

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARIA
INGENIERÍA EN DISEÑO DE PRODUCTOS
SANTIAGO - CHILE



UNIVERSIDAD TECNICA
FEDERICO SANTA MARIA

Diseño de Aplicación Móvil para Usuarios con Discapacidad Visual en Metro de Santiago

VALENTINA MAGDALENA VERGARA ESTRADA
MEMORIA DE TITULACIÓN PARA OPTAR AL TÍTULO DE

INGENIERA EN DISEÑO DE PRODUCTOS
PROFESOR GUIA - ROBERTO DURÁN

DICIEMBRE -2025



CONSTANCIA DE VALIDACIÓN Y CONFIDENCIALIDAD DE MONOGRAFÍA A REPOSITORIO ACADÉMICO

1.- IDENTIFICACIÓN DEL TRABAJO ACADÉMICO

Tipo de monografía (marcar una opción): ☒ Memoria o trabajo de título ☐ Tesis de Postgrado

Título del trabajo: **Diseño de Aplicación Móvil para Usuarios con Discapacidad Visual en Metro de Santiago.**

Nombre del candidato(a): **Valentina Magdalena Vergara Estrada**

Carrera / Grado: **Ingeniería en Diseño de Productos.**

Campus: **San Joaquín** Departamento: **Departamento de Ingeniería en Diseño**

2.- VALIDACIÓN DEL PROFESOR GUÍA/DIRECTOR DE TESIS

Yo, Roberto Alejandro Durán Novoa, en mi calidad de profesor(a) guía/director(a) del trabajo académico mencionado anteriormente **DEJO CONSTANCIA** que:

- He revisado esta versión del documento y corresponde a la versión final aprobada del trabajo.
- El trabajo cumple con los requisitos académicos y de formato establecidos por la institución.

3.- EVALUACIÓN DE CONFIDENCIALIDAD POR PROPIEDAD INDUSTRIAL (marcar una opción)

☒ El trabajo **NO contiene** información que amerite confidencialidad y puede ser publicado de inmediato en repositorio con acceso abierto.

☐ El trabajo **CONTIENE** información con potenciales implicancias de propiedad industrial o intelectual y requiere un periodo de confidencialidad (**embargo**) por (**marcar una opción**):

☐ 6 meses ☐ 12 meses ☐ 2 años ☐ 3 años ☐ 5 años ☐ 10 años

Fundamentación de la necesidad de confidencialidad (obligatorio si se solicita embargo):

4.- FIRMAS

Profesor(a) guía o director(a) de memoria o tesis:

Fecha: 22/12/2025

Firma:

Estudiante o Candidato(a):

Fecha: 22/12/2025

Firma:

Este formulario debe ser insertado como página 2 de la memoria o tesis, completado y firmado por estudiante y profesor(a) antes de la entrega en portal PRISMA de Biblioteca USM.

Dedicatoria

*Este trabajo está dedicado a todos los estudiantes
que alguna vez han pensado en que no llegarán a la meta
o sienten que se les está acabando el tiempo,
si yo lo logré, ustedes también pueden.*

Agradecimientos

Agradezco profundamente a todas las personas que me acompañaron y apoyaron durante este proceso. En especial a mi familia y a Victor, quienes estuvieron conmigo en los momentos más difíciles, cuando sentía que no podía continuar y que todo mi esfuerzo de años podía quedar en el olvido.

La universidad fue para mí una etapa llena de desafíos, pero también una fuente invaluable de experiencias y aprendizajes. Gracias a ella conocí a personas que hoy son muy significativas en mi vida y viví oportunidades que marcaron mi crecimiento personal. Una de ellas fue la posibilidad de realizar un intercambio de seis meses en el extranjero. Aunque implicó aprender a vivir sola en un país muy distinto, con un idioma nuevo y costumbres desconocidas, pude disfrutar intensamente cada momento. Hoy atesoro esos recuerdos y amistades que surgieron allí, espero pronto volver a reencontrarme con Jenny, Ege, Pablo y Haiame.

También agradezco la orientación de mi profesor guía, quien, a pesar de que asumir este proyecto implicaba enfrentarse a un proceso completamente nuevo, decidió acompañarme, apoyarme y guiarme en cada etapa. Espero que le pueda dar otra oportunidad al programa si un alumno solicita su ayuda en el proceso de la memoria.

Por último, pero no menos importante, agradezco a mis compañeros de equipo por esforzarse en sacar este proyecto adelante, por madrugar programando y corrigiendo el código. Muchas gracias Antonia, Pablo y Rodrigo por ser mi último grupo de trabajo en la universidad.

Índice

Dedicatoria	1
Agradecimientos	2
1. Resumen	5
2. Abstract	6
3. Introducción	7
3.1. Situación Actual	7
3.2. Desafíos del Entorno	8
3.3. ¿Qué se puede ver?	8
3.4. ¿Qué se puede hacer?	9
3.5. Estado del Arte	10
4. Requisitos del Sistema	16
4.1. Requisitos Funcionales	16
4.2. Requisitos no Funcionales	16
4.3. Requisitos de Interfaces	16
4.4. Requisitos de Ambiente	16
4.4.1. Hardware de Desarrollo	16
4.4.2. Software de Desarrollo	16
4.5. Entorno de Soporte y Desarrollo	17
4.6. Perfiles de Usuario	17
4.6.1. Joven con Discapacidad Visual	17
4.6.2. Adulto con Discapacidad Visual	17
4.6.3. Persona Mayor con Discapacidad Visual	18
4.6.4. Cuidador o Familiar	18
4.6.5. Asistente o Trabajador del Metro	18
4.6.6. Revisión de Requerimientos	19
5. Planificación del Proyecto	20
5.1. Marco General	20
5.2. Objetivo General (GOAL)	20
5.3. Objetivos Específicos	20
5.4. Actividades y Milestones	20
5.5. Metodología de trabajo	22
5.6. Carta Gantt	26
6. Diseño de Arquitectura del Sistema	27
6.1. Diagrama de Contexto	27
6.2. Diagrama de Arquitectura	27
6.3. Enumeración de Módulos	28
6.4. Matriz de Requisitos Funcionales y Módulos	29
6.5. Revisión de Arquitectura de Diseño	29
7. Módulos	30
7.1. Módulo 1: Investigación del Usuario y Diseño de Producto (UX_PD)	30
7.2. Módulo 2: Captura, Corrección y Preprocesamiento Visual (Ex_Im)	38
7.3. Módulo 3: Detección de Objetos y Dirección (Det_OI + Dir_OI)	40
7.4. Módulo 4: Interacción por Voz (Rec_Prompts + TaV_Dir)	42

8. Diseño de Interfaces	44
8.1. Principios de Diseño	44
8.2. Flujos de Interacción	44
8.3. Modelo de navegación	45
8.4. Componentes de la Interfaz	47
8.4.1. Interfaz de Usuario	47
8.4.2. Interfaz de Desarrollador	51
8.5. Justificación del Diseño	51
8.6. Resultados finales de Interfaz de Usuario	52
9. Revisión de Prototipo	53
9.1. Contexto General del Prototipo	53
9.2. Dataset	54
9.3. Entrenamiento Modelos CV	56
9.4. Interface Móvil	60
10. Gestión de Riesgos	63
10.1. Supuestos	63
10.2. Dependencias	63
10.3. Restricciones	63
10.4. Riesgos	63
10.5. Cómo Mitigar	64
11. Verificación y Validación	65
11.1. Verificación y Validación	65
11.1.1. Verificación:	65
11.1.2. Validación:	65
11.2. Plan de Testing	66
11.3. Análisis Crítico y Evaluación de Riesgos	66
12. Resultados Fases Metodológicas	67
13. Conclusión	68
13.1. Conclusiones	68
13.2. Trabajo Futuro	68

1. Resumen

En el siguiente trabajo de titulación de Ingeniería en Diseño de Productos se presenta el desarrollo de AIMES, un sistema de asistencia inteligente orientado a mejorar la movilidad y autonomía de personas con discapacidad visual en el Metro de Santiago. El objetivo principal del proyecto fue diseñar e implementar una solución accesible capaz de entregar orientación espacial en tiempo real mediante visión computacional y retroalimentación auditiva.

Dentro del equipo de trabajo, el rol de la estudiante se centró en el diseño de experiencia e interfaz de usuario, alineado con las competencias propias de la Ingeniería en Diseño de Productos. Su labor consistió en analizar las necesidades específicas de las personas con discapacidad visual, definir los métodos de interacción con el sistema y asegurar que la interfaz desarrollada fuese accesible y coherente con los lineamientos de accesibilidad digital. Esta asignación se determinó en función de su formación centrada en el usuario, lo cual permitió complementar el trabajo del resto del equipo, encargado de la implementación técnica de los módulos de visión computacional, procesamiento de imagen y programación de la aplicación.

El trabajo comenzó con una revisión exhaustiva de literatura sobre accesibilidad, movilidad urbana y asistencia tecnológica, junto con investigación de campo en la Biblioteca Central para Ciegos y entrevistas a usuarios reales. A partir de esta base, se aplicó una metodología de diseño iterativa que permitió definir requerimientos, generar conceptos y desarrollar tanto la interfaz accesible como los módulos técnicos del sistema.

La solución final integra captura y procesamiento de imagen, detección de elementos críticos de la estación y entrega de instrucciones auditivas claras. Los resultados del proceso evidencian una interfaz personalizable y compatible con lectores de pantalla, junto con un funcionamiento técnico orientado a la detección precisa de escaleras, ascensores, pisos podotáctiles y señalización relevante. Si bien no se logró implementar la navegación hacia salidas específicas ni rutas alternativas de emergencia, AIMES sienta una base sólida para futuras mejoras y aporta significativamente al diseño de tecnologías inclusivas.

Palabras clave: Accesibilidad, visión computacional, movilidad urbana, discapacidad visual, diseño inclusivo.

2. Abstract

The following thesis for the degree of Product Design Engineering, presents the development of AIMES, an intelligent assistance system designed to improve mobility and autonomy for visually impaired people in the Metro of Santiago. The primary objective of the project was to design and implement an accessible solution capable of providing real-time spatial guidance through computer vision and auditory feedback.

Within the project team, the student's role focused on user experience and interface design, aligned with the core competencies of Product Design Engineering. Her work involved analyzing the specific needs of visually impaired individuals, defining appropriate interaction methods for the system, and ensuring that the developed interface was accessible and consistent with established digital accessibility guidelines. This role was assigned based on her user-centered design background, which effectively complemented the work of the rest of the team, who were responsible for the technical implementation of the computer vision modules, image processing, and application programming.

The work began with an extensive literature review on accessibility, urban mobility, and assistive technologies, complemented by field research at the Central Library for the Blind and interviews with real users. Based on this foundation, an iterative design methodology was applied to define requirements, generate concepts, and develop both the accessible interface and the system's technical modules.

The final solution integrates image capture and preprocessing, detection of key station elements, and delivery of clear, concise auditory instructions. The results of the process indicate a customizable interface fully compatible with screen readers, alongside effective detection of critical elements such as stairs, elevators, tactile paving, and station signage. Although navigation toward specific exits and emergency alternative routes could not be implemented, AIMES establishes a solid foundation for future enhancements and contributes significantly to the development of inclusive technologies.

Keywords: Accessibility, computer vision, urban mobility, visual impairment, inclusive design.

3. Introducción

3.1. Situación Actual

En Chile, de acuerdo a la Encuesta Nacional de Discapacidad y Dependencia (ENDIDE, 2022), organizada por el Observatorio Social del Ministerio de Desarrollo Social y Familia[1], se presentan los siguientes resultados respecto a personas que presentan alguna discapacidad visual:

- Se estima que aproximadamente 4,6 millones de adultos presentan algún grado de pérdida de visión; de ellos, cerca de 153 mil viven con ceguera total.
- La Región Metropolitana concentra el 22,7 % de los adultos con pérdida de visión y el 24,9 % de los adultos con ceguera en Chile.
- Cerca de 12 mil niños y adolescentes presentan casos de ceguera, mientras que 535 mil tienen ceguera total.
- 59 % de las personas ciegas y el 76 % con pérdida de visión indican que pueden tomar decisiones por sí mismas.
- 7,8 % de las personas ciegas usa bastón, y menos del 2 % utiliza tecnología accesible como lectores de pantalla.

Esta información contrasta con investigaciones hechas por la Organización Mundial de la Salud (OMS, 2023)[2]:

- En el mundo hay al menos 2.200 millones de personas con deterioro de la visión cercana o lejana. Como mínimo en mil millones de estos casos, la discapacidad visual podría haberse evitado o minimizado con el tratamiento adecuado.
- Las principales causas a nivel mundial de la discapacidad visual y la ceguera son los errores de refracción y las cataratas.
- Aunque la pérdida de visión puede afectar a personas de todas las edades, la mayoría de las personas con discapacidad visual y ceguera superan los 50 años.

Esta situación impone desafíos urgentes en materia de accesibilidad e inclusión, especialmente en espacios públicos de alta afluencia como infraestructura de transporte público el Metro de Santiago. En este escenario, se hace necesario potenciar nuevas iniciativas que permitan mejorar la calidad de vida y la autonomía de las personas con discapacidad visual, especialmente en su movilidad y acceso a servicios esenciales.

3.2. Desafíos del Entorno

El Metro de Santiago constituye uno de los sistemas de transporte público más relevantes del país, caracterizado por su alta concurrencia y dinamismo. Sin embargo, el entorno de cada estación puede variar significativamente, lo que dificulta una experiencia uniforme y segura para personas con discapacidad visual. En la figura 1 se aprecian ejemplificaciones.

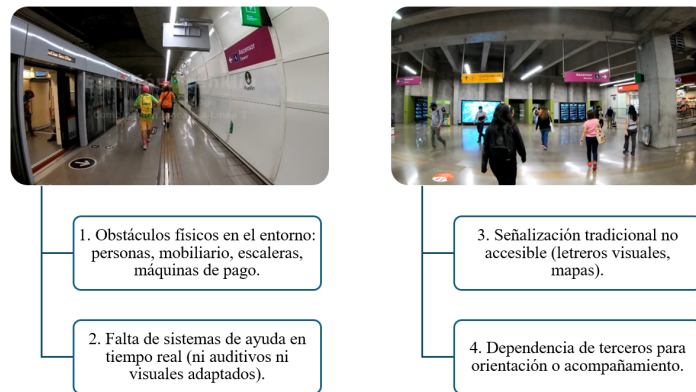


Figura 1: Entornos de conflicto inicial. ¿Cómo orientar a una persona con ceguera a través de estos pasillos?

Como problema central está la falta de autonomía y seguridad para personas con discapacidad visual en entornos de transporte urbano.

Factores de conflicto

- **Limitaciones de señalética tradicional:** La señalética visual no es interpretada por personas ciegas o con baja visión, lo que les dificulta identificar salidas, destinos y obstáculos.
- **Entorno dinámico e impredecible:** Las estaciones tienen flujo de personas y cambios constantes (construcciones, bloqueos, nuevas rutas) que dificultan la planificación anticipada.
- **Soluciones en infraestructura limitadas:** Sistemas como pisos podotáctiles y señaléticas escritas en braille que indican caminos únicos hacia ascensores preferenciales y que son utilizados indebidamente. [3]
- **Soluciones tecnológicas generales:** Aplicaciones de asistencia existentes como Be My Eyes[4] y Seeing AI[5] no están integradas al entorno ni ofrecen respuesta contextualizada y en tiempo real. Por otro lado, la aplicación de Lazarillo[6] [7] sí cumple con estas características de integración, pero se limita a ser un mapa de navegación en estaciones puntuales.
- **Aumento del riesgo:** La imposibilidad de identificar obstáculos o rutas incrementa la probabilidad de accidentes y dependencia de terceros, lo que limita la autonomía.

3.3. ¿Qué se puede ver?

Las personas con discapacidad visual están constantemente expuestas a incomodidades dentro del transporte público. No siempre cuentan con la posibilidad de pedir ayuda, ni existe garantía de que puedan recibirla de manera oportuna.

Como consecuencia, su frecuencia como usuarias del sistema de transporte público tiende a ser baja, lo que refuerza su exclusión del espacio urbano y disminuye sus oportunidades de integración social y laboral.

3.4. ¿Qué se puede hacer?

La integración de tecnologías emergentes en la infraestructura de transporte puede ofrecer soluciones reales y efectivas para las personas con discapacidad visual. Es fundamental adoptar un enfoque accesible desde el diseño de los sistemas, priorizando la autonomía y dignidad de los usuarios.

Además, se hace indispensable establecer alianzas estratégicas con empresas tecnológicas y fundaciones especializadas en inclusión, con el objetivo de desarrollar herramientas que respondan directamente a las necesidades de este grupo de personas.

Elementos relevantes:

- **Usuario:** El usuario principal es la persona con discapacidad visual que necesita desplazarse de manera independiente por estaciones de metro. El sistema busca facilitar su movilidad brindando orientación en tiempo real mediante retroalimentación auditiva, identificación de señales y detección de obstáculos. Usuarios secundarios incluyen familiares, cuidadores, personal del Metro y organizaciones de apoyo, que podrían configurar, supervisar o asistir en la implementación del sistema.
- **Producto:** El producto es un sistema de asistencia inteligente para personas con discapacidad visual que combina, procesamiento digital de imágenes en tiempo real, detección de señales y obstáculos, retroalimentación auditiva mediante texto a voz (TTS) y recepción de comandos por voz (STT).
- **Mercado:** El mercado objetivo abarca soluciones de tecnología asistiva y accesibilidad urbana, orientadas a personas con discapacidad visual. Se considera la posible colaboración con actores clave como Metro de Santiago (por su interés en accesibilidad e inclusión), organizaciones gubernamentales y ONGs dedicadas a la inclusión de personas con discapacidad.

Es importante destacar que se busca desarrollar este proyecto sin fines de lucro, por esta razón se habla de una colaboración con actores clave y no de posibles clientes.

3.5. Estado del Arte

Esta sección considera dos aspectos para importantes durante el desarrollo de AIMES: Contextual y Tecnológico. Ambos tienen el enfoque de establecer las bases con las que se desarrollan las funcionalidades, diseño de UX y PD, diseño de calidad de vida y de integración de tecnologías en el sistema.

Contextual

Si bien el desarrollo de este proyecto representa una oportunidad significativa para promover la autonomía de personas con discapacidad visual, también implica un desafío importante: la baja familiarización de este grupo con tecnologías móviles y digitales. Según ENDIDE 2022, solo el 1,1 % de las personas adultas ciegas y el 0,1 % de quienes tienen pérdida de visión utilizan herramientas tecnológicas de accesibilidad como lectores de pantalla o software especializado. Esta baja adopción tecnológica evidencia una brecha digital profunda, que podría afectar directamente la usabilidad y efectividad de soluciones tecnológicas si no se diseñan considerando sus habilidades, experiencias previas y contexto cotidiano.[8]

Por ello, este proyecto se plantea como una investigación exploratoria, que no solo busca implementar una solución, sino también comprender los hábitos, barreras y posibilidades reales de adopción tecnológica en esta población. No se pretende imponer una herramienta, sino co-diseñar una experiencia de asistencia accesible, gradual y ajustada a la realidad de los usuarios, con el fin de generar impacto social realista y sostenible.

El entorno objetivo para implementar la solución AIMES es el Metro de Santiago, por su arquitectura reconocible a gran escala. La dificultad se eleva al recordar las imágenes en la figura 1, por la misma razón, se debe limitar el alcance de la infraestructura a alimentar en el dataset para el sistema. Se consideran las líneas más recientes, como la línea 6 o la línea 3, para acotar el desarrollo y aprendizaje de AIMES, ya que son líneas que generalmente siguen el mismo patrón de espacios claros donde destacan los colores de las señales frente a los sobrios de las paredes, suelos, techos y ornamentas.

Tecnológico

Consideraciones deben realizarse respecto a qué hay presente en el mercado y cómo AIMES se acerca a estos productos. El cuadro 1 muestra una comparativa de distintas soluciones tecnológicas respecto a AIMES, considerando distintos atributos como el tipo de entrada y salida, si funcionan en tiempo real, qué ofrecen y si son de uso local o en línea.

Aplicación	Entrada	Salida	Tiempo real	Navegación	Offline
AIMES	Imagen + Video	Voz	Sí	Sí (Metro)	Sí
Seeing AI [5]	Imagen	Voz	No (foto)	No	No
Be My Eyes [4]	Video en vivo	Voluntario remoto	Sí	No	No
Aira [9]	Video en vivo	Agente humano	Sí (humano)	Sí (humano)	No
Lookout (Google) [10]	Imagen + Video	Voz	Sí	No	Sí (parcial)

Cuadro 1: Comparación de aplicaciones de asistencia visual

Por otro lado, respecto a las tendencias y novedades tecnológicas consideradas, se hizo revisión a distintos artículos que implementan innovaciones propias como arquitecturas conocidas que cumplan con tarea de detección de objetos y complementado con retroalimentación auditiva.

Respecto a las tendencias, se aprecia una alta integración de tecnologías de Ultralytics YOLO y MobileNet [11], con resultados diversos dependiendo de la tarea asignada y los recursos disponibles.

Para contrastar con las tendencias, se realiza una revisión a distintas tecnologías emergentes haciendo un trace-back desde el eslabón más antiguo hasta su innovación más reciente y de interés:

Segment Anything Model (SAM): El modelo Segment Anything (SAM), propuesto por Meta AI[12][13], introdujo un paradigma generalista para la segmentación automática de objetos, permitiendo generar máscaras de alta calidad a partir de entradas simples como puntos, cajas o máscaras previas. A través de su arquitectura basada en Vision Transformers (ViT-H), SAM demostró una capacidad sin precedentes para segmentar objetos sin requerir entrenamiento específico por clase. A partir de la versión 8.0.89, el framework de Ultralytics incorporó soporte para SAM, permitiendo combinarlo fácilmente con tareas de detección en el ecosistema YOLOv8. Esta integración ofrece nuevas posibilidades para sistemas como AIMES, especialmente en tareas de anotación automatizada o análisis visual más detallado.

Avance a Faster SAM / MobileSAM: MobileSAM es una implementación eficiente del modelo original, basada en el trabajo *Faster Segment Anything* de Zhang et al. (2023)[14][15], que utiliza un codificador Tiny-ViT más liviano y una técnica de destilación desacoplada para preservar precisión con menor carga computacional. Esta versión permite realizar segmentación interactiva en tiempo real sobre dispositivos edge, alineándose con los requerimientos de AIMES para operar localmente sin depender de servidores remotos ni conectividad constante.

Lanzamiento de Ultralytics YOLOv1 (2015)[16]: La familia de modelos YOLO (You Only Look Once) marcó un hito en la detección de objetos en tiempo real desde su primera versión en 2015. Su enfoque unificado y de extremo a extremo permitió realizar detección con alta velocidad y precisión en una sola pasada por la red. Esta arquitectura ha servido como base para múltiples aplicaciones móviles y edge, lo que la convierte en un referente fundamental para soluciones como AIMES, donde se requiere detección eficiente en tiempo real.

Lanzamientos de YOLOv11 y YOLOv12 (2024–2025)[17]: Las versiones más recientes de YOLOv11 y YOLOv12 han incorporado mecanismos de atención más refinados, fusión de características jerárquicas y mejoras en el entrenamiento semi-supervisado. Estas versiones apuntan a una mayor robustez frente a condiciones adversas y menor dependencia de grandes conjuntos de datos anotados, fortaleciendo su aplicabilidad en soluciones en tiempo real sobre plataformas embebidas.

Implementación de YOLO para ejecución remota (2023): Los autores de *Implementing Visual Assistant Using YOLO and SSD for Visually-Impaired Persons*[18] presentan una arquitectura de aplicación en la que los modelos de detección de objetos, específicamente YOLO y SSD, se ejecutan de manera remota en un servidor, mientras que el dispositivo del usuario (por ejemplo, un smartphone) actúa como cliente ligero. Esta solución permite superar las limitaciones de procesamiento local en dispositivos móviles al enviar las imágenes capturadas hacia el servidor, realizar allí la inferencia, y retornar los resultados al dispositivo, donde se entrega retroalimentación auditiva al usuario. Aunque este enfoque impone dependencias de conectividad y latencia de red, demuestra ser una opción viable para desplegar modelos complejos de visión por computador en tiempo real para personas con discapacidad visual, sin requerir hardware de alto rendimiento en el dispositivo cliente. Sin embargo, los cons están presentes en la latencia adquirida por la carga y subida de datos, como el riesgo de falta de conectividad por la movilidad del dispositivo.

DETR (2020): El modelo *DEtection TRansformer* (DETR)[19] introdujo por primera vez una arquitectura basada completamente en Transformers para la tarea de detección de objetos. A diferencia de los enfoques tradicionales basados en anchors, DETR reformula el problema como una tarea de predicción directa de objetos, lo que reduce la necesidad de postprocesamiento. Aunque su uso está más enfocado a dispositivos con alta capacidad de cómputo, su innovación ha inspirado múltiples variantes eficientes relevantes para sistemas como AIMES.

RepViT (2024): El artículo *RepViT: Revisiting Mobile CNN from ViT Perspective*[20] propone una nueva familia de arquitecturas de redes convolucionales eficientes para dispositivos móviles, inspiradas en el diseño de Transformers (ViT). Su principal contribución radica en combinar la eficiencia estructural de CNNs con principios de diseño de ViT, logrando superar a modelos como MobileViT y EfficientFormer tanto en precisión como en latencia. Es relevante para AIMES al presentar un enfoque moderno y liviano para tareas de visión por computador en tiempo real, especialmente útil en dispositivos edge.

Arquitectura DCNN Two Branches (2025): El enfoque propuesto por los autores en *Efficient Real-Time Pathfinding for Visually Impaired Individuals*[21] trata sobre una arquitectura de red neuronal convolucional profunda con dos ramas paralelas para segmentación semántica en tiempo real, diseñada específicamente para asistir a personas con discapacidad visual en la navegación. Una rama se enfoca en capturar detalles espaciales mediante canales anchos y capas superficiales (“demarcation path”), mientras que la otra extrae características semánticas profundas (“path with rich features”). Esta configuración permite balancear precisión y eficiencia computacional, alcanzando un 72,6% de mIoU y una velocidad de 120 fotografías por segundo (FPS) en GPU RTX 4090, favoreciendo la implementación en dispositivos portátiles con restricción de recursos. El modelo mejora la comprensión del entorno al identificar caminos y obstáculos, incrementando la autonomía y seguridad de los usuarios.

MobileNet, CNN para smartphones (2017): MobileNet[22] introduce un diseño eficiente basado en convoluciones separables en profundidad que reduce significativamente la complejidad computacional y el tamaño del modelo sin comprometer demasiado la precisión. Esta arquitectura está optimizada para correr en dispositivos con recursos limitados como smartphones, facilitando aplicaciones en tiempo real. Gracias a su bajo consumo de memoria y capacidad para mantener un rendimiento competitivo, MobileNet ha sido ampliamente adoptado en sistemas embebidos y móviles, sentando las bases para el desarrollo de redes ligeras aplicadas en sistemas de asistencia visual y otras aplicaciones móviles.

MobileViT, integración de transformer (2021): MobileViT[23] es una arquitectura híbrida que combina la eficiencia de las convoluciones de MobileNet con la capacidad de modelado global de los Transformers. A diferencia de modelos puramente convolucionales, MobileViT introduce bloques de atención que permiten capturar dependencias espaciales de largo alcance sin incrementar significativamente la complejidad computacional. Su diseño modular y compacto lo hace especialmente adecuado para tareas de visión por computador en tiempo real en dispositivos móviles. Esta arquitectura ofrece una buena relación entre precisión y eficiencia, lo que la convierte en una opción viable para reemplazar o complementar backbones convencionales en sistemas embebidos como AIMES, donde se requiere procesar simultáneamente detección, segmentación y OCR con bajo consumo de recursos.

Evaluación de modelos de Deep Learning en smartphones (2021): El artículo *Smart at What Cost? Characterizing Mobile Deep Learning Inference Performance*[24] presenta un estudio holístico sobre el uso y desempeño de modelos de redes neuronales profundas (DNNs) en aplicaciones móviles del mundo real. A través del análisis de más de 16 mil aplicaciones populares en la Google Play Store, los autores caracterizan las arquitecturas, optimizaciones y cargas computacionales asociadas a los modelos DNN utilizados en dispositivos móviles de distintas gamas y generaciones. Para ello, desarrollaron *gaugeNN*, una herramienta que automatiza la extracción, medición y evaluación del rendimiento de estos modelos. El estudio revela una creciente penetración de DNNs en el ecosistema móvil, así como importantes desafíos en términos de latencia, consumo energético y compatibilidad con hardware heterogéneo. Además, se destacan las brechas entre las técnicas avanzadas propuestas en la literatura y su adopción práctica, enfatizando la necesidad de herramientas de despliegue optimizadas para un entorno tan dinámico y limitado en recursos como el móvil.

La revisión de arquitecturas y modelos recientes evidencia una clara tendencia hacia redes neuronales cada vez más eficientes y especializadas para dispositivos con recursos limitados, como smartphones o dispositivos embebidos, sin sacrificar la precisión y velocidad requeridas en aplicaciones en tiempo real. Tecnologías como YOLO y sus evoluciones, junto con propuestas recientes como RepViT, MobileSAM y arquitecturas especializadas de segmentación, ofrecen bases sólidas para el desarrollo de sistemas que combinan detección y segmentación con retroalimentación auditiva. En particular, la implementación de YOLOv12 se presenta como una opción idónea para la detección rápida y precisa, mientras que modelos ligeros basados en RepViT o versiones optimizadas de arquitecturas dual-branch para segmentación permiten complementar la comprensión del entorno con un bajo costo computacional. Nuestro trabajo apunta a integrar estas innovaciones para construir una solución optimizada que permita una asistencia visual en tiempo real accesible, robusta y práctica para personas con discapacidad visual, maximizando tanto la autonomía como la seguridad en entornos urbanos y complejos como el Metro de Santiago.

VOSK (2022): VOSK[25] es una librería de reconocimiento de voz de código abierto que permite realizar transcripción de voz a texto completamente offline. Está optimizada para dispositivos con recursos limitados, como teléfonos móviles, y soporta múltiples idiomas, incluyendo español. VOSK fue seleccionada para AIMES por su bajo consumo de recursos, velocidad de respuesta aceptable y capacidad de funcionar sin conexión, lo cual es clave en entornos como estaciones de Metro donde la conectividad puede ser variable.

RHVoice (2022): RHVoice[26] es un motor de síntesis de voz (Text-to-Speech) desarrollado bajo licencia libre, con soporte para varios idiomas y voces naturales, incluyendo opciones en español. Su rendimiento estable en entornos móviles y su posibilidad de operación completamente offline lo hacen especialmente apto para sistemas accesibles como AIMES, que requieren respuestas auditivas claras, empáticas y sin depender de la nube.

Kivy (2011): Kivy[27] es un framework de desarrollo multiplataforma en Python orientado a la creación de aplicaciones con interfaces gráficas y soporte táctil. Durante las primeras etapas de prototipado de AIMES, Kivy fue utilizado en conjunto con Buildozer para generar aplicaciones Android empaquetadas en formato APK. No obstante, su uso fue finalmente descartado debido a limitaciones en la integración con APIs nativas de Android, la falta de soporte avanzado para funcionalidades de accesibilidad del sistema operativo, el acceso restringido a servicios de audio y sensores en tiempo real, y el rendimiento subóptimo al trabajar con visión artificial y módulos de voz. Esto motivó la migración hacia desarrollo nativo con Android Studio, lo cual permitió mayor estabilidad, control de permisos, acceso directo a componentes del sistema y cumplimiento estricto de estándares de accesibilidad móvil.

WCAG (2018): La *Web Content Accessibility Guidelines*[28] (WCAG) es un conjunto de directrices que promueve la perceptibilidad, operabilidad, comprensibilidad y robustez de los contenidos digitales. Este enfoque busca asegurar una experiencia inclusiva para personas con discapacidad visual, especialmente aquellas con baja visión o daltonismo, al garantizar que los sitios web y aplicaciones sean accesibles a través de diversas tecnologías de asistencia, como lectores de pantalla, así como mediante el uso de contrastes adecuados, navegación mediante teclado y otros métodos de interacción inclusivos.

Android Studio (2013): Android Studio[29] es el entorno de desarrollo integrado (IDE) oficial para la creación de aplicaciones Android, basado en IntelliJ IDEA y mantenido por Google. Ofrece herramientas avanzadas para diseño de interfaces, depuración en tiempo real, emulación de dispositivos, integración con Gradle y optimización de rendimiento. En AIMES, Android Studio se utiliza como entorno central para empaquetar, compilar y desplegar la aplicación móvil, garantizando compatibilidad con versiones recientes de Android y acceso a APIs nativas necesarias para sensores, audio en tiempo real y permisos del sistema.

Kotlin (2011): Kotlin[kotlin] es un lenguaje de programación moderno y conciso desarrollado por JetBrains, diseñado para ejecutarse sobre la JVM y plenamente interoperable con Java. Su sintaxis segura y expresiva, junto con características como null-safety, funciones de extensión y soporte nativo para programación asíncrona mediante corutinas, lo han convertido en el lenguaje recomendado para el desarrollo Android. En AIMES, Kotlin permite construir lógica más robusta, reducir errores comunes y optimizar tareas en tiempo real, como la gestión de sensores, procesamiento de audio y comunicación con módulos de visión computacional.

Estas tecnologías enfocadas en el diseño de aplicaciones móviles son clave para la integración del modelo de CV en un software compactado y compatible con estándares de accesibilidad para personas con discapacidad visual. La correcta implementación de la interfaz, siguiendo los lineamientos de la WCAG, permitirá que las personas con discapacidad visual interactúen de manera efectiva y autónoma con la aplicación, garantizando una experiencia fluida y sin barreras, mejorando su accesibilidad a la información y servicios proporcionados por la aplicación móvil.

En los cuadros 2 y 3 se exponen resúmenes de las tecnologías destacables y de interés para este proyecto.

Tendencia	Descripción	Implementación en AIMES
Modelos ligeros de CV (MobileNet, RepViT)	Redes convolucionales optimizadas para dispositivos móviles con baja latencia y alta precisión.	Obligatorio
OCR en tiempo real	Reconocimiento de texto en ambientes dinámicos como estaciones o paneles de trenes.	Obligatorio
Fusión multimodal	Combinación de visión, texto y voz para una experiencia más robusta.	Obligatorio
Interacción por voz personalizada	Asistentes con prompts adaptados al contexto y preferencias del usuario.	Obligatorio
Edge Computing	Procesamiento directamente en el dispositivo para evitar depender de internet.	Obligatorio
Mapeo semántico del entorno	Reconocimiento contextual de objetos y señales para entender el entorno.	Deseable
Framework multiplataforma (Kivy) [27]	Desarrollo de aplicaciones móviles con GUI en Python, adaptable a dispositivos Android.	Deseable
Framework Android Studio	Desarrollo de aplicaciones móviles con GUI en Kotlin. APK inmediata desde su compilador.	Obligatorio

Cuadro 2: Tendencias tecnológicas en asistencia visual y relación con AIMES

Innovación	Descripción
MobileViT[23] (2021)	Híbrido entre MobileNet y ViT, combina eficiencia estructural de CNNs con el modelado global de Transformers.
RepViT[20] (CVPR 2024)	CNNs móviles inspiradas en ViT que superan a MobileViT y EfficientFormer en precisión y latencia para edge computing.
MobileSAM[14] (2023)	Versión ligera de SAM basada en Tiny-ViT, permite segmentación eficiente en dispositivos móviles.
YOLOv12[17][30] (2025)	Versión más reciente y robusta de la familia YOLO, con mecanismos de atención jerárquicos y optimización para ejecución en edge.
Arquitectura DCNN Two-Branches[21] (2025)	Red especializada con dos rutas paralelas para segmentación en tiempo real, pensada para asistencia a personas con discapacidad visual.
VOSK[25] (2022)	RMotor de reconocimiento de voz offline, eficiente en móviles, con soporte multilingüe.
RHVoice[26] (2022)	Motor de texto a voz de código abierto con voces naturales y operación completamente offline.

Cuadro 3: Novedades tecnológicas recientes aplicables a sistemas como AIMES

A continuación, en la figura 2, se muestra cómo las tecnologías descritas se relacionan a cada módulo del proyecto:

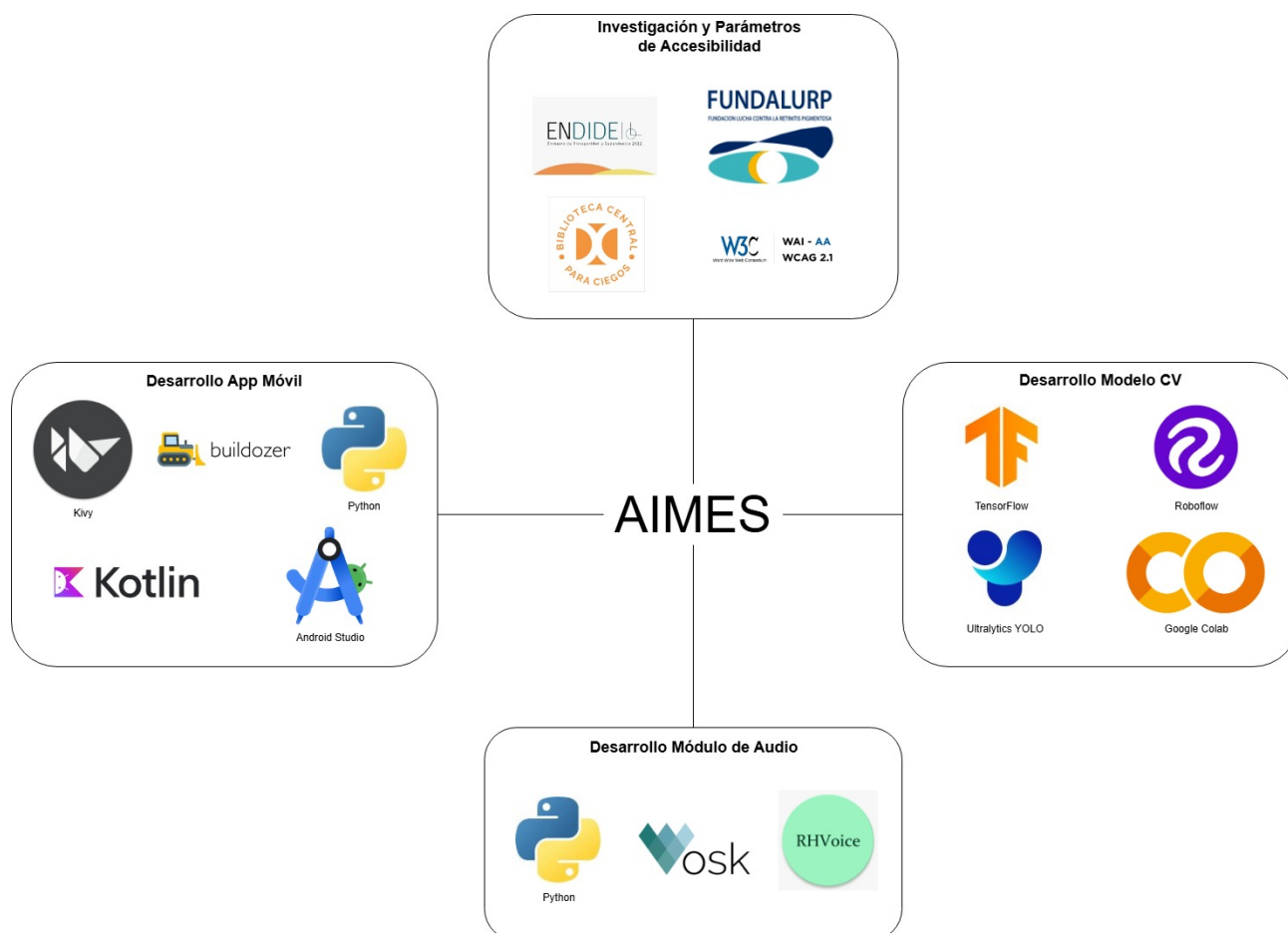


Figura 2: Relación de las tecnologías a utilizar con los módulos de trabajo

4. Requisitos del Sistema

4.1. Requisitos Funcionales

1. Captura de imágenes en tiempo real: Se utilizará una cámara (de teléfono móvil o externa) para obtener información visual del entorno.
2. Modelo de Detección de señales y texto de Metro de Santiago: Integración de modelos de *Computer Vision* para detección de señales y texto.
3. Comunicación Verbal: El sistema sea capaz de obtener y considerar instrucciones del usuario.
4. Orientación a través de voz para el usuario: El sistema debe proveer información al usuario a través de audio.

4.2. Requisitos no Funcionales

1. Compatibilidad: Los distintos componentes que conforman la solución se puedan comunicar entre sí.
2. Portabilidad: El software se encuentre disponible para el usuario, desarrollado en Android.
3. Confiabilidad: Debe estar operando en toda la experiencia del usuario.
4. Usabilidad: Debe ser operable mediante comandos de voz claros y concisos, con asistencia de voz entendible para el usuario.

4.3. Requisitos de Interfaces

1. Cámara del dispositivo móvil: Cámara para detectar el entorno del usuario, con el fin de capturar imágenes para procesarlas.
2. Aplicación móvil: Permite la interacción, configuración, recepción de comandos de voz y asistencia audible.
3. Auriculares o salidas de audio: Para emitir las instrucciones de navegación al usuario.

4.4. Requisitos de Ambiente

4.4.1. Hardware de Desarrollo

1. Computador: Para el diseño, codificación y entrenamiento de modelos de visión artificial y reconocimiento de voz a texto.
2. Cámara: Para simular la entrada de video en pruebas locales del sistema.
3. Auriculares o parlantes externos: Para probar la retroalimentación auditiva (text-to-speech).

4.4.2. Software de Desarrollo

- Entrenamiento de modelo de detección de señales y texto: OpenCV y Roboflow.
 - Python: Consideración del framework Kivy [27] y Ultralytics [31] para desarrollo interfaz móvil y modelos de CV en lenguaje Python.
 - Android Studio: Utilización de Android Studio para diseño y compilación de ejecutable para su emulación directa en dispositivo móviles.
 - Para la Asistencia por voz: Librería Compatible con Kivy. Potencialmente Plyer.[32]
 - Sistema Operativo: Windows, para desarrollar la solución. Android para migración de solución a aplicaciones móviles.
-

4.5. Entorno de Soporte y Desarrollo

El entorno de soporte y desarrollo de AIMES combina hardware accesible y software de código abierto con herramientas de despliegue nativo en dispositivos móviles. El diseño, entrenamiento de modelos y pruebas iniciales se realizan en un computador de escritorio, mientras que los emuladores de dispositivos Android permiten validar las interacciones y latencias en condiciones similares al uso real.

En la etapa de prototipado temprano, se evaluó el uso de Kivy y su empaquetado mediante Buildozer para la generación de APKs. Sin embargo, esta alternativa fue descartada debido a limitaciones en el acceso a APIs nativas de Android, la falta de soporte robusto para accesibilidad a nivel de sistema y la necesidad de integrar módulos de audio, visión y permisos con baja latencia.

En consecuencia, la arquitectura móvil de AIMES pasó a desarrollarse de forma nativa en Android Studio, lo que permite un mayor control sobre la interfaz de usuario, uso de componentes accesibles del sistema operativo, integración directa con motores STT/TTS, sensores y librerías de visión. Este ecosistema asegura una transición fluida desde prototipos funcionales hacia una aplicación móvil estable, escalable y alineada con las buenas prácticas de desarrollo Android.

4.6. Perfiles de Usuario

Esta sección presenta los distintos perfiles de usuario identificados para el desarrollo de AIMES, considerando las características, necesidades y contextos de las personas con discapacidad visual que podrían utilizar la aplicación. Los rangos etarios fueron determinados en base a la información obtenida del **Estudio de Personas con Discapacidad Visual en Chile según ENDIDE 2022** [8]. Estos antecedentes indican que la población con discapacidad visual se concentra principalmente en adultos y adultos mayores, aunque también se consideraron jóvenes adultos, quienes muestran mayor familiaridad con el uso de dispositivos móviles.

4.6.1. Joven con Discapacidad Visual

- **Socioeconómico y cultural:** Nivel socioeconómico diverso, pero prevalecen medio o medio bajo; generalmente con acceso a educación superior o media completa. [33]
- **Ocupacional:** Estudiante universitario o trabajador en formación.
- **Etario:** 18–29 años.
- **Características físicas, fisiológicas, psicológicas:** Se presume una buena salud física. Independiente en su vida diaria. Tiene autonomía en toma de decisiones. No posee afecciones mentales que afecten su independencia.
- **Otros:** Busca independencia total en el desplazamiento. Se infiere que utiliza smartphone.

4.6.2. Adulto con Discapacidad Visual

- **Socioeconómico y cultural:** Nivel socioeconómico diverso. Puede tener escolaridad media o técnica.[33]
- **Ocupacional:** Trabajador formal/informal o cesante.
- **Etario:** 30–59 años.
- **Características físicas, fisiológicas, psicológicas:** Capacidad física variable y sin afecciones mentales que puedan perjudicar su autonomía. Puede presentar condiciones asociadas como glaucoma.
- **Otros:** Uso diario u ocasional de ayudas técnicas. Se presume que al menos está familiarizado con las nuevas tecnologías.

4.6.3. Persona Mayor con Discapacidad Visual

- **Socioeconómico y cultural:** Nivel socioeconómico bajo a medio. Poco familiarizado con la tecnología. Riesgo de exclusión digital.[33]
- **Ocupacional:** Jubilado o dependiente de apoyo familiar o estatal.
- **Etario:** 60 años o más.
- **Características físicas, fisiológicas, psicológicas:** Puede tener limitaciones físicas asociadas a la edad. Bajo manejo de tecnología. Posible comorbilidad (existencia de dos o más enfermedades).
- **Otros:** Requiere sistema simplificado. Mayor necesidad de soporte o asistencia con la tecnología.

4.6.4. Cuidador o Familiar

- **Socioeconómico y cultural:** Nivel socioeconómico diverso. Generalmente de entorno directo del usuario final. Compromiso afectivo.[33]
- **Ocupacional:** Familiar, acompañante o cuidador contratado.
- **Etario:** 20–65 años.
- **Características físicas, fisiológicas, psicológicas:** Buen estado físico y mental. Capacidad para asistir y supervisar.
- **Otros:** Rol secundario pero clave en configuración y soporte del sistema.

4.6.5. Asistente o Trabajador del Metro

- **Socioeconómico y cultural:** Empleado del sistema de transporte público. Capacitado en atención a usuarios.[33]
- **Ocupacional:** Personal de atención al pasajero.
- **Etario:** 25–60 años.
- **Características físicas, fisiológicas, psicológicas:** Buen estado físico. Formación en atención al público. Capacidad para resolver situaciones de emergencia.
- **Otros:** Puede actuar como punto de apoyo ocasional. No opera directamente el sistema.

Se ha identificado dos categorías de usuarios: principales, que incluyen jóvenes, adultos y adultos mayores con discapacidad visual; y secundarios, conformados por cuidadores, familiares y asistentes del metro. Sobre la base de esta clasificación, se ha determinado que el enfoque principal será el adulto con discapacidad visual, dado que la prevalencia de esta condición es significativamente mayor en dicho grupo etario.[1, 8]

4.6.6. Revisión de Requerimientos

Los requerimientos funcionales fueron revisados para asegurar que cubran las características mínimas necesarias para el MVP, permitiendo pruebas en terreno y la validación inicial del sistema con usuarios reales. Se confirmó que estos requisitos permiten generar una versión demostrable y operativa de AIMES, sobre la cual será posible escalar módulos adicionales de accesibilidad, navegación y personalización en fases posteriores.

En cuanto a los requerimientos no funcionales y de interfaces, la revisión verificó que estos aportan estructura y consistencia al proyecto, al orientar su implementación hacia estándares de accesibilidad, tiempos de respuesta definidos, operación offline y compatibilidad con dispositivos Android de gama media. Esta evaluación permitió asegurar coherencia entre las distintas categorías de requisitos y su alineación con los objetivos de inclusión, rendimiento y usabilidad del proyecto.

Respecto a los requisitos de desarrollo, se re-evaluaron ciertas librerías y frameworks inicialmente considerados en el prototipo, como Kivy y sus módulos de audio, los cuales presentaron incompatibilidades con APIs nativas de Android y limitaciones en la integración con servicios de accesibilidad. Como resultado, se decidió migrar hacia un desarrollo nativo en Android Studio, lo que habilita el uso directo de motores de STT/TTS del sistema, acceso a sensores, permisos avanzados y componentes de interfaz accesibles. Esta transición asegura mayor estabilidad técnica, soporte a largo plazo y cumplimiento de las guías de accesibilidad oficiales de Android y WCAG.

5. Planificación del Proyecto

5.1. Marco General

AIMES se apoya en tres pilares fundamentales:

- i La necesidad social de mejorar la movilidad e inclusión de personas con discapacidad visual en sistemas de transporte masivo como el Metro de Santiago.
- ii Los avances en visión artificial, reconocimiento de voz y procesamiento en dispositivos móviles, que permiten implementar soluciones en tiempo real.
- iii Metodologías de diseño centradas en el usuario, que aseguran que la solución responda a requerimientos reales y se valide en entornos cotidianos.

Este marco proporciona la base conceptual y tecnológica sobre la que se estructuran los objetivos, actividades y entregables del proyecto.

5.2. Objetivo General (GOAL)

Desarrollar un sistema de asistencia inteligente que proporcione apoyo parcial a personas con discapacidad visual para desplazarse de forma más autónoma y segura en el Metro de Santiago, utilizando procesamiento de imágenes, detección de objetos e interacción por voz. Esta versión inicial busca ser la base para un sistema de orientación integral que pueda ser desarrollado y ampliado en trabajos futuros.

5.3. Objetivos Específicos

- OE1. Analizar información recopilada de personas con discapacidad visual mediante investigación contextual, para guiar el diseño del sistema.
- OE2. Capturar imágenes del entorno, preprocesándolas en tiempo real, para mantener un bajo consumo en dispositivos móviles con recursos limitados, mediante el uso de librerías de procesamiento digital de imágenes.
- OE3. Detectar señalética y obstáculos presentes en el entorno mediante técnicas de visión computacional, para entregar información visual relevante que apoye en la toma de decisiones del usuario durante su desplazamiento. La detección inicial estará limitada a elementos esenciales para la orientación, seguridad o toma de decisiones del usuario (los elementos esenciales serán definidos en la sección 7).
- OE4. Generar indicaciones espaciales claras y entregadas de manera oportuna, entendiendo “oportuna” como aquella información que se proporciona en el momento exacto en que el usuario la necesita para tomar decisiones seguras durante su desplazamiento, con un tiempo de respuesta menor o igual a 200 ms. Mediante un sistema de retroalimentación auditiva basado en la información visual capturada, para guiar al usuario en tiempo real durante su desplazamiento por la estación.
- OE5. Permitir la operación del sistema priorizando la interacción por voz, manteniendo opciones táctiles como respaldo, mediante la implementación de un módulo de interacción por comandos de voz y respuesta sonora, para facilitar el uso autónomo y accesible del sistema en distintos niveles de discapacidad visual.

5.4. Actividades y Milestones

A continuación se detallan específicamente las actividades y milestones del proyecto, organizadas según cada módulo correspondiente. La definición completa de los módulos, incluyendo sus objetivos y alcances, será presentada posteriormente en este informe. Además, la Carta Gantt incluida en la sección 5.6 complementa esta información, mostrando la planificación temporal de las tareas.

Módulo 1: Investigación del Usuario y Diseño de Producto (UX_PD)

- Milestone 1.1: Perfilamiento de usuario y contexto de uso.
 - 1.1.1: Investigación sobre movilidad de personas con discapacidad visual.
 - 1.1.2: Entrevistas con usuarios.
 - 1.1.3: Registro de puntos críticos en estaciones del Metro.
- Milestone 1.2: Diseño de experiencia e interacción.
 - 1.2.1: Definición de flujos de interacción.
 - 1.2.2: Diseño de voz UX (tono, estilo de mensajes)
 - 1.2.3: Configuración de interfaz por voz.
- Milestone 1.3: Prototipo de interfaz adaptable a móvil.
 - 1.3.1: Wireframes accesibles y diseño de producto.
 - 1.3.2: Diseño de UI accesible con feedback por voz.
 - 1.3.3: Implementación de configuración de volumen y preferencias.
 - 1.3.4: Implementación de seguridad y cifrado en la aplicación.
 - 1.3.5: Evaluación con usuarios.
 - 1.3.6: Validación de producto final.

Módulo 2: Captura, Corrección y Preprocesamiento Visual (Ex_Im)

- Milestone 2.1: Captura desde cámara móvil.
 - 2.1.1: Captura de video con OpenCV.
 - 2.1.2: Evaluación de latencia y estabilidad.
 - 2.1.3: Compatibilidad con app móvil.
- Milestone 2.2: Corrección de imagen intervenida.
 - 2.2.1: Dataset con stickers, rayas, manchas.
 - 2.2.2: Entrenamiento de autoencoder o GAN.
 - 2.2.3: Integración al pipeline visual.
- Milestone 2.3: Preprocesamiento.
 - 2.3.1: Escalado y ajuste de color.
 - 2.3.2: Canalización hacia modelo de *computer vision*.
 - 2.3.3: Testeo en dispositivos móviles.

Módulo 3: Detección de Objetos y Dirección. (Det_OI + Dir_OI)

- Milestone 3.1: Entrenamiento con dataset propio.
 - 3.1.1: Anotación de señales en el Metro.
 - 3.1.2: Entrenamiento con aumentaciones.
 - 3.1.3: Validación con mAP y FPS.
 - Milestone 3.2: Detección y clasificación.
 - 3.2.1: Filtrar clases relevantes.
-

- 3.2.2: Priorización de elementos visuales.
- 3.2.3: Overlays y validación en tiempo real.
- Milestone 3.3: Dirección espacial hacia objetivo.
 - 3.3.1: Estimar posición en cuadrantes.
 - 3.3.2: Generar instrucciones simples.
 - 3.3.3: Pruebas con videos del metro.

Módulo 4: Interacción por Voz (Rec_Prompt + TaV_Dir)

- Milestone 4.1: Recepción del comando por voz.
 - 4.1.1: Integración de STT.
 - 4.1.2: Interpretación de intención.
 - 4.1.3: Robustez en entorno ruidoso.
- Milestone 4.2: Asociación con escena visual.
 - 4.2.1: Mapeo de clases detectadas.
 - 4.2.2: Activar dirección al destino.
 - 4.2.3: Confirmación auditiva.
- Milestone 4.3: Retroalimentación auditiva.
 - 4.3.1: Motor TTS.
 - 4.3.2: Frases claras y empáticas.
 - 4.3.3: Ajustes para uso con auriculares.

5.5. Metodología de trabajo

Para enfrentar de manera efectiva los desafíos identificados en el proyecto AIMES, se adopta una metodología combinada basada en dos enfoques complementarios: Design Thinking para el análisis y comprensión de las necesidades del usuario, y Scrum para el desarrollo técnico iterativo e incremental del sistema.

- **Scrum:** Es un marco ágil de trabajo que permite desarrollar productos complejos de manera iterativa, enfocándose en entregas rápidas, colaboración constante y mejora continua.[34]
- **Design Thinking (DT):** Es una metodología centrada en el usuario que busca resolver problemas mediante la empatía, la ideación creativa y la experimentación con prototipos.[35]

1. Fase de Comprensión y Diseño Centrado en el Usuario (DT)

Objetivo: Comprender en profundidad los desafíos y necesidades de las personas con discapacidad visual en entornos urbanos como el Metro de Santiago, garantizando que la solución tecnológica responda a problemas reales.

Actividades a desarrollar:

- **Empatizar:** Realización de entrevistas, observaciones y levantamiento de datos sobre la movilidad de personas con discapacidad visual. Identificación de puntos críticos en estaciones (escaleras, accesos, andenes, señalética).
- **Definir:** Formulación precisa de los problemas principales:
 - Falta de señalización accesible.

- Obstáculos físicos imprevisibles.
- Necesidad de orientación auditiva en tiempo real.
- **Idear:** Generación de soluciones posibles, priorizando el uso de visión computacional y retroalimentación por voz.
- **Prototipar:** Desarrollo de primeros esquemas de interacción y flujos de experiencia de usuario.
- **Testear:** Validación preliminar con usuarios para ajustar el enfoque antes de iniciar el desarrollo técnico.

Resultado esperado: Una definición clara y validada de requisitos funcionales y técnicos basados en la experiencia del usuario.

2. Fase de Desarrollo Iterativo e Incremental (Scrum)

Objetivo: Construir el sistema mediante iteraciones cortas (sprints), enfocándose en entregar funcionalidades específicas, probadas y mejoradas continuamente. Se aprecia en la figura 3.

Organización por Sprints:

Sprint	Objetivo Técnico	Actividades	Resultado Esperado
Sprint 1	Captura y preprocesamiento de imágenes.	- Configuración de OpenCV. - Captura de video en tiempo real. - Implementación de modelos de corrección visual.	Flujo estable de imágenes preprocesadas.
Sprint 2	Detección de señales y obstáculos.	- Entrenamiento de modelos YOLO. - Filtrado de clases relevantes (señalética, obstáculos).	Detección precisa en condiciones reales.
Sprint 3	Generación de indicaciones espaciales.	- Algoritmo de direccionamiento basado en posición relativa de objetos. - Validación de instrucciones comprensibles.	Instrucciones claras de navegación para el usuario.
Sprint 4	Interacción por voz y retroalimentación auditiva.	- Integración de reconocimiento de voz (STT). - Implementación de retroalimentación auditiva (TTS).	Comunicación fluida entre usuario y sistema.

Figura 3: Tabla de Sprints

Eventos Scrum aplicados:

- **Sprint Planning:** Definición de las actividades y objetivos de cada sprint.
- **Daily Scrum:** Reuniones breves para sincronizar avances y detectar bloqueos.
- **Sprint Review:** Demostración de avances funcionales para recibir feedback.
- **Sprint Retrospective:** Reflexión interna sobre el proceso y ajustes para el siguiente sprint.

3. Fase de Validación e Iteración Final

Objetivo: Verificar que la solución completa sea funcional, accesible y alineada a las necesidades del usuario.

Actividades:

- Integración de los módulos desarrollados: captura, detección, navegación y voz.
- Pruebas en entornos reales o simulados (por ejemplo, estaciones de metro).
- Recopilación de feedback de usuarios reales con discapacidad visual.
- Ajustes finales basados en observaciones de uso real.

Resultado esperado: Un sistema funcional que mejora la movilidad y autonomía de personas con discapacidad visual en entornos urbanos complejos.

Mediante la combinación de **DT** y **Scrum** en el proyecto se espera:

- Entender en profundidad al usuario y diseñar soluciones centradas en sus verdaderas necesidades.
- Construir un sistema robusto, modular y escalable de manera eficiente.
- Adaptarse rápidamente a cambios y mejoras basadas en validaciones prácticas.

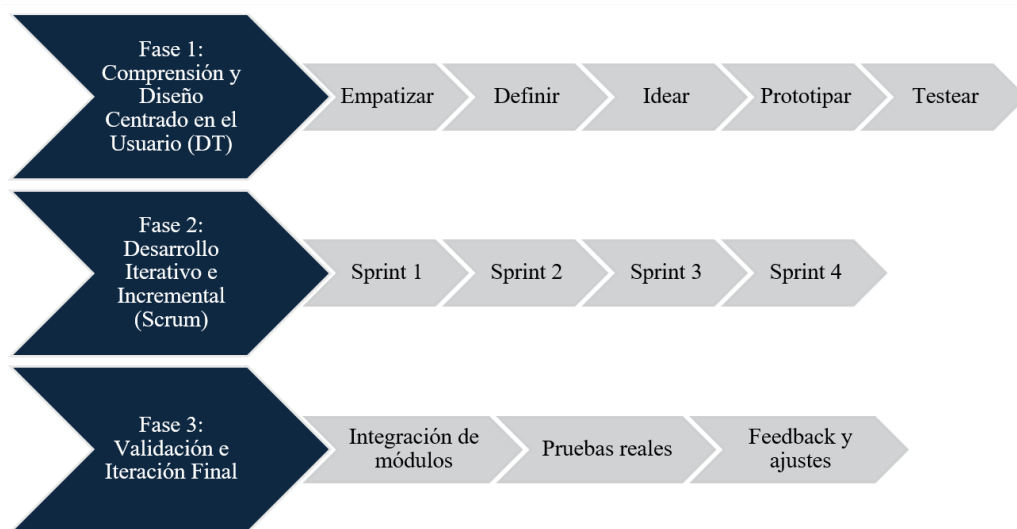


Figura 4: Diagrama de la Metodología

5.6. Carta Gantt

La Carta Gantt es una herramienta esencial de gestión y planificación, a continuación para ofrecer una visión clara y estructurada del trabajo, se presenta la figura 5 con las tareas asignadas a cada participante del proyecto.

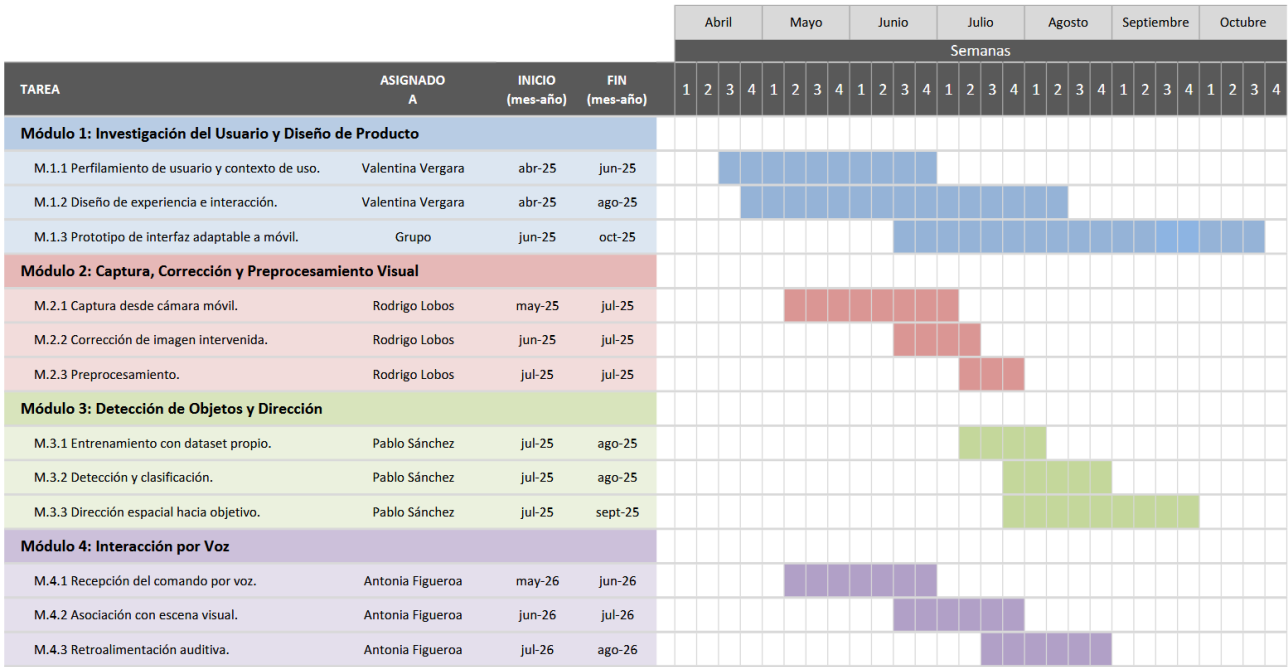


Figura 5: Carta Gantt del proyecto AIMES

6. Diseño de Arquitectura del Sistema

6.1. Diagrama de Contexto

En la figura 6 se muestra cómo interactúan los distintos dispositivos con el sistema. Se muestran las entradas hacia el sistema (desde la cámara y del usuario) y la salida (asistente por voz hacia la salida de audio):

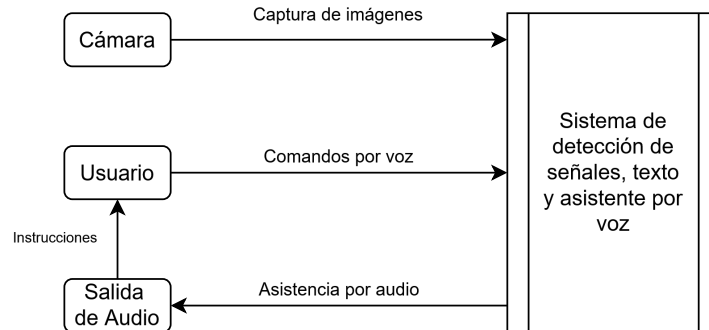


Figura 6: Diagrama del sistema

Tal como aparece en el diagrama, la cámara se encarga de entregar la entrada visual del sistema para su procesamiento y detección, mientras que el usuario puede enviar comandos por voz. La salida del sistema corresponde a asistencia por audio tras el procesamiento de las entradas, generando instrucciones comprensibles para el usuario.

6.2. Diagrama de Arquitectura

La figura 7 detalla los principales componentes funcionales del sistema de asistencia por voz para usuarios con discapacidad visual. La entrada de imágenes se procesa mediante un módulo de preprocesamiento y detección, que identifica objetos relevantes en el entorno. En caso de contener texto, se aplica reconocimiento óptico de caracteres (OCR) para extraer información direccional. Si no contiene texto, el sistema evalúa si el objeto pertenece a una ruta (como escaleras, ascensores o pisos podotáctiles) o si es un obstáculo, generando alertas en consecuencia. Paralelamente, los comandos de voz del usuario son interpretados mediante STT, integrándose con los módulos de orientación y asistencia para personalizar la respuesta. La información final se entrega al usuario mediante síntesis de voz utilizando TTS. El sistema también almacena la última asistencia emitida para mantener coherencia y continuidad en las indicaciones.

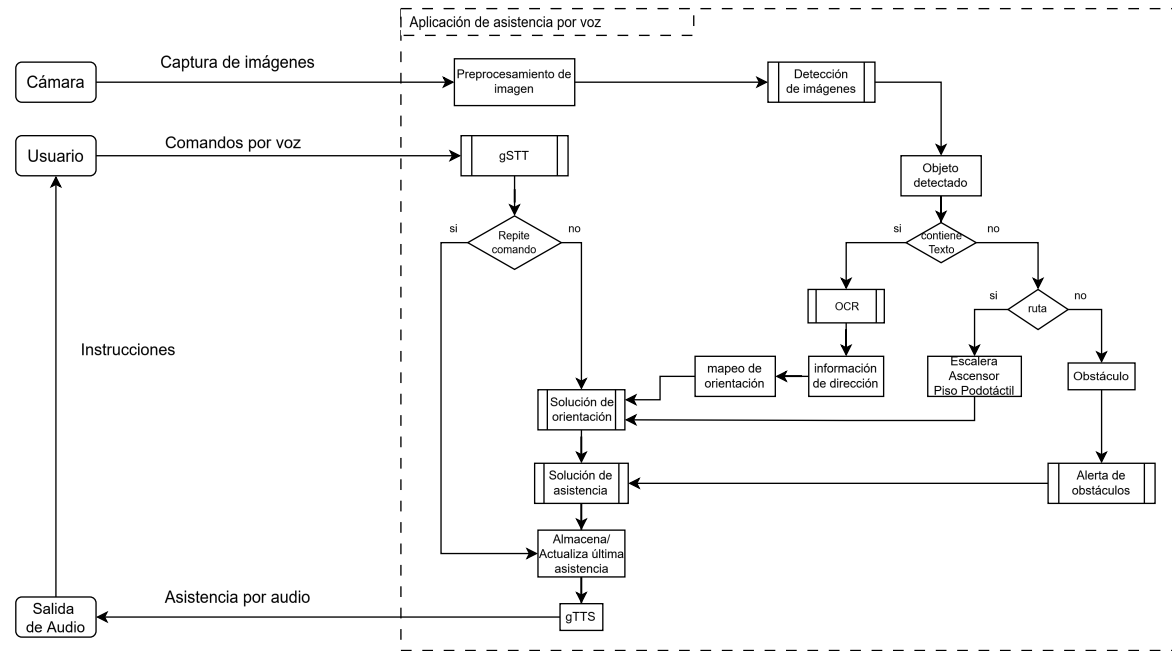


Figura 7: Diagrama de arquitectura

6.3. Enumeración de Módulos

Módulo	Propósito	Sección
Investigación del Usuario y Diseño de Producto (UX_PD)	Comprender el contexto de uso, necesidades y barreras de personas con discapacidad visual para diseñar una experiencia de interacción accesible y funcional.	7.1
Captura, Corrección y Preprocesamiento Visual (Ex_Im)	Obtener video del entorno en tiempo real desde el dispositivo móvil y procesarlo para su uso por los módulos de detección.	7.2
Detección y Dirección con modelo de <i>computer vision</i> (Det_OI + Dir_OI)	Detectar señales y obstáculos visuales, y entregar instrucciones para dirigir al usuario hacia su objetivo.	7.3
Interacción por Voz (Rec_Prompt + TaV_Dir)	Permitir que el usuario interactúe con el sistema usando comandos de voz y reciba retroalimentación auditiva personalizada.	7.4

Cuadro 4: Resumen de Módulos del Sistema AIMES

6.4. Matriz de Requisitos Funcionales y Módulos

A continuación se muestra la matriz donde se trabajará en completar cada requisito presentado:

Requisito	Módulo 1	Módulo 2	Módulo 3	Módulo 4
Captura de imágenes en tiempo real		X		
Modelo de detección de señales y texto del Metro			X	
Reconocimiento de texto a voz (comandos del usuario)				X
Orientación a través de voz para el usuario	X		X	X
Uso de cámara del móvil para detección		X		
Aplicación móvil para interacción	X			X
Auriculares o salida de audio para instrucciones				X
Hardware de desarrollo (PC, cámara, auriculares)	X	X	X	X
Software de desarrollo (OpenCV, Android Studio, etc.)	X	X	X	X

Cuadro 5: Matriz de trazabilidad entre requisitos y módulos del sistema

6.5. Revisión de Arquitectura de Diseño

La figura 7, correspondiente al diagrama de arquitectura, fue revisada con el fin de comprobar la coherencia entre los bloques definidos, los módulos y los requisitos del sistema. Se identificaron componentes de mayor complejidad —como *Preprocesamiento de imagen*, *Mapeo de orientación*, *Solución de orientación* y *Solución de asistencia*— que requieren pasos intermedios adicionales y una definición más precisa de sus funciones.

Durante esta revisión se constató que los módulos representan adecuadamente las intenciones del proyecto y permiten orientar el progreso en función de los hitos establecidos. No obstante, se enfatiza la necesidad de refinar o, en caso necesario, descartar bloques cuya implementación no aporte directamente a los objetivos de AIMES. De esta manera, la arquitectura mantiene su coherencia y escalabilidad a lo largo del desarrollo.

7. Módulos

A continuación se describen los módulos del sistema AIMES, detallando su propósito, alcance, dependencias, supuestos, restricciones y estructura funcional.

7.1. Módulo 1: Investigación del Usuario y Diseño de Producto (UX_PD)

Propósito

Comprender el contexto de uso, necesidades y barreras de personas con discapacidad visual para diseñar una experiencia de interacción accesible y funcional, a la vez de compilar diversas implementaciones que puedan ofrecer seguridad a la aplicación y al usuario de amenazas al sistema y a la experiencia del usuario.

Descripción-Desarrollo

Durante el desarrollo del módulo, se realiza una *investigación dura*, que consiste en recopilar la mayor cantidad de información de distintos tópicos de interés para integrar en el sistema AIMES. Gran parte de estas investigaciones son incluidas en el Estado-del-Arte que se seguirá actualizando hasta que concluya el módulo 1. Durante el desarrollo de los siguientes módulos, se comienza una *investigación ligera*, que consiste en la actualización pasiva del Estado-del-Arte, considerando en todo momento que el enfoque es extraer información que aporte a la implementación de los módulos y no a expandir el conocimiento. El criterio para determinar cuándo termina la etapa 1 es finalizada la estructura a continuación.

Alcance

Este módulo abarca desde la investigación inicial hasta la validación de un prototipo funcional de interfaz, incluyendo entrevistas, registro en terreno, definición de flujos de interacción, evaluación de una interfaz con retroalimentación auditiva, estudio de trabajos relacionados a la implementación de medidas de seguridad al usuario para proteger su dispositivo smartphone, evitar la interacción innecesaria con datos sensibles del usuario, permitir en un 99,99 % la disponibilidad de la aplicación e implementar métodos para mitigar riesgos definidos en sección 8.

Dependencias

Es un módulo inicial e independiente, pero sus resultados orientan el diseño y funcionamiento de los módulos técnicos posteriores. Es importante considerar que no se limita a lo que esté en el mercado, publicado o propuesto, sino también al conocimiento que portan los integrantes de esta memoria y cómo lo aportarán al mundo académico.

Supuestos:

- Acceso a personas con discapacidad visual para entrevistas.
- Posibilidad de realizar observaciones y pruebas en estaciones reales.
- Existencia de artículos para las tecnologías de interés.
- Existencia de distintas implementaciones para evaluar la mejor opción en términos de rendimiento, efectividad y eficiencia.

Restricciones:

- Tiempo limitado para el diseño iterativo y validación en terreno.
 - Disponibilidad de usuarios para pruebas durante el desarrollo.
 - Capacidad de artículos disponibles.
-

Estructura:

- *Entradas:* Insumos de investigación, entrevistas, registros de terreno.
- *Ejecución:* Análisis de contexto, definición de flujos, diseño de interfaz y evaluación.
- *Salidas:* Perfil de usuario, wireframes accesibles, prototipo funcional validado.

Hallazgos visita Biblioteca Central para ciegos y entrevista Don Iván

Durante la mañana del lunes 16 de junio se realizó una visita a la Biblioteca Central para Ciegos, la cual fue guiada por el coordinador general. Esta instancia permitió observar y comprender algunas de las prácticas cotidianas de las personas con discapacidad visual, así como recoger información clave para orientar el diseño del sistema AIMES.

En primer lugar, se identificó que las personas ciegas utilizan el lector de pantalla NVDA para acceder al sistema operativo Windows en computadores, lo que permite una navegación completamente auditiva. Esta interacción se realiza casi exclusivamente mediante teclado, aplicando técnicas de dactilografía y el uso de comandos y atajos, descartando el uso del mouse. En cuanto a dispositivos móviles, los usuarios utilizan los sistemas de accesibilidad integrados: VoiceOver en iPhone y TalkBack en Android. Además, se constató que la mayoría de las personas ciegas se conectan a clases impartidas por la Biblioteca a través de sus celulares, evidenciando una alta adopción del teléfono como dispositivo principal de interacción.

También se abordaron aspectos editoriales, destacándose que existe un estándar para la elaboración de libros accesibles, el cual regula el uso de colores, tipografías y tamaño de ilustraciones para personas con baja visión o ceguera. En relación con la movilidad en el entorno del Metro de Santiago, se destacó la importancia de los pisos podotáctiles como una guía esencial de desplazamiento, señalando que su referencia dentro del sistema a desarrollar no es redundante, sino una herramienta valiosa. Asimismo, se destacó la necesidad de que el sistema considere rutas alternativas o de emergencia, ya que las personas ciegas requieren planificación anticipada ante cortes o fallas de servicio. Finalmente, se reafirmó que el uso del bastón es habitual entre personas ciegas que se desplazan de forma independiente, por lo que el sistema debe diseñarse como un complemento a esta herramienta y no como un reemplazo.

Adicional a la visita a la biblioteca central para ciegos, se realizó una entrevista en terreno, específicamente en la línea 6 del Metro de Santiago, a Don Iván, una persona con discapacidad visual que se desplaza de forma independiente por la red de Metro. La conversación tuvo un enfoque exploratorio y empático, con el objetivo de comprender sus hábitos, dificultades y expectativas frente a una posible solución tecnológica como el sistema AIMES.

Don Iván comentó que se moviliza con frecuencia utilizando su bastón como principal herramienta de orientación. A pesar de su autonomía, expresó varias dificultades que enfrenta en el entorno del Metro. Una de las más recurrentes es la interacción con las pantallas táctiles públicas, particularmente las máquinas de recarga Bip!, donde no tiene certeza sobre en qué parte está tocando. En ese sentido, señaló que una aplicación que ofrezca asistencia para ubicar correctamente su dedo o para leer la pantalla sería una gran ayuda.

También mencionó que sería útil contar con información sobre los servicios cercanos dentro de las estaciones, como cajeros automáticos o tiendas (por ejemplo, Oxxo o Pronto), ya que en ocasiones necesita acceder a estos puntos, pero no tiene referencias claras sobre su ubicación. Asimismo, indicó que al momento de bajar del tren muchas veces no sabe si lo hizo al inicio, al medio o al final del andén, lo cual complica su orientación posterior dentro de la estación.

Otro aspecto que consideró relevante fue la posibilidad de que el sistema indique las salidas específicas del Metro, permitiéndole elegir una dirección concreta según su destino. Finalmente, sugirió que una cámara externa, independiente del teléfono y complementada con un altavoz, podría ser una opción más segura y eficiente para obtener información del entorno sin comprometer su dispositivo móvil.

Don Iván se mostró muy dispuesto a colaborar con el proyecto y manifestó interés en participar en futuras pruebas, así como en difundir la iniciativa en la asociación de personas ciegas a la que pertenece.

Requisitos específicos acorde al contexto

A partir de la información recopilada, se establecen los siguientes requisitos específicos para el diseño del sistema:

1. **Compatibilidad con lectores de pantalla móviles:** Los usuarios utilizan VoiceOver (iPhone) y TalkBack (Android) para acceder a aplicaciones en sus celulares. Por lo tanto, el sistema debe asegurar compatibilidad total con estos lectores de pantalla, sin interferir en su funcionamiento.
2. **Retroalimentación auditiva clara y precisa:** El sistema debe ofrecer instrucciones por voz mediante TTS, con indicaciones breves, oportunas y comprensibles en entornos ruidosos como el Metro.
3. **Referencia a pisos podotáctiles:** Se destacó que no es redundante mencionar los pisos podotáctiles, ya que funcionan como guía física en las estaciones. La aplicación debe ser capaz de detectar y comunicar su ubicación.
4. **Orientación hacia servicios cercanos dentro de la estación:** El sistema debe ofrecer al usuario información sobre la ubicación de servicios relevantes en estaciones de Metro, como cajeros automáticos, tiendas de conveniencia (Oxxo, Pronto, etc.), u otros puntos de interés.
5. **Soporte para navegación hacia salidas específicas:** El sistema debe permitir que el usuario elija una salida concreta de la estación (por ejemplo: hacia una calle o avenida), y que reciba instrucciones precisas para llegar a ella desde su posición actual.
6. **Orientación espacial al bajar del tren:** El sistema debe ser capaz de determinar y comunicar al usuario si descendió al inicio, medio o final del andén, para facilitar la planificación de su ruta dentro de la estación.
7. **Consideración de rutas alternativas ante emergencias:** Los usuarios ciegos planifican sus rutas con anticipación, por lo tanto, si una ruta se interrumpe, el sistema debe sugerir opciones de trayecto alternativas, para evitar desorientación y dependencia de terceros.
8. **Consumo energético bajo:** Dado que los usuarios confían en sus celulares durante largos trayectos, el sistema debe tener bajo consumo de recursos y operar de forma fluida sin agotar la batería del dispositivo.
9. **Uso de cámara externa y altavoz seguro:** Se recomendó explorar el uso de una cámara externa al teléfono (como en un wearable o bastón inteligente), junto con un altavoz externo, para mejorar la calidad de orientación sin comprometer la posición del celular.
10. **Asistencia al interactuar con pantallas táctiles de uso público:** Se recomendó integrar una función o módulo que ayude al usuario a entender dónde está apuntando con el dedo al interactuar con máquinas como las de carga Bip! o tótems informativos.

A continuación se muestra la Figura 8 comparando los requisitos ideales con los reales.



Figura 8: Diagrama de requisitos

Diagrama de funciones

A continuación, se presenta el diagrama del sistema a desarrollar, el cual permite descomponer la función general del producto en tareas más específicas que guían su desarrollo estructurado. Según Ulrich y Eppinger, un diagrama de funciones “representa las actividades que debe cumplir un producto y las relaciones entre ellas, sin hacer referencia directa a componentes físicos”, facilitando identificar soluciones técnicas para cada necesidad del usuario.[36]

En cada diagrama se destaca quienes serían los actores responsables de cada actividad/función.

La Figura 9 muestra una representación esquemática de las funciones ideales o deseables dentro del sistema AIMES, organizadas según sus objetivos de accesibilidad, navegación y soporte técnico.

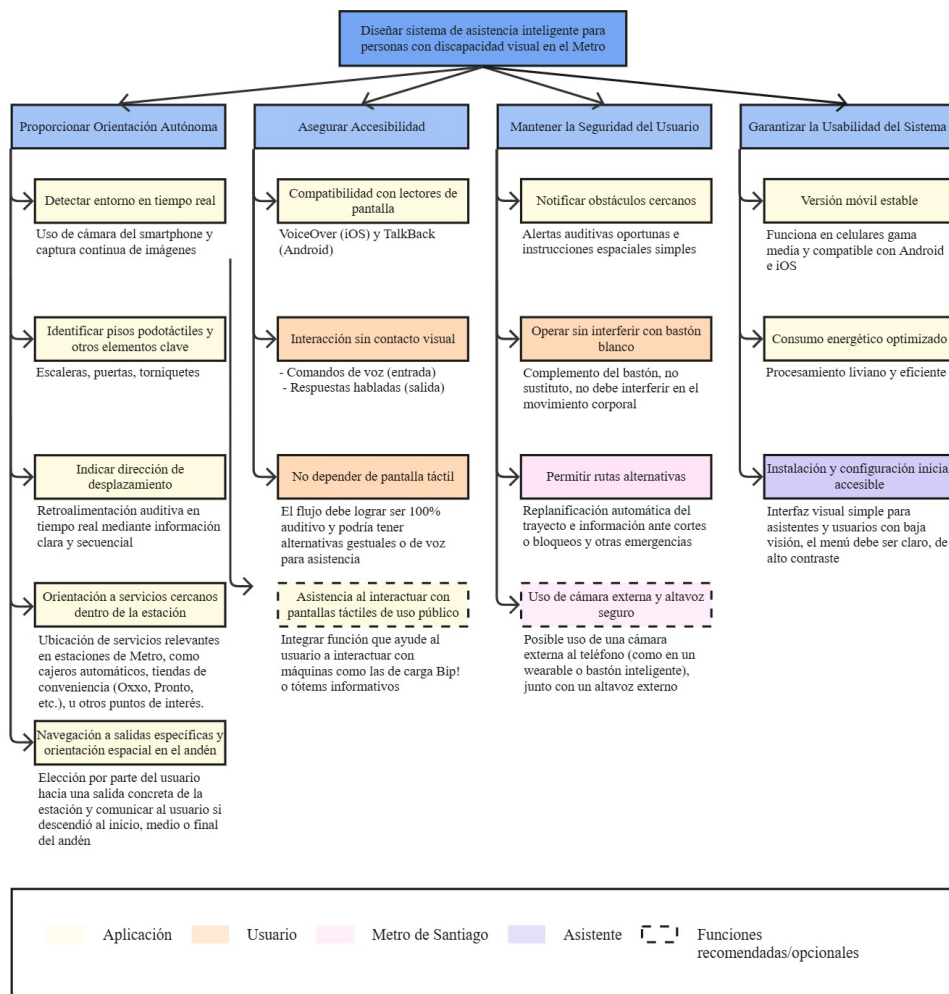


Figura 9: Diagrama de funciones ideales

Por otro lado, la Figura 10 muestra un esquema con las funciones indispensables para el sistema, este de igual manera se encuentra organizado según sus objetivos de accesibilidad, navegación y soporte técnico.

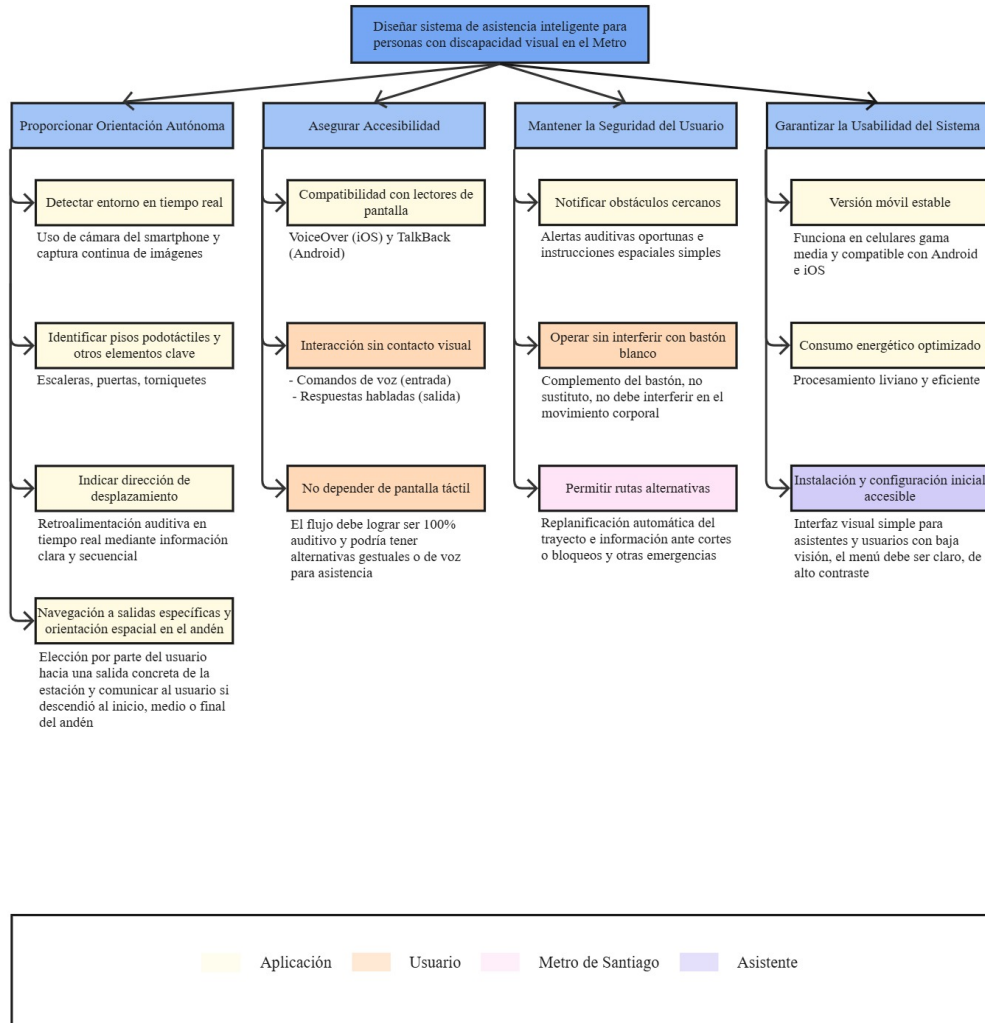


Figura 10: Diagrama de funciones reales

Diagrama de conceptos

A continuación, se presenta el diagrama de conceptos desarrollado para el sistema AIMES, el cual permite explorar distintas soluciones técnicas posibles para cumplir con las funciones clave del producto. Según Ulrich y Eppinger, un diagrama de conceptos “representa gráficamente las ideas o soluciones alternativas que pueden implementarse para lograr las funciones del producto”, ayudando a visualizar combinaciones viables de tecnologías antes de definir un diseño final. En este caso, el diagrama permite organizar y analizar propuestas asociadas a accesibilidad, orientación espacial, interacción con el entorno y eficiencia operativa.[36]

Es importante destacar que la Figura 11 muestra la situación ideal donde los conceptos no se ven necesariamente en el diseño de la interfaz, sino que en las funcionalidades y operación del sistema.

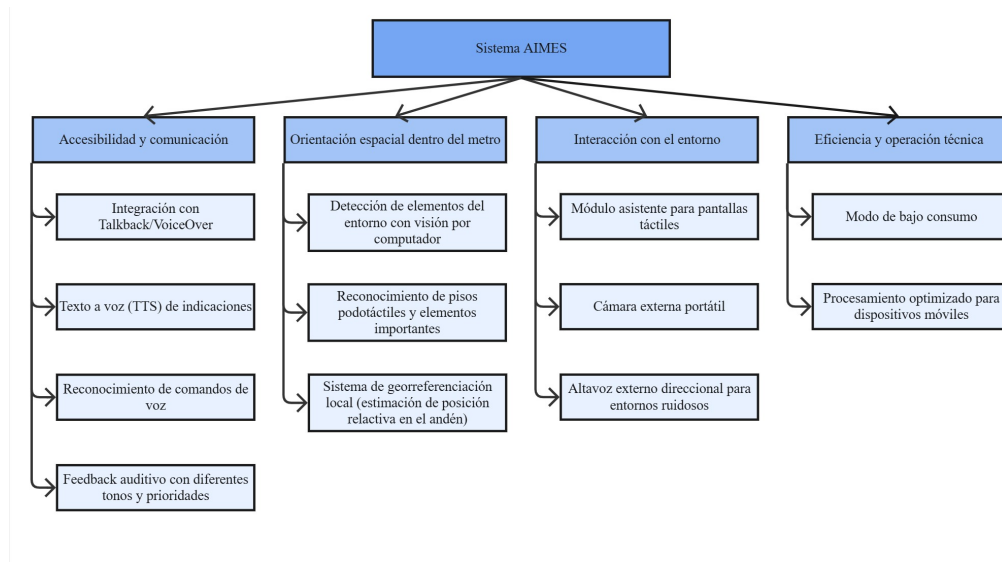


Figura 11: Diagrama de conceptos

Test de Usabilidad

Evaluar la facilidad de uso, comprensión de instrucciones auditivas y nivel de autonomía que proporciona AIMES a personas con discapacidad visual dentro del entorno de la línea 6 del Metro de Santiago.

Metodología del Test Tipo de prueba: Moderada, en entorno controlado (simulación de estación de Metro).

Usuarios: Personas con discapacidad visual (mínimo 3 con ceguera total y 3 con baja visión).

Duración estimada: 20 minutos por participante.

Formato: Individual, con observador silencioso y registro de voz.

Fase del prototipo: Funcional en MVP (Producto Mínimo Viable) (modo voz + navegación básica).

Tareas por Evaluar

1. Inicio del sistema por voz

- **Indicador de éxito:** El usuario logra activar el sistema sin ayuda.

2. Selección de destino (salida específica)

- **Indicador de éxito:** El usuario comprende cómo elegir una salida y confirma su selección por voz.

3. Seguimiento de instrucciones auditivas

- **Indicador de éxito:** El usuario sigue correctamente las instrucciones hasta un punto predeterminado (ej: torniquete o salida).

4. Detección de bloqueo y ruta alternativa

- **Indicador de éxito:** El usuario reacciona adecuadamente al aviso de bloqueo y sigue la ruta alternativa propuesta.

5. Reconocimiento del entorno inmediato (ej: pisos podotáctiles o escaleras)

- **Indicador de éxito:** El usuario reconoce verbalmente lo que el sistema le informa y reacciona en consecuencia.

- Tasa de éxito por tarea
- Errores cometidos y recuperaciones
- Tiempo promedio para completar cada tarea
- Necesidad de asistencia externa
- Observaciones cualitativas del comportamiento
- Medición del nivel de confianza del usuario

Al final del test, se aplicará una escala tipo Likert de 5 niveles con la siguiente pregunta:

¿Qué tan seguro(a) te sentiste usando la aplicación para orientarte dentro de la estación?

Opciones:

- Muy inseguro
- Inseguro
- Neutro
- Seguro
- Muy seguro

Además, se incluirá una pregunta abierta:

¿Qué elementos del sistema te generaron más confianza y cuáles te generaron dudas?

7.2. Módulo 2: Captura, Corrección y Preprocesamiento Visual (Ex_Im)

Propósito

Obtener video del entorno en tiempo real desde el dispositivo móvil y procesarlo para que sea interpretable por los módulos de detección.

Descripción-Desarrollo

Durante el desarrollo del Módulo 2, se deben establecer y aplicar diversas restricciones y criterios técnicos relacionados con las imágenes que se capturan y procesan en tiempo real desde el dispositivo móvil. Esto incluye parámetros como la orientación y campo de visión de la cámara, la calidad de imagen, la estabilidad de la captura y la calidad de textura de los objetos presentes en la escena.

En la **fase de captura de video (Milestone 2.1)**, se utilizará la librería OpenCV para implementar mecanismos de adquisición de video, evaluando su latencia y estabilidad, y asegurando su integración fluida con la aplicación móvil. Esta etapa define los requisitos mínimos de entrada para garantizar datos visuales consistentes y utilizables.

Durante la fase de **corrección de imagen (Milestone 2.2)**, considerando la calidad de textura de los objetos en las imágenes se abordarán interferencias visuales como stickers, manchas o rayas sobre la cámara mediante técnicas de reconstrucción visual, como autoencoders o GANs entrenados con un dataset específico. Estos métodos permiten recuperar o limpiar las imágenes defectuosas sin requerir intervención del usuario.

Finalmente, en la fase de **preprocesamiento (Milestone 2.3)**, las imágenes serán escaladas y ajustadas en color antes de ser canalizadas hacia los modelos de visión por computadora del Módulo 3. Este procesamiento se optimizará para ejecución en dispositivos móviles, garantizando eficiencia y precisión.

El usuario no debe notar las mitigaciones técnicas aplicadas. Al usuario se le debe comunicar la importancia de manipular adecuadamente el dispositivo durante la captura visual, para asegurar una calidad óptima en el análisis de imagen.

Alcance

Este módulo incluye el preprocesamiento de imágenes con OpenCV para realizar corrección de imágenes intervenidas mediante modelos generativos y preprocesamiento (escalado, ajustes de color) para alimentar eficientemente los modelos de detección.

Dependencias

Requiere del diseño de interfaz definido por el UX y acceso a datasets anotados para entrenamiento de modelos de corrección.

Supuestos:

- La cámara del dispositivo tiene resolución y estabilidad adecuadas.
- Disponibilidad de ejemplos reales de interferencias visuales.

Restricciones:

- Recursos computacionales limitados en el dispositivo móvil.
- Latencia mínima requerida para procesamiento en tiempo real.

Estructura:

- *Entradas:* Captura de video, datasets de entrenamiento.
 - *Ejecución:* Corrección de imagen, preprocesamiento, canalización hacia detección.
 - *Salidas:* Frames optimizados para procesamiento posterior.
-

7.3. Módulo 3: Detección de Objetos y Dirección (Det_OI + Dir_OI)

Propósito

Detectar señales y obstáculos visuales en tiempo real, y generar instrucciones espaciales claras para orientar al usuario.

Descripción-Desarrollo

El desarrollo del Módulo 3 inicia con la definición de métricas clave de evaluación, alineadas con los criterios establecidos en la investigación exploratoria del Módulo 1. Estas métricas, como el *Mean Average Precision (mAP)* y la velocidad de inferencia (*FPS*), permiten cuantificar la precisión y eficiencia del modelo de visión por computadora en condiciones reales, y establecer comparaciones frente a soluciones de referencia en la literatura.

Durante la **fase de entrenamiento (Milestone 3.1)**, se construirá un dataset personalizado a partir de imágenes y videos del entorno del Metro, con énfasis en la anotación precisa de señales y obstáculos. Se aplicarán técnicas de *data augmentation* para robustecer el modelo ante variaciones en condiciones de iluminación, ángulos y oclusiones, y se validará el rendimiento utilizando las métricas previamente definidas. El dataset estará almacenado en la plataforma de Roboflow para el libre acceso.

En la **fase de detección y clasificación (Milestone 3.2)**, se implementarán modelos optimizados para dispositivos móviles, seleccionando únicamente las clases visuales más relevantes para la experiencia del usuario. Se priorizarán aquellos elementos que influyan directamente en la navegación (como señaléticas con flechas direccionales, salidas, puertas, pisos podotáctiles y otros), y se evaluará la calidad de los resultados mediante visualización en tiempo real mediante *overlays* integrados.

Finalmente, en la **fase de dirección espacial (Milestone 3.3)**, se diseñarán e implementarán mecanismos para estimar la posición relativa de los objetos detectados en un sistema de cuadrantes simples encima del frame actual. A partir de esta estimación, el sistema generará instrucciones claras y comprensibles (ej. “gira a la izquierda”, “sigue derecho”) para guiar al usuario. Estas instrucciones se validarán con una serie de pruebas reales desde usar videos grabados en estaciones del Metro hasta pruebas en terreno a documentar.

A lo largo del desarrollo, se pondrá énfasis en la eficiencia computacional, optimizando el rendimiento del modelo para su ejecución fluida en dispositivos móviles. Esto implica un equilibrio entre precisión y consumo de recursos, en línea con las limitaciones de hardware típicas de smartphones, y aprovechando el conocimiento técnico derivado del Módulo 1 para informar decisiones de implementación.

Alcance

Incluye la captura y anotación de dataset para entrenar un modelo de detección de objetos (YOLO, MobileViT o MobileSAM), generación de overlays, construcción, selección y cálculo de direcciones según la posición relativa de los elementos detectados.

Dependencias

En base a los módulos 1 y 2, *UX_PD* y *Ex_Im* respectivamente, se consideran los perfiles y frames preprocesados para satisfacer las necesidades del usuario. Esta satisfacción se realiza construyendo las órdenes de orientación considerando los distintos objetos de interés que conforman la escena y son detectados en el frame actual, permitiendo la ubicación del usuario y un posible camino a atravesar.

Supuestos:

- Disponibilidad de datasets anotados en escenarios reales del Metro.
- Funcionamiento de modelo de *computer vision* con métricas adecuadas en móviles.

Restricciones:

- Limitaciones de procesamiento y consumo energético.
- Alta variabilidad visual y movimiento de personas en estaciones.

Estructura:

- *Entradas:* Frames preprocesados, modelo de *computer vision* entrenado.
 - *Ejecución:* Detección y clasificación, generación de dirección espacial.
 - *Salidas:* Instrucciones visuales y espaciales priorizadas.
-

7.4. Módulo 4: Interacción por Voz (Rec_Prompts + TaV_Dir)

Propósito

Facilitar la interacción sin contacto visual ni táctil, permitiendo comandos por voz y retroalimentación auditiva adaptada al entorno.

Descripción-Desarrollo

Al igual que en el Módulo 2, y considerando el propósito específico de este módulo, se incorporarán tecnologías de reconocimiento y síntesis de voz (*speech-to-text* y *text-to-speech*) para mejorar la accesibilidad e inclusión de la aplicación. El desarrollo se enfocará en garantizar una comunicación fluida y efectiva entre el usuario y el sistema, tanto de manera activa (comandos e interacciones) como pasiva (mensajes e instrucciones auditivas), en entornos urbanos ruidosos.

Durante la **fase de recepción del comando por voz (Milestone 4.1)**, se integrará la funcionalidad de STT que sea capaz de interpretar la intención del usuario incluso bajo condiciones acústicas complejas. Se evaluará la capacidad del sistema para operar con ruido de fondo (como el del Metro de Santiago), y se establecerán mecanismos de validación que garanticen que las instrucciones vocales son correctamente interpretadas.

En la **fase de asociación con la escena visual (Milestone 4.2)**, se implementará un método de determinación de escena mediante las clases prioritarias detectadas (del Módulo 3) y los comandos del usuario, habilitando la generación de direcciones espaciales contextualizadas. Cada acción del sistema será confirmada mediante retroalimentación auditiva clara, para asegurar que el usuario comprenda y valide cada decisión del sistema.

En la **fase de retroalimentación auditiva (Milestone 4.3)**, se implementará un motor TTS adaptado al uso móvil, con construcción de frases *empáticas* (simples de estructura, asertivas al informar y de fácil comprensión), de vocabulario técnico adecuado y duración optimizada. Además, se configurarán parámetros de salida auditiva para que funcionen correctamente en distintos contextos de uso, incluyendo audífonos o altavoces del dispositivo.

Los **criterios pasivos** establecidos incluirán límites en la duración de los mensajes, claridad del lenguaje, adecuación del vocabulario técnico y ritmo del habla. Los **criterios activos** se centrarán en permitir una interacción funcional que abarque el control de la aplicación, modificación de parámetros de captura y el intercambio eficiente de información.

Durante las etapas de prueba, se evaluará el desempeño del sistema en entornos reales del Metro de Santiago, donde se considera una amplia variedad de fuentes de ruido, interrupciones y condiciones ambientales. Estas pruebas serán esenciales para medir la flexibilidad y eficacia del sistema en situaciones de uso cotidiano, así como para garantizar una orientación efectiva y empática al usuario dentro de la infraestructura.

Alcance

Considera desde la integración de STT y TTS a través de la misma librería (como PyTTS para Python o la API nativa de TTS de Android), reconocimiento de voz e interpretación de intención, mapeo de las distintas detecciones visuales para la construcción de respuestas auditivas que entregan al usuario los mensajes de orientación mencionados en el módulo 3.

Dependencias

Depende del diseño de experiencia definido en UX_PD y de las detecciones generadas por Det_OI.

Supuestos:

- Disponibilidad de librerías de STT y TTS optimizadas para dispositivos móviles.
- El usuario utilizará auriculares o el sistema contará con salida de audio clara.

Restricciones:

- Condiciones de ruido ambiente que afectan el reconocimiento de voz.
- Latencia aceptable para mantener la conversación fluida.

Estructura:

- *Entradas:* Comandos de voz, elementos visuales detectados.
 - *Ejecución:* Reconocimiento de voz, mapeo con contexto visual, generación de respuesta.
 - *Salidas:* Respuestas auditivas claras, empáticas y contextualizadas.
-

8. Diseño de Interfaces

El diseño de interfaces en AIMES se centra en construir una experiencia accesible, empática y funcional para personas con discapacidad visual, priorizando la autonomía del usuario en entornos de alta demanda como estaciones de Metro. Dado que los usuarios objetivo no interactúan con la interfaz de manera visual, el enfoque se basa en la interacción auditiva y por voz, adaptada a distintos niveles de discapacidad visual.

8.1. Principios de Diseño

El desarrollo de la interfaz se guía por los siguientes principios:

- **Accesibilidad:** Toda funcionalidad debe estar disponible mediante entrada y salida por voz, sin requerir visión ni interacción táctil compleja.
- **Claridad y simplicidad:** Las respuestas deben ser breves y fácilmente comprensibles.
- **Retroalimentación continua:** El sistema debe entregar confirmaciones auditivas claras tras cada acción, hasta 2 repeticiones de confirmación son dadas por defecto; Si el usuario no confirma y entrega una nueva acción al sistema, la solución prosigue con esa nueva acción dada.
- **Personalización:** El usuario podrá configurar parámetros como volumen de voz, velocidad del habla, entre otros.

8.2. Flujos de Interacción

La interfaz (Ver Fig.12) se organiza en flujos de voz que responden a los siguientes ciclos:

1. **Inicio del sistema:** Activación por comando de voz.
2. **Captura y análisis:** Toma de imagen, detección de objetos y evaluación espacial.
3. **Entrega de instrucciones:** El sistema informa la presencia y dirección de objetos relevantes.
4. **Confirmación del usuario:** El sistema puede requerir o entregar confirmación auditiva, de hasta 2 repeticiones (por defecto).
5. **Adaptación de contenido:** El sistema ajusta el feedback según personalización.

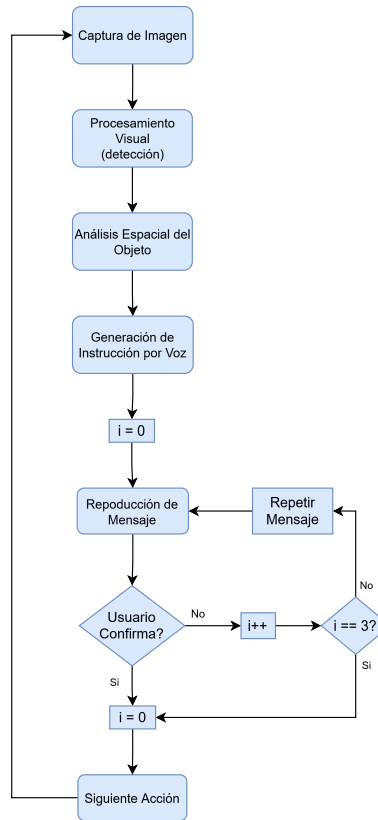


Figura 12: Flujo de Interacción de Interfaz

8.3. Modelo de navegación

El proceso comienza con un comando de activación como “Iniciar guía” (Ver Fig.13), tras lo cual el sistema proporciona una breve explicación sobre su funcionamiento. Si el usuario desea omitir esta introducción, puede utilizar el comando “Omitir”, y el sistema quedará a la espera de nuevas instrucciones.

Una vez recibido un comando, el sistema confirma si este fue correctamente comprendido (Ver Fig.14). En caso afirmativo, se inicia el proceso de navegación, donde el usuario puede utilizar uno de los siguientes comandos principales:

1. “Me dirijo a..., estoy en...”: para indicar origen y destino.
2. “Quiero utilizar escaleras”: para solicitar preferencia por escaleras.
3. “Quiero ir a la salida...”: para solicitar orientación hacia una salida específica.

Si el sistema no detecta señales, escaleras u otros elementos necesarios para ejecutar la navegación correctamente, se genera una alerta o mensaje de error auditivo, informando al usuario sobre la situación y sugiriendo posibles alternativas (Ver Fig.15).

Este modelo de navegación es secuencial y guiado, lo que evita que el usuario tenga que memorizar múltiples comandos o rutas complejas. Además, comandos como “Configuraciones” permiten ajustar parámetros como el volumen o la velocidad de lectura, adaptándose así a las preferencias del usuario.

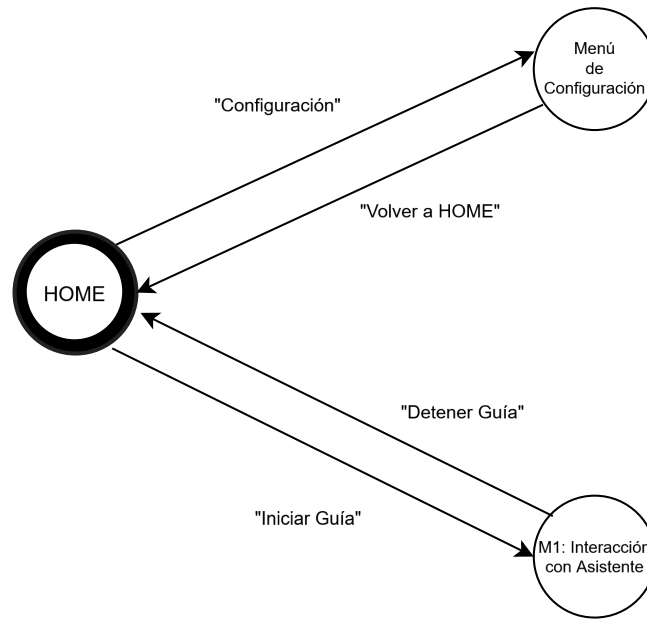


Figura 13: Modelo de Navegación: Inicio de guía o asistencia

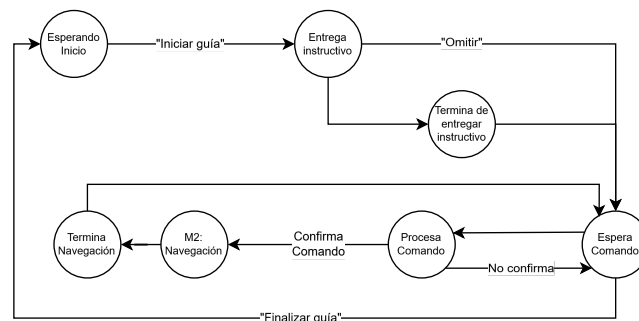


Figura 14: Modelo de Navegación: Inicio de guía o asistencia

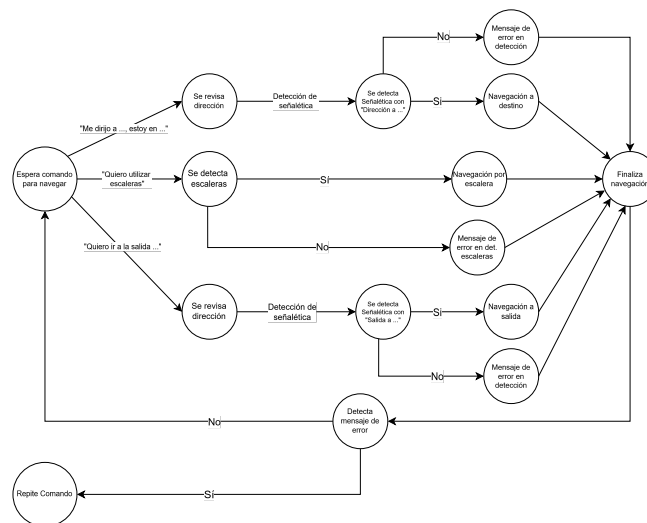


Figura 15: Modelo de Navegación: Inicio de guía o asistencia

8.4. Componentes de la Interfaz

8.4.1. Interfaz de Usuario

- **Menú:** Permite activar funciones, recibir ayuda o configurar preferencias mediante comandos hablados, tales como: volumen, repeticiones y velocidad de habla.
- **Motor TTS (texto a voz):** Convierte respuestas del sistema en instrucciones auditivas claras.
- **Módulo STT (voz a texto):** Transforma las instrucciones habladas del usuario en comandos para el sistema.
- **Botones Generales:** Corresponden a los elementos táctiles que aparecen en pantalla y permiten activar funciones clave como instrucciones, menú, preferencias o ayuda. Están diseñados con alto contraste, etiquetas claras y tamaño amplio para facilitar su uso por personas con baja visión o en entornos donde la voz no sea viable, se integran como alternativa como forma complementaria a la interacción por voz.

A continuación, se presenta el diseño preliminar de la interfaz de AIMES, el cual fue desarrollado siguiendo los principios de accesibilidad establecidos por las WCAG 2.1.

El diseño contempla tres versiones iterativas, cada una de las cuales responde a distintos niveles de refinamiento visual y funcional:

La Figura 16 muestra una representación esquemática de la **interfaz de usuario** de AIMES, propone una estructura minimalista con botones de gran tamaño, bajo nivel de complejidad y contraste básico.



Figura 16: Interfaz Gráfica AIMES

La Figura 17 introduce funciones clave como la visualización de estados y la inclusión de íconos accesibles que refuerzan el contenido textual.



Figura 17: Interfaz Gráfica AIMES 2

La Figura 18 se perfila como la más robusta visualmente, incorporando un fondo negro, íconos a color de alto contraste y una jerarquía visual clara, lo que favorece su uso por personas con baja visión.

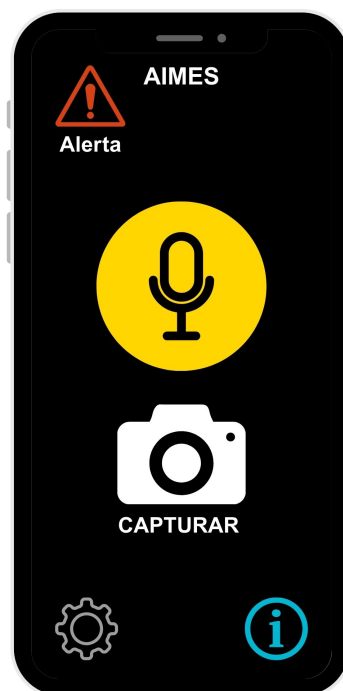


Figura 18: Interfaz Gráfica AIMES 3

Por otra parte, es importante mencionar que el botón de alerta está diseñado para ofrecer respuestas rápidas y adaptadas a distintas situaciones de urgencia.

Las opciones desarrolladas son las siguientes:

- **Opción 1:** Envío automático de un mensaje predefinido con la ubicación en tiempo real a un contacto previamente configurado, confirmando la acción mediante retroalimentación auditiva.
- **Opción 2:** Incorporación de dos botones, uno para contactar directamente a un contacto predefinido y otro para llamar a servicios de emergencia como policía, bomberos o ambulancias.
- **Opción 3:** Inclusión de mensajes predeterminados con nombre del usuario y ubicación GPS, además de la posibilidad de personalizar mensajes para situaciones de menor urgencia.
- **Opción 4:** Combinación de las funcionalidades anteriores, incorporando 3 botones, contacto predefinido, servicios de emergencia y mensajes predeterminados con interfaz accesible para su selección.

A continuación se muestra en la figura 19 la interfaz gráfica de las opciones propuestas.

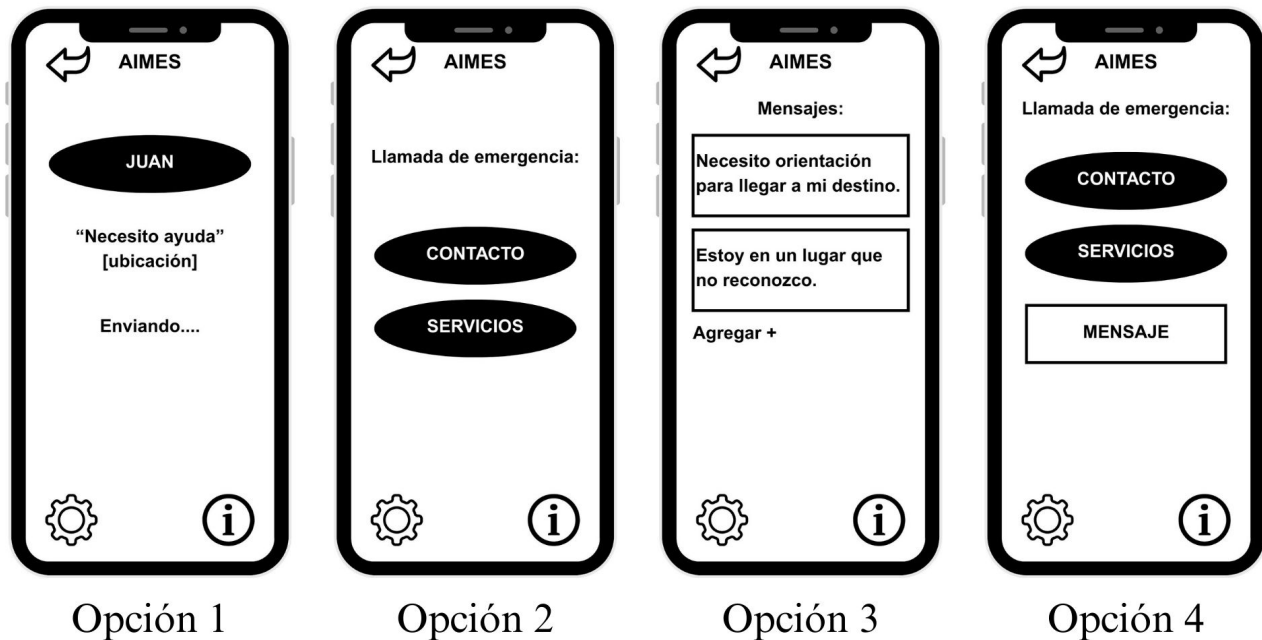


Figura 19: Interfaz Gráfica Botón de Alerta

El tamaño de los íconos fue definido a partir de las recomendaciones de diseño inclusivo, estableciendo un mínimo de 40px para facilitar su visualización, y manteniendo un espaciado adecuado entre elementos interactivos para prevenir errores táctiles.

La elección de colores, por su parte, se fundamentó en la necesidad de asegurar contrastes superiores a 4.5:1, recomendados por WCAG, y en evitar combinaciones problemáticas para personas con daltonismo.

A continuación, se muestra en la Figura 20 la paleta de colores utilizada junto a sus respectivos códigos.



Figura 20: Paleta de colores Interfaz Gráfica AIMES 3

Se debe recalcar que estos **diseños son preliminares**, realizados en base a la investigación y hallazgos recientes, por lo tanto, están sujetos a cambios, es necesario recibir retroalimentación de una mayor cantidad de usuarios para definir el diseño final de la interfaz.

8.4.2. Interfaz de Desarrollador

Además de la interfaz dirigida al usuario final, se ha diseñado una **interfaz provisoria para uso de desarrolladores**, con el objetivo de facilitar el testeo y validación de los módulos implementados durante las etapas tempranas del proyecto.

Esta interfaz no está orientada a la usabilidad por parte de personas con discapacidad visual, sino a brindar acceso directo a métricas internas clave del sistema.

- **Vista de cámara en tiempo real:** Muestra el video capturado con los *bounding boxes* de los objetos detectados.
- **Métricas de rendimiento visual:** Incluye el número de **FPS**, precisión del modelo (mAP estimado) y clases detectadas por frame.
- **Métricas de rendimiento de voz:** Visualización de la latencia del módulo STT, confirmación de intención y tiempo de respuesta en TTS.

Esta interfaz está implementada dentro de la misma aplicación, pero en una vista distinta accesible solo para testeo interno. El diseño incluye elementos visuales mínimos, optimizados para revisión técnica más que para experiencia de usuario, como lo pueden ver en la figura 21.



Figura 21: Interfaz de Desarrollador AIMES

8.5. Justificación del Diseño

Este diseño se basa en principios de accesibilidad universal y fue ideado tras una serie de investigaciones de aplicaciones que ya trabajaban nuestra problemática principal 3.5, además del análisis de los usuarios finales. Por esto, se prioriza la experiencia auditiva, considerando limitaciones reales como ruido ambiental, señalética deficiente y complejidad espacial.

La inclusión de una **interfaz de desarrollador** se justifica por la necesidad de contar con una herramienta que permita validar, depurar y medir el rendimiento de los módulos implementados en dispositivos móviles. Dado que la interfaz del usuario final está orientada a la simplicidad y no presenta información de sistema, esta vista paralela resulta fundamental para verificar métricas como latencia, exactitud de detección, carga computacional o errores en ejecución, antes de su integración final.

8.6. Resultados finales de Interfaz de Usuario

La presente sección expone los resultados obtenidos en el desarrollo de la interfaz de usuario del sistema AIMES, destacando los aspectos de diseño, accesibilidad y usabilidad logrados durante el proyecto. Estos resultados se fundamentan en los requerimientos definidos previamente, así como en los principios de diseño centrado en el usuario aplicados a personas con discapacidad visual.

A continuación, en la figura 22 se presenta el diseño actual de la interfaz de usuario de la aplicación, junto con el desglose de cada uno de sus componentes.

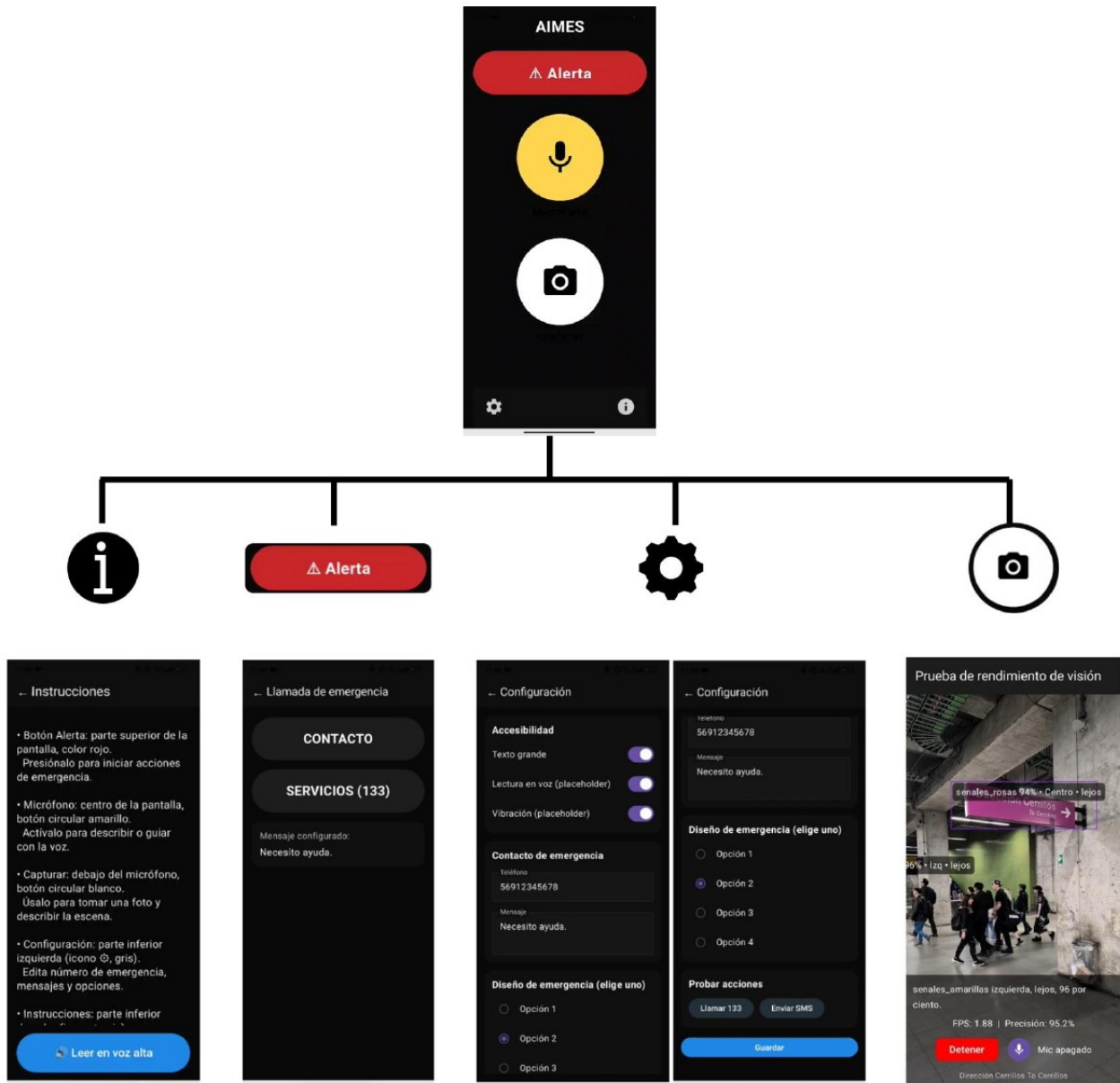


Figura 22: Interfaz de Usuario y sus componentes

9. Revisión de Prototipo

9.1. Contexto General del Prototipo

El prototipo, contemplando los módulos descritos en la sección 5, es abarcado desde tres aristas principales:

1. Confección de Dataset:

Mediante la toma y compilación de imágenes de la infraestructura de Metro de Santiago, se realiza un dataset que será almacenado en la plataforma de Roboflow. Se crean dos variantes: Señales-Detección (que formatea las imágenes para modelos de detección) y Señales-Segmentación (que formatea las imágenes para modelos de segmentación). A continuación, se indican las 13 clases que serán capturadas para entrenar los modelos de CV:

- a) Ascensores
- b) Entrada (entrada de torniquetes a la estación)
- c) Escaleras Mecánicas
- d) Escaleras Normales
- e) Estación (logo de la estación en las paredes de los andenes)
- f) Podo Círculo (suelo indicativo de intersección de líneas de pisos podotáctiles)
- g) Podo Línea (líneas podotáctiles)
- h) Señales Amarillas (indicativas de la línea 2)
- i) Señales Azules (indicativas de la línea 4)
- j) Señales Cafés (indicativas de la línea 3)
- k) Señales Rojas (indicativas de la línea 1)
- l) Señales Rosas (indicativas de la línea 6)
- m) Señales Verdes (indicativas de la línea 5)

2. Entrenamiento de Modelos de CV:

Una vez completada la etapa de etiquetado, las dos variantes del dataset son utilizadas para el entrenamiento de modelos de visión computacional bajo YOLOv11, en sus versiones orientadas a *detección* y *segmentación de instancias*. El entrenamiento se realiza en entorno de Google Colab con GPU dedicada, siguiendo una metodología iterativa que incluye: división del dataset en proporciones 92/5/3 para entrenamiento, validación y prueba; ajuste de hiperparámetros como tamaño de imagen y *batch size*; y el uso de métricas estándar como mAP@50, mAP@50-95, F1-score y *latency* sobre dispositivo móvil.

Se consideran los modelos de detección para realizar un aporte exploratorio para variaciones del modelo YOLOv11n.

Por otro lado, la integración del modelo de segmentación YOLO11n-segmentation, se justifica para el uso de máscaras devueltas por el modelo para el ajuste del algoritmo que oriente al usuario a través de las detecciones hechas en los ambientes de prueba.

3. Desarrollo de Aplicación Móvil:

El desarrollo de la aplicación móvil se realiza de forma nativa en Android Studio, utilizando Kotlin como lenguaje principal. Esto permite integrar directamente los modelos de visión computacional exportados en formato `.tflite` o `.onnx`, habilitando inferencia en tiempo real mediante aceleración por hardware cuando está disponible. La aplicación incorpora un flujo de interacción basado en voz, operando con las APIs nativas de Android para reconocimiento de voz (`SpeechRecognizer`) y síntesis de voz (`TextToSpeech`), garantizando retroalimentación auditiva accesible sin depender de conexión a internet.

La interfaz es desarrollada bajo los lineamientos de accesibilidad de Android y WCAG 2.1, con navegación asistida por comandos hablados, contraste visual adecuado y soporte completo para usuarios con ceguera o baja visión. Los módulos principales integrados en esta fase incluyen: (1) captura y preprocesamiento de imagen desde la cámara del dispositivo, (2) ejecución del modelo de CV para detección y segmentación de señales, (3) generación de instrucciones auditivas orientadas a movilidad dentro de estaciones, y (4) registro de métricas internas para validación en terreno.

9.2. Dataset

La tarea de captura de imágenes a lo largo del desarrollo del proyecto, concluye en el dataset presente en [37], construido por el equipo a cargo de AIMES almacenado en la plataforma Roboflow en formato de modelos de detección. El dataset presenta la siguiente distribución de instancias etiquetadas por clase:

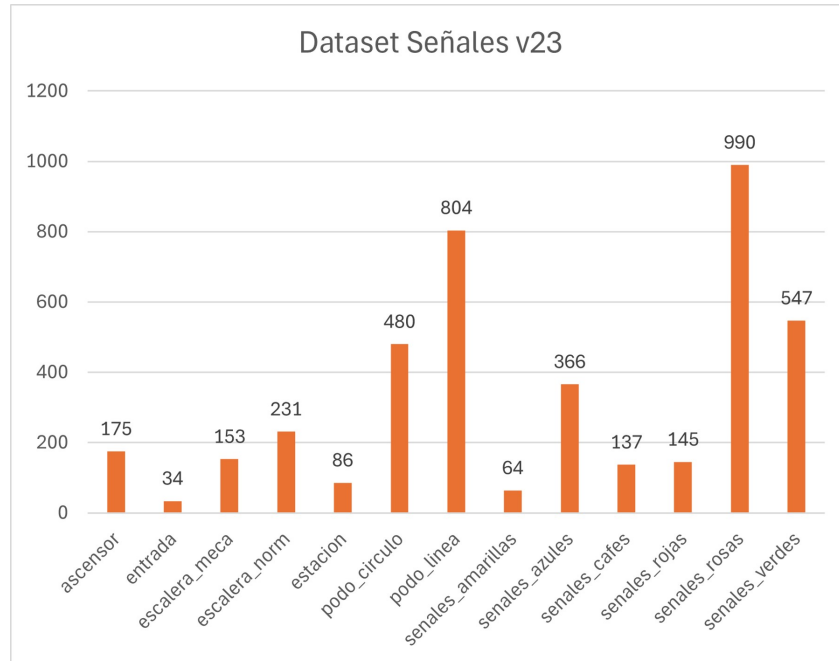


Figura 23: Distribución de instancias etiquetadas para las clases presentes en el dataset de señales-detección.

Por otro lado, el dataset de la figura 24 corresponde a una distribución ajustada de las imágenes, pero en formato para modelos de segmentación que se puede encontrar en [38]. A continuación, su distribución de instancias etiquetadas por clase:

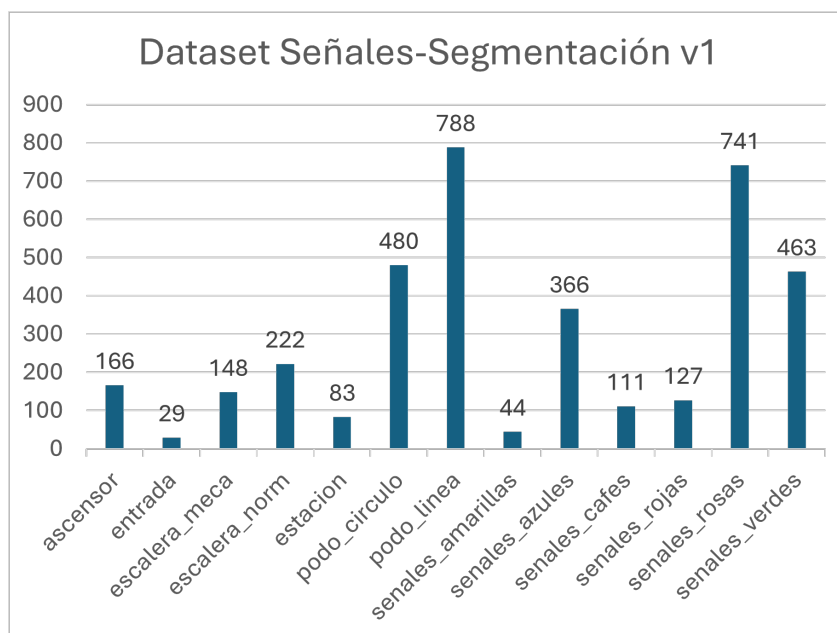


Figura 24: Distribución de instancias etiquetadas para las clases presentes en el dataset de señales-segmentación.

De la figura 23, se aprecia que es la versión 23 del dataset que ha sido construido desde 2023 para el propósito de AIMES. A día de la confección de este avance de hito, la distribución de las etiquetas indica un claro desequilibrio que beneficia las clases como 'señales rosadas', predominantes en la línea 6 del Metro de Santiago. Por otro lado, para 24, la versión 1 fue específicamente creada para la integración de un modelo de segmentación, por lo que es un repositorio independiente y reciente en Roboflow.

9.3. Entrenamiento Modelos CV

El equilibrio antes mencionado, puede afectar en los entrenamientos de los modelos de *computer vision* que sean utilizados por el equipo, los cuales consisten actualmente de:

Modelos de Detección

- YOLOv11n
- YOLOv11n con agregado de capa P2.
- YOLOv11n con agregado de capas basadas en el transformer RepViT.

Los modelos anteriores tienen diferencias en la cantidad de parámetros utilizados como en qué tipo de capas son aplicadas para cada paso en el entrenamiento. La figura 25 indica cantidades y el resumen de precisión en base al mAP.

Característica	YOLOv11n + p2	YOLOv11n + RepViT	YOLOv11n Tradicional
Backbone	C2f-ELAN	C2f-ELAN	C2f-ELAN
Configuración	Agrega escala P2 y bloques C3k2 para objetos pequeños.	Backbone tipo RepViT con bloques C3k2 y SPPF.	Por defecto
Precisión mAP@0.5 All Classes	0.56	0.631	0.698
Parámetros estimados	~2.74 millones	> 5 millones	~2.7 millones
Épocas	100	100	100

Figura 25: Comparación de modelos: parámetros, arquitecturas y resultados de mAP obtenidos.

Utilizando la librería de Ultralytics, los modelos son entrenados con el dataset y se obtienen las siguientes estadísticas desde el *trainer*:

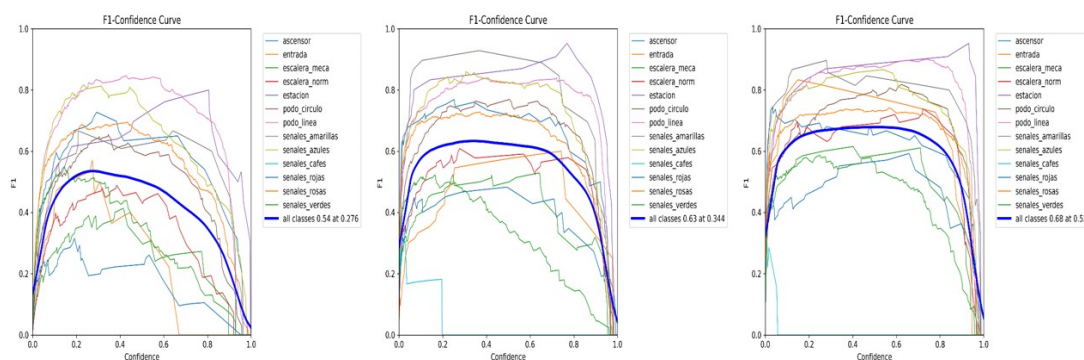


Figura 26: Resultados de Confianza F1. En orden de izquierda a derecha: YOLOv11n + P2, YOLOv11n + RepViT y YOLOv11n + Tradicional.

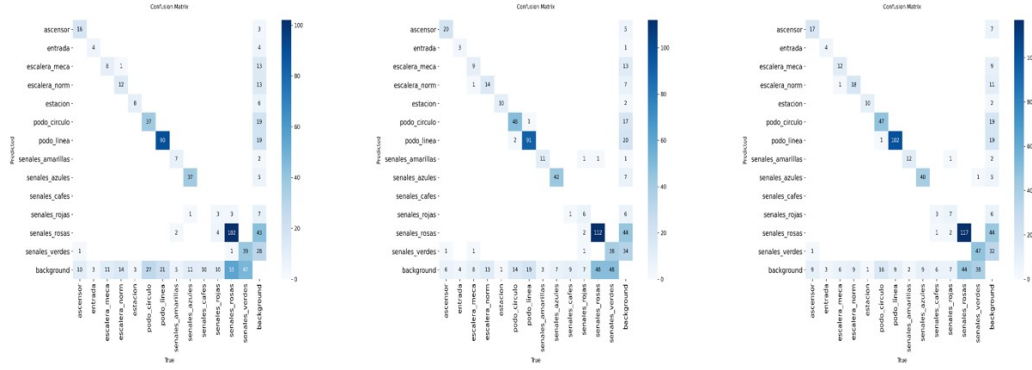


Figura 27: Resultados de Matriz de Confusión. En orden de izquierda a derecha: Yolov11n + P2, Yolov11n + RepVit y Yolov11n + Tradicional.

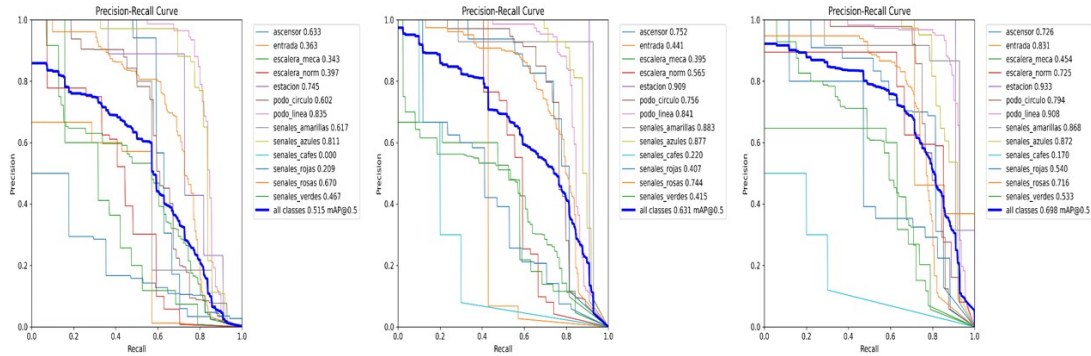


Figura 28: Resultados de Precision-Recall. En orden de izquierda a derecha: Yolov11n + P2, Yolov11n + RepVit y Yolov11n + Tradicional.

Los resultados de las métricas F1 (Figura 26), Matriz de Confusión (Figura 27) y Precision-Recall (Figura 28) evidencian un comportamiento consistente respecto al desequilibrio del dataset. Las clases con menor representación, como señales cafés o verdes, obtienen menores valores de F1 y mayor confusión. En contraste, las clases más numerosas muestran un mejor rendimiento en precisión y recall, lo que confirma que la distribución de instancias es un factor crítico para el desempeño general de los modelos.

Modelo de Segmentación

Se utiliza el siguiente modelo de segmentación entrenado con el dataset en el formato del mismo tipo:

- YOLOv11n-Seg (versión nano con cabeza de segmentación)

En la tabla 29 se aprecia que fue entrenado con características similares a los modelos de detección.

Característica	YOLOv11n-Segmentación
Backbone	C2f-ELAN
Configuración	Por defecto
Precisión mAP@0.5 All Classes	0.715
Parámetros calculados	~2.900.000
Image Size	512
Épocas	100

Figura 29: Parámetros, arquitecturas y resultados de mAP obtenidos de YOLO11n-segment.

Los parámetros de medición obtenidos corresponden a las figuras 30, 31 y 32.

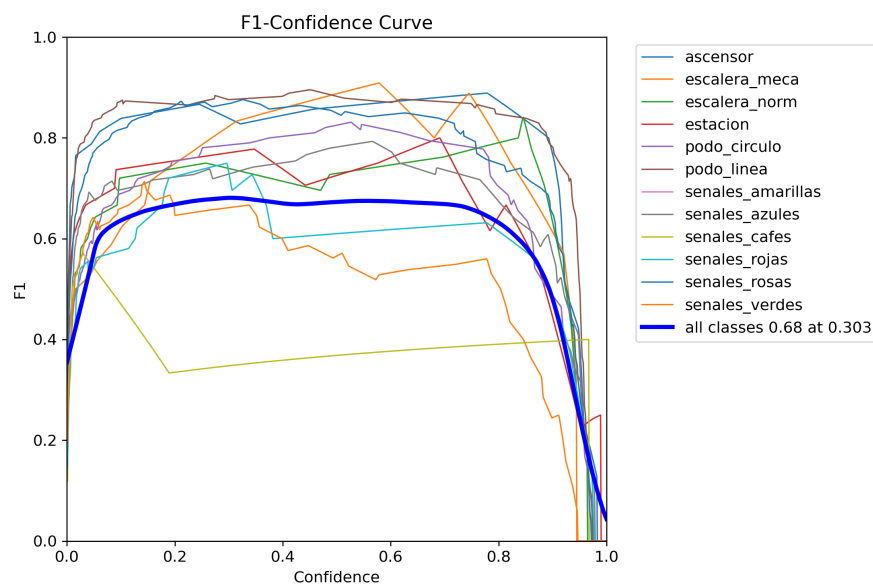


Figura 30: Curva de F1-score por *confidence threshold* del modelo YOLOv11n-Seg durante validación.

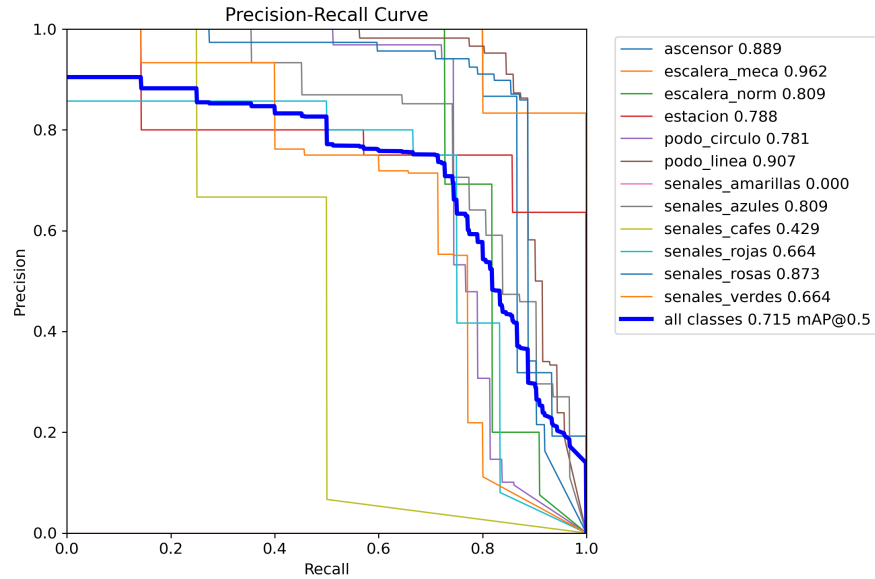


Figura 31: Curva Precision-Recall (PR) del modelo YOLOv11n-Seg.

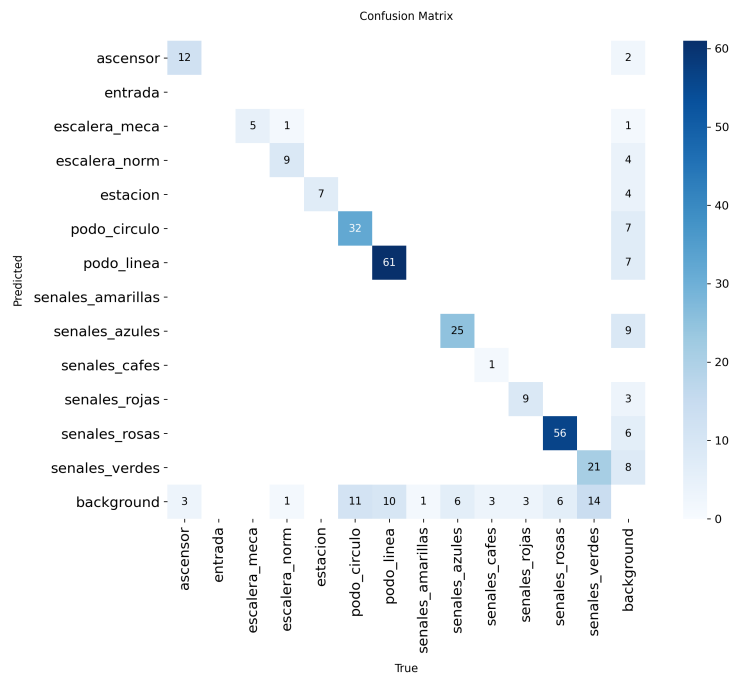


Figura 32: Matriz de confusión resultante del modelo YOLOv11n-Seg una vez finalizado el entrenamiento.

De ambos tipos de modelos de CV, se puede apreciar que hay buena respuesta para el dataset creado. Sin embargo, se decide avanzar con el modelo de segmentación para sacar provecho a los resultados máscara que se obtengan de él.

9.4. Interface Móvil

Para la interfaz móvil, se buscaba inicialmente cumplir con las interacciones descritas en el diagrama 15 y aproximarse al diseño conceptual mostrado en la figura 18.

En una primera etapa se desarrolló un prototipo utilizando el framework Kivy, mostrado en la figura 33. Este prototipo funcionó correctamente en entorno de escritorio (Windows), pero presentó fallas críticas al ser empaquetado como APK para dispositivos Android, lo que impidió su despliegue en el entorno real previsto. Debido a estas limitaciones, principalmente relacionadas con compatibilidad, rendimiento y acceso a APIs nativas de Android, se decidió descartar Kivy como solución definitiva.

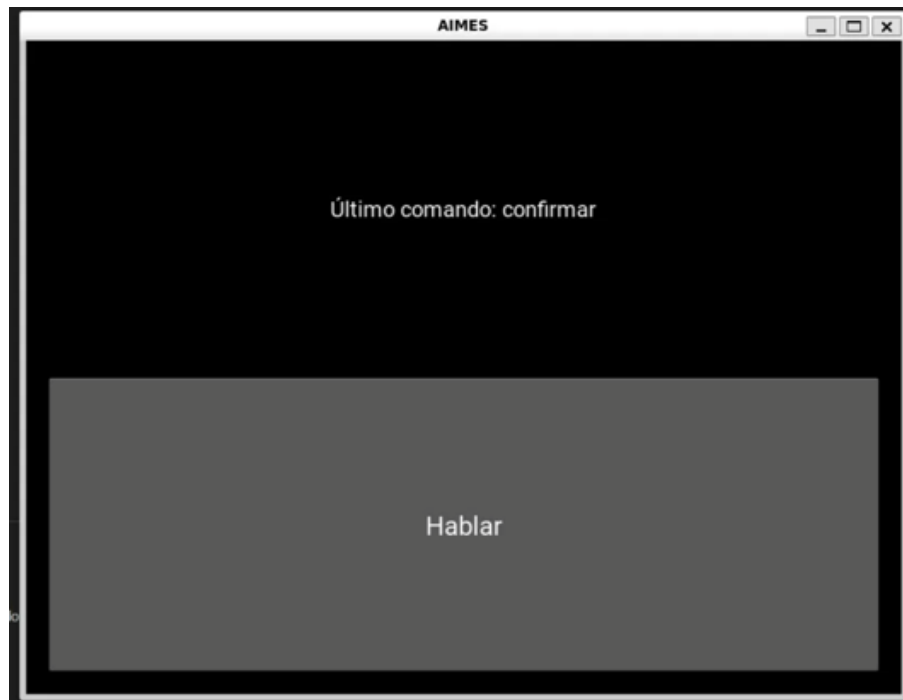


Figura 33: Primer prototipo de la interfaz móvil desarrollado en Kivy (descartado en fase posterior).

El desarrollo de la interfaz móvil fue retomado desde cero utilizando Android Studio con Kotlin, lo que permitió integrar funciones nativas, soporte estable para cámara y TTS, así como diseño visual acorde a requisitos planteados en la sección 3. Las nuevas vistas de interfaz se presentan en las figuras 34, 35 y 36.

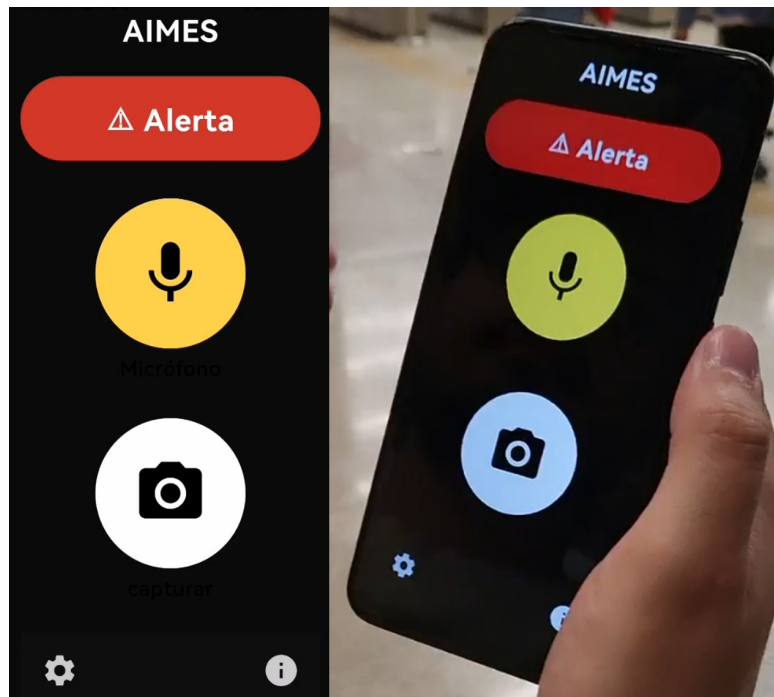


Figura 34: Menú principal del prototipo de interfaz móvil desarrollado en Android Studio.

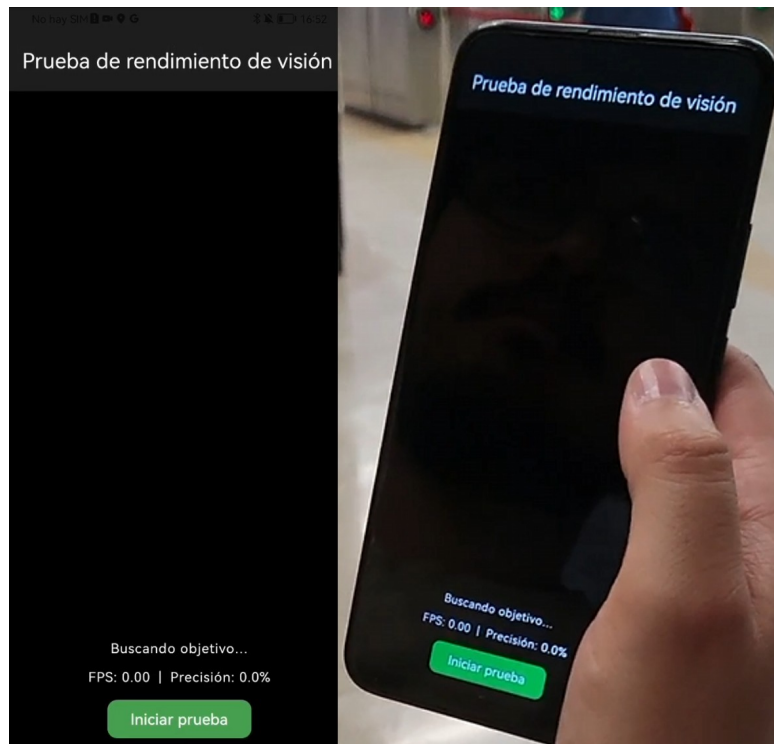


Figura 35: Pantalla de acceso a la funcionalidad de captura de imágenes.

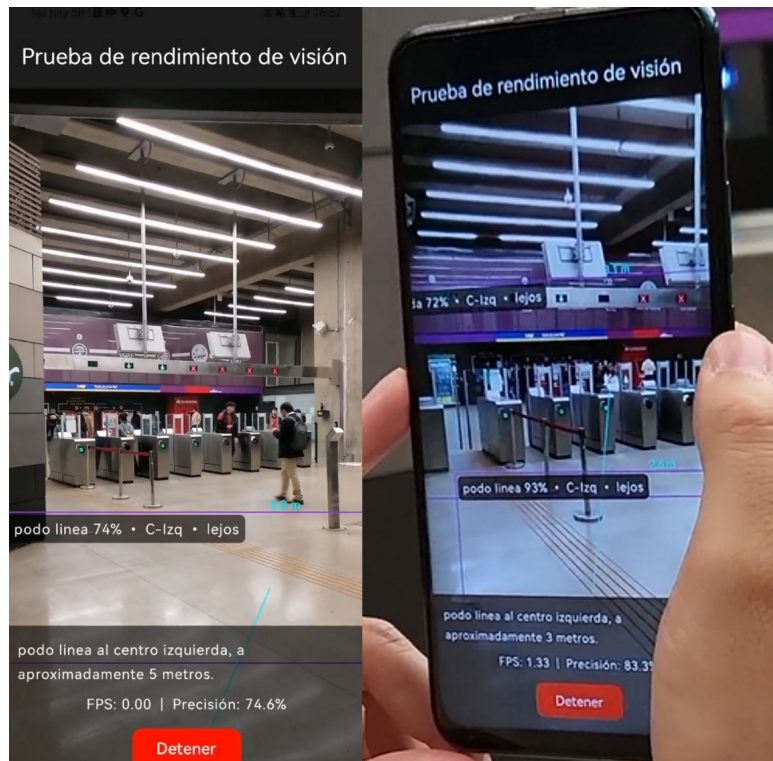


Figura 36: Funcionalidad de captura de imágenes en estado activo. Destacando las mediciones de desempeño y registro de detecciones por parte de YOLOv11n-segment.

A partir de las imágenes presentadas, se observa el funcionamiento efectivo del sistema en la infraestructura del Metro de Santiago.

Para complementar esta evidencia, se ha dispuesto un repositorio en OneDrive que contiene:

- Cuatro registros en video de pruebas realizadas en terreno.
- Capturas adicionales del flujo de interacción de la aplicación.

El material complementario asociado a esta etapa se encuentra disponible en un repositorio en línea. Contactar con los autores para acceder al enlace.

10. Gestión de Riesgos

10.1. Supuestos

1. Se asume que el usuario opera la aplicación dentro de entornos previamente configurados y entrenados para su correcto funcionamiento. Esto es, el caso particular de la Línea 6 del Metro de Santiago.
2. Se asume que, en horarios de alta concurrencia, la aplicación puede pasar a segundo plano para priorizar la guía de otras personas.
3. Se asume que los ascensores para personas con movilidad reducida estará siempre disponible.

10.2. Dependencias

1. La efectividad del sistema depende directamente del entorno físico: señalética clara, iluminación suficiente, y características estructurales reconocibles por los modelos de visión.
2. En situaciones de alto flujo, la orientación del usuario también dependerá parcialmente de la colaboración de terceros presentes en el entorno.
3. El funcionamiento continuo y confiable de la cámara del dispositivo es indispensable para la operación del sistema.

10.3. Restricciones

1. La cámara debe mantenerse a una distancia no superior a un metro del usuario, preferiblemente acoplada al cuerpo o bastón, para garantizar un ángulo de visión consistente.
2. La orientación de la cámara debe ser fija en posición horizontal para preservar el marco de referencia de visión que se espera de la aplicación.
3. La aplicación debe estar optimizada para funcionar correctamente en dispositivos móviles de gama media, limitando así el tamaño de la app, el uso de memoria RAM y la demanda de procesamiento.
4. Tiempo de respuesta esperado $\leq 200\text{ms}$; FPS esperado: ≥ 30 ; Confianza de predicción: $\geq 80\%$.

10.4. Riesgos

1. La efectividad del sistema podría verse comprometida si el usuario presenta movilidad reducida o movimientos inestables, afectando la calidad de la captura visual.
 2. En entornos altamente congestionados, personas u objetos móviles pueden generar oclusión visual, disminuyendo la precisión del modelo de detección y reconocimiento.
 3. Existe el riesgo de que el nivel de accesibilidad de la interfaz no cumpla con los estándares esperados para usuarios con discapacidad visual, especialmente si no se prueba con población real.
 4. Puede haber inconsistencias o errores en la información capturada si los carteles están dañados, mal iluminados o modificados respecto al entrenamiento del sistema.
 5. Puede haber una pérdida de oportunidad de indicación si hay un tiempo de respuesta excesivo entre la captura de escena y la entrega de instrucciones.
 6. La falta de conexión a Internet (en caso de dependencia parcial para procesamiento o actualizaciones) podría degradar la experiencia o desactivar algunas funciones críticas.
-

10.5. Cómo Mitigar

1. **Movilidad reducida o movimientos inestables del usuario:** La efectividad del sistema podría verse comprometida si el usuario presenta movilidad reducida o movimientos inestables, afectando la calidad de la captura visual.
Mitigación: Implementar estabilización digital de imagen en tiempo real y técnicas de corrección de desenfoque. Entrenar los modelos con datos recolectados en condiciones de movimiento no ideal.
 2. **Oclusión visual en entornos congestionados:** Personas u objetos móviles pueden generar oclusión visual, disminuyendo la precisión del modelo de detección y reconocimiento.
Mitigación: Incluir mecanismos de predicción temporal y seguimiento multiobjeto para inferir información parcial. Reforzar el entrenamiento con datos de alta concurrencia.
 3. **Interfaz no accesible para usuarios con discapacidad visual:** Existe el riesgo de que el nivel de accesibilidad de la interfaz no cumpla con los estándares esperados.
Mitigación: Aplicar pruebas con usuarios reales desde etapas tempranas del desarrollo (co-diseño inclusivo) y ajustarse a las normas WCAG y recomendaciones de la Unión Mundial de Ciegos.
 4. **Errores por señalética defectuosa o no reconocible:** Puede haber inconsistencias si los carteles están dañados, mal iluminados o modificados respecto al entrenamiento.
Mitigación: Incorporar modelos robustos al cambio de dominio y añadir un módulo OCR adaptativo que combine contexto con reconocimiento textual. Actualizar periódicamente los datos de entrenamiento.
 5. **Retraso en la entrega de instrucciones:** Puede haber pérdida de oportunidad si el tiempo de respuesta entre captura e indicación es excesivo.
Mitigación: Optimizar los pipelines de inferencia y audio. Implementar un sistema de prioridad para eventos críticos y pruebas sistemáticas de latencia.
 6. **Falta de conexión a Internet:** Si el sistema depende de conectividad para ciertas funciones, una desconexión podría degradar su rendimiento.
Mitigación: Diseñar el sistema para operar principalmente offline, manteniendo procesamiento en el dispositivo. Las actualizaciones pueden gestionarse de forma diferida o periódica.
-

11. Verificación y Validación

11.1. Verificación y Validación

El proceso de verificación y validación constituye una etapa crítica para asegurar la calidad y confiabilidad del sistema AIMES. Mientras la verificación se centra en responder a la pregunta “¿estamos construyendo el sistema correctamente?”, la validación busca responder “¿estamos construyendo el sistema adecuado?”.

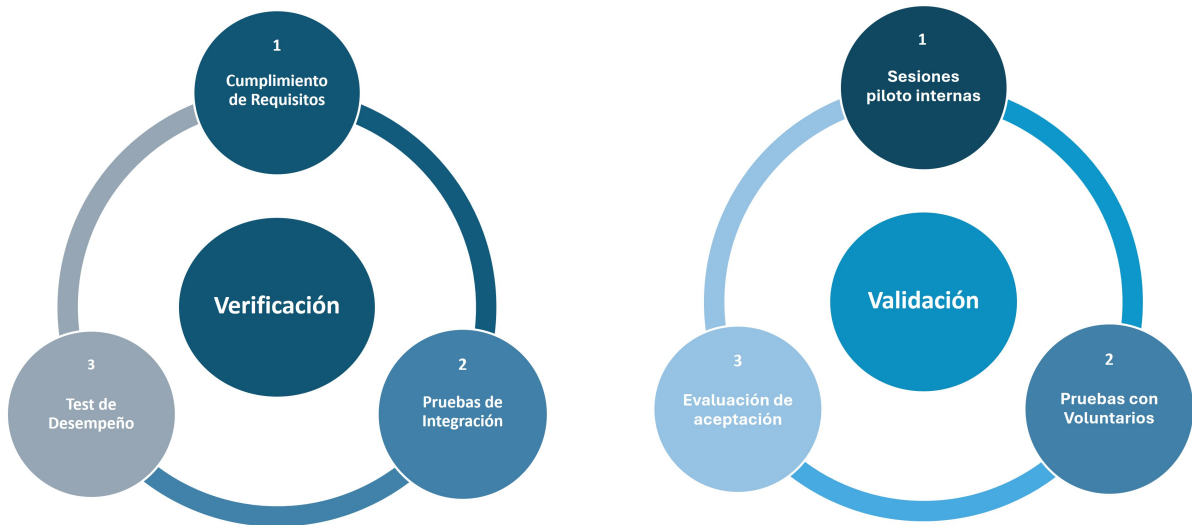


Figura 37: Diagramas de Verificación y Validación, con tres pasos para cada etapa.

En la figura 37 se muestra la separación de ambos procesos, cada uno compuesto por tres pasos principales:

11.1.1. Verificación:

1. Cumplimiento de requisitos: Garantizar que los requerimientos funcionales y no-funcionales estén correctamente documentados y sean consistentes.
2. Pruebas de Integración: Confirmar que los módulos definidos responden a los requerimientos planteados y son técnicamente viables.
3. Test de Desempeño: Comprobar que cada componente implementado cumple su función en entornos controlados.

11.1.2. Validación:

1. Sesiones piloto internas: Validar que el sistema responde de manera adecuada en escenarios simulados.
2. Pruebas con Voluntarios: contrastar el desempeño del sistema con el entorno real del Metro de Santiago, incorporando usuarios con discapacidad visual.
3. Evaluación de Aceptación: evaluar si el sistema satisface las necesidades prácticas de accesibilidad y asistencia, identificando mejoras antes de la implementación final.

De esta manera, la verificación asegura que la implementación es coherente con el diseño y los requisitos, mientras que la validación garantiza que AIMES realmente aporta valor y cumple con su propósito de asistencia inclusiva.

11.2. Plan de Testing

Se complementa al Test de Usabilidad descrito en la sección 5.1 del presente informe.

Al menos dos sesiones deben ser llevadas a cabo, tanto por usuarios voluntarios con ceguera, con capacidad visual menor o un usuario vidente que se le impida ver durante el transcurso de la prueba. Estas sesiones tienen como propósito evaluar tanto la efectividad técnica del sistema como la percepción de usabilidad por parte de los participantes.

El plan de testing considera las siguientes etapas:

1. **Sesiones internas de validación:** Realizadas por el equipo desarrollador, permiten identificar errores de integración y ajustar la configuración de los modelos antes de exponerlos a los usuarios externos.
2. **Pruebas con usuarios voluntarios:** Se desarrollarán en un entorno controlado (laboratorio o sala de simulación) para medir precisión, tiempos de respuesta y comprensión de las instrucciones auditivas.
3. **Pruebas en terreno:** Sesiones en estaciones reales del Metro de Santiago, con foco en el flujo de desplazamiento, ruido ambiental y variabilidad de iluminación. Aquí se evaluará la utilidad práctica del sistema en condiciones reales.
4. **Retroalimentación y ajustes:** Cada sesión concluye con entrevistas breves y encuestas de percepción (también contemplados en el Test de Usabilidad), cuyos resultados se sistematizan para mejorar la siguiente iteración de pruebas.

De esta manera, el plan de testing asegura un ciclo iterativo de mejora, combinando verificación técnica con validación en usuarios reales, para garantizar que AIMES avance hacia un MVP confiable y alineado con las necesidades de su público objetivo.

11.3. Análisis Crítico y Evaluación de Riesgos

A priori, se han identificado los siguientes riesgos, tanto los que han sido mencionados a lo largo del presente documento, como otros a considerar durante la implementación de AIMES:

- **Desequilibrio del dataset:** Puede reducir la precisión en señales con baja representación. Se mitiga mediante técnicas de aumento de datos y entrenamiento iterativo. Contemplado como paso complementario y posterior a la entrega del proyecto.
- **Compatibilidad en Android:** La exportación desde prototipos en Windows a APK presenta inestabilidades. Se plantea migración gradual y pruebas incrementales en móviles, descartando cualquier librería no compatible y que, las que sí sean compatibles, su re-implementación sea más cara para el tiempo estimado de entrega.
- **Latencia del sistema:** Retrasos superiores a 2 segundos comprometen la utilidad de la asistencia. Se evalúa qué optimizar de los modelos y reducción de parámetros para el tamaño.
- **Factores externos:** Condiciones como ruido ambiental, flujo de personas o fallas en la conectividad pueden degradar la experiencia. Se mitiga con robustez en el preprocesamiento y fallback offline en la app, como con la planificación de un *Happy Path* que asegure a los desarrolladores el mínimo cumplimiento de los requisitos funcionales descritos en la sección 2.

12. Resultados Fases Metodológicas

El desarrollo del proyecto AIMES se trabajó siguiendo la metodología definida en la sección 5.4, la cual combinó principios de Design Thinking y un enfoque iterativo inspirado en SCRUM. Mediante el uso de esta estructura metodológica fue posible avanzar de manera ordenada, flexible y centrada en el usuario final, permitiendo que cada fase de desarrollo aportara aprendizajes a medida que se avanzaba con el proyecto.

A lo largo del proceso el equipo enfrentó diversos desafíos, como la limitación de acceso a usuarios reales en el contexto de Metro, ajustes en los tiempos de entrega y dificultades técnicas propias del diseño accesible. Sin embargo, la metodología planteada inicialmente facilitó la adaptación a estas circunstancias sin afectar la calidad del proyecto. Gracias al modelo iterativo, fue posible refinar la investigación, ajustar los requisitos específicos, rediseñar interfaces y validar decisiones mediante pruebas en terreno.

Los hallazgos obtenidos de cada etapa de la metodología fueron de gran relevancia:

- **Empatizar:** La visita a la Biblioteca Central para Ciegos, el análisis de estudios como ENDIDE 2022 y la entrevista con usuarios frecuentes (Don Iván) entregaron retroalimentación real sobre comportamientos, frustraciones y expectativas en la movilidad diaria.
- **Definir:** A partir de los hallazgos, se identificaron necesidades críticas como la orientación espacial, la claridad auditiva y la compatibilidad con tecnologías de asistencia. Esto permitió formular un problema bien delimitado y enfocado en la accesibilidad.
- **Idear:** Se generó un amplio conjunto de posibles soluciones, modelado mediante diagramas de funciones y conceptos, que permitió visualizar la arquitectura del sistema y las interacciones clave entre los módulos.
- **Prototipar:** Se desarrolló el diseño de la interfaz siguiendo pautas WCAG 2.1, con iconografía de alto contraste, textos amplios y navegación totalmente accesible mediante lectores de pantalla.
- **Evaluar:** Mediante pruebas en terreno se identificaron oportunidades de mejora, especialmente en la interacción por voz y el reconocimiento de elementos clave dentro de la estación. Los resultados confirmaron que el diseño cumplía con las funciones críticas, pero que aún queda mucho por mejorar y pulir.

En conjunto, las fases metodológicas permitieron avanzar hacia un prototipo funcional, demostrando que la flexibilidad del proceso, sumada a iteraciones constantes, fue clave para construir una solución centrada en la experiencia del usuario con discapacidad visual. AIMES logró así un equilibrio entre la viabilidad técnica, accesibilidad y aporte social, cumpliendo con los objetivos establecidos en el plan de trabajo inicial.

A continuación, se detalla en la figura 38 cómo se distribuyó el trabajo de cada integrante de AIMES para desarrollar la solución.

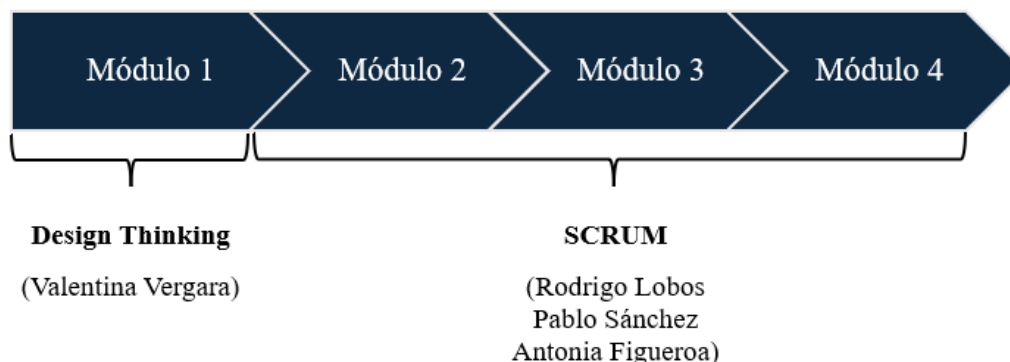


Figura 38: Distribución de trabajo por integrante

13. Conclusión

13.1. Conclusiones

El desarrollo actual de AIMES ha permitido validar la factibilidad técnica del sistema, logrando la integración de visión por computador, procesamiento en tiempo real y retroalimentación auditiva dentro de una aplicación móvil operativa. Se ha demostrado que modelos de detección y segmentación como YOLOv11 nano-segment pueden ejecutarse localmente en dispositivos Android sin dependencia de servidores externos, manteniendo tiempos de inferencia compatibles con una experiencia en tiempo real. No obstante, esta versión preliminar sólo permite la detección de elementos relevantes del entorno, pero aún no proporciona orientación espacial significativa al usuario, por lo que el sistema *funciona*, pero no *asiste*. Esta brecha define el punto crítico de transición entre un prototipo funcional y una herramienta de asistencia real.

Uno de los principales aportes del presente trabajo es la creación de un dataset original de infraestructura del Metro de Santiago, enfocado en accesibilidad y etiquetado en español. Este recurso constituye una contribución inédita en el contexto nacional, permitiendo el estudio de señalética, accesos, rutas podotáctiles, escaleras, ascensores y elementos arquitectónicos urbanos desde una perspectiva de inclusión y movilidad asistida. Su publicación permitirá reducir la brecha de datasets con especificidad latinoamericana y fomentar nuevas investigaciones en accesibilidad urbana basada en IA.

Asimismo, mediante el uso de benchmarks, se exploró el rendimiento de modelos Ultralytics YOLO en su versión más reciente, evaluando su desempeño en dispositivos móviles de gama media. Esto aporta evidencia al campo de edge AI, particularmente en relación a la precisión, latencia y consumo energético en escenarios de uso crítico, como el transporte público.

Finalmente, se consolidó el desarrollo de la aplicación móvil mediante la migración desde un prototipo inicial en Kivy hacia una implementación nativa con Android Studio y Kotlin, debido a limitaciones detectadas en compatibilidad con APIs de accesibilidad, empaquetado en `.apk` y soporte para librerías de STT/TTS.

En conjunto, estos avances demuestran que AIMES ha superado la fase de validación técnica, consolidando tres aportes concretos: (1) un dataset propio y replicable, (2) un modelo funcional ejecutado en tiempo real en un teléfono móvil, y (3) una aplicación operativa que integra visión asistiva y respuesta auditiva.

13.2. Trabajo Futuro

Para que AIMES pueda ser utilizado de forma segura y efectiva por personas con discapacidad visual en entornos reales, la siguiente etapa debe centrarse en la incorporación de capacidades de orientación espacial. Esto implica que el sistema no sólo detecte objetos, sino que sea capaz de informar su ubicación relativa (dirección, distancia, accesibilidad, riesgos), y traducir dicha información en instrucciones accionables, claras y continuas.

Entre las líneas de desarrollo priorizadas se identifican:

- Integración de retroalimentación multimodal (voz + vibración + posibles señales hápticas).
- Validación con usuarios reales bajo protocolos de accesibilidad, usabilidad y ética aplicada, siguiendo lineamientos WCAG y Android Accessibility.
- Optimización energética del modelo y aplicación para garantizar autonomía suficiente en situaciones de uso prolongado.
- Divulgación y documentación formal del dataset para su uso abierto por la comunidad científica a través de Roboflow.

El objetivo de esta futura fase será transformar AIMES desde un producto demostrativo hacia una herramienta asistiva operativa en terreno, escalable y socialmente válida, contribuyendo al desarrollo de tecnologías de movilidad inclusiva basadas en inteligencia artificial.

Referencias

- [1] Ministerio de Desarrollo Social y Familia. *Resultados Encuesta Nacional de Discapacidad y Dependencia 2022: Población adulta*. [https://observatorio.ministeriodesarrollosocial.gob.cl/storage/docs/endide/230128_Resultados_ENDIDE_2022\(Poblacion-adulta\).pdf](https://observatorio.ministeriodesarrollosocial.gob.cl/storage/docs/endide/230128_Resultados_ENDIDE_2022(Poblacion-adulta).pdf). Gobierno de Chile. 2022.
- [2] World Health Organization: WHO. *Ceguera y discapacidad visual*. es. Ago. de 2023. URL: <https://www.who.int/es/news-room/fact-sheets/detail/blindness-and-visual-impairment>.
- [3] Francia Quezada. «Reportaje revela que ambulantes han dañado cerca del 30 % de los ascensores del Metro de Santiago». es-ES. En: *BioBioChile - La Red De Prensa Más Grande De Chile* (ene. de 2023). URL: <https://www.biobiochile.cl/noticias/nacional/region-metropolitana/2023/06/05/reportaje-revela-que-ambulantes-han-danado-cerca-del-30-de-los-ascensores-del-metro-de-santiago.shtml>.
- [4] Erum Salam. «I tried Be My Eyes, the popular app that pairs blind people with helpers». En: *The Guardian* (jul. de 2019). URL: <https://www.theguardian.com/lifeandstyle/2019/jul/12/be-my-eyes-app-blind-people-helpers>.
- [5] Perkins School for the Blind. *Microsoft seeing AI and low Vision Review – Perkins School for the Blind*. en-US. Oct. de 2022. URL: <https://www.perkins.org/resource/microsoft-seeing-ai-and-low-vision-review/>.
- [6] Perkins School for the Blind. *Lazarillo: a free accessible GPS app for the blind and visually impaired – Perkins School for the Blind*. en-US. Jul. de 2024. URL: <https://www.perkins.org/resource/lazarillo-free-accessible-gps-app-blind-and-visually-impaired/>.
- [7] Metro de Santiago. Nov. de 2023. URL: <https://www.metro.cl/noticias/lazarillo-la-aplicacion-que-ayuda-a-las-personas-ciegas-se-expande-en-la-red-de-metro-y-alcanza-las-13-estaciones>.
- [8] Fundación Luz. *Estudio: Personas con discapacidad visual en Chile según ENDIDE 2022. Análisis de población adulta y NNA*. <https://fundacionluz.cl>. 2023.
- [9] Perkins School for the Blind. *Using Aira with low vision – Perkins School for the Blind*. en-US. Jul. de 2023. URL: <https://www.perkins.org/resource/aira-low-vision-review/>.
- [10] Perkins School for the Blind. *Visual assistance apps: Post roundup – Perkins School for the Blind*. en-US. Ago. de 2023. URL: <https://www.perkins.org/resource/visual-assistance-apps-post-roundup/>.
- [11] Ghatan Fauzi Nugraha et al. «Object Detection for the Visually Impaired: A Systematic Literature Review». En: *2024 10th International Conference on Wireless and Telematics (ICWT)*. 2024, págs. 1-16. DOI: 10.1109/ICWT62080.2024.10674727.
- [12] Ultralytics. *SAM (Segment Anything Model)*. URL: <https://docs.ultralytics.com/models/sam/>.
- [13] Alexander Kirillov et al. «Segment Anything». En: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. de 2023, págs. 4015-4026.
- [14] Chaoning Zhang et al. *Faster Segment Anything: Towards Lightweight SAM for Mobile Applications*. 2023. arXiv: 2306.14289 [cs.CV]. URL: <https://arxiv.org/abs/2306.14289>.
- [15] Ultralytics. *MobileSAM (Mobile Segment Anything Model)*. URL: <https://docs.ultralytics.com/es/models/mobile-sam/>.
- [16] Joseph Redmon et al. *You only look once: Unified, Real-Time Object Detection*. Ene. de 2015. URL: <https://arxiv.org/abs/1506.02640>.
- [17] So Development. *Comparing YOLOv11 and YOLOv12: A Deep Dive into the Next-Generation Object Detection Models*. en-US. Feb. de 2025. URL: <https://so-development.org/comparing-yolov11-and-yolov12-a-deep-dive-into-the-next-generation-object-detection-models/>.
- [18] Ratnesh Litoriya et al. «Implementing Visual Assistant using YOLO and SSD for Visually-Impaired Persons». En: *Journal of Automation, Mobile Robotics and Intelligent Systems* 17.4 (feb. de 2024), págs. 79-87. DOI: 10.14313/JAMRIS/4-2023/33. URL: <https://www.jamris.org/index.php/JAMRIS/article/view/844>.

-
- [19] Nicolas Carion et al. *End-to-End Object Detection with Transformers*. 2020. arXiv: 2005.12872 [cs.CV]. URL: <https://arxiv.org/abs/2005.12872>.
 - [20] Ao Wang et al. «Rep ViT: Revisiting Mobile CNN From ViT Perspective». En: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2024, págs. 15909-15920. DOI: 10.1109/CVPR52733.2024.01506.
 - [21] Tadeh Ghahremanians y Hossein Mahvash Mohammadi. «Efficient Real-Time Pathfinding for Visually Impaired Individuals». En: *IEEE Access* 13 (2025), págs. 71323-71334. DOI: 10.1109/ACCESS.2025.3562247.
 - [22] Andrew G. Howard et al. *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*. 2017. arXiv: 1704.04861 [cs.CV]. URL: <https://arxiv.org/abs/1704.04861>.
 - [23] Sachin Mehta y Mohammad Rastegari. *MobileViT: Light-weight, General-purpose, and Mobile-friendly Vision Transformer*. 2022. arXiv: 2110.02178 [cs.CV]. URL: <https://arxiv.org/abs/2110.02178>.
 - [24] Mario Almeida et al. *Smart at what cost? Characterising Mobile Deep Neural Networks in the wild*. 2021. arXiv: 2109.13963 [cs.LG]. URL: <https://arxiv.org/abs/2109.13963>.
 - [25] Alpha Cephei Org. *Vosk Webpage*. 2022. URL: <https://alphacephei.com/vosk/>.
 - [26] Olga Yakovleva. *RHVoice github*. 2022. URL: <https://github.com/RHVoice/RHVoice>.
 - [27] Kivy Authors. *Kivy Documentation*. 2011. URL: <https://kivy.org/doc/stable/>.
 - [28] World Wide Web Consortium. *Web Content Accessibility Guidelines (WCAG) 2.1*. <https://www.w3.org/TR/WCAG21/>. 2018.
 - [29] *Android Studio App Tools - Android Developers*. URL: <https://developer.android.com/studio>.
 - [30] Rahima Khanam y Muhammad Hussain. *A Review of YOLOv12: Attention-Based Enhancements vs. Previous Versions*. 2025. arXiv: 2504.11995 [cs.CV]. URL: <https://arxiv.org/abs/2504.11995>.
 - [31] URL: <https://github.com/ultralytics>.
 - [32] Kivy. *GitHub - kivy/plyer: Plyer is a platform-independent Python wrapper for platform-dependent APIs*. URL: <https://github.com/kivy/plyer>.
 - [33] Instituto Nacional de Estadísticas (INE). *Censo 2024 – Primeros resultados*. <https://www.ine.gob.cl>. INE. 2024.
 - [34] Ken Schwaber y Jeff Sutherland. *The Scrum Guide™: The Definitive Guide to Scrum – The Rules of the Game*. <https://scrumguides.org/scrum-guide.html>. Scrum.org. 2020.
 - [35] Tim Brown. *Change by Design: How Design Thinking Creates New Alternatives for Business and Society*. Harvard Business Press, 2009.
 - [36] Karl T. Ulrich. *Diseño y desarrollo de productos (5ª ed.)* McGraw-Hill, 2012.
 - [37] AIMES. *Senales Computer Vision Model*. 2025. URL: <https://universe.roboflow.com/aimes/senales-cnijl>.
 - [38] AIMES. *Senales-segmentation Computer Vision Dataset*. 2025. URL: <https://universe.roboflow.com/aimes/senales-segmentation-zvzpo>.