Ciencia y Educación
(L-ISSN: 2790-8402 E-ISSN: 2707-3378)
Vol. 6 No. 3
Marzo del 2025

DESARROLLO DE UN SISTEMA DE ALERTA PARA LA PREVENCIÓN DE ACCIDENTES DE TRÁNSITO MEDIANTE RECONOCIMIENTO DE SEÑALES Y VISIÓN ARTIFICIAL

DEVELOPMENT OF AN ALERT SYSTEM FOR THE PREVENTION OF TRAFFIC ACCIDENTS THROUGH SIGN RECOGNITION AND ARTIFICIAL VISION

Autores: [1]José Luis Jinez Tapia, [2]Paola Alexandra Ortiz Encalada, [3]José Luis Erazo Parra y [4]Rodrigo Patricio Toasa Jimenes.

[1]ORCID ID: https://orcid.org/0000-0002-4113-0579
[2]ORCID ID: https://orcid.org/0000-0001-8765-2308
[3]ORCID ID: https://orcid.org/0000-0003-3149-6718
[4]ORCID ID: https://orcid.org/0000-0001-8744-0794
[1]E-mail de contacto: jjinez@unach.edu.ec
[2]E-mail de contacto: portiz@unach.edu.ec
[3]E-mail de contacto: jlerazo@unach.edu.ec
[4]E-mail de contacto: rodrigo.toasa@unach.edu.ec
Afiliación: [1*2*3*4*]Universidad Nacional de Chimborazo, (Ecuador).
Articulo recibido: 14 de Enero del 2025
Articulo revisado: 16 de Enero del 2025
Articulo aprobado: 3 de Marzo del 2025
[1]Ingeniero en Electrónica y Telecomunicaciones graduado en la Universidad Nacional de Chimborazo, (Ecuador). Máster Universitario en Ingeniería de Telecomunicaciones graduado en la Università Della Calabria, (Italia). Magíster en Electricidad mención en Energías Renovables y Eficiencia Energética graduado en la Pontificia Universidad Católica del Ecuador, (Ecuador).
[2]Ingeniera Industrial graduada en la Universidad Nacional de Chimborazo, (Ecuador). Magíster en Seguridad Industrial mención Prevención de Riesgos y Salud Ocupacional graduada en la Universidad Nacional de Chimborazo, (Ecuador). Magíster en Producción y Operaciones Industriales graduada en la Universidad Técnica de Ambato, (Ecuador).
[3]Tecnólogo en Programación de Sistemas graduado en el Instituto Tecnológico Superior República Federal de Alemania, (Ecuador). Ingeniero en Sistemas e Informática graduado en la Universidad Regional Autónoma de los Andes, (Ecuador). Magíster en Informática Empresarial graduado en la Universidad Regional Autónoma de los Andes, (Ecuador).
[4]Ingeniero Automotriz graduado en la Escuela Superior Politécnica de Chimborazo, (Ecuador). Máster Universitario en Ingeniería Matemática y Computación graduado de la Universidad Internacional de la Rioja (España). Máster en Diseño Mecánico graduado en la Escuela Superior Politécnica de Chimborazo, (Ecuador).

Resumen

Los sistemas de visión artificial tienen un rol fundamental, permitiendo la captura y análisis de imágenes en las vías. Su integración facilita la identificación de señales en tiempo real derivadas del video. El propósito de la investigación se centró en diseñar y evaluar un sistema basado en visión por computadora para detectar señales de tránsito, líneas de carretera y proximidad entre vehículos. La metodología se centró en un estudio experimental centrado en la estructura del sistema, los dispositivos a utilizarse como cámara, alarmas, el análisis de las señales de tránsito en PYTHON, alcoholímetro, así como su evaluación. Dentro de los resultados, se implementó un prototipo electrónico de alerta automática para prevenir accidentes mediante visión artificial, capaz de reconocer señales de tránsito preventivas y horizontales, y alertar al conductor sobre posibles riesgos en la vía. El sistema incluye seguridad por alcohol y distancia, emitiendo señales sonoras ante peligros potenciales. Sin embargo, el reconocimiento de imágenes se ve afectado por la disminución de la luz natural. Este proyecto destaca por el uso de visión artificial y sus librerías, facilitando el procesamiento de imágenes y abriendo posibilidades para futuras mejoras, como el reconocimiento de ojos y tapabocas.

Palabras clave: Sistema de alerta, Prevención, Accidentes de tránsito, Reconocimiento, Señal artificial, Visión artificial.

Abstract

Artificial vision systems play a fundamental role, allowing the capture and analysis of images on the roads. Its integration facilitates the identification of real-time signals derived from the video. The purpose of the research

focused on designing and evaluating a system based on computer vision to detect traffic signs, road lines and proximity between vehicles. The methodology focused on an experimental study focused on the structure of the system, the devices to be used such as cameras, alarms, the analysis of traffic signs in PYTHON, a breathalyzer, as well as their evaluation. Among the results, an electronic automatic alert prototype was implemented to prevent accidents through artificial vision, capable of recognizing preventive and horizontal traffic signs, and alerting the driver about possible risks on the road. The system includes alcohol and distance safety, emitting sound signals in case of potential dangers. However, image recognition is affected by the decrease in natural light. This project stands out for the use of artificial vision and its libraries, facilitating image processing and opening possibilities for future improvements, such as eye and mask recognition.
Keywords: Warning system, Prevention, Traffic accidents, Recognition, Signal vision, Artificial vision.

Sumário

Os sistemas de visão artificial desempenham um papel fundamental, permitindo a captura e análise de imagens nas estradas. Sua integração facilita a identificação em tempo real de sinais derivados do vídeo. O objetivo da pesquisa centrou-se em projetar e avaliar um sistema baseado em visão computacional para detectar sinais de trânsito, linhas rodoviárias e proximidade entre veículos. A metodologia centrou-se num estudo experimental focado na estrutura do sistema, nos dispositivos a utilizar como câmaras, alarmes, na análise de sinais de trânsito em PYTHON, bafômetro, bem como na sua avaliação. Entre os resultados, foi implementado um protótipo de alerta eletrônico automático para prevenção de acidentes por meio de visão artificial, capaz de reconhecer sinais de trânsito preventivos e horizontais, e alertar o motorista sobre possíveis riscos na via. O sistema inclui álcool e segurança à distância, emitindo sinais sonoros em caso de potenciais perigos. No entanto, o reconhecimento da imagem é afetado pela diminuição da luz natural. Este projeto se destaca pela utilização da visão artificial e suas bibliotecas, facilitando o processamento de imagens e abrindo possibilidades para melhorias futuras, como reconhecimento de olhos e máscaras.
Palavras-chave: Sistema de alerta, Prevenção, Acidentes de trânsito, reconhecimento, sinalização artificial, Visão artificial.

Introducción

Los sistemas de apoyo visual para conductores por medios electrónicos son de gran importancia, ya que son dispositivos que ayudan a la seguridad en la carretera. Su aplicación en los vehículos aporta comodidad y permite la reducción de accidentes, los cuales generalmente se producen por cansancio y falta de claridad de las señales de tránsito. Estos sistemas alertan al conductor y mejoran su visualización de las señales viales, las cuales cumplen funciones informativas, preventivas y reglamentarias (Mateo, 2017). Diferentes estudios se han enfocado en la detección de señales de tránsito, especialmente aquellas de intersección como "Pare" y "Ceda el paso" (Flores et al., 2018), así como en señales informativas presentes en calles y carreteras (Chicaiza, 2017). Estos sistemas resultan indispensables para la seguridad de conductores y pasajeros.

Los sistemas de visión artificial tienen un rol fundamental, permitiendo la captura y análisis de imágenes en las vías. Su integración facilita la identificación de señales en tiempo real derivadas del video (Gamán, 2015). Se propone el desarrollo de un sistema basado en visión por computadora para detectar señales de tránsito, líneas de carretera y proximidad entre vehículos. Se integra un de detector de alcohol para los conductores. El sistema operará en tiempo real con el objetivo de emitir alertas

sonoras, permitiendo a una conducción más segura y confiable.

Materiales y Métodos

En la siguiente sección se detalla la estructura del sistema, los dispositivos a utilizarse como cámara, alarmas, el análisis de las señales de tránsito en PYTHON, alcoholímetro. La implementación del sistema se lo ejecuto siguiendo el esquema que presenta a continuación, en la figura 1. Para el desarrollo del prototipo electrónico de alerta automática, se desarrolla diferentes códigos de programación para el reconocimiento de señales preventivas, líneas de carril y un sistema de seguridad de distancia y alcoholímetro. Este proyecto cuenta con una cámara y sensores conectados a la tarjeta Raspberry Pi.

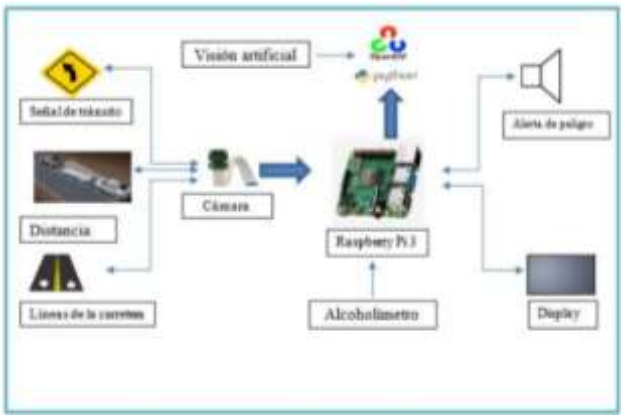

Figura 1: *Esquema general del sistema*

Instalación del software OpenCV en tarjeta Raspberry Pi

Para el desarrollo de un código de programación para la detección de colores y figuras en OpenCV, se realizaron los siguientes procesos; crear un algoritmo para el reconocimiento de señales de tránsito preventivas, desarrollo de un algoritmo de programación para detectar las líneas de carril. Diseño de un sistema de seguridad para detectar el nivel de alcohol del conductor y distancia entre vehículos, pruebas de funcionamiento del prototipo para la verificación de su funcionamiento y análisis de resultados.

Instalación de la librería OpenCv de Python en la tarjeta Raspberry Pi

OpenCv es una herramienta muy indispensable en el proyecto que vamos a realizar ya que nos facilita al momento de desarrollar nuestro algoritmo Una vez instalado el sistema operativo Raspberry Pi OS en la tarjeta Raspberry Pi procederemos abrir la ventana de comandos para digitalizar python3 con este comando pretendemos ver que Python tenemos instalado en la tarjeta Raspberry Pi.

Detección de colores y figuras en OpenCV

Para el desarrollo del siguiente código de programación, nos permitirá el reconocimiento de un color en específico el cual será el amarillo, se desglosa en los siguientes pasos para la detección del color; la imagen o fotograma a procesar, trasformar la imagen de BGR a HSV, determinar el rango del color y visualización. Lo primero que se realiza es la importación de las librerías después la lectura de la imagen, también se lo realiza en tiempo real para lo cual se activa la cámara y se reconoce la imagen. Cuando una imagen es reconocida por OpenCV por defecto la lee en BGR por ello es necesario la trasformación a HSV (Matiz, Saturación, Brillo). Se emplea este espacio de color ya que es más sencillo el reconocimiento del color Amarillo. Se necesita dos valores límites para determinar el rango del color uno inicial y uno final, se busca H ya que S y V toman valores entre 100 a 255 y de 20 a 255 respectivamente, el valor de H toma valores de 15 y 45. Y por último se visualiza los resultados de la detección del color amarillo.

Detección de figuras

Se realizó la detección de bordes Canny para identificar la figura geométrica siguiendo una

serie de pasos. En primer lugar, se filtró la imagen utilizando un filtro gaussiano con el fin de reducir el ruido y suavizar los bordes. Posteriormente, se aplicó el cálculo de detección de bordes Canny, seguido de la supresión de píxeles fuera del borde para eliminar respuestas no deseadas. Finalmente, se implementó la umbralización por histéresis, la cual empleó dos umbrales, uno máximo y otro mínimo, con el objetivo de determinar si un píxel formaba parte de un borde. La operación de filtrado se llevó a cabo mediante la convolución, la cual consistió en recorrer píxel a píxel la imagen utilizando una máscara de N × N, donde N representó el número de píxeles empleados en el procesamiento. La imagen fue convolucionada con el filtro gaussiano en las direcciones "X" y "Y" para mejorar la detección de bordes.

Detección de objetos mediante Haar Cascades en Python

La aplicación del método de Haar Cascade se dividió en dos partes: el cálculo de características de Haar y la creación de imágenes integrales. Para su implementación, se requirió una gran cantidad de imágenes tanto positivas como negativas. El cálculo de características de Haar consistió en realizar operaciones en regiones rectangulares dentro de una ventana específica, donde se calculó la suma de las intensidades de los píxeles en cada región y se determinó la diferencia entre dichas sumas. Para el reconocimiento de señales de tránsito preventivas, se empleó este método, para lo cual se creó una base de datos conformada por imágenes positivas, que contenían señales de tránsito preventivas, e imágenes negativas, que no las incluían. Después de obtener la base datos de las imágenes se entrenó el clasificador, el cual fue utilizado para reconocer las señales de tránsito preventivas empleando el método de Haar Cascade. Para entrenar el clasificador se empleó el software Cascade Trainger Gui. En la ventana se procede a configurar tres aspectos fundamentales para el entrenamiento que son; abrir la ubicación donde se encuentra la carpeta de las imágenes positivas y negativas, especificar el número de imágenes positivas que se obtuvo en este caso fueron 45 e insertar el número de imágenes negativas que se obtuvo en nuestro caso fueron 300. Una vez realizado el entrenamiento se genera una carpeta llamada clasiffier, donde está ubicado el clasificador nombrado en este caso como cascade.xml. Empleando este clasificador se construyó el código de programación para detectar las señales de tránsito preventivas en tiempo real

Detección de líneas de carretera con OpenCV

Para detectar las líneas de carril, se llevaron a cabo los siguientes pasos. En primer lugar, se realizó la importación de las librerías necesarias, entre ellas OpenCV, NumPy y Matplotlib. Posteriormente, se procedió a la lectura de la imagen utilizando la función "cv2.imread" de OpenCV, lo que permitió cargar la imagen para su posterior procesamiento. Posteriormente se convirtió la imagen a escala de grises por lo que se utiliza la función de opencv "cv2.cvtColor". Para poder eliminar el ruido de la imagen se aplica la función de "Gaussianblur" para lo cual se debe tener en cuenta los valores del tamaño Kernel como también la desviación estándar. Para aplicar la detección de bordes, se utilizó la función "cv2.Canny". Primero, se realizó la lectura de la imagen y se transformó a escala de grises para facilitar el procesamiento. Posteriormente, se establecieron dos valores importantes: el umbral bajo y el umbral alto, los cuales permitieron definir los bordes de la imagen. En este caso, la detección de bordes

permitió resaltar y dibujar las líneas de la carretera de manera efectiva.

Tabla 1. *Especificación de valores Canny*

Parámetros	Valor
Umbral bajo	50
Umbral alto	150

Fuente: Elaboración propia.

Región de interés

Se delimitó una región en (X,Y) donde se dibujó un triángulo con el objetivo de que, dentro de esta área, solo se encontraran las líneas del carril. Para ello, se determinaron los vértices del triángulo. Una vez hallados los vértices, la región contenida dentro del triángulo se consideró como la región de interés, permitiendo la detección precisa de las líneas del carril. Para la detección de líneas rectas, se empleó el espacio de Hough mediante la función "HoughLinesP" de OpenCV. Este método permitió identificar y trazar líneas dentro de la región de interés previamente delimitada, facilitando así la detección precisa de las líneas del carril en la imagen procesada. Para la detección de líneas rectas, se utilizó el espacio de Hough mediante la función "cv2.HoughLinesP" de OpenCV.

Tabla 2. *Parámetros para el espacio de Hough*

Parámetros	Valor
rho(ρ)	1
Theta(θ)	1° = πrad/180
Umbral mínimo de votos	15
Longitud mínima de línea	7
Espacio máximo entre líneas	3
Grosor de línea	1

Fuente: Elaboración propia.

Este método permitió identificar segmentos de línea en la imagen procesada, basándose en la transformación de Hough probabilística. La detección se realizó dentro de la región de interés previamente delimitada, lo que facilitó la identificación precisa de las líneas del carril en la imagen. Después de aplicar la trasformada Hough según los parámetros ya establecidos sobre la imagen se observa la detección de las líneas del carril.

Figura 2: *Detección de las líneas del carril*

Distancia de seguridad entre vehículos

Se realizó un código de programación para determinar la distancia de seguridad entre vehículos, para evitar una posible eventualidad proporcionando al conductor la capacidad de reaccionar de manera oportuna ante posibles imprevistos y así evitar accidente de tránsito. De acuerdo al Art.175 de la ley de tránsito se determina una distancia mínima entre vehículos de 3 metros a una velocidad de 60km/h. Se realizó un dispositivo con la capacidad de emitir alertas sonoras en función a la distancia, está conformado por un sensor ultrasónico para obtener la distancia, una tarjeta Raspberry Pi 4 para procesar la información y un buzzer para emitir las alertas sonoras. El código de programación se desarrolló en el lenguaje Python, se realizó los siguientes pasos; importar las librerías GPIO y Time, declarar los puertos como entrada o salida, asignar las variables y ejecutar el código de programación. Para obtener la distancia se conectó el sensor ultrasónico a la Raspberry pi en los puertos GPIO configurados como entrada. Los datos del sensor ultrasónico son procesados por código de

programación realizado en el lenguaje Python y ejecutado por la tarjeta Raspberry Pi.

Detección del nivel de alcohol en un conductor

Según el código orgánico integral penal, el límite máximo establecido es de 0.3 mg/L de alcohol en la sangre (El Universo, 2020). Se desarrollo un dispositivo que permite medir el nivel de alcohol del conductor, para asegurarse que se encuentre dentro de los límites permitidos por la ley y así evitar posibles sanciones o accidentes de tránsito que se puedan ocasionar por el estado etílico del conductor.

El dispositivo está conformado por un sensor para medir el nivel de alcohol, un buzzer para emitir alertas sonoras y una tarjeta Raspberry Pi para procesar la información. El sensor de alcohol se encuentra conectado a la raspberry Pi 4B en los puertos GPIO configurados como entrada. Los valores analógicos obtenidos del sensor no son directamente los valores de alcohol en el aire, se debe aplicar la ecuación para traducir las lecturas del sensor en miligramos por litro (mg/L). A continuación, se observa la ecuación 1:

$$alcohol = 0.4091\left(\frac{Rs}{5463}\right)^{-1.497}$$

Código de programación para la toma de imágenes positivas y negativas

```
import cv2
import numpy as np
import imutils
import os
Datos = 'n'
if not os.path.exists(Datos):
    print('Carpeta creada:', Datos)
    os.makedirs(Datos)
cap = cv2.VideoCapture(0,
cv2.CAP_DSHOW)
x1, y1 = 190, 80
x2, y2 = 450, 398
count = 0
while True:
    ret, frame = cap.read()
    if ret == False: break
    imAux = frame.copy()
    cv2.rectangle(frame,(x1,y1),(x2,y2),(255,0,
    0),2)
    objeto = imAux[y1:y2,x1:x2]
    objeto = imutils.resize(objeto, width=38)
    k = cv2.waitKey(1)
    if k == ord('s'):
        cv2.imwrite(Datos+'/objeto_{}.jpg'.form
        at(count),objeto)
        print('Imagen
        almacenada:','objeto_{}.jpg'.format(coun
        t))
        count = count + 1
    if k == 27:
        break
    cv2.imshow('frame', frame)
cv2.imshow('objeto', objeto)
cap.release()
cv2.destroyAllWindows()
```

Código de programación para el reconocimiento de señales preventivas

```
import cv2
import numpy as np
from matplotlib import pyplot as plt
cap = cv2.VideoCapture(0,cv2.CAP_DSHOW)
path =
'C:\\Users\\EDISON\\PycharmProjects\\Openc
vTest\\classifier\\cascade.xml'
faces = cv2.CascadeClassifier(path)
while True:
    ret,frame = cap.read()
    gray = cv2.cvtColor(frame,
    cv2.COLOR_BGR2GRAY)
    toy = faces.detectMultiScale(gray
                                 scaleFactor=5,
                                 minNeighbors=91,
                                 minSize=(70,78))
    for (x,y,w,h) in toy:
```

```
        cv2.rectangle(frame, (x,y), (x+w,y+h),
        (0, 255, 0), 2)
        cv2.putText(frame,'Curva_Abierta_Dere
        cha',(x,y-
        10),2,0.7,(0,255,0),2,cv2.LINE_AA)
        cv2.imshow('frame',frame)
        if cv2.waitKey(1) == 27:
            break
cap.release()
cv2.destroyAllWindows
```

Código de programación para detención de líneas de carril

```
import cv2
import numpy as np
def average_slope_intercept(image, lines):
    left_fit =[]
    right_fit = []
    for line in lines:
        x1, y1, x2, y2 = line.reshape(4)
        parameters = np.polyfit((x1, y1), (x2, y2),
        1)
        print(parameters)
def canny(image):
    gray = cv2.cvtColor(lane_image,
    cv2.COLOR_RGB2GRAY)
    blur = cv2.GaussianBlur(gray, (5, 5), 0)
    canny = cv2.Canny(blur, 50, 150)
    return canny
def display_lines(image, lines):
    line_image = np.zeros_like(image)
    if lines is not None:
        x1, y1, x2, y2 = line.reshape(4)
        cv2.line(line_image, (x1, y1), (x2, y2),
        (255, 0, 0), 10)
    return line_image
def region_of_interest(image):
    height = image.shape[0]
    polygons = np.array([[(200, height), (1100,
    height), (550, 250)]])
    mask = np.zeros_like(image)
    cv2.fillPoly(mask, polygons, 255)
    masked_image = cv2.bitwise_and(image,
    mask)
    return masked_image
image = cv2.imread('test_image.jpg')
lane_image = np.copy(image)
canny_image = canny(lane_image)
cropped_image =
region_of_interest(canny_image)
lines = cv2.HoughLinesP(cropped_image, 1,
np.pi/180, 100, np.array([]),
minLineLength=100,
maxLineGap=10)
averaged_lines =
average_slope_intercept(lane_image, lines)
line_image = display_lines(lane_image, lines)
combo_image = cv2.addWeighted(lane_image,
0.8, line_image, 1, 1)
cv2.imshow('LECTURA', combo_image)
cv2.waitKey(0)
```

Código de programación para distancia entre vehículos

```
#Importamos de las librerias
import RPi.GPIO as GPIO
import time
TRIG = 23 #El GPIO al cual conectamos la
señal TRIG del sensor
ECHO = 24 #El GPIO al cual conectamos la
señal ECHO del sensor
GPIO.setmode(GPIO.BCM) # Establecemos el
modo según el cual nos referiremos a los GPIO
de nuestra RPi
GPIO.setup(TRIG, GPIO.OUT) #
Configuramos el pin TRIG como una salida
GPIO.setup(ECHO, GPIO.IN) # Configuramos
el pin ECHO como una salida
try:
    # Implementamos un loop infinito
    while True:
        # Ponemos en bajo el pin TRIG y después
        esperamos 0.5 seg para que el transductor
        se estabilice
        GPIO.output(TRIG, GPIO.LOW)
```

```
        time.sleep(0.5)
        # Ponemos en alto el pin TRIG esperamos
        10 uS antes de ponerlo en bajo
        GPIO.output(TRIG, GPIO.HIGH)
        time.sleep(0.00001)
        GPIO.output(TRIG, GPIO.LOW)
        # En este momento el sensor envía 8
        pulsos ultrasónicos de 40kHz y coloca su
        pin ECHO
        en alto
        # Debemos detectar dicho evento para
        iniciar la medición del tiempo
        while True:
            pulso_inicio = time.time()
            if GPIO.input(ECHO) ==
            GPIO.HIGH:
                break
        while True:
            pulso_fin = time.time()
            if GPIO.input(ECHO) ==
            GPIO.LOW:
                break
        # Tiempo medido en segundos
        duracion = pulso_fin - pulso_inicio
        distancia = (34300 * duracion) / 2
        # Imprimimos resultado
        print("Distancia: %.2f cm" % distancia)
finally:
GPIO.cleanup()
```

Resultados y Discusión

Resultados de la detección de señales de tránsito preventivas

Las pruebas del dispositivo se realizaron de manera experimental, para lo cual se instaló el artefacto en el vehículo de pruebas y se realizó el recorrido de una ruta previamente establecida. Para la selección de la ruta se tomó en cuenta ciertos criterios como: cantidad de señales de tránsito preventivas, nivel de tráfico vehicular y velocidad del recorrido. El vehículo de pruebas realizó el recorrido a una velocidad de 50Km/h, demoró en completar el recorrido una hora, donde se detectó 126 señales de tránsito preventivas como, aproximación de semáforos, Curva abierta izquierda, curva abierta derecha, reductor de velocidad, peatones en la vía, animales en vía, cruce peatonal con prioridad, aproximación a redondel, empalme lateral en curva derecha, cruce de vías, niños y curva cerrada ala derecha. A continuación, se presentan varios escenarios en donde se pone a prueba cada una de las variables estudiadas en relación al horario, en el primero se analizó el horario de 12pm – 2pm, y en el segundo de 4am – 6am.

Escenario 1

En este escenario la toma de datos se realizó en el horario de 12pm a 2pm

Figura 3: *Escenario 1 desarrollado entre las 12pm – 2pm*

Una vez realizado el reconocimiento de las señales de tránsito preventivas obtenidas en el escenario 1 mediante el sistema desarrollado en este proyecto, se obtuvieron los resultados mostrados en la tabla 3.

Tabla 3. *Porcentaje de aciertos de reconocimiento de las señales de tránsito*

Tipo de señal	# de Señales	# de Señales detectadas	Porcentaje
Aproximación de semáforo	22	18	81,8%
Curva abierta izquierda	12	10	83,3%
Reductor de velocidad	12	11	91,6%
Peatones en la vía	34	28	82,3%
Animales en la vía	2	2	100%
Cruce peatonal con prioridad	12	7	58,3%

Tipo de señal	# de Señales	# de Señales detectadas	Porcentaje
Curva abierta a la derecha	16	16	100%
Aproximación a redondel	4	3	75%
Empalme lateral en curva derecha	4	2	50%
Cruce de vías	4	4	100%
Niños	2	2	100%
Curva cerrada derecha	2	2	100%
TOTAL	**126**	**105**	**83,3%**

Fuente: Elaboración propia.

Escenario 2

En este escenario la toma de datos se realizó en el horario de 12pm a 2pm

Figura 4: *Escenario 2 de 4am – 6am*

Una vez realizado el reconocimiento de las señales de tránsito preventivas obtenidas en e escenario 2 mediante el sistema desarrollado en este proyecto, se obtuvieron los resultados mostrados en la tabla 4.

Tabla 4. *Porcentaje de aciertos de reconocimiento - escenario 2*

Tipo de señal	# de Señales	# de Señales detectadas	Porcentaje
Aproximación de semáforo	22	13	59%
Curva abierta izquierda	12	5	41,66%
Reductor de velocidad	12	7	58,3%
Peatones en la vía	34	20	58,8%
Animales en la vía	2	0	0%
Cruce peatonal con prioridad	12	5	41,6%
Curva abierta a la derecha	16	10	62,5%
Aproximación a redondel	4	2	50%
Empalme lateral en curva derecha	4	1	25%
Cruce de vías	4	2	50%
Niños	2	0	0%
Curva cerrada derecha	2	1	50%
TOTAL	**126**	**66**	**52,38%**

Fuente: Elaboración propia.

Tabla 5: *Pruebas de muestras emparejadas*

ales Reconocidas	Horario	norte	Medios de comunicación	Desv. Desviación	Desv. Error promedio
	Madrugada	12	5,50	6.142	1.773
	Tarde	12	8,75	8,292	2.394

Fuente: Elaboración propia.

Tabla 6: *Pruebas de muestras emparejadas*

	Prueba de Levene		Prueba T para la igualdad de medias						
	F	Sig	t	gl	Sig (Bilateral)	Diferencia de medias	Error estándar	Inferior	Superior
Se asumen varianzas iguales	1,171	0,291	-1,091	22	0,287	-3,25	2,979	-9,248	2,928
No se asumen varianzas iguales			-1,091	20,279	0,288	-3,25	2,979	-9,458	2,958

Fuente: Elaboración propia.

Como p = 0.291 significa que las medias entre las señales detectadas de 12 a 14 pm y las señales detectadas de 4 a 6 am son significativamente diferentes, se concluye que, una correcta iluminación permite la visualización de las señales de mejor manera, caso contrario se dificulta su reconocimiento.

Resultados de la detección de líneas de carril

Para la realización de las líneas de carril se tomó en cuenta el área donde se va a realizar la detención de líneas de carril para lo cual se realizó un algoritmo donde ya se detalla paso a paso lo que se realizó para obtener los resultados que se muestran a continuación donde se visualiza el área de trabajo el radio de curvatura también si el conductor va en las siguientes direcciones ya que son izquierda, derecha, recto.

Resultados de la distancia de seguridad con otro vehículo

A continuación, se muestran los resultados obtenidos en relación a la distancia de seguridad

Tabla 7: *Valores reales con respecto a valores obtenidos del sensor ultrasónico*

Valor real (cm)	Valor medio (cm)	Error(cm)
50	49	1
100	96	4
150	144	6
200	191	9
250	243	7
300	290	10
350	339	11
400	388	12
450	427	23
500	470	30

Fuente: Elaboración propia

Tabla 8: *Estadísticas de grupo*

Valor	Grupo	norte	Medios de comunicación	Desv. Desviación	Desv. Error promedio
	REAL	10	275,00	151.383	47.871
	MEDIO	10	263,70	143.488	45.375

Fuente: Elaboración propia

Tabla 9: *Prueba T para muestras independientes*

	Prueba de Levene		Prueba T para la igualdad de medias						
	F	Sig.	t	gl	Sig (Bilateral)	Diferencia de medias	Error estándar	Inferior	Superior
Se asumen varianzas iguales	0,034	0,857	0,171	18	0,866	11,3	65,959	-127,254	150
No se asumen varianzas iguales			0,171	17,949	0,866	11,3	65,959	-127,303	149,903

Fuente: Elaboración propia.

En la tabla se observa que entre el valor de las medias del valor real y el valor medido son significativamente diferentes, por tanto, el sensor es válido para este sistema. Para realizar las pruebas de funcionamiento del sistema de seguridad entre vehículos, se consideraron tres distancias como son: 400cm, 200cm, 100cm. Unos leds indicarán a que distancia se encuentra el vehículo. Cuando no se encuentra ningún vehículo no se activa la alerta sonora, en estas circunstancias el conductor se encuentra totalmente seguro. Por otro lado, cuando el conductor se encuentra en una distancia no permitida, se activa la alerta sonora para dar a conocer al conductor sobre el rebasamiento de dicha distancia.

Resultados del nivel de alcohol en el conductor

Para comprobar el funcionamiento del dispositivo se procedió a realizar pruebas en 2 partes: con presencia de alcohol y sin presencia de alcohol. En la tabla 10 se realiza pruebas de nivel de alcohol a cinco sujetos, los cuales se encuentran en condiciones óptimas para poder conducir.

Tabla 10. Pruebas de nivel de alcohol a sujetos de prueba sin presencia de alcohol

Sujeto	Prueba 1 (mg/L)	Prueba 2 (mg/L)	Prueba 3 (mg/L)	Prueba 4 (mg/L)	Prueba 5 (mg/L)	Average (mg/L)
Sujeto 1	0.01	0.07	0.06	0.06	0.06	0.05
Sujeto 2	0.05	0.05	0.05	0.05	0.05	0.05
Sujeto 3	0.05	0.05	0.05	0.04	0.05	0.05
Sujeto 4	0.04	0.04	0.04	0.04	0.04	0.04
Sujeto 5	0.03	0.03	0.03	0.03	0.03	0.03

Fuente: Elaboración propia.

Tabla 11. Pruebas de nivel de alcohol a sujetos de prueba con presencia de alcohol

Subject	Prueba 1 (mg/L)	Prueba 2 (mg/L)	Prueba 3 (mg/L)	Prueba 4 (mg/L)	Prueba 5 (mg/L)	Average (mg/L)
Sujeto 1	4.20	4.21	4.27	5.10	5.18	4.59
Sujeto 2	4.27	4.30	5.20	5.23	5.37	4.87
Sujeto 3	7.92	7.85	7.60	7.42	7.37	7.63
Sujeto 4	6.75	6.71	6.48	6.20	6.03	6.43
Sujeto 5	3.95	3.90	3.86	3.80	3.06	3.11

Fuente: Elaboración propia.

En la tabla 11 se demuestra los datos tomados a cinco sujetos, que presentan un nivel de alcohol excesivo para lo cual no estarían aptos para conducir un vehículo. Estas pruebas se realizaron en dos escenarios, donde se tomó en

cuenta las horas del día debido a su iluminación natural. Una correcta iluminación permite la visualización de las señales de mejor manera, caso contrario se dificulta su reconocimiento. Los resultados encontrados en este estudio en cuanto al reconocimiento de señales de tránsito preventivas, se obtuvo un porcentaje de aciertos en el día de un 83.3% y en la noche de un 52.3%, esto significa que el dispositivo, al tener la presencia de luz natural responde con un porcentaje alto al reconocimiento de señales y al encontrarse con luz artificial se obtiene un porcentaje medio. Con respecto a los resultados de la detección de líneas de carril se aplica la trasformada Hough debido a su alta eficiencia y por la facilidad de representar las líneas que se encuentren, el funcionamiento del dispositivo se encuentra en un nivel aceptable, pero siempre y cuando las líneas de carril estén claramente pintadas sobre la vía. El análisis de la distancia entre vehículos se realiza entre dos valores (real, medio), ya que se observa un valor de significancia de 0.857 esto significa que el sensor ultrasónico aplica para el sistema de alerta. Los resultados del nivel de alcohol en el conductor se lo realizan a un grupo de sujetos obteniendo un resultado promedio de 0.05 sin la presencia de alcohol, un valor promedio de 5.54 con presencia de alcohol en estas condiciones el conductor no puede manejar.

Conclusiones

En este trabajo se implementó un prototipo electrónico de alerta automática para la prevención de accidentes mediante el reconocimiento de señales con visión artificial, el sistema es capaz de alertar al conductor sobre un posible suceso en la vía. Se desarrolló un código de programación para el reconocimiento de señales de tránsito preventivas y horizontales, el procesamiento de imágenes está limitado debido a que mientras disminuye la luz natural será más complicado el reconocimiento de las señales. Se implementó un sistema de seguridad de alcohol y distancia, mediante señales sonoras alertar al conductor frente a un posible accidente tránsito. El aporte importante que se le ha dado a este proyecto de titulación es la utilización de visión artificial ya que nos ha facilitado el trabajo de este proyecto con sus librerías y tratamiento de imágenes y mejoras en trabajos futuros (reconocimiento de ojos, reconocimiento de tapabocas).

Referencias Bibliográficas

Agermoso. (2015). Obtenido de http://licenciadeconducirrd.blogspot.com/2015/07/senales-de-transito.html

Amazon. (s.f.). Obtenido de https://www.amazon.es/LABISTS-C%C3%A1mara-Oficial-Raspberry-Compatible/dp/B07VTMNS2BANT

Chimborazo. (2015). Obtenido de Agencia Nacional de Tránsito: https://www.ant.gob.ec/index.php/noticias/boletines-provinciales/63-noticias-2/boletineschimborazo/1169-ant-chimborazo-registra-una-reduccion-de-siniestros-en-un-15-enjunio-de-2015#.XeZlxb5MSUk

Arce K., y Vasco I. (2018). Reconocimiento de patrones en imágenes de video para el monitoreo de eventos de tráfico Vehicular. Santiago de Cali.

Barba L. (2015). Utilización de métodos de visión artificial para PC como apoyo en la automoción. Madrid.

Casanova M. (2014). Diseño, Construcción e instalación de un alcoholímetro electrónico con dispositivo de bloqueo de un vehículo. Riobamba.

Cazau P. (2006). Introducción a la investigación en Ciencias Sociales. Buenos Aires.

Chicaiza F. (2017). Sistema Electrónico de alerta automática para el reconocimiento de señales de tránsito reglamentarias, preventivas e informativas en la ciudad de Ambato.

Cristina (2015). Slideshare. Obtenido de https://www.slideshare.net/Cristina1128/ecu

ador-connuevatipologadeaccidentesdetrnsito
Cruzado D. (2015). Detección y reconocimiento de señales de tráfico. Madrid.
EcuRed. (s.f.). Obtenido de https://www.ecured.cu/Modelo_HSV
Educación Vial. (2018). Obtenido de http://educacionvial201813.blogspot.com/2018/11/tipos-de-senales-de-transito-senalesde.
Educavial (2015). Slideshare. Obtenido de https://es.slideshare.net/educavial/seales-horizontales-49533482
El Universo. (2020). Obtenido de https://www.eluniverso.com/noticias/2020/09/14/nota/7977406/alguien-puede-negarsehacerse-prueba-alcoholemia-que-dice-ley/
Electronilab. (s.f.). Obtenido de https://electronilab.co/tienda/sensor-de-distancia-de-ultrasonidohc-sr04/
Estarita, J., Jiménez, A., Brochero, J., Escobar, H., y Moreno, S. (2018). Sistema de Reconocimiento de objetos en tiempo real. Colombia.
Flores M., Conlago C., Yunda J. y Aldás, M. (2018). Implementación de un algoritmo para la detención de señales de tránsito del Ecuador: Pare, Ceda el paso y Velocidad.
Gamán, l. R. (2015). Utilización de métodos de visión artificial para PC como apoyo en la automoción. Madrid.
Historia de la Informática. (2013). Obtenido de https://histinf.blogs.upv.es/2013/12/18/raspberry-pi/
INEN. (2011). Señalización Vial. Parte 1. Señalización Vertical. Quito. Obtenido de https://www.obraspublicas.gob.ec/wpcontent/uploads/downloads/2015/04/LOTAIP2015_reglamento-tecnico-ecuatoriano-rteinen-004-1-2011.pdf
Lara R., y Mares M. (2016). Reconocimiento de Patrones en Imágenes con un Sistema Embebido. Jalisco.
Marcano, M. (2018). Issuu. Obtenido de https://issuu.com/mariamarcan1996/docs/la_investigacion_experimental_pdf
Master autoescuela. (s.f.). Obtenido de https://www.autoescuelamasterbolivia.com/blog/senalesverticales/
Mateo J. (2017). Implementacion de un software para la detección y reconocimiento de señales de tráfico en tiempo real a partir de un video capturado en un vehículo en circulación.
Mendieta V. (2013). "Detección y Reconocimiento de Semáforos por Visión Artificial". Madrid.
Mujtaba, H. (2020). Great Learning. Obtenido de https://www.mygreatlearning.com/blog/viola-jones-algorithm/
Naylamp Mechatronics. (s.f.). Obtenido de https://naylampmechatronics.com/blog/42_Tutorialsensores-de-gas-MQ2-MQ3-MQ7-y-MQ13.html
Organización Mundial de la Salud (2018). Obtenido de https://www.who.int/es/news-room/fact-sheets/detail/road-traffic-injuries
Peden M. (2009). Informe Sobre la Situación Mundial de la Seguridad Vial. Suiza.
Pico G. (2019). Sistema Avanzado de Asistencia Al Conductor Empleando Visión Artificial en vehículos de Transporte Público. Ambato-Ecuador.
Rambal. (s.f.). Obtenido de https://rambal.com/raspberry/736-pantalla-raspberry-pi-5-in.html
Rueda B. P. (2017). Desarrollo de un prototipo portatil para el reconocimiento de señales dactilológicas mediante visión artificial. Quito.
Sailema F. (2017). Sistema Electrónico de alerta automática para el reconocimiento de señales de tránsito reglamentarias, preventivas e informáticas en la ciudad de Ambato. Ambato.
Stackoverflow. (2020). Obtenido de https://stackoverflow.com/questions/10948589/choosing-thecorrect-upper-and-lower-hsv-boundaries-for-color-detection-withcv?lq=1
Tutor de Programación. (2017). Obtenido de https://acodigo.blogspot.com/p/tutorialopencv

Villalón G., Torres M., y Flores, M. (2017). Sistema de detención de señales de tráfico para la localización de intersecciones viales y frenado anticipado. *Revista Iberoamericana de Automática e Informática Industrial, 1-11*.

Yanque J. (2016). SCRIBD. Obtenido de https://es.scribd.com/document/321852334/Metodologia-de-La-Investigacion-metodos-consulta