UNIVERSIDAD NACIONAL

Facultad de Ciencias Exactas y Naturales

Escuela de Informática

Cálculo objetivo de la distancia interpupilar mediante el uso de inteligencia artificial para el seguimiento de tratamientos de estrabismo.

Para optar por el grado de Licenciatura en Informática

con énfasis en sistemas web

Estudiante:

Ing. Alejandro Sánchez Ortega

I Ciclo, 2020

Heredia, Costa Rica

Dedicatoria:

Para Elizabeth,

Dios y la vida me premiaron con la mejor abuela, fuerte, trabajadora e inspiradora.

Esto es para ti, Eli.

Gracias por tanto apoyo, gracias por tanto amor.

Agradecimientos:

Antes que nada, agradezco a Dios por permitir culminar este proceso, el cual no ha sido fácil, sin embargo, muchas personas estuvieron a mi lado durante todo este largo camino; profesores, amigos y colegas.

A mi familia, que nunca dejó de respaldar mi esfuerzo. A Michelle, que creyó en mi aún cuando ni yo mismo lo hacía.

Sin mas, un afectuoso agradecimiento a esos que creyeron en mí.

Ing. Alejandro Sánchez Ortega

TABLA DE CONTENIDOS

TABLA DE FIGURAS

CAPÍTULO I
INTRODUCCIÓN

La presente investigación “Cálculo objetivo de la distancia interpupilar mediante el uso de inteligencia artificial para el seguimiento de tratamientos de Estrabismo”, nace con el propósito de darle trazabilidad a las diferentes fases del tratamiento de estrabismo por medio del uso de tecnologías no tradicionales, como lo es la inteligencia artificial.

Existen distintos métodos manuales para el diagnóstico del estrabismo en las personas, como por ejemplo el Cover test y el Test de Hirschberg, los cuales, en cuanto a efectividad y precisión, dependen de la experiencia y la habilidad visual del evaluador o médico a cargo de dichas pruebas, por lo cual los resultados obtenidos pueden ser imprecisos y poco confiables.

Por lo descrito anteriormente, se propone hacer uso de tecnológicas no tradicionales en el área de la medicina, en la cual su uso (tecnología) es una tarea cotidiana, no solo en Costa Rica sino en el mundo.

Este documento presenta la investigación, el desarrollo y la prueba de la aplicación realizada. A continuación, se detalla la descripción de los antecedentes, planteamiento del problema, justificación, objetivos del proyecto; luego se desarrolla el marco teórico; seguido por el procedimiento metodológico de la investigación, también se realiza un análisis retrospectivo y finalmente se detallan conclusiones y recomendaciones obtenidas del proyecto.

CAPÍTULO I: INTRODUCCIÓN

1.1 Antecedentes

El uso de la tecnología en medicina posee múltiples beneficios, como lo son:

- Tratamiento de enfermedades cardiovasculares, las cuales han reducido drásticamente el número de personas que mueren a causa de un ataque al corazón o que sufren de una insuficiencia cardiaca.
- Diabetes, las personas que sufren de esta enfermedad ahora tienen acceso a tecnologías muy exactas de monitoreo de glucosa en la sangre, reduciendo considerablemente el riesgo de sufrir complicaciones.
- Entre muchos otros beneficios en procedimientos quirúrgicos, tratamiento de cataratas y otros males. (Clinic-Cloud ,2016, p.1).

Las ventajas del uso de la tecnología en salud redundan principalmente en los pacientes siendo los principales beneficiados. Sin embargo, los médicos también obtienen beneficios, por ejemplo, al utilizar diferentes tecnologías pueden reducir el margen de error al realizar un procedimiento médico. Estos beneficios que van más allá de la gestión de pacientes, citas e incluso hospitales que ayudan a la industria médica a manejar de manera correcta sus recursos. (Clinic-Cloud ,2016, parr.5).

En cuanto a la computación cognitiva, se menciona que “el objetivo de la computación cognitiva es crear sistemas informáticos automatizados que sean capaces de resolver problemas sin necesidad de asistencia humana” (Rouse, 2017, parr.1). Por lo tanto, se planteó la alternativa de incluirlo en la presente investigación, la cual trató de automatizar mediante el uso de la inteligencia artificial un proceso que se realiza de forma manual. Este concepto hace una referencia implícita al concepto de inteligencia artificial.

La inteligencia artificial en la industria de la medicina tiene algunos antecedentes y empresas líderes que deben ser mencionadas, un claro ejemplo de esto es la empresa IBM, que se destaca como pionero del uso de esta tecnología, además es el desarrollador del súper computador llamado Watson que es un sistema informático que lee y entiende el lenguaje natural siendo entrenado con miles de datos y respondiendo de manera asertiva preguntas

en cuestión de segundos, entre muchas otras capacidades que el mismo posee, como lo son el procesamiento de imágenes, entre otros (IBM Watson,2016, p.2).

Como parte de las tareas que posee actualmente IBM es la aplicación de Watson en la rama de la salud en la especialidad de oncología, la cual parte de la historia clínica del paciente con la capacidad de obtener y analizar el significado y el contexto de los datos estructurados y no estructurados en las historias clínicas e informes. Seguidamente identifica las posibles opciones de tratamiento, basado en la evidencia y en los datos recolectados de casos que fueron registrados en el pasado, dadas estas posibles opciones, los médicos analizan cual es que mejor se ajusta a las condiciones del paciente contando con la gran ventaja que Watson brinda estas opciones en pocos minutos, y a los expertos les toma cerca de dos semanas realizar el mismo proceso. La aplicación de este software ya cuenta con algunos casos de éxito y muchos hospitales están interesados en la utilización de este (Muñoz, 2016).

Otro de los conceptos importantes dentro de la investigación es el estrabismo, el cual según *American Academy of Ophthalmology* se define como un problema visual que hace que los ojos no estén alineados correctamente y apunten en diferentes direcciones (Jambrina, 2014, p.1).

Como se mencionó anteriormente, el uso de la tecnología en la medicina es de suma importancia. Por esta razón este proyecto se orienta al desarrollo de un prototipo el cual sea capaz de calcular el ángulo de desviación por medio el reflejo de Hirschberg, que funcione como herramienta valiosa para el examinador que deba realizar la prueba para obtener el ángulo de desviación, debido que tiene el inconveniente de estar en manos de la observación que realice el examinador, por tanto, este examen no es preciso.

1.2 Planteamiento del problema

Para diagnosticar el estrabismo se realizan una serie de exámenes los cuales abordan básicamente dos aspectos muy importantes: el primero corresponde a un aspecto sensorial y el segundo tiene que ver con un aspecto motor. Para este último aspecto se hace un estudio de la desviación mediante las pruebas de reflejo de Hirschberg y Cover test.

Estas pruebas sirven para confirmar el diagnóstico de estrabismo y determinar el ángulo de desviación y además un estudio de los movimientos oculares. La solución que se pretende desarrollar trata de resolver la problemática que existe al realizar el diagnóstico para obtener el ángulo de desviación por medio del reflejo de Hirschberg ya que este método tiene el gran problema que la medición depende solamente de la observación que realiza el examinador, es decir existe una alta probabilidad de que el examinador falle en su diagnóstico ya que depende mucho de su habilidad visual y sus años de experiencia realizando esta prueba.

El prototipo propuesto por desarrollar en esta investigación permite cooperar con el diagnóstico para la obtención del ángulo de desviación por medio del reflejo de Hirschberg, ya que este método implica una medición que depende solamente de la observación que realiza el examinador. Por lo tanto, se deduce que existe una alta probabilidad de que el examinador falle en su diagnóstico, porque depende de la habilidad visual y años de experiencia realizando esta prueba, además, que los datos que el examinador tiene la capacidad de recolectar no son exactos

Asimismo, la herramienta tiene la característica de recolectar datos de la distancia interpupilar y el ángulo de desviación de cada ojo, en las diferentes etapas del tratamiento, brindando datos que pueden ayudar al examinador a dar seguimiento al tratamiento.

1.3. Justificación

El número de personas afectadas por estrabismo alrededor del mundo es considerable, ya que para el 2006 se estimaba que podrían ser de 130 a 260 millones de personas afectadas (National Eye Institute Congressional Justification for FY, 2006). Además, la prevalencia de estrabismo en los Estados Unidos de Norteamérica es de aproximadamente 7.5 millones de personas (Research to Prevent Blindness, NISE, NSF) lo que significa que 1 de cada 36 o lo que es lo mismo 2.76% de personas sufre de esta enfermedad.

La presente investigación da trazabilidad en las diferentes fases del tratamiento de estrabismo, por medio de la creación de una herramienta tecnológica que se desempeñe como apoyo para los especialistas encargados de detectar este mal visual en las personas.

Es importante apoyar las ciencias médicas mediante el uso de la tecnología, razón por la cual este proyecto se orienta al desarrollo de un prototipo de una aplicación capaz de calcular objetivamente la distancia interpupilar mediante el uso de inteligencia artificial para el seguimiento de tratamientos de estrabismo.

Asimismo, se procura automatizar la toma de medidas de la distancia interpupilar, misma que se obtiene aplicando procedimientos médicos como el test del reflejo de Hirschberg, con la finalidad de calcular datos precisos y confiables.

Dada la evidencia de casos de estrabismo que se mencionó con anterioridad en este documento, acerca de la gran cantidad de personas afectadas por este mal visual, es vital resaltar la importancia del desarrollo de una herramienta como la que se propone, aun cuando existen otros métodos para dar seguimiento a tratamientos de estrabismo, se

pretende brindar al examinador una herramienta que esté a su alcance en todo momento, como lo es una aplicación móvil.

Finalmente, este proyecto de graduación tiene elementos innovadores, los cuales se enumeran a continuación:

1. Esta investigación se destaca por generar un producto amigable y de fácil al acceso al ser una aplicación móvil.

2. La empresa patrocinadora de este proyecto, que lleva por nombre Ártico Soluciones, corresponde a una pyme, dentro de lo que corresponde a una pequeña empresa.

3. La utilización de la inteligencia artificial en cuanto a soluciones de software en el mercado costarricense, debido que existen pocas empresas en el país que apliquen dicha tecnología en sus proyectos.

4. La solución trata de automatizar un procedimiento que actualmente se realiza de manera visual dejando toda la responsabilidad al evaluador.

1.4. Objetivos del Proyecto

1.4.1 Objetivo general

Facilitar el de cálculo objetivo de la distancia interpupilar mediante el uso de inteligencia artificial para el seguimiento de tratamientos de estrabismo.

1.4.2 Objetivos específicos

I. Investigar los métodos convencionales de obtención del ángulo de convergencia en el diagnóstico de estrabismo para el desarrollo de software orientado a una arquitectura de tipo cliente-servidor con la finalidad de establecer las herramientas y lenguajes de programación adecuados para el procesamiento de imágenes y datos.

II. Desarrollar una aplicación de software que se ejecute del lado del servidor, que realice el cálculo objetivo de la distancia interpupilar y el ángulo de desviación de cada ojo.

III. Probar los datos calculados en el servidor de aplicaciones mediante el desarrollo de la aplicación móvil en un ambiente de desarrollo para validar su confiabilidad en conjunto con expertos en el tema.

CAPÍTULO II
MARCO TEÓRICO

CAPÍTULO II: MARCO TEÓRICO

2.1 Marco referencial

Se investigó acerca de los avances en medicina, específicamente en el padecimiento de estrabismo, tratando de conocer antecedentes de aplicaciones móviles utilizadas para calcular la distancia interpupilar o relacionadas al estrabismo y también investigaciones formales relacionadas a este tema, los hallazgos obtenidos son los siguientes:

- En la tienda virtual para la descarga de aplicaciones móviles de dispositivos de la marca Apple, no existe ninguna aplicación dedicada al cálculo de la distancia interpupilar. Pero existen dos aplicaciones referentes al estrabismo de manera didáctica, es decir, tratan de enseñar como se realizan pruebas para el diagnóstico de estrabismo, estas son:

    1. Occuler.
    2. Oftalmología pediátrica y estrabismo.

- En el caso de la tienda virtual para la descarga de aplicaciones móviles que utilicen el sistema operativo Android, existen algunas aplicaciones con relación al estrabismo o problemas de la vista, con fines didácticos y otra con la finalidad de proveer herramientas para realizar ciertos exámenes de la vista. Sin embargo, ninguna de ellas utiliza inteligencia artificial para obtener la distancia interpupilar como lo propone esta investigación.

Otro aspecto relevante de la investigación es la existencia de un sistema experto para el diagnóstico de enfermedades en los ojos utilizando el método de encadenamiento, este sistema fue realizado por la Universidad Negeri Manado en Indonesia, en la cual el lenguaje de programación utilizado fue PHP y MySQL como base de datos. La misma, trata de desarrollar un sistema experto que utiliza inteligencia artificial, con un alcance definido el cual es trabajar con 16 tipos de enfermedades y con 41 tipos de síntomas diferentes, el cual

es capaz de aprender e imitar el comportamiento humano, en este caso el de los doctores, con la finalidad de solventar el problema de oferta y demanda en el sentido de que es muy superior el número de pacientes en contra del número de doctores. (Munaiseche et al, 2018, p.1)

De acuerdo con la información revisada, no existen aplicaciones móviles públicas disponibles en ninguna de las dos tiendas en línea para dispositivos móviles. Tampoco se logró encontrar alguna investigación que se enfoque específicamente en el cálculo de la distancia interpupilar para dar seguimiento a tratamientos de estrabismo, por lo que se torna importante y pionera esta investigación.

El desarrollo de software propuesto en versión 1.0 consiste en un software con arquitectura de tipo cliente-servidor, siendo el servidor quién realiza los cálculos complejos y el cliente quien captura y despliega los datos al usuario final.

Esta solución para el cálculo objetivo de la distancia interpupilar mediante el uso de inteligencia artificial para el seguimiento de tratamientos de estrabismo, pretende hacer más ágil y exacto el proceso de toma de medidas para el examinador. Con dichos resultados y mayor precisión, el paciente podrá disfrutar de soluciones confiables y personalizadas.

Por otro lado, los métodos y tecnologías no tradicionales que se utilizaron para realizar este proyecto, como lo son el uso de la inteligencia artificial y *machine learning*, para la detección y análisis de imágenes para resolver un problema complejo, se vuelven especialmente efectivos ya que la aplicación de algoritmos ayuda en la obtención de soluciones precisas.

2.2 Marco conceptual

2.2.1 Conceptos referentes a oftalmología

Oftalmología

En el artículo publicado por Ucha (2013) define la oftalmología como "especialidad médica que se ocupa del estudio de los ojos, lo cual incluye el tratamiento de las enfermedades y los defectos típicos que padece la visión, tal es el caso del astigmatismo y estrabismo, entre otros" (parr. 1)

Estrabismo

Según los médicos Serrano y Gaviria en su investigación, Estrabismo y ambliopía conceptos básicos para el médico de atención primaria, el estrabismo significa "la desalienación ocular, ya sea debido a anomalías en la visión binocular o anomalías en el control neuromuscular de la motilidad ocular esto conlleva a que las fóveas de ambos ojos no estén simultáneamente alineadas con el objeto de fijación" (Serrano y Gaviria, 2011).

Eje visual

El eje visual es "una proyección lineal imaginaria que une a la foveola, el ápice corneal y el punto de fijación además indica que su importancia radica en que corresponde a la trayectoria de acceso de los rayos luminosos paraxiales y centrales que estimulan la fóvea" (Guerrero,2017).

Eje óptico

El eje óptico es "una proyección axial imaginaria que se extiende desde la pupila (disco óptico) hasta el ápice corneal y sirve como referente visual para realizar la oftalmoscopia". (Guerrero,2017).

Convergencia

Según Marín y Cinta (2007) el término de convergencia tiene dos significados diferentes, uno describe la posición relativa de los ejes visuales cuando se encuentran en un punto próximo dado de visión, y el otro se refiere a los movimientos relativos de los ejes visuales cuando la fijación cambia desde un punto más lejano a un punto más próximo. (Figura 1)

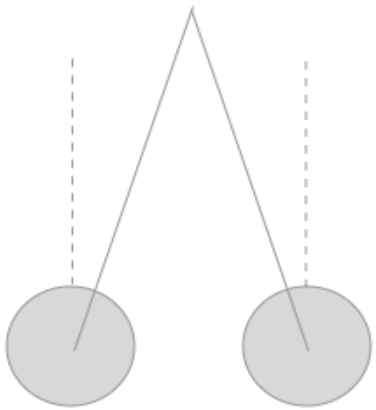


Figura 1. Convergencia desde un objeto lejano hasta un objeto más próximo.

Fuente: Elaboración propia.

Distancia Interpupilar

La distancia interpupilar se define como la distancia existente entre la pupila del ojo derecho y la del ojo izquierdo a una determinada distancia de visión, o bien la distancia entre los centros pupilares de los ojos. Además, indican que es una medida útil ya que con ella se pueden detectar malformaciones craneofaciales entre otras cosas (Díaz y Moreno, 2018).

Fóvea

Para definir dicho concepto es necesario hacer referencia al Gran diccionario de la lengua española, donde se menciona que dicha parte que compone el ojo se define como

“pequeña depresión de la retina que constituye el punto de máxima nitidez visual” (RAE, 2016)

Motilidad ocular

La motilidad ocular es “el movimiento automático, espontáneo y coordinado de los ojos que posibilita al cerebro componer una imagen tridimensional perfecta. Para realizar correctamente estos movimientos es preciso que los músculos de los ojos implicados tengan un funcionamiento óptimo” (Clínica Baviera,2018, parr. 1).

2.2.2 Herramientas tecnológicas

Aplicaciones móviles

El desarrollo de aplicaciones móviles es similar a otras plataformas, adicionalmente requieren atención aquellos elementos que no están presentes en los modelos tradicionales de desarrollo.

Las aplicaciones móviles son instaladas en los sistemas operativos de los dispositivos inteligentes, similar a como se hacen en las computadoras, pero con la diferencia que la aplicación tiene acceso sobre todo el hardware del dispositivo lo cual no se puede realizar con una computadora personal. Entre otras diferencias se pueden citar el contar con una pantalla más pequeña y el tener una batería limitada para el uso diario (Iversen y Eierman, 2013).

Las tecnologías móviles están recibiendo una atención significativa por parte de los mundos de los negocios y la tecnología de la información. La tecnología representa un cambio dramático para aquellos que pueden sacar provecho económico de ella. Las aplicaciones móviles son la base de la innovación del cómo llegar a los clientes, lo cual

permitió un rediseño en el proceso de los negocios de los productos de software (Iversen y Eierman, 2013).

Las aplicaciones para Android se desarrollan en un ambiente Linux para dispositivos móviles, mediante la utilización de un kit de desarrollo de software o SDK por sus siglas en inglés, el cual es un proyecto de gratuito y de código abierto que le provee al desarrollador todo un ecosistema para crear aplicaciones robustas, seguras y optimizadas para los teléfonos inteligentes o tabletas (Cinar,2012). Mientras en el caso de las aplicaciones móviles para dispositivos iOS estas se popularizaron con la aparición del iPhone y iPad, estos programas se desarrollan utilizando el lenguaje de programación llamado Objetive-C el cual fue creado por Apple. (Neuburg,2013). La liberación de los nuevos entornos de desarrollo para IOS y Mac se realizaron en la conferencia mundial de desarrolladores de Apple, en inglés Apple Worldwide Developers Conference Neuburg,2013.

Inteligencia artificial

La inteligencia artificial es uno de los ejes principales de esta investigación, se vuelve necesario desarrollar dicho concepto tomando en cuenta que es un insumo con el cual se desea desarrollar el proyecto.

La inteligencia artificial se define como: "una ciencia que tiene como objetivo el diseño y construcción de máquinas capaces de imitar el comportamiento inteligente de las personas" (Álvarez,2002 p.95).

En ese mismo sentido, Russell y Norvig definen la inteligencia artificial bajo un comportamiento humano, el enfoque de la prueba de Turing de la siguiente manera “La prueba de Turing propuesta por Alan Turing (1950), se diseñó para proporcionar una definición operacional y satisfactoria de inteligencia." (2004, p.3).

Los mismos autores de dicho libro indican que "un computador debería poseer las siguientes capacidades:

- Procesamiento de lenguaje natural
- Representación del conocimiento
- Razonamiento automático
- Aprendizaje automático
- Visión computacional
- Robótica "(Russell y Norvig, 2004, p.3)

Machine Learning

Ruiz (2018) define machine learning como "una rama de la inteligencia artificial encargada de crear programas de software capaces de generalizar comportamientos a partir de los datos recibidos"(p. 5). También indica que este no es un concepto nuevo pero que en los últimos años ha tenido un auge exponencial en su uso y aplicación, esto debido al aumento en la capacidad de procesamiento de las computadoras modernas y su bajo coste.

Modelos de machine learning

Es importante mencionar la relevancia de los modelos de entrenamiento de machine learning para la investigación, porque las computadoras son capaces de "aprender" mediante el uso de modelos de entrenamiento a los cuales se les ingesta grandes cantidades de información, y con esta es posible predecir comportamientos futuros.

Para definir dicho concepto el ingeniero en machine learning, Bhattacharjee (2017) menciona que son "un modelo de machine learning que puede ser una representación matemática de un proceso del mundo real. Para generar un modelo de machine learning es

necesario proveer datos de entrenamiento a un algoritmo de machine learning del cual aprender" (p.1).

Framework

Según Saavedra (2009) el concepto de framework se emplea en muchos ámbitos en el desarrollo de sistemas de software, se pueden encontrar frameworks para el desarrollo de aplicaciones médicas, de visión por computador entra otros. Adicionalmente, Gutiérrez (2018) define este concepto como "una estructura de software compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación". En otras palabras, un framework se puede considerar como una aplicación genérica incompleta y configurable a la que podemos añadirle las últimas piezas para construir una aplicación concreta.

Por otro lado, los objetivos principales que intenta conseguir un framework son "acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones" (Saavedra, 2009).

Biblioteca de programación

Según Jones (2008) "las bibliotecas fueron diseñadas para agrupar funcionalidades similares en una sola unidad. Estas unidades pueden ser compartidas con otros desarrolladores de software y esto permite lograr una programación modular" (p.1).

Lenguaje de programación

Rodríguez (2003) define un lenguaje de programación como el lenguaje artificial que se utiliza para expresar programas en un ordenador. También indica, que cada ordenador, según su arquitectura, "entiende" cierto tipo de instrucciones elementales (lenguaje de máquina). Como consecuencia, para facilitar el trabajo de los programadores existen también lenguajes de alto nivel, los cuales no dependen de la arquitectura del ordenador.

2.2.3 Metodología de desarrollo

La metodología de desarrollo de software en cascada es denominada así por la posición de las fases en el desarrollo de esta, que parecen caer en cascada "por gravedad" hacia las siguientes fases. Este enfoque metodológico que ordena rigurosamente las etapas del proceso para el desarrollo de software, de tal forma que el inicio de cada etapa debe esperar a la finalización de la etapa anterior (Maida y Pacienzia, 2015, p.39).

Según Maida y Pacienzia (2015) el ciclo de vida del software en cascada es el siguiente:

- Ingeniería y Análisis del Sistema: Debido que el software es siempre parte de un sistema mayor, el trabajo comienza estableciendo los requisitos de todos los elementos del sistema y luego asignando algún subconjunto de estos requisitos al software.

- Análisis de los requisitos del software: El proceso de recopilación de los requisitos se centra e intensifica especialmente en el software. El ingeniero de software (Analistas) debe comprender el ámbito de la información del software, así como la función, el rendimiento y las interfaces requeridas.

- Diseño: El diseño del software se enfoca en cuatro atributos distintos del programa: la estructura de los datos, la arquitectura del software, el detalle procedimental y la caracterización de la interfaz. El proceso de diseño traduce los requisitos en una representación del software con la calidad requerida antes de que comience la codificación.

- Codificación: El diseño debe traducirse en una forma legible para la máquina. El paso de codificación realiza esta tarea. Si el diseño se realiza de una manera detallada la codificación puede realizarse mecánicamente.

- Prueba: Una vez que se ha generado el código comienza la prueba del programa. La prueba se centra en la lógica interna del software, y en las funciones externas, realizando pruebas que aseguren que la entrada definida produce los resultados que realmente se requieren.

En la siguiente figura se ejemplifica más ampliamente el ciclo de vida de la metodología de desarrollo de software en cascada comparándola con otras.

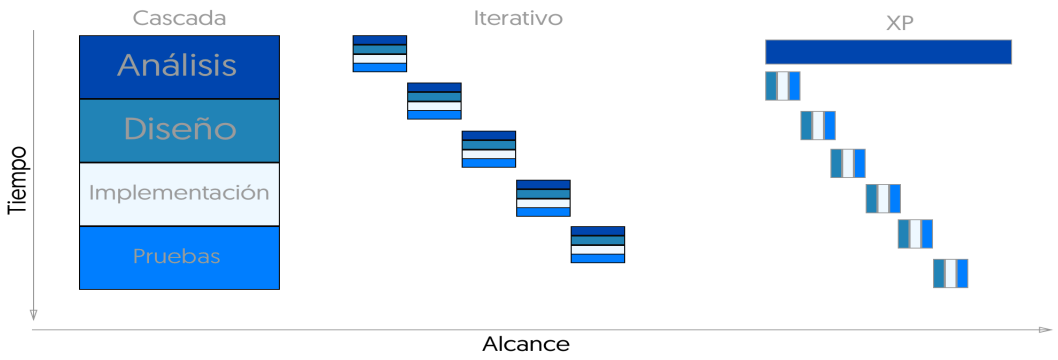


Figura 2 . Comparación de ciclos de vida de metodologías de software ágiles.

Fuente: Elaboración propia

CAPÍTULO III
METODOLOGÍA

CAPÍTULO III: METODOLOGÍA

3.1 Proceso de investigación

Para el análisis, diseño, desarrollo e implementación del sistema de cálculo objetivo de la distancia interpupilar mediante el uso de inteligencia artificial para el seguimiento de tratamientos de estrabismo en la Universidad Nacional, se ha seleccionado la metodología de trabajo de cascada.

Es esencial destacar, que tanto el proceso de investigación como el proceso de desarrollo, fue realizado exclusivamente por una persona, por lo cual es efectiva la aplicación de esta metodología ya que, para iniciar una nueva fase del desarrollo, se debe finalizar con la fase anterior y esto favorece la fluidez y la entrega de los objetivos planteados.

Debido a la naturaleza de este proyecto, la cual comprende el análisis de imágenes y cálculos matemáticos para la obtención de la distancia Interpupilar y el ángulo de convergencia en pacientes estrábicos, se utilizó la inteligencia artificial desde dispositivos móviles iOS y Android. A su vez, se optó por desarrollar una parte del sistema en las plataformas móviles antes mencionadas utilizando tecnologías como JavaScript con el framework react native. La otra parte del proyecto comprende la plataforma web, encargada del análisis y procesamiento de imágenes utilizando el lenguaje de programación Python[1], el cual por sus características particulares facilita el desarrollo de este tipo de proyectos.

A lo largo de los planteamientos hechos, el reconocimiento de imágenes no es una tarea complicada para los seres humanos, somos capaces de reconocer fácilmente la diferencia entre un león y un jaguar, un hombre y una mujer, entre otros, esto gracias a que nuestro cerebro hace parecer que esta tarea no es complicada. Sin embargo, para las máquinas esto es un gran reto, en los últimos años el campo del machine learning en el reconocimiento y análisis de imágenesha tenido un avance, es por eso que para este proyecto

[1] Python es un lenguaje de programación multiparadigma (orientación a objetos, imperativo y funcional).

en particular es necesario incluir bibliotecas de desarrollo de software de código abierto como OpenCV. Esta biblioteca permite mediante el entrenamiento de un modelo de machine learning para facilitar a las máquinas el reconocimiento de ciertas imágenes o comportamientos establecidos en el mismo (Russell y Norvig, 2014).

3.2 Descripción de las tecnologías para el desarrollo del prototipo

Python

Para el desarrollo de software del proyecto se utilizó el lenguaje de programación Python en su versión 3.6.4, por las cualidades que tiene que son especialmente favorables para proyectos que involucran específicamente el análisis de imágenes.

La historia del lenguaje de programación Python inicia con Guido Van Rossum quien ideó el lenguaje a finales de los años 80 y comenzó a implementarlo en diciembre del año 1989 (Marco, 2018).En febrero de 1991 publica la primera versión 0.9.0. Luego vino la versión 1.0 que se publicó en enero de 1994, y seis años más tarde se publica la versión 2.0 en octubre de 2000. La versión actual de dicho lenguaje, la cual es la 3.0, se publicó en diciembre de 2008. El desarrollo de Python lo lleva a cabo un colectivo de programadores que trabaja en conjunto con la fundación Python Software Foundation. Es importante destacar que Guido Van Rossum dirigió el desarrollo de Python hasta Julio del 2018.

El proyecto fue realizado en la versión 3.6.4, es importante aludir como se identifican las versiones de Python por tres números X.Y.Z, en la que Bartolomé Sintes Marco indica:

- X corresponde a las grandes versiones de Python (1, 2 y 3), incompatibles entre sí: Los principales cambios introducidos en Python 2 fueron las cadenas Unicode, las comprensiones de listas, las asignaciones aumentadas, los nuevos métodos de cadenas y el recolector de basura para referencias cíclicas.

Los principales cambios introducidos en Python 3 fueron la separación entre cadenas Unicode y datos binarios, la función print(), cambios en la sintaxis, tipos de datos, comparadores, etc. Por el momento, no hay planes de crear una nueva versión Python 4, incompatible con las anteriores.

- Y corresponde a versiones importantes en las que se introducen novedades en el lenguaje, pero manteniendo la compatibilidad (salvo excepciones).

  Desde hace unos años, las versiones X.Y se publican aproximadamente cada año y medio y se mantienen durante cinco años, excepto la versión 2.7, que se mantendrá por lo menos durante diez años, hasta 2020.
- Z corresponde a versiones menores que se publican durante el período de mantenimiento, en las que sólo se corrigen errores y fallos de seguridad.

  Normalmente, se publica una última versión X.Y.Z justo antes de que una versión X.Y deje de mantenerse. Algunas empresas comerciales ofrecen el mantenimiento de versiones antiguas una vez acabado el mantenimiento oficial. (Marco, 2018).

OpenCV

En este punto es importante destacar que el núcleo de este proyecto está basado en esta biblioteca de código abierto por sus cualidades en la detección y análisis de imágenes. OpenCV, fue desarrollado originalmente por la visión de Intel. Es importante subrayar que es gratuito para uso comercial y la investigación bajo una licencia BSD. Además, esta biblioteca es multiplataforma y funciona en los sistemas operativos Mac OSX, Windows y Linux. La misma se centra principalmente hacia procesamiento de imagen en tiempo real.

Algunas de las funciones básicas de OpenCV son:

1. Captura en tiempo real.
2. Importación de archivos de vídeo.
3. El tratamiento básico de imágenes (brillo, contraste, umbral.)
4. Detección de objetos (cara, cuerpo).

En su documentación, OpenCV facilita muchos de los ejemplos y funcionalidades utilizados en este proyecto, como la utilización de reconocimiento facial mediante el uso de HaarCascades entre otras. Estos ejemplos pueden ser consultados en el siguiente sitio web: https://docs.opencv.org/master/examples.html

Node.js

Node.js fue lanzado inicialmente solo para el sistema operativo Linux en el 2009. El gestor de paquetes (NPM) fue lanzado en el 2010, sin embargo, el soporte nativo para Windows fue lanzado en el 2012.

Por otro lado, Node.js es un entorno que trabaja en tiempo de ejecución, es de código abierto y además es multiplataforma, de esta manera permite crear a los desarrolladores toda clase de herramientas del lado del servidor en el lenguaje de programación Javascript. La ejecución en tiempo real está pensada para usarse fuera del contexto de un explorador web.

Algunas de las ventajas que posee Node.js son las siguientes:

- Rendimiento: Node.js ha sido diseñado para optimizar el rendimiento y la escalabilidad en aplicaciones web.
- Simpleza: El código está escrito en simple Javascript lo que significa que los desarrolladores pierden menos tiempo ocupándose de la complejidad que posee otros lenguajes de programación utilizados del lado del servidor.
- Gestor de paquetes de Node.js (NPM): Proporciona a miles de paquetes reutilizables y de código libre en muchos de los casos.
- Portabilidad: Al ser multiplataforma brinda una gran versatilidad y portabilidad entre los diferentes sistemas operativos.

React Native

Primeramente, React native es un framework o marco de trabajo que permite a los desarrolladores de software implementar aplicaciones móviles nativas utilizando el lenguaje de programación Javascript. El mismo fue lanzado en marzo del 2015 por Facebook.

Seguidamente, debido a las múltiples ventajas que brinda React Native en el desarrollo de aplicaciones móviles, como por ejemplo reducir a la mitad el tiempo de desarrollo de una aplicación nativa para los sistemas operativos móviles como lo son iOS y Android debido a que se realiza un solo desarrollo de software utilizando el lenguaje de programación Javascript y se obtienen dos ejecutables, para desplegar la aplicación en los sistemas operativos móviles antes mencionados fue seleccionado como framework de desarrollo para la captura y despliegue de resultados al usuario final (Escacena,2018).

Git

Git forma parte de esta sección ya que todo el código fuente generado en este proyecto está versionado, lo que significa que es posible dar trazabilidad de todos los cambios realizados desde las primeras fases de desarrollo.

Es básicamente un software de control de versiones diseñado por Linus Trolvalds y a su vez un control de versiones se define como la gestión de los diversos cambios que se realizan sobre los elementos de algún producto o una configuración de este. Fue creado pensado en la eficiencia y confiabilidad del mantenimiento de versiones de aplicaciones cuando estas tienen un gran número de archivos de código fuente, los cuales deben ser trabajados por uno o más desarrolladores. Facilita de manera efectiva la forma de distribuir, gestionar y asignar el trabajo a los diferentes miembros del equipo.

Algunas de las características más importantes de Git son:

1. Rapidez en la gestión de ramas, debido a que Git nos dice que un cambio será fusionado mucho más frecuentemente de lo que se escribe originalmente.
2. Gestión distribuida; Los cambios se importan como ramas adicionales y pueden ser fusionados de la misma manera como se hace en la rama local.
3. Gestión eficiente de proyectos grandes.
4. Realmacenamiento periódico en paquetes.

Django

Django es un framework o marco de trabajo de aplicaciones web gratuito y de código abierto escrito en Python. Al ser Django un entorno web permite crear de manera rápida y eficientes servicios web para la comunicación entre el servidor y el cliente.

Fue desarrollado inicialmente entre 2003 y 2005 por un equipo que era responsable de crear y mantener sitios web de periódicos. Después de crear varios sitios, el equipo empezó a tener en cuenta y reutilizar muchos códigos y patrones de diseño comunes. Este código común se convirtió en un framework web genérico, que fue de código abierto, conocido como proyecto "Django" en julio de 2005.

El mismo ha continuado creciendo y mejorando desde su primer hito, el lanzamiento de la versión (1.0) en septiembre de 2008, hasta el reciente lanzamiento de la versión 1.11 (2017). Cada lanzamiento ha añadido nuevas funcionalidades y solución de errores, que van desde soporte de nuevos tipos de bases de datos, motores de plantillas, caching, hasta la adición de funciones genéricas y clases de visualización (que reducen la cantidad de código que los desarrolladores tiene que escribir para numerosas tareas de programación).

3.3 Selección de la muestra

Los participantes de la investigación, es decir, las personas para la cual va dirigido el proyecto son especialistas oftalmólogos encargados de calcular la distancia Interpupilar para el debido seguimiento de tratamientos de estrabismo. Esta investigación es acompañada por el doctor Optometrista Miguel Zarama.

Además, para este proyecto se contó con el apoyo de la empresa patrocinadora Ártico Soluciones es una pyme dedicada al desarrollo de software en Costa Rica con capital y recursos completamente costarricenses, creada a finales del año 2016 y establecida como sociedad anónima a inicios del 2017 la cual utiliza tecnologías disruptivas para la creación de sus productos comerciales. Esta empresa toma papel de patrocinador en la implementación de la solución de software a desarrollar.

Parte de la historia de Ártico Soluciones S.A. es importante destacar que fue fundada a finales del año 2016 por un grupo de tres talentosos ingenieros en sistemas de información con muchos años de experiencia en el campo de tecnologías de la información en Costa Rica. Desde ese entonces, se han gestionado todos los trámites legales que competen para figurar de manera legal en el país. Por otro lado, esta empresa actualmente desarrolla diferentes productos y servicios en el mercado costarricense.

El objetivo general de Ártico Soluciones S.A. es brindar al mercado empresarial local e internacional soluciones tecnológicas de calidad que satisfagan sus necesidades de manera simple y eficaz, buscando siempre la mejora continua, cumpliendo con su responsabilidad social y ambiental.

La visión de la empresa se orienta en ser una empresa líder y responsable que brinde al mercado local e internacional soluciones tecnológicas de punta y de calidad. Mientras su misión es brindar a nuestros clientes soluciones tecnológicas de calidad que satisfagan sus

necesidades de manera simple y eficaz, buscando siempre la mejora continua, cumpliendo con nuestra responsabilidad social y ambiental. Soluciones brindadas por colaboradores de un alto perfil profesional y con un desarrollo humano integral.

CAPÍTULO IV
PROPUESTA DE SOLUCIÓN

CAPÍTULO IV: PROPUESTA DE SOLUCIÓN

4.1 Diagnóstico

El diagnóstico y tratamiento de estrabismo se debe realizar desde edades tempranas para que en caso de que se presente pueda ser tratado a la brevedad. Por ejemplo, en un recién nacido los movimientos oculares son irregulares e incoordinados como lo menciona Pérez (2011) en su documento Diagnóstico y tratamiento de estrabismo "entre la segunda y tercera semana de vida, empiezan a coordinar los movimientos de la cabeza y los ojos y aparece el reflejo de fijación. A los tres meses, siguen un objeto con movimientos aún algo erráticos y a los seis meses son capaces de mantener un paralelismo ocular si la maduración cerebral es normal para su edad. La agudeza visual va incrementándose a lo largo del desarrollo hasta los cinco o seis años, en que alcanza la unidad de visión" (p.1).

A su vez, en pacientes tanto adultos como niños, si tienen un ojo alineado y el otro desviado en cualquier dirección, el objeto estimulará por una parte la fóvea del ojo alineado, y por otra, el área extrafoveal del ojo estrábico, creando a nivel cortical una imagen de confusión o de diplopía.

- **Diplopía**: En la siguiente imagen se muestra cómo en el ojo desviado no se proyecta la fóvea y por tanto la calidad de la imagen es mala. Lo que significa que en este caso el cerebro no elimina la imagen y por ello tienden a ver una sombra como si fuese una imagen doble.

- **Confusión**: En este caso la fóvea del ojo desviado no tiene la misma imagen que la fóvea del ojo fijador, lo que significa que en este caso el cerebro no la elimina y se superponen imágenes distintas, en el caso de la Figura 3 se ejemplifica como la estrella sobre el círculo.

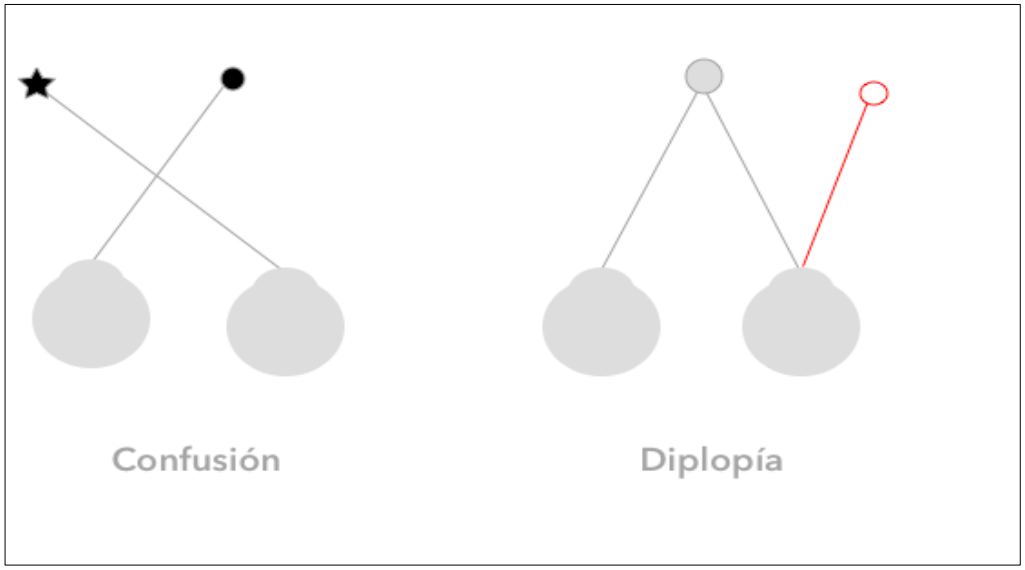


Figura 3. Comparación de confusión y diplopía.

Fuente: Elaboración propia

Tipos de estrabismo

Existen múltiples clasificaciones de estrabismo dependiendo de dos parámetros:

1. Según la forma de manifestarse:
    a. Estrabismos latentes que sólo se manifiestan al hacer el Cover test.
    b. Estrabismos manifiestos.

2. Según el sentido de la desviación (Ver Figura 4):

I. Horizontales
    a. Exotropia.
    b. Endotropia.

II. Verticales:
    a. Hipertropia.

b. Hipotropia

III. Torsionales:

a. Inciclotropia

b. Exciclotropia

3. Según la etimología:

a. No paralíticas.

b. Endotropia.

c. Exotropia.

d. Estrabismos verticales.

e. Paralíticas.

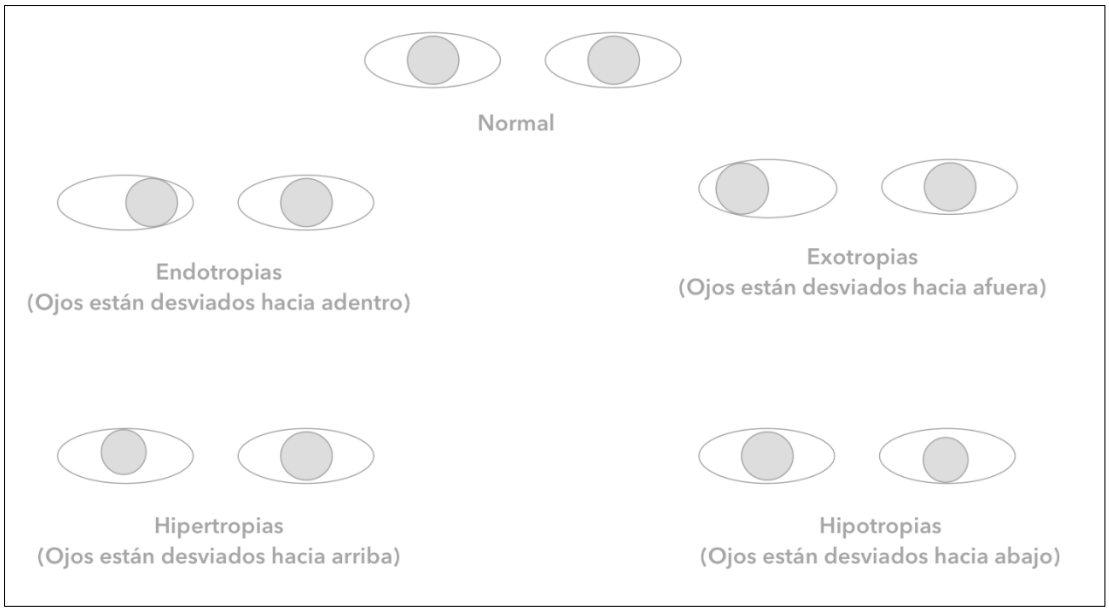


Figura 4 Clasificación de tipos más comunes de estrabismo.

Fuente: Elaboración propia

Diagnóstico

Cuando se sospecha un estrabismo hay una serie acciones que se deben realizar, por ejemplo:

1. Realizar una buena anamnesis esto se refiere a un conjunto de datos que se recogen en la historia clínica de un paciente con un objetivo diagnóstico, como lo son antecedentes familiares de estrabismo, antecedentes personales como problemas en el embarazo o en el parto, la edad de aparición, la forma de aparición (brusca o intermitente) entre otras.

2. Inspección: En este caso se puede mencionar elementos muy puntuales como desviación en los ojos (endotropia, exotropia, hipertropia o hipotropia) o posiciones anómalas de la cabeza (tortículis compesadoras por estrabismo.

3. Agudeza visual

4. Motilidad ocular

Se hizo especial énfasis en el punto 4 de motilidad ocular, ya que este proyecto trata de automatizar algunos de los exámenes o diagnóstico que aquí se detallarán mediante el uso de aprendizaje de máquina o machine learning.

Algunas de las acciones que se realizan para diagnosticar la motilidad ocular son las siguientes:

I. Deducciones, versiones y vergencias

II. Estudio de las nueve posiciones de la mirada para evaluar la acción de cada musculo.

III. Test de Hirschberg:

Consiste en la proyección de una luz sobre cada ojo y observar donde cae el reflejo luminoso con respecto al centro de la pupila.

- Si no existe un estrabismo, el reflejo cae en el centro de la pupila de cada ojo.
- Si en un ojo cae en el centro y en el otro, en el borde pupilar, se dice que hay un estrabismo de 15 grados.
- Si el reflejo cae entre el borde pupilar y el limbo, el estrabismo es de 25 a 30 grados.
- Si el reflejo cae a nivel límbico será de 45 grados.
- Si estos reflejos se proyectan por dentro de la pupila, estamos ante una exotropia. - Si se proyectan por fuera, el paciente presenta una endotropia. (Pérez, 2011, p.2).

Sí alguna de las situaciones mencionadas anteriormente se presenta en el test de Hirschberg en un paciente, se obtiene como resultado el tipo de estrabismo que podría presentar el mismo.

En la Figura 5, se observa el ejemplo de como se ve el reflejo de Hirschberg, señalado como un punto blanco dentro de la figura. En el punto A se observa una persona con una mirada normal y en los otros puntos (B, C, D) el diagnóstico de algún tipo estrabismo.

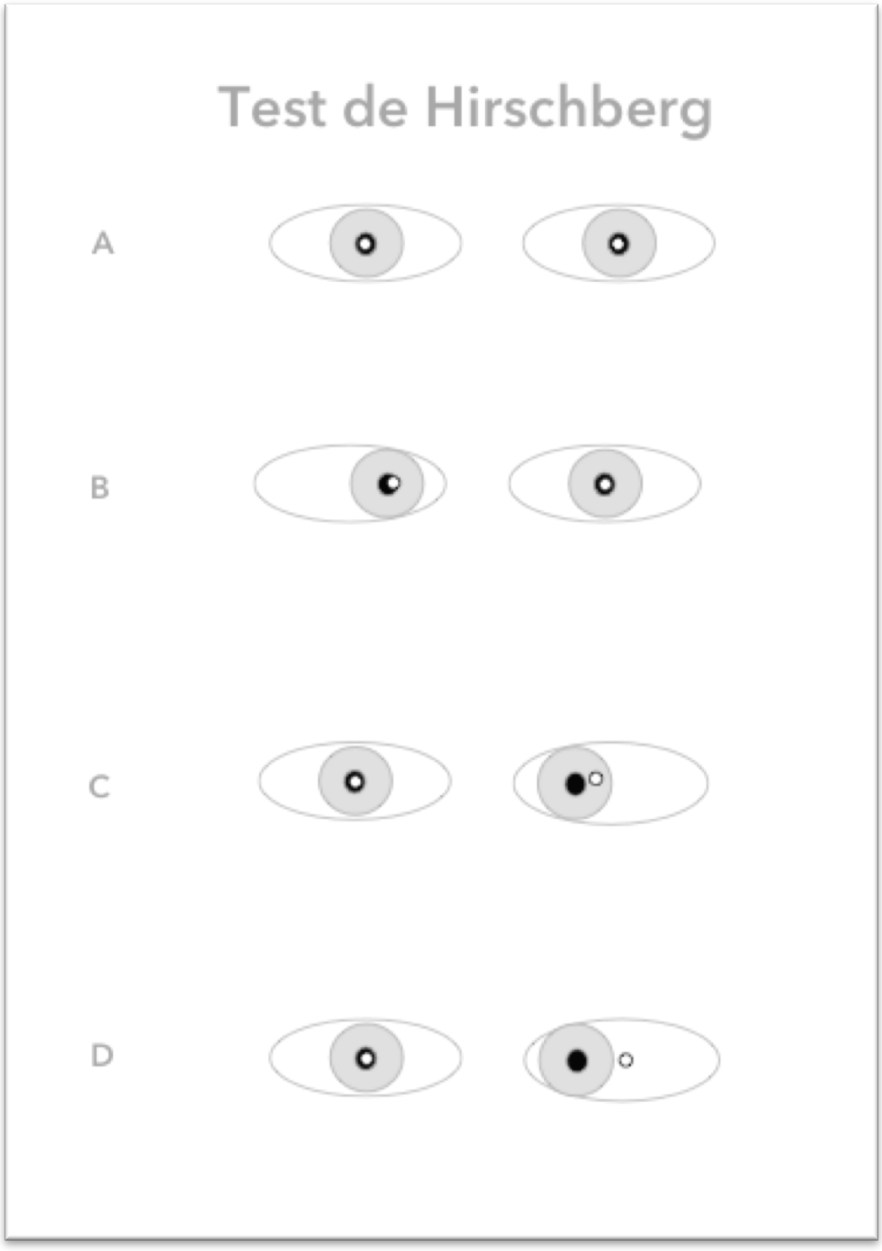


Figura 5 Test de Hirschberg.

Fuente: Elaboración propia.

Cover test

Otra práctica común, y a su vez fundamental en el diagnóstico de estrabismo, es el cover test o test de la pantalla, seguidamente se describe brevemente en que consiste este proceso.

Se basa en los movimientos de fijación que hace el ojo desviado al ocluir el ojo fijador. El procedimiento se realiza ocluyendo uno de los ojos y haciendo fijar con el otro ojo un objeto, observaremos el comportamiento del ojo destapado. Después repetimos la maniobra tapando el otro ojo y estudiamos el ojo no ocluido.

En las endotropias, el movimiento de fijación se hace desde dentro hacia fuera y en las exotropias, desde fuera hacia dentro; en las hipertropias, desde arriba hacia abajo, y en las hipotropias, desde abajo hacia arriba.

Para diagnosticar las forias, se efectúa el covertest alternante y se estudia el comportamiento del ojo tapado. Ocluyendo alternativamente durante dos o tres segundos cada ojo por separado, se produce un movimiento de refijación al destapar el ojo ocluido. Cuando dejamos de ocluir, se recupera el paralelismo.

También se usa el cover test con barras de prismas para medir los grados de un estrabismo (test de Krimski). El cover test se debe realizar siempre en visión lejana y cercana (si el paciente colabora) ya que hay estrabismos que tienden a ser mayores o a descompensarse más en visión lejana (exotropias) y otros, en visión cercana (incomitancia lejos-cerca) (Pérez, 2011, p.3).

Esta prueba tiene la gran ventaja que no requiere equipos costosos para su ejecución, se puede realizar en niños y adultos, evalúa la acomodación y la convergencia, entre otros beneficios. Sin embargo, una de sus principales desventajas es que debe ser realizado por un profesional con experiencia.

En relación con la segunda parte del primer objetivo específico, a continuación, se indica que basado en los hallazgos previamente mencionados acerca del estrabismo, se establecerán las herramientas y lenguajes de programación adecuados para el procesamiento de imágenes y datos, con la finalidad de cumplir satisfactoriamente con el proceso de desarrollo de software.

Con base en los datos previamente expuestos se decidió utilizar una arquitectura de tipo cliente servidor, según se presenta en la Figura 6.

Figura 6 Arquitectura cliente-servidor

Fuente: Elaboración propia.

Para este desarrollo en específico se seleccionó como lenguaje principal para el cálculo y análisis de imágenes, el lenguaje de programación denominado Python, ya que para este tipo de desarrollos es especialmente práctico, debido a que posee una gran cantidad de bibliotecas dedicadas a la inteligencia artificial y una comunidad grande de programadores que la respalda. También se seleccionó la biblioteca de software libre OpenCV y el framework para entornos web denominado Django. Más adelante se justificará a detalle esta afirmación.

Respecto a la captura de imágenes que se necesita llevar a cabo en el proyecto, el lenguaje de programación seleccionado para realizar dicha tarea fue Javascript con la biblioteca de software React Native.

Detección de rostros utilizando HaarCascades

Este apartado está basado en la documentación de la biblioteca OpenCV, la cual indica que la detección de objetos utilizando clasificadores de HaarCascade es un método efectivo propuesto por Paul Viola y Michael Jones en su investigación Rapid Object Detection using a Boosted Cascade of Simple Features en el año 2001. En este caso está basado en un enfoque de Machine learning donde las funciones en cascada son entrenadas con muchas imágenes positivas y negativas.

En este caso, se indica en la documentación, que se necesitan muchas imágenes de rostros positivas, es decir, imágenes en las que efectivamente exista un rostro e imágenes negativas donde no existan rostros para entrenar al clasificador. Seguidamente se requiere extraer características de dichas imágenes.

De esta forma, es a grandes rasgos cómo funcionan los HaarCascades en OpenCV. A continuación, se muestra ejemplificado en las Figuras 7 y 8 en las cuales se trata de mostrar que por ejemplo en una imagen es escala de grises una sección negra, una blanca y otra negra dentro de un rostro puede representar una nariz.

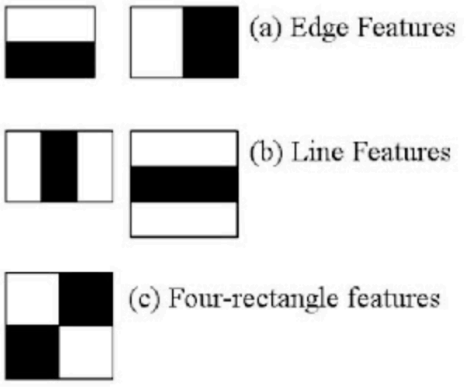


Figura 7 Características del Haar-cascade.

Obtenido de OpenCV.org el 19 de octubre de 2018

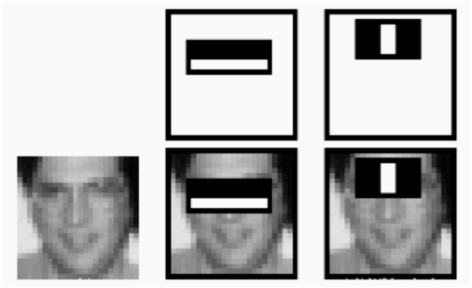


Figura 8 Características del Haar-cascade aplicado a un rostro.

Obtenido de OpenCV.org el 19 de octubre de 2018

Con la aplicación de este algoritmo a una imagen de un rostro, se pueden obtener los siguientes resultados (Ver Figura 9).

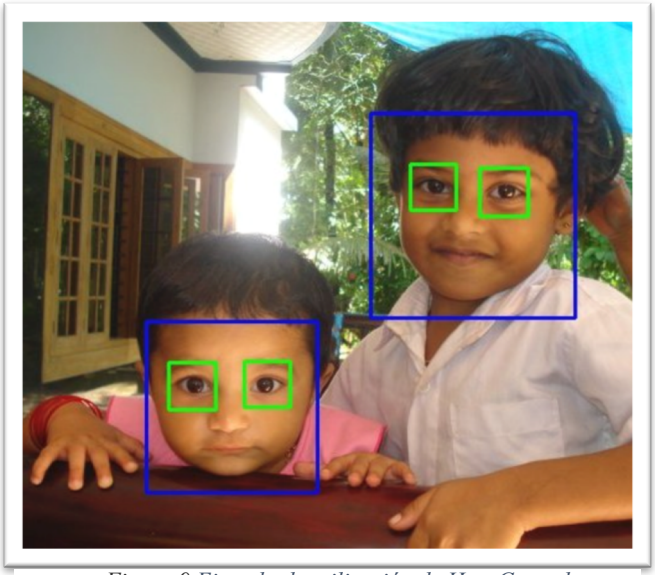


Figura 9 Ejemplo de utilización de HaarCascades en rostros.

Fuente:Obtenido de OpenCV.org el 19 de octubre de 2018

4.2 Propuesta de solución

La aplicación de software que se ejecutó del lado del servidor para la gestión y análisis de imágenes fue desarrollado en Python, en dos versiones. La primera, una versión de escritorio, se utilizó para pruebas, que posteriormente fue migrada a su versión web utilizando el framework Django lo cual la convierte en la segunda versión de la aplicación.

A continuación, se describirán los aspectos más importantes de la versión web, ya que esta es la que se utiliza actualmente para realizar toda la gestión de contenido y análisis de imágenes.

1. Instalación

En esta sección se detalla qué software se debe instalar, y cómo debe ser ejecutado el mismo para poner en funcionamiento esta aplicación, los pasos a seguir también se encuentran en el archivo “léeme” de la aplicación.

(1) Primeramente, se instala Python 3.x.

(a) Para instalar Python lo primero que se debe hacer es contar con una máquina con sistema operativo, Mac Os, Windows o Linux. Seguidamente se debe descargar el software de la siguiente página web https://www.python.org/downloads/ e instalar.

(b) El segundo paso reside en crear un ambiente virtual o virtual env en el cual instalaremos las dependencias del software en un ambiente independiente, el cual no afecte el desempeño del sistema operativo.

(c) Esto lo podemos hacer ejecutando el siguiente comendo, la terminal o línea de comandos: python3 -m nombreDelNuevoEnv /path/to/new/virtual/environment

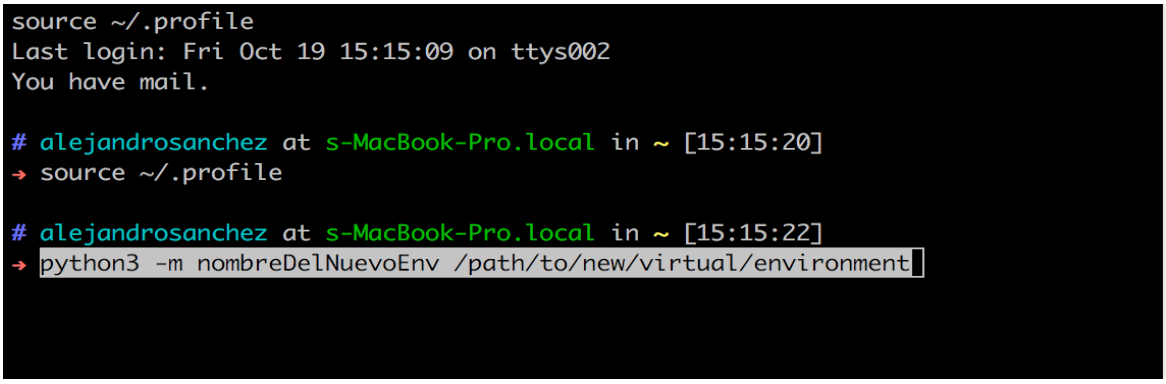


Figura 10 Creación de virtual env en Mac Os.

Fuente: Elaboración propia.

(2) En seguida se debe activar o poner en práctica, el nuevo ambiente mediante la ejecución del siguiente comando en la misma ruta que creamos el ambiente.

(a) Activar el nuevo env [source nombreDelNuevoAnv/bin/activate]

(b) En caso de que se quiera desactivar el ambiente se debe ejecutar el ambiente: source nombreDelNuevoAnv/bin/actívate.

(c) Para instalar los requerimientos de la aplicación utilizamos el gestor de dependencias **pip** de Python ejecutando desde la terminal, posicionados en la ruta donde se encuentra el archivo requirements.txt el siente comando: pip install -r requirements.txt

(3) Posteriormente, todas las imágenes que se procesen serán almacenadas en una base de datos sqllite y se debe ejecutar las migraciones en el proyecto, ejecutando el comando: python3 manage.py runserver

(4) Finalmente, para correr la aplicación se debe ejecutar el siguiente comando: python3 manage.py runserver

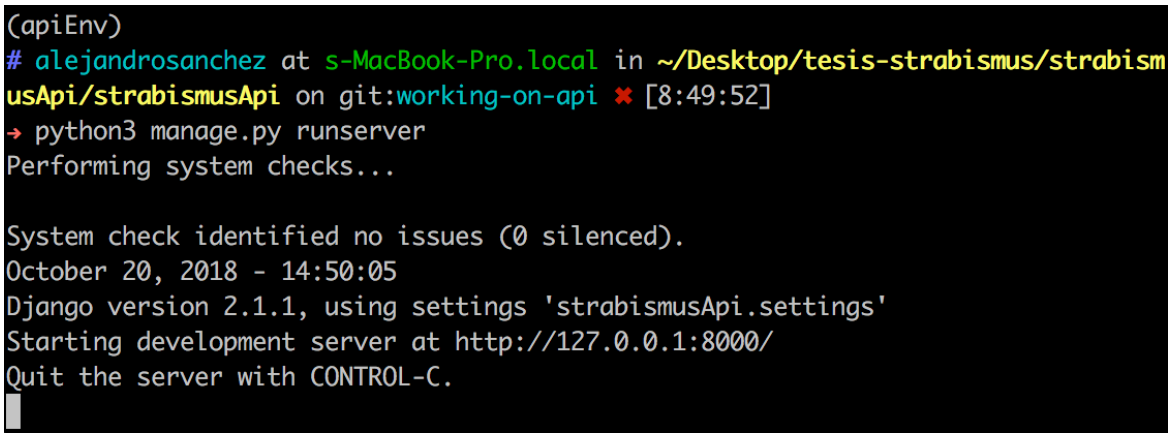


Figura 11. Ejecución de la aplicación Django en Mac Os.

Fuente: Elaboración propia.

Cuando se logra poner en ejecución la aplicación Django, en la terminal, se puede ver que la misma corre de manera predeterminada en el puerto 8000. Django tiene la gran ventaja que de manera automática presenta los servicios web que tenemos disponibles en la aplicación accediendo al sitio: http://localhost:8000/ (Ver Figura 12)

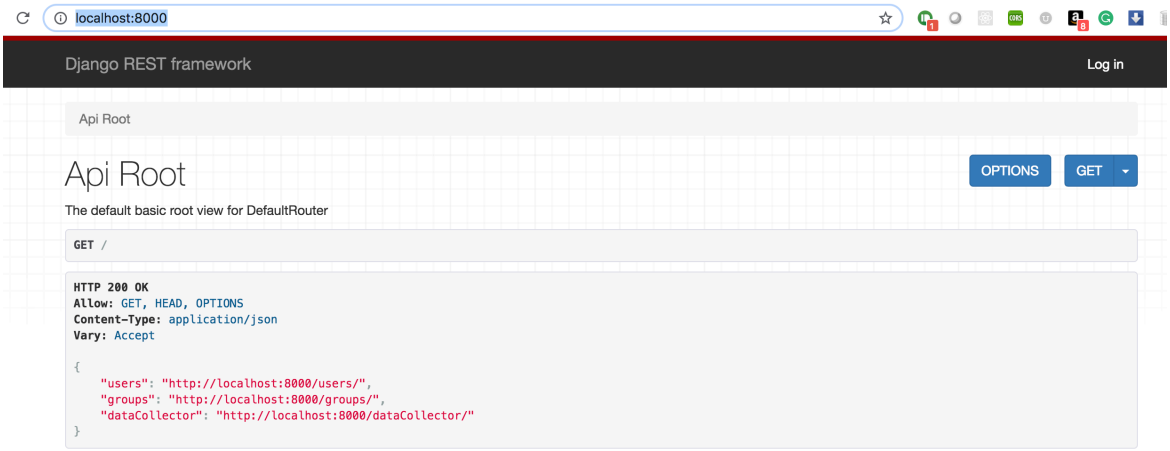


Figura 12. Acceso a la aplicación Django desde un navegador web.

Fuente: Elaboración propia.

En este caso, el primer método web y segundo, son creados por Django de manera automática, el tercero, llamado dataCollector (Ver figura 12) es el encargado de realizar todo el procesamiento y devolver resultados.Para realizar pruebas de este servicio web se puede utilizar herramientas como Postman para ejecutar peticiones al servidor de aplicaciones.

A continuación, se muestra un ejemplo de petición:

1. Tipo: POST.
2. Url: http://localhost:8000/api/strabismusDetector
3. Ejemplo de cuerpo:

{

"email":"test@test.com",

"fullname":"Postman fullname",

"deviceID":"postman1",

"minRadius": 30,

"maxRadius":30,

"image": "/9j/4SZRRXhpZgAATU0AKg"

}

Nota: La imagen debe ser enviada en formato base64.

2. Cálculo de ángulos y distancias para el ojo

En esta sección se detalla el desarrollo del funcionamiento principal del sistema, y como se realiza el cálculo de distancias y se obtiene el ángulo de cada ojo.

ReadImage

Está función es la primera que se utiliza, ya que el servicio web recibe la imagen en formato base64 el cual debe ser transformado en una imagen que pueda ser soportada por OpenCV como se observa en la Figura 13.

```
def readImage(b64Image):
    """Read and return an image."""
    img = base64.b64decode(b64Image);
    npimg = np.fromstring(img, dtype=np.uint8);
    source = cv2.imdecode(npimg, 1)
    return source
```

Figura 13. Función para leer imágenes.

Fuente: Elaboración propia.

GetEyes

Esta función tiene la tarea de obtener las ROI (Region of interest o región de interés) de nuestra imagen, la cual en este caso se podría interpretar como dos sub-imágenes, una para el ojo derecho y otra para el ojo izquierdo mediante la utilización de un clasificador en cascada. Entonces existen dos puntos importantes, ya que OpenCV trabaja buscando grises más claros y más oscuros dentro de una imagen, antes de utilizar la función de HaarCascade se debe transformar la imagen en escala de grises para que pueda ser interpretada por OpenCV. Esta función retorna un arreglo con dos objetos, el primero contiene las dos sub-imágenes con cada ojo, y el segundo contiene la imagen completa transformada en escala de grises (Ver Figura 14).

Figura 14. Función para obtener la sub-imágenes de los ojos.

Fuente: Elaboración propia.

GetLeftPupilCenter y getRightPupilCenter

Esta es una de las funciones principales de la aplicación, es la que se encarga de encontrar la pupila en el ojo de la persona. Existen dos funciones que son similares, una llamada getLeftPupilCenter y la otra getRightPupilCenter, una se encarga del ojo izquierdo y la otra función del ojo derecho, respectivamente.

En la Figura 15 se puede observar que se reciben cinco parámetros:

- Img: Representa la imagen original.
- Eyes: Representa el arreglo con los dos ojos.
- Gray: Representa la imagen en escala de grises.
- MinRadius: el radio mínimo del ojo.
- MaxRadius: el radio mínimo del ojo.

```
def getRightPupilCenter(img, eyes, gray, minRadius, maxRadius):
    ex, ey, ew, eh = eyes[1]
    ## Drawing a rectangle around the eye
    ##cv2.rectangle(img,(ex,ey),(ex+ew,ey+eh),(0,255,0),2)
    roi_gray2 = gray[ey:ey + eh, ex:ex + ew]
    roi_color2 = img[ey:ey + eh, ex:ex + ew]
    circles = cv2.HoughCircles(roi_gray2, cv2.HOUGH_GRADIENT, 1, 200, param1=200, param2=1, minRad
    try:
        for i in circles[0, :]:
            # draw the outer circle
            cv2.circle(roi_color2, (i[0], i[1]), i[2], (0, 255, 0), 1)
            print("drawing circle")
            # draw the center of the circle
            cv2.circle(roi_color2, (i[0], i[1]), 2, (0, 255, 0), 2)
            print("drawing center circle")

            return (i[0], i[1])
    except Exception as e:
        print(e)
```

Figura 15. Función para obtener la posición de la pupila.

Fuente: Elaboración propia.

En esta parte uno de los principales puntos a destacar, es la utilización de la función *HoughCircles* de la biblioteca OpenCV, la cual tiene la capacidad de encontrar circunferencias en una imagen dada o en una región de interés.

En relación con la información anteriormente mencionada, se debe comprender que la función se utiliza para encontrar círculos en las imágenes y en este caso es especialmente efectiva, ya que con el procesamiento previo al cual se ha sometido la imagen, hasta este punto, existen dos sub-imágenes, una para cada ojo. A partir de lo anterior, se quiere encontrar el círculo más grande dentro de la sub-imagen que en este caso representaría la pupila del ojo.

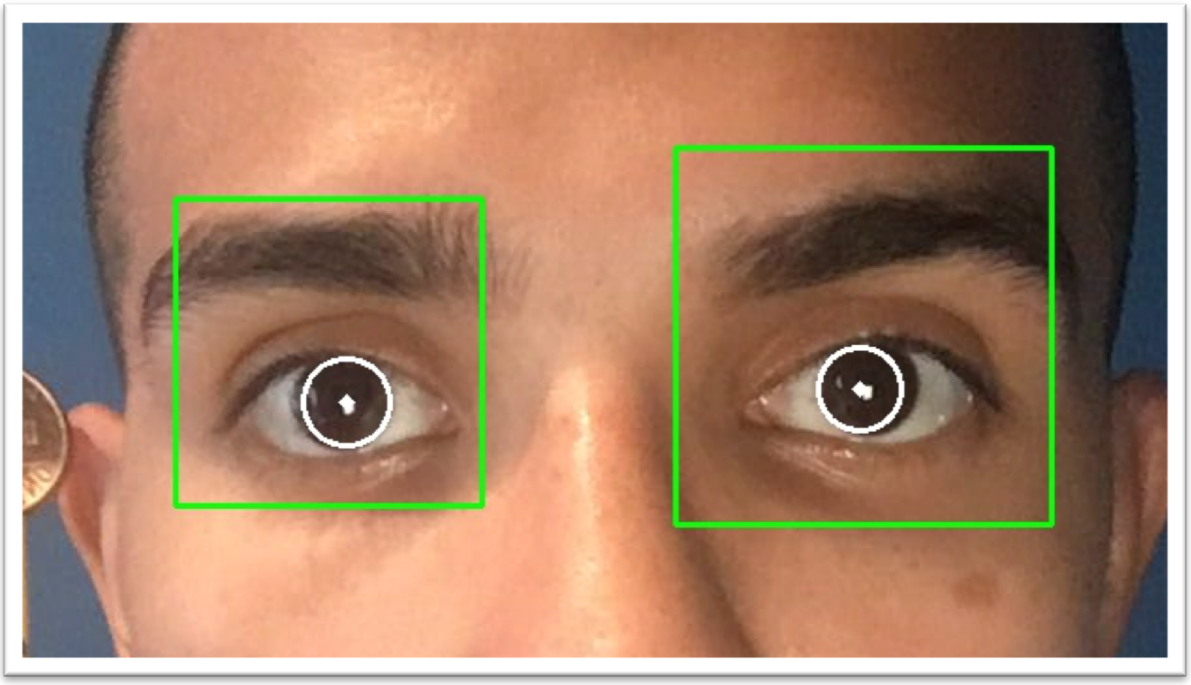


Figura16. Utilización HoughCircles en la región de interés.

Fuente: Elaboración propia.

GetNose

Encontrar la nariz es uno de los tres puntos que se necesitan para obtener el ángulo y la distancia nasopupilar, por lo tanto, estos tres puntos serían:

1. Pupila izquierda.

2. Pupila derecha
3. Tabique.

```python
def getNose(img, gray):
    detector = dlib.get_frontal_face_detector()
    predictor = dlib.shape_predictor(os.path.join(dirname, 'assets/shape_predictor_68_face_landmarks.

    # detect faces in the grayscale image
    rects = detector(gray, 1)
    # loop over the face detections
    for (i, rect) in enumerate(rects):
        # determine the facial landmarks for the face region, then
        # convert the landmark (x, y)-coordinates to a NumPy array
        shape = predictor(gray, rect)
        shape = face_utils.shape_to_np(shape)

        for (name, (i, j)) in face_utils.FACIAL_LANDMARKS_IDXS.items():
            print(name)
            # TODO: Go directly to the right index instead of iterate and use this if statement
            if name == "nose":
                # loop over the subset of facial landmarks, drawing the
                # specific face part
                for (x, y) in shape[i:j]:
                    cv2.circle(img, (x, y), 1, (0, 255, 0), 2)
                    return (x, y)
```

Loading...

Figura 17 Función para encontrar la nariz en la imagen.

Fuente: Elaboración propia.

Para resolver este problema se utiliza la biblioteca dlib, la cual contiene algoritmos complejos de machine learning para resolver problemas del mundo real. En este caso con la utilización de la función shape predictor utilizando un modelo de machine learning pre-entrenado de puntos de referencia que devuelve la posición de las coordenadas de 68 puntos en una imagen correspondientes al rostro (Ver Figura 18)

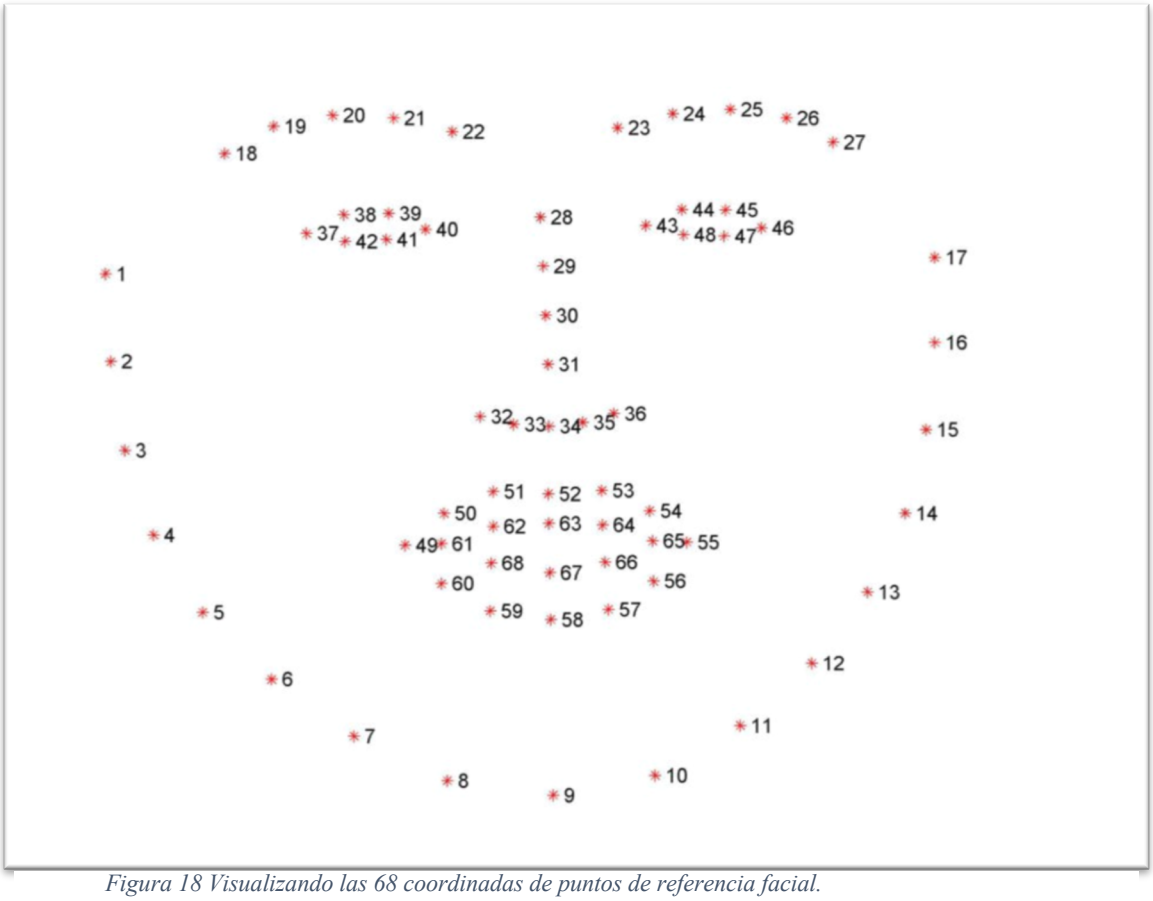


Figura 18 Visualizando las 68 coordinadas de puntos de referencia facial.

Fuente: Tomado de: The iBUG - Facial point annotations.

En el caso de la aplicación, no se necesitan los 68 puntos, si no solo el punto 28 como se marca en la Figura 18. Esto aplicado a una imagen se muestra de la siguiente manera (Ver Figura 19).

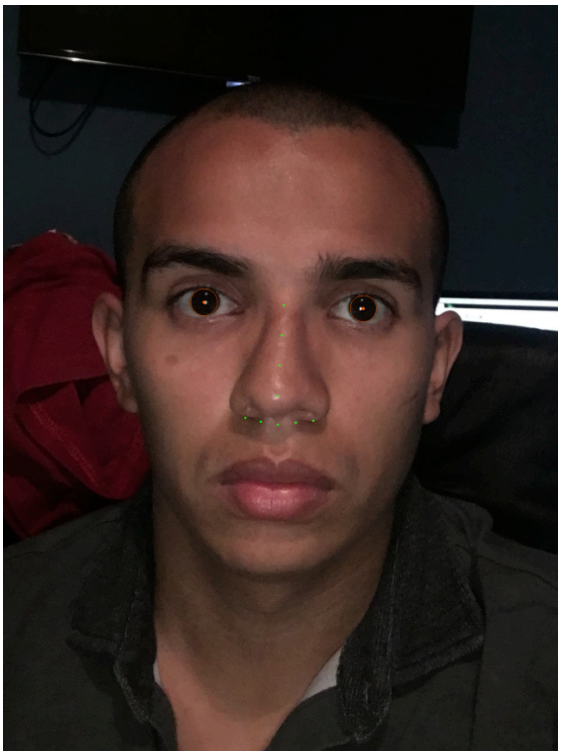


Figura 19. Detectando la nariz con dlib y landmarks.

Fuente: Elaboración propia.

findSquares

Antes de proceder a la explicación acerca de cómo se desempeña esta función, es importante entender el porqué de su existencia.

No obstante, es significativo resaltar que todas las distancias que se han obtenido hasta el momento, se encuentran expresadas en pixeles, lo que significa que no tienen validez en el mundo real ya que la cantidad de pixeles que tiene una distancia, varía según la distancia y la resolución en que se tome la fotografía a la persona.

Artefactos de referencia

Para solventar la conversión de pixeles a centímetros, se inició con la creación de una serie de artefactos que serían tomados como referencia en la fotografía.

Inicialmente, el primer artefacto fue una moneda de un centavo estadounidense al lado izquierdo de la imagen, con la idea de encontrar ese círculo con una función de HoughCircles, sabiendo que la medida del diámetro de un centavo es de 19,05 mm. Sin embargo, a la hora de identificar la moneda, se producían muchos falsos positivos encontrándola (Ver Imagen 21).

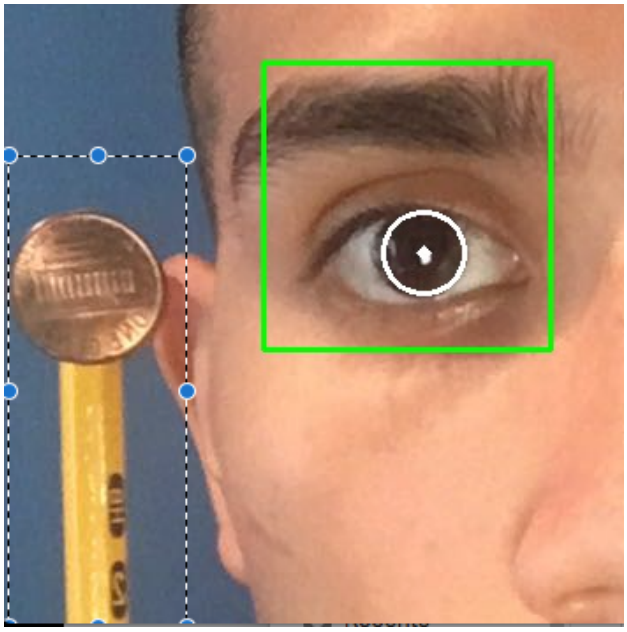


Figura 20 Artefacto de prueba 1, moneda de centavo estadounidense.

Fuente: Elaboración propia.

Luego de este intento fallido, se tomó la decisión de cambiar el objeto de referencia por uno cuadrado, es por eso por lo que se utiliza una función para encontrar cuadrados (Figura 20).

El primer artefacto con esta figura geométrica se elaboró utilizando una paleta de cartón, a la cual se le colocó en un extremo un cuadrado rojo de 2cm de diagonal, y este, al igual que la moneda, en muchas pruebas el sistema no era capaz de detectarlo. Debido a ese inconveniente se tomó la decisión de pintar dicha figura de color blanco, y como consecuencia de dicho cambio de color, la detección mejoró en el sistema (Ver Figura 22).

Sin embargo, este artefacto tiene el gran problema que la medida exacta del cuadrado depende de la posición en la que el usuario o paciente sostenga el artefacto en su mano.

Figura 21 Artefacto de prueba 2, Cuadrado y paleta.

Fuente: Elaboración propia.

Finalmente, se tomó la decisión de utilizar el cuadrado, pero montado sobra un armazón de aros de lentes (Ver Figura 22), con la finalidad de que la posición sea lo más precisa posible para mitigar el porcentaje de error de la medida del cuadrado.

Figura 22 Artefacto de prueba 3, Cuadrado y aros de lentes.

Fuente: Elaboración propia.

Es por esta razón, que para solventar este problema se decidió utilizar objetos de tamaño conocido, como referencia para poder realizar una conversión de pixeles a centímetros de manera confiable. (Ver Figura 20).

```python
def findSquares(img):
    img = cv2.GaussianBlur(img, (5, 5), 0)
    squares = []
    for gray in cv2.split(img):
        for thrs in range(0, 255, 26):
            if thrs == 0:      You, a month ago • Agregando API para consumir el servicio de calculo d
                bin = cv2.Canny(gray, 0, 50, apertureSize=5)
                bin = cv2.dilate(bin, None)
            else:
                _retval, bin = cv2.threshold(gray, thrs, 255, cv2.THRESH_BINARY)
            bin, contours, _hierarchy = cv2.findContours(bin, cv2.RETR_LIST, cv2.CHAIN_APPROX_SIMPLE)
            for cnt in contours:
                cnt_len = cv2.arcLength(cnt, True)
                cnt = cv2.approxPolyDP(cnt, 0.02 * cnt_len, True)
                if len(cnt) == 4 and cv2.contourArea(cnt) > 1000 and cv2.isContourConvex(cnt):
                    cnt = cnt.reshape(-1, 2)
                    max_cos = np.max([angleCos(cnt[i], cnt[(i + 1) % 4], cnt[(i + 2) % 4]) for i in ra
                    if max_cos < 0.1:
                        squares.append(cnt)
    return squares
```

Figura 23 . Función para encontrar cuadrados.

Fuente: Elaboración propia

CalibratePixelPerMetric

Esta función, básicamente lo que hace es llamar a la función findSquares antes mencionada, para encontrar el cuadrado de referencia, seguidamente se marca su diagonal y se obtiene la medida de esa diagonal en pixeles. La diagonal del cuadrado mide 2.2 cm por lo que solo basta con dividir la cantidad de pixeles obtenidos de esa diagonal entre 2.2 cm para obtener la equivalencia de pixeles por centímetros para esa imagen en específico sin importar la resolución de la imagen como se puede observar en la Figura 24.

```python
def calibratePixelPerMetric(img):
    pixelsPerCentimeter = None
    squares = findSquares(img)
    cnt = squares[2]
    cv2.drawContours(img, [cnt], 0, (0, 255, 0), 3)
    smallestContourArea = 0
    smallestContour = 0
    for cnt in squares:
        contourArea = cv2.contourArea(cnt)
        if (smallestContourArea == 0):
            smallestContour = cnt
            smallestContourArea = contourArea
        if (contourArea < smallestContourArea):
            smallestContour = cnt
            smallestContourArea = contourArea
    # This finds the bounding rectangle
    # x,y are the co-ordinates of left-top point and w,h are width and height respectively
    x, y, w, h = cv2.boundingRect(smallestContour)
    cv2.line(img, (x, y), (x + w, y + h), (0, 0, 255), 5)
    font = cv2.FONT_HERSHEY_SIMPLEX
    cv2.putText(img, '2cm', (x + int(w / 2), y + int(h / 2)), font, 1, (0, 0, 0), 1)
    pixelsInReferenceObj = dist.euclidean((x, y), (x + w, y + h))
    ## Reference object real size is: 2cm
    if pixelsPerCentimeter is None:
        pixelsPerCentimeter = pixelsInReferenceObj / 2.2

    print(pixelsInReferenceObj / pixelsPerCentimeter)
    return pixelsPerCentimeter
      You, a month ago • Agregando API para consumir el servicio de calculo de estrabismo.
```

Figura 24. Código fuente de la función calibratePixelPerMetric.

Fuente: Elaboración propia.

4.3 Validación de la propuesta

Pruebas

Inicialmente se debe recalcar que para evitar inconvenientes de cualquier índole (legal, derechos, emocional) las pruebas se realizaron con el investigador. También es importante destacar que se realizaron pruebas en conjunto con el doctor Optometrista Miguel Zarama, quién ha guiado y acompañado la investigación durante este proceso.

Algunas de las pruebas son exitosas y otras tienden a fallar debido a diferentes factores. No hay que olvidar que el *machine learning* está basado en modelos probabilísticos, es por esta razón que en diferentes ocasiones se puede obtener un falso

positivo o un resultado erróneo de esta aplicación, sin embargo, se logró documentar tanto pruebas exitosas como erróneas, las cuales serán expuestas a continuación.

Pruebas erróneas

En esta sección se documentaron algunos de los resultados erróneos o problemáticos, en los cuales diferentes factores afectan el desempeño de la aplicación, como lo son la calidad de la imagen, también se presentan elementos que entorpecen el análisis dentro de los que se encuentran fondos de la imagen con colores o figuras llamativas, entre otros.

En el primer caso, el software confundió el orificio de la nariz con el ojo derecho, esto tiende a suceder cuando el fondo de la imagen es irregular (Ver Figura 25).

Figura 25. Prueba errónea 1

Fuente: Elaboración propia.

En la Figura 26, se trata de resaltar un detalle que tiene la aplicación, este es el ajuste del tamaño de la circunferencia de la pupila, ya que como cada persona tiene diferente las medidas de la pupila, este valor debe ser modificado hasta lograr encontrar el tamaño de la circunferencia que se necesita para ajustarlo al paciente.

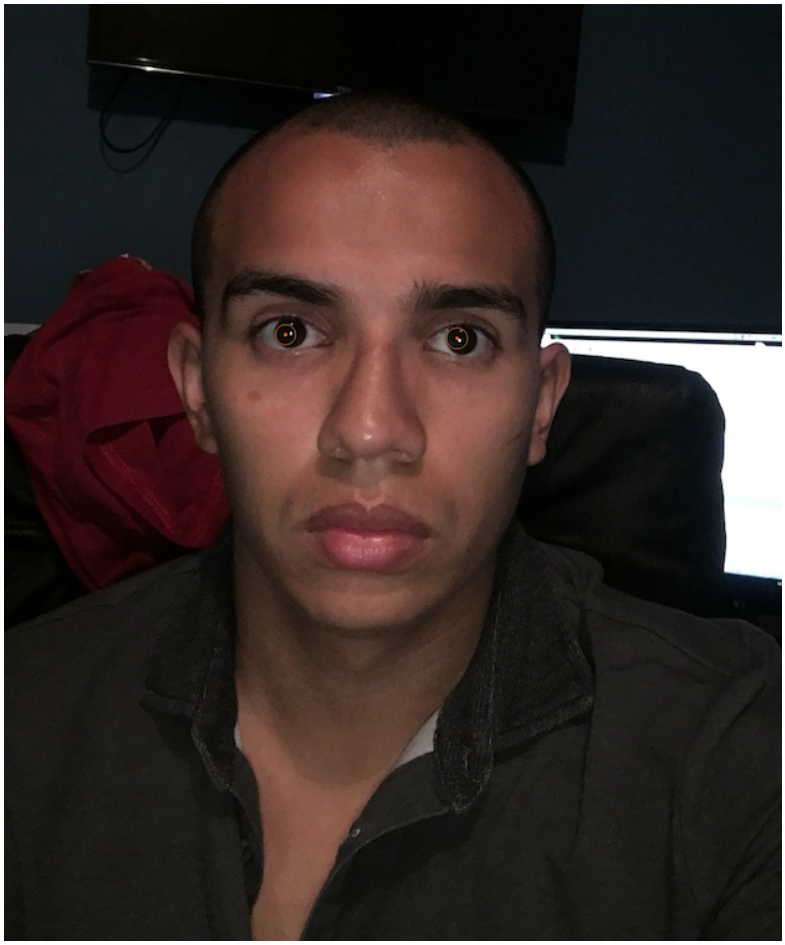


Figura 26 Prueba errónea 2.

Fuente: Elaboración propia.

Pruebas exitosas

El caso de éxito que se presenta a continuación, es uno que se realizó con medidas conocidas del autor de esta investigación, las cuales fueron obtenidas por el Doctor Zarama en su consultorio.

En la Figura 27 se pueden observar las medidas interpupilares tomadas a

una distancia de 40cm del paciente. En este caso se evidencia que la misma es de 6.38cm.

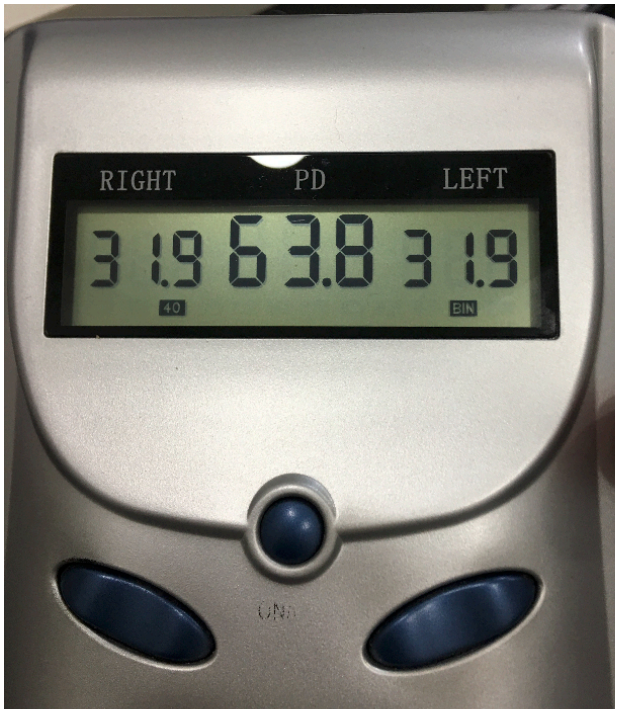


Figura 27. Medias interpupilares de Alejandro Sánchez.

Fuente: Elaboración propia.

Por otro lado, en la Figura 28 muestra una fotografía analizada por la aplicación, la que se puede comparar con la anterior y observar puntualmente que:

1. En la Figura 27 la distancia interpupilar del ojo izquierdo es de 3.19cm y la Figura 28 es de 3.12
2. En la Figura 27 la distancia interpupilar del ojo derecho es de 3.19cm y la Figura 28 es de 3.2
3. En la Figura 27 la distancia nasopupilar es de 6.38cm y la figura 28 es de 6.388cm

Con los hallazgos anteriormente descritos, se puede indicar que la aplicación desarrollada puede llegar a ser muy precisa si se toman ciertas recomendaciones antes de tomar la fotografía para el respectivo análisis, como se mencionó anteriormente.

Figura 28. Prueba exitosa.

Fuente: Elaboración propia.

Aplicación móvil para el Cálculo objetivo de la Distancia Interpupilar

Figura 29. Métodos de autenticación

Fuente: Elaboración propia.

En esta sección se trata de demostrar mediante capturas de pantalla la funcionalidad de la aplicación móvil para el Cálculo objetivo de la Distancia Interpupilar, que fue desarrollada en *React Native*, por tanto, puede ser ejecutada en dispositivos iOS y Android. Tanto, los diseños como la funcionalidad fueron realizados por el autor de esta investigación.

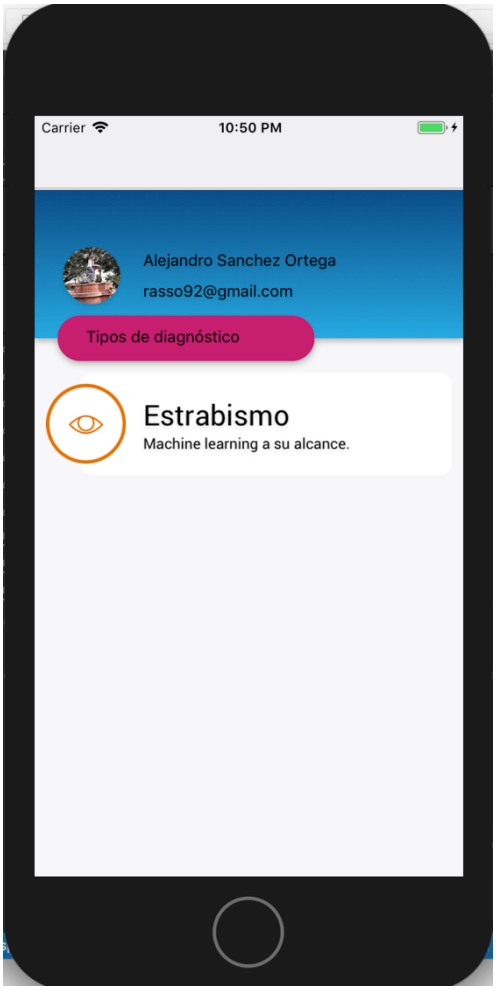
A continuación, se detallan las partes de la aplicación móvil.

Pantalla de bienvenida (Ver figura 29).

En este caso, la aplicación cuenta con diferentes métodos de autenticación, los cuales son:

- Facebook.
- Google.
- Correo electrónico

Los tres métodos son completamente funcionales dentro de este prototipo.



Lista de tipos de diagnóstico

En esta sección, se demuestran los tipos de diagnóstico o acciones que se pueden llevar a cabo dentro de la aplicación.

Figura 30. Lista de tipos de diagnóstico.

Fuente: Elaboración propia.

Figura 31. Lista de tipos de diagnóstico.

Fuente: Elaboración propia.

Opciones

Al seleccionar la opción de "Estrabismo" en el menú anterior, se despliega el menú de la **Figura 31**, el cual da la posibilidad de realizar un nuevo análisis, o ver el último análisis.

La opción de ver el último análisis aún no está disponible, sin embargo, es una acción perfectamente viable dentro del flujo de la aplicación.

Figura 32. Lista de tipos de diagnóstico.

Fuente: Elaboración propia.

Captura de datos

En esta pantalla se capturan los datos necesarios para el análisis, las cuales son:

1. La fotografía que se desea analizar, la cual puede ser tomada en el momento o seleccionar una de la galería de fotos del dispositivo utilizado.

2. El radio mínimo y máximo para el ajuste de la circunferencia de la pupila.

Además, se despliega una leyenda con indicaciones, la cual establece que la fotografía debe ser tomada a exactamente a una distancia de 40 cm entre la persona y el dispositivo móvil.

Despliegue de resultados

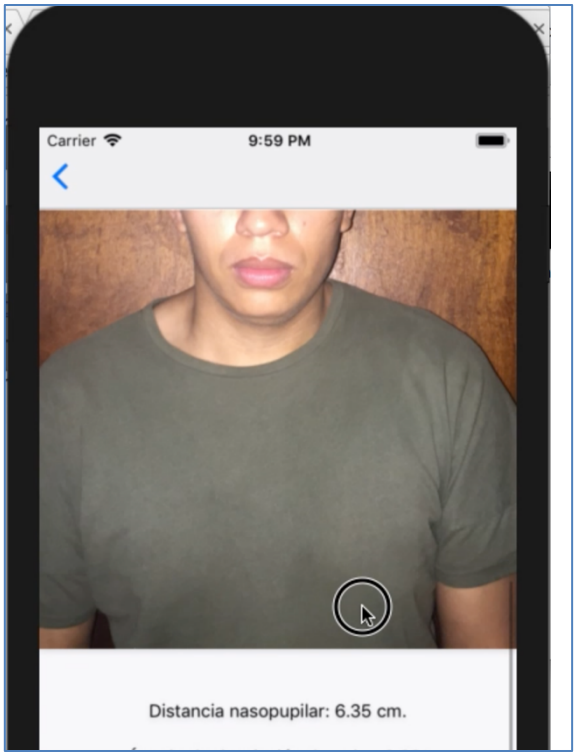


Figura 33. Despliegue de resultados.

Fuente: Elaboración propia.

En la Figura 33 se muestra como se despliegan los siguientes resultados:

1. Distancia nasopupilar
2. Distancias interpupilares.
3. Ángulo de cada ojo.

CAPÍTULO V
CONCLUSIONES Y RECOMENDACIONES

CAPÍTULO V: CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

A continuación, se presentan las principales conclusiones del proyecto:

1. La estrategia utilizada en este proyecto fue efectiva por el cumplimiento de todos los objetivos. Se llevó a cabo dos desarrollos de software de manera integra para lograr el resultado esperado, implementando funcionalidades para el análisis de imágenes, así como también un producto amigable para el usuario.

2. Del estudio de la bibliografía de los métodos convencionales de obtención del ángulo de convergencia en el diagnóstico de estrabismo para el desarrollo del software, se concluye que no se puede menospreciar los exámenes que se hacen actualmente de manera rudimentaria, ya que con estos se obtienen resultados valiosos, sin embargo, con una herramienta como la propuesta se puede llegar a automatizar este procedimiento con datos confiables que no dependen completamente de la habilidad examinador.

3. Se desarrolló una aplicación de software que se ejecuta del lado del servidor, en una arquitectura de tipo cliente-servidor en el framework para desarrollo web llamado Django, encargada del procesamiento y análisis de imágenes. Así también se encarga de la comunicación con el cliente, que en este caso en específico es la aplicación para dispositivos móviles.

4. El análisis y procesamiento de imágenes efectivo del proceso se realizó mediante la investigación y aprendizaje de cursos virtuales. No fue un proceso sencillo debido a la falta de experiencia en el tema del investigador, sin embargo, los resultados son sumamente satisfactorios y prometedores.

5. A pesar de que se logra cumplir con todos los objetivos del proyecto, se considera fundamental que el uso de esta aplicación aún no puede ser en producción debido que no se asegura la veracidad de los datos obtenidos porque no ha sido probada en pacientes con estrabismo.

6. La aplicación móvil a pesar de que aún no está disponible para el público en general, tiene la característica de que puede ser ejecutada tanto en dispositivos que ejecuten el sistema operativo Andriod como iOS. De esta manera se podría instalar a la mayoría de los dispositivos móviles de la actualidad brindando un mayor alcance a la misma.

7. Por otro lado, el realizar un desarrollo de software para un área, que al menos en este caso no es convencional como lo es la medicina, representó un gran reto. En este caso fue enriquecedor el acompañamiento del Dr. Miguel Zarama, en los temas que se trataron de abordar, para lo cual fue complicado no solo comprender sino también desarrollar la funcionalidad esperada.

8. Con respecto al cálculo de la distancia interpupilar mediante el uso de inteligencia artificial para el seguimiento de tratamientos de estrabismo, se podría deducir que a pesar de que se logró realizar satisfactoriamente la obtención de la distancia Interpupilar, y con esta calcular el ángulo de cada ojo, el software desarrollado aún tiene deficiencias que pueden afectar su desempeño.

9. Dentro de los factores que afectan el desempeño del algoritmo puede encontrarse la calidad del dispositivo de captura de imágenes, la luz y el entorno en general en el cual se tome la fotografía, que pueden interferir en los resultados.

5.2 Limitaciones

1. La fotografía debe ser tomada a una distancia de 40cm entre el paciente y el dispositivo móvil, ya que el cálculo del ángulo de cada ojo está definido partiendo de la premisa de que la fotografía fue tomada a esa distancia.

2. Existen objetos distractores a la hora de tomar la fotografía, considerando que el algoritmo que realiza el cálculo de la distancia interpupilar depende de figuras tales como círculos, en el caso de las pupilas, y cuadrados en el caso del artefacto calibrador, es vital que a la hora de capturar la fotografía se eviten los objetos distractores dentro de la misma.

3. El artefacto calibrador corresponde a un aro de lentes con cuadrados blancos adheridos a sus costados, los cuales tienen una diagonal de 2,2 cm y como consecuencia se recomienda el uso de un artefacto similar, el cual no obstaculice la visibilidad de los ojos.

4. Se debe considerar que en el punto de madurez en el que se encuentra este software no todas las personas pueden ser sujeto a análisis, ya que el algoritmo que se plasmó requiere, por ejemplo, un rostro completo, es decir, si el sujeto a analizar no tuviese nariz, el software no será capaz de obtener la medida. Es por esta razón que se debe tener esta consideración antes de realizar el análisis.

5.3 Trabajos futuros

1. Una de las recomendaciones para futuras investigaciones, es realizar pruebas en una población de personas que padezcan de estrabismo con la finalidad de validar y afianzar la validez de los datos obtenidos con la herramienta desarrollada.

2. Se recomienda utilizar Python como lenguaje de programación por ser ampliamente utilizado en la industria de la inteligencia artificial y cuenta con gran respaldo en la comunidad de ingenieros informáticos.

3. Se sugiere el uso del framework Django debido que adiciona al proyecto escalabilidad y otra serie de características en caso de que se desee mejorar la aplicación en el futuro

REFERENCIAS

Referencias bibliográficas

A Hands-on Guide to Building Apps with iOS and Android. Nueva

Álvarez, L. (2002). Antropología Social e Inteligencia Artificial. Macy, 95-110.

Andrés, A. (2015, Agosto) Qué Es Git. Recuperado 15 de octubre de 2018 de https://codigofacilito.com/articulos/que-es-git

Android Apps with Eclipse. (s. f.). Recuperado 19 de octubre de 2018, de https://books.google.es/books?hl=es&lr=&id=bPJnCEC0JkIC&oi=fnd&pg=PP2&dq=what+is+android+apps&ots=JOuyiJ3v3h&sig=VtuLfsqrw2IfDNj0PdNkaM_Ikdo#v=onepage&q=what%20is%20android%20apps&f=false

Anónimo. (2014, Julio 29). ¿Qué es el Estrabismo? - American Academy of Ophthalmology. Recuperado 10 de Agosto del 2017 de: https://www.aao.org/salud-ocular/enfermedades/estrabismoA.

Anónimo. (2015, Agosto 13). Statistics about Squint. Recuperado 03 de Septiembre del 2017 de http://www.rightdiagnosis.com/s/squint/stats.htm

Anónimo. (2015, Junio 8). Strabismus statistics. Recuperado 03 de Septiembre del 2017 de http://www.eyesapart.com/2005/06/08/strabismus-statistics/

Anónimo. (2016, Junio) Introducción a machine learning. Recuperado 11 de Noviembre del 2018 de https://es.slideshare.net/zanorte/introduccion-a-machine-learning

Anónimo. (2016). Gran Diccionario de la Lengua Española. Recuperado 11 de Noviembre 2018 de https://es.thefreedictionary.com/f%c3%b3vea

Anónimo. (2018) Introducción a Express/Node. Recuperado 15 de Octubre del 2018 de https://developer.mozilla.org/es/docs/Learn/Server-side/Express_Nodejs/Introduction

Bhattacharjee, J. (2017, octubre 27). Some Key Machine Learning Definitions. Recuperado 29 de octubre de 2018 de https://medium.com/technology-nineleaps/some-key-machine-learning-definitions-b524eb6cb48

Baviera, C. (2018, septiembre 12). Definición de motilidad ocular y sus principales disfunciones. Recuperado 29 de octubre de 2018, de https://www.clinicabaviera.com/blog/definicion-de-motilidad-ocular-y-sus-principales-disfunciones/

Díaz Hernandez, A., & Moreno Ramirez, C. E. (s.f). Variabilidad de la visión interpupilar en visión lejana y visión cercana. Recuperado 15 de Noviembre del 2018 de https://www.imagenoptica.com.mx/pdf/revista35/variabilidad.htm

Duckman, R. H. (2006). Visual Development, Diagnosis, and Treatment of the Pediatric. Philadelphia, Estados Unidos: Edwards Brothers.

Escacena, J. (2018) Desarrollando aplicaciones móviles nativas con React Native. Recuperado 15 de Octubre del 2018 de https://www.paradigmadigital.com/dev/desarrollando-aplicaciones-moviles-nativas-con-react-native/

Florencia, U. (2013). Oftalmología. Recuperado 11 de Noviembre del 2018 de https://www.definicionabc.com/salud/oftalmologia.php

Garvey, K. A., Dobson, V., Messer, D. H., Miller, J. M., & Harvey, E. M. (2010, Abril). Prevalence of strabismus among preschool, kindergarten, and 1st grade Tohono Oódham children. Recuperado 03 de Septiembre del 2017 de: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2847803/

Guerrero, V. J.(2017). Función visual humana, En: Guerrero Vargas, J.J(2017) Optometría clínica(3a ed.) & Cuidado Primario de la visión humana. pp 10-11. Cúcuta, Colombia: Editores Clinikbox.

Gutiérrez J. J. (2018) ¿Qué es un framework web?. Recuperado 11 de Noviembre del 2018 de http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf

Hernández, D. A. & Moreno, R. C. (2018) Variabilidad de la distancia interpupilar en visión lejana y visión cercana. Recuperado 8 de Noviembre del 2018 de http://www.imagenoptica.com.mx/pdf/revista35/variabilidad.htm

IBM Watson. (2016, Julio). La tecnología cognitiva que abre una nueva era de la computación. Recuperado 03 de Septiembre del 2017 de: http://www.agoraceg.org/banco-conocimiento/ibm-watson-la-tecnologia-cognitiva-que-abre-una-nueva-era-de-la-computacion

IBM. (2016, Noviembre 28). Products and services. Recuperado 25 de Agosto del 2017, de https://www.ibm.com/watson/products-services/

Iversen J. & Eierman M. (2013). Learning Mobile App Development. New York:Addison-Wesley

Jambrina, O. D. (2017, Junio 12). Diagnóstico del estrabismo. Recuperado 18 de Agosto del 2017 de: http://www.webconsultas.com/estrabismo/diagnostico-del-estrabismo-2488

Jones, T. (2008, Agosto 20). Anatomy of Linux dynamic libraries. Recuperado 10 de noviembre del 2018 de https://www.ibm.com/developerworks/linux/library/l-dynamic-libraries/

Joskowicz J. (2008). Reglas y Prácticas en Extreme Programming. Recuperado 7 de Noviembre del 2017 de: http://iie.fing.edu.uy/~josej/docs/XP%20-%20Jose%20Joskowicz.pdf

Kaehler A. & Bradski G.(2011) OpenCV. Recuperado 15 de Octubre del 2018 de https://www.ecured.cu/OpenCV

Kent, B. & Fowler, M. (2001, Marzo ,23) Planning extreme programming. Recuperado 7 de noviembre del 2017 de: http://www.awprofessional.com/articles/article.asp?p=20972&redir=1&rl=1

Maida, E.G. & Pacienzia, J. (2015) Metodologías de desarrollo de software [en línea]. Tesis de Licenciatura en Sistemas y Computación. Facultad de Química e Ingeniería "Fray Rogelio Bacon". Universidad Católica Argentina, 2015. Disponible en: http://bibliotecadigital.uca.edu.ar/repositorio/tesis/metodologias-desarrollo-software.pdf [Fecha de consulta: Recuperado 25 de Mayo del 2018]

Munaiseche et al.(2018) IOP Conf. Ser.: Mater. Sci. Eng. 306 012023 Recuperado 18 de noviembre de 2018 de http://iopscience.iop.org/article/10.1088/1757-899X/306/1/012023/pdf

Muñoz, D. (2016, Agosto 12). Supercomputadora Watson de IBM fue clave en caso de leucemia en Japón. Recuperado 03 de Septiembre del 2017 de: https://www.fayerwayer.com/2016/08/supercomputadora-watson-de-ibm-fue-clave-en-caso-de-leucemia-en-japon/

Navarro, P. M. (2016, Septiembre 28). Ventajas y beneficios de la tecnología en la salud. Recuperado 03 de Septiembre del 2017 de: https://clinic-cloud.com/blog/ventajas-beneficios-de-la-tecnologia-en-la-salud/

Neuburg, M. (2013, febrero 28). Programming iOS 6. Recuperado 26 de septiembre de 2018, de http://shop.oreilly.com/product/0636920029717.do

Pérez, B. L. (2011, Mayo-Agosto). Diagnóstico y tratamiento de un estrabismo. Recuperado 17 de octubre de 2018 de http://portal.scptfe.com/wp-content/uploads/2013/12/2011-2-5.c.pdf

Puell M. & María C. (2007). Óptica Fisiológica: el sistema óptico del ojo y la visión binocular. Universidad Complutense de Madrid.

Rodríguez S. J. (2003, Noviembre) Introducción a la programación. España: Editorial Club Universitario. Recuperado 8 de Noviembre del 2018 de https://www.editorial-club-universitario.es/pdf/405.pdf

Rouse, M. (2017). Computación cognitiva, o cómputo cognitivo. Recuperado 13 de Febrero del 2019 de: https://searchdatacenter.techtarget.com/es/definicion/Computacion-cognitiva-o-computo-cognitivo

Ruiz, A. (2018, diciembre 14). Algunos conceptos detrás de análisis Machine Learning. Recuperado 18 de diciembre de 2018, de https://www.geo-sapience.com/algunos-conceptos-machine-learning/

Russel, S. J., & Norvig, P. (2004). Inteligencia artificial un enfoque moderno (2da ed.). Madrid: pearson education. S.a.

Saavedra, E. G. (2009) Framework para el desarrollo de aplicaciones Web. Revista de Software Libre ATIX, 32-41, (En línea). Recuperado 08 de mayo 2018 de http://osl.ugr.es/descargas/atix08.pdf.

Sagonas C. & Zafeiriou S. (2013). Facial point annotacions. Recuperado 20 de octubre de 2018 de https://ibug.doc.ic.ac.uk/resources/facial-point-annotations/

Serrano Camacho, J. C., & Gaviria Bravo, M. L. (2011, Agosto-Noviembre). Estrabismo y ambliopía, conceptos básicos para el médico de atención primaria. MedUNAB. Recuperado 8 de Noviembre del 2018 de https://revistas.unab.edu.co/index.php/medunab/article/view/1561

Sintes, M. B. (2018, setiembre). Historia de Python. Recuperado 10 de Octubre del 2018 de: http://www.mclibre.org/consultar/python/otros/historia.html