Capítulo 2. Marco Teórico

El propósito del presente capítulo, es realizar un estudio de diferentes modelos de detección e identificación facial, con el objetivo de analizar sus características. Adicionalmente se revisarán prototipos de sistemas de video vigilancia para identificar las características y consideraciones para su funcionamiento.

2.1 Detección e Identificación Facial

Para lograr la identificación facial, el primer paso es detectar un rostro, existen diversos algoritmos para detección facial, algunos localizan y extraen regiones, otros utilizan filtros los cuales permiten identificar rostros.

Principalmente existen dos tipos de reconocimiento: intrusivo y no intrusivo (Sinsinwar & Dwivedi, 2014), en el intrusivo el usuario está consciente sobre el reconocimiento, por ejemplo, el escaneo de la palma de la mano, el reconocimiento facial es no intrusivo ya que sin la necesidad de que la persona coopere puede ser identificada para ser autenticado.

(Kim, Kumar, Pavlovic, & Rowley, 2008), abordan el problema de rastreo e identificación facial en el mundo real, tomando como base videos con una gran cantidad de ruido. Obteniendo resultados superiores al 80% en el seguimiento de personas.

Existen otras técnicas para el reconocimiento facial muy efectivas, con resultados de exactitud muy altos, como el que proponen (Gumus, Kilic, Sertbas, & Ucan, 2010), donde utilizan básicamente dos métodos, *wavelet decomposition y Eigenfaces* basado, en *Principal*

Component Analysis (*PCA*) como técnicas principales para la reducción de datos y la extracción de características.

2.1.1 AdaBoost- Algoritmo *Viola Jones*

(Viola & Jones, 2001) proponen un método de aprendizaje automático para detección visual de objetos, el cual, es capaz de procesar imágenes muy rápidamente y lograr promedios muy altos de detección de objetos en tiempo real. Adaboost es un algoritmo de aprendizaje, el cual combina una colección de clasificadores débiles para formar un clasificador robusto. (Viola & Jones, 2001) hacen tres importantes contribuciones:

1. Proponen una nueva representación de una imagen, llamada imagen integral.
2. Un algoritmo de aprendizaje.
3. Un método para combinar e incrementar en cascada clasificadores más complejos.

El concepto de imagen integral permite hacer los cálculos muy rápidamente, a través de una transformación previa a todos los pixeles de la imagen, con el objetivo de generar una matriz sobre la cual se van a realizar los cálculos al recorrer diversas máscaras de diferentes escalas sobre la imagen llamadas características *Haar-Like* (Figura 2).

La imagen integral en la ubicación (*x, y*) contiene la suma de los píxeles por encima y a la izquierda de (*x, y*) incluyéndolos, en la siguiente ecuación (ec. 1) se representa este proceso:

$$ii(x, y) \sum_{x' \leq x, y' \leq y} i(x', y'), \tag{1}$$

Donde $ii(x, y)$ es la imagen integral y $i(x, y)$ es la imagen original. Usando las siguientes ecuaciones para calcular la suma rectangular (ec. 2 y ec. 3):

$$s(x, y) = s(x, y - 1) + i\,(x, y) \quad (2)$$

$$ii(x, y) = ii(x - 1, y) + s(x, y) \quad (3)$$

Al hacer uso de la imagen integral, cualquier suma rectangular puede ser calculada con cuatro referencias.

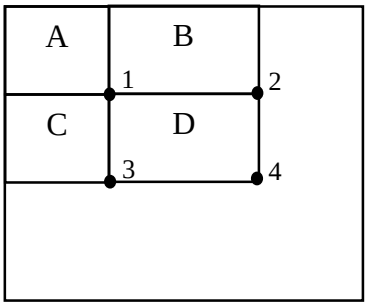

Figura 1. Representación de suma rectangular.

La suma de pixeles dentro del rectángulo D (Figura 1), se puede calcular de la siguiente manera: el valor de la imagen integral en el punto 1, es la suma de pixeles del rectángulo A. el valor en el punto 2 es A+B, en el punto 3 es A+C, en el punto 4 es A+B+C+D. La suma finalmente puede ser calculada como 4+1- (2+3).

Este proceso permite realizar los cálculos muy rápidamente, al aplicar sobre una imagen las características *Haar-Like*, logrando de esta manera, realizar la detección de rostros en tiempo real.

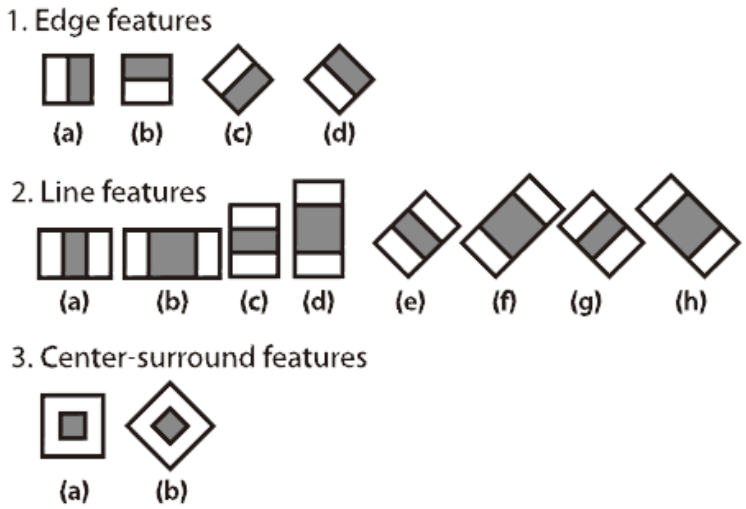

Figura 2. Características *Haar-Like.*

Cada clasificador fue entrenado previamente para detectar rostros con las características *Haar-Like*, estos clasificadores se aplican en cascada en una región de interés para poder detectar un rostro, descartando rápidamente las regiones que no incluyen un rostro, una región se considera un rostro solamente si cada uno de los clasificadores aprueba que es un rostro.

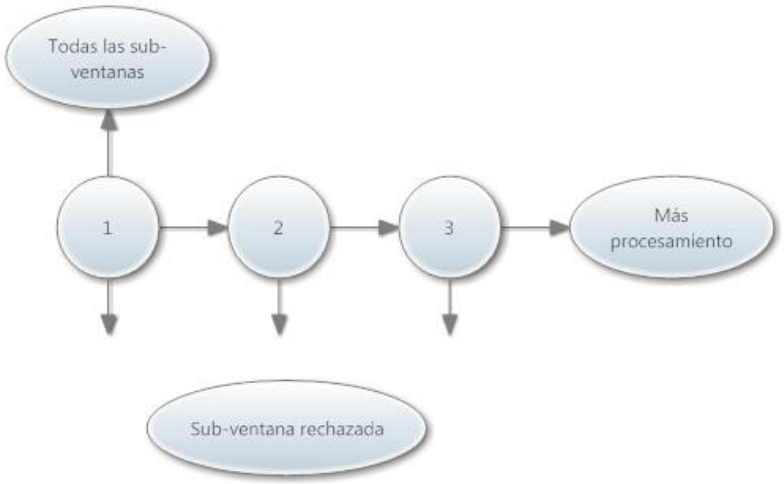

Diagrama 5. Clasificadores *Haar-Like* en cascada.

2.1.2 Eigenfaces

(Turk & Pentland, 1991) presentan un enfoque, para la detección e identificación de rostros humanos y un sistema de reconocimiento facial en tiempo casi real, el cual rastrea y localiza la cabeza de un sujeto y luego identifica a la persona, a través de la comparación de las características de la cara de individuos conocidos.

Este enfoque trata el reconocimiento facial como un problema de reconocimiento de dos dimensiones, aprovechando el hecho de que las caras se encuentran normalmente en posición vertical y por lo tanto pueden ser descritos por un pequeño conjunto de características de dos dimensiones.

El proceso de reconocimiento es el siguiente

- Se obtiene el conjunto de imágenes de caras para el entrenamiento y se calculan los *eigenfaces* los cuales definen el espacio de rostros.
- Cuando se encuentra una nueva cara se calculan los pesos basados en la imagen de entrada y los *n eigenfaces* proyectando la imagen de entrada en cada uno de los *eigenfaces*.
- Determinar si una imagen es una cara, esto se lleva a cabo al revisar si la imagen está suficientemente cerca al espacio de rostros.
- Si es una cara, clasificar el patrón de peso ya sea como persona conocida o persona desconocida.
- Si la misma cara desconocida es vista en repetidas ocasiones, se calcula sus características de patrón de peso y se incorpora en las caras conocidas.

El espacio de la cara está definido por los *eigenfaces,* que son los vectores propios (*eigenvectors*) de un conjunto de caras, los cuales no corresponden necesariamente a características aisladas tales como los ojos, las orejas y la nariz.

El proceso de reconocimiento, tiene la habilidad de aprender a reconocer nuevas caras de una manera no supervisada.

2.1.3 OpenCV

(Bradski & kaehler, 2008) mencionan que *OpenCV* son librerías de código abierto para visión en computadora. La librería está escrita en lenguaje C y C++, las cual se puede ejecutar en diversos sistemas operativos. *OpenCV* está escrita en C optimizado, la cual puede tomar ventaja de los múltiples núcleos de un procesador.

Uno de los principales objetivos de *OpenCV,* es proveer una infraestructura simple de utilizar, que ayude a las personas a construir aplicaciones sofisticadas de visión por computadora, de una manera más rápida.

OpenCV tiene una estructura de 5 componentes principales:

1. Componente CV.- Contiene los algoritmos básicos de procesamiento de imagen y visión por computadora de alto nivel.
2. Componente MLL.- Es la librería de aprendizaje la cual incluye múltiples clasificadores estadísticos y herramientas de agrupación.
3. Componente HighGUI.- Contiene rutinas de entrada y salida (I/O) y funciones para almacenamiento y carga de video e imágenes.

4. Componente CXCore.- Contiene la estructura básica de datos y contenidos, soporte para archivos XML y funciones de dibujo.
5. Componente CvAux.- contiene algoritmos experimentales de segmentación *Eigen objects*, modelos de Markov, *Stereo Vision, 3D tracking,* entre otros.

La librería *OpenCV* implementa un conjunto de funciones para realizar detección de rostros, para lograrlo se hacen llamadas a funciones que reciben como parámetros el archivo XML (donde se especifican las características de detección para el que fue entrenado), los cuales indican si se encuentra un rostro o un objeto en la imagen, el factor de escala, el número de pixeles vecinos a recorrer, el tipo de detección y el tamaño de la ventana.

2.1.4 Formatos de Archivo de Video

Los videos digitales se pueden almacenar en archivos de distintos formatos, existen múltiples tipos de formatos de video, los cuales utilizan diferentes algoritmos de compresión (llamados *codec*), para efectos de esta investigación no se describirán dichos algoritmos, ya que en la mayoría de los casos es tecnología propietaria y no hay referencias de su funcionamiento interno.

Windows Media Video, es el nombre genérico para hacer referencia al conjunto de algoritmos de compresión para tecnologías de video, desarrolladas por Microsoft y que forman parte de la plataforma Windows Media.

Los formatos de video más utilizados son:

- AVI
- MPEG

- MOV
- WMV
- RM
- FLV

Para llevar a cabo el almacenamiento en video, se utilizará la librería *AForge.NET Framework*, la cual permitirá guardar en video, los fotogramas capturados por la cámara en formato AVI.

Características principales del Formato AVI (*Audio Video Interleaved*):

- Es el formato estándar para almacenar video digital.
- Por lo regular se utiliza el códec DV (Digital Video) para almacenar video desde una cámara digital.
- Contiene una calidad excelente.
- El formato AVI puede ser visualizado en la mayoría de reproductores.

Con base en estas características y las ventajas que proporciona, se seleccionó el formato AVI para realizar el almacenamiento de la secuencia de video, en el dispositivo de almacenamiento, que para el caso del prototipo será el disco duro.

2.2 Sistemas de Video Vigilancia

No es relevante cuál sea la aplicación de video vigilancia, si se graba en video es necesario almacenarlo (Milestone Systems, Best Practices in Video Surveillance Storage, 2008). Basados en la idea anterior, es muy importante considerar que la capacidad de

almacenamiento de todos los dispositivos, tiene un límite y es necesario maximizar el rendimiento de estos recursos.

Otra propuesta plantea la detección de rostros, basado en *Tracking-Learning-Detection (TLD),* (Kalal, Mikolajczyk, & Matas, 2010) implementan un detector genérico y un validador diseñados para el rastreo facial en tiempo real, resistente a oclusiones y cambios de apariencia. El detector entrenado fuera de línea localiza caras frontales y el validador en línea entrenado decide cuales caras corresponden al sujeto localizado.

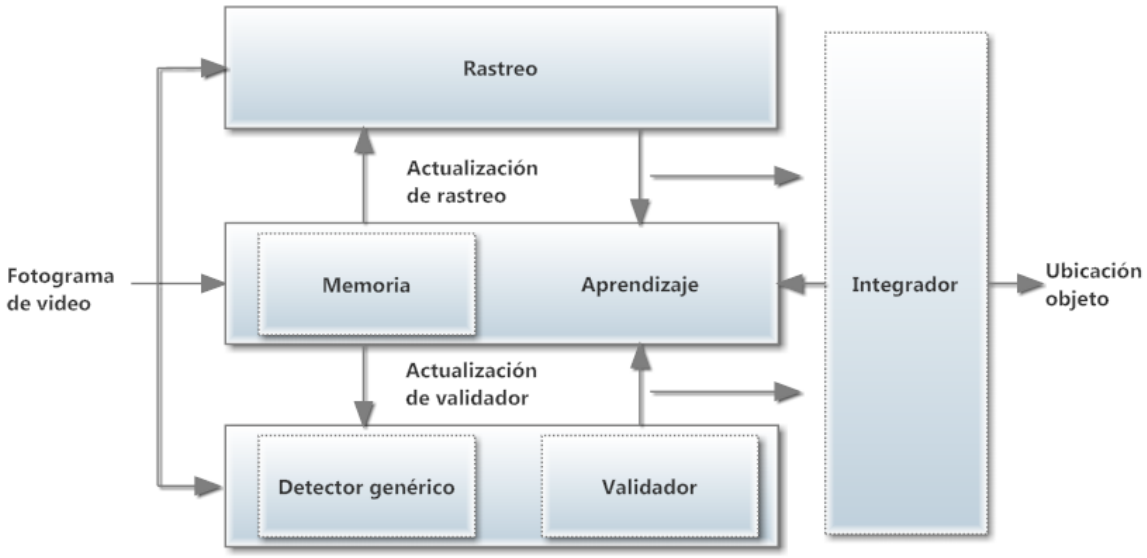

Diagrama 6. Arquitectura modelo *TLD.*

En el modelo TLD, cada fotograma es procesado por un rastreador, un detector y sus salidas son enviadas a un integrador el cual su función es estimar la ubicación del objeto. La ubicación del objeto, la detección y trayectoria son analizadas por un bloque de aprendizaje que genera datos de entrenamiento para el detector. El detector genérico regresa las regiones que corresponden a cierta clase, el validador decide si el objeto representa una instancia específica seleccionada para el rastreo.

En las pruebas que realizaron, un sujeto aparece en 12,222 fotogramas, la secuencia completa de video fue de 35,471 fotogramas, El modelo TLD rastreó y detectó correctamente al principio de la secuencia, pero falló en la segunda parte. La prueba tuvo un precisión del 70% con respecto al rastreo-detección.

Finalmente podemos decir que es de suma relevancia, que la información que se almacena en video incluya contenido importante para el usuario. En el caso de cámaras de video vigilancia la información relevante corresponde a información donde aparecen personas.

Con base en la revisión del estado del problema, estudios de mercado y tomando en cuenta las características de las diferentes propuestas que se analizaron en el capítulo, se seleccionó para su implementación el algoritmo AdaBoost para el desarrollo del prototipo por sus características de precisión, velocidad, rendimiento en tiempo real y procesamiento de fotogramas en video para el desarrollo de sistemas mejorados de almacenamiento, así mismo las técnicas de procesamiento de video, seleccionando hardware y software a utilizar, fueron precisados los objetivos, hipótesis y actividades que permiten definir las líneas de investigación de los siguientes capítulos.