ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

CARRERA DE ELECTRÓNICA Y AUTOMATIZACIÓN

DESARROLLO DE UN SISTEMA EMBEBIDO BASADO EN FPGA Y VISIÓN ARTIFICIAL IMPLEMENTADO EN UN VEHÍCULO PARA DETECTAR Y ALERTAR SOBRE LA PRESENCIA DE CANES EN LA VÍA.

Trabajo de Integración Curricular

Tipo: Dispositivo Tecnológico

Presentado para optar al grado académico de:

INGENIERO/A EN ELECTRÓNICA Y AUTOMATIZACIÓN

AUTORES:

CARLOS JULIO DOMINGUEZ OROZCO

GIUSTINNE SOLANGE LLIGUIN AMBATO

Riobamba – Ecuador

2023

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

CARRERA DE ELECTRÓNICA Y AUTOMATIZACIÓN

DESARROLLO DE UN SISTEMA EMBEBIDO BASADO EN FPGA Y VISIÓN ARTIFICIAL IMPLEMENTADO EN UN VEHÍCULO PARA DETECTAR Y ALERTAR SOBRE LA PRESENCIA DE CANES EN LA VÍA.

Trabajo de Integración Curricular

Tipo: Dispositivo Tecnológico

Presentado para optar al grado académico de:

INGENIERO EN ELECTRÓNICA Y AUTOMATIZACIÓN

AUTORES:

CARLOS JULIO DOMINGUEZ OROZCO

GIUSTINNE SOLANGE LLIGUIN AMBATO

DIRECTOR: ING. JORGE LUIS HERNÁNDEZ AMBATO

Riobamba – Ecuador

2023

Nosotros, Carlos Julio Domínguez Orozco y Giustinne Solange Lliguin Ambato, declaramos que el presente Trabajo de Integración Curricular es de mi autoría y los resultados de este son auténticos. Los textos en el documento que provienen de otras fuentes están debidamente citados y referenciados.

Como autores asumimos la responsabilidad legal y académica de los contenidos del Trabajo de Integración Curricular; el patrimonio intelectual pertenece a la Escuela Superior Politécnica de Chimborazo.

Riobamba, 13 de noviembre de 2023

Carlos Julio Domínguez Orozco

060540584-4

Giustinne Solange Lliguin Ambato

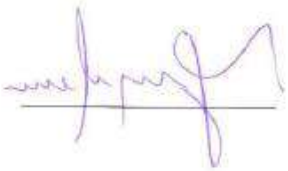
060437359-7

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

CARRERA ELECTRÓNICA Y AUTOMATIZACIÓN

El Tribunal del Trabajo de Integración Curricular certifica que: El Trabajo de Integración Curricular; Tipo: Dispositivo Tecnológico, **DESARROLLO DE UN SISTEMA EMBEBIDO BASADO EN FPGA Y VISIÓN ARTIFICIAL IMPLEMENTADO EN UN VEHÍCULO PARA DETECTAR Y ALERTAR SOBRE LA PRESENCIA DE CANES EN LA VÍA**, realizado por el señor y la señorita: **CARLOS JULIO DOMINGUEZ OROZCO y GIUSTINNE SOLANGE LLIGUIN AMBATO**, ha sido minuciosamente revisado por los Miembros del Tribunal del Trabajo de Integración Curricular, el mismo que cumple con los requisitos científicos, técnicos, legales, en tal virtud el Tribunal Autoriza su presentación.

	FIRMA	FECHA
Ing. Franklin Geovanni Moreno Montenegro **PRESIDENTE DEL TRIBUNAL**		2023-11-13
Dr. Jorge Luis Hernández Ambato **DIRECTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR**		2023-11-13
Ing. Jorge Vicente Yuquilema Illapa **ASESOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR**		2023-11-13

DEDICATORIA

A Dios por guiarme y bendecirme en cada momento de mi vida. A mi padre Carlos Humberto Dominguez Sinchiri por el apoyo inquebrantable sus sabios consejos y enseñarme el valor del esfuerzo, que han sido la brújula en mis decisiones. A mi madre Magda del Roció Orozco Poma por su amor incondicional y fe constante, que han sido mi fuerza en cada paso. A mis hermanos, Juan, Carina, Rodrigo, Mónica, María y Jhon, por su aliento y confianza en mí. A toda mi familia, les dedico este logro compartido, porque cada uno ha aportado a mi crecimiento como persona, con amor.

Carlos

Dedico este trabajo de tesis a mi querido Padre, Bolívar Lliguin, cuyo amor incondicional y sabios consejos han sido el faro que ha guiado mis pasos en el camino de la vida. Gracias por ser mi pilar fundamental y por encaminarme con amor y dedicación hacia la búsqueda de la excelencia en cada decisión que tomo, tanto en mi servicio a Dios como en mi profesión. A mis adorados tíos, quienes han sido un sólido apoyo en mi desarrollo personal y profesional. A mi amado esposo, Jimmy Aldaz, siendo mi compañero de vida, mi mayor motivación y mi fuente de amor y comprensión inagotables. Tu apoyo inquebrantable y tu confianza en mí me impulsan a esforzarme día a día para alcanzar mis sueños y metas.

Giustinne

AGRADECIMIENTO

A mis padres, cuyo sacrificio, esfuerzo y amor han sido el cimiento sobre el cual he logrado culminar esta carrera, A mi familia por todo el apoyo sobre mí. A los docentes de la Facultad de Informática y Electrónica por compartir sus conocimientos en mi formación como profesional, en especial a mi director de tesis el Ing. Jorge Hernández por su invaluable orientación a lo largo de este trabajo de titulación

Carlos

El principal agradecimiento a Dios quien me ha dado la fortaleza e inteligencia para cumplir cada meta trazada en mi desarrollo personal e intelectual. A mi familia por confiar en mi y creer ciegamente en la capacidad para conseguir mis sueños y de manera especial a mi Tutor el Ing. Jorge Luis Hernandez quien nunca desistio en enseñarme y orientarme en el desarrollo de este trabajo de titulación. Este logro es el resultado del amor, apoyo y la influencia positiva de cada una de las personas que han formado parte de cada etapa de mi vida,

Giustinne

ÍNDICE DE CONTENIDO

ÍNDICE DE TABLAS.....XII
ÍNDICE DE ILUSTRACIONES.....XV
ÍNDICE DE ANEXOS.....XVII
RESUMEN.....XVIII
SUMMARY.....XIX
INTRODUCCIÓN.....1

CAPÍTULO I

1. DIAGNOSTICO DEL PROBLEMA.....2

1.1 Planteamiento del problema.....2

1.2 Justificación.....2
1.2.1 Justificación Teórica.....2
1.2.2 Justificación aplicativa.....3

1.3 Objetivos.....5
1.3.1 Objetivo General.....5
1.3.2 Objetivos Específicos.....5

CAPÍTULO II

2. MARCO TEÓRICO.....6

2.1 Generalidades de los Sistemas Embebidos.....6

2.2 Elementos Hardware de los Sistemas Embebidos para Visión Artificial.....7
2.2.1 Microprocesador.....7
2.2.2 Microcontrolador.....8
2.2.3 Tarjetas de Desarrollo.....9

2.2.3.1 Tipos de Tarjetas de Desarrollo 9
2.2.4 Sensores 10
2.2.4.1 Sensor de Imagen 11
2.2.4.2 Dispositivos de captura de imagen 11
2.2.5 Actuadores 12

2.3 Herramientas Software para Visión Artificial en Sistemas Embebidos 13
2.3.1 Técnicas de Inteligencia artificial 13
2.3.1.1 Visión por Computadora o Visión Artificial 14
2.3.1.2 Machine Learning 16
2.3.2 Redes Neuronales Artificiales 18
2.3.2.1 Neurona Artificial 18
2.3.2.2 Características de las RNA 21
2.3.2.3 Elementos de una RNA *22*
2.3.2.4 Arquitecturas de RNAs *23*
2.3.3 Técnicas para el entrenamiento de RNA 24
2.3.3.1 Algoritmo de Aprendizaje por Transferencia (Transfer Learning) 25

2.4 Plataformas de desarrollo para Sistemas Embebidos 26
2.4.1 Lenguajes de Programación 27
2.4.1.1 Python 27
2.4.1.2 C++ 27
2.4.1.3 VHDL 27
2.4.2 Entornos de Desarrollo 28
2.4.2.1 Vivado 28
2.4.2.2 Jupyter 28
2.4.2.3 Arduino IDE 28
2.4.3 Librerías para RNAs e Inteligencia Artificial 29
2.4.3.1 OpenCV 29
2.4.3.2 Tensor Flow 29

CAPÍTULO III

3. MARCO METODOLÓGICO 30

3.1 Metodología de Desarrollo 30

3.2 Parámetros Generales ... 32
3.2.1 Requerimientos ... *32*
3.2.2 Herramientas Hardware y Software para el desarrollo ... 33
3.2.2.1 Herramientas Hardware ... 33
3.2.2.2 Herramientas Software ... 36
3.2.3 Elaboración del Conjunto de Datos ... 36
3.2.3.1 Recolección de Datos ... 37
3.2.3.2 Acondicionamiento de los Datos ... 40

3.3 Modelo de Entrenamiento RNA desarrollado ... 40
3.3.1 Selección de la Arquitectura de la RNA ... 40
3.3.2 Selección de la Técnica de Entrenamiento ... 42
3.3.3 Proceso de Entrenamiento ... 43
3.3.4 Verificación de la RNA ... 44
3.3.4.1 Pruebas de Entrenamiento ... 45

3.4 Sistema embebido ... 45
3.4.1 Configuración de la Tarjeta de Desarrollo ... 46
3.4.2 Diagrama de bloques de la programación DDC sobre la PYNQ Z1. ... 49
3.4.3 Implementación del DDC ... 58

3.5 Validación y Pruebas ... 59
3.5.1 Pruebas de Funcionamiento ... 59
3.5.1.1 Pruebas de Funcionamiento en Ambientes Controlados ... 60
3.5.1.2 Pruebas de Funcionamiento en Ambientes no Controlados ... 61
3.5.2 Verificación de los Requerimientos ... 62

CAPÍTULO IV

4. PROPUESTA Y DISEÑO DEL PROTOTIPO ... 64

4.1 Esquema Electrónico para DDC. ... 64

4.2 Diseño de la PCB para activación de alarmas ... 65

4.3 Diseño de la carcasa para DDC (Dispositivo Detector de Canes) ... 66

4.4 Algoritmo General del Dispositivo DDC 69

CAPÍTULO V

5. VALIDACIÓN DEL PROTOTIPO 71

5.1 Pruebas de Validación del Modelo Entrenado 71
5.1.1 Curva de Precisión de Entrenamiento y Pruebas 71
5.1.2 Curva de Pérdida de Entrenamiento y Pruebas 72

5.2 Pruebas Estáticas de Experimentación sobre el S.E implementado 73
5.2.1 Pruebas de Campo en Ambiente Controlado 73
5.2.1.1 Distancia de detección de presencia de canes 74
5.2.1.2 Tiempo de Detección a la distancia de 3 metros 76
5.2.1.3 Tiempo de Detección a la distancia máxima 78
5.2.1.4 Diferenciación Persona-Can 80
5.2.1.5 Detección varios canes 81
5.2.2 Pruebas de Campo en Ambiente no Controlado 82
5.2.2.1 Distancia de Detección de la presencia de canes 82
5.2.2.2 Tiempo de Detección a la distancia mínima en 3 jornadas 85
5.2.2.3 Tiempo de Detección a la distancia máxima en 3 jornadas 87
5.2.2.4 Diferenciación Persona-Can 89
5.2.2.5 Detección varios canes 89

5.3 Verificaciones Estadísticas de los Tiempos de Respuesta 90
5.3.1 Verificación estadística del tiempo de respuesta a 3 metros en 3 jornadas en un ambiente no controlado. 91
5.3.2 Verificación estadística del tiempo de respuesta a 5.5 metros en 3 jornadas en un ambiente no controlado 94

5.4 Pruebas Dinámicas de Experimentación sobre el S.E implementado 97
5.4.1 Entorno de Evaluación 97
5.4.1.1 Delimitación de la Zona de Pruebas y Vehículo Participante 97
5.4.1.2 Sujetos de Prueba 98
5.4.1.3 Detección del DDC dentro del margen de referencia 99

5.5 Análisis de costos para la fabricación del DDC 101

CONCLUSIONES..... 102
RECOMENDACIONES..... 103
BIBLIOGRAFÍA
ANEXOS

ÍNDICE DE TABLAS

Tabla 2-1. Características diferenciales entre los sistemas embebidos y los sistemas tradicionales. 6
Tabla 2-2. Tabla comparativa sobre los atributos de las Tarjetas de Desarrollo. 9
Tabla 2-3. Descripción teórica de los tipos de sensores considerados en los S.E. 11
Tabla 2-4. Características y funcionalidades de los dispositivos de captura de imagen 12
Tabla 2-5. Descripción conceptual y gráfica de los tipos de actuadores. 13
Tabla 2-6. Técnicas de Inteligencia Artificial para reconocimiento de patrones. 14
Tabla 2-7. Subcampos existentes dentro del Machine Learning y su descripción. 17
Tabla 2-8. Principales componentes que conforman la Neurona. 19
Tabla 2-9. Principales Funciones de Activación para las Redes Neuronales Artificiales. 20
Tabla 2-10. Principales elementos que conforman una Red Neuronal Artificial. 22
Tabla 2-11. Arquitectura para el Entrenamiento de las Redes Neuronales Artificiales. 23
Tabla 2-12. Descripción de las capas existentes dentro de las Redes Neuronales Convolucionales (CNN). 24
Tabla 2-13. Técnicas para el entrenamiento de las Redes Neuronales Artificiales. 24
Tabla 3-1. Características técnicas de la tarjeta FPGA PynQ Z1 de Xilinx. 34
Tabla 3-2. Principales características técnicas y especificaciones de la cámara web 720px 35
Tabla 3-3. Componentes hardware para el desarrollo del Dispositivo Detector de Canes. 35
Tabla 3-4. Descripción de software, librerías y lenguajes de programación utilizados en el S.E. 36
Tabla 3-5. Atributos considerados en la adquisición de datos para el entrenamiento de la RNA. 38
Tabla 3-6. Cronograma de los días y hora por día requeridos para la adquisición de datos. 38
Tabla 3-7. Descripción detallada de las jornadas requeridas para la recolección de datos. 39
Tabla 3-8. Lugares territoriales y tipo de vías que fueron consideradas para la recolección de datos. 39
Tabla 3-9. Descripción detallada del proceso que sigue Google Colab para el entrenamiento. 44
Tabla 3-10. Bibliotecas necesarias para acceder a las funcionalidades, periféricos y procesar video de la tarjeta de desarrollo. 50
Tabla 3-11. Tipos de pruebas a realizar en ambientes controlados 60
Tabla 3-12. Tipos de pruebas a realizar en ambientes no controlados 61
Tabla 3-13. Lista de verificación de los requerimientos 62
Tabla 5-1. Prueba de distancia de detección de la presencia de canes. 74

Tabla 5-2. Representación y análisis de los datos recolectados en la detección de canes en ambientes controlados. 75
Tabla 5-3. Tiempo de detección de la presencia de canes a 3 metros de distancia. 76
Tabla 5-4. Prueba de Shapiro-Wilk para los datos del tiempo de detección a 3 metros en ambientes controlados. 77
Tabla 5-5. Tiempo de detección de la presencia de canes a 5.5 metros de distancia. 78
Tabla 5-6. Prueba de Shapiro-Wilk para los datos del tiempo de detección a 5.5 metros en ambientes controlados. 80
Tabla 5-7. Diferenciación Persona-can del Sistema Embebido en un Ambiente Controlado. ... 81
Tabla 5-8. Descripción de 4 casos de detección de varios canes a diferente distancia. 82
Tabla 5-9. Distancia de detección del DDC en ambientes no controlados. 83
Tabla 5-10. Representación y análisis de los datos recolectados en la detección de canes en ambientes no controlados. 84
Tabla 5-11. Tiempo de respuesta del DDC en 3 jornadas a una distancia mínima en un ambiente no controlado 85
Tabla 5-12. Descripción de la media y desviación estándar del tiempo de detección a 3 metros en 3 jornadas en un ambiente no controlado. 86
Tabla 5-13. Prueba de Shapiro-Wilk para el tiempo de detección a 3 metros obtenidos en 3 jornadas en una en ambientes no controlados. 86
Tabla 5-14. Tiempo de respuesta por parte del DDC durante la mañana a una distancia máxima. 87
Tabla 5-15. Descripción de la media y desviación estándar del tiempo de detección a 5.5 metros en 3 jornadas en un ambiente no controlado. 88
Tabla 5-16. Prueba de Shapiro-Wilk para el tiempo de detección a 5.5 metros obtenidos en 3 jornadas en una en ambientes no controlados. 88
Tabla 5-17. Diferenciación Persona-can del Sistema Embebido en un Ambiente no Controlado. 89
Tabla 5-18. Descripción de 4 casos de detección de varios canes a diferente distancia en un ambiente no controlado. 90
Tabla 5-19. Intervalos de confianza para la desviación estándar de las 3 jornadas a una distancia de 3 metros. 91
Tabla 5-20. valor p para los conjuntos de datos del tiempo de detección en 3 jornadas a una distancia de 3 metros. 92
Tabla 5-21. Análisis de varianza en la Prueba de diferencias significativas para las 3 jornadas a 3 metros de distancia. 93
Tabla 5-22. Prueba de Tukey para las 3 jornadas a 3 metros de distancia. 93

Tabla 5-23. Intervalos de confianza para la desviación estándar de las 3 jornadas a una distancia de 5.5 metros. 95
Tabla 5-24. valor p para los conjuntos de datos del tiempo de detección en 3 jornadas a una distancia de 5.5 metros. 95
Tabla 5-25. Análisis de varianza en la prueba de diferencias significativas en las 3 jornadas a 5.5 metros de distancia. 96
Tabla 5-26. Prueba de Tukey para las 3 jornadas a 5.5 metros de distancia. 97
Tabla 5-27. Detección del Dispositivo dentro del margen de referencia en el escenario de prueba. 99
Tabla 5-28. Datos cualitativos obtenidos de las 5 pruebas en las diferentes velocidades. 100
Tabla 5-29. Costos de fabricación del Dispositivo Detector de Canes. 101

ÍNDICE DE ILUSTRACIONES

Ilustración 2-1. Principales componentes hardware de un Sistema Embebido. 7
Ilustración 2-2. Representación general de un Microprocesador utilizado en los S.E. 8
Ilustración 2-3. Descripción general de la estructura de un Microcontrolador básico 8
Ilustración 2-4. Componentes de un sistema de visión artificial 15
Ilustración 2-5. Técnicas para el procesamiento de imágenes 16
Ilustración 2-6. Descripción Gráfica de la constitución de la Neurona. 18
Ilustración 2-7. Principales características de las Redes Neuronales Artificiales. 22
Ilustración 2-8. Diagrama de flujo Técnica de entrenamiento Transfer Learning 25
Ilustración 3-1. Arquitectura del Dispositivo Detector de Canes. 30
Ilustración 3-2. FPGA PYNQ Z1 de Xilinx. 33
Ilustración 3-3. Cámara Web HD 720px Webcam 34
Ilustración 3-4. Adaptación de los atributos para la recolección de datos en el proyecto. 37
Ilustración 3-5. Parámetros representativos dentro de la Arquitectura de una RNA 41
Ilustración 3-6. Ventajas consideradas para la selección de la Técnica de Entrenamiento 42
Ilustración 3-7. Configuración de la FPGA 47
Ilustración 3-8. App Fing para la obtención de dirección IP 48
Ilustración 3-9. Diagrama de bloques del algoritmo DDC 49
Ilustración 3-10. Diagrama de Flujo para la activación de la salida de video HDMI 51
Ilustración 3-11. Diagrama de flujo de la configuración de la cámara 52
Ilustración 3-12. Diagrama de flujo para la Visualización de Video 54
Ilustración 3-13. Diagrama de Flujo del algoritmo detector de canes. 57
Ilustración 3-14. Diagrama de bloque representativo de la Arquitectura del Dispositivo 58
Ilustración 3-15. Siluetas de canes elaboradas en MDF 59
Ilustración 4-1. Esquematico general de conexiones del DDC. 64
Ilustración 4-2. Diseño de la PCB utilizado para la activación de alarmas. 65
Ilustración 4-3. Diseño de la tapa para el corte laser. 66
Ilustración 4-4. Diseño de la parte interna de la base de la carcasa del DDC. 67
Ilustración 4-5. Vista de la parte externa de la base de la carcasa del DDC. 67
Ilustración 4-6. Vista frontal de la estructura del DDC. 68
Ilustración 4-7. Vista lateral de la estructura del DDC. 68
Ilustración 4-8. Diseño de la Portada colocada en la parte superior del DDC. 68
Ilustración 4-9. Diagrama de flujo del algoritmo general del dispositivo DDC 70
Ilustración 5-1. Descripción de la curva de precisión obtenido en el Modelo Entrenado. 72

Ilustración 5-2. Descripción de la curva de pérdida obtenido en el Modelo Entrenado. 73
Ilustración 5-3. Gráfico Scatter de los resultados obtenidos en la detección de canes en Ambientes Controlados. 75
Ilustración 5-4. Gráfico Scatter del Tiempo de Detección del DDC a tres metros de distancia. 78
Ilustración 5-5. Gráfico Scatter del Tiempo de Detección del DDC a 5.5 metros de distancia. 80
Ilustración 5-6. Gráfico Scatter de los resultados obtenidos en la detección de canes en Ambientes no Controlados. 84
Ilustración 5-7. Descripción Gráfica del escenario de pruebas dinámicas del DDC. 98
Ilustración 5-8. Canes participantes en las pruebas dinámicas del DDC. 98
Ilustración 5-9. Gráfico Scatter de los resultados obtenidos en la detección de canes en pruebas dinámicas. 100

ÍNDICE DE ANEXOS

ANEXO A: ALGORITMO DETECTOR DE CANES
ANEXO B: TABLA DE VALORES CRÍTICOS PARA LA PRUEBA DE TUKEY
ANEXO C: ESTRUCTURA DEL DISPOSITIVO DETECTOR DE CANES
ANEXO D: EVIDENCIA FOTOGRÁFICA PRUEBAS EN AMBIENTE CONTROLADO
ANEXO E: EVIDENCIA FOTOGRAFICA PRUEBAS EN AMBIENTE NO CONTROLADO
ANEXO F: EVIDENCIA FOTOGRÁFICA DE PRUEBAS DINÁMICAS

RESUMEN

De acuerdo con datos de la Agencia Nacional de Transito (ANT) revela que existe un índice de accidentes relacionados con animales en las vías, por lo tanto, el objetivo de la presente investigación fue en desarrollar un Sistema Embebido basado en FPGA y visión artificial implementado en vehículos para detectar y alertar la presencia de canes. La metodología consistió en cuatro etapas, considerando como pilar fundamental la implementación de un modelo entrenado de Red Neuronal Artificial (RNA) a través de la técnica de transferencia de aprendizaje. Para lo cual, con la finalidad de enriquecer y entrenar esta RNA, se recolectó un conjunto de 2000 imágenes, de las cuales 1500 fueron dedicadas al entrenamiento, 500 para pruebas. En la plataforma Google Colab se desarrolló el entrenamiento, y el desarrollo del algoritmo de detección se realizó en Jupyter Notebook, obteniendo dos modos de operación: normal y detector. En la etapa de validación de la RNA entrenada se obtuvo la curva de aprendizaje, en la cual, se demostró una impresionante precisión y mínima perdida de datos. Mientras que, la robustez y eficacia del sistema se sometió a pruebas de campo y se evaluaron detecciones a distintas distancias, en un rango de 1 a 10 metros, revelando que la máxima distancia de detección efectiva es de 5.5 metros. Además, como medida de precaución, se estableció una distancia mínima de 3 metros para mitigar posibles situaciones de riesgo. Las pruebas de tiempo de respuesta, realizadas en ambientes controlados y no controlados, arrojaron resultados consistentes y confiables, que mediante la prueba de Tukey se determinó un rendimiento óptimo en las jornadas matutinas. El costo de fabricación de este sistema embebido se estableció en un total de $421.55. En este contexto se concluye que el dispositivo fundamenta su funcionalidad y se recomienda el uso de otra placa hardware para optimizar el rendimiento dejando así esta investigación como una base sólida para futuros desarrollos.

Palabras clave: <SISTEMA EMBEBIDO>, <RED NEURONAL>, <TRANSFERENCIA DE APRENDIZAJE>, <MATRIZ DE PUERTA PROGRAMABLE EN CAMPO (FPGA)>, <DETECTOR DE CANES>, <ALGORITMO DE DETECCION>.

SUMMARY

According to the National Transit Agency (NTA), there is an accident rate related to animal son the roads, therefore, the present research aimed to develop an Embedded System based on FPGA and computer vision implemented in vehicles to detected and alert the presence of dogs. The methodology consisted of four stages, considering implementing a trained Artificial Neural Network (ANN) model using the transfer learning technique as a fundamental pilar. For this purpose, to enrich and train this ANN, a set of 2,000 images was collected, of which 1,500 were allocated for training and 500 for testing. The training was developed on the Google Colab platform, and the detection algorithm was developed in Jupyter Notebook, obtaining two modes of operation: normal and detector. In the validation stage of the trained ANN, the learning curve was obtained, demonstratinng impressive accuracy and minimal data los. The robustness and effectiveness of the System were subjected to field tests, conducted in controlled and uncontrolled enviroments, yielded consistent and reliable results, with the Tukey test determining optimal performance in the morning sessions. The manufacturing cost of this embedded system was established at a total of $421.55. In this context, it is concluded that the device supports its functionality, and it is recommended to use another hardware board to optimize performance, thus leaving this research as a solid foundation for future developments.

Keywords: <EMBEDDED SYSTEM>, <NEURAL NETWORK>, <TRANSFER LEARNING >, <FIELD-PROGRAMMABLE GATE ARRAY (FPGA) >, <CANINE DETECTOR >, <DETECTION ALGORITHM

INTRODUCCIÓN

En Ecuador los atropellamientos de animales en las vías es un tema poco tratado, sin embargo, gracias a los datos de la Agencia Nacional de Tránsito mencionó, entre los meses de enero a mayo del 2017 se han registrado 11.930 accidentes de los cuales 154 de ellos fueron causados por presencia de animales en las vías, a pesar de no contar con cifras de cuantos canes se atropellan a diario en el Ecuador, Emaseo, por medio de su programa animales al cielo retira un aproximado de 20 cuerpos en la vía principal de ciudad de Quito (El Comercio 2017).

El atropellar animales se puede considerar como maltrato animal de acuerdo con la Ley Orgánica de Bienestar Animal (LOBA) en Ecuador (Hernández Bustos y Fuentes Terán 2018), mediante la reforma del Art-249 del Código Orgánico Integral Penal se menciona que la persona que cause maltrato o muerte de mascotas o animales de compañía será sancionada con pena privativa de libertad de dos a seis meses (COIP, 2021).

En tal virtud, conociendo previamente estos datos se establece que el enfoque de este trabajo es el desarrollo de un dispositivo detector de canes surge con la necesidad de brindar protección a dicha mascota y que el conductor ser quien utilice dicho dispositivo. Para lo cual a lo largo del presente trabajo de integración curricular se plantea cinco capítulos desglosados de la siguiente manera: capítulo 1 diagnóstico del problema, capitulo 2 marco teórico, capitulo 3 marco metodológico, capitulo 4 propuesta y diseño del dispositivo, capitulo 5 validación del prototipo.

En el capítulo inicial se abarca diversos aspectos cruciales para el desarrollo del tema planteado, tales como identificando el planteamiento del problema, justificación teórica y práctica además de la metodología empleada para el abordaje de la temática. La segunda sección se realiza el análisis bibliográfico focalizando en los sistemas embebidos, RNA y técnicas de entrenamiento, además de algunos conceptos que sirven de base para el desarrollo de este trabajo. En el tercer capítulo, se explora la metodología a desarrollar y cuáles son los elementos que influyen para la fabricación del dispositivo tanto como la parte software (entornos y algoritmos) y hardware. El cuarto capitulo se refiere al diseño estructural del dispositivo, esquemas de conexión, y ensamblaje del dispositivo, así como también la especificación del algoritmo general del dispositivo detector de canes. Por último, en el quinto capítulo se procede al análisis minucioso y la interpretación obtenidos de las pruebas realizadas al dispositivo ya ensamblado. Concluyendo con un análisis de costos para la fabricación del prototipo.

CAPÍTULO I

1. DIAGNOSTICO DEL PROBLEMA

1.1 Planteamiento del problema

¿Es posible la creación de un sistema embebido basado en FPGA y Visión Artificial implementado en un vehículo para detectar y alertar sobre la presencia de canes en la vía?

1.2 Justificación

1.2.1 Justificación Teórica

En la actualidad la visión artificial ha tenido un crecimiento exponencial por medio de las nuevas tecnologías y el desarrollo de muchas investigaciones relacionadas con esta temática se focaliza en mejorar el bienestar y convivir del ser humano y su entorno. En este sentido, el portal web (ATIA Innovation, 2021) menciona que los algoritmos de visión artificial permiten identificar en tiempo real la posición de objetos, identifica el espacio y distancia a la que se encuentra lo que facilita la interacción con el medio que lo rodea.

Los estudiantes de la Universidad Autónoma de Bucaramanga realizaron un trabajo de titulación cuyo tema era, El diseño de un sistema de visión artificial para la detección de Zopilote negro en aeródromos, los autores proponen a mediante la investigación se demostró que un sistema de visión artificial es capaz de realizar la detección de objetos, se para pruebas se tomó como referencia una caja roja para ponerlo como obstáculo y así trazar trayectorias que ayuden a evitar obstáculos. También proponen que la velocidad a la que se requería los datos de procesamiento de imagen era una gran limitante debido a las capacidades de la batería (Vera González & Valle Ortiz, 2020).

En la ciudad de Valladolid en el 2014, Roberto Muñoz Manso realizo un trabajo de titulación con el tema Sistema de visión artificial para la detección y lectura de matrículas, el autor propone que en sistemas de detección artificial que mueven, se requiere crear un sistema que sea invariante a condiciones externas como la iluminación, rotación, etc. También menciona y valora que el sistema es efectivo a largas distancias, debido a que los casos de fallos son mínimos y los resultados son los que se esperaba (Manso, 2014).

Las tarjetas de desarrollo FPGAs para visión artificial de alta resolución nos brinda alta velocidad de procesamiento de imágenes y un ecosistema ampliado de proveedores, Las FPGA se convierten en la plataforma preferida para el diseño de sistemas embebidos, debido a que posee un gran ancho de banda y los sistemas inteligentes instalados tienen un tamaño reducido y bajo consumo de energía (Llorente, 2019). El uso FPGAs como la tarjeta de desarrollo es muy importante para la investigación, y no nos limitamos al uso común de Arduino o Raspberry Pi, con los avances tecnológicos se ha creado diseños de FPGAs que cuentan con cámara para aprendizaje automático que son destinados a sistemas embebidos inteligentes y que serán utilizados en esta investigación.

El uso de la cámara es de suma importancia pues emula la visión humana, al estar conectada a la tarjeta de desarrollo su principal objetivo es la adquisición de imágenes para ser leídas por el algoritmo de visión artificial el mismo que se encarga de procesar secuencialmente todas las tareas de entrenamiento que son indicadas por el sistema operativo a través del software. (Pérez Cueto, 2009). En este sentido se puede entender que en la etapa de envió y adquisición de imágenes exista un pequeño retardo de tiempo hasta la visualización de esta, en el caso de la investigación hasta que el sistema de alerta se active acorde la detección de canes en la vía.

Por lo tanto, de acorde a la información recabada y citada se determina que es factible la realización de este trabajo de investigación, siendo un sistema embebido basado en FPGA y visión artificial implementado en un vehículo para la detección y alerta de canes en las vías que en la actualidad no se puede encontrar en comercialización en Ecuador, dando paso así a futuros trabajos de investigación y desarrollo de sistemas de detección que se vean interesados en mejorar o a su vez desarrollar su propio sistema embebido partiendo de la utilización de FPGA y visión artificial como herramienta fundamental para su desarrollo.

1.2.2 Justificación aplicativa

El tema de investigación busca diseñar y construir un prototipo de un sistema embebido basado en FPGA y visión artificial implementado en un vehículo para la de detección y alerta de canes en las vías debido a la falta de desarrollo tecnológico orientado a esta problemática.

En el transcurso del trabajo de titulación conforme avance el proceso de investigación los elementos que conforman el sistema embebido de prevención serán seleccionados acorde a los requerimientos planteados. Para ello es necesario considerar varios aspectos y parámetros relacionados con el ambiente de trabajo, sin olvidar aquellos materiales eléctricos/electrónicos necesarios al momento de realizar el diseño y creación del sistema. En el caso de la

implementación será necesario considerar el alquiler de un vehículo que nos brinde todas las condiciones adecuadas para que el sistema funcione de manera idónea.

El desarrollo del dispositivo detector de canes en las vías implementado en vehículos con FPGA en primera instancia se desarrolla en cuatro etapas. La primera etapa se encarga de establecer los requerimientos necesarios para llevar a cabo el dispositivo. La segunda etapa se basa en la recolección de datos, programación y entrenamiento del algoritmo de visión artificial para que pueda ser capaz de detectar la presencia de canes en las vías a partir de un conjunto de imágenes relacionadas con este propósito, misma que se llevará a cabo mediante un computador y herramientas de programación que permitan implementar este tipo de algoritmos y obtener un modelo de visión artificial entrenado para tal fin.

La tercera etapa se basa en reconstruir o trasladar el modelo de visión artificial entrenado, para detectar canes en las vías, sobre una tarjeta de Field Programmed Gate Array (FPGA), que será el núcleo del sistema embebido de esta investigación, misma que recibirá las imágenes en tiempo real desde una cámara conectada a dicha tarjeta. En esta etapa también se realiza la implementación del sistema de advertencia hacia el conductor, el mismo que puede ser alertado por cualquiera de las siguientes formas: Audible, Táctil o una combinación de ambos como la Vibro acústica.

La última etapa se centra en la realización de pruebas de evaluación del sistema desarrollado. Las pruebas deberán ser desarrolladas en ambientes controlados y no controlados, como, por ejemplo: ambientes de iluminación controlada y no controlada, así como también la participación segura de mascotas que sirvan para evaluar el número de aciertos positivos, negativos, falsos positivos, falsos negativos, distancia de detección en condiciones de vehículo detenido y en movimiento. Se debe recalcar que las pruebas a bordo del vehículo serán ejecutadas siempre y cuando existan todas las garantías de que las mismas podrán ser ejecutadas bajo condiciones de seguridad para los sujetos participantes.

1.3 Objetivos

1.3.1 Objetivo General

Desarrollar un sistema embebido basado en FPGA y Visión Artificial implementado en un vehículo para detectar y alertar sobre la presencia de canes en la vía.

1.3.2 Objetivos Específicos

- Investigar sobre los sistemas de detección y alerta de canes en la vía desarrollados y reportados en literatura, así como los componentes hardware y software que lo conforman, normativas de uso, características y vulnerabilidades que pueden tener.

- Definir los requerimientos operativos que deberá cumplir el prototipo de sistema embebido de detección y alerta de presencia de canes en las vías, así como los componentes hardware y herramientas software que se utilizarán en su desarrollo.

- Desarrollar el modelo entrenado para la detección de canes en la vía mediante algoritmos de visión artificial orientado a utilizar la menor cantidad de recursos de procesamiento y compatible con plataformas embebidas como FPGAs.

- Implementar el modelo entrenado para la detección de canes en la vía sobre una tarjeta de FPGA dotada de una cámara para la adquisición de imágenes en tiempo real, así como del sistema de alerta hacía el conductor en un vehículo real.

- Realizar las pruebas necesarias, bajo condiciones controladas y no controladas de iluminación, para evaluar las tasas de detección y alerta de presencia de canes en la vía en condiciones de tránsito vehicular seguras para los sujetos participantes.

CAPÍTULO II

2. MARCO TEÓRICO

En este segundo capítulo se describe conceptos relaciones con sistemas embebidos, inteligencia artificial, visión artificial, además se detalla que entornos de software se utiliza para la programación de tarjetas de desarrollo, para finalizar se detalla las redes neuronales desde su concepto hasta sus procesos de entrenamiento como también la implementación de los actuadores y sistemas de alarmas vehiculares.

2.1 Generalidades de los Sistemas Embebidos

Es un sistema basado en microprocesadores que cumple una tarea específica, sometida a una serie de requerimientos delimitados por el usuario ya sea en el diseño o en la funcionalidad.

"Un sistema embebido o sistema incrustado es un sistema cuya función principal no es computacional, pero es controlado por un computador integrado, puede ser un microcontrolador o un microprocesador. La palabra embebido implica que se encuentra dentro del sistema general"(David y Pérez 2009).

Los sistemas embebidos poseen una estructura compuesta del hardware similar al de un computador, diseñado específicamente para satisfacer los requerimientos inmersos en el software los mismos que serán controlados y ejecutados por un sistema operativo (Cayssials 2014).

En la *Tabla 2-1.*, se detallas algunas características que diferencian un sistema embebido de un sistema computacional tradicional.

Tabla 2-1. Características diferenciales entre los sistemas embebidos y los sistemas tradicionales.

Sistema Embebido	Sistema Computacional Tradicional
Ejecución de tareas específicas	Ejecución de cierta cantidad de programas
Limitación en su diseño	Alta demanda de recursos en el diseño
Bajo costo de implementación	Elevado costo de implementación
Tamaño reducido	Tamaño prominente
Baja demanda energética	Alta demanda energética

Fuente: (Salas 2015).

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

2.2 Elementos Hardware de los Sistemas Embebidos para Visión Artificial

Como en todo sistema, el termino hardware se refiere a la parte tangible que constituye un sistema embebido, que en conjunto con el software realizan tareas específicas. Dichos componentes se distinguen de los sistemas tradicionales por su tamaño, funcionalidad, capacidad de procesamiento.

Todo sistema embebido dispone de un hardware, constituido físicamente por circuitos integrados interconectados sobre tarjetas impresas que conforman la memoria, unidad de control, unidad lógico-aritmético y las unidades de entrada/salida (sensores y actuadores).

La Ilustración 2-1., describe de manera general los principales componentes hardware que formarían parte dentro del Sistema Embebido si los requerimientos consideran Visión Artificial.

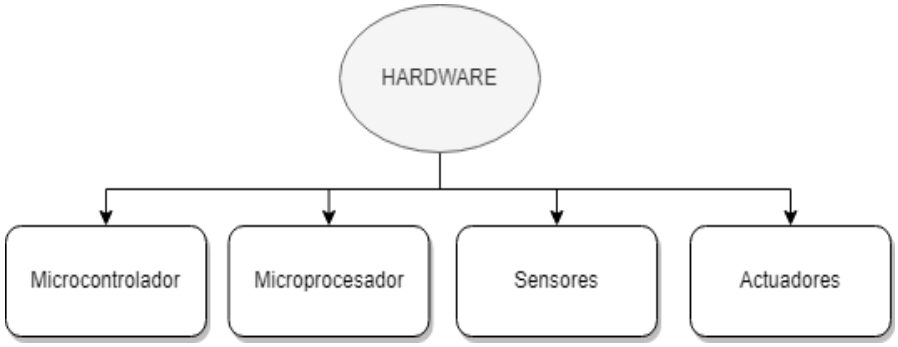


Ilustración 2-1. Principales componentes hardware de un Sistema Embebido.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

2.2.1 Microprocesador

Según Martín del Brío, (1999) se conoce como el "componente electrónico que realiza en una única pastilla el procesador (CPU) de una maquina programable de tratamiento de la información."
El microprocesador es considerado por David & Pérez, (2009) como un componente LSI que realiza una gran cantidad de funciones o tareas en una sola pieza de circuito integrado. Por ello se puede considerar como el puesto de mando de todo el sistema embebido siendo el que posee un control mayoritario en todo el proceso incluido en la velocidad de funcionamiento.

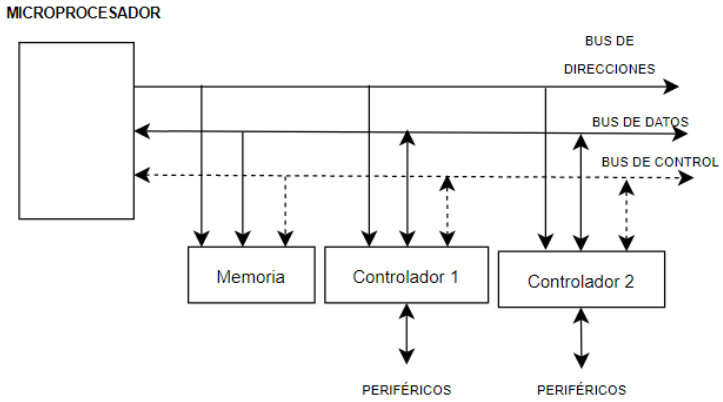


Ilustración 2-2. Representación general de un Microprocesador utilizado en los S.E.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

2.2.2 Microcontrolador

"Un microcontrolador es un circuito integrado programable que contiene todos los componentes necesarios para controlar el funcionamiento de una tarea determinada" (Palacios, Remiro y López 2014).

Según Aguayo, (2004), el microcontrolador es en definitiva un circuito integrado que incluye todos los componentes que se pueden considerar en un computador. Debido a su reducido tamaño es posible montar el controlador en el propio dispositivo al que gobierna (controlador embebido).

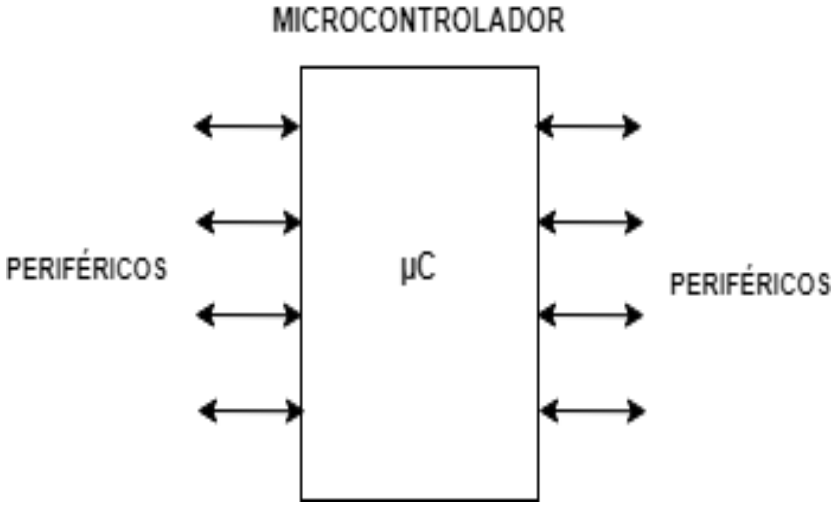


Ilustración 2-3. Descripción general de la estructura de un Microcontrolador básico

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

2.2.3 Tarjetas de Desarrollo

De acuerdo con los autores González & Silva (2013, p. 2), las tarjetas de desarrollo son herramienta utilizadas para el diseño y prototipado rápido de sistemas digitales o analógicos, son consideradas un elemento útil para mejorar el proceso de diseño al reducir el tiempo de prueba del diseño.

2.2.3.1 Tipos de Tarjetas de Desarrollo

- **Raspberry Pi**

  Raspberry Pi es una computadora de bajo costo y tamaño compacto que está diseñada para promover el aprendizaje y la experimentación en el campo de la informática. Se trata de una placa de circuito impreso que incluye un procesador, memoria, puertos de entrada y salida, y una variedad de interfaces que permiten conectarlo a otros dispositivos y componentes.

- **FPGA**

  Una FPGA (Field-Programmable Gate Array) es un tipo de dispositivo electrónico programable que se utiliza para implementar circuitos digitales. Consiste en una matriz de bloques lógicos configurables y elementos de interconexión programables. Estos elementos permiten a los diseñadores configurar y reconfigurar la funcionalidad del hardware después de su fabricación (Woods et al. 2009).

Comparativa entre Tarjetas de Desarrollo.

Mediante la *Tabla 2-2.*, se hace una retroalimentación de las principales consideraciones de las tarjetas de desarrollo más comunes en la actualidad.

Tabla 2-2. Tabla comparativa sobre los atributos de las Tarjetas de Desarrollo.

Características	Raspberry Pi	FPGA
Procesador	ARM Cortex-A/72	ARM Cortex-M3
Precio	$100	$180
Entradas Analógicas	N/A	N/A
Entradas Digitales	26-40	132-540 terminales
Voltaje de entrada	5v	1.2-5v

Características	Raspberry Pi	FPGA
Red	Eth, Wifi	Eth, Wifi
RAM	1.2 o 4 Gb	75 a 1355 KB
Tiempo de procesamiento	Alto	Alto
Gráfica		

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023

2.2.4 Sensores

Un sensor es un dispositivo que convierte una magnitud física en una señal eléctrica o óptica que puede ser interpretada y procesada por un sistema electrónico. Los sensores pueden medir una amplia variedad de magnitudes físicas, como temperatura, presión, humedad, posición, aceleración, velocidad, entre otras. Los sensores son ampliamente utilizados en una gran variedad de aplicaciones, desde la industria automotriz y aeroespacial hasta la medicina y la agricultura, y son esenciales para la recopilación de datos en tiempo real y el control de procesos automatizados (Dorf y Bishop 2005).

El dispositivo que convierte una magnitud física en una señal eléctrica o óptica que puede ser interpretada y procesada por un sistema electrónico. Los sensores pueden medir una amplia variedad de magnitudes físicas, como temperatura, presión, humedad, posición, aceleración, velocidad, entre otras. Los sensores son ampliamente utilizados en una gran variedad de aplicaciones, desde la industria automotriz y aeroespacial hasta la medicina y la agricultura, y son esenciales para la recopilación de datos en tiempo real y el control de procesos automatizados (Dorf y Bishop 2005).

Dunn, (2011) , presenta una conceptualización personal de una posible clasificación de los sensores presentes en la industria y por medio de la Tabla 2-3., se viraliza dicha información.

Tabla 2-3. Descripción teórica de los tipos de sensores considerados en los S.E.

Tipos de sensores	Descripción
Sensores activos	Se dice de aquellos que emiten una señal para detectar la presencia de un objeto.
Sensores pasivos	Son aquellos que detectan la radiación electromagnética que emiten los objetos.
Sensores analógicos	Miden una magnitud física continua y proporcionan una señal de salida proporcional a la magnitud medida.
Sensores digitales	Se encargan de medir una magnitud física y proporcionan una salida en forma de señal digital.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023

2.2.4.1 Sensor de Imagen

Es el elemento responsable de convertir la luz en señales eléctricas, permitiendo capturar y digitalizar imágenes. Estos sensores son ampliamente utilizados en cámaras digitales, video cámaras, escáneres y otros dispositivos de captura de imágenes. La tecnología más comúnmente utilizada en estos sensores es el CCD (dispositivos de acoplamiento de carga). Que combina en un solo chip los componentes fotosensibles. Así como la lógica de control y la circuitería asociada. En estos dispositivos, la señal eléctrica transmitida por los fotodiodos depende de la intensidad luminosa que reciben, su espectro y el tiempo de integración, es decir, el tiempo durante el cual los fotodiodos son sensibles a la luz incidente (Tello 2006).

2.2.4.2 Dispositivos de captura de imagen

Son aparatos electrónicos que posibilitan la obtención imágenes o fotografías en formato digital estas pueden ser estáticas o en movimiento, para posterior procesamiento o visualización (Martínez Verdú 2002). Para una buena adquisición de imágenes se debe considerar diversos factores, tales como la luminosidad, la interferencia, el fondo de la imagen, la resolución, para lograr un procesamiento y estudio de la imagen en un sistema de visión artificial.

En la *Tabla 2-4.,* se resume las características y funcionalidades de algunos dispositivos de captura de imagen más comunes.

Tabla 2-4. Características y funcionalidades de los dispositivos de captura de imagen

Dispositivo	Descripción
Cámaras digitales	Son portátiles que utilizan un sensor de imagen, Entre las características tenemos la resolución, zoom óptico, pantalla LCD, almacenamiento de datos, modos de exposición, conectividad, batería.(Cosoi P. 2002)
Teléfonos móviles	En la mayoría de los teléfonos modernos viene equipados con una cámara integrada, que son capaces de capturar imágenes de alta calidad e incluso grabar videos.
Escáneres	Es un dispositivo que se utiliza para digitalizar imágenes impresas o documentos. Estos están disponibles en una variedad de resoluciones
Cámaras de seguridad	Dispositivos que se utilizan para capturar imágenes en lugares públicos o privados. Pueden ser monitoreadas en tiempo real o pueden grabar para su posterior revisión.(Dominguez et al. 2016)
Cámaras deportivas	Se utilizan para capturar imágenes y videos de actividades deportivas extremas o al aire libre. Estas son resistentes y están diseñadas para soportar condiciones adversas

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023

2.2.5 Actuadores

Es un componente o dispositivo que convierte una señal de entrada en una acción física en un sistema automatizado. Los actuadores pueden ser eléctricos, hidráulicos, neumáticos o mecánicos, y se utilizan en una amplia variedad de aplicaciones, como el control de válvulas, la automatización de procesos industriales, el control de movimientos en robots y sistemas de maquinaria, y en la industria del automóvil, entre otros (Corona, Abarca y Mares 2014).

Los actuadores eléctricos convierten una señal eléctrica en una acción mecánica, como un motor que convierte la electricidad en movimiento. Los actuadores hidráulicos y neumáticos utilizan fluidos comprimidos para mover un objeto o realizar una acción, como un cilindro hidráulico que mueve un brazo robótico. Los actuadores mecánicos, como los solenoides, utilizan un campo magnético para realizar una acción mecánica, como abrir o cerrar una válvula.

En la Tabla 2-5., se maneja la información detallada de los tipos de actuadores más utilizados en la actualidad dentro de un sistema embebido.

Tabla 2-5. Descripción conceptual y gráfica de los tipos de actuadores.

Actuadores	**Descripción**	**Gráfica**
Mecánicos	Actuadores que generan movimiento físico.	
	- Motores de corriente continua (DC) - Motores de paso - Servomotores - Motores de corriente alterna (AC) - Actuadores piezoeléctricos	
Electromagnéticos	Actuadores que utilizan campos electromagnéticos.	
	- Solenoides - Electroimanes - Válvulas electromagnéticas	
Ópticos	Actuadores que generan luz o trabajan con luz.	
	- LED (Diodo Emisor de Luz) - Pantallas	
Sonido	Actuadores utilizados para producir sonido.	
	- Altavoces	

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023

2.3 Herramientas Software para Visión Artificial en Sistemas Embebidos

Los sistemas embebidos para la visión artificial requieren de software especializado para realizar el procesamiento de imágenes y realizar tareas de reconocimiento mediante el análisis visual. A continuación, se presenta algunos de los elementos software utilizados en los sistemas embebidos para la visión artificial.

2.3.1 Técnicas de Inteligencia artificial

La inteligencia artificial (IA) abarca una amplia gama de áreas que se desarrollan bajo el concepto de IA. Cada uno de estos cuenta con técnicas y herramientas para llegar a sus objetivos. En la *Tabla* 2-6., se describen las técnicas de inteligencia artificial utilizadas en la actualidad para el reconocimiento de patrones.

Tabla 2-6. Técnicas de Inteligencia Artificial para reconocimiento de patrones.

Técnica	Descripción
Algoritmos Genéticos	Técnicas basadas en la selección natural y evolución biológica, utilizadas para la optimización y búsqueda de soluciones.
Lógica Difusa	Método que maneja la imprecisión y la incertidumbre, permitiendo la representación y razonamiento en situaciones ambiguas.
Aprendizaje Automático o Machine Learning	Algoritmos y modelos que permiten a los sistemas aprender automáticamente a partir de datos y mejorar su rendimiento con la experiencia.
Visión por Computadora o Visión Artificial	Técnicas para la interpretación y análisis de imágenes y videos, utilizadas en aplicaciones de reconocimiento facial, detección de objetos, etc.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023

2.3.1.1 Visión por Computadora o Visión Artificial

La visión artificial, también conocida como visión por computador, es la ciencia que se encarga de programar un ordenador para procesar imágenes y videos, y en algunos casos, comprender su contenido (Alvear Puertas et al. 2017). En otras palabras, es la capacidad de las máquinas de ver y analizar el mundo que la rodea de manera similar a la que lo hacen los seres humanos.

De acuerdo con Sanabria y Archila (2011, p. 180), la visión artificial se realiza de manera similar al proceso de la visión humana. La entrada básica es una imagen capturada por una cámara. La imagen se representa como una matriz de puntos, donde cada punto corresponde a un valor de una función bidimensional f (x, y), donde x e y son las coordenadas espaciales y f representa la intensidad o brillo de la imagen. En el caso de las imágenes en blanco y negro, f es un valor único, mientras que, para las imágenes en color, se utiliza el modelo de color aditivo RGB, que combina tres matrices de puntos.

En la *Ilustración 2-4.* se muestra el sistema de visión artificial que consta de algunos componentes principales, entre los que están: adquisición de imágenes, procesamiento, iluminación, objeto, reconocimiento y la interpretación de la escena.

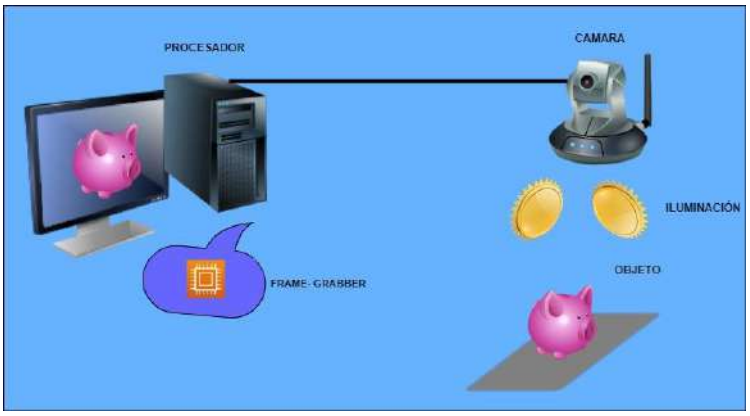


Ilustración 2-4. Componentes de un sistema de visión artificial

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2022

Procesamiento de la imagen. - Según, (Juan y Chacón 2011) son operaciones para ajustar los datos de una imagen y mejorar su análisis en etapas posteriores. Mientras que, (Esqueda 2002) menciona que las técnicas que trabajan directamente con los valores de los píxeles de la imagen conforman el procesamiento espacial. El procesamiento de imágenes se rige acorde a la siguiente ecuación.

$$S(x,y) = F(I(x,y))$$

Donde: $I(x,y)$: imagen original, $S(x,y)$: Imagen resultante, F : Transformación

En el procesamiento de imágenes, engloba un conjunto de técnicas y algoritmos, así como también de ciencias en la computación, matemáticas y astronomía, que tienen como finalidad mejorar la calidad, extraer información relevante o transformar de diversas maneras la imagen (Cuevas, Zaldivar y Perez 2012).

En el procesamiento de imágenes, engloba un conjunto de técnicas y algoritmos, así como también de ciencias en la computación, matemáticas y astronomía, que tienen como finalidad mejorar la calidad, extraer información relevante o transformar de diversas maneras la imagen (Cuevas, Zaldivar y Perez 2012).

En este sentido, se puede desarrollar operaciones aritméticas más comunes son la suma, resta, multiplicación y división. Sin embargo, es importante destacar que para realizar cualquiera de estas es necesario que el par de imágenes tengan el mismo tamaño (Esqueda 2002).

En la *Ilustración 2-5*. se muestra algunas de las técnicas más comunes de procesamiento de imágenes para extraer características, entre las cuales destacan las siguientes.

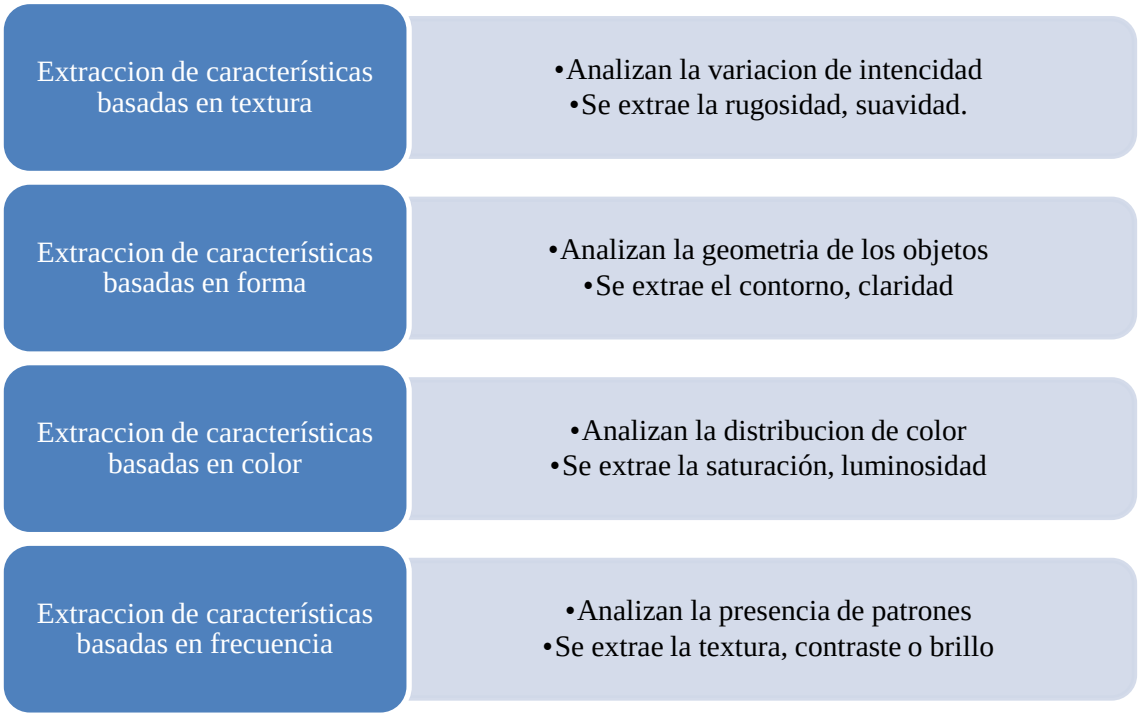


Ilustración 2-5. Técnicas para el procesamiento de imágenes

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023

2.3.1.2 Machine Learning

El Machine Learning es un enfoque de la inteligencia artificial que permite a las máquinas aprender automáticamente a partir de datos y mejorar su rendimiento en una tarea específica sin necesidad de ser programadas explícitamente. En lugar de escribir reglas y algoritmos específicos, el Machine Learning se basa en algoritmos y modelos que pueden aprender de los datos y realizar predicciones o tomar decisiones basadas en esa experiencia aprendida (Goodfellow et al. 2016).

El Machine Learning es un campo de estudio que se enfoca en el desarrollo de algoritmos y modelos que brindan la facilidad de aprender y mejorar automáticamente a través de la experiencia. Dentro del amplio campo del Machine Learning, existen varios subcampos que se centran en diferentes enfoques y aplicaciones. En la Tabla 2-7., se proporciona una descripción de los subcampos que se pueden encontrar en la actualidad.

Tabla 2-7. Subcampos existentes dentro del Machine Learning y su descripción.

Subcampo	Descripción
Aprendizaje supervisado	Los modelos se entrenan utilizando ejemplos de entrada y salida esperada.
Aprendizaje no supervisado	Los modelos se entrenan en conjuntos de datos sin etiquetas para descubrir patrones o estructuras ocultas.
Aprendizaje por refuerzo	Los modelos aprenden a través de la interacción con un entorno y retroalimentación en forma de recompensas.
Aprendizaje profundo o Deep Learning.	Se enfoca en entrenar redes neuronales profundas con múltiples capas de procesamiento.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023

- **Aprendizaje supervisado**

Según Bishop, (2006), es un enfoque del Machine Learning donde se utilizan datos de entrenamiento que contienen ejemplos etiquetados. En este enfoque, se proporciona a la máquina un conjunto de datos de entrada junto con las salidas deseadas correspondientes. La tarea del algoritmo de aprendizaje es construir un modelo o función que pueda mapear las entradas a las salidas correctas. El aprendizaje supervisado se basa en el principio de que los datos de entrenamiento contienen información suficiente para inferir una regla general que puede usarse para predecir las salidas correspondientes a nuevas instancias de datos. El modelo resultante se evalúa utilizando datos de prueba o validación para medir su capacidad de generalización.

- **Aprendizaje no supervisado**

El objetivo del aprendizaje no supervisado es descubrir patrones, estructuras o relaciones ocultas en los datos sin la guía de salidas conocidas. Los algoritmos de aprendizaje no supervisado buscan agrupar los datos en conjuntos o categorías similares o extraer características y representaciones significativas de los datos (Bishop, 2006).

- **Deep Learning**

Es un subcampo del aprendizaje automático que se enfoca en el entrenamiento de redes neuronales profundas para aprender a realizar tareas complejas, como la clasificación de imágenes o el reconocimiento de voz. En lugar de requerir características específicas y diseñadas a mano para cada tarea, el Deep Learning utiliza capas de procesamiento para aprender características y patrones relevantes de manera automática a partir de los datos de entrada. Debido a su capacidad

para aprender de manera jerárquica, el Deep Learning ha demostrado ser extremadamente efectivo en una variedad de tareas de aprendizaje automático, superando a otros métodos tradicionales en muchos casos (Goodfellow et al. 2016).

2.3.2 Redes Neuronales Artificiales

En la actualidad, las redes neuronales artificiales se han convertido en una herramienta fundamental en el campo del aprendizaje automático y la inteligencia artificial. Estas redes están inspiradas en el funcionamiento del cerebro humano y se utilizan para resolver una amplia gama de problemas en diversas áreas, como reconocimiento de imágenes, procesamiento del lenguaje natural, predicción y toma de decisiones. Una red neuronal artificial es un conjunto de algoritmos y modelos matemáticos que imitan la forma en que el cerebro humano procesa y analiza información. La red neuronal está compuesta por capas de neuronas interconectadas que reciben entradas, procesan la información y producen salidas. Cada neurona toma una entrada, realiza una operación matemática y produce una salida, que se transmite a las neuronas de la siguiente capa. El aprendizaje de una red neuronal se produce mediante la adaptación de los pesos sinápticos de las conexiones entre las neuronas, lo que permite que la red se ajuste a los datos de entrenamiento y pueda generalizar y realizar predicciones precisas en nuevos datos(Aggarwal C, 2018).

2.3.2.1 Neurona Artificial

La red neuronal artificial está compuesta por siete elementos esenciales como se muestra en la *Ilustración 2-6*, estos son aplicables para la neurona en diferentes roles, ya sea como entrada, salida o en las capas intermedias (Jaya 2022).

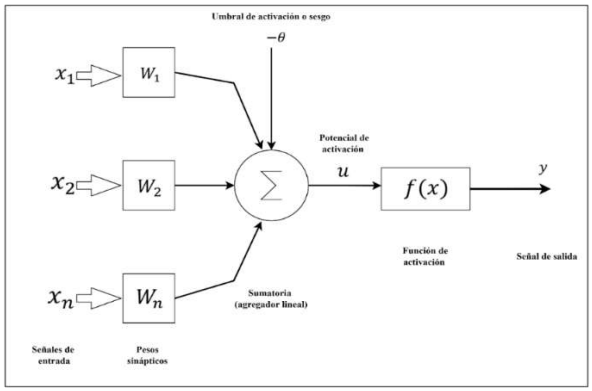


Ilustración 2-6. Descripción Gráfica de la constitución de la Neurona.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

La Tabla 2-8., **se** encarga de describir los componentes que forman parte de las neuronas acompañado de la representación y descripción de estas.

Tabla 2-8. Principales componentes que conforman la Neurona.

Componentes	Representación	Descripción
Señales de entrada	$\{x_1, x_2, x_3 \ldots\ldots, x_n\}$	Señales que provienen del entorno externo, y representan los valores de las variables de una aplicación particular
Pesos Sinápticos	$\{w_1, w_2, w_3 \ldots\ldots, w_n\}$	Ponderaciones de las variables de entrada para cuantificar su relevancia de la funcionalidad de la neurona
Agregador Lineal	$\sum$	Sumatoria de los pesos de las señales de entrada para generar una señal de activación
Umbral de activación o sesgo	θ	Variable que especifica que umbral debe tener el agregador lineal para producir un valor de activación
Potencial de activación	u	Valor de la diferencia entre el agregador lineal y el umbral de activación
Función de activación	$f(u)$	Función que limita a la salida de la neurona dentro de un rango razonable.
Señal de salida	y	Resultado final generado por la neurona dado un conjunto de señales de entrada. También se puede usar como señal de entrada para otras neuronas interconectadas.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

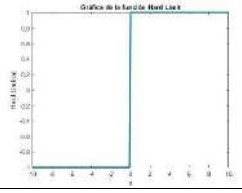
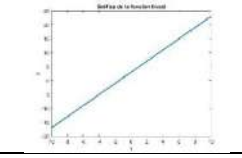
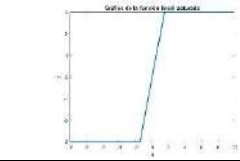
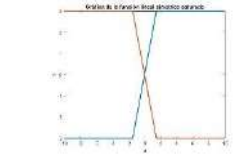
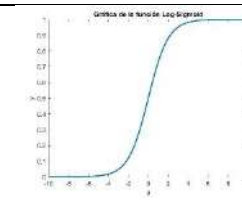
- ***Funciones de Activación***

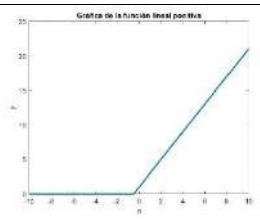
Las funciones de activación son una parte fundamental de las redes neuronales artificiales, que se utilizan para modelar relaciones no lineales en los datos de entrada. Estas funciones aplican una transformación no lineal a los datos de entrada, lo que les permite representar y aprender patrones complejos en los datos. Las funciones de activación también son utilizadas para introducir la no

linealidad en las redes neuronales, lo que les permite modelar relaciones no lineales en los datos de entrada (Goodfellow et al., 2016).

En la Tabla 2-9., se puede apreciar las funciones de activación más comunes acompañadas de su relación entrada salida y su respectiva gráfica.

Tabla 2-9. Principales Funciones de Activación para las Redes Neuronales Artificiales.

Nombre	**Relación entrada / salida**	**Gráfico**
Hard Limit	$a = 0 \quad n < 0$ $a = 0 \quad n \leq 0$	
Symmetrical Hard Limit	$a = -1 \quad n < 0$ $a = +1 \quad n \leq 0$	
Linear	$a = n$	
Saturating Linear	$a = -1 \quad n < 0$ $a = +1 \quad 0 \leq n \leq 0$ $a = 1 \quad n > 0$	
Symmetric Saturating Linear	$a = -1 \quad n < -1$ $a = n \quad -1 \leq n \leq 1$ $a = 1 \quad n > 1$	
Log-Sigmoid	$a = \frac{1}{1 + e^{-n}}$	
Hyperbolic Tangent Sigmoid	$a = \frac{e^{n} - e^{-n}}{e^{n} + e^{-n}}$	

Nombre	Relación entrada / salida	Gráfico
Positive Linear	$a = 0 \quad n < 0$ $a = n \quad n \geq 0$	

Fuente:(Taipe 2021)

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

2.3.2.2 Características de las RNA

Las Redes Neuronales Artificiales tienen varias características que les permite realizar actividades de entrenamiento y aprendizaje a partir de información o históricos de entrada, mediante algoritmos de entrenamiento. En la *Ilustración 2-7*, enlista de manera concisa las principales características que se ha mencionado anteriormente, sabiendo que el aprendizaje adaptativo se refiere a la capacidad de una RNA para ajustar sus parámetros internos o su estructura en función de los datos de entrada y las señales de retroalimentación.

Al hablar de la adaptación a las nuevas tecnologías hace referencia a la flexibilidad y adaptabilidad que tienen las RNA a diferentes entornos y tecnologías, pudiendo ser entrenadas y utilizadas en una amplia variedad de aplicaciones; esto se lleva a cabo por la capacidad de aprender y organizar (autoorganización) sus propias representaciones internas de los datos, pudiendo identificar patrones y estructuras subyacentes en los datos sin necesidad de una supervisión explícita. La operación en tiempo real permite a las RNA procesar y responder a los datos de entrada en tiempo real, lo que significa que pueden tomar decisiones o realizar predicciones en tiempo casi instantáneo, todo esto hace a la RNA que cuente con la capacidad de manejar y recuperarse de fallos o perturbaciones en los datos de entrada. Pueden ser robustas frente al ruido, datos faltantes o errores en los datos de entrada, esta característica garantiza un rendimiento confiable incluso en condiciones imperfectas.

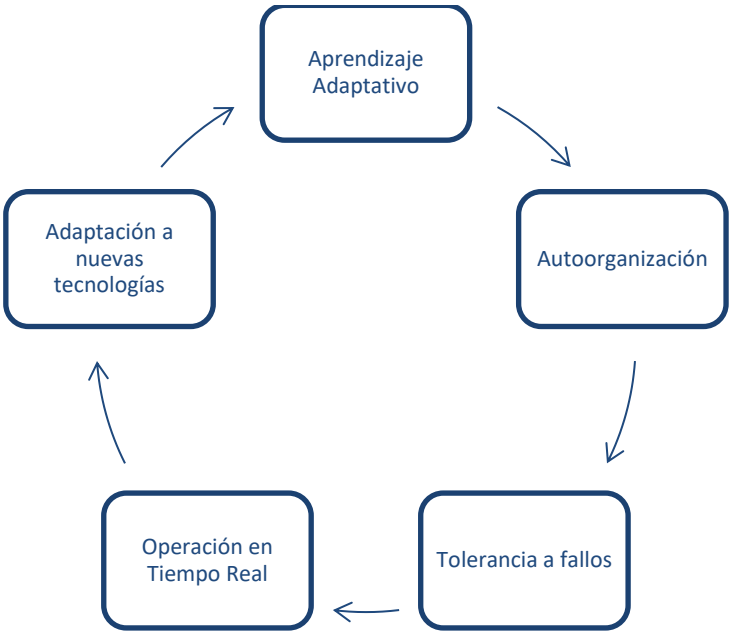


Ilustración 2-7. Principales características de las Redes Neuronales Artificiales.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

2.3.2.3 Elementos de una RNA

Las redes neuronales artificiales están compuestas por diferentes elementos interconectados que trabajan en conjunto para lograr un funcionamiento eficiente. El elemento fundamental en una red neuronal artificial es la neurona artificial o nodo. Cada neurona recibe una o más entradas, las procesa mediante una función de activación y produce una salida. La función de activación puede ser lineal o no lineal y determina la respuesta de la neurona en función de las entradas recibidas. En la Tabla 2-10., se describe de mamera detallada los elementos fundamentales que hacen parte de la estructura de una Red Neuronal Artificial. La presencia de cada una de ellas depende de la arquitectura y requerimientos que deba cumplir la RNA.

Tabla 2-10. Principales elementos que conforman una Red Neuronal Artificial.

Elemento	Descripción
Neurona	Unidad básica encargada del procesamiento en una RNA, la misma que recibe enésimas entradas, realiza una operación y genera enésimas salidas.
Capa	Se dice del conjunto de neuronas que procesan una entrada para producir una salida, se las conoce también como niveles de abstracción.
Conexión	Vinculo existente entre la salida de una neurona con la entrada de otra, permitiendo que la información fluya bidireccionalmente entre ellas.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

2.3.2.4 Arquitecturas de RNAs

Las arquitecturas de las redes neuronales se refieren a la estructura y organización de las capas y conexiones de una red neuronal artificial. Estas arquitecturas determinan cómo se procesa la información y se realiza el aprendizaje dentro de la red.

En la Tabla 2-11., se refleja la información sobre las arquitecturas de las redes neuronales más comunes que se pueden encontrar en la actualidad en el campo de la inteligencia artificial.

Tabla 2-11. Arquitectura para el Entrenamiento de las Redes Neuronales Artificiales.

Tipo de RNA	Descripción
Perceptrón	Red Neuronal de una sola capa que se utiliza para clasificar patrones linealmente separables.
Redes Neuronales Multicapa	Esta red cuenta con una o varias capas ocultas que se utiliza para clasificar patrones no linealmente separables.
Redes Neuronales Convolucionales (CNN)	RNA especializada en el procesamiento de imágenes, se utilizan capas de convolución y pooling para extraer características y reducir la dimensión de la imagen.
Redes Neuronales Recurrentes (RNN)	Red Neuronal en la que la salida de una neurona se retroalimenta como entrada a la misma o a otras nuevas neuronal de la red. Se utiliza para procesas datos secuenciales.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

- **Redes Neuronales Convolucionales (CNN)**

Las CNNs se inspiran en la organización del sistema visual en los seres humanos y utilizan capas de convolución para extraer características relevantes de las imágenes. Estas capas convolucionales aplican filtros convolucionales a las imágenes de entrada, lo que permite detectar patrones locales como bordes, texturas y formas. A medida que la información se propaga a través de la red, las capas convolucionales se encargan de extraer características de mayor nivel de abstracción (Goodfellow et al. 2016). Según la *Tabla 2-12.*, se puede apreciar que son varias las capas inmiscuidas dentro una red neuronal convolucional.

Tabla 2-12. Descripción de las capas existentes dentro de las Redes Neuronales Convolucionales (CNN).

Capa	Descripción
Capa de entrada	Recibe las imágenes o datos de entrada.
Capas convolucionales	Aplican filtros convolucionales para extraer características locales de las imágenes.
Capas de activación	Aplican una función de activación no lineal para introducir la no linealidad en la red.
Capas de agrupación (pooling)	Reducen la dimensionalidad de las características, preservando las más relevantes.
Capas de normalización	Normalizan los valores de las características para mejorar la estabilidad del entrenamiento.
Capas completamente conectadas	Realizan la clasificación o regresión final basada en las características extraídas.
Capa de salida	Produce la salida final de la red neuronal, que puede ser una clasificación, una regresión, etc.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

2.3.3 Técnicas para el entrenamiento de RNA

En la Tabla 2-13., se presenta una visión general de las principales técnicas de entrenamiento utilizadas en las redes neuronales. Cada técnica juega un papel importante en el proceso de entrenamiento y puede ser aplicada de acuerdo con las necesidades específicas del modelo y el problema en cuestión.

Tabla 2-13. Técnicas para el entrenamiento de las Redes Neuronales Artificiales.

Técnica	Descripción
Retropropagación	Propagación del error hacia atrás para ajustar los pesos de las conexiones. Utiliza el gradiente descendente (Borracci y Rubio 2003).
Regularización	Agrega términos para evitar el sobreajuste, como la regresión de peso (L1 y L2) y la deserción.
Mini lotes (Mini-batch)	Entrena la red neuronal en mini lotes en lugar de en lotes completos para mejorar la eficiencia y convergencia.

Aprendizaje por transferencia (Transfer Learning)	Utiliza pesos pre-entrenados en una tarea para entrenar una red neuronal en una tarea relacionada. (Fei et al. 2022).

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

2.3.3.1 Algoritmo de Aprendizaje por Transferencia (Transfer Learning)

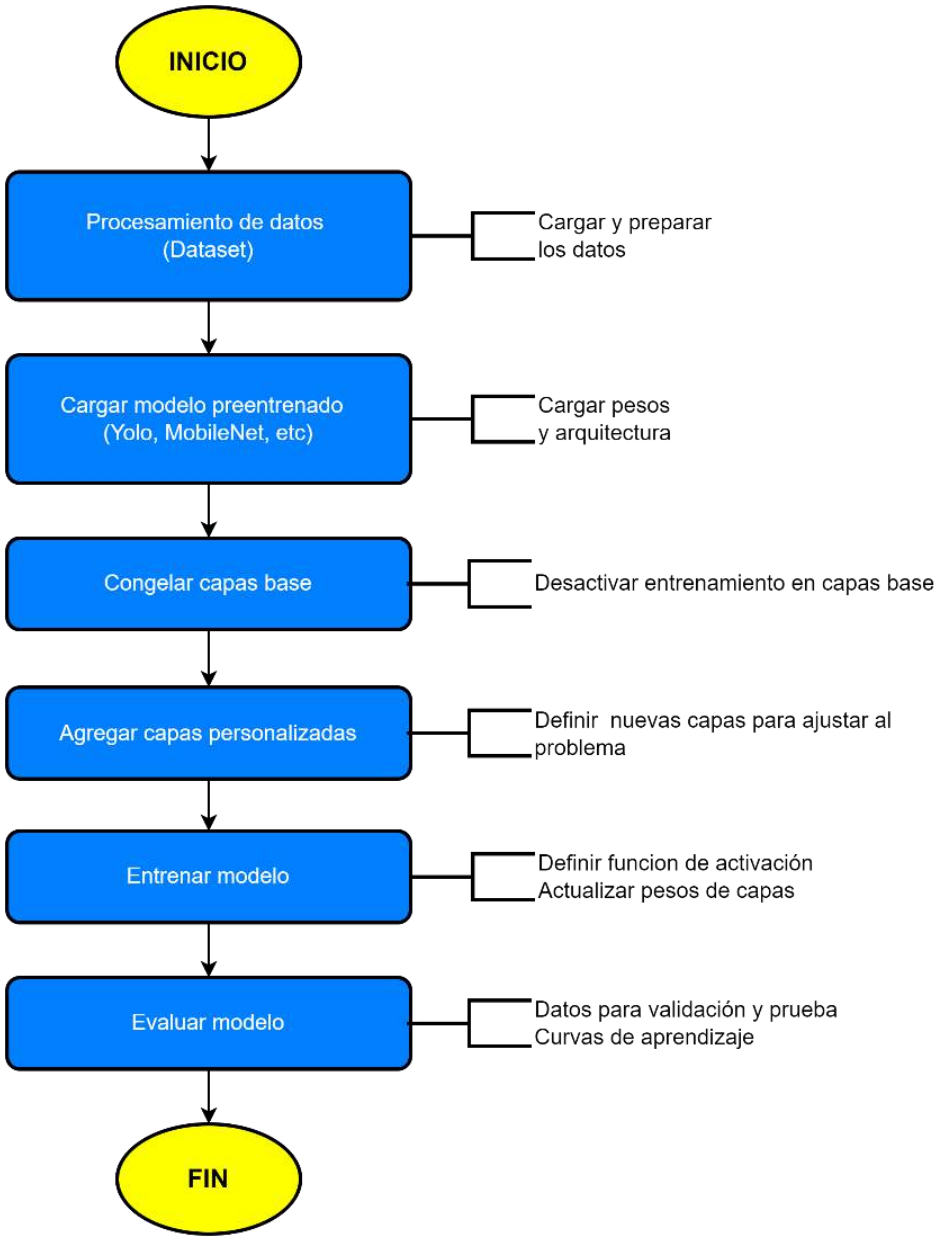


Ilustración 2-8. Diagrama de flujo Técnica de entrenamiento Transfer Learning

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

En la Ilustración 2-8. se representa en forma de diagrama de flujo el algoritmo de la técnica de entrenamiento transferencia por aprendizaje que consta de 8 pasos detallados a continuación.

- **Procesamiento de datos:** Este paso incluye la carga de datos y preparación de los datos de entrenamiento y validación.

- **Cargar modelo pre-entrenado**: Se carga un modelo pre-entrenado de CNN, que generalmente ha sido entrenado en un conjunto de datos grande y diverso.

- **Congelar capas base:** Aquí, se desactiva el entrenamiento en las capas base del modelo pre-entrenado para que los pesos no se actualicen durante el entrenamiento.

- **Agregar capas personalizadas:** Se añaden capas adicionales personalizadas encima de las capas base para adaptar el modelo a tu problema específico.

- **Compilar modelo:** En esta etapa, se define la función de pérdida, el optimizador y las métricas para compilar el modelo antes del entrenamiento

- **Entrenar modelo:** Se alimentan los datos de entrenamiento al modelo y se actualizan los pesos solo en las capas personalizadas.

- **Evaluar modelo**: Aquí se alimentan los datos de validación o prueba al modelo entrenado y se obtienen métricas de rendimiento, como la precisión o la pérdida.

- **Utilizar el modelo**: En la última etapa, se cargan los pesos entrenados, se alimentan nuevos datos para inferencia y se obtienen las predicciones del modelo (MathWorks, 2023).

2.4 Plataformas de desarrollo para Sistemas Embebidos

Las plataformas de desarrollo pueden ser específicas para un lenguaje de programación o tener soporte para múltiples lenguajes. También pueden variar en términos de su enfoque, ya sea para el desarrollo de aplicaciones móviles, aplicaciones web, inteligencia artificial, Internet de las cosas (IoT) u otros dominios específicos.

2.4.1 Lenguajes de Programación

2.4.1.1 Python

Python es un lenguaje de programación de alto nivel, interpretado y de propósito general. Es conocido por su simplicidad, legibilidad y facilidad de uso, lo que lo convierte en una opción popular tanto para principiantes como para programadores experimentados.

Python se destaca por su sintaxis clara y concisa, lo que facilita la escritura y comprensión de código. Además, ofrece una amplia biblioteca estándar que proporciona una amplia gama de funcionalidades para tareas comunes, como manipulación de archivos, networking, procesamiento de texto, matemáticas y más (Zelle, 2010).

2.4.1.2 C++

Según Stroustrup, (2013) es un lenguaje de programación de propósito general que se basa en el lenguaje C, es un lenguaje de alto nivel que permite la programación orientada a objetos, así como la programación genérica y la programación de bajo nivel. C++, se caracteriza por ser un lenguaje eficiente y flexible, que ofrece un alto nivel de abstracción y permite la manipulación directa de recursos del sistema, maneja una sintaxis rica y ofrece características como el uso de clases y objetos, herencias, polimorfismos, plantillas, sobrecargas de operadores, manejo de excepciones, entre otros.

2.4.1.3 VHDL

VHDL (VHSIC Hardware Description Language), es un estándar utilizado para describir y modelas sistemas digitales. Fue desarrollado originalmente por el Departamento de Defensa de los Estados Unidos en la década de 1980. Se utiliza para diseñar, simular y verificar circuitos digitales complejos, ofreciendo una sintaxis estructurada y basada en texto que permite describir las señales, componentes, interconexiones y comportamientos de un sistema digital. Permite definir la funcionalidad de los componentes y su interacción, así como simular y verificar el sistema antes de la implementación física (IEEE 2009).

VHDL (VHSIC Hardware Description Language), es un estándar utilizado para describir y modelas sistemas digitales. Fue desarrollado originalmente por el Departamento de Defensa de los Estados Unidos en la década de 1980. Se utiliza para diseñar, simular y verificar circuitos

digitales complejos, ofreciendo una sintaxis estructurada y basada en texto que permite describir las señales, componentes, interconexiones y comportamientos de un sistema digital. Permite definir la funcionalidad de los componentes y su interacción, así como simular y verificar el sistema antes de la implementación física (IEEE 2009).

2.4.2 Entornos de Desarrollo

2.4.2.1 Vivado

Según Xilinx Inc, (2021), vivado es un entorno de desarrollo integrado (IDE) utilizado para el diseño de sistemas digitales, especialmente para dispositivos lógicos programables (FPGAs) y sistemas en chip (SoCs). Utiliza el lenguaje de programación C/C++, con ensamblaje de sistema gráfico, su diseño de productividad de alto nivel ultrarrápido (UG1197), acelera la creación y verificación de diseños 15 veces más que las metodologías basadas en RTL.

2.4.2.2 Jupyter

Es un software de interfaz web de código abierto en el cual permite incluir texto, video, audio, imágenes y la ejecución de código en varios lenguajes a través de un navegador. Jupyter ha sido capaz de combinar tres de los lenguajes de programación de código abierto más utilizados en el ámbito científico como son: Julia, Python y R, evitando así las limitaciones de usar un solo leguaje (Cabrera y Días 2020).

2.4.2.3 Arduino IDE

Según Banzi, (2011), Arduino es una plataforma de hardware y software de código abierto diseñada para facilitar la creación de proyectos electrónicos interactivos. Se compone de una placa de desarrollo con microcontrolador y un entorno de programación que permite escribir y cargar código en la placa.

2.4.3 Librerías para RNAs e Inteligencia Artificial

2.4.3.1 OpenCV

Según Suarez et al., (2014), OpenCV es una biblioteca de visión por computadora y aprendizaje automático de código abierto. Se enfoca en proporcionar un conjunto de herramientas y algoritmos que permiten a los desarrolladores procesar y analizar imágenes y videos de manera eficiente. OpenCV se puede utilizar en una variedad de lenguajes de programación, como C++, Python y Java, y ofrece funciones para tareas como detección de objetos, seguimiento de objetos, reconocimiento facial, calibración de cámaras, procesamiento de imágenes, entre otros.

2.4.3.2 Tensor Flow

Tensor Flow es una biblioteca de código abierto para el aprendizaje automático y la inteligencia artificial. Fue desarrollada por el equipo de investigación de Google Brain y se ha convertido en una de las bibliotecas más populares y ampliamente utilizadas en el campo del aprendizaje automático(Giancarlo Zaccone, 2018).

CAPITULO III

3. MARCO METODOLÓGICO

En este capítulo se describe las pautas que conforman el Dispositivo Detector de Canes.

3.1 Metodología de Desarrollo

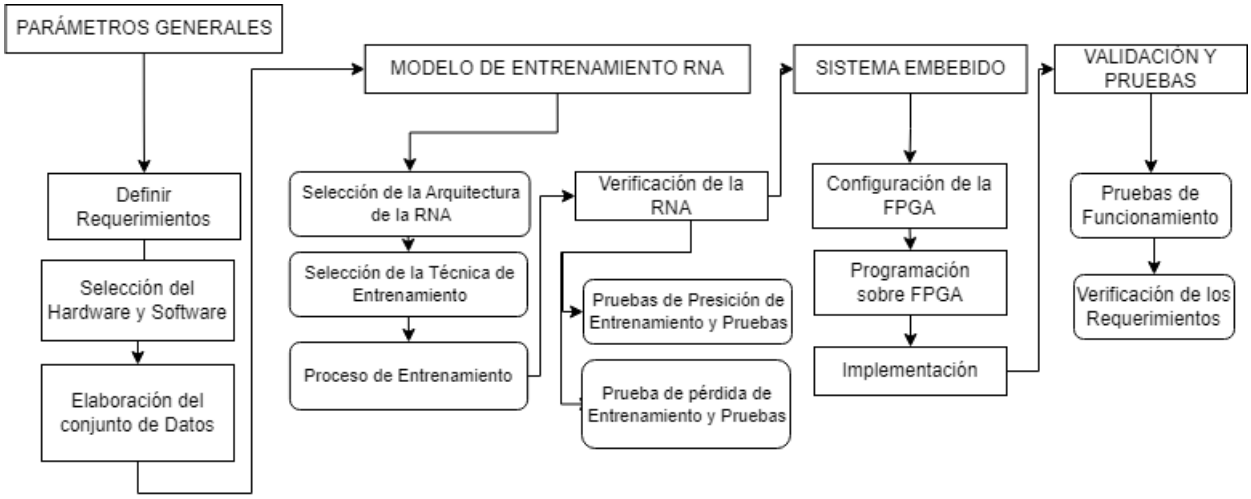


Ilustración 3-1. Arquitectura del Dispositivo Detector de Canes.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023

El presente Proyecto de Integración Curricular tiene como objetivo el desarrollo de un Sistema Embebido basado en FPGA y Visión Artificial implementado en un vehículo para detectar y alertar sobre la presencia de canes en las vías. Para esto, el proceso metodológico, resumido en la Ilustración 3-1.

El proyecto está constituido por cuatro etapas, la **PRIMERA ETAPA** hace referencia a los parámetros generales del sistema embebido, esta es la fase inicial del desarrollo del sistema como tal, donde se establecen los fundamentos y requisitos iniciales para el dispositivo detector de canes. Dentro de esta etapa se encuentran tres procesos, el primero es la definición de parámetros, donde se establece de manera precisa los objetivos del sistema, así como los requerimientos funcionales que debe cumplir. Se detalla el propósito del sistema embebido y las características que se espera que tenga. El segundo proceso se trata de la selección de hardware y software donde se decide que componentes serán utilizados en el sistema. Además, se eligen las herramientas software para desarrollar el sistema. El tercer proceso se encarga de la elaboración del conjunto

de datos, donde se recopila y prepara un conjunto de datos representativos que serán utilizados para entrenar y probar la red neuronal encargada de la detección de los canes.

La **SEGUNGA ETAPA,** llamada Modelo de Entrenamiento de la RNA, se relaciona con el desarrollo y entrenamiento de la red neuronal que será la base del dispositivo detector de canes. Para esto se considera cuatro procesos: el primer proceso que se encarga de la selección de la arquitectura de la RNA, donde se elige la arquitectura más adecuada para la red neuronal que se utilizada. Esto incluye la determinación del número de capas, el tipo de capas y otros parámetros propios de la arquitectura. El segundo proceso es la selección de la técnica de entrenamiento, donde se decide que técnica de entrenamiento se utilizara para ajustar los parámetros de la red neuronal. El tercer proceso lleva el nombre de proceso de entrenamiento y es considerado el más relevante de esta etapa, puesto que es donde se lleva a cabo el entrenamiento de la red neuronal utilizando el conjunto de datos preparado en la primera etapa y considerando los parámetros seleccionados en el primer y segundo proceso de esta etapa. El cuarto proceso llamado verificación de la RNA es donde se evalúa la eficacia de la red neuronal entrenada mediante pruebas de precisión y perdida de entrenamiento.

La **TERCERA ETAPA,** conocida como SISTEMA EMBEBIDO, **s**e enfoca en el acondicionamiento del hardware y software seleccionado. Esta etapa se conforma de tres procesos primordiales, los cuales son: el primer proceso conocido como Configuración de la FPGA, es donde se establecen las configuraciones adecuadas en la FPGA para garantizar la operación correcta del sistema y la interconexión adecuada con los componentes. El segundo proceso lleva el nombre de Programación sobre la FPGA por ser el proceso encargado de programar la tarjeta de desarrollo con el código necesario para el funcionamiento del sistema. Esto incluye la integración de la red neuronal entrenada en el hardware y la interacción con otros componentes. El proceso tres conocido como Implementación contiene la implementación física del sistema, que puede incluir la integración de sensores, actuadores, tarjetas de desarrollo y estructuras que contengan estos componentes.

En la **ETAPA CUATRO,** llamada Verificación y Pruebas, se realizarán las pruebas de validación necesarias para evaluar el funcionamiento del dispositivo detector de canes y asegurar que el sistema cumple con los requerimientos establecidos. Existen dos procesos: el primer proceso llamado Pruebas de Funcionamiento, hace referencia a la respuesta del dispositivo en ambientes controlados y no controlados tantos estáticos como dinámicos. Mientras que el segundo proceso, conocido como Verificación de los requerimientos, es donde se evalúa si el sistema cumple con todos los requisitos especificados en la Etapa 1 en el proceso llamado Definición de parámetros.

3.2 Parámetros Generales

3.2.1 Requerimientos

En base a la revisión bibliográfica descrita en el capítulo anterior se puede detallar los requerimientos de diseñador para poder cumplir con el objetivo general. Los requerimientos son:

- Dimensionar correctamente la estructura del dispositivo de tal manera que su presencia no obstaculice la visión del conductor, ni afecte el correcto funcionamiento del vehículo.
- El sistema detector de canes se implementará utilizando la tarjeta de desarrollo FPGA PYNQ Z1
- La comunicación módulo de adquisición de datos- tarjeta de desarrollo debe ser mediante conexión USB.
- La cámara debe ser colocada en el exterior del vehículo.
- El dispositivo detector de canes DDC será alimentado con una fuente externa de 6V, 4.5Ah.
- Considerar dos modos de operación; modo normal y modo detector.
- El modo encendido estará encargado de avisar si el dispositivo ha sido alimentado correctamente, esto lo realiza mediante la activación de un actuador (led).
- El modo normal dará aviso por medio de un actuador (led) que existe tráfico de datos entre el módulo de adquisición de datos y la tarjeta de desarrollo.
- Cuando este modo detector se dará aviso mediante un indicador (led), y cuando se haya detectado canes, los actuadores, buzzer y vibrador se activan brindando alerta al conductor.
- El dispositivo deberá ser capaz de detectar canes en diferentes condiciones de luz ambiente.
- El dispositivo DDC deberá ser capaz de detectar canes en un rango de distancia que abarque desde 1 hasta 10 metros, con un tiempo de respuesta seguro y eficiente.
- El dispositivo DDC debe ser capaz de diferenciar canes y personas
- El dispositivo DDC debe ser capaz de detectar a uno o más canes.

3.2.2 Herramientas Hardware y Software para el desarrollo

A continuación, se describe los elementos utilizados para la implementación del sistema embebido para la detección de canes en las vías, una descripción y especificaciones técnicas del funcionamiento. Así como las consideraciones más relevantes de cada elemento.

3.2.2.1 Herramientas Hardware

- **Tarjeta de Desarrollo FPGA PYNQ - Z1**

Es una placa de desarrollo basada en la plataforma Zynq-7000 de Xilinx. La Zynq-7000 es una familia de SoC (System-on-Chip) que combina un procesador ARM Cortex-A9 de doble núcleo con una matriz FPGA programable. La tarjeta de desarrollo PYNQ Z1 está diseñada para facilitar el desarrollo de sistemas embebidos y acelerados por hardware utilizando la combinación de un procesador ARM y una FPGA.

La FPGA PYNQ Z1 cuenta con una amplia variedad de periféricos y conectores, incluyendo puertos Ethernet, USB, HDMI, Pmod y GPIO, que permiten la conexión con otros dispositivos y sensores. Además, la placa es compatible con el entorno de desarrollo PynQ, que proporciona una interfaz de programación Python para acceder y controlar tanto el procesador ARM como la lógica programable en la FPGA (Xilinx, 2020).

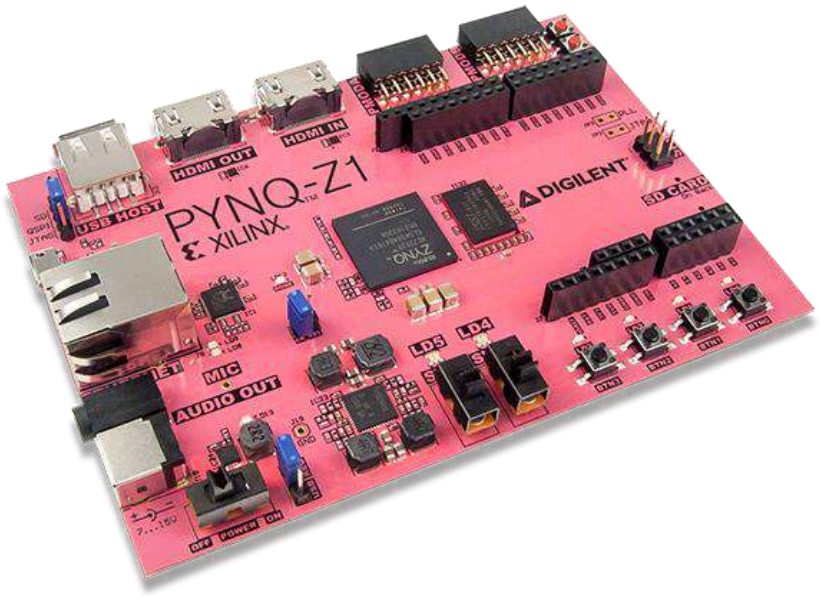


Ilustración 3-2. FPGA PYNQ Z1 de Xilinx.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

Mediante la Tabla 3-1, se observa las principales características técnicas sobre la tarjeta de Desarrollo FPGA PynQ Z1.

Tabla 3-1. Características técnicas de la tarjeta FPGA PynQ Z1 de Xilinx.

Características Técnicas	Detalles
Familia de FPGA	Xilinx Zynq-7000
Procesador	ARM Cortex-A9 de doble núcleo
Memoria RAM	512 MB DDR3
Memoria Flash	16 MB QSPI Flash
FPGA Lógica	28,000 celdas lógicas
Pines GPIO	40 pines GPIO de propósito general
Interfaz de red	Puerto Ethernet 10/100 Mbps
Puertos USB	2 puertos USB 2.0
Conectividad de video	Puerto HDMI
Conectores de expansión	2 conectores Pmod, 1 conector XADC
Conectividad de depuración	Puerto UART y JTAG
Voltaje de alimentación	5V DC

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

- **Dispositivo de captura de imagen Cámara Web 720px**

Como se muestra en la *Ilustración 3-3*, La cámara utiliza una interfaz USB para conectarse a diferentes dispositivos, como computadoras o sistemas embebidos. Está diseñada para ofrecer una calidad de imagen excepcional y es ampliamente utilizada en aplicaciones por visión por computadora, consta de enfoque automático y alta sensibilidad (León Orozco 2023).

Ilustración 3-3. Cámara Web HD 720px Webcam

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

En la Tabla 3-2, se describe las características técnicas de funcionamiento de la cámara web 720px Webcam.

Tabla 3-2. Principales características técnicas y especificaciones de la cámara web 720px

Características Técnicas	Detalles
Conectividad	USB 2.0
Color	Negro
Velocidad de fotogramas	30 FPS
Tipo de videocámara	Webcam
Característica especial	Micrófono de reducción de ruido
Resolución	720 p 1280x720
Lente	Recubrimiento de vidro de 6 capas
Compatibilidad	Windows, Linux, Mac- OS X 10.4.8 o posterior.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

- **Otros Componentes Hardware**

Mediante la Tabla 3-3., se presenta una descripción breve de los componentes hardware utilizados en el proceso de creación y desarrollo del dispositivo detector de canes, cuyos componentes principalmente se encargan de generar los avisos de alerta según el modo de operación en el que se encuentre el DDC.

Tabla 3-3. Componentes hardware para el desarrollo del Dispositivo Detector de Canes.

Componentes Hardware	Descripción	Voltaje de Operación	Consumo de corriente	Temperatura de operación
Pulsador	Dispositivo utilizado para abrir o cerrar un circuito eléctrico (Pandey 2022).	5-12V	20mA	
Motor Vibrador	Genera vibraciones o pulsaciones en respuesta a una señal eléctrica (Novatronic 2020).	5V	100mA	-20°C a 70°C
Buzzer	Dispositivo electromecánico que produce sonido o zumbidos audibles (Mecafenix, 2018).	5V	30mA	-20°C a 70°C

Componentes Hardware	Descripción		Voltaje de Operación	Consumo de corriente	Temperatura de operación
LED'S	Verde	Diodo emisor de luz, sus funciones son de señalización, y mejoras estéticas (Déleg, 2017)	1.6 - 3.7 V	5mA	-40°C a 70°C
	Azul		2.4 – 3.7 V	20mA	-40°C a 70°C

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

3.2.2.2 Herramientas Software

El entrenamiento de una red neuronal puede ser computacionalmente intensivo, especialmente si se trabaja con conjuntos de datos grandes y redes neuronales profundas. Por medio de la *Tabla 3-4*, se mencionan algunos recursos que son importantes para llevar a cabo el entrenamiento de manera eficiente.

Tabla 3-4. Descripción de software, librerías y lenguajes de programación utilizados en el S.E.

		Descripción
SOFTWARE	Jupyter	Entorno desarrollador del algoritmo del S.E.
	Google Collab	Entorno desarrollador de la RNA.
Librerías	Open CV	Procesamiento de imágenes
	Tensor Flow	Arquitectura de la RNA
Lenguajes de Programación	Python	Lenguaje Base
	Arduino	Activación de salidas de la FPGA

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

3.2.3 Elaboración del Conjunto de Datos

La Elaboración del Conjunto de Datos, es una fase relevante en la preparación de cualquier sistema o modelo que involucre la utilización de datos. Este proceso engloba dos tareas fundamentales: la recolección de datos y el acondicionamiento de los datos.

3.2.3.1 Recolección de Datos

La recolección de datos es una etapa fundamental en el entrenamiento de una red neuronal artificial, implica obtener muestras de datos representativas del problema que se desea resolver y organizarlos de manera adecuada.

Durante la recolección de datos, es importan te considerar los siguientes aspectos:

Definición del problema: se debe tener claro cuál es el problema que se desea abordar y que tipo de datos son relevantes para resolverlo. Esto permite enfocar la recolección de datos de manera efectiva.

Selección de fuentes de datos: Se identifican las fuentes de datos disponibles, que pueden ser bases de datos existentes, conjuntos de datos públicos, sensores, registros, encuestas, entre otros.

Captura de datos: se realiza la adquisición de los datos desde las fuentes seleccionadas. Esto puede implicar la extracción de datos de bases de datos, el registro de mediciones o la recopilación de información a través de encuestas u otras formas de interacción con los usuarios.

La *Ilustración 3-4.*, describe de manera grafica los aspectos mencionados anteriormente relacionados con los parámetros de recolección de datos necesarios en este proyecto.

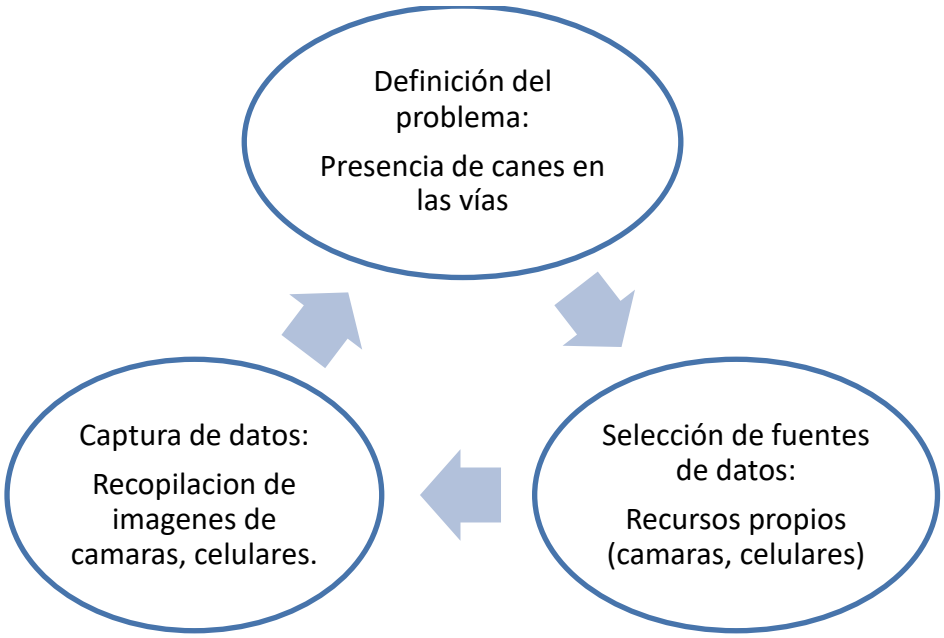


Ilustración 3-4. Adaptación de los atributos para la recolección de datos en el proyecto.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

En la Tabla 3-5., se puede evidenciar los siete parámetros primordiales que se consideraron para la adquisición de datos para la detección de la presencia de canes en las vías acompañado de la cantidad necesaria en cada uno.

Tabla 3-5. Atributos considerados en la adquisición de datos para el entrenamiento de la RNA.

Parámetro	**Cantidad**
Numero de Datos (imágenes)	2000
Días	60
Horas por día	4
Jornadas por día	Dia / Noche
Lugares de muestreo	100
Condiciones ambientales	75% despejado – 25% lluvioso
Especificación general de los canes	Raza mestiza

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

- Número de datos: es necesario manejar un número elevado de datos (imágenes) que en este caso fueron 2000 imágenes. Haciendo una asignación de 1500 imágenes específicamente para el entrenamiento de la red neuronal artificial mientras que el restante (500 imágenes) son utilizadas al momento de realizar las pruebas de la red neuronal artificial como tal.

- Días/Horas por Día: se trabajó en un periodo de nueves semanas, de la semana1 a la semana 8 se recolecta datos todos los siete días de la semana en un lapso de 4 horas por día, mientras que en la semana 9 se recolecta la información tan solo los lunes, martes, viernes y sábado en el mismo lapso de horas como se muestra en la Tabla 3-6.

Tabla 3-6. Cronograma de los días y hora por día requeridos para la adquisición de datos.

	Lunes	**Martes**	**Miércoles**	**Jueves**	**Viernes**	**Sábado**	**Domingo**
Semana1	X	X	X	X	X	X	X
Semana2	X	-	X	X	X	X	X
Semana3	X	X	X	X	X	X	X
Semana4	X	X	X	X	X	X	X
Semana5	X	X	X	X	X	X	X
Semana6	X	X	X	X	X	X	X
Semana7	X	X	X	X	X	X	X
Semana8	X	X	X	X	X	X	X
Semana9	X	X	-	-	X	X	-

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

Nota: X representa las 4 horas por días utilizadas para la recolección de datos.

- Jornadas por día: mientras más condiciones externas sean consideradas al momento de la recolección de datos se puede obtener una información diversa para que el entrenamiento pueda contener el mayor número de posibilidades que se puedan presentar. Siendo así se vio necesario obtener información en dos jornadas diarias (DIA y NOCHE) distribuyendo dos horas para cada una. En la Tabla 3-7, se detalla de manera precisa las jornadas consideradas y el número de horas que se ocupó en las mismas, tomando en cuenta el total de días en los que se realizó la recolección.

Tabla 3-7. Descripción detallada de las jornadas requeridas para la recolección de datos.

	N. de horas	Días
Jornada 1: DIA	2 horas	60
Jornada 2: NOCHE	2 horas	60

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

- Lugares de muestreo: mediante la Tabla 3-8., se describe cuáles fueron los lugares considerados y el número de zonas requeridos en cada una de ellas.

Tabla 3-8. Lugares territoriales y tipo de vías que fueron consideradas para la recolección de datos.

Lugar	Zonas	Descripción
Riobamba	3	Circunvalaciones Avenidas Zona urbana
Guano	2	Zona Urbana Carretera Principal
Ambato	2	Zona Urbana Carretera Principal Avenidas
Quito	4	Zona Urbana Autopistas Avenidas Carreteras Principales
Pallatanga	2	Zona Urbana Carretera Principal
Macas	2	Zona Urbana Carretera Principal
Baños	2	Zona Urbana Carretera Principal
Lago Agrío	2	Zona Urbana Carretera Principal

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

3.2.3.2 Acondicionamiento de los Datos

El acondicionamiento de datos también conocido como preprocesamiento de datos, es una etapa fundamental en el entrenamiento de una red neuronal artificial. Consiste en preparar y transformar los datos de entrada de manera adecuada antes de ser utilizados para entrenar la red neuronal, el objetivo principal del acondicionamiento de datos es mejorar la calidad de los datos y asegurar que estén en un formato compatible con la red neuronal. Existen algunas tareas a realizar al momento de acondicionar los datos antes de proceder al entrenamiento, esto depende de la técnica de entrenamiento elegida. En el caso del entrenamiento para la detección de canes en las vías, se he seleccionado la técnica de Transfer Learning o transferencia de aprendizaje, la misma que se describirá más adelante.

Para esta técnica es necesario de manera primordial la limpieza de datos, donde se identifica y trata los datos incorrectos, ruidosos o faltantes; esto puede incluir eliminar registros duplicados corregir errores de formato, manejar valores faltantes o anomalías.

3.3 Modelo de Entrenamiento RNA desarrollado

Esta etapa es esencial en la construcción y perfeccionamiento de sistemas basados en aprendizaje automático, debido a la integración de cuatro procesos importantes que determinan la eficacia y capacidad de generación de la red neuronal artificial.

3.3.1 Selección de la Arquitectura de la RNA

Al momento de seleccionar la arquitectura de la red neuronal artificial se debe analizar los requerimientos tanto de hardware como de software que debe cumplir el sistema embebido utilizado en la detección de canes en las vías. Se analizo de manera comparativa cinco parámetros presentes entre las clases de arquitecturas nombradas en el marco teórico, los cuales se ven reflejados en la *Ilustración 3-5*.

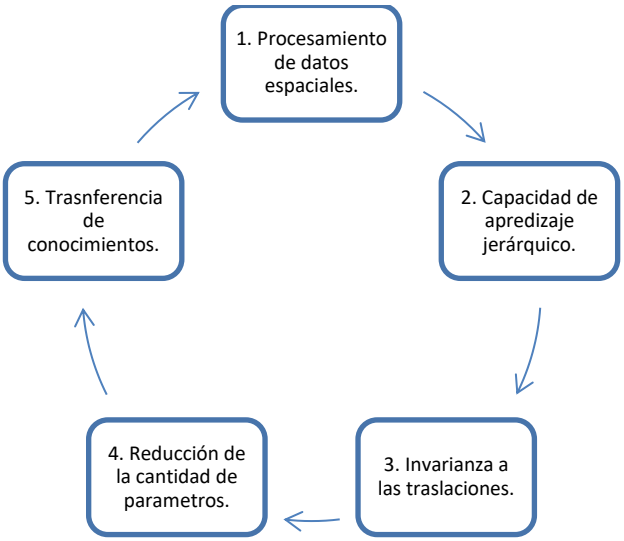


Ilustración 3-5. Parámetros representativos dentro de la Arquitectura de una RNA

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

Dando como resultado que las Redes Neuronales Convolucionales son las más optimas dentro de esta aplicación cumpliendo con:

- Capacidad para procesar datos con una estructura espacial, como imágenes, aprovechando la propiedad de la convolución para capturar patrones locales en los datos, bordes, texturas y características distintas. Debido a que las CNN compuestas por capas convolucionales y de agrupación (pooling) que le permiten el aprendizaje jerárquico de los datos, estas capas convolucionales extraen características más abstractas y de mayor nivel a medida que se profundiza en la red. La invarianza a las traslaciones es otro parámetro que se acopla perfectamente con estas redes debido a que son inherentemente invariantes a las traslaciones en los datos de entrada, esto significa que la posición exacta de un objeto en una imagen no afecta la capacidad de la red para reconocerlo.

- Utilización de operaciones de convolución y agrupación que permite reducir la cantidad de parámetros requeridos en comparación con las redes neuronales totalmente conectadas. Esto no solo hace que las CNN sean más eficientes en cuanto a la memoria y los recursos computacionales necesarios, sino también ayuda a evitar el sobreajuste al tener menos parámetros que aprender.
- Disponibilidad natural a la transferencia de conocimiento que manejan estas redes, demostrando ser eficaces en la extracción de características generales de las imágenes durante la fase de entrenamiento. Con esto, dichas características pueden transferirse y reutilizarse en diferentes tareas de reconocimiento de imágenes y clasificación.

3.3.2 Selección de la Técnica de Entrenamiento

Dentro del Marco teórico, la *Tabla 2-13*, describe las técnicas más comunes utilizadas en el entrenamiento de las redes neuronales artificiales. Entre ellas se encuentra la Técnica de Transfer Learning que ha sido seleccionada como la técnica más idónea dentro de este sistema embebido. La *Ilustración 3-6,* menciona las ventajas que se consideraron al momento de tomar la decisión.

Ilustración 3-6. Ventajas consideradas para la selección de la Técnica de Entrenamiento

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

La escasez de datos hace mención del manejo de una cantidad mínima de datos para entrenar una red neuronal, permitiendo aprovechar los conocimientos adquiridos de modelos previamente entrenados en conjuntos de datos más grandes. La eficiencia en tiempo y recursos puede ser un obstáculo grande en el caso de poseer recursos hardware limitados, sin embargo, esta técnica permite aprovechar el trabajo ya realizado en la etapa de entrenamiento previo, lo que reduce significativamente el tiempo y los recursos necesarios para obtener un modelo de alta calidad. Los modelos previamente entrenados en tareas relacionadas tienden a tener una capacidad de generalización mejorada, esto significa que el modelo puede capturar características generales y patrones útiles en los datos que se aplican en el sistema de detección de canes en las vías.

Al entrenar una red neuronal desde cero en conjuntos de datos pequeños, existe el riesgo de sobreajuste, donde el modelo memoriza los datos de entrenamiento en lugar de aprender patrones generales. Al aplicar Transfer Learning, se utiliza la información aprendida en conjuntos de datos más grandes, lo que ayuda a prevenir el sobreajuste y mejora la capacidad del modelo para generalizar correctamente en nuevos datos. La transferencia de conocimiento de modelos

previamente entrenados permite adaptarse más fácilmente a nuevos dominios de datos. Por ejemplo, si se entrena un modelo en imágenes de naturaleza y se desea aplicar a imágenes de automóviles, el Transfer Learning permite aprovechar los conocimientos sobre la detección de características básicas como bordes y texturas, y adaptarlos específicamente

- **Acondicionamiento de la Arquitectura con la Técnica de Entrenamiento**

Esta etapa es considerada crucial dentro del entrenamiento de las redes neuronales artificiales, debido a que permite acoplar la arquitectura seleccionada para el entrenamiento siendo en este caso una Red Neuronal Convolucional con la técnica elegida (Transfer Learning) para entrenar dicha red. Una correcta elección de los dos parámetros descritos anteriormente puede tener gran impacto en el rendimiento de la red dentro de la aplicación seleccionada al momento de la implementación.

En este apartado se gestiona las clases a entrenar por el modelo y la función de transferencia que va a comandar el mismo. Considerando que el objetivo es la detección de la presencia de canes en las vías el número de clases a entrenar es uno, sin embargo, sabiendo que la técnica seleccionada nos permite trabajar con un algoritmo entrenado por grandes desarrolladores y que cuenta con un aproximado de 92 clases de objetos para la detección, se opta por considerar dos números de clases para reducir al máximo el número de errores en el entrenamiento.

Mientras que en el caso de la función de transferencia se elige trabajar con Softmax, sabiendo que su principal objetivo es asignar probabilidades representativas a cada clase en función de las salidas de la red neuronal. Para esta elección también fue de gran influencia el tipo de técnica que se escogió con anterioridad debido a que, en el contexto del entrenamiento de una red neuronal, la función de transferencia Softmax se utiliza en la capa de salida junto con la función de pérdida de entropía cruzada (cross-entropy loss). La función de pérdida compara las probabilidades predichas por la red con las etiquetas verdaderas y busca minimizar la diferencia entre ellas durante el proceso de aprendizaje.

3.3.3 Proceso de Entrenamiento

El proceso de entrenamiento es el procedimiento donde el modelo de la RNA “aprende” a través de iteraciones y ajustes basados en datos de entrenamiento. Durante este proceso, el objetivo es que la RNA adquiera la capacidad de generalizar y hacer predicciones precisas en nuevos datos, mejorando su capacidad de reconocimiento de patrones. La Tabla 3-9, hace mención del proceso de entrenamiento que se desarrolla dentro del entorno Google Colab.

Tabla 3-9. Descripción detallada del proceso que sigue Google Colab para el entrenamiento.

Paso	Descripción
Preprocesamiento de datos	Se realiza el preprocesamiento necesario en los datos, como normalización, redimensionamiento, etc.
Cargar un modelo pre-entrenado	Importamos un modelo pre-entrenado, en este caso es Mobilenet v2, que ha sido previamente entrenado en un conjunto de datos grande y diverso.
Congelar capas del modelo pre-entrenado	Congelar algunas capas del modelo pre-entrenado para evitar que se actualicen durante el entrenamiento.
Agregar capas personalizadas	Agregar dos nuevas capas personalizadas al modelo, como capas densas adicionales, para adaptarlo a la tarea de detectar la presencia de canes en las vías.
Compilar el modelo	Configurar el modelo para el entrenamiento, especificando la función de activación (Softmax).
Entrenar el modelo	Utilizar el conjunto de datos de entrenamiento para entrenar el modelo, especificando el número de épocas (30).
Evaluar el modelo	Utilizar el conjunto de datos de prueba para evaluar el rendimiento del modelo, calcular métricas como precisión, pérdida, etc.
Ajustar el modelo (opcional)	Realizar ajustes en la arquitectura o hiperparámetros del modelo según sea necesario y volver a entrenarlo.
Guardar el modelo entrenado	Guardar el modelo entrenado en formato. pbtx para posteriormente cargar en la tarjeta de desarrollo Pynq Z1.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

3.3.4 Verificación de la RNA

La verificación de una red neuronal entrenada es un proceso crítico para evaluar su desempeño y garantizar su funcionamiento correcto. Consiste en realizar pruebas y análisis exhaustivos para verificar que la red neuronal produce resultados precisos y confiables en una variedad de situaciones y datos de entrada.

3.3.4.1 Pruebas de Entrenamiento

- **Prueba de precisión de entrenamiento y pruebas**

La precisión de entrenamiento y pruebas es una medida utilizada para evaluar el desempeño de una red neuronal durante el proceso de entrenamiento y validación. Estas métricas indican la capacidad de la red para generalizar y hacer predicciones precisas sobre datos no vistos previamente.

La precisión de entrenamiento se refiere a la capacidad de la red para clasificar correctamente los datos de entrenamiento, es decir, la proporción de ejemplos de entrenamiento que se clasifican correctamente. La precisión de pruebas, también conocida como precisión de validación o precisión de evaluación, se refiere a la capacidad de la red para clasificar correctamente los datos de prueba. Estos datos de prueba se utilizan para evaluar el rendimiento del modelo en situaciones del mundo real y para verificar si la red ha aprendido patrones y características generalizables.

- **Prueba de pérdida de entrenamiento y pruebas**

La pérdida de entrenamiento y pruebas, también conocida como función de pérdida o función de costo, es una medida utilizada para evaluar cuán bien se está desempeñando una red neuronal durante el entrenamiento y la evaluación.

La pérdida de entrenamiento se refiere a la cantidad de error o discrepancia entre las salidas predichas por la red neuronal y las salidas reales correspondientes en el conjunto de datos de entrenamiento. Es una medida de qué tan bien está aprendiendo la red a partir de los datos de entrenamiento y ajustando sus pesos y sesgos para minimizar la discrepancia entre las predicciones y las salidas reales. La pérdida de pruebas, también conocida como pérdida de validación o pérdida de evaluación, se refiere a la cantidad de error en las predicciones de la red neuronal cuando se aplica a datos de prueba o datos no vistos previamente. Es una medida de qué tan bien la red está generalizando y haciendo predicciones precisas en nuevos datos.

3.4 Sistema embebido

A continuación, se presenta de forma detallada como se realiza el proceso de configuración de la tarjeta de desarrollo PYNQ Z1, así como también el proceso de programación sobre la FPGA y la implementación de tanto la parte software como hardware para obtener como resultado el sistema embebido con la capacidad de realizar la detección de canes.

3.4.1 Configuración de la Tarjeta de Desarrollo

En este apartado, se detalla paso a paso el proceso de configuración de la tarjeta de desarrollo PYNQ Z1, desde la instalación del software necesario, la configuración del entorno de desarrollo, la conexión de la tarjeta a un ordenador, la instalación de controladores y bibliotecas en la FPGA.

Para la configuración de la FPGA se necesita de algunos requisitos ente los que tenemos:

- Computadora con navegador (Chrome, Safari, Mozilla, Opera, etc.)
- Cable de Red Ethernet
- Cable micro USB
- Tarjeta Micro-SD de 8GB con imagen precargada

A continuación, se enumeran los pasos a seguir para la configuración de la tarjeta de desarrollo FPGA Pynq Z1.

Instalación del software: Es el primer paso crucial para configurar adecuadamente la tarjeta de desarrollo, por tal motivo se procede a la descarga del software necesario, esta descarga se la realiza directamente desde la página web oficial del fabricante http://www.pynq.io/board.html en donde se encuentra las últimas versiones de software y controladores. Una vez descargada la imagen (software), esta se la guarda en una micro USB para insertar en la ranura de la tarjeta de desarrollo.

1. Para configurar el arranque desde la tarjeta microSD, es necesario ajustar el JP4/ Boot Jumper en posición SD esto se logra colocando el Jumper sobre los dos pines superiores JP4, tal como se muestra en Ilustración 3-7.

2. Para la alimentar la PYNQ-Z1 a través de un cable micro USB, es necesario configurar el JP5/ Power Jumper en la posición USB. Otra alternativa para alimentar la tarjeta, se puede utilizar un regulador de potencia de 12V y configurar el Power Jumper en la posición REG.

3. A continuación, insertar la tarjeta microSD cargada con la imagen PYNQ-Z1 en la ranura correspondiente ubicada debajo de la placa.

4. Conectar el cabe USB desde el Ordenador/Laptop al puerto PROG-UART.

5. Conectar el cable de red Ethernet al puerto Ethernet de la placa.

6. Encender la PYNQ-Z y verificar la secuencia de arranque siguiendo las siguientes instrucciones.

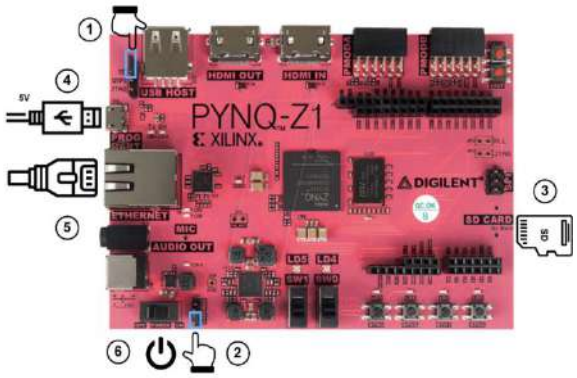


Ilustración 3-7. Configuración de la FPGA

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

Encendido de la tarjeta de desarrollo PYNQ-Z1: Como se menciona en el paso 6 de la configuración de la tarjeta de desarrollo, se mueve el interruptor de POWER a la posición ON para encender la placa. De manera inmediata se iluminará un LED color Rojo LD13 para confirmar que la placa tiene energía, después de unos segundos, el LED Amarillo/Verde LD12 se iluminara para mostrar que el dispositivo Pynq está funcionando de manera correcta. Después de aproximadamente un minuto, se debe notar que dos LEDs Azules LD4 y LD5 parpadean al mismo tiempo, junto con los LEDs Amarillo/Verde LD0-LD3. Posteriormente los LEDs Azules seguirán pardeando mientras que los LEDs Amarillo/Verde permanecen encendidos. Esto indica que el sistema operativo se inició correctamente y está listo para usarse.

Conexión de red: Después de haber configurado correctamente la placa es necesario establecer la conexión para comenzar a utilizar el entorno de programación Jupyter. Por tal motivo es necesario conectarse desde un enrutador de red con un servicio DHCP para que la tarjeta de desarrollo adquiera automáticamente una dirección IP.

Conexión al entorno de programación Jupyter: Una vez que la tarjeta de desarrollo Pynq Z1 haya adquirido su dirección IP, es momento de establecer la conexión para acceder al entorno de programación Jupyter. Para hacer esto, es necesario descargar una aplicación móvil llamada Fing, mediante esta aplicación que nos ayudara a obtener la dirección IP que se le otorgo a la PYNQ. cómo se lo muestra en la Ilustración 3-8.

Ilustración 3-8. App Fing para la obtención de dirección IP
Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

Una vez que identificamos la dirección IP de la PYNQ nos dirigimos a un navegador web y en la barra de búsqueda se va a digitar lo siguiente http://*dirección IP de placa*:9090. Si la tarjeta esta correctamente configurada, y una vez de iniciar sesión con las siguientes credenciales:

- Usuario: xilinx
- Contraseña: xilinx

3.4.2 Diagrama de bloques de la programación DDC sobre la PYNQ Z1.

En este apartado, describiremos el algoritmo adjuntado en el **ANEXO A**, que nos permitirá adecuar la red neuronal y el hardware en el sistema embebido. Este proceso se realiza mediante cinco bloques tal y como se muestra en la Ilustración 3-9, para posteriormente detallar paso a paso cada uno de los bloques.

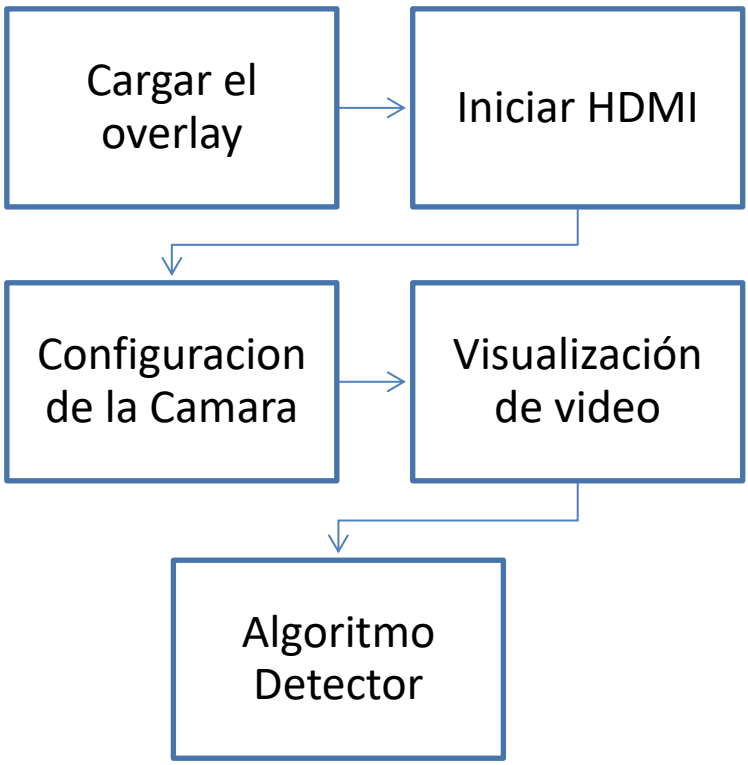


Ilustración 3-9. Diagrama de bloques del algoritmo DDC

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

- **Bloque 1: Cargar Overlay**

En este bloque implica habilitar y configurar componentes que nos permite acceder a funcionalidades adicionales, librerías de terceros, así como también frameworks que serán utilizadas durante el desarrollo del software DDC. Mismas que son descritas en la Tabla 3-10.

Tabla 3-10. Bibliotecas necesarias para acceder a las funcionalidades, periféricos y procesar video de la tarjeta de desarrollo.

Bibliotecas	Descripción
BaseOverlay	Bibliotecas PYNQ (Python for Zynq) proporciona una interfaz para acceder a los recursos y periféricos para configurar los componentes adicionales de la PYNQ Z1
from pynq.lib.video import *	Esta biblioteca brinde soporte para el procesamiento de video en plataformas PYNQ. Proporciona funciones para captura, mostrar y manipular las señales de video utilizados en los periféricos de la tarjeta de desarrollo.
from pynq.buffer import PynqBuffer	Representa un buffer de memoria en la plataforma PYNQ, para administrar la memoria y los datos que se intercambian entre el hardware y software.
base = BaseOverlay("base.bit")	Se utiliza para cargar el Overlay en la tarjeta PYNQ. El argumento "base.bit" especifica el archivo bitstream que contiene la descripción de hardware y lógica programable necesaria para el funcionamiento.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

- **Bloque 2: Iniciar HDMI salida**

En la Ilustración 3-10. se explica mediante un diagrama de flujo el bloque 2 que consiste en configurar y habilitar la salida de video de HDMI de la tarjeta de desarrollo PYNQ con una resolución de 640x480 a 60Hz. El objeto "VideoMode", se utiliza para definir la configuración de video deseada para posteriormente iniciar la salida HDMI y se imprime un mensaje para indicar que la salida HDMI ha sido habilitada correctamente.

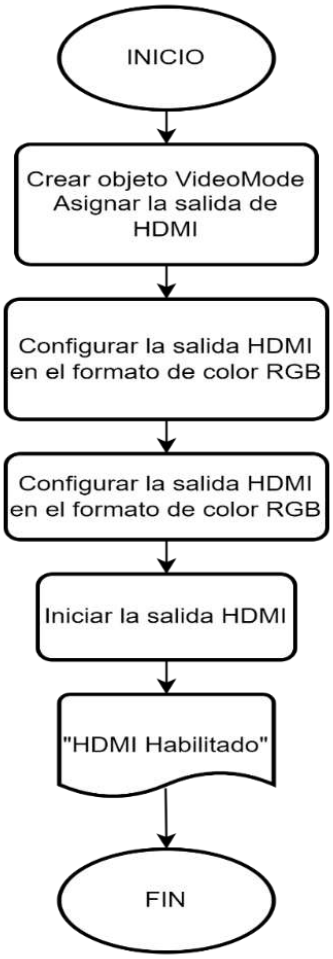


Ilustración 3-10. Diagrama de Flujo para la activación de la salida de video HDMI

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

- **Bloque 3: Configuración de la cámara**

La Ilustración 3-11., representa un diagrama de flujo el cual describe el proceso para la configuración de la cámara para el DDC el cual nos permitirá la captura de la imagen en tiempo real para la futura detección de canes.

- Se definen las siguientes bibliotecas:

  os. - Proporciona funciones relacionadas con el sistema operativo, como configuración de variables de entorno (Bahit Eugenia, 2013).

  cv2.- Importa la biblioteca de OpenCV, que proporciona funciones y herramientas para el procesamiento de imágenes y videos (Zyprian Florian, 2023).
- Se configura como variable de entorno para que OpenCV no muestre mensajes de registro, esto evita que se impriman mensajes innecesarios durante la ejecución
- Definimos la variable **frame_in_w** con el calor de ancho deseado para el marco de la imagen capturada por la cámara.

- Definimos la variable **frame_in_h** con el calor de la altura deseada para el marco de la imagen capturada por la cámara.
- Crear un objeto **videoIn=cv.VideoCapture(0)** que mediante OpenCV se utiliza para la captura imágenes de la cámara
- El comando **videoIn.set** establece el ancho y la altura del marco de la imagen capturada y establecer el tamaño del búfer del objeto **videoIn** en 2 fotogramas antes de sobrescribir los anteriores.
- Imprimir si la cámara está habilitada o no, así como también información de la dirección de la cámara y el mensaje de que la cámara está configurada.

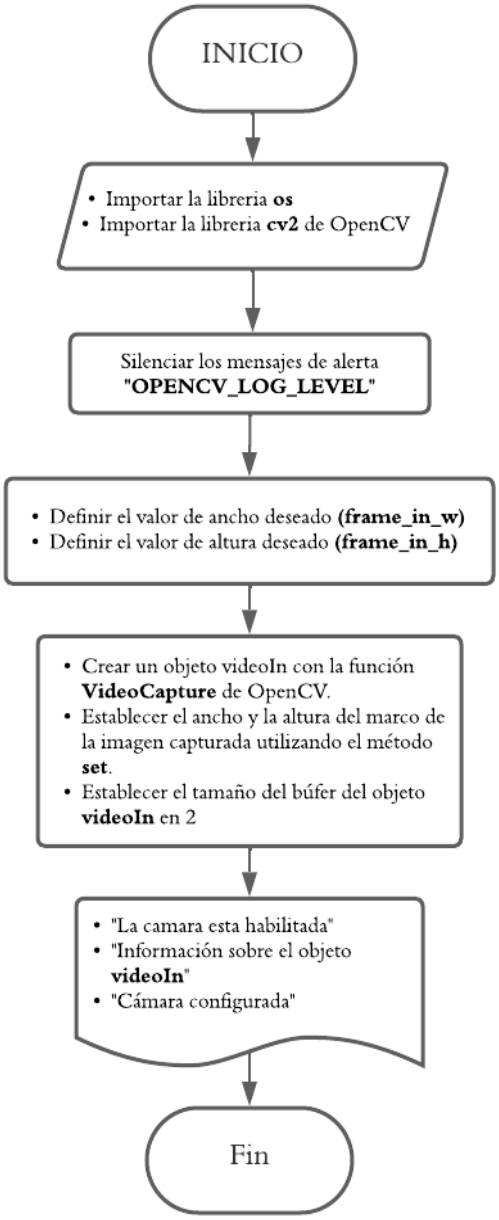


Ilustración 3-11. Diagrama de flujo de la configuración de la cámara

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

- **Bloque 4: Visualización de video**

La *Ilustración 3-12.*, representa un diagrama de flujo y explica de manera más detallada el algoritmo que realiza la captura de video desde una cámara y muestra los Frames en una ventana utilizando OpenCV y PYNQ. A continuación, se explica que hace cada parte del diagrama de flujo para una mejor comprensión.

Inicio

- Importar las siguientes bibliotecas y/o librerías:
  numpy: Esta librería es aquella que proporciona funcionalidades para realizar operaciones numéricas en matrices y arreglos.
  time: Modulo encargado de proporcionar funciones relacionadas con el tiempo, como medir la duración de ejecución.
- Definir una variable contadora la misma que se va a inicializar en cero. "**contador=0**", y que se utilizará para llevar el registro del número de frames procesados.
- La variable "**num_frames=100**" define el número total de frames que se capturan y serán procesados.
- Se inicializa una variable **readError** en cero para contar los errores de lectura de la cámara
- En la variable "**org**" definir las coordenadas (x, y) de la esquina superior izquierda donde se colocará el texto en cada frame.
- Registramos el tiempo de inicio antes de capturar los frames con el comando "**start=time.time()**". Y elegimos la fuente, escala y el grosor del texto que se va se utiliza para mostrar el contador de frames.

Bucle de repetición

- Comienza el ciclo de repetición para procesar los frames y se pregunta si la lectura fue exitosa
- Si se verifica que la lectura del frame fue exitosa se incrementa el contador en 1 para llevar el registro del número de frames procesados.
- Añadir el número del contador al frame capturado utilizando la función "**putText**" de OpenCV.
- Crear un nuevo frame de salida de HDMI mediante la función "**outframe = hdmi_out.newframe()**".
- El contenido del frame capturado se le asigna al frame de salida "**outframe[:] = frame_vga**".
- En el frame de salida HDMI se muestra en pantalla
- Se registra el tiempo de finalización después de procesar todos los frames

Fin del bucle

- Se imprime el mensaje “si se ve” para indicar que la visualización ha sido exitosa
- Se muestra el número de FPS durante la ejecución
- Se imprime el número errores de lectura de la cámara
- Con la función **videoIn.release()** para liberar la cámara y finalizar la captura de video.

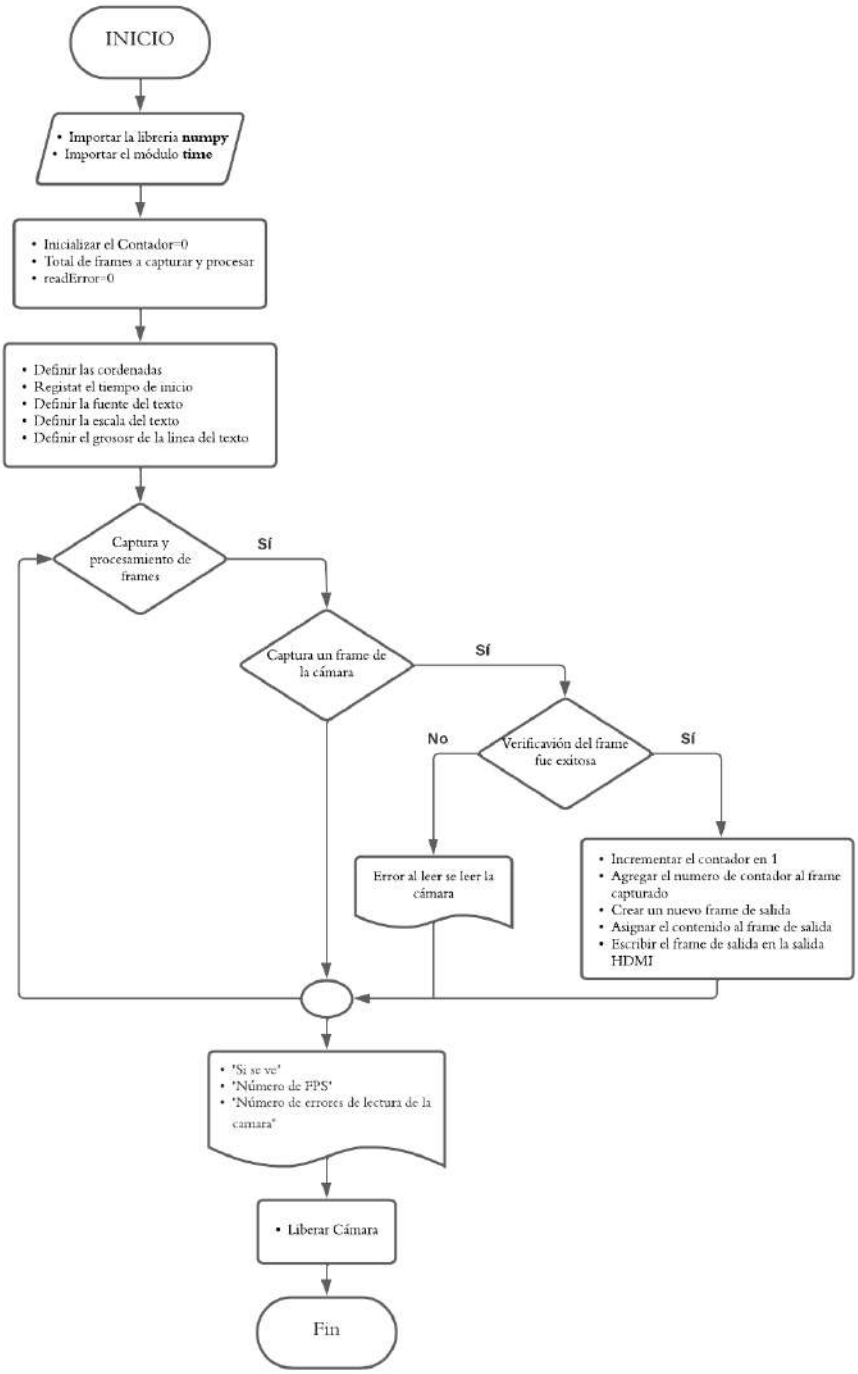


Ilustración 3-12. Diagrama de flujo para la Visualización de Video

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023

- **Bloque 5: Algoritmo Detector de Canes**

En la Ilustración 3-13., *se* representa el diagrama de flujo sobre el algoritmo de detección de canes en tiempo real con la interacción de la tarjeta de desarrollo PYNQ Z1 para controlar actuadores externos según se haya realizado la detección de canes.

Inicio

- Importar las siguientes bibliotecas y/o librerías: cv2, time, os, imutils y pynq.lib.arduino.
  imutils: Esta librería proporciona herramientas de OpenCV que brinda una seri de funciones y métodos para realizar operaciones comunes en imágenes y videos de manera más sencilla y concisa.

  pynq.lib.arduino: Modulo encargado de proporcionar a la PYNQ la comunicación con una placa Arduino así como también controlar el funcionamiento de los LEDs, motor, y buzzer en función de la detección de los canes.

- Configuración de la placa y los componentes hardware, se configura diferentes pines para controlar varios componentes de hardware como el motor, los LEDs y el buzzer.
  Pin 11, utilizado para controlar el buzzer, Pin 12, utilizado para controlar el motor, Pin 10, utilizado como indicador LED de inicio, Pin 9, utilizado como indicador LED en el modo normal, Pin 8, utilizado como indicador LED en el modo detector

- Configuración de la Detección de Objetos. En esta sección, se carga el modelo entrenado previamente por transferencia de aprendizaje y se configura algunos de los parámetros para su uso. **modelodir:** Es la dirección del archivo del modelo entrenado en formato (.pbtx). **pesosdir:** Es el archivo de los pesos del modelo entrenado (.pb).

  Estos archivos contienen la arquitectura y los pesos del modelo entrenado y la biblioteca OpenCV nos proporciona funciones para cargar este tipo de modelos y utilizarlos para detectar objetos en imagen o video.

- Aplicación de funciones para la detección de objetos. Las funciones más importantes son:
  Translate(objeto): Esta función recibe el nombre de una clase detectada. **getobjetos(imagen, thresg,NMS, dibujar=True, objetos[])** Esta función realiza la

detección de objetos en una imagen y devuelve información sobre los objetos detectados. Dibuja una caja delimitadora y etiquetas del objeto detectado.

Bucle de repetición

- El algoritmo entra a un ciclo infinito que se realiza las siguientes acciones: Se lee el estado del botón conectado al pin 13, si el botón esta presionado entra al modo NORMAL es decir se desactiva la detección de canes y se apagan los actuadores (buzzer, motor y algunos LEDs). Si el botón no está presionado entra al modo DETECTOR es decir se desactiva la detección de canes y se apagan los actuadores (buzzer, motor y algunos LEDs)

- Se sigue capturando imágenes de la cámara en cada iteración del bucle. Si la detección esta activada (**DETECTOR=True**), el programa utiliza el modelo de detección de objetos para detectar personas y canes en la imagen. Si detecta un perro, se activa el buzzer y se cuentan las detecciones consecutivas. Si no se detecta canes, el buzzer se desactiva y se cuentan los cuadros consecutivos sin detección.

Fin del Bucle de repetición

- Se termina el buble si se presiona el botón conectado al pin 13. Luego, el programa libera la captura de video, restablece el estado de los compones hardware y se finaliza el algoritmo.

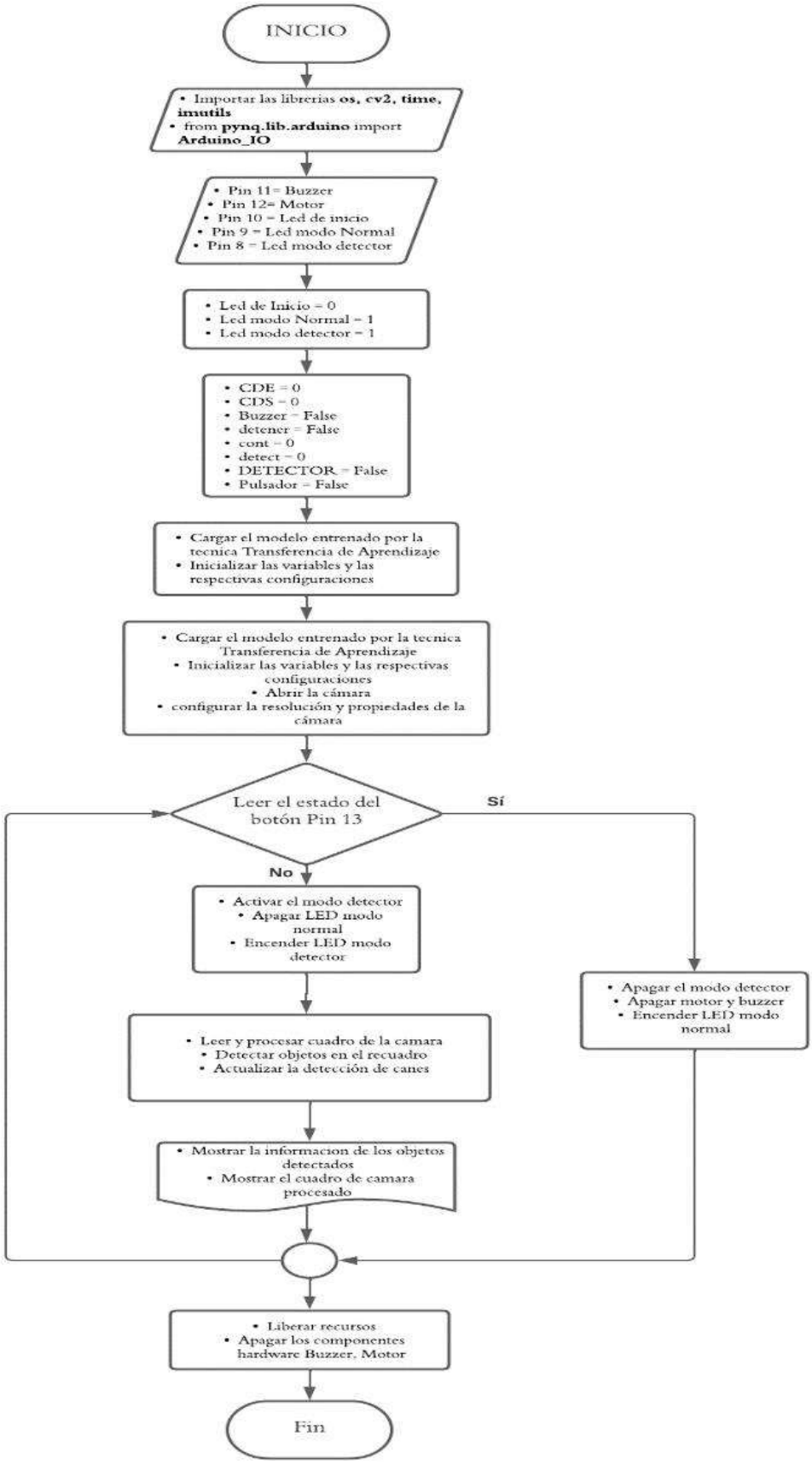


Ilustración 3-13. Diagrama de Flujo del algoritmo detector de canes.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023

3.4.3 Implementación del DDC

Por medio de la Ilustración 3-14, se describe la estructura por bloques de la arquitectura implementada del DDC, la misma se interpreta de la siguiente manera: como parte primordial se encuentra la tarjeta de desarrollo FPGA PYNQ Z1 en la cual se ha realizado la programación explicada en los apartados anteriores, misma que es encarga de realizar todo el procesamiento de la información y dispositivos conectados a la tarjeta de desarrollo. Por otra parte, consta de una fuente de suministro de energía de 5V de DC que tiene la capacidad de brindar la corriente necesaria para el perfecto funcionamiento del dispositivo detector de canes DDC.

Por otra parte, se encuentra los modos de operación que son interactuados por un botón pulsador, el cual al ser presionado nos permitirá cambiar de modos de operación, es decir, al modo detector y al modo normal.

Para finalizar, en el puerto USB de la PYNQ Z1 se encuentra el módulo de adquisición de imágenes que se encarga de enviar toda la información que se presente en el entorno para que a su vez acorde al modo de operación y la detección de canes el módulo de actuadores dentro del sistema embebido se active o se desactive según sea el caso de operación y/o detección de presencia de canes.

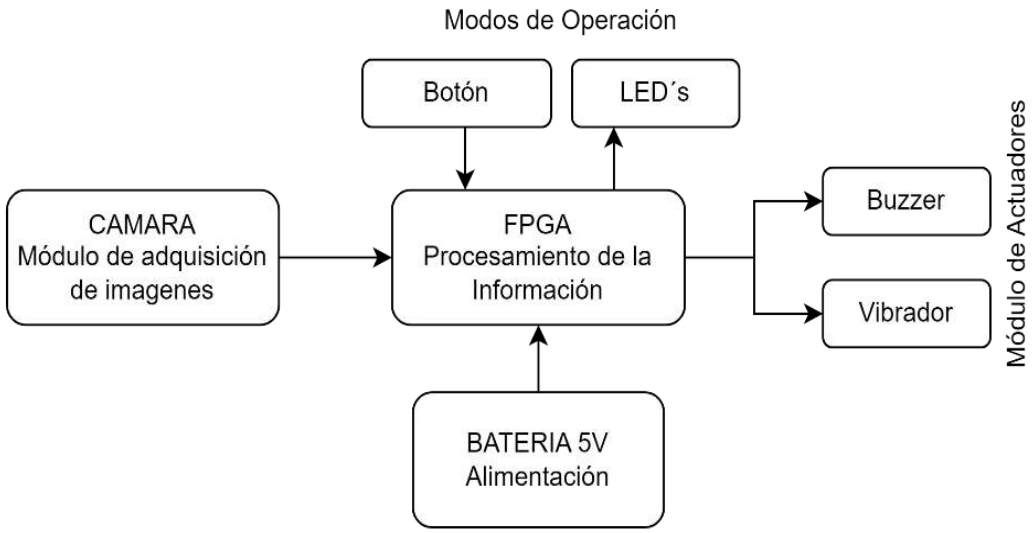


Ilustración 3-14. Diagrama de bloque representativo de la Arquitectura del Dispositivo

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

3.5 Validación y Pruebas

En este apartado abarca le descripción detallada de los entornos en los cuales se llevarán a cabo las pruebas, considerando diferentes escenarios tanto como en ambientes controlados y no controlados. Además, se va realiza la verificación de los requerimientos previamente establecidos.

3.5.1 Pruebas de Funcionamiento

El proceso de pruebas de funcionamiento se desarrolla siguiendo un enfoque sistemático. En el cual se establece los casos de pruebas en los ambientes controlados y no controlados. Esto con el fin de tener los parámetros claros para evaluar y validar el rendimiento del dispositivo DDC.
En este sentido, se recopila información en dos tipos de ambientes. Cabe mencionar que estas pruebas fueron con dos siluetas de canes en MDF simulando la presencia de este, tal como se muestra en la Ilustración 3-15.

Ilustración 3-15. Siluetas de canes elaboradas en MDF

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

3.5.1.1 Pruebas de Funcionamiento en Ambientes Controlados

En la Tabla 3-11., se explica las cómo van a ser desarrolladas las pruebas de funcionamiento en ambientes controlados. Para estas pruebas se adecuo un escenario exclusivo de 12x4 metros, y se marcó sobre el piso diez puntos referenciales a una distancia de separación de un metro entre ellas, colocando el dispositivo en un sitio fijo al inicio de dicha señalización. Estas pruebas serán fundamentales para verificar el funcionamiento básico del DDC y obtener resultados óptimos.

Tabla 3-11. Tipos de pruebas a realizar en ambientes controlados

Tipo de Prueba	**Condiciones**	**Recopilación de datos**
Prueba de detección	Colocamos la silueta del can a una distancia de un metro para posteriormente mover la silueta a distancias progresivas y llegar hasta una distancia de 10m.	Se evalúa si el dispositivo detecto se marca como un valor de "1" y cuando no detecta adquiere un valor de "0"
Tiempo de detección a 3 metros	Colocamos la silueta del can a una distancia fija de 3 metros, cubrimos la cámara del DDC y con la ayuda de un cronometro al descubrir la cámara tomamos el tiempo que se demora en detectar el can, este proceso lo repetimos por 20 veces.	Se recopila los segundos que tarda el DDC en detectar al can.
Tiempo de detección a 5.5 metros	Colocamos la silueta del can a una distancia fija de 5.5 metros, cubrimos la cámara del DDC y con la ayuda de un cronometro al descubrir la cámara tomamos el tiempo que se demora en detectar el can, este proceso lo repetimos por 20 veces.	Se recopila los segundos que tarda el DDC en detectar al can.
Prueba de diferenciación Persona - Can	Nos colocamos a lado de la silueta del can a una distancia de 3 metros hasta los 7 metros, y verificamos si el dispositivo diferencia personas con canes ayudándonos de una pantalla HDMI.	Recopilar los datos según sea el caso de diferenciación de persona- can.
Pruebas de detección varios canes	Colocamos primera silueta a una cualquier distancia, y la segunda silueta de can la colocamos a una distancia diferente de la primera y verificamos si identifico a los canes ayudándonos de una pantalla HDMI	Recopilar los datos según la detección de los diferentes canes

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

3.5.1.2 Pruebas de Funcionamiento en Ambientes no Controlados

Para probar el dispositivo en situaciones no controladas, se toma en cuenta condiciones de iluminación adversar como factor más influyente, así como también se desarrolla en calles alternativas con el fin de cambios de escenarios, con el fin de asegurarse de que el sistema pueda manejar casos variados.

Tabla 3-12. Tipos de pruebas a realizar en ambientes no controlados

Tipo de prueba	Descripción	Horarios
Prueba de detección	Colocamos la silueta del can a una distancia de un metro para posteriormente mover la silueta a distancias progresivas y llegar hasta una distancia de 10m.	Alternativos
Tiempo de detección a 3 metros	Colocamos la silueta del can a una distancia fija de 3 metros, cubrimos la cámara del DDC y con la ayuda de un cronometro al descubrir la cámara tomamos el tiempo que se demora en detectar el can, este proceso lo repetimos por 20 veces.	07:00 AM 13:00 PM 19:00 PM
Tiempo de detección a 5.5 metros	Colocamos la silueta del can a una distancia fija de 5.5 metros, cubrimos la cámara del DDC y con la ayuda de un cronometro al descubrir la cámara tomamos el tiempo que se demora en detectar el can, este proceso lo repetimos por 20 veces.	07:00 AM 13:00 PM 19:00 PM
Prueba de diferenciación Persona - Can	Nos colocamos a lado de la silueta del can a una distancia de 3 metros hasta los 7 metros, y verificamos si el dispositivo	Alternativos
Pruebas de detección varios canes	Colocamos primera silueta a una cualquier distancia, y la segunda silueta de can la colocamos a una distancia diferente de la primera y verificamos si identifico a los canes ayudándonos de una pantalla HDMI	Alternativos

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

3.5.2 Verificación de los Requerimientos

Para llevar a cabo la verificación de los requerimientos establecidos al inicio de este capítulo, se lo realiza mediante el uso de una tabla de verificación conocida como checklist, con el fin de garantizar su cumplimiento integral del dispositivo.

En la Tabla 3-13., se muestra la lista de verificación, cuya herramienta es esencial para realizar un seguimiento de cada requisito y evaluar el estado de cumplimiento de cada requerimiento.

Tabla 3-13. Lista de verificación de los requerimientos

Requerimientos	**¿Cumplido?**	
	Si	**No**
Las dimensiones del dispositivo no obstaculizan la visión del conductor, ni afecta el funcionamiento del vehículo	✓	
El sistema detector esta implementado en una FPGA	✓	
La comunicación de la cámara y la FPGA es mediante conexión USB	✓	
La cámara está ubicada en una zona exterior del vehículo	✓	
El dispositivo DDC es alimentado con una fuente externa de 6V, 4.5Ah	✓	
Existe el modo de operación normal	✓	
Existe el modo de operación detector	✓	
Existe un led que da aviso si el dispositivo se ha alimentado correctamente	✓	
Existe un led que da un aviso si el dispositivo está operando el modo normal	✓	
Existe un led que da un aviso si el dispositivo está operando el modo detector	✓	
En el modo de operación normal los actuadores “Buzzer y vibrador” se desactivan y el dispositivo ejecuta acciones de video	✓	
En el modo de operación detector los actuadores “Buzzer y vibrador” se activan cuando el dispositivo haya detectado la presencia de canes	✓	

Requerimientos	¿Cumplido?	
	Si	No
El dispositivo es capaz de detectar canes con la iluminación ambiente de la mañana “7:00 am”	✓	
El dispositivo es capaz de detectar canes con la iluminación ambiente de la tarde “13:00 pm”	✓	
El dispositivo es capaz de detectar canes con la iluminación ambiente de la noche e iluminación externa del vehículo “19:00 pm”	✓	
El DDC en el ambiente no controlado en el horario de la mañana 7:00 am. A una distancia de 3 metros, el tiempo de detección es menor a 3.5 segundos.	✓	
El DDC en el ambiente no controlado en el horario de la tarde 13:00 pm. A una distancia de 3 metros, el tiempo de detección es menor a 4 segundos.	✓	
El DDC en el ambiente no controlado en el horario de la tarde 19:00 pm. A una distancia de 3 metros, el tiempo de detección es menor a 5 segundos.	✓	
El DDC en el ambiente no controlado en el horario de la mañana 7:00 am. A una distancia de 5.5 metros, el tiempo de detección es menor a 8 segundos.	✓	
El DDC en el ambiente no controlado en el horario de la tarde 13:00 pm. A una distancia de 5.5 metros, el tiempo de detección es menor a 9 segundos.	✓	
El DDC en el ambiente no controlado en el horario de la tarde 19:00 pm. A una distancia de 5.5 metros, el tiempo de detección es menor a 10 segundos.	✓	
El dispositivo DDC es capaz de diferenciar canes - personas	✓	
El dispositivo DDC es capaz de detectar uno o más canes	✓	

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

CAPITULO IV

4. PROPUESTA Y DISEÑO DEL PROTOTIPO

4.1 Esquema Electrónico para DDC.

El esquema de conexiones muestra cómo se interconectan los componentes eléctricos en un circuito, permite visualizar de manera clara y precisa cómo se conectan los diferentes elementos para que el circuito encargado de la detección y alerta de la presencia de canes en las vías funcione de la mejor manera. Este dispositivo está compuesto por una tarjeta de desarrollo FPGA PynQ Z1 de Xilinx, cargas inductivas (motor vibrador y buzzer), indicadores led, interruptor y Cámara USB 1080 P OV2710 como se muestra en la Ilustración 4-1.

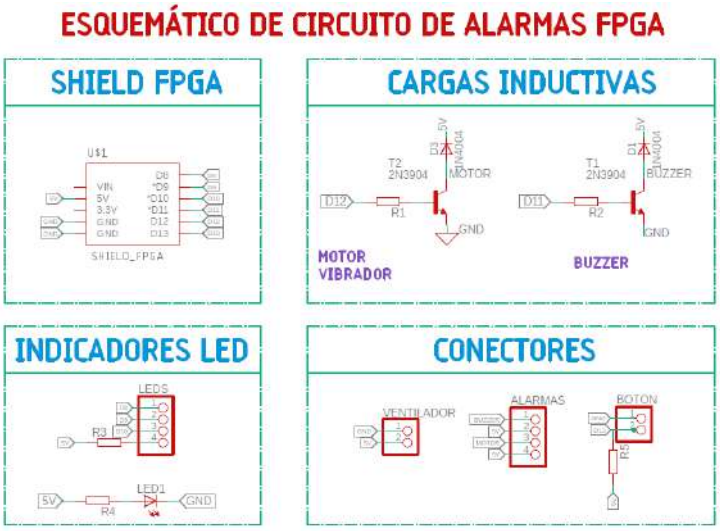


Ilustración 4-1. Esquematico general de conexiones del DDC.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

Acorde a la Ilustración 4-1, las conexiones del dispositivo detector de canes vendrían dado de la siguiente manera:

- La alimentación del dispositivo estará conectada en los pines de 5V y GND de la tarjeta de desarrollo FPGA PynQ Z1

- Dentro de la carga inductiva, el motor vibrador está conectado al colector del transistor 2N3904 y el emisor de dicho transistor está conectado a GND y en la base del transistor se coloca una resistencia que se conectara al pin 12 de la FPGA.

- La otra carga inductiva (buzzer zumbador) también está conectada al colector de un transistor 2N3904, mientras que el emisor está conectado a GND y su base se conecta con una resistencia al pin 13 de la FPGA.

- De los pines D8, D9 y D10 de la tarjeta de desarrollo PynQ Z1 se conecta al Diodo Led RGB respectivamente con los pines R, G y B compartiendo una común llamada GND.

- Dentro de la FPGA se le conecta a la salida del pin 13 una resistencia en serie con el interruptor, dicha conexión va a la salida de 5V de la misma tarjeta, con el objetivo de encender y cambiar el modo de operación del dispositivo.

- La cámara USB 1080 P OV2710 se conecta al puerto USB de la Tarjeta de Desarrollo FPGA PynQ Z1.

4.2 Diseño de la PCB para activación de alarmas

El diseño de la PCB se realizó en el software desarrollador Eagle, seleccionado los componentes electrónicos necesarios para el sistema embebido utilizado para la detección de la presencia de canes en las vías y se colocaron en el esquemático, estableciendo la conexión entre ellos y posterior a eso se genera los archivos de producción para la impresión de esta.

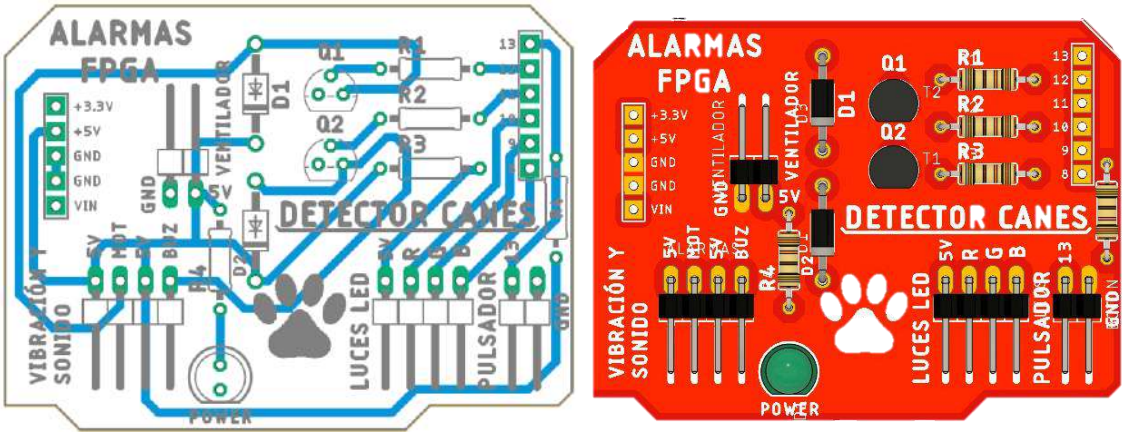


Ilustración 4-2. Diseño de la PCB utilizado para la activación de alarmas.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

4.3 Diseño de la carcasa para DDC (Dispositivo Detector de Canes)

- **Diseño de la parte superior de la Carcasa del DDC**

En el diseño de la estructura externa del Sistema Embebido, se consideró que la tapa superior estará realizada en el software SolidWorks versión 2023 y posterior a ello se guarda el diseño en un formato SVG (Scalable Vector Graphics) el cual es interpretado por la máquina de corte laser que serán encargada de crear esta pieza en material acrílico.

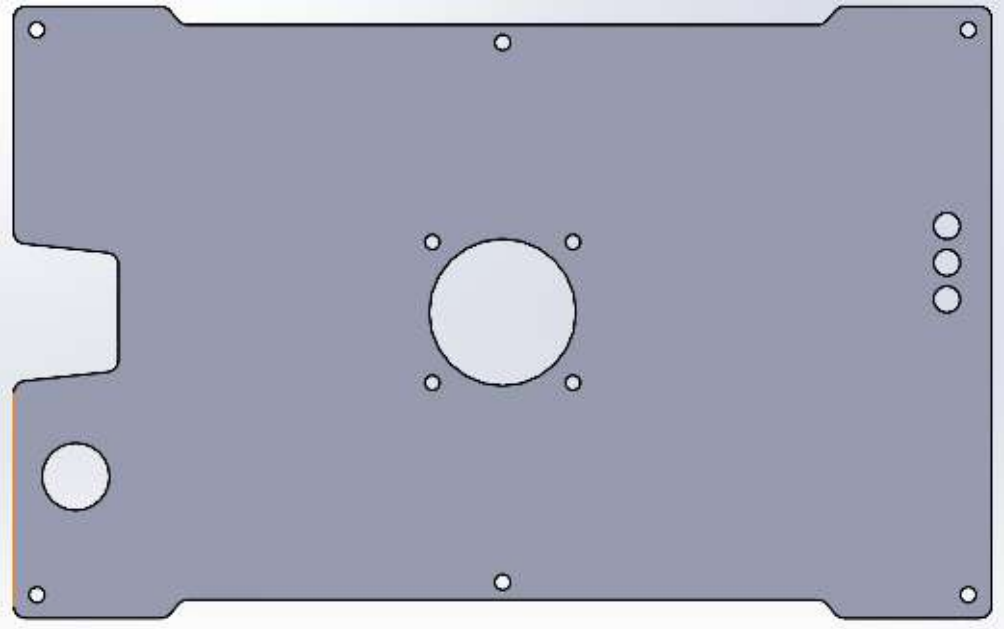


Ilustración 4-3. Diseño de la tapa para el corte laser.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

- **Diseño posterior de la Carcasa del DDC**

En la Ilustración 4-4, se observa el diseño gráfico de la base de la estructura de la parte interna de la carcasa del dispositivo, mientras que en la Ilustración 4-5, se observa la estructura interna de la base de la estructura del sistema embebido, estos diseños están realizados en el software SolidWorks versión 2023.

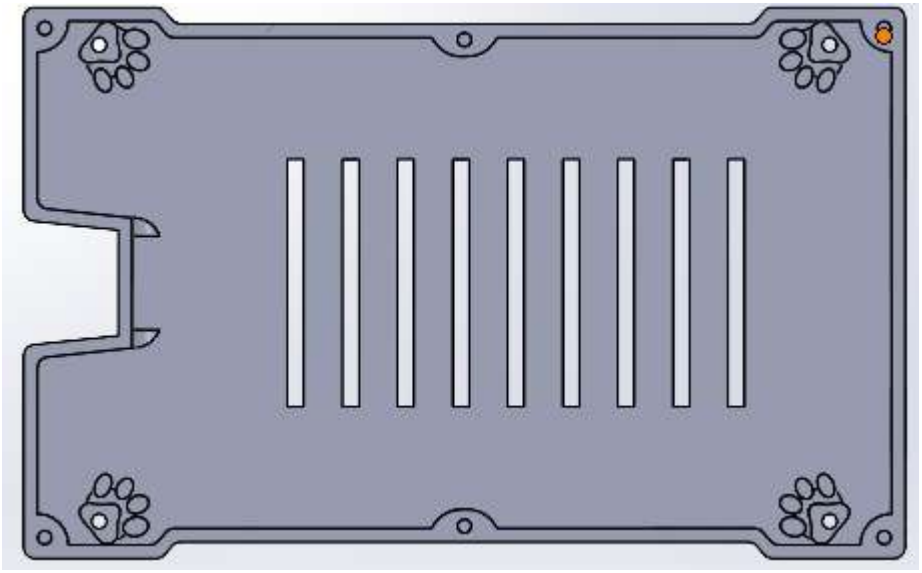


Ilustración 4-4. Diseño de la parte interna de la base de la carcasa del DDC.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

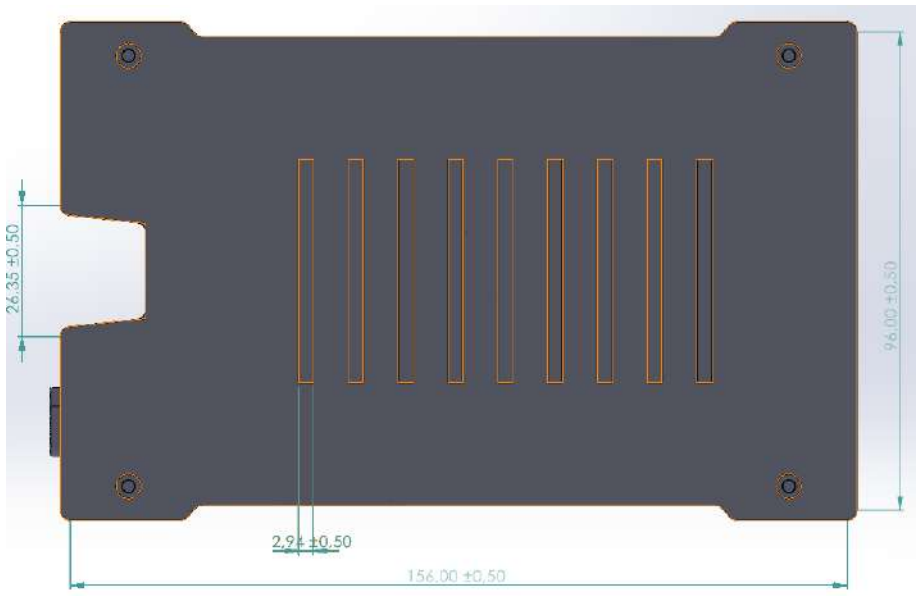


Ilustración 4-5. Vista de la parte externa de la base de la carcasa del DDC.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

- **Diseño lateral de la Carcasa del DDC**

Mediante la Ilustración 4-6, y la Ilustración 4-7, se representa gráficamente el diseño de la parte frontal y lateral del Dispositivo Detector de Canes en las vías, los mismos que fueron elaborados en el software diseñador SolidWorks versión 2023.

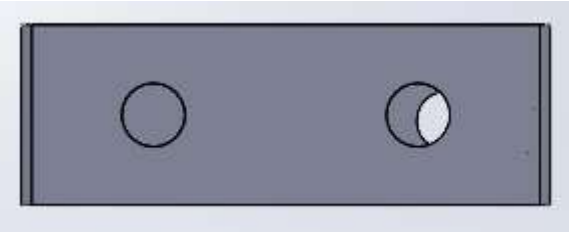


Ilustración 4-6. Vista frontal de la estructura del DDC.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

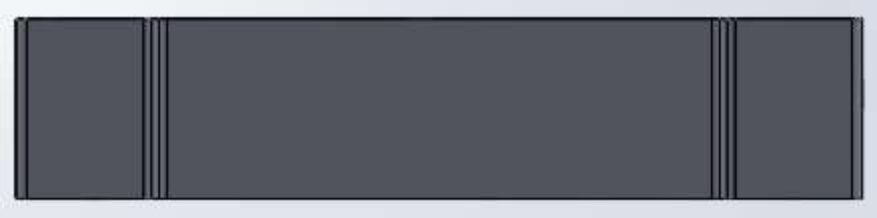


Ilustración 4-7. Vista lateral de la estructura del DDC.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

- **Diseño de la portada de la Carcasa del DDC**

Por medio de la portada se busca representar mediante símbolos gráficos y literarios todo lo que estará inmerso dentro de este sistema embebido. La Ilustración 4-8, evidencia mediante las gráficas de la izquierda el objetivo principal de este dispositivo como es la detección de la presencia de canes, mientras que en la parte derecha de la portada se observa los modos de funcionamiento con los que cuenta el dispositivo.

Ilustración 4-8. Diseño de la Portada colocada en la parte superior del DDC.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

4.4 Algoritmo General del Dispositivo DDC

El algoritmo general del dispositivo DDC se aplica desde dos entornos de programación, el proceso de desarrollo y entrenamiento de la red neuronal mediante la técnica de transferencia de aprendizaje se ejecuta en la plataforma Google Colab, para posteriormente la integración del modelo resultante en el algoritmo de detección en la plataforma Jupyter Notebook. Por tal motivo se exploran las etapas de procesamiento de imágenes, detección y activación de alarmas, destacando la interconexión de estas fases para lograr un sistema funcional. El resultado del algoritmo general del dispositivo DDC combina la potencia de transferencia de aprendizaje con la aplicación práctica de detección y activación de alarmas.

En este sentido, en la Ilustración 4-9., se detalla en forma de diagrama de flujo los pasos esenciales del algoritmo de detección y alarma de canes.

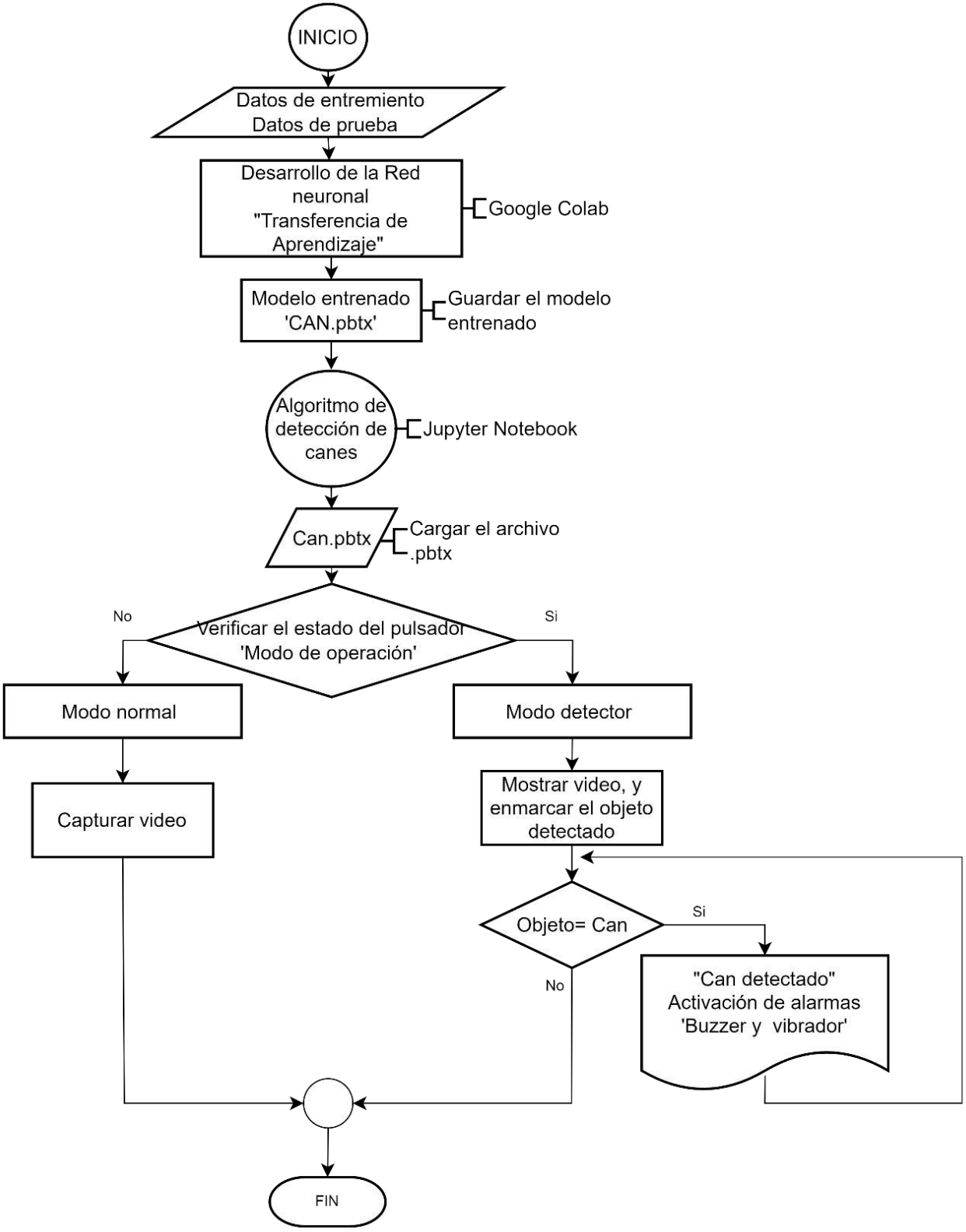


Ilustración 4-9. Diagrama de flujo del algoritmo general del dispositivo DDC

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

CAPITULO V

5. VALIDACIÓN DEL PROTOTIPO

La validación del Dispositivo diseñado para la detección de la presencia de canes en las vías consiste en verificar y evaluar la precisión y confiabilidad del dispositivo para detectar de manera efectiva la presencia de canes en las carreteras. Durante la validación, se llevan a cabo diferentes pruebas y análisis para evaluar el rendimiento del dispositivo en condiciones reales. Esto implica la recopilación de datos de detección en diferentes escenarios, como diferentes distancias de detección, tiempo de respuesta, curvas de aprendizaje, pruebas de normalidad y la precisión general del sistema.

En este apartado se considerará la obtención de los resultados por medio de tablas y el resultado obtenido será representado de manera estadística por barras y acompañados de una descripción de cada una.

5.1 Pruebas de Validación del Modelo Entrenado

5.1.1 Curva de Precisión de Entrenamiento y Pruebas

La Ilustración 5-1, describe una gráfica creciente con tendencia a la unidad que representa la precisión con la que cuenta el modelo entrenado con respecto a las 30 épocas consideradas para el entrenamiento. Al manejar un número considerable de datos al momento de entrenar la red neuronal (1500 imágenes) permite tener una exactitud igual a la unidad, la misma, que equivale al 100% haciendo que el modelo de entrenamiento sea confiable en su toma de decisiones.

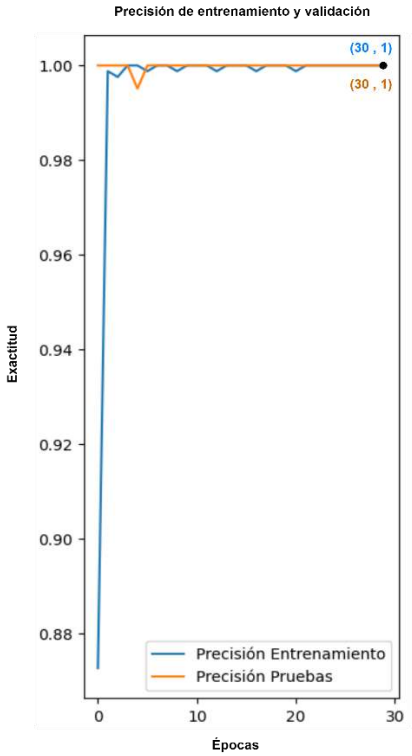


Ilustración 5-1. Descripción de la curva de precisión obtenido en el Modelo Entrenado.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

5.1.2 Curva de Pérdida de Entrenamiento y Pruebas

Haciendo referencia lo descrito anteriormente sobre el número de datos utilizados tanto para el entrenamiento de la red neuronal como para las pruebas de funcionamiento de esta. Se puede observar en la Ilustración 5-2, como se va comportando el modelo entrenado a medida que va avanzando en las épocas de entrenamiento. A medida que el entrenamiento avanza hasta cumplir las 30 épocas se evidencia que la tendencia de la gráfica con respecto a la perdida de los datos en el entrenamiento alcanza un valor de exactitud de 0.0012, mientras que la perdida en términos de exactitud de los datos de validación es de 0.0015.

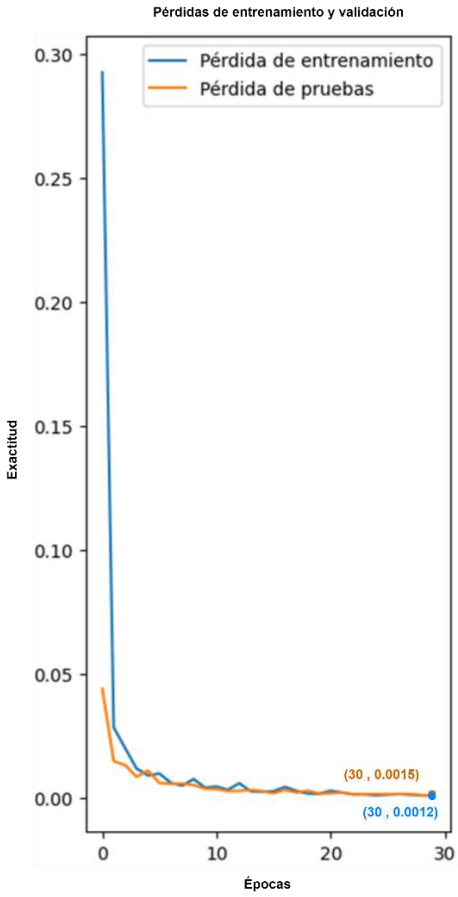


Ilustración 5-2. Descripción de la curva de pérdida obtenido en el Modelo Entrenado.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

5.2 Pruebas Estáticas de Experimentación sobre el S.E implementado

Mediante las pruebas se verifica que el DDC cumpla con los requerimientos y especificaciones establecidas al inicio de la investigación. Obteniendo información de las variables más representativas que intervienen dentro del funcionamiento del dispositivo y analizando su comportamiento ante diversas condiciones.

5.2.1 Pruebas de Campo en Ambiente Controlado

Estas pruebas se realizan en condiciones cuidadosamente controladas y reproducibles, donde se puede manipular y ajustar los parámetros de manera precisa. Se adecuo un escenario exclusivamente para esta prueba (12x4 metros), se utilizó herramientas confiables de medición para colocar sobre el piso un máximo de diez puntos de referencias a una distancia de separación de un metro entre ellas. Posterior a ello se colocó el dispositivo al inicio de dicha señalización con el objetivo que el primer punto de referencia se encuentra a una distancia de un metro de

distancia y finalmente se coloca al otro extremo de las referencias una silueta bidimensional de un can elaborada en MDF.

5.2.1.1 Distancia de detección de presencia de canes

Entre las pruebas de verificación que se debe realizar al DDC se encuentra la Distancia de Detección que hace referencia a la distancia a la cual el dispositivo puede identificar la presencia de un can con precisión. La *Tabla 5-1*, describe de manera textual si el dispositivo detecto o no al can dentro del ambiente controlado.

Tabla 5-1. Prueba de distancia de detección de la presencia de canes.

Distancia (metros)	**Prueba 1**	**Prueba 2**	**Prueba 3**	**Prueba 4**	**Prueba 5**
1	Detecto	Detecto	Detecto	Detecto	Detecto
2	Detecto	Detecto	Detecto	Detecto	Detecto
3	Detecto	Detecto	Detecto	Detecto	Detecto
4	Detecto	Detecto	Detecto	Detecto	Detecto
5	Detecto	Detecto	Detecto	Detecto	Detecto
6	Detecto	No Detecto	No Detecto	Detecto	Detecto
7	No Detecto	No Detecto	Detecto	No Detecto	No Detecto
8	No Detecto	Detecto	No Detecto	No Detecto	No Detecto
9	No Detecto	No Detecto	Detecto	No Detecto	No Detecto
10	No Detecto	No Detecto	No Detecto	No Detecto	No Detecto

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

Para el análisis e interpretación de los datos recolectados, se coloca el valor de 1 en los casos donde el dispositivo detecto y se representó con la palabra “Detecto”, mientras que se coloca 0 en los casos donde no se detectó la presencia del can y se representó con la palabra “No Detecto”. La *Tabla 5-2*, describe la interpretación de los datos que se mencionó en el párrafo anterior y el análisis estadístico (promedio, desviación estándar y varianza) utilizado para el conjunto de datos. En cuanto al valor del promedio se evidencia que se tiene una respuesta favorable del dispositivo hasta los seis metros de distancia con un 60% de detección. Mientras que la desviación y la varianza van tomando ciertos valores a medida que la distancia de prueba es mayor debido a la perdida de precisión en la detección de la presencia del can.

Tabla 5-2. Representación y análisis de los datos recolectados en la detección de canes en ambientes controlados.

Distancia (metros)	Prueba1	Prueba2	Prueba3	Prueba4	Prueba5	Promedio en %	Desviación	Varianza
1	1	1	1	1	1	100%	0	0
2	1	1	1	1	1	100%	0	0
3	1	1	1	1	1	100%	0	0
4	1	1	1	1	1	100%	0	0
5	1	1	1	1	1	100%	0	0
6	1	0	0	1	1	60%	0,55	0,49
7	0	0	1	0	0	20%	0,45	0,40
8	0	1	0	0	0	20%	0,45	0,40
9	0	0	1	0	0	20%	0,45	0,40
10	0	0	0	0	0	0%	0	0

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

La Ilustración 5-3, representa la relación entre la distancia (metros) y el promedio obtenido en las pruebas realizada al dispositivo detector de canes. En el eje vertical, se muestra el valor del promedio de las pruebas, lo que proporciona una medida consolidada de su desempeño. En el eje horizontal, se representa la distancia a la cual se llevaron a cabo estas pruebas. Mediante el grafico de dispersión se puede hacer un análisis descriptivo que arroja la distancia mínima (3 metros) a la que el conductor puede tomar una decisión sin afectar el comportamiento del vehículo y una distancia máxima a la que el dispositivo detecta la presencia de canes sin ninguna complicación (5.5 metros).

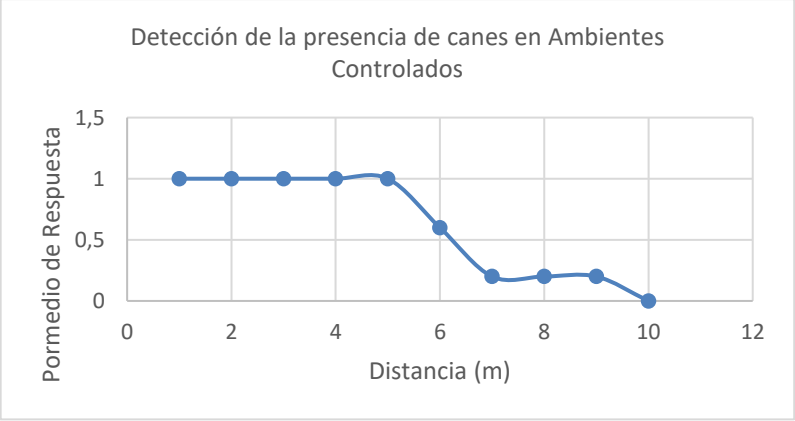


Ilustración 5-3. Gráfico Scatter de los resultados obtenidos en la detección de canes en Ambientes Controlados.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

5.2.1.2 Tiempo de Detección a la distancia de 3 metros

El tiempo de detección es un parámetro crucial al momento de validar el funcionamiento de un sistema embebido. Un tiempo de detección rápido y preciso permite accionar de manera oportuna al dispositivo. La Tabla 5-3, hace referencia al tiempo obtenido en veinte pruebas recolectadas en un ambiente controlado a una distancia de detección mínima (3 metros) para que el accionar del conductor pueda llevarse con normalidad. En la parte final se evidencia los valores estadísticos de tendencia central y la dispersión del conjunto de datos, se describe una media de 3.43 segundos que representa el tiempo promedio que el dispositivo tarda en detectar la presencia del can, la desviación de 0.38 segundos señala la variabilidad típica en los tiempos de detección con respecto a la media y la varianza de 0.14 segundos refleja la magnitud de las diferencias entre los valores individuales y la media.

Tabla 5-3. Tiempo de detección de la presencia de canes a 3 metros de distancia.

Numero de Prueba	**Tiempo**
1	3.85 seg
2	3.92 seg
3	3.08 seg
4	2.83 seg
5	3.54 seg
6	2.91 seg
7	2.98 seg
8	2.80 seg
9	3.62 seg
10	3.72 seg
11	3.57 seg
12	3.36 seg
13	3.68 seg
14	3.95 seg
15	3.64 seg
16	3.30 seg
17	2.98 seg
18	3.78 seg
19	3.67 seg
20	3.51 seg

ANALISIS ESTADISTICO	
Media	3.43 seg
Desviación	0.38 seg
Varianza	0.14seg

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

La Tabla 5-4, evidencia los valores obtenidos al realizar la prueba de Shapiro Wilk al conjunto de datos del tiempo de detección a una distancia de 3 metros en ambientes controlados. El estadístico de Shapiro-Wilk arroja un resultado de 0.91, este valor sugiere que los datos no se alejan significativamente de una distribución normal. El p-valor de 0.063 es un indicador clave para las pruebas de hipótesis, indicando que no hay suficiente evidencia para rechazar la hipótesis nula debido a que es mayor que el nivel de significancia (Alpha). En resumen, los valores obtenidos muestran una tendencia de distribución normal.

Tabla 5-4. Prueba de Shapiro-Wilk para los datos del tiempo de detección a 3 metros en ambientes controlados.

Estadístico	0.91
p-valor	0.063
Alpha	0.05
Normal	Si

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

La Ilustración 5-4, muestra la variabilidad en el tiempo de detección en función del número de pruebas realizadas en el dispositivo detector de canes. En el eje vertical, se representa el tiempo requerido para la detección en cada prueba, mostrando valores que oscilan entre los 2.5 y 4 segundos. En el eje horizontal, se representa el número de pruebas con las que se trabajó.

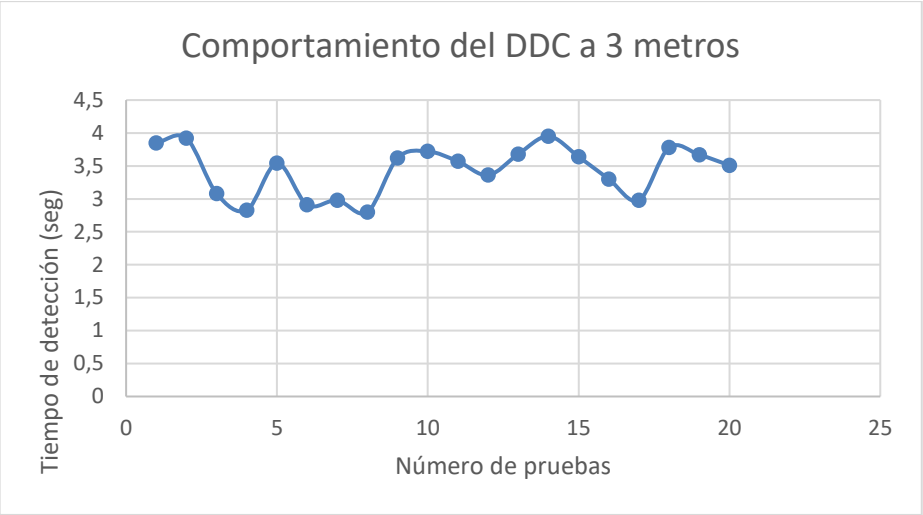


Ilustración 5-4. Gráfico Scatter del Tiempo de Detección del DDC a tres metros de distancia.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

5.2.1.3 Tiempo de Detección a la distancia máxima

La Tabla 5-5, evidencia los veinte datos recolectados del dispositivo detector de la presencia de canes en las vías a una distancia máxima de 5.5 metros, se considera este valor como máximo acorde a los resultados obtenidos en la prueba de detección de la presencia de canes en las vías. Mediante el análisis de tendencia central y dispersión se revela una media de 7.76 segundos, que indica el tiempo promedio necesario para la detección en esta distancia especifica. La desviación estándar de 0.69 segundos sugiere que los tiempos de detección varían alrededor de esta media en un grado relativamente moderado. Mientras que la varianza 0.48 segundos refuerza la idea de que los tiempos de detección no son uniformes, ya que muestra la dispersión de los valores individuales con respecto a la media.

Tabla 5-5. Tiempo de detección de la presencia de canes a 5.5 metros de distancia.

Numero de Prueba	**Tiempo**
1	8.43 seg
2	8.55 seg
3	7.35 seg
4	8.49 seg
5	7.82 seg
6	7.29 seg
7	7.46 seg

Numero de Prueba	**Tiempo**
8	7.86 seg
9	6.98 seg
10	6.70 seg
11	7.18 seg
12	6.64 seg
13	6.67 seg
14	7.62 seg
15	8.13 seg
16	8.42 seg
17	8.17 seg
18	8.65 seg
19	8.59 seg
20	8.27 seg
Media	7.76 seg
Desviación Estándar	0.69 seg
Varianza	0.48 seg

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

La Tabla 5-6, describe los resultados obtenidos en la prueba de normalidad de Shapiro Wilk que se aplicó al conjunto de datos del tiempo de detección de un dispositivo a una distancia máxima (5.5 metros) en un ambiente controlado.

El valor del estadístico de prueba es de 0.91, indicando que los datos no se alejan significativamente de una distribución normal. El p-valor obtenido es de 0.063, que es mayor que el nivel de significancia (Alpha) establecido de 0.05. Esto sugiere que no hay suficiente evidencia para rechazar la hipótesis nula de normalidad. En conjunto, estos resultados establecen que los datos siguen una distribución aproximadamente normal.

Tabla 5-6. Prueba de Shapiro-Wilk para los datos del tiempo de detección a 5.5 metros en ambientes controlados.

Estadístico	0.91
p-valor	0.063
Alpha	0.05
Normal	Si

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

La Ilustración 5-5, muestra el comportamiento oscilante que varía en el rango de 6 a 9 segundos en el eje vertical, representando el tiempo que tarda el DDC en detectar la presencia de canes en un ambiente controlado, determinado por el número de pruebas representadas en el eje horizontal. El patrón oscilante indica que a medida que aumenta el número de pruebas, el tiempo de detección experimenta una tendencia variante, evidenciando un comportamiento no lineal.

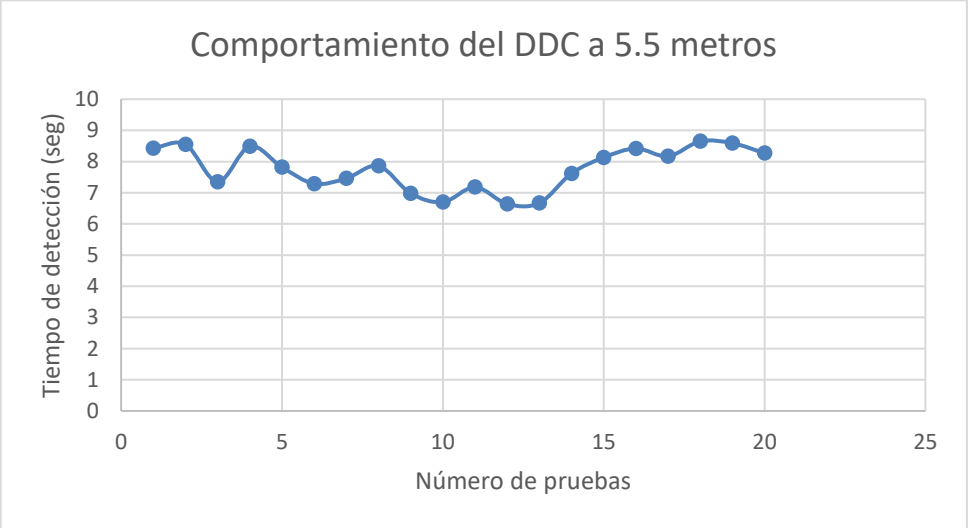


Ilustración 5-5. Gráfico Scatter del Tiempo de Detección del DDC a 5.5 metros de distancia.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

5.2.1.4 Diferenciación Persona-Can

Para evaluar la eficiencia y fiabilidad del Dispositivo Detector de Canes se analiza el comportamiento de dicho dispositivo ante la presencia de más de un objeto frente a su módulo de adquisición de imágenes. Siendo uno de los objetos una persona de características estándar y el otro objeto la silueta bidimensional del can. La Tabla 5-7, revela una relación entre la distancia de detección y la capacidad de diferenciar entre dos objetos. En distancias comprendidas entre 3 y 5.5 metros, el dispositivo muestra un rendimiento efectivo al distinguir entre la persona y el can, lo que resulta en una activación exitosa de las alarmas debido a la detección del can. Este comportamiento coherente en este rango indica una funcionalidad satisfactoria del detector.

Sin embargo, entre los 6 y 7 metros, la tabla refleja una disminución en la capacidad de diferenciación entre persona y can. En estos casos, el dispositivo no logra distinguir claramente entre los dos objetos, lo que resulta la no activación de la alarma que alerta la presencia del can. Esto ratifica el resultado obtenido en la detección de presencia de canes, que indica que en distancias más allá de 5.5 metros, el dispositivo experimenta ciertas limitaciones.

Tabla 5-7. Diferenciación Persona-can del Sistema Embebido en un Ambiente Controlado**.**

Distancia (metros)	Detección Can	Detección Persona	Observación
3	X		Detecto el can y activo las alarmas
3.5	X		Detecto el can y activo las alarmas
4	X		Detecto el can y activo las alarmas
5	X		Detecto el can y activo las alarmas
5.5	X		Detecto el can y activo las alarmas
6	X	X	Detecto la presencia de los dos objetos.
7	-	-	No detecto la presencia de ninguno

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

5.2.1.5 Detección varios canes

Sabiendo que el objetivo primordial de este dispositivo es detectar la presencia de canes en las vías, se vio la necesidad de analizar cuál sería el comportamiento del dispositivo ante la presencia de varios canes en el trayecto. Al tratarse de un ambiente controlado se consideró el uso de dos siluetas bidimensional de canes a diferentes distancias. En la Tabla 5-8, se ha realizado cuatro análisis independientes para examinar el desempeño del DDC frente a la presencia simultanea de dos canes a distintas distancias. En todos los casos se observa que el dispositivo muestra una tendencia consistente a detectar con mayor precisión al can que se encuentra en la distancia más cercana. Estos resultados demuestran una relación consistente entre la proximidad del can al dispositivo y su capacidad de detección, a medida que los canes se acercan al dispositivo hasta llegar a una distancia mínima de 3 metros, la probabilidad de detección aumenta notablemente.

Tabla 5-8. Descripción de 4 casos de detección de varios canes a diferente distancia.

Canes	3 metros	3.5 metros	4 metros	5 metros	5.5 metros	Observación
Can 1	X					Detecto al Can 1
Can 2			X			
Can 1			X			Detecto al Can 2
Can 2	X					
Can 1			X			Detecto al Can 1
Can 2					X	
Can 1					X	Detecto al Can 2
Can 2		X				

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

5.2.2 Pruebas de Campo en Ambiente no Controlado

Dichas pruebas se llevan a cabo en situaciones reales y variadas, donde el dispositivo se enfrenta a condiciones no predecibles y cambiantes, en este caso se consideró a la iluminación como un factor no controlado.

5.2.2.1 Distancia de Detección de la presencia de canes

La Tabla 5-9, recopilo datos obtenidos al realizar las pruebas realizadas con el DDC de detección de la presencia de canes en ambientes no controlados a diversas distancias. Se recolectaron datos de 10 distancias distintas, y para cada una de estas distancias se realizaron cinco pruebas. Durante cada prueba, se registró si el dispositivo “Detecta” o “No Detecto” la presencia del can en función de la respuesta obtenida.

Tabla 5-9. Distancia de detección del DDC en ambientes no controlados.

Distancia (metros)	Prueba 1	Prueba 2	Prueba 3	Prueba 4	Prueba 5
1	Detecto	Detecto	Detecto	Detecto	Detecto
2	Detecto	Detecto	Detecto	Detecto	Detecto
3	Detecto	Detecto	Detecto	Detecto	Detecto
4	Detecto	Detecto	Detecto	Detecto	Detecto
5	Detecto	Detecto	Detecto	Detecto	Detecto
6	Detecto	Detecto	No Detecto	No Detecto	Detecto
7	Detecto	No Detecto	No Detecto	No Detecto	No Detecto
8	No Detecto	No Detecto	No Detecto	No Detecto	No Detecto
9	No Detecto	No Detecto	No Detecto	No Detecto	No Detecto
10	No Detecto	No Detecto	No Detecto	No Detecto	No Detecto

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

Se considera el mismo criterio utilizado en el análisis de la prueba de la distancia de detección de la presencia de canes en ambientes controlados, donde se coloca el valor de 1 en los casos donde el dispositivo detecto y se representó con la palabra “Detecto”, mientras que se coloca 0 en los casos donde no se detectó la presencia del can y se representó con la palabra “No Detecto”.

Se realiza el análisis de tendencia central y dispersión mediante los datos interpretados en la Tabla 5-10, que correlaciona la distancia de detección y el número de pruebas efectuadas en un ambiente no controlado al dispositivo detector de canes. En el caso del promedio indica que el dispositivo logra una detección del 100% a distancias que oscilan entre 1 y 5 metros. A una distancia de 6 metros, la tasa de detección disminuye a un 60%, mientras que, para distancias entre los 7 y 10 metros, la detección se reduce al 0%. Esta tendencia sugiere que el DDC tiene una alta eficiencia en distancias cercanas.

En el caso de la desviación estándar y la varianza, presentes solo a 6 y 7 metros con 0,55 y 0,45 respectivamente, indican una variabilidad en los resultados en esas distancias específicas.

Tabla 5-10. Representación y análisis de los datos recolectados en la detección de canes en ambientes no controlados.

Distancia (metros)	Prueba1	Prueba 2	Prueba 3	Prueba 4	Prueba 5	Promedio en %	Desviación	Varianza
1	1	1	1	1	1	100%	0	0
2	1	1	1	1	1	100%	0	0
3	1	1	1	1	1	100%	0	0
4	1	1	1	1	1	100%	0	0
5	1	1	1	1	1	100%	0	0
6	1	0	0	1	1	60%	0,55	0,49
7	1	0	0	0	0	20%	0,45	0,40
8	0	0	0	0	0	0%	0	0
9	0	0	0	0	0	0%	0	0
10	0	0	0	0	0	0%	0	0

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

La Ilustración 5-6, representa los resultados obtenidos en el análisis al DDC realizado en un ambiente no controlado. En el eje Y se encuentra el promedio de las cinco pruebas efectuadas, mientras que en el eje X se representa la distancia a la que se llevó a cabo estas pruebas. La grafica de dispersión proporciona una representación visual de cómo el dispositivo se comporta en un entorno no controlado. Evidenciando la eficiencia total hasta los 5 metros de distancia, a partir de ahí el dispositivo presenta dificultades en la detección en el caso de los 6 y 7 metros de distancia, a partir de los 7 metros el dispositivo deja de detectar totalmente la presencia del can.

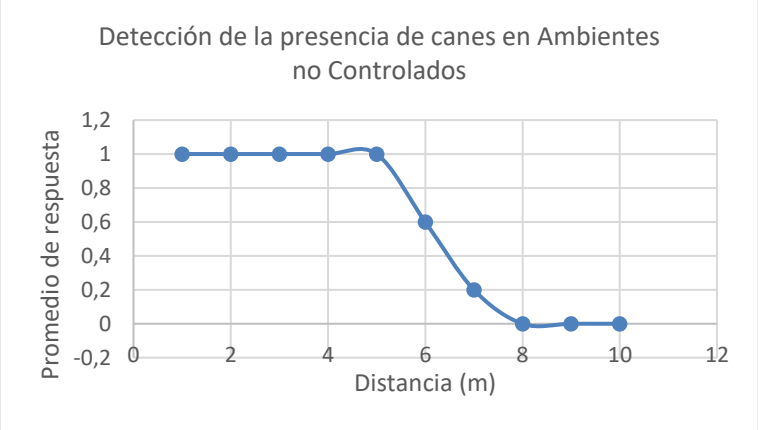


Ilustración 5-6. Gráfico Scatter de los resultados obtenidos en la detección de canes en Ambientes no Controlados.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023

5.2.2.2 Tiempo de Detección a la distancia mínima en 3 jornadas

Acorde a lo mencionado en el apartado anterior el tiempo tiene un papel relevante al momento de considerar a un dispositivo detector de canes como un sistema embebido apto para ponerlo en funcionamiento en ambientes no controlados. La Tabla 5-11, presenta los resultados de 20 pruebas realizadas en un ambiente no controlado utilizando el dispositivo detector de canes en tres diferentes jornadas: 07:00 am, 13:00 pm, 19:00 pm. Los datos recopilados muestran como el tiempo de detección varia en las tres jornadas especificadas, estos resultados son de especial interés para comprender como la iluminación natural cambia a lo largo del día y como esto puede afectar la capacidad del dispositivo para detectar la presencia de canes.

Tabla 5-11. Tiempo de respuesta del DDC en 3 jornadas a una distancia mínima en un ambiente no controlado.

Numero de Prueba	**Tiempo de detección a las 07:00 AM**	**Tiempo de detección a las 13:00 PM**	**Tiempo de detección a las 19:00 PM**
1	2.80 seg	3.48 seg	4.95 seg
2	2.5 seg	4.56 seg	3.87 seg
3	2.11 seg	4.98 seg	4.56 seg
4	3.54 seg	3.56 seg	2.98 seg
5	4.25 seg	2.97 seg	3.57 seg
6	4.37 seg	3.47 seg	4.52 seg
7	2.86 seg	4.25 seg	4.61 seg
8	3.84 seg	4.89 seg	4.84 seg
9	2.39 seg	4.71 seg	3.99 seg
10	4.18 seg	3.65 seg	3.82 seg
11	2.45 seg	3.14 seg	4.71 seg
12	2.93 seg	3.59 seg	4.26 seg
13	2.75 seg	4.06 seg	4.58 seg
14	2.35 seg	4.85 seg	3.56 seg
15	3.46 seg	2.76 seg	3.29 seg
16	3.97 seg	3.68 seg	4.13 seg
17	3.68 seg	3.92 seg	4.75 seg
18	4.08 seg	4.51 seg	3.64 seg
19	2.58 seg	4.33 seg	4.09 seg
20	3.77 seg	3.21 seg	3.57 seg

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

Mediante la Tabla 5-12 donde se analizaron tres jornadas diferentes (mañana -mediodía - noche). Se observó que, durante la mañana, con un promedio de 3.24 segundos y una desviación estándar de 0.74 segundos, el dispositivo exhibe un tiempo de detección más optimo en comparación con el mediodía (promedio de 3.93 segundos y una desviación estándar de 0.68 segundos) y la noche (promedio de 4.11 segundos y una desviación estándar de 0.56 segundos). Estos datos sugieren que la jornada matutina presenta condiciones ambientales más favorables para la detección precisa y rápida de canes.

Tabla 5-12. Descripción de la media y desviación estándar del tiempo de detección a 3 metros en 3 jornadas en un ambiente no controlado.

	Jornada 07:00 am	**Jornada 13:00 pm**	**Jornada 19:00 pm**
Media	3.24 seg	3.93 seg	4.11 seg
Desviación Estándar	0.71 seg	0.68 seg	0.56 seg

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

Se llevo a cabo el análisis de la normalidad mediante la prueba de Shapiro-Wilk en un conjunto de datos que representa el tiempo de detección de un dispositivo a una distancia de tres metros, realizado en tres jornadas distintas (07:00 am, 13:00 pm y 19:00 pm) en ambientes no controlados. En la Tabla 5-13, se observa que en la jornada de las 07:00 am, el valor estadístico obtenido es de 0.91, con un p-valor de 0.085 y un nivel de significancia (Alpha) de 0.05, lo que indica que se verifica la normalidad en los datos. En la jornada de las 13:00 pm, el valor estadístico fue de 0.95 con un p-valor de 0.42 y en la última jornada a las 19:00 pm, se obtuvo un valor estadístico de 0.94 con un p-valor de 0.31. En ambos casos, con un nivel significancia de 0.05, también se confirma la normalidad en los datos. Estos resultados sugieren que los datos siguen una distribución aproximadamente normal en todas las jornadas y permiten fundamentar el siguiente análisis que se realiza con los conjuntos de datos que se tiene en la Tabla 5-11.

Tabla 5-13. Prueba de Shapiro-Wilk para el tiempo de detección a 3 metros obtenidos en 3 jornadas en una en ambientes no controlados.

Estadístico	0.91	0.95	0.94
p-valor	0.085	0.42	0.31
Alpha	0.05	0.05	0.05
Normal	Si	Si	Si

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

5.2.2.3 Tiempo de Detección a la distancia máxima en 3 jornadas

La Tabla 5-14, presenta los tiempos de detección registrados en tres jornadas distintas (07:00 AM, 13:00 PM y 19:00 PM) a una distancia constante de 5.5 metros mediante el dispositivo detector de canes. Los valores reflejan el tiempo en segundos que el dispositivo tardó en detectar la presencia de los canes en cada jornada. Los datos recopilados revelan variaciones en los tiempos de detección en las diferentes jornadas, lo que sugiere posibles influencias de factores ambientales o de iluminación en el proceso de detección.

Tabla 5-14. Tiempo de respuesta por parte del DDC durante la mañana a una distancia máxima.

Numero de Prueba	Tiempo de detección a las 07:00 AM	Tiempo de detección a las 13:00 PM	Tiempo de detección a las 19:00 PM
1	8.28 seg	9.14 seg	9.78 seg
2	8.15 seg	8.76 seg	8.94 seg
3	7.96 seg	8.91 seg	9.54 seg
4	7.52 seg	9.45 seg	8.75 seg
5	7.56 seg	7.93 seg	8.61 seg
6	7.84 seg	8.64 seg	9.05 seg
7	8.14 seg	8.72 seg	9.65 seg
8	6.98 seg	7.54 seg	9.47 seg
9	7.63 seg	7.48 seg	8.87 seg
10	6.54 seg	9.15 seg	9.63 seg
11	8.41 seg	9.06 seg	9.12 seg
12	8.24 seg	8.64 seg	9.85 seg
13	8.34 seg	8.71 seg	8.84 seg
14	8.61 seg	8.69 seg	9.65 seg
15	6.91 seg	9.05 seg	9.35 seg
16	6.82 seg	8.87 seg	9.64 seg
17	7.14 seg	7.46 seg	9.71 seg
18	7.28 seg	7.58 seg	9.07 seg
19	8.04 seg	9.07 seg	8.56 seg
20	8.63 seg	8.19 seg	8.73 seg

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

La Tabla 5-15, evidencia los resultados estadísticos en tres franjas horarias (07:00 am– 13:00 pm – 19:00 pm). Los datos arrojaron resultados consistentes, mostrando notables disparidades en los tiempos de respuesta del prototipo. En la mañana, con un promedio de 7.75 segundos y una desviación estándar de 0.63 segundos, se alcanzó el tiempo de detección más eficiente en comparación con las mediciones al mediodía (promedio de 8.55 segundos y una desviación de 0.63 segundos) y en la noche (promedio de 9.24 segundos y una desviación estándar de 0.43 segundos). Estos indicativos, además de señalar la influencia de las condiciones ambientales favorables en la mañana, resaltan un incremento en el tiempo de detección a medida que se aumenta la distancia.

Tabla 5-15. Descripción de la media y desviación estándar del tiempo de detección a 5.5 metros en 3 jornadas en un ambiente no controlado.

	Jornada 07:00 am	**Jornada 13:00 pm**	**Jornada 19:00 pm**
Media	7.75 seg	8.55 seg	9.24 seg
Desviación Estándar	0.63 seg	0.63 seg	0.43 seg

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

Se realizó la prueba de Shapiro-Wilk en un conjunto de datos que representa el tiempo de detección del DDC a una distancia de 5.5 metros en tres jornadas distintas en un ambiente no controlado y se obtuvieron los resultados evidenciados en la *Tabla 5-16*. Para la primera jornada, el estadístico fue de 0.95 y el p-valor fue 0.284, comparados con un nivel de significancia (alpha) de 0.05. Estos valores indican que los datos sí siguen una distribución normal. En la segunda jornada, el estadístico fue 0.87, el p-valor fue 0.012 y el nivel de significancia se mantuvo en 0.05. En este caso, los datos no siguen una distribución normal. Para la tercera jornada, el estadístico fue 0.91, el p-valor fue 0.07 y nuevamente se consideró un nivel de significancia de 0.05. Los datos en esta tercera prueba siguen una distribución normal.

Tabla 5-16. Prueba de Shapiro-Wilk para el tiempo de detección a 5.5 metros obtenidos en 3 jornadas en una en ambientes no controlados.

Estadístico	0.95	0.87	0.91
p-valor	0.284	0.012	0.07
Alpha	0.05	0.05	0.05
Normal	Si	No	Si

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

5.2.2.4 Diferenciación Persona-Can

En el contexto de un ambiente no controlado, la Tabla 5-17, muestra claramente la capacidad del dispositivo detector de canes para distinguir entre una persona y un can en función de la distancia de detección. En distancias comprendidas entre los 3 y 5.5 metros, el dispositivo demuestra una discriminación efectiva, identificando correctamente tanto a la persona como al can, lo que resulta en la activación exitosa de las alarmas de presencia de canes. Sin embargo, en distancias de 6 y 7 metros, la capacidad de diferenciación parece disminuir, ya que no se logra una discriminación clara entre los dos objetos. Esto se refleja en la falta de activación de la alarma en estas distancias.

Tabla 5-17. Diferenciación Persona-can del Sistema Embebido en un Ambiente no Controlado.

Distancia (metros)	Detección Can	Detección Persona	Observación
3	X		Detecto el can y activo las alarmas
3.5	X		Detecto el can y activo las alarmas
4	X		Detecto el can y activo las alarmas
5	X		Detecto el can y activo las alarmas
5.5	X		Detecto el can y activo las alarmas
6	X	X	Detecto la presencia de los dos objetos.
7	-	-	No detecto la presencia de ninguno

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

5.2.2.5 Detección varios canes

La Tabla 5-18, presenta un análisis detallado de cuatro casos, cada uno examinando el funcionamiento del DDC en una jornada especifica 07:00 am, dentro de un entorno no controlado. En cada caso, se evaluó la respuesta del dispositivo ante la presencia simultanea de dos canes ubicados a diferentes distancias. Los resultados consistentemente indican que el dispositivo demuestra una habilidad destacada para detectar el can que se encuentra más cercano en todas las instancias.

Tabla 5-18. Descripción de 4 casos de detección de varios canes a diferente distancia en un ambiente no controlado.

Canes	3 metros	3.5 metros	4 metros	5 metros	5.5 metros
Can 1	X				
Can 2			X		
Can 1			X		
Can 2	X				
Can 1			X		
Can 2					X
Can 1					X
Can 2		X			

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

5.3 Verificaciones Estadísticas de los Tiempos de Respuesta

Se ha seleccionado como técnica de verificación estadística la prueba de Tukey, esta metodología es aplicada para determinar si existe alguna variación significativa en el tiempo de detección entre las jornadas mencionadas. En particular, el método de Tukey va permitir realizar múltiples comparaciones entre las medias de las tres jornadas mientras controla el error tipo I, evitando así la posibilidad de obtener falsos.

Al identificar las diferencias significativas de tiempo de detección en diferentes jornadas mediante las pruebas de igualdad de varianza, el método de Tukey proporciona información valiosa sobre cuando el dispositivo detector de la presencia de canes muestra un mejor o peor rendimiento en términos de tiempo de detección en ambientes no controlados debido a las comparaciones múltiples entre los conjuntos de datos y presentando las diferencias significativas entre sí.

Para mayor precisión en la obtención de los resultados se hace uso del software estadístico Minitab, el mismo que proporciona herramientas para realizar todo tipo de análisis estadístico.

5.3.1 Verificación estadística del tiempo de respuesta a 3 metros en 3 jornadas en un ambiente no controlado.

Validación de las varianzas

- **Hipótesis nula** Todas las varianzas son iguales
- **Hipótesis alterna** Por lo menos una varianza o media es diferente
- **Nivel de significancia** α=0.05

En la Tabla 5-19, se muestran los resultados obtenidos mediante el software Minitab usando intervalo de confianza de Bonferroni del 95%, de donde se observa que:

- En la Jornada 1 (07:00 am), se puede estar un 98.33% seguro que la desviación estándar está dentro del intervalo de confianza (0.613397; 1.02001).

- En la Jornada 2 (13:00 pm), se puede estar un 98.33% seguro que la desviación estándar está dentro del intervalo de confianza (0.530796; 0.98357).

- En la Jornada 3 (19:00 pm), se puede estar un 98.33% seguro que la desviación estándar está dentro del intervalo de confianza (0.429041; 0.84789).

Tabla 5-19. Intervalos de confianza para la desviación estándar de las 3 jornadas a una distancia de 3 metros.

Muestra	N	Desv.Est.	IC
Jornada 1 (07:00 am)	20	0,742	(0,613397; 1,02001)
Jornada 2 (13:00 pm)	20	0,67	(0,530796; 0,98357)
Jornada 3 (19:00 pm)	20	0,56	(0,429041; 0,84789)
Nivel de confianza individual=98.33% Nivel de confianza de la población=95%			

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

En la Tabla 5-20, se aprecia dos valores p de la prueba de igualdad de varianzas que son: el valor p del método de comparaciones múltiples y el valor p del método de Levene, este último valor p es el que debemos utilizarlo para nuestro análisis.

El valor p del método de Levene, evalúa si las varianzas entre los grupos son estadísticamente iguales o diferentes mediante la siguiente ecuación:

Si:

$$\text{valor p} > \alpha; \qquad \text{las varianzas son iguales}$$

De no cumplirse esta sentencia, se define que las varianzas son desiguales.

Tabla 5-20. valor p para los conjuntos de datos del tiempo de detección en 3 jornadas a una distancia de 3 metros.

Método	**Estadística de prueba**	**Valor p**
Comparaciones múltiples	—	0,206
Levene	1,79	0,177

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

Reemplazando el valor p de Levene obtenido en la Tabla 5-20, en la ecuación descrita anteriormente tenemos:

$$\text{valor p} > \alpha$$

$$0.177 > 0.05$$

Entonces se concluye que el valor p de Leve si cumple la condición de ser mayor que el nivel de significancia por ende **se admite la Hipótesis nula, determinando que las varianzas en los tiempos de detección en las tres jornadas a 3 metros de distancia son estadísticamente iguales.**

Validación de las medias

- **Hipótesis nula** Todas las medias son iguales
- **Hipótesis alterna** Por lo menos una media es diferente
- **Nivel de significancia** α=0.05

Al verificar que las varianzas en las tres jornadas son estadísticamente iguales, se puede continuar con el cálculo de la prueba de las diferencias significativas de las medias de las tres jornadas. Para ello es necesario trabajar únicamente con el valor p de la Tabla 5-21 y realizar la siguiente comparación:

Si

$$\text{valor p} \leq \alpha; \qquad \text{las diferencias entre las medias son estadisticamente significativas}$$

$$\text{valor p} > \alpha; \qquad \text{las diferencias entre las medias no son estadisticamente significativas}$$

Tabla 5-21. Análisis de varianza en la Prueba de diferencias significativas para las 3 jornadas a 3 metros de distancia.

Fuente	GL	SC Ajust.	MC Ajust.	Valor F	Valor p
Factor	2	8,427	4,2134	9,50	0,000
Error	57	25,281	0,4435		
Total	59	33,708			

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

Por lo tanto, reemplazando nuestro valor p en la condición anterior, tenemos:

$$\text{valor p} \leq \alpha$$

$$0.000 \leq 0.05$$

De acuerdo con la condición anterior, se rechaza la hipótesis nula y se acepta la hipótesis alterna, la misma que indica que **no todas las medias de los tiempos de detección en tres jornadas a 3 metros de distancia son iguales.**

Verificación de diferencias significativas

Finalmente, en la Tabla 5-22, se obtiene los resultados absolutos de la prueba de Tukey donde se conoce que las medias que no comparten una letra son significativamente diferentes. De esta manera se encuentra que:

- La jornada 2 (13:00 pm) y jornada 3(19:00) no presentan diferencias significativas en los tiempos de detección.

- La jornada 1(07:00 am) muestra diferencias significativas con las jornadas 2 y 3 (13:00 pm y 19:00 pm), respectivamente.

Tabla 5-22. Prueba de Tukey para las 3 jornadas a 3 metros de distancia.

Factor	N	Media	Agrupación	
Jornada 3 (19:00 pm)	20	4,114	A	
Jornada 2 (13:00 pm)	20	3,928	A	
Jornada 1 (07:00 am)	20	3,243		B

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

5.3.2 Verificación estadística del tiempo de respuesta a 5.5 metros en 3 jornadas en un ambiente no controlado.

Se repite el análisis realizado en el tiempo de respuesta a 3 metros de distancia en 3 jornadas en un ambiente controlado. El mismo que empieza elaborando la prueba de igualdad de varianza que debe cumplir la siguiente condición:

Validación de las varianzas

- **Hipótesis nula** Todas las varianzas son iguales
- **Hipótesis alterna** Por lo menos una varianza es diferente
- **Nivel de significancia** α=0.5

Para llegar a admitir una de las hipótesis, primero debemos estimar la desviación estándar de cada tiempo de detección en las tres jornadas diferentes y posterior a ello determinar si las varianzas son estadísticamente iguales.

En el caso de la desviación estándar la Tabla 5-23, muestra los valores asignados a cada tiempo de detección en las 3 jornada obtenidos en el software estadístico Minitab, los mismos que se interpretan de la siguiente manera:

- En la Jornada 1 (07:00 am), se puede estar un 98.33% seguro que la desviación estándar está dentro del intervalo de confianza (0.479447; 0.946614)

- En la Jornada 2 (13:00 pm), se puede estar un 98.33% seguro que la desviación estándar está dentro del intervalo de confianza (0.449892; 0.993110)

- En la Jornada 3 (19:00 pm), se puede estar un 98.33% seguro que la desviación estándar está dentro del intervalo de confianza (0.353514; 594254)

Tabla 5-23. Intervalos de confianza para la desviación estándar de las 3 jornadas a una distancia de 5.5 metros.

Muestra	N	Desv.Est.	IC
Jornada 1 (07:00 am)	20	0,632080	(0,479447; 0,946614)
Jornada 2 (13:00 pm)	20	0,627146	(0,449892; 0,993110)
Jornada 3 (19:00 pm)	20	0,430036	(0,353514; 0,594254)
Nivel de confianza individual=98.33%			
Nivel de confianza de la población=95%			

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

Si:

$$\text{valor p} > \alpha; \qquad \text{las varianzas son iguales}$$

De no cumplirse esta sentencia, se define que las varianzas son desiguales.

$$\text{valor p} > \alpha$$

$$0.421 > 0.05$$

Tabla 5-24. valor p para los conjuntos de datos del tiempo de detección en 3 jornadas a una distancia de 5.5 metros.

Método	Estadística de prueba	Valor p
Comparaciones múltiples	—	0,104
Levene	0,88	0,421

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

Entonces se concluye que el valor p de Levene si cumple la condición de ser mayor que el nivel de significancia por ende **se admite la Hipótesis nula donde se asume que las varianzas en los tiempos de detección en las tres jornadas a 5.5 metros de distancia son estadísticamente iguales.**

Validación de las medias

- **Hipótesis nula** Todas las medias son iguales
- **Hipótesis alterna** Por lo menos una media es diferente
- **Nivel de significancia** α=0.05

Partiendo del cumplimiento de la igualdad de varianzas en las tres jornadas, se procede a realizar la prueba de diferencias significativas en Minitab. Obteniendo como resultado la Tabla 5-25, que revela información sobre el análisis que se realiza a la varianza de los conjuntos de datos.

De esta tabla tomamos el valor p, el mismo que nos permite conocer si nuestros conjuntos manejan diferencias significativas entre sí por medio de la siguiente condición:

Si

valor $p \leq \alpha$; las diferencias entre las medias son estadisticamente significativas

valor $p > \alpha$; las diferencias entre las medias no son estadisticamente significativas

Reemplazando nuestro valor p en la condición anterior, tenemos:

$$\text{valor } p \leq \alpha$$

$$0.000 \leq 0.05$$

Tabla 5-25. Análisis de varianza en la prueba de diferencias significativas en las 3 jornadas a 5.5 metros de distancia.

Fuente	GL	SC Ajust.	MC Ajust.	Valor F	Valor p
Factor	2	22,23	11,1141	34,10	0,000
Error	57	18,58	0,3259		
Total	59	40,81			

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

De acuerdo a la condición anterior se descarta la hipótesis nula y se admite la hipótesis alterna que describe que **no todas las medias de los tiempos de detección en tres jornadas a 5.5 metros de distancia son iguales.**

Verificación de diferencias significativas

- La prueba de Tukey brinda una visión más definida mediante la La jornada 1(07:00 am) muestra diferencias significativas con las jornadas 2 (13:00 pm)
- La jornada 2 (13:00 pm) muestra diferencias significativas con la jornada 3 (19:00 pm).
- La jornada 3 (19:00 pm) muestra diferencias significativas con la jornada 1 (07:00 am).

Tabla 5-26, de cuáles son las diferencias que existen entre los tiempos de distancia en las tres jornadas. Describiendo esta información de la siguiente manera:

- La jornada 1(07:00 am) muestra diferencias significativas con las jornadas 2 (13:00 pm)
- La jornada 2 (13:00 pm) muestra diferencias significativas con la jornada 3 (19:00 pm).
- La jornada 3 (19:00 pm) muestra diferencias significativas con la jornada 1 (07:00 am).

Tabla 5-26. Prueba de Tukey para las 3 jornadas a 5.5 metros de distancia.

Factor	N	Media	Agrupación		
Jornada 3 (19:00 pm)	20	9,2405	A		
Jornada 2 (13:00 pm)	20	8,552		B	
Jornada 1 (07:00 am)	20	7,751			C

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

Los resultados obtenidos en esta sección hacen referencia al análisis estadístico de diferencias significativas presente en los tiempos de detección en tres jornadas diferentes a determinadas distancias en ambientes no controlados.

5.4 Pruebas Dinámicas de Experimentación sobre el S.E implementado

La implementación del dispositivo en un vehículo para realizar las pruebas dinámicas es fundamental al momento de validar el comportamiento del prototipo en situaciones reales y en movimiento. Las pruebas dinámicas proporcionan una oportunidad para verificar la sensibilidad con la que cuenta el dispositivo ante la presencia de un can. También se pueden presentar problemas o limitaciones que no se habrían detectado en las pruebas estáticas como es el movimiento inoportuno del can.

5.4.1 Entorno de Evaluación

5.4.1.1 Delimitación de la Zona de Pruebas y Vehículo Participante

Como parte de los objetivos específicos se mencionó un entorno de tránsito vehicular seguro para los sujetos participantes, siendo la zona periurbana la que cumple con esta mención. De manera específica se evalúa el dispositivo en la Ciudadela los Shyris delimitando entre las calles Cuenca e Ibarra, donde se estableció dos puntos de referencia distancia mínima (3 metros) y distancia máxima (5 metros) con respecto al vehículo en donde se situarán los sujetos participantes(canes).

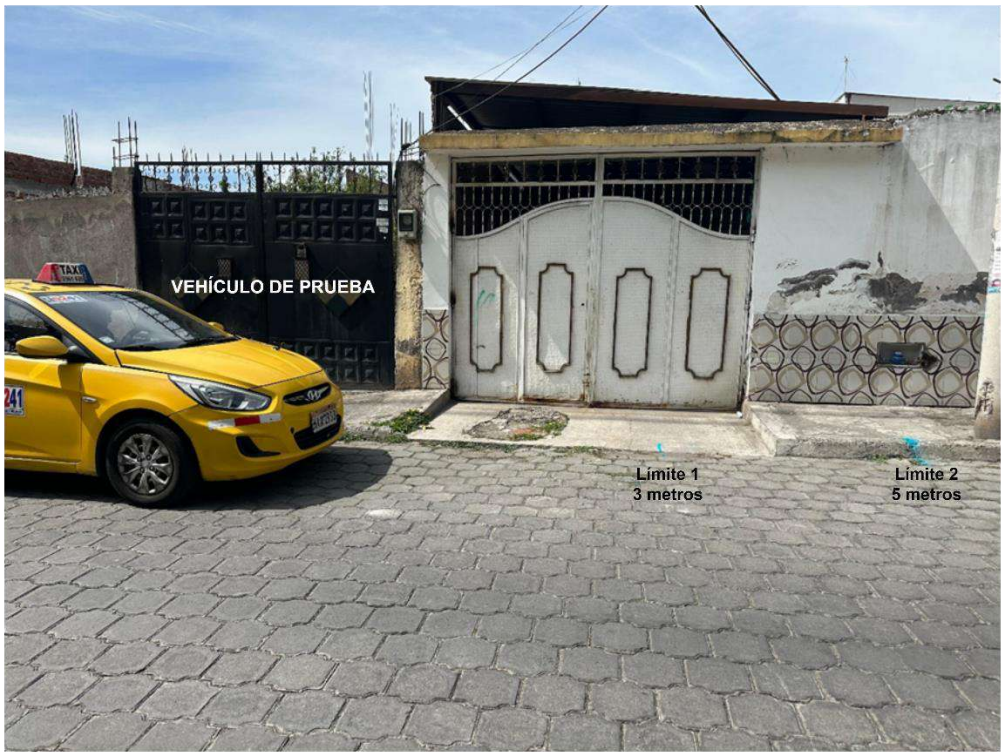


Ilustración 5-7. Descripción Gráfica del escenario de pruebas dinámicas del DDC.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

5.4.1.2 Sujetos de Prueba

A diferencia de las pruebas estáticas donde se consideró las siluetas de dos canes en tamaño real ahora evaluaremos el comportamiento del prototipo con la ayuda de dos canes, El Can 1 de raza Husky siberiano y el Can 2 de raza Golden, los cuales son de tamaño similar a las siluetas. Los canes estarán manipulados por su dueño generando un ambiente seguro para los canes y el vehículo participante.

Ilustración 5-8. Canes participantes en las pruebas dinámicas del DDC.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

5.4.1.3 Detección del DDC dentro del margen de referencia

Tras la ejecución de las pruebas dinámicas en la Tabla 5-27, se evidencia que, a las velocidades de 5, 10 y 15 km/h, el dispositivo demostró un rendimiento excepcional al detectar correctamente a los canes en todas las cinco pruebas realizadas a cada una de estas velocidades. A una velocidad de 20 km/h, el dispositivo mantuvo un alto nivel de precisión, detectando correctamente a los canes en tres de las cinco pruebas. Este resultado indica una ligera disminución en la tasa de detección en comparación con velocidades más bajas. A 25 km/h, el dispositivo mostró una capacidad de detección reducida, detectando correctamente a los canes en solo dos de las cinco pruebas realizadas. Esto sugiere una disminución significativa en la precisión a medida que la velocidad aumenta. A una velocidad de 30 km/h, el dispositivo presentó un rendimiento limitado, siendo capaz de detectar a los canes en solo una de las cinco pruebas realizadas. Esto indica una marcada disminución en la capacidad de detección a esta velocidad.

Finalmente, a una velocidad de 35 km/h, el dispositivo no logró detectar a ningún can en ninguna de las cinco pruebas realizadas. Esto indica que el dispositivo experimenta dificultades significativas en la detección de canes a esta velocidad.

Tabla 5-27. Detección del Dispositivo dentro del margen de referencia en el escenario de prueba.

Velocidad del vehículo	Prueba 1	Prueba 2	Prueba 3	Prueba 4	Prueba 5
5 km/h	Detecto	Detecto	Detecto	Detecto	Detecto
10 km/h	Detecto	Detecto	Detecto	Detecto	Detecto
15 km/h	Detecto	Detecto	Detecto	Detecto	Detecto
20 km/h	Detecto	Detecto	Detecto	No Detecto	Detecto
25 km/h	No Detecto	Detecto	Detecto	No Detecto	No Detecto
30 km/h	No Detecto	No Detecto	Detecto	No Detecto	No Detecto
35 km/h	No Detecto	No Detecto	No Detecto	No Detecto	No Detecto

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

Mediante la Tabla 5-28, se realiza una transformación de la información cualitativa a cuantitativa para realizar una interpretación adecuada de los datos

Tabla 5-28. Datos cualitativos obtenidos de las 5 pruebas en las diferentes velocidades.

Velocidad del vehículo	Prueba 1	Prueba 2	Prueba 3	Prueba 4	Prueba 5	Promedio
5 km/h	Detecto	Detecto	Detecto	Detecto	Detecto	100% - 1
10 km/h	Detecto	Detecto	Detecto	Detecto	Detecto	100% - 1
15 km/h	Detecto	Detecto	Detecto	Detecto	Detecto	100% - 1
20 km/h	Detecto	Detecto	Detecto	No Detecto	Detecto	60% - 0.6
25 km/h	No Detecto	Detecto	Detecto	No Detecto	No Detecto	40% - 0.4
30 km/h	No Detecto	No Detecto	Detecto	No Detecto	No Detecto	20% - 0.2
35 km/h	No Detecto	No Detecto	No Detecto	No Detecto	No Detecto	0% - 0

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

La Ilustración 5-9, evidencia de manera grafica el comportamiento que tiene el dispositivo detector de canes implementado en un vehículo de manera dinámica en diversas velocidades. Es claro que la efectividad del dispositivo depende de la distancia en la que se encuentra del can mientras más cerca se encuentre el dispositivo del can dentro del límite referencial su detección será optima.

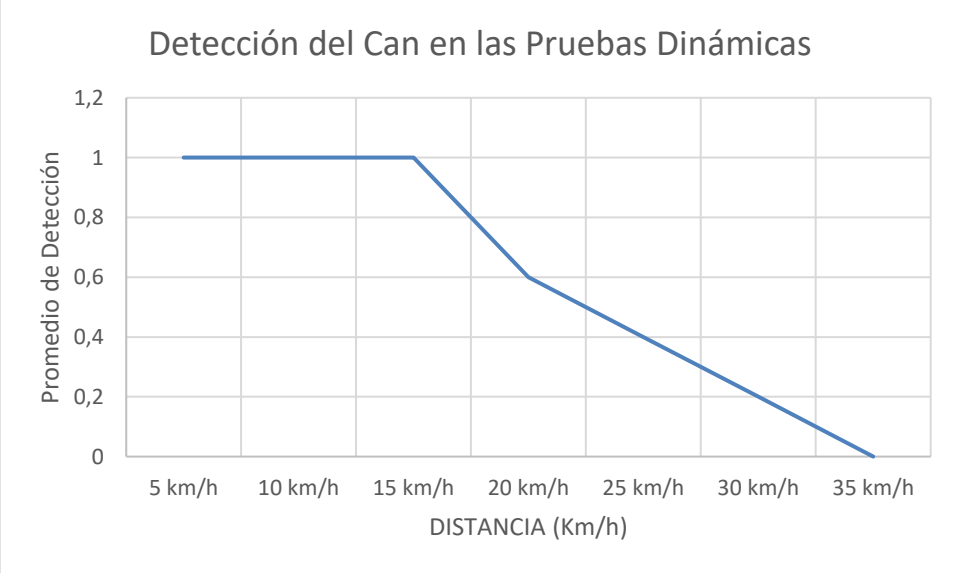


Ilustración 5-9. Gráfico Scatter de los resultados obtenidos en la detección de canes en pruebas dinámicas.

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

5.5 Análisis de costos para la fabricación del DDC

En la *Tabla 5-29.*, se detalla el costo total estimado para la construcción del dispositivo detector de canes. En esta tabla, se incluyen los costos unitarios asociados a cada elemento esencial para la creación del dispositivo, junto a la cantidad requerida de dichos elementos. Los valores presentados reflejan los costos de adquisición en el contexto local de Ecuador.

El desglose de costos presentado en la *Tabla 5-29.*, no solo nos sirve de guía para el análisis económico, sino también proporciona una visión general de cómo se distribuirán los recursos en la implementación del proyecto.

Tabla 5-29. Costos de fabricación del Dispositivo Detector de Canes.

Cantidad	Elemento	Costo unitario (USD)	Costo total (USD)
1	FPGA PYNQ Z1	$ 300,00	$ 300,00
1	Cámara	$ 20,00	$ 20,00
1	Motor vibrador 5-12V	$ 1,50	$ 1,50
2	Buzzer 5V	$ 0,80	$ 1,60
3	Diodos LEDs	$ 0,15	$ 0,45
1	Impresión 3D de la carcasa del DDC	$ 20,00	$ 20,00
1	Corte laser de la tapa de la carcasa	$ 3,00	$ 3,00
1	Batería 6V-4Ah	$ 15,00	$ 15,00
1	Otros elementos	$ 10,00	$ 10,00
Total		$ 370,45	$ 371,55
Mano de obra		$	50,00
Costo total del DDC		$	421,55

Realizado por: Domínguez Carlos, Lliguin Giustinne, 2023.

Tomando como referencia la *Tabla 5-29.*, se establece que el costo total de fabricación del DDC es de $421,55. De tal manera, este valor se desglosa acorde a los siguientes porcentajes de costos de fabricación: siendo el 88.13% en costos de materiales necesarios para implementación del dispositivo, y el 11.87% en la inversión del tiempo y esfuerzo empleado en mano de obra en la creación y fabricación del dispositivo DDC.

CONCLUSIONES

- Se desarrolló un prototipo de sistema embebido basado en FPGA y Visión Artificial implementado en un vehículo para detectar y alertar sobre la presencia de canes en la vía. El prototipo fundamenta su funcionalidad en el módulo de adquisición de datos (Cámara), procesador de la información (FPGA), selector de modos de operación (Botones y Led´s) y un módulo de actuadores (Buzzer y Vibrador). La alimentación del prototipo es acondicionada desde una batería para alimentación externa.
- Para el tratamiento de las imágenes y detección de canes se utilizó una Red Neuronal Convolucional (CNN), la misma que después de 30 épocas de entrenamiento y validación presentó una pérdida de datos o de detección cercano al 0.001%, lo que implica que se ha logrado un nivel de precisión aceptable en la detección de los canes.
- Durante las pruebas de campo realizadas en ambientes controlados, el dispositivo operó de forma exitosa en el rango de 3 a 5.5 metros con un promedio de 3.43 segundos y 7.76 segundos, respectivamente. Así mismo, en la diferenciación persona-can el desempeño del prototipo fue satisfactorio cumpliendo con la activación de las alarmas al detectar la presencia del can. En la detección simultanea de canes se evidenció una relación constante entre la proximidad del can al dispositivo, detectando con precisión al can que se encuentra en la distancia más cercana.
- Partiendo de los resultados de las pruebas en ambientes no controlados a 3 y 5.5 metros de distancia se evidenció un mejor tiempo de detección en la jornada 1 (07:00 am) atribuyendo a la presencia de mejores condiciones de iluminación natural que benefician al funcionamiento de la cámara. En el caso de la diferenciación persona-can, el prototipo funciona de manera satisfactorio y activa las alarmas con precisión, al igual que en la detección simultanea de canes.
- Por medio de los resultados obtenidos en las pruebas dinámicas de detección se considera que el rendimiento efectivo del prototipo es directamente proporcional a la velocidad con la que cuenta el vehículo de prueba hasta una distancia de 25 km/h, a partir de ahí la detección disminuye de manera notable llegando a una velocidad de 35 km/h donde el dispositivo denota una detección nula.
- Tras realizar el análisis económico necesario para la elaboración del prototipo Dispositivo Detector de Canes (DDC) se estableció un costo total de $421.55, sin embargo, al no existir un dispositivo con el cual pueda ser comparable, no es posible determinar una comparativa económica.

RECOMENDACIONES

- Basados en las pruebas dinámicas se considera necesario examinar otras plataformas hardware para optimizar el rendimiento del algoritmo de detección de canes en la vía para futuras investigaciones, permitiendo desarrollar una comparativa del tiempo de respuesta obtenido con la tarjeta de desarrollo empleada en este proyecto.

- Sería conveniente desarrollar en proyectos futuros la implementación de un código que habilite la inicialización automática del programa de detección en el prototipo. Esto eliminaría la necesidad de activar el sistema desde un ordenador antes de colocarlo al vehículo, lo que potenciaría la comodidad y la eficiencia en la utilización del prototipo.

- Se sugiere implementar un módulo de adquisición de datos (cámara) de mayor resolución y específicamente desarrollada para exteriores, lo que permitirá mejorar significativamente la calidad del video capturado y, por ende, brindaría al dispositivo una mejor capacidad de respuesta frente a la detección de la presencia del can en la vía.

- Se recomienda trabajar sobre un desarrollo más sintetizado y reducido de la CNN, específicamente orientado a la detección únicamente de canes, capaz de reducir el consumo de recurso computacionales de la plataforma hardware del sistema embebido y que sea capaz de lograr mejores tiempos de respuesta.

BIBLIOGRAFÍA

AGGARWAL, C.C., 2018. *Neural Networks and Deep Learning*. New York: Springer. vol. 1.

ALVEAR PUERTAS, V., ROSERO MONTALVO, P., PELUFFO ORDÓÑEZ, D. y PIJAL ROJAS, J., 2017. Internet de las Cosas y Visión Artificial, Funcionamiento y Aplicaciones: Revisión de Literatura. *Enfoque UTE*,

BANZI, M., 2011. *Getting Started with Arduino, 2nd Edition - O'Reilly Media*. S.l.: s.n.

BISHOP, C.M., 2006. *Pattern recognition and machine learning Solutions to exercis*. S.l.: s.n. vol. 4.

BORRACCI, R. a y RUBIO, M., 2003. Aplicabilidad de redes neuronales artificiales para la predicción de los resultados individuales de la cirugía cardíaca. Estudio preliminar. *Rev Argent Cardiol*, vol. 71,

CABRERA, E. y DÍAS, E., 2020. *Manual de uso de Jupyter notebook para aplicaciones docentes*. Madrid: Universidad Complutense de Madrid.

CAYSSIALS, R., 2014. *Sistemas Embebidos en FPGA*. Alpha Editions. Argentina: AlfaOmega Grupo Editor Argentino S.A. vol. 1.

COIP, 2021. Código Orgánico Integral Penal. *Registro Oficial - Ógano del Gobierno del Ecuador*,

CORONA, L.G., ABARCA, G.S. y MARES, J., 2014. Sensores y actuadores aplicaciones con Arduino. *Publicacion En Internet*, no. October, ISSN 2198-6452.

COSOI P., E., 2002. Cómo elegir una cámara digital. *Revista chilena de pediatría*, vol. 73, no. 5, DOI 10.4067/s0370-41062002000500014.

CUEVAS, E., ZALDIVAR, D. y PEREZ, M., 2012. Procesamiento digital de imágenes con MatLAB y SIMULINK. *Revista S&T 10(21)*,

DAVID, A. y PÉREZ, A., 2009. Sistemas Embebidos y Sistemas Operativos Embebidos. *CICORE*, ISSN 1316-6239.

DÉLEG, M., 2017. Tecnología LED. *Universidad Politécnica Salesiana*, no. 2,

DOMINGUEZ, L.D., PEREZ, A.J., RUBIALES, A.J., D'AMATO, J.P. y BARBUZZA, R., 2016. Herramientas para la detección y seguimiento de personas a partir de cámaras de seguridad. *XXII Congreso Argentino de Ciencias de la Computación (CACIC 2016)*,

DORF, R. y BISHOP, R., 2005. *Sistemas de Control Moderno*. PEARSON PRENTICE. Madrid: PEARSON EDUCACIÓN S.A. vol. 10. ISBN 8420544019.

DUNN, P.F., 2011. *Fundamentals of sensors for engineering and science*. S.l.: s.n.

EL COMERCIO, 2017. Atropellar animales en las vías puede generar sanción por maltrato. *https://www.elcomercio.com/narices-frias/atropellar-animales-ecuador-sanciones-maltratoanimal.html*.

ESQUEDA, J., 2002. Fundamentos de Procesamiento de Imágenes. *Conatec*.

EUGENIA BAHIT, 2013. Módulos de sistema. *https://uniwebsidad.com*.

FEI, Y., ZHANG, H., WANG, Y., LIU, Z. y LIU, Y., 2022. LTPConstraint: a transfer learning based end-to-end method for RNA secondary structure prediction. *BMC Bioinformatics*, vol. 23, no. 1, ISSN 14712105. DOI 10.1186/s12859-022-04847-z.

GIANCARLO ZACCONE, Md.R.Karim., 2018. *Deep Learning with TensorFlow : Explore Neural Networks and Build Intelligent Systems with Python, 2nd Edition*. S.l.: s.n.

GOODFELLOW, I., BENGIO, Y. y COURVILLE, A., 2016. *Deep Learning*. London: Cambrigde. vol. 1. ISBN 0662035613.

HERNÁNDEZ BUSTOS, M.B. y FUENTES TERÁN, V.M., 2018. La Ley Orgánica de Bienestar Animal (LOBA) en Ecuador: análisis jurídico. *Derecho Animal. Forum of Animal Law Studies*, vol. 9, no. 3, DOI 10.5565/rev/da.328.

IEEE, 2009. *VHDL Language Reference Manual*. S.l.: s.n. vol. 2008.

JAYA, J.W., 2022. *Evaluación del uso de redes neuronales artificiales en la predicción de resultados de un proceso de isomerización para su uso como herramienta didáctica en la materia de simulación de procesos*. Trabajo de Titulación. Riobamba: Escuela Superior Politécnica de Chimborazo.

JUAN, R.Q. y CHACÓN, a, 2011. Redes neuronales artificiales para el procesamiento de imágenes, una revisión de la última década. *Riee&C*, vol. 9, no. 1, ISSN ISSN 1870 9532.

LEÓN OROZCO, I., 2023. Camara Web Pc HD 720P Webcam Windows MAC 720px reales. *https://www.mediaprice.com.ec/producto/camara-web-pc-hd-720p-1280x720-webcam/*.

MARTÍN DEL BRÍO, B., 1999. *Sistemas electrónicos basados en microprocesadores y microcontroladores*. S.l.: Prensas Universitarias de Zaragoza. ISBN 8477335168.

MATHWORKS, 2023. Transfer learning para entrenar modelos de Deep Learning. *https://la.mathworks.com/discovery/transfer-learning.html*.

MECAFENIX, 2018. *Que es el buzzer y cómo funciona (zumbador) - Ingeniería Mecafenix*. 2018. S.l.: s.n.

NOVATRONIC, 2020. Motor Vibración 5V. *https://novatronicec.com/index.php/product/motor-vibracion-5v/*.

PALACIOS, E., REMIRO, F. y LÓPEZ, L., 2014. *Microcontrolador PÍC16F84*. 3. España: RA-MA, S.A.

PANDEY, O.N., 2022. Switching Theory and Logic Design (STLD). *Electronics Engineering*. S.l.: s.n.,

REVISTA ESPECIFICAR, 2018. *Todo sobre los actuadores*. 2018. S.l.: s.n.

SALAS, S., 2015. *Todo sobre sistemas embebidos*. Diana Felix. Perú: Universidad Peruana de Ciencias Aplicadas S.A.C. vol. I.

SANABRIA, J.S. y ARCHILA, J.F., 2011. Detección y análisis de movimiento usando visión artificial. *Scientia et Technica Año XVI*,

STROUSTRUP, B., 2013. *the C++ Programming Language 4Th Edition*. S.l.: s.n.

SUAREZ, O., FERNÁNDEZ, M., VÁLLEZ, N., BUENO, G., PATÓN, J. y SALIDO, J., 2014. *OpenCV Essentials*. UK: Packt Publishing.

TAIPE, J., 2021. *Simulación y predicción por RNA de la composición de Tetrahidrofurano separado de una mezcla azeotrópica tetrahidrofurano - agua utilizando DWSIM*. Trabajo de titulación. Riobamba: Escuela Superior Politécnica de Chimborazo, Facultad de Ciencias.

TELLO, J.C., 2006. La visión artificial y las operaciones morfológicas en imágenes binarias. *Campus Multidisciplinar en Percepción e Inteligencia (CMPI)*,

WOODS, R., MCALLISTER, J., LIGHTBODY, G. y YI, Y., 2009. *FPGA-Based Implementation of Signal Processing Systems*. S.l.: s.n.

XILINX, 2023. Pynq-Z1: Python Productivity for Zynq. *Xilinx S.A.*

XILINX INC, 2021. Vivado Design Suite User Guide: Getting Started (UG910).

ZELLE, J., 2010. Python programming: an introduction to computer science. *Beedle & Associates, Inc*,

ZYPRIAN FLORIAN, 2023. Guía maestra OpenCV hecha para desarrolladores Python. *https://konfuzio.com*.

ANEXOS

ANEXO A: ALGORITMO DETECTOR DE CANES

DETECTOR DE CAN - SSD MOBILINET

Es necesario el uso de una cámara usb, el modelo y los archivos de configuración deben encontrarse en la misma carperta que este archivo.

Step 1: Cargar el overlay

In [1]:
```
from pynq.overlays.base import BaseOverlay
from pynq.lib.video import *
from pynq.buffer import PynqBuffer
base = BaseOverlay("base.bit")
print("iniciando")
```

```
iniciando
```

Paso 2: Iniciar HDMI salida

In [2]:
```
# monitor configuration: 640*480 @ 60Hz
#Mode = VideoMode(640,480,24,120)
hdmi_out = base.video.hdmi_out
Mode = VideoMode(640,480,24,120)
hdmi_out.configure(Mode,PIXEL_BGR)
hdmi_out.start()
print("HDMI habilidado")
```

```
HDMI habilidado
```

Paso 3: Configurar Cámara

In [3]:
```
import os
os.environ["OPENCV_LOG_LEVEL"]="SILENT"
import cv2

# Tamaño de imagen configurado
frame_in_w =640
frame_in_h =480

videoIn = cv2.VideoCapture(0)
videoIn.set(cv2.CAP_PROP_FRAME_WIDTH, frame_in_w);
videoIn.set(cv2.CAP_PROP_FRAME_HEIGHT, frame_in_h);
videoIn.set(cv2.CAP_PROP_BUFFERSIZE, 2) # por ejemplo, búfer de tamaño 2
print("capture device is open: " + str(videoIn.isOpened()))
print(videoIn)
print("cámara configurada")
```

```
capture device is open: True
<VideoCapture 0xadaa97a0>
cámara configurada
```

Paso 4: Visualizar video

In [4]:
```
import numpy as np
import cv2
import time
contador=0
num_frames = 100
readError=0
org = (50, 50)
start = time.time()
font = cv2.FONT_HERSHEY_SIMPLEX
fontScale = 1
# Blue color in BGR
color = (255, 0, 0)
# Line thickness of 2 px
thickness = 2

for i in range (num_frames):
    ret, frame_vga = videoIn.read()
    #frame_vga=cv2.cvtColor(frame_vga, cv2.COLOR_BGR2RGB)
    if (ret):
        contador+= 1
        cv2.putText(frame_vga, str(contador), org, font,fontScale, color, thickness, cv2.LINE_AA)
        outframe = hdmi_out.newframe()
        outframe[:] = frame_vga
        hdmi_out.writeframe(outframe)
    else:
        print("Error while reading from camera.")
        readError += 1

end = time.time()
print("si se ve")
print("Frames per second: " + str((num_frames-readError) / (end - start)))
print("Number of read errors: " + str(readError))
videoIn.release()
```

```
si se ve
Frames per second: 22.95543045126372
Number of read errors: 0
```

In [7]:
```
from pynq.lib.arduino import Arduino_IO
# Настройте контакт 13 как выход
button_pin = Arduino_IO(base.ARDUINO, 13, 'in')
button_value = button_pin.read()
for i in range(10):
    print(button_value)
```

In [ ]:

```
import cv2
import time
import os
import imutils
from pynq.lib.arduino import Arduino_IO
# Настройте контакт 13 как выход
led_pin = Arduino_IO(base.ARDUINO, 11, 'out') #buzzer
led_pin1 = Arduino_IO(base.ARDUINO, 12, 'out') #motor
led_pin2 = Arduino_IO(base.ARDUINO, 10, 'out') #Led inicio
led_pin3 = Arduino_IO(base.ARDUINO, 9, 'out')# Led modo1
led_pin4 = Arduino_IO(base.ARDUINO, 8, 'out')# Led modo2

led_pin2.write(0)# Proceso de inicio
led_pin3.write(1)# Proceso de inicio
led_pin4.write(1)# Proceso de inicio

CDE=0
CDS=0
BUZZER=False
detener=False
cont=0
detect=0
DETECTOR=False
pulsado=False

os.environ["OPENCV_LOG_LEVEL"] = "SILENT"
# font
font = cv2.FONT_HERSHEY_SIMPLEX
# org
org = (50, 50)
# fontScale
fontScale = 1
# Blue color in BGR
color = (255, 0, 0)
# Line thickness of 2 px
thickness = 2
videoIn = cv2.VideoCapture(0)
# camera (input) configuration 320x240
frame_in_w = 640
frame_in_h = 480
print("cámara configurada")
videoIn.set(cv2.CAP_PROP_FRAME_WIDTH, frame_in_w)
videoIn.set(cv2.CAP_PROP_FRAME_HEIGHT, frame_in_h)
videoIn.set(cv2.CAP_PROP_BUFFERSIZE, 1)  # por ejemplo, búfer de tamaño 2
print("capture device is open: " + str(videoIn.isOpened()))
print(videoIn)
```

```
readError = 0
contador = 0
dobj = 0
dobjpre = 0
cnombres = []
sonido = False
carchivo = "coco.names"

contador = 0
with open(carchivo, "rt") as f:
    cnombres = f.read().rstrip("\n").split("\n")

modelodir = "ssd_mobilenet_v3_large_coco_2020_01_14.pbtxt"
pesosdir = "frozen_inference_graph.pb"

net = cv2.dnn_DetectionModel(pesosdir, modelodir)
net.setInputSize(180, 180)
net.setInputScale(1.0 / 120.5)  # (1.0/ 120.5)
net.setInputMean((120.5, 120.5, 120.5))  # ((120.5, 120.5, 120.5))
net.setInputSwapRB(True)


def translate(objeto):
    # print(objeto)
    trasnlateaobj = []
    if objeto == "person":
        objeto = "Persona"
    if objeto == "dog":
        objeto = "Perro"
    else:
        objeto = objeto
    return objeto


def getobjetos(imagen, thresh, NMS, dibujar=True, objetos=[]):
    IDclase, confidencia, caja = net.detect(
        imagen, confThreshold=thresh, nmsThreshold=NMS)
    # print(IDclase,bbox)
    if len(objetos) == 0:
        objetos = cnombres
        # print( len(IDclase))
```

```
    objetoinfo = []
    listaobj = []
    listaid = []

    if len(IDclase) != 0:
        for classId, confidence, box in zip(IDclase.flatten(), confidencia.flatten(), caja):
            className = cnombres[classId - 1]
            if className in objetos:
                dobj = len(className)

                objetoinfo.append([box, className])
                trasnlateaobj = translate(className)
                listaobj.append(trasnlateaobj)
                listaid.append(classId)

                if (dibujar):

                    cv2.rectangle(imagen, box, color=(210, 100, 0), thickness=2)
                    cv2.rectangle(imagen, (box[0], box[1], box[2], 22), color=(210, 100, 0), thickness=-1)
                    cv2.putText(imagen, trasnlateaobj.upper(), (box[0]+10, box[1]+15),font, 0.4, (255, 255, 255), 1)
                    cv2.putText(imagen, str(round(confidence*100, 1)), (box[0]+100, box[1]+15),font, 0.4, (255, 255, 2

    return imagen, objetoinfo, listaobj, listaid


num_frames = 90
start = time.time()
font = cv2.FONT_HERSHEY_SIMPLEX
readError = 0
framecon=0
listaobj = []
#for i in range(num_frames):
estado1=False
contador2=0

button_pin = Arduino_IO(base.ARDUINO, 13, 'in')
button_value = button_pin.read()

while True:
        button_value = button_pin.read()

        if(base.buttons[1].read()==1):
            break


      if(button_value == 1):
          DETECTOR=False
          BUZZER=False
          CDS=0
          CDE=0
          led_pin.write(0)
          led_pin1.write(0)
          led_pin3.write(1)
          led_pin4.write(0)

      if(button_value == 0):
          DETECTOR=True
          led_pin3.write(0)
          led_pin4.write(1)


      ret, frame_vga = videoIn.read()
      #frame_vga=cv2.cvtColor(frame_vga, cv2.COLOR_BGR2RGB)


      if(ret):
          dim = (640, 480)
          frame_vga = cv2.resize(frame_vga, dim, interpolation = cv2.INTER_AREA)
          if(DETECTOR):
              cv2.putText(frame_vga, "Detector", (280, 32), font, 1, (210,200, 255), 2, cv2.LINE_AA)
          else:
              cv2.putText(frame_vga, "NORMAL", (280, 32), font, 1, (210,200, 255), 2, cv2.LINE_AA)
```

```
            if (ret):
                contador += 1
                framecon+=1
                if(framecon==2):
                    if(DETECTOR):
                        if(BUZZER):
                            led_pin.write(1)
                            led_pin1.write(1)
                        else:
                            led_pin.write(0)
                            led_pin1.write(0)
                        valores, objetoinfo, listaobj, listaid = getobjetos(frame_vga, 0.53, 0.1, dibujar=True, objeto
                    framecon=0
                #cv2.putText(frame_vga, str(contador), org, font,fontScale, color, thickness, cv2.LINE_AA)
                cv2.putText(frame_vga, str(listaobj), (7, 70), font, 0.6, (210,100, 1), 2, cv2.LINE_AA)

                if("Persona" in listaobj):
                    CDE+=1
                    if(CDE==3):
                        print("PERSONA DETECTADA")
                        BUZZER=True
                        CDS=0
                else:
                    CDS+=1

                    if(CDS==3):
                        print("SIN DETECTAR")
                        BUZZER=False
                        CDE=0



                # print(listaobj)
                outframe = hdmi_out.newframe()
                outframe[:] = frame_vga
                hdmi_out.writeframe(outframe)
            else:
                #print("Error de cámara")
                readError += 1
        else:
            print("no hay camara")
            break




end = time.time()

print("FPS: " + str((num_frames-readError) / (end - start)))
print("Errores: " + str(readError))
led_pin.write(0)# motor
led_pin1.write(0)# buzzer
led_pin2.write(1)# terminado
led_pin3.write(1)# modo1
led_pin4.write(1)# modo2

videoIn.release()
```

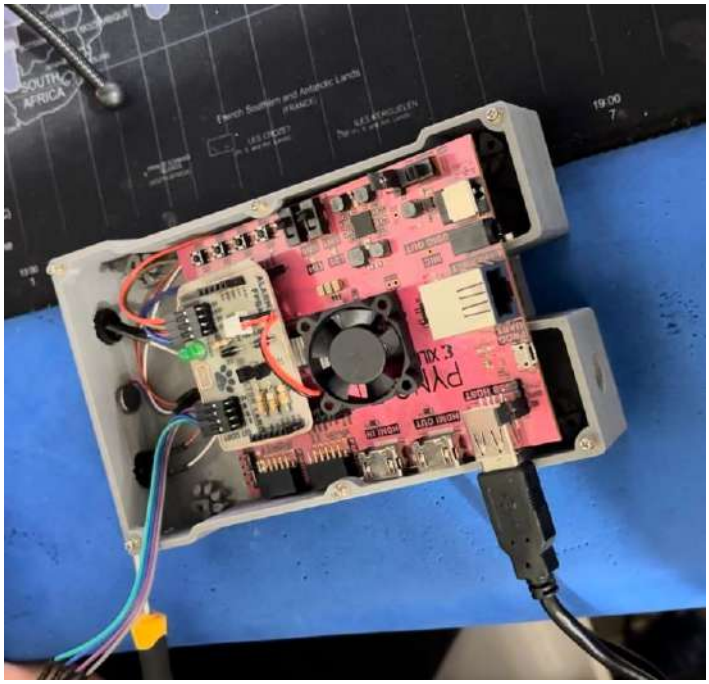
ANEXO B: TABLA DE VALORES CRÍTICOS PARA LA PRUEBA DE TUKEY

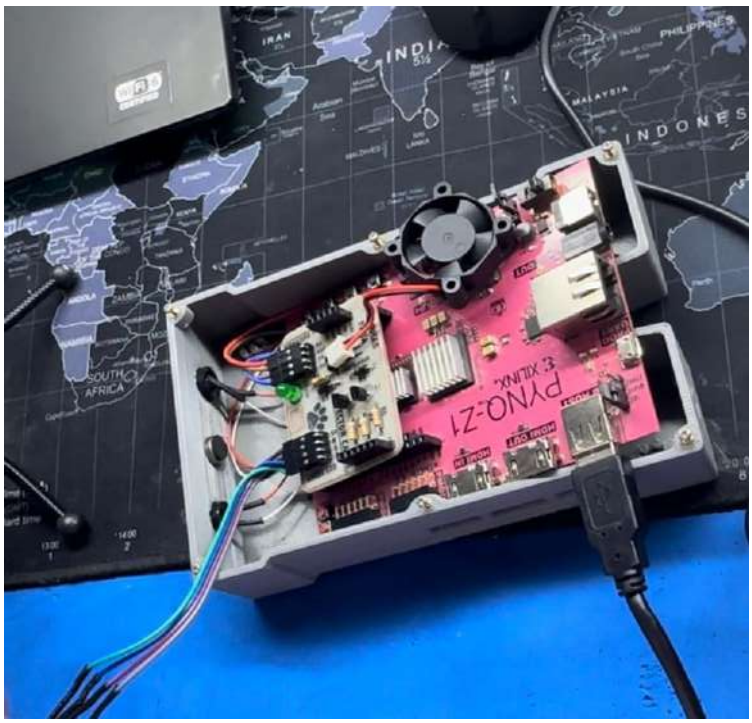
Valores críticos para la prueba de *Tukey*.

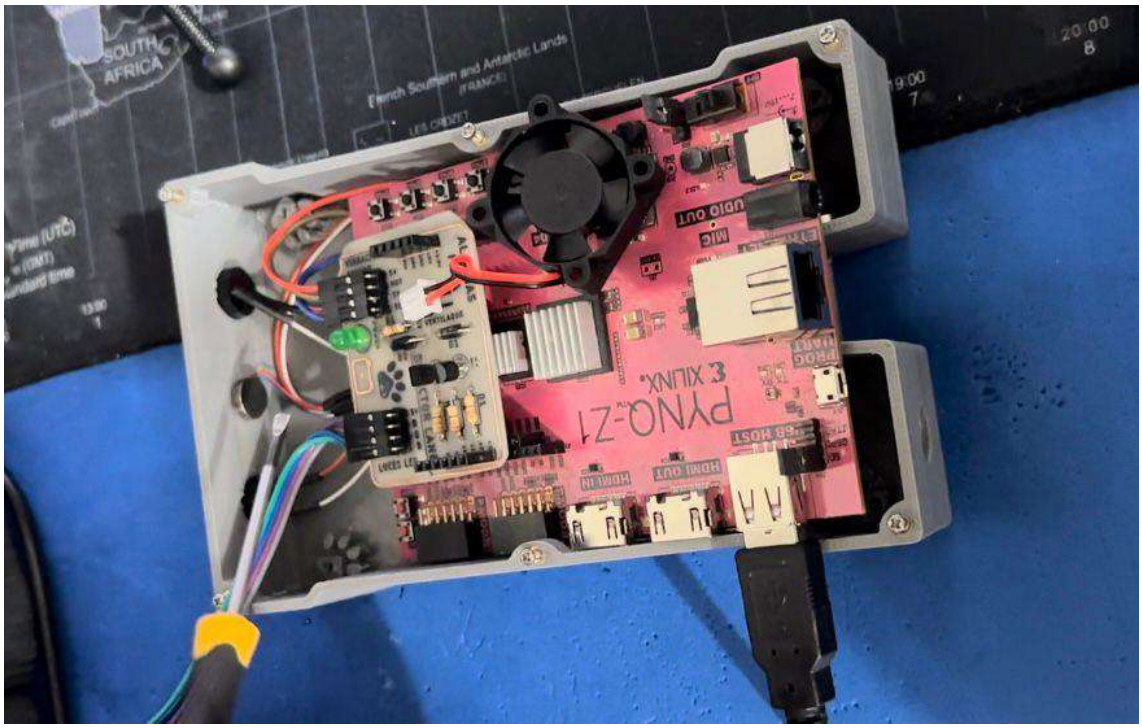
$q_{\alpha}(v_1, v_2)$

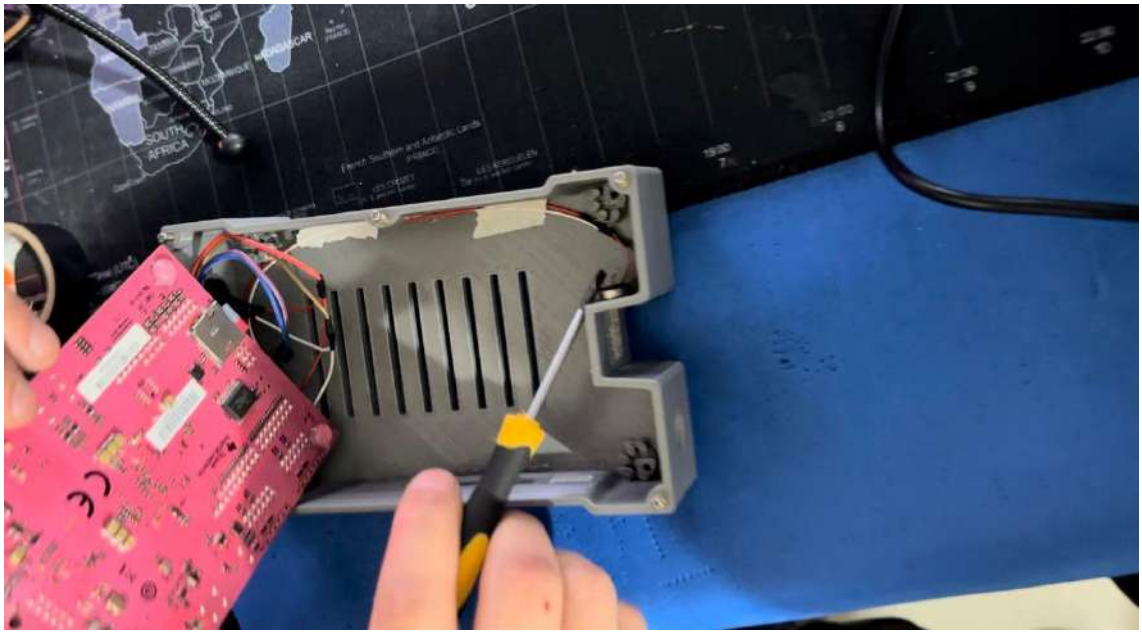
v_2 ↓	α ↓	v_1 2	3	4	5	6	7	8	9	10	11
1	**0.05**	18.00	29.98	32.82	37.08	40.41	43.12	45.40	47.36	49.07	50.59
	0.01	90.03	135.0	164.3	185.6	202.2	215.8	227.2	237.0	245.6	253.2
2	**0.05**	6.10	8.33	9.80	10.88	11.74	12.44	13.03	13.54	13.99	14.39
	0.01	14.04	19.02	22.29	24.72	26.63	28.20	29.53	30.68	31.69	32.59
3	**0.05**	4.50	5.91	6.82	7.50	8.04	8.48	8.85	9.18	9.46	9.72
	0.01	8.26	10.62	12.17	13.33	14.24	15.00	15.64	16.20	16.69	17.13
4	**0.05**	3.93	5.04	5.76	6.29	6.71	7.05	7.34	7.60	7.83	8.03
	0.01	6.51	8.12	9.17	9.96	10.58	11.10	11.55	11.93	12.27	12.57
5	**0.05**	3.64	4.60	5.22	5.67	6.03	6.33	6.58	6.80	6.99	7.17
	0.01	5.70	6.97	7.80	8.42	8.91	9.32	9.67	9.97	10.24	10.48
6	**0.05**	3.46	4.34	4.90	5.31	5.63	5.89	6.12	6.32	6.49	6.65
	0.01	5.24	6.33	7.03	7.56	7.97	8.32	8.61	8.87	9.10	9.30
7	**0.05**	3.34	4.16	4.68	5.06	5.36	5.61	5.82	6.00	6.16	6.30
	0.01	4.95	5.92	6.54	7.01	7.37	7.68	7.94	8.17	8.37	8.55
8	**0.05**	3.26	4.04	4.53	4.89	5.17	5.40	5.60	5.77	5.92	6.05
	0.01	4.74	5.63	6.20	6.63	6.96	7.24	7.47	7.68	7.87	8.03
9	**0.05**	3.20	3.95	4.42	4.76	5.02	5.24	5.43	5.60	5.74	5.87
	0.01	4.60	5.43	5.96	6.35	6.66	6.91	7.13	7.32	7.49	7.65
10	**0.05**	3.15	3.88	4.33	4.65	4.91	5.12	5.30	5.46	5.60	5.72
	0.01	4.48	5.27	5.77	6.14	6.43	6.67	6.87	7.05	7.21	7.36
11	**0.05**	3.11	3.82	4.26	4.57	4.82	5.03	5.20	5.35	5.49	5.61
	0.01	4.39	5.14	5.62	5.97	6.25	6.48	6.67	6.84	6.99	7.13
12	**0.05**	3.08	3.77	4.20	4.51	4.75	4.95	5.12	5.27	5.40	5.51
	0.01	4.32	5.04	5.50	5.84	6.10	6.32	6.51	6.67	6.81	6.94
13	**0.05**	3.06	3.73	4.15	4.45	4.69	4.88	5.05	5.19	5.32	5.43
	0.01	4.26	4.96	5.40	5.73	5.98	6.19	6.37	6.53	6.67	6.79
14	**0.05**	3.03	3.70	4.11	4.41	4.64	4.83	4.99	5.13	5.25	5.36
	0.01	4.21	4.89	5.32	5.63	5.88	6.08	6.26	6.41	6.54	6.66
15	**0.05**	3.01	3.67	4.08	4.37	4.60	4.78	4.94	5.08	5.20	5.31
	0.01	4.17	4.83	5.25	5.56	5.80	5.99	6.16	6.31	6.44	6.55
16	**0.05**	3.00	3.65	4.05	4.33	4.56	4.74	4.90	5.03	5.15	5.26
	0.01	4.13	4.78	5.19	5.49	5.72	5.92	6.08	6.22	6.35	6.46

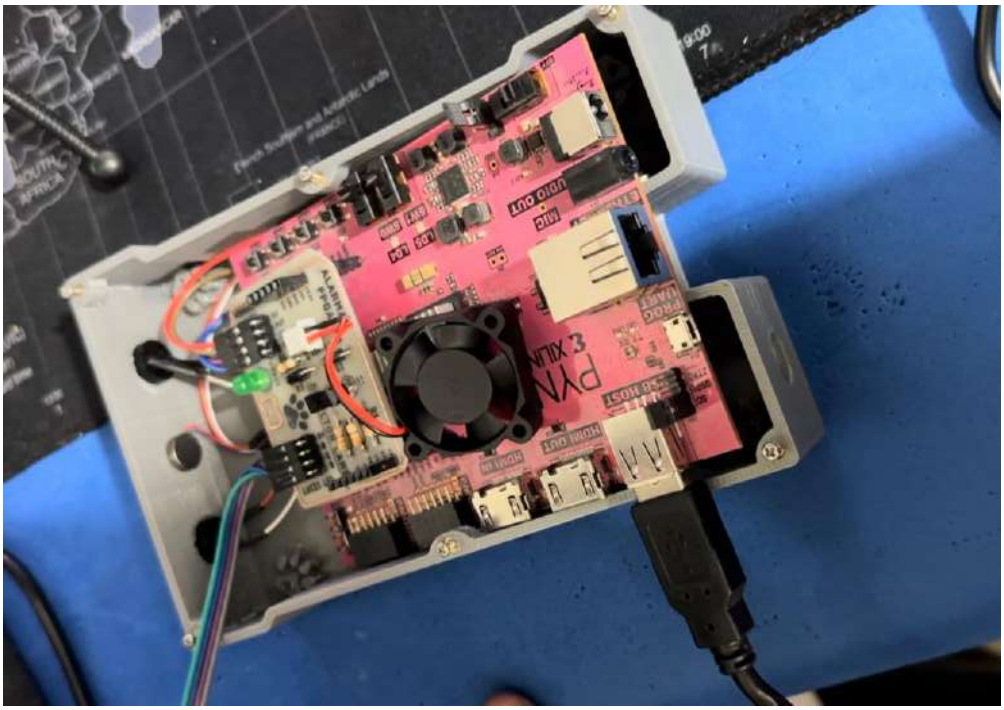
ANEXO C: ESTRUCTURA DEL DISPOSITIVO DETECTOR DE CANES



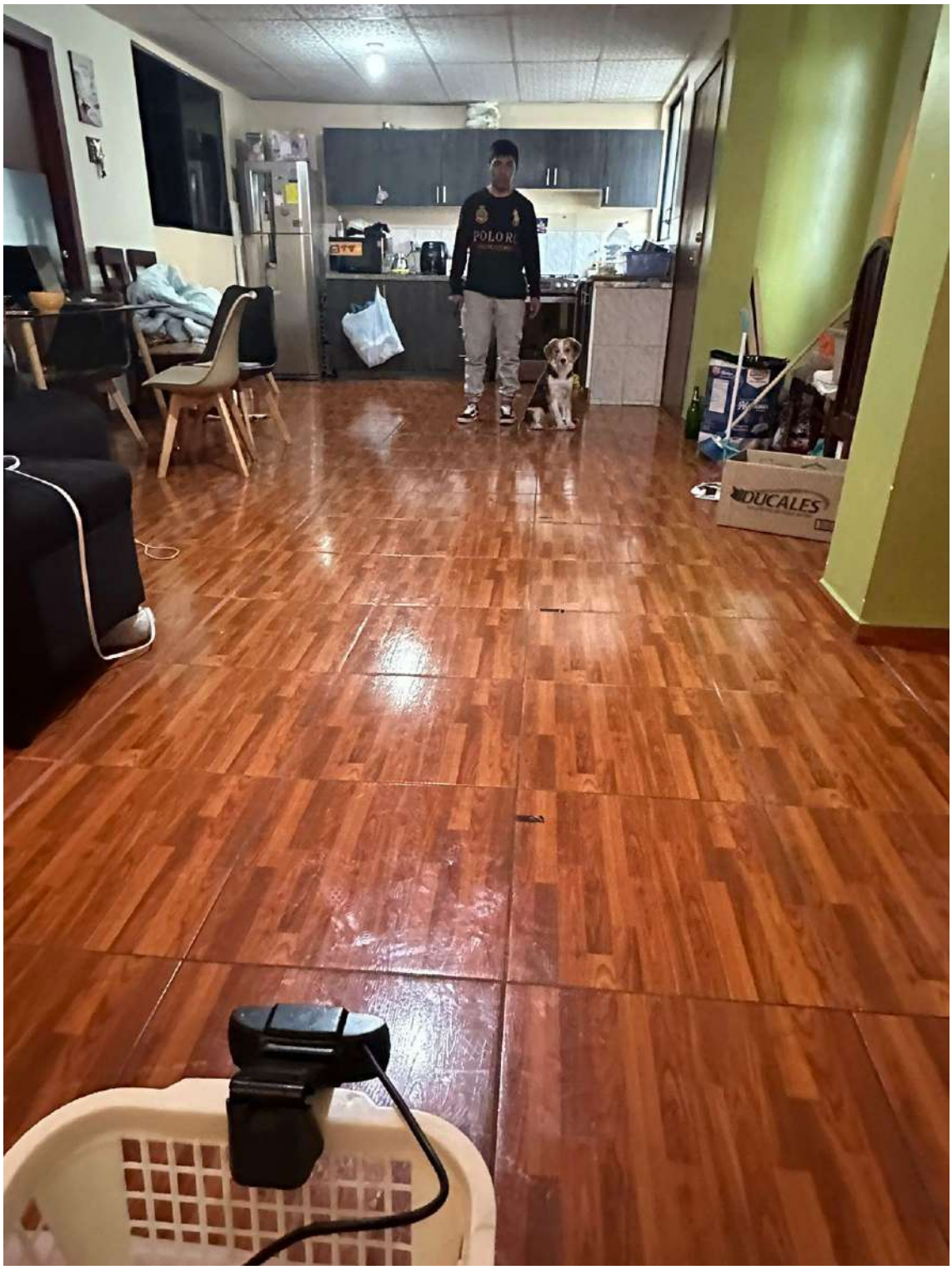


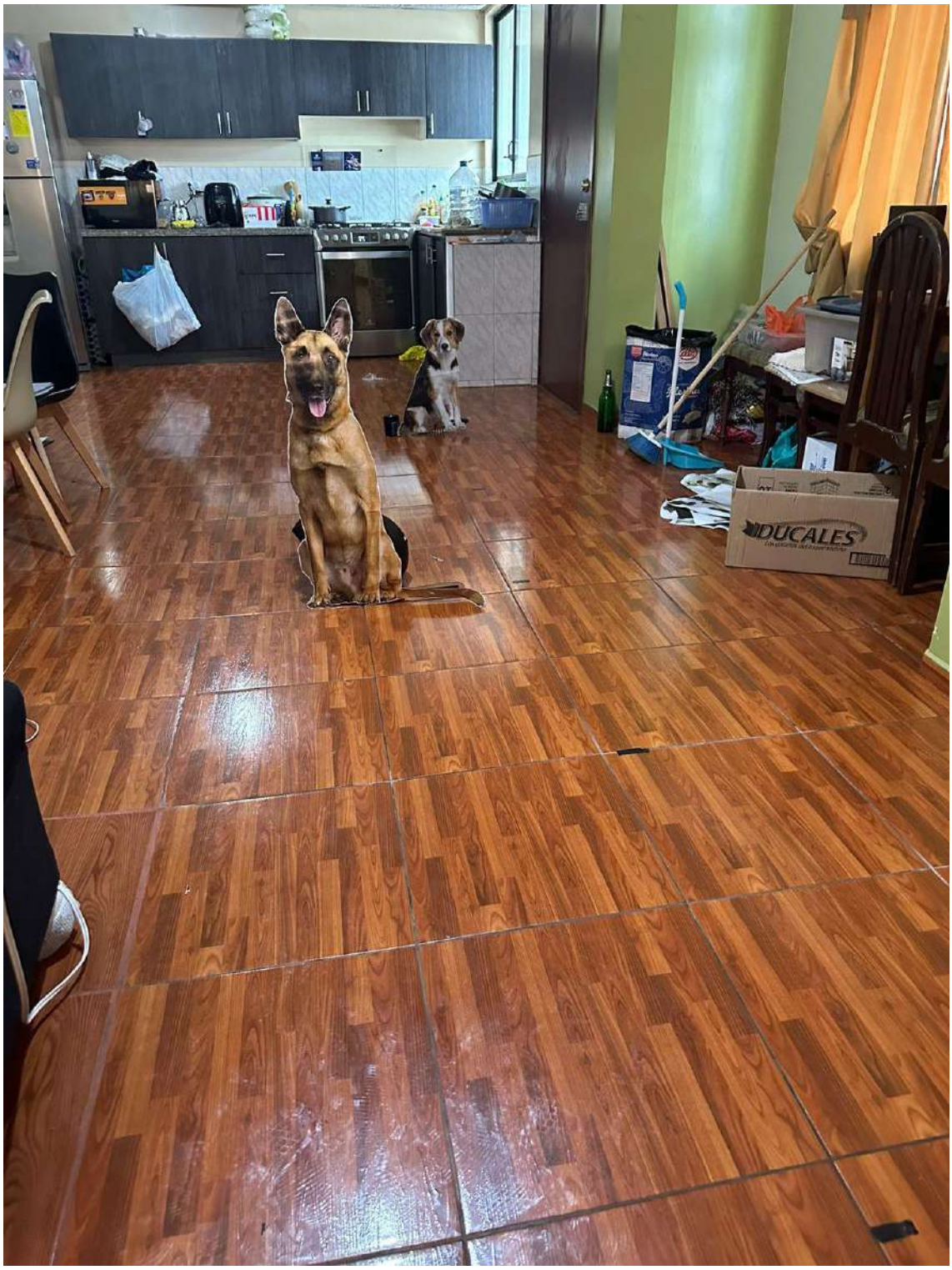

SOUTH AFRICA
French Southern and Antarctic Lands
PYNQ-Z1
XILINX
HDMI OUT
HDMI IN
MIC

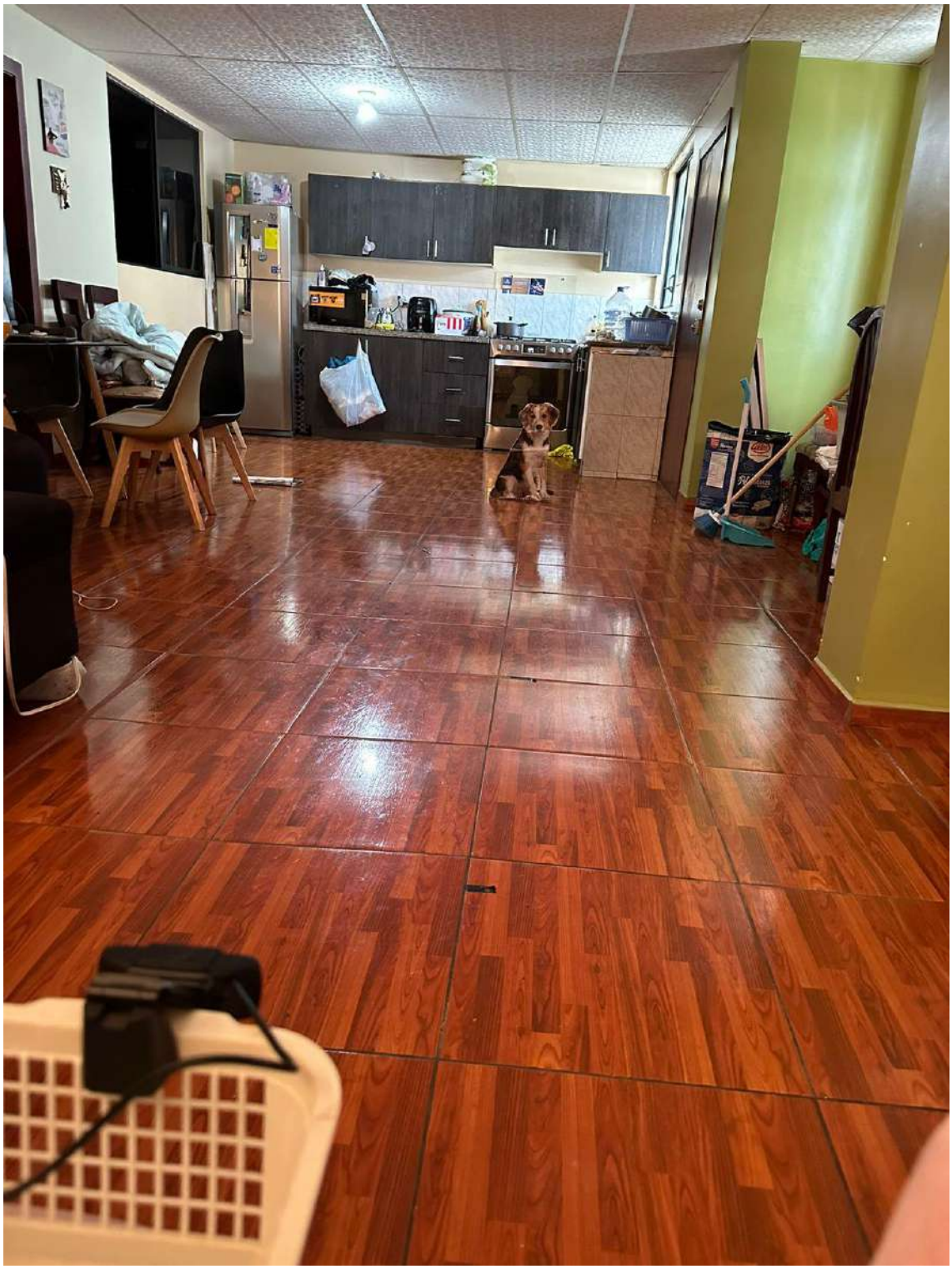

SOUTH AFRICA
French Southern and Antarctic Lands

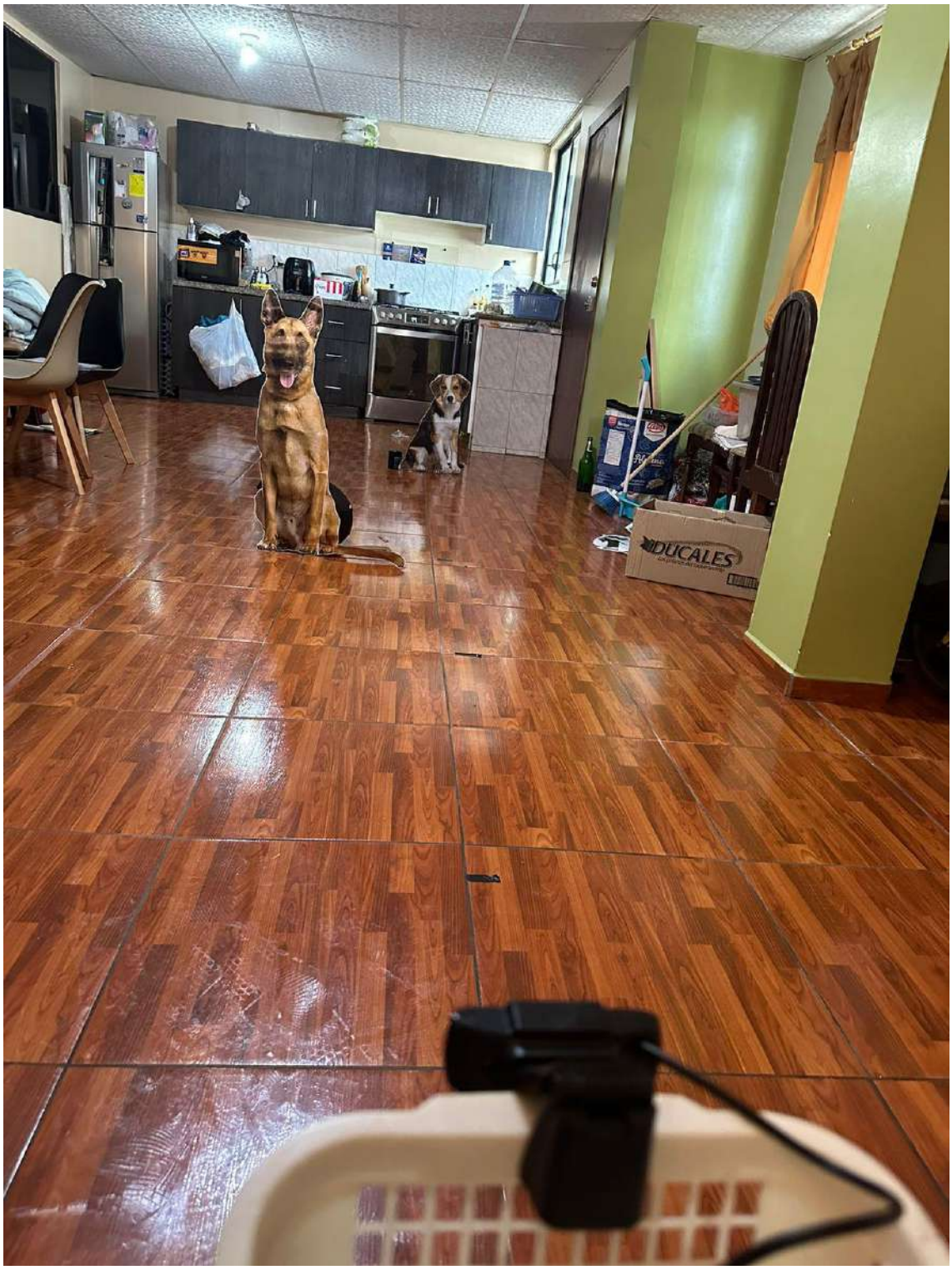


ANEXO D: EVIDENCIA FOTOGRÁFICA PRUEBAS EN AMBIENTE CONTROLADO


DUCALES


DUCALES




DUCALES

ANEXO E: EVIDENCIA FOTOGRAFICA PRUEBAS EN AMBIENTE NO CONTROLADO

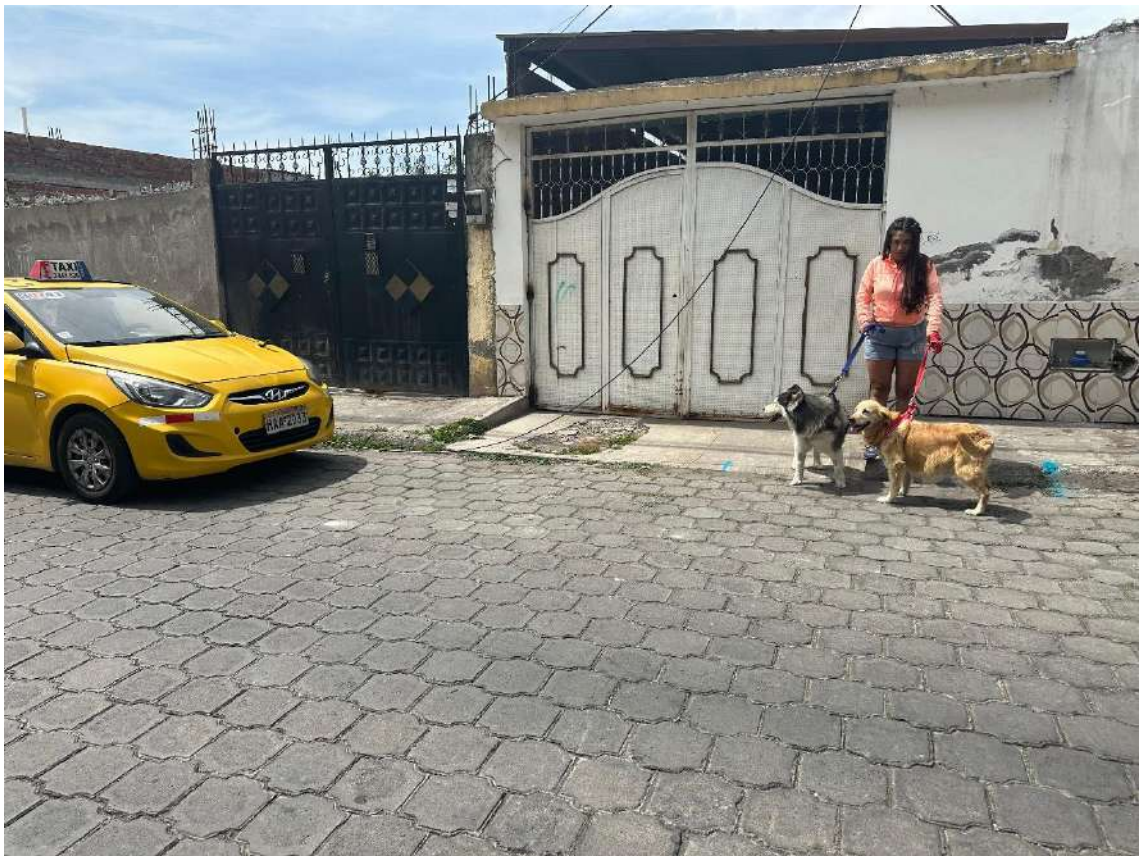

POR FAVOR
PROHIBIDO LIBAR
EM ESTA ZONA
GOLD
Monkey
licoreria

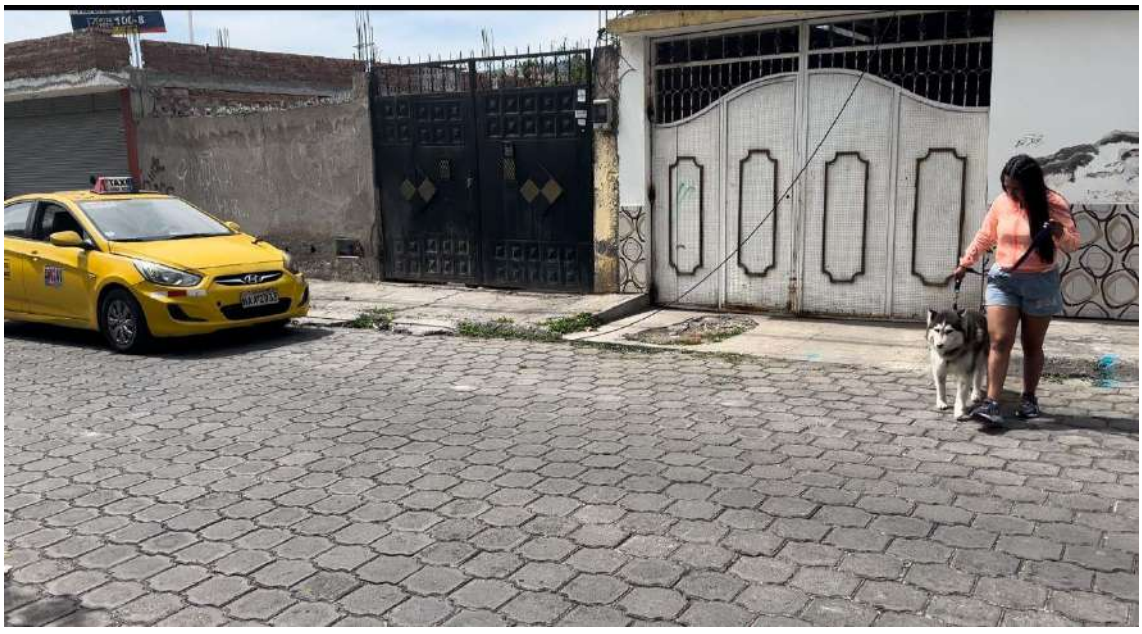

POR FAVOR
PROHIBIDO LIBAR
EM ESTA ZONA
GOLD
Monkey
licoreria

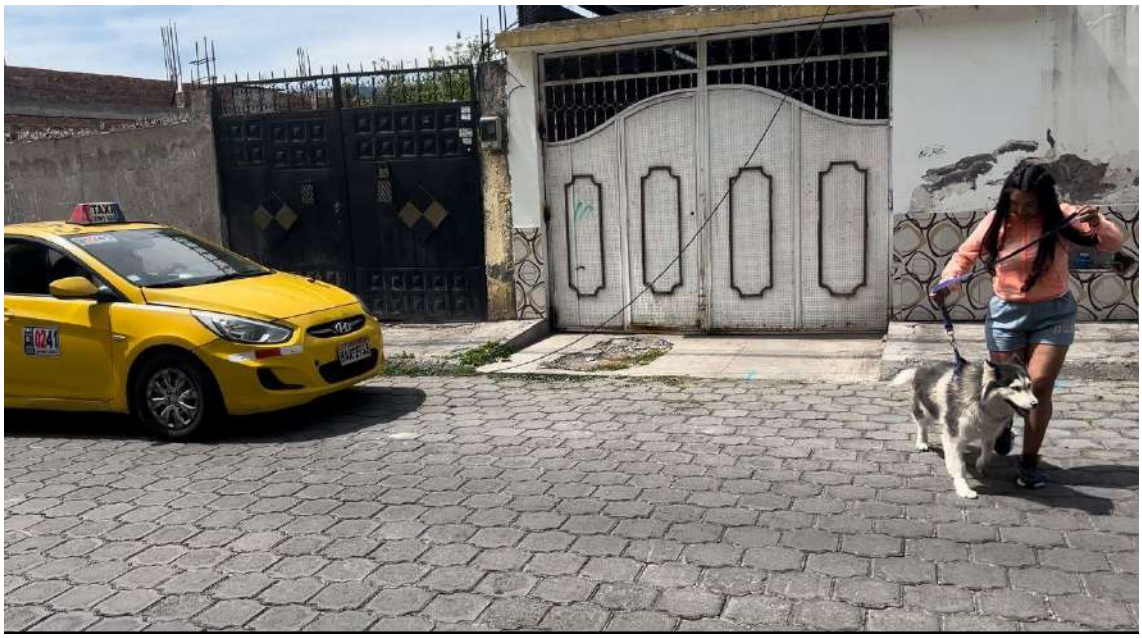

POR FAVOR
PROHIBIDO LIBAR
EN ESTA ZONA
Gold
Monkey
Licorería
EDUCACION
Y RESPETO


TIBER
EDUCACION
Y RESPETO

ANEXO F: EVIDENCIA FOTOGRÁFICA DE PRUEBAS DINÁMICAS






TAXI
0241


TAXI
COOPERATIVA
M ALDONADO
0241

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

DIRECCIÓN DE BIBLIOTECAS Y RECURSOS DEL APRENDIZAJE

UNIDAD DE PROCESOS TÉCNICOS

REVISIÓN DE NORMAS TÉCNICAS, RESUMEN Y BIBLIOGRAFÍA

Fecha de entrega: 23/11/2023

INFORMACIÓN DE LOS AUTORES
Nombres – Apellidos: CARLOS JULIO DOMINGUEZ OROZCO GIUSTINNE SOLANGE LLIGUIN AMBATO
INFORMACIÓN INSTITUCIONAL
Facultad: INFORMÁTICA Y ELECTRÓNICA
Carrera: ELECTRÓNICA Y AUTOMATIZACIÓN
Título a optar: INGENIERÍA EN ELECTRÓNICA Y AUTOMATIZACIÓN
f. Analista de Biblioteca responsable: Ing. Fernanda Arévalo M.