UNIVERSIDAD CARLOS III DE MADRID

DEPARTAMENTO DE INGENIERÍA DE SISTEMAS Y AUTOMÁTICA

TRABAJO DE FIN DE GRADO

SEGUIMIENTO DE LOS DEDOS DE UNA MANO MEDIANTE TECNOLOGÍA RGBD

AUTOR: Álvaro González García

TUTOR: David Álvarez Sánchez

GRADO EN INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA

Leganés, MADRID.

Febrero de 2014

Quería agradecer a todas las personas que me han ayudado día a día a la realización de este proyecto. Han sido muchos meses, y me habéis servido de gran ayuda:

Gracias en especial a mi familia: mamá, papá y Alex;

a mi tutor, David, que siempre ha estado disponible para ayudarme;

y por supuesto a mis amigos, que han estado apoyando y pendientes de mí en todo momento, en especial a Rodrigo.

ÍNDICE

ÍNDICE DE FIGURAS

I. RESUMEN

Uno de los aspectos en los que se está centrando el mundo de la robótica es el campo del tratamiento de imágenes y de la visión artificial.

La Visión Artificial es una disciplina que pretende dotar a los ordenadores de la capacidad de poder simular la visión humana. Algo que para nuestro cerebro es trivial como distinguir un objeto del fondo, no lo es para un ordenador. En esto consiste la Visión Artificial, en extraer información del mundo físico a través del procesamiento de una o varias imágenes.

Este proyecto se centra en uno de los campos de la visión artificial como es el seguimiento de objetos, y para ello se han utilizado dos filtros diferentes, el Filtro de Kalman, y el Filtro de Partículas.

El objetivo principal del proyecto es la utilización de estos filtros en una secuencia de imágenes tridimensionales, obtenidas en tiempo real mediante el sensor Kinect, para seguir una mano (incluyendo estados de oclusión) con el fin de obtener conclusiones sobre la eficacia y diferencias entre ambos filtros.

II. ABSTRACT

One of the aspects in which is focusing the robotic world is the field of Image Processing and Computer Vision.

Machine Vision is a discipline that aims to give computers the ability to simulate human vision. Something that for our brain is trivial like distinguish between an object and the background, it is not for a computer. This is the aim for Artificial Vision, extracting information from the physical world through processing one or more images.

This project focuses on one of the fields of Computer Vision such as object tracking, and for this I have used two different filters, the Kalman Filter and the Particle Filter.

The main objective of the project is the implementation of these filters in a sequence of three-dimensional images obtained in real time using the Kinect sensor to track a hand (including statements of occlusion) in order to draw conclusions about the effectiveness and differences between these filters .

III. INTRODUCCION

A. VISION ARTIFICIAL

Aristóteles recoge en su obra que “visión es saber qué hay, y dónde, mediante la vista”. Muchos años después, J. J. Gibson definió visión como “recuperar de la información de los sentidos (vista) propiedades validas del mundo exterior” [2]. El neurocientífico David Marr, a su vez, dijo que “visión es un proceso que produce, a partir de las imágenes del mundo exterior, una descripción que es útil para el observador y que no tiene información irrelevante” [4].

En el hombre, la vista es el sentido más importante y a su vez el más complejo, consistiendo en el 75% de la información procesada por el cerebro. La importancia de este sentido para el ser humano es la razón por la que se ha querido transformar y transmitir toda esta información a aplicaciones robóticas. Esta transformación de una imagen del mundo real a un formato digital inteligible por un ordenador y el procesamiento por parte de éste, es lo que se conoce como visión artificial o visión computacional.

1. EVOLUCIÓN HISTÓRICA DE LA VISIÓN ARTIFICIAL

En la década de los 50 había mucha confianza en el poder de los ordenadores. Como la acción de ver era fácil para nosotros, se creía que sería fácil transmitirlo a los ordenadores.

En los 60, viendo que los avances obtenidos eran muy escasos, llegó una época de frustración. Se dieron cuenta que se sabía muy poco del proceso de visión en el hombre. Y se buscaron diversas soluciones para solventar el problema de que la información de la cámara es una proyección bidimensional de objetos tridimensionales.

A partir de 1980 fue cuando esta materia se empezó a llamar visión computacional. Aumentaron notablemente las vías de investigación y el espectacular avance tecnológico permitió el desarrollo de cámaras y ordenadores mucho más potentes y a unos precios muy asequibles.

Con el avance sufrido en este campo, se empezó a usar la Visión Artificial en numerosas aplicaciones en el mundo de la robótica, control de calidad, aplicaciones militares, seguridad, y en muchos más campos.

2. FASES DEL PROCESAMIENTO DE IMÁGENES

Podemos dividir la visión artificial en dos niveles: nivel bajo y nivel alto [1].

En la visión de bajo nivel incluimos aquellas fases del procesamiento de imágenes en las que se trabaja para extraer propiedades de los puntos o píxeles que componen la imagen. En este nivel se realizan, en este orden, las siguientes fases:

-La primera fase, que es puramente sensorial, es la adquisición de la imagen a través de una cámara.

-La segunda etapa es el preprocesamiento, que consiste en el tratamiento digital de las imágenes capturadas para facilitar las fases posteriores. Para ello se aplican a las imágenes ciertos tratamientos como pueden ser filtros, transformaciones geométricas, etc. Forman parte de esta etapa de preprocesamiento acciones como eliminación de ruido, realce del contorno, mejora del contraste, etc.

El nivel alto se encarga de agrupar los elementos obtenidos en el nivel bajo y tratarlos con el fin de que el ordenador sea capaz de interpretar los resultados obtenidos. En este nivel se realizan las siguientes fases o etapas:

-Fase de segmentación, que consiste en aislar los elementos que interesan de una escena, para comprenderla.

-A veces se le aplican a las imágenes transformaciones morfológicas para orientarla a la posterior clasificación.

- Una fase útil para determinadas aplicaciones es la descripción de objetos, que consiste en extraer información como área, perímetro del contorno, número de huecos,...

-En la etapa de reconocimiento o clasificación se pretende distinguir los objetos segmentados mediante el análisis de los resultados obtenidos en las fases anteriores, comparándolos con algunas características que se establecen previamente para diferenciarlos.

Estas fases no siempre se llevan a cabo todas ellas, o no siempre en orden secuencial, sino que en ocasiones es conveniente o necesaria una realimentación hacia atrás cuando falla alguna de las fases.

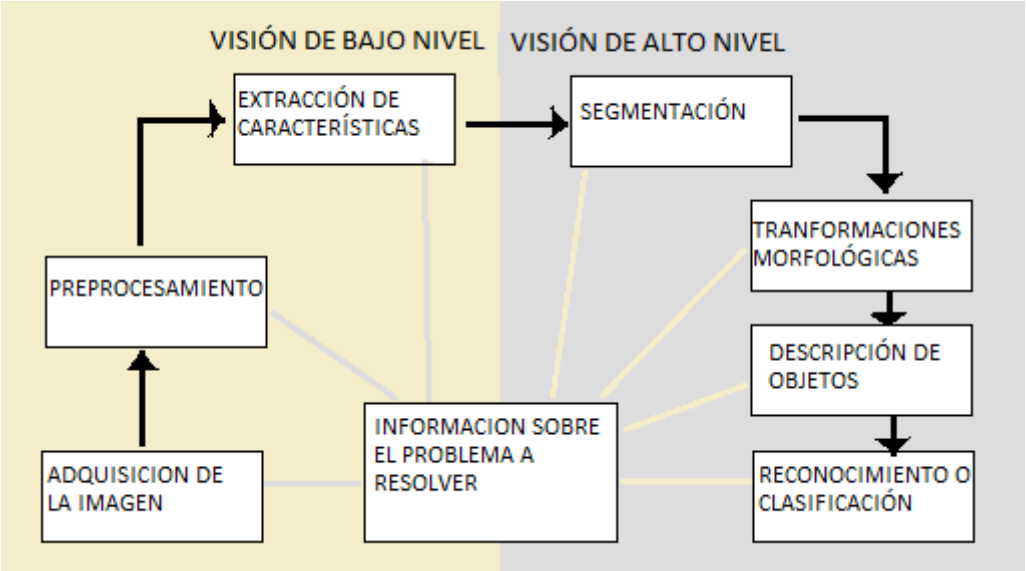


Figura 1: Etapas de una aplicación de visión artificial

B. MOTIVACIONES

En la actualidad existe una gran expansión de la visión por computador y sus aplicaciones tanto en la vida cotidiana como en el ámbito de la investigación.

El uso del mando a distancia comienza a desaparecer debido a la incorporación de cámaras capaces de detectar el movimiento del usuario para su manejo.

Pero existen más aplicaciones en las que se pueden utilizar el cuerpo como mecanismo de control. Esta tecnología también se encuentra aplicada en los videojuegos como es el caso de la cámara Kinect de Microsoft o su antecesora Eye Toy de Sony.

Pero la detección del movimiento del usuario no solo tiene aplicaciones interactivas sino que actualmente también se utiliza para el manejo de robots mediante gestos, aplicaciones en biomedicina para analizar el movimiento de las personas, etc.

El auge del uso del cuerpo humano para el manejo de sistemas crea la necesidad de la investigación de su funcionamiento, centrándose en la cámara Kinect y sus librerías para ordenador.

C. OBJETIVOS

Este proyecto se centra en uno de los problemas clásicos de la visión artificial: el seguimiento de un objeto en una secuencia de imágenes. Para ello, se emplean métodos probabilísticos como el Filtro de Kalman y el Filtro de Partículas.

El objetivo principal de este proyecto consiste en utilizar dichos filtros para el seguimiento de una mano en secuencias de imágenes 3D en tiempo real. Dicho seguimiento se realizará tanto cuando el objeto a seguir (la mano) está visible, como cuando está en estado de oclusión.

Para alcanzar este objetivo final, es necesario alcanzar una serie de objetivos secundarios:

1. Manejar la librería PCL para el tratamiento de nubes de puntos.
2. Adquirir nubes de puntos tridimensionales en tiempo real a través del sensor Kinect de Microsoft.
3. Aprender técnicas de tratamiento de imágenes en 3D.
4. Localizar los puntos característicos de la mano.
5. Comprensión del funcionamiento del Filtro de Kalman y el Filtro de Partículas.
6. Aplicar ambos filtros al seguimiento de la mano en tiempo real.
7. Adaptar ambos filtros al modo de oclusión.
8. Señalar el punto que predicen estos filtros con cuadrados de colores para que el funcionamiento de los filtros sea visible para el usuario.
9. Comparar el Filtro de Kalman y el Filtro de Partículas.

IV. ESTADO DEL ARTE

Para realizar el seguimiento de un objeto, lo primero que hay que hacer es localizar el objeto que vamos a seguir, y a continuación, aplicar a dicho objeto el algoritmo de seguimiento correspondiente.

A. LOCALIZACIÓN DEL OBJETO

El proceso de localizar el objeto a seguir, se puede dividir en cuatro fases [8]: adquisición de datos, filtrado, segmentación y reconocimiento del objeto.

1. Adquisición de datos:

Los seres humanos percibimos las imágenes a través de los ojos. Al tener dos ojos, el cerebro recibe dos imágenes diferentes, una procedente de cada ojo. La capacidad que tiene un ser vivo de integrar estas dos imágenes en una sola por medio del cerebro es lo que se conoce como visión estereoscópica. Gracias a la combinación de las dos imágenes percibimos la realidad en tres dimensiones.

Las cámaras estereoscópicas tratan de imitar el comportamiento de los ojos, consistiendo no en una sola cámara, sino dos, separadas por una distancia conocida, y es el ordenador el que se encarga de integrar las dos imágenes para formar una sola imagen tridimensional. Además, es necesario conocer la distancia a cada punto de la imagen. Los datos de profundidad de cada punto de la imagen pueden ser obtenidos habiendo contacto con dichos puntos, o sin él:

- Las tecnologías que precisan de contacto son más lentas, pero más precisas.

- Las tecnologías que no precisan contacto pueden dividirse en activas, si emiten algún tipo de radiación para obtener la forma del objeto; o pasivas, si no emiten sino que detectan la radiación ambiental.

En este proyecto los datos los vamos a adquirir con tecnología que no precisa contacto, por lo que vamos a centrar el estudio en este tipo de tecnologías para obtener la información. Algunos métodos son [3]:

a) Triangulación estereoscópica

Para determinar la profundidad es necesario encontrar la correspondencia entre los puntos de ambas imágenes. Este problema es una de las limitaciones de este método, ya que es difícil establecer una correspondencia entre puntos que están en una región de intensidad o color homogéneos.

Las imágenes basadas en triangulación estereoscópica producen generalmente estimaciones de profundidad fiables para el conjunto de puntos visibles por ambas cámaras.
La medición con esta técnica es pasiva, y no requiere condiciones especiales de iluminación de la escena.

En 1991, el científico Huang [9] utilizó este método para determinar la forma de un objeto a partir de su sombra (shape-from-shading) [11]. Al igual que Zheng Zhenrong [10], para captar el objeto visto desde todas las perspectivas, usó una mesa rotatoria. Su estudio se basó en detectar los puntos angulosos de las figuras, como las esquinas; pero esto no es eficaz para objetos esféricos.

Jiajun Zhu usó ocho cámaras (4 pares de cámaras). Las colocó de tal forma que cubrieran los 360 grados del objeto [13]. Jiajun Zhu hizo que cada cámara capturara dieciocho imágenes de alta resolución en color. Cada una de las imágenes que capturaba con una cantidad de luz diferente, y con esto conseguía capturar la posición del objeto. El objetivo de esta disposición es la obtención de mapa de profundidad de mucha calidad. Este método tenía el inconveniente es que el tiempo de procesado de imagen era muy alto.

Moro propuso otra forma para determinar la profundidad de una imagen utilizando cámaras estereográficas. Extraía ciertos modelos de los objetos de la imagen previamente enseñados [14]. Esa necesidad de tener modelos entrenados era la mayor limitación del método propuesto por Moro.

b) Método de luz estructurada

Se diseña un patrón de luz con el que iluminar la escena. La profundidad se determina mediante la imagen de la luz reflejada. La luz estructurada puede ser en forma de líneas horizontales y verticales, puntos o patrones como los cuadrados del tablero de ajedrez. Siendo la más común la luz estructurada en forma de rayas.

La precisión de este método depende de la forma de la estructura de la luz, de la calidad óptica de la cámara, y de la longitud de onda de la luz. Este método sin contacto activo requiere de una calibración inicial.

La cámara Kinect, que es la que usaremos en este proyecto, utiliza este método para calcular la profundidad. Cruz publicó un artículo [19] donde se exponen los últimos avances y la gran variedad de aplicaciones en las que puede ser integrada esta cámara.

c) Time-of-flight

Otra técnica para medir la profundidad es la conocida como time-of-flight (tiempo de vuelo). Las cámaras que utilizan la técnica de tiempo de vuelo son dispositivos relativamente nuevos, que capturan escenas en tres dimensiones.

Su base científica es parecida a la que utiliza el radar o el lidar, la diferencia con estos es que el radar y el lidar utilizan ondas de radiofrecuencia y no ondas de luz. Consiste en la emisión de ondas de luz infrarroja, la distancia se calcula midiendo el tiempo que tarda esa onda en ir y volver. Por tanto incluimos esta técnica en el grupo de sin contacto activo.
Los resultados que obtenemos con cámaras de este tipo tienen mucha precisión, si bien es una tecnología relativamente cara.

Charles Pheatt propuso un método utilizando una cámara digital y dos diodos laser, y que necesitaba en torno a 300 escaneos para reconstruir un objeto no muy complejo, utilizando una mesa rotatoria para capturar el objeto desde todas las perspectivas [20].
Un paso más dio Hagebeuker [21] que introdujo un nuevo tipo de cámaras, las Photonic Mixer Device, PMD, que utilizando la técnica de time-of flight consigue mejorar los resultados que se obtenían hasta entonces.

d) Interferometría

La interferometría es una técnica que consiste en combinar la luz proveniente de diferentes receptores para obtener una imagen de mayor resolución. Esta técnica es difícil de implementar para longitudes de onda más corta.

El principio físico utilizado es que dos ondas de luz que coinciden en fase se amplifican mientras que dos ondas en oposición de fase se cancelan, existiendo también cualquier combinación intermedia. Esto permite, mediante medición del grado de cancelación o amplificación de dos haces láser, realizar mediciones de superficies menores a la longitud de onda.

Para estas aplicaciones se pueden utilizar diferentes tipos de interferómetros, como el interferómetro Twyman-Green y el interferómetro de Fizeau.

Uno de los primeros usos de la interferometría fue en el famoso experimento de Michelson y Morley (1887)[15], que demostró la inexistencia del éter, proporcionando las primeras evidencias experimentales en las que más tarde se asentaría la relatividad especial. El interferómetro de Michelson permite medir distancias con una precisión muy alta. Su funcionamiento se basa en la división de un haz coherente de luz en dos haces para que recorran caminos diferentes y luego converjan nuevamente en un punto. De esta forma se obtiene lo que se denomina la figura de interferencia, que permitirá medir pequeñas variaciones en cada uno de los caminos seguidos por los haces.

2. Segmentación

La segmentación es el proceso que divide una imagen en regiones u objetos cuyos píxeles poseen atributos similares, y constituye uno de los procesos más importantes de un sistema automatizado de visión, ya que permite extraer los objetos de la imagen para su posterior descripción y reconocimiento [16].

Las distintas técnicas de segmentación pueden encuadrarse en tres grupos fundamentales: técnicas de umbralización, técnicas basadas en regiones y técnicas basadas en la detección de bordes.

- El método de *segmentación por umbralización* tiene en cuenta el valor de intensidad de los píxeles, para decidir si estos forman parte de un objeto de interés.

  El objetivo de este método, es encontrar de una manera óptima los valores característicos de la imagen que establecen la separación del objeto de interés, con

respecto a las regiones que no pertenecen al mismo. Si los valores de intensidad del objeto y del resto de la imagen difieren claramente, entonces el histograma mostrará una distribución bimodal, con dos máximos distintos, separados por una región vacía, con lo cual se logrará una separación perfecta entre el objeto y el fondo, al establecer un valor umbral ubicado en esta región del histograma. Por lo tanto cada píxel de la imagen, es asignado a una de dos categorías, dependiendo si el valor umbral es excedido o no.

El problema muchas veces está en encontrar el valor a tomar como umbral entre objetos. Esto se debe a la presencia de ruido o a que el objeto y el fondo no tienen un único color sino un intervalo de ellos que se solapan en algunos valores. Cuando surge este problema, hay varios métodos para solucionarlo:

Posiblemente el método más intuitivo y fácil sea el mínimo entre máximos, que sirve para una umbralización en la que los dos objetos a separar tienen colores diferentes, y observando el histograma se observan dos picos. Se establece el umbral en el punto entre los dos picos donde menor sea la frecuencia de color.

Si se tiene un conocimiento a priori de la imagen, sabiendo el porcentaje de pixeles que componen un objeto, utilizamos el método P-Tile para establecer el umbral, observando el histograma, se toma el valor que delimite ese tanto por ciento. También se puede saber a priori el número de objetos, el tamaño, etc.

Otra forma es mediante medidas iterativas. Se considera el histograma como dos gaussianas con igual desviación específica, y se toma como umbral el valor medio de la imagen. De esta forma se divide el histograma en dos partes de las que se obtienen sus medias. Y se elige el nuevo valor según:

$$T = \frac{\mu_1 + \mu_2}{2}$$

Se vuelven a calcular las dos medias con este nuevo umbral, si son distintas, se repite el proceso tantas veces como sea necesario.

Nobuyuki Otsu diseñó otro método que adoptó su nombre. Él supuso que el histograma es la suma de dos gaussianas. Y su método establece que el umbral debe minimizar la suma ponderada de cada una de las varianzas de los objetos presentes.

$$q_1(t) = \sum_{i=1}^{t} P(i) \qquad q_2(t) = \sum_{i=t+1}^{I} P(i)$$

$$\sigma_w{}^2 = q_1(t)\sigma_1{}^2 + q_2(t)\sigma_2{}^2$$

El método de Kittle-Illingoworth toma dos probabilidades de que aparezca un nivel de gris en la imagen: la probabilidad del histograma de la imagen real, P(i); y la probabilidad del histograma si fuera la suma de dos gaussianas ,f(i); y establece un parámetro J que representa el coste de equivocarse al tomar f(i) en lugar de P(i). Hay que encontrar el valor que minimiza J.

$$J = \sum_{i=1}^{I} P(i) log \frac{P(i)}{f(i)} = \sum_{i=1}^{I} P(i) log P(i) - \sum_{i=1}^{I} P(i) log f(i)$$

Otra forma es el Método de Partición de la Unidad, PUM, que usa la definición de entropía:

$$H = -\sum p_i \log(p_i)$$

En donde hay que maximizar f(t):

$$f(t) = \frac{H_t}{H_T} \frac{log P_t}{\log(\max(p_0, p_1, \dots, p_t)} + \left[1 - \frac{H_t}{H_T}\right] \frac{\log(1 - P_t)}{\log(\max(p_{t+1}, p_{t+2}, \dots, p_N)}$$

$$H_T = -\sum_{i=0}^{i=N} p_i \log(p_t)$$

Y por último, el método Kapur, que también usa la entropía, pero corrige algunos términos. Busca maximizar H_t y H_T.

A continuación se muestra una tabla comparativa de todos los métodos mencionados, donde se aprecia para cuatro imágenes diferentes, el resultado que se obtendría al aplicar un método de segmentación u otro:

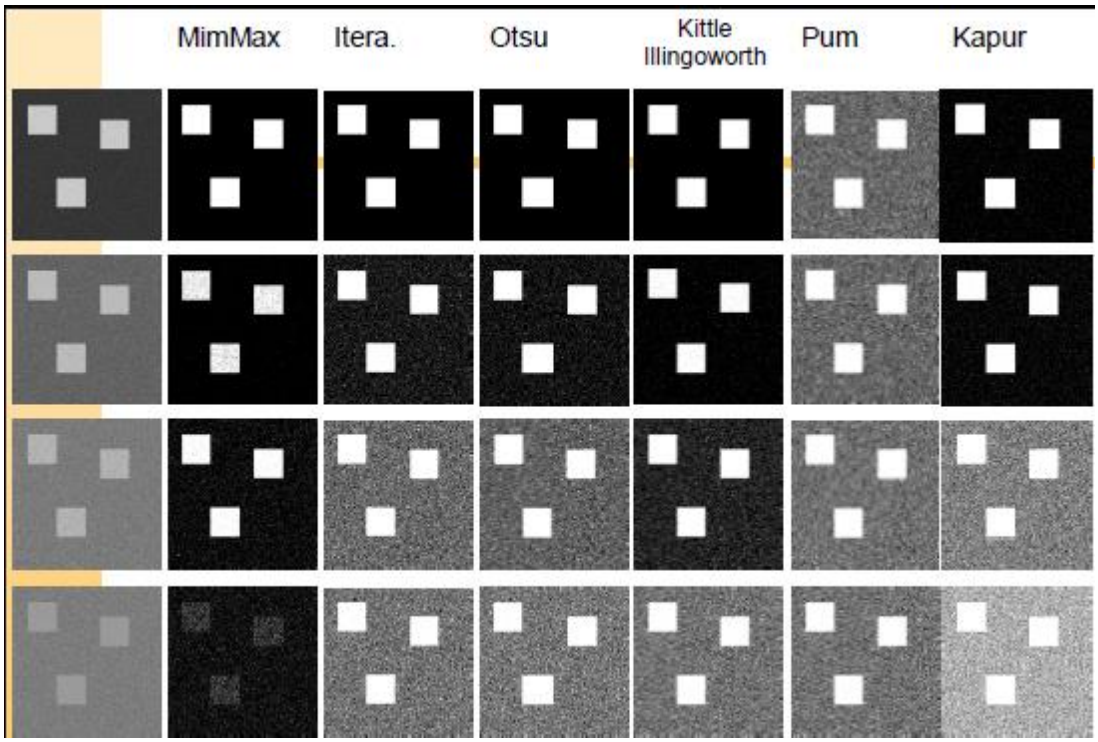


Figura 2: Tabla comparativa de los métodos de umbralización

- Los *métodos de segmentación basados en regiones* tienen en cuenta un conjunto de píxeles de la imagen, en los que se analizan características como la posición en el espacio de intensidades, relaciones topológicas (conectividad) y características de las fronteras entre dos conjuntos. Dependiendo de cómo sea analizada la posición en el espacio y las relaciones espaciales existentes entre los píxeles, se pueden encontrar *métodos de Clasificación* y métodos por *Crecimiento de Regiones*.

  Los *métodos de Clasificación* determinan primero una partición del espacio de intensidades y utilizan luego las relaciones de conectividad para determinar una región.

  La *segmentación basada en conectividad de regiones* considera que una región en una imagen está constituida por un conjunto de píxeles contiguos y conectados. La conectividad de un píxel con otro puede ser de 4-conectividad si existe relación entre dos píxeles colindantes horizontalmente o verticalmente, o bien 8-conectividad si los píxeles se tocan en forma diagonal también. Dos píxeles que no son colindantes también pueden tener 4 u 8- conectividad si existe un camino de uno al otro a través de píxeles conectados.

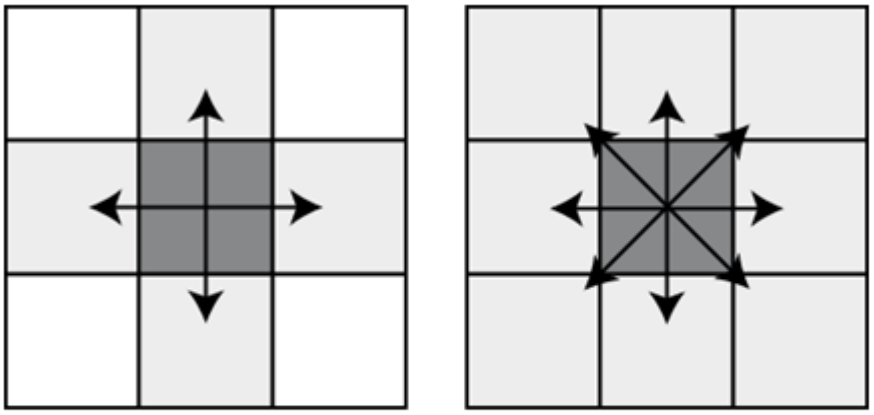


Figura 3: Conectividad 4 frente a conectividad 8

En el procesamiento de la imagen se determina si cada píxel es miembro o no de un objeto mediante lo que se conoce como proceso de etiquetación: un píxel a considerar puede pertenecer a un objeto ya identificado (si tiene conectividad con pixeles del objeto), o puede ser el primer píxel de un objeto nuevo. Cuando ya están identificados todos los píxeles de cada objeto, se determinará la situación de estos objetos en la imagen.

Los *métodos de Crecimiento de Regiones* utilizan de manera simultánea tanto la partición del espacio de intensidades como el estudio de la relación de conectividad, para determinar una región. Estos métodos, consideran a cada píxel como un nodo de un grafo, luego analizando una vecindad de un píxel cualquiera, se unen por medio de un arco todos aquellos píxeles con características similares. La intensidad de los niveles de gris es la característica de similitud buscada, de forma que dos píxeles estarán conectados si estos difieren en menos de un umbral preestablecido, y si se encuentran en una determinada vecindad.

- La segmentación de una imagen puede también llevarse a cabo mediante la *detección de los límites de cada región (detección de bordes)*, es decir, localizando los lugares donde se produce un cambio significativo de los niveles de intensidad de los píxeles (detección de bordes). Esta operación puede efectuarse utilizando operadores de detección de bordes basados en la primera y/o segunda derivada. En el primer caso, como consecuencia del ruido, iluminación no uniforme y otros, los operadores gradiente rara vez definen por completo la frontera de los objetos, necesitándose, en

tal caso, algoritmos que se encarguen de realizar la unión de los píxeles detectados mediante el seguimiento del contorno.

Estos métodos basados en la detección de bordes, tienen algunos problemas como los objetos que están unidos, ocultamientos, o discontinuidades producidas por el ruido en la imagen. Si tratas de eliminar este ruido, puedes eliminar también los bordes de objetos pequeños.

Para realizar este tipo de segmentación, hay una serie de métodos locales, como los basados en el gradiente con los operadores de Roberts, Sobel, Prewitt, o el operador isotrópico, pero todos estos métodos son muy sensibles al ruido y a cortes en el borde.

También hay métodos globales, como la transformada de Hough, que busca formas geométricas en toda la imagen, encontrando los parámetros de la forma (recta, circunferencia, elipse,…) que contenga más puntos en la imagen.

3. Reconocimiento

El reconocimiento de los objetos que captamos es una de las principales complicaciones de la visión artificial. Cuando este problema esté completamente resuelto y un robot sea capaz de interpretar el entorno y reconocer todos los objetos que le rodean, los robots serán capaces de tomar decisiones por sí mismos.

Esa es la razón por la que hay multitud de estudios centrados en esta área. Una de las líneas de trabajo que más se está investigando hoy en día está relacionada con las bases de datos, como por ejemplo la base de datos publicada en Lai [5], que contiene 300 objetos organizados en 51 categorías diferentes.

El investigador Tang crea una amplia base de datos con numerosos datos y posiciones de diferentes objetos cotidianos [6]. La base de datos de Tang es muy eficaz porque cada objeto tiene una gran cantidad de datos. Hasta el punto de que el ordenador es capaz de solucionar obstrucciones, cambios de iluminación, etc.

Otro método para el reconocimiento de objetos cotidianos en una escena con un enfoque diferente es el que utiliza Ahn [18], que utiliza un sistema de visión para robots conocido como SLAM (Simultaneous localization and mapping). Pero lo usa para el reconocimiento de objetos. Para ello primero extrae las características invariantes, luego hace un agrupamiento RANSAC (RANdom SAmple Consensus) y por último calcula información métrica precisa para actualizar el SLAM.

B. SEGUIMIENTO DEL OBJETO

El seguimiento de objetos es el proceso de estimar, a lo largo del tiempo, la ubicación de uno o más objetos móviles mediante el uso de una cámara. La rápida mejora en cuanto a calidad y resolución de los sensores de imagen, junto con el incremento de la potencia de cálculo en la última década, ha favorecido la creación de nuevos algoritmos y aplicaciones mediante el seguimiento de objetos.

Las cámaras de video capturan información sobre los objetos de interés en forma de conjunto de píxeles o puntos. Al modelar la relación entre el aspecto del objeto de interés y el valor de los píxeles correspondientes, un seguidor de objetos valora la ubicación de este objeto en el tiempo. La relación entre el objeto y la proyección de su imagen es muy compleja y puede depender de más factores que no sean solamente la posición del objeto, lo que implica que el seguimiento de objetos sea una tarea difícil. Por ejemplo, los cambios de aspecto del objeto en el plano de la imagen dificultan el seguimiento, causado por uno o más de los siguientes factores [17]:

- Cambios de posición. El objeto de interés varia su aspecto cuando se proyecta sobre el plano de la imagen, por ejemplo, al girar.
- Iluminación ambiente. La dirección, la intensidad y el color de la luz de ambiente influyen en el aspecto del objeto de interés. De la misma manera, los cambios en la iluminación global son con frecuencia un reto en las escenas al aire libre.
- Ruido. El proceso de adquisición de imágenes introduce en la señal de la imagen un cierto grado de ruido que depende de la calidad del sensor. Las observaciones del objeto de interés pueden dañarse y por tanto afectar al rendimiento del seguidor.
- Oclusiones. Puede ser que un objeto de interés no se observe bien cuando sea parcial o totalmente tapado por otros objetos en la escena. Las oclusiones son generalmente debidas a:
    - Un objeto de interés que se mueve detrás de un objeto estático, como por ejemplo una columna.
    - Otros objetos que se mueven en la escena de manera que entorpecen la visión de un objeto de interés.

El seguimiento de objetos es una tarea muy importante dentro del campo del procesado de video. Podemos hacer una clasificación de técnicas según tres grandes grupos: seguimiento de

puntos, seguimiento de núcleo (kernel) y seguimiento de siluetas. Como se muestra en el siguiente esquema:

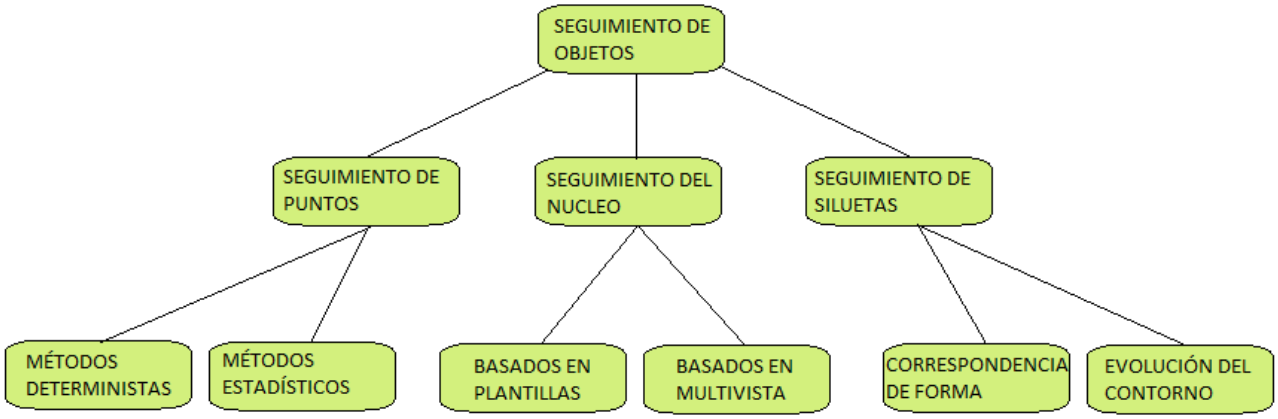


Figura 4: Esquema de seguimiento de objetos

1. Técnicas de seguimiento de puntos

Los objetos detectados en secuencias de imágenes consecutivas quedan representados cada uno por uno o varios puntos, y la asociación de éstos está basada en el estado del objeto en la imagen anterior, que puede incluir posición y movimiento.

Esta técnica puede presentar problemas en escenarios donde el objeto presenta oclusiones. Las técnicas de seguimiento de puntos se pueden clasificar a su vez en dos grandes categorías:

a) Métodos estadísticos.

Solucionan los problemas de seguimiento teniendo en cuenta las observaciones y las incertidumbres del modelo para la valoración del estado del objeto que se está siguiendo. Modelan propiedades del objeto tales como la posición, la velocidad y la aceleración. Las observaciones consisten normalmente en la posición del objeto dentro de la imagen, que se obtiene mediante algoritmos de detección. Algunos de los métodos más utilizados son los que desarrollaremos en este proyecto como el Filtro de Kalman y el Filtro de Partículas.

b) Métodos deterministas.

Determinan el coste de correspondencia a través de una predicción futura del comportamiento del objeto a partir del anterior. Este coste se define normalmente como una combinación de las siguientes restricciones:

- Proximidad.
- Velocidad máxima.
- Cambios de velocidad pequeños.
- Movimiento común.
- Rigidez.
- Uniformidad por proximidad.

2. Técnicas de seguimiento del núcleo

Las técnicas de seguimiento del núcleo realizan un cálculo del movimiento del objeto, el cual está representado por una región inicial, de una imagen a la siguiente. El movimiento del objeto se expresa en general en forma de movimiento paramétrico o mediante el campo de flujo calculado en los siguientes fotogramas. Podemos distinguir dos categorías:

- Seguimiento utilizando plantillas y modelos de apariencia basados en densidad de probabilidad. El método que más se utiliza en esta categoría es el *llamado template matching*.
- Seguimiento basado en modelos multivista. Se utiliza cuando el aspecto del objeto cambia drásticamente y como consecuencia se pierde el seguimiento de este objeto.

3. Técnicas de seguimiento de siluetas

Estas técnicas se realizan mediante la valoración de la región del objeto en cada imagen utilizando la información que contiene. Esta información puede ser en forma de densidad de aspecto o de modelos de forma que son generalmente presentados con mapas de márgenes. Dispone de dos métodos:

- Correspondencia de forma.
- Seguimiento del contorno.

V. CONCEPTOS TEÓRICOS

A. ESPACIO DE COLOR

El color es una sensación que producen los rayos luminosos en los órganos visuales y que es interpretada en el cerebro. Se trata de un fenómeno físico-químico donde cada color depende de la longitud de onda.

El espectro electromagnético es la distribución energética del conjunto de las ondas electromagnéticas. El espectro electromagnético se extiende desde la radiación de menor longitud de onda, como los rayos gamma y los rayos X, pasando por la luz ultravioleta, la luz visible y los rayos infrarrojos, hasta las ondas electromagnéticas de mayor longitud de onda, como son las ondas de radio [22].

La luz es una radiación electromagnética en el rango de frecuencias de los 405 a los 790 THz (740 a 380 nm). Este rango de frecuencias es lo que se conoce como la radiación o espectro visible. [25]

En los objetos, la sensación de color se produce por la absorción de frecuencias incidentes. Los cuerpos iluminados absorben parte de las ondas electromagnéticas y reflejan las restantes. Dichas ondas reflejadas son captadas por el ojo y, de acuerdo a la longitud de onda, son interpretadas por el cerebro. En condiciones de poca luz, el ser humano sólo puede ver en blanco y negro [23].

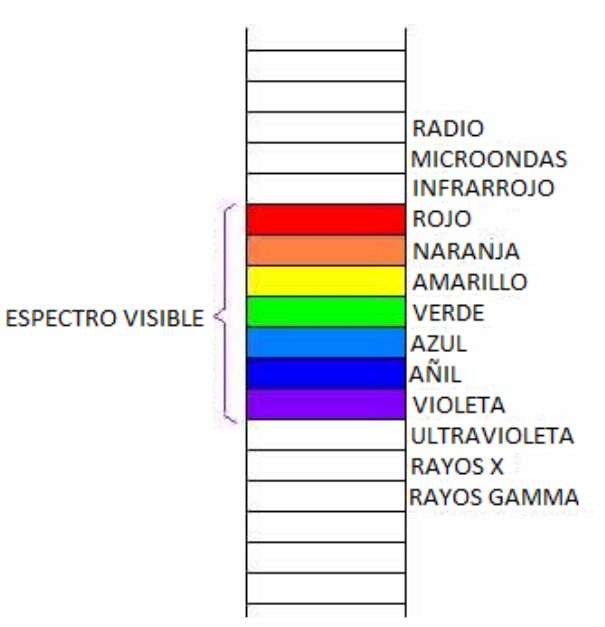


Figura 5: Espectro de la radiación

Para el estudio y representación de los colores tal y como los percibimos los humanos, se han creado varios modelos o espacios de color. Son modelos matemáticos abstractos, que describen la forma en que se representan los colores mediante tuplas de números. El estudio de los modelos de color es importante, porque el modelo condiciona cómo se captura, almacena, procesa, transmite y genera el color.

No todos los modelos de color son completos y algunos son mejores que otros dependiendo de la aplicación. Así, el modelo de color RGB es el que más se ajusta al modo de captura y generación en imagen digital. El espacio de color HSV separa crominancia (color) y luminancia (brillo). Estos dos modelos nombrados son los que vamos a utilizar en nuestro proyecto, si bien hay más modelos como el Color LAB, los modelos YIQ y YUV, CMYK, etc.

1. RGB

EL espacio de color RGB está basado en un modelo aditivo de mezcla, con 3 colores primarios: R-red (rojo), G-green (verde), B-blue(azul). La combinación aditiva de estos colores primarios produce todo el rango de colores representables en RGB.

El espacio RGB tiene forma de cubo de lado 1. El punto (R=0, G=0 , B=0) es el negro, y el (R=1,G=1,B=1) es el blanco. Surgen tres colores secundarios: cian, magenta y amarillo.

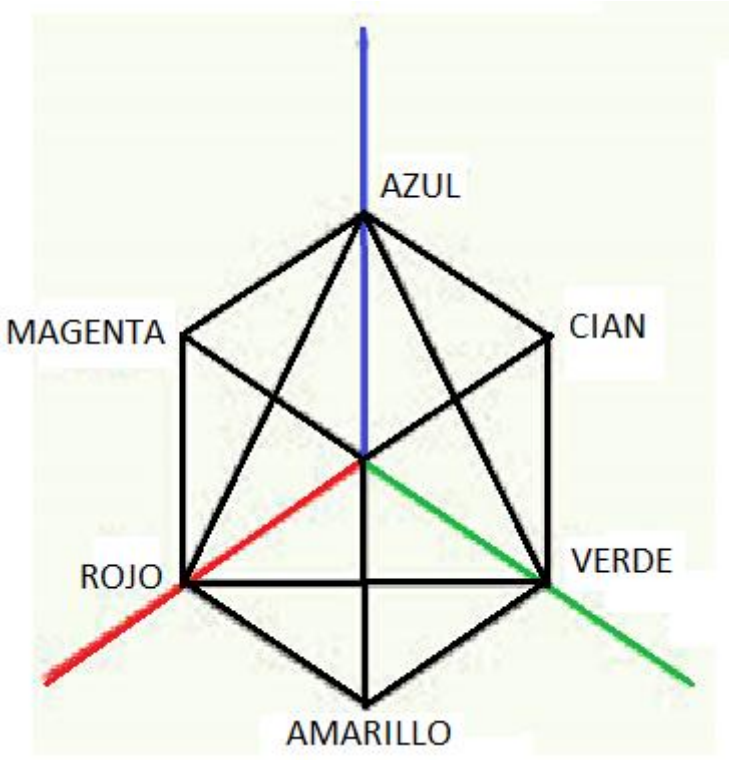


Figura 6: Espacio de color RGB

El espacio RGB es el más utilizado en la práctica, pero no es completo, ya que existen colores que no se pueden obtener con la combinación de R, G y B.

En el proyecto, la adquisición de imágenes se realiza con una cámara Kinect, que adquiere los datos de color en formato RGB.

2. HSV

El espacio de color HSV, o HSI está pensado para ser fácilmente interpretable y legible por un humano, usa términos más familiares cuando hablamos de color.

El espacio HSV se suele representar como un cono o como un cilindro, y consta de los componentes: H-matiz (hue), S-saturación, V-valor de intensidad.

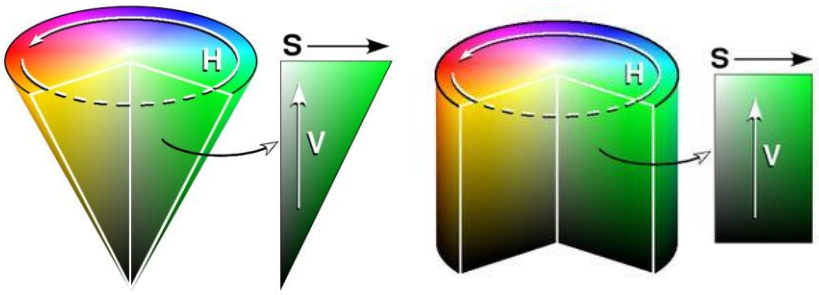


Figura 7: Espacio de color HSV

El valor de intensidad, o luminosidad (V) es la cualidad de ser más claro o más oscuro. Su valor se suele dar o en tanto por ciento o en tanto por uno, siendo lo correspondiente al rango de blanco a negro.

La saturación (S) diferencia del color respecto a un gris con la misma intensidad. Cuanto más diferente, más saturado. También se suele expresar en tanto por ciento o en tanto por uno, e indica el radio de la rueda cromática.

El matiz de un color (H) es su ángulo dentro de la rueda cromática. Su valor está comprendido entre 0º y 359º.

Este espacio de color es poco sensible a cambios de iluminación, ya que un cambio de ésta, solo alteraría de forma significativa el valor de V, dejando H y S prácticamente iguales.

B. MEDIA ARITMÉTICA RECORTADA

En matemáticas y estadística ,la media aritmética (también llamada promedio o simplemente media) de un conjunto finito de números es el valor característico de una serie de datos cuantitativos objeto de estudio que parte del principio de la esperanza matemática o valor esperado, se obtiene a partir de la suma de todos sus valores dividida entre el número de sumandos. Expresada de forma más intuitiva, podemos decir que la media es la cantidad total de la variable distribuida a partes iguales entre cada observación. Una de las limitaciones de la media aritmética es que se trata de una medida muy sensible a los valores extremos; valores muy grandes tienden a aumentarla mientras que valores muy pequeños tienden a reducirla. Esa es la razón por la que existe la media recortada, o también llamada media truncada.

La media recortada tiene por finalidad evitar la distorsión que los valores extremos no compensados causan a la media aritmética. Para obtener la media recortada se excluyen un porcentaje de valores extremos superiores e inferiores de la distribución [24]. La media recortada también se hace quitando valores sólo por el lado inferior o sólo por el lado superior. La aplicación de estos casos es para cuando sólo se concibe la aparición de valores anormales bien por encima o por debajo de los valores normales.

C. CENTRO DE GRAVEDAD

El centro de gravedad es el punto de aplicación de la resultante de todas las fuerzas de gravedad que actúan sobre las distintas porciones materiales de un cuerpo, de tal forma que el momento respecto a cualquier punto de esta resultante aplicada en el centro de gravedad es el mismo que el producido por los pesos de todas las masas materiales que constituyen dicho cuerpo. El centro de gravedad de un cuerpo no corresponde necesariamente a un punto material del cuerpo. Así, el centro de gravedad de una esfera hueca está situado en el centro de la esfera que, obviamente, no pertenece al cuerpo [26].

D. FILTRADO

Los ficheros generados en la adquisición de datos en 3D son muy densos, con gran cantidad de puntos, la mayoría de ellos innecesarios para nuestro objetivo. Para reducir notablemente carga computacional al tratamiento de la información realizamos un filtrado, que lo que hace es eliminar muchos de los puntos que no nos interesan.

En nuestro proyecto utilizamos un filtro de profundidad, el cual elimina todos los puntos que no estén en una franja de valores definida por nosotros. Este filtro lo proporciona la librería de filtrado de PCL, y se puede aplicar en cualquiera de los tres ejes.

Para demostrar de forma gráfica el resultado de aplicar un filtro de profundidad, se exponen las dos siguientes imágenes. La primera es una nube de puntos a la que no se le ha aplicado el filtro de profundidad; y la segunda, es una imagen con un filtro de profundidad, que elimina todos los puntos que estén a más de 1.2m del origen en el eje Z:

Figura 8: Imagen de la izquierda, sin filtrar. Imagen de la derecha, filtrada.

E. FILTRO DE KALMAN

Rudolf E. Kalman nació en Budapest en 1930, emigró a Estados Unidos durante la Segunda Guerra Mundial y se doctoró en el M.I.T. en Ingeniería Eléctrica en 1954.En 1960, R.E. Kalman publicó un famoso artículo describiendo una solución recursiva al problema de filtrado lineal de datos discretos. A partir de ahí, muchas investigaciones se han beneficiado de este trabajo, especialmente en campos como la navegación asistida o autónoma.

El filtro de Kalman proporciona un buen marco para la estimación de una variable, de la que se dispone de medidas a lo largo del tiempo. Se trata de una técnica de estimación Bayesiana empleada para seguir sistemas dinámicos observados mediante sensores ruidosos.
En el ámbito de la visión artificial el filtro de Kalman es un algoritmo recursivo que se utiliza para estimar la posición de un punto o característica en movimiento y la incertidumbre de la medida, en el siguiente estado o imagen. Se trata de buscar la característica (punto, borde,

esquina, región, etc.) en un área determinada de la siguiente imagen alrededor de la posición predicha, en la que estamos seguros de encontrar la característica dentro de un cierto grado de confianza.

El filtro de Kalman estima variables de estado de un proceso con realimentación. Calcula el estado del proceso en algún instante y entonces obtiene información (se realimenta) de la medida. Por tanto, las ecuaciones del filtro se pueden clasificar en dos tipos: *actualización del tiempo* y *actualización de las medidas*. Las primeras son responsables de proyectar hacia el futuro los estimadores del estado actual y de la covarianza del error, para obtener los estimadores a priori del siguiente estado. Las ecuaciones de actualización de las medidas son responsables de la realimentación, incorporando una nueva medida a los estimadores a priori para obtener unos estimadores a posteriori mejorados. Las ecuaciones de actualización del tiempo pueden ser interpretadas como ecuaciones de *predicción*, mientras que las de actualización de la medida pueden pensarse como ecuaciones de *corrección*.

El método estima el estado $\boldsymbol{x}$ perteneciente a $\mathfrak{R}^{\boldsymbol{n}}$ de un proceso controlado en tiempo discreto que es gobernado por la ecuación diferencial del tipo:

$$\boldsymbol{x_{k+1} = A_k x_k + B u_k + w_k}$$

con una medida z correspondiente a la observación y perteneciente a $\mathfrak{R}^{\boldsymbol{m}}$ que es:

$$\boldsymbol{z_k = H_k x_k + v_k}$$

Las variables aleatorias $\boldsymbol{w_k}$ y $\boldsymbol{v_k}$ representan el ruido del proceso y de la medida respectivamente y se asume que son independientes.

La matriz $\boldsymbol{A_{NxM}}$ relaciona el estado en tiempo $\boldsymbol{k}$ con el estado en tiempo $\boldsymbol{k+1}$. Esta relación se manifiesta en las siguientes ecuaciones:

$$\boldsymbol{x_{x_{k+1}} = x_{x_k} + v_x t}$$

$$\boldsymbol{x_{y_{k+1}} = x_{y_k} + v_y t}$$

$$\boldsymbol{x_{z_{k+1}} = x_{z_k} + v_z t}$$

$$\boldsymbol{v_{x_{k+1}} = v_{x_k}}$$

$$\boldsymbol{v_{y_{k+1}} = v_{y_k}}$$

$$\boldsymbol{v_{z_{k+1}} = v_{z_k}}$$

Cuando el filtro de Kalman se aplica a la Visión Artificial, el estado ***x*** se corresponde con el vector posición del objeto en la imagen determinado por las coordenadas de posición x_x , x_y y x_z, y las coordenadas de velocidad v_x , v_y y v_z. La observación ***z*** en cambio, es únicamente un vector de tres componentes z_x , z_y y z_z, correspondiente a las coordenadas de la posición observada del objeto de interés. Dando como resultado la matriz ***A:***

$$A=\begin{pmatrix}1 & 0 & 1 & 0\\ 0 & 1 & 0 & 1\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{pmatrix}$$

La matriz B_{Nx1} relaciona la entrada control ***u*** perteneciente a $\Re^1$ con el estado ***x***.

Y la matrizH_{NxM} relaciona el estado con la medida z_k.

$$H_k=\begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\end{pmatrix}$$

El filtro de Kalman proporciona una ecuación que computa un estimador del estado a posteriori $\hat{x}_k$ como combinación lineal del estimador a priori $\hat{x}_{\bar{k}}$ y la diferencia ponderada entre la observación actual z_ky una predicción de medida$H_k\hat{x}_{\bar{k}}$:

$$\hat{x}_k=\hat{x}_{\bar{k}}+K(z_k-H_k\hat{x}_{\bar{k}})$$

La diferencia $(z_k-H_k\hat{x}_{\bar{k}})$ se llama comúnmente *innovación de la medida* o simplemente *residuo* y refleja la discrepancia entre la predicción de la medida $H_k\hat{x}_{\bar{k}}$ y la observación actual z_k. La matriz K_{NxM} llamada *ganancia de Kalman* o factor de mezcla establece la cantidad de influencia del error entre nuestra estimación y la medida:

$$K_k=P_{\bar{k}}H_k^T(H_kP_{\bar{k}}H_k^T+R_k)^{-1}$$

Siendo $P_{\bar{k}}$ el estimador de la covarianza del error a priori y R_k la covarianza del error medido. Vemos que si R_k se aproxima a **0**, la ganancia ponderará el residuo con mayor peso. Por el contrario, cuando $P_{\bar{k}}$ se aproxime a **0**, la ganancia ponderará menos el residuo.

Otra forma más intuitiva de ver la ponderación de ***K***, es que cuando la covarianza del error de medida $R_k$se aproxime a ***0***, tendremos más confianza en la observación actual z_k, mientras que la medida predicha $H_k\hat{x}_{\bar{k}}$ perderá confianza en la misma medida. Por otra parte, cuando el estimador de la covarianza del error a priori $P_{\bar{k}}$ se aproxime a ***0*** se perderá confianza en la medida z_k y la de la medida predicha $H_k\hat{x}_{\bar{k}}$ se incrementará.

En nuestro sistema $P_{\bar{k}}$ y R_k son matrices de dimensión 4x4, y Q_k es una matriz de 2x2. Matrices que inicializaremos con distintos valores para estudiar los diferentes resultados que proporcionan.

Las ecuaciones específicas para las actualizaciones del tiempo o predicción son:

$$\hat{\boldsymbol{x}}_{\overline{\boldsymbol{k+1}}} = \boldsymbol{A_k}\hat{\boldsymbol{x}}_{\overline{\boldsymbol{k}}} + \boldsymbol{Bu_k}$$

$$\hat{\boldsymbol{P}}_{\overline{\boldsymbol{k+1}}} = \boldsymbol{A_k P_k A_k^T} + \boldsymbol{Q_k}$$

Y las ecuaciones para la actualización de la medida o corrección son:

$$\boldsymbol{K_k} = \boldsymbol{P_{\overline{k}} H_k^T (H_k P_{\overline{k}} H_k^T + R_k)^{-1}}$$

$$\hat{\boldsymbol{x}}_{\boldsymbol{k}} = \hat{\boldsymbol{x}}_{\overline{\boldsymbol{k}}} + \boldsymbol{K}(\boldsymbol{z_k} - \boldsymbol{H_k}\hat{\boldsymbol{x}}_{\overline{\boldsymbol{k}}})$$

$$\boldsymbol{P_k} = (\boldsymbol{I} - \boldsymbol{K_k H_k})\boldsymbol{P_{\overline{k}}}$$

La descripción del filtro de Kalman con sus ecuaciones puede verse en el siguiente diagrama.

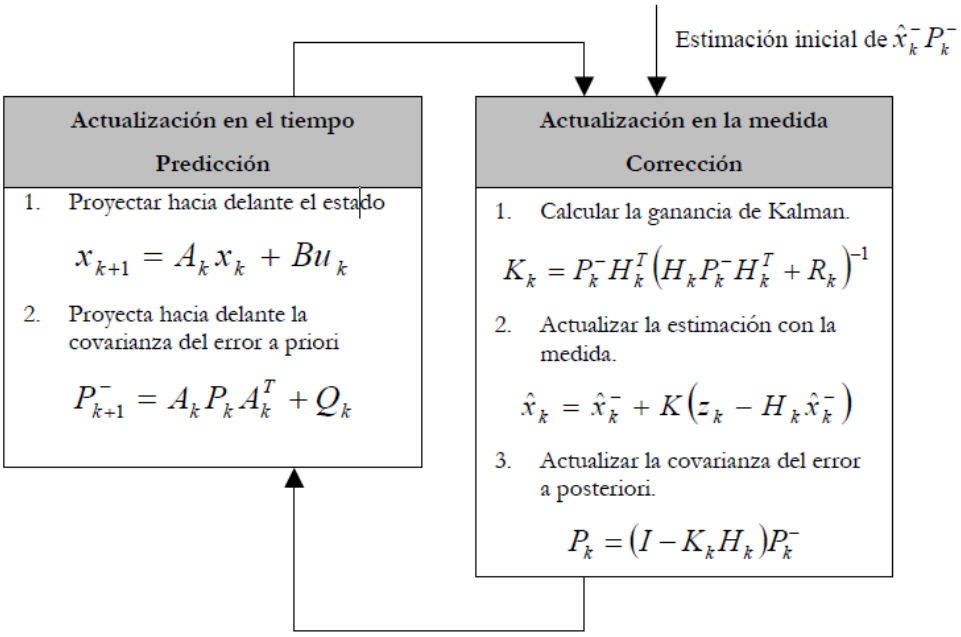


Figura 9: Esquema del filtro de Kalman

F. FILTRO DE PARTÍCULAS

1. Método de Monte Carlo

El método fue llamado así en referencia al Principado de Mónaco por ser “la capital del juego de azar”, al tomar una ruleta como un generador simple de números aleatorios.
El nombre y el desarrollo sistemático de los métodos de Monte Carlo datan aproximadamente de 1944 y fueron favorecidos por el desarrollo de la computadora electrónica.

El uso real de los métodos de Monte Carlo como una herramienta de investigación, viene del trabajo de la bomba atómica durante la Segunda Guerra Mundial. Este trabajo involucraba la simulación directa de problemas probabilísticos de hidrodinámica concernientes a la difusión aleatoria de neutrones en material de fusión.

Aún en la primera etapa de estas investigaciones, John von Neumann y Stanislao Ulam refinaron esta curiosa “Ruleta rusa” y los métodos “de división”. Sin embargo, el desarrollo sistemático de estas ideas tuvo que esperar el trabajo de Harris y Herman Kahn en 1948. Aproximadamente en el mismo año, Fermi, Metropolis y Ulam obtuvieron estimadores para los valores característicos de la ecuación de Schrödinger para la captura de neutrones a nivel nuclear.

El Método de Monte Carlo da solución a una gran variedad de problemas matemáticos posibilitando la realización de experimentos con muestreos estadísticos en una computadora. El método es aplicable a cualquier tipo de problema, ya sea estocástico o determinístico. A diferencia de los métodos numéricos que se basan en evaluaciones en N puntos en un espacio M-dimensional para producir una solución aproximada, el método de Monte Carlo tiene un error absoluto en la estimación que decrece en $1/\sqrt{N}$ en virtud del Teorema Central de Límite.

En la práctica, las pruebas aleatorias se sustituyen por resultados de ciertos cálculos realizados con números aleatorios.

2. Filtro de Partículas

El Filtro de Partículas es un método secuencial de Monte Carlo aplicable a cualquier transición de estados o modelo de medida, y en la visión artificial se utiliza para el seguimiento de objetos en secuencias de imágenes.

Lo propusieron N. Gordon, D. Salmond y A. Smith en 1993 [27], como Filtro Bootstrap para implementar Filtros Bayesianos recursivos. La densidad requerida del vector de estados era representada como un conjunto de muestras aleatorias, las cuales se actualizan y propagan por el algoritmo. El filtro de partículas supone una nueva manera de representar y generar recursivamente una aproximación del estado de la Función de Densidad de Probabilidad (FDP). La idea central es representar la FDP requerida como un conjunto de muestras aleatorias, más que como una función sobre el espacio de estados. La aproximación Bayesiana se utiliza para construir la FDP, que define el estado del objeto en el instante actual.

El Filtro de Partículas representa la densidad a posteriori mediante una distribución de partículas (muestras) en el espacio de estados. Las partículas son estados posibles del proceso, que se pueden representar como puntos en el espacio de estados de dicho proceso. Este enfoque se ha desarrollado de forma independiente en los últimos años en campos como la estadística, la economía o la visión artificial (Kitagawa 1987 [28]; West 1992 [29]; Gordon, Salmond, y Smith 1993 [30]; Isard y Blake 1996 [31]; Kitagawa 1996 [32]; Carpenter, Clifford, y Fernhead 1997 [33]; Pitt y Shephard 1999 [34]).

Los nombres con los que se le ha denominado son: Método Secuencial de Monte Carlo (Sequential Monte- Carlo Methods), Algoritmo Condensation (Condensation Algorithm), Filtro Bootstrap (Bootstrap Filter), Filtro de la Supervivencia del Más Apto, aunque últimamente se está utilizando el término Filtro de Partículas (Particle Filtres) para todos ellos. Todos estos algoritmos son propuestas similares que propagan las partículas (muestras de la función de densidad a posteriori), utilizando:

- el modelo de movimiento: $p(x_t|x_{t-1}, a_t)$
- el modelo de verosimilitud:$p(x_t|x_t)$

de forma que el peso combinado de las partículas de una región aproxima la integral de la función de densidad a posteriori en esa región. En concreto, el Filtro de Partículas representa la densidad a posteriori mediante un conjunto discreto de N partículas $(m_t, \dots, m_N)$ y sus probabilidades asociadas $(\pi_t, \dots, \pi_N)$.

Inicialmente, el conjunto de partículas se escoge a partir de la distribución a priori $p(x_0)$. Si no existe información a priori, entonces las partículas se distribuyen uniformemente por el espacio de estados. Posteriormente, en cada instante de tiempo t, se actualizan las N partículas en función de la acción anterior a_{t-1} y la observación actual z_k. Para ello, se aplica el modelo de movimiento $p(x_t|x_{t-1}, a_{t-1})$ a cada una de las N partículas, generando un nuevo conjunto de partículas. Las partículas nuevas representan la predicción de la variable de estado, sin considerar la observación, se obtiene el peso π^i asociado a cada partícula.

El conjunto de pesos de cada partícula es proporcional a la probabilidad de su estado y a la suma normalizada de sus pesos. La densidad de los pesos es igual al producto de la densidad previa (del muestreo) y la probabilidad (de los pesos). En la Figura 10 se representa un muestreo discreto de una función de densidad de probabilidad continua mediante partículas, cuyo tamaño hace referencia al peso asignado a las mismas.

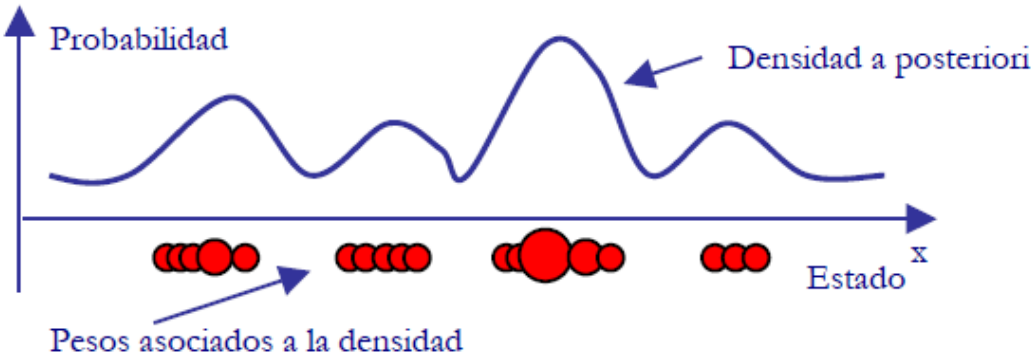


Figura 10: Representa el muestreo y los pesos obtenidos al aplicar el Filtro.

En un último paso, se remuestrea el conjunto de partículas, extrayendo (con reemplazo) N partículas del conjunto actual, proporcional al peso de cada una. En este nuevo conjunto tendrán más probabilidad de desaparecer aquellas partículas para las que no hay evidencia de verosimilitud o, lo que es lo mismo, que tenga menor peso. Una vez construido el nuevo conjunto de partículas, según la probabilidad de éstas se asocia un peso a cada una. Este nuevo conjunto de partículas constituye una representación muestral de la probabilidad a posteriori.

En la Figura 11, se representa gráficamente la evolución de las partículas en cada fase, suponiendo que las partículas están estimando un único parámetro, distribuido en el eje horizontal. El área de los círculos representa el peso de cada partícula.

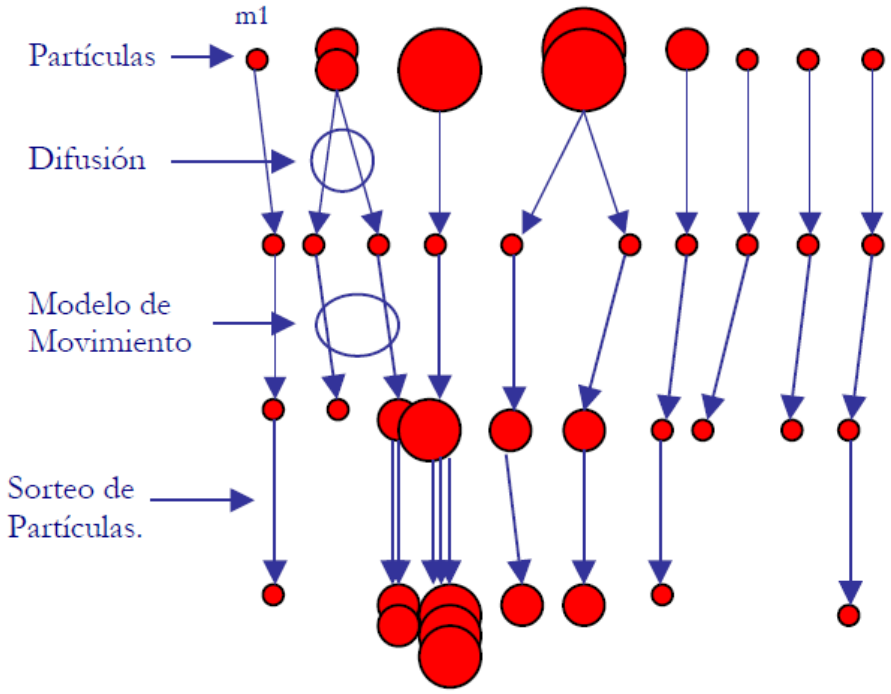


Figura 11: Funcionamiento del Filtro de Partículas.

Los elementos que se consideran en el método del Filtro de Partículas son:

- Los vectores de medidas Z, en este caso, serán provenientes de las imágenes. Las medidas dependen del estado del objeto, y el estado del objeto se deriva estadísticamente de las medidas.

- El Modelo de Movimiento F, se utiliza para predecir la posición del objeto en el instante actual, a partir de la densidad de probabilidad del instante anterior ($x_{t+1} = F(x_t)$).

- El Modelo de Verosimilitud $P(Z_t|x_t)$, la estimación de la función de probabilidad condicional a posteriori $P(Z_t|x_t)$ define la verosimilitud de la medida observada dado un punto del espacio de estados (es decir, dado un estado del sistema).

Para poder implementar el algoritmo de actualización es necesario un conjunto bien ponderado de partículas en el instante t con pesos iguales $1/\mathrm{N}$, y así actualizar este conjunto para reflejar las nuevas medidas obtenidas en el instante t +δt . El algoritmo de actualización queda como sigue:

1. Propagar cada partícula m_i en el tiempo utilizando el modelo de movimiento F del objeto para obtener un conjunto actualizado de partículas $\{m_t^*\}$.

2. Obtener un nuevo vector de medidas Z y evaluar la densidad de probabilidad posterior π_t^* para cada m_t^*, $\pi_t^* = p(m_t^*|Z)$ que cuantifica la verosimilitud de (un estado) m_t^* dado un vector de medidas Z. Esto puede ser escrito usando la regla de Bayes anteriormente descrita:

$$\mathrm{Pr}(\mathrm{m_t^*}|\mathrm{Z}) = \frac{\mathrm{p}(\mathrm{Z}|\mathrm{m_t^*})\mathrm{p}(\mathrm{m_t^*})}{\mathrm{p}(\mathrm{Z})}$$

Donde:

$\mathrm{p}(\mathrm{Z})$ es la probabilidad a priori de la medida que se asume constante y conocida.

$\mathrm{p}(\mathrm{m_t^*}) = \frac{1}{\mathrm{N}}$, por lo que: $\mathrm{P}(\mathrm{m_t^*}|\mathrm{Z}) = \mathrm{Kp}(\mathrm{Z}|\mathrm{m_t^*})$ donde $\mathrm{p}(\mathrm{Z}|\mathrm{m_t^*})$ puede ser calculado sin inversión de las ecuaciones de medida.

3. Volver a muestrear a partir del conjunto $\{m_t^*\}$ con probabilidades π_t^* y generar un nuevo conjunto bien ponderado $\{m_t\}$ con pesos iguales $(\frac{1}{N})$ para cada partícula. El muestreo de cada partícula consiste en un estado, un peso y otra información posible (una covarianza por instante).

4. Repetir los pasos 1 y 3 para instantes sucesivos.

En la Figura 12, se puede ver cómo evolucionan las partículas en el tiempo durante una ejecución del Filtro de Partículas.

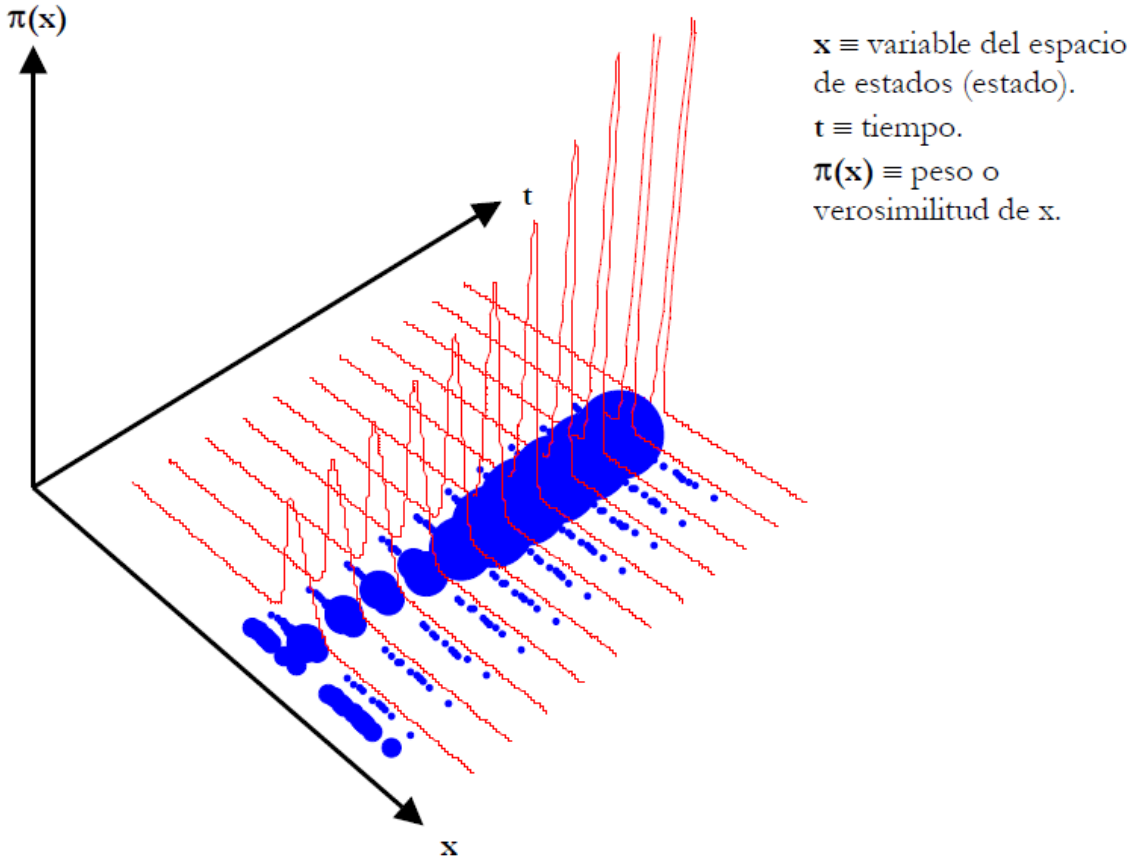


Figura 12: Evolución de la densidad de probabilidad y de su modelado a través de partículas.

El Filtro de Partículas representa una poderosa herramienta para el tratamiento de procesos del mundo real, evitando hacer cualquier tipo de suposiciones sobre las características intrínsecas del proceso como hacen los filtros clásicos, como el Filtro de Kalman.

La diferencia fundamental entre ambos algoritmos estriba en el hecho de que los filtros de Kalman pueden representar solamente la estimación del estado por una gausiana uni-modal, los Filtros de Partículas pueden representar densidades multi-modales complejas empleando una gran cantidad de partículas aleatoriamente muestreadas.

G. PIN HOLE

El modelo pin hole es un modelo con el que se realizaban las fotografías en el pasado. El objetivo era captar un objeto tridimensional en una fotografía (2 dimensiones). Las cámaras basadas en el modelo pin hole consistían en una caja estanca , donde se introducía un objeto fotosensible a la luz, y a la que se le abría un pequeño orificio por donde se dejaba entrar la luz durante un periodo de tiempo suficiente para que en el objeto fotosensible se quedara impresa la fotografía.

Con este método de fotografía, a cada punto de la fotografía le corresponde un punto del objeto captado, como se observa en la siguiente imagen.

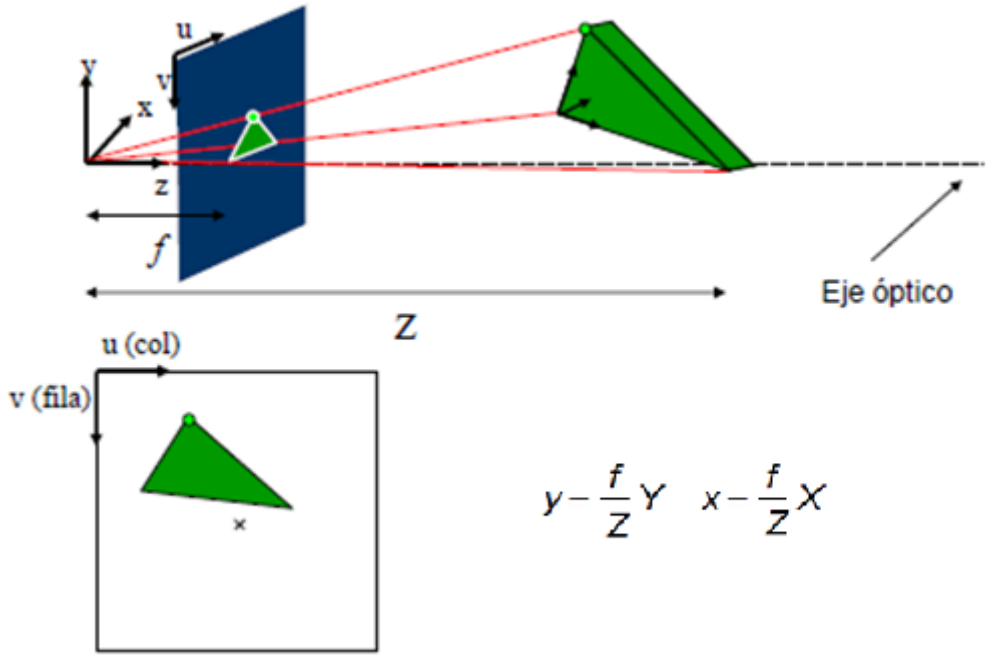


Figura 13: Modelo pin hole

Para conocer esa correspondencia entre puntos, hay que basarse en el modelo pin hole, que es lo que hemos hecho en este proyecto:

Una vez realizada la predicción mediante el Filtro de Kalman o el Filtro de Partículas, abrimos un interfaz donde se muestra la predicción del filtro recuadrada para que sea visible para el usuario. Esta interfaz que muestra la predicción es una imagen 2D, pero todo el proceso que nos lleva hasta la predicción se realiza mediante el tratamiento de los datos tal y como los proporciona la cámara, es decir, datos en 3D; por lo que es necesario hacer una transformación de la nube de puntos en 3D a una imagen en 2D.

Además, los filtro de Kalman y de Partículas realizan una predicción cuyo resultado son los valores correspondientes a las 3 dimensiones (X, Y y Z), pero para mostrarlo en la imagen 2D hay que buscar la correspondencia entre ese punto de la nube en 3D, con ese mismo punto de la imagen 2D, y es para eso para lo que utilizamos el modelo pin hole.

VI. HERRAMIENTAS USADAS

A. KINECT

La cámara Kinect es un dispositivo que creó Alex Kipman para la compañía Microsoft, enfocada a integrarla en su videoconsola Xbox 360. Su presentación fue en la Expo Electronic Entertainment de 2009 y se anunció como la nueva generación de entretenimiento en el hogar, ya que mediante una cámara puedes ser el protagonista, en primera persona, de un videojuego.

El invento de Kipman tuvo gran aceptación y no sólo como una cámara para la videoconsola, sino que hoy en día se está integrando en muchos robots por ser una cámara de captura tridimensional a un precio asequible.

1. ESPECIFICACIONES TÉCNICAS

Las especificaciones técnicas de la Kinect son las siguientes:

- Sensores y actuadores
  - La Kinect cuenta con dos cámaras de tecnología CMOS:
    - ✓ Una cámara RGB cuyo objetivo es captar la información de color de todo aquello que se encuentre dentro de su rango de visión
    - ✓ Una cámara de infrarrojos cercanos, NIR, la cual está complementada con una fuente de luz infrarroja que dota a la Kinect de la capacidad de obtener datos incluso en ausencia de luz. Esta cámara infrarroja de corto alcance es la que obtiene los datos de profundidad.

Figura 14: Cámara kinect de Microsoft

- El dispositivo Kinect también está dotado de cuatro micrófonos colocados horizontalmente que lo que hacen es situar la fuente de sonido, así como eliminar el ruido ambiente.
- En la base, hay colocado un pequeño motor que permite el movimiento ascendente-descendente de la cámara. Así la Kinect sigue a su objetivo a lo largo de sus movimientos, aumentando el rango de visión. El motor de la base esta suplementado con un acelerador de tres ejes para aumentar la precisión en el movimiento del motor.
- Cuenta también con un ventilador interno para la disipación del calor.

El procesamiento de los mapas lo realiza mediante el Chip PrimeSense PS1080-A2. En la siguiente imagen se puede observar un despiece de la cámara Kinect de Microsoft, con todos sus componentes.

Figura 15: Despiece de la cámara Kinect.

- Rango de Visión
  - En horizontal: 57 grados.
  - En vertical: 43 grados.
  - Rango de inclinación (gracias a los motores): ±27 grados.
  - Profundidad de detección: Para la obtención de unos datos fiables el objeto captado debe estar a una distancia de entre 0.8m y hasta 8m.

- Flujo de datos

    - Sensor de profundidad: 320 x 240 y 16 bit por pixel, velocidad 30 fps
    - Sensor a color: 640 x 480 y 32 bit por pixel, velocidad 30 fps
    - Audio: 16 bits y velocidad 16 kHz

2. MÉTODO DE OBTENCIÓN DE LA PROFUNDIDAD

La mayoría de las cámaras de obtención de datos de profundidad mediante rayos infrarrojos se basan en el "tiempo de vuelo" para hacer la captura. Este método consiste en emitir un haz de rayos infrarrojos que rebota en un objeto y vuelve a la cámara. La distancia que hay entre el objeto a la cámara se calcula contando el tiempo que tarda el rayo en ir y volver, como ya se explicó en el apartado IV.A.1.c . Las cámaras que se basan en este método tienen un elevado coste ya que los tiempos medidos son muy pequeños, y por tanto requieren potentes y precisos cronómetros incorporados.

Otro método en el que se basan las cámaras infrarrojas consiste en medir el desfase de la longitud de onda que se produce entre el rayo que se emite, y el que recibe la cámara.

La Kinect ha conseguido ser una cámara de datos de profundidad que no tiene un precio tan elevado, esto es gracias a que el sistema de obtención los datos no es ninguno de los dos métodos mencionados anteriormente. Para medir la distancia no se centra solo en la posición del objeto, sino también en la detección y codificación de los haces de luz reflejada por los objetos.

La cámara infrarroja emite un rayo de luz conocido que al incidir en un objeto forma una malla. Los datos de distancia de cada punto los calcula midiendo la deformación de los puntos que componen esa malla.

Figura 16: Método de obtención de la profundidad

El chip PrimeSense PS1080-A2 de procesamiento de imágenes descompone la imagen eliminando el ruido que producen los rayos infrarrojos reflejados por la luz del sol.

La medición de profundidad se realiza haciendo una triangulación de cada punto de la malla infrarroja que se lanzó, con su correspondiente punto en la malla recibida. Con esto se genera el mapa de profundidad.

B. SOFTWARE

1. LINUX

GNU/Linux es uno de los términos empleados para referirse a la combinación del núcleo o *kernel* libre similar a Unix denominado Linux con el sistema GNU. Su desarrollo es uno de los ejemplos más prominentes de software libre; todo su código fuente puede ser utilizado, modificado y redistribuido libremente por cualquiera bajo los términos de la GPL (Licencia Pública General de GNU) y otra serie de licencias libres.

En el proyecto, la distribución Debian GNU/Linux utilizada ha sido Ubuntu, que es un sistema operativo completo basado en GNU/Linux, disponible de forma libre con soporte para la comunidad y los profesionales, que permite realizar todas las tareas diarias que actualmente se llevan a cabo con un ordenador.

Que sea un software libre quiere decir que cumple las siguientes libertades:

1. Libertad de usar el programa, con cualquier propósito.
2. Libertad de estudiar cómo funciona el programa y modificarlo, adaptándolo a tus necesidades.
3. Libertad de distribuir copias del programa.
4. Libertad de mejorar el programa y hacer públicas esas mejoras a los demás, de modo que toda la comunidad se beneficie.

2. PCL

PCL (Point Cloud Library) es un proyecto independiente de código abierto que incluye numerosos y avanzados algoritmos para nubes de puntos 3D y procesamiento de la geometría . El desarrollo de esta biblioteca se inició en marzo de 2010 a manos de Willow Garage , y el lanzamiento oficial de PCL fue en mayo de 2011.

PCL está escrito en C + + y liberado bajo la licencia BSD . Fue desarrollado por un gran consorcio de investigadores e ingenieros de todo el mundo. La biblioteca contiene algoritmos para el filtrado, la estimación, la reconstrucción de superficies, la segmentación, etc. Estos algoritmos se pueden utilizar para la percepción. En el caso de la robótica para casos como filtrar los valores extremos de los datos ruidosos, concatenar nubes de puntos 3D, crear superficies a partir de nubes de puntos y visualizarlos, segmentar partes importantes de una escena, extraer puntos clave y los descriptores de cómputo para reconocer objetos en el mundo en base a su apariencia geométrica, y demás aplicaciones.

3. BIBLIOTECA DE FILTRADO BAYESIANO

Orocos (Open RObot COntrol Software) es un proyecto que surgió con el objetivo de desarrollar un software libre de propósito general con un marco de trabajo modular para controlar robots. Este proyecto tiene librerías escritas en C++, una de ellas es la biblioteca de filtrado bayesiano, que es la que usamos en nuestro proyecto.

La biblioteca de filtrado bayesiano, BFL (Bayesian Filtering Library) proporciona algoritmos para el procesamiento de información recursiva y algoritmos de estimación basados en el teorema de Bayes.

Algunos de los algoritmos que proporciona esta biblioteca son el filtro de Kalman (normal y extendido), el filtro de partículas (normal y extendido también), métodos secuenciales de Monte Carlo, etcétera. Estos algoritmos se pueden ejecutar en tiempo real y ser utilizados para la estimación cinemática y dinámica de aplicaciones.

Esta biblioteca ha sido muy importante para la consecución del proyecto, ya que de ella se obtuvieron los filtros de Kalman y de Partículas.

4. BIBLIOTECA OPEN CV

OpenCV es una biblioteca libre de visión artificial originalmente desarrollada por Intel Corporation en 1999 para el tratamiento de imágenes. Desde que apareció su primera versión se ha utilizado en infinidad de aplicaciones, desde sistemas de seguridad con detección de movimiento, hasta aplicaciones de control de procesos donde se requiere reconocimiento de objetos. Esto se debe a que su publicación se da bajo licencia BSD, que permite que sea usada libremente para propósitos comerciales y de investigación con las condiciones en ella expresadas.

Las funciones y clases que componen la librería están escritas en C y C++ y además son multiplataforma, pudiéndose utilizar tanto en aplicaciones para GNU/Linux, MacOS y Microsoft.

Una de las principales ventajas de ser una librería de código abierto es su gran uso por parte de comunidades que comparten sus experiencias y códigos con los que ayudar en su uso y aprendizaje

La principal característica de OpenCV es la funcionalidad, claridad y sencillez de uso de sus funciones. Los algoritmos están basados en estructuras de datos. Dispone de más de 500 funciones dedicadas a:

-Procesamiento general de imágenes.

- Segmentación

- Descriptores de geometría

- Imágenes Piramidales

- Calibración de cámaras

-Transformaciones geométricas

-Detección de rostros

- Máquinas de aprendizaje

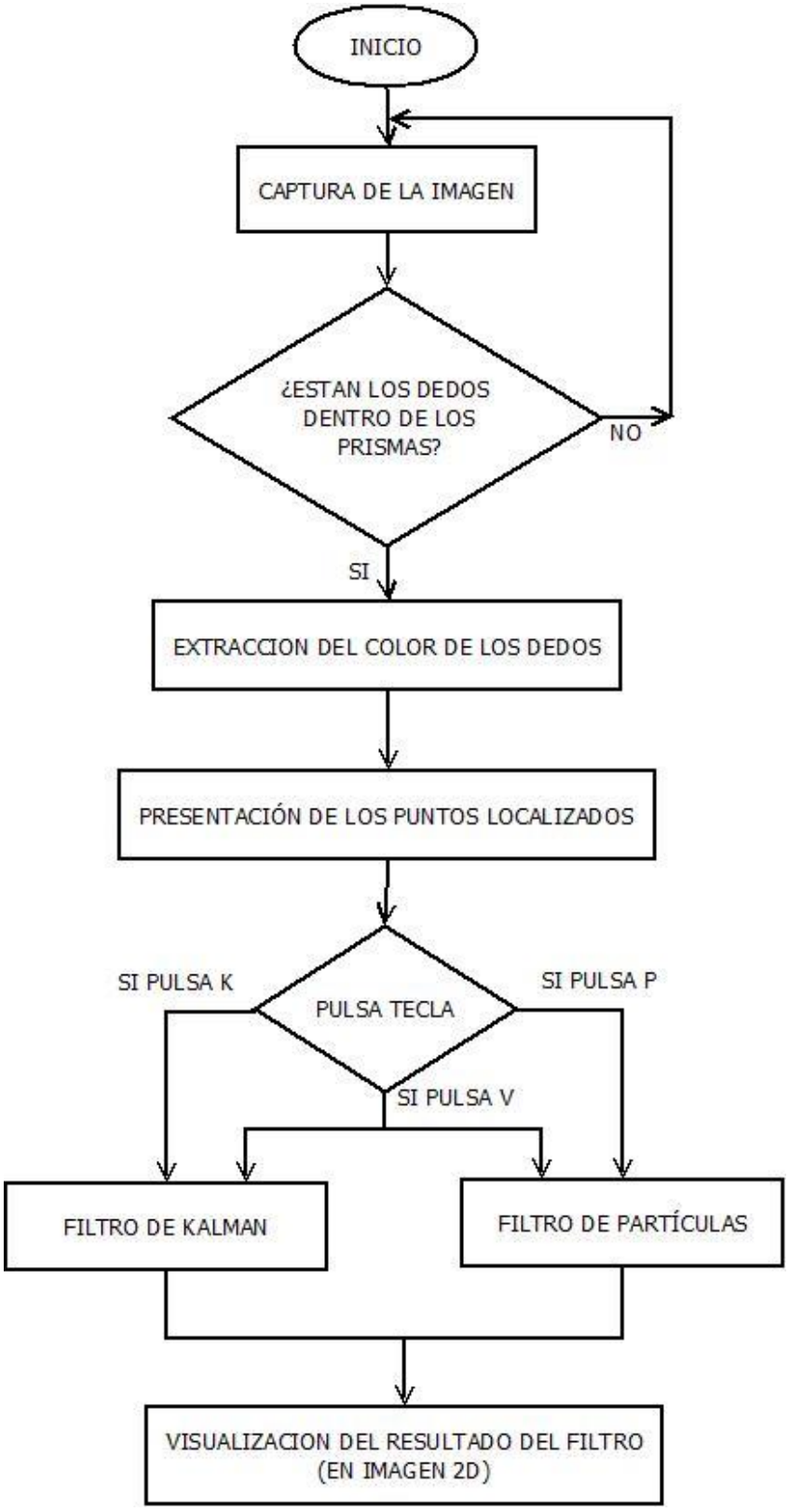
- Etc.

En nuestro caso utilizaremos estas librerías para transformar las nubes de puntos obtenidas mediante Kinect en imágenes bidimensionales y para representar los puntos del resultado de los filtros de Kalman y de Partículas.

VII. MÉTODO DE LA SOLUCIÓN DISEÑADA

A. DIAGRAMA DE ESTADOS

A continuación se muestra un diagrama de estados resumido del algoritmo desarrollado.



Cada uno de esos estados será explicado detalladamente a continuación, y si es preciso, también se mostraran un diagrama de ellos.

B. CAPTURA DE LA IMAGEN

La adquisición de imágenes en tiempo real se realiza mediante la cámara Kinect, la cual nos da la información del entorno en forma de nubes de puntos. Esta información se transfiere al ordenador, donde el tratamiento de los datos lo realizamos con la librería PCL, orientada al tratamiento de nubes de puntos.

Las nubes de puntos que captura la Kinect tienen una resolución de 640x480, lo cual nos da un total de 307200 puntos. Cada punto tiene tres coordenadas cartesianas (XYZ) y tres coordenadas de color (RGB).

A continuación se muestra un ejemplo de una nube de puntos capturada por la Kinect.

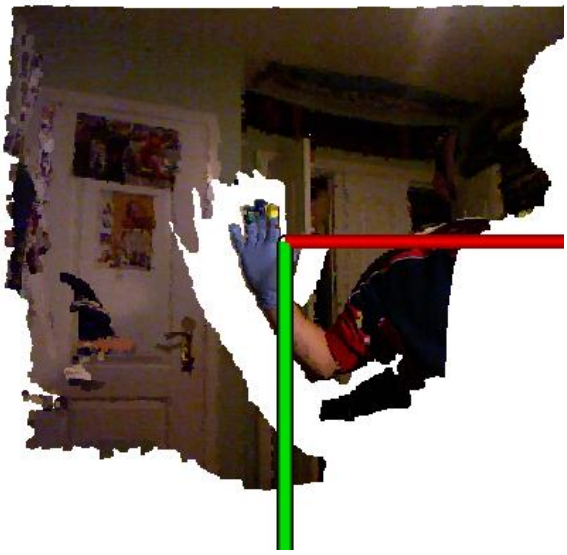


Figura 17: Ejemplo de una nube de puntos

C. LOCALIZACIÓN DE LOS PUNTOS CARACTERÍSTICOS DE LA MANO

Para la realización de este proyecto, hemos utilizado un guante, y cada dedo ha sido etiquetado con un color diferente, tal y como se muestra en la siguiente imagen:

Figura 18: Guante utilizado en el proyecto.

El objetivo principal es hacer un seguimiento de los dedos mediante el Filtro de Kalman o el Filtro de Partículas, para ello, lo primero que tenemos que hacer es localizar cada uno de los cinco dedos de la mano. Para ello, hemos diseñado un algoritmo que consiste en:

Mostramos por pantalla cinco prismas vacíos, como se ve en la siguiente imagen:

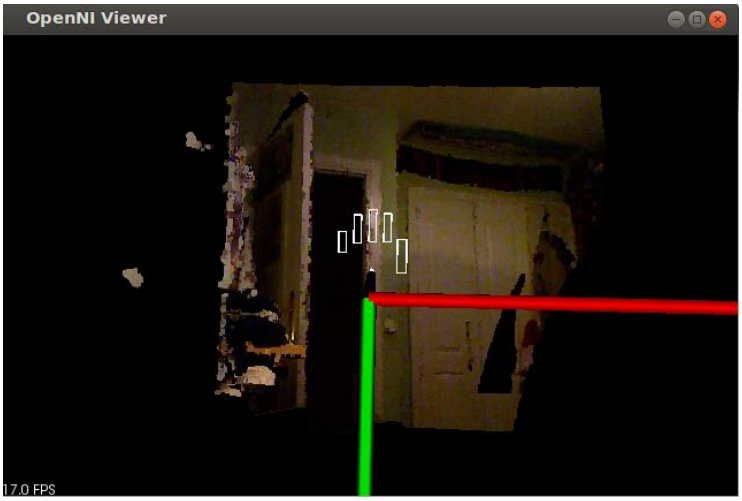


Figura 19: Prismas de los dedos vacíos

Tenemos que meter los cinco dedos a la vez en cada uno de los prismas, tal y como se muestra en la siguiente imagen:

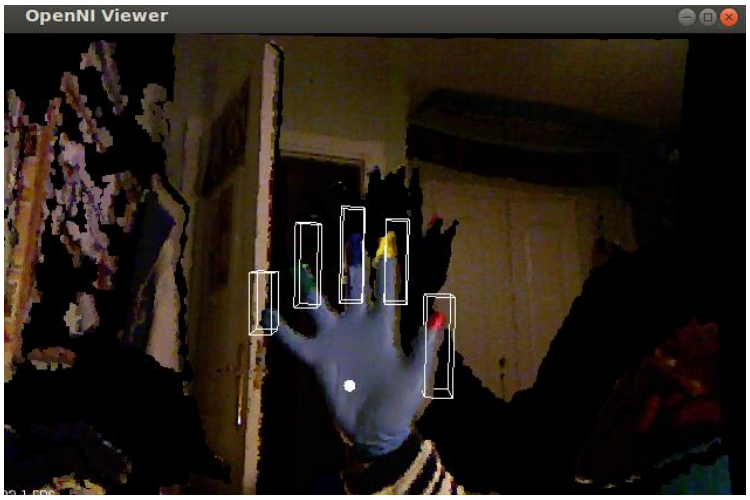


Figura 20: Prismas con los dedos introducidos ampliada.

El programa no considera que esté introducido el dedo en el prisma hasta que no haya un mínimo de 50 puntos dentro del prisma. En la terminal mostramos cuantos puntos hay introducidos en cada prisma para facilitar al usuario la acción de meter todos los dedos. En la siguiente figura se ve un ejemplo en el que están todos los dedos introducidos en los prismas:

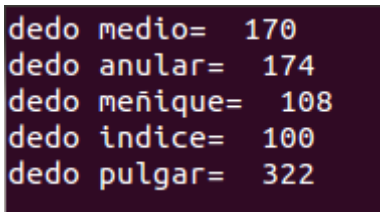


Figura 21: Vista de la terminal con todos los dedos dentro de los prismas.

1. LOCALIZACIÓN DEL PUNTO DE MAYOR ALTURA DE CADA PRISMA

Una vez estén metidos todos los dedos al mismo tiempo en los prismas, el programa busca, para cada dedo (cada prisma), cual es el punto de mayor altura.

Para ello, compara la coordenada “Y” de todos los puntos de cada prisma, y la que mayor sea, esa será la posición del punto más alto.

En la siguiente figura se muestra para una captura, cuáles serían los puntos localizados para cada dedo:

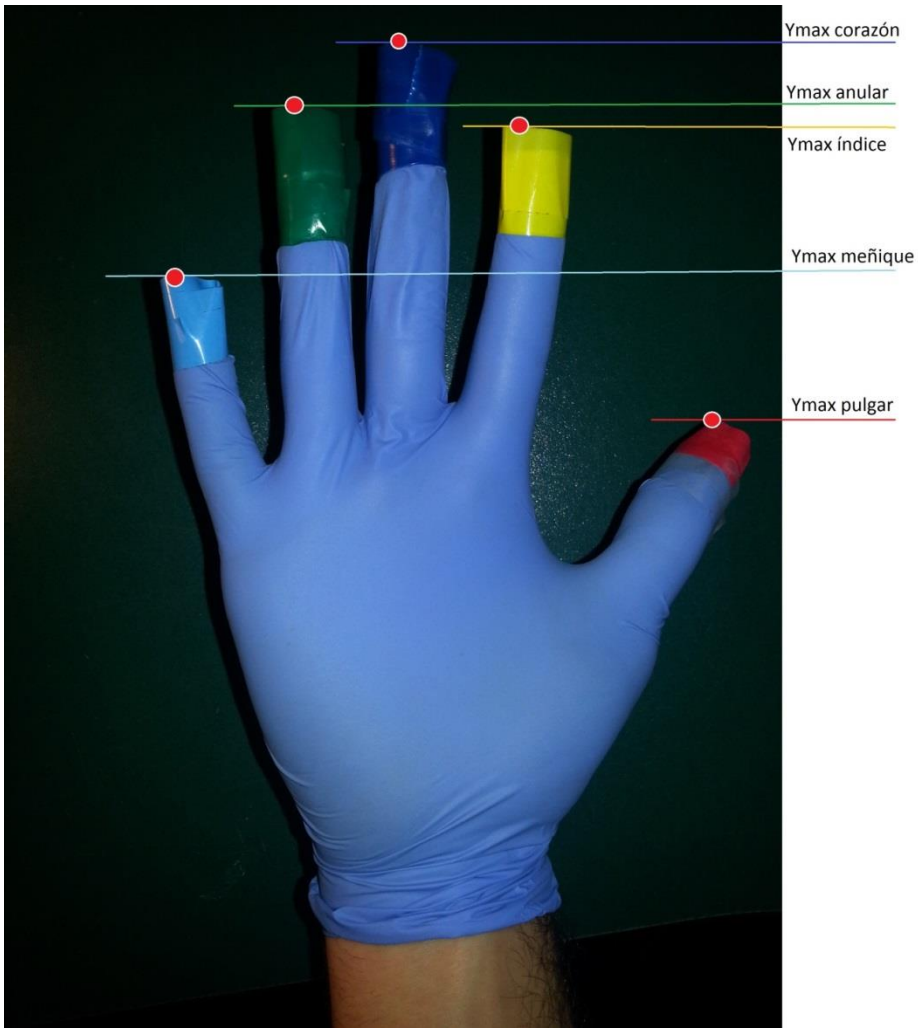


Figura 22: Puntos localizados mediante Y máxima.

2. EXTRACCIÓN DEL COLOR DE LOS DEDOS

Cuando ya se ha localizado el punto de mayor altura de cada dedo, el programa tiene que asignar a cada dedo su color. Ese color se determinará mediante los pasos que a continuación se detallan:

a) MEDIA ARITMÉTICA

Partiendo del punto localizado, se establece un pequeño cuadrado de 1cm de lado, cuya arista superior está centrada en el punto:

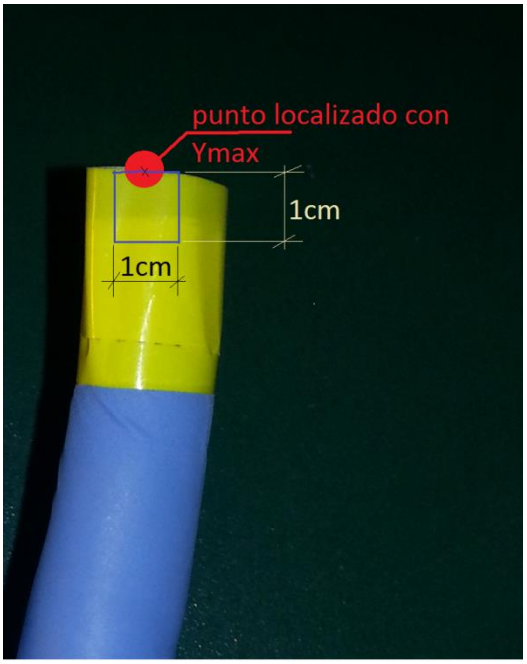


Figura 23: Cuadrado dibujado en torno a cada punto localizado.

El algoritmo está programado para que cuente todos los puntos que hay dentro del cuadrado, extraiga la información de color de cada uno de ellos, y haga una media aritmética recortada un 10% (tal y como se explicó en el apartado IV.B) de cada una de las componente de color (RGB), de tal forma que ya sabremos de qué color es cada dedo.

b) CONVERSION DE RGB A HSV

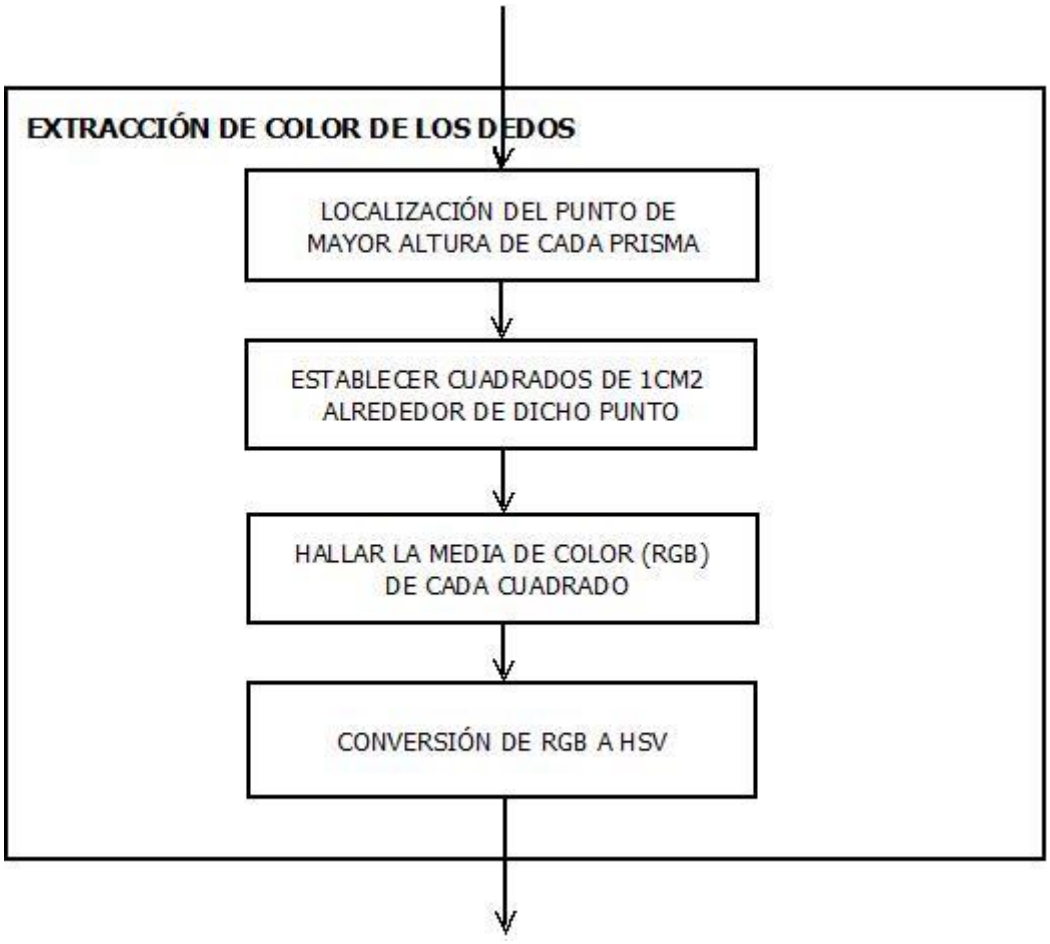
Como la información de color en el espacio RGB no era suficientemente fiable, decidimos hacer una conversión del resultado de las medias de R, G y B de cada dedo al espacio de color HSV.

A continuación se muestra un ejemplo de lo que se ve en la terminal de una extracción de color de los dedos, donde está el número de puntos que tenía cada cuadrado, y las componentes H, S y V resultantes de hacer la media de todos esos puntos, y convertirlos al espacio de color HSV:

```
puntos corazon: 31
h_corazon: 203.837 s_corazon: 0.722004 v_corazon: 0.159686
puntos anular: 30
h_anular: 171.798 s_anular: 0.807867 v_anular: 0.108007
puntos menique: 33
h_menique: 136.984 s_menique: 0.168425 v_menique: 0.371678
puntos indice: 53
h_indice: 28.7767 s_indice: 0.843867 v_indice: 0.389603
puntos pulgar: 53
h_pulgar: 209.657 s_pulgar: 0.747648 v_pulgar: 0.24128
```

Figura 24: Extracción de color de los dedos.

Todo este proceso queda resumido con el siguiente diagrama:



Una vez llegados a este punto, el programa ya sabe cuáles son las coordenadas HSV de cada dedo, de tal forma que el seguimiento de los dedos se hará buscando en cada actualización de

la nube, un conjunto de puntos que tengan coordenadas de HSV muy parecidas a las establecidas.

D. PRESENTACIÓN DE LOS PUNTOS LOCALIZADOS

Cuando ya estén localizados los puntos de todos los dedos, el programa cambia de visualizador para enseñar por pantalla al usuario el resultado mediante pequeñas esferas situadas en los puntos localizados, como se muestra en las siguientes imágenes.

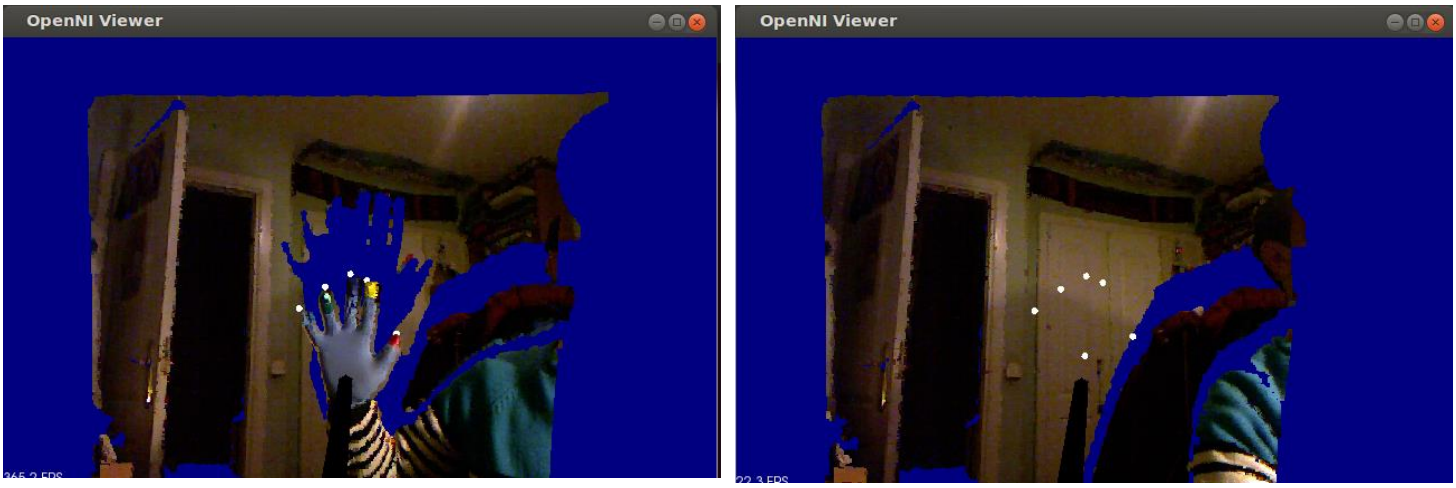


Figura 25: Visualizador que muestra donde han sido localizados los dedos.

El único objeto que tiene este paso es que el usuario se cerciore de que los puntos han sido bien localizados, previniendo así un mal funcionamiento del programa en los pasos posteriores.

Una vez se haya corroborado la localización correcta de los dedos, el programa espera a que el usuario pulse la tecla “b” para seguir adelante en el algoritmo.

Lo siguiente que hará el programa será aplicar uno de los filtros programados. Para ello, se le deja elegir al usuario si lo que quiere es aplicar el Filtro de Kalman, el Filtro de Partículas, o si lo que quiere es que se apliquen ambos filtros simultáneamente.

La siguiente imagen es una captura de pantalla del terminal en el momento de elección de modo de predicción. En este caso se pulsó la "v", conduciéndonos al modo de comparación de filtros.

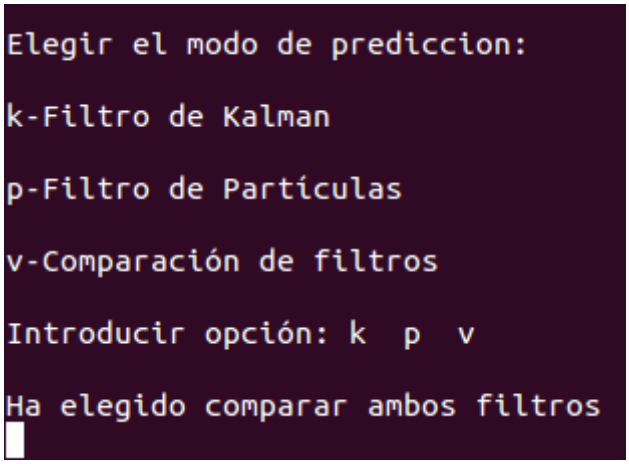


Figura 26: Elección de modo de predicción.

E. FILTRO DE KALMAN

El objetivo del filtro de Kalman es predecir las posiciones en las que van a estar los dedos de una manera óptima, minimizando el índice del error cuadrático medio.

El filtro de Kalman que hemos incorporado a nuestro código es el que proporciona la librería BFL*, The Bayesian Filtering Library*.

Uno de los objetivos del proyecto era poder realizar la predicción para estados de oclusión, es decir, que cuando los dedos dejen de verse, el filtro siga prediciendo la posición en la que debería estar el dedo. Para hacer esto, hemos distinguido dos modos de funcionamiento para nuestro filtro de Kalman, un primer modo para cuando la mano está visible; y otro segundo modo de seguimiento, para cuando la mano está oculta.

Para los dos modos del filtro de Kalman a la nube de puntos se le aplica un filtro de profundidad para eliminar todos los puntos que estén a más de 1.5m de la cámara, para reducir la carga computacional del filtro.

Para saber en cuál de los dos modos de seguimiento está el filtro, lo primero es localizar los dedos, si los encontramos, nos meteremos en el modo de mano visible; por lo contrario, si no están en el rango de visión de la Kinect, el filtro se meterá en el modo de oclusión.

Para localizar los dedos, lo que hacemos es buscar en todos los puntos de la nube, aquellos cuya componente de color HSV sean similares a las que ya están definidas para cada dedo (como se detalló en el apartado VI.C.2, “Extracción del color de los dedos”).

1. MANO VISIBLE

Si se han encontrado puntos que coincidan con el color del dedo, lo que se hace es calcular el centro de gravedad de todos los puntos que hayan sido localizados. Ese centro de gravedad será el punto que querremos predecir en el estado siguiente.

También es necesario calcular la velocidad que lleva ese punto en el estado actual. Dicha velocidad, al ser tomada en intervalos de tiempo muy pequeños, la consideramos constante. Se calcula la velocidad independientemente para cada componente XYZ, para ello aplicamos la fórmula siguiente:

$$\boldsymbol{v} = \frac{\boldsymbol{CG} - \boldsymbol{C\acute{G}}}{\boldsymbol{t}}$$

Donde $\boldsymbol{CG}$ es el centro de gravedad del estado actual, $\boldsymbol{C\acute{G}}$ es el centro de gravedad del estado anterior, y t es el tiempo que transcurre entre los dos cálculos del centro de gravedad.

Una vez ya sabemos la posición del punto (CG), y la velocidad (v) en el estado actual, ya podemos aplicar el filtro que proporciona la librería de Orocos, y que nos dará para un valor de predicción para el estado siguiente.

Todo este proceso se repite para cada dedo individualmente. A continuación se muestra un ejemplo de una predicción, con su respectiva posición y velocidad para el dedo corazón:

```
FILTRO KALMAN VISIBLE CORAZÓN
medida Kalman [3](0.0193338,-0.0217105,0.53)
velocidad Kalman [3](-0.0167603,0.00752817,-0.0214217)
 Prediccion Kalman =
[3](0.0113471,-0.0141261,0.491771)
 Covarianza Kalman =
[3,3]((0.00235329,0,0),(0,0.00235329,0),(0,0,0.00235329))
```

Figura 27: Predicción del filtro de Kalman en estado visible

La “medida Kalman” es la posición del centro de gravedad del dedo corazón en el estado actual, que en ese momento lleva esa velocidad. Y la “predicción Kalman” es la posición en la que el filtro de Kalman predice que estará el centro de gravedad del dedo corazón en el estado siguiente.

2. MANO EN ESTADO DE OCLUSION

Si a la hora de localizar los dedos no fue encontrado ningún punto del mismo color que los dedos, eso quiere decir que los dedos están fuera del rango de visión de la cámara.

Si eso ocurriera, se activa el modo del filtro de Kalman en estado de oclusión, y para poder predecir en este caso, es necesario hacer dos aproximaciones:

1. Asumir que la velocidad que va a llevar el dedo durante el tiempo de oclusión, va a ser constante e igual a la velocidad que llevaba el dedo en el estado anterior a estar oculto.
2. Al no saber la posición real del dedo en el estado actual, se considera como válida la predicción que hizo el filtro en el estado anterior.

Un ejemplo de predicción en estado de oclusión para el dedo corazón, sería el siguiente:

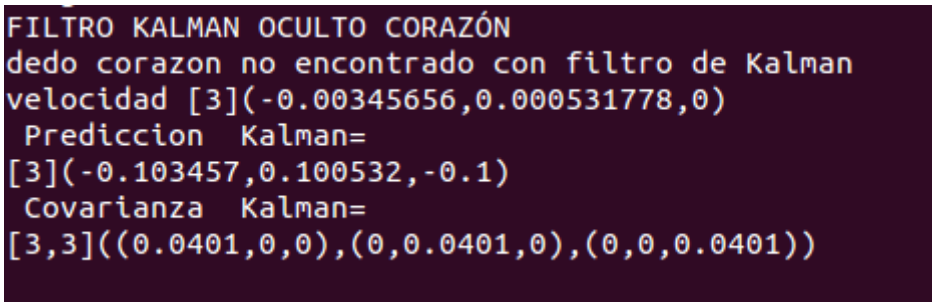
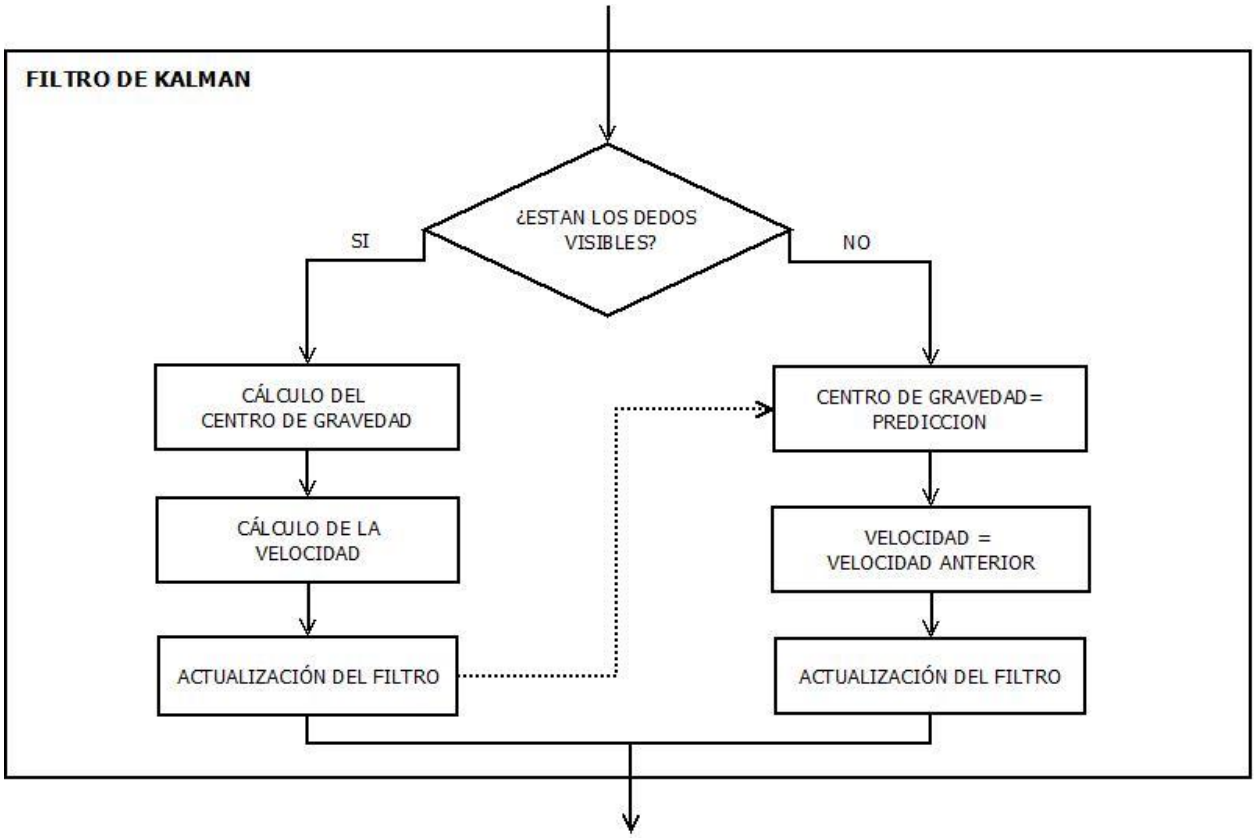


Figura 28: Predicción del filtro de Kalman en estado oculto.

La primera predicción siempre debe ser con los dedos visibles, porque para aplicar el modo en oclusión debo tener al menos una predicción anterior.

Si el filtro está actuando en modo de oclusión, y vuelve a localizar el dedo, pasa automáticamente al modo de Kalman visible, tomando de nuevo los valores reales del centro de gravedad.

El filtro de Kalman se puede esquematizar con el siguiente diagrama de estados:



F. FILTRO DE PARTICULAS

El filtro de partículas también lo proporciona la librería de filtrado bayesiano BFL.

El funcionamiento del filtro es muy parecido al del filtro de Kalman. Para que no sea repetitivo, no se va a explicar todo el filtro, simplemente se detallaran las diferencias que hay con respecto al filtro de Kalman.

En el filtro de partículas, como se explicó en el apartado IV.G “Filtro de Partículas”, se lanzan aleatoriamente unas muestras que mediante un número suficiente de iteraciones, se van aproximando a la medida real. Esa es la primera diferencia con el filtro de Kalman, hemos tenido que establecer el número de muestras, teniendo en cuenta que cuantas más muestras, menos error habrá en la predicción del filtro, pero un número muy alto de éstas ralentizan mucho el proceso de filtrado. El número de muestras fue determinado experimentalmente, estableciéndolo en 600 muestras.

1. MANO VISIBLE

Al igual que en el filtro de Kalman, cuando el filtro de partículas actúa en modo visible, al filtro se le administran la posición del centro de gravedad del dedo y la velocidad del estado actual, y el filtro de partículas estima la posición del centro de gravedad en el estado siguiente.

```
FILTRO PARTICULAS VISIBLE CORAZÓN
medida particulas [3](0.0389305,-0.051891,0.764)
velocidad particulas [3](-0.00156222,0.0020823,-0.0306581)
 Prediccion particulas =
[3](0.0102331,0.0883077,0.737224)
 Covarianza particulas =
[3,3]((0.00163529,0.00156419,0.00206265),(0.00156419,0.0106248,0.0128902),(0.002
06265,0.0128902,0.015785))
```

Figura 29: Filtro de partículas en modo visible.

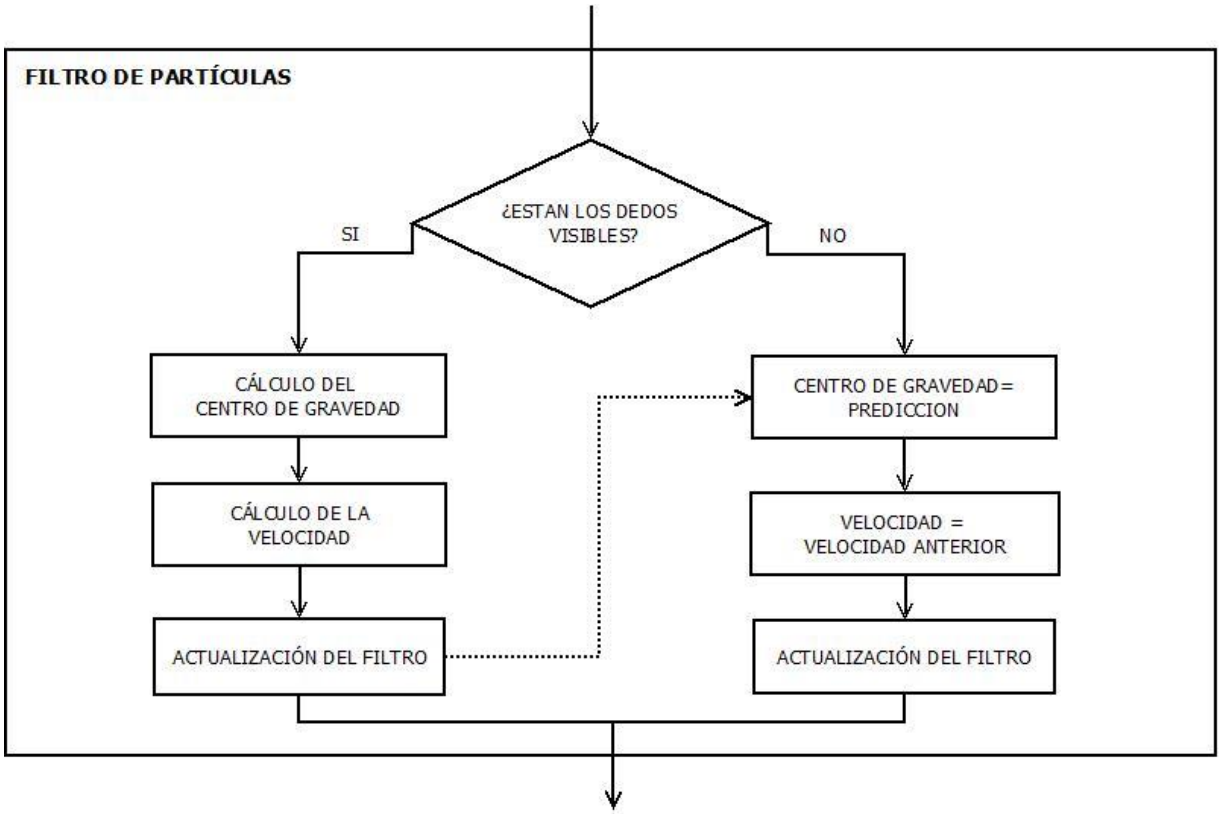
2. MANO EN ESTADO DE OCLUSIÓN

De nuevo el funcionamiento de este filtro es similar al del filtro de Kalman, por tanto, cuando no se conoce la posición del centro de gravedad actual, la predicción se realiza suponiendo la velocidad constante, y el centro de gravedad actual se considera que es el valor predicho por el filtro en el estado anterior.

```
FILTRO KALMAN OCULTO CORAZÓN
dedo corazon no encontrado con filtro de Kalman
velocidad [3](-0.00345656,0.000531778,0)
 Prediccion  Kalman=
[3](-0.103457,0.100532,-0.1)
 Covarianza  Kalman=
[3,3]((0.0401,0,0),(0,0.0401,0),(0,0,0.0401))
```

Figura 30: Filtro de partículas en modo de oclusión.

El diagrama de estados correspondiente al filtro de partículas seria:



G. VISUALIZACIÓN DEL RESULTADO DEL FILTRO

Una vez ya se ha aplicado el filtro de Kalman o el filtro de Partículas, el siguiente objetivo propuesto es abrir una interfaz para que el usuario aprecie de forma visual el resultado de la predicción del filtro. Esto lo hemos decidido hacer dibujando un cuadrado alrededor de la predicción de cada dedo.

Hemos considerado que este resultado será mejor percibido por el usuario si en lugar de mostrarlo en una nube de puntos tridimensional, se muestra en una imagen bidimensional. Esto es debido a que para el usuario se ve más nítida una imagen en 2 dimensiones que una nube de puntos tridimensional, como se ve en la siguiente imagen:

Figura 31: Nube 3D frente a imagen 2D.

Para mostrarlo como imagen 2D, es necesario realizar los siguientes pasos:

1. CONVERSIÓN DE LA NUBE DE PUNTOS A UNA IMAGEN 2D

Lo primero que hay que hacer es convertir la nube de puntos que nos proporciona la Kinect (imagen 3D) en una captura en dos dimensiones. Para ello utilizamos una de las funciones que nos proporciona la librería openCV.

2. CALCULO DEL PIXEL CORRESPONDIENTE A LA POSICION 3D

Tanto el Filtro de Kalman como el de Partículas, devuelven como resultado una predicción de la posición de cada dedo. Esa predicción tiene tres componentes, las correspondientes a las coordenadas de cada eje (XYZ). Pero como hemos decidido mostrar el resultado en una imagen en dos dimensiones, es necesario establecer una correspondencia entre el punto que nos devuelve el filtro, con su respectivo punto en una imagen bidimensional.

El problema surge en que esa correspondencia no es suficiente con eliminar la coordenada del eje Z (eje de profundidad), ya que el origen O de los ejes en el sistema tridimensional no está situado en el mismo punto que en la imagen bidimensional, como queda expuesto en la siguiente imagen:

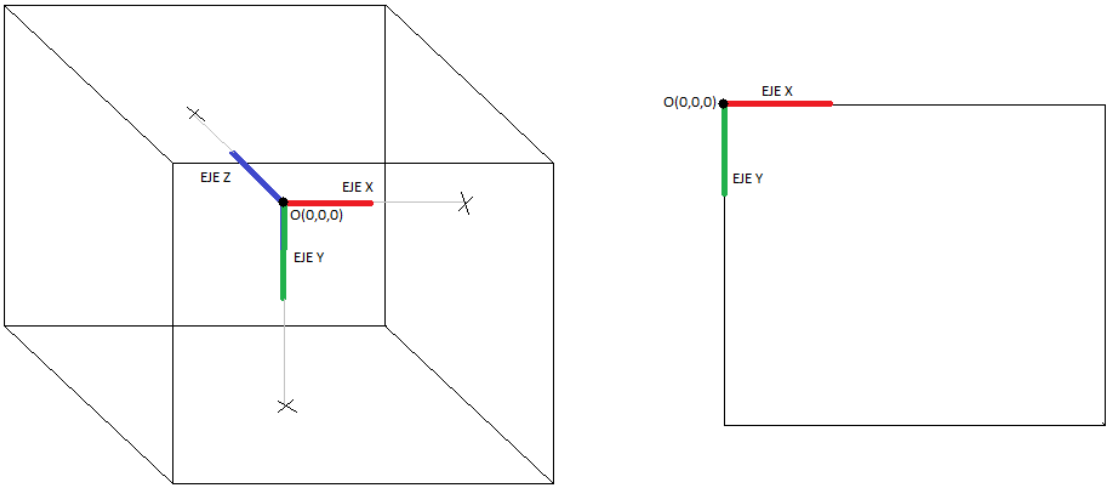


Figura 32: Comparación ejes en imagen 2D y nube 3D

Para establecer la correspondencia entre el punto tridimensional y el mismo punto en la imagen bidimensional hacemos uso del algoritmo Pin Hole, explicado en el apartado IV.G.

Por ejemplo, como se ve en la siguiente imagen, para un punto de la nube de puntos que está en la posición (0.1 , 0.02 , 1.5), le corresponde el pixel (331.359 , 236.826) de la imagen en 2D:

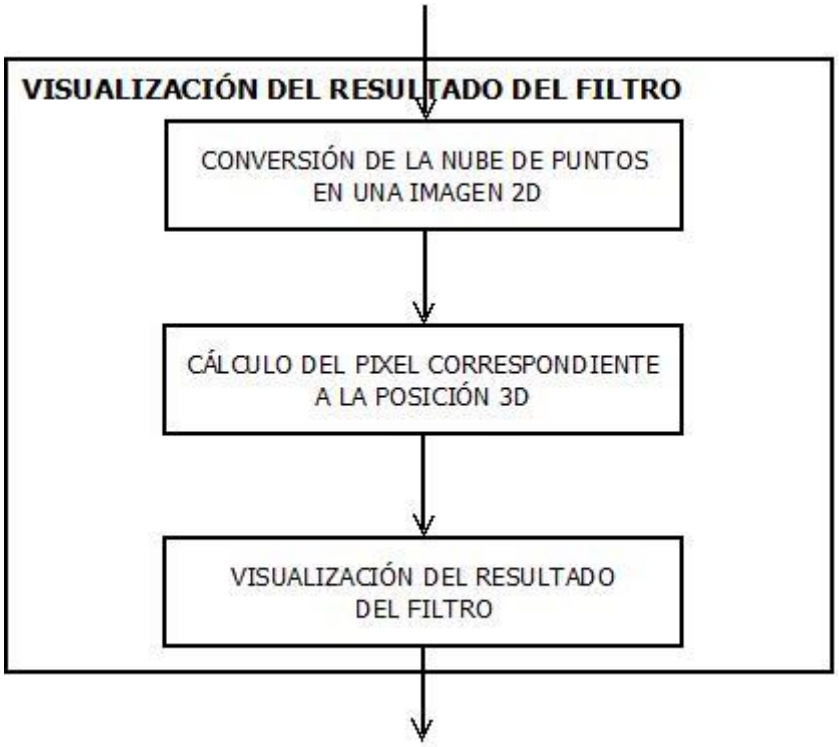
```
Prediccion x=0.01
Prediccion y=0.02
Prediccion z=1.5
Coordenadas2D=(331.359,236.826)
```

Figura 33: Resultado de Pinhole

3. VISUALIZACIÓN DEL RESULTADO DEL FILTRO

Cuando ya sabemos cuál es el punto en la imagen 2D que queremos resaltar, solo falta dibujar un pequeño rectángulo alrededor de dicho punto. Para eso utilizamos otra de las funciones de la librería OpenCV. Obteniendo como resultado los cuadrados que resaltan el punto que predice el filtro de Kalman o el Filtro de Partículas.

El diagrama de flujo correspondiente a esta última parte seria el que se muestra a continuación:



A continuación se van a mostrar una serie de imágenes donde se recoge el resultado final del trabajo.

Primero se muestran dos imágenes en las que están todos los dedos visibles. La diferencia entre ellas es que traté de mover solamente el dedo pulgar para demostrar que el filtro sigue a cada dedo de forma independiente:

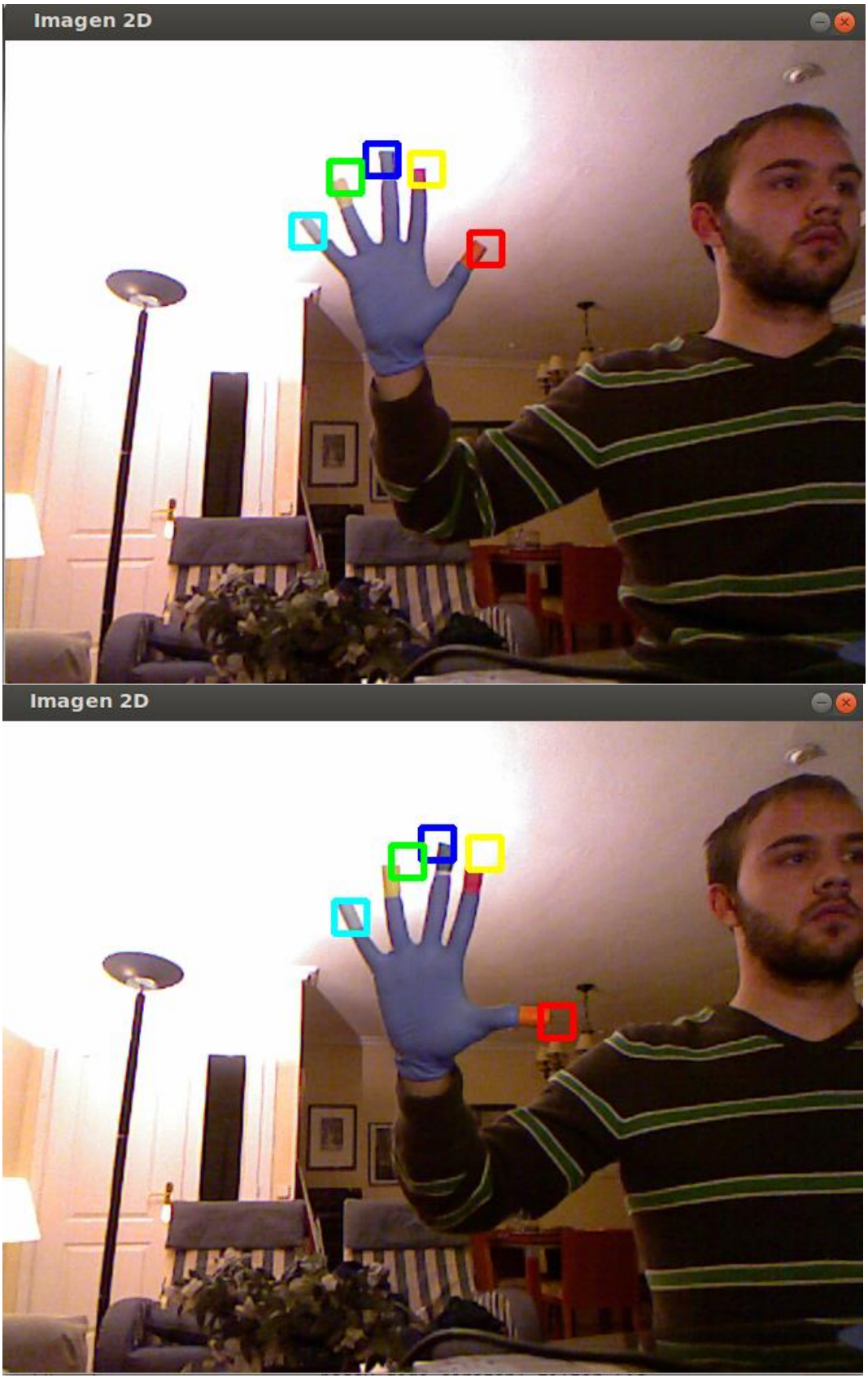


Figura 34: Resultado final de la predicción del filtro con todos los dedos visibles.

En la siguiente imagen se demuestra el funcionamiento del filtro en estado de oclusión, cuando todos los dedos están tapados:

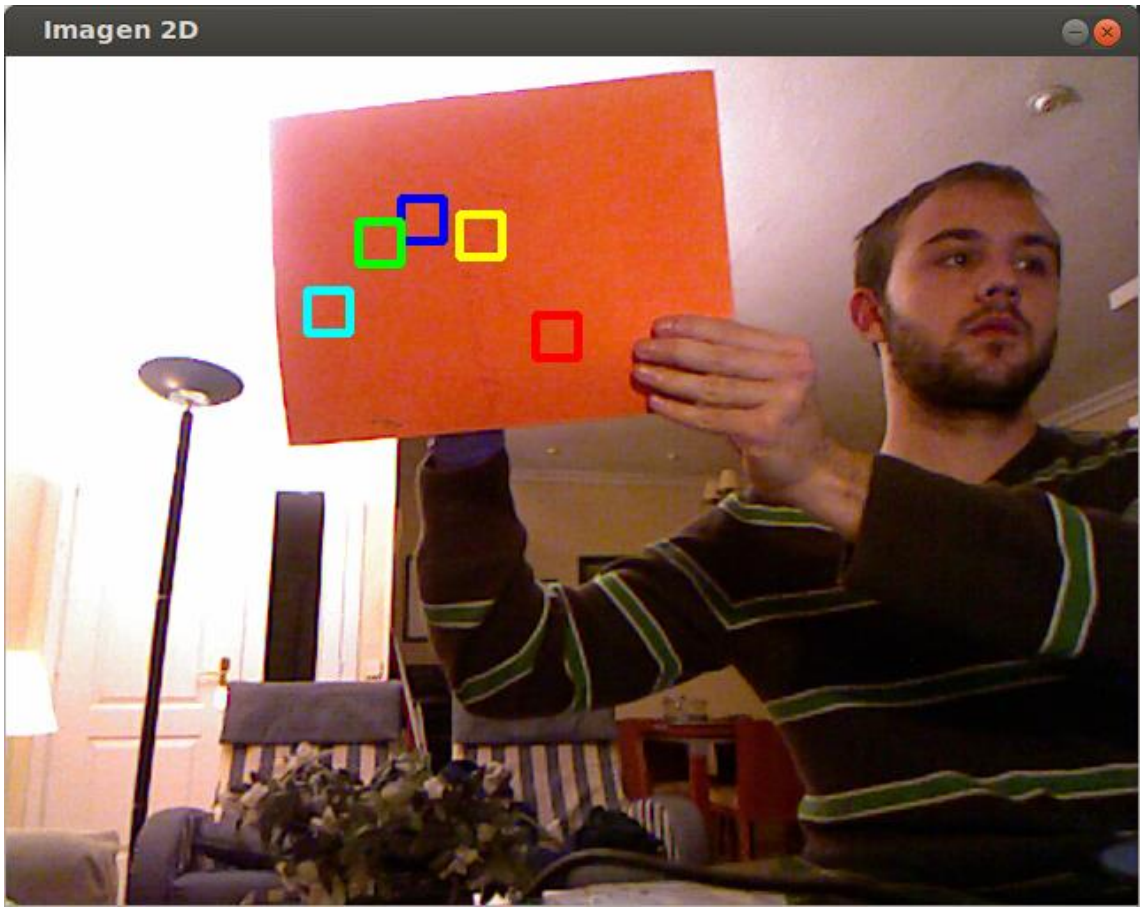


Figura 35: Resultado final de la predicción del filtro con todos los dedos ocultos.

Y por último, se muestra una imagen en la que se combinan ambos modos de funcionamiento del filtro, ya que hay dos dedos en modo visible, y los otros tres están ocultos:

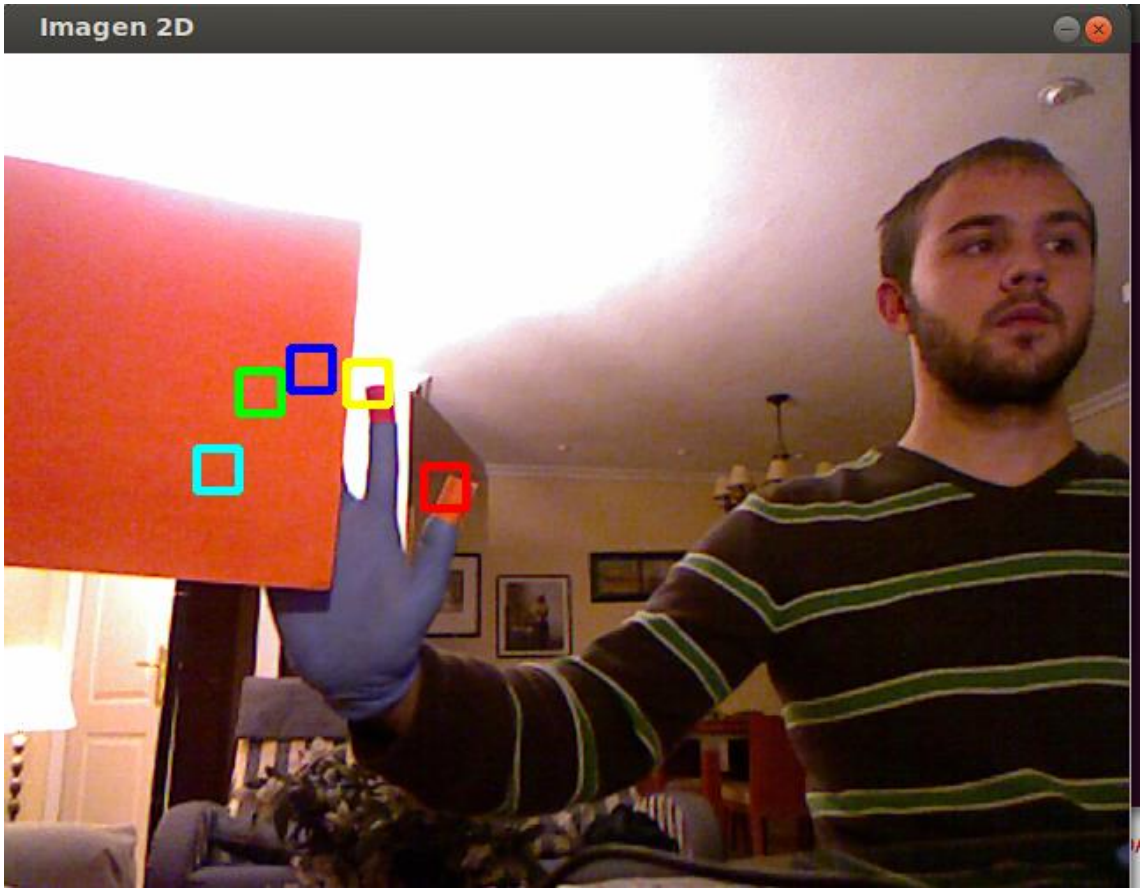


Figura 36: Resultado final de la predicción del filtro combinando ambos modos de predicción: visible y oculto.

VIII. RESULTADOS EXPERIMENTALES

En este capítulo, se pondrán a prueba muchas partes del código para demostrar su efectividad, así como justificar algunas de las decisiones tomadas en el proyecto.

A. FILTRADO DE PROFUNDIDAD

A lo largo del proyecto, se ha realizado varias veces un filtrado de profundidad, ya explicado en el apartado "IV.D". Este filtrado se realiza para eliminar gran cantidad de puntos de la nube de forma que se reduce la carga computacional.

A continuación se muestra una tabla donde se compara, para 8 capturas de nubes de puntos diferentes, el número de puntos eliminados:

	Puntos antes del filtrado	Puntos eliminados	Puntos de la nube filtrada	Porcentaje de puntos eliminados (%)
Nube 1	307200	250368	56832	81,50
Nube 2	307200	245999	61201	80,08
Nube 3	307200	269093	38107	87,60
Nube 4	307200	250286	56914	81,47
Nube 5	307200	258823	48377	84,25
Nube 6	307200	259238	47962	84,39
Nube 7	307200	257232	49968	83,73
Nube 8	307200	252490	54710	82,19
				83.15
	MEDIA DEL PORCENTAJE DE PUNTOSDE LAS OCHO NUBES			

Como queda demostrado, este filtro elimina casi un 85% de información de la nube innecesario para nuestro objetivo final, y conseguimos agilizar notablemente el tiempo de procesado.

B. EXTRACCIÓN DEL COLOR DE LOS DEDOS

1. MEDIA ARITMÉTICA RECORTADA

Hemos querido demostrar la necesidad de hacer la media aritmética recortada en lugar de una media aritmética simple.

Este proceso lo llevamos a cabo a la hora de calcular el color de los dedos, realizando una media de cada componente de color RGB de todos los puntos que componen cada dedo. Como ya quedó explicado en el apartado "IV.B" y "VI.C.2.a", se recorta un 10% de los valores de cada extremo.

En la siguiente tabla se compara, para 10 zonas de colores diferentes, el resultado que se obtiene aplicando una media simple, y el resultado que se obtiene al aplicar la media aritmética recortada al 10%. También se muestra la diferencia entre haber hecho la media de una forma u otra:

	MEDIA ARITMÉTICA SIMPLE			MEDIA ARITMÉTICA RECORTADA			DIFERENCIA		
	R	G	B	R	G	B	R	G	B
ZONA 1	121	54	70	125	53	64	4	1	6
ZONA 2	45	78	240	47	78	244	2	0	4
ZONA 3	98	201	44	101	205	35	3	4	9
ZONA 4	206	45	111	206	47	114	0	2	3
ZONA 5	147	68	96	145	66	101	2	2	5
ZONA 6	88	72	83	89	70	84	1	2	1
ZONA 7	158	69	108	170	68	93	12	1	15
ZONA 8	12	100	72	14	103	73	2	3	1
ZONA 9	97	74	187	94	77	189	3	3	2
ZONA 10	64	49	25	75	69	28	11	20	3

Hemos optado por realizarlo con la media aritmética recortada porque la presencia de posibles brillos en la imagen alteraría el resultado de las componentes de color. A pesar de que, en general, las diferencias en R, G y B no son notables, se han localizado algunas ocasiones en que la diferencia si es significativa.

2. LOCALIZACIÓN DEL PUNTO DE MAYOR ALTURA

En este apartado se quiere justificar la razón por la cual se decidió realizar la media de un conjunto de valores en lugar de tomar el color del dedo localizado. Como se explicó en el apartado “VI.C.1” y “VI.C.2.a”, el algoritmo primero detecta el punto de mayor altura de cada dedo, y a continuación se traza un cuadrado de 1cm x 1cm como se muestra en la siguiente imagen:

Figura 37: Imagen del puto localizado con el cuadrado correspondiente.

Se calcula la media de cada componente de color de todos los puntos del cuadrado en lugar de tomar directamente las coordenadas de color del punto más alto. En la siguiente tabla se muestra un estudio con 50 ensayos en los que se tomaron las coordenadas de color de ambas formas, y comprobando en cuántos de ellos el resultado era el buscado:

SOLAMENTE CON EL PUNTO DE Y MAXIMA	NUMERO DE ENSAYOS TOTALES	NUMERO DE ENSAYOS CON RESULTADO CORRECTO*	PORCENTAJE DE EFECTIVIDAD
DEDO PULGAR COLOR ROJO	50	26	**52**
DEDO INDICE COLOR AMARILLO	50	34	**68**
DEDO CORAZÓN COLOR AZUL	50	29	**58**
DEDO ANULAR COLOR VERDE	50	22	**44**
DEDO MEÑIQUE COLOR CIÁN	50	28	**56**

HACIENDO LA MEDIA DE TODOS LOS PUNTOS DEL CUADRADO	NUMERO DE ENSAYOS TOTALES	NUMERO DE ENSAYOS CON RESULTADO CORRECTO*	PORCENTAJE DE EFECTIVIDAD
DEDO PULGAR COLOR ROJO	50	42	**84**
DEDO INDICE COLOR AMARILLO	50	48	**96**
DEDO CORAZÓN COLOR AZUL	50	40	**80**
DEDO ANULAR COLOR VERDE	50	39	**78**
DEDO MEÑIQUE COLOR CIÁN	50	43	**86**

(*)NOTA: El resultado de las tres coordenadas de color se comprobaba en un editor de fotografía para ver el color formado. Se ha considerado como ensayo correcto aquellos colores resultantes que no diferían mucho del color del dedo.

Podemos observar que este caso ha sido completamente necesario, ya que si hubiéramos optado por mantener las coordenadas de color del punto de mayor altura, estaríamos cogiendo un color equivocado en aproximadamente el 50% de las ocasiones, y aplicando la media a los puntos del cuadrado, conseguimos mejorar la eficiencia, obteniendo el color correcto en un 85% aproximadamente de los casos.

C. FILTRO DE KALMAN Y FILTRO DE PARTÍCULAS

A la hora de aplicar tanto el filtro de Kalman como el de partículas, nosotros le proporcionamos al filtro la posición del dedo en el estado actual (centro de gravedad de todos los puntos del color del dedo), y el filtro predice la posición de ese punto en el estado siguiente. Con esto conseguimos tener en el estado actual una predicción del estado siguiente.

A continuación, para estudiar la eficacia de las predicciones de los filtros, se presentan unas tablas en las que comparamos la posición real en el estado siguiente, con la predicción que hace el filtro en el estado actual (en el estado actual predice el estado siguiente). Dichas tablas las hacemos para, por ejemplo, el dedo índice, ya que como el filtro es el mismo para los cinco dedos, la eficacia será la misma para los cinco. Además realizamos el estudio para un mismo movimiento del dedo, de tal forma que las posiciones reales en ambas tablas son las mismas, así conseguimos una comparación entre ambos filtros más precisa.

Valor Medido Real			Predicción de filtro de Kalman			Error absoluto		
Eje x (mm)	Eje y (mm)	Eje z (mm)	Eje x (mm)	Eje y (mm)	Eje z (mm)	Eje x (mm)	Eje y (mm)	Eje z (mm)
-215	4	847	-198	12	851	17	5	4
-182	11	889	-201	17	896	19	6	7
-157	14	934	-141	20	939	16	6	5
-119	15	957	-127	16	962	8	1	5
-69	32	1024	-75	22	1017	6	10	7
-10	66	1134	8	74	1114	3	8	2
-44	96	1045	-39	101	1067	5	5	22
-13	109	1162	-15	102	1174	2	7	1.2
47	117	1211	32	121	1227	15	4	16
88	120	1230	94	119	1233	6	1	3
117	132	1289	124	147	1282	7	15	7
126	142	1294	129	151	1299	3	9	5
155	164	1305	160	172	1314	5	8	9
187	180	1288	179	172	1301	8	8	13
209	203	1333	216	199	1329	7	4	4
ERROR PROMEDIO DE LA PREDICCIÓN						**8,47**	**6,47**	**7,79**

Valor Medido Real			Predicción de filtro de Partículas			Error absoluto		
Eje x (m)	Eje y (m)	Eje z (m)	Eje x (m)	Eje y (m)	Eje z (m)	Eje x (mm)	Eje y(mm)	Eje z (mm)
-215	4	847	-219	8	851	4	3	4
-182	11	889	-191	20	898	9	2	9
-157	14	934	-149	6	926	8	4	8
-119	15	957	-120	16	958	1	4	1
-69	32	1024	-69	32	1024	0	7	0
-10	66	1134	-3	59	1127	7	5	7
-44	96	1045	-50	90	1039	6	4	6
-13	109	1162	-4	100	1153	9	6	9
47	117	1211	35	129	1199	12	1	12
88	120	1230	79	111	1221	9	0	9
117	132	1289	108	141	1298	9	8	9
126	142	1294	132	136	1300	6	10	6
155	164	1305	153	162	1303	2	5	2
187	180	1288	190	195	1285	3	15	3
209	203	1333	205	208	1337	4	5	4
ERROR PROMEDIO DE LA PREDICCIÓN						**5,93**	**5,27**	**5,93**

NOTA: El error calculado es el error absoluto, es decir, el valor absoluto de la diferencia entre el valor real y el valor predicho. No se ha calculado el error relativo porque no tiene sentido ya que no hay fondo de escala con el que comparar el error absoluto.

Observamos que los errores que se cometen en ambos filtros no son muy significativos, alcanzando rara vez un error en la predicción mayor de 1cm.

Cabe mencionar que no se ha estudiado la eficacia de los filtros para los modos en oclusión ya que cuando el objeto no está visible, no tenemos un valor de la posición real, y por lo tanto no podemos estimar el error que está cometiendo el filtro.

De modo más intuitivo, se ha diseñado una gráfica en la que se sitúan una serie de puntos por los que pasa el dedo, con sus correspondientes puntos predichos, de tal forma que se aprecia mucho mejor el significado que puede tener el error. Dicha grafica solo se ha realizado en los ejes X e Y para que pueda ser representado en dos dimensiones, eliminando las componentes reales y predichas del eje Z.

Para el filtro de Kalman sería esta distribución:

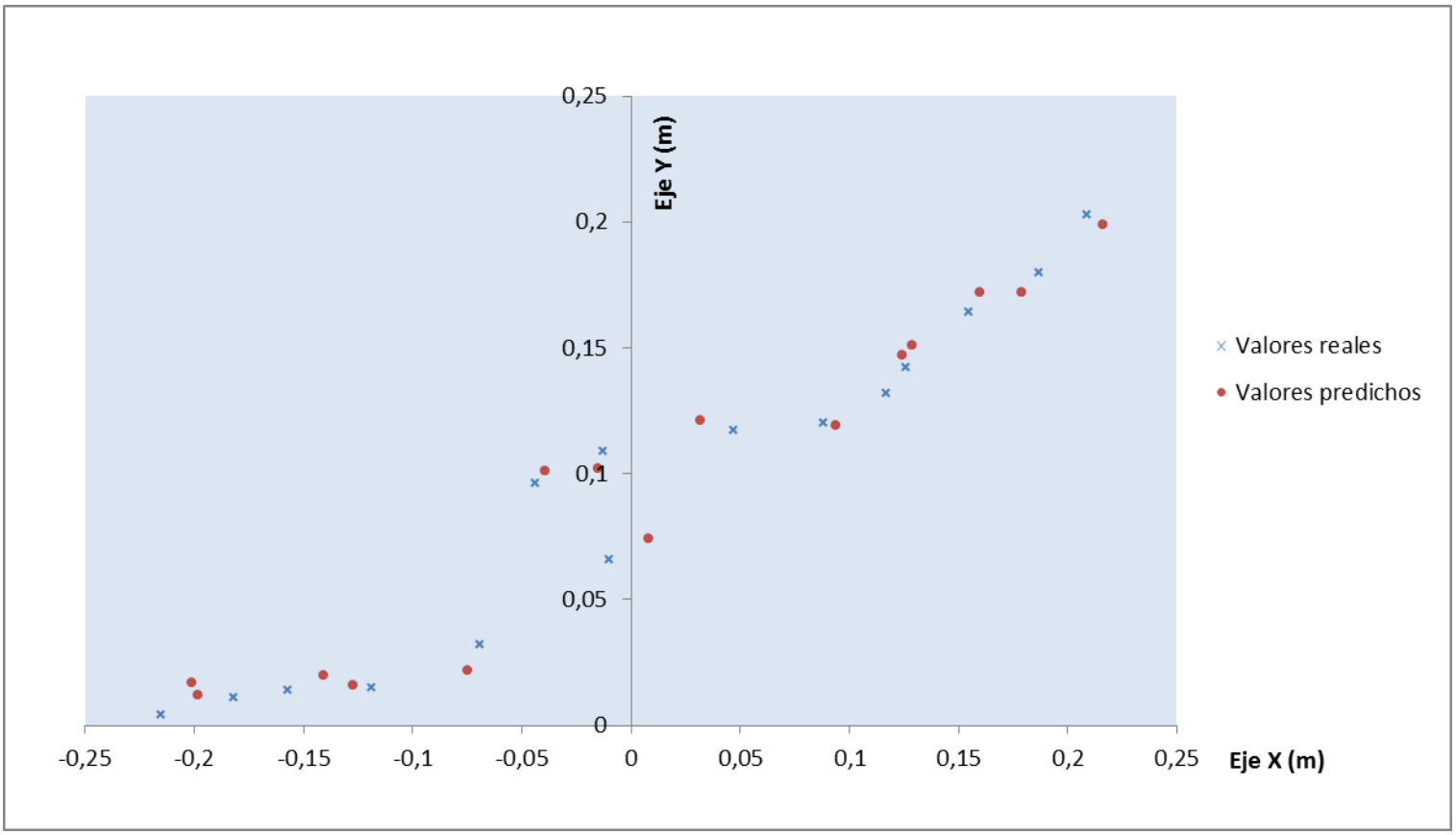


Figura 38: Representación gráfica de las predicciones del filtro de Kalman

Y para el filtro de partículas, esta otra:

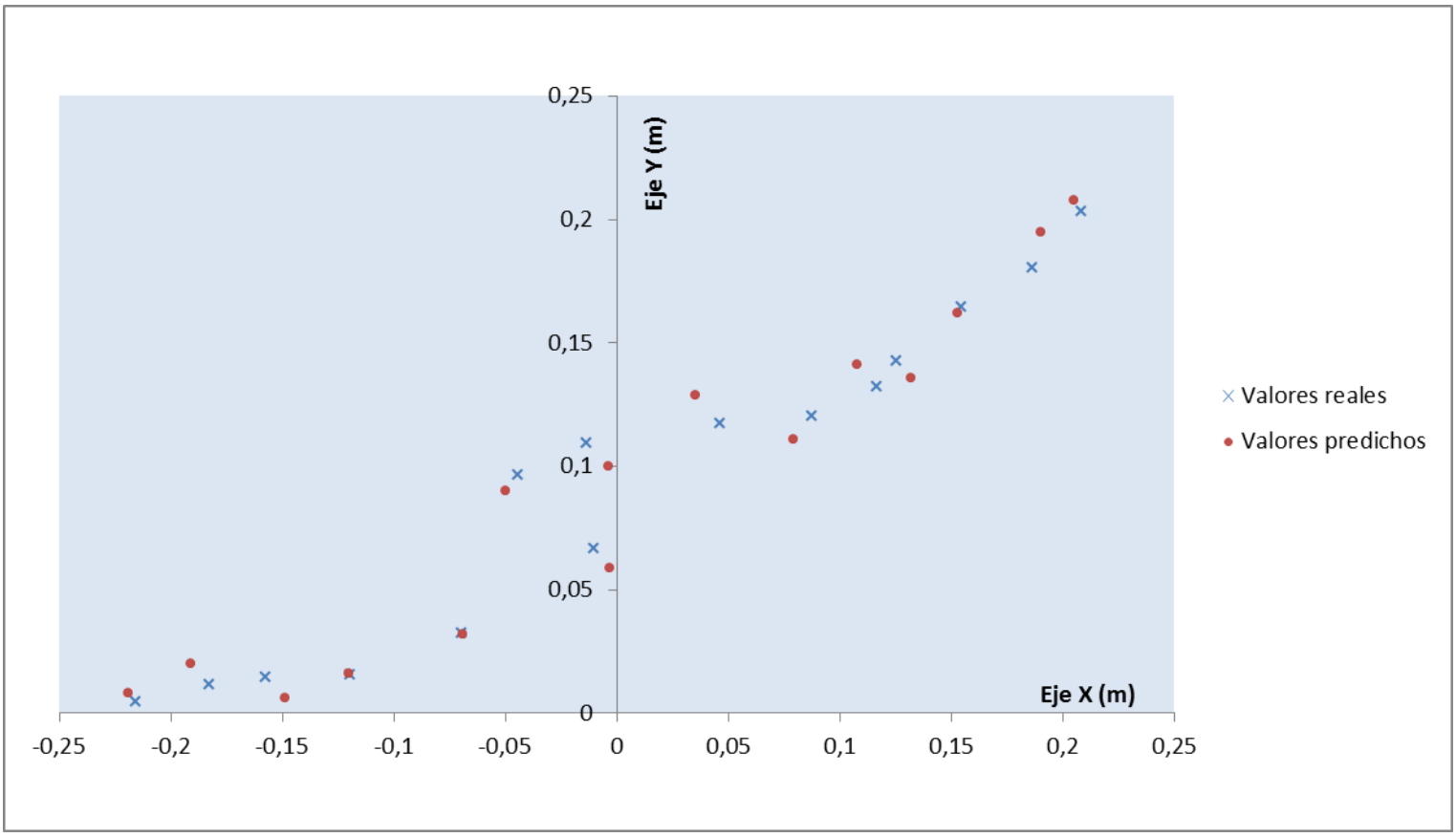


Figura 39: Representación gráfica de las predicciones del filtro de Partículas

D. TIEMPOS EN CADA PROCESO

El algoritmo que hemos diseñado, es en tiempo real, lo cual supone que la imagen tiene que estar actualizándose continuamente. Por ello, todo el programa es un bucle cerrado que se recorre constantemente. A continuación se exponen los tiempos aproximados que conlleva cada proceso que se lleva a cabo, lo cual nos da una idea aproximada del tiempo que tarda en recorrerse el bucle completo, y por tanto, cada cuanto tiempo se actualiza la imagen.

PROCESO		TIEMPO (ms)	
Captura de la imagen con Kinect		33	
Filtrado de profundidad		18	
Extracción del color de los dedos	Localización del punto más alto	31	
	Media aritmética recortada	10	
	Conversión de RGB a HSV	12	
Filtrado de seguimiento (de los 5 dedos)	Kalman	105	
	Partículas		162
Conversión imagen 3D a imagen 2D		41	
TIEMPO TOTAL		**250**	**307**

Se obtienen dos tiempos totales: uno es el correspondiente a haber aplicado el filtro de Kalman, y el otro al de partículas. Se aprecia que es ligeramente más lento el filtro de partículas.

> NOTA: No se ha tenido en cuenta el tiempo que tarda el usuario en meter los dedos dentro de los prismas, ya que eso solo tiene que hacerlo en el primer proceso, y el tiempo depende del usuario, no del programa.

IX. CONCLUSIONES, MEJORAS Y AVANCES FUTUROS

Una vez concluido el proyecto, podemos afirmar que se han aplicado un filtro de Kalman y un filtro de Partículas al seguimiento de objetos, en este caso, los dedos de una mano, en secuencias de imágenes tridimensionales.

Se han adquirido conocimientos nuevos en técnicas de tratamientos tanto de imágenes como, sobre todo, de nubes de puntos, las cuales han sido tratadas con PCL, una librería que sin duda me será muy útil para trabajos e investigaciones futuros.

Se ha estudiado el filtro de Kalman y filtro de Partículas que proporciona la biblioteca Orocos, para poder aplicarlo a nuestro programa y que tuviera la finalidad que deseábamos. Se han coleccionado datos necesarios para evaluar el funcionamiento de los filtros tanto en estado visible, como en estado de oclusión.

También se han adquirido unos pequeños conocimientos de la librería OpenCV para tratamiento de imágenes. Esto nos fue necesario para poder trabajar con una imagen bidimensional en lugar de una nube de puntos con el fin de recuadrar el resultado final de los filtros.

Además del trabajo visible, fue necesario un periodo de aprendizaje para la captura y procesamiento de nubes de puntos. En esta etapa se aprendieron las funciones básicas para el tratamiento de imágenes y nubes de puntos, teniendo que estudiar la documentación previa e implementar programas de prueba.

Los resultados que hemos obtenidos de los filtros son bastante aceptables, aunque todas las predicciones tienen un pequeño error asociado. Esos pocos milímetros que difieren de la medida real se deben a que ambos filtros lo que hacen es predecir la posición que ocupará el dedo en un estado futuro, asumiendo un movimiento rectilíneo no acelerado, pero al ser una predicción de un futuro cercano, no puede ser un resultado exacto.

Comparando ambos filtros, los dos realizan perfectamente la función para la que fueron diseñados, si bien cabe mencionar que el filtro de partículas puede obtener resultados más exactos ya que depende del número de muestras que queramos lanzar. Para un número muy alto de muestras, la predicción será mucho más aproximada a la medida real que la que pudiera hacer el filtro de Kalman, aunque este filtro también es más lento que el de Kalman, por tanto, dependiendo de la aplicación para la que se quiera utilizar, será mejor uno u otro.

Los resultados que se han obtenido pueden ser útiles para el alumnado, para la enseñanza de los filtros de Kalman y de Partículas, ya que se podrán explicar de forma intuitiva este tipo de técnicas de seguimiento, que por su carga teórica suelen ser difíciles de comprender.

Como posibles futuras líneas de trabajo, se podría trabajar en la implementación de algún otro filtro de agrupamiento como el K-medias para el cálculo de estimadores del filtro. Se podría trabajar también en tratar de reducir los errores que cometen los filtros creados, que suelen ser de unos pocos milímetros. Podría ser interesante también mejorar el algoritmo de tal forma que para la localización y seguimiento de los dedos no fuera necesaria la utilización de un guante de colores.

X. CAPÍTULOS ADICIONALES

A. PRESUPUESTO

La consecución final de este proyecto tiene dos costes asociados: los costes de los materiales necesarios, y los costes atribuidos a la contratación de un ingeniero para la realización del código. Se exponen más detalladamente a continuación:

COSTE DE LOS MATERIALES		
CONCEPTO	UNIDADES	IMPORTE (€)
Cámara Kinect de Microsoft	1	150
Ordenador	1	500
TOTAL		**650**

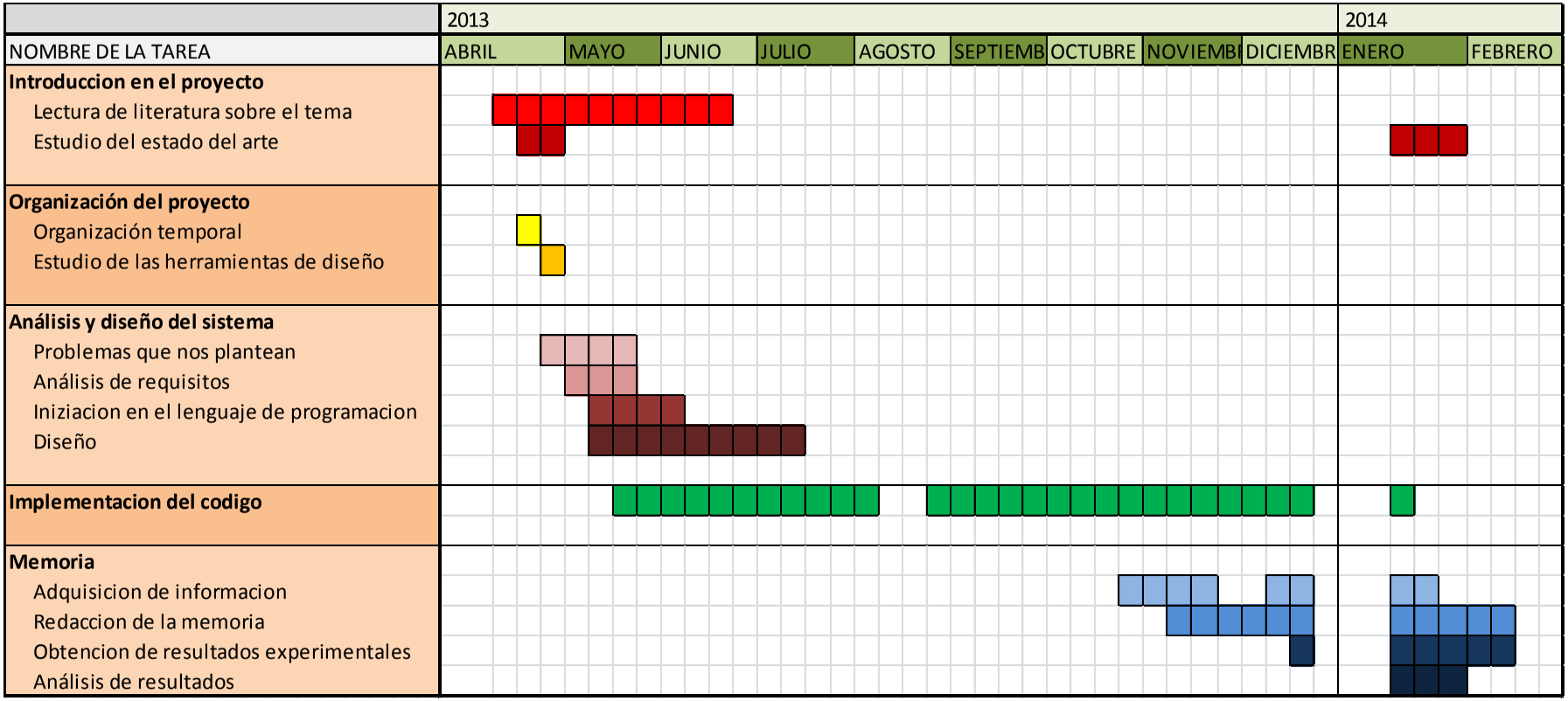
Para calcular el gasto asociado a la ingeniería, hay que estimar las horas de trabajo. No todas las horas están remuneradas con un mismo importe. Una estimación de los gastos que supondría la contratación de un ingeniero se detalla en la siguiente tabla:

COSTE INGENIERÍA			
CONCEPTO	Nº DE HORAS	PRECIO/HORA (€)	IMPORTE (€)
Documentación e investigación	100	12.5	1250
Diseño e implementación	120	15	1800
Pruebas experimentales	10	10	100
TOTAL	230		**3150**

Una vez detallados los costes asociados al proyecto, se puede calcular el coste final del proyecto, que sería según se detalla a continuación:

PRESUPUESTO GENERAL	
TIPO DE COSTE	IMPORTE(€)
Materiales	650
Ingeniería	3150
Subtotal	3800
I.V.A (21%)	798
TOTAL	**4598**

B. PLANIFICACIÓN



XI. BIBLIOGRAFÍA

[1] David A. Forsyth and Jean Ponce (2003). Computer Vision, A Modern Approach. Prentice Hall.

[2] J. J. Gibson, The Ecological Approach to Visual Perception. Boston: Houghton Mifflin, 1979.

[3] http://en.wikipedia.org/wiki/Range_imaging

Última visita: 24 de noviembre de 2013

[4] D. Marr, Vision. San Francisco: Freeman, 1982.

[5] K. Lai, L. Bo, X. Ren, and D. Fox. A large-scale hierarchical multi-view rgbd object dataset. In Robotics and Automation (ICRA), 2011 IEEE International Conference on pages 1817–1824. IEEE, 2011.

[6] J. Tang, S. Miller, A. Singh, and P. Abbeel. A textured object recognition pipeline for color and depth image data.

[7] D.I.B. Hagebeuker. A 3d time of flight camera for object detection. 2007.

[8] Bernd Jähne (1997). Practical Handbook on Image Processing for Scientific Applications. CRC Press.

[9] Huang D, Swanson EA, Lin CP, Schuman JS, Stinson WG, Chang W, Hee MR, Flotte T, Gregory K, Puliafito CA, et al. Optical coherence tomography. Science. 1991 Nov 22.

[10] Hui Zheng, Jie Yuan, and Renshu Gu. A novel method for 3d reconstruction on uncalibrated images. In Proceedings of the Third International Conference on Internet Multimedia Computing and Service, ICIMCS '11, pages 138–141, New York, NY, USA, 2011. ACM.

[11] Y.Y. Huang and M.Y. Chen. 3d object model recovery from 2d images utilizing corner detection. In System Science and Engineering (ICSSE), 2011 International Conference on, pages 76–81. IEEE, 2011.

[12] N. J. Gordon, D. J. Samond, A. F. M. Smith, "Novel Approach to Nonlinear/NonGaussian Bayesian State Estimation", IEE Proc.-F, vol. 140, no. 2, 1993.

[13] J. Zhu, G. Humphreys, D. Koller, S. Steuart, and R.Wang. Fast omnidirectional 3d scene acquisition with an array of stereo cameras. In 3-D Digital Imaging and Modeling, 2007. 3DIM'07. Sixth International Conference on, pages 217–224. IEEE, 2007.

[14] A. Moro, E. Mumolo, and M. Nolich. Building virtual worlds by 3d object mapping. In Proceedings of the 2010 ACM workshop on Surreal media and virtual cloning, pages 31–36. ACM, 2010.

[15] http://www.biografiasyvidas.com/biografia/m/michelson.htm

Última visita: 28 de noviembre de 2013

[16] Javier González Jiménez. “Visión por Computador”. Ed. Paraninfo. 2000.

[17] D. Zhang and G. Lu (2001). Segmentation of moving objects in image sequence: A review. Circuits, Systems and Signal Process. vol. 20, no. 2, pp. 143-183, 2001.

[18] S. Ahn, M. Choi, J. Choi, andW.K. Chung. Data association using visual object recognition for ekf-slam in home environment. In Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on, pages 2588–2594. IEEE, 2006.

[19] L. Cruz, D. Lucio, and L. Velho. Kinect and rgbd images: Challenges and applications. SIBGRAPI Tutorial, 2012.

[20] D.I.B. Hagebeuker. A 3d time of flight camera for object detection. 2007.

[21] C. Pheatt, J. Ballester, and D. Wilhelmi. Low-cost three-dimensional scanning
using range imaging. Journal of Computing Sciences in Colleges, 20(4):13–19, 2005.

[22] http://www.astrofisicayfisica.com/2012/06/que-es-el-espectro-electromagnetico.html
Última visita: 28 de noviembre de 2013

[23] http://definicion.de/color/
Última visita: 5 de diciembre de 2013

[24] http://www.uv.es/webgid/Descriptiva/25_otras_medias.html
Última visita: 12 de diciembre de 2013

[25] http://dis.um.es/~ginesgm/files/doc/pav/tema5.pdf
Última visita: 5 de diciembre de 2013

[26]http://www.monografias.com/trabajos-pdf4/centro-gravedad-centroide/centro-gravedad-centroide.pdf
Última visita: 13 de diciembre de 2013

[27] J.F Vélez, A.B Moreno, Á. Sánchez y J.L Esteban, “Visión por Computador”,
Universidad Rey Juan Carlos, 2003.

[28] G. Kitagawa, “Non-Gaussian state space modeling of nonstationary time series”, Journal of the American Statistical Association, Vol.82, No.400, 1032-63, 1987.

[29] M. West, “Modeling wih mixtures. Bayesian Statistics”, 4, 503-524, 1992.

[30] M. Isard, A. Blake, “Contour tracking by Stochastic Propagation of Conditional density”, in Proc. 4th European Conf. Computer Vision, 1996, pp. 343-356.

[31] N. J. Gordon, D. J. Samond, A. F. M. Smith, “Novel Approach to Nonlinear/NonGaussian Bayesian State Estimation”, IEE Proc.-F, vol. 140, no. 2, 1993.

[32] G.Kitagawa, “Monte Carlo Filter and Smoother for Non-Gaussian Nonlinear State Space Modes”, J. Comput. Graph. Statistics, vol. 5, pp. 1-25, 1996.

[33] J. Carpenter, P. Clifford, P.Fernhead, “An improved particle filter for non-linear problems”. Technical report, Dept. of Statistics, University of Oxford, 1997.

[34] M. K. Pitt, N. Shephard, “Filtering Via Simulation: Auxiliary Particle Filters”, Journal of the American Statistical Association, vol. 94, no. 446, pp. 590-599, 1999.