UNIVERSIDAD AUTÓNOMA DE YUCATÁN

FACULTAD DE MATEMÁTICAS

MAESTRÍA EN CIENCIAS DE LA COMPUTACIÓN

ESTIMACIÓN GENERAL DE LA MIRADA CON BASE EN LA GEOMETRÍA DE LA ESCENA USANDO REDES NEURONALES

T E S I S

EN OPCIÓN AL GRADO DE:

MAESTRO EN CIENCIAS DE LA COMPUTACIÓN

PRESENTA:

ING. RAJIV RACIEL GONZÁLEZ GONZÁLEZ

DIRECTOR DE TESIS:

DR. ARTURO ESPINOSA ROMERO

MÉRIDA, YUCATÁN, MÉXICO, 2018

Estimación general de la mirada con base en la geometría de la escena usando redes neuronales

Ing. Rajiv Raciel González González

Maestría Ciencias de la Computación
Facultad de Matemáticas
Universidad Autónoma de Yucatán
2018

Resumen

En el presente trabajo se propone un sistema para estimar la mirada de las personas de manera general, esto quiere decir que a partir de estimaciones discretas de la pose de la cabeza se asocian regiones segmentadas en un plano en frente de ellas representando la mirada. El proceso descrito a lo largo del documento tiene un enfoque basado principalmente en dos etapas: la fase de calibración del experimento y la fase de clasificación. En la primera etapa se utilizan técnicas de optimización numérica, geometría, álgebra lineal y calibración de cámaras con el objetivo de que la captura de datos para la clasificación sea lo más precisa posible. En la segunda etapa se utilizan en su mayoría herramientas de aprendizaje automático como el detector de rostros de Viola y Jones, descriptores de características HOG y redes neuronales con el propósito de clasificar la mirada de las personas.
Los métodos probados en las diferentes fases son descritos ampliamente en el documento. Cada una de las fases es descrita de manera independiente pero se mantienen dentro del contexto del objetivo principal.

Dedicatoria y agradecimientos

A mi madre por su cariño y apoyo incondicional, sin ella nada de esto sería posible.

A Jazmín mi novia y amiga, por ser alguien especial en mi vida e ilusionarse junto conmigo en todos mis proyectos.

A mis hermanos Roger y Juan por su apoyo y consejos que me brindan.

A mi sobrina Sofía que llena mi vida de alegría.

Al Dr. Arturo Espinosa Romero, por su paciencia, interés, optimismo y asesoramiento mostrado en la supervisión de la tesis.

Al Consejo Nacional de Ciencia y Tecnología (CONACYT), que me otorgó la beca con la cual pude sustentarme económica durante el estudio de la maestría.

Declaración

Por este medio declaro que yo escribí esta tesis y que describe el trabajo de mi tesis de Maestría en Ciencias de la Computación

Índice general

Índice de figuras

Capítulo 1

Introducción

1.1. Introducción

Conocer la posición, orientación y movimiento (pose) de la cabeza son cuestiones de gran importancia, ya que pueden ayudar a las personas a interactuar con las computadoras de una forma más natural en comparación de como se realiza actualmente. Un aspecto importante relacionado con lo anterior, es conocer lo que están observando las personas, a tal problema se le conoce como estimación de la mirada y ha sido estudiado ampliamente en los últimos años. Sus aplicaciones se extienden en el campo de la realidad virtual, video conferencias y publicidad, donde pueden ser utilizadas en sistemas publicitarios con pantallas, carteles, bancos de exterior, mesas, etc. y en combinación con el tiempo que miran las personas además de detectar su género y edad, se puede obtener información de suma importancia en cuanto a las preferencias del público y cómo son sus intereses en relación a sus características físicas. Además, conociendo dicha información de las personas en tiempo real se pueden implementar sistemas publicidad interactiva con los clientes y de este modo captar más su atención en relación a los sistemas clásicos (estáticos) actuales.

Por medio del análisis de imágenes capturadas a personas y en combinación con algoritmos de inteligencia artificial es posible conocer la posición y orientación de sus cabezas con respecto a un marco de referencia, la información obtenida puede ayudar a estimar la mirada de las personas, lo cual es el objetivo principal del presente trabajo de tesis: estimar la mirada de las personas en un plano enfrente de ellas.

Actualmente existen diversas técnicas para conocer la mirada, las cuales en su mayoría utilizan la pose del rostro en conjunto con la ubicación de los iris de ambos ojos, lo cual resulta en sistemas de estimación precisos pero al mismo tiempo limitados a utilizar múltiples cámaras, una distancia corta entre la cámara y las personas o sistemas con cámaras con lentes especiales para detectar los ojos e iris en distancias amplias con buena resolución, dando como resultado sistemas de estimación de mirada computacionalmente costosos.

En el presente trabajo de tesis se propone una posible solución a la estimación de la mirada a través de un sistema con una sola cámara monocular y herramientas de visión computacional, aprendizaje automático y optimización numérica, el sis-

tema tiene la capacidad de realizar la estimación sin la necesidad de localizar los ojos de la persona y como resultado se pretende obtener un rango de operación amplio de distancia entre la persona y la cámara.

1.2. Objetivo general

Detectar el rostro de las personas y estimar su mirada en un plano virtual situado enfrente de ellas, la estimación se realiza asociando regiones en dicho plano con información de textura de los rostros detectados y su posición.

1.3. Objetivos específicos

- Generar una base de datos de rostros humanos con la información de pose asociada.
- Desarrollar un sistema de realidad aumentada para la generación de puntos 3D con coordenadas específicas en una escena tomando como marco de referencia la cámara.
- Desarrollar el sistema para que funcione en un rango de distancia amplio entre la persona y la cámara.
- Implementación de un algoritmo clasificador para asociar la región que se observa con poses determinadas.

1.4. Estructura de la tesis

El presente trabajo de tesis se encuentra dividido por capítulos de la siguiente forma:

- El capítulo 2 describe el estado del arte, es decir, una recopilación de la información relacionada con el tema del proyecto de tesis.
- En el capítulo 3 se presenta el marco teórico, antecedentes y fundamentos que se requieren y se utilizan en este proyecto.
- En el capítulo 4 se presenta la metodología, la cual incluye la planeación, la teoría, el diseño y la implementación del sistema estimador de la mirada de las personas.
- El capítulo 5 describe el diseño y resultados experimentales llevados a cabo durante el desarrollo del sistema.
- En el capítulo 6 se presentan y discuten las conclusiones del proyecto, asi como también se describe el trabajo futuro.

Capítulo 2

Estado del arte

En el área de visión computacional la estimación de la pose de una cabeza es el proceso de inferir la orientación y la posición de una cabeza humana a partir de las imágenes [1]. El estimador debe ser robusto a distorsiones causadas por la cámara, a las expresiones faciales, al cabello, al vello facial y a objetos que ocluyan la cabeza, por ejemplo unos lentes o sombreros. La estimación de la pose está intrínsecamente ligado con la **estimación de la mirada**, por si misma, a partir de estimación de la pose de la cabeza se puede inferir una estimación general de la mirada, lo cual es útil para un gran número de aplicaciones prácticas [3]. En en la figura 2.1 se presentan dos posibles aplicaciones de los sistemas de estimación general de la mirada, en la parte superior izquierda de la imagen se muestra un cuarto inteligente donde la dirección del rostro despliega el foco de atención del usuario. La parte inferior de la imagen muestra un sistema de vehículos inteligentes: la vigilancia del conductor está basada en el análisis de la pose de la cabeza.

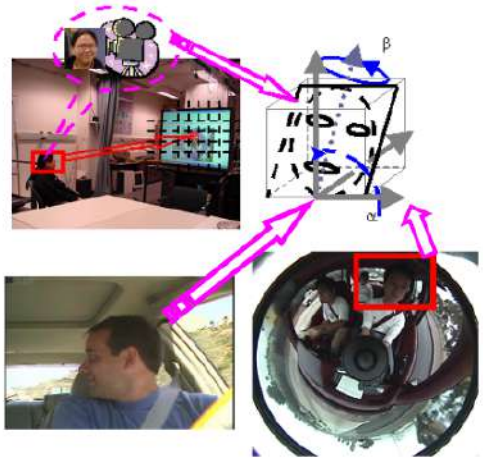

Figura 2.1: Aplicaciones de sistemas de estimación general de la mirada

Existen algunos aspectos que se deben tomar en cuenta al realizar investigación en este tema: se necesita previamente localizar la cabeza de las personas; la cabeza humana puede ser modelada como un objeto rígido sin tomar en cuenta el resto del cuerpo humano; la estimación de la pose de la cabeza se hace con

respecto a un marco de referencia centrado en la cámara y de manera más general a un sistema global de coordenadas; para estimar la mirada de las personas con precisión en cualquier configuración, un sistema seguidor de ojos debe ser complementado con el sistema de estimación de la pose de la cabeza [2].
Actualmente se han realizado una gran variedad de métodos que se pueden emplear para solucionar el problema de la estimación de la pose de la cabeza, razón por la cual motivó a Murphy-Chutorian y Manubhai [1] a desarrollar una taxonomía sobre los enfoques más importantes, en ella se presenta la clasificación de los métodos tomando como principal parámetro de clasificación el enfoque fundamental sobre el cual subyace la implementación del método. A continuación se presenta lo más destacado de cada enfoque:

- *Métodos de plantillas apariencias.*- Este método utiliza métricas de comparación basadas en imágenes para hacer correspondencias de la pose de una cabeza con un conjunto de ejemplos (plantillas) con las poses etiquetadas, a continuación se realiza la correspondencia con las plantillas más similares. El método no requiere entrenamiento con ejemplos negativos, solo requiere los ejemplos etiquetados.
  Desventajas:

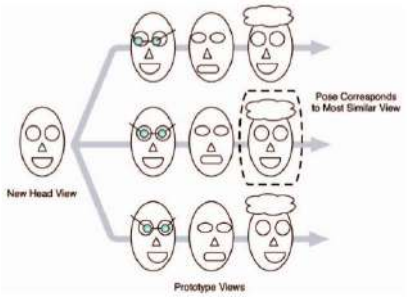

Figura 2.2: Plantillas de apariencias

  (a) Solo estiman poses discretas
  (b) Se requiere saber dónde se localiza el rostro
  (c) Ineficiente con una cantidad grande de ejemplos de entrenamiento,

  Como posible solución a los dos últimos problemas, en [12] y [28] proponen entrenar un conjunto de maquinas de vectores de soporte para detectar y localizar el rostro, y subsecuentemente usan los vectores de soporte como plantillas de apariencias para estimar la pose de la cabeza.

- *Métodos detectores en forma de arreglo.*- Se entrenan múltiples detectores de rostro con diferentes poses discretas, la diferencia con el método anterior es que la imagen de entrada es evaluada con un detector entrenado con bastantes imágenes y un algoritmo supervisado.
  Desventajas:

  (a) Se requieren demasiadas imágenes de entrenamiento

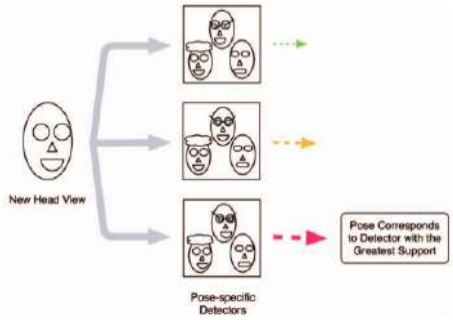

Figura 2.3: Detectores en arreglo

(b) Es difícil implementar el arreglo con un número de detectores extenso en un sistema en tiempo real

(c) Podrían darse ambigüedades en la clasificación (múltiples clasificaciones positivas).

Ventajas:

(a) La detección y localización no son etapas diferentes

(b) Tienen buen desempeño con imágenes de alta y baja resolución

- *Métodos de regresión no lineal.*- Estiman la pose mediante el aprendizaje de una función no lineal de mapeo desde un espacio de imágenes a una o más direcciones de pose. Con un conjunto de entrenamiento se puede construir un modelo que estime de forma discreta o continua una pose. Se ha demostrado éxito en este enfoque utilizando como algoritmo de aprendizaje las máquinas de regresión de soporte (SVR) [14], por ejemplo como se realizó en [15], [16] y [17].

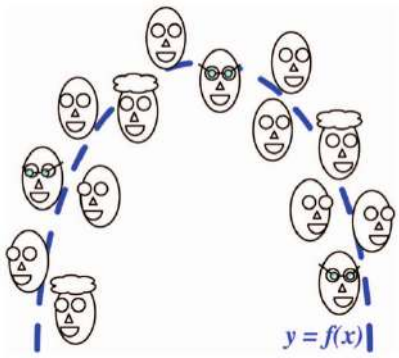

Figura 2.4: Regresión no lineal

Desventajas:

(a) No es claro cómo utilizan la herramienta específica de la regresión

(b) La estimación realizada es tosca (coarse) en las ubicaciones discretas

(c) Son propensas al error

Las ventajas de utilizar este método con redes neuronales son:

(a) Bastante rápidos

(b) Solo requieren imágenes etiquetadas para el entrenamiento

(c) Son los que arrojan los resultados más exactos en la práctica

- *Métodos de variedad-embebida o manifold.-* Estos métodos buscan manifolds de baja dimensión que modelen la variación continua en la pose de la cabeza. Nuevas imágenes pueden ser incrustadas en estos manifolds y después utilizarlos para la estimación con técnicas como la regresión en el espacio embebido. Cualquier algoritmo de reducción de dimensionalidad puede ser considerado como un intento de un manifold embebido, pero la dificultad yace en crear un algoritmo que exitosamente recupere la pose de la cabeza mientras ignora otras fuentes de variación en la imagen.

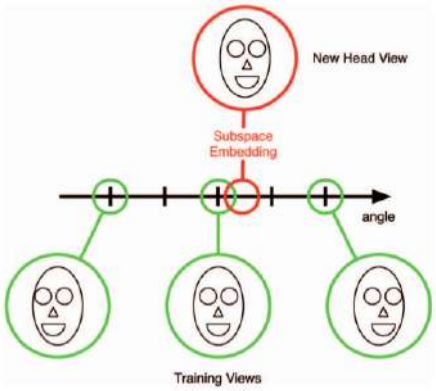

Figura 2.5: Manifold

Desventajas:

(a) Al igual que con los detectores en forma de arreglo la estimación de la pose es tosca ya que la estimación se deriva de un conjunto discreto de mediciones.

(b) La heterogeneidad en los datos de entrenamiento que es común en los escenarios para el entrenamiento en el mundo real representa un problema. Para contrarrestar lo anterior es necesario entrenar un manifold con múltiples personas [18], pero a menudo es imposible obtener un muestreo regular de poses de cada individuo.

Ventajas:

(a) Todas las técnicas de variedad-embebida que utilicen un enfoque lineal tienen la gran ventaja de que la reducción de las dimensiones del modelo de la pose se puede lograr con unas simples multiplicaciones de matrices lo cual es veloz en comparación con otros métodos, [19] y [20].

- *Modelos flexibles*.- Los modelos flexibles toman un enfoque diferente al de los métodos previos. En esta técnica un modelo no rígido es adaptado a la imagen de tal forma que se ajusta a la estructura facial de cada individuo, este método requiere datos de entrenamiento con las características del rostro anotadas, permite hacer comparaciones a nivel características en lugar de a nivel global de apariencias. Básicamente los modelos son plantillas basadas en grafos deformables de puntos característicos locales del rostro como las esquinas de los ojos, nariz, boca, etc. Para entrenar este sistema las ubicaciones de las características de los rostros son etiquetadas a mano en cada imagen de entrenamiento, estas características pueden ser extraídas de vistas de múltiples personas y extra invarianza puede ser lograda almacenando muchos descriptores en cada nodo, a esto se le conoce como Elastic Bunch Graph, además, tiene la capacidad de representar objetos no rigidos o deformables.
  El proceso de hallar las correspondecias (matching) se realiza con el algoritmo Elastic Graph Matching [21], con éste se compara un conjunto de grafos con la imagen nueva, para ello se coloca el grafo sobre la imagen y se calculan las distancias mínimas de la característica en cada locación del nodo del grafo.
  Para realizar la estimación de la pose un conjunto de grafos es creado en cada pose discreta y es comparado con una nueva vista de la cabeza, el conjunto de grafos con mayor similitud asigna una pose discreta a la cabeza.
  Desventajas:

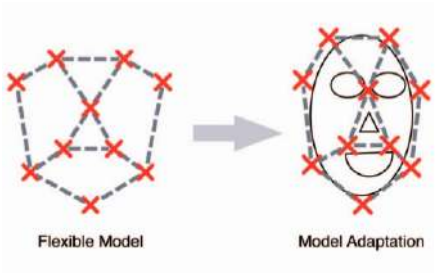

Figura 2.6: Modelos flexibles

(a) La estimación de la pose es discreta, requiriendo un número extenso de conjuntos de grafos y comparar los conjuntos con todas deformaciones resulta computacionalmente costoso.

(b) No es eficiente estar localizando todas las características de la cara en cada frame.

(c) Podría no estimar exitosamente la pose del rostro en imágenes de baja resolución y con el rostro lejos de la cámara.

Ventajas:

(a) Los modelos de apariencia activa (AAM) [22] los cuales evolucionaron de modelos flexibles tienen buena invarianza al error de localización

de la cabeza ya que ellos se adaptan a las imágenes y encuentran la posición exacta de las características de la cara.

Métodos geométricos.- Utilizan la forma de la cabeza y la configuración de las características locales para estimar la pose [23]. El movimiento de yaw (véase la figura 4.3) puede ser calculado con la distancia entre los ojos, roll con el ángulo que se forma entre la línea de los ojos y el horizonte, finalmente el pitch se calcula con la distancia de la punta de la nariz y la línea de los ojos.

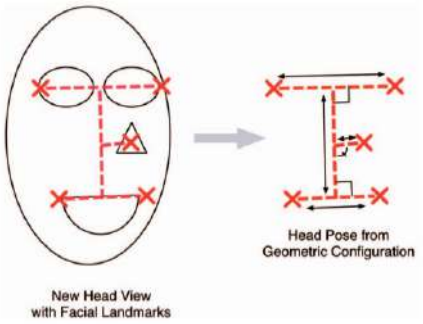

Figura 2.7: Métodos geométricos

Desventajas:

(a) Se necesita que la cabeza se encuentre cerca de la cámara para ver todas las líneas del rostro, como en [24].

(b) Es difícil encontrar las características con alta precisión y exactitud.

(c) Son métodos sensibles a las oclusiones en la cara.

Ventajas:

(a) Los métodos geométricos son rápidos y simples, con unas pocas características del rostro se puede obtener un decente estimador de la pose de la cabeza.

(b) Con simples características geométricas se puede estimar la pose de la cabeza, por ejemplo: el movimiento de yaw puede ser obtenido creando el triángulo entre los ojos y la boca y buscando el triángulo la desviación a partir de un triángulo isósceles [25].

2.0.1. Comparación de publicaciones

Los resultados reportados en la tabla 2.1 son algunas investigaciones similares a las que se propone en este trabajo de tesis en cuanto al aspecto de realizar una estimación general de la pose de la cabeza (o mirada) y experimentando con pocas poses discretas, excepto por la publicación de la primera fila [49] la cual se añade a la tabla debido a que es la que registra mejores resultados (en esta área) experimentando con bastantes poses. La precisión de la clasificación es el porcentaje de aciertos con las etiquetas de las poses.

Cuadro 2.1: Estimación de la pose "coarse"

Publicación	Precisión de la claficación	No. de poses discretas
J. Wu [49]	90 %	93
Voit [50]	39.4 %	8
Voit [54]	34.9 %	8
Zhang [51]	87 %	5
B. Ma [52]	97.14 %	7
Brown [3]	91 %	9
N. Krüger [53]	92 %	5

2.0.2. Métodos para la recolección del ground truth

Para evaluar y comparar los sistemas de estimación de pose se necesita de un método preciso para medir los datos de campo (ground truth) en un conjunto de datos evaluativos, generalmente el "ground truth" es esencial para el entrenamiento en cualquier método de estimación de la pose. La siguiente lista describe los métodos más comunes para capturar estos datos de campo, en orden del menos preciso (burdo) al más preciso:

- Sugerencia direccional.- Se les indica a las personas que miren hacia objetos en la habitación en la que se encuentran y se capturan las posiciones de la cabeza [26], no es muy preciso este método ya que entre otras cosas se asume que una persona tiene la habilidad de dirigir su cabeza con exactitud hacia los objetos.

- Sugerencia direccional con apuntador láser.- Igual al anterior pero con un láser [27], presenta los mismos problemas que el método anterior y las personas tienden a mover sus cabezas durante la captura de datos.

- Anotación manual.- Otras personas anotan la posición de la pose.

- Arreglo de cámaras.- Se capturan imágenes con múltiples cámaras a la misma persona con diferentes ángulos.

- Sensores magnéticos.- Los sensores se colocan en la cabeza de las personas y se obtienen las medidas del sensor [28]. Entre las desventajas es que son sensibles al ruido y presencia de metales.

- Sensores inerciales.- Utilizan acelerómetros y giroscopios y se reduce el ruido con un filtro de kalman. No miden posición solo orientación en tres grados de libertad [29].

- Sistemas ópticos de captura de movimiento.- Son robustos y muy costosos, son usados en captura profesional del movimiento articulado del cuerpo [17].

2.0.3. Características comunes de los métodos

Así como existen bastantes diferencias entre los métodos también hay unas características que prevalecen:

- Como métrica para conocer el desempeño de los métodos se utiliza el error de la media absoluta angular para pitch, roll y yaw.
- Mientras más poses se deseen estimar la clasificación se vuelve más difícil y propensa a cometer más errores.
- Un gran número de los enfoques en algún punto del proceso de estimación utilizan un algoritmo de aprendizaje automático, los más comunes son las redes neuronales [3], máquinas de vectores de soporte[30], y los métodos de boosting como el floatboost [31].
- Se logra bastante precisión en la estimación mediante la utilización de cámaras de visión estéreo.
- La variación de la pose está restringida a un rango que contiene todas las características visibles del rostro (que se utilizan para la clasificación) desde una vista frontal.

Capítulo 3

Marco teórico

Para realizar la investigación relacionada a la estimación de la mirada de las personas es necesario conocer algunos conceptos, definiciones y conocimientos que sirvan de apoyo al desarrollo de la presente tesis. En este capítulo se describen las bases para llevar a cabo el proyecto, dichas bases consisten principalmente de geometría proyectiva, calibración de cámaras, algoritmos de aprendizaje supervisado y optimización numérica.

3.1. Geometría proyectiva

La geometría proyectiva es una rama de la geometría que estudia como se proyectan los objetos tridimensionales en un espacio bidimensional (véase la figura 3.1), el cual se conoce como el espacio proyectivo y para trabajar en dicho espacio se utilizan coordenadas homogéneas.

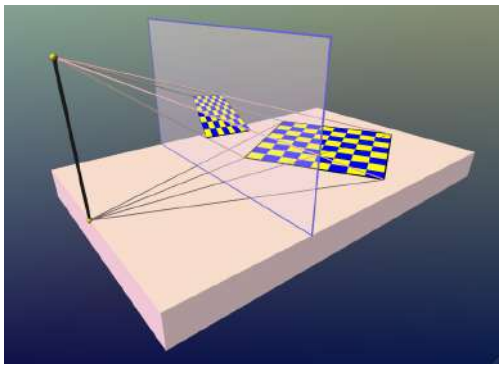

Figura 3.1: Geometría proyectiva

3.1.1. Coordenadas homogéneas

En coordenadas homogéneas un punto bidimensional está definido por tres coordenadas. De tal modo que un punto (x, y) se representa de forma homogénea como $(\frac{x}{w}, \frac{y}{w}, w)$, generalmente se suele dar el valor de 1 al tercer valor w.

Cuadro 3.1: Transformaciones proyectivas

Transformación	Matriz H	Invariantes
Euclideana	$\begin{bmatrix} R & T \\ 0^T & 1 \end{bmatrix}$	Distancia entre dos puntos, ángulo entre dos líneas y área
Similitud	$\begin{bmatrix} SR & T \\ 0^T & 1 \end{bmatrix}$	Ángulos entre rectas y razón entre distancias
Afín	$\begin{bmatrix} A & T \\ 0^T & 1 \end{bmatrix}$	Razón de áreas y el paralelismo
Proyectiva	$\begin{bmatrix} A & T \\ V^T & v \end{bmatrix}$	Razón cruzada

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x/w \\ y/w \\ w \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{3.1}$$

Análogamente sucede en el caso de tres dimensiones, es decir, para un punto tridimensional (x, y, z) se introduce una cuarta coordenada y se representa en forma homogénea como $(\frac{x}{w}, \frac{y}{w}, \frac{z}{w}, w)$, donde generalmente a la cuarta coordenada se le asigna 1, $w = 1$.

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} x/w \\ y/w \\ z/w \\ w \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{3.2}$$

Las coordenadas homogéneas resultan muy útiles ya que permiten integrar la traslación junto con la rotación y la escala, permitiendo tratar todas las transformaciones geométricas como multiplicaciones de matrices.

3.1.2. Tipos de transformaciones

Existen 4 tipos de transformaciones geométricas [44], las cuales se enlistan a continuación en la tabla 3.1 y se muestran ejemplos en la figura 3.2:

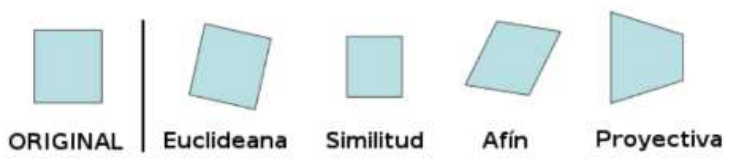

Figura 3.2: Transformaciones proyectivas

3.2. Calibración de cámaras y corrección de imágenes

Los métodos de calibración tienen como objetivo conocer los valores de los parámetros intrínsecos (internos) y extrínsecos (externos) de la cámara que se requiere modelar, haciendo uso de patrones geométricamente conocidos capturados en imágenes. Los métodos de calibración más populares son: el algoritmo de Tsai [34], el de Kanatani [35] y finalmente el de Zhang [36]. A continuación se describen los conceptos básicos del modelado de una cámara proyectiva y la implementación realizada por Bouguet [32] en MATLAB, la cual se utiliza para calibrar la cámara utilizada en este trabajo.

3.2.1. Conceptos básicos de la calibración

Los parámetros extrínsecos de la cámara indican su posición y orientación con respecto al marco de referencia del mundo. Estos parámetros son la matriz de rotación R y el vector de traslación t que constituyen la transformación de cuerpo rígido que superpone el sistema de referencia de la cámara con el sistema de referencia del mundo, aplicada como:

$$x_c = Rx_w + t \tag{3.3}$$

donde $x_c = (x_{c1}, x_{c2}, x_{c3})^T$ es un punto bajo el sistema de referencia de la cámara y $x_w = (x_{w1}, x_{w2}, x_{w3})$ es un punto bajo el del sistema de referencia del mundo. Los parámetros intrínsecos no dependen de la posición de la cámara, sino únicamente de su construcción y de la posición de los lentes que conforman su sistema óptico. Estos parámetros se introducen en una matriz de 3x3 que representa una proyección en perspectiva. Esta transformación calcula la posición de las proyecciones en el plano de la imagen de los puntos tridimensionales. Además, convierte las unidades de longitud de metros a pixeles, y de esta forma poder modelar la captura de las imágenes del sensor CCD, el cual por naturaleza de la digitalización convierte las unidades de los puntos. A dicha transformación se le conoce como modelo de cámara ideal y se representa como:

$$u_1 = \frac{f x_{c1}}{s_x x_{c3}} + u_{o1}, \quad u_2 = \frac{f x_{c2}}{s_y x_{c3}} + u_{o2} \tag{3.4}$$

donde

- $(u_1, u_2)^T$ es un punto en la imagen.
- f es la distancia del centro de la cámara al plano de la imagen y es igual a la distancia focal del sistema óptico si el objeto se encuentra en el infinito.
- $(u_{o1}, u_{o2})^T$ es el punto de intersección del eje óptico de la cámara con el plano de la imagen y se conoce como punto principal.
- s_x y s_y son las distancias entre los centros de las celdas del sensor en metros o bien, el tamaño de los pixeles en cada una de sus dimensiones.

El modelo de la cámara pin-hole con los parámetros internos y externos es un conjunto de transformaciones lineales en cascada que cambian el marco de referencia de los puntos en la escena, primero del sistema del mundo al sistema de la cámara, y de este último al sistema de la imagen. El modelo completo se presenta a continuación:

$$\lambda = \begin{bmatrix} u_1 \\ u_2 \\ 1 \end{bmatrix} = \begin{bmatrix} \epsilon f/s_x & \gamma & u_{o1} \\ 0 & f/s_y & u_{o2} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & t \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} x_{w1} \\ x_{w2} \\ x_{w3} \\ 1 \end{bmatrix} \tag{3.5}$$

reescribiendo algunas matrices queda:

$$\lambda \begin{bmatrix} u_1 \\ u_2 \\ 1 \end{bmatrix} = K D_u G \begin{bmatrix} x_{w1} \\ x_{w2} \\ x_{w3} \\ 1 \end{bmatrix} \tag{3.6}$$

- K es la matriz de calibración o de parámetros.
- D_u es una matriz que permite realizar la multiplicación de matrices de diferente dimensión y comúnmente se le conoce como matriz dummy.
- G es la matriz correspondiente a la transformación euclideana de los parámetros extrínsecos.
- ϵ es un factor de escala conocido como razón de aspecto y se refiere a la proporción entre el ancho y el alto de los pixeles.
- γ representa el efecto de asimetría en la cuadrícula de pixeles en el sensor de la cámara ocasionado por los primeros capturadores de imágenes, en los cuales el escaneo horizontal era línea por línea del sensor. En los modernos capturadores la lectura se hace en paralelo, por lo tanto la asimetría es considerada nula.

A la matriz resultante de la multiplicación de la matrices de parámetros instrínsecos y extrínsecos se le conoce como matriz de proyección en perspectiva: Π, y relaciona a un punto en el espacio proyectivo: P^3, con otro punto en el plano proyectivo: P^2. La representación a bloques de Π es:

$$\Pi = K D_u G \tag{3.7}$$

3.2.2. Matlab Calib-tool

Calib-tool [32] es una herramienta para calibración de cámaras desarrollada en MATLAB, que cuenta con una interface gráfica de usuario simple y amigable para uso no restringido a personas expertas en el área de visión computacional. Esta implementación es utilizada en el presente trabajo de tesis para modelar la cámara y con ayuda de esta calibración estimar el plano del piso y adicionalmente

corregir las distorsiones de las imágenes capturadas.
Esta herramienta de MATLAB cuenta con la implementación de un conjunto de métodos, todos publicados independientemente, que combinados alcanzan un alto grado de robustez y confiabilidad en su estimación de parámetros. Los métodos más importados son los mencionados en la sección anterior: [34], [35] y [36].
Calib-tool requiere un mínimo de 2 imágenes (sin número máximo) de un patrón de referencia plano como el de un tablero de ajedrez, sin embargo, mientras mayor sea el número de imágenes el error será menor en las estimaciones. La herramienta cuenta con una etapa de inicialización en la cual se estiman las homografías usando el algoritmo de Zhang.
El cálculo de los parámetros intrínsecos se realiza a partir de homografías y se utilizan las propiedades de ortogonalidad de los puntos de fuga de líneas perpendiculares en la escena para estimarlos, en esta etapa Calib-tool utiliza el algoritmo Katani para lograr dicha estimación. Posteriormente con esos parámetros internos y las homografías se extraen los parámetros externos. La etapa de refinamiento está basada en el algoritmo de Zhang en el apartado de: *estimación de máxima verosimilitud*, donde se considera que cada punto característico en las imágenes del patrón está corrompido por una distribución de ruido independiente y forma común, con el cálculo de la diferencia en posición de estos puntos con los proyectados en las imágenes se obtiene una función a minimizar mediante el algoritmo de Levenberg-Marquardt [37].
Calib-tool no considera la distorsión en su estimación inicial, pero en la etapa de optimización no-lineal utiliza un modelo de parámetros no intrínsecos propuestos en [38], el cual considera un modelo polinomial par de distorsión radial de sexto orden, más de dos coeficientes extras que corresponden a la distorsión tangencial. De esta forma el toolbox de MATLAB calcula los parámetros intrínsecos y extrínsecos, adicionalmente provee de un modelo de distorsión radial y tangencial que puede ser usado para corregir todas las imágenes capturadas con la misma cámara.

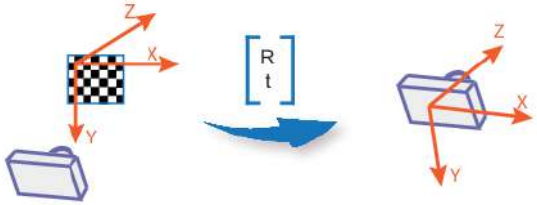

Figura 3.3: Calibración mediante tablero de ajedrez

3.3. Optimización numérica

El interés en los procesos de optimización dio un paso importante con la llegada de las computadoras digitales a principios de los cincuentas. En años recientes las técnicas de optimización han avanzado rápidamente y se ha logrado un considerable progreso, de igual forma, con el paso de los años las computadoras se

han vuelto más veloces, más versátiles y más eficientes, como consecuencia de ello ahora es posible resolver problemas de optimización los cuales se creía hace unos años eran intratables.
Al proceso de optimización se le conoce como el proceso de obtener lo "mejor", si es que es posible determinar lo que es bueno o malo en un problema. En la práctica, alguien siempre desea el mayor o mejor (por ejemplo salario recibido) o lo menor o mínimo (por ejemplo gastos en el hogar). Por lo tanto la palabra óptimo es tomada como lo máximo o mínimo dependiendo de las circunstancias, mientras optimizar hace referencia el proceso de obtener lo óptimo.
Los problemas de optimización son necesarios en bastantes disciplinas, como en la ingeniería, física, matemáticas, economía, administración, comercio e incluso política. Por ejemplo en ingeniería (área típica de aplicación de optimización) se utiliza en la modelación, caracterización en el diseño de dispositivos, circuitos y sistemas.
La mayoría de los problemas en la vida cotidiana tienen varias soluciones y ocasionalmente un infinito número de soluciones. Suponiendo que el problema tiene más de una solución, la optimización puede ser lograda encontrando la mejor solución del problema en términos de algunos criterios de desempeño. Si el problema admite una única solución (un conjunto único de parámetros es aceptable) entonces la optimización no puede ser aplicada.
Existen los siguientes enfoques para realizar optimización:

- Métodos analíticos
- Métodos gráficos
- Métodos experimentales
- Métodos numéricos

3.3.1. Métodos numéricos

El enfoque más importante para realizar optimización está basado en métodos numéricos. En este enfoque se utilizan procedimientos númericos iterativos para generar una serie de soluciones progresivamente mejores al problema de optimización, comenzando con una estimación inicial para la solución. El proceso acaba cuando se satisface algún criterio de convergencia, por ejemplo, cuando los cambios en las variables independientes o el criterio de desempeño de iteración a iteración llega a ser insignificante.
Los métodos numéricos pueden ser usados para resolver problemas de optimización altamente complejos los cuales no se pueden resolver analíticamente. Además, éstos pueden ser programados en una computadora y en consecuencia los métodos numéricos han remplazado la mayoría de los otros enfoques de optimización.
El problema más básico de optimización es ajustar las variables $x_1, x_2, ..., x_n$ de tal forma que minimice la cantidad F. Este problema puede ser establecido matemáticamente como:

$$minimizar \qquad F = f(x_1, x_3, ..., x_n) \tag{3.8}$$

Donde la cantidad F es normalmente conocida como el objetivo o la función costo. La función objetivo podría depender de un número extenso de variables, por ejemplo 100 o más. Normalmente se utiliza notación matricial para simplificar la notación de las variables. Si x es un vector columna con los elementos $x_1, x_2, ..., x_n$, la transpuesta de x, puede ser expresada como el vector fila:

$$x^T = [x_1\ x_2\ ...\ x_n] \tag{3.9}$$

En algunas aplicaciones un número de distintas funciones de x necesitan ser optimizadas simultáneamente, por ejemplo, si se necesita resolver el siguiente sistema de ecuaciones no lineales simultáneamente:

$$f_i(x) = 0 \quad para \quad i = 0, 1, 2, ..., m \tag{3.10}$$

Entonces se busca un vector x el cual reduzca todas las $f_i(x)$ a cero simultáneamente. En tal problema las funciones a ser optimizadas pueden ser utilizadas para construir un vector:

$$F(x) = [f_1(x) f_2(x) ... f_m(x)]^T \tag{3.11}$$

El problema puede ser resuelto encontrando un punto $x = x^*$ de tal forma que $F(x^*) = 0$. A menudo un punto x^* que reduce todas las $f_i(x)$ a cero simultáneamente podría no existir, pero una aproximación $F(x^*) = 0$ podría estar disponible lo cual en la práctica es suficiente.

3.3.2. Mínimos cuadrados

Como se mencionó al final de la sección anterior existen problemas que pueden ser resueltos definiendo una función objetivo adecuada en términos de los elementos de la función $F(x)$. La función objetivo debe ser una cantidad escalar y su optimización debe llevar a las optimizaciones simultáneas de los elementos (funciones) de $F(x)$, por lo tanto se debe utilizar algún tipo de norma, esto quiere decir que una función objetivo puede ser definida en términos de la norma L_p:

$$F(x) \equiv L_p = \{\sum_{i=1}^{m} |f_i(x)|^p\}^{1/p} \tag{3.12}$$

Donde p es un entero. Si $p = 2$ la norma euclideana es minimizada y si la raiz cuadrada es omitida, la suma de los cuadrados es minimizada.

$$F(x) \equiv L_2 = \{\sum_{i=1}^{m} |f_i(x)|^2\}^{1/2} \tag{3.13}$$

A tal problema se le conoce comúnmente como el problema de los **mínimos cuadrados**.
Uno de los métodos más populares de optimización numérica para resolver problemas de mínimos cuadrados no lineales es el de **Levenberg-Marquardt**. Sea el problema a solucionar:

$$F(x) = \frac{1}{2} \sum_{i=1}^{m} [f_i(x)]^2 \tag{3.14}$$

y $J_i(x)$ denota el jacobiano de $f_i(x)$, entonces el método de Levenberg-Marquardt busca en la dirección dada por la solución de p de las ecuaciones:

$$(J_k^T J_k + \lambda_k I)p_k = -J_k^T, \tag{3.15}$$

donde la λ_k es un escalar no negativo e I es la matriz identidad. Dado que el algoritmo de Levenberg-Marquardt es del tipo de métodos con región de confianza se debe elegir un buen valor λ_k (o Δ) para asegurar un descenso en las funciones.

3.4. Detector de rostros de Viola y Jones

Como se mencionó anteriormente para estimar la mirada con precisión se necesita un sistema de detección y seguimiento de ojos, además de múltiples cámaras y un sistema de estimación de pose de la cabeza, cabe destacar que el proyecto que se presenta más que estimar la mirada con precisión, se pretende estimar la mirada general de las personas representada como una región en un plano virtual enfrente de ellas.
Como primer paso para la estimación de la mirada se debe detectar dónde se encuentran los rostros en la escena, ya que el sistema consiste de varias etapas, se requiere que la detección se haga lo más rápido y eficiente posible, por lo tanto se decidió utilizar uno de los algoritmos de detección más populares y rápidos que existen: el clasificador basado en características tipo Haar en cascada. Este método fue propuesto por Paul Viola y Michael Jones en 2001 en el artículo Rapid Object Detection using a Boosted Cascade of Simple Features [33], el método consta principalmente de las secciones presentadas a continuación.

3.4.1. Imagen integral

En esta etapa se utiliza una representación intermedia de la imagen conocida como imagen integral o también como tablas de áreas sumadas [40]. En la imagen integral a cada pixel de la imagen se le asocia un valor representando la suma de los valores de los píxeles que se encuentran arriba y a la izquierda de él, así como también se le suma su valor de píxel. Este tipo de representación tiene como principal ventaja la rápida estimación del valor de los píxeles de subregiones de la imagen. El detector de Viola y Jones clasifica las imágenes basado en el valor de

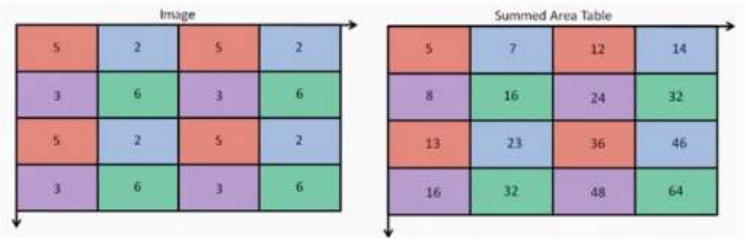

Figura 3.4: Ejemplo de cálculo de la imagen integral

simples características, los sistemas basados en características operan más rápido

que los basados en píxeles. La imagen integral puede ser usada para estimar el valor de características tipo Haar, este tipo de características se visualizan como rectángulos adyacentes blancos y negros, el valor que generan se calcula de la diferencia de la suma de los pixeles en el área blanca menos la suma de los del área negra, la adyacencia que presentan los rectángulos permite reutilizar algunos valores. El conjunto de características rectangulares ofrece una buena representación de las imágenes.

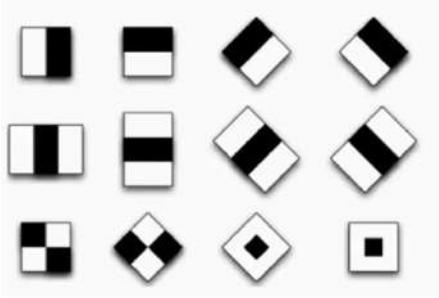

Figura 3.5: Características Haar

3.4.2. Algoritmo AdaBoost

Los métodos de Boosting [41] básicamente consisten en combinar la eficiencia de varios clasificadores débiles para producir un poderoso consejo o comité clasificador. El algoritmo Adaboost es el método más usado y práctico de los algoritmos de Boosting. La idea general del AdaBoost consiste en que los ejemplos que son clasificados erróneamente obtienen mayor peso en las iteraciones siguientes, esto significa que los ejemplos que se encuentran cerca de la frontera de decisión son generalmente más difíciles de clasificar y por lo tanto se les asigna mayores pesos después de unas cuantas iteraciones. La idea de reasignación de pesos en el conjunto de entrenamiento es esencial en los métodos de Boosting.
El detector de Viola y Jones se entrena con conjuntos de imágenes etiquetadas como positivas y negativas, el algoritmo de AdaBoost se utiliza para entrenar al clasificador y seleccionar un conjunto pequeño con las características Haar más importantes de una amplia biblioteca de posibles características. Como resultado final del proceso de entrenamiento el algoritmo de AdaBoost produce un clasificador robusto que tiene la forma de un perceptrón, una combinación de clasificadores débiles en el que cada clasificador tiene asociado una sola característica Haar y un umbral. Resumiendo lo anteriormente mencionado, la utilización del algoritmo de clasificación AdaBoost permite encontrar las características que mejor separan los ejemplos positivos de los negativos.
Una de las ventajas clave por la cual AdaBoost fue elegido como algoritmo seleccionador de características por los investigadores Viola y Jones en su detector, es la increíble velocidad que presenta a comparación de otros métodos publicados hasta esa fecha como en [42], comparando ambos métodos mediante un clasificador de 200 características, con Adaboost puede ser construido en alrededor 10^{11}

operaciones de procesador mientras que con [42] se logra construir en 10^{16} operaciones.
Durante los experimentos realizados por Viola y Jones se notó que las primeras características seleccionadas por AdaBoost tenían bastante sentido y son fáciles de interpretar, por ejemplo, una de esas característica parecía enfocarse en la propiedad de que la región de los ojos es por lo regular más oscura que la región de la nariz y las mejillas.

Figura 3.6: Características tipo Haar sobrepuestas

3.4.3. Filtro con estructura en cascada.

El último componente importante del Detector de Viola y Jones es la estructura en cascada que se forma con la combinación de unos cuantos clasificadores mejorados (boosted), estos clasificadores son usados para rechazar la mayoría de las subregiones negativas (sin rostros de personas) y poner atención en regiones de la imagen más prometedoras, es decir, las regiones que presenten el rostro de alguna persona. Este método es muy eficiente ya que la mayoría de las sub-ventanas o subregiones son rechazadas en etapas tempranas. Las sub-ventanas de la entrada del detector pasan a través de una serie de nodos, cada nodo toma una decisión binaria y dependiendo de la decisión la subregión se mueve al siguiente nodo o se rechaza. El número de clasificadores débiles presentes en cada nodo incrementa conforme la sub-ventana se va moviendo a los siguientes nodos, por ejemplo, en el primer nodo contiene un clasificador débil, el segundo 10, el tercero 25, el cuarto 50 y así sucesivamente, figura 3.7. Teniendo pocos clasificadores en etapas tempranas es otra forma de mejorar la velocidad del detector y esto de alguna forma compensa el costo de evaluar cada característica Haar en diferentes escalas y posiciones de la imagen.

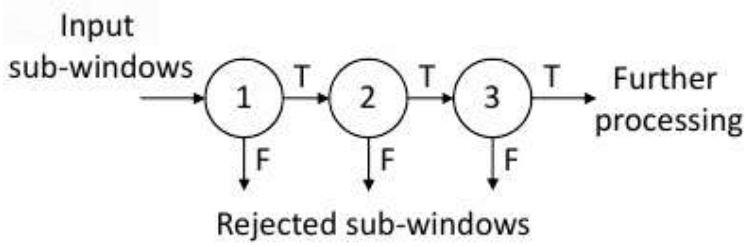

Figura 3.7: Diagrama de nodos

El detector de rostros descrito en esta sección tiene incontables aplicaciones y debido la velocidad de la detección puede ser utilizado en aplicaciones que requieran detección en tiempo real. Uno de los aspectos más interesantes del método de Viola y Jones es que no se limita a la detección de rostros, puede ser modificado para sistemas detectores de otro tipo de patrones en las imágenes, por ejemplo, automóviles, peatones y recientemente utilizado para la detección de la enfermedad de Chagas, analizando las muestras de sangre de los infectados [43].

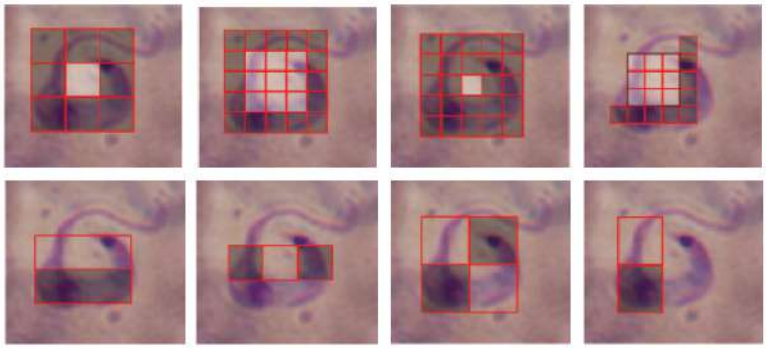

Figura 3.8: Detección de chagas en la sangre mediante el método de Viola y Jones

3.5. Descriptores HOG

Se comenzará esta sección describiendo de manera general qué es un descriptor de características "feature descriptor" y luego se detallará en específico los descriptores tipo HOG. Un descriptor de características de una imagen o de una región de ella es aquel que simplifica la representación de dicha imagen extrayendo información útil y desechando toda la información extraña.
Típicamente un descriptor convierte una imagen de tamaño igual a *ancho* x *largo* x 3 (canales) a un vector de características de longitud n, en el artículo original acerca de HOG [45] se utiliza como ejemplo una imagen de tamaño 64 x 128 x 3 por lo que la longitud del vector n es de 3780. A partir de ahora se utilizará como ejemplo las mismas dimensiones que las utilizadas en el artículo original, sin embargo, hay que tener en cuenta que se puede utilizar HOG para cualquier dimensión en las imágenes.
Al utilizar un descriptor de características en el presente proyecto se busca que cumpla con los siguientes requisitos:

- Ayudar a que las imágenes con rostros (pueden ser de diferentes personas) que miran la misma región en pantalla sean agrupados en el mismo conjunto.
- Ayudar a que las imágenes con rostros que miran diferentes regiones en pantalla sean agrupados en conjuntos diferentes.

En los descriptores HOG la distribución (histograma) de las direcciones de los gradientes (gradientes orientados) son usados como características. Los gradientes (derivadas en x y y) de una imagen son útiles debido a que la magnitud de los

gradientes es mayor alrededor de los bordes y las esquinas (regiones con cambios abruptos de intensidad) y es bien sabido que los bordes y las esquinas ofrecen más información sobre la forma del objeto que las regiones planas.

3.5.1. Cálculo del histograma de gradientes orientados

A continuación se describirá como se realiza el cálculo del descriptor HOG tomando como ejemplo una región de la imagen 3.9 para detectar peatones, como se realiza en [45].

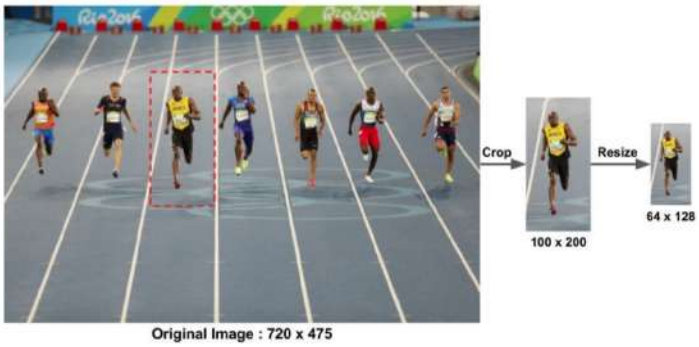

Figura 3.9: Imagen de ejemplo para HOG

Paso 1: Preprocesamiento y normalización

El descriptor HOG es calculado en una región de la imagen de 64x128, generalmente dichas regiones son analizadas a multiples escalas y en muchas locaciones, la única restricción es que las regiones siendo analizadas tengan una proporción fija. En este caso las regiones necesitan tener una proporción de 1:2. Por ejemplo, ellas pueden ser 100x200, 128x256 pero no 101x205.
En la figura 3.9 se puede observar que se eligió una imagen de 720x475 y en ella se seleccionón una región de 100x200 para calcular el descriptor HOG, esta región se redimensionó a 64x128.

Paso 2: Calcular los gradientes de la imagen

Para calcular el descriptor HOG es necesario calcular primero los gradientes horizontales y verticales, esto se logra fácilmente filtrando la imagen con los siguientes kernels:

Figura 3.10: Kernels

Posteriormente al filtrado de la imagen se calcula la magnitud y dirección del gradiente con las siguientes fórmulas:

$$g - \sqrt{g_x^2 + g_y^2} \tag{3.16}$$

$$\theta - arctan(\frac{g_y}{g_x}) \tag{3.17}$$

En la figura 3.11 se observan los resultados de los gradientes: Nótese que la

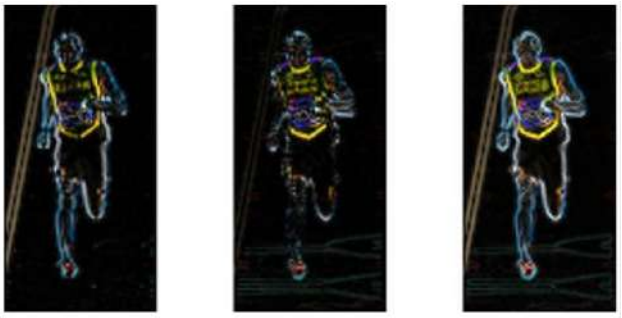

Figura 3.11: Izquierda: valor absoluto del gradiente x. Centro: valor absoluto del gradiente. Derecha: magnitud del gradiente

magnitud del gradiente se intensifica donde sea que haya cambios bruscos de intensidad y ninguno de los gradientes anteriores se intensifica en las regiones suaves de la imagen.
Los gradientes de la imagen remueven bastante información no esencial y destaca los contornos, por ejemplo, en la imagen de los gradientes 3.11 se pudiera deducir facilmente que es una persona corriendo.
En cada pixel de HOG el gradiente tiene una magnitud y una dirección. La magnitud del gradiente de un pixel es el máximo de la magnitud de los gradientes los tres canales (en imágenes RGB), y el ángulo es el ángulo correspondiente al máximo gradiente.

Paso 3: Cálculo del Histograma de gradientes

En este paso la región seleccionada es dividida en celdas de 8x8 y un histograma de gradientes es calculado para cada celda de 8x8. El criterio para la selección de la dimensión de las celdas (8x8), se realiza en base a la dimensión del objeto en la imagen que se quiere analizar, tomando el ejemplo del corredor (cuya dimensión es de 64x128), las celdas de 8x8 son lo suficientemente grandes para capturar características interesantes (el rostro, la parte superior de la cabeza, etc.)

En la figura 3.12 se presenta una celda del objeto analizado con los valores de sus gradientes, en el cuadro del centro se pueden apreciar los pixeles en una celda con flechas sobrepuestas mostrando la dirección y magnitud de los gradientes. Nótese como la dirección de la flecha apunta hacia la dirección del cambio de intensidad y la magnitud muestra que tan grande es la diferencia.
A continuación se crea el histograma de gradientes de las celdas de 8x8, el histograma contiene 9 bins (contenedores o celdas) correspondientes a los ángulos

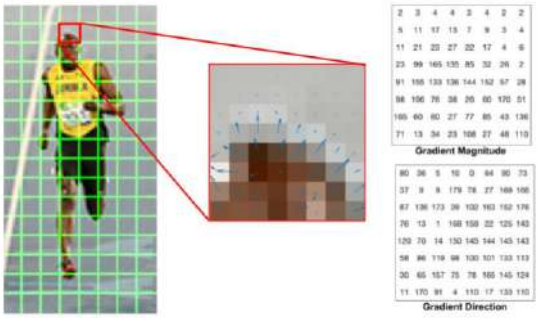

Figura 3.12: Valores de los gradientes de celda 8x8

0, 20, 40...160. En 3.13 se encuentra el proceso del cálculo del histograma, la selección del bin está basado en la dirección y el valor que va dentro del bin es la magnitud del gradiente.

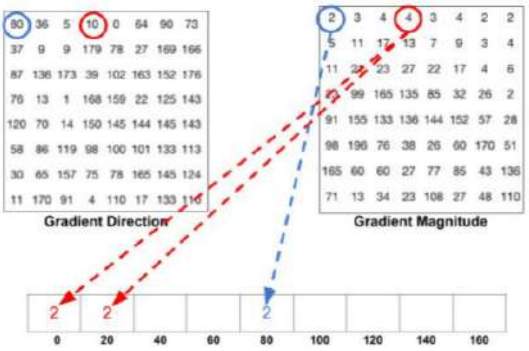

Figura 3.13: Cálculo histograma de gradientes

La contribución de todos los pixeles en la celda de 8x8 es sumada para crear el histograma de 9 bins, para la celda de la figura 3.12 el histograma final se encuentra en la imagen 3.14, se puede notar que hay más elementos en los bins cerca de los 0 y 180 grados, esto indica que los gradientes están apuntando arriba o abajo.

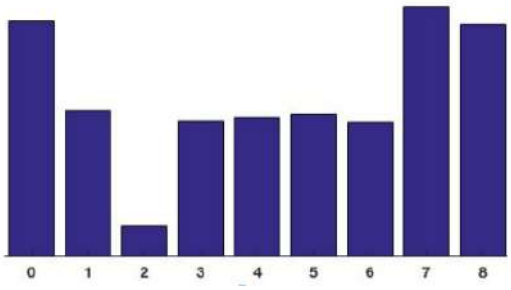

Figura 3.14: Histograma final de celda de 8x8

Paso 4: Normalización del bloque de 16x16

El histograma creado en las etapas anteriores está basado en los gradientes de la imagen, sin embargo, los gradientes en una imagen son sensibles a las variaciones de luz, para contrarrestar este problema en el artículo original de HOG el autor recomienda normalizar el histograma, él propone cuatro diferentes tipos de normalizaciones donde la más común es la normalización con la norma L2 o euclídea. Además, Dalal sugiere realizar la normalización en bloques mayores del histograma correspondiente a la celda de 8x8 ya que de acuerdo a sus experimentos se obtienen resultados superiores.
En el ejemplo del corredor y siguiendo las instrucciones de la investigación original, se concatenan cuatro histogramas de 8x8 (histograma de 9x1) para formar un bloque de 16x16 y así obtener un vector normalizado de 36x1. Luego, la ventana (bloque de 16x16) es desplazado cada 8 pixeles y de nuevo se calcula el vector normalizado. En la figura 3.15 se puede apreciar el proceso anterior del desplazamiento de la ventana.

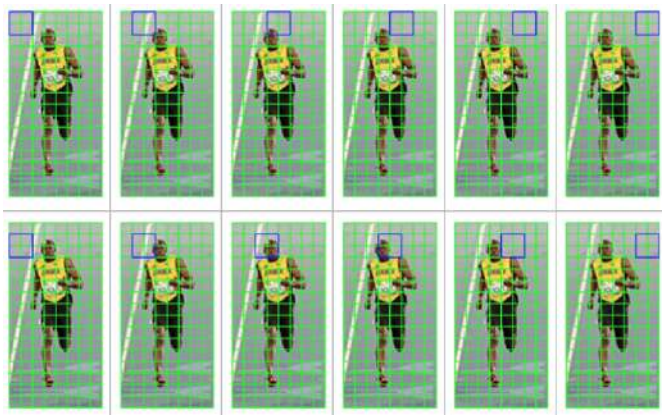

Figura 3.15: Normalización llevada a cabo desplazando los bloques de 16x16

Paso 5. Cálculo del vector de características HOG

El vector final de características para la región seleccionada se obtiene de la concatenación de los vectores de dimensión de 36x1, tomando en consideración que hay 7 posiciones horizontales posibles de la región de la imagen, 15 verticales y cada bloque tiene un longitud de 36 elementos, el vector final tiene: 7*15*36=3780 elementos.
Para la visualización únicamente se grafican los histogramas normalizados de 9x1 en las celdas de 8x8. En la imagen 3.16 se puede percibir que las direcciones dominantes del histograma capturan la forma de la persona.

3.6. Redes neuronales artificiales

Una red neuronal artificial es un paradigma de aprendizaje automático inspirado en las neuronas de los sistemas nerviosos de los animales, en donde las

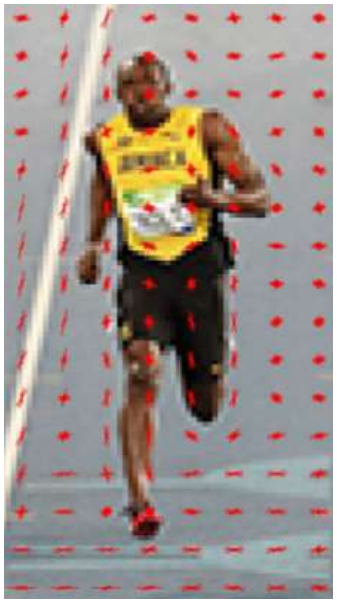

Figura 3.16: Direcciones dominantes

entradas son multiplicadas por unos variables llamadas *pesos*, lo cual resulta en una *activación* y forman la salida de la red. Las investigaciones en redes neuronales comenzaron en 1943, cuando McCulloch y Pitts dieron una definición formal de una neurona en el artículo "A logical Calculus of the ideas Immanent in Neurons Activity" [5].
En 1958 Rosenblatt inventó el perceptrón, el cual es una red neuronal simple de conexiones hacia adelante (feedforward) [6]. La desventaja del algoritmo del perceptrón es que solo converge en conjuntos de datos linealmente separables y no es capaz de solucionar problemas no lineales como por ejemplo el problema XOR. Lo anterior fue probado en 1958 por Minsky y Papert en su monografía "Perceptrons"[7], sin embargo, ellos también demostraron que esta desventaja puede ser contrarrestada con una arquitectura de dos capas conectadas hacia adelante.
Fue hasta 1985 cuando Rumelhart, Hinton y Williams [8] presentaron una regla de aprendizaje para las redes neuronales artificiales con unidades escondidas en su trabajo: "Learning Internal Representations by Error Propagation". El descubrimiento original de la propagación hacia atrás (backpropagation) es atribuido en realidad a Werbos, quien describió el algoritmo en su tesis doctoral en 1974 en la universidad de Harvard.
A continuación se describe el algoritmo de propagación hacia atrás.

1. Inicializar los pesos con valores aleatorios.

2. Se introducen los valores en el vector de entrada de la red neuronal.

3. Evaluar la salida de la red después de una propagación hacia adelante de la señal.

4. Calcular $\delta_j = (y_j - d_j)$ donde d_j es la etiqueta salida de la neurona j y y_j

es la salida real: $y_j = g(\sum_i w_{ij}x_i) = (1 + e^{-\sum_i w_{ij}x_i})^{-1}$, (en caso de que la función de activación sea de tipo sigmoide).

5. Para todas las demás neuronas (desde la primera hasta la última capa) calcular $\delta_j = \sum_k w_{jk}g'(x)\delta_k$, donde δ_k es el δ_j de la subsiguiente capa y $g'(x) = y_k(1 - y_k)$

6. Actualizar los pesos con $w_{ij}(t + 1) = w_{ij}(t) - \eta y_i y_j(1 - y_j)\delta_j$, donde η es la tasa de aprendizaje.

7. Se verifica si se ha cumplido uno de los criterios de terminación. Ir al paso 2 para un número fijo de iteraciones o un error.

El error de la red está definido como:

$$E = \frac{1}{2}\sum_{j=1}^{m}(d_j - y_j)^2 \tag{3.18}$$

Las redes neuronales son utilizadas en el sistema propuesto para la determinar la región en pantalla que las personas están observando, para ello se tomará como vector de entrada de la red neuronal las imágenes de los rostros de las personas y otros datos 3D de ellas.

Capítulo 4

Metodología

En el presente capítulo se detallarán las etapas y procesos necesarios del sistema para lograr la estimación de la mirada de las personas, esto incluye desde los métodos para el cálculo de los ángulos de la mirada a un objeto en específico, el proceso de generación y selección de las marcas en el piso donde se colocan las personas y las marcas en la pantalla para la captura de datos, el proceso de optimización numérica para minimizar el error en la matriz de rotación, cálculo de la ecuación del plano del piso, la extracción de las características del rostro con HOG y finalmente como se implementa la red neuronal.

4.1. Descripción general del sistema

El sistema de estimación de la mirada que se propone en este proyecto de tesis funciona para posiciones discretas, es decir, se estiman regiones del plano enfrente de las personas. La metodología del sistema consta de dos fases principales: la fase de calibración del experimento y la fase de clasificación de la mirada.
El objetivo final de la fase de calibración es generar una secuencia de pares de puntos en el piso, en los cuales las personas deben colocarse para capturar fotografías de sus caras observando marcas en una pantalla o pared enfrente de ellas, las imágenes capturadas se utilizan para entrenar el algoritmo de aprendizaje automático en la fase de clasificación. Cabe destacar que se debe conocer con precisión donde se encuentran sus rostros y del objeto que miran con respecto a un marco de referencia centrado en la cámara.
La fase de calibración tiene las siguientes etapas:

- **Análisis de la geometría en la escena de experimentación**.- Se analizan tres aspectos: la ubicación 3D de la cabeza, el objeto que deben mirar las personas y el comportamiento que presentan al desplazarlos, esto se hace con la finalidad de obtener las ecuaciones que describen la mirada y conocer los límites de experimentación, es decir, hasta qué distancias de la cámara vale la pena realizar la captura de datos.

- **Obtención de la ecuación del plano del piso**.- Para obtener la ecuación del plano donde se realizan los experimentos en primer lugar se debe calibrar

la cámara con la que se trabaja, ya que a partir de ella se obtiene la matriz de parámetros intrínsecos K. A continuación se aplican técnicas de visión computacional para hallar una matriz de homografía entre un plano 3D y una imagen de dicho plano, se descompone la homografía y con ayuda de la matriz K se obtiene la rotación y traslación que existe entre la cámara y el piso, a partir de estas matrices es sencillo obtener la ecuación que describe el plano del piso.

- **Optimización numérica**.- Durante la metodología y experimentación se demostrará que la ecuación del plano obtenida en la etapa anterior resulta inexacta, esto se debe a que hay una distancia lejana entre la cámara y el plano, por lo que es necesario aplicar una corrección a la imagen y un algoritmo de optimización numérica para minimizar el error de dicha ecuación.

- **Algoritmos de generación de secuencia óptima de marcas de experimentación**.- Conociendo donde se encuentra el plano del piso se procede a generar una secuencia de puntos que yazgan en él, en los que exista una variación amplia (en los pares consecutivos) en los ejes X y Z, con el objetivo de que esta variación produzca pares de rotaciones amplias en la mirada, con ello se pretende obtener un conjunto de datos de entrenamiento heterogéneo.

En la fase de clasificación se utilizan las marcas que se producen al final de la fase anterior para realizar la captura de los datos (fotografías) de entrenamiento de la red neuronal. Luego de entrenar la red se presentan tres etapas principales para lograr la clasificación de la mirada, véase la figura 4.1:

- **Detección del rostro**.- Se utiliza el algoritmo detector de rostros de Viola y Jones para localizar el rostro de la persona en la imagen.

- **Extracción de características del rostro**.- Se extraen las características del rostro que incluyen la posición en 3D y las características de la imagen, las cuales son utilizadas por el clasificador.

- **Clasificador**.- Se encarga de producir en la salida una región en la pantalla que representa lo que está observando la persona detectada en etapas previas, el algoritmo de clasificación es una red neuronal artificial de retropropagación entrenada con anterioridad.

A continuación se describirán con detalle cada una de estas etapas.

4.2. Análisis de la geometría en la escena de experimentación

Uno de los problemas que hay que tomar en cuenta y definir en el proyecto antes de realizar los experimentos, es describir la relación que existe entre la

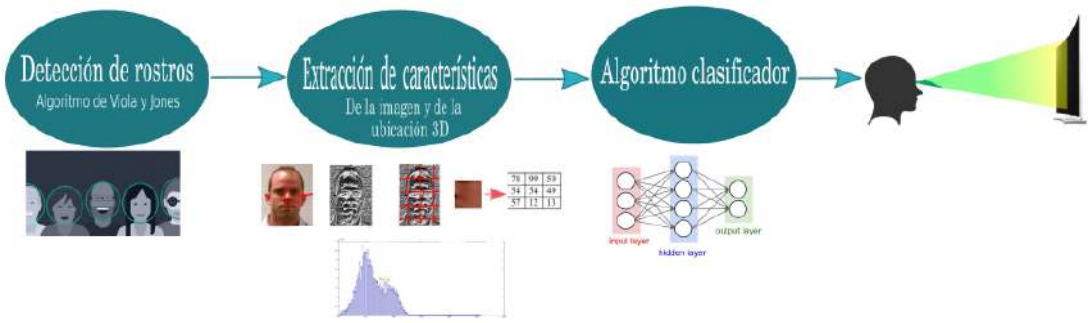

Figura 4.1: Etapas de la clasificación

ubicación (en metros) de lo que está mirando el sujeto de estudio y la de la pose de su cabeza medida en ángulos. Esto involucra considerar diferentes características como: la estatura de la persona, la distancia de la persona a la pantalla, cuál es el marco de referencia, etc. Esta sección consiste en describir cómo se da la ecuación que representa la relación entre el objeto desplegado en pantalla y la orientación de la cabeza, primero se detallan las características geométricas del escenario donde se realiza la captura de datos. Todas las unidades de medida de las variables de distancia y posición están en metros.

La pantalla que observa el sujeto del experimento se coloca en una pared a una

Figura 4.2: Parámetros en la pantalla

distancia de L_y metros del piso, tiene unas dimensiones de W de ancho y L de alto: figura 4.2. El sistema de coordenadas tiene su origen en la cámara y se ubica encima de la pantalla y a la mitad del ancho W, la recta que se encuentra a lo largo de W es paralela al eje X y la que está a lo largo de L es paralela al eje Y. Durante los experimentos las personas estarían paradas en un mismo plano (piso) y enfrente de la pantalla a una distancia variable D_z con respecto al eje Z. Hay un punto en específico de interés en el rostro de las personas y es el que se utiliza para realizar el análisis, este punto se define como P y se encuentra a la mitad de la linea que une los ojos, figura 4.3. El vector $\vec{v}$ sale del punto P y es ortogonal al plano de la cara, el punto P se encuentra a una distancia H_y con respecto al piso y es dependiente de la estatura de la persona. Otro parámetro que se debe mencionar es el que hace referencia a la distancia que hay entre la persona y el origen sobre el eje X, es decir, la componente X del punto P de la persona y

lo denotaremos como a. Todas las variables anteriores se pueden apreciar en la imagen 4.4

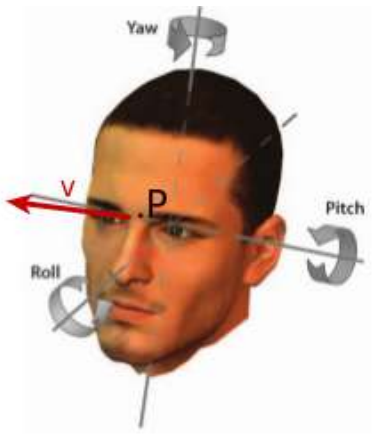

Figura 4.3: Tipos de movimientos de la cabeza

Como se puede observar en la figura 4.3 las personas tenemos tres tipos de movimiento de la cabeza: yaw, pitch y roll, sin embargo, para el análisis de este proyecto solo se toman en cuenta el movimiento de yaw y pitch, debido a que estos son los movimientos que se dan cuando una persona mueve su cabeza para observar un objeto en dos dimensiones desplazándose en un plano.
El objeto de la pantalla que la persona observa es representado como S y el vector $\vec{v}$ representa la mirada de la persona o lo que es lo mismo $\vec{PS}$, esto quiere decir que si se proyecta el vector sobre la misma dirección hacia la pantalla debe intersectar S.
El objeto S se desplazará a través de toda la pantalla (X, Y) metros, en el marco de referencia de la imagen el origen es la esquina superior izquierda de la pantalla, a esta esquina le corresponde la coordenada $(0, 0)$, (X, Y) es la esquina inferior derecha y $(\frac{X}{2}, \frac{Y}{2})$ el centro de la pantalla. La pose de la cabeza se mide mediante dos ángulos: ϕ_x y ϕ_y, donde:

- ϕ_y.- Observando hacia el plano $Y - Z$ representa que tanto se desvía el vector $\vec{v}$.
- ϕ_x.- Observando hacia el plano $X - Z$ (desde arriba de la persona y la pantalla) representa que tanto se desvía el vector $\vec{v}$.

De igual manera el ángulo ϕ_x indica el movimiento yaw que la persona realiza al mirar como se traslada el objeto S y el ángulo ϕ_y el movimiento pitch.

4.2.1. Cálculo de ϕ_y y ϕ_x. Primer caso

Una vez definido el escenario y los aspectos a tomar en cuenta (figura 4.4), ahora falta demostrar cuál es la relación que se puede hallar entre la posición del objeto en la pantalla y la pose de la cabeza a través de los ángulos ϕ_x y ϕ_y, el cálculo de dichos ángulos depende de la orientación de la cámara con respecto al

piso, a continuación se presentan dos formas de obtenerlos.
El primer caso es con la cámara colocada sobre la pantalla de tal forma que su

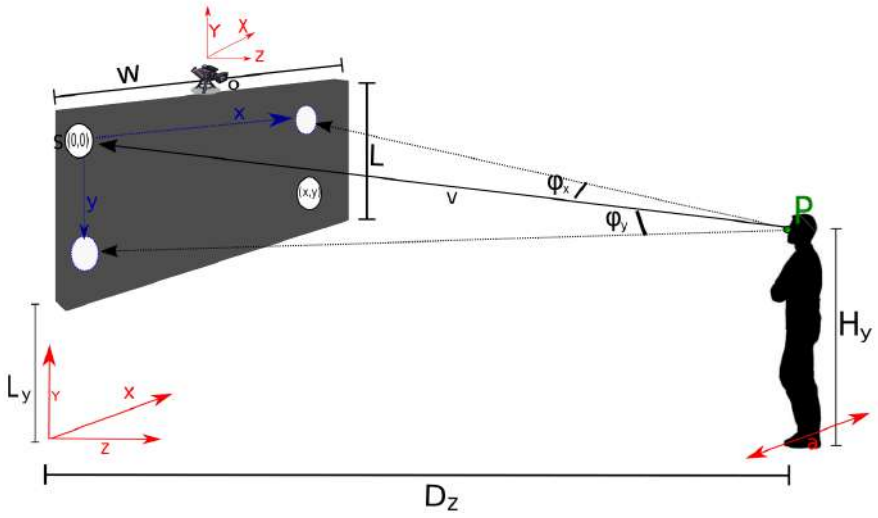

Figura 4.4: Escenario y parámetros principales

eje Z es paralelo al plano del piso, esto quiere decir que el eje Y de la cámara yace sobre el plano de la pantalla y forma un ángulo recto con el piso.
El análisis comienza observando la escena desde el plano $Y - Z$, primero se definen dónde se encuentran con respecto al sistema de coordenadas los puntos más importantes para el análisis: S y P. Como se había mencionado previamente el sistema de coordenadas se encuentra centrado en la cámara, a la mitad de la pantalla con respecto al eje X y a una pequeña altura c_y con respecto al eje Y, suponiendo que el objeto a observar está en la coordenada (x, y) de la pantalla, el movimiento de S yace sobre el plano $X - Y$ por lo que su componente en Z es igual cero, entonces:

$$S = [S_x, S_y, S_z]^T = [x - W/2, -(c_y + y), 0]^T \tag{4.1}$$

Para hallar la ubicación de P se toma la distancia D_z a la que se encuentra del origen sobre el eje Z, como se mostró previamente la persona se desplaza a metros sobre el eje X y se encuentra a una distancia sobre el eje y de $L_y + L - H_y$, por lo que P se encuentra en:

$$P = [P_x, P_y, P_z]^T = [a, -(L_y + L - H_y), D_z]^T \tag{4.2}$$

Ahora que ya se definieron donde se encuentran los puntos S y P se necesitan los ángulos que se producen entre la posición del vector $\vec{v} = \vec{PS}$ y cada uno de los ejes que son de interés, estos ángulos se conocen como cosenos directores del vector $\vec{v}$ (imagen 4.5) y únicamente son necesarios los que se forman con el eje Y y X. Las fórmulas de los cosenos directores son:

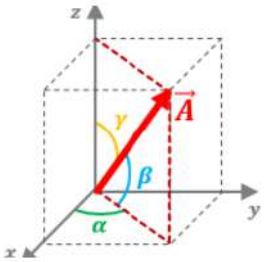

Figura 4.5: Cosenos directores

$$cos(\phi_y) = \frac{\vec{v}_y}{|\vec{v}|} \tag{4.3}$$

$$cos(\phi_x) = \frac{\vec{v}_x}{|\vec{v}|} \tag{4.4}$$

donde $|\vec{v}|$ se halla mediante:

$$\sqrt{\vec{v}_x^2 + \vec{v}_y^2 + \vec{v}_z^2} = \sqrt{(x - \frac{W}{2} - a)^2 + (-c_y - y + (L_y + L - H_y))^2 + (-D_z)^2} \tag{4.5}$$

Por lo tanto los ángulos de desviación son:

$$\phi_y = cos^{-1}(\frac{-c_y - y + (L_y + L - H_y)}{\sqrt{(x - \frac{W}{2} - a)^2 + (-c_y - y + (L_y + L - H_y))^2 + (-D_z)^2}}) \tag{4.6}$$

$$\phi_x = cos^{-1}(\frac{x - \frac{W}{2} - a}{\sqrt{(x - \frac{W}{2} - a)^2 + (-c_y - y + (L_y + L - H_y))^2 + (-D_z)^2}}) \tag{4.7}$$

4.2.2. Cálculo de ϕ_y y ϕ_x. Segundo caso

En este caso para el cálculo de ϕ_x y ϕ_y no es necesario que la cámara se encuentre en una orientación específica (paralela al piso), ahora los ángulos se calculan de manera independiente a la orientación de la camára. Por lo tanto se resuelve la limitante de que la cámara no pueda estar a una altura mayor que la de la persona, ya que si esto sucede no se alcanza a visualizar en la imagen el rostro de las personas en posiciónes cercanas a la cámara, estas posiciones son las más importantes para la captura de datos.
Para el segundo caso únicamente se necesita conocer cual es la ecuación del piso con respecto a la cámara, la ubicación de la persona en el plano y su estatura. En las sección 4.3 se explica como se obtienen los parámetros del plano del piso n (normal del plano) y d (distancia más cercana al origen) mediante técnicas de visión computacional y geometría proyectiva.

Localización de P

Como se mostró en el caso anterior para hallar la mirada de las personas en los dos ángulos únicamente se necesita aplicar la fórmula de cosenos directores al punto P (cabeza de la persona) y S (objeto en pantalla). En este caso se comienza el análisis considerando el punto que yace en el piso donde se colocan las personas, sea este punto $F = [F_x, F_y, F_z]$, F se puede obtener estableciendo dos valores en la ecuación del plano, uno en el eje x y el otro en el eje z y el valor de y se consigue despejando la variable y de la ecuación:

$$d = Ax + By + Cz \tag{4.8}$$

$$y = \frac{d - Ax - Cz}{B} \tag{4.9}$$

Donde $x = F_x$, $y = F_y$ y $z = Fz$. Para encontrar P simplemente se traslada del origen a F y se multiplica la estatura de la persona H_y en dirección negativa (regla de la mano derecha) por la norma de la normal del plano, lo anterior se debe a que las personas al estar paradas sobre el piso siempre están paradas de manera ortogonal, el resultado de esta operación es la ubicación de P en 3 dimensiones, véase la figura 4.6.

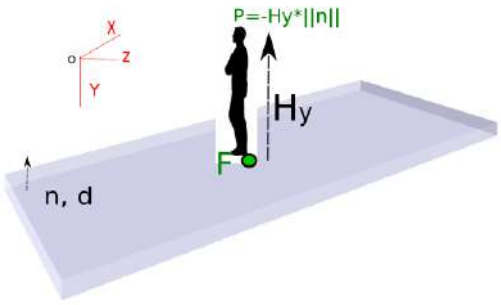

Figura 4.6: Cálculo de P

Localización de S

Para conocer con precisión cuál es la ubicación en pantalla de la figura S con la cámara en cualquier orientación es de gran ayuda utilizar de una unidad Pan-Tilt. Una unidad Pan-Tilt es un dispositivo que permite colocar camáras en posiciones bastante precisas, imagen 4.7.

Para continuar con el análisis hay que considerar los siguientes elementos de la unidad Pan-Tilt y la cámara: O_c es el eje de rotación de la cámara (punto focal); O_u es el eje de rotación de la unidad Pan-Tilt; D_{cu} la distancia entre O_c y O_u en el eje Y; θ es el ángulo que rota la unidad Pan-Tilt alrededor del eje X; $S_1 = (S_{1x}, S_{1y}, S_{1z})$ es la ubicación inicial de la figura en pantalla antes de aplicar la rotación de la cámara con la unidad y finalmente S_2 es la ubicación después de la rotación, todos los elementos están en la imagen 4.8. Partiendo de la ubicación

Figura 4.7: Unidad Pan-Tilt

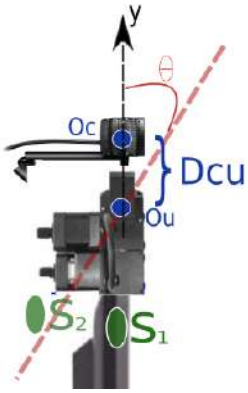

Figura 4.8: Parámetros de la unidad Pan-Tilt y la cámara

de S_1, la localización de S_2 se realiza tomando en cuenta los parámetros anteriores mediante los siguientes pasos:

- Colocando la unidad Pan-Tilt con la cámara de manera que el objetivo de la cámara quede de forma perpendicular a la del plano de la pantalla en la cual se proyectarán las figuras que los sujetos de experimentación observan, figura 4.9.

Figura 4.9: Objetivo de la cámara

- Se traslada D_{cu} metros el punto S_1 hacia el eje de rotación de la cámara O_c, únicamente en el eje Y.
- Se rota θ grados el punto mediante la matriz de rotación del eje X.

- Finalmente se regresa el punto D_{cu} metros.

Los pasos anteriores se llevan a cabo con la siguiente operación:

$$S_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(-\theta) & -sen(-\theta) \\ 0 & sen(-\theta) & cos(-\theta) \end{bmatrix} * \begin{bmatrix} S_{1x} \\ S_{1y} - D_{cu} \\ S_{1z} \end{bmatrix} + \begin{bmatrix} S_{1x} \\ S_{1y} + D_{cu} \\ S_{1z} \end{bmatrix} \tag{4.10}$$

Conociendo S y P en el segundo caso, los ángulos de la mirada se calculan exactamente igual que como se calculan en el primer caso, es decir, mediante los cosenos directores.

4.3. Obtención de la ecuación del plano del piso

Como se mencionó al inicio del capítulo se deben colocar marcas (con el marco marco de referencia centrado en la cámara) en el piso, donde las personas se colocan para capturar sus datos, los cuales son utilizados para entrenar el algoritmo de aprendizaje automático. Las marcas se pueden pintar midiendo sus coordenadas a partir de la cámara mediante un flexómetro o cinta métrica, sin embargo, al realizarlo de esta forma se tienen los siguientes problemas:

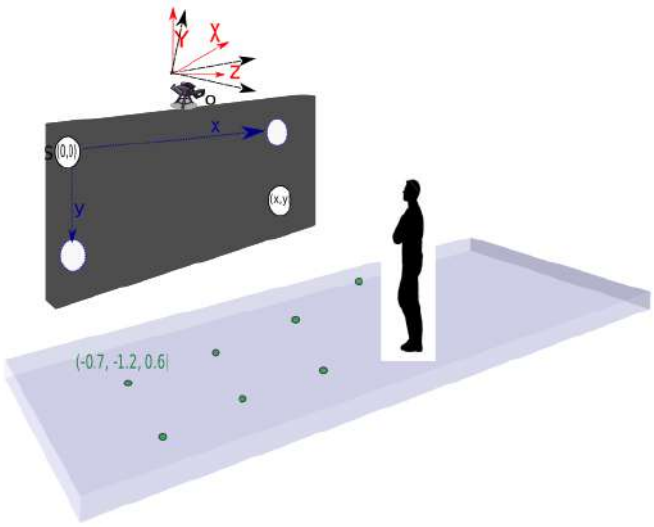

Figura 4.10: Marco de referencia rotado

- En caso de que el eje Z de la cámara esté paralelo al piso (sección 4.2.1).- Sino no se coloca adecuadamente el flexómetro sobre la misma dirección de los ejes del marco de referencia, tiende a incrementarse el error conforme las marcas se alejan.

- La cámara está rotada con respecto al piso, (sección 4.2.2).- Si se rota la cámara de igual forma se rota el marco de referencia y es más complicado colocar las marcas, lo anterior se puede ver ilustrado en la figura 4.10.

Como solución a los problemas anteriores se pueden utilizar algoritmos y herramientas de visión computacional, geometría y optimización numérica para colocar en el piso de manera más precisa (a utilizar un simple flexómetro) las marcas, el método consiste primordialmente en conocer cuál es la ubicación y orientación del plano del piso con respecto a la cámara y proyectar las marcas en éste. Las proyecciones son visualizadas en una pantalla, por lo tanto simplemente se buscan y se pintan en el piso.

4.3.1. Proyección de marcas mediante visión computacional, geometría y optimización numérica

Antes de proyectar las marcas o puntos en el piso se requiere conocer la matriz de parámetros intrínsecos K de la cámara con la que se trabaja, figura 4.11. La obtención de esta matriz se logra mediante el software de Matlab Calib-tool (sección 3.2.2) dando como resultado en una primera experimentación la siguiente matriz:

$$K = \begin{bmatrix} fx & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 616.164 & 0 & 325.528 \\ 0 & 616.82 & 228.66 \\ 0 & 0 & 1 \end{bmatrix}$$

Figura 4.11: Calibración de cámara

Homografía

La matriz de homografía H es uno de los elementos más importantes para la obtención del plano del piso, esta matriz se encarga de mapear puntos en el plano del piso con los respectivos puntos en una imagen de ese plano. Para generar

la homografía se debe crear una matriz $scnP_{mxn}$ con algunos de esos puntos del plano, establecer uno de ellos como origen y colocar las coordenadas de los demás puntos en torno a éste, finalmente los elementos (puntos) de dicha matriz deben tener las medidas reales en metros que hay entre ellos:

$$imP = H * scnP = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_3 \end{bmatrix} * scnP \tag{4.11}$$

La matriz de homografía se descompone y a partir de ella se encuentra la matriz de rotación y el vector de traslación que existe entre el plano del piso y la cámara, es decir, este método se basa en un modelo conocido 3D y su correspondencia en el plano de la imagen 2D.
Inicialmente se creó un software para capturar con el mouse 4 puntos marcados en el piso siguiendo un orden en específico, la matriz con los puntos establecidos con las medidas reales se denomina $scnP_{4x4}$ y la matriz obtenida con los puntos capturados con el mouse se denomina imP_{3x4}. Los puntos establecidos en el piso tienen forma cuadrangular y están separados entre sí a $0.4m$, tomando como origen el punto de la esquina superior izquierda y como primera captura, como segundo punto la esquina inferior izquierda, tercer punto la esquina inferior derecha y último la esquina superior derecha:

$$scnP = \begin{bmatrix} 0 & 0 & 0.4 & 0.4 \\ 0 & 0.4 & 0.4 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

donde cada columna es un punto, la primera fila representa la coordena en el eje X, la segunda en Y y la tercera en Z. Nótese que son coordenadas homogéneas y el valor en z es cero ya que todos los puntos yacen sobre el mismo plano. Como se puede observar la matriz $scnP$ está en metros, por lo tanto se requiere que imP igual esté en metros, esto se consigue simplemente multiplicando la matriz imP por la inversa de la matriz K.
Como resultado del proceso anterior se obtiene una homografía inexacta, ya que al comprobar si mapea los puntos de la escena a la imagen, ésta no lo realiza correctamente y los puntos presentan bastante error. El error en la matriz de homografía podría deberse a los siguientes motivos:

- Hay una distancia considerable entre los puntos del plano y la imagen, con una distancia aproximada de 2m desde la cámara hasta el punto más cercano (parámetro d de la ecuación del plano).
- La captura de los puntos en la imagen mediante el mouse es realizada por simple inspección y seleccionándolos, lo cual podría generar capturas erróneas.
- Son pocos puntos para calcular la homografía, la literatura recomienda mientras más puntos sean posibles mejor.

Para solucionar los inconvenientes anteriores se optó por utilizar más puntos, para ello se coloca una manta con un tablero de ajedrez con 48 intersecciones o en este caso 48 puntos con una longitud de $12cm$ cada lado, esto quiere decir que los puntos están separados por $0.12m$. Además, con el objetivo de que sea más precisa la captura y automática se modifica el sistema para localizar las intersecciones automáticamente mediante bibliotecas de OpenCV y no estar capturando manualmente los puntos. Ahora la matriz $scnP$ tiene una dimensión de $4x48$ mientras imP es de $3x48$. En la imagen 4.12 se pueden notar el buen desempeño que tiene algoritmo al encontrar las intersecciones en la manta con el tablero de ajedrez.

Figura 4.12:

Rotación y traslación de la cámara

Con la homografía H se calcula la pose de la cámara (matriz de rotación R_{3x3} y vector de traslación T_{3x1}) con respecto al plano del piso y a partir de la pose se puede calcular la ecuación del plano.

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_3 \end{bmatrix}$$

La primera columna de H contiene la primera columna de R (R_1), es decir, la rotación al rededor del primer eje, la segunda columna es la segunda columna de R (R_2) y la tercera columna de H es el vector de traslación T, solamente queda por calcular la tercera columna de R (R_3), la cual tiene que ser ortogonal a las dos primeras, por lo tanto se puede obtener mediante el producto cruz de la primera columna de H con la segunda. Adicionalmente antes de calcular la tercera columna de la matriz de rotación es necesario normalizar R y T debido a

redundacias. La normalización se realiza de la siguiente manera:

$$norm1 = ||[h_{11}h_{21}h_{31}]^T|| \tag{4.12}$$

$$norm2 = ||[h_{12}h_{22}h_{32}]^T|| \tag{4.13}$$

$$normT = \frac{norm1 + norm2}{2} \tag{4.14}$$

$$R = \begin{bmatrix} \frac{h_{11}}{norm1} & \frac{h_{12}}{norm2} & R_{13} \\ \frac{h_{21}}{norm1} & \frac{h_{22}}{norm2} & R_{23} \\ \frac{h_{31}}{norm1} & \frac{h_{32}}{norm2} & R_{33} \end{bmatrix}, T = \begin{bmatrix} \frac{T_x}{normT} \\ \frac{T_y}{normT} \\ \frac{T_z}{normT} \end{bmatrix}$$

Luego simplemente se aplica el producto cruz para obtener la tercera columna

$$R_3 = R_1 \otimes R_2 \tag{4.15}$$

Para hallar la ecuación del plano con respecto a la cámara (marco de referencia): $Ax + By + Cz = d$, únicamente es necesario la R y T de la cámara, y un punto de la escena de los que se utilizaron para el cálculo de la homografía.

Cálculo de la ecuación del plano

Como es bien sabido la normal n del plano Π está dada por los componentes: A, B y C de la ecuación del plano y la distancia más cercana del plano al origen por el parámetro d, [46], lo anterior se puede apreciar en la figura 4.13.

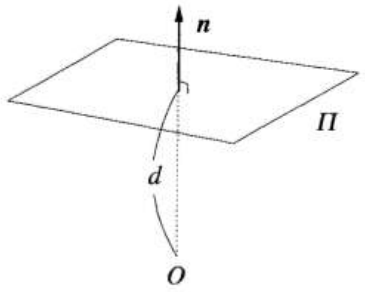

Figura 4.13: Representación de un plano en el espacio

Tomando como marco de referencia el plano (puntos establecidos en $scnP$) la normal está dada por el eje Z, por lo tanto para obtener la normal con respecto al marco de referencia en la cámara únicamente se rota este vector con la matriz de rotación:

$$n = R * \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

Sea $scnP1_{3x1}$ el primer punto de la matriz de puntos establecidos en la escena, entonces la distancia d del plano al origen es este punto rotado y trasladado hacia la cámara, finalmente al resultado se le aplica el producto punto con la normal al plano:

$$scnPR = R * scnP1 + T \tag{4.16}$$

$$d =< N, scnPR > \tag{4.17}$$

Reproyección de puntos en la imagen

Todo el proceso anterior se realiza con el objetivo de conocer la ecuación del plano y proyectar marcas sobre él, las marcas (donde se van a colocar las personas para la captura de sus datos) deben ser visualizadas en una imagen para pintarlas con mayor facilidad y precisión a utilizar un flexómetro o cinta métrica. Lo anterior se consigue multiplicando cada uno de estos puntos por la matriz de parámetros intrínsecos K y de este modo pasar las coordenadas de las marcas de metros a pixeles.
Antes de pasar a esa etapa de proyección es necesario verificar que los parámetros R, T, n y d encontrados en la sección anterior sean correctos, a continuación se presenta una manera de hacerlo:

- La verificación de n y d consiste en utilizar los puntos obtenidos de imP y proyectarlos desde el origen del sistema de coordenadas en dirección del plano del piso (formando rectas) tomando especial atención en las coordenadas que se intersectan con el plano (véase la figura 4.14), estas intersecciones deben ser las mismas que las coordenadas de scP. Sea L la recta en el espa-

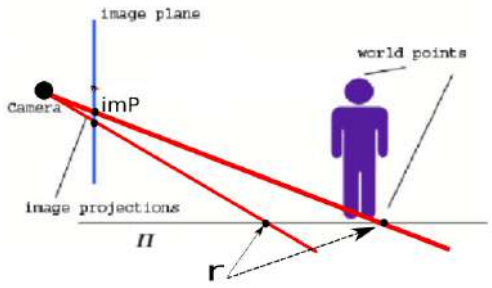

Figura 4.14: Intersección de rectas en el suelo

cio, rh el vector que va del punto más cercano de L al origen y m el vector unitario que indica la orientación de L.

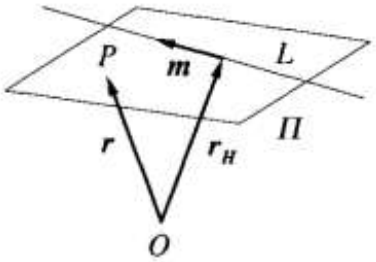

Figura 4.15: Intersección de una recta con un plano

La intersección de L con el plano Π en r está dada por:

$$r = r_H + \frac{d - < n_\Pi, r_H >}{< n_\Pi, m >} m \tag{4.18}$$

En este caso las lineas L (proyecciones de imP) parten de la cámara, por lo tanto el vector rh es igual cero. Simplificando la ecuación, la intersección de cada proyección de imP queda de la siguiente manera:

$$r = \frac{d}{< n_{\Pi}, m >} m \tag{4.19}$$

- La verificación de R y T simplemente se realiza tomando los puntos de $scnP$ rotándolos, trasladándolos (hacia la cámara) y proyectándolos en la imagen, como se describe en la sección 3.2.1, en caso de que los cálculos de la R y T sean correctos estas proyecciones deben coincidir con las de imP.

4.3.2. Optimización con Levenberg-Marquardt

Como se mostrará más adelante en la etapa experimental la rotación y traslación que se obtienen utilizando la descomposición de la matriz de homografía (método de la sección anterior) no son lo suficientemente precisas para comenzar a capturar los datos. Por lo tanto es recomendable utilizar un algoritmo de optimización numérica para minimizar el error que existe en la rotación y traslación, y en consecuencia en la ecuación del plano . Para llevar a cabo la optimización numérica en el presente trabajo se utiliza el algoritmo de Levenberg-Marquardt debido a que es uno de los más populares y eficientes, con él se busca el vector de parámetros p que minimice.

$$F(p) = \frac{1}{2} \sum_{1}^{m} (f_i(p))^2 \tag{4.20}$$

donde $f : \mathbb{R}^n \to \mathbb{R}, i = 1, ..., m$ son las funciones dadas y $m \geq n$. Las funciones a optimizar deben tener las siguientes condiciones: tener incluidos los parámetros de R y T, estar descritas de tal forma que sean iguales o cercanas a cero e incluir los 48 puntos de la manta del tablero de ajedrez, por lo tanto el número de ecuaciones a minimizar considerando los tres componentes de cada punto es $m = 144$ y se plantean de la siguiente forma:

$$f_{xyz}(p) = ||X_{rt} - X_{intersec}|| \tag{4.21}$$

donde X_{rt} es el conjunto de puntos $scnP$ rotados y trasladados, y $X_{intersec}$ es el conjunto de puntos imP que se proyectan e intersectan con el plano. Por cada conjunto de tres ecuaciones (cada punto) se tiene:

$$f_{xyz}(p) = \{R * X_{scn} + T\} - \{\frac{d}{< n_{\Pi}, X_{im} >} X_{im}\} \tag{4.22}$$

Como se explicó anteriormente la n y d del plano dependen de la rotación y traslación por lo que es importante también ir actualizando estos parámetros en cada paso de la optimización.
En problemas de optimización numérica la redundancia de las matrices de rotación es inconveniente y a menudo es preferible una representación mínima [47]. La

representación más simple está basada en el teorema de Euler, él cual menciona que cada matriz de rotación puede ser descrita por un eje de rotación y un ángulo alrededor de él. Una compacta representación del eje y el ángulo es un vector de rotación de tres dimensiones cuya dirección es el eje y cuya magnitud es el ángulo en radianes, por lo tanto en esta investigación se optimizará el eje de rotación y su magnitud del ángulo.
De acuerdo a [48] mediante la fórmula de Rodrigues cualquier matriz de rotación se puede realizar a través de la rotación alrededor de un eje fijo ω por un cierto ángulo $\| \omega \|$ con la siguiente ecuación:

$$R = I + \frac{\hat{\omega}}{\| \omega \|} sin(\| \omega \|) + \frac{\hat{\omega}^2}{\| \omega \|^2}(1 - cos(\| \omega \|)) \tag{4.23}$$

Donde los valores del eje y el ángulo de una matriz R:

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

se obtienen por medio de:

$$\| \omega \| = cos^{-1}(\frac{traza(R) - 1}{2}), \frac{\omega}{\| \omega \|} = \frac{1}{2sin(\| \omega \|)} \begin{bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{bmatrix} \tag{4.24}$$

y $\hat{\omega}$ es una matriz simétrica oblícua. Los 7 parámetros optimizar son los siguientes:

$$p = [\omega_x, \omega_y, \omega_z, \| \omega \|, T_x, T_y, T_z] \tag{4.25}$$

En el algoritmo de Levenberg-Marquardt se debe definir cuáles son cada una de las funciones a optimizar, tomando los 48 puntos de la ecuación 4.22 se tienen en total 144 funciones a optimizar.
Para comenzar con el proceso de optimización falta expresar las funciones de 4.22 en términos de $\omega_x, \omega_y, \omega_z, \| \omega \|, T_x, T_y, T_z$, esto se logra aplicando la fórmula de Rodrigues y simplificando con álgebra básica. Debido a que son bastantes ecuaciones resultantes y cada una es muy extensa, a continuación se presenta únicamente las 3 componentes de una función de optimización $f_{xyz}(p)$ de la ecuación 4.21, esto quiere decir que se optimizarán 48 de cada una de este tipo:

- Componente x:

$$\begin{aligned} f_{xi}(p) = &[(\omega_x^2 - 1)(1 - cos(\| \omega \|)) + 1]X_{scnxi} + \\ &[\omega_x\omega_y(1 - cos(\| \omega \|)) - \omega_z(sin(\| \omega \|))] X_{scny} + \\ &[\omega_y sin(\| \omega \|) + \omega_x\omega_z(1 - cos(\| \omega \|))] X_{scnzi}\} + T_x \end{aligned} \tag{4.26}$$

- Componente y:

$$\begin{aligned} f_{yi}(p) = &[\omega_z sin(\| \omega \|) + \omega_x\omega_y(1 - cos(\| \omega \|))]X_{scnxi} + \\ &\left[(\omega_y^2 - 1)(1 - cos(\| \omega \|)) + 1\right] X_{scnyi} - \\ &[\omega_x sin(\| \omega \|) + \omega_y\omega_z(1 - cos(\| \omega \|))] X_{scnzi} + T_y \end{aligned} \tag{4.27}$$

- Componente z:

$$\begin{aligned} f_{zi}(p) = &[\omega_x\omega_z(1 - cos(\parallel \omega \parallel)) - \omega_y \sin(\omega)]X_{scnxi}+ \\ &[\omega_x sin(\parallel \omega \parallel) + \omega_y\omega_z(1 - cos(\parallel \omega \parallel))]\, X_{scnyi}+ \\ &[(\omega_z^2 - 1)(1 - cos(\parallel \omega \parallel)) + 1]\, X_{scnzi} + T_z \end{aligned} \tag{4.28}$$

4.4. Generación de secuencia óptima de marcas de experimentación

Durante la captura de datos de las personas hay infinitas combinaciones de posiciones de marcas en el piso y en la pantalla que se pueden utilizar, inclusive si las regiones de captura en ambos casos están limitadas en áreas pequeñas, sin embargo, lo más importante es capturar información en las marcas más relevantes, las cuales son aquellas en las que se da mayor variación en los ángulos de la mirada: ϕ_x y ϕ_y, con el objetivo de que haya variedad en el conjunto de datos de entrenamiento. Para que se cumpla lo anterior estos ángulos de mirada deben ser mayores a los umbrales $thres_x$ y $thres_y$ resultantes del análisis que se lleva a cabo en la sección 5.2.
Al final de esta etapa se genera una secuencia de marcas en el piso en donde a cada una le corresponden dos posiciones diferentes en la pantalla, esto quiere decir que por cada vez que se pare una persona en una marca en el plano del piso se capturará dos veces su rostro en orientaciones diferentes. A continuación se detallará como se realiza el proceso para la selección de los puntos más significativos (con mayor variación).

4.4.1. Malla de puntos en el plano del piso y pantalla

De modo que se requiere que la captura de datos (ubicación, ángulos de mirada e imagen) no le lleve bastante tiempo a cada persona, se desarrolló esta etapa tomando en cuenta que sea automatizada y rápida. El primer paso consiste en generar el arreglo bidimensional de puntos en el plano del piso, es necesario considerar en donde se encuentra el plano del piso en relación al marco de referencia (cámara), por lo tanto se deben tomar en cuenta los parámetros de n y d. Con base en el análisis de los ángulos ϕ_x y ϕ_y (véase la sección 5.2 para más detalle) se determina el tamaño del paso entre cada marca en el eje X y Z del piso, y cuáles son los límites máximos y mínimos en ambos ejes para pintarlas.
Posteriormente se genera una matriz de puntos en el piso que representa todas las posiciones posibles en las cuales las personas se pueden colocar para capturar sus datos, cada una de éstas es representada por la variable F, además, se calcula para cada posición F cuál es la ubicación de la cabeza de la persona en la escena, representada por P, y se calcula como se explica en la sección 4.2.2. Una vez localizada la P se descartan aquellas que no se encuentren en el campo visual de la cámara, ya que de nada sirve capturar una imagen del rostro de la persona si no se encuentra el rostro en ella. Para determinar si el rostro se encuentra en el campo visual solamente se proyecta en la imagen la ubicación de P mediante la matriz de calibración K de la cámara, con el resultado obtenido se verifica que

esté dentro de los límites visibles de la imagen, si es así P se considera para la siguiente etapa del algoritmo.
Sean $P_1, P_2, P_3...$ el conjunto de puntos en la escena que representan el rostro en diferentes posiciones resultantes de la etapa anterior, se elige P_1 (un punto en específico en el arreglo bidimensional) y se determina en ese punto cuál es el ángulo ϕ_{x1} y ϕ_{y1} para una figura en pantalla con una posición fija S_1, inmediatamente se calcula la rotación de la mirada con respecto a otro punto en la malla P_2 pero con la misma S_1, los nuevos ángulos son ϕ_{x2} y ϕ_{y2}. A continuación se calcula rotación en la mirada que existe entre ambas posiciones:

$$rot_x = \phi_{x1} - \phi_{x2} \tag{4.29}$$

$$rot_y = \phi_{y1} - \phi_{y2} \tag{4.30}$$

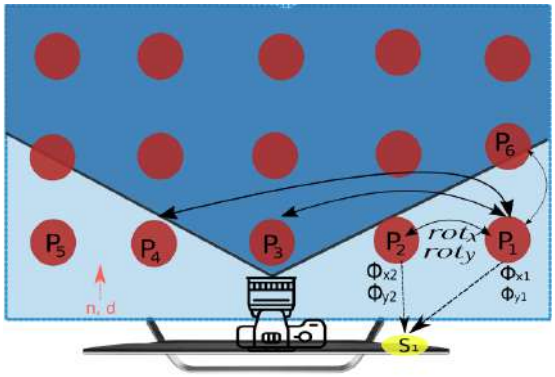

Figura 4.16: Arreglo de puntos en el piso

Lo anterior se calcula para las demás P de la matriz siempre con respecto a P_1 (véase la figura 4.16 para un mejor entendimiento), donde el área con un tono más bajo de azul representa el campo visual de la cámara. Al terminar de calcular todas las rotaciones de P_1 se procede a calcular todas las rotaciones de mirada ahora con respecto a P_2 con todos los demás puntos y así sucesivamente para los demás puntos del arreglo bidimensional. Al final se genera una matriz de matrices de rotaciones de ángulos de la mirada y esto es con respecto a una posición de la figura en pantalla fija que se observa, en este ejemplo S_1.
En el siguiente paso se generan matrices de matrices para varias ubicaciones de S en pantalla. Después de generar estas matrices, filtrar las rostros: P que presentan mayor variación en rotación y validar que los rostros P se encuentren dentro del campo visual de la cámara, se generan las **instancias** de posibles puntos para capturar datos, cada instancia consta de los siguientes elementos:

$$F1x, F1y, F1z, F2x, F2y, F2z, Sx, Sy, Sz, rot_x, rot_y \tag{4.31}$$

Los tres primeros datos indican la primera posición en el plano del piso del sujeto de experimento, los siguientes tres datos hacia que punto se movió, los siguientes tres indican la posición de la figura en pantalla para esa instancia y los últimos dos datos señalan la rotación en el vector de la mirada que hubo al pasar de $F1$ a $F2$ con respecto al eje X y al Y.

4.4.2. Búsqueda de secuencia óptima mediante algoritmo de ordenamiento

Este algoritmo tiene como objetivo analizar las instancias obtenidas en la etapa anterior para hallar una secuencia de n elementos que presente por cada par de puntos en el piso consecutivos un desplazamiento amplio en los ejes X y Z, de igual manera los movimientos consecutivos del objeto en pantalla deben presentar un desplazamiento amplio en los ejes X y Y. Lo anterior tiene como finalidad que en cada instancia de la secuencia haya una variación significativa en la mirada, esto es: rot_x y rot_y. A continuación se describe un algoritmo basado en ordenamientos, búsquedas y agrupamientos para elegir de entre todo el conjunto de instancias posibles una secuencia óptima.

Primero se almacenan todas las instancias en una matriz donde cada fila es una de ellas y cada columna es uno de los 11 valores que tiene cada instancias. En el siguiente paso se le añaden 4 campos a cada instancia: subconjunto, inicio de subconjunto, final de subconjunto y relación entre los dos puntos (o marcas) de la misma instancia, es decir, F1 y F2.

Ahora se añaden los valores para el campo de subconjunto de todas las instancias de la siguiente manera: primero se ordenan las filas de menor a mayor con base en $F1x$, después se agrupan en conjuntos que tienen la misma $F1x$ (por lo tanto los conjuntos son aquellos con el mismo $F1x$) y cada conjunto se ordena de menor a mayor ahora con base en $F1z$ y se asigna el valor de subconjunto a aquellas instancias que también tienen la misma $F1z$, esto quiere decir que los subconjuntos son aquellos grupos de instancias en las que la columna $F1x$ es igual para todas instancias del conjunto y lo mismo para la columna de $F1z$.

En el siguiente paso se recorren todas las filas de datos y se colocan los valores para los 3 nuevos campos faltantes con ayuda de la etiqueta de subconjunto, la cual sirve para identificar en que número de elemento comienzan y terminan los subconjuntos, y cuantos elementos posee cada uno.

A continuación se elige el número de instancias finales que se utilizarán para los experimentos y se crea el vector que almacenará dichas instancias finales: $vInstances$. El primer elemento de $vInstances$ es seleccionado aleatoriamente del primer subconjunto de la matriz de datos, de esta instancia se analiza el elemento $F2(F2x, F2y, F2z)$ buscando en cual instancia diferente de toda la matriz hay un $F1(F1x, F1y, F1z)$ con exactamente los mismos valores y se procede a analizar el subconjunto al que pertenece este nuevo $F1$, lo anterior se puede ver ilustrado en la figura 4.17.

Con este nuevo $F1$ se verifican dos condiciones: que el subconjunto al que perte-

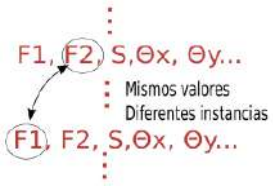

Figura 4.17: Búsqueda de siguiente instancia

nece no tenga una instancia en la lista $vInstances$ y que su S (Sx, Sy, Sz) no se encuentre en $vInstances$, si se da alguna de estas condiciones se selecciona otra instancia del mismo subconjunto, si no es así se añade esta nueva instancia en la lista de $vInstances$, y se procede a analizar el nuevo elemento de la lista como el anterior, es decir, analizando su $F2$, y así sucesivamente hasta añadir el número de elementos deseado.
Al final el algoritmo genera la lista secuencial de instancias en la que el elemento $F2$ de cada una es igual a $F1$ de la siguiente instancia, se diseña de esta manera para que la captura de datos de las personas sea lo más rápido posible. En la figura 4.18 se puede apreciar un diagrama de flujo del algoritmo descrito en esta sección.

4.5. Red neuronal

El algoritmo clasificador propuesto para esta etapa de la tesis es una red neuronal de propagación hacia atrás (sección 3.6), donde las neuronas de entrada son el vector de características y las neuronas de salida representan la región en pantalla que se observa. En esta sección se describe como se implementa la red neuronal en este proyecto de investigación, comenzando con el tratamiento de datos (incluidas las imágenes) requerido para representarlos como el vector de características.

4.5.1. Vector de características de entrada

Con las imágenes y la información capturada se preparan los datos para la red neuronal, para ello se generan dos etapas de tratamiento de datos, la primera consiste de los siguientes pasos:

1. Se procesa cada una de las imágenes capturadas con el algoritmo de Viola y Jones, con el objetivo de realizar la detección de rostros.

2. De las imágenes con rostros detectados se extrae la región donde se encuentra el rostro mediante un ROI (región de interés).

3. El ROI de la cara de la persona se guarda para posterior análisis junto con los datos de ubicación 3D de la cabeza y la ubicación (igual 3D) de lo que estaba observando en pantalla al momento de la captura, esto es P y S.

4. Finalmente se completa la información con la resolución original del ROI del rostro ya que en una etapa posterior es necesario redimensionar todos los rostros a una misma resolución.

La salida de este proceso tiene un conjunto de 8 datos por cada rostro: *la imagen, la dimensión original del rostro*, P_x, P_y, P_z, S_x, S_y, S_z. Utilizar la resolución original del ROI del rostro como una característica del vector sirve como un indicador de la distancia sobre el eje Z de la persona a la cámara.
La segunda etapa consiste simplemente en acomodar los datos resultantes del

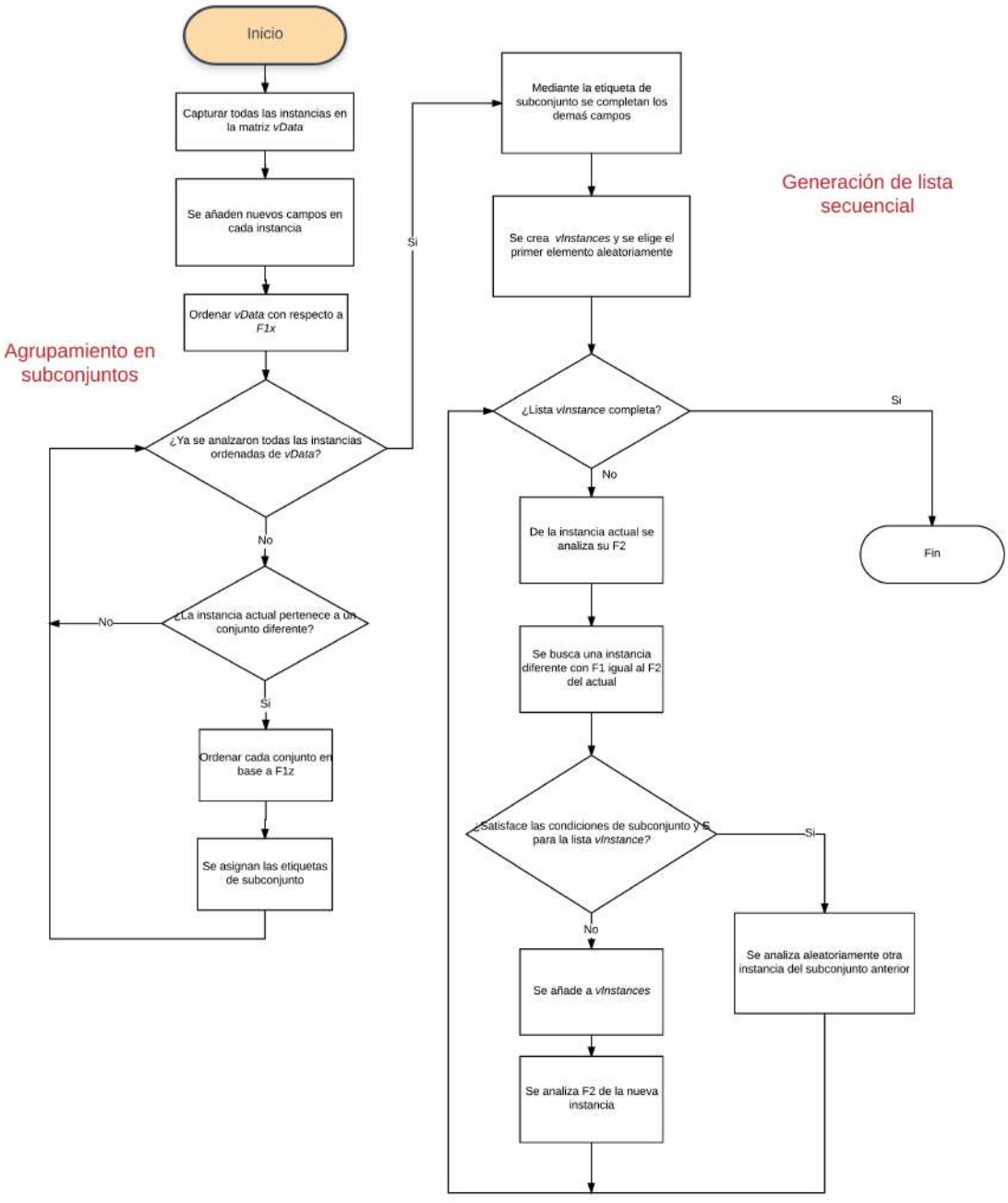

Figura 4.18: Diagrama de flujo

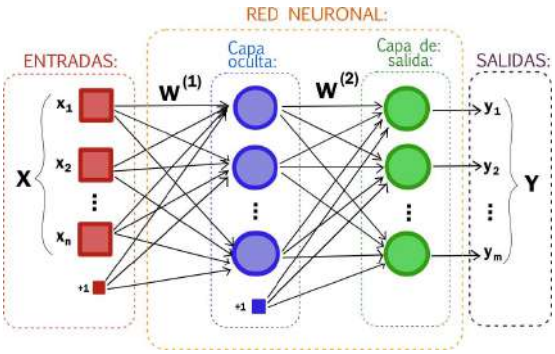

Figura 4.19: Red neuronal artificial

Figura 4.20: Características

proceso anterior en forma del vector de características que se requiere en las neuronas de entrada de la red. En cada ejemplo de entrenamiento se redimensiona la imagen del rostro a un estándar $dimE$, en donde cada pixel representa una característica del vector. Por lo tanto al final se obtiene el vector de características con $dimE+4$ elementos, donde los 4 elementos restantes son la dimensión original del rostro y los valores de P.

HOG

El otro vector de características con el que se experimenta se compone de los descriptores HOG, es decir, en vez de utilizar la imagen del rostro completa como vector, se utilizan los descriptores HOG extraídos del rostro. Con ello se busca mejorar la precisión de la clasificación (porcentaje de aciertos) y minimizar el tiempo que le toma a la red neuronal propagarse, ya que como se mencionó en la sección 3.5 este tipo de descriptores extraen la información útil y desechan la extraña de una imagen.
Los parámetros más importantes a considerar para el cálculo del vector de descriptores HOG en los ejemplos de entrenamiento son los siguientes:

- *winSize*.- Este parámetro es la dimensión de la ventana sobre la cual el descriptor es calculado. Usualmente se coloca la misma dimensión que el de la imagen,

- *blockSize*.- Es el tamaño de las celdas en las cuales se calculan los histogramas que se van desplazando a través de la imagen. Este parámetro se usa para contrarrestar la variación en la iluminación, un tamaño de bloque

grande hace que los cambios de iluminación locales sean menos significativos, mientras que los bloques pequeños le dan más peso a las variaciones locales de iluminación.

- *cellSize*.- La elección de este elemento está basada en la escala de las características importantes para realizar la clasificación. Un tamaño pequeño podría resultar en un vector de características muy extenso, mientras que un valor grande para *cellSize* podría no capturar información relevante.

- *nbin*.- Es el tamaño de los intervalos contenedores "bins" en el histograma de gradientes.

4.5.2. Neuronas de salida

En la red neuronal que se utiliza en este proyecto cada neurona de salida representa una partición de la pantalla que la persona está observando, por ejemplo, si solamente se tienen dos neuronas de salida la pantalla será particionada en dos regiones. Durante la etapa experimental se utilizan diferentes topologías y número de neuronas de salida de la red.
Con respecto a lo anterior es necesario asociar la figura de lo que observa la persona S (etiqueta) a una neurona de salida. Como se conocen las dimensiones de la pantalla donde se desplaza S únicamente se determina para cada ejemplo de entrenamiento en que región de la pantalla se encuentra y se le coloca un uno en esa neurona, en caso contrario se coloca un cero, lo anterior está representado en la imagen 4.21.
Al momento de realizar estimaciones con los datos de prueba ya con la red

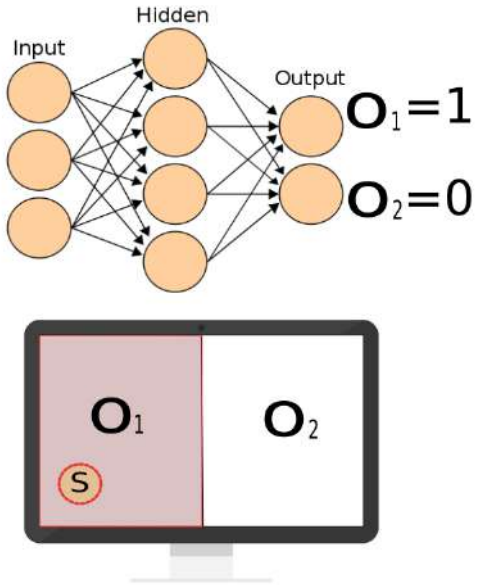

Figura 4.21: Neuronas de salida

entrenada los resultados en el vector de salida no son enteros como las etique-

tas en el entrenamiento (1 y 0), la red arroja valores decimales, por lo tanto para tomar una clasificación como acertada se elige la neurona con el valor má alto (cercano a uno), por ejemplo, si la red arrojó el siguiente vector: $[O_1, O_2] = [-0.0555573, 1.05556]$ se elige O_2 como la clase estimada correctamente por la red neuronal.

4.6. Resumen

En el presente capítulo se ha desarrollado una metodología que consta de dos etapas, la primera es para la captura precisa de los datos de entrenamiento y la otra consiste en la implementación de un algoritmo de clasificación. El objetivo de la primera etapa es generar una secuencia de marcas en el piso donde se colocan las personas para capturar sus datos y de igual forma marcas en la pantalla que deben observar, el sistema debe conocer con precisión cuáles son las coordenadas de la cabeza de la persona y de lo que observa, y mediante los cosenos directores se puede inferir los ángulos de la mirada.
Cabe destacar que gran parte de la metodología se basa en encontrar las coordenadas (cabeza de la persona y objeto que miran) necesarias para el cálculo de la mirada. El sistema que se propone obtiene estas coordenadas de interés a partir de la ubicación del plano del piso con respecto a la cámara. En este capítulo se presentó un enfoque para localizar el plano partiendo de la matriz de homografía que hay entre el plano 3D del piso y el plano 2D de la imagen, luego, se descompone esa matriz y con herramientas de geometría y álgebra lineal se encuentra la rotación y traslación en tres dimensiones que hay entre estos planos, con estos elementos se puede inferir fácilmente la ecuación del plano, sin embargo, dado que existe una distancia lejana entre el plano y la cámara se generan errores, por lo tanto el último paso para la obtención de la ecuación del plano consiste en la minimización del error mediante el algoritmo de optimización numérica Levenberg-Marquardt.
La segunda etapa de la metodología describe como se utilizan los datos capturados de entrenamiento y la forma de introducirlos en la red neuronal, de igual manera se menciona la interpretación de las neuronas de salida.

Capítulo 5

Experimentos

En este capítulo se describen los experimentos que se llevaron a cabo para probar el sistema estimador de la mirada, al igual que con la metodología los experimentos están divididos en los que pertenecen a la etapa de calibración del experimento y los que pertenecen a la clasificación. Cabe señalar que se probaron y reportaron los resultados de experimentar con diferentes topologías para la red neuronal, además se analizó y comparó utilizar diferentes características en el vector de entrada de la red.

5.1. Materiales

Todos los experimentos se llevaron a cabo en una computadora con sistema operativo Linux ubuntu versión 16.04.2 LTS, con procesador Intel(R) Core(TM) i7-5500U CPU @ 2.40GHz y 8GB de memoria RAM. Además, se utilizó una unidad Pan-Tilt para colocar con precisión la cámara como la que se encuentra en la imagen 4.7 y una cámara logitech C920 (figura 5.1) .

Figura 5.1: Cámara Logitech C920

5.2. Análisis del comportamiento de los ángulos de mirada ϕ_x y ϕ_y

Durante la etapa de entrenamiento se requiere capturar bastantes imágenes de personas que se encuentran enfrente de la cámara y con el rostro en diferentes orientaciones. Para lograr que la persona varíe su mirada o lo que es lo mismo la orientación de su cabeza se requiere:

- Se varíe la posición del objeto en pantalla que el sujeto del experimento está observando
- Se debe variar la posición de la persona en el lugar del experimento

Como se mencionó en la sección 4.4.1 una instancia de captura es la ubicación de una persona en el lugar del experimento P, la posición del objeto S y la orientación de la cabeza (medida a través de los ángulos ϕ_x y ϕ_y). Habiendo señalado lo anterior, si se llega a dar una extensa cantidad de instancias de captura por cada persona variando los elementos P y S, llevaría en consecuencia a bastante tiempo de captura, lo cual sería inconveniente para los sujetos del experimento, por lo tanto se realizó un análisis del comportamiento de los ángulos en diferentes circunstancias, el análisis fue realizado mediante el software matemático Octave y a continuación se discuten los resultados.
El análisis presentado a continuación se realizó con respecto al caso 1 de la sección 4.2.1 suponiendo que se utiliza una pantalla de 42 pulgadas, el marco de referencia se encuentra en la cámara encima de la pantalla y todas las unidades de medida están en metros.

- Ancho de la pantalla: $W = 0.9282$
- Largo de la pantalla: $L = 0.5523$
- Distancia de la pantalla al piso: $Ly = 1.5$
- Distancia de la cámara a la pantalla: $Cy = 0.05$
- Estatura del sujeto del experimento: $Hy = 1.6$
- Movimiento de la persona a través del eje Z: de 0.1 a 3 con pasos de 0.51786
- Movimiento de la persona a través del eje X: de -3 a 3 con pasos de 0.109
- Movimiento de la figura en pantalla a través del eje X: de $-\frac{W}{2}$ a $\frac{W}{2}$ con pasos de 0.01658
- Movimiento de la figura en pantalla a través del eje Y: de $-Cy$ a $-(Cy+L)$ con pasos de 0.05178

Véase la imagen 4.4 para un mejor entendimiento de los parámetros anteriores.

En la gráfica de la figura 5.2 se puede observar la variación de los ángulos ϕ_x y ϕ_y con la persona en una posición posición fija: $P = [0, -(L_y + L - H_y), 1]^T$, la variación de la figura en pantalla S es en el eje X y es graficada tomando como origen la esquina superior izquierda de la pantalla. El mayor cambio que ocurre en los ángulos es con respecto a ϕ_x a que pasa de 2 a 1.1 radianes, los cuales son 0.9 radianes o 51.5662 grados, esto parece tener mucho sentido ya que el desplazamiento se hace en el eje de ϕ_x.
Resulta peculiar que a pesar de que no hay desplazamiento en el eje Y de S se da una ligera variación en el ángulo ϕ_y de la mirada de la persona, esto se debe al

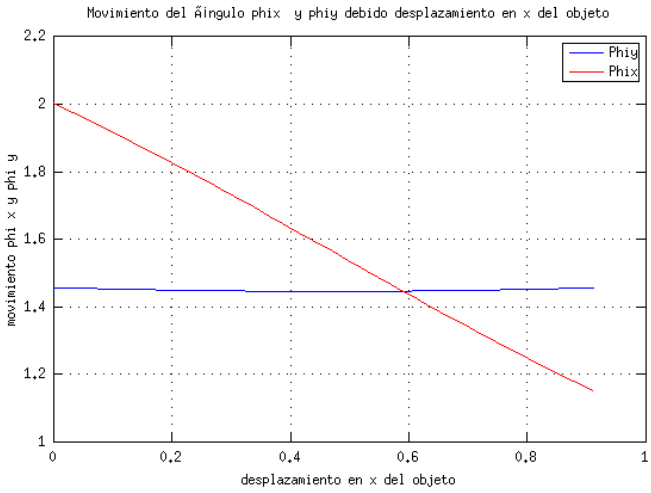

Figura 5.2: Desplazamiento de S en el eje X

punto de fuga. El punto de fuga es el lugar geométrico en el cual las proyecciones de las rectas paralelas a una dirección dada en el espacio, no paralelas al plano de proyección convergen, lo anterior se puede ver ilustrado en la imagen 5.3, el punto de fuga es el lugar donde convergen todas líneas "paralelas"de color verde, y la línea del horizonte es la recta horizontal de color azul.

Cuando el objeto en pantalla S inicialmente se encuentra en una posición superior

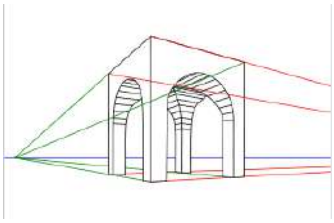

Figura 5.3: Punto de fuga

a la de los ojos y se mueve hacia uno de los extremos de la pantalla, éste pareciera moverse hacia abajo (hacia el horizonte) y cuando comienza el movimiento de S en una posición inferior a la de los ojos de la persona y se mueve hacia un extremo, éste pareciera moverse hacia arriba, por lo tanto la mirada de las personas en el eje Y tiende a moverse ligeramente.

En la gráfica de la figura 5.4 se puede apreciar que el objeto que observan únicamente se desplaza en el eje Y, en este caso el único desplazamiento de la mirada es con respecto al ángulo ϕ_y. Al desplazarse el objeto 0.5523 la mirada varía de 1.2 a 1.7 radianes, es decir, 0.5 radianes o lo que e lo mismo 28.6479 grados.

La gráfica 5.5 describe el caso en el que únicamente la persona se mueve sobre el eje X de -3 a 3m permaneciendo S fijo en el centro de la pantalla, el mayor desplazamiento lo tiene el ángulo ϕ_x. De esta gráfica se puede concluir que la mayor variación en el ángulo se da entre -1 y 1, ya que la mirada varía de 0.8 a 2.36 radianes, si la persona se mueve más allá de esta distancia la mirada tiende a

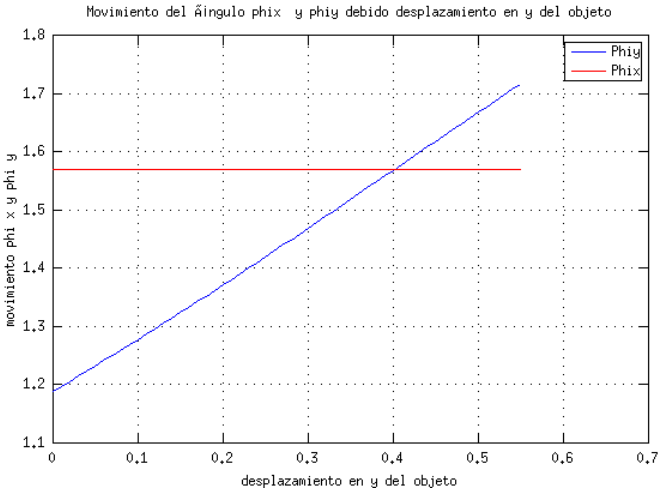

Figura 5.4: Desplazamiento de S en el eje X

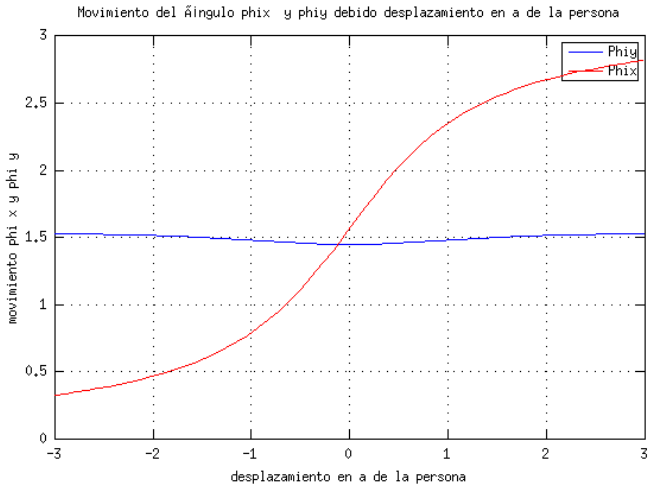

Figura 5.5: Desplazamiento de la persona en el eje X

estabilizarse y no valdría la pena hacer experimentos en estas zonas, la varíación de la mirada es mínima. Lo anterior se ve con mayor claridad en la gráfica 5.6, aquí se amplio el rango en el que se mueve la persona sobre el eje X, como se puede observar ϕ_x tiende a estabilizarse.

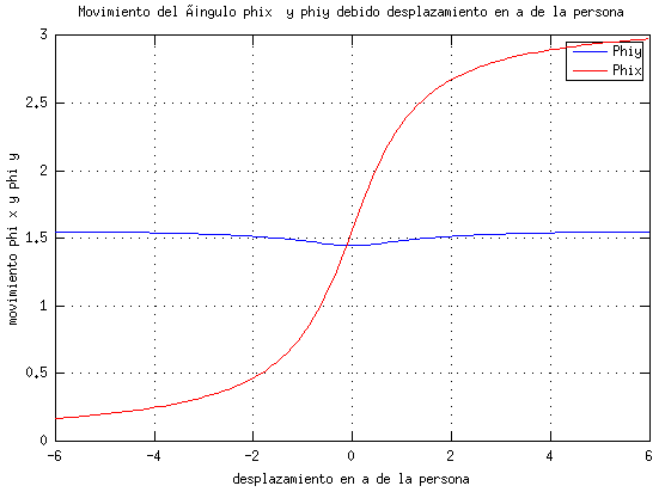

Figura 5.6: Desplazamiento de P en el eje X

El último análisis corresponde al caso cuando el objeto que observan en pantalla no se mueve y la persona únicamente se mueve sobre el eje Z, S está en el centro de la pantalla: $[S_x, S_y, S_z]^T = [\frac{W}{2}, -(c_y + \frac{L}{2}), 0]^T$ y la persona se encuentra en $[P_x, P_y, P_z]^T = [0, -(L_y + L - H_y), vecZ]^T$ donde $vecZ$ es un vector que va desde 0.1 a $3m$.

En la gráfica 5.7 se puede apreciar que mientras se va alejando la persona el

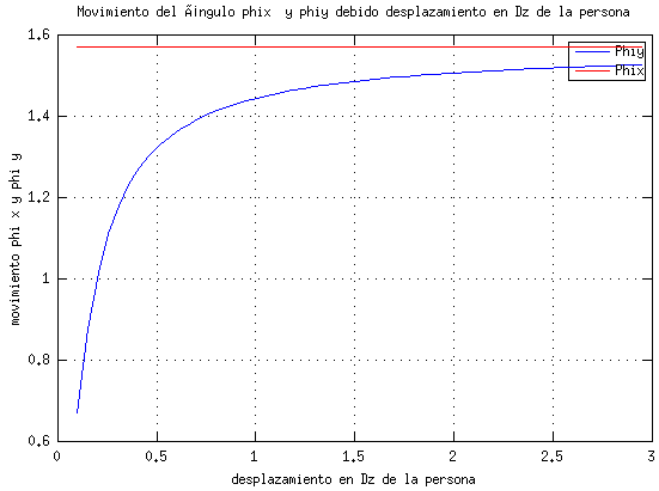

Figura 5.7: Desplazamiento en el eje Z de la persona

único ángulo de la mirada que varía es ϕ_y, el cual va de 0.6 a 1.55 radianes, esto quiere decir que hay una variación en el ángulo de 0.95 radianes o 51.56 grados hacia abajo cuando la persona se aleja, además se puede observar que a partir

de 1.5m la variación que hay es mínima y de 0 a 0.8m es cuando se da la mayor variación. Lo anterior se puede explicar con la linea de fuga, mientras la persona se aleja el objeto pareciera moverse hacia abajo y establecer con la mirada de la persona en 1.6 radianes o 91 grados, en el horizonte.

5.2.1. Simulación processing

Además del análisis hecho con Octave se realizó una simulación mediante el software Processing con el objetivo de observar de manera interactiva y en tiempo real el comportamiento de los ángulos en diferentes posiciones de las personas y variar la posición en el eje X y Y de la figura en pantalla que se observa. Los parámetros del escenario son los mismos a los utilizados en el análisis anterior: W, L, LY, Cy y Hy. La simulación es una vista desde arriba mirando hacia el plano $X - Z$ del sistema.

En la imagen 5.8 se puede apreciar una captura de la simulación, en ella se

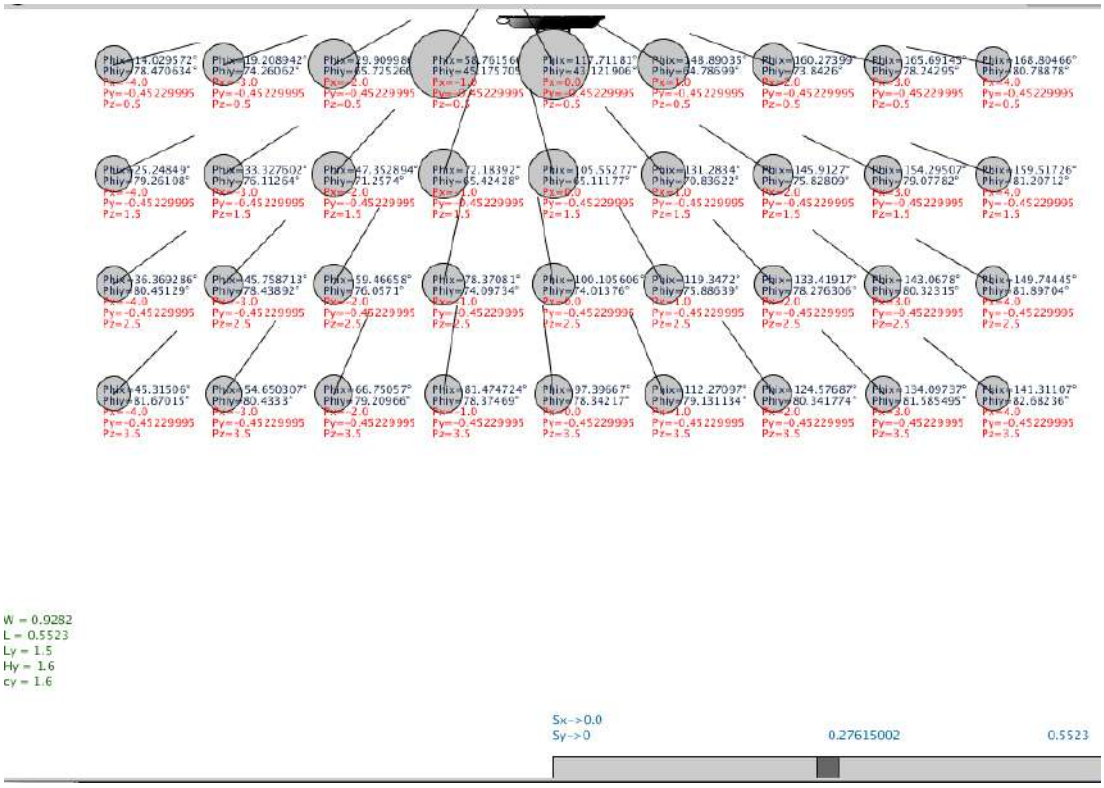

Figura 5.8: Simulación en Processing

despliegan 36 posiciones de personas separadas por 1m en el eje X y 0.5m en el eje Z. La variación de S en el eje Y se da por medio de la barra deslizante que se encuentra en la esquina inferior derecha, va de 0 a 0.5523m y el desplazamiento en el eje X se logra haciendo click en la figura (óvalo blanco) en pantalla que se encuentra en la parte superior de la ventana de la simulación y sin soltar el mouse se arrastra la figura de izquierda a derecha. Los valores de los ángulos ϕ_x y ϕ_y son mostrados en cada una de las posiciones, donde el valor de ϕ_y es representado

por el radio del círculo de cada persona y la línea que sale de dicho círculo es el ángulo ϕ_x.
Por medio la simulación se demostró como al arrastrar el objeto en pantalla S únicamente en el eje X además de variar ϕ_x también varía el ángulo ϕ_y de todas las instancias, esto se puede corroborar en los valores de los ángulos y en el área del círculo.

5.3. Obtención de la ecuación del plano del piso

Retomando la sección 4.3 de la metodología y en especial el apartado de **reproyección de puntos en la imagen**, se realizaron experimentos para verificar que la estimación de la rotación y traslación fuera precisa a partir simplemente de la descomposición de la matriz de homografía. El resultado de comprobación se puede ver en la imagen 5.9, las marcas rojas son las proyecciones en la imagen de las intersecciones de las rectas imP con el plano y las verdes son los puntos de scP rotados, trasladados y proyectados en la cámara.
Como se puede observar los cálculos de los parámetros n y d del plano son precisos, mientras que la R y T presentan error y parece incrementarse al alejarse del origen.

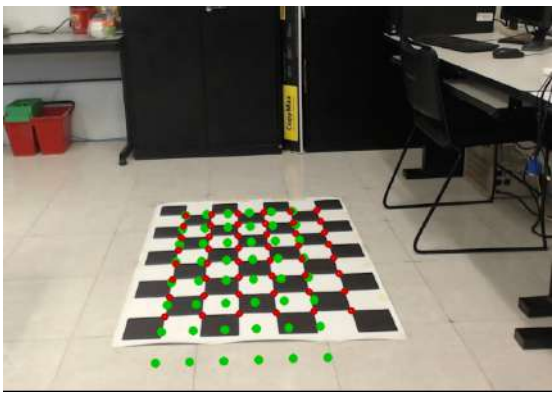

Figura 5.9: Reproyección de puntos

Se realizó un análisis de las distancias en pixeles que hay entre los dos conjuntos de puntos imP y scP en la imagen con una resolución 720x480p, se obtuvo en el primer punto (primera fila de arriba a abajo y la primera columna de izquierda a derecha) 0.27 pixeles de diferencia en el eje X y 0.15 en Y, en el último punto (fila 8 y columna 6) la distancia es de 52 pixeles en X y 107 en Y.
De igual manera se realizó un análisis de los conjuntos de puntos pero en tres dimensiones en metros, es decir, antes de proyectar los puntos en la imagen. En el primer punto hay una distancia de $0.0002978m$ en X, $0.00003559m$ en Y y $0.000467m$ en Z; y en el último punto $0.265m$ en X, 0.0268 en Y y $0.128m$ en Z.
Como sucedió en el caso de dos dimensiones el error se va expandiendo conforme los puntos se alejan del primer punto. El error en R y T es mayor que el que hay en n y d, esto se debe a que estos parámetros del pano se calculan con respecto al primer punto (véase la ecuación 4.17) el cual presenta menos error, sin embargo,

es necesario que R y T sean más precisos porque se utilizan en etapas posteriores, en consecuencia se optó por optimizar estos parámetros mediante el algoritmo de Levenberg-Marquardt.

5.3.1. Optimización con Levenberg-Marquardt

Para evaluar los resultados de aplicar el algoritmo de optimización numérica mencionado en la sección 4.3.2 se creó un programa tomando en cuenta todos los aspectos de dicha sección, y para probar y comparar la eficiencia se utilizan los mismo 48 puntos. Al programa le toma 20 iteraciones llegar al mínimo con un valor de:

$$F(p) = 1.588771e^{-3} \tag{5.1}$$

Se compararon los resultados del mismo modo que en la sección anterior, midiendo la distancia de los puntos en metros rotados y trasladados $scnP$ con los puntos que resultan de la intersección de las rectas de imP con el plano del piso, esto se realiza aplicando los parámetros: R, T, n y d resultantes de la optimización. En el primer punto se obtuvo un error de $-0.00145m$ en X, $0.00594m$ en Y y en Z $0.01289m$. En el último punto en el cual se notaba más error de la rotación y traslación, después de la optimización se obtuvieron los siguientes resultados: en X $0.0449m$, en Y $0.0423m$ y en Z $0.1424m$. El mayor cambio en la disminución del error se puede apreciar en el último punto en la coordenada X ya que pasa de un error de $26.5cm$ a $4cm$. En la figura 5.10 se puede observar la comparación de

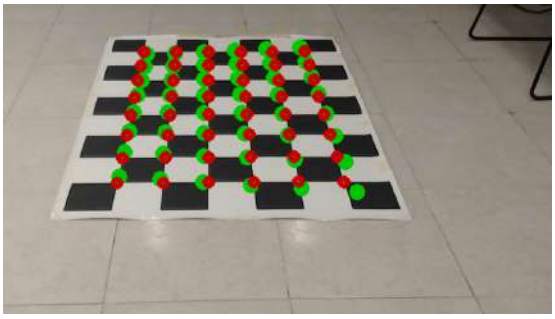

Figura 5.10: Reproyección de puntos después de la optimización

los puntos después de optimizar. Los resultados son superiores en comparación de lo que había antes de la optimización.

5.3.2. Correción de la imagen y recalibración de la cámara

En la sección anterior se mostró que a pesar de utilizar optimización numérica aún hay un ligero error de $4cm$ en el peor de los caso y como se busca que la proyección de los puntos 3D en la imagen con los puntos establecidos sean la más precisa posible, se recalibró la cámara (a diferentes resoluciones) con la que se realizan los experimentos mediante herramientas de OpenCV y esta vez se corrigió la distorsión de la imagen mediante los coeficientes de distorsión obtenidos de la

recalibración. Todo lo anterior es con el objetivo de aumentar la precisión de la estimación de la matriz de rotación, el vector de traslación y los parámetros de la ecuación del plano.

En la imagen de la izquierda de la figura 5.11 se puede apreciar la imagen antes

Figura 5.11: Corrección de la imagen

de la corrección y en la de la derecha aplicando la corrección.

Después de aplicar lo anterior (recalibración y corrección de la cámara) y antes de optimizar se obtienen resultados muy superiores a los conseguidos hasta ahora, ya que $F(p) = 4.10149e-07$, el resultado de comparar los puntos y dibujarlos se puede apreciar en la figura 5.12.

Aplicando nuevamente la optimización numérica a partir del resultado anterior

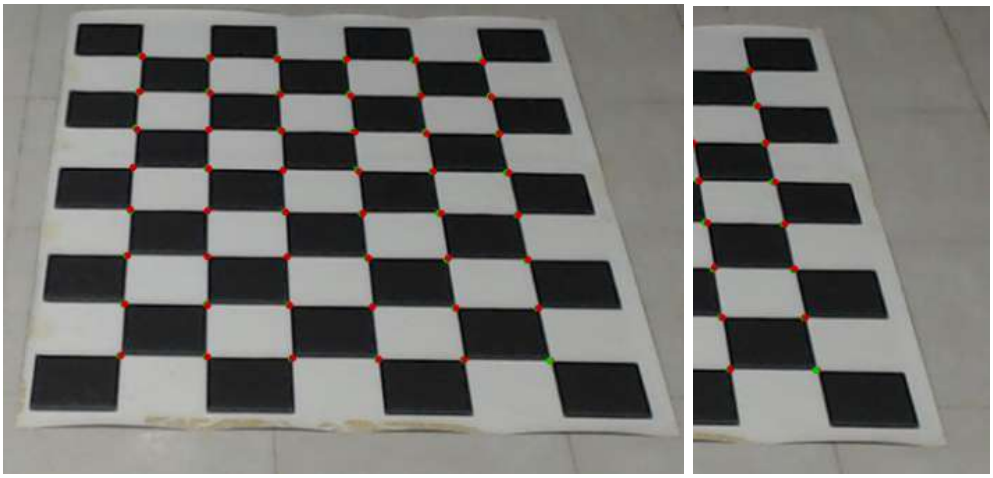

Figura 5.12: Estimación de R y T antes de la optimización

se obtienen resultados aún mejores, ya que después de 1000 iteraciones se obtiene $F(p) = 2.908337e-10$, el resultado gráfico se ve en la imagen 5.13.

5.4. Generación de secuencia óptima de marcas en el piso y pantalla

En el apartado 4.4 se describe el algoritmo para la generación de instancias óptimas con las cuales se realiza la captura de datos necesarios para el algoritmo

Figura 5.13: Estimación de R y T después de la optimización

clasificador, a continuación se presentan los resultados de implementar dicho algoritmo y que parámetros se eligieron para su elaboración.
La captura de datos se realizó en el CLIR de la Facultad de Matemáticas de la UADY por lo que los parámetros iniciales para la generación de las instancias están limitados al área del laboratorio, y en vez de una pantalla se utilizó para los experimentos un pintarrón, los más importantes a considerar son los siguientes:

- Umbrales con los cuales se puede decidir si hay una variación sigficativa en la mirada.- De acuerdo al análisis desarrollado en 5.2 una buena elección es $thres_x = 0.3$ y $thres_y = 0.1$ radianes.

- Ecuación del plano del piso.- Los valores de los parámetros del plano después de la optimización numérica son: $n = (-1.513929e-02m,\ 9.542600e-01m,\ 3.126682e-01m)$ y $d = 2.157112m$

- Rotación de la unidad Pan-Tilt.- Es necesario conocer el ángulo para el cálculo de S (sección 4.2.2): θ=-18.52 radianes, el ángulo es negativo debido a la regla de la manos derecha.

- Eje X.- La posición de la primera marca es en la coordenada $-1.9m$, el intervalo de distancia entre cada punto es de $0.3m$, lo anterior es con respecto al eje X. Se seleccionaron 11 columnas de marcas.

- Eje Z.- La posición de la primera marca es en la coordenada $0.8m$, el intervalo es de $0.3m$ y cuenta con 10 filas de marcas. Lo anterior es con respecto al eje Z.

- Dimensiones del pintarrón donde se desplaza la figura S.- En eje Y $1.17m$ y en X $2.375m$

- La figura en el pintarrón S tiene permitido desplazarse 23 posiciones a lo largo del eje Y y 47 a lo largo del eje X.

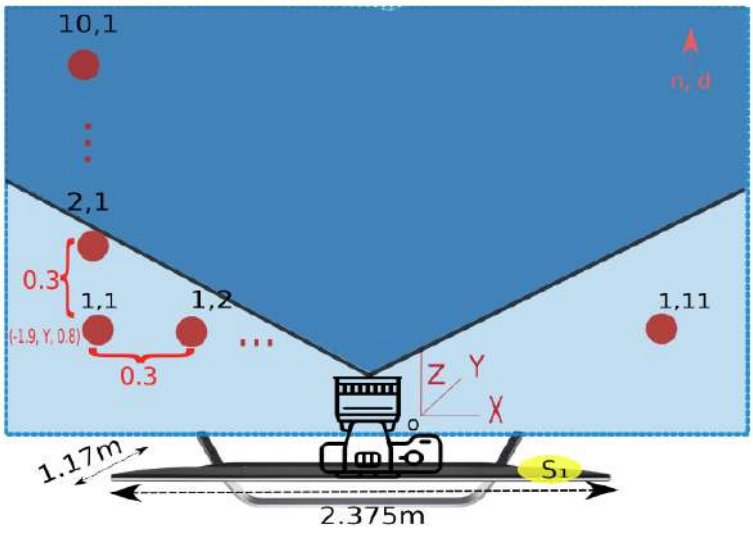

Figura 5.14: Parámetros iniciales para el algoritmo de generación de marcas

El algoritmo se ejecutó tomando en cuenta los parámetros anteriores y guardando las instancias en un archivo de texto. El programa genera el archivo con $642,114$ instancias posibles en $56.9Mb$ empleando un tiempo de 5.02s. Cabe destacar que el número de instancias generadas se ve afectado aumentando o disminuyendo el número de posibles posiciones en pantalla, y el número de columnas y filas en la malla del piso, por ejemplo, al utilizar en lugar de 23 posiciones en el eje Y se utilizan 117 y en el eje X en vez de 47 se eligen 237 da como resultado un archivo de $1.4Gb$ con 16'333,248 instancias y le toma a la computadora 1 minuto y 39s generar las instancias. Tomando esto en consideración para el algoritmo de la siguiente etapa (búsqueda de secuencia óptima) no es conveniente utilizar demasiadas instancias, ya que le toma bastante tiempo encontrar una secuencia óptima de puntos.

Figura 5.15: Instancias iniciales para el algoritmo de generación de marcas

De acuerdo a la descrito en la sección 4.4.2 (Búsqueda de secuencia óptima

mediante algoritmo de ordenamiento) se desarrolló el algoritmo tomando como entrada las 642, 114 instancias generadas en la etapa anterior y se experimentó con la generación de 20 instancias secuenciales. Los ordenamientos, búsquedas y agrupamientos en todas las instancias le toman al algoritmo 138m 28s generar la lista con los 20 elementos, en la figura 5.16 se encuentra un ejemplo de la salida de este algoritmo con la lista de las 20 instancias óptimas, nótese que hay instancias con una distancia máxima en el eje Z de hasta 3.5m de la cámara.

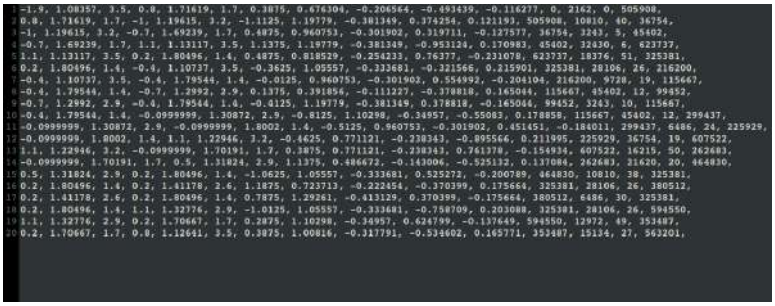

Figura 5.16: Secuencia óptima de marcas

5.5. Protocolo de captura de datos

En las secciones anteriores del presento capítulo ya se han descrito casi todos los elementos para realizar la captura de datos:

- Estimación del plano del piso a través de la matriz de rotación y traslación optimizadas.
- Identificación 3D de la ubicación del rostro de las personas y de la figura que observan en pantalla.
- Ecuaciones para describir los ángulos de la mirada.
- Generación de secuencia de instancias (marcas en el piso) óptimas para la captura.

Sin embargo aún falta describir como se lleva a cabo la captura de los datos necesarios para el algoritmo de entrenamiento.

5.5.1. Proyección en tiempo real de instancias óptimas para dibujar en el piso

El protocolo de captura comienza pintando en el piso las instancias óptimas generadas a través el algoritmo descrito en la sección 4.4, para ello se desarrolló un programa que recibe como entrada la lista de las instancias de marcas, las proyecta y pinta en tiempo real sobre lo que esté capturando la cámara, esto se realiza como guía o referencia al momento de pintarlas físicamente en el piso del

laboratorio de experimentación.
Como se mencionó en el apartado 4.4.1 las instancias se eligen con la restricción (entre otras) de que el rostro de la persona P esté dentro del campo visual de la cámara, esto significa que pueden haber marcas F en el piso que no se encuentren dentro del campo visual de la cámara pero si su rostro, por ejemplo, aquellas que se encuentran más cercanas a la cámara, imagen 5.17.

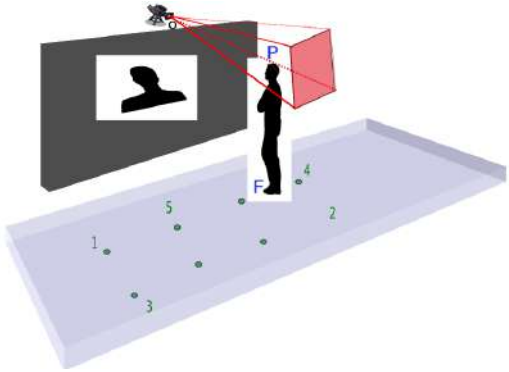

Figura 5.17: Campo visual

Dibujo de rectas graduadas

Para solucionar el problema de instancias F generadas no visibles debido a la restricción del campo visual, se desarrolló un algoritmo que utiliza la rotación, traslación y parámetros del plano para dibujar en el piso de la escena las marcas F, en todos los casos se dibujan lineas graduadas paralelas a los ejes X y Z del marco de referencia. A las rectas se les dibuja una pequeña marca cada 10 cm, además, se dibuja a un lado de las marcas el valor de su coordenada 3D. Con esto se busca tomar como referencia las marcas y las rectas graduadas que si se alcanzan a ver y moverse sobre los mismos ejes hasta marcar los puntos F del conjunto de instancias que no se alcanzan a visualizar en pantalla, esto se puede realizar con ayuda de un flexómetro.
Al algoritmo se le añadió la opción de simular la altura de los sujetos de experimentación mediante una recta ortogonal al plano del piso, esto se realizó por dos motivos:

- Verificar que la estimación de rotación, traslación y plano obtenidos de etapas anteriores son precisos
- Corroborar que el rostro de las personas P que vayan a realizar el experimento se encuentra dentro del campo visual de la cámara

Los resultados del algoritmo anterior se pueden ver en la figura 5.18 y como se puede apreciar hay unas marcas en las que no se alcanza a visualizar la F pero la P si. Las marcas están enumeradas para indicarle el orden que la persona deberá seguir durante la captura de sus datos.
Las rectas amarillas fueron colocadas y pintadas con respecto a la altura real del

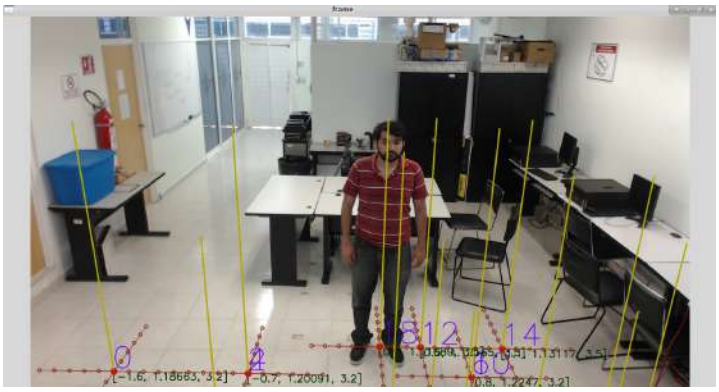

Figura 5.18: Rectas graduadas

sujeto de experimentación que aparece en la misma imagen. Las rectas amarillas (ortogonales al plano) indican que la estimación de la normal del plano n fue calculada con precisión, y en cuanto al parámetro de la distancia d hay un pequeño error de 3cm.
Ya con las marcas pintadas en el plano del piso y pintarrón el siguiente paso consistió simplemente en ir capturando las fotografías de las personas siguiendo la secuencia de instancias generadas previamente, en la imagen 5.19 se encuentra una de las capturas realizadas en esta etapa de los experimentos. En las siguientes secciones se describirá la segunda fase del proyecto: clasificación de la mirada utilizando aprendizaje automático.

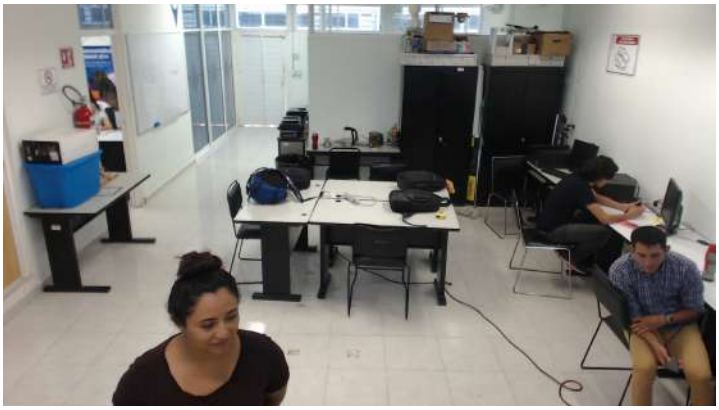

Figura 5.19: Imagen obtenida durante la etapa de captura

5.6. Red neuronal

En esta sección se describirá todo lo relacionado a la experimentación con la red neuronal, por ejemplo: la etapa de preparación de datos para introducir a la red, entrenamiento, resultados al variar la topología, etc.

5.6.1. Detector de rostros y filtrado de los ejemplos útiles

La primera etapa de esta sección consiste en preparar las imágenes y datos obtenidos de la captura para introducirlos en la red neuronal (incluyendo P y S), esto conlleva a que en cada imagen se obtenga y guarde la región (ROI) en donde se encuentra el rostro de las personas, por lo tanto se desarrolló un programa con el algoritmo de Viola y Jones para detectar el rostro y guardar la región donde se encuentre.
Existen algunos casos en los que a pesar de que el rostro está visible en la imagen el algoritmo de Viola y Jones no logra detectarlo, en estos casos se descarta la imagen, ya que para lograr estimar la mirada de las personas con el sistema que se desarrolla es fundamental detectar previamente el rostro con el algoritmo de Viola y Jones, además, es tedioso y tardado ir seleccionando manualmente las regiones donde se localizan los rostros en las imágenes.
En la figura 5.20 se pueden ver algunos ejemplos de las imágenes capturadas y procesadas con el detector de Viola y Jones.

Figura 5.20: Imágenes procesadas con Viola y Jones

En la imagen 5.21 se presenta un falso positivo del algoritmo detector, el cual es descartado para el conjunto de entrenamiento.
Se capturaron en total 383 imágenes y después de procesarlas con el detector de Viola y Jones se obtuvieron 244 rostros útiles para el conjunto de entrenamiento. En la figura 5.22 se encuentran algunos de los rostros que si se utilizan para el conjunto de entrenamiento. Las resoluciones de los rostros capturados son variadas, pero como se mencionó en la sección 4.5.1, se deben redimensionar todas a un estándar, debido a que el número de características y neuronas de entrada depende de dicha resolución, la resolución elegida para estas pruebas fue de 75x75 pixeles.

5.6.2. Entrenamiento y pruebas

Con los datos listos para el algoritmo de clasificación se procedió a dividirlos en dos conjuntos, el de entrenamiento y el de prueba, a continuación se presentan

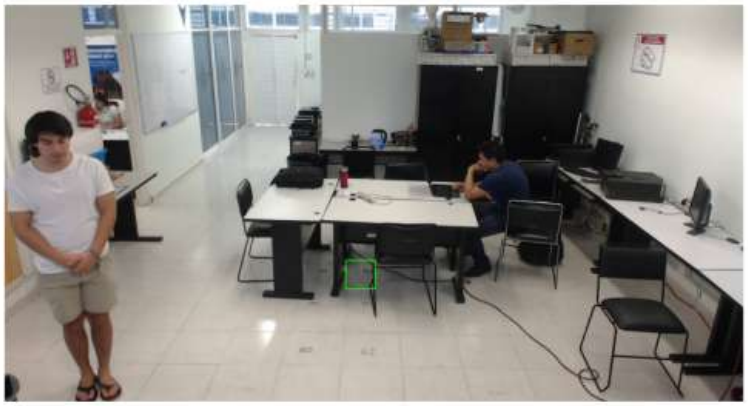

Figura 5.21: Imagen procesada con Viola y Jones

Figura 5.22: Imágenes de los rostros guardadas

los resultados de experimentar con 204 imágenes o ejemplos de entrenamiento y 40 de prueba. Cabe destacar que como criterios de paro se utilizaron: un máximo de 1000 iteraciones o un error menor o igual a 0.00001. La eficiencia se mide como el porcentaje de aciertos encontrados en el conjunto de prueba y los aciertos son determinados como se explica en la sección 4.5.2

Dos neuronas de salida

Primero se experimentó discretizando el área de visualización de S (pintarrón) que observaban los sujetos de los experimentos en 2 secciones de misma longitud, partiendo el eje X a la mitad (véase la figura 5.23).

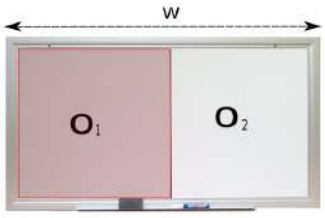

Figura 5.23: Pintarrón divido en dos secciones

En la tabla 5.1 se presentan los resultados de utilizar una capa oculta, se puede notar que incluso con una neurona en dicha capa oculta se consiguen buenos resultados (95 % de aciertos) en 0.057 ms, es decir, le toma 0.57 ms a la red

Cuadro 5.1: Una capa oculta

Topología	Eficiencia	Tiempo
2-2	95 %	0.057077ms
5-2	92.5 %	0.063579ms
10-2	85 %	0.10723ms
25-2	97.5 %	0.212ms
30-2	97.5 %	0.279ms
50-2	87.5 %	0.459ms

Cuadro 5.2: Dos capas ocultas

Topología	Eficiencia	Tiempo
2-2-2	60 %	0.060679ms
5-2-2	95 %	0.062157ms
5-4-2	97.5 %	0.050279ms
5-5-2	97.5 %	0.06059ms
10-10-2	92.5 %	0.115969ms
50-10-2	97.5 %	0.466266ms

neuronal propagar un ejemplo de prueba hacia delante. Los mejores resultados con una capa se logran a partir 25 neuronas, sin embargo el tiempo se ve afectado incrementando 3.7 veces más.

En la tabla 5.2 se puede notar que utilizando dos capas ocultas (5 neuronas en la primera y 4 en la segunda) se producen resultados bastante buenos y en un tiempo menor al empleado cuando se utiliza solamente una capa.

En la tabla 5.3 se encuentran los resultados de utilizar 3 capas ocultas, aunque se logra el mismo máximo posible de 97.5 % que en las topologías anteriores el tiempo empleado es mayor.

Tres neuronas de salida

Ahora se presentan los resultados de utilizar tres clases en la clasificación de la mirada, por lo tanto en la capa de salida de la red hay tres neuronas. En la imagen 5.24 se encuentra ilustrada la segmentación del pintarrón en este caso.

Con solo una capa oculta (tabla 5.4) el mejor resultado que se obtuvo es 87.5 %

Cuadro 5.3: Tres capas ocultas

Topología	Eficiencia	Tiempo
2-2-2-2	50 %	0.068744ms
5-5-5-2	62.5 %	0.060773ms
4-5-5-2	95 %	0.056476ms
5-10-5-2	72.5 %	0.089464ms
5-2-10-2	97.5 %	0.061379ms
50-20-50-2	97.5 %	0.548939ms

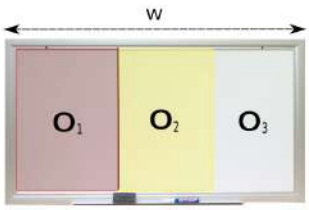

Figura 5.24: Pintarrón divido en tres secciones para tres neuronas de salida

Cuadro 5.4: Una capa oculta

Topología	Eficiencia	Tiempo
2-3	47.5 %	0.055246ms
4-3	87.5 %	0.05233ms
5-3	85 %	0.063973ms
20-3	80 %	0.182204ms

utilizando 4 neuronas en dicha capa.

En la tabla 5.5 se puede observar que usando dos capas ocultas se obtienen resultados superiores a usar una, incluso se llega a una eficiencia de 97.5 %. Los mejores resultados se consiguen al colocar más de 10 neuronas en las capas ocultas. En la tabla 5.6 se puede apreciar que a pesar de haber más neuronas y capas no se alcanzan resultados tan buenos como con dos capas ocultas. De los experimentos con tres clases se puede concluir que se alcanzan resultados similares a utilizar dos clases: 97.5 %

Cuatro neuronas de salida

En esta sección se presentan los resultados de utilizar cuatro clases en la clasificación llevando a cabo una partición del pintarrón como la de la figura 5.25.

En la tabla 5.7 se presentan los resultados de utilizar una capa oculta, como se puede apreciar éstos son inferiores que los obtenidos en los experimentos anteriores con un número menor de clases, ya que en el mejor de los casos se obtiene un 80 % de eficiencia en 0.122ms.

En la tabla 5.8 se puede observar que la red neuronal tiene un mejor desempeño al utilizar dos capas ocultas y sobre todo si se utilizan en ellas más de 20 neuronas

Cuadro 5.5: Dos capas ocultas

Topología	Eficiencia	Tiempo
5-2-3	62.5 %	0.059546ms
3-2-3	85 %	0.080533ms
10-5-3	75 %	0.104578ms
12-10-3	95 %	0.103967ms
18-10-3	97.5 %	0.168328
20-20-3	92.5 %	0.163037ms

Cuadro 5.6: Tres capas ocultas

Topología	Eficiencia	Tiempo
2-2-2-3	40 %	0.06782ms
10-15-10-3	72.5 %	0.181769ms
10-15-15-3	82.5 %	0.110591ms
18-20-20-3	85 %	0.190253ms
18-25-10-3	95 %	0.177355ms
40-32-15-3	95 %	0.409067ms

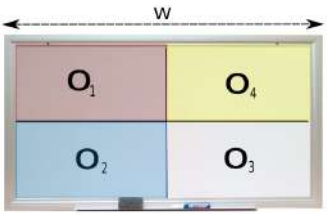

Figura 5.25: Pintarrón divido en cuatro regiones

Cuadro 5.7: Una capa oculta

Topología	Eficiencia	Tiempo
2-4	67.5 %	0.053ms
5-4	72.5 %	0.105ms
12-4	80 %	0.122ms
19-4	67.5 %	0.19ms
27-4	60 %	0.27ms

Cuadro 5.8: Dos capas ocultas

Topología	Eficiencia	Tiempo
5-3-4	72.5 %	0.064ms
20-12-4	85 %	0.3167ms
25-16-4	92.5 %	0.213ms
30-9-4	90 %	0.29ms
30-20-4	90 %	0.282ms

Cuadro 5.9: Tres capas ocultas

Topología	Eficiencia	Tiempo
5-12-19-4	67.5 %	0.78ms
20-15-20-4	77.5 %	0.164ms
22-20-20-4	87.5 %	0.157ms
20-15-17-4	80 %	0.424ms
25-9-10-4	72.5 %	0.26ms

llegando incluso a una eficiencia de 92.5 %.

Utilizando tres capas ocultas el desempeño del algoritmo decae y lo máximo que se puede obtener es 87.5 % de aciertos con una topología de $22 - 20 - 20 - 4$ en 0.157ms (tabla 5.9).

Entrenamiento y experimentos con descriptores HOG

A continuación se presentan y discuten los resultados de la red neuronal con una metodología similar a la anterior pero en lugar de utilizar toda la imagen del rostro(de 75x75 pixeles) como vector de características, se utilizan los descriptores HOG, con ello se busca un tiempo de propagación de la red menor y una eficiencia superior. Los valores de los parámetros utilizados para generar el vector de características son en algunos casos los mismos que los utilzados en el ejemplo de la sección 3.5 y del artículo original [45]:

- *winSize*.- Es el tamaño del rostros ya redimensionado: 75x75 pixeles
- *blockSize*.- 15x15
- *cellSize*.- 5x5
- *nbin*.- El autor de HOG recomienda utilizar un valor de 9 para capturar gradientes entre 0 y 180 grados, el mismo valor fue aplicado en estos experimentos

Tomando en cuenta los parámetros anteriores el vector de características HOG resultante para la red neuronal pasa de 5629 a 2029 elementos.

En la tabla 5.10 están los resultados con dos neuronas de salida (2 segmentos en la pantalla), éstos son superiores ya que se logra 100 % de eficiencia con solo una capa oculta de 5 neuronas en tan solo 0.025ms, tal eficiencia no se logró

Cuadro 5.10: Dos clases utilizando descriptores HOG

Topología	Eficiencia	Tiempo
2-2	97.5 %	0.024ms
5-2	100 %	0.025ms
2-2-2	97.5 %	0.0235ms
5-2-2	100 %	0.0438ms
2-2-2-2	100 %	0.029ms

Cuadro 5.11: Tres clases utilizando descriptores HOG

Topología	Eficiencia	Tiempo
2-3	95 %	0.0289ms
27-3	97.5 %	0.084ms
5-2-3	95 %	0.0415ms
10-3-3	97.5 %	0.065ms
10-15-10-3	100 %	0.06558ms

con ninguna topología con 2 clases utilizando toda la imagen como vector de características. En ese caso la mayor eficiencia fue de 97.5 % y se obtuvo con una oculta de 25 neuronas en 0.212ms.

Los resultados de utilizar tres neuronas de salida se muestran en la tabla 5.11, sin utilizar HOG lo máximo de eficiencia que se lograba con este número de clases era 97.5 % ahora se alcanza hasta un 100 % usando 3 capas ocultas. Incluso con una capa se logra 97.5 % mientras que en los experimentos anteriores (sin HOG) lo máximo obtenido con una capa era 80 %.

En la tabla 5.12 se presentan los resultados con 4 neuronas de salida en la red neuronal y con los descriptores HOG como parte del vector de características. Al igual que en los casos anteriores en éste los resultados son mejores a utilizar toda la imagen como vector de características ya que el mejor de esos resultados con 4 clases era de 92.5 % en 0.213ms con una topología de 25-16-4 y ahora con HOG se llega incluso a 95 % de aciertos en 0.0415ms y dos capas ocultas.

Cuadro 5.12: Cuatro clases utilizando descriptores HOG

Topología	Eficiencia	Tiempo
2-4	85 %	0.096ms
27-4	90 %	0.117ms
7-26-4	92.5 %	0.0595ms
9-20-4	95 %	0.0386ms
9-5-21-4	95 %	0.0415ms

5.7. Discusión

Se observó durante la experimentación que es posible conocer los parámetros extrínsecos de la cámara y los parámetros de un plano en el piso con precisión utilizando una sola fotografía, mediante la descomposición de una matriz de homografía entre el plano del piso y el de la imagen. Cabe destacar que se puede contrarrestar los errores generados por la distancia amplia que presenta la cámara del piso ($d = 2.15m$) con el algoritmo de optimización numérica Levenberg-Marquardt.
En la fase de clasificación como era de esperarse al ir aumentando el número de clases (neuronas de salida) en la red disminuye la eficiencia, de igual manera, se demostró que al utilizar los descriptores HOG como parte del vector de características de la red se obtienen resultados superiores en la clasificación en todos los casos y en menor tiempo en comparación de utilizar la imagen completa como vector de características, llegando incluso a 95 % de aciertos en el caso más complejo de los experimentados, es decir, con cuatro neuronas de salida. Además, no está descrito en las tablas presentadas pero el tiempo empleado en el entrenamiento utilizando HOG es menor en todos los casos ya que pasa de un tiempo promedio de 1 minuto (sin HOG) a 5s.

Capítulo 6

Conclusión

El trabajo presentado en esta tesis consiste en el desarrollo de un sistema robusto que tiene como objetivo estimar de manera general la mirada de las personas a partir de fotografías de sus rostros y de otros datos como la ubicación en tres dimensiones de la cabeza, dicha estimación está representada por una región de un plano o pantalla enfrente de ellas que yace sobre el mismo plano $X-Y$ que el marco de referencia (cámara) del sistema. A partir de este objetivo se desarrolló el sistema con una especial atención en la recolección precisa de los datos para el conjunto de entrenamiento del algoritmo de clasificación.
De acuerdo a los resultados obtenidos se demostró que utilizando las imágenes de los rostros detectados y datos geométricos precisos en el conjunto de entrenamiento se logra estimar con alto porcentaje de efectividad la mirada. De igual manera se observó durante la experimentación que los resultados son satisfactorios inclusive con el rostro en una distancia amplia (hasta 3.5m) de la cámara.
Durante las pruebas se mostró como el sistema presenta un desempeño superior con respecto a velocidad de estimación y porcentaje de aciertos, utilizando el descriptor de características HOG como parte del vector de neuronas de entrada en vez de toda la imagen, debido a que este algoritmo se encarga de desechar la información "inútil"de la imagen del rostro, llegando a resultados para dos y tres clases de 100 % de efectividad y 95 % para cuatro clases.

6.0.1. Contribuciones del proyecto

Una de las principales contribuciones del proyecto radica en la recolección que se hizo de una base de datos de 383 imágenes de entrenamiento con rostros de diferentes personas en diferentes orientaciones y ubicaciones. Además, se desarrolló un protocolo para llevar acabo una captura automatizada, precisa y veloz de los datos de las personas una vez calibrado el escenario de experimentación. Este protocolo está descrito de manera detallada de modo que fácilmente se puedan realizar más rondas de capturas de datos y experimentar en sistemas similares que requieran imágenes de rostros de personas.
Otra importante contribución de este trabajo es la que se encuentra en el apartado correspondiente a la estimación del plano del piso en una distancia lejana de la cámara (marco de referencia) mediante una sola imagen. La metodología

propuesta también incluye la implementación de un algoritmo de optimización numérica para refinar dicha estimación.
Se presentó un sistema de realidad aumentada para dibujar coordenadas 3D con precisión en tiempo real en la escena de experimentación, como se aprecia en la imagen 5.18, donde se simula la estatura del sujeto de experimentación mediante rectas que salen del plano del piso. Una de las principales aplicaciones que pudiera tener este sistema está en el área educativa, ya que se pudiera utilizarse para enseñar de manera interactiva como funcionan los sistemas cartesianos tridimensionales.

6.0.2. Trabajo futuro

El proyecto descrito en este documento puede ser mejorado o ampliado en trabajos futuros, se sugieren los siguientes trabajos para lograrlo:

- Se recomienda experimentar con más clases (neuronas de salida) por dos motivos:
  - Analizar y verificar que tanto disminuye el desempeño al intentar estimar mas clases.
  - Todas las publicaciones en esta área reportan resultados a partir de 5 clases. Sería interesante comparar la eficiencia de este sistema con los demás trabajos existentes.
- Experimentar quitando los datos 3D del vector de características y trabajando solo con HOG para analizar que tanto es afectado el sistema.
- El sistema puede ser implementado para funcionar en tiempo real fácilmente, ya que el único dato con el que no se cuenta al momentos de capturar el rostro es la ubicación en tres dimensiones de la cabeza de las personas o como se denominó a lo largo de este documento: P. Este parámetro puede ser encontrada ubicando los pies de las personas, lo cual se puede conseguir con una técnica como la presentada en [55] (véase la imagen 6.1).
  La ubicación de los pies es igual al parámetro F que yace en el plano del piso, por lo tanto se puede pasar a coordenada 3D este parámetro proyectando una recta que parte de este punto de la imagen 2D hacia el plano del piso en busca del punto donde intersecta con el plano (como se realiza en la sección 4.3.1), dicho punto de intersección es la F, finalmente se proyecta un vector normal al plano partiendo de F hacia la región en la imagen donde se encuentra el rostro, este vector que se forma es la P que se necesita introducir como parte del vector de características en la red neuronal.
- Es importante capturar más imágenes con el objetivo de analizar el comportamiento del sistema y hacerlo más robusto. En artículos similares utilizan al menos 1000 imágenes.
- Se diseñó y fabricó una tarjeta electrónica con un microcontrolador, zigbee y una unidad de medición inercial (IMU) para la captura de la rotación (en

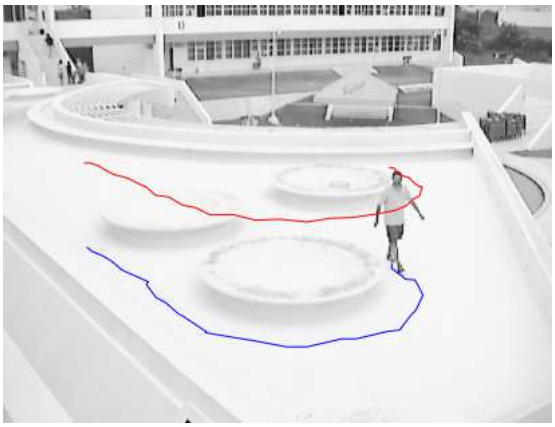

Figura 6.1: Trayectoria de una persona

cuaterniones) de la cabeza con el objetivo de realizar la recolección de datos de manera más precisa a indicarle a las personas que miren en determinadas posiciones, o para descartar las imágenes en las que las personas no dirigen su cabeza con exactitud hacia los objetos. Valdría la pena experimentar con esta tarjeta al momento de realizar dicha captura de imágenes.

Referencias

[1] Murphy-Chutorian, E., & Trivedi, M. M. Head pose estimation in computer vision: A survey. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(4), 607-626, 2009

[2] Wang, J. G., & Sung, E. Study on eye gaze estimation. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 32(3), 332-350, 2002.

[3] Brown, L. M., & Tian, Y. L. Comparative study of coarse head pose estimation. In Motion and Video Computing, 2002. Proceedings. Workshop on, 125-130, 2002.

[4] Dalal, N., & Triggs, B. Histograms of oriented gradients for human detection. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference* , Vol. 1, pp. 886-893, 2005

[5] McCulloch, W. S., & Pitts, W. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4), 115-133. 1943

[6] Rosenblatt, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6), 386. 1958

[7] Minsky, M., & Papert, S. Perceptrons. 1969.

[8] Rumelhart, D. E., Hinton, G. E., & Williams, R. J. Learning internal representations by error propagation *California Univ San Diego La Jolla Inst for Cognitive Science*, (No. ICS-8506). 1985.

[9] Andreas Antoniou, & Wu-Sheng Lu *Practical Optimization, Algorithms and Engineering Applications*. (2007).

[10] Gill F., Murray W., Wright M.& *Practical Optimization*. (1981).

[11] Sanchez-Machay I. Automatización de un sistema de captura de fotografía aérea para la generación de mosaicos de imágenes, (2014)

[12] Ng, J., & Gong, S. Composite support vector machines for detection of faces across views and pose estimation. *Image and Vision Computing*. 20(5), 359-368, 2002

[13] Ng, J., & Gong, S. Multi-view face detection and pose estimation using a composite support vector machine across the view sphere. *Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems, Proceedings. International Workshop* . (pp. 14-21) 1999.

[14] Drucker, H., Burges, C. J., Kaufman, L., Smola, A. J., & Vapnik, V. Support vector regression machines. Advances in neural information processing systems (pp. 155-161). 1997

[15] Li, Y., Gong, S., & Liddell, H. Support vector regression and classification based multi-view face detection and recognition. *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference* (pp. 300-305). 2000

[16] Li, Y., Gong, S., Sherrah, J., & Liddell, H. Support vector machine based multi-view face detection and recognition. *Image and Vision Computing*, 22(5), 413-427. 2004

[17] Murphy-Chutorian, E., Doshi, A., & Trivedi, M. M. Head pose estimation for driver assistance systems: A robust algorithm and experimental evaluation. *Intelligent Transportation Systems Conference,* (pp. 709-714). 2007

[18] Yan, S., Zhang, Z., Fu, Y., Hu, Y., Tu, J., & Huang, T. Learning a person-independent representation for precise 3D pose estimation. *Multimodal Technologies for Perception of Humans*, 297-306. 2008

[19] Duda, R. O., Hart, P. E., & Stork, D. G. *Pattern classification. John Wiley & Sons* 2012

[20] McKenna, S. J., & Gong, S. Real-time face pose estimation *Real-Time Imaging* 4(5), 333-347. 1998

[21] Lades, M., Vorbruggen, J. C., Buhmann, J., Lange, J., Von Der Malsburg, C., Wurtz, R. P., & Konen, W. Distortion invariant object recognition in the dynamic link architecture *IEEE Transactions on computers,* 42(3), 300-311. 1993

[22] Cootes, T. F., Edwards, G. J., & Taylor, C. J. Active appearance models *IEEE Transactions on pattern analysis and machine intelligence,* 23(6), 681-685, 2001

[23] Wilson, H. R., Wilkinson, F., Lin, L. M., & Castillo, M. Perception of head orientation *Vision research* , 40(5), 459-472. 2000

[24] Horprasert, T., Yacoob, Y., & Davis, L. S. Computing 3-d head orientation from a monocular image sequence. *Proceedings of the Second International Conference*, (pp. 242-247). 1996

[25] Nikolaidis, A., & Pitas, I. Facial feature extraction and pose determination *Pattern Recognition,* 33(11), 1783-1791. 2000

[26] Gourier, N., Hall, D., & Crowley, J. L. Estimating face orientation from robust detection of salient facial structures *FG Net Workshop on Visual Observation of Deictic Gestures*, 2004

[27] Rae, R., & Ritter, H. J. Recognition of human head orientation based on artificial neural networks. *IEEE Transactions on neural networks*, 9(2), 257-265. 1998

[28] Ng, J., & Gong, S. Multi-view face detection and pose estimation using a composite support vector machine across the view sphere *Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems, 1999. Proceedings. International Workshop*, 14-21, 1999

[29] Morency, L. P., Rahimi, A., & Darrell, T. Adaptive view-based appearance models *Computer Vision and Pattern Recognition, 2003. IEEE Computer Society Conference* (Vol. 1, pp. I-I). 2003

[30] Huang, J., Shao, X., & Wechsler, H. Face pose discrimination using support vector machines (SVM) *Pattern Recognition, 1998. Proceedings. Fourteenth International Conference* Vol. 1, pp. 154-156. 1998

[31] Gui, Z., & Zhang, C. 3D head pose estimation using non-rigid structure-from-motion and point correspondence *TENCON 2006. 2006 IEEE Region 10 Conference*, pp. 1-3. 2006

[32] Bouguet, J. Y. (2002). Camera calibration tool-box for matlab *http : //www.vision.caltech.edubouguetjcalib_doc*. 2002

[33] Viola, P., & Jones, M. Rapid object detection using a boosted cascade of simple features. *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference*, Vol. 1, pp. I-I. 2001

[34] Tsai, R. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. *IEEE Journal on Robotics and Automation,* 3(4), 323-344. 1987

[35] Kanatani, K. Geometric computation for machine vision *Oxford University Press, Inc.*. 1993

[36] Zhang, Z. (1999). Flexible camera calibration by viewing a plane from unknown orientations *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference* Vol. 1, pp. 666-673. 1999

[37] Moré, J. J. The Levenberg-Marquardt algorithm: implementation and theory *Numerical analysis. Springer, Berlin, Heidelberg* pp. 105-116. 1978

[38] Heikkila, J., & Silven, O. A four-step camera calibration procedure with implicit image correction. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference*, pp. 1106-1112. 1997

[39] Aldana-Iut J. Calibración del sistema cámara-ojo para reconstrucción 3D de estructuras en imágenes de retina. 2009

[40] Crow, F. C. Summed-area tables for texture mapping. *ACM SIGGRAPH computer graphics*, 18(3), 207-212, 1984.

[41] Friedman, J., Hastie, T., and Tibshirani, R. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics*, 28(2), 337-407, 2000.

[42] John, G. H., Kohavi, R., & Pfleger, K. (1994). In Irrelevant features and the subset selection problem. *Machine learning: proceedings of the eleventh international conference.* (pp. 121-129). 1994

[43] Uc-Cetina, V., Brito-Loeza, C., & Ruiz-Piña, H. Chagas Parasite Detection in Blood Images Using AdaBoost. *Computational and mathematical methods in medicine*, 2015.

[44] Hartley R & Zisserman A. A Multiple View Geometry in Computer Vision. 2003

[45] Dalal, N., & Triggs, B. Histograms of oriented gradients for human detection. *Computer Vision and Pattern Recognition. IEEE Computer Society Conference*, Vol. 1, pp. 886-893. 2005

[46] Kanatanu, K. *Statistical Optimization for Geometric Computation: Theory and Practice*, 1996

[47] Rubner, Y., & Tomasi, C. Perceptual metrics for image database navigation *Springer Science & Business Media.* Vol. 594. 2013

[48] Ma, Y., Soatto, S., Kosecka, J., & Sastry, S. S. An invitation to 3-d vision: from images to geometric models *Springer Science & Business Media* Vol. 26. 2012

[49] Wu, J., Pedersen, J. M., Putthividhya, D., Norgaard, D., & Trivedi, M. M. A two-level pose estimation framework using majority voting of gabor wavelets and bunch graph analysis *In Proc. ICPR Workshop Visual Observation of Deictic Gestures.* (2004, August).

[50] Voit, M. CLEAR 2007 evaluation plan: Head pose estimation. 2007

[51] Zhang, Z., Hu, Y., Liu, M., & Huang, T. Head pose estimation in seminar room using multi view face detectors *In International Evaluation Workshop on Classification of Events, Activities and Relationships. Springer, Berlin, Heidelberg.* (pp. 299-304). (2006, April)

[52] Ma, B., Zhang, W., Shan, S., Chen, X., & Gao, W. Robust head pose estimation using LGBP *In Pattern Recognition, 2006. ICPR 2006. 18th International Conference.* (Vol. 2, pp. 512-515). IEEE. (2006, August).

[53] Krüger, N., Pötzsch, M., & von der Malsburg, C. Determination of face position and pose with a learned representation based on labelled graphs. *Image and vision computing.* 15(8), 665-673. (1997)

[54] Voit, M., Nickel, K., & Stiefelhagen, R. Neural network-based head pose estimation and multi-view fusion *International Evaluation Workshop on Classification of Events, Activities and Relationships . Springer, Berlin, Heidelberg.* (pp. 291-298). 2006

[55] Alonzo J. Reconstrucción de la estructura tridimensional de una escena, a partir de trayectorias de cuerpos en movimiento. 2007