Universidad Nacional Autónoma de México

Facultad de Ingeniería

Sistema de detección y rastreo de personas en tiempo real para el robot Golem-II+

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

Ingeniero en Computación

PRESENTA:

Romero Cordero Rogelio Adrián

DIRECTOR DE TESIS:
Dr. Ivan Vladimir Meza Ruiz

2015

Agradecimientos

A mis padres, Petra Cordero Hernández y Rogelio Romero Sánchez quienes me concedieron el hermoso placer de vivir, por todos los esfuerzos que tuvieron que concebir para sacar a nuestra familia adelante a pesar de cualquier obstáculo que se encontraban, ellos son la luz que guía cada paso de mi vida.

A mis hermanos, Ana Lilia, Luis Alberto y Carlos por pelear, por jugar, por entender, por creer en mi, pero sobre todo, por siempre estar conmigo y ayudarme cuando lo necesitaba.

A Salma, quien me ha brindado su apoyo incondicional, su cariño, confianza y comprensión, además del gran amor que siempre me ha entregado. Por los hermosos momentos que hemos compartido, siempre tendrás una gran importancia en mi pensamiento, en mi vida y en mi corazón.

A mi director de tesis, Ivan Vladimir que me brindó su apoyo, paciencia y consejos no sólo para la realización de este trabajo, sino para otros aspectos de mi desarrollo académico.

Quiero expresar mi más sincero agradecimiento a Gibran, quien lamentablemente no pudo aparecer como coodirector de este trabajo, por su gran aporte y participación activa en la realización de esta tesis, además de su buena disposición y paciencia para explicarme cuando no entendía, no cabe duda que su participación ha enriquecido el trabajo realizado.

A la Facultad de Ingeniería y a la UNAM por brindarme la oportunidad de obtener el aprendizaje que ha ayudado en gran parte a desarrollar mi forma de pensar y actuar. Por brindarme los medios por los cuales he formado mi carrera profesional.

Agradezco a todas las personas que hicieron posible el desarrollo y consumación de este proyecto, ha sido un largo camino y no pude haberlo logrado sin su apoyo.

Al proyecto Golem-III: Un robot guía para el museo Universum (ICyTDF/209/2012), el cual me brindó un apoyo económico para la realización de este trabajo.

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

Un robot de servicio puede ser semi o completamente autónomo y puede realizar servicios útiles para el bienestar de los humanos, excluyendo operaciones de manufactura tal y como lo menciona la IFR (International Federation of Robotics) [1]. En otro sentido también podría entenderse como un robot que asiste a los seres humanos. Con base en esta definición, un robot de servicio podría llegar a hacer un sin fin de tareas, es por esto que la IFR ha hecho una clasificación de los diferentes tipos de robots de servicio en dos grandes ramas que son:

- Robots domésticos
    - *Tareas domésticas*: mayordomo, compañía, asistente, limpieza, etc.
    - *Entretenimiento*: juguetes, paseos, educación y entrenamiento para niños, etc.
    - *Ayuda a discapacitados*: rehabilitación personal, transporte, etc.
    - *Transporte personal*: sillas de ruedas robotizadas.
    - *Seguridad y vigilancia en el hogar*: cámaras, puertas, ventanas automáticas, etc.

- Robots de servicio profesionales
    - *Robots de campo*: agricultores, ordeñadores, silvicultores, sistemas de minería, robots espaciales, etc.
    - *Limpieza profesional*: de pisos, de ventanas y paredes incluyendo los que escalan edificios, de tuberías y tanques, automóviles, aviones, etc.
    - *Inspección y mantenimiento*: para industrias, alcantarillas, plantas, etc.
    - *Construcción y demolición*: demolición nuclear, desmantelamiento, dar soporte y mantenimiento en construcciones, etc.
    - *Sistemas de logística*: entrega de correo, vehículos automatizados, etc.
    - *Robótica médica*: sistemas de diagnóstico, rehabilitación, etc.
    - *Defensa, rescate y aplicaciones de seguridad*: bombero, antibombas, aeroplano no transportado, etc.

De esta forma se llegan a una serie de cuestiones acerca de cómo el robot puede entender a un ser humano y de qué forma lo hace, ya sea de forma hablada, escrita o con lenguaje natural.

Uno de los principales problemas a resolver para que un robot de servicio tenga una mejor interacción con los humanos de manera autónoma, es que el robot pueda percibir a las personas que se encuentran a su alrededor. Esta percepción puede ser por medio de visión computacional, que se hace a través de imágenes, con audio a través de la voz, de su temperatura con una cámara térmica, entre otros.

En este trabajo se usarán las imágenes proporcionadas por una cámara fotográfica, y mediante una técnica de extracción de características llamadas Histogramas de Gradientes Orientados (HOG *Histograms of Oriented Gradients* en inglés) se realizará la detección de personas en el entorno. Después de esto, se llevará a cabo un rastreo de las personas detectadas.

1.1. Definición del problema

Detectar a una persona con un robot de servicio no es una tarea sencilla debido a que existe una gran variedad de problemas que deberán considerarse al momento de desarrollar el sistema.

La mayoría de los dispositivos con los que se hace la detección de personas y para los que se ha hecho una mayor investigación, se asume que el dispositivo no está en movimiento, sin embargo existen varios esfuerzos para portarlo a robots móviles [2]. Esto quiere decir que el nivel de dificultad aumenta al implementarse en un robot móvil como lo es un robot de servicio, puesto que el dispositivo que obtiene las imágenes podría capturar imágenes con ruido o que no tengan bien definidos los objetos de su alrededor. Debido al movimiento, las siluetas de los objetos podrían deformarse, además el robot puede perder la localización de la persona en la imagen más rápidamente.

Existen diversos dispositivos que se han utilizado para detectar personas, cada uno de estos dispositivos tienen sus propias ventajas en cuanto a la percepción del entorno. Algunos ejemplos de estos dispositivos son: los sensores

láser que solo detectan la distancia que hay entre el dispositivo y los objetos que están frente a él, o las cámaras térmicas que utilizan la diferencia de la temperatura de objetos comunes y la temperatura de los humanos [2].

La solución que se usó en este trabajo se basó en la cámara fotográfica, que captura imágenes del entorno y se procesan con una serie de algoritmos para detectar la parte superior de una persona desde los hombros hasta la cabeza. Existen una serie de desafíos que se tendrán que considerar, por ejemplo, los cambios en la apariencia de una persona (como su postura), la orientación (si está de frente, de perfil o dando la espalda al robot), la distancia (ya que si está muy lejos no podrá ser detectada una persona), los obstáculos temporales y la iluminación del entorno (la persona puede parecer diferente dependiendo de la cantidad de luz y el modo en el que incide en ella).

Actualmente los sistemas de detección de personas son en su mayoría para aplicaciones de vigilancia y se usan para saber cuántas personas entran o salen de un lugar, tratar de saber quiénes son y si ya han estado antes, entre otras.

Los sistemas que ya han sido aplicados a robots de servicio, se enfrentan al problema de la movilidad del robot. Sus ambientes ahora son cambiantes, de tal manera que puede tener características muy diferentes a las que se tenían en el momento de detectar con éxito a una persona; bajo estos cambios puede ser que pierda su localización.

Ningún sistema puede predecir estos cambios ni deja de sufrir las consecuencias de ellos, por lo tanto, se trata de hacer un sistema robusto, que tenga

la mínima sensibilidad a cambios de iluminación, de forma (debido al movimiento) y de orientación tanto del robot como de las personas detectadas, etc.

El cuadro 1.1 proporciona un pequeño resumen de lo anteriormente descrito, así como una recapitulación de lo visto.

Resumen	
Robots de Servicio	Proporcionan servicios para el bienestar de los humanos
Necesario para la interacción	▪ Percepción ▪ Localización
Desafíos en visión (cámara)	▪ Postura ▪ Orientación ▪ Distancia ▪ Iluminación ▪ Movimiento
Ambiente	Dinámico
Sistema robusto	Que se mantengan las detecciones aún cuando existan cambios en el entorno.

Cuadro 1.1. *Cuadro resumen.*

1.2. Objetivo

El objetivo general de este trabajo es:

- Desarrollar un sistema de detección y rastreo de personas eficiente y

en tiempo real para el robot Golem-II+ que detectará la parte superior del cuerpo humano desde los hombros hasta la cabeza y realizará un rastreo de la persona en la escena.

Los objetivos específicos son:

- Recolectar dos conjuntos de imágenes uno que servirá para el entrenamiento del detector y otro que será para la evaluación del mismo.
- Entrenar diferentes tipos de detectores y evaluar el desempeño de cada uno de ellos en dos diferentes conjuntos de imágenes.
- Elegir un detector definitivo para realizar el rastreo, con base a los resultados de la evaluación.
- Desarrollar el módulo de rastreo que utilizará el filtro de Kalman y el algoritmo Húngaro.
- Evaluar el sistema completo en videos tomados directamente de la cámara del robot Golem-II+.

1.3. Metodología

Se implementará un módulo de detección de personas a partir de los hombros hasta la cabeza, ya que estas partes del cuerpo tienen una mayor probabilidad de que al ser detectadas sea un cuerpo humano, debido a que presentan características que son diferentes a otros objetos de la vida cotidiana, además de que tienden a tener una apariencia más estable y por

lo tanto menos variaciones en su forma. De la misma manera se desarrollará otro módulo de detección de personas con la diferencia en que esté solo detectará la cabeza, con el fin de hacer una comparación entre estos dos detectores y determinar cual se usará.

Para llevar a cabo la detección de personas, se realizará el entrenamiento de un detector utilizando una técnica de aprendizaje supervisada llamada *Máquina de Vectores de Soporte* o *SVM (Support Vector Machine)* [3], que a partir de muestras de entrada detecta patrones y realiza tareas de clasificación y regresión. Para este caso la clasificación será de dos clases, es decir, una clase será: *personas*, y la otra clase será: *no personas*.

Para lograr esto se recolectará un conjunto de imágenes (corpus) que contendrán cuadros en donde se señalizará el lugar donde se encuentran los hombros y la cabeza de cada una de las personas que se encuentren en la imagen, a este conjunto de imágenes se le denominará "positivas". También se seleccionará otro gran conjunto de imágenes a las que se les llamará "negativas" las cuales no tendrán ningún cuerpo humano.

Una vez finalizado el entrenamiento del detector, este intentará localizar a las personas en la imagen, cuando se detecte el lugar en donde se encuentra una persona, esta información se usará para iniciar el rastreo.

El rastreo tomará las localizaciones que sean entregadas por el detector y a partir de estas se hará una predicción de la localización de la persona para la siguiente imagen con un método que se llama: filtro de Kalman. Como parte del rastreo primero se tiene que hacer una asociación de los cuadros encontrados por el detector, pues puede arrojar más de un cuadro en la imagen y esto quiere decir que pueden existir más de una persona en la escena

o que puede haber detecciones que no señalen a una persona. Para hacer esta asociación se usará un algoritmo que resuelve problemas de asignación llamado algoritmo Húngaro. Con este método sabremos qué cuadro pertenece a la persona que había sido detectada con anterioridad.

Cuando se conoce la relación entre las detecciones presentes y las detecciones pasadas, se procede a estimar la siguiente localización de la persona en la escena con las predicciones del filtro de Kalman.

1.4. Organización de la tesis

La tesis está organizada de la siguiente forma:

Capítulo 1.- Se menciona la definición del problema, el objetivo de la tesis, la metodología a usar y la forma en que se organiza este trabajo.

Capítulo 2.- Se mencionarán algunos de los dispositivos y algunas de las técnicas que se utilizan con ellos para la detección y el rastreo de personas, sobre todo los que pueden y suelen ser utilizados en robots de servicio.

Capítulo 3.- Esta orientado para explicar más a detalle la técnica de extracción de características que se usó en este trabajo y la forma en que se lleva a cabo todo el proceso de detección en el sistema propuesto.

Capítulo 4.- Se hace un análisis tanto del método que se utilizó para el rastreo como de las técnicas que usaron para llevar a cabo esta fase.

Capítulo 5.- Tiene como objetivo explicar la forma en la que se plantearon las evaluaciones de cada fase, así como los resultados que entregaron.

Capítulo 6.- Se dan a conocer las conclusiones, las dificultades encontradas y el trabajo futuro.

Capítulo 2

Antecedentes

En este capítulo se dan a conocer algunos de los dispositivos comunes para la detección de personas, así como una breve descripción de algunos trabajos que se han desarrollado usando uno o más de estos dispositivos.

Los robots de servicio podrían llegar a realizar tareas elaboradas, que para nosotros los humanos son repetitivas y relativamente fáciles de hacer, por ejemplo, tender una cama, limpiar un cuarto, llevar o cargar algunos utensilios; incluso el robot podría manipular ciertos instrumentos para hacer tareas aun más complejas como lavar trastes, escribir, cocinar, etc.

Para llevar a cabo este tipo de tareas es necesario que un robot perciba lo que hay en su entorno, desde las personas que pueda atender, hasta los obstáculos que pueda encontrar. La información que obtiene de la percepción servirá para realizar tareas como: no chocar, hacer de forma correcta los movimientos que necesita, evadir obstáculos móviles y estáticos, localizar personas, entre otras.

Esta percepción se logra obteniendo información del entorno, que puede ser en forma de imágenes, señales de sonido, profundidad de los objetos, la

temperatura de cada objeto, etc. Una vez obtenida ésta, se puede usar para detectar a las personas que se encuentran en la imagen e incluso guardar los rostros para después recordar a estas personas. La información también se puede procesar para obtener otro tipo de datos, los cuales el robot interpretará y usará para llevar a cabo las tareas que le han sido asignadas.

Para tener esta percepción es necesario contar con algún dispositivo que capture el entorno y se lo facilite al robot para que esté pueda procesarlo. Existen diversos dispositivos que permiten este tipo de percepción, a continuación nos enfocamos en los más comunes del entorno del robot de servicio.

Los sistemas que se especializan en detección de personas se pueden dividir en tres principales tipos, de acuerdo al dispositivo que usan para resolver el problema, a continuación se listan algunos de los dispositivos más utilizados:

- Cámaras: Basadas en luz.
  - Cámaras fotográficas: Capturan una sola imagen a la vez.
  - Cámaras estereoscópicas: Capturan dos imágenes a la vez, intentan emular la visión humana.
  - Cámaras térmicas: Crean una imagen a partir de la temperatura del entorno.
- Otros sensores: Basados en distancia.
  - Láser: Determina la distancia a un objeto.
  - Sensor infrarrojo: Es un sensor de movimiento y obtiene profundidades del entorno.

- Híbridos: Son sistemas que utilizan dos o más dispositivos para detectar personas.

A continuación se analizan estos dispositivos con más detalle.

2.1. Cámaras

Una cámara es un dispositivo utilizado para capturar imágenes, esto lo hace tomando la información de la luz visible que incide en los objetos del entorno. Esta información se convierte en una imagen, que es la que envía la cámara al robot o al dispositivo al que esté conectada.

La detección de personas en imágenes tiene como objetivo localizar a una o varias personas que se encuentren en la escena, regresando una región en donde se ha determinado la presencia de la o las personas.

Ésta es una tarea muy difícil, ya que el cuerpo humano es altamente deformable, además de que puede adoptar diferentes posturas, por ejemplo, estar sentado, recostado, parado, etc., que dificultan aún más la detección. Otros factores que también influyen, como la distancia a la que se encuentra la(s) persona(s) del dispositivo, la iluminación del ambiente, el peso y la altura de las personas etc., representan un gran desafío en la correcta y rápida detección de personas.

Otro aspecto que dificulta la detección de las personas es que éstas pueden estar parcialmente obstruidas por cosas que están en el ambiente, como muebles, paredes, o sólo por que el usuario está muy cerca al robot. En muchas ocasiones lo único que se alcanza a ver son sólo algunas partes del cuerpo de la o las personas. Por esta razón se optó por hacer la detección de la cabeza,

o de la cabeza junto con los hombros, ya que son las partes que mejor se distinguen de otras partes del cuerpo humano y que son menos obstruidas por los muebles de una casa o de una oficina debido a que éstos lugares son en los que comúnmente operan los robots de servicio.

La cámara es el dispositivo más usado en detección de personas, por ser un dispositivo accesible para la mayoría de la gente. Además el tiempo que tarda en tomar una captura es muy rápida, en comparación con algunos otros dispositivos.

2.1.1. Cámaras fotográficas

La gran mayoría de los métodos que se han implementado, se basan en el procesamiento de las imágenes capturadas por éste tipo de dispositivos. Por ejemplo, uno de los trabajos más reconocidos para la detección de rostros es el de Viola y Jones [4], en donde presentan un framework que es capaz de procesar imágenes muy rápido, teniendo un alto porcentaje de detecciones. Este trabajo se basa en una técnica llamada AdaBoost en la que se genera una serie de detectores simples, que individualmente no son muy prácticos debido a que su margen de éxito esta un poco arriba del 50 %, pero que usándolos en *cascada* hacen que la detección se vuelva eficiente.

Este método descarta rápidamente las regiones de la imagen que no sean un rostro y se concentra en las regiones en las que sea más probable que se encuentre.

Roth et al. [5], hace un framework donde la idea básica es iniciar con un sistema de detección simple y explotarlo con una gran cantidad de imágenes

en videos que no están etiquetados. La idea principal es usar clasificadores reconstructivos y discriminativos, en un co-entrenamiento iterativo, para obtener cada vez mejores detectores. Su propuesta consiste en tres fases: *el detector de movimiento* que entrega las regiones de la imagen donde encuentre algo que se mueva, *el modelo reconstructivo* que realiza una detección del cuerpo humano para hacer una reconstrucción de la región de la imagen que entregó el detector de movimiento y por último, *el modelo discriminativo* que realiza otra fase de detección pero ésta es más robusta que la del modelo reconstructivo. El método de detección que usan, basado en el movimiento de fondo, se hace tomando áreas de la imagen donde se detecte movimiento en una serie de imágenes consecutivas, que son tomadas mediante una cámara de vigilancia en condiciones estáticas. La segunda y tercera fase de éste sistema se retroalimentan entre si con las detecciones realizadas por cada una de ellas, con ésto, aseguran la robustez y fiabilidad del sistema.

Otro de los trabajos importantes para la detección que usa imágenes capturadas por una cámara fotográfica y en el cual se basa este trabajo, es el de Dalal y Triggs [3]. En este trabajo se hace un estudio acerca de un tipo de características que se extraen de las imágenes, estas características son llamadas gradientes orientados.

Las orientaciones de cada gradiente son agrupadas en histogramas, creando lo que Dalal y Triggs[3] llaman Histogramas de Gradientes Orientados (HOG). Estos histogramas forman un vector que describe una región de la imagen, al cual se le llama descriptor HOG.

Mediante una técnica de aprendizaje automático estos descriptores HOG

son procesados para obtener un clasificador, que hará la distinción entre las regiones de una imagen que sean personas o que no sean personas. De esta forma, Dalal desarrolló un clasificador de personas de cuerpo completo. En la evaluación que hizo para este detector usando este tipo de características, mostró que tiene un buen rendimiento y que ademas es menos propenso a equivocarse. Dalal también exploró la idea de combinar detectores de partes del cuerpo humano. Su hipótesis es que estos detectores combinados podían ser más robustos a oclusiones.

Dalal [6] entrenó 3 detectores diferentes: cabeza hasta los hombros, torso y piernas. Combinó la salida de cada uno de los detectores con base en la relación del cuerpo humano que existe entre ellos, utilizando diferentes técnicas de fusión de datos. Por ejemplo, una técnica simple que usaron fue únicamente combinar las tres diferentes detecciones(una de cabeza hasta los hombros, una de torso y una de piernas) en una sola detección.

Otra técnica usada por Dalal [6] fue el de calcular el centroide de la persona en base a la localización de las partes del cuerpo, arrojadas por sus correspondientes detectores (previamente descritos), y con base en este centroide combinar las detecciones.

Sus resultados muestran que esta fusión de datos entre los diferentes detectores incrementan el desempeño, dando un mayor número de detecciones correctas. Sin embargo para este caso, ya no solo se está llevando a cabo la detección de un cuerpo completo, sino tres partes diferentes de él. Esto podría afectar el tiempo en el que se lleva a cabo la detección, además del tiempo que se lleva en el proceso de fusión de los datos arrojados por cada detector.

Estos sistemas propuestos se basan en cámaras fijas y tenemos que recordar que nuestro objetivo es aplicarlo a un robot móvil.

Figura 2.1. *Imágenes de los resultados finales del detector de cuerpo completo. Fuente: Dalal [6].*

2.1.2. Cámaras estereoscópicas

Estas cámaras intentan emular la visión humana utilizando dos cámaras fotográficas de color. Con la obtención de estas dos imágenes, donde la diferencia entre ellas nos da información de la profundidad (disparidad), puede ser utilizada para calcular la distancia entre algún objeto y la cámara.

Las cámaras estereoscópicas no son tan utilizadas ya que no están al alcance de todas las personas, en comparación con las cámaras fotográficas que además obtienen resultados similares en detección de personas. Otro aspecto es que el costo computacional al usar cámaras estereoscópicas es más elevado que con una cámara fotográfica, debido a que se tiene que hacer el procesamiento de la detección en dos imágenes, ademas de la fusión de datos entre los resultados de estas dos. Por otro lado, la información adquirida solo es complementaria, es decir, la información de la distancia puede no ser ocupa-

da para hacer la detección de una persona en la imagen, aunque para otras tareas podría ser de utilidad.

Como ejemplo de investigación para esta área en un robot de servicio, Junki Satake y Jun Miura [7], utilizan las imágenes de una cámara estereoscópica para hacer detección y rastreo de personas para un robot móvil. Ellos proponen un método de detección usando plantillas de profundidad para la peculiar forma del cuerpo humano de la cabeza y los hombros, aplicado a una imagen con una profundidad densa.

Junki Satake y Jun Miura [7] también desarrollan un verificador para eliminar los falsos positivos.

Para el rastreo en este trabajo, se usa una técnica que permite estimar continuamente la posición y la velocidad de las personas en las coordenadas locales del robot. Estas estimaciones son usadas para el control correcto del movimiento del robot.

Las partes notables de este trabajo son que su robot puede seguir a una persona en específico, logrando reconocer la diferencia entre la persona que está rastreando de las otras personas que detecta y que es robusto a obstrucciones ocasionales. Sin embargo, el algoritmo que usan no contempla situaciones donde las personas estén muy cerca unas de otras sin importar la diferencia en profundidad de las personas. Para poder resolver este problema ellos proponen usar otro tipo de información visual como el color o la textura.

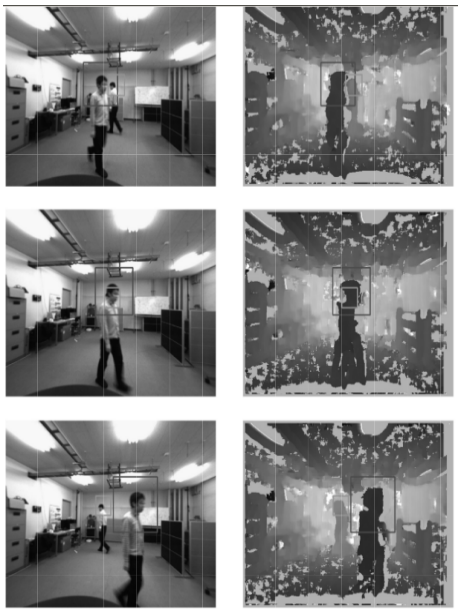

Figura 2.2. *Un correcto rastreo de personas con oclusión temporal. Fuente: Junki Satake y Jun Miura [7].*

2.1.3. Cámaras térmicas

Una cámara térmica es un dispositivo que crea una imagen a partir de la radiación térmica. La principal diferencia que existe entre este tipo de cámaras y las cámaras fotográficas, es que éstas últimas capturan una imagen en el rango del espectro electromagnético de la luz visible, que se encuentra aproximadamente entre los 400 y los 700 nm mientras que las térmicas lo capturan en el rango infrarrojo entre los 50 μm y 1 mm de longitud de onda.

Las cámaras térmicas son usadas en la detección de personas porque tienen algunas ventajas respecto a otros dispositivos usados para la detección; los problemas como la iluminación, la perspectiva de la imagen, la distancia

entre el dispositivo y la persona, entre otros, son de menor importancia, debido a que cualquier objeto que tenga una temperatura mayor al del ambiente, tendrá un color distintivo en las imágenes captadas por las cámaras térmicas, respecto a otros objetos que tienen una temperatura menor o igual a la del ambiente. Bajo este concepto, es relativamente más fácil detectar personas en este tipo de imágenes, ya que el problema se reduce únicamente a la forma del cuerpo humano, debido a que las personas tendrán un color diferente al del ambiente.

André Treptow et al. [8], combinan un filtro de partículas con dos modelos alternativos, que son factibles para el rastreo en tiempo real con una cámara térmica y que se explican a continuación. El primero de los modelos utilizados, es un modelo de contorno, que mide la probabilidad de que el objeto tenga forma de una persona.

El segundo modelo está basado en simples valores de características grises. Ellos aplican el sistema desarrollado a un robot móvil del tipo PeopleBot (especializado en interacción con personas) para un proyecto al que llaman "guardia de seguridad robótico", donde una de las tareas del robot es identificar gente en un edificio mientras patrulla por los corredores. Como la cámara térmica no necesita de luz visible, el robot puede distinguir personas incluso en completa oscuridad, lo cual es una ventaja considerable para un guardia de seguridad.

Uno de los problemas que presenta este sistema es que el modelo que adoptan comúnmente falla cuando las personas hablan, estrechan las manos,

caminan en grupos o cualquier tipo de acercamiento que tenga una persona a otra, ya que utiliza un filtro de partículas haciendo sensible el sistema al movimiento cercano entre personas.

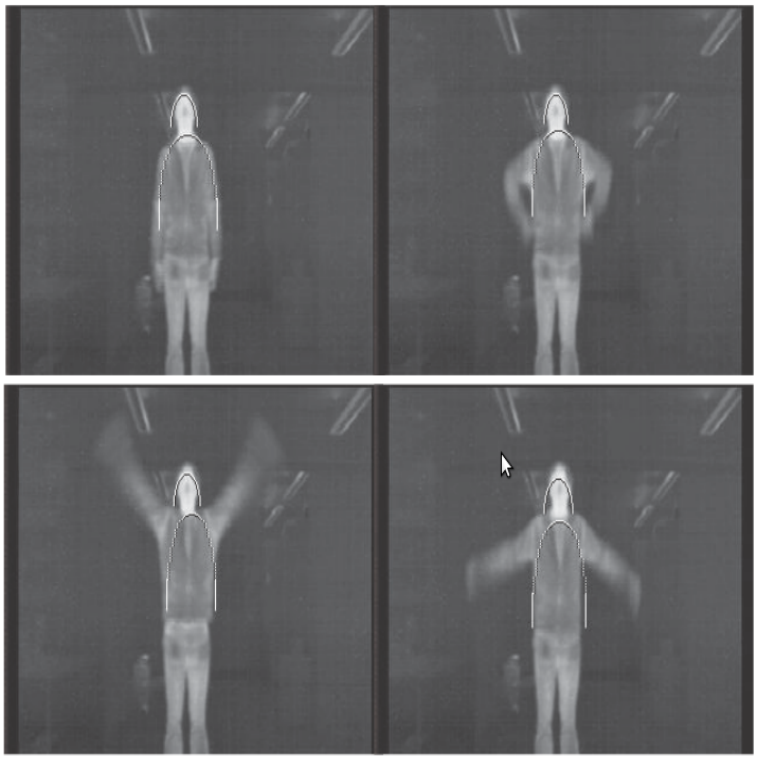

Figura 2.3. *Rastreo de una persona con una cámara térmica mediante el sistema de André Treptow et al. Fuente: André Treptow et al. [8].*

2.2. Otros sensores

En esta sección se explica la forma en que se utilizan algunos sensores que no entran en la clasificación de cámaras pero que también son utilizados para realizar la detección de personas.

2.2.1. Láser

El láser es un dispositivo electrónico que, basado en la emisión inducida, amplifica de manera extraordinaria un haz de luz monocromático y coherente. La palabra láser proviene del acrónimo en inglés de *ligth amplification by simulated emission of radiation*, que quiere decir, amplificación de luz mediante emisión inducida de radiación[9].

Los láser que comúnmente se utilizan en los robots de servicio son los láser de tipo *rangefinder*, que utiliza un haz de luz para determinar la distancia respecto a un objeto. La mayoría de estos dispositivos funcionan lanzando un pulso del láser a un objetivo y miden el tiempo que tarda el haz de luz en regresar al dispositivo. Con esta medida de tiempo y conociendo la velocidad de la luz, se puede calcular aproximadamente a que distancia se encuentra el objetivo.

Belloto[2] y Kornienko[10], describen métodos que se usaron para detectar personas usando láser. Estos métodos obtienen la posición de objetos que parezcan piernas humanas que aparecen en los resultados de la lectura arrojados por el láser, con base a un modelo que contiene patrones comunes de las diferentes posiciones de las piernas de las personas. Belloto[2] utiliza la información de varios sensores, entre ellos el láser, para realizar la detección. Tiene un detector de rostros que utiliza una cámara fotográfica y con el conjunto de información de los dos detectores (piernas y rostros) se usa para hacer el seguimiento de la persona que ha sido detectada. Becerra[11] también hace detección a través de láser, además, hace un análisis de diferentes

métodos de rastreo con base en estas detecciones y un módulo de planeación de movimientos que toma como base las detecciones que fueron tomadas a partir del láser.

Desafortunadamente el patrón que se busca con el láser es muy común en otros objetos de la vida cotidiana como mesas o sillas o cualquier cosa que tenga dos postes separados un poco entre ellos. Otra de las dificultades a las que se enfrentan este tipo de detectores es a reconocer cuando la persona tiene las piernas juntas ya que en este caso es aun más complicado hacer la distinción entre una persona y cualquier otro objeto que tenga una forma parecida. Estos casos provocan muchos falsos positivos en las detecciones de personas través del láser.

En la figura 2.4 podemos observar un ejemplo de sensor láser, este es el usado por el robot Golem II+. Se trata de un láser Hokuyo UTM-30LX. A la derecha, una ilustración de la trayectoria que sigue el láser y cómo llega al receptor, mediante esta técnica se calcula la distancia entre el láser y el objetivo.

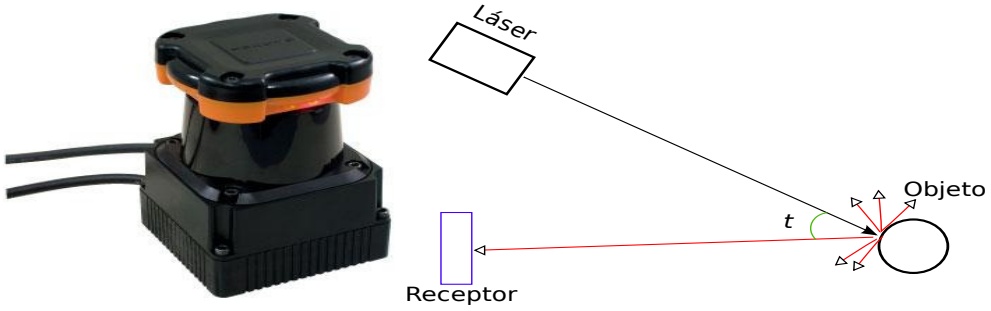

Figura 2.4. *Láser que usa el robot Golem-II+.*

2.2.2. Sensor infrarrojo

El dispositivo más común que utiliza sensores infrarrojos es el Kinect. El Kinect es un dispositivo sensor de movimiento creado por Microsoft para la consola Xbox 360 como reemplazo al control manual. Está constituido por una cámara RGB, un emisor de luz infrarroja, un sensor de profundidad infrarrojo que recibe la luz infrarroja del emisor y mide la profundidad que hay entre un objeto y el sensor, un arreglo de micrófonos y un acelerómetro de 3 ejes[12].

El Kinect es comúnmente usado porque trabaja sobre niveles de luz muy bajos, los resultados que ofrece son invariantes al color y la textura, y resuelve ambigüedades de la silueta en las diferentes poses del cuerpo humano. También simplifica enormemente la sustracción de la imagen de fondo.

El Kinect es un dispositivo que ofrece excelentes resultados, aunque entre algunas de sus desventajas es su precio, que no es tan bajo como el de algunas cámaras fotográficas, aunque tampoco es tan elevado. Este dispositivo requiere de una distancia mínima y una máxima para que se pueda hacer la detección y el rastreo de personas. El Kinect sólo puede diferenciar 6 personas a la vez y sólo puede hacer rastreo detallado con 2 de estas personas [12], aunque este punto puede ser solo definido por el o los métodos de detección que utiliza. Otra de las desventajas que presenta es que las personas detectadas siempre deben estar de pie.

Lu Xia et al. [13] utilizan la información de profundidad del Kinect para

hacer la detección de personas. El método que proponen esta basado en dos modelos para la detección de la cabeza, uno de ellos de 2 dimensiones, que distingue el contorno de la cabeza y los autores lo utilizan principalmente para obtener aquellas regiones del entorno que puedan ser personas. El otro es un modelo de 3 dimensiones que detecta la superficie de la cabeza y éste se utiliza para construir una estimación de la cabeza de la persona a partir de la región detectada por el primer modelo. También proponen un método para encontrar el contorno del cuerpo humano completo a partir de las detecciones que se hicieron de la cabeza. Este método utiliza la información de la profundidad dada por el Kinect. También exploran un algoritmo de rastreo basado en el movimiento de los objetos resultados de la detección. Este rastreo se hace en base a la velocidad del objeto y se realiza una estimación de la posición de los pixeles(del objeto detectado) en base a esta velocidad.

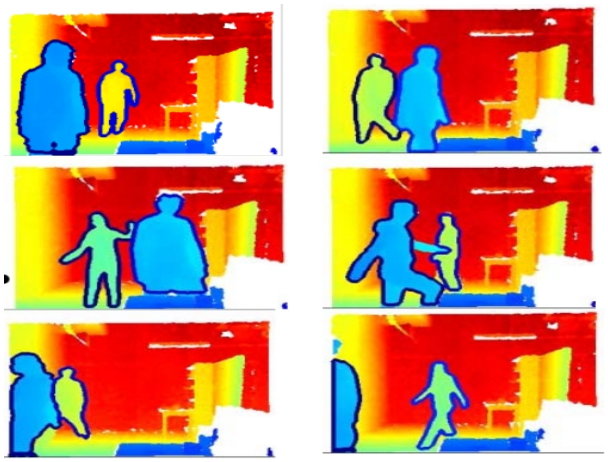

Figura 2.5. *Detector de personas utilizando la información del Kinect. Fuente: Lu Xia et al.[13].*

2.3. Híbridos

Algunos trabajos optan por usar varios dispositivos y hacer un sistema que tome las mejores características de cada uno de ellos. La mayoría de los trabajos que siguen este método no utilizan más de dos dispositivos, ya que implica un mayor desarrollo y no garantiza que los resultados sean muy diferentes a los obtenidos con solo dos.

Nicola Belloto[2] propone una solución al rastreo de personas para un robot móvil realizando la fase de detección con base en dos módulos, uno de ellos lleva a cabo la detección de piernas y el otro realiza la detección de rostros. El sistema utiliza un algoritmo basado en láser para la detección de piernas, en este algoritmo utilizan patrones de piernas tanto de personas que están de pie como personas que están caminando, incluso cuando el robot se esta moviendo. Una vez que se obtiene la información del rostro, esta se fusiona con la información de la posición de los pies de la persona en base a las posiciones que se obtienen de cada una de las partes. Para este sistema, la fase de rastreo se hizo con el filtro de Kalman y con un algoritmo de asociación de datos llamado *Nearest-Neighbor* que encuentra, en cada secuencia de imágenes, la relación entre las personas que ya han sido detectadas previamente y donde fueron detectadas actualmente. El filtro de Kalman se explica detalladamente en el capitulo 4 de este trabajo.

Este sistema trabaja correctamente la mayoría de las veces, solo cuando dos personas están muy juntas entre ellas con una velocidad y orientación similar, causa que los rastreos se intercambien.

S. Burion [14] elabora una propuesta para desarrollar un robot que ayude a las personas en caso de desastre(incendios, derrumbes, inundaciones, etc.), ya que las personas que ayudan en este tipo de incidentes típicamente tienen que tomar decisiones rápidas bajo estrés e intentar salvar a otras personas bajo su propio riesgo. Los robots que ayudan en caso de desastre son llamados Robots de Búsqueda Urbana y Rescate"(USAR por las siglas en inglés de *Urban Search And Rescue Robots*).

El objetivo del trabajo de S. Burion, es el de proveer a un robot móvil de un conjunto de sensores para hacer la detección de personas. Los sensores que usan son los siguientes:

Sensor piroeléctrico: esta hecho de un material cristalino que genera una pequeña carga eléctrica cuando es expuesto al calor en forma de radiación infrarroja. El dispositivo que usan fue diseñado especialmente para detectar personas, si alguna parte del cuerpo humano cruza o pasa por enfrente del sensor esté se activa. Si se cruza otro tipo de elemento, no va activar al sensor.

Cámara fotográfica: para detectar el movimiento en dos imágenes consecutivas haciendo una diferencia entre los pixeles que no cambiaron y los que si cambiaron. Una de las limitaciones de esta propuesta es que la cámara no debe estar en movimiento.

Micrófono: lo usan para detectar voz humana. Este tipo de estrategia es usada porque suponen que en caso de desastre se detienen todas las actividades para tratar de escuchar a una víctima que esté gritando.

Cámara infrarroja: toma las imágenes térmicas en escala de grises. El método aprovecha que en la imagen la piel es de un color blanco uniforme y

toman esta característica para detectar personas. El objetivo es detectar las partes más blancas de la imagen ya que estas partes podrían ser porciones del cuerpo humano.

La solución que S. Burion propone para la fusión de los enfoques, es asignar un cierto valor de confianza a cada uno de los dispositivos, dependiendo de los resultados que arrojaron cada uno de ellos al ser evaluados. Pero cada fusión de enfoque hace que la detección se vuelva lenta, aunque aumenta la precisión de la detección, es decir, debido a que cada dispositivo arroja información de la posibilidad de que en una región se encuentre una persona, si la mayoría o todos los dispositivos concuerdan en que ahí hay una persona, la probabilidad de que sea cierto es muy alta.

2.4. Motivo de elección

La interacción del robot con las personas es uno de los aspectos más importantes que puede tener un robot y es una de las formas en que se realiza un intercambio de información entre cada parte. La detección de personas para un robot de servicio es un punto crucial para esta interacción, ya que proporciona información del lugar en donde las personas se encuentran.

Para este sistema, se decidió usar la cámara fotográfica ya que es uno de los dispositivo más usados para el reconocimiento de personas y que ha tenido resultados muy aceptables, además, se cree que la información de la localización que se extrae de la imágenes ofrecidas por la cámara, puede ser fusionada más fácilmente que con la información de los demás dispositivos.

La cámara fotográfica es también un dispositivo que puede conseguirse con mayor facilidad que el resto de los aquí mencionados y al alcance de la mayoría de las personas.

Dispositivo	Precio	Fortalezas	Desventajas
Cámara fotográfica	bajo	precio	baja resolución, costo computacional medio, sensibilidad a la iluminación
Cámara estereoscópica	medio	información de profundidad	alto costo computacional, sensibilidad a la iluminación
Cámara térmica	alto	fácil distinción humana, no le afecta la iluminación, bajo costo computacional	difícil adquisición
Láser	medio	precisión en mediciones, rápido, no le afecta la iluminación	medición en sólo 1 dimensión
Sensor infrarrojo	medio	fácil distinción humana, no le afecta la iluminación	pocas personas, alcance limitado

Cuadro 2.1. *Cuadro resumen de dispositivos de sensado para detección de personas.*

El cuadro 2.1 muestra una comparación entre los diferentes dispositivos mencionados, mostrando algunas de sus fortalezas que los hacen adecuados para la detección de personas. Se puede observar fácilmente que la cámara fotográfica es una opción apropiada para la tarea.

Capítulo 3

Detección de personas basado en HOG

En este capítulo se presenta el método que se usa para llevar a cabo la detección de personas en imágenes. Se da a conocer cómo se hace la extracción de características HOG y cómo se usan para este sistema. Se analiza el método que obtiene el núcleo de la detección: *el clasificador* y cómo se usa en el proceso para hacer la separación entre personas y no personas, mediante un algoritmo de aprendizaje automático que toma ejemplos. También se explica la forma en que se hace la detección de personas paso a paso, desde que se obtiene la imagen hasta que se sabe la cantidad y localización de las personas en esa imagen.

El método que se usó para resolver el problema de detección, es el propuesto por Dalal y Triggs [3] para la detección de personas, ya que ha mostrado un buen desempeño en la detección de objetos y personas y éste no se ha portado a robots de servicio.

Aunque es un buen método de detección, éste ha sido aplicado en su ma-

yoría a problemas de vigilancia, donde todas o casi todas las personas aparecen de cuerpo completo y en un ambiente donde la cámara nunca se mueve.

Este método propone un conjunto de características que funcionan en detección de objetos y reconocimiento [3], [15], y estas características han sido extensamente aceptadas como de las mejores para capturar bordes o información local de formas [16].

3.1. Histogramas de gradientes orientados

HOG es el acrónimo del inglés *Histograms of Oriented Gradients* que significa *Histogramas de Gradientes Orientados*, y es un método de extracción de características, donde la idea es que la apariencia y forma local de un objeto, puede ser caracterizada por la manera en que la dirección e intensidad de la iluminación, cambian en el borde (o contorno) del objeto en una región específica de la imagen.

Para saber el cambio en la dirección y la intensidad de iluminación en la región, se hace uso de vectores que describen estos cambios. A estos vectores se les conoce como *vectores gradiente*.

3.1.1. Cálculo de los gradientes

El vector gradiente $\nabla\psi$ de un campo escalar ψ evaluado en un punto p, es un vector que indica la dirección hacia donde el campo varía más rápidamente

y el módulo de este vector indica el ritmo de variación del campo en ese punto evaluado. El gradiente de una función f en coordenadas cartesianas se define de la siguiente forma:

$$\nabla f = \frac{\delta f}{\delta x} i + \frac{\delta f}{\delta y} j \tag{3.1}$$

El gradiente de una imagen captura el cambio en la intensidad o en el color de la imagen y la dirección hacia donde cambia. Se puede obtener aplicando un filtro a la imagen. Para aplicar este filtro, se hace una convolución entre la matriz de pixeles de la imagen y una matriz kernel o la matriz de filtro. Para este caso se usarán dos filtros, uno aproxima la derivada en x y el otro en y, obteniendo dos imágenes que tendrán la derivada de cada pixel en sus respectivos ejes. Las matrices que se usarán como filtro para este caso son conocidas como *Filtro de Sobel* y son las siguientes:

$$S_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}, S_y = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix}$$

Aplicando una operación de convolución entre alguna de estas dos matrices y la matriz de pixeles de la imagen obtenemos las aproximaciones de las derivadas, S_x se usa para la aproximación en el eje x y S_y para la aproximación en el eje y, tal y como se muestra en las siguientes expresiones:

$$Gx = S_x * M \tag{3.2}$$

$$Gy = S_y * M \tag{3.3}$$

Donde M es la matriz de pixeles de la imagen a la que se le quiere aplicar el filtro, G_x es la matriz que contiene las aproximaciones de las derivadas en el eje x y G_y para el eje y.

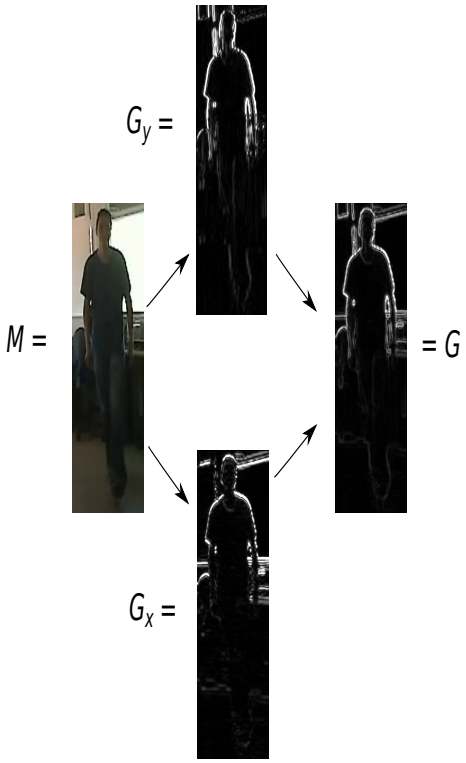

Figura 3.1. *Ejemplo del cálculo de las derivadas mediante el filtro de Sobel y el resultado de combinarlas.*

Para calcular el gradiente en cada pixel de la imagen se utilizan las siguientes ecuaciones:

$$\theta_{(i,j)} = \arctan\left(\frac{Gy_{(i,j)}}{Gx_{(i,j)}}\right) \tag{3.4}$$

$$|G_{(i,j)}| = \sqrt{\left(Gx_{(i,j)}\right)^2 + \left(Gy_{(i,j)}\right)^2} \tag{3.5}$$

Con la ecuación 3.4 obtendremos el ángulo que nos indica la dirección del vector y la ecuación 3.5 se usa para obtener el módulo del vector que nos indica la intensidad de cambio del campo en ese punto.

En la Figura 3.1 se obtienen las derivadas G_y y G_x a partir de la imagen M para después sumarlas y obtener la imagen G. Esto nos da un ejemplo visual de la forma en que se aplica el filtro de Sobel.

Al aplicar este filtro a la imagen de entrada obtenemos como resultado otra imagen que contendrá la información de los bordes de las personas y de los objetos que se encuentren en la imagen. Ésto reduce la cantidad de información que tiene que ser procesada debido a que solo se tomarán en cuenta estos bordes de la imagen.

Esta síntesis de información se utiliza tanto para hacer el entrenamiento del clasificador, como para analizar la imagen en la que se hará la detección de personas, esto hace que la detección se haga más rápida que si se analizara la imagen tal cual llega al sistema. También permite que con el contorno de las personas se distinga mejor entre una persona y cualquier otro objeto del ambiente, debido a que de esta forma se puede percibir de manera más rápida su ubicación debido a la forma característica de nuestro cuerpo.

3.1.2. Método para la extracción de características HOG

En este método las imágenes son divididas en pequeñas regiones llamadas *celdas*. Para cada celda se acumula un histograma de direcciones de gradiente u orientaciones de contorno sobre los pixeles de la celda.

Para considerar los cambios en iluminación y contraste de las celdas adyacentes, los gradientes deben normalizarse localmente, por lo que se necesita agrupar a las celdas en regiones más largas, a estas regiones se les llama bloques.

Este método de extracción de características consta de 5 fases, donde se empieza por darle un tratamiento a la imagen para tener ciertas condiciones especiales que hacen que los cálculos posteriores sean más rápidos. Después se divide la imagen en celdas, las celdas se agrupan en bloques y con estos bloques se hacen los cálculos de los histogramas para formar el descriptor HOG de la imagen.

A continuación se describen las 5 fases para el cálculo de lo descriptores HOG:

1. Se le aplica una normalización a la imagen para reducir los efectos de iluminación.

2. Se hacen los cálculos para obtener los gradientes de primer orden de la imagen, como fue descrito en la sección 3.1.1. Estos gradientes capturan los bordes, siluetas y alguna otra información de textura. También dan una reducción adicional a la influencia de las variaciones en iluminación de la imagen.

3. Se divide la imagen en pequeñas regiones espaciales llamadas celdas. Para cada celda se acumula un histograma del gradiente o de orientaciones del borde, sobre todos los pixeles de la celda. Esta combinación de histogramas a nivel celda forman la representación de un *histograma de orientaciones*. Cada histograma de orientaciones divide el rango del ángulo del gradiente en una serie de subrangos fijos, por ejemplo, la orientación de un gradiente puede esta contenida en un rango de $[0° - 360°]$, entonces se pueden hacer rangos fijos de: $[0° - 20°), [20° - 40°)...[340° - 360°]$ para agrupar las direcciones de los gradientes y no se almacene tanta información (Dalal[6] verificó que con una amplitud de $20°$ por rango, se obtienen resultados aceptables). Con esto se hace una suma de las magnitudes de los gradientes que tienen ángulo de orientación comprendido entre alguno de esos valores para formar el histograma de orientaciones.

4. Se toman grupos locales de celdas a los que se llama bloques, y normaliza el contraste de los bloques antes de pasar a la siguiente fase. Cada bloque se forma con una cantidad fija de celdas. Éstos se forman horizontalmente recorriendo una por una las columnas de celdas de izquierda a derecha y verticalmente se forman recorriendo una por una las filas de arriba hacia abajo. Esta forma de obtener los bloques hace que algunas celdas estén compartidas por varios bloques, lo que resulta en una mejora de rendimiento. A estos bloques normalizados se les llama *Descriptor de Histograma de Gradientes Orientados* (Descriptor HOG).

5. Esta fase consiste en recolectar los descriptores HOG de todos los bloques de la cuadrícula densamente traslapada que cubre la ventana de detección en un vector combinado de características para usarlo en el clasificador de ventanas.

La Figura 3.2 muestra el efecto de cada una de las fases descritas, las fases inician después de que se obtiene la imagen de entrada.

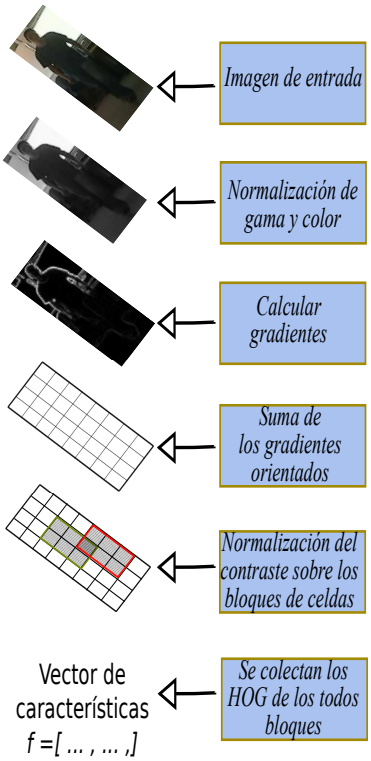

Figura 3.2. *Fases de la extracción de características HOG, para la obtención del descriptor HOG de una imagen*

Con este método de extracción de características HOG se tiene una manera más rápida de análisis de información de la imagen ya que provee de

una especie de *resumen* o *síntesis*, haciendo la extracción sólo de la información que es relevante para el proceso al que se someterá. Esto ayuda a que la detección de personas se haga más rápido, siendo esto fundamental en un robot móvil, sobre todo si se busca una respuesta rápida del robot.

3.2. Máquina de vectores de soporte (SVM) lineal

Una máquina de vectores de soporte (del inglés *Support Vector Machine - SVM*) es un método de aprendizaje automático que a partir de los datos de entrada, reconoce patrones en estos datos y resuelve problemas de clasificación binaria.

El método crea un clasificador discriminatorio que se define mediante un hiperplano que separa los datos. Estos datos deben estar etiquetados como positivos y como negativos para que la salida del método sea un hiperplano óptimo que categorizará nuevos datos de entrada.

3.2.1. Clasificación

El clasificador que se obtiene mediante máquinas de vectores de soporte, separa datos en dos clases diferentes. Bajo este concepto, los datos de entrada estarán etiquetados como positivos con la etiqueta +1 y como negativos con un -1. A cada dato de entrada se le llama vector, ya que se trata de un

conjunto de datos asociados, que para este caso consiste en el vector de características HOG previamente descrito. A cada vector se le denotará como x_i, siendo el *iésimo* elemento del conjunto de vectores.

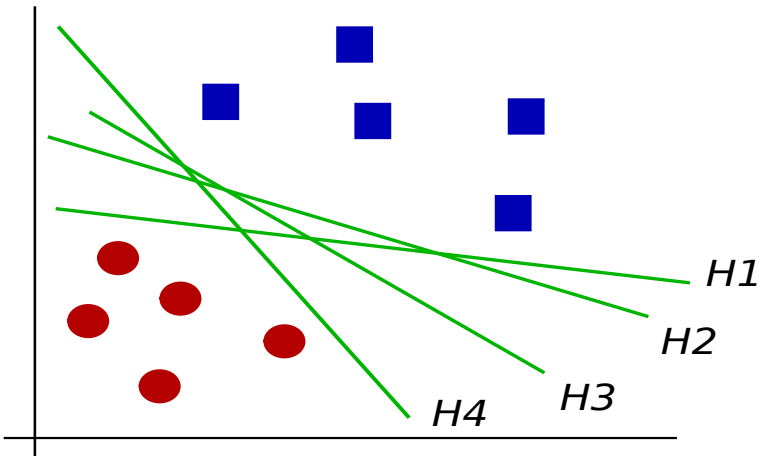

Figura 3.3. *Hiperplanos que dividen dos clases de objetos*

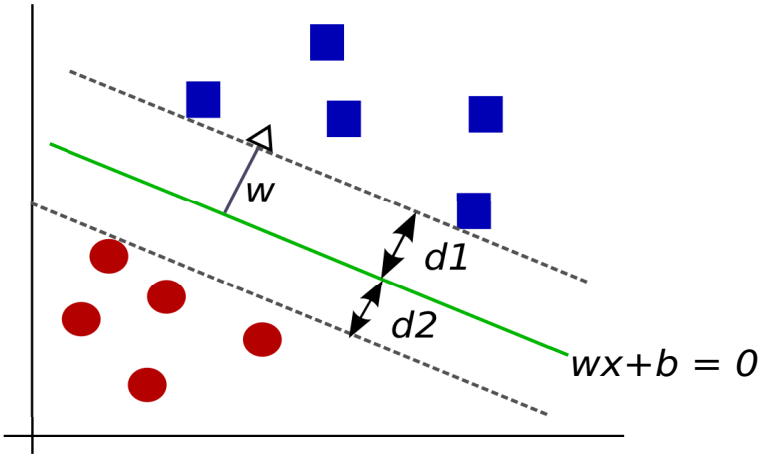

Figura 3.4. *Hiperplano óptimo que divide las dos clases de objetos*

Como se aprecia en la Figura 3.3, pueden existir múltiples soluciones al

problema de clasificación, por este motivo, se tiene que encontrar la solución óptima al problema. Para esto se debe hallar el hiperplano que separe las clases con la distancia mayor entre los vectores cercanos.

En un espacio de una sola dimensión (por ejemplo una recta), un hiperplano es un punto ya que divide el espacio en dos lineas. En un espacio de dos dimensiones (por ejemplo un plano) un hiperplano es una recta ya que divide el espacio en dos mitades. En un espacio tridimensional un hiperplano es un plano debido a que divide el espacio en dos. Haciendo una generalización del concepto: En un espacio afín de dimensión n, las variedades de dimensión $n-1$ se llaman *hiperplanos* [17].

La ecuación 3.6 es la forma general de un hiperplano, mientras que la ecuación 3.7 nos muestra la condición que debe cumplir un hiperplano para separar dos clases de datos:

$$f(x) = wx + b \tag{3.6}$$

$$wx + b = 0 \tag{3.7}$$

donde w es el vector conocido como el *vector de pesos* y que es normal al hiperplano, b es conocido como *bias* y es la separación que hay desde el origen hasta el hiperplano.

En nuestro caso tenemos L vectores de entrenamiento, donde cada vector de entrada x_i tiene D atributos y cada uno de los vectores pertenece a una

clase $y_i = -1$ o $y_i = +1$, por consiguiente los datos de entrada tendrán la siguiente forma:

$$\{x_i, y_i\} \text{ donde } i = 1 \ldots L,\ y_i \in \{-1, +1\},\ x_i \in \Re^D$$

$$wx + b = +1 \quad para \quad y_i = +1 \tag{3.8}$$

$$wx + b = -1 \quad para \quad y_i = -1 \tag{3.9}$$

$$y_i\,(wx + b) = +1 \tag{3.10}$$

Lo que se quiere es separar los vectores de acuerdo al valor de su etiqueta y_i, usando dos hiperplanos diferentes como lo muestran las ecuaciones 3.8 y 3.9.

En la figura 3.4 podemos apreciar tres hiperplanos, $P1$, $P2$ y P. Donde P corresponde al hiperplano óptimo, $P1$ corresponde al hiperplano paralelo a P donde la separación entre éstos dos corresponde a la distancia entre el hiperplano P y el vector más cercano etiquetado como $+1$. $P2$ se define de la misma forma pero ahora usando el vector más cercano etiquetado con -1. LLamaremos *margen* a la suma de la distancia entre los hiperplanos $P1$ y P más la distancia entre $P2$ y P.

Para encontrar el hiperplano óptimo P es necesario encontrar un hiperplano paralelo a $P1$ y a $P2$ que tengan el *margen* tan grande como sea posible. La distancia entre dos hiperplanos paralelos, por ejemplo: $wx + b_1 = 0$ y $wx + b_2 = 0$ se obtiene mediante la expresión 3.11.

$$distancia = \frac{|b_1 - b_2|}{||w||} \tag{3.11}$$

Si replanteamos las ecuaciones de $P1$ y $P2$ como se muestra en las ecuaciones 3.12 y 3.13 respectivamente, podemos usar la ecuación 3.11 para calcular la distancia entre $P1$ y $P2$ que es igual a la suma de la distancia entre $P1$ y P mas la distancia entre $P2$ y P. En la ecuación 3.14 se sustituyen los valores de estos hiperplanos y al final se encuentra que el *margen* o la distancia entre $P1$ y $P2$ está dada por la ecuación 3.15.

$$wx + (b + 1) = 0 \tag{3.12}$$

$$wx + (b - 1) = 0 \tag{3.13}$$

$$distancia = \frac{|(b + 1) - (b - 1)|}{||w||} \tag{3.14}$$

$$distancia = \frac{2}{||w||} \tag{3.15}$$

El problema ahora es maximizar esa distancia o *margen*, que es equivalente a minimizar la función $L(w)$ de la ecuación 3.16 sujeta a ciertas restricciones. Las restricciones modelan el requerimiento del hiperplano para clasificar correctamente todos los ejemplos de entrenamiento x_i. Entonces la ecuación $L(w)$ con la restricción es:

$$\text{mín}\, L(w) = \frac{1}{2}w^2 \tag{3.16}$$

Sujeto a $y_i(wx_i + b) >= 1 \quad \forall i$.

Una vez que se obtiene la ecuación del hiperplano que será nuestro clasificador, sólo basta con sustituir el dato de entrada en x en esa ecuación y dependiendo de las etiquetas de los datos de entrenamiento, basta con verificar el signo del resultado y sabremos si se trata de un positivo o de un negativo, o en otras palabras, sabremos si se trata o no de una persona para el caso del detector.

Existe un parámetro que hace más flexible la decisión de la clasificación, éste se conoce como el *umbral de clasificación* y se trata de un número que nos indica una distancia mínima que debe existir entre el hiperplano y el vector de características que se está clasificando. Esta distancia solo se mide cuando el vector se ha clasificado como una persona. Si esta distancia es menor al umbral, entonces ese vector se tomará como si no fuera una persona.

El clasificador que se obtiene es el núcleo de la detección, debido a que éste separa las regiones de la imagen que nos interesan y que probablemente sean personas.

3.3. Metodología para la detección

El clasificador binario es sólo una parte del detector. El objetivo del detector es localizar precisa y correctamente a las personas que se encuentren en la imagen en todas las escalas posibles. Para que esto sea posible primero se deben encontrar las regiones de la imagen que parezcan ser una persona, para ello tendremos que elegir los candidatos que cumplan con esta condición.

Una vez que tenemos todos estos posibles candidatos en diferentes escalas, éstos se agrupan para tener solo un cuadro por cada posible persona detectada.

3.3.1. Ventana deslizante

El método más simple para encontrar los candidatos es llamado *ventana deslizante* o *escaneo exhaustivo* que consiste en recorrer una imagen mediante una *ventana*, que no es más que una región de la imagen que está delimitada por un cierto tamaño que se elige dependiendo de las proporciones del objeto que se quiere detectar. Por ejemplo, Dalal[6] utiliza una ventana con un tamaño de 64x128 pixeles ya que se adapta más en relación al aspecto del cuerpo humano.

Este algoritmo inicia desde la parte superior izquierda de la imagen y se va moviendo n pixeles hacia la derecha, hasta que llega a la última columna de pixeles en la imagen, después de eso, regresa hasta el lugar donde inició (esquina superior izquierda) y recorre n pixeles hacia abajo. Se repite este proceso hasta que se llega a la esquina inferior derecha de la imagen.

Una vez que termina, se puede elegir entre dos técnicas diferentes: hacer un escalamiento de la imagen o hacer el escalamiento pero de la ventana de detección, por un **factor de escala**. Dependiendo de la técnica elegida la imagen será escalada y se tendrá el mismo tamaño de ventana o, se tomará la ventana escalada dejando intacta la imagen para que el proceso comience de nuevo con los nuevos tamaños. Esto se repite una y otra vez re-escalando la

imagen o la ventana en cada iteración hasta que la imagen sea más chica que el tamaño de la ventana de detección en los dos casos.

Cuando se elige hacer el cambio de escala en la imagen, este cambio afectará la imagen de tal manera que ésta se va reduciendo. Cuando se elige que el cambio sea haga en la ventana de detección éste aumentará el tamaño de la ventana. Así, al final siempre se tendrá la imagen más pequeña que la ventana y cumpliéndose esta condición se podrá detener al algoritmo.

La técnica elegida para este trabajo fue el incremento en la ventana de detección, aunque se podría decir que las dos técnicas son lo mismo: una es la inversa de la otra.

En las Figuras 3.5 y 3.6 se ilustra como es el proceso de ventana deslizante: en la Figura 3.5 se comienza en la parte superior izquierda y se va recorriendo hacia la derecha y luego hacia abajo, hasta recorrer toda la imagen. El proceso se repite en las diferentes escalas, éste es un ejemplo usando la técnica de escalamiento de imagen sin cambiar el tamaño de la ventana. En la Figura 3.6 se muestra ahora el cambio de tamaño en la ventana tomando todas las escalas posibles.

Figura 3.5. *Método ventana deslizante, ejemplo 1.*

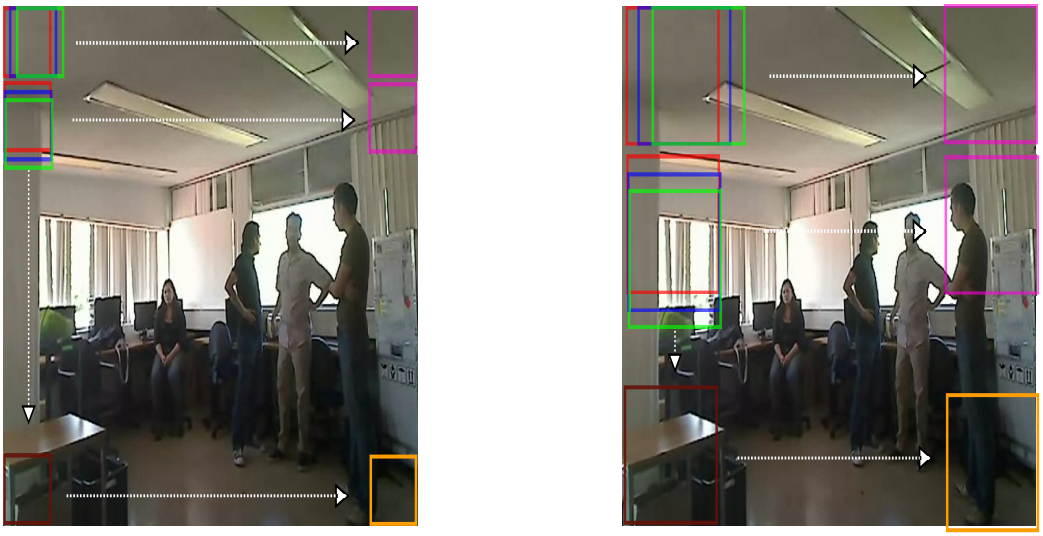

Figura 3.6. *Método ventana deslizante, ejemplo 2.*

3.3.2. Agrupación de los cuadros

El resultado del proceso anterior es una serie de cuadros que nos marcan la posible localización de una persona. El inconveniente es que son muchos cuadros muy cerca unos de otros y ésto se debe a que el escaneo puede encontrar a la misma persona una y otra vez en diferentes escalas de la imagen.

Para solucionar esto se usa una técnica para fusionar estos cuadros de una manera eficiente y precisa. Para realizar esta fusión o agrupamiento se debe considerar lo siguiente (Dalal[6]):

- Entre más cuadros estén sobrepuestos en una región de la imagen, entonces más alta es la probabilidad de que sea un verdadero positivo.
- Cuadros que estén cerca unos de otros deben ser fusionados, pero los cuadros que se encontraron en escalas muy diferentes no deben ser fusionados.

Una vez hecho lo anterior los cuadros que tengan tamaños y posiciones similares, se combinan en uno sólo, para así obtener el cuadro que mejor se acople a la posible persona detectada.

3.3.3. Algoritmo de detección completo

En este capítulo hemos revisado las bases del algoritmo de detección usado, los descriptores HOG que se utilizan para no usar toda la información de la imagen, las máquinas SVM que con base a ejemplos devuelve un clasificador de ventanas y el algoritmo de la ventana deslizante, que como se observa

en la Figura 3.8 agrupa todas las ventanas que sean probables detecciones para combinarlas y al final devolver sólo una ventana con la detección encontrada.

Estos procedimientos forman la fase de detección del sistema y son la base esencial para que posteriormente se pueda hacer el rastreo de las personas detectadas que se explica en el siguiente capítulo.

A continuación se presenta el algoritmo completo para la detección de personas.

Algorithm 1 Algoritmo de Detección

Require: Imagen de prueba con medidas W_ixH_i, Clasificador binario de ventanas de $W_n x, H_n$, Tamaño del paso de escala S_r.
Ensure: Vector de cuadros con las posiciones de las personas detectadas.

$S_s = 1$
$S_e = min(W_i/W_n, H_i/H_n)$
$S_n = floor\left(\frac{\log(S_e/S_s)}{\log(S_r)} + 1\right)$
while $W_n < W_i$ or $H_i < H_n$ **do**
    $y = 0$
    repeat
        repeat
            $x = 0$
            Colocar la ventana de detección en las coordenadas (x, y)
            Extraer las características HOG en el área de la ventana y se guardan en un vector V
            if la distancia entre V y el clasificador es menor al umbral de clasificación **then**
                almacenar x, y, H_n y W_n en un vector D
            end if
            $x+ = 1$
        until $x = W_i$
    until $y = H_i$
    $W_n* = S_r$
    $H_n* = S_r$
end while
Combinar cuadros similares que se encuentren en D
return D

Figura 3.7. *Pseudocódigo del algoritmo de detección*

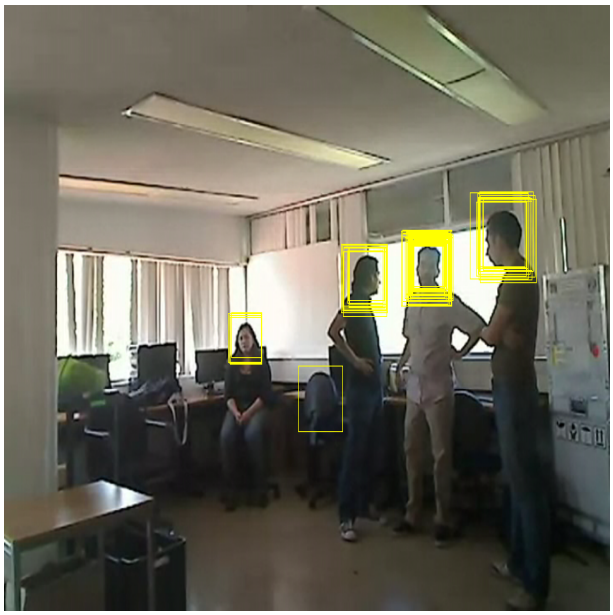

Figura 3.8. *Conjunto de cuadros al terminar el método ventana deslizante, deben ser agrupados para una detección homogénea.*

Capítulo 4

Rastreo multi-objetivo con el filtro de Kalman

En este capítulo se da a conocer la forma en la que se hizo el rastreo de personas. Para ello se necesita que las personas que ya se detectaron sean reconocidas en la siguiente imagen como las mismas personas, con ésto sabremos como se mueve cada una de las personas en el entorno. El algoritmo Húngaro resuelve este problema de relacionar las detecciones pasadas con las nuevas y será explicado con ejemplos que facilitarán su comprensión. Una vez que se conoce esta relación será momento de hacer la estimación de los cuadros con el filtro de Kalman. Al final del capítulo se describe el algoritmo completo del rastreo que se utilizó en este trabajo.

Cuando se ha detectado a las personas presentes en una imagen, lo siguiente es comenzar a realizar el rastreo o seguimiento de estas personas para tratar de conocer la manera en que se mueven e intentar predecir como lo seguirán haciendo. En 1960 R. E. Kalman publicó su famoso artículo que

describía una solución recursiva al problema del filtrado lineal de datos discretos, Kalman[18].

Desde ese tiempo y debido en gran parte a los avances en la computación digital, el filtro de Kalman ha sido objetivo de muchas investigaciones y aplicaciones Welch[19]. El filtro es muy poderoso en ciertos aspectos: mantiene las estimaciones del pasado, del presente y evalúa los estados futuros, y puede hacerlo aún cuando no se conozca con precisión la naturaleza del problema. Prácticamente, el filtro de Kalman es uno de los descubrimientos más grandes en la historia de la teoría de estimación estadística y posiblemente uno de los descubrimientos más grandes del siglo XX, Grewal[20].

Antes de proceder con el núcleo del rastreo, se tiene que tener alguna forma de relacionar las detecciones presentes con las pasadas que pertenezcan a la misma persona detectada. La manera en que esta relación se lleva a cabo es con el *Algoritmo Húngaro* que hace la correspondencia entre dos conjuntos, dependiendo de un costo que se tiene entre cada elemento de los conjuntos.

El nombre de este algoritmo fue dado por H. W. Kuhn en reconocimiento al trabajo de dos matemáticos húngaros: J.Egerváry y D.König, ya que él se basó en los trabajos de ellos. Kuhn plantea el problema de correspondencia como un problema de asignación de trabajos al personal, siendo la meta del

algoritmo encontrar la mejor asignación de un conjunto de personas o trabajadores a un conjunto de trabajos, Kuhn[21].

En la Figura 4.1 se muestra la arquitectura básica del sistema de rastreo, éste contiene tres bloques principales, el algoritmo Húngaro que devuelve la relación entre qué detecciones actuales corresponden a las estimaciones hechas con detecciones anteriores, el control del rastreo almacena, administra y etiqueta estas relaciones, así como el control del flujo de las detecciones y estimaciones, el filtro de Kalman tiene como entrada la relación de correspondencia entre las detecciones, hace una estimación de la siguiente localización de las detecciones y la regresa al control del rastreo, este control lleva acabo el rastreo de cada persona por separado.

A continuación se describirán los algoritmos utilizados para realizar el rastreo y después se presentará el proceso final de la fase de rastreo que se realizó para nuestro sistema.

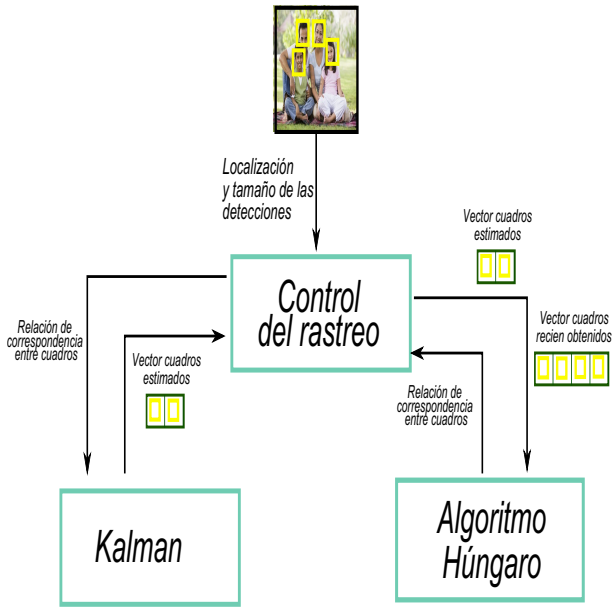

Figura 4.1. *Arquitectura del programa de rastreo.*

4.1. Algoritmo Húngaro

Este método es un algoritmo de optimización combinatoria que resuelve el problema de asignación en tiempo polinómico y fue planteado por H. W. Kuhn basándose en el trabajo de dos matemáticos húngaros [21]. El algoritmo resuelve el problema de asignación entre dos conjuntos que pueden estar relacionados entre ellos, estos dos conjuntos podrían verse como un grafo bipartito donde la relación se representa mediante una arista. Otra forma de representarlo, es con una matriz que relaciona a estos dos conjuntos, siendo uno representado por las filas y el otro por las columnas [21]. Kuhn menciona que la solución del problema puede verse como la mejor forma de asignar un conjunto de individuos a un conjunto de trabajos.

Individuo i	Calificado para trabajo(s) j
1	1, 2 y 3
2	3 y 4
3	4
4	4

Cuadro 4.1. *Ejemplo de individuos que están disponibles para trabajar.*

Antes de explicar el algoritmo se tiene que conocer el problema que se trata de resolver y los conceptos que se usarán para entender el método. Este se explicará mediante la analogía que usó Kuhn ya que de esta manera es mucho más fácil de entender.

4.1.1. El problema de asignación simple

Suponiendo que se tienen cuatro individuos que están disponibles para realizar cuatro trabajos, también se hace la suposición de que no todos los individuos están calificados para realizar todas las tareas, de modo que los individuos califican para los trabajos como se muestra en el Cuadro 4.1.

La información del Cuadro 4.1 puede ser presentada efectivamente como una matriz, en donde cada fila representa a cada individuo, cada columna representa cada trabajo y la intersección representa sí el individuo está o no calificado para realizar ese trabajo, tal y como se muestra en la siguiente matriz:

$$Q = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Como se puede observar, en la intersección entre la fila uno y la columna uno, hay un número 1 indicando que ese individuo está calificado para realizar ese trabajo, en la última columna se indica que el mismo individuo no está calificado para realizar ese trabajo y por lo tanto aparece un 0 en esa posición de la matriz.

El problema también puede ser representado mediante un grafo bipartito como se muestra en la Figura 4.2, donde cada individuo y cada trabajo se representan mediante un vértice en conjuntos separados. Cada arista representa que un individuo está calificado para cierto trabajo, por ejemplo, el individuo 1 está conectado al trabajo 1 indicando que es apto para realizarlo. Si el problema es visto de esta manera, la solución será aquella donde se consiga el mayor número de aristas posibles, teniendo sólo una arista para cada par de vértices.

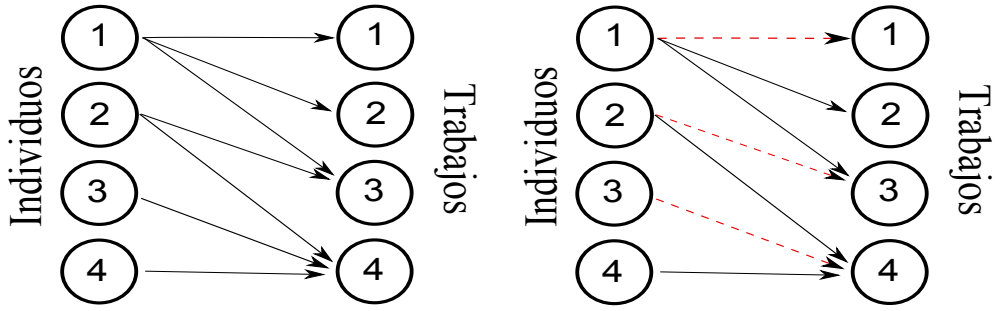

Figura 4.2. *(izquierda) Grafo bipartito del ejemplo representando el cuadro 4.1. (derecha) Una de las soluciones óptimas representada mediante aristas segmentadas.*

La pregunta del problema de asignación simple es:

¿Cuál es el mayor número de trabajos que pueden ser asignados a individuos calificados?. Con no más de 1 individuo asignado a un trabajo.

O en términos que se adecuan a la matriz:

¿Cuál es el mayor número de unos que pueden ser elegidos de Q, sin que sean de la misma fila o columna?

Se puede empezar a asignar trabajos a cualquier individuo que no tenga asignado un trabajo, siempre y cuando esté calificado para hacerlo. De este modo podemos asignar los individuos 1 y 2 a los trabajos 3 y 4 respectivamente y esta información puede ser indicada en la matriz colocando asteriscos, como se muestra a continuación :

$$Q = \begin{bmatrix} 1 & 1 & 1* & 0 \\ 0 & 0 & 1 & 1* \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Una vez que sea imposible incrementar las asignaciones de trabajos, de tal manera que no haya más individuos que puedan ser asignados a un solo trabajo para el que califique, se dice que la asignación está *completa*. La asignación que se hizo anteriormente es completa debido a que ya no hay individuos que califiquen para los trabajos restantes.

Sin embargo, lo que se hizo con la matriz Q puede mejorarse haciendo una transferencia de trabajos. Por ejemplo, al mover el individuo 1 al trabajo 1 y el individuo 2 al trabajo 3 se obtiene una asignación temporalmente *incompleta*, pero a partir de esta se puede hacer la última relación, la del individuo 3 o 4 al trabajo 4 para completar la asignación. Se puede decir que esta forma es la óptima ya que otras transferencias de trabajos solo involucrarían al individuo 1 por que solo él puede hacer los trabajos 1 y 2 pero no los dos al mismo tiempo. Transferir al individuo 1 al trabajo 2 también seria una solución óptima al igual que mover el individuo 4 al trabajo 4.

$$Q = \begin{bmatrix} 1* & 1 & 1 & 0 \\ 0 & 0 & 1* & 1 \\ 0 & 0 & 0 & 1* \\ 0 & 0 & 0 & 1 \end{bmatrix}, Q = \begin{bmatrix} 1 & 1* & 1 & 0 \\ 0 & 0 & 1* & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1* \end{bmatrix},$$

$$Q = \begin{bmatrix} 1 & 1* & 1 & 0 \\ 0 & 0 & 1* & 1 \\ 0 & 0 & 0 & 1* \\ 0 & 0 & 0 & 1 \end{bmatrix}, Q = \begin{bmatrix} 1* & 1 & 1 & 0 \\ 0 & 0 & 1* & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1* \end{bmatrix}$$

Estas matrices representan algunas de las posibles asignaciones óptimas para el ejemplo presentado.

4.1.2. El método Húngaro

El nombre de este algoritmo fue dado por H. W. Kuhn en reconocimiento al trabajo de dos matemáticos húngaros. Kuhn plantea el problema de correspondencia como un problema de asignación, siendo la meta del algoritmo encontrar la mejor asignación de un conjunto de personas o trabajadores a un conjunto de trabajos, Kuhn[21].

El algoritmo modela un problema de asignación como una matriz de costos de dimensión nxm, donde cada elemento representa el costo de asignar el enésimo trabajador al emésimo trabajo.

Supongamos que hay n individuos $(i = 1, ..., n)$ que están disponibles para n trabajos $(j = 1, ..., n)$ y que ahora, los elementos de la matriz que relacionan a los individuos con los trabajos, representan el *costo* de asignar un individuo a un trabajo. En otras palabras, el elemento $r_{i,j}$ de la matriz R representa el monto que le cuesta al individuo i hacer el trabajo j.

Ahora la asignación de los trabajos no es tan fácil como lo que se mostró en la sección anterior. A continuación se muestran los pasos del algoritmo Húngaro para realizar la asignación óptima de los individuos a los trabajos a través de un ejemplo.

Dada la matriz de costos R, donde cada elemento $a_{i,j}$ representa el tiempo

en horas que le cuesta a cada individuo i realizar el trabajo j, encontrar la asignación de trabajos óptima mediante el algoritmo Húngaro que minimice el tiempo que tardarán todos los individuos en realizar los trabajos.

$$R = \begin{bmatrix} 14 & 5 & 8 & 7 \\ 2 & 12 & 6 & 5 \\ 7 & 8 & 3 & 9 \\ 2 & 4 & 6 & 10 \end{bmatrix}$$

Figura 4.3. *Matriz R, ejemplo para resolver mediante el algoritmo Húngaro*

1. **Minimizar la matriz**:

   - Encontrar el valor f_i más pequeño de cada renglón y restarlo a todos los valores del reglón.

$$\begin{matrix} f_0 = 5 \\ f_1 = 2 \\ f_2 = 3 \\ f_3 = 2 \end{matrix} \quad R' = \begin{bmatrix} 9 & 0 & 3 & 2 \\ 0 & 10 & 4 & 3 \\ 4 & 5 & 0 & 6 \\ 0 & 2 & 4 & 8 \end{bmatrix}$$

Figura 4.4. *Paso* 1*, resta de valores más pequeños por fila*

   - Encontrar el valor c_j más pequeño de cada columna y restarlo a todos los elementos de la columna de la nueva matriz.

$$\begin{array}{l} c_0 = 0 \\ c_1 = 0 \\ c_2 = 0 \\ c_3 = 2 \end{array} \quad R' = \begin{bmatrix} 9 & 0 & 3 & 0 \\ 0 & 10 & 4 & 1 \\ 4 & 5 & 0 & 4 \\ 0 & 2 & 4 & 6 \end{bmatrix}$$

Figura 4.5. *Paso* 1*, resta de valores más pequeños por columna*

Esto hará que se tenga al menos un cero en cada renglón y/o columna. Cada cero representa, dependiendo de la posición del cero en la matriz (i, j), que el individuo i puede ser asignado al trabajo j. A esta última matriz se le conoce como *matriz de costos reducida*

2. **Se marcan con líneas las filas y/o columnas** que tengan ceros, haciendo el menor número de líneas posibles. Todos los ceros deben estar marcados. Si el número de líneas es igual al número de filas entonces se salta al paso 5, si no, se continua al siguiente paso.

$$R' = \left[\begin{array}{cccc} \cancel{9} & \cancel{0} & \cancel{3} & \cancel{0} \\ 0 & 10 & 4 & 1 \\ \cancel{4} & \cancel{5} & \cancel{0} & \cancel{4} \\ 0 & 2 & 4 & 6 \end{array}\right]$$

Figura 4.6. *Paso* 2 *del algoritmo húngaro, debido a que el número de líneas no es igual al número de filas se continúa.*

3. **Encontrar el elemento menor** de las filas y columnas que no están

marcadas, y restarlo a los demás elementos de la misma fila/columna. Si un elemento está cubierto por dos líneas en vez de restar el elemento menor encontrado, se suma.

$$R' = \begin{bmatrix} 10 & 0 & 3 & 0 \\ 0 & 9 & 3 & 0 \\ 5 & 5 & 0 & 4 \\ 0 & 1 & 3 & 5 \end{bmatrix}$$

Figura 4.7. *Paso* 3*, el número menor de los no marcados es 1, fila 2, columna 4.*

4. **Se vuelven a marcar las líneas** en las filas/columnas que tengan ceros y se verifica si el número de líneas es igual al número de renglones de la matriz, si no es igual se vuelve al paso 3, en caso contrario se continua.

$$R' = \begin{bmatrix} \cancel{10} & 0 & 3 & \cancel{0} \\ 0 & 9 & 3 & 0 \\ \cancel{5} & \cancel{5} & 0 & \cancel{4} \\ 0 & 1 & 3 & 5 \end{bmatrix}$$

Figura 4.8. *Debido a que el número de líneas es igual al número de filas el algoritmo avanza al paso* 5*.*

5. **Se hacen las asignaciones** de los individuos a los trabajos eligiendo un solo un cero por columna o por renglón. Teniendo cuidado que ningún renglón y ninguna columna se repitan al elegir los ceros.

$$R' = \begin{bmatrix} 10 & \boxed{0} & 3 & 0 \\ 0 & 9 & 3 & \boxed{0} \\ 5 & 5 & \boxed{0} & 4 \\ \boxed{0} & 1 & 3 & 5 \end{bmatrix}$$

Figura 4.9. *Asignaciones correspondientes de los individuos a los trabajos.*

Y para verificar que los costos son los menores posibles se hace observando los costos n de la matriz original, únicamente conservando las posiciones otorgadas por el algoritmo húngaro. Se observa en la Figura 4.10, que ha sido una asignación óptima ya que la suma de los costos es la menor que se pudo haber obtenido de todas las combinaciones posibles tal y como menciona Kuhn[21].

$$R = \begin{bmatrix} 14 & \boxed{5} & 8 & 7 \\ 2 & 12 & 6 & \boxed{5} \\ 7 & 8 & \boxed{3} & 9 \\ \boxed{2} & 4 & 6 & 10 \end{bmatrix}$$

Figura 4.10. *Verificación de los costos como asignación óptima.*

4.1.3. Implementación

Para el caso de nuestro trabajo la analogía cambia un poco, ahora en vez de ser individuos serán los cuadros que se tienen de las detecciones pasadas y en vez de ser trabajos se tratará de los cuadros más recientemente detectados.

El costo que se definió como la relación entre estos cuadros es la intersección que existe entre cada uno de ellos, debido a que los cuadros que tengan una mayor intersección son los que cuentan con mayor probabilidad de pertenecer a la misma persona. De aquí surge un problema ya que el algoritmo trata de encontrar el menor costo, que para este caso representa la menor intersección entre los cuadros y lo que en realidad se busca es encontrar los cuadros con los cuales se tenga el área más grande de intersección. Para resolver este problema sólo se representó como el inverso de la intersección. Es decir:

$$costo = \frac{1}{area\_de\_interseccion} \tag{4.1}$$

Esta fue la única consideración que se hizo, en todo lo demás se siguió el algoritmo tal cual se describió.

4.2. Filtro de Kalman

El filtro de Kalman es un conjunto de ecuaciones matemáticas que proveen de un método computacional para estimar el estado de un proceso, de tal forma que se minimice el error Welch[19].

Permite hacer una inferencia en un sistema dinámico lineal, que es un

modelo Bayesiano similar al modelo oculto de Markov pero que el espacio de estados de las variables latentes es continuo y donde todas las variables, ya sean latentes u observadas tienen distribución Gaussiana, Faragher[22].

El trabajo más famoso que se hizo cuando el filtro de Kalman fue recientemente descubierto fue el que se usó en la computadora de navegación del Apollo que llevo a Neil Armstrong a la luna y que lo trajo de regreso. Hoy en día en cada dispositivo de navegación de los satélites, en cada smartphone, y en muchos videojuegos se utiliza el filtro de Kalman, Faragher[22]

El problema general del filtro de Kalman es tratar de estimar un estado futuro $x \in \Re^n$ de un proceso controlado en tiempo discreto con la siguiente representación:

Ecuación de estado:

$$x_k = Fx_{k-1} + Bu_k + w_k \tag{4.2}$$

Donde x_k es el vector que contiene los componentes del estado en el momento actual (por ejemplo: posición, velocidad, etc.) y F es la matriz de nxn que relaciona el estado en el tiempo anterior $(k-1)$, con el estado actual k, mejor conocida como *Matriz de transferencia* o *Matriz de transición.*

Mediante el parámetro u_k se permite tener un control externo en el sistema y consiste en un vector de dimensión c que tiene como componentes las entradas de control tales como ángulo de dirección, ajustes de aceleración, fuerza de frenado, etc. B es una matriz de nxc que relaciona estas entradas de control con el cambio de estado.

La variable w_k es una variable aleatoria que usualmente es llamada *ruido*

del proceso y que está asociada con los eventos aleatorios que afectan directamente el estado actual del sistema.

Esta ecuación se encarga de hacer una proyección del estado anterior, tratando de hacer una predicción del estado actual en el que se encuentra el sistema. También se utilizan algunos otros factores que pueden afectar al sistema, que no dependen del estado anterior del sistema tales como el ruido y las variables de control en u_k que nos sirven en casos en el que se hace un cambio directo sobre el sistema y que éste puede ser conocido (por ejemplo algún cambio en la aceleración).

Mediante una serie de mediciones, el sistema puede aproximarse al estado actual de un entorno. Estas mediciones se denotan mediante la variable z_k. Las mediciones pueden ser representadas mediante la siguiente expresión:

$$z_k = H_k x_k + v_k \tag{4.3}$$

Donde H_k es la matriz de transformación que mapea los parámetros del vector de estado al dominio de la medición y v_k es el error de la medición.

Para entender mejor este algoritmo, se propone el siguiente ejemplo. Consideraremos un tren que recorre una vía y que su vector de estado está definido por su posición y su velocidad:

$$x_k = \begin{bmatrix} a_k \\ b_k \end{bmatrix} \tag{4.4}$$

Donde a_k es la posición del tren en la vía y b_k es su velocidad.

El conductor del tren en algún momento tendrá que acelerar o frenar el tren, lo cual, se toma como la entrada de control para el sistema. Esta entrada se considerará como una función de fuerza aplicada f_k sobre la masa del tren m. Esta información es almacenada en el vector u_k:

$$u_k = \frac{f_k}{m} \tag{4.5}$$

La relación entre la fuerza aplicada vía el freno o el acelerador durante el periodo Δk, que es igual al tiempo que pasa desde $k-1$ hasta k; la posición y la velocidad del tren están dadas por las siguientes ecuaciones:

$$a_k = a_{k-1} + (b_{k-1}\Delta k) + \frac{f_k(\Delta k)^2}{2m} \tag{4.6}$$

$$b_k = b_{k-1} + \frac{f_k \Delta k}{m} \tag{4.7}$$

Estas ecuaciones pueden representarse en forma de matrices de la siguiente manera:

$$\begin{bmatrix} a_k \\ b_k \end{bmatrix} = \begin{bmatrix} 1 & \Delta k \\ 0 & 1 \end{bmatrix} \begin{bmatrix} a_{k-1} \\ b_{k-1} \end{bmatrix} + \begin{bmatrix} \frac{(\Delta k)^2}{2} \\ \Delta k \end{bmatrix} \frac{f_k}{m} \tag{4.8}$$

Por comparación definimos la matriz de transición y la matriz de control como:

$$F = \begin{bmatrix} 1 & \Delta k \\ 0 & 1 \end{bmatrix} \qquad B = \begin{bmatrix} \frac{(\Delta k)^2}{2} \\ \Delta k \end{bmatrix} \tag{4.9}$$

El estado verdadero de x_k no puede ser directamente observado, pero el filtro de Kalman nos permite estimarlo combinando los modelos del sistema con las mediciones ruidosas de ciertos parámetros o funciones lineales de los parámetros. Por lo tanto, las estimaciones de interés del vector de estado estarán definidos por funciones de densidad de probabilidad, en vez de ser valores discretos. El filtro de Kalman está basado en funciones de densidad de probabilidad Gaussianas. Para que éstas sean descritas necesitamos conocer sus varianzas y covarianzas, las cuales se almacenan en la *matriz de covarianza* P. Los valores que se encuentran en la diagonal principal de la matriz P son las varianzas asociadas con los parámetros del vector de estado y los valores que no están en la diagonal principal son las covarianzas entre los elementos del vector de estado. La matriz P se define como:

$$P_k^- = E[(x_k - x_k^-)(x_k - x_k^-)^T] \tag{4.10}$$

4.2.1. Fases del filtro de Kalman

El filtro de Kalman involucra dos fases: La fase de predicción, donde a partir de los estados anteriores se hace una proyección para estimar el siguiente estado y la fase de corrección, que nos da como resultado una corrección de esta estimación, haciendo una mezcla entre lo que se predijo que pasaría en la fase de predicción y las mediciones obtenidas.

Entonces, las ecuaciones que definen la fase de predicción son las siguientes:

$$x_k^- = Fx_{k-1} + Bu_{k-1} + w_k \tag{4.11}$$

$$P_k^- = FP_{k-1}F^T + Q_{k-1} \tag{4.12}$$

Donde Q_{k-1} es la matriz de covarianza del ruido del proceso, asociada con las entradas ruidosas de control.

Para la fase de corrección la ganancia de Kalman nos permite calcular los valores actualizados del vector de estado x_k y de la matriz P_k cuando las nuevas mediciones están disponibles. Las ecuaciones para la actualización son:

$$x_k = x_k^- + K_k(z_k - H_k x_k^-) \tag{4.13}$$

$$P_k = (I - K_k H_k)P_k^- \tag{4.14}$$

Donde K es una matriz de nxm y nos dice qué tanto peso debemos darle a la diferencia entre las mediciones y el estado que se predijo, es decir, el error que hay entre estas dos. A esta matriz se le llama *ganancia de Kalman* o *factor de mezcla*. Y puede ser definida de la siguiente forma:

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + R)^{-1} = \frac{P_k^- H_k^T}{(H_k P_k^- H_k^T + R)} \tag{4.15}$$

Donde R es la matriz de covarianza del ruido de las mediciones. Como podemos observar, si la matriz de covarianza del ruido se aproxima a cero, entonces el valor de K será muy grande y le dará un mayor peso a la diferencia entre las mediciones y el estado predicho. Por otra parte, si el valor de P_k^-

se aproxima a cero, la ganancia K tendrá un valor pequeño y por lo tanto la diferencia entre las mediciones y el estado predicho se tomará menos en cuenta. Específicamente sería:

$$\lim_{R \to 0} K_k = H^{-1} \tag{4.16}$$

$$\lim_{P_k^- \to 0} K_k = 0 \tag{4.17}$$

Otra forma de decir cómo es que la ganancia K afecta a la corrección, es pensando que mientras la covarianza del error en la medición R se aproxima a cero, la medición actual z_k es más y más *confiable*, mientras que la medición en el estado predicho Hx_k^- es cada vez menos *confiable*. Por otra parte, mientras la covarianza en el error estimada P_k^- se aproxima a cero, la medición actual z_k es cada vez menos *confiable* y la medición predicha Hx_k^- es cada vez más y más *confiable*.

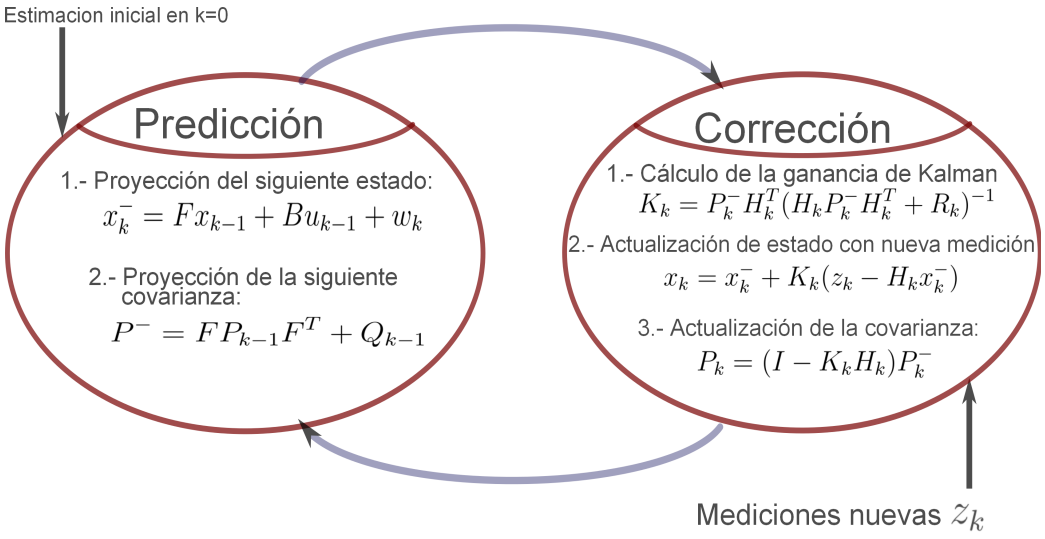

Figura 4.11. *Proceso iterativo del filtro de Kalman.*

Retomando el ejemplo del tren, se considerará solo una dimensión para simplificar la explicación del método, se tratará de hacer la estimación de la posición del tren (o más precisamente, la antena que se encuentra en el techo del tren). La información disponible es:

- Predicciones basadas en la última posición y velocidad conocidas del tren.
- Mediciones que se obtienen de la antena.

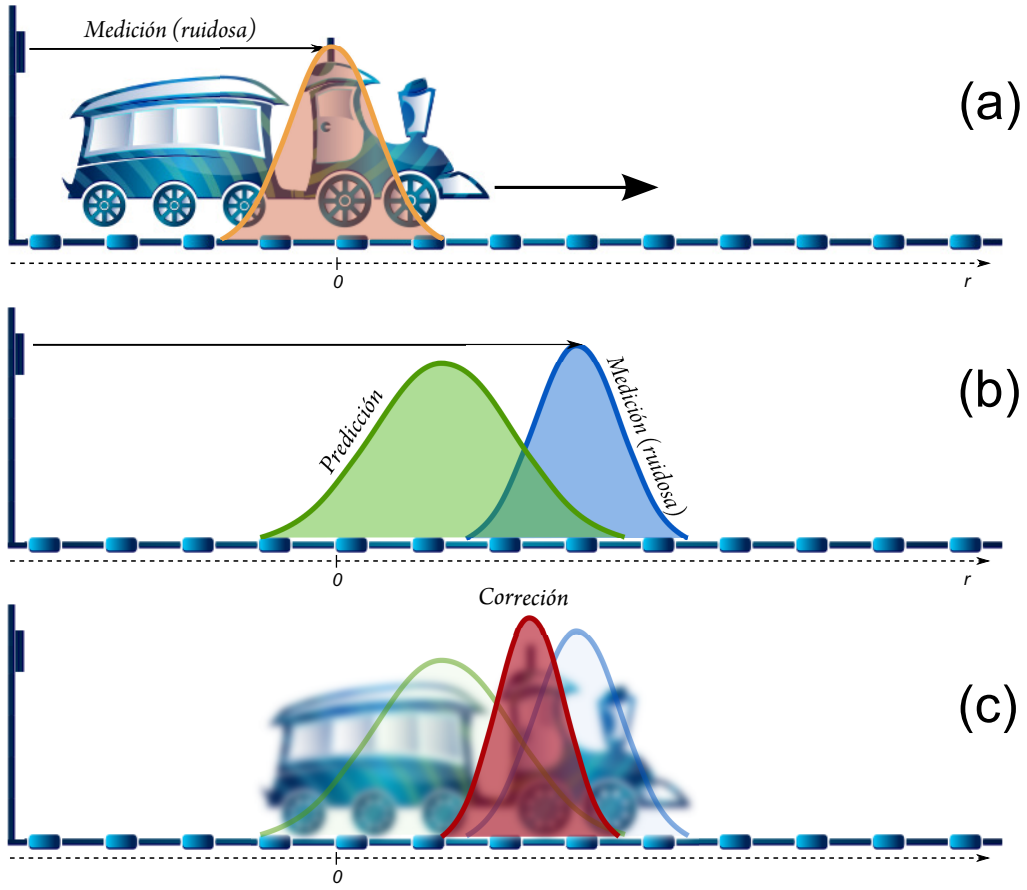

Figura 4.12. *Ejemplo del filtro de Kalman basado en la posición de un tren (Con datos de Faragher[22], elaboración propia).*

Se puede hacer la combinación de esta información haciendo uso del filtro de Kalman para obtener la mejor estimación posible de la localización del tren.

El estado inicial del sistema (en el tiempo $k = 0$) representado en la Figura 4.12(a) es conocido debido a una precisión razonable y está dado por una distribución de probabilidad Gaussiana. En el siguiente instante de tiempo,

$k = 1$, podemos estimar la nueva posición del tren, basada en las limitaciones conocidas tales como la velocidad y posición en el tiempo $k = 0$, la aceleración o la desaceleración.

En la práctica podemos tener conocimiento sobre las variables de control, tales como el freno o el acelerador que usa el conductor del tren que modifican esta aceleración o desaceleración. En cualquier caso, tenemos una predicción de la nueva posición del tren, representada en la Figura 4.12(b) por una nueva distribución de probabilidad Gaussiana con valores de media y varianza nuevos, matemáticamente representada por la ecuación 4.11. La varianza ha aumentado, representando una reducción en la certeza de nuestra estimación de la posición, comparado con $k = 0$, debido a la incertidumbre asociada al ruido del proceso que viene por las aceleraciones y/o desaceleraciones entre los tiempos $k = 0$ y $k = 1$.

En $k = 1$, también se hace una medición para saber dónde se encuentra el tren usando su sistema de posicionamiento y esta medición está representada por la distribución de probabilidad en la Figura 4.12(b). La mejor estimación que se puede hacer sobre la posición del tren es combinando nuestro conocimiento de la predicción y de la medición.

Ésto se logra multiplicando estas dos distribuciones, como esta representado en la Figura 4.12(c) en la distribución de en medio.

Una propiedad clave de las funciones Gaussianas es que el producto de dos nos da como resultado otra función Gaussiana, ésto permite que se puedan

multiplicar funciones Gaussianas a través del tiempo sin hacer que el algoritmo se vuelva más complejo o incremente el número de términos, después de cada instante de tiempo, una nueva distribución es totalmente representada por una función Gaussiana. Ésta es la clave para las propiedades recursivas elegantes del filtro de Kalman.

La predicción que está representada en la Figura 4.12(b) está dada por la ecuación:

$$y_1(r;\mu_1,\sigma_1) = \frac{1}{\sqrt{2\pi\sigma_1^2}} e^{-\frac{(r-\mu_1)^2}{2\sigma_1^2}} \tag{4.18}$$

y la distribución de la medición representada en la Figura 4.12(b) por:

$$y_2(r;\mu_2,\sigma_2) = \frac{1}{\sqrt{2\pi\sigma_2^2}} e^{-\frac{(r-\mu_2)^2}{2\sigma_2^2}} \tag{4.19}$$

La información que estas dos distribuciones proveen es fusionada al multiplicarlas. Como ejemplo se hace la representación de la distribución en la Figura 4.12(c), la cual es la mejor estimación de la posición en ese periodo de tiempo. Esta fusión puede ser representada como:

$$y_{fus}(r;\mu_1,\sigma_1,\mu_2,\sigma_2) = \left(\frac{1}{\sqrt{2\pi\sigma_1^2}} e^{-\frac{(r-\mu_1)^2}{2\sigma_1^2}}\right)\left(\frac{1}{\sqrt{2\pi\sigma_2^2}} e^{-\frac{(r-\mu_2)^2}{2\sigma_2^2}}\right) \tag{4.20}$$

$$y_{fus}(r;\mu_1,\sigma_1,\mu_2,\sigma_2) = \frac{1}{2\pi\sqrt{\sigma_1^2\sigma_2^2}} e^{-\left(\frac{(r-\mu_1)^2}{2\sigma_1^2}+\frac{(r-\mu_2)^2}{2\sigma_2^2}\right)} \tag{4.21}$$

Los términos cuadráticos en esta nueva función pueden expandirse para

reescribirse en la forma de una función Gaussiana:

$$y_{fus}(r;\mu_{fus},\sigma_{fus}) = \frac{1}{\sqrt{2\pi\sigma_{fus}^2}} e^{-\frac{(r-\mu_{fus})^2}{2\sigma_{fus}^2}} \tag{4.22}$$

donde:

$$\mu_{fus} = \frac{\mu_1\sigma_2^2 + \mu_2\sigma_1^2}{\sigma_1^2 + \sigma_2^2} \tag{4.23}$$

$$\mu_{fus} = \mu_1 + \frac{\sigma_1^2(\mu_2 - \mu_1)}{\sigma_1^2 + \sigma_2^2} \tag{4.24}$$

por otra parte:

$$\sigma_{fus}^2 = \frac{\sigma_1^2\sigma_2^2}{\sigma_1^2 + \sigma_2^2} \tag{4.25}$$

$$\sigma_{fus}^2 = \sigma_1^2 - \frac{\sigma_1^4}{\sigma_1^2 + \sigma_2^2} \tag{4.26}$$

Estas dos últimas ecuaciones (4.24 y 4.26) representan los pasos de actualización de la medición del algoritmo del filtro de Kalman y podemos hacer ciertas comparaciones con los parámetros de las ecuaciones que hemos visto antes, recordando las ecuaciones con las que se hace la predicción:

$$x_k = x_k^- + K_k(z_k - H_k x_k^-) \tag{4.27}$$

$$P_k = P_{k-1} - K_k H_k + P_{k-1} \tag{4.28}$$

Si reescribimos las ecuaciones 4.24 y 4.26 para que sean similares:

$$\mu_{fus} = \mu_1 + K(\mu_2 - H\mu_1) \tag{4.29}$$

$$\sigma_{fus}^2 = \sigma_1^2 - KH\sigma_1^2 \tag{4.30}$$

Comparando estas ecuaciones resulta que podemos encontrar equivalencias entre los parámetros descritos anteriormente como:

- $\mu_{fus} \rightarrow x_k$: es el vector estado derivado de la fusión de datos.
- $\mu_1 \rightarrow x_{k-1}$: es el vector estado antes de la fusión, este puede ser la predicción.
- $\sigma_{fus}^2 \rightarrow P_k$: la matriz de covarianza, después de fusionar los datos.
- $\sigma_1^2 \rightarrow P_{k-1}$: la matriz de covarianza antes de la fusión.
- $\mu_2 \rightarrow z_k$: el vector de mediciones.
- $\sigma_2^2 \rightarrow R$: la matriz de covarianza del ruido de las mediciones.
- $H \rightarrow H_k$: La matriz de transformación que mapea los parámetros del vector estado al dominio de las mediciones y que para este caso:

$$H = 1 \tag{4.31}$$

- $K \rightarrow K_k$: La ganancia de Kalman que en este caso es:

$$K = \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2} \tag{4.32}$$

Con todos estos parámetros definidos tenemos completas las ecuaciones para empezar el algoritmo y hacer las predicciones y correcciones pertinentes en cada instante de tiempo, para poder tener estimaciones minimizando el ruido de la posición del tren.

4.2.2. Consideraciones

Para el caso de nuestro sistema, se hizo que la estimación fuera de la posición y el tamaño de los cuadros que habían sido detectados. El vector de estado que se utilizó para el sistema es:

$$x_k = \begin{bmatrix} a_k \\ b_k \\ w_k \\ h_k \\ \dot{a_k} \\ \dot{b_k} \\ \dot{w_k} \\ \dot{h_k} \end{bmatrix} \tag{4.33}$$

Donde a_k es la posición del punto superior derecho del cuadro en el eje x, b_k es la posición del mismo punto en el eje y, w_k el ancho del cuadro y h_k el largo a partir de este punto. $\dot{a_k}$, $\dot{b_k}$, $\dot{w_k}$, $\dot{h_k}$ representan las respectivas variaciones de velocidad de cada uno de los parámetros anteriores.

Las definiciones para los parámetros son:

$$a_k = a_{k-1} + \dot{a_{k-1}}\delta k \tag{4.34}$$

$$b_k = b_{k-1} + \dot{b_{k-1}}\delta k \tag{4.35}$$

$$w_k = w_{k-1} + \dot{w_{k-1}}\delta k \tag{4.36}$$

$$h_k = h_{k-1} + \dot{h_{k-1}}\delta k \tag{4.37}$$

Por lo tanto la matriz de transición es:

$$F = \begin{bmatrix} 1 & 0 & 0 & 0 & \delta k & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & \delta k & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & \delta k & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & \delta k \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.38}$$

Para este caso no se considerarán entradas de control, ya que la aceleración del robot no se puede saber, a menos que haya comunicación con otros módulos del robot, por lo tanto, solo se considerarán las velocidades de los parámetros anteriores, ya que éstas se pueden conocer en base a las posiciones anteriores.

4.2.3. Algoritmo completo de rastreo

Para describir el algoritmo completo, primero se presenta el pseudocódigo del algoritmo Húngaro, después el del filtro de Kalman para que al final se entienda cómo estos dos trabajan en conjunto.

Para que se tuviera un control independiente de cada una de las personas que están siendo rastreadas en la escena, se hizo una clase para crear objetos que contengan la información del rastreo para solo una persona. Esta clase es llamada *Track* y contiene información como: localización actual de la persona en la escena representada por un cuadro, un identificador numérico y un

contador que nos dice el número de veces consecutivas que el objeto ha sido refrescado, es decir, el número de frames consecutivos que la persona ha sido detectada de nuevo en la escena. Este control se puede ver en la Figura 4.1 donde se muestra la arquitectura del rastreo, el módulo de control se encarga de crear, eliminar y modificar los objetos track.

A continuación, en la Figura 4.13, se da a conocer el pseudocódigo del algoritmo Húngaro donde ya se mencionan los objetos track descritos anteriormente:

Algorithm 2 Algoritmo Húngaro

Require: Vector de cuadros 1: $v1$, vector de cuadros track: *tracks*
Ensure: Vector de enteros que nos dice la relacion entre los vectores de entrada.

```
1:  i = j = 0
2:  for item en v1 do
3:      for track en tracks do
4:          if area_interseccion(item, track) > 0 then
5:              matriz_costos[i][j] = 1/area_interseccion(item, track)
6:          else
7:              matriz_costos[i][j] = 1
8:          end if
9:          j++
10:     end for
11:     i++
12: end for
13: for cada renglon en matriz_costos do
14:     menor = min(renglon)
15:     restar menor a cada elemento de renglon
16: end for
17: for cada columna en matriz_costos do
18:     menor = min(columna)
19:     restar menor a cada elemento de columna
20: end for
21: Marcar con lineas las filas y columnas que tengan ceros
22: while (número de lineas) != (número de filas) do
23:     Encontrar menor de todos los elementos no marcados en matriz_costos
24:     Restar menor a los elementos no marcados
25:     if un elemento esta cubierto 2 veces then
26:         elemento+ = menor
27:     end if
28:     Marcar con lineas las filas y columnas que tengan ceros
29: end while
30: for i = 0 hasta i <= columnas en matriz_costos do
31:     bandera = 0
32:     for j = 0 hasta i <= filas en matriz_costos do
33:         if matriz_costos[i][j] == 0 then
34:             asignaciones[i] = j
35:             bandera = 1
36:         end if
37:         if bandera! = 1 then
38:             asignaciones[i] = −1
39:         end if
40:     end for
41: end for
42: return asignaciones
```

Figura 4.13. *Pseudocódigo del algoritmo Húngaro*

El vector de salida de este algoritmo se explica en la Figura 4.14, donde $v1$ y $v2$ son los vectores que contienen cuadros de las detecciones los cuales son sometidos al algoritmo Húngaro, las flechas representan las asignaciones correspondientes de los cuadros y los cuadros grises que se encuentran arriba del vector representan los índices de sus elementos.

El vector de salida de cada caso siempre será del mismo tamaño que el vector $v1$ de entrada, ésto para tener una referencia de lo que se pudo asignar respecto a este vector.

En el índice 0 del vector de salida se encuentra un 1, indicando que el cuadro que se encuentra en el vector $v1$ en el índice 0 corresponde al cuadro del vector $v2$ que se encuentra en el índice 1, así como lo indica la flecha que une los cuadros y que representa esta asignación.

En el ejemplo de la derecha los vectores son de diferentes tamaños lo que quiere decir que se encontraron personas nuevas en la escena, cuando esto sucede, únicamente se coloca un -1 en el vector de salida, indicando que no se tiene una correspondencia con algún cuadro.

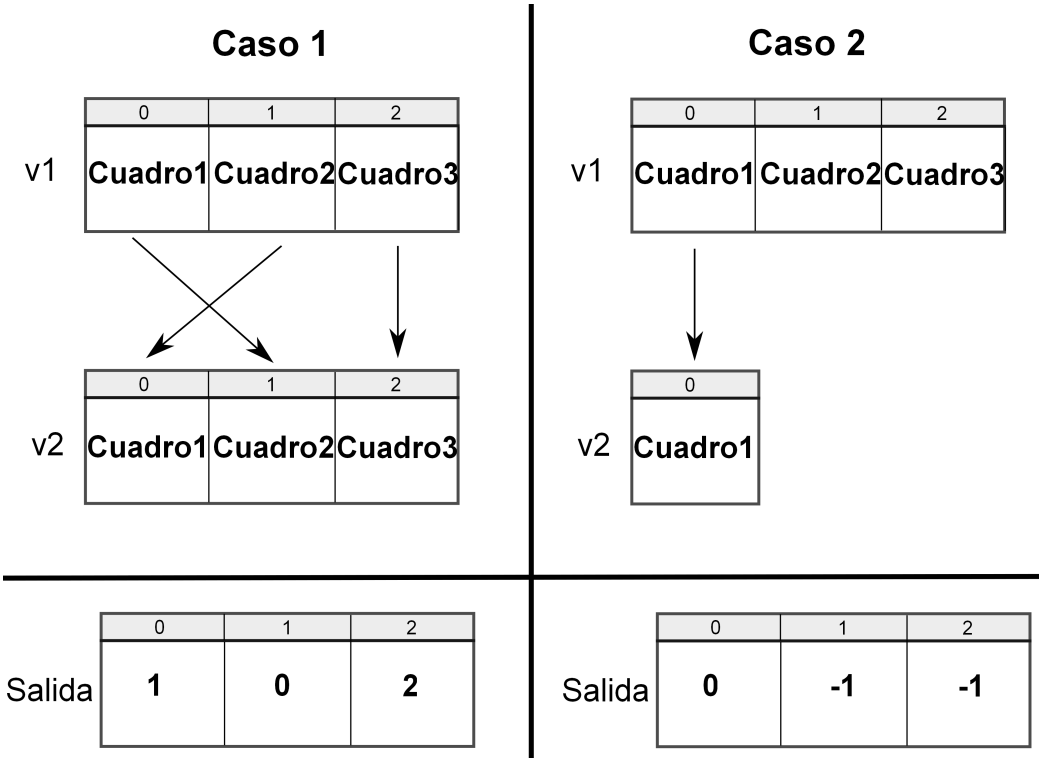

Figura 4.14. *Entradas y salidas del algoritmo Húngaro.*

Algorithm 3 Clase Track (Filtro de Kalman)

```
1:  limite_inferior
2:  limite_superior
3:  modificado = 0
4:  procedure Track(item cuadro_inicial item lim_inferior item lim_superior)
5:      estado_actual = cuadro_inicial
6:      limite_inferior = lim_inferior
7:      limite_superior = lim_superior
8:      tiempo_de_vida = 1
9:  end procedure
10: procedure Corregir(void)
11:     Calcular la ganancia de Kalman
12:     Actualizar el estado actual con la medición
13:     Actualizar el valor de la covarianza
14: end procedure
15: procedure Predecir
16:     Proyectar siguiente estado en base al actual
17:     Proyectar siguiente covarianza
18: end procedure
19: procedure Refrescar
20:     Predecir()
21:     Corregir()
22:     modificado = 1
23:     tiempo_de_vida = tiempo_de_vida + 1
24: end procedure
```

Figura 4.15. *Pseudocódigo de la definición de clase Track*

Algorithm 4 Algoritmo Completo de Rastreo

Require: Vector de detecciones: v, Imagen capturada por la cámara: I, Limite inferior: $lim\_inferior$ y Limite superior $lim\_inferior$
Ensure: Vector de enteros que nos dice la relacion entre los vectores de entrada.

```
1:  Inicializar vector tracks vacío
2:  while I sea valida do
3:      Llamar al Algoritmo Húngaro con v y cuadros de tracks
4:      Inicializar vector: asignaciones = AlgoritmoHungaro(v, tracks)
5:      for i = 0 hasta tamanio(asignaciones) do
6:          if asignaciones[i] = −1 then
7:              Crear nuevo Track con cuadro inicial v[i], lim_inferior, lim_superior
8:              Agregar Track al vector tracks
9:          else
10:             Refrescar track: track.Refescar(v[i])
11:         end if
12:     end for
13:     for i = 0 hasta tamaño(tracks) do
14:         if (tracks[i].tiempo_de_vida >= lim_inferior) then
15:             Dibujar en la imagen I el cuadro en tracks[i]
16:         end if
17:         if tracks[i].modificado = FALSO then
18:             Reduce tiempo de vida de tracks[i]
19:             if tracks[i].tiempo_de_vida = 0 then
20:                 Elimina tracks[i]
21:                 i − −
22:             end if
23:         else
24:             Aumenta tiempo de vida tracks[i]
25:         end if
26:         tracks[i].modificado = 0
27:     end for
28:     Actualizar vector v y la imagen I
29: end while
```

Figura 4.16. *Pseudocódigo del algoritmo completo del rastreo*

Para el caso del algoritmo que define el Filtro de Kalman, los parámetros $limite\_inferior$ y $limite\_superior$ se establecieron como el mínimo y máximo número de detecciones consecutivas respectivamente. El $limite\_inferior$ nos sirve para determinar el valor mínimo de estas detecciones que se requieren para que se empiecen a dibujar los cuadros en la imagen.

Ésto sirve para que aquellas detecciones que aparecen por debajo de este límite no sean mostradas, con ésto se disminuye el número de falsos positivos

mostrados en la imagen. De igual manera el parámetro *limite_superior* nos sirve para establecer un límite de crecimiento en el tiempo de vida de cada *track* dado por el parámetro *tiempo_de_vida*. Este último parámetro va aumentando dependiendo del número de detecciones consecutivas determinadas por el algoritmo Húngaro, por ejemplo, si el algoritmo Húngaro nos dice que el *cuadro_inicial* de un *track* coincide con un cuadro de la siguiente imagen, el tiempo de vida del *track* en cuestión aumenta en uno, quedando con $tiempo\_de\_vida = 2$, diciéndonos que la detección ya ha estado en 2 imágenes consecutivas. Por el contrario, si el algoritmo no nos regresa una coincidencia para el *track* que, supongamos, tiene $tiempo\_de\_vida = 2$, entonces este parámetro se reducirá en uno. Este valor llamado *tiempo_de_vida* también nos sirve para mantener los *tracks* que probablemente están siguiendo a una persona correctamente.

Para tener un control sobre el seguimiento de las personas de un ambiente, se usa el algoritmo completo de rastreo. Este control se lleva cabo mediante los parámetros que son arrojados por las implementaciones del algoritmo Húngaro y del filtro de Kalman.

Primero, se empieza por el vector de asignaciones que envía el algoritmo Húngaro, si en este vector existen cuadros que no pudieron ser asignados a algún *track* se crea un nuevo *track*, para que esta probable nueva persona detectada pueda ser rastreada en el ambiente. De lo contrario, si el cuadro puede ser asignado a un *track* existente, este se *refresca* con el cuadro que le ha asignado el algoritmo Húngaro. El parámetro del *track* "modificadoçambia

al valor de 1 si es que este fue refrescadoz ésto quiere decir que se enviará éste nuevo estado o cuadro de detección actual al filtro de Kalman para que se haga una estimación de la posición de la persona en el entorno.

El parámetro "modificado"nos indica si el *track* en cuestión fue utilizado para ese conjunto de detecciones actual, si éste no fue utilizado, entonces se disminuye su *tiempo_de_vida*, haciendo que mientras menos veces sea utilizado tenga más probabilidad de ser eliminado, indicando que la persona que había sido detectada con anterioridad ha salido del entorno o que ya no puede ser detectada.

El algoritmo completo de rastreo nos provee de un control sobre todos los *tracks* que se crean para hacer el seguimiento de las personas que se encuentran en una serie de imágenes. Para ello, este algoritmo hace uso del filtro de Kalman definido como una clase, a partir de la cual se crean los objetos del arreglo *tracks*. Para encontrar la relacion entre los cuadros de estos objetos $Track$ y las detecciones actuales se hace uso del algoritmo Húngaro. Estos tres algoritmos en conjunto más la detección de personas forman el ***Sistema de Detección y Rastreo***.

Capítulo 5

Experimentos y resultados

Este capítulo presenta los experimentos y resultados usados para medir el desempeño de nuestro sistema. En estos experimentos se probaron diferentes parámetros, tanto en la fase de detección como en la fase de rastreo, que son clave para modificar el comportamiento del sistema. Para la fase de detección se evalúa:

- El efecto que tiene el cambio del tamaño de la ventana de detección.
- La manera en que el cambio de la escala modifica la tasa de detecciones.
- El umbral que determina qué tan flexible es la decisión de la detección.

Para la fase de rastreo las partes que se evalúan son las siguientes:

- Variación en el rango de límites del coeficiente de confianza.
- Cambio en el número de frames en donde se efectúa la detección.

Un aspecto interesante que debe ser resaltado es la diferencia que hay entre la detección de sólo cabezas y entre la de cabezas con hombros. Si bien se había definido que la forma particular que tiene la cabeza con los hombros,

que es en forma de un símbolo omega, podría ser mejor diferenciada entre la mayoría de los objetos cotidianos, podría no ser tan bueno cuando existen personas en una imagen que están de perfil, o que están casi de perfil. El detector de cabezas tendría una ventaja en este tipo de casos, ya que la forma ovalada de la cabeza no cambia demasiado o no en todos los casos. En este capítulo tendremos la comparativa entre los resultados de ambos detectores y se dará a conocer cuál de ellos se encontró con las mejores puntuaciones en las mismas condiciones.

Para iniciar el capítulo se hará una descripción de la creación del corpus que se analizó en nuestro trabajo, de los sitios donde se obtuvieron o de cómo fueron creados, tamaños, resoluciones, entre otras características de las imágenes y los videos recopilados.

5.1. Corpus

El corpus de evaluación que se usó en el trabajo está constituido de imágenes y videos. Éstos fueron tomados de diferentes fuentes, las cuáles se mencionan en cada uno de los apartados siguientes.

5.1.1. Videos

Los videos que se usaron como parte del corpus de nuestro trabajo se tomaron directamente del robot Golem-II+ utilizando una webcam Logitech QuickCam Pro 9000. Ésto con la idea de hacer las pruebas en el ambiente en que el sistema se estará usando. Estos videos son de una resolución de $640x480$ pixeles y duran de 3 a 5 minutos en formato *webm*. Se trató de

hacer que las personas siguieran ciertos movimientos, como caminar alrededor, quedarse parados, etc. para que al evaluar el sistema se conociera su comportamiento. Estos movimientos fueron:

- Quedarse parados, algunos mirando hacia el robot, otros de perfil mientras el robot se mueve hacia adelante, se detiene y después da una vuelta completa en su propio eje.
- Que una de las personas rodee al robot mientras éste sigue girando sobre su propio eje.
- Con el robot quieto, las personas se mueven frente a él tratando de atravesarse unas con otras, con el objetivo de probar qué tan eficiente es el sistema contra obstrucciones de otras personas.
- Con las mismas características del punto anterior, sólo que ahora el robot también se encuentra en movimiento.
- Seguir a una persona mientras camina y que en algunos puntos otras personas se atraviesen en el camino del robot.
- Que el robot entre a una habitación y gire mientras algunas personas se encuentran sentadas y otras paradas muy cerca de él.

Cada uno de estos videos tienen objetivos diferentes para el análisis de la eficiencia del sistema.

5.1.2. Conjuntos de imágenes

Para la colección de imágenes positivas para el entrenamiento, se usaron los conjuntos preestablecidos de *INRIA Person Dataset*[23]. En este trabajo

sólo se usaron las imágenes que se encuentran en el directorio *Train*, tanto las positivas como las negativas. De este conjunto de imágenes, se usaron 614 imágenes positivas de diferentes tamaños, en donde se encuentran personas en diferentes posturas y que predominan las personas que están de pie mirando hacia la cámara, así como 1218 imágenes negativas que también son de diferentes tamaños.

Otro de los conjuntos usados fue el *CBCL PEDESTRIAN DATABASE #1* [24] del *Centro de Aprendizaje Biológico y Computacional* del MIT, que contiene 924 imágenes con medidas fijas de 64 por 128 pixeles en formato PPM. Las imágenes son de personas que se encuentran en la calle, algunas están de frente a la cámara y otras dando la espalda. Estas imágenes fueron agregadas al conjunto de imágenes positivas para el entrenamiento de los detectores.

Por último, se agregó otra selección de imágenes, las cuales fueron tomadas de diversos conjuntos con diferentes características en las posturas, pero lo que más se buscaba era que las personas estuvieran sentadas, ya que la posición de los hombros y la cabeza son un poco diferentes a cuando están de pie. Las selecciones anteriores fueron tomadas de *The Images of Groups Dataset*[25], estas imágenes tienen la principal característica de que son agrupaciones de personas, de diferentes países y diferentes vestimentas, lo cual hace que las imágenes positivas tengan más variedad. Estas imágenes son de diferentes tamaños y cada conjunto es diferente en cuanto a cantidad.

Como resultado se obtuvo una selección de 2000 imágenes como conjunto positivo, y 1218 imágenes como el conjunto negativo para la parte de entrenamiento de los detectores.

Para la parte de evaluación se hicieron otros dos conjuntos, uno que fue una recopilación de imágenes obtenidas de Creative Commons[26] y la otra fueron extracciones de frames de los videos que se obtuvieron del robot para la evaluación del sistema completo.

Las imágenes del primer conjunto tienen una resolución de 800x600 pixeles, donde las personas se encuentran en diferentes poses: de perfil, de espaldas, sentados, recostados, etc. en lugares de fondo muy diferentes, obteniendo 50 imágenes.

Para el otro conjunto, se eligieron las imágenes casi aleatoriamente de los videos con la restricción de que en las imágenes siempre hubiera por lo menos una persona. Como se mencionó anteriormente la resolución es de $640x480$ y se obtuvo un total de 140 imágenes para este conjunto.

Figura 5.1. *Ejemplos de imágenes recopiladas para el entrenamiento de los detectores. Fuente: INRIA[23].*

Figura 5.2. *Ejemplos de imágenes recopiladas para el entrenamiento de los detectores. Fuente: CBCL[24].*

Figura 5.3. *Ejemplos de imágenes recopiladas para el entrenamiento de los detectores. Fuente: AMP[25]*

Figura 5.4. *Ejemplos del conjunto de imágenes extraídas de Internet para la evaluación. Fuente: Creative Commons[26].*

Figura 5.5. *Ejemplos del conjunto de imágenes extraídas de la cámara de Golem-II+ para la evaluación.*

Todas estas imágenes: el conjunto de entrenamiento y el conjunto de evaluación, fueron marcadas por una persona, señalando los lugares donde

identificaba a los individuos en la imagen. Estos cuadros marcados serán tomados como la *verdad*, es decir, como si esos cuadros marcados fueran exactamente el o los lugares donde se encuentran las personas y que el número de estos cuadros son exactamente la cantidad de individuos que existen en la imagen.

<table>
<tr><th colspan="4">Corpus</th></tr>
<tr><th colspan="2">Videos</th><th colspan="2">Imágenes</th></tr>
<tr><th>Descripción</th><th>Resolución</th><th>Descripción</th><th>Cantidad y características</th></tr>
<tr><td>Personas sin moverse, el robot moviéndose</td><td>640x480 pixeles</td><td>De pie, mirando a la cámara, otras de perfil</td><td>614 imágenes positivas</td></tr>
<tr><td>Una persona rodeando al robot, éste le sigue Los otros no se mueven</td><td>640x480 pixeles</td><td>Personas de pie, de espaldas y mirando hacia la cámara</td><td>924 imágenes positivas</td></tr>
<tr><td>El robot no se mueve, las personas se mueven a su alrededor</td><td>640x480 pixeles</td><td>Grupos de personas, fotos familiares, algunos de pie, otros sentados</td><td>462 imágenes positivas</td></tr>
<tr><td>El robot y las personas caminan por el lugar</td><td>640x480 pixeles</td><td>Casas, paisajes, parques, cualquier ambiente sin personas</td><td>1218 imágenes negativas</td></tr>
<tr><td>El robot sigue a una persona que le da la espalda, mientas otros se atraviesan</td><td>640x480 pixeles</td><td>Total entrenamiento</td><td>2000 imágenes positivas 1218 negativas</td></tr>
<tr><td rowspan="2">El robot entra a una habitación con personas y explora la habitación</td><td rowspan="2">640x480 pixeles</td><td>Gente caminando, sentada, de perfil, de espaldas</td><td>50 imágenes de 800x600 pixeles</td></tr>
<tr><td>Imágenes extraídas de los videos</td><td>140 imágenes de 640x480 pixeles</td></tr>
<tr><td>Total</td><td>6 videos de aprox. 5 min. cada uno</td><td>Total</td><td>190 imágenes para evaluación</td></tr>
</table>

Cuadro 5.1. *Resumen de los elementos que conforman el corpus*

5.2. Diseño experimental

En los experimentos que se realizaron se analiza el rendimiento de las fases. Primero se hicieron los experimentos para evaluar sólo la fase de detección haciendo una variación en los parámetros que más adelante se mencionarán. En cada cambio se medían los resultados que se arrojaban, para después graficar y hacer un análisis. A partir del análisis de estas gráficas se hizo la elección del detector que sería puesto en el sistema completo. Además de tomar la decisión de qué detector sería el mejor para usarse, el de cabezas o el de cabezas-hombros.

Para la fase de rastreo, se analizan los resultados del sistema completo con el detector que previamente se eligió. Para ello también se hacen cambios en algunos parámetros y se miden sus resultados para que posteriormente se construyan gráficas que nos ayuden a tomar una decisión final acerca de los parámetros que mejor se ajustan al sistema.

5.2.1. Experimentos de detección

De todos los parámetros que pueden cambiarse, sólo se eligieron tres, debido a que son los que afectan a la detección de manera más significativa que los demás. Estos parámetros son:

- El tamaño de la ventana de detección.
- El umbral para la distancia del clasificador.
- El tamaño del cambio en la escala.

El objetivo de cambiar estos parámetros es para conocer el comportamiento de los detectores en diferentes imágenes tomadas en distintos lugares, personas, poses de las personas y diferentes distancias; y en base a las mediciones que se hicieron, elegir el que mejor eficiencia tenga, es decir, el que tenga menor número de falsos positivos, falsos negativos y un mayor número de verdaderos positivos.

Para los clasificadores se utilizó una ventana cuadrada de 4 tamaños distintos. La decisión se basó en el detector de personas por partes, que Dalal propuso en su tesis [6]. Dalal propone una ventana cuadrada de 32x32 para la parte de la cabeza-hombros ya que es proporcional a su detector de personas de cuerpo completo que tiene una medida de 64x128. En nuestro caso se optó por evaluar el rendimiento de clasificadores con un tamaño de ventana de 24x24, 32x32, 48x48, y 64x64 para observar su comportamiento y decidir cúal de ellos era el más conveniente para el robot Golem-II+ y usarlo en la fase de rastreo.

Para este proceso se siguieron los pasos en el orden en el que se listan a continuación:

- **Recolectar dos conjuntos de imágenes:** tanto positivas (con personas) como negativas (sin personas).
- **Etiquetar, recortar y redimensionar:** se etiquetan las imágenes positivas en el lugar en donde se encuentra una persona tanto de su

cabeza como de su cabeza y hombros en etiquetas separadas, para después recortar esa porción de la imagen y redimensionar a los tamaños de ventana anteriormente presentados.

- **Obtener los clasificadores(entrenamiento):** se hace la extracción de las características HOG de cada imagen recortada, así como de porciones aleatorias del mismo tamaño de las imágenes negativas. Estas características se usan como entrada para el algoritmo *SVM* que creara un clasificador entre las clases *cabeza* y *no cabeza* o *cabeza-hombros* y *no cabeza-hombros*

Con los clasificadores que se obtuvieron se hizo una variación en los parámetros del umbral de distancia y en el tamaño del cambio de escala. Para la parte del tamaño de cambio en la escala se partió desde el valor más bajo que puede ser utilizado, que es 1.01 hasta 1.5 donde se notó que ya casi no se hacían detecciones en la imagen de entrada, haciendo incrementos de 0.01. Para el último parámetro, el umbral para la distancia del clasificador, se partió desde el valor más bajo aceptado que es 0.1 hasta el valor de 2.0, debido a que también se notó que con este valor había ya muy pocas detecciones, con incrementos de 0.1 para éste. A continuación se muestran los rangos anteriormente mencionados:

	Valor inferior	Valor superior	Incremento
Escala	1.01	1.50	0.01
Umbral distancia	0.1	2.0	0.1

Cuadro 5.2. *Rangos de valores para la escala y el umbral.*

En este proceso de variación de parámetros se fueron registrando el número de falsos positivos, verdaderos positivos y falsos negativos, tomando en cuenta lo siguiente:

- Verdaderos positivos: son aquellos cuadros donde el detector señala la existencia de una persona y efectivamente la hay. Para que un cuadro detectado se tome como un verdadero positivo debe satisfacer que el área entre los dos cuadros, sea mayor al 33 % del área del cuadro de tamaño mayor, es decir, debe satisfacer la condición siguiente:

$$\frac{|S \cap D|}{max(|S|, |D|)} > 0{,}33 \tag{5.1}$$

  donde:

  S es el cuadro marcado.

  D es el cuadro detectado.

  $|A|$ es área de un cuadro A.

- Falsos positivos: son aquellos cuadros en donde el detector marca que en ese lugar hay una persona y no la hay.

- Falsos negativos: son aquellos cuadros en donde el detector no marca que hay una persona, pero en realidad si la hay.

5.2.2. Experimentos de rastreo

Para esta fase, el análisis que se hace es ahora con el sistema completo, ya que el rastreo depende directamente de la fase de detección, por ello, esta serie de experimentos se aplican a las dos fases. En estos experimentos se mide el rendimiento que tiene el sistema modificando las propiedades de los objetos *track* que fueron definidos en el capítulo **Rastreo multi-objetivo con el filtro de Kalman**.

El primer experimento que se hizo fue el de cambiar los valores: $lim\_inferior$ y $lim\_superior$ de los objetos *track*, ésto con el objetivo de medir el efecto que causa en el número de *falsos positivos* que marca el rastreo. Este cambio en los valores se fue haciendo de acuerdo al número de imágenes consecutivas que se encontrarán, entonces si el parámetro que se cambia es $lim\_inferior$, éste entre más grande sea, más exigente será el rastreo, es decir, para que un *track* sea considerado para mostrarse como un rastreo válido de una persona, éste deberá ser mayor, y análogamente entre más pequeño sea, menos exigente será el sistema para que un objeto *track* sea considerado como un rastreo válido. El valor de este parámetro se cambia esperando encontrar el valor ideal en el que se tenga un menor número de falsos positivos, ya que entre más exigente sea el sistema, menor número de falsos positivos se espera obtener, pero tratando de que esta exigencia sea la más baja posible, ya que si el sistema es muy riguroso será difícil que se obtenga el rastreo de algún individuo.

Otro de los parámetros que se hace variar es el $lim\_superior$. Este parámetro define el número máximo que alcanza el atributo *tiempo_de_vida* y se utiliza para que este último no crezca indefinidamente. Se espera que este valor no esté tan alejado del $lim\_inferior$ ya que al tener una mayor diferencia entre estos dos, un objeto *track* que ya ha sido marcado como válido y que por alguna razón éste deje de serlo, tardará demasiado en marcarse como inválido, por ejemplo, si una persona pasa caminando en frente de la cámara y ésta está el suficiente tiempo como para que empiece a ser rastreada pero de repente sale del rango de vista de la cámara, entonces el rastreo debería detenerse, sin embargo si el valor del $lim\_inferior$ es muy alto, éste tardará mucho en dejar de tratar de seguir rastreando a la persona.

Cuando se empezó a hacer este experimento, se observó que el sistema tardaba demasiado en detectar a las personas en el video que se estaba capturando, ésto es debido a que en la fase de detección, entre más pequeño sea el tamaño de ventana de detección más se tardará en hallar a las personas en cada imagen. Para solucionar este problema se planteó la propuesta de no tomar todas las imágenes del video, es decir, saltarse algunas imágenes que están siendo capturadas por la cámara y no hacer detección en ellas. Para observar los efectos de este experimento se hicieron varias pruebas y en cada una de ellas se cambiaba el número de imágenes que se ignoraban entre cada detección.

5.3. Resultados

En esta sección se presentan los resultados obtenidos de los experimentos, tanto de la fase de detección como del sistema completo que contiene la fase de rastreo.

Como primera parte se darán a conocer los detalles de las gráficas que se usaron para comparar los detectores, las cuales se muestran en las Figuras 5.6 y 5.7. Estas gráficas son conocidas como *Detection Error Tradeoff* (DET), que dibujan una curva relacionando los tipos de error. Para el sistema completo, la evaluación se hace mediante gráficas más sencillas, pero que también muestran una forma de comparar los diferentes puntos de operación del sistema.

5.3.1. Detección

La forma adecuada de presentar el funcionamiento del detector, es generando una curva para ver el comportamiento en los diferentes puntos de operación. El punto de operación es un parámetro que puede ser ajustado para llevar a cabo la toma de decisiones del detector.

En la detección, se aprecian dos tipos de error: el detector puede dar tanto una falsa alarma(falso positivo) como una detección fallida (falso negativo). La curva DET(*Detection Error Tradeoff*) ha sido desarrollada para apreciar el funcionamiento de un detector de forma más sencilla, debido a que se grafica la desviación normal en ambos ejes, dando un tratamiento uniforme para ambos tipos de error y usa una escala logarítmica para ambos ejes, ésto

mejora la forma de observar el funcionamiento de los sistemas y produce gráficas que son cercanas a líneas rectas, facilitando ver la diferencia entre ellas en un punto de operación, Sánchez[27], Martin et al.[28].

Para el eje de las ordenadas de la gráfica, se calcula la *tasa de pérdidas* o *miss rate* en inglés y se calcula de la siguiente manera:

$$MissRate = \frac{\#falsos\_negativos}{\#verdaderos\_positivos + \#falsos\_negativos} \tag{5.2}$$

Para el eje de las abscisas, se presenta el *número de falsos positivos por ventana* o $FPPW$(*False Positive Per Window* en inglés) que se calcula obteniendo el número de ventanas que puede tener una imagen dividida entre el número de falsos positivos de la imagen, es decir:

$$FPPW = \frac{\#falsos\_positivos}{\#ventanas} \tag{5.3}$$

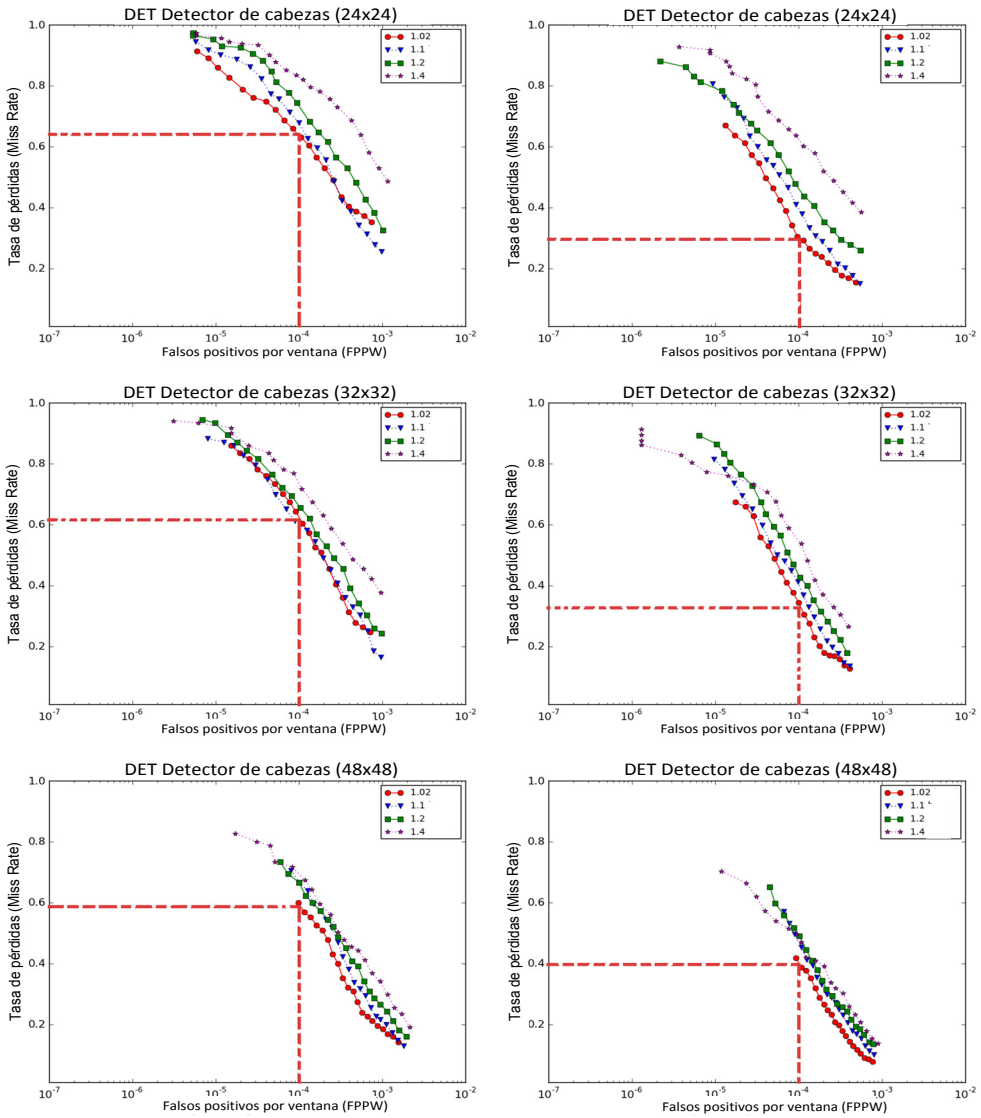

Figura 5.6. *Resultados para los detectores de cabezas con un tamaño de ventana de 24x24(arriba), 32x32(medio) y 48x48(abajo) pixeles. Usando el conjunto de Internet(izquierda) y el extraído del robot Golem-II+ (derecha).*

La Figura 5.6 muestra las gráficas obtenidas por los detectores de cabezas. Como se puede observar, un número menor en los falsos positivos por ventana

(FPPW) es necesario para un detector práctico. En la mayoría de las gráficas se observa una tendencia a que entre menor sea el valor de este parámetro, mayor será el valor de la tasa de pérdidas o *miss rate*. Ésto quiere decir que entre menos falsos positivos por ventana, mayor será el número de veces que falle en una detección. El objetivo entonces es hallar el detector donde los valores de FPPW y *miss rate* sean los menores posibles en su respectiva curva DET, indicando que el detector tiene el mejor desempeño respecto a los otros.

Para cumplir este objetivo, se mantuvo fijo un valor de FPPW como punto de referencia, debido a que entre menor sea éste, mayor será el valor de *miss rate*. Se eligió como punto de referencia un valor de 10^{-4} en FPPW ya que éste es aproximadamente el valor medio que se encuentra en los resultados, además este valor también es usado por Dalal como punto de referencia para comparar sus resultados y es el más apropiado en la práctica. En las gráficas se muestra una línea segmentada indicando el valor de 10^{-4} en FPPW y su correspondiente valor de *miss rate*.

Cada curva en las gráficas es diferenciada por el tamaño de incremento en la escala. Se observa que entre menor sea éste incremento de tamaño, menores serán los valores de FPPW y *miss rate* por lo que la curva con valor pequeño se apreciará más abajo que las demás curvas en la gráfica, es por eso que se toma como referencia la escala menor. Las 3 gráficas de la Figura 5.6 de la primera columna se construyeron usando el conjunto de imágenes que se obtuvieron de Internet, éstas muestran que su valor de *miss rate* oscila alrededor de 0.6, esto quiere decir que ese detector en ese punto de operación de cada 100 cabezas, 60 no serán detectadas y 40 serán detectadas, para ese

tipo de imágenes. Si se observa la siguiente columna de gráficas, las cuales pertenecen al conjunto de imágenes extraídas directamente de la cámara del robot Golem-II+, este valor mejora. Si se observa la primera gráfica de esta columna el valor de *miss rate* es de aproximadamente 0.3 indicando que de cada 100 cabezas, 30 no serán detectadas.

Bajo estos términos, el detector con un tamaño de ventana de 24x24 con un incremento de escala de 1.02 muestra los mejores resultados para las curvas de los detectores de cabezas.

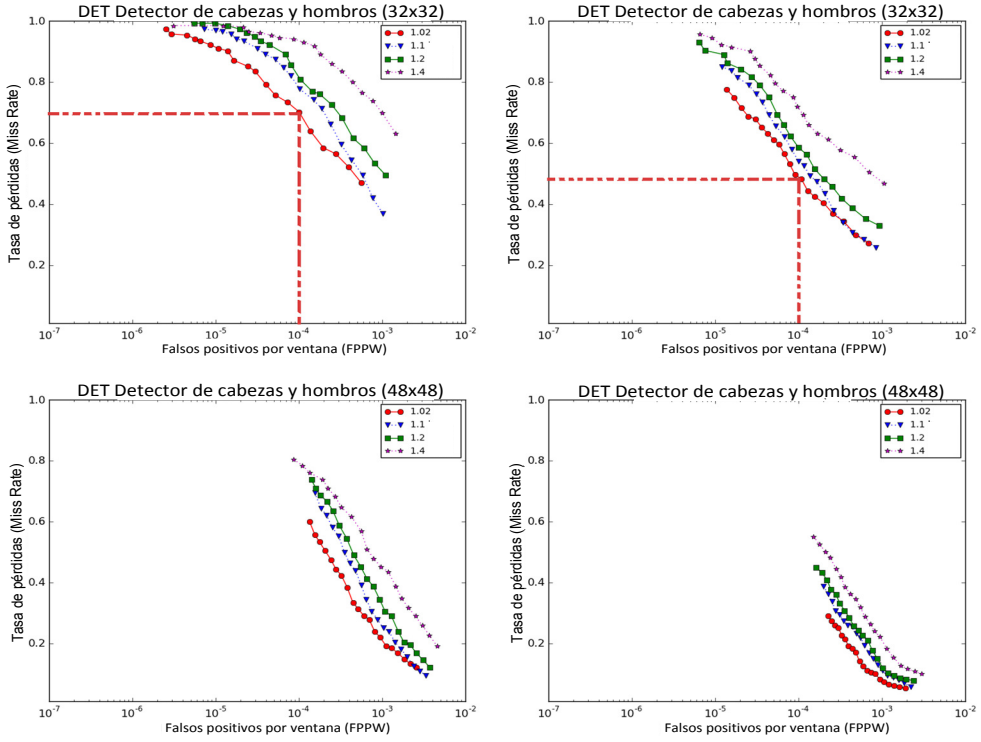

Figura 5.7. *Resultados para los detectores de cabezas y hombros con un tamaño de ventana de 32x32(arriba)y 48x48(abajo) pixeles. Usando el conjunto de Internet(izquierda) y el extraído del robot Golem-II+ (derecha).*

Para los detectores de cabezas y hombros, los resultados no son tan favorables. Como se ve en la Figura 5.7 para el detector con una ventana de 32x32 el valor de *miss rate* es de aproximadamente 0.7 para las imágenes de Internet y de alrededor de 0.5 en el conjunto de Golem. En la misma figura para las gráficas de la segunda fila, las líneas segmentadas ya no se dibujaron debido a que la curva ya no llega a los 10^{-4} en FPPW. Las gráficas que se omitieron les pasó algo similar, se encontraban muy a los extremos sin llegar al punto de referencia y éstas ya no fueron reportadas debido a que su rendimiento observado en las gráficas no es el adecuado.

5.3.2. Rastreo

Para estos experimentos se obtuvieron las gráficas que se muestran en la Figura 5.8. Para construir éstas, se usaron los videos del corpus que fueron extraídos del robot Golem-II+. Para realizar la evaluación se tuvo que hacer un conteo de las personas en cada video, éste se hizo registrando el número de personas en cada rango de imágenes, por ejemplo, si en un video aparecían 3 personas de la imagen 1 a la imagen 100, entonces se registraba $[1-100]$ 3, y si en los siguientes 100 aparecían 2, entonces se registraba $[101-200]$ 2 y así sucesivamente hasta que se etiquetaron todos los videos. Una vez etiquetados, éstos se sometieron al sistema de rastreo y se capturaron los resultados de la siguiente manera: se hizo una especie de histograma basándose en el etiquetado de los videos, este histograma se trata de un vector que va de -6 a 6. Si los resultados del rastreo en el mismo rango de imágenes que fueron etiquetadas en un video eran exactamente el mismo número de personas, entonces eso quiere decir que el rastreo no se equivocó ninguna vez, por lo

tanto el índice 0 del vector histograma se sumaba en 1, si el resultado del rastreo nos daba una persona menos entonces el vector se incrementaba en uno pero en el índice -1 y así sucesivamente. Como ejemplo, digamos que un video A de 200 imágenes se etiquetó de la siguiente manera:

- $[0 - 50]$ 2
- $[51 - 150]$ 3
- $[151 - 200]$ 4

y los resultados del rastreo fueron los siguientes:

- $[0 - 20]$ 2
- $[21 - 120]$ 3
- $[121 - 200]$ 2

Para el análisis podemos hacer la siguiente tabla, que nos muestra la diferencia de personas que se encontraron entre los rangos de imágenes.

Imagen inicial	Imagen final	Diferencia de personas
0	20	0
21	50	+1
51	120	0
121	150	-1
151	200	-2

Cuadro 5.3. *Diferencia de personas en cada rango de imágenes.*

Debido a que el primer rango de imágenes del video etiquetado va de $[0-50]$ y el primer rango del resultado del rastreo va de $[0-20]$, estos rangos no pueden ser comparados directamente, por lo tanto se usa el rango más pequeño para que pueda hacerse la diferencia correspondiente. Dicha diferencia se visualiza en el primer renglón de la tabla anterior, el resultado es que en los dos rangos se tiene el mismo número de personas, por lo tanto la diferencia es 0. Para los siguientes rangos se tiene que hacer un rango auxiliar, por ejemplo, el siguiente rango en el resultado del rastreo va de $[21-120]$ pero aún no se han tomado las 30 imágenes restantes del primer rango del video etiquetado, por lo tanto el rango auxiliar irá de $[21-50]$, tomando así las 30 imágenes faltantes del rango etiquetado. Este proceso se repite hasta que se tienen todas las diferencias entre todos los rangos del video.

Una vez que se termina este proceso, se procede a construir el vector histograma, éste se obtiene haciendo un conteo de cada diferencia según el valor del índice del vector. Para el caso del ejemplo anterior el histograma sería el siguiente:

-6	**-5**	**-4**	**-3**	**-2**	**-1**	**0**	**1**	**2**	**3**	**4**	**5**	**6**
0	0	0	0	0	1	2	1	1	0	0	0	0

Cuadro 5.4. *Vector histograma que muestra las veces que el sistema se equivocó o acertó.*

Finalmente se grafican los índices del histograma y con esto sabremos el número de veces que el rastreo se hizo correctamente y cuántas veces se equivocó.

En las gráficas de la Figura 5.8 se muestran los resultados de este proceso de evaluación. En la primera fila de la imagen se indica que estas gráficas se

hicieron sin saltarse imágenes del video, la segunda fila saltándose una imagen y así sucesivamente. Las columnas corresponden a varias configuraciones de $lim\_inferior$ y $lim\_superior$, las cuales siguen la descripción del Cuadro 5.5.

Columna 1		Columna 2	
lim_inferior	lim_superior	lim_inferior	lim_superior
5	10	10	20
5	30	10	15
10	15	15	20
10	30	20	30
15	20	30	35

Cuadro 5.5. *Cuadro que muestra la relación entre los límites de los frames.*

La elección de estos valores se hizo tratando de cubrir aspectos tales como: que la diferencia entre estos limites fuera pequeño(diferencia de 5), medio(diferencia de 10) y alto(diferencia de 20), así como ver lo que pasaba cuando estos limites se hacían más grandes, repitiendo estos aspectos mencionados.

Observando las gráficas de la primera fila, cuando el límite inferior es pequeño, el rastreo tiende a equivocarse menos que cuando es más grande. Por ejemplo en el caso en el que $lim\_inferior = 5$ con $lim\_superior = 10$ saltando 0 imágenes del video (Figura 5.8 gráfica del primer renglón y primera columna), podemos observar que en el índice cero del eje de las abscisas el puntaje es de 90, refiriéndose a que bajo esta configuración de limites,

el rastreo detectó a todas las personas del video en 90 rangos de imágenes. Si ésto lo comparamos con la gráfica en donde el $lim\_inferior = 10$ con $lim\_superior = 20$ saltando cero imágenes(Figura 5.8, gráfica de la primera fila y segunda columna), el número de rangos en los que el rastreo no se equivoca es de aproximadamente 55, indicándonos que el rastreo se desempeña mejor cuando $lim\_inferior$ es menor, debido a que estas dos configuraciones son las que obtienen el puntaje más alto cuando el rastreo se equivoca 0 veces. Tomando lo anterior como base podemos entonces comparar el caso cuando $lim\_inferior = 5$ - $lim\_superior = 10$ con el caso $lim\_inferior = 5$ - $lim\_superior = 30$ (Figura 5.8, gráfica de la primera fila primera columna), ésto con la finalidad de saber qué pasa cuando esta diferencia incrementa. En la Figura 5.8 cada columna muestra una configuración diferente en cuanto a los parámetros $lim\_inferior$ y $lim\_superior$, así como cada gráfica muestra un valor diferente de imágenes ignoradas.

Podemos ver que cuando este valor de $lim\_superior$ es 30, el puntaje es de aproximadamente 22, mucho menor que el caso donde $lim\_superior = 10$ que tiene aproximadamente un puntaje de 90. Este cambio hace que disminuya considerablemente el desempeño del rastreo, de hecho si se observa la gráfica, este par de valores es el más bajo de todas las configuraciones para cuando el número de rangos de imágenes que se equivoca el rastreo es igual a cero. Ésto nos indica que entre más grande sea la diferencia entre estos valores, menor será el desempeño.

Si comparamos las gráficas fila por fila, observaremos una tendencia de disminución de las veces que el rastreo se equivoca cero veces, ésto quiere decir que entre más saltos de imágenes haya entre ejecuciones del rastreo, entonces menor será su desempeño, pero entre menor sea este salto de imágenes, el rastreo será mucho más lento y no se obtendrán los resultados esperados. También podemos observar que entre más saltos de imágenes se hagan, entonces se reducirá el número de observaciones que se realizan, es por ésto que la gráfica se ve más pequeña. En conclusión, se deberá elegir una configuración en la que se haga un compromiso entre la velocidad del rastreo y su precisión.

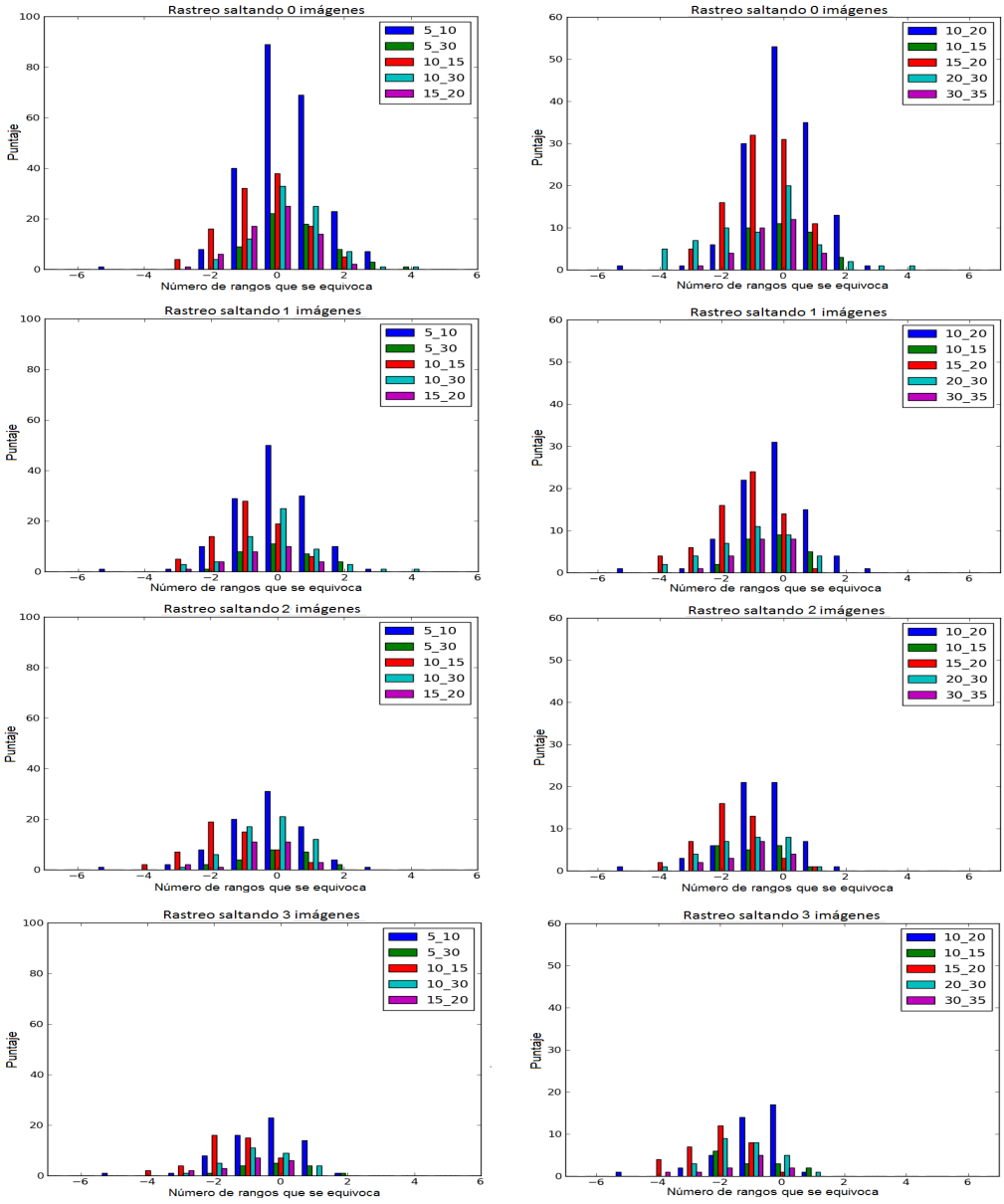

Figura 5.8. *Resultados de la evaluación del rastreo del sistema.*

Capítulo 6

Conclusiones y trabajo futuro

En este último capítulo se presentan las conclusiones obtenidas del trabajo realizado, en donde se presenta una propuesta para un sistema de detección y rastreo de personas para el robot móvil de servicio Golem-II+.

El sistema propuesto en este trabajo se adaptó para usarse directamente en el robot y que pudiera usar esta funcionalidad en el concurso internacional *Robocup@Home* presentando resultados aceptables.

Debido a que la interacción con los humanos es uno de los puntos más importantes de un robot de servicio, se decidió implementar un sistema de detección y rastreo para el robot móvil de servicio Golem-II+, con el que se sepa la posición de las personas en una serie de imágenes tomadas en tiempo real por la cámara del robot, y que el sistema sea capaz de seguir a las personas detectadas en todas estas imágenes. Para desarrollar este sistema primero se debe obtener un detector eficiente que permita conocer las posiciones de todas las personas en una imagen, teniendo el menor número de falsos positivos posibles y que éste realice la detección lo más rápido posible.

La siguiente fase es que a partir de las detecciones obtenidas, el sistema pueda ser capaz de relacionarlas con las detecciones de la próxima imagen, tomando en cuenta las posiciones anteriores, realizando una retroalimentación en cada detección y con ésto realizar el rastreo de las personas.
Para obtener la fase de detección del sistema se implementaron 4 detectores de cabezas de personas a partir de un conjunto de 2000 imágenes positivas de ejemplo, donde la gente se aprecia en posturas y ambientes diferentes, cada uno con un tamaño de ventana de detección distinta, con tamaños de 24x24, 32x32, 48x48 y 64x64 pixeles. Después de su implementación, se hicieron pruebas con diferentes configuraciones, cambiando las escalas de las ventanas al momento de la detección y variando también el umbral de distancia con el que se acepta o no que una ventana sea una detección. Todo ésto para realizar evaluaciones sobre sus desempeños, midiendo el número de veces que los detectores se equivocan con respecto a la cantidad de ventanas que se evalúan, para así tener una referencia en la cual se puedan comparar y así encontrar el detector con la configuración que tenga la mejor eficiencia, además de observar la tendencia en cuanto al cambio de configuraciones que éstos presentaron.

También se implementaron otros 4 detectores de hombros y cabezas bajo los mismos tamaños y objetivos que los anteriores, se hicieron el mismo tipo de evaluaciones para encontrar aquel que tiene el mejor rendimiento entre ellos.

Otro de los objetivos por el cual se entrenaron este segundo tipo de de-

tectores fue bajo la premisa de que podían tener mejor rendimiento que los anteriores, debido a que la forma que se trata de detectar es más distinguible que sólo la forma de la cabeza.

Se evaluó y comparó cada uno de los detectores y se encontraron diferentes puntos a tomar en cuenta, por ejemplo, en general entre más pequeña sea la ventana de detección, más precisa será la detección de personas, pero al ser más pequeña esta ventana, más poder de computo será demandado debido a que se tienen más ventanas a ser analizadas por imagen y por lo tanto la velocidad del detector disminuye.

También se encontró que el detector de cabezas es más eficiente que el de la cabeza hasta el hombro, ésto se cree que es debido a que las características que se buscan en el de cabezas son más comunes que las de cabezas y hombros, ésto es por que si los hombros se mueven aunque sea solo un poco, es más probable que no se detecte debido a que la variación de la postura es mucho más notable. En el detector de cabezas también ocurren este tipo de variaciones, pero la forma de la cabeza cuando se cambia de postura la variación no es tan notable, siempre y cuando la cabeza se mantenga erguida, es decir, que la persona no doble el cuello, por ejemplo cuando mira hacia el suelo.

El detector de cabezas que obtuvo el mejor resultado y que fue el usado para el sistema, es el que tiene la ventana de detección más pequeña(24x24) con una tasa de fallo de aproximadamente 30 % para imágenes que usa el

robot. Pero debido a que la velocidad con la que hace las detecciones es la más baja, se decidió optar por el de 48x48, que tiene una tasa de fallo de aproximadamente 40 % para las mismas imágenes, ésta no es una gran diferencia entre estos dos detectores, pero la diferencia de velocidades si es muy notable entre ellos.

Con el detector definido, lo siguiente fue desarrollar el módulo de rastreo que hace uso de éste como base para su funcionamiento. Este módulo utiliza el algoritmo Húngaro para relacionar las detecciones en el tiempo de cada persona visible por la cámara del robot, ésto es, conocer las relaciones entre las detecciones anteriores y las detecciones actuales. Después de aplicar el algoritmo Húngaro, se procede a usar el filtro de Kalman, éste se encarga de predecir el lugar en donde la detección actual se va a encontrar, tomando en cuenta el cambio en el movimiento que las detecciones anteriores han tenido. Con la medición actual(última detección) se hace una corrección de la predicción, otorgándonos una estimación de la posición real de la persona, minimizando el ruido que el detector nos de en cada una de las detecciones.

Al conjunto de acciones que hacen estos algoritmos se le conoce como rastreo. Con el rastreo integrado, el sistema completo fue sometido a ciertas pruebas que evalúan su correcto desempeño. A este nivel del sistema, se encontraron otros detalles que hacen que se vea afectada la velocidad a la que éste hace las detecciones y el rastreo. Debido a lo anterior, se optó que los experimentos también se hicieran en diferentes partes del video, sin tomar en cuenta algunas de las imágenes para que la detección no se haga en to-

das perjudicando menos a la velocidad del sistema y ver como se reflejaba en el rendimiento. Se encontró que entre más imágenes sean omitidas menos preciso será el sistema, ocasionando que falle más veces que no omitiendo imágenes.

Las evaluaciones aplicadas al sistema, han sido tanto en ambientes en el que comúnmente el robot se encontrará y demostrado que el sistema es estable, robusto, que presenta un alto grado de fiabilidad, pero también han demostrado que tienen muchos aspectos que pueden ser mejorados para obtener resultados aún más satisfactorios. Por ejemplo, haciendo que éste presente un número de falsos positivos y falsos negativos que prácticamente sean cero. Ésto se puede obtener incrementado el número de imágenes de muestra donde haya personas con posturas diferentes a las que se usaron para el aprendizaje de los detectores. Otro aspecto importante que puede ser mejorado es la parte del filtro de Kalman, en este punto puede ahondarse más en el modelo que se utiliza para hacer la proyección de los estados pasados y así obtener la predicción del estado actual. Para mejorar aún más el sistema, se puede pensar en hacer una fusión de datos con el *Kinect*, ya que su principal ventaja es obtener información de profundidad en el ambiente.

Al obtener esta información de profundidad, lo único que se tendría que hacer para saber la posición de una persona en un espacio de 3 dimensiones, sería relacionar la información que presenta el rastreo en dos dimensiones (x, y) con la información sin procesar que presenta el *Kinect* en 3 dimensiones (x, y, z) de toda la escena. Con esta técnica podrían hacerse filtros de falsos positivos que mantengan la coherencia de las detecciones, por ejemplo,

si el rastreo ésta siguiendo a algo que muestra demasiada profundidad, con poca profundidad o que se muestren saltos de profundidad muy pronunciados, estos rastreos tienen una alta probabilidad de ser falsos positivos.

En Resumen, las contribuciones que se hicieron son:

- Desarrollo de varios detectores de cabezas y de cabezas-hombros.
- Comparación de los detectores desarrollados.
- Propuesta y desarrollo del sistema de rastreo.
- Evaluación del sistema de rastreo.
- Desarrollo de un programa para etiquetación de imágenes.

Los resultados más interesantes que se encontraron son:

- El detector de cabezas encuentra a más personas en las imágenes de entrada que el detector de cabezas-hombros, debido a la diferencia de la forma.
- El detector que se decidió usar es el correspondiente a la ventana de detección de tamaño medio (48x48), debido a que los de tamaño más pequeño (24x24 y 32x32) son lentos y las tasas de fallo no son muy distantes entre los tres.
- El sistema propuesto es capaz de detectar y rastrear a varias personas a la vez, conociendo la diferencia entre qué detección pertenece a qué persona a través de todas las imágenes capturadas.

Bibliografía

[1] International Federation of Robotics. *Consultado online en http://cbcl.mit.edu*, 2015.

[2] N. Bellotto and H. Hu. **International Federation of Robotics, "Service Robots"**. *IEEE Transactions on Systems, Man, and Cybernetics B, Cybernetics, vol. 39, no 1, pp. 167-181*, Feb 2009.

[3] N. Dalal and B. Triggs. **Histograms of Oriented Gradients for Human Detection.** *Conference on Computer Vision and Pattern Recognition, San Diego, USA*, Junio 2005.

[4] P. Viola and M. Jones. **Rapid Object Detection using a Boosted Cascade of Simple Features.** *Accepted Conference on Computer Vision And Pattern Recognition*, 2001.

[5] P. Roth, H. Grabner, D. Skocaj, H. Bischof, and A. Leonardis. **Online conservative learning for person detection**. *Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, 2005.

[6] N. Dalal. **Finding People in Images and Videos**. *Institut National Polytechnique de Grenoble, Francia*, Julio 2006.

[7] J. Satake and J. Miura. **Robust Stereo-Based Person Detection and Tracking for a Person Following Robot**. *Toyohashi University of Technology, ICRA-2009 Workshopon Person Detection an Tracking, Kobe, Japón*, 2009.

[8] A. Treptow, G. Cielniak, and T. Duckett. **Real-time people tracking for mobile robots using thermal vision**. *En Robotics and Autonomous System*, 2006.

[9] Real Academia de la Lengua Española. **Diccionario de la lengua Española**. *Consultado online en http://www.ifr.org/*, 2014.

[10] L. Kornienko and L. Kleeman. **An autonomous human body parts detector using a laser range finder**.

[11] M. Becerra. **Sistema de Seguimiento de Personas para un Robot Movil de Servicio.** *UNAM*, 2012.

[12] Microsoft Corporation. **Microsoft**. *Consultado online en http://www.microsoft.com/*, 2014.

[13] L. Xia, C. Chen, and J. K. Aggarwal. **Human Detection Using Depth Information by Kinect**.

[14] S. Burion. **Human Detection for Robotic Urban Search and Rescue**. *Intitut de Production Robotique*, Febrero 2004.

[15] P. Felzenszwalb, D. McAllester, and D. Ramanan. **A discriminatively trained, multiscale, deformable part model**. *CVPR*, 2008.

[16] X. Wang, T. X. Han, and S. Yan. **An HOG-LBP Human Detector with Partial Occlusion Handling**.

[17] C. Ivorra Castillo. **Geometría**.

[18] R. E. Kalman. **A New Approach to línear Filtering and Prediction Problems**. *Research Institute for Advanced Study. Baltimore, Maryland, Estados Unidos de America*, 1960.

[19] G. Welch and G. Bishop. **An Introduction to the Kalman Filter**. *Universidad del norte de Carolina*, Julio 2006.

[20] M. S. Grewal and A. P. Andrews. **Kalman Filtering Theory and Practice Using MATLAB**. *Tercera Edición. Publicado por Jonh Wiley e Sons, Inc*, 2008.

[21] H. W. Kuhn. **The Hungarian Method for the Assigment Problem**. *Universidad Princeton, Estados Unidos*, 1955.

[22] R. Faragher. **Understanding the Basis of the Kalman Filter Via Simple and Intuitive Derivation**. *IEEE Signal Processing Magazine*, Septiembre 2012.

[23] Learning and Recognition in Vision. **Inria, Laboratoire Jean Kuntzmann**. *Consultado online en http://pascal.inrialpes.fr*, 2014.

[24] Poggio Lab. **Center for Biological & Computational Learning**. *Consultado online en http://cbcl.mit.edu*, 2014.

[25] Advanced Multimedia Processing Lab. **Cornell University, School of Electrical and Computer Engineering**. *Consultado online en http://chenlab.ece.cornell.edu*, 2014.

[26] Creative Commons. **Creative Commons**. *Consultado online en http://creativecommons.org/*, 2014.

[27] S. Sánchez, C. Henao, and M. Alvarez. **Evaluación de Algoritmos de Detección de Complejos QRS Mediante las Curvas de funciónamineto ROC,DET y EPC.** *cientia et Technica Año XIII, No 34. Universidad Tecnológica de Pereira*, Mayo 2007.

[28] A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki. **The DET Curve in Assessment of Detection Task Performance**. *Estados Unidos de America.*