

Sistema de Monitoreo y Alerta Temprana basado en Inteligencia Artificial para Áreas Protegidas

Autores:

Fabrizio Martin Contigiani
Gabriel Orlando Da Silva Schmies

Tutor:

Dr. Ing. Sergio Eduardo Moya

27 de diciembre de 2025

Resumen

El presente trabajo describe el diseño e implementación de un sistema de monitoreo y alerta temprana para áreas protegidas, que combina tecnologías de Internet de las Cosas (IoT) con Inteligencia Artificial (IA) para la detección automática de fauna silvestre, personas y vehículos.

El sistema está compuesto por una red de nodos de captura basados en microcontroladores ESP32 equipados con cámaras, los cuales se comunican mediante el protocolo ESP-MESH para transmitir imágenes hacia un nodo raíz. Este nodo actúa como puerta de enlace, reenviando las imágenes a un servidor remoto a través de Transmission Control Protocol (TCP)/IP.

En el servidor, las imágenes son procesadas por un servicio de inferencia basado en SpeciesNet, un modelo de detección de objetos desarrollado por Google que utiliza YOLOv5. Este modelo permite identificar y clasificar especies de fauna silvestre, así como detectar la presencia de humanos y vehículos, generando alertas automáticas ante posibles intrusiones.

La arquitectura del servidor incluye una aplicación web desarrollada en Django para la gestión de imágenes, un bot de Telegram para el envío de notificaciones en tiempo real, y una base de datos PostgreSQL para el almacenamiento persistente. Todo el sistema está contenedorizado mediante Docker para facilitar su despliegue.

Los resultados demuestran la viabilidad de implementar un sistema de vigilancia inteligente de bajo costo para áreas protegidas, capaz de operar de manera autónoma y alertar a los administradores ante eventos relevantes.

Palabras Clave - Cámaras Trampa, Internet de las Cosas, Inteligencia Artificial, Monitoreo de Fauna, Detección de Intrusos, ESP-MESH, SpeciesNet, YOLO

Índice general

Índice de figuras	5
Índice de cuadros	6
Glosario	7
Siglas	9
1. Introducción	10
1.1. Contexto y motivación	10
1.2. Estructura del documento	11
2. Antecedentes	12
2.1. Trabajos relacionados	12
2.2. Soluciones comerciales existentes	12
2.3. Estado del arte	12
2.4. Análisis comparativo	12
3. Planteamiento del Problema	13
3.1. Áreas protegidas y conservación de fauna silvestre	13
3.2. Problemática de la vigilancia en áreas remotas	13
3.3. Sistemas de monitoreo tradicionales	13
3.3.1. Cámaras trampa convencionales	13
3.3.2. Limitaciones actuales	13
3.4. Necesidad de detección de intrusos	13
3.5. Justificación del proyecto	13
4. Objetivos y Alcance	14
4.1. Objetivo general	14
4.2. Objetivos específicos	14
4.3. Alcance del proyecto	14
4.4. Limitaciones	14
5. Marco Teórico	15
5.1. Internet de las Cosas (IoT)	15
5.1.1. Arquitecturas IoT	15
5.1.2. Protocolos de comunicación inalámbrica	15

5.2.	Redes Mesh	15
5.2.1.	Topologías de red	15
5.2.2.	ESP-MESH y Mwifi	15
5.3.	Inteligencia Artificial aplicada a visión por computadora	15
5.3.1.	Redes neuronales convolucionales (CNN)	15
5.3.2.	Detección de objetos con YOLO	15
5.3.3.	SpeciesNet de Google	15
5.4.	Tecnologías de desarrollo	15
5.4.1.	Microcontroladores ESP32	15
5.4.2.	ESP-IDF y ESP-MDF	15
5.4.3.	Contenedorización con Docker	15
5.4.4.	Framework Django	15
6.	Metodología	16
6.1.	Enfoque metodológico	16
6.2.	Etapas del desarrollo	16
6.3.	Herramientas y tecnologías utilizadas	16
6.4.	Métricas de evaluación	16
7.	Diseño del Sistema	17
7.1.	Arquitectura general	18
7.2.	Diseño del hardware	18
7.2.1.	Selección de componentes	18
7.2.2.	Nodo de captura con cámara	18
7.2.3.	Nodo raíz	18
7.2.4.	Alimentación y consumo energético	18
7.3.	Diseño de la red mesh	18
7.3.1.	Topología de la red	18
7.3.2.	Protocolo de comunicación	18
7.3.3.	Formato de datos	18
7.4.	Diseño del servicio de detección	18
7.4.1.	Servidor de inferencia con SpeciesNet	18
7.4.2.	Detección de animales, humanos y vehículos	18
7.4.3.	Anotación de imágenes con bounding boxes	18
7.5.	Diseño del servidor de aplicación	18
7.5.1.	Arquitectura de servicios	18
7.5.2.	Gestión de imágenes	18
7.5.3.	Interfaz web	18
7.5.4.	Bot de Telegram y sistema de alertas	18
8.	Implementación	19
8.1.	Nodo mesh (mesh-node)	20
8.1.1.	Firmware del nodo de captura	20
8.1.2.	Captura de imágenes	20
8.1.3.	Compresión y transmisión	20
8.2.	Nodo raíz (root-node)	20
8.2.1.	Firmware del nodo raíz	20

8.2.2.	Conexión con servidor TCP	20
8.2.3.	Gestión de la red mesh	20
8.3.	Servicio de detección (wildlife-detection)	20
8.3.1.	Contenedor Docker con SpeciesNet	20
8.3.2.	API de inferencia con LitServe	20
8.3.3.	Procesamiento de imágenes	20
8.4.	Servidor de aplicación (server)	20
8.4.1.	Aplicación Django	20
8.4.2.	Integración con SpeciesNet	20
8.4.3.	Bot de Telegram y sistema de alertas	20
8.4.4.	Base de datos PostgreSQL	20
8.4.5.	Despliegue con Docker Compose	20
9.	Pruebas y Resultados	21
9.1.	Ambiente de pruebas	21
9.2.	Pruebas de conectividad y red mesh	21
9.2.1.	Alcance de la red	21
9.2.2.	Latencia de transmisión	21
9.2.3.	Estabilidad de la conexión	21
9.3.	Pruebas de detección	21
9.3.1.	Detección de fauna silvestre	21
9.3.2.	Detección de humanos	21
9.3.3.	Detección de vehículos	21
9.4.	Evaluación del modelo de IA	21
9.4.1.	Precisión y recall	21
9.4.2.	Tiempo de inferencia	21
9.5.	Pruebas de consumo energético	21
9.6.	Pruebas del sistema de alertas	21
9.6.1.	Tiempo de respuesta	21
9.7.	Análisis de resultados	21
10.	Conclusiones	22
10.1.	Conclusiones generales	22
10.2.	Aportes del trabajo	22
10.3.	Trabajos futuros	22
10.4.	Recomendaciones	22
	Bibliografía	23
A.	Esquemáticos del hardware	24
B.	Código fuente relevante	25
B.1.	Firmware del nodo mesh	25
B.2.	Firmware del nodo raíz	25
B.3.	Servidor de detección	25
B.4.	Aplicación Django	25

C. Manual de instalación y configuración	26
C.1. Configuración del firmware	26
C.2. Despliegue del servidor	26
C.3. Configuración del bot de Telegram	26
D. Manual de usuario	27
E. Especificaciones técnicas	28
F. Análisis de viabilidad económica	29

Índice de figuras

Índice de cuadros

Glosario

- bot de Telegram** Programa automatizado que interactúa con usuarios a través de la plataforma de mensajería Telegram. 1, 10
- cámara trampa** Dispositivo de captura de imágenes activado por movimiento, utilizado comúnmente para monitoreo de fauna silvestre. 10
- caza furtiva** Práctica ilegal de captura o matanza de animales silvestres sin autorización, a menudo de especies protegidas o en peligro de extinción. 10
- código abierto** Software cuyo código fuente está disponible públicamente para que cualquiera pueda estudiarlo, modificarlo y distribuirlo. 11
- detección de objetos** Técnica de visión por computadora que identifica y localiza objetos dentro de una imagen, generalmente mediante bounding boxes. 10
- Django** Framework de desarrollo web de alto nivel escrito en Python, que sigue el patrón modelo-vista-plantilla. 1
- Docker** Plataforma de contenedorización que permite empaquetar aplicaciones junto con sus dependencias para facilitar el despliegue. 1
- ESP-MESH** Protocolo de red mesh desarrollado por Espressif para microcontroladores ESP32, basado en Wi-Fi. 1, 10
- ESP32** Microcontrolador de bajo costo y bajo consumo con Wi-Fi y Bluetooth integrados, fabricado por Espressif Systems. 1, 10
- inferencia** Proceso de utilizar un modelo de aprendizaje automático entrenado para realizar predicciones sobre nuevos datos. 1
- mesh** Topología de red donde cada nodo puede conectarse con múltiples nodos vecinos, permitiendo rutas alternativas para la transmisión de datos. 11
- nodo** Dispositivo individual que forma parte de una red mesh. 1
- nodo raíz** Nodo principal de una red mesh que actúa como puerta de enlace hacia redes externas. 1
- PostgreSQL** Sistema de gestión de bases de datos relacional de código abierto. 1

SpeciesNet Modelo de detección y clasificación de fauna silvestre desarrollado por Google, basado en YOLOv5. 1, 10

tiempo real Procesamiento de datos que ocurre con latencia mínima, permitiendo respuestas inmediatas a eventos. 10, 11

YOLOv5 Versión 5 del modelo You Only Look Once, arquitectura de red neuronal para detección de objetos en tiempo real. 1

Siglas

IA Inteligencia Artificial. 1, 10, 11

IoT Internet de las Cosas. 1, 10, 11

TCP Transmission Control Protocol. 1

Capítulo 1

Introducción

La conservación de la fauna silvestre y la protección de áreas naturales representan desafíos críticos en la actualidad. La pérdida de biodiversidad, la caza furtiva y la intrusión humana en ecosistemas protegidos amenazan el equilibrio ecológico y la supervivencia de numerosas especies. Ante esta problemática, surge la necesidad de implementar sistemas de monitoreo que permitan vigilar estas áreas de manera continua y eficiente.

Tradicionalmente, el monitoreo de fauna silvestre se ha realizado mediante cámara trampa, dispositivos que capturan imágenes cuando detectan movimiento. Sin embargo, estos sistemas convencionales presentan limitaciones significativas: requieren revisión manual periódica, no permiten alertas en tiempo real, y generan grandes volúmenes de datos que deben ser analizados manualmente por expertos.

El avance de tecnologías como el IoT y la IA ofrece nuevas posibilidades para superar estas limitaciones. La combinación de redes de sensores inalámbricos con algoritmos de detección de objetos permite desarrollar sistemas capaces de identificar automáticamente fauna silvestre, personas y vehículos, generando alertas inmediatas ante eventos relevantes.

El presente trabajo propone el diseño e implementación de un sistema de monitoreo y alerta temprana para áreas protegidas, que integra una red de nodos de captura basados en ESP32 comunicados mediante ESP-MESH, un servidor de procesamiento con IA basado en SpeciesNet, y un sistema de notificaciones a través de bot de Telegram.

1.1. Contexto y motivación

Las áreas protegidas enfrentan amenazas constantes que van desde la caza ilegal de especies en peligro hasta la intrusión de personas no autorizadas y vehículos en zonas restringidas. Los guardaparques y administradores de estas áreas frecuentemente carecen de los recursos humanos y tecnológicos necesarios para mantener una vigilancia efectiva sobre extensas superficies de terreno, muchas veces en ubicaciones remotas con acceso limitado a infraestructura de comunicaciones.

Las cámaras trampa tradicionales, aunque útiles para la investigación científica, no fueron diseñadas para la vigilancia en tiempo real. Las imágenes capturadas permanecen almacenadas en tarjetas de memoria que deben ser recolectadas físicamente, lo que implica visitas frecuentes al campo y retrasos significativos entre la captura de un evento y su descubrimiento. Para cuando se detecta una intrusión o un acto de caza furtiva, los responsables ya se encuentran lejos del área.

Es cierto que en la actualidad existen cámaras trampa más modernas que incorporan conectividad celular o satelital, permitiendo el envío remoto de imágenes. Sin embargo, estos dispositivos suelen tener un costo significativamente mayor, lo que limita su adopción masiva especialmente en países en vías de desarrollo, donde paradójicamente se concentra gran parte de la biodiversidad mundial. Además, muchas de estas soluciones comerciales dependen de servicios en la nube propietarios con costos de suscripción recurrentes.

La motivación de este proyecto surge de la necesidad de transformar el paradigma del monitoreo pasivo hacia un sistema activo de vigilancia inteligente, pero de manera accesible y de bajo costo. Un sistema que no solo capture imágenes, sino que las transmita en tiempo real, las analice automáticamente mediante algoritmos de IA, y genere alertas inmediatas cuando se detecten eventos de interés, ya sea la presencia de fauna silvestre para fines de investigación, o la detección de intrusos para fines de seguridad. Todo esto utilizando componentes de hardware económicos y software de código abierto.

1.2. Estructura del documento

El presente documento se organiza en diez capítulos que describen de manera progresiva el desarrollo del sistema propuesto:

Capítulo 1 - Introducción: Presenta el contexto general del proyecto, la motivación y la estructura del documento.

Capítulo 2 - Antecedentes: Revisa trabajos relacionados, soluciones comerciales existentes y el estado del arte en sistemas de monitoreo de fauna y detección de intrusos.

Capítulo 3 - Planteamiento del Problema: Describe la problemática de las áreas protegidas, las limitaciones de los sistemas tradicionales y la justificación del proyecto.

Capítulo 4 - Objetivos y Alcance: Define los objetivos generales y específicos, así como el alcance y las limitaciones del trabajo.

Capítulo 5 - Marco Teórico: Presenta los fundamentos teóricos sobre IoT, redes mesh, IA aplicada a visión por computadora, y las tecnologías utilizadas.

Capítulo 6 - Metodología: Describe el enfoque metodológico, las etapas de desarrollo y las herramientas empleadas.

Capítulo 7 - Diseño del Sistema: Detalla la arquitectura general, el diseño del hardware, la red mesh, el servicio de detección y el servidor de aplicación.

Capítulo 8 - Implementación: Presenta la implementación de cada componente: nodos mesh, nodo raíz, servicio de detección y servidor de aplicación.

Capítulo 9 - Pruebas y Resultados: Documenta las pruebas realizadas y analiza los resultados obtenidos en conectividad, detección, rendimiento y consumo energético.

Capítulo 10 - Conclusiones: Resume las conclusiones generales, los aportes del trabajo, trabajos futuros y recomendaciones.

Capítulo 2

Antecedentes

- 2.1. Trabajos relacionados
- 2.2. Soluciones comerciales existentes
- 2.3. Estado del arte
- 2.4. Análisis comparativo

Capítulo 3

Planteamiento del Problema

- 3.1. Áreas protegidas y conservación de fauna silvestre
- 3.2. Problemática de la vigilancia en áreas remotas
- 3.3. Sistemas de monitoreo tradicionales
 - 3.3.1. Cámaras trampa convencionales
 - 3.3.2. Limitaciones actuales
- 3.4. Necesidad de detección de intrusos
- 3.5. Justificación del proyecto

Capítulo 4

Objetivos y Alcance

- 4.1. Objetivo general
- 4.2. Objetivos específicos
- 4.3. Alcance del proyecto
- 4.4. Limitaciones

Capítulo 5

Marco Teórico

5.1. Internet de las Cosas (IoT)

5.1.1. Arquitecturas IoT

5.1.2. Protocolos de comunicación inalámbrica

5.2. Redes Mesh

5.2.1. Topologías de red

5.2.2. ESP-MESH y Mwifi

5.3. Inteligencia Artificial aplicada a visión por computadora

5.3.1. Redes neuronales convolucionales (CNN)

5.3.2. Detección de objetos con YOLO

5.3.3. SpeciesNet de Google

5.4. Tecnologías de desarrollo

5.4.1. Microcontroladores ESP32

5.4.2. ESP-IDF y ESP-MDF

5.4.3. Contenedorización con Docker

5.4.4. Framework Django

Capítulo 6

Metodología

- 6.1. Enfoque metodológico
- 6.2. Etapas del desarrollo
- 6.3. Herramientas y tecnologías utilizadas
- 6.4. Métricas de evaluación

Capítulo 7

Diseño del Sistema

7.1. Arquitectura general

7.2. Diseño del hardware

7.2.1. Selección de componentes

7.2.2. Nodo de captura con cámara

7.2.3. Nodo raíz

7.2.4. Alimentación y consumo energético

7.3. Diseño de la red mesh

7.3.1. Topología de la red

7.3.2. Protocolo de comunicación

7.3.3. Formato de datos

7.4. Diseño del servicio de detección

7.4.1. Servidor de inferencia con SpeciesNet

7.4.2. Detección de animales, humanos y vehículos

7.4.3. Anotación de imágenes con bounding boxes

7.5. Diseño del servidor de aplicación

7.5.1. Arquitectura de servicios

7.5.2. Gestión de imágenes

7.5.3. Interfaz web

7.5.4. Bot de Telegram y sistema de alertas

Capítulo 8

Implementación

8.1. Nodo mesh (mesh-node)

8.1.1. Firmware del nodo de captura

8.1.2. Captura de imágenes

8.1.3. Compresión y transmisión

8.2. Nodo raíz (root-node)

8.2.1. Firmware del nodo raíz

8.2.2. Conexión con servidor TCP

8.2.3. Gestión de la red mesh

8.3. Servicio de detección (wildlife-detection)

8.3.1. Contenedor Docker con SpeciesNet

8.3.2. API de inferencia con LitServe

8.3.3. Procesamiento de imágenes

8.4. Servidor de aplicación (server)

8.4.1. Aplicación Django

8.4.2. Integración con SpeciesNet

8.4.3. Bot de Telegram y sistema de alertas

8.4.4. Base de datos PostgreSQL

8.4.5. Despliegue con Docker Compose

Capítulo 9

Pruebas y Resultados

9.1. Ambiente de pruebas

9.2. Pruebas de conectividad y red mesh

9.2.1. Alcance de la red

9.2.2. Latencia de transmisión

9.2.3. Estabilidad de la conexión

9.3. Pruebas de detección

9.3.1. Detección de fauna silvestre

9.3.2. Detección de humanos

9.3.3. Detección de vehículos

9.4. Evaluación del modelo de IA

9.4.1. Precisión y recall

9.4.2. Tiempo de inferencia

9.5. Pruebas de consumo energético

9.6. Pruebas del sistema de alertas

9.6.1. Tiempo de respuesta

9.7. Análisis de resultados

Capítulo 10

Conclusiones

10.1. Conclusiones generales

10.2. Aportes del trabajo

10.3. Trabajos futuros

10.4. Recomendaciones

Bibliografía

- [1] J. Redmon, S. Divvala, R. Girshick y A. Farhadi, «You Only Look Once: Unified, Real-Time Object Detection,» *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, págs. 779-788, 2016.

Apéndice A

Esquemáticos del hardware

Apéndice B

Código fuente relevante

B.1. Firmware del nodo mesh

B.2. Firmware del nodo raíz

B.3. Servidor de detección

B.4. Aplicación Django

Apéndice C

Manual de instalación y configuración

C.1. Configuración del firmware

C.2. Despliegue del servidor

C.3. Configuración del bot de Telegram

Apéndice D

Manual de usuario

Apéndice E

Especificaciones técnicas

Apéndice F

Análisis de viabilidad económica