

Estimación de la mirada

Rajiv Raciel González González

2017

Resumen

Índice general

Resumen	I
Contenidos	IV
Lista de figuras	VI
Agradecimientos	VII
Nomenclatura	VII
1. Introducción	1
1.1. Introducción	1
1.2. Objetivo general	1
1.3. Objetivos específicos	1
1.4. Estructura de la tesis	2
2. Estado del arte	3
2.0.1. Métodos para la recolección del ground truth	7
2.0.2. Características comunes de los métodos	7
3. Marco teórico	9
3.1. Detector de rostros de Viola y Jones	9
3.1.1. Imagen integral	10
3.1.2. Algoritmo AdaBoost	11
3.1.3. Filtro con estructura en cascada.	12
3.2. Matrices de Rotación	13
3.3. Homografía	13
3.4. Modelo pinhole de cámara	13
3.5. Calibración de cámaras y corrección de imágenes	13
3.6. Optimización numérica y Levenberg-Marquardt	13
3.7. Aprendizaje Automático	13
3.8. Redes neuronales artificiales	13
3.9. Descriptores HOG	13

3.9.1. Cálculo del histograma de gradientes orientados	14
4. Metodología	19
4.1. Descripción general del sistema	19
4.2. Mapeo de coordenadas a ángulos	20
4.3. Ecuaciones de la mirada	22
4.3.1. Desviación de ϕ_y y ϕ_x . Primer caso	22
4.3.2. Desviación de ϕ_y y ϕ_x . Segundo caso	24
4.4. Generación de marcas más significativas en el piso y la pantalla	26
4.4.1. Malla de puntos en el plano del piso y pantalla	26
4.4.2. Generación de secuencia óptima mediante algoritmo de ordenamiento	28
4.5. Rotaciones en los pares de posiciones	28
4.5.1. Cálculo de todas las rotaciones posibles	28
4.6. Proyección de marcas en el piso	28
4.6.1. Proyección de marcas mediante visión computacional, geometría y optimización numérica	30
4.6.2. Optimización Levenberg-Marquardt	34
4.7. Red neuronal	36
4.7.1. Vector de características de entrada	36
4.7.2. Neuronas de salida	37
5. Experimentos	39
5.1. Materiales	39
5.1.1. Dispositivo para adquirir datos	39
5.2. Análisis del comportamiento de los ángulos de mirada ϕ_x y ϕ_y . .	44
5.2.1. Simulación processing	48
5.3. Proyección de marcas mediante visión computacional, geometría y optimización numérica	49
5.3.1. Reproyección de punto en la imagen	49
5.3.2. Optimización Levenberg-Marquardt	51
5.3.3. Corrección de la imagen y recalibración de la cámara	51
5.4. Protocolo de captura de datos	52
5.4.1. Proyección en tiempo real de instancias óptimas para marcar en el piso	54
5.4.2. Captura de datos	56
5.5. Red neuronal	56
5.5.1. Detector de rostros y guardado de los ejemplos útiles . . .	56
5.5.2. Entrenamiento y pruebas	58

6. Resultados	64
7. Conclusiones	65
A. Example Appendix	66
B. Referencias	67
B.1. Referencias	67

Índice de figuras

3.1.	Ejemplo de imagen integral	10
3.2.	Características Haar	10
3.3.	Características tipo Haar sobreuestas	11
3.4.	Diagrama de nodos	12
3.5.	Detección de chagas en la sangre mediante el método de Viola y Jones	13
3.6.	Imagen de ejemplo para HOG	14
3.7.	Kernels	15
3.8.	Izquierda: valor absoluto del gradiente x. Centro: valor absoluto del gradiente y. Derecha: magnitud del gradiente	15
3.9.	Valores de los gradientes de celda 8x8	16
3.10.	Cálculo histograma de gradientes	16
3.11.	Histograma final de celda de 8x8	17
3.12.	Normalización en bloque llevada a cabo desplazando bloques de 16x16	18
3.13.	18
4.1.	21
4.2.	21
4.3.	Escenario y aspectos a tomar en cuenta	22
4.4.	24
4.5.	25
4.6.	25
4.7.	26
4.8.	29
4.9.	31
4.10.	Representación de un plano en el espacio	33
4.11.	intersección de una recta con un plano	34
4.12.	Red neuronal artificial	36
4.13.	Características	37
4.14.	Neuronas de salida	38

5.1.	40
5.2.	40
5.3.	41
5.4.	42
5.5.	42
5.6.	43
5.7.	tarjeta con el IMU	44
5.8.	batería recargable	44
5.9.	tarjeta receptora de datos	45
5.10.	46
5.11.	46
5.12.	47
5.13.	47
5.14.	48
5.15.	49
5.16.	50
5.17.	Reproyección de puntos	50
5.18.	Reproyección de puntos	52
5.19.	Corrección de la imagen	52
5.20.	Estimación de R y T antes de la optimización	53
5.21.	Estimación de R y T después de la optimización p	53
5.22.	Campo visual	54
5.23.	Campo visual	55
5.24.	Imagen obtenida durante la etapa de captura	56
5.25.	Imagenes procesadas con Viola y Jones	57
5.26.	Imagen procesada con Viola y Jones	57
5.27.	Imágenes de los rostros guardadas	57
5.28.	Pintarrón divido en dos secciones	58
5.29.	Pintarrón divido en tres secciones	59
5.30.	Pintarrón divido en cuatro secciones	61

Agradecimientos

CONACYT

Capítulo 1

Introducción

1.1. Introducción

Conocer la posición, orientación y movimiento 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 a como se realiza actualmente, dando como resultado en útiles aplicaciones como: video conferencias, controles o interfaces hombre-máquina especiales, aplicaciones de realidad virtual. Además, si se toma en cuenta hacia dónde está mirando la persona y el tiempo que lleva en cierta posición puede proporcionar información adicional que ayudaría a inferir si está muy atenta a lo que está observando o si la persona está durmiendo.

Por medio del análisis de imágenes capturadas a personas y en combinación con algoritmos de aprendizaje automático es posible conocer la posición y orientación de sus cabezas con respecto a un marco de referencia, la información obtenida puede ayudar a conocer qué es lo que están observando 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.

1.2. Objetivo general

Detectar el rostro de las personas y estimar su pose relativa a un plano virtual situado enfrente de ellas, 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 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

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 [Murphy et al, 2009]. El estimador debe ser robusto a distorsiones causadas por la cámara, a las expresiones faciales, al cabello, al vello facial y a objetos que ocluyen la cabeza como por ejemplo unos lentes o sombreros.

Existen algunos aspectos que se deben tomar en cuenta al realizar investigación en de 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. [Wang et al, 2009]. Actualmente se han realizado una gran diversidad 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 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. Las clasificaciones son:

- *Métodos de apariencias de plantillas.*- 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, hace la correspondencia con las más similares de las plantillas. El método no requiere entrenamiento con ejemplos negativos, solo requiere los ejemplos etiquetados.

Desventajas:

- (a) Solo estiman posiciones discretas
- (b) Se requiere saber dónde se localiza el rostro
- (c) Ineficiente con muchos ejemplos
- (d) Ineficiente con imágenes de la misma persona del entrenamiento y durante la prueba

■ *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.

Ventajas:

- (a) La detección y localización no son etapas diferentes
- (b) Son buenos con imágenes con alta y baja resolución

Desventajas:

- (a) Se requieren bastantes imágenes de entrenamiento
- (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).

■ *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.

Desventajas:

- (a) No es claro cómo se utilizará 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 entrenamiento
- (c) Arrojan los resultados más exactos en la práctica.

- *Métodos de variedad-embebida o manifold.*- Este método busca 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.

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 muchos escenarios de entrenamiento en el mundo real, representan un problema. Para contrarrestar lo anterior es necesario entrenar un manifold con múltiples personas, 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 bastante veloz en comparación con otros métodos.
- *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 anotados (comentados), 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 locaciones 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 rígidos o deformables.

El proceso de pareo (matching) se realiza con el algoritmo Elastic Graph

Matching, 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 la estimación de la pose un conjunto de grafos es creado para cada pose discreta y es comparada con una nueva vista de la cabeza. El conjunto de grafos con mayor similitud asigna una pose discreta a la cabeza.

Desventajas:

- (a) La estimación de la pose es discreta, requiriendo muchos conjuntos de grafos y comparar los conjuntos con muchas deformaciones resulta costoso computacionalmente.
- (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) 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. El movimiento de yaw puede ser calculada con la distancia entre los ojos, roll con el ángulo que se forma con la línea de los ojos y el horizonte y pitch con la distancia de la punta de la nariz y la línea de los ojos.

Desventajas:

- (a) Se necesita que el rostro se encuentre cerca de la cámara para ver todas las líneas del rostro.
- (b) Es difícil es encontrar las características con alta precisión y exactitud.
- (c) Son métodos sensibles a las occlusiones 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 muy 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 puro.

2.0.1. 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, 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, 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. Entre las desventajas es que son sensibles al ruido y presencia de metales.
- Sensores iniciales.- Utilizan acelerómetros y giroscopios y se reduce el ruido con un filtro de kalman. No miden posición solo orientación en tres DOF.
- Sistemas ópticos de captura de movimiento.- Son robustos y muy costosos, son usados en captura profesional del movimiento articulado del cuerpo.

2.0.2. Características comunes de los métodos

Así como existen muchas 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 [Brown et al, 2002], support vector machine [Huang et al, 1998], y los métodos de boosting como el floatboost [Zhang et al, 2006].
- Se logra mucha 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 estimación de la mirada de las personas en un plano virtual enfrente de ellas es necesario conocer algunos conceptos, definiciones y algunos conocimientos que sirvan de apoyo al desarrollo del presente trabajo de tesis. En este capítulo se describen las bases para llevar a cabo el proyecto, dichas bases consisten principalmente de visión computacional, geometría proyectiva, algoritmos de aprendizaje supervisado, algoritmos de optimización numérica....

3.1. Detector de rostros de Viola y Jones

Como ya se ha mencionado anteriormente para estimar la mirada con precisión se necesita un sistema de detección y seguimiento de ojos además de múltiples cámaras y el sistema de estimación de pose de la cabeza, habiendo señalado lo anterior cabe destacar que el proyecto que se desarrollará más que estimar la mirada con precisión (además de la pose de la cabeza) representándola como un vector normal al iris, se pretende estimar una región en un plano virtual enfrente de ellas, la región indicaría lo que están observando las personas de la escena que tienen enfrente.

En el presente proyecto para la estimación de la pose de la cabeza de las personas en primer lugar se debe detectar dónde se encuentran los rostros (como se ha mencionado) para luego estimar su pose, ya que el sistema consiste de muchas etapas, se requiere que la detección se haga lo más rápido posible y eficientemente, por lo tanto se decidió utilizar uno de los algoritmos de detección más utilizados 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, el método consta principalmente de las siguientes etapas.

3.1.1. Imagen integral

En esta etapa se utiliza una representación intermedia de la imagen conocida como imagen integral o también conocida como tablas de áreas sumadas [Crow 1984]. 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 pixel. 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

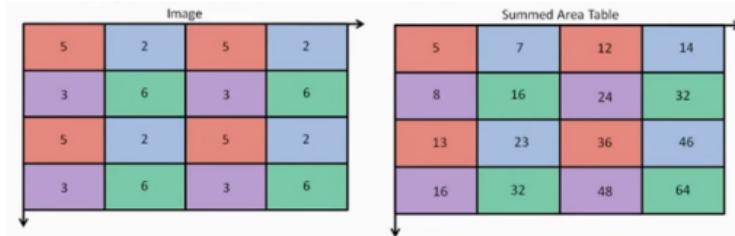


Figura 3.1: Ejemplo de imagen integral

imágenes basado en el valor de simples características, los sistemas basados en características operan muchos más rápido que los basados en píxeles. La imagen integral puede ser usada para estimar el valor de características tipo Haar, las características Haar se visualizan como rectángulos adyacentes blancos y negros, el valor que generan se calcula de la diferencia de la suma de los píxeles 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 muy buena representación de imágenes la cual soporta un efectivo entrenamiento.

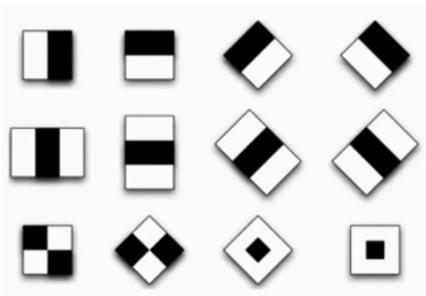


Figura 3.2: Características Haar

3.1.2. Algoritmo AdaBoost

Los métodos de Boosting [Friedman et al 2000] básicamente consisten en combinar la eficiencia de muchos 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 entrenó con conjuntos de imágenes etiquetadas como positivas y negativas, el algoritmo de AdaBoost fue utilizado para entrenar al clasificador y seleccionar un conjunto pequeño con las características Haar más importantes de una muy 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 fue elegido AdaBoost como algoritmo seleccionador de características por Viola y jones en su detector, es la increíble velocidad que presenta; usando AdaBoost, un clasificador de 200 características puede ser construido en alrededor 10-11 operaciones de procesador. Las primeras características seleccionadas por AdaBoost tienen mucho sentido y son fáciles de interpretar, una de esas características parece 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.

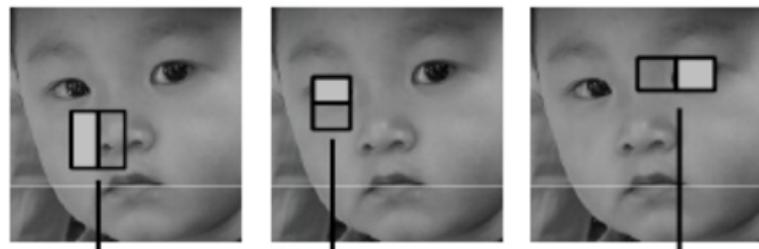


Figura 3.3: Características tipo Haar sobreuestas

3.1.3. 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 mayor??a 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 mayor??a de las sub-ventanas o subregiones son rechazadas en etapas tempranas. Se le denominó estructura en ?cascada? debido a la forma que presenta al momento de procesar las sub- ventanas. 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 as?? sucesivamente. 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 caracter??stica Haar en diferentes escalas y posiciones de la imagen.

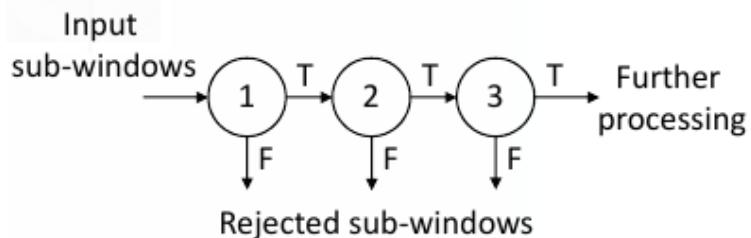


Figura 3.4: Diagrama de nodos

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, mediante las muestras de sangre de los infectados [Uc Cetina et al 2015].

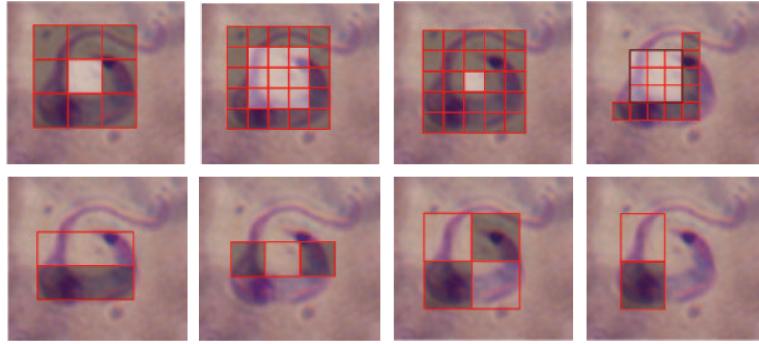


Figura 3.5: Detección de chagas en la sangre mediante el método de Viola y Jones

3.2. Matrices de Rotación

3.3. Homografía

3.4. Modelo pinhole de cámara

3.5. Calibración de cámaras y corrección de imágenes

3.6. Optimización numérica y Levenberg-Marquardt

3.7. Aprendizaje Automático

3.8. Redes neuronales artificiales

3.9. Descriptores HOG

Se comenzará esta sección describiendo de manera general qué es un descriptor de características “feature descriptor” y luego se hará mención en específico de los descriptores tipo HOUGH. Una descriptor de características de una imagen o de una región de una imagen es que simplifica la imagen extrayendo información útil y desecharando toda la información extraña. Tipicamente 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 [Dalal et al, 2005] 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.

La siguiente pregunta que sale a la vista es ¿qué tipo de características son úti-

les para la etapa de clasificación del presente trabajo de tesis?. Al utilizar el descriptor de características se busca que cumpla 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 porque la magnitud de los gradientes es mayor alrededor de los bordes y las esquinas (regiones con cambios abruptos de intensidad) y es bien conocido que los bordes y las esquinas ofrecen mucha más información sobre la forma del objeto que las regiones planas.

3.9.1. 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 3.6 para detectar peatones, como en el artículo original.

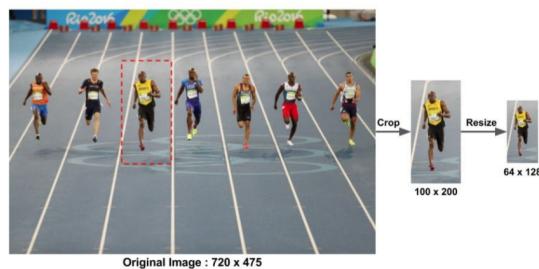


Figura 3.6: Imagen de ejemplo para HOG

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 3.6 se puede observar que se eligió una imagen de 720x475 y en ella se seleccionó 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 facilmente filtrando la imagen con los siguientes kernels:

-1	0	1

-1
0
1

Figura 3.7: Kernels

La magnitud y dirección del gradiente se obtienen aplicando las siguientes fórmulas:

$$g = \sqrt{g_x^2 + g_y^2} \quad (3.1)$$

$$\theta = \arctan \frac{g_y}{g_x} \quad (3.2)$$

En la figura 3.8 se observan los resultados de los gradientes: Nótese que la magni-



Figura 3.8: Izquierda: valor absoluto del gradiente x. Centro: valor absoluto del gradiente y. Derecha: magnitud del gradiente

tud 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 3.8 se pudiera deducir facilmente que es una persona corriendo.

En HOG, en cada pixel, 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.)

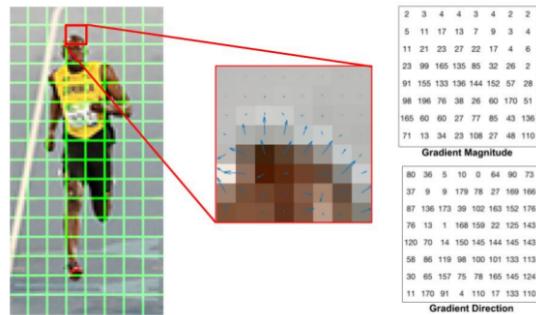


Figura 3.9: Valores de los gradientes de celda 8x8

En la figura 3.9 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 sobrepuertas 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 correspondientes a los ángulos 0, 20, 40...160. En 3.10 se encuentra el proceso del cálculo del histograma, la selección del bin (celda del arreglo) está basado en la dirección y el valor que va dentro del bin es la magnitud del gradiente.

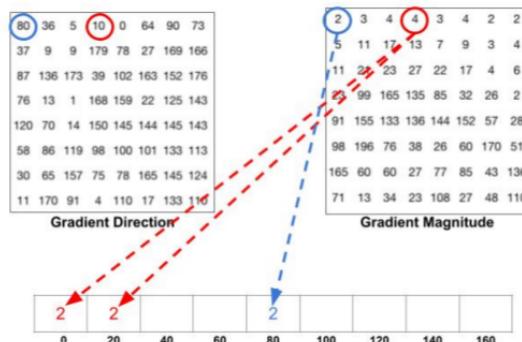


Figura 3.10: Cálculo histograma de gradientes

La contribución de todos los pixeles en la celda de 8x8 son sumados para

crear el histograma de 9 bins, para la celda de la figura 3.9 el histograma final se encuentra en la imagen 3.11, se puede notar que hay más en los bins cerca de los 0 y 180 grados, esto indica que los gradientes están apuntando arriba o abajo.

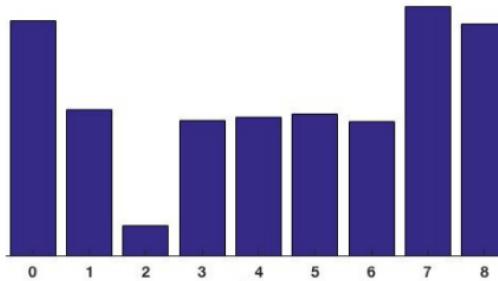


Figura 3.11: Histograma final de celda de 8x8

Paso4: 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 [Dalal et al, 2005] 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 investigación original, se concatenan cuatro histogramas de 8x8 (histograma de 9x1) para formar un bloque de 16x16 y en consecuencia 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 3.12 se puede apreciar el proceso anterior del desplazamiento de la ventana

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 una longitud de 36 elementos, el vector final tiene: $7 \times 15 \times 36 = 3780$ elementos.

Para la visualización únicamente se grafican los histogramas normalizados de 9x1 en las celdas de 8x8, imagen 3.13. Se puede percibir que las direcciones dominantes del histograma capturan la forma de la persona.

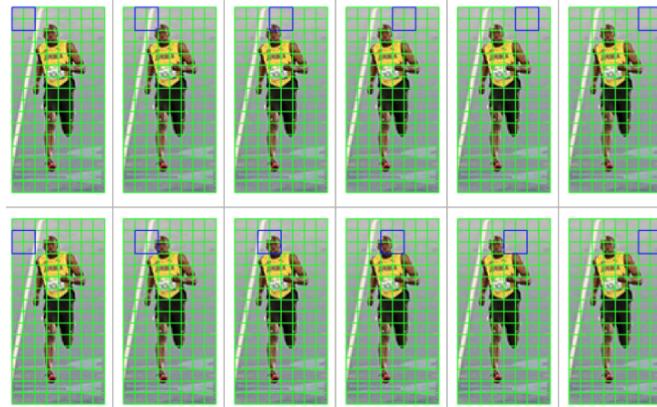


Figura 3.12: Normalización en llevada a cabo desplazando los bloques de 16x16

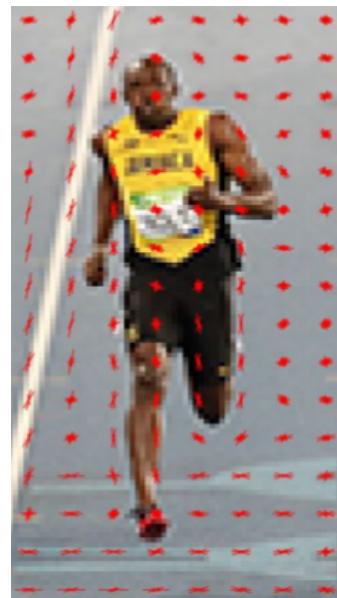


Figura 3.13:

Capítulo 4

Metodología

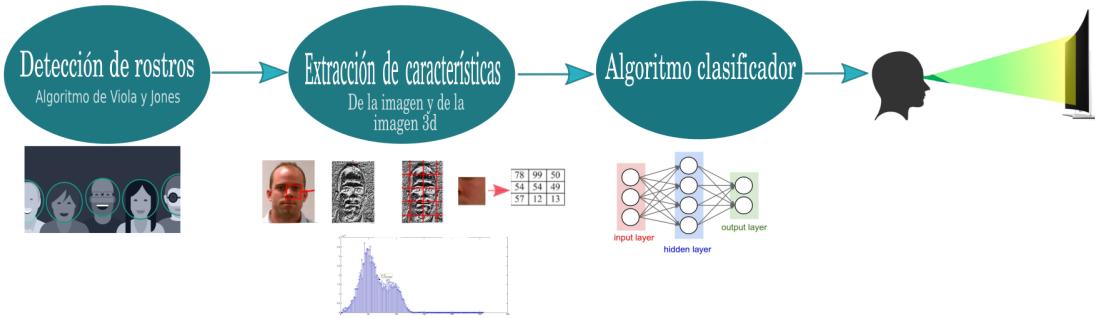
En el presente capítulo se detallarán las etapas y procesos necesarios del sistema para lograr la estimación de la mirada de las personas, esto incluye desde los métodos para el cálculo de los ángulos de mirada de las personas a un objeto en específico, el proceso de generación y selección de las marcas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

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 de lo que se observa, es decir, se estiman regiones del plano enfrente de las personas. La estructura del sistema ya calibrado y funcionando consta de 3 etapas:

- 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 3d del rostro que incluyen la posición en 3d, y las características de la imagen que ayudan al 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 es una red neuronal artificial de retropropagación entrenada con anterioridad.

Una de las etapas cruciales de este proyecto es la captura precisa de datos, entre esos datos está la ubicación del rostro en tres dimensiones en la escena, para lograr lo anterior se debe conocer el plano del piso en el que las personas habitan,



esto es, la ecuación matemática que describe dicho plano, el cual se obtiene a partir de la estimación de la matriz de rotación y traslación que tiene la cámara con respecto al piso. Además en este capítulo se detalla como implementar el algoritmo de Levenberg-Marquardt para optimizar la matriz de rotación y traslación. Otro apartado importante de la metodología incluye determinar en cuales puntos del piso ~~en los que vale la pena~~ que se coloquen las personas para capturar sus datos. Finalmente se describe como se realiza la implementación de la red neuronal teniendo con las imágenes de los rostros y los datos 3d como vector de características. A continuación en la metodología se describen las herramientas y técnicas necesarias para estimar la ecuación del plano.

4.2. Mapeo de coordenadas a ángulos

Uno de los problemas que hay que tomar en cuenta y definir en el presente trabajo antes de realizar los experimentos, es describir la relación que existe entre la ubicación en metros de lo que está mirando el sujeto de estudio en la pantalla y la pose de la cabeza de esta persona medida en ángulos, tomando en cuenta diversos aspectos 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 realiza el mapeo (o la ecuación) de dicho objeto desplegado en pantalla y la orientación de la cabeza, primero se describirá la geometría del escenario.

La pantalla que observa el sujeto del experimento se coloca en una pared a una distancia de L_y metros del piso, la pantalla tiene unas dimensiones de W m. de ancho y L m. de alto. 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 a Y** . Las personas se encontrarán paradas en un mismo plano (piso) y enfrente de la pantalla a una distancia D_z . Hay un punto en específico del rostro de las personas que es de interés 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. El vector \vec{v} sale del punto P y es ortogonal al plano de la cara, el



Figura 4.1:

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 .

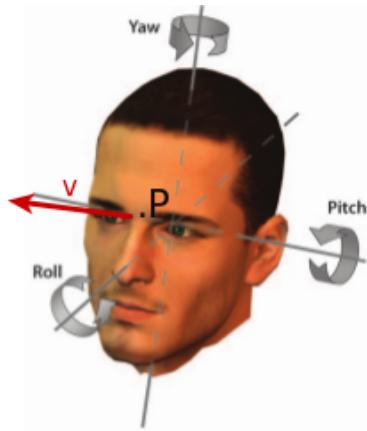


Figura 4.2:

Como se puede observar en la figura 1.2 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, ya que estos son los movimientos que se dan cuando una persona mueve su cabeza para observar el objeto en dos dimensiones de la pantalla.

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} , cuando la persona se encuentre mirando S el vector \vec{v} debe apuntar a S , 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, tomando

como marco de referencia 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 .

- ϕ_y .- Observando desde el plano $Y - Z$ representa que tanto se desvía el vector \vec{v}
- ϕ_x .- Observando desde 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.

4.3. Ecuaciones de la mirada

Una vez definido el escenario y los aspectos a tomar en cuenta, ahora falta demostrar cual es la relación que se puede hallar entre la posición del objeto en la pantalla y la pose de la cabeza mediante los ángulos ϕ_x y ϕ_y .

4.3.1. Desviación de ϕ_y y ϕ_x . Primer caso

Para facilitar el análisis en esta sección se ilustra el escenario con todos los elementos en la figura 2.1

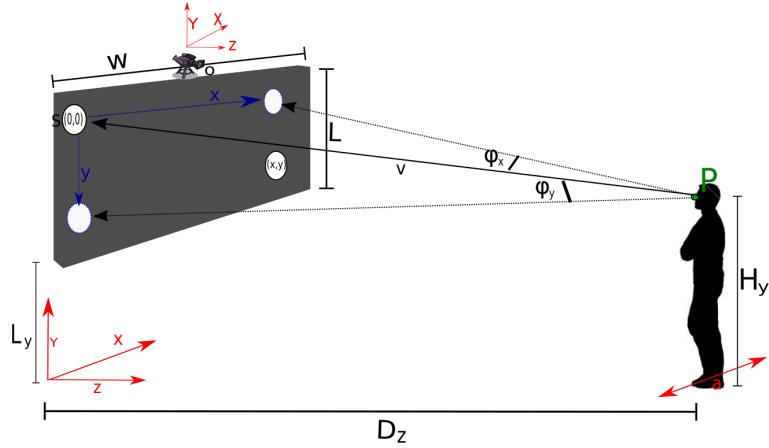


Figura 4.3: Escenario y aspectos a tomar en cuenta

Este caso es suponiendo que la cámara está colocada sobre la pantalla de tal forma que su eje Z es paralelo al piso, esto quiere decir que el eje Y de la cámara yace sobre el plano de la pantalla.

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, los cuales son: S y P . Supongamos que el objeto a observar está en la coordenada (x, y) de la pantalla, 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 m. con respecto al eje y , el movimiento de S yace sobre el plano $X - Y$ por lo que su componente en z es igual cero, entonces:

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

Para hallar las componentes de P se toma la distancia D_z a la que se encuentra del origen sobre el eje Z , como se había mencionado la persona se desplaza a m sobre el eje X y se encuentra a una distancia sobre el eje y de $L_y + L - H_y$:

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

Ahora que ya se definieron donde se encuentran los puntos que se utilizarán con respecto al marco de referencia centrado en la cámara, se necesitan los ángulos 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} y únicamente son necesarios los que se forman con el eje Y y el X . Las fórmulas de los cosenos directores son:

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

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

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

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

Por lo tanto los ángulos de desviación son:

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

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

4.3.2. Desviación de ϕ_y y ϕ_x . Segundo caso

El segundo caso para calcular ϕ_y y ϕ_x es un tanto similar al primero pero con la diferencia de que para obtener dichos ángulos no es necesario que la cámara (origen) se encuentre en una rotación específica (paralela al piso) como en el caso anterior. Ahora los ángulos se calculan de manera independiente a la orientación de la cámara, para lograr lo anterior 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 secciones siguientes 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 mencionó anteriormente 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). Sea $F = [F_x, F_y, F_z]$ un punto que yace en el piso donde se coloca la persona. 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 el la variable y de la ecuación:

$$y = \frac{d - Ax - Cz}{B} \quad (4.8)$$

De la ecuación anterior $F_x = x$, $F_y = y$ y $Fz = z$. Para encontrar P simplemente se parte de F y multiplicamos 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 una manera ortogonal, el resultado de esta operación es la ubicación de P en 3 dimensiones.

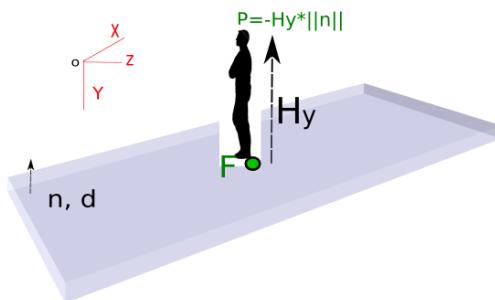


Figura 4.4:

Localización de S

Para conocer con precisión cual es la ubicación en pantalla del objeto con respecto a la cámara como marco de referencia con la cámara en cualquier orientación se hizo uso de una unidad Pan-Tilt. Una unidad Pan-Tilt es un dispositivo que permite colocar cámaras en posiciones muy precisas, imagen 4.5.

Sea O_c el eje de rotación de la cámara (punto focal), O_u el eje de rotación de la

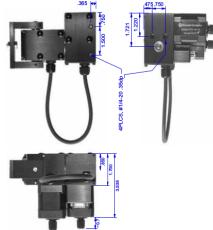


Figura 4.5:

unidad Pan-Tilt, D_{cu} la distancia en el eje y entre O_c y O_u , θ el ángulo que rota la unidad Pan-Tilt alrededor del eje X , $S_1 = (S_{1x}, S_{1y}, S_{1z})$ la ubicación inicial de la figura en pantalla antes de la rotación de la unidad y S_2 la ubicación después de la rotación, figura 4.7 y 4.6. Conociendo S_1 con respecto al marco de referencia de la cámara la localización de S_2 se realiza tomando en cuenta los parámetros anteriores mediante los siguientes pasos:

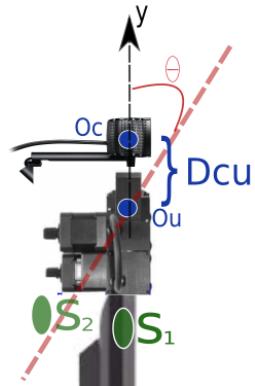


Figura 4.6:

- Colocando la unidad Pan-Tilt con la cámara de manera que el objetivo de la cámara quede de forma perpendicular a la al plano de la pantalla en la cual se proyectarán las figuras que los sujetos de experimentación observarán, lo anterior observando desde el plano $Y - Z$ del marco de referencia.
- Se traslada el punto S_1 hacia el eje de rotación de la cámara O_c mediante D_{cu} metros, únicamente desplazándolo en el eje Y .



Figura 4.7:

- Se rota θ grados el punto a través de la matriz de rotación del eje X .
- Finalmente se regresa el punto D_{cu} metros.

Lo anterior se realiza con la siguiente ecuación:

$$S_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(-\theta) & -\sin(-\theta) \\ 0 & \sin(-\theta) & \cos(-\theta) \end{bmatrix} * \begin{bmatrix} S_{1x} \\ S_{1y} - D_{cu} \\ S_{1z} \end{bmatrix} + \begin{bmatrix} S_{1x} \\ S_{1y} + D_{cu} \\ S_{1z} \end{bmatrix} \quad (4.9)$$

4.4. Generación de marcas más significativas en el piso y la pantalla

Como se puede apreciar en el análisis anterior pueden haber infinitas combinaciones de posiciones de marcas en el piso donde se colocan las personas y figuras en la pantalla, inclusive si las regiones en pantalla y piso donde se realiza la captura de datos 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, es decir, ϕ_x y ϕ_y , esto se hace con el objetivo de que haya variedad en el conjunto de datos de entrenamiento.

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.

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 con el propósito de 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 . En base al análisis de los ángulos ϕ_x y ϕ_y se determina el tamaño del paso en metros en el eje X y Z del piso, y cuales son sus límites máximos y mínimos en ambos ejes.

Tomando en cuenta lo anterior 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: F , además se calcula para cada posición cual es la ubicación de la cabeza de la persona en la escena: P , como se explicó 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 de la cámara únicamente se proyecta en la imagen la ubicación de P mediante la matriz de calibración de la cámara, con el resultado obtenido se verifica que esté dentro de los límites visibles de la imagen, si es así, P se encuentra dentro del campo visual de la cámara

Sea cada punto que representa el rostro $P_1, P_2, P_3\dots$ Sea P_1 un punto en el arreglo bidimensional que representa el rostro, se determina cual es el ángulo ϕ_{x1} y ϕ_{y1} para una figura S_1 en la pantalla en una posición fija, a continuación se calcula cual es la rotación de la mirada con respecto a otro punto en la malla: P_2 cuyos ángulos son ϕ_{x2} y ϕ_{y2} :

$$rot_x = \phi_{x1} - \phi_{x2} \quad (4.10)$$

$$rot_y = \phi_{y1} - \phi_{y2} \quad (4.11)$$

Lo anterior se calcula para las demás P de la matriz siempre con respecto a P_1 , al terminar de calcular todas las rotaciones de P_1 se pasa a calcular todas las rotaciones de mirada de P_2 con todos 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, esto es 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 encuentre dentro del campo visual de la cámara se generan las instancias de posibles puntos para capturar datos, cada instancia con el siguiente formato:

$$F1x, F1y, F1z, F2x, F2y, F2z, Sx, Sy, Sz, rot_x, rot_y \quad (4.12)$$

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. Generación de secuencia óptima mediante algoritmo de ordenamiento

El algoritmo comienza con la obtención de todas las instancias de rotaciones posibles, es decir la salida de la etapa anterior.

4.5. Rotaciones en los pares de posiciones

Se eligió como forma de representación de la pose de la cabeza los cuaterniones, los cuales son obtenidos mediante el IMU, como se había mencionado el dispositivo se colocará en la cabeza de las personas durante el entrenamiento para la generación del "ground truth", sin embargo, aún falta considerar los siguientes detalles:

- ¿Cuáles posiciones en el piso con respecto al marco de referencia centrado en la cámara se utilizaron para capturar la información(imagen rostro y pose en cuaterniones)?.
- ¿A qué distancia estarán las posiciones entre si y hasta que distancia máxima se tomarán datos, es decir, cual es el mínimo en X y Z que se debe seleccionar? ya que como se observó en la sección de *Análisis del comportamiento de los ángulos ϕ_x y ϕ_y* dependiendo de los parámetros del experimento hay valores en los cuales ya no vale la pena hacer análisis ya que las variaciones en los ángulos es muy pequeño o inexistente.
- ¿Cuáles posiciones para la figura en pantalla se deben utilizar?.
- ¿Cuántas posiciones de figura en pantalla y personas en el piso se considerarán?
- ¿Cómo lidiar con las pequeñas variaciones en las mediciones del IMU que se den en diferentes personas en los mismas instancias de experimentación?

4.5.1. Cálculo de todas las rotaciones posibles

4.6. Proyección de marcas en el piso

Como se mencionó anteriormente para la etapa de entrenamiento del algoritmo de aprendizaje automático se debe tener control sobre las capturas que se van a

tomar de cada sujeto de experimento incluyendo las posiciones en la escena donde se colocarán al capturar los datos. Una vez calculada las mejores posiciones en la escena para capturar los datos con las que se va a entrenar el algoritmo, la siguiente etapa consiste en colocar estos puntos con marcas en el piso para saber durante los experimentos en qué posiciones se deben colocar las personas. Como se había mencionado las marcas son con respecto al marco de referencia centrado en la cámara, la cual se encuentra sobre la pantalla, dichas marcas se pueden pintar en el piso 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:

- Hay bastante inexactitud, por ejemplo en caso de que no se mida sobre la misma dirección de los ejes debido a pequeños errores humanos y esto tiende a agravarse cuando las marcas son lejanas.
- El sistema de coordenadas centrado en la cámara no es paralela al plano del piso donde se colocarán las marcas ya que se debe rotar en dirección al piso para así tener una mejor vista de los rostros de las personas que se encuentran más cercanas a la cámara. Por lo tanto si rotamos la cámara de igual manera rotamos el marco de referencia y es más complicado colocar las marcas, lo anterior se puede ver ilustrado en la figura 11.1.

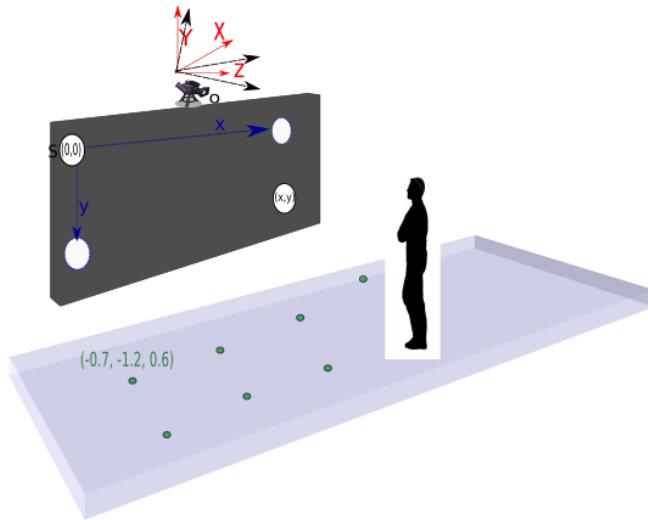


Figura 4.8:

Como solución al problema anterior se desarrollaron algoritmos utilizando visión computacional, geometría y optimización numérica para colocar en el piso de manera más precisa las marcas, el método consiste en conocer cual es la distancia y orientación del plano del piso donde se situarán las personas y proyectar las marcas en éste tomando como marco de referencia la cámara.

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

Para utilizar el método que se describirá a continuación es necesario conocer la matriz de parámetros intrínsecos K . La obtención de esta matriz se logró mediante el software de Matlab: Camera Calibration Toolbox, dando como resultado la siguiente matriz.

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 616,164 & 0 & 325,528 \\ 0 & 616,82 & 228,66 \\ 0 & 0 & 1 \end{bmatrix}$$

Homografía

Durante la primera etapa se debe hallar la homografía que mapea puntos en el plano del piso y los respectivos puntos en la imagen de ese plano, todas coordenadas deben estar en metros, los puntos en el plano del piso deben tener las medidas reales que hay entre ellos, tener uno de los puntos como origen y con respecto a éste establecer las coordenadas de las demás.

La matriz de homografía se descompone y a partir de ella se encuentra la matriz de rotación y el vector de translación 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 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 $scnPts_{4x4}$ y la matriz obtenida con los puntos capturados con el mouse se denomina $imgPts_{3x4}$. 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 establecemos la siguiente matriz:

$$scnPts = \begin{bmatrix} 0 & 0 & 0,4 \\ 0 & 0,4 & 0,4 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

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 $scnPts$ está en metros por lo tanto se requiere que $imgPts$ igual esté en metros, esto se consigue simplemente multiplicando la matriz $imgPts$ por la inversa de la matriz K .

Como resultado del proceso anterior se obtuvo una homografía inexacta ya que al comprobar si mapeaba los puntos de la escena a la imagen ésta no lo hacía correctamente, los puntos tenían 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
- La captura de los puntos en la imagen mediante el mouse es realizada por simple inspección y seleccionando los puntos lo cual podría generar capturas erróneas.
- Son pocos puntos para calcular la homografía

Para solucionar los inconvenientes anteriores se optó por utilizar más puntos, para ello se colocó 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 estarán separados a 0,12m, además con el objetivo de que sea más precisa la captura y automática se modificó el programa para localizar las intersecciones automáticamente mediante bibliotecas de openCV y no estar capturando manualmente los puntos. Ahora la matriz $scnPts$ tiene una dimensión de $4x48$ y $imgPts$ $3x48$. En la imagen 11.2 se pueden notar el buen desempeño que tiene algoritmo al encontrar las intersecciones en la manta con el tablero de ajedrez.



Figura 4.9:

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, además, se puede calcular la ecuación del plano la cual será útil más adelante.

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

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 calcular 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 redundancias. La normalización se realiza de la siguiente manera:

$$norm1 = \|[h_{11}h_{21}h_{31}]^T\| \quad (4.13)$$

$$norm2 = \|[h_{12}h_{22}h_{32}]^T\| \quad (4.14)$$

$$normT = \frac{norm1 + norm2}{2} \quad (4.15)$$

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

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

$$R_3 = R_1 \otimes R_2 \quad (4.16)$$

Para hallar la ecuación del plano con respecto a la cámara (marco de referencia): $Ax + By + Cz = d$ únicamente es necesario la R , 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 al plano Π está dado por los componentes: A , B y C de la ecuación del plano y la distancia más cercana del plano al origen por: d , figura

Tomando como marco de referencia el plano (puntos establecidos en $scnPts$) 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 rotamos este vector con la matriz de rotación:

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

Sea $scnPts1_{3x1}$ 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

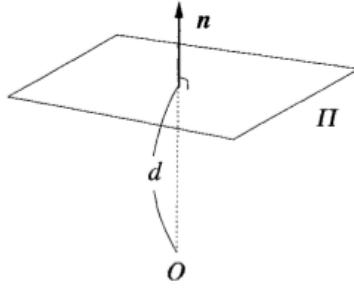


Figura 4.10: Representación de un plano en el espacio

la cámara, y al resultado se le aplica el producto punto con la normal al plano:

$$scnPtsRot = R * scnPts1 + T \quad (4.17)$$

$$d = \langle N, scnPtsRot \rangle \quad (4.18)$$

Reproyección de puntos en la imagen

Todo el proceso anterior se realiza con el objetivo de poder proyectar puntos en la imagen sobre el plano del piso tomando como marco de referencia la cámara. Lo anterior se consigue únicamente estableciendo la matriz con las marcas en el piso que se necesitan y proyectándolas en la imagen multiplicando por la matriz de parámetros intrínsecos para pasar las coordenadas a pixeles.

Antes de pasar a esa etapa es necesario verificar que los parámetros encontrados en la sección anterior: R , T , n y d son correctos.

- La verificación de n y d consiste en utilizar los puntos obtenidos de $imgPts$ y proyectarlos desde el origen hacia el plano (formando rectas) y encontrar las coordenadas en las cuales se intersectan dichas proyecciones con el plano, estas intersecciones deben ser las mismas que las coordenadas de $scnPts$. Sea L la recta en el espacio, rh el vector que va del punto más cercano de L al origen y m el vector unitario que indica la orientación de L .

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

$$r = r_H + \frac{d - \langle n_\Pi, r_H \rangle}{\langle n_\Pi, m \rangle} m \quad (4.19)$$

En este caso las líneas (proyecciones de $imgPts$) parten de la cámara por lo tanto el vector rh es igual cero, simplificando la ecuación la intersección queda de la siguiente manera:

$$r = \frac{d}{\langle n_\Pi, m \rangle} m \quad (4.20)$$

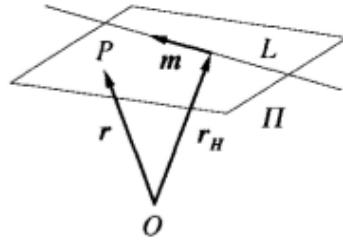


Figura 4.11: intersección de una recta con un plano

- La verificación de R y T simplemente se realiza tomando los puntos de $scnPts$ rotándolos, trasladándolos y proyectándolos en la imagen, en caso de que los cálculos de la R y T sean correctos estas proyecciones deben ser las mismas que $imgPts$.

4.6.2. Optimización Levenberg-Marquardt

Como se mencionará más adelante en la etapa experimental la rotación y translación no son lo suficientemente precisas para empezar a realizar experimentos. Por lo tanto se opta por utilizar un algoritmo de optimización numérica para minimizar el error que existe en la rotación y translación. Se eligió el algoritmo de Levenberg-Marquardt debido a que es uno de los más eficientes, con él se busca el vector de parámetros p que minimice.

$$F(p) = \frac{1}{2} \sum_1^m (f_i(p))^2 \quad (4.21)$$

donde $f : \mathbb{R}^n \rightarrow \mathbb{R}$, $i = 1, \dots, m$ son las funciones dadas y $m \geq n$. Las funciones a optimizar deben tener los parámetros de R y T , estar descritas de tal forma que sean iguales o cercanas a cero, e incluir a los 48 puntos, por lo tanto el número de ecuaciones a minimizar es $m = 144$ (tomando en cuenta los tres componentes de cada punto) y se plantean de la siguiente forma:

$$f_{xyz}(p) = ||X_{rt} - X_{intersec}|| \quad (4.22)$$

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

$$f_{xyz}(p) = \{R * X_{scn} + T\} - \{\frac{d}{\langle n_\Pi, X_{img} \rangle} X_{img}\} \quad (4.23)$$

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. La representación más simple la cual está basada en el teorema de Euler dice que cada 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 que se optimizará el eje de rotación y su magnitud del ángulo de rotación, ya que como indica la fórmula de Rodrigues cualquier matriz de rotación se puede realizar mediante la rotación alrededor de un eje fijo ω por un cierto ángulo $\|\omega\|$:

$$R = I + \frac{\hat{\omega}}{\|\omega\|} \sin(\|\omega\|) + \frac{\hat{\omega}^2}{\|\omega\|^2} (1 - \cos(\|\omega\|)) \quad (4.24)$$

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:

$$\|\omega\| = \cos^{-1} \left(\frac{\text{traza}(R) - 1}{2} \right), \frac{\omega}{\|\omega\|} = \frac{1}{2\sin(\|\omega\|)} \begin{bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{bmatrix} \quad (4.25)$$

Los 7 parámetros optimizar son los siguientes:

$$p = [\omega_x, \omega_y, \omega_z, \|\omega\|, T_x, T_y, T_z] \quad (4.26)$$

En el algoritmo de levenberg-marquardt se debe definir cuales son cada una de las funciones y no es suficiente con 48 ecuaciones como la de 11.10 ya que ésta involucra multiplicaciones de matrices de 3 filas (son conjuntos de puntos de 3 coordenadas) teniendo en realidad 144 funciones a optimizar. Por lo tanto es necesario expresar la ecuación 11.10 en términos de $\omega_x, \omega_y, \omega_z, \|\omega\|, T_x, T_y, T_z$ con ayuda de la fórmula de Rodrigues para poder utilizar el algoritmo de optimización. Por lo que para cada una de las x de las 48 de la ecuación 11.10 de X_{rt} (la parte de $X_{intersec}$ está más claro como usarla debido a la ecuación 11.11) la función de optimización es:

$$\begin{aligned} f_{xi}(p) = & [(\omega_x^2 - 1)(1 - \cos(\|\omega\|)) + 1] X_{scnxi} + \\ & [\omega_x \omega_y (1 - \cos(\|\omega\|)) - \omega_z (\sin(\|\omega\|))] X_{scny} + \\ & [\omega_y \sin(\|\omega\|) + \omega_x \omega_z (1 - \cos(\|\omega\|))] X_{scnzi} \} + T_x \end{aligned} \quad (4.27)$$

Para cada una de los puntos en y de los 48 la función de optimización es:

$$f_{yi}(p) = [\omega_z \sin(\|\omega\|) + \omega_x \omega_y (1 - \cos(\|\omega\|))] X_{scnxi} + \\ [(\omega_y^2 - 1)(1 - \cos(\|\omega\|)) + 1] X_{scnyi} - \\ [\omega_x \sin(\|\omega\|) + \omega_y \omega_z (1 - \cos(\|\omega\|))] X_{scnzi} + T_y \quad (4.28)$$

Finalmente en los puntos pertenecientes a la coordenada z la función es:

$$f_{zi}(p) = [\omega_x \omega_z (1 - \cos(\|\omega\|)) - \omega_y \sin(\omega)] X_{scnxi} + \\ [\omega_x \sin(\|\omega\|) + \omega_y \omega_z (1 - \cos(\|\omega\|))] X_{scnyi} + \\ [(\omega_z^2 - 1)(1 - \cos(\|\omega\|)) + 1] X_{scnzi} + T_z \quad (4.29)$$

4.7. Red neuronal

El algoritmo clasificador propuesto para esta etapa de la tesis es una red neuronal de propagación hacia atrás, donde las neuronas de entrada es el vector de características y las neuronas de salida representan región en la pantalla que se observa, a continuación se describe como se implementa la red neuronal en el presente proyecto comenzando con el tratamiento de datos incluidas las imágenes para representarlos como el vector de características.

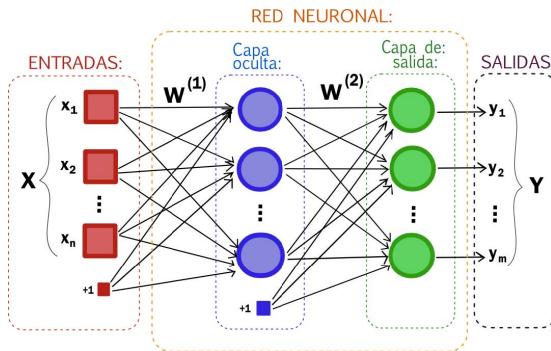


Figura 4.12: Red neuronal artificial

4.7.1. 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 procesan cada una de las imágenes capturadas con el algoritmo de Viola y Jones para la detección de rostros
2. De las imágenes con rostros detectados se extrae la región en la imagen donde se encuentra el rostro mediante un ROI.

3. El ROI de la cara de la persona se guarda para posterior análisis junto con los datos de ubicación 3d en las escena del rostro 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 por cada rostro un conjunto de 8 datos los cuales son: *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 una indicador de la distancia sobre el eje Z de la persona, más adelante en la etapa experimental se entrará más en detalle.



Figura 4.13: Características

La segunda etapa consiste simplemente en acomodar los datos resultantes del 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 $dimE$ para todos, 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 restantes elementos son la dimensión original del rostro y P .

4.7.2. Neuronas de salida

En este proyecto cada neurona de salida representa una partición de la pantalla que la persona está observando, esto quiere decir que si por ejemplo si solamente se tienen dos neuronas de salida la pantalla será particionada en dos regiones, durante la etapa experimental se experimentará con diferentes topologías de la red y número de neuronas de salida. Con respecto a lo anterior es necesario mapear la figura de lo que observa la persona S (etiqueta) a una neurona de salida, ya que se conocen las dimensiones de la pantalla donde se desplaza S , en consecuencia únicamente se determina en qué región de la pantalla se encuentra

S para cada ejemplo de entrenamiento y se le coloca un uno a esa neurona, en caso contrario se coloca un cero, figura 4.14.

Una vez entrenada la red y cuando se está haciendo la predicción con los datos de

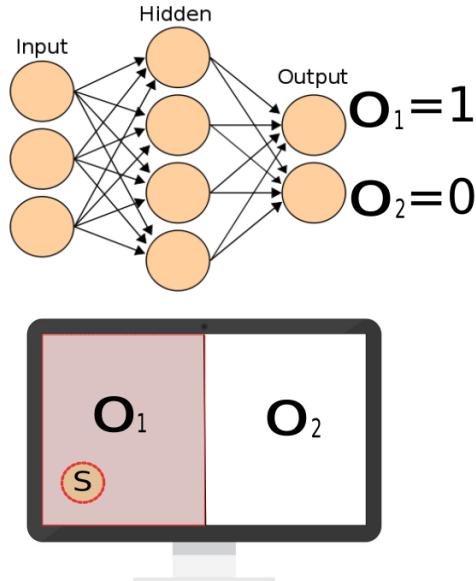


Figura 4.14: Neuronas de salida

prueba, los resultados en el vector de salida no son enteros como las etiquetas en el entrenamiento (1 y 0), la red arroja valores decimales, por lo tanto para tomar una predicción como acertada se elige la neurona con el valor más 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 resultante por la red neuronal.

Capítulo 5

Experimentos

En este capítulo se describen los experimentos....

5.1. Materiales

5.1.1. Dispositivo para adquirir datos

Durante la etapa de adquisición de datos para el algoritmo de entrenamiento, además de conocer la ubicación de la persona y la de la figura que observan en pantalla es necesario capturar una imagen del rostro de la persona y conocer (mediante un aparato de medición) la orientación de la cabeza, lo anterior es necesario para evaluar y comparar el sistema de estimación de pose.

Como señalan en [Murphy-Chutorian, 2009] el método que se utilice para obtener las medidas de la orientación de la cabeza en el conjunto de entrenamiento debe ser preciso y entre los métodos más eficientes proponen los sistemas captura ópticos de movimiento y los sensores iniciales, sin embargo, los primeros son muy costosos, por lo que en el desarrollo del presente trabajo de tesis se optó por utilizar un sensor inercial, el cual consiste de acelerómetros y giroscopios a menudo acoplados con algún tipo de filtro para reducir ruido.

En este apartado se describirá el proceso de diseño y fabricación del dispositivo que realiza la medición de la orientación.

De modo que la tarjeta electrónica debe ir en la cabeza de las personas, durante el diseño se tomó mucho en cuenta el tamaño y el peso que tendría, ya que de ser muy grande y pesado podría ser muy incomodo para las personas durante los experimentos.

IMU

El imu seleccionado para este proyecto es el BNO055 de Bosch (figura 5.1). El sensor utiliza algoritmos que mezclan los datos del acelerómetro, magnetómetro

y giroscopio en una salida estable de la orientación de los tres ejes, el sensor es capaz de arrojar los datos en cuaterniones, vectores y ángulos de Euler. El BNO055 se alimenta con voltaje de 3.3 a 5v, mide 2.67x2.032cm y para obtener los datos utiliza comunicación I2C.

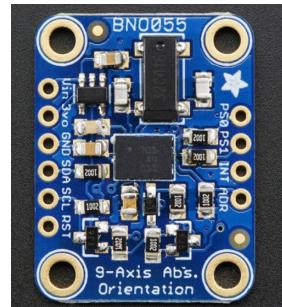


Figura 5.1:

Microcontrolador

Debido a las restricciones de peso y tamaño se utilizó un arduino micro, el cual es la versión más pequeña del arduino con 4.8x1.8cm y esto se debe a que es una versión más limitada en cuanto a periféricos y módulos, sin embargo, es adecuada para el uso que se le da en el proyecto, únicamente se utilizará para obtener los datos del imu mediante comunicación I2C y enviarlos a la computadora por comunicación serial. El arduino se alimenta mínimo con 7v para que su regulador interno regule a 5v el microcontrolador que usa.



Figura 5.2:

Xbee

Durante los experimentos las personas se estarán moviendo en diferentes posiciones de la escena y mirando diferentes lugares en la pantalla, como se había mencionado la tarjeta electronica se colocará en su cabeza, por lo que utilizar un cable (de comunicación serial) entre la computadora y la tarjeta para obtener los datos de la pose sería muy inadecuado ya que éste debe ser bastante largo, le pesaría a la tarjeta y afectaría su posición en la cabeza, y finalmente podría

afectar la experiencia de la persona el tener un cable muy extenso cerca de ella. Tomando en cuenta las consideraciones anterior se decidió trabajar con módulos xbee.

Los xbee son módulos inalámbricos creados para la comunicación inalámbrica entre ellos, su finalidad es la eliminación de cables en la comunicación serial. Para este proyecto se utilizaron los xbee s1 (figura 5.3) los cuales son los más pequeños, de bajo consumo y simples de utilizar debido a que su configuración punto a punto es bastante sencilla y únicamente se requiere declarar un xbee como emisor y el otro como receptor. Los xbee s1 se alimentan con 3.3v.

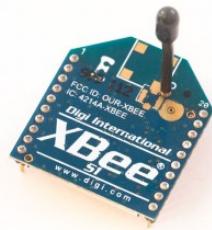


Figura 5.3:

Diseño de la tarjeta

Para alimentar todo el circuito es necesario al menos 7v, ya que como se había mencionado eso necesita el Arduino micro y es el dispositivo que requiere mayor voltaje, el imu funciona con 5v por lo que se le puede conectar el pin de 5v que tiene el Arduino, sin embargo aún se tiene el inconveniente de que el xbee y su comunicación funcionan con 3.3v, y las baterías recargables de iones de litio como la que se pretende utilizar por su reducido tamaño y peso, arrojan solo 3.7v.

Para solucionar el problema de los 7v se utilizó la bomba de carga TPS61093, figura 5.4, en una configuración para elevar el voltaje a 7v. La comunicación entre el arduino y el xbee (diferentes niveles de voltaje) se logró mediante el cambiador de nivel TXS0102, y finalmente para obtener 3.3v de la batería se utilizó el regulador TPS73633. Se pudo alimentar el xbee a través del pin de 3.3v del arduino, sin embargo, el xbee consume hasta 60mA lo cual es demasiado para el regulador de 3.3v del Arduino y podría afectar su funcionamiento.

En la imagen 5.5 se puede ver el esquemático de la tarjeta y en la 5.6 el pcb. Adicionalmente se creó una pequeña tarjeta para la recepción de datos a través del otro xbee y envío a la computadora mediante un cable con convertidor de rs232 a usb. Más adelante se detallará el programa para la recepción de datos.



Figura 5.4:

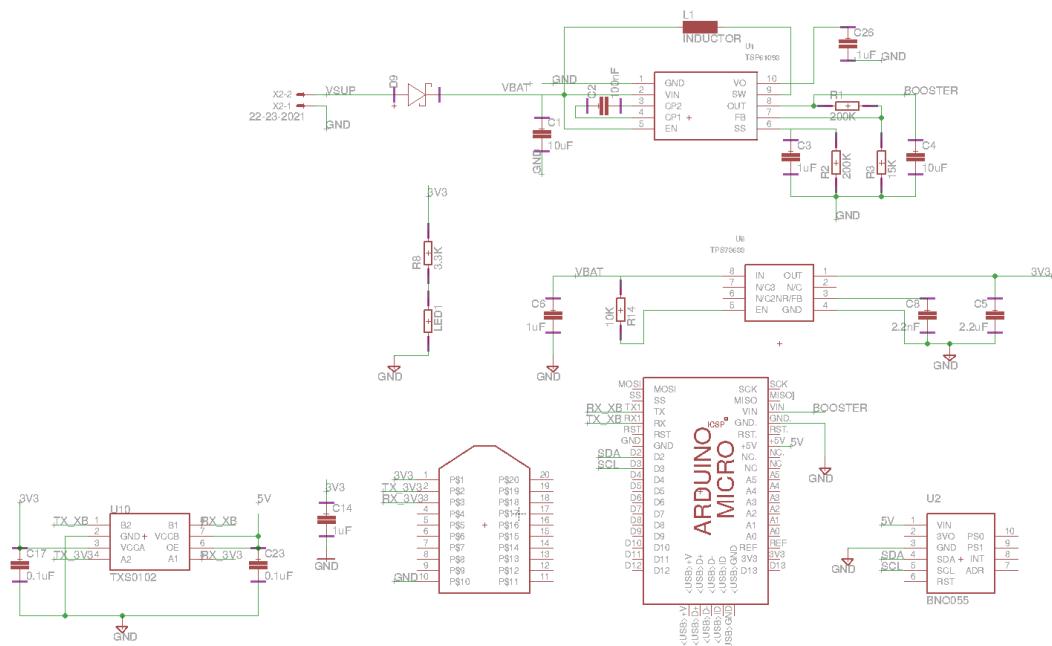


Figura 5.5:

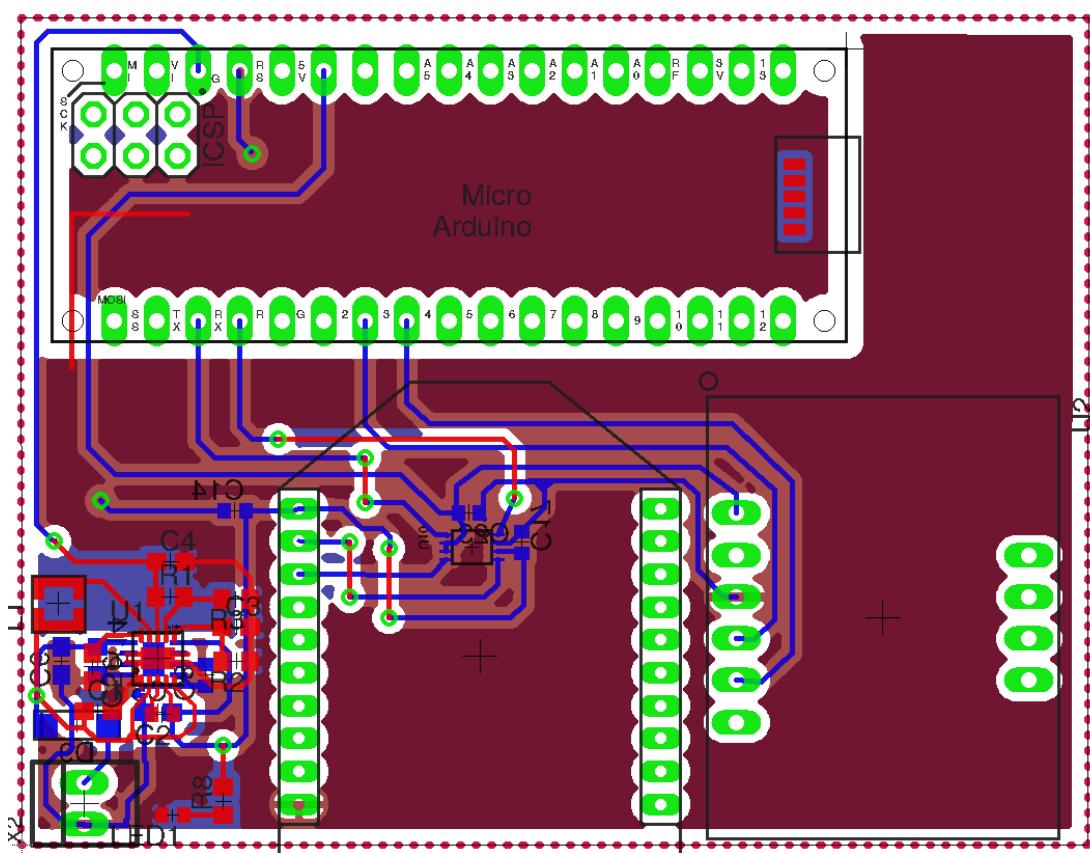


Figura 5.6:

tarjeta electrónica

En las siguientes imágenes se presentan: la tarjeta electrónica con el imu, arduino y xbee ya fabricada; la batería recargable de 3.7v y la tarjeta que conecta el xbee receptor de datos con la computadora para poder enviar los datos del imu hacia la computadora.

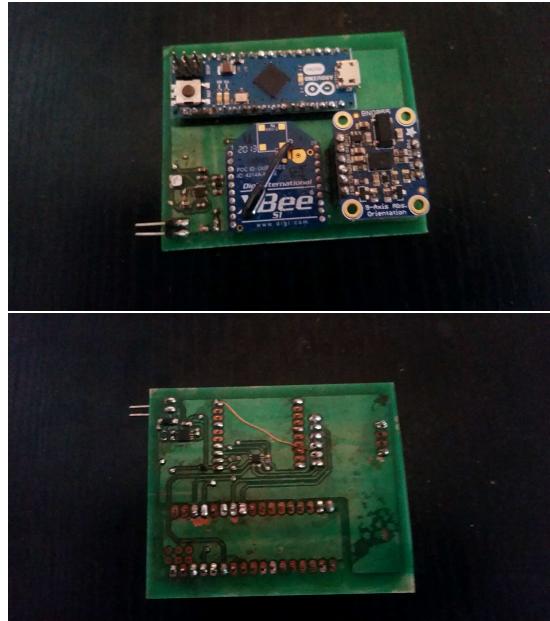


Figura 5.7: tarjeta con el IMU



Figura 5.8: batería recargable

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 de frente a la pantalla, con el rostro en diferentes orientaciones debido a

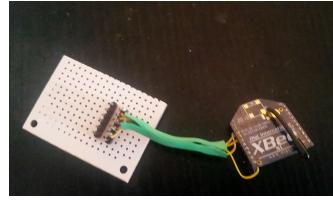


Figura 5.9: tarjeta receptora de datos

que está mirando el objeto en pantalla en diferentes posiciones, para lograr esto se requieren dos cosas durante la etapa de captura de datos:

- 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

Sea una instancia de captura de datos la captura de la imagen de una persona anotando su ubicación en el lugar del experimento, la posición del objeto y la orientación de la cabeza (ϕ_x y ϕ_y).

Se llega a dar una extensa cantidad de instancias de captura por cada persona variando los parámetros mencionados, lo que llevaría a bastante tiempo de captura de instancias por cada persona y sería muy inconveniente para ellas, por lo tanto se realizó un análisis antes de realizar los experimentos del comportamiento de los ángulos en diferentes circunstancias, el análisis fue realizado mediante el software matemático Octave y a continuación se discuten los resultados. Suponiendo que es una pantalla de 42 pulgadas y el marco de referencia se encuentra en la cámara encima de la pantalla.

- Ancho de la pantalla: $W = 0,9282m$
- Largo de la pantalla: $L = 0,5523m$
- Distancia de la pantalla al piso: $Ly = 1,5m$
- Distancia de la cámara a la pantalla: $Cy = 0,05m$
- Estatura del sujeto del experimento: $Hy = 1,6m$
- Movimiento de la persona a través del eje Z : de $0,1m$ a $3m$ con pasos de $0,51786m$
- Movimiento de la persona a través del eje X : de $-3m$ a $3m$ con pasos de $0,109m$
- Movimiento de la figura en pantalla a través del eje X : de $-\frac{W}{2}m$ a $\frac{W}{2}m$ con pasos de $0,01658m$

- Movimiento de la figura en pantalla a través del eje Y: de $-Cy$ a $-(Cy+L)m$ con pasos de $0,051786m$

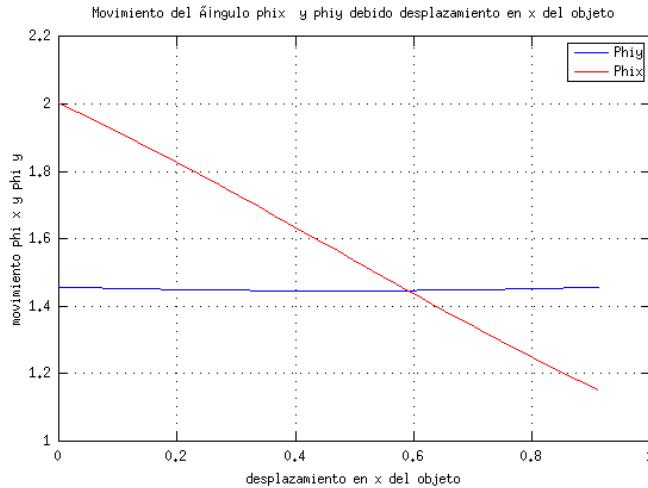


Figura 5.10:

En la gráfica de la figura 3.1 se puede observar la variación de los ángulos ϕ_x y ϕ_y con la persona en una única posición: $P = [0, -(L_y + L - H_y), 1]^T$, la variación de la figura en pantalla (en el eje x y y) de lo que observan es graficada con el marco de referencia centrado en la esquina superior izquierda de la pantalla, sin embargo los cálculos se hacen tomando la posición de la cámara como marco de referencia. El mayor cambio que ocurre en los ángulos es con respecto a ϕ_x de 2 a 1.1 radianes que son 0.9 radianes o 51.5662 grados, lo cual parece tener mucho sentido ya que el desplazamiento se hace en el eje de ϕ_x .

Lo que resulta peculiar es que a pesar de que no hay desplazamiento en el eje y de la figura si hay una ligera variación en el ángulo ϕ_y de la mirada de la persona, esto se debe al 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 figura 3.2, 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 la

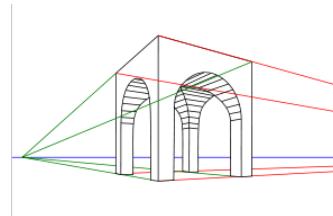


Figura 5.11:

figura inicialmente se encuentra en una posición superior a la de los ojos y se mueve hacia uno de los extremos de la pantalla, ésta pareciera moverse hacia abajo (hacia el horizonte) y cuando inicialmente la figura se encuentra en una posición inferior a la de los ojos de la persona y se mueve hacia un extremo, ésta pareciera moverse hacia arriba por lo tanto la mirada de las personas en el eje y tiende a moverse.

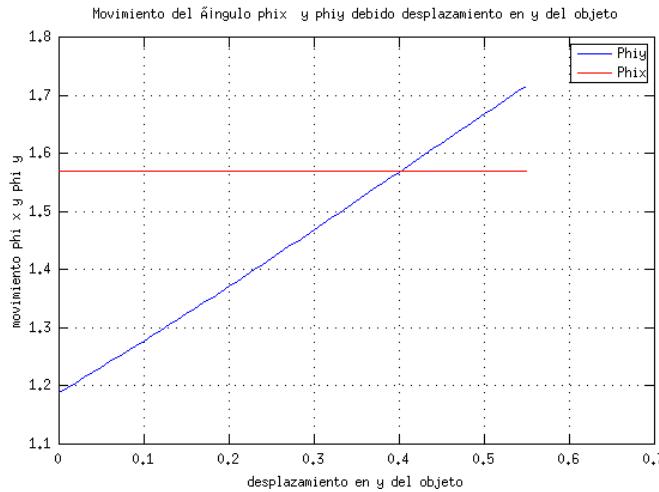


Figura 5.12:

En la gráfica de la figura 3.3 se puede apreciar que el objeto que observan las personas en pantalla 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 de $0,5523m$ la mirada varía de $1,2$ a $1,7$ radianes, es decir, $0,5$ radianes o lo que es lo mismo $28,6479$ grados.

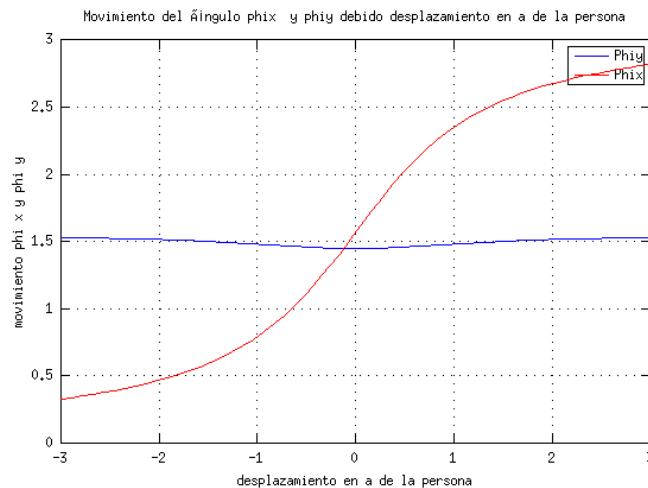


Figura 5.13:

La gráfica 3.4 describe el caso en el únicamente la persona se mueve sobre el

eje X de -3 a $3m$, 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 $1m$ la cual la mirada varía de $0,8$ a $2,36$ radianes, si la persona se mueve más allá de esta distancia la mirada tiende a estabilizarse y no valdría la pena hacer experimentos en estas zonas. Lo anterior se ve con mayor claridad en la figura 3.5, aquí se amplio el rango en el que se mueve la persona sobre el eje X y en consecuencia ϕ_x tiende a estabilizarse.

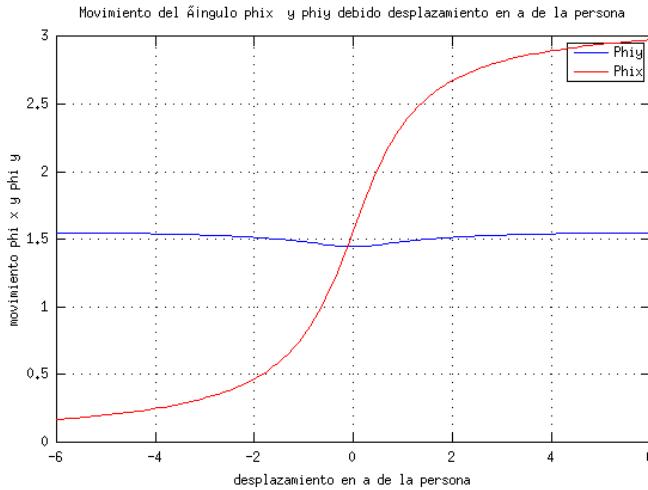


Figura 5.14:

El último análisis corresponde al caso cuando la figura en pantalla no se mueve y la persona únicamente se mueve sobre el eje Z . La figura que observa se encuentra en el centro de la pantalla, es decir en: $[S_x, S_y, S_z]^T = [\frac{W}{2}, -(c_y + \frac{L}{2}), 0]^T$ 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 3.6 se puede apreciar que mientras se va alejando la persona el único ángulo de la mirada que varía es ϕ_y , 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 muy pequeña, 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

Se realizó una simulación mediante el software processing del análisis anterior, esto se hizo para poder observar el comportamiento de los ángulos en diferentes posiciones de las personas al mismo tiempo y variar en tiempo real y de manera

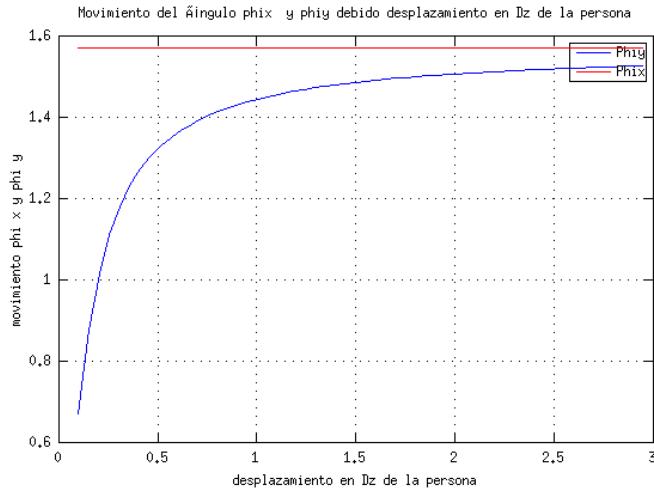


Figura 5.15:

interactiva la posición en el eje X y Y de la figura en pantalla. Los parámetros del escenario son los mismos a los utilizados en la análisis anterior: W , L , LY , Cy y Hy .

En la figura 4.1 se puede apreciar una figura de la simulación, en ella se despliegan 36 posiciones de personas separadas por 1m en el eje X y 0.5m en el eje Y . La variación 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 Y 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 ángulos ϕ_x y ϕ_y además de ser mostrados en cada una de las posiciones son representados mediante el radio del círculo que representa cada persona y la línea que sale de dicho ángulo, ϕ_x es la línea y ϕ_y es el radio.

Mediante la simulación de igual manera se demostró como al arrastrar la figura únicamente en X no solo varía ϕ_x sino también ϕ_y de todas las instancias, esto se puede corroborar en los valores de los ángulos y en el área del círculo.

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

5.3.1. Reproyección de punto en la imagen

Retomando la sección 4.6.1 se realizaron experimentos para verificar que la estimación de la rotación y traslación fueran precisos a partir simplemente de la descomposición de la homografía. El resultado de comprobación se puede ver en la

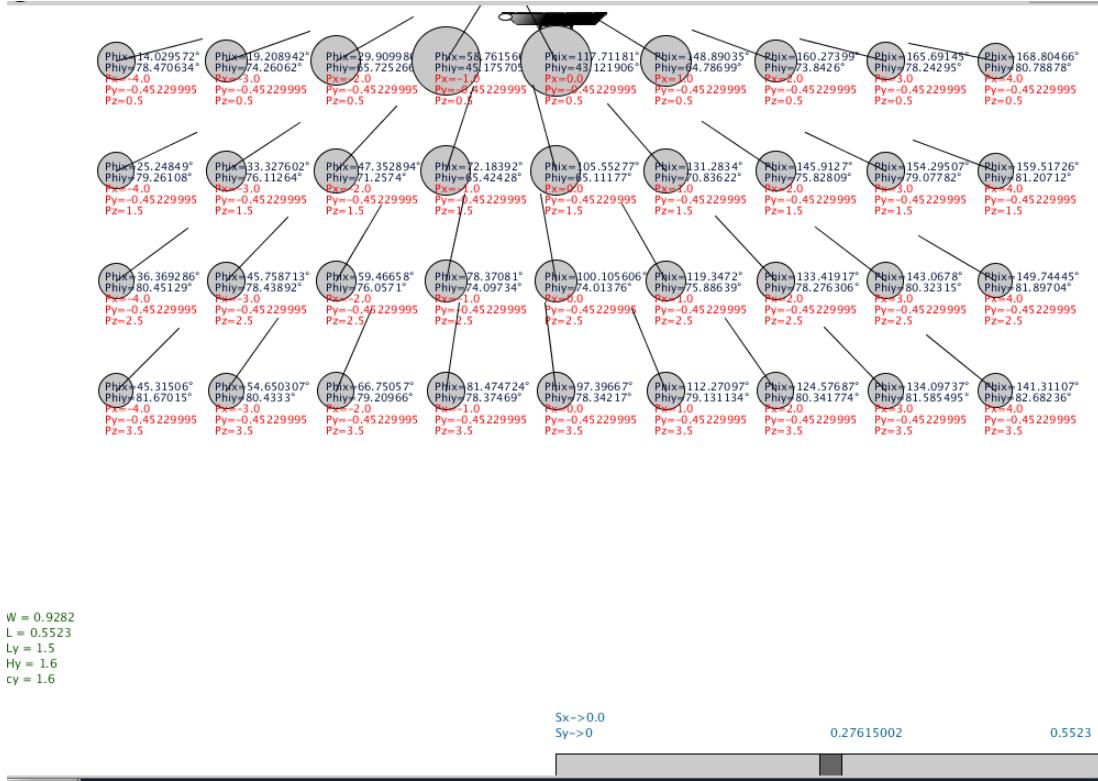


Figura 5.16:

Imagen 5.17, las marcas rojas son la proyección en la imagen de la intersecciones con el plano y las verdes son los puntos de $scnPts$ rotados, trasladados y proyectados. Como se puede observar el cálculo de la n y d del plano es bastante precisa, mientras que la R y T tienen error, y parece ser mayor en el desplazamiento sobre el eje Y .

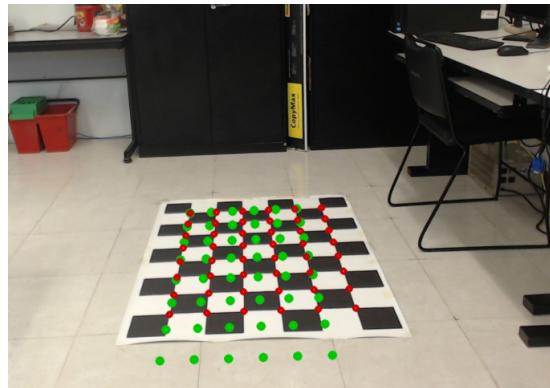


Figura 5.17: Reproyección de puntos

Se realizó un análisis de la distancia que hay entre los que se obtiene de los puntos rotados, trasladados y proyectados con lo que debiera ser ($imgPts$) en pixeles y se obtuvo que 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) y la distancia es de: 52 pixeles en X y 107 en Y .

El error en metros en tres dimensiones con respecto al primer y último punto es el siguiente: 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 se puede observar el error se va expandiendo conforme los puntos se alejan del primer punto. El error en R y T es mucho mayor que el que hay en n y d esto se debe a que estos últimos parámetros se calculan con respecto al primer punto el cual como ya se analizó tiene menos error, sin embargo; es necesario que R y T sean más precisos porque se utilizan bastante en etapas posteriores, por lo que es necesario optimizar estos parámetros.

5.3.2. Optimización Levenberg-Marquardt

Para evaluar los resultados de aplicar el algoritmo de optimización numérica mencionado en la sección 4.6.2 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:

$$F(p) = 1,588771e^{-3} \quad (5.1)$$

Se compararon los resultados del mismo modo que en la sección anterior, midiendo la distancia de los puntos en metros rotados y trasladados con la R y T resultantes de la optimización con lo que en realidad debería ser, esto es, los puntos que resultan de la intersección de la proyección de los puntos con el plano del piso. En el primer punto se obtuvo un error de: en X $-0,00145m$, en Y $0,00594m$ y en Z $0,01289m$. En el último en el cual se notaba más el error de la rotación y traslación, ya con la optimización se obtuvieron los siguientes resultados: en X $0,0449m$, en Y $0,0423m$ y en Z $0,1424m$. La mayor diferencia se puede ver el último punto en la coordenada X ya que pasa de una diferencia de $26,5cm$ a $4cm$. En la figura 11.6 se puede apreciar la comparación de los puntos después de optimizar y los resultados son superiores a comparación de lo que hay antes de la optimización.

Sin embargo, en el último punto (esquina inferior derecha)

5.3.3. Corrección de la imagen y recalibración de la cámara

Como se busca que la proyección de los puntos 3d en la imagen con los puntos seleccionados sean lo más precisos posibles se recalibró la cámara (a diferentes

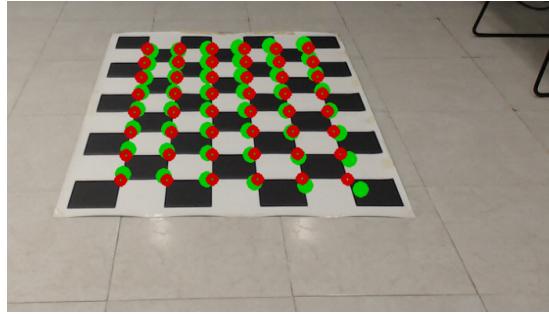


Figura 5.18: Reproyección de puntos

resoluciones) con la que se realizan los experimentos mediante herramientas de OpenCV y esta vez se corrigió la distorsión de la cámara 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 y el vector de traslación.

En la imagen de la izquierda de la figura 4.26 se puede apreciar la imagen antes



Figura 5.19: Corrección de la imagen

de la corrección y en la de la derecha con la corrección aplicada.

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 5.20.

Aplicando nuevamente la optimización numérica a partir del resultado anterior da resultados aún mejores ya que después de 1000 iteraciones se obtiene un mínimo de $F(p) = 2,908337e - 10$ y el resultado gráfico se ve en la imagen 5.21.

5.4. Protocolo de captura de datos

En este punto del proyecto de tesis ya se han descrito casi todos los elementos para realizar la captura de datos:

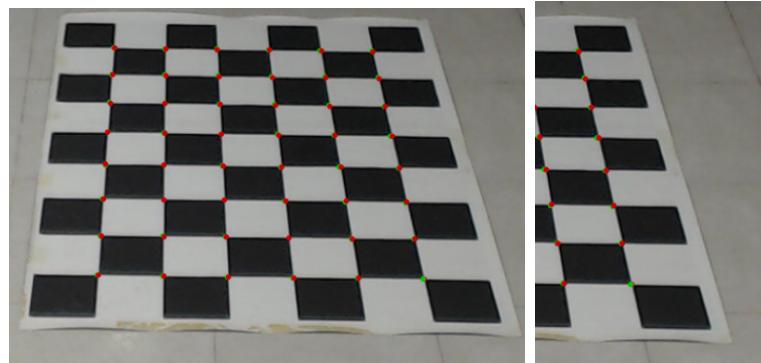


Figura 5.20: Estimación de R y T antes de la optimización



Figura 5.21: Estimación de R y T después de la optimización p

- 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 el ángulo de la mirada
- Generación de secuencia de instancias (marcas en el piso) óptimas para la captura

Sin embargo aún falta describir el proceso de captura de datos, los cuales serán utilizados en el algoritmo de entrenamiento.

5.4.1. Proyección en tiempo real de instancias óptimas para marcar en el piso

El protocolo de captura comienza marcando en el piso las instancias óptimas obtenidas mediante el algoritmo descrito en la sección 4.4, para ello se desarrolló un programa que recibe como entrada las instancias de marcas, las proyecta y pinta en tiempo real sobre lo que esté capturando la cámara, esto se realiza como guía para o referencia al momento de marcarlas físicamente en el piso del laboratorio de experimentación.

Como se ha mencionado los puntos se eligieron tomando en cuenta una región limitada en base a análisis previos y con la restricción (entre otras) de que el rostro de la persona P estuviera dentro del campo visual de la cámara. Esto significa que pudieran darse marcas en el piso F que no se encuentran dentro del campo visual de la cámara, por ejemplo aquellas que se encuentran más cercanas a la cámara, imagen 5.22.

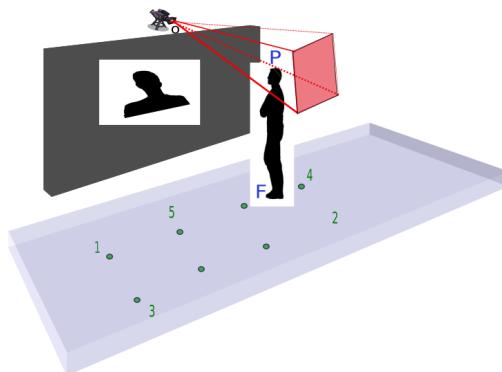


Figura 5.22: Campo visual

Dibujo de rectas graduadas

Para solucionar el problema de de instancias F generadas no perceptibles debido a la restricción del campo visual, se desarrolló un algoritmo que utiliza la rotación traslación y parámetros del plano, y dibuja en el piso de la escena las marcas F, en todos los casos se dibujan líneas graduadas paralelas a los ejes X y Y del marco de referencia. A las rectas se les dibuja una pequeña marca cada 10cm, además se dibujan a un lado de las marcas 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 ver 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 5.23 y como se puede apreciar hay unas marcas en las que no se alcanza a apreciar la *F* pero la *P* si. Las marcas están enumeradas, esto indica el orden que la persona deberá seguir para la captura de sus datos.

La recta amarilla fue colocada y pintada con respecto a la altura real del sujeto

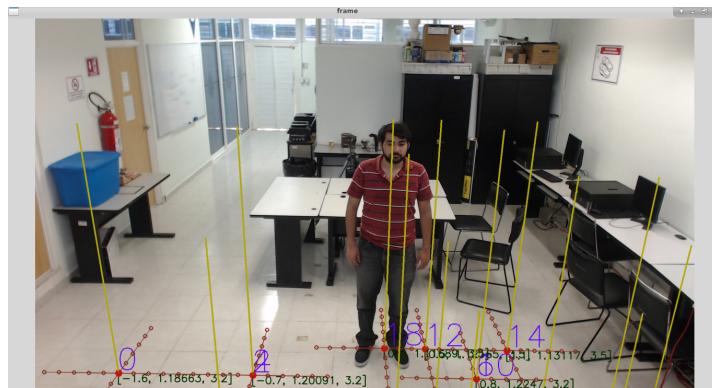


Figura 5.23: Campo visual

de experimentación que aparece en la misma imagen. La orientación de la recta tomado en cuenta que es ortogonal al plano parece estar correcta esto indica que la estimación de la normal del plano *n* fue estimada con precisión, sin embargo la longitud de la recta se pasa por 3cm.

5.4.2. Captura de datos



Figura 5.24: Imagen obtenida durante la etapa de captura

5.5. Red neuronal

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

5.5.1. Detector de rostros y guardado de los ejemplos útiles

La primera etapa de esta sección consiste en preparar las imágenes y datos obtenidos de la captura de datos para introducirlos en la red neuronal (incluyendo P y S), esto conlleva 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. Hay casos en los que a pesar de que el rostro está visible en la imagen, el algoritmo de Viola y Jones no logra detectar el rostro, en este caso descartamos la imagen, ya que para lograr estimar la mirada de las personas con el sistema que se desarrolla es fundamental previamente el rostro con el algoritmo de Viola y Jones.

En la figura siguiente 5.25 se pueden ver algunos ejemplos de las imágenes capturadas y procesadas con el detector de Viola y Jones.

En la imagen 5.26 se presenta un falso positivo del algoritmo detector, el cual es rechazado para el conjunto de entrenamiento. Del total de las 383 después de procesarlas con el detector se obtuvieron 244 rostros útiles para el conjunto de entrenamiento. En la figura 5.27 se encuentran algunos de los rostros que si se



Figura 5.25: Imagenes procesadas con Viola y Jones



Figura 5.26: Imagen procesada con Viola y Jones



Figura 5.27: Imágenes de los rostros guardadas

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

utilizan para el conjunto de entrenamiento. Las resoluciones de los rostros capturados son muy variadas, pero como se mencionó en la sección 4.7.1, se deben redimensionar todas a un estándar, ya que el número de características y neuronas de entrada depende de dicha resolución. La resolución elegida para estas pruebas fue de 75x75 pixeles.

5.5.2. Entrenamiento y pruebas

Ya con los datos listos para el algoritmo de clasificación estos se dividen en dos conjuntos, el de entrenamiento y el de prueba, en una primera iteración se probó con 204 imágenes o ejemplos de entrenamiento y 40 de prueba.

Dos neuronas de salida

Primero se experimentó discretizando la pantalla (pintarrón) que observaban los sujetos de los experimentos en 2 secciones de misma longitud, partiendo el eje X a la mitad 5.28. Con una capa oculta se obtuvieron los siguientes resultados: Se puede apreciar en 5.1 que incluso con una capa oculta y una neurona se

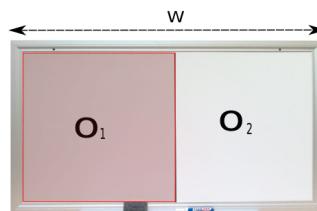


Figura 5.28: Pintarrón dividido en dos secciones

consiguen unos resultados de 95 % en tan solo 0.057ms que le toma a un ejemplo propagarse hacia adelante. Los mejores resultados con una capa se obtienen 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 que el empleado cuando se utiliza solamente una capa oculta.

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

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

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

Tres neuronas de salida

A continuación se presentan los resultados de utilizar tres clases en la clasificación de la mirada, por lo tanto el en la capa de salida de la neuronal hay tres neuronas. En la figura 5.29 se encuentra ilustrada la segmentación del pintarrón.

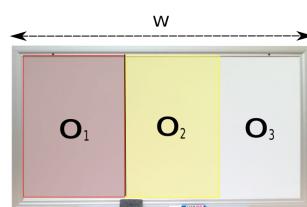


Figura 5.29: Pintarrón divido en tres secciones

Con solo una capa oculta 5.4 el mejor resultado que se obtuvo es 87.5 % utilizando 4 neuronas en dicha capa.

En la tabla 5.5 se puede observar que usando dos capas ocultas se obtienen resultados muy superiores a usar solo una capa, incluso se llega a una eficiencia de 97.5 %. Los mejores resultados se consiguen al colocar má de 10 neuronas en las capas ocultas. En 5.6 se puede apreciar que a pesar de haber más neuronas y

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

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: 3 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

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

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 bastante buenos como los obtenidos con solo dos clases: 97.5 %

Cuatro neuronas de salida

En esta sección se presentan los resultados de utilizar cuatro clases en la clasificación de la mirada, llevando a cabo una partición de la pantalla como la de la figura 5.30.

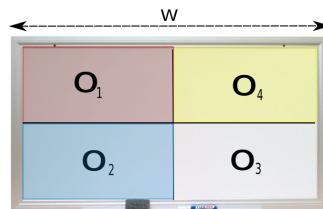


Figura 5.30: Pintarrón dividido en cuatro secciones

En 5.7 se presentan los resultados de utilizar una capa oculta, en seguida se puede apreciar que los resultados 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 mas de 20 neuronas, llegando incluso a una eficiencia hasta 92.5 %.

Utilizando tres capas ocultas el desempeño del algoritmo decae y lo máximo

Cuadro 5.9: 3 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

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

que se puede obtener es 87.5 % de eficiencia con una topología 22 – 20 – 20 – 4 en 0.157ms, tabla 5.9.

Entrenamiento y experimentos con descriptores HOG

A continuación se presentan 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. El vector de características de la red pasa de 5629 elementos a 2029.

En la tabla 5.10 se pueden apreciar los experimentos de utilizar dos neuronas de salida (2 segmentos en la pantalla), los resultados son muy superiores ya que se logra 100 % de eficiencia con tan solo una capa oculta de 5 neuronas en tan solo 0.025ms, tal eficiencia no se logró 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 3 neuronas de salida se encuentran en 5.11, sin utilizar HOG lo máximo de eficiencia que se lograba con 3 clases era 97.5 % ahora con HOG se alcanza hasta un 100 % usando 3 capas ocultas. Incluso con una capa se alcanza 97.5 % mientras que en los experimentos anteriores (sin HOG) lo máximo obtenido con una capa era 80 %.

En 5.12 se pueden apreciar los resultados de utilizar 4 neuronas de salida en la red neuronal y con los descriptores HOG como parte del vector de características. Los resultados son superiores a utilizar toda la imagen como características ya que el mejor de esos resultados con 4 clases era de 92.5 % en 0.213ms con una

Cuadro 5.11: 3 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

Cuadro 5.12: 4 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

topología de 25-16-4 y ahora con HOG se llega incluso a 95 % de eficiencia en 0.0415ms y dos capas ocultas.

Se puede concluir que al utilizar los descriptores HOG como parte del vector de características se obtienen resultados superiores en todos los casos de clasificación y en mucho menor tiempo. Además, no está descrito en las tablas pero el tiempo empleado en el entrenamiento utilizando HOG es mucho menor ya que pasa de un tiempo promedio de 1 minuto (sin HOG) a 5s.

Capítulo 6

Resultados

Capítulo 7

Conclusiones

Apéndice A

Example Appendix

Appendices are usually labelled with letters separate to ordinary chapters.

Apéndice B

Referencias

B.1. Referencias

Bibliografía

- [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] 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