DEPARTAMENTO DE ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

TRABAJO DE TITULACIÓN, PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

TEMA: DESARROLLO DE UN SISTEMA DE CLASIFICACIÓN AUTOMÁTICA DE TIPOS DE TERRENOS EMPLEANDO TÉCNICAS DE MACHINE LEARNING.

AUTOR: GUEVARA BONILLA, MARCO ANTONIO

DIRECTOR: ING. LARCO BRAVO, JULIO CÉSAR

SANGOLQUÍ

2019

CERTIFICACIÓN

i

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES

CERTIFICACIÓN

Certifico que el trabajo de titulación, ***"DESARROLLO DE UN SISTEMA DE CLASIFICACIÓN AUTOMÁTICA DE TIPOS DE TERRENOS EMPLEANDO TÉCNICAS DE MACHINE LEARNING"*** fue realizado por el señor ***Guevara Bonilla, Marco Antonio*** el mismo que ha sido revisado en su totalidad, analizado por la herramienta de verificación de similitud de contenido; por lo tanto cumple con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de Fuerzas Armadas ESPE, razón por la cual me permito acreditar y autorizar al señor ***Guevara Bonilla, Marco Antonio*** para que lo sustente públicamente.

Sangolquí, 27 de mayo del 2019

Ing. Julio Larco

Director del proyecto

AUTORÍA DE RESPONSABILIDAD

ii

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES

AUTORÍA DE RESPONSABILIDAD

Yo, ***Guevara Bonilla, Marco Antonio*** con cédula de identidad N°: 1803137320, declaro que el contenido, ideas y criterios del trabajo de titulación: ***"DESARROLLO DE UN SISTEMA DE CLASIFICACIÓN AUTOMÁTICA DE TIPOS DE TERRENOS EMPLEANDO TÉCNICAS DE MACHINE LEARNING"*** es de mi autoría y responsabilidad, cumpliendo con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de Fuerzas Armadas ESPE, respetando los derechos intelectuales de terceros y referenciando las citas bibliográficas.

Consecuentemente el contenido de la investigación mencionada es veraz.

Sangolquí, 27 de mayo del 2019

Marco Guevara

CI: 1803137320

AUTORIZACIÓN

iii

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES

AUTORIZACIÓN

*Yo, **Guevara Bonilla, Marco Antonio** autorizo a la Universidad de las Fuerzas Armadas ESPE publicar el trabajo de titulación: **"DESARROLLO DE UN SISTEMA DE CLASIFICACIÓN AUTOMÁTICA DE TIPOS DE TERRENOS EMPLEANDO TÉCNICAS DE MACHINE LEARNING"** en el Repositorio Institucional, cuyo contenido, ideas y criterios son de mi responsabilidad.*

Sangolquí, 27 de mayo del 2019

Marco Guevara

CI: 1803137320

DEDICATORIA

A mi familia y amigos.

Marco Antonio Guevara Bonilla

AGRADECIMIENTO

A mi familia por todo su apoyo.

A todos aquellos que estuvieron presentes durante el desarrollo de esta tesis.

Al Ing. Julio Larco por su guía académica y apoyo en el desarrollo de este proyecto.

Marco Antonio Guevara Bonilla

ÍNDICE DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

El presente proyecto de investigación se enfoca en el desarrollo de un sistema que permita clasificar automáticamente por lo menos cuatro tipos de terrenos, como por ejemplo, pavimento, asfalto, grava y césped, empleando técnicas de Machine Learning. El alcance de este proyecto contempla el diseño e implementación del hardware para la adquisición de las señales de vibración recibidas por un sensor inercial (IMU, del inglés Inertial Measurement Unit), también constará de un GPS (Global Position System) que indicará la ruta en donde se realizó la prueba, el dispositivo estará ubicado en una bicicleta, la cual será el vehículo de prueba a utilizar. Adicionalmente, los datos recolectados se guardarán en una memoria micro SD, cuya información posteriormente se almacenarán en una base de datos, desde donde se tomará la información para clasificar los terrenos utilizando técnicas de Machine Learning, asimismo con las coordenadas geográficas proporcionadas por el GPS se mostrará la trayectoria que recorrió la bicicleta en un mapa, para lo cual se usara el recurso de google maps, para finalmente mostrar mediante una interfaz gráfica de usuario (GUI, del inglés Graphical User Interface) el tipo de suelo que atravesó, de manera off-line.

PALABRAS CLAVE

- **APRENDIZAJE AUTOMÁTICO**
- **IMU (INERTIAL SENSOR)**
- **GPS (POSICIONAMIENTO GLOBAL)**

ABSTRACT

The present research project focuses on the development of a system that automatically classifies at least four types of terrain, such as pavement, asphalt, gravel and turf, using Machine Learning techniques. The scope of this project includes the design and implementation of hardware for the acquisition of vibration signals received by an inertial measurement unit (IMU), will also consist of a GPS (Global Position System) that will indicate the route where the test was performed, the device will be located on a bicycle, which will be the test vehicle to use. Additionally, the collected data will be stored in a micro SD memory, whose information will later be stored in a database, from where the information will be taken to classify the terrains using Machine Learning techniques, also with the geographical coordinates provided by the GPS will show the trajectory of the bicycle on a map, for which the google maps resource will be used, to finally show through a graphical user interface (GUI) that type of terrain took, off-line.

KEYWORDS

- **MACHINE LEARNING**
- **IMU (INERTIAL SENSOR)**
- **GPS (GLOBAL POSITIONING)**

CAPÍTULO 1

1.1. Antecedentes

En la actualidad existen algoritmos de clasificación de terrenos utilizados para adaptar el modo de manejo de vehículos terrestres no tripulados, así como *Rovers* planetarios, vehículos militares y robots de exploración, estos algoritmos fueron desarrollados con el fin de determinar el tipo de suelo a los cuales puede estar expuesto el vehículo autónomo, uno de estos suelos puede ser el asfalto que tiene una superficie plana y poco resbaladiza, permitiendo que el recorrido sea seguro, otros terrenos como arena o grava pueden causar inconvenientes en la trayectoria, debido a su superficie resbaladiza e irregular (Weiss et al., 2006).

La capacidad de clasificar automáticamente el terreno atravesado por un vehículo será útil en muchos escenarios, ya que permitirá la formación de estimaciones de varias propiedades físicas (es decir, fricción, cohesión, ángulo de fricción interna, etc.). Estas estimaciones pueden utilizarse para vincular el rango de maniobras seguras en un algoritmo automatizado de prevención de riesgos (Ward. et al., 2009).

Para el desarrollo de los algoritmos de clasificación es necesario recopilar información de los terrenos, esta puede ser adquirida a través de sensores, que pueden ser de contacto y sin contacto. En los estudios con los sensores de contacto, las características como la frecuencia de vibración de la señal se utilizan para clasificar la superficie, (Brooks et al., 2005) usó un robot con un acelerómetro en su chasis. Usaron la medición de vibración del chasis para clasificar las superficies, (Weiss et al., 2006) incluye un acelerómetro a la carrocería del vehículo para medir las vibraciones en la dirección perpendicular al suelo. Luego, utilizaron la transformada rápida de

Fourier (FFT, del inglés *Fast Fourier Transform*) y la densidad espectral de potencia (PSD, del inglés *Power Spectral Density*) para entrenar un algoritmo, aplicando máquinas de vectores de soporte (SVM, del inglés *Support Vector Machines*) para clasificar las superficies. (Park et al.,2012) diseñó un robot móvil y utilizó los datos de los sensores de los neumáticos, de esta manera desarrollaron un método para extraer las características del terreno.

Los estudios de sensores sin contacto usaron principalmente sensores ópticos, sonares o acústicos para la clasificación del terreno. En un trabajo realizado por (Manduchi et al., 2005) se presentaron nuevos algoritmos de procesamiento de señales los cuales se adecuaron para la navegación autónoma, estos algoritmos consideraban dos sistemas de sensores compuestos por una cámara estéreo a color y un radar láser (LADAR, Laser Radar) de eje simple, que se complementaron entre sí para formar un sensor único, con el cual se obtuvo un sistema de clasificación basado en colores para etiquetar los obstáculos detectados; y un algoritmo para el análisis de datos LADAR que permitió discriminar entre hierba y obstáculos como troncos de árboles o rocas. (Bellutta et al., 2000; Castano et al., 2001; Talukder et al., 2002; Larson et al., 2004) usaron una sola cámara y desarrollaron una nueva técnica de clasificación del terreno. (Lee et al., 2011) utilizó una cámara con dispositivo de carga acoplada (CCD, del inglés *Charge Coupled Device*); extrajeron colores y texturas de los datos del sensor y clasificaron las superficies.

1.2. Justificación

La clasificación y caracterización del terreno es uno de los problemas más desafiantes asociados con la movilidad de los vehículos terrestres no tripulados (UGV, del inglés Unmanned Ground Vehicles), debido a que estos operan en diferentes tipos de superficies con características extremadamente diversas, que pueden afectar el rendimiento y la estabilidad de los mismos.

Por esta razón es importante verificar que tipo de terreno atraviesa el vehículo en un momento determinado; al utilizar esta información el vehículo puede adaptar y optimizar su modo de manejo para atravesar cualquier tipo de superficie.

Es por eso que el presente proyecto desarrollará un clasificador automático para cuatro tipos de suelos. Las anteriores investigaciones mencionadas utilizaron vehículos autónomos con sistemas embebidos, ocasionando que el costo sea elevado. Considerando esto, para factibilidad del proyecto se usará un vehículo de bajo costo como lo es una bicicleta para realizar las pruebas de los algoritmos, y posteriormente en trabajos futuros se podrá probar estos algoritmos en vehículos motorizados. También se aportará con una base de datos que estará disponible en la Universidad de las Fuerzas Armadas – ESPE y servirá como aporte para futuras investigaciones.

1.3. Alcance del proyecto

El proyecto consiste en desarrollar un sistema que permita la clasificación de cuatro tipos de terrenos (adoquín, asfalto, césped, grava) empleado técnicas de *Machine Learning*.

Este proceso inicia con la adquisición de datos obtenidos por medio de un sensor inercial, para después empezar con la extracción de las características a partir de las señales que se obtendrán del sensor inercial que requiere de un conjunto inicial de datos. La siguiente etapa consiste en la selección de las características para simplificar modelos y hacerlos más fácil de interpretar, también acortará los tiempos de entrenamiento.

Posteriormente se medirá el desempeño del clasificador mediante indicadores que mostrarán si los datos recolectados son suficientes para lograr una clasificación correcta. Es recomendable

que el sensor inercial entregue datos procesados para facilitar la clasificación del terreno y la generación de la base de datos.

El método de investigación que se aplicará en este proyecto para cumplir con los objetivos planteados es explicativo debido a que se detallarán cada uno de los procesos que surjan durante la operación del sistema, mientras la metodología que se usará es cuantitativa.

1.4. Objetivos

1.4.1. General

- Desarrollar un sistema de clasificación automática de tipos de terreno empleando técnicas de *Machine Learning*.

1.4.2. Específicos

- Realizar el estado del arte sobre sensores inerciales e identificación de terrenos con *Machine Learning*.
- Diseñar e implementar el *hardware* del sistema que permita la adquisición de las señales de vibración de un sensor inercial y la posición geográfica por GPS.
- Crear una base de datos para almacenar la información de las señales de vibración y las posiciones geográficas obtenidas.
- Generar el clasificador de terrenos aplicando *Machine Learning*.
- Desarrollar una interfaz gráfica de usuario que permita identificar el tipo de terreno y la ruta donde se encuentra.
- Establecer y realizar un protocolo de pruebas.

- Identificar el desempeño del clasificador.

CAPÍTULO 2

MARCO TEÓRICO

2.1. Señal digital

Es un tipo de señal que tiene dos niveles eléctricos específicos que varían en el tiempo, llevando información según el código previamente utilizado. Estos niveles eléctricos están representados por símbolos como el 1 que indica un estado alto y el 0 que indica un estado en bajo, la representación de los niveles eléctricos por medio de los símbolos 1 y 0 ayudan a que la lógica y la aritmética binaria sea más sencilla de aplicar.

Los niveles eléctricos también van a depender de la familia lógica que se utilice en los dispositivos, por ejemplo, en la familia TTL (del inglés, *Transistor Transistor Logic*) se tiene que los valores por debajo de 0.8 voltios son interpretados como 0 mientras que los valores por encima de 2 voltios son interpretados como 1. En cambio, en la familia CMOS los valores entre 0 y 2.25 voltios representan a un 0 y a partir de 2.75 voltios los valores representan a un 1. Esta representación de los niveles provee a la señal digital de una gran inmunidad al ruido (Miyara, 2004).

2.2. Muestreo

Es un proceso por el cual se toman datos cada cierto tiempo de una señal continua, con este proceso se logra convertir una señal analógica en una señal digital, también se debe considerar que el tiempo de adquisición de los datos se conoce como periodo de muestreo y el inverso de este como frecuencia de muestreo. En la Figura 1 se puede observar cómo se representa una señal continua luego de ser muestreada. (Cortés & Cano, 2008) (Oppenheim, 2009).

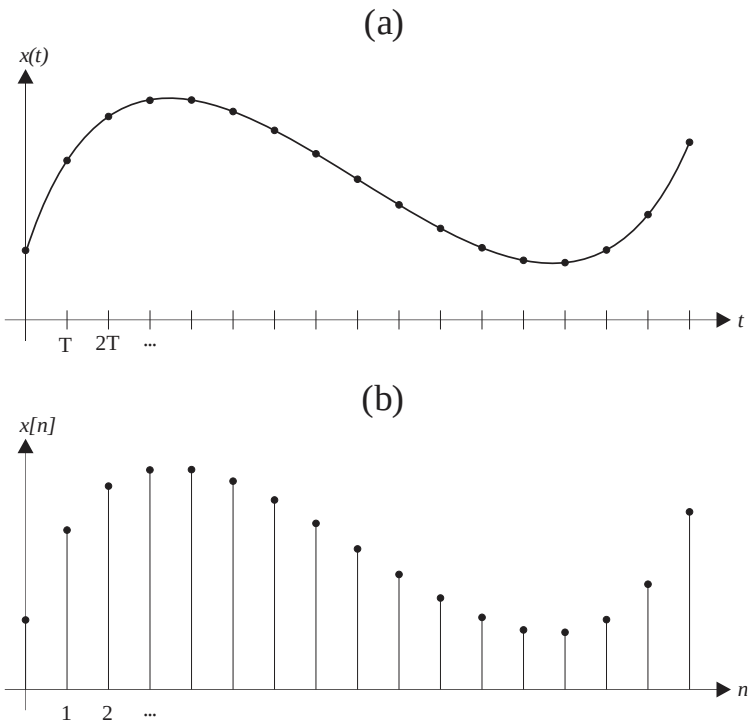


Figura 1. Muestreo de una señal continua: (a) señal continua, (b) señal muestreada

El teorema de Nyquist menciona que para poder reconstruir una señal de banda limitada, su tasa de muestreo debe ser mayor al doble de la frecuencia máxima que alcanza dicha señal.

$$f_s \geq 2f_{max}$$

Donde:

f_s es la frecuencia con la cual se realizará el muestreo de la señal dada.

f_{max} es la frecuencia máxima existente en la señal dada.

2.3. Transformada de Fourier

La transformada de Fourier es una herramienta muy útil para el procesamiento de señales, tanto en la física como en la ingeniería, el objetivo de esta herramienta es obtener una representación en frecuencia de las señales representadas en el dominio del tiempo, esto permite un análisis más amplio de las señales y facilita el estudio de las mismas (Meyer-Baese, 2007).

2.3.1. Transformada de Fourier en tiempo continuo

Las señales con periodo finito en el tiempo se pueden descomponer en una suma de exponenciales complejas, las cuales representan los armónicos en frecuencias que componen la señal. Este tipo de señales se pueden representar por medio de series de Fourier, haciendo que el periodo tienda al infinito, la sumatoria se puede convertir en una integral y por medio de una deducción matemática se obtienen las siguientes ecuaciones (Meyer-Baese, 2007) (Saltos, 2014).

$$X(f) = \int_{-\infty}^{+\infty} x(t) \cdot e^{-j \cdot 2 \cdot \pi \cdot f \cdot t} \cdot dt \tag{1}$$

$$x(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} X(f) \cdot e^{j \cdot 2 \cdot \pi \cdot f \cdot t} \cdot df \tag{2}$$

Donde (1) representa la transformada de Fourier de una señal $x(t)$, y (2) la ecuación de la transformada inversa de Fourier.

2.3.2. Transformada de Fourier en Tiempo Discreto

La dificultad de procesar la Transformada de Fourier para señales reales y el alto costo computacional han llevado a la búsqueda de alternativas que faciliten el procesamiento de las señales, es así que se llega a obtener la transformada de Fourier en Tiempo Discreto (DFT, del

inglés *Discrete Fourier Transform*), la idea de la DFT es discretizar las variables continuas y delimitar el número de muestras tanto en tiempo como en frecuencia (Meyer-Baese, 2007) (Oppenheim, 2009).

Para obtener las ecuaciones de la DFT se realiza un análisis similar al de la transformada de Fourier de tiempo continuo, tomando una secuencia finita x(n) y una longitud N.

$$X(k) = \sum_{n=0}^{N-1} x(n) \cdot e^{\frac{-j \cdot 2 \cdot \pi \cdot k \cdot n}{N}} \tag{3}$$

$$x(n) = \frac{1}{N} \cdot \sum_{k=0}^{N-1} X(k) \cdot e^{\frac{j \cdot 2 \cdot \pi \cdot k \cdot n}{N}} \tag{4}$$

Donde (3) es la ecuación de la transformada discreta de Fourier mientras que (4) es la ecuación de la transformada inversa de Fourier en Tiempo Discreto (IDFT).

2.4. Transformada rápida de Fourier

El cálculo de la DFT implica realizar una suma de N multiplicaciones complejas para cada salida, en total se obtiene N^2 multiplicaciones complejas y N^2 sumas complejas para realizar una DFT de N puntos, es decir que se tiene una complejidad de N^2, por lo cual se buscaron maneras más eficientes de realizar los cálculos de la DFT, estos algoritmos son conocidos como la Transformada Rápida de Fourier (FFT, del inglés *Fast Fourier Transform*). Los algoritmos de la FFT permiten realizar cálculos más eficientes reduciendo la complejidad al orden de $N * \log(N)$ (Meyer-Baese, 2007) (Oppenheim, 2009).

2.5. Validación cruzada

La validación cruzada (del inglés *Cross Validation*) es una técnica de evaluación utilizada para estimar la exactitud de un modelo de aprendizaje automático. Es utilizado comúnmente en el aprendizaje de máquinas para comparar y seleccionar un modelo predictivo, dado que es fácil de entender e implementar.

2.5.1. Validación cruzada k-Fold

Es un procedimiento de remuestreo utilizado para evaluar modelos de aprendizaje automático en una muestra de datos definida. El procedimiento tiene un único parámetro llamado k que se refiere al número de grupos en los que se debe dividir una muestra de datos, este procedimiento se denomina a menudo validación cruzada k-Fold. Cuando se elige un valor específico para k, se puede utilizar en lugar de k el valor elegido para referenciar el modelo, por ejemplo, si k=10 se convierte en 10 veces la validación cruzada, es decir validación cruzada 10-Fold (James, 2013).

El valor de k debe ser elegido cuidadosamente, un valor mal elegido puede dar como resultado un modelo erróneo, por lo general los valores que se le asigna a k es de 5 o 10, ya que se ha demostrado empíricamente que estos valores producen estimaciones de tasa de error pequeñas, además un valor de k = 10 es muy común en el aprendizaje automático (James, 2013).

Para concretar el procedimiento de la validación cruzada k-Fold, a continuación, se muestra un ejemplo:

Primero se debe tener las muestras de los datos, para este ejemplo se escogieron 20.

$$muestras = [1, 2, 3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20]$$

El siguiente paso es elegir un valor para k, esto fijara en grupos de cuanto se dividirán los datos, para este caso se utilizará un valor de k = 4, eso significa que se mezclarán los datos y luego serán divididos en 4 grupos, cada grupo tendrá 5 muestras ya que se tiene 20 en total.

Fold 1: [1,2,3,4,5]

Fold 2: [6,7,8,9,10]

Fold 3: [11,12,13,14,15]

Fold 4: [16,17,18,19,20]

Luego se utiliza las muestras obtenidas para evaluar los modelos del algoritmo de aprendizaje automático, los modelos que se obtienen son los siguientes:

Modelo 1

- Datos de entrenamiento: Fold 1, Fold 2, Fold 3.
- Datos de prueba: Fold 4.

Modelo 2

- Datos de entrenamiento: Fold 1, Fold 2, Fold 4.
- Datos de prueba: Fold 3.

Modelo 3

- Datos de entrenamiento: Fold 1, Fold 3, Fold 4.
- Datos de prueba: Fold 2.

Modelo 4

- Datos de entrenamiento: Fold 2, Fold 3, Fold 4.
- Datos de prueba: Fold 5.

Por último, los modelos se descartan después de ser evaluados.

2.6. Aprendizaje de máquina

El aprendizaje de máquina (del inglés *Machine Learning*) es un campo de las ciencias de la computación y una rama de la inteligencia artificial, cuyo objetivo es desarrollar técnicas que permitan a una máquina o computadora aprender y cambiar su comportamiento de manera autónoma basándose en el análisis de datos y técnicas de aprendizaje (Padilla, 2010)(Godoy, 2015).

2.6.1. Aprendizaje supervisado

Esta técnica de aprendizaje se divide en dos etapas, una de entrenamiento y la otra de prueba. En la etapa de entrenamiento se utiliza un 70% del total de los datos disponibles para enseñar al algoritmo a encontrar patrones y relaciones en los datos recolectados, mientras que en la etapa de prueba se utilizan los datos restantes para validar el rendimiento del algoritmo.

2.7. Máquinas de soporte vectorial

Las máquinas de soporte vectorial (SVM), son un conjunto de algoritmos de aprendizaje supervisado que fueron desarrollados en los años 90 por Vladimir Vapnik, las SVM tienen un mayor desempeño que otros algoritmos de aprendizaje tradicional como las redes neuronales y es

una herramienta muy utilizada para resolver problemas de clasificación (Carmona, 2016) (Betancourt, 2005).

El objetivo de las SVM es hallar un hiperplano que separe de forma óptima los datos de cada clase (ver Figura 2), existe un número infinito de hiperplanos que se pueden generar, pero la idea es seleccionar uno que equidista los datos más cercanos de cada clase para obtener un margen máximo, estos datos se denominan vectores de soporte (Carmona, 2016).

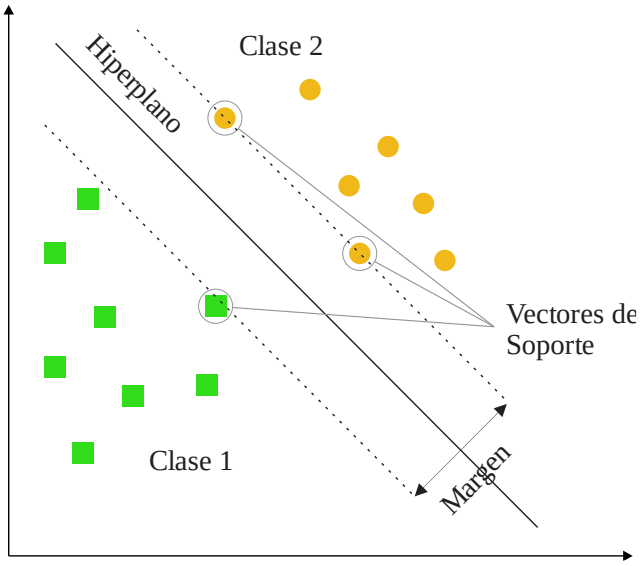


Figura 2. Hiperplano de separación óptimo

2.8. Matlab®

Matlab® es un software matemático con un lenguaje de programación propio, disponible para varias plataformas como por ejemplo Windows®, Mac OS®, GNU/Linux®, entre otras (MathWorks, s.f.). Esta herramienta se utilizó para la implementación del sistema de clasificación automática de tipos de terrenos, en su versión R2018b.

Matlab® cuenta con funciones y aplicaciones que son utilizadas para el análisis de señales en el dominio del tiempo y de la frecuencia, las cuales serán de gran ayuda para extraer características de las muestras obtenidas y así poder cumplir con los objetivos planteados en el proyecto.

CAPÍTULO 3

DISEÑO E IMPLEMANTACION DEL HARDWARE

En este capítulo se describen los elementos constitutivos del sistema de clasificación, así como las características necesarias para su implementación. Además, se detalla la programación necesaria para el correcto funcionamiento del hardware, por último, se especifican los protocolos de comunicación utilizados por el sensor y los módulos.

En este proyecto se pretende obtener un sistema que permita el reconocimiento de cuatro tipos de terreno y el geoposicionamiento de los mismos. Para cumplir con esto es necesario el uso de un sensor para medir las vibraciones producidas en el vehículo de prueba, así como, de un GPS para obtener las coordenadas del posicionamiento. La información estará almacenada en una memoria extraíble, por lo cual, el uso de un módulo de almacenamiento es necesario. Además, es indispensable el uso de una tarjeta programable para realizar el procesamiento de los datos y la incorporación de todo el hardware.

A continuación, se detallan los requerimientos que se tuvieron en cuenta para la selección de los elementos del hardware, así como la descripción de cada uno.

3.1. Diseño del hardware

El sistema está constituido por varios elementos físicos que conforman un sistema electrónico unificado. Dentro de estos elementos se encuentran: el acelerómetro, el módulo de almacenamiento, el módulo de posicionamiento global y el controlador, todos con funciones específicas para el correcto funcionamiento del sistema.

En la Figura 3 se muestra el diagrama general de bloques del sistema de clasificación y a continuación se darán los requerimientos técnicos mínimos necesarios para el correcto funcionamiento del sistema.

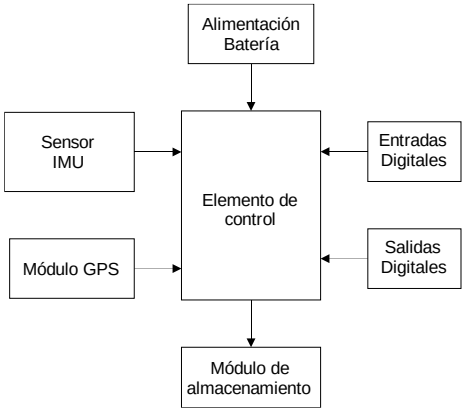


Figura 3. Diagrama general del Sistema de clasificación

3.1.1. Requerimientos del hardware

En este apartado del trabajo se detallarán los requerimientos mínimos que debe cumplir cada uno de los elementos que conforman la parte electrónica del sistema.

3.1.1.1. Controlador

Esta unidad es la encargada de controlar las actividades del sensor y los módulos por medio del microcontrolador, este elemento puede realizar el control por medio de los protocolos de comunicación, permitiendo así el sensado y almacenamiento de los datos. En la Tabla 1 se muestra las características mínimas que debe cumplir este elemento.

Tabla 1
Requerimientos del controlador

Elemento	Características	Función
Controlador programable	Alimentación de 5V Pines digitales de entrada y salida Comunicación serial Comunicación I2C Comunicación SPI Tamaño reducido	Controlar el funcionamiento del sensor y módulos electrónicos por medio de los protocolos de comunicación.

I2C. Es un protocolo de comunicación serial síncrono, envía los datos bit a bit de manera coordinada a través de una sola vía de comunicación.

SPI (del inglés *Serial Peripheral Interface*). Es en un protocolo de comunicación síncrono, puede enviar y recibir información gracias a su modo full dúplex, esto permite que dos dispositivos puedan comunicarse al mismo tiempo.

3.1.1.2. Sensor

Para la adquisición de datos es necesario utilizar un acelerómetro, este elemento permite medir la vibración que se produce en el vehículo de prueba al atravesar un terreno. (Weiss et al., 2006) menciona en su artículo que es posible adaptar un acelerómetro al chasis del vehículo para medir la vibración perpendicular al suelo. En la Tabla 2 se detallan las características mínimas requeridas para el sensor y la función que cumple dentro del sistema.

Tabla 2
Requerimientos del sensor

Sensor	Características	Función
Acelerómetro	Alimentación de 5V Comunicación I2C Rango 2g/4g/8g Salida digital	Medir la vibración que se produce en el vehículo de prueba

3.1.1.3. Módulo GPS

Para obtener la posición geográfica de los terrenos en las diferentes zonas de prueba se utiliza un módulo GPS, además gracias a las posiciones geográficas se puede trazar rutas de los diferentes terrenos y visualizarlos en un mapa. En la Tabla 3 se muestra las características mínimas que debe cumplir el módulo GPS.

Tabla 3
Requerimientos del módulo GPS

Módulo	Características	Función
GPS	Alimentación de 5V Comunicación serial	Obtener los valores de latitud y longitud

3.1.1.4. Módulo de almacenamiento

Para guardar los datos recolectados por el acelerómetro y las coordenadas del GPS se requiere de un módulo de almacenamiento, este módulo almacena los datos recolectados en una tarjeta de memoria permitiendo llevar la información a un computador para luego procesar los mismos. En la Tabla 4 se muestra las características mínimas que debe cumplir este módulo.

Tabla 4
Requerimientos del módulo de almacenamiento

Módulo	Características	Función
MicroSD Card	Alimentación de 5V Comunicación SPI Ranura para tarjeta de memoria	Almacenar en la tarjeta de memoria los datos del sensor y las coordenadas del GPS

3.1.1.5. Alimentación

Para poner en marcha el sistema electrónico es necesario contar con una alimentación de corriente continua, en la Tabla 5 se especifican los requerimientos mínimos.

Tabla 5
Requerimientos de alimentación de corriente continua

Alimentación	Características	Función
Batería	Voltaje de 5V	Energizar el sistema electrónico.

3.1.2. Elección de los componentes del sistema

A continuación, se explica cómo ha sido escogido cada elemento que compone el hardware, así como sus características principales. Cada elemento será escogido según los requerimientos descritos anteriormente, ya que se busca construir un sistema que tenga un nivel de fiabilidad alto y que sea accesible, es decir con un costo moderado.

3.1.2.1. Elección del controlador

El controlador es el elemento central del sistema electrónico, por eso es importante escoger el que mejor se adapte y cubra las necesidades del proyecto, ya que el control, la captura y el tratamiento de los datos son necesarios para correcto funcionamiento del sistema.

Existe una gran variedad de tarjetas programables en el mercado como por ejemplo Arduino, Raspberry PI, entre otras, estas tarjetas incorporan sistemas completos con microcontroladores potentes permitiendo que el desarrollo del hardware sea sencillo.

Para este proyecto se decidió usar una placa Arduino debido a su versatilidad, fácil uso y bajo costo, de la cual se tuvo dos opciones que se muestran la Tabla 6.

Tabla 6

Comparación de los controladores

	Arduino Nano	**Arduino UNO**
Microcontrolador	ATmega328P SMD	ATmega328P
Alimentación	5 V	5 V
Consumo de energía	19 mA	46 mA
Pines de E/S digitales	14	14
Protocolos de comunicación	Serial, I2C, SPI	Serial, I2C, SPI
Tamaño	18 x 45 mm	53.4 x 68.6 mm
Precio	$ 7	$ 10

Los modelos de tarjetas programables mostrados en la Tabla 6 cumplen con las características mínimas requeridas en el proyecto, su mayor diferencia está en el tamaño y consumo de energía, por lo tanto, cualquier opción es válida. En este proyecto se necesita que el dispositivo electrónico sea pequeño por lo tanto el controlador Arduino Nano es la mejor opción.

3.1.2.2. Elección del acelerómetro

Existen varios tipos de sensores que se pueden utilizar en diferentes aplicaciones electrónicas, por ejemplo, sensores para medir la presión, distancia, temperatura, etc. Dentro de esta amplia gama de sensores se necesita uno que mida la aceleración que se produce en la bicicleta al

desplazarse por los terrenos seleccionados. Para este proyecto se tuvieron tres opciones que se muestran en la Tabla 7.

Tabla 7

Comparación de los sensores

	ADXL335	**MPU6050**	**BNO055**
Alimentación	5V	5V	5V
Consumo de energía	350 uA	500 uA	12.3 mA
Protocolo de comunicación	----------	I2C	I2C
Rango	3g	2g, 4g, 8g y 16g	
Salida	Analógicas	Digital	Digital
Precio en dólares	7	4	34

El sensor ADXL335 fue descartado por no cumplir el requerimiento de salida digital, ya que al tener salida analógica se tendría que implementar un filtro digital para tener lecturas más estables.

El BNO055 es un sensor inercial (IMU) compuesto por un acelerómetro, giroscopio, magnetómetro y un software de fusión de sensores, este sensor cumple con los requerimientos mínimos planteados anteriormente pero también incorpora varias características que no se usaran en el proyecto por lo cual también se lo descarto.

El MPU6050 también es un sensor inercial (IMU) compuesto por un acelerómetro y un giroscopio, este elemento fue el seleccionado para adquirir los datos de aceleración que se produzca en la bicicleta, ya que cumple los requerimientos mínimos necesarios.

3.1.2.3. Elección del módulo GPS

El GPS es un sistema de posicionamiento que permite conocer la ubicación exacta en cualquier parte del mundo, en este proyecto se aprovechará esta tecnología para ubicar los terrenos y graficar las diferentes rutas recorridas. En la Tabla 8 se especifican los módulos GPS que se tuvieron como opción.

Tabla 8
Comparación de los módulos GPS

	Adafruit Ultimate GPS	**GPS6MV2**
Alimentación	5V	5V
Consumo de corriente	25 mA	100 mA
Protocolo de comunicación	Serial	Serial
Precio en dólares	40	17

Los módulos GPS mostrados en la Tabla 8 tienen características similares y cumplen con los requisitos mínimos para la implementación del proyecto, sin embargo, se eligió trabajar con el módulo GPS6MV2 por ser el más accesible en el mercado.

3.1.2.4. Elección del módulo de almacenamiento

Para almacenar los datos que recoja el sensor se necesita el uso de tarjetas de memoria externas como las SD o microSD. En la Tabla 9 se observan las características de las dos opciones de módulos que se escogieron para el proyecto.

Tabla 9
Comparación de los módulos de almacenamiento

	Modelo MOD-MSDC	**Modelo MD0043**
Alimentación	5V	5V
Consumo de energía	200 mA	200 mA
Comunicación	SPI	SPI
Ranura para tarjeta	Micro SD	SD
Precio en dólares	7	7

Ambos módulos tienen características similares y pueden cumplir la misma función en el proyecto, la cual es almacenar los datos recogidos por el sensor, en este caso se trabajará con el módulo de lectura MicroSD por ser más moderno y compacto, además las memorias SD tienen un precio elevado en comparación con las memorias microSD.

3.1.2.5. Elección de la batería

La alimentación de corriente continua será suministrada por una batería de 5 a 12 voltios, el consumo de corriente de todo el sistema es de aproximadamente 350 mA y se estima que el tiempo de sensado de los datos sea de una hora, por lo cual la capacidad de la batería debe ser mayor a 350 mAh. En la Tabla 10 se muestran las opciones de batería que se seleccionaron para el proyecto.

Tabla 10
Comparación de las baterías

	Batería de plomo	**Batería LiPO**
Voltaje	12 V	7.4 V
Capacidad	2300 Ah	1000 mAh
Medidas	179 x 35 x 61 mm	68 x 35 x 12.7 mm
Precio en dólares	11	15

Aunque la batería de plomo tiene el menor precio queda descartada debido a su tamaño y peso, ya que esto dificulta el montaje en la bicicleta, por tal motivo la batería LiPo (Polímero de Litio) es la que se usara en este proyecto ya que tiene las mejores prestaciones, siendo de un tamaño reducido y ligera.

3.1.3. Descripción del hardware

En este apartado se detallan las características de los elementos que fueron seleccionados según los requerimientos mínimos del sistema.

3.1.3.1. Arduino Nano

El Arduino Nano es una pequeña placa electrónica programable basada en el microcontrolador ATmega328P, incorpora un chip FTDI- FT232RL, este chip permite conectar el Arduino con un computador médiate USB, también posee 14 pines digitales que pueden funcionar como entradas y salidas, la placa funciona con una tensión de 5 voltios así que puede ser alimentada directamente por medio del USB, además incorpora un regulador de voltaje permitiendo alimentar la tarjeta con la batería LiPo de 7.4 voltios.

Cuenta con tres protocolos de comunicación:

- Los pines 0 (RX) y 1 (TX) se utilizan para recibir y transmitir datos serie TTL.
- Los pines 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK) son compatibles con la comunicación SPI.
- Los pines 4 (SDA) y 5 (SCL) permiten la comunicación I2C

En la Tabla 11 se muestran las especificaciones técnicas del Arduino Nano y en la Figura 4 la distribución de los pines.

Tabla 11

Especificaciones técnicas del Arduino Nano

Microcontrolador	ATmega328P
Voltaje de operación	5 V
Voltaje de entrada recomendado	7-12 V
Pines de E/S digitales	14 de los cuales 6 proveen de salida PWM
Entradas analógicas	8
Corriente máxima por cada pin de E/S	40 mA
Consumo de energía	19 mA
Frecuencia de reloj	16 MHz
Dimensiones	18 x 45 mm

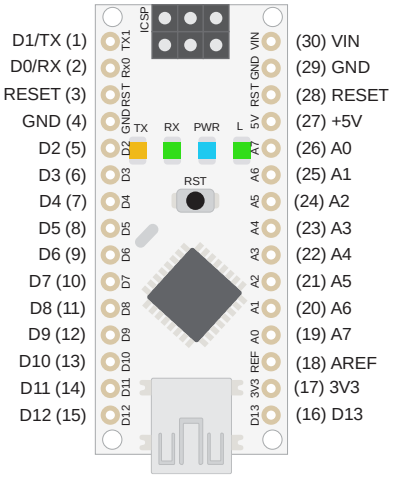


Figura 4. Esquema de distribución de los pines del Arduino Nano

3.1.3.2. MPU6050

El MPU6050 es una unidad de medición inercial de seis grados de libertad que combina en un solo integrado un acelerómetro y un giroscopio de 3 ejes (x, y, z) con una alta precisión, tiene una conversión de análogo a digital de 16 bits y un protocolo de comunicación I2C lo que permite trabajar con la placa Arduino, adicionalmente incorpora un regulador de voltaje de 3.3 voltios, con lo cual se puede alimentar con los 5 voltios de la placa. En la Figura 5 se aprecia el modulo físico de la unidad de medición inercial.

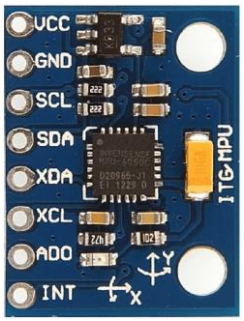


Figura 5. Módulo MPU6050

Las especificaciones técnicas del módulo MPU6050 se muestran en la Tabla 12.

Tabla 12

Especificaciones técnicas del módulo MPU6050

Voltaje de operación	3.3 a 5 V
Corriente de operación del acelerómetro	500 uA
Rango del acelerómetro	2g/4g/8g/16g
Rango del giroscopio	250Grad/Seg, 500Grad/Seg, 1000Grad/Seg, 2000Grad/Seg
Protocolo de comunicación	I2C
Conversión A/D	16 bits
Dimensiones	20 x 16 x 3 mm

3.1.3.3. Módulo GPS GY-GPS6MV2

Este módulo está equipado con un chip receptor NEO 6M de la marca UBLOX y cuenta con una antena cerámica que realizar la conexión con los satélites. El modulo puede ser alimentado con 5 voltios ya que tiene incorporado un regulador de voltaje de 3.3 voltios. También posee un protocolo de comunicación serial por medio del cual se envían las coordenadas geográficas hacia el Arduino. En la Figura 6 se muestra el hardware del módulo GPS.

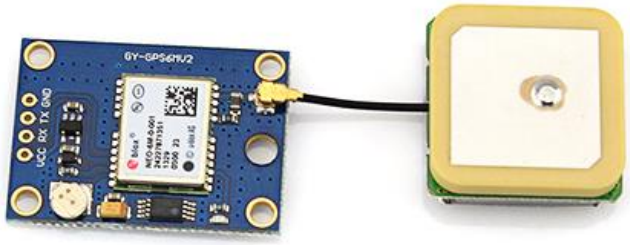


Figura 6. Módulo GY-GPS6MV2 con su antena cerámica

En la Tabla 13 se muestran las especificaciones técnicas del módulo GPS modelo GY-GPS6MV2

Tabla 13
Especificaciones técnicas del módulo GY-GPS6MV2

Voltaje de operación	3 a 5 V
Consumo de energía	100 mA
Antena	Cerámica
Protocolo de comunicación	Serial (9600bps)
Dimensión del modulo	23 x 30 mm
Dimensión de la antena	12 x 12 mm

3.1.3.4. Módulo de almacenamiento MOD-MSDC

Este módulo cuenta con una ranura para insertar una memoria Micro SD, está diseñado para acceder a la memoria por medio del protocolo de comunicación SPI, tiene incorporado un regulador de voltaje de 3.3 voltios, esto permite que el modulo sea alimentado por 5 voltios. Además, tiene un circuito de conversión de voltaje para realizar comunicaciones a 3.3 voltios o 5 voltios. En la Figura 7 se muestra el módulo de almacenamiento, y las especificaciones técnicas del mismo se las describen en la Tabla 14.

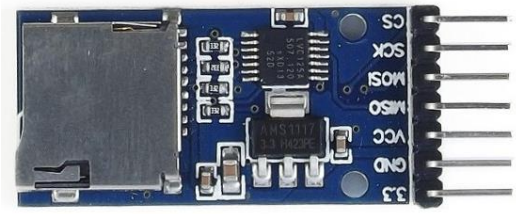


Figura 7. Módulo MOD-MSDC

Tabla 14

Especificaciones técnicas del módulo MOD-MSDC

Voltaje de operación	3.3 o 5 V
Consumo de energía	200 mA
Protocolo de comunicación	SPI
Ranura de memoria	Micro SD
Dimensión	42 x 25 mm

3.1.3.5. Batería LiPO

La batería seleccionada es una LiPo de 7.4 voltios con una capacidad de 1000 mAh, la capacidad indica cuanta corriente continua puede suministrar la batería en una hora, en este caso la batería es de 1000 mAh lo que indica que se descargará por completo en una hora si tiene una demanda de 1000 miliamperios, esta batería está compuesta por dos celdas, el valor de cada celda es de 3.7 voltios, la duración de las baterías LiPO es de 2 a 3 años aproximadamente. En la Figura 8 se muestra la batería seleccionada.

Figura 8. Batería LiPo

En la Tabla 15 se detallan las características de la batería seleccionada.

Tabla 15

Especificaciones técnicas de la batería LiPo

Voltaje	7.4 V - 2S
Capacidad	1000 mAh
Constante de descarga	25 – 50 C
Peso	58 g
Medidas	68 x 35 x 12.7 mm
Conector de carga	JST-XH
Conector de descarga	XT60

3.2. Conexiones

De acuerdo al diagrama de bloques mostrado en la Figura 1, el sistema electrónico está compuesto por un sensor MPU6050 conectado al Arduino por medio de comunicación I2C, también incorpora un módulo GPS y un módulo de almacenamiento, ambos conectados a la placa por medio de sus protocolos de comunicación serial y SPI respectivamente. En la Figura 9 se muestra el diagrama esquemático del circuito electrónico.

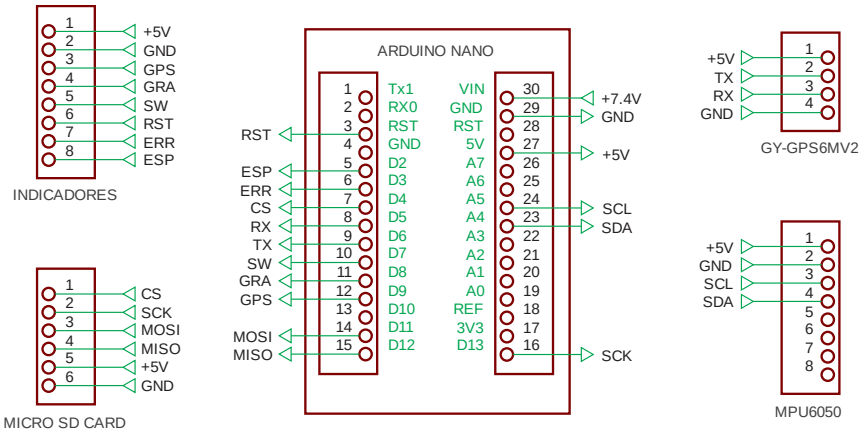


Figura 9. Diagrama esquemático del circuito electrónico

3.3. Integración del circuito electrónico

Para obtener un circuito electrónico unificado es importante realizar la integración de todos los componentes del hardware, es por eso que se diseñó una placa de circuito impreso (PCB). En la Figura 10 se puede observar el diseño PCB para la conexión de los elementos, la misma que consta de las siguientes partes:

- 30 pines para conectar la placa Arduino Nano.
- 6 pines para conectar el módulo de almacenamiento.

- 4 pines para conectar el módulo de GPS.
- Un molex macho de 8 pines para conectar los indicadores.
- Un USB tipo B para conectar el módulo MPU6050.
- Una bornera de dos pines para conectar la batería LiPo de 7.4 voltios.

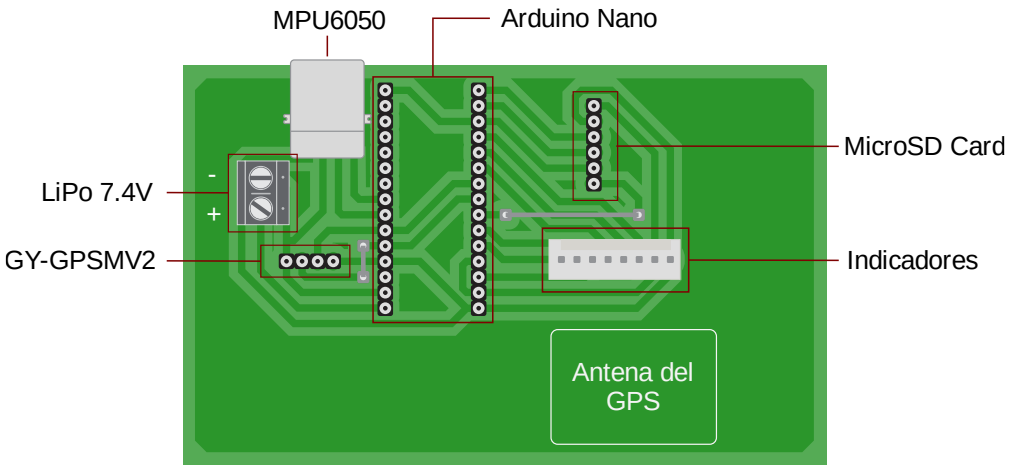


Figura 10. PCB del circuito electrónico

También se diseñó un contenedor que se ajuste a las medidas del PCB para proteger la placa de los golpes y la corrosión. El contenedor fue diseñado en el software SketchUp 2018, luego se extrajo su modelo en 3D en formato STL y una vez obtenido el diseño se realizó el prototipo en una impresora 3D. La Figura 11 representa el modelo en tres dimensiones de la caja y la tapa del contenedor.

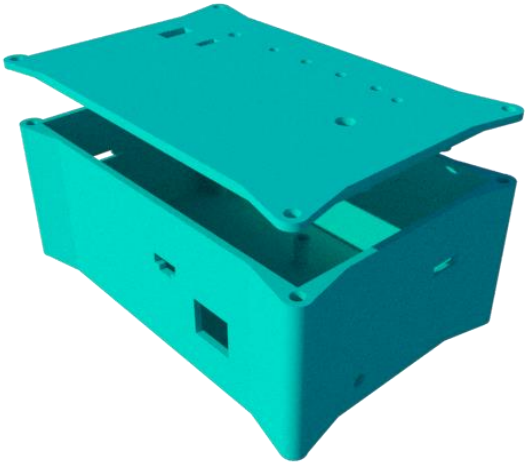


Figura 11. Diseño del contenedor

3.4. Ubicación del sensor en la bicicleta

Los marcos de las bicicletas están diseñados para atenuar las vibraciones producidas por el suelo, sin embargo, los valores de vibración más altos se producen cerca de las ruedas (Pérez, 2016). Por tal motivo, el acelerómetro se ubicó cerca del buje trasero, en el triángulo formado por las vainas inferiores, ver Figura 12.

Figura 12. Ubicación del sensor MPU6050

3.5. Montaje final

En la Figura 13 se muestra la ubicación de los elementos que componen el sistema electrónico, los elementos estarán colocados en la bicicleta de la siguiente manera:

- La placa que contiene el controlador y los módulos se ubicará en el manubrio junto con la batería.
- El acelerómetro estará ubicado en las vainas inferiores de la bicicleta.

Figura 13. Montaje final del hardware

3.6. Diseño del programa Arduino

Para llevar a cabo todas las acciones de control del proyecto, es necesario desarrollar un programa para el controlador Arduino, el software que se utiliza para desarrollar el código fuente

es el IDE de Arduino, el cual es un entorno de programación de código abierto multiplataforma de fácil uso y tiene un lenguaje de programación basado en C++.

El IDE de Arduino incorpora un conjunto de herramienta de programación por lo cual es considerado un programa de aplicación, está compuesto por un editor de código, un compilador y un depurador, además incorpora las herramientas para cargar el código verificado en la memoria flash del Arduino.

Para el desarrollo del software del proyecto, se implementó un programa que sigue la estructura y secuencia lógica mostrada en la Figura 14 y en el Anexo 5 la programación realizada.

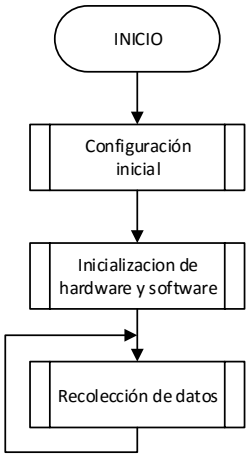


Figura 14. Diagrama de flujo principal

3.6.1.1. Configuración inicial

Para el desarrollo del programa lo primero que se realizo es la configuración inicial, aquí se declaran las librerías y variables necesarias para el correcto funcionamiento del programa.

En la Figura 15 se muestra el diagrama de flujo de la configuración inicial y en el Anexo 5 la programación realizada.

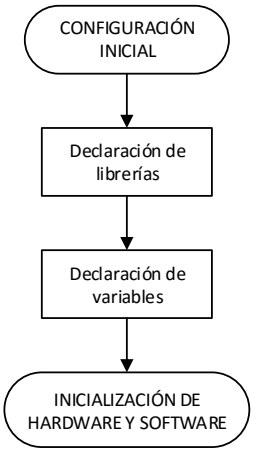


Figura 15. Diagrama de flujo de la configuración inicial

Declaración de librerías

Una librería es un conjunto de funciones que se incorpora en el programa para facilitar la programación, haciendo que el código sea sencillo de realizar y fácil de entender. Las librerías que se utilizaron en el programa se detallan a continuación.

- `SoftwareSerial.h`: Librería necesaria para realizar la comunicación serie desde cualquier entrada o salida digital.
- `TinyGPS.h`: Librería para analizar los flujos de datos NMEA (del inglés *National Marine Electronics Association*) proporcionados por los módulos GPS.
- `SdFat.h`: Esta librería proporciona acceso de lectura / escritura a los sistemas de archivos FAT16 / FAT32 en tarjetas de memoria.
- `I2Cdev.h`: La librería permite realizar la comunicación con dispositivos por medio de I2C.

- `MPU6050.h`: Librería para obtener los datos del acelerómetro y giroscopio del módulo MPU5060.
- `Wire.h`: Esta librería permite comunicarse con dispositivos I2C y es un complemento para la librería I2Cdev.h

Declaración de variables

Las variables deben declararse antes de que puedan ser utilizadas. Para declarar una variable se empieza definiendo su tipo, asignándole siempre un nombre y un valor inicial.

Para el programa de Arduino se declararon las variables con un nombre y tipo de dato de acuerdo a la función que realicen, el código cuenta con variables para el manejo de los datos de la aceleración, obtención de coordenadas geográficas del GPS, para los leds indicadores, el muestreo de los datos, entre otras.

Inicialización de hardware y software

Luego de realizar la configuración inicial, se pasa a la parte de inicialización de hardware y software, aquí se configura los diferentes protocolos de comunicación necesarios para conectar la placa Arduino con el sensor y los módulos, de igual manera se configuran los pines digitales de la placa, para que trabajen como entrada o salida.

En la Figura 16 se muestra el diagrama de flujo de la inicialización de hardware y software

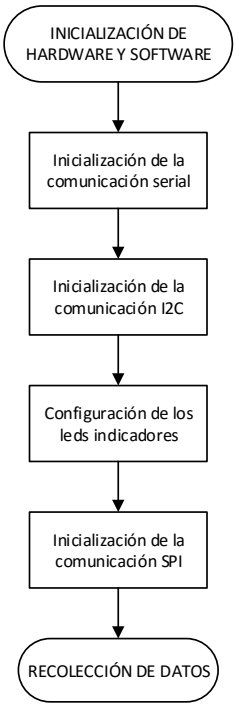


Figura 16. Diagrama de flujo de la inicialización del hardware y software.

El código de la inicialización del hardware y software se muestra en el Anexo 5.

Inicialización de la comunicación serial

La comunicación serial es usada para conectar el controlador con el modulo GPS, los datos técnicos del módulo GPS establecen que la velocidad de transmisión soportada por este es de 9600 baudios, con este parámetro se realiza la configuración del puerto serie de Arduino para trabajar a la misma velocidad y de esta manera establecer la comunicación.

Inicialización de la comunicación I2C

Para establecer la comunicación I2C entre el módulo MPU5060 y el Arduino Nano se tiene que inicializar de la librería `Wire.h` médiate la instrucción `Wire.begin`.

Configuración de los leds indicadores

Se van a utilizar cuatro pines digitales del Arduino Nano, tres pines serán usados para los leds indicadores los cuales muestran la pausa de grabación de los datos, la grabación de los datos y el error en la lectura de la tarjeta de memoria, estos pines estarán configurados como salidas, el otro pin estará configurado como entrada y servirá para controlar el almacenamiento de los datos.

Inicialización de la comunicación SPI

Para realizar la comunicación SPI entre el módulo microSD y el Arduino Nano, se tiene que inicializar de la librería `SdFat.h` médiate la instrucción `SD.begin`.

Recolección de datos

Luego de tener configurado los protocolos de comunicación y los pines digitales, se procede a la recolección de datos, el proceso inicia cuando el switch de control de grabación este activado, luego se comprueba que la tarjeta de memoria se encuentre en la ranura microSD del módulo, caso contrario se indicara un error en la lectura de la tarjeta, una vez verificado esto el programa empieza a recoger los valores leídos por el acelerómetro y el módulo GPS, estos valores son recogidos cada 10 ms es decir que se tomaran 1000 muestras cada segundo así como lo describe (Weiss et al., 2006), una vez que la información haya sido procesada, se almacena en la memorita micro SD.

El programa incorpora salidas digitales para las luces indicadoras, estas permiten saber cuándo el circuito se encuentra en estado de grabación, espera o error.

En la Figura 17 se detalla el proceso de recolección de datos médiate el diagrama de flujo y en el Anexo 5 se observa la programación realizada.

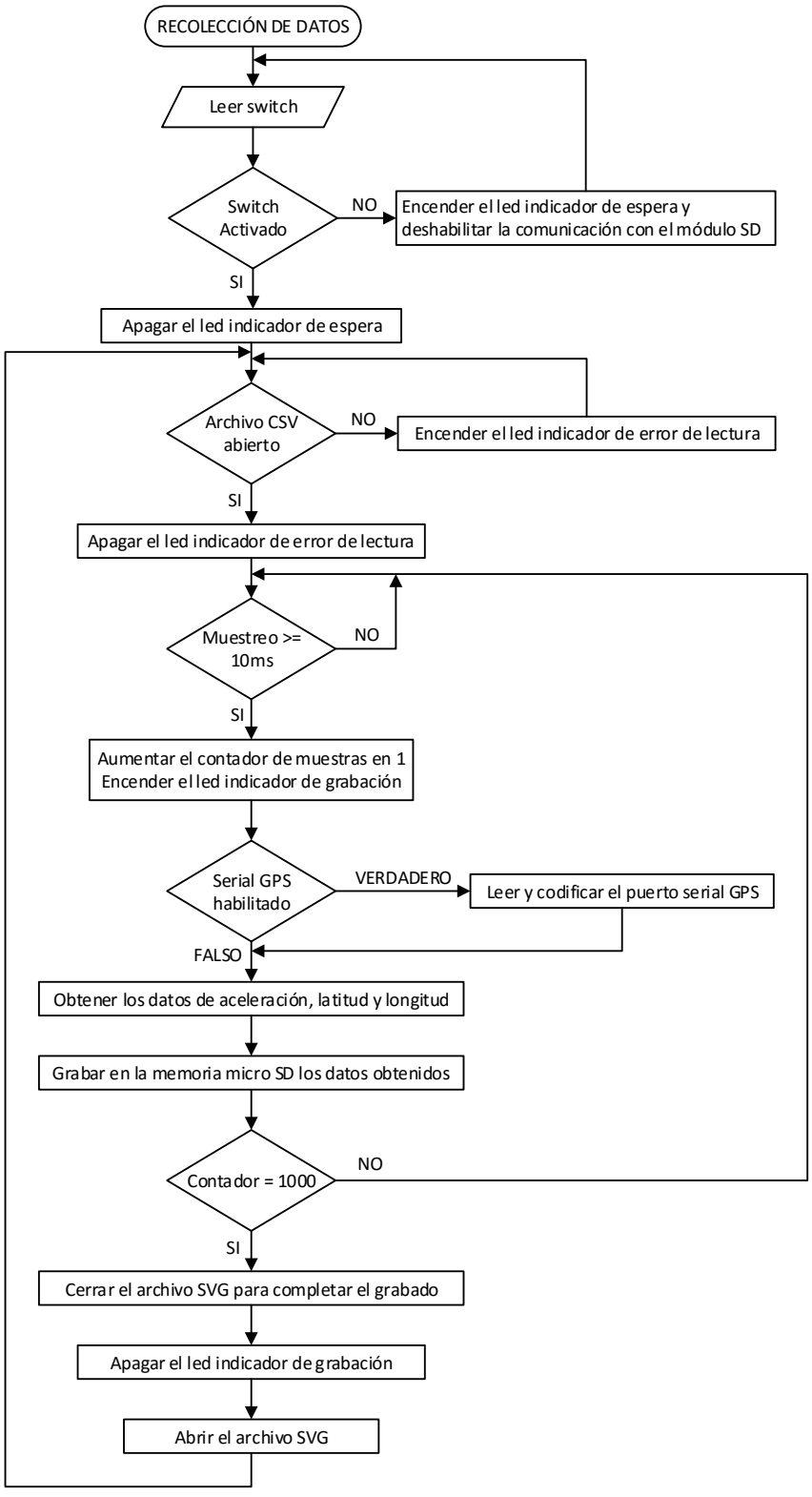


Figura 17. Diagrama de flujo de la recolección de datos

CAPÍTULO 4

IMPLEMENTACIÓN DEL CLASIFICADOR DE TIPOS DE TERRENOS

En este capítulo se detallan los elementos que constituyen la base de datos de la diferente información de los terrenos, se explicará la metodología que se utilizó para extraer y seleccionar las características de la misma, por último, se explicará la configuración que se realizó para obtener el modelo del clasificador.

4.1. Descripción general

Con el hardware presentado en el capítulo anterior se capturo los datos de los diferentes terrenos mencionados en el apartado 1.3, luego estos datos fueron almacenados en una memoria extraíble y llevados hacia un computador para empezar la etapa de entrenamiento del clasificador.

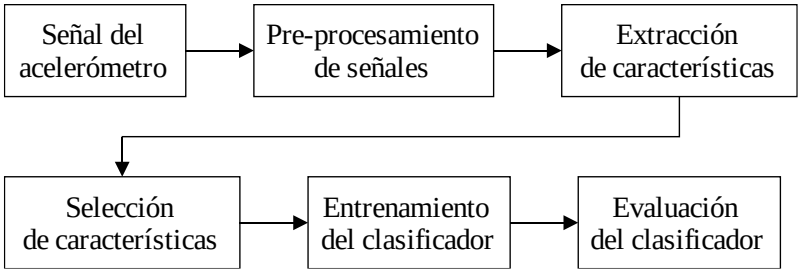


Figura 18. Proceso para la implementación del clasificador de terrenos

En la Figura 18 se muestra de forma general el proceso implementado para el entrenamiento del clasificador de terrenos. Este inicia con la lectura de los datos en su formato original .CSV, que contienen información de la aceleración, latitud y longitud que se adquirió de cada tipo de terreno, dichos datos son leídos por Matlab® de manera automática mediante una función específica.

En la etapa de pre-procesamiento se normalizo la señal, luego esta fue utilizada para realizar la extracción de características en el dominio de la frecuencia mediante la transformada de Fourier, posterior a esto cada una de las características de interés que serán explicadas en el apartado 4.4 se etiquetaron según la clase, con estos datos, se entrenó el clasificador por medio de SVM. Finalmente, los resultados obtenidos se analizaron con las matrices de confusión.

Para realizar la clasificación de los terrenos se utilizó la aplicación *Classification Learner* de Matlab®, esta aplicación cuenta con varios modelos de clasificación, los cuales pueden ser entrenados y evaluados a partir de datos de entrada debidamente etiquetados por clase. Entre los modelos de clasificación están los arboles de decisión, las SVM, la regresión logística, vecinos más cercanos, etc. (MathWorks, s.f.). Para esté proyecto en específico se utilizó las SVM según lo descrito en el apartado 2.7 del capítulo 2.

4.2. Base de datos

La base de datos contendrá los valores de aceleración producida por la vibración que se originó en la bicicleta al atravesar los diferentes terrenos, así como los valores de latitud y longitud obtenidos por el GPS.

La base de datos está constituida por tres columnas, la primera está formada por los valores de aceleración que se adquirieron del sensor, las siguientes columnas contienen los valores de latitud y longitud respectivamente, valores que se consiguieron del GPS. Estos datos fueron recolectados por el hardware explicado en el capítulo anterior, y almacenados en una memoria micro SD, lo que permitió llevar la información hacia un computador. En la Tabla 16 se muestra un ejemplo de la distribución de la base de datos.

Tabla 16

Distribución de la base de datos para un terreno de adoquín.

Aceleración (Amplitud)	Latitud (Grados)	Longitud (Grados)
13.153449	-0.179441	-78.468302
12.470868	-0.179441	-78.468302
10.346486	-0.179441	-78.468302
7.989786	-0.179441	-78.468302

La base de datos está conformada por 400 archivos, los cuales se dividen en cuatro grupos de 100 para cada tipo de terreno. Cada archivo está formado por 1000 muestras, que sumadas son 100000 y 400000 en total, se obtuvo 1000 muestras para cada archivo debido a que la frecuencia de muestreo fue de 1 kHz. De los 100 archivos se utilizó 70 para el entrenamiento del clasificador y 30 para evaluar su desempeño, correspondiente al 70% y 30% respectivamente. Esto se lo realizó para cada tipo de terreno, es decir, que en total se utilizó 280 archivos para el entrenamiento y 120 para probar el desempeño, en la Tabla 17 se muestra la distribución de los archivos con el número de datos correspondientes para cada tipo de terreno.

Tabla 17

Número de muestras de la base de datos

Terreno	Archivos	Muestras	Entrenamiento			Desempeño		
			Archivos	Datos	%	Archivos	Datos	%
Adoquín	100	100000	70	70000	70	30	30000	30
Asfalto	100	100000	70	70000	70	30	30000	30
Césped	100	100000	70	70000	70	30	30000	30
Grava	100	100000	70	70000	70	30	30000	30
Total	400	400000	280	280000	70	120	120000	30

4.2.1. Adquisición y lectura de archivos .CSV

Los datos se organizaron por carpetas con el nombre de cada terreno, y dentro de cada una estarán los archivos necesarios para el entrenamiento y comprobación del clasificador. Para leer los archivos .CSV en Matlab® se utilizó la función `csvread`, esta función es utilizada por el archivo *Feacture_Extraction_1.m* el cual se encuentra en el Anexo 1, `csvread` almacena los valores de los archivos seleccionados en una matriz de 280000x1 ya que se requiere trabajar con los datos de entrenamiento del clasificador, esto permite separar los datos en vectores de 1000x1 para luego realizar un procesamiento y análisis de los mismos, esto se realizará con todos los archivos de los cuatro tipos de terrenos. En la Figura 19 se presenta la señal del adoquín obtenida de uno de los archivos de entrenamiento y está formada por 1000 muestras.

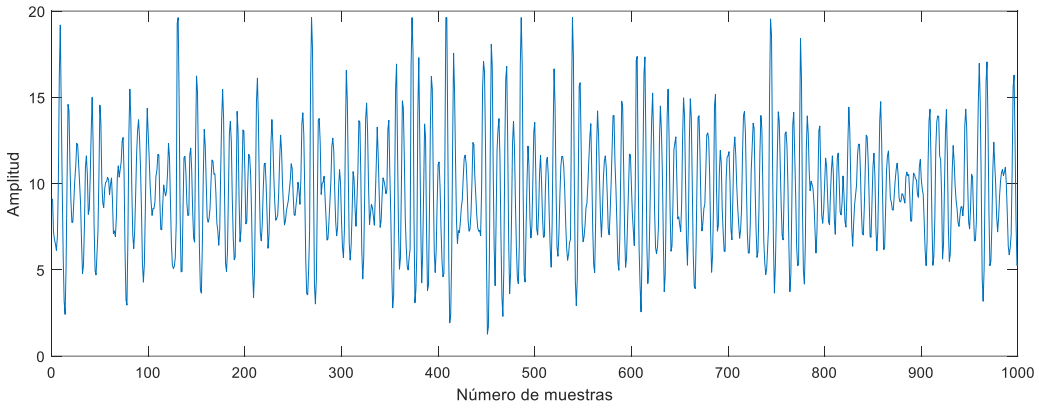


Figura 19. Señal del adoquín compuesta por 1000 muestras.

4.3. Procesamiento de las señales

Antes de empezar a trabajar con las señales primero se realizó un filtrado para eliminar el ruido, Matlab® cuenta con varias herramientas para el procesamiento de señales, en este caso se requería

eliminar el ruido por lo cual se usó la función `filter`, esta herramienta permite filtrar la señal utilizando una función de transferencia racional, la cual está definida por los coeficientes del numerador y denominador. Se optó por un filtro de media móvil ya que es muy utilizado para eliminar el ruido de las señales, este filtro desliza una ventana a lo largo de las muestras para calcular sus promedios y de esta manera obtener una señal menos ruidosa (MathWorks, 2018).

La ecuación (5) define la función para el filtro de media móvil que fue usado en este proyecto.

$$y(n) = \frac{1}{windowSize}\Big(x(n) + x(n-1) + \cdots + x\big(n-(windowSize-1)\big)\Big) \qquad (5)$$

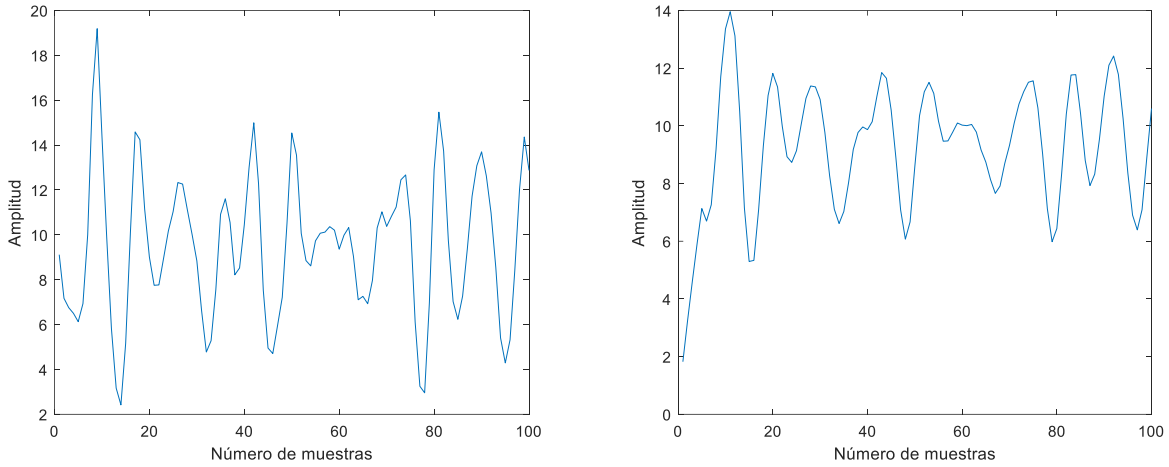
Donde:

$n = Vector\ temporal$

$x = Señal\ de\ entrada$

$y = Salida\ del\ filtro$

El tamaño de la ventana ($windowSize$) es la encargada de modificar el comportamiento del filtro, mientras más grande sea este valor mayor será el suavizado, sin embargo el aumentar el tamaño de la ventana puede traer consecuencias negativas ya que no solo se eliminaría el ruido sino que también parte de la información, es por eso que el tamaño de la ventana depende del tipo de señal que se tenga, sin embargo los valores habituales son entre 3 y 10. Para este proyecto se decidió utilizar un tamaño de ventana igual a 5 muestras. En la Figura 20 se muestra una señal de adoquín sin filtrar y otra aplicada el filtro de media móvil.



a) b)

Figura 20. Filtrado de una señal de adoquín. a) Señal sin filtro, b) Señal filtrada

Luego de eliminar el ruido la señal tiene que ser normalizada, para ello lo primero que se hizo fue eliminar la componente DC, esto se logró restando el valor medio de la señal filtrada, después se la dividió para el valor máximo de la señal, de esta manera se normalizó la señal entre valores de -1 y 1. Con este proceso se llegó a obtener la señal de la Figura 21 la cual fue utilizada para la extracción de características.

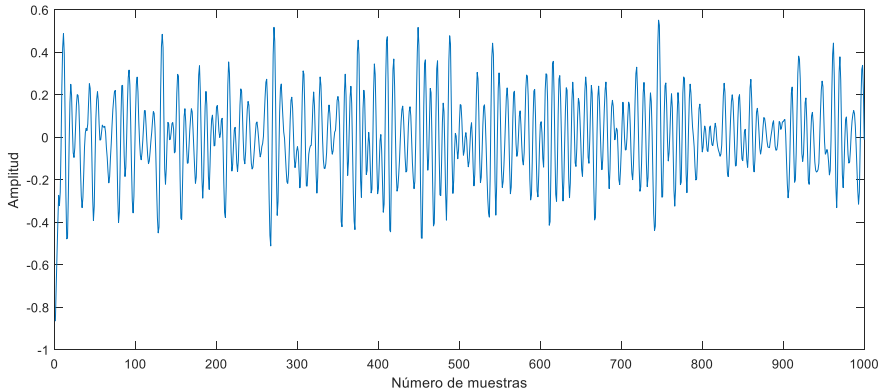


Figura 21. Señal filtrada y normalizada del adoquín.

4.4. Extracción de características

Para que el clasificador funcione de manera correcta se lo debe entrenar con ciertas características de las señales, las cuales ayudan a tomar una decisión de una manera más eficiente. Para este proyecto se extrajeron características tanto en el dominio del tiempo como en la frecuencia para ser evaluadas. De la señal en el tiempo se obtuvo como características la media, varianza, picos y valles, mientras que de la señal en frecuencia se extrajo la densidad espectral, la frecuencia de trabajo y el área bajo la curva. En la Tabla 18 se describe cada una de las características extraídas.

Tabla 18

Características extraídas en el dominio del tiempo y de la frecuencia

Tiempo	
Características	**Descripción**
Media	$\bar{X} = \frac{\sum(x)}{n}$
Varianza	$S^2 = \frac{\sum(X - \bar{X})^2}{n - 1}$
Promedio de Picos	$\overline{X_{picos}} = \frac{\sum(Picos)}{Número\ de\ picos}$
Promedio de Valles	$\overline{X_{valles}} = \frac{\sum(Valles)}{Número\ de\ valles}$
Frecuencia	
Características	**Descripción**
Densidad espectral	$x(k) = \sum_{n=0}^{N-1} x(n) \cdot e^{\frac{-j \cdot 2 \cdot \pi \cdot k \cdot n}{N}}$
Frecuencia de trabajo	$\max[F(w)]$
Área bajo la curva	$\text{trapz}[F(w)]$

- Pico: es la amplitud o valor máximo de la señal.
- Valle: es el pico negativo o valor mínimo de la señal.
- max: es una función de Matlab® utilizada para encontrar el valor máximo de la señal.
- trapz: es una función de Matlab® que sirve para encontrar el área bajo la curva de una señal.

En la Tabla 19 se muestra un fragmento de las características que se obtuvieron para el terreno de adoquín, los valores de cada fila se adquirieron a partir de 1000 muestras agrupadas en un archivo CSV, se tiene 70 archivos para cada tipo de terreno, es decir que en total se tiene 280 archivos a los cuales se les extrajo las características, estas serán usadas en la siguiente sección.

Tabla 19

Características extraídas para un terreno de adoquín

Media	Varianza	Promedio de Picos	Promedio de Valles	Densidad espectral	Frecuencia de trabajo (Hz)	Área bajo la curva (u^2)
0.0009	0.0371	0.2108	-0.2141	0.0057	11.7188	0.0351
-0.0015	0.0367	0.2167	-0.2250	0.0071	11.5234	0.0364
-0.0136	0.0267	0.1642	-0.1936	0.0068	11.7188	0.0273
0.0063	0.0363	0.2257	-0.2197	0.0083	11.5234	0.0363
0.0011	0.0395	0.2277	-0.2351	0.0076	11.7188	0.0371

Una vez extraídas las características en tiempo y frecuencia, se realizó pruebas para seleccionar las que mejor se adapten al sistema de clasificación; las cuales consistieron en realizar el entrenamiento del clasificador con las características seleccionadas tanto en tiempo como en frecuencia. En el capítulo 5 de pruebas y resultados se detalla este proceso de mejor manera.

4.5. Configuración del clasificador

Antes de realizar la configuración del clasificador es importante mencionar que se trabajó con los datos de entrenamiento de los cuatro tipos de terreno correspondientes a 70x4 = 280 archivos, con esto se logró obtener 280 datos de cada una de las características seleccionadas, en la Figura 22 se muestra como se obtienen las características para un terreno de adoquín, este proceso se repite para cada uno de los terrenos y así se obtienen los 280 datos.

70 archivos para el entrenamiento del terreno de adoquín

Archivos	1	2	3	4	···	70
Muestras	1000	1000	1000	1000	···	1000

Archivo 1	Vector de características 1
Archivo 2	Vector de características 2
Archivo 3	Vector de características 3
⋮	
Archivo 70	Vector de características 70

Figura 22. Vectores de características para un terreno de adoquín

Además, se tuvo que etiquetar los datos extraídos, esto se logró por medio de un código el cual se encuentra en el Anexo 2. En el código primero se creó una variable "`x`" la cual almacena las características extraídas, luego se creó otra variable llamada "`y`" esta contiene las etiquetas de cada clase, seguido a esto se convirtió la variable "`y`" a tipo categorial, por ultimo con ayuda del comando `table` se logró incorporar las características con las etiquetas de clase en una sola variable denominada "`z`", la cual fue utilizada para realizar el entrenamiento del clasificador. Esta variable quedo conformada por dos columnas, la una contiene las características y la otra las

etiquetas, obteniendo un tamaño para la variable "z" de 280x2, en la Tabla 20 se muestra la distribución de las características y las etiquetas.

Tabla 20

Fragmento de la tabla de características

Características ("x")			Etiquetas ("y")
0.0056991	11.719	0.035083	ADOQUIN
0.0071324	11.523	0.036411	ADOQUIN
0.006771	11.719	0.027284	ADOQUIN
0.0083059	11.523	0.036261	ADOQUIN

En el apartado 4.1 se mencionó que para realizar el entrenamiento del clasificador se usó la aplicación *Classification Learner* de Matlab® y a continuación se describen los paso que se realizaron para lograr este objetivo.

4.5.1. Acceso a la aplicación

Se usó Matlab®, y se accedió a la pestaña APPS la cual se encuentra en la parte superior del software, una vez ahí se observaron varias aplicaciones, de las cuales se seleccionó la que tiene el nombre *Classification Learner*.

Al acceder a la aplicación se desplego una ventana principal con algunas herramientas necesarias para el entrenamiento y evaluación del modelo de clasificación, las cuales se pueden observar en la Figura 23.

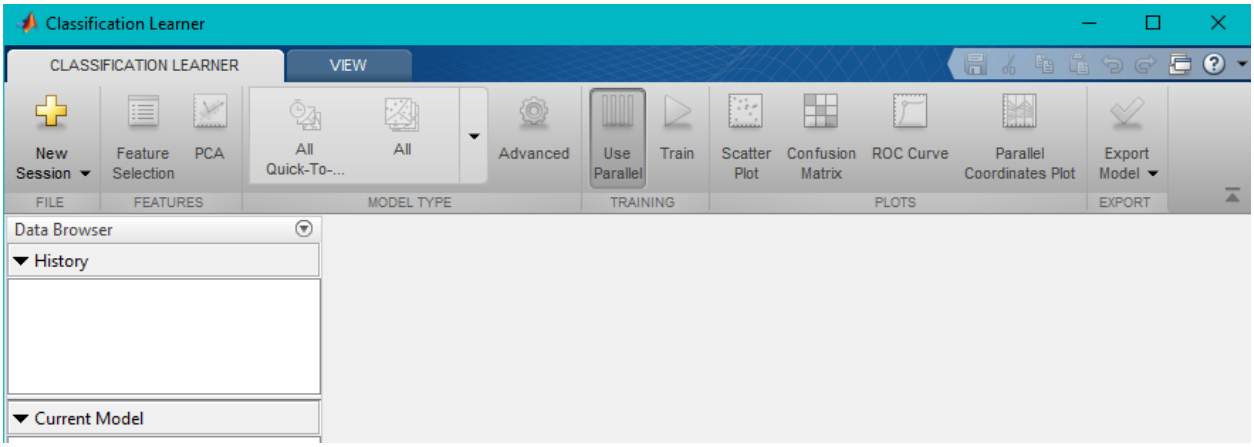


Figura 23. Ventana principal de la aplicación

4.5.2. Nueva sesión

Al acceder a la pestaña *New Session* se desplego una ventana con varias opciones como se muestra en la Figura 24.

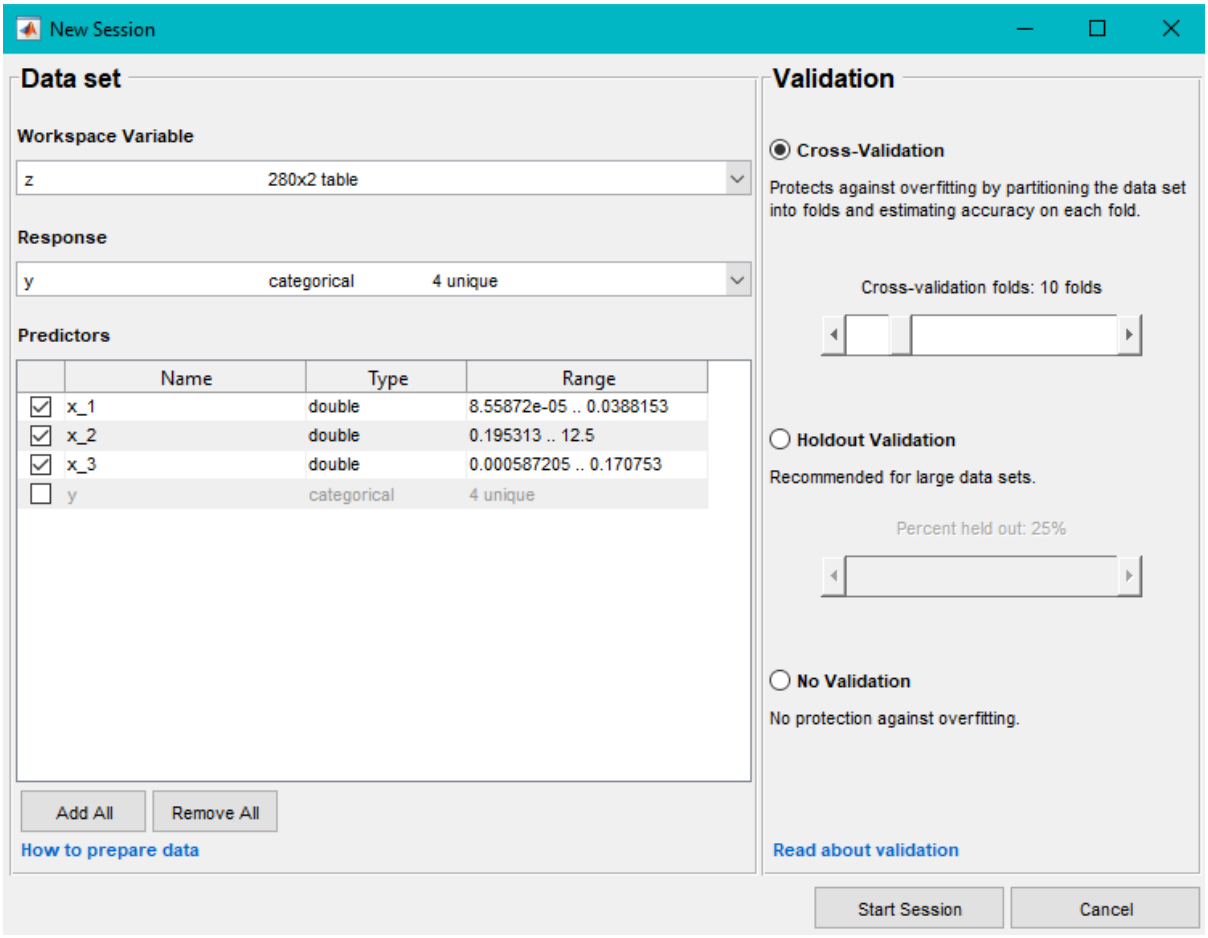


Figura 24. Ventana de configuración de una nueva sesión

La configuración que se siguió para crear una nueva sesión es la siguiente:

- En la etiqueta *Workspace Variable* se escogió la variable que contiene la información necesaria para el entrenamiento del clasificador, en el apartado 4.6 se indicó que el nombre de la variable es "`z`".
- En *Cross-Validation* (revisar apartado 2.5) se escogió un valor de 10 ya que (Refaeilzadeh, Tang & Liu, 2008) en su artículo mencionan que en el aprendizaje automático la validación cruzada con 10 iteraciones es lo más común.

- Por ultimo para iniciar la sesión se dio clic en el botón *Start Session.*

4.5.3. Entrenamiento del algoritmo

La aplicación *Classification Learner* incorpora varios algoritmos de aprendizaje, de los cuales se utilizó SVM debido a que en trabajos anteriores dio buenos resultados, para mayores detalles diríjase al apartado 2.6. Para estimar qué tipo de SVM arroja mejores resultados se trabajó con Linear SVM, Quadratic SMV y Cubic SVM.

Luego de ver seleccionado los algoritmos de aprendizaje se los debe entrenar, este proceso se lo realizo dando un clic en el botón *Train,* esto se realiza para cada algoritmo escogido.

La aplicación empezó a realizar el entrenamiento para cada algoritmo, y al finalizar esta operación se desplegaron los resultados en la parte izquierda de la ventana donde se mostró la exactitud de cada uno. En la Tabla 21 se exponen los resultados que se obtuvieron.

Tabla 21
Exactitud de los modelos de clasificación

Modelo SVM	Exactitud
Linear	93.2%
Quadratic	92.1%
Cubic	91.8%

4.6. Interfaz grafica

Para probar el desempeño del clasificador se creó una interfaz gráfica de usuario utilizando el comando `GUIDE` de Matlab®, esta interfaz ayudará a visualizar de manera sencilla la señal del terreno que se está analizando, así como la posición del mismo en un mapa, ya que uno de los

requerimientos para la creación de la misma era el poder observar el posicionamiento georeferenciado de los terrenos que fueron muestreados. En el capítulo 5 la GUI será de gran ayuda para obtener los resultados de evaluación de cada tipo de terreno y obtener así porcentajes de predicción de los mismos.

En la Figura 25 se muestra la GUI, esta cuenta con dos botones, el primero se utiliza para abrir y buscar el archivo que se quiere analizar, el segundo botón sirve para iniciar la clasificación una vez escogido el archivo. Los resultados se visualizan de dos formas, la una mediante un gráfico que muestra la forma de la señal y la otra mediante texto el cual indica que tipo de terreno que se reconoció, además se muestra una figura que contiene el mapa y sector en donde se realizó la prueba del terreno analizado.

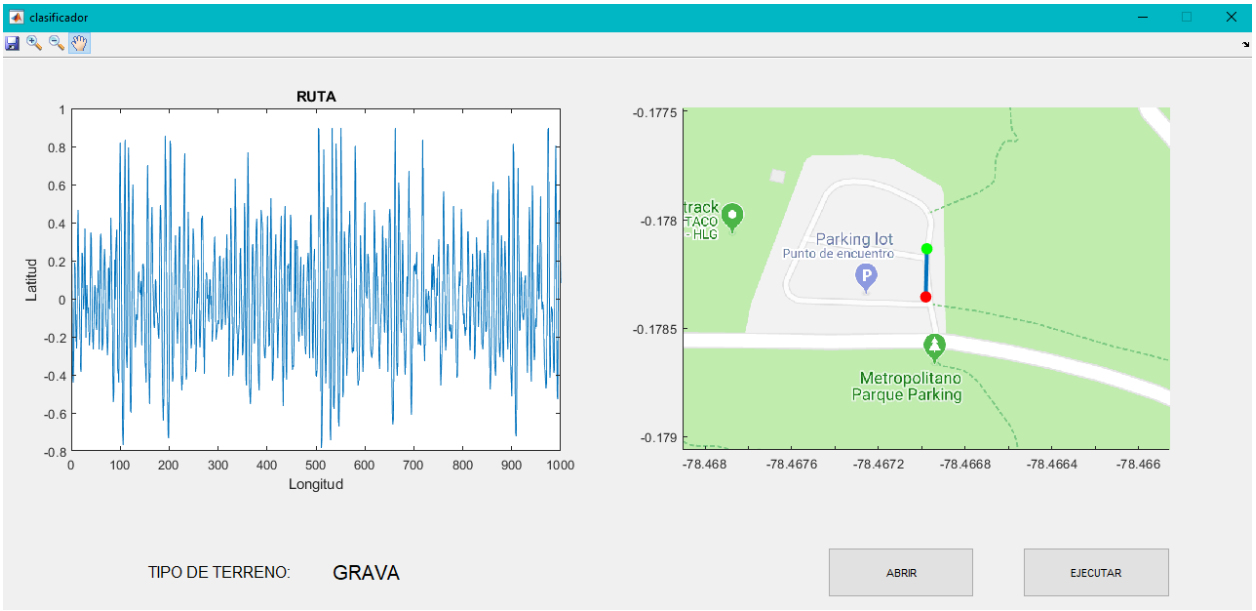


Figura 25. Interfaz gráfica de usuario

CAPÍTULO 5

PRUEBAS Y RESULTADOS

En este capítulo se muestran las pruebas realizadas para escoger las características que mejor se adapten al sistema (características temporales y de frecuencia), una vez escogidas las características, se muestran los modelos de algoritmos seleccionados para realizar el entrenamiento del clasificador, así mismo se explica cuál fue el seleccionado para utilizarlo en las pruebas de campo.

En las pruebas realizadas se utilizó el 30% del total de los datos recolectados para cada terreno lo que corresponde a 30 archivos con 1000 muestras cada uno, estos archivos fueron evaluados por el algoritmo de clasificación obteniendo como resultado predicciones correctas e incorrectas para cada tipo de terreno, estos valores se los va poder observar a lo largo del capítulo.

Por último, se realizó una prueba de campo, la cual consistió en tomar muestras de los diferentes terrenos que se encuentran en el campus de la Escuela Politécnica Nacional para luego ser evaluados por el algoritmo de clasificación seleccionado, arrojando como resultado un gráfico en el cual se puede observar la ruta y los diferentes terrenos que se pudieron predecir.

5.1. Pruebas para seleccionar las características extraídas

Se evaluaron las características del tiempo y de la frecuencia de tres maneras, en la primera se utilizaron las características tanto del tiempo como de la frecuencia, en la segunda solo las características del tiempo y en la tercera se utilizaron las de frecuencia, luego se procede a analizar cada una de estas mediante el entrenamiento del algoritmo de clasificación para poder observar cual arroja mayor precisión al momento de clasificar los terrenos.

Tabla 22

Precisión de los clasificadores con características de tiempo y frecuencia

Características	Exactitud
Tiempo y frecuencia	92.8%
Tiempo	91.8%
Frecuencia	**93.2%**

En la Tabla 22 se muestra los valores de precisión que se obtuvieron para cada caso, dando como resultado que al utilizar las características temporales y de frecuencia se consigue una precisión de 92.8%, para las cuatro características que se adquirieron en el dominio del tiempo el clasificador arrojo un valor de precisión de un 91.8%, mientras que con las tres características que se adquirieron en el dominio de la frecuencia arrojo un valor de 93.2%, es así que después de realizar las pruebas de entrenamiento, se eligió trabajar con las características del domino de frecuencia, ya que se tuvo mejores resultados, además al utilizar un menor número de características se logró reducir los tiempos de procesamiento, tanto en el entrenamiento como en las pruebas del clasificador.

5.2. Algoritmo del modelo 1

Para obtener el modelo 1 se utilizó un sistema lineal de SVM, compuesto por una función kernel tipo lineal y un método multi-clase Uno-contra-Uno (del inglés, *One-vs-One*, OVO). El entrenamiento se realizó para cuatro tipos de terrenos (adoquín, asfalto, césped, grava) de los cuales se extrajeron tres características (Densidad espectral, frecuencia de trabajo, área bajo la curva) que fueron descritas en el apartado 4.5, el entrenamiento del modelo 1 arrojo una exactitud del 93.2%. El cual se muestra en la Tabla 23.

Tabla 23

Exactitud del modelo 1

Modelo SVM	Exactitud
Linear	93.2%

En la matriz de confusión de la Figura 26 se puede observar los porcentajes de aciertos y fallas de los terrenos, mostrando que del 100% de los datos de adoquín, el sistema predijo que el 97% son adoquín y el 3% césped, además se observa que del 100% de los datos de asfalto, el sistema predijo que el 99% son asfalto y el 1% césped, asimismo del 100% de los datos de césped, el sistema predijo que el 90% son césped, el 3% es adoquín y el 7% es grava, por ultimo del 100% de los datos de grava, el sistema predijo que el 90% son grava, que el 1% es adoquín y el 9% césped. En la Tabla 24 se muestra los porcentajes de aciertos y fallas totales para cada terreno.

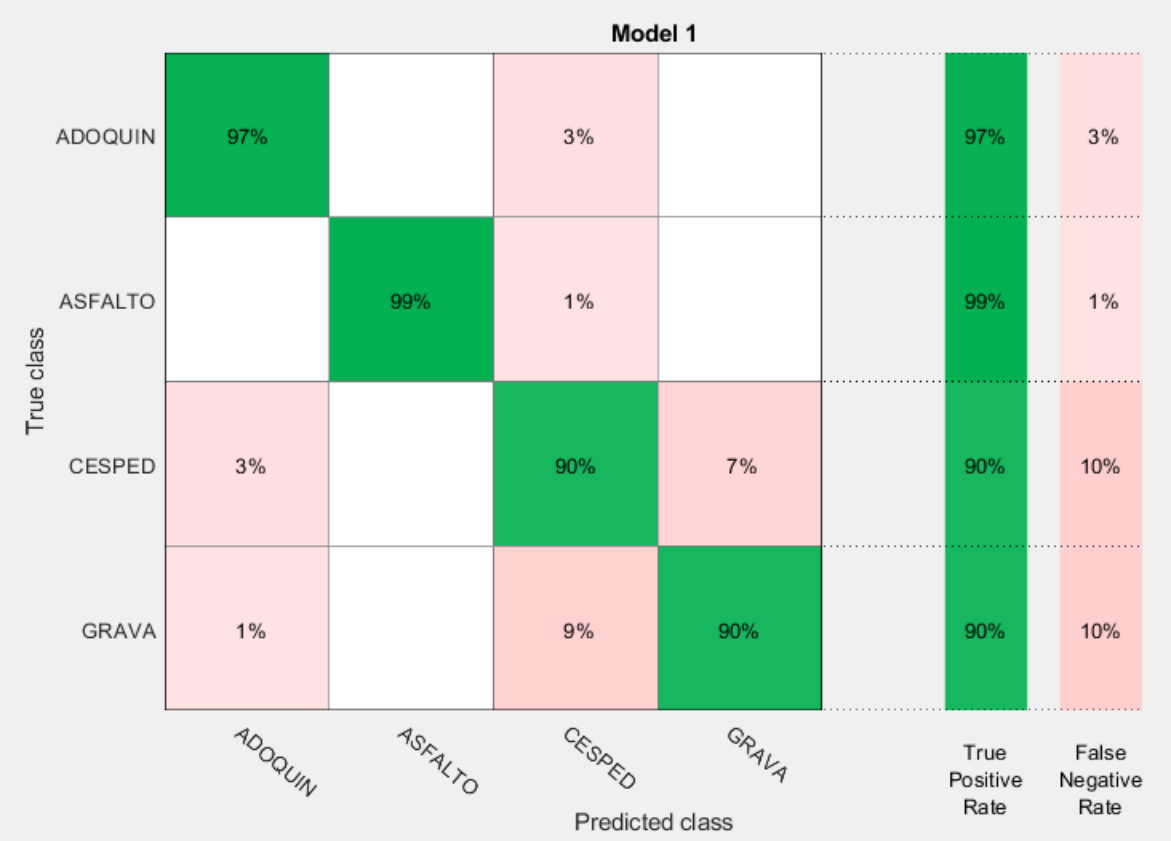


Figura 26. Matriz de confusión del modelo 1

Tabla 24

Porcentaje de aciertos y fallas

Terrenos	**Aciertos**	**Fallas**
Adoquín	97%	3%
Asfalto	99%	1%
Césped	90%	10%
Grava	90%	10%

5.3. Algoritmo del modelo 2

Para obtener el modelo 2 se utilizó un sistema cuadrático de SVM, compuesto por una función kernel tipo cuadrático y un método multi-clase One vs One. El entrenamiento se realizó para cuatro tipos de terrenos (adoquín, asfalto, césped, grava) de los cuales se extrajeron tres características (Densidad espectral, frecuencia de trabajo, área bajo la curva) que fueron descritas en el apartado 4.5, el entrenamiento del modelo 2 arrojo una exactitud del 92.1%. El cual se muestra en la Tabla 25.

Tabla 25

Exactitud del modelo 2

Modelo SVM	Exactitud
Quadratic	92.1%

En la matriz de confusión de la Figura 27 se puede observar los porcentajes de aciertos y fallas de los terrenos, mostrando que del 100% de los datos de adoquín, el sistema predijo que el 97% son adoquín y el 3% césped, además se observa que del 100% de los datos de asfalto, el sistema predijo que el 97% son asfalto y el 3% adoquín, asimismo del 100% de los datos de césped, el sistema predijo que el 87% son césped, el 3% es adoquín y el 10% es grava, por ultimo del 100% de los datos de grava, el sistema predijo que el 87% son grava, que el 1% es adoquín, el 1% asfalto y el 10% césped. En la Tabla 26 se muestra los porcentajes de aciertos y fallas totales para cada terreno.

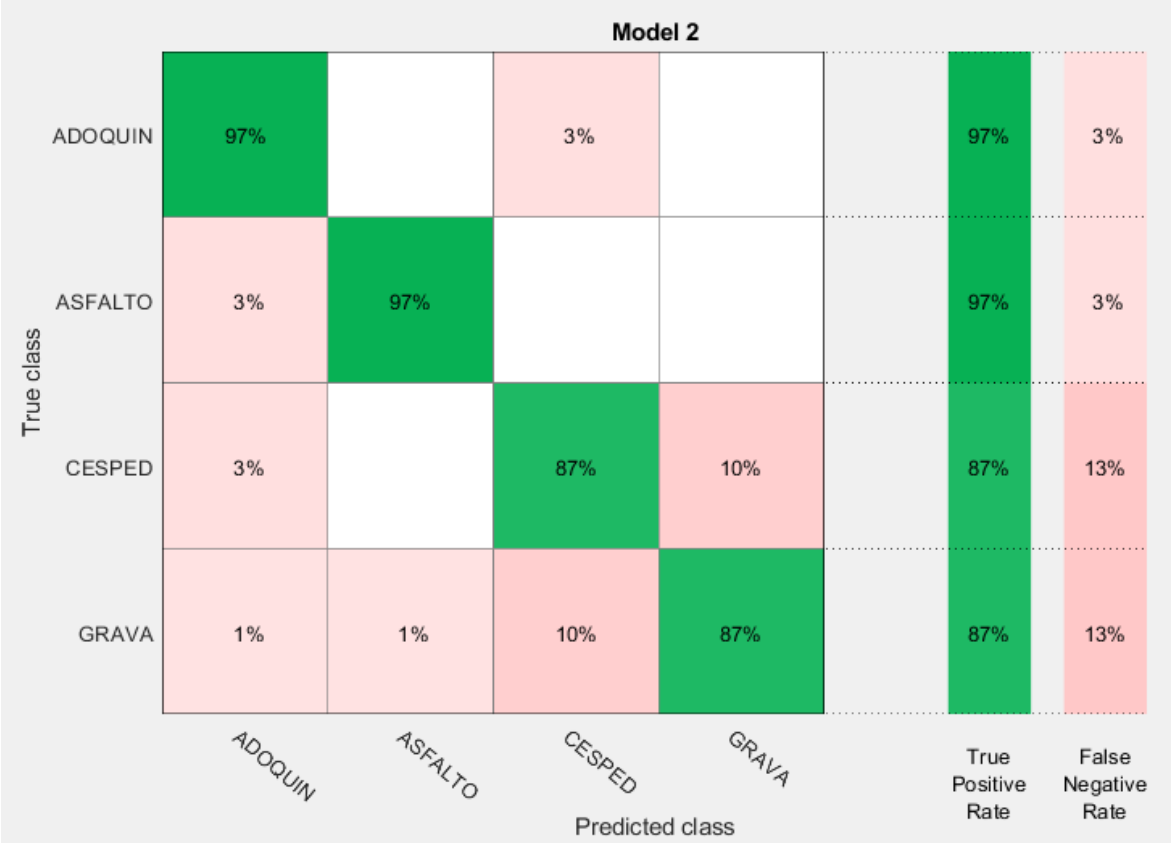


Figura 27. Matriz de confusión del modelo 2

Tabla 26

Porcentaje de aciertos y fallas

Terrenos	**Aciertos**	**Fallas**
Adoquín	97%	3%
Asfalto	97%	3%
Césped	87%	13%
Grava	87%	13%

5.4. Algoritmo del modelo 3

Para obtener el modelo 3 se utilizó un sistema lineal de SVM, compuesto por una función kernel tipo cubica y un método multi-clase One vs One. El entrenamiento se realizó para cuatro tipos de terrenos (adoquín, asfalto, césped, grava) de los cuales se extrajeron tres características (Densidad de espectral, frecuencia de trabajo, área bajo la curva) que fueron descritas en el apartado 4.5, el entrenamiento del modelo 3 arrojo una exactitud del 91.8%. El cual se muestra en la Tabla 27.

Tabla 27

Exactitud del modelo 3

Modelo SVM	Exactitud
Cubic	91.8%

En la matriz de confusión de la Figura 28 se puede observar los porcentajes de aciertos y fallas de los terrenos, mostrando que del 100% de los datos de adoquín, el sistema predijo que el 96% son adoquín, el 3% césped y el 1% grava, además se observa que del 100% de los datos de asfalto, el sistema predijo que el 97% son asfalto, el 1% adoquín y el 1% césped, asimismo del 100% de los datos de césped, el sistema predijo que el 84% son césped, el 3% es adoquín y el 13% es grava, por ultimo del 100% de los datos de grava, el sistema predijo que el 93% son grava y el 7% césped. En la Tabla 28 se muestra los porcentajes de aciertos y fallas totales para cada terreno.

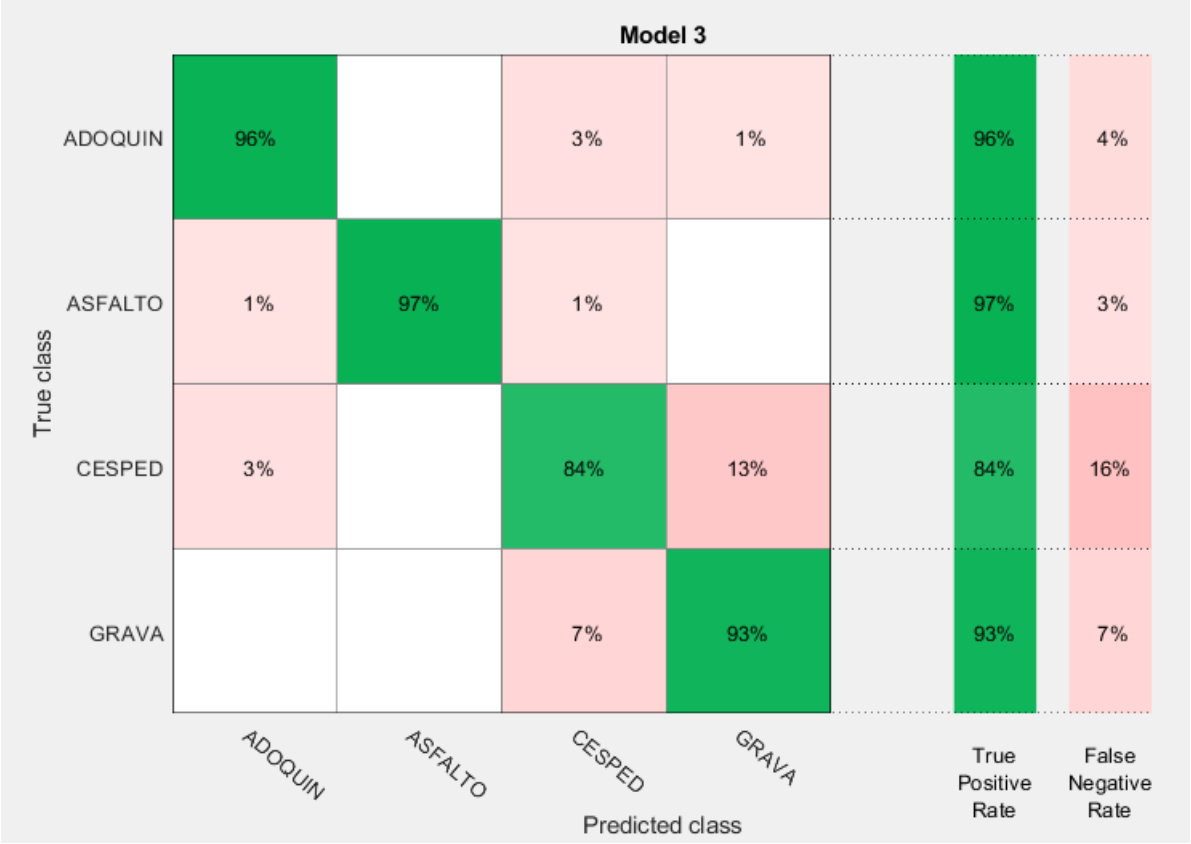


Figura 28. Matriz de confusión del modelo 3

Tabla 28

Porcentaje de aciertos y fallas

Terrenos	**Aciertos**	**Fallas**
Adoquín	97%	3%
Asfalto	97%	3%
Césped	84%	16%
Grava	93%	7%

5.5. Evaluación del algoritmo

Luego del entrenamiento de los diferentes modelos de clasificación se decidió utilizar el modelo 1 para realizar las pruebas y evaluaciones de campo, ya que posee el mayor porcentaje de exactitud.

La base de datos quedó conformada por 400 archivos para los cuatro tipos de terrenos como está especificado en la Tabla17 del apartado 4.2, se dividió la base de datos en 30% para pruebas y 70% para entrenamiento, correspondiente a 120 y 280 archivos respectivamente, estos fueron utilizados para obtener los modelos descritos anteriormente y una vez que el modelo fue determinado, el 30% de los datos que se mencionó anteriormente serán utilizados para la evaluación del algoritmo. A continuación, se presenta la evaluación del algoritmo con cada tipo de terreno.

5.5.1. Evaluación del algoritmo con adoquín

Los datos se los recolecto en la Av. Amazonas, parque La Carolina y parque Metropolitano de la ciudad de Quito, Ecuador, el terreno analizado en este apartado es adoquín, ver Figura 29. El 30% de los datos para la evaluación del clasificador corresponde a 30 archivos tipo .CSV cada uno con 1000 muestras, al comprobar el desempeño del algoritmo para el adoquín se obtuvo 28 datos acertados y 2 erróneos de los cuales uno corresponde a césped y otro a grava, teniendo una eficiencia del 93,3% para clasificar este tipo de terreno, en la Tabla 29 se muestra las predicciones correctas y erróneas con sus porcentajes.

Figura 29. Tipo de terreno adoquín

Tabla 29

Evaluación del algoritmo de clasificación para el adoquín

Muestras	Terreno	Predicción	Porcentaje
28	Adoquín	Correcto	93.3%
1	Césped	Incorrecto	3.3%
1	Grava	Incorrecto	3.3%

5.5.2. Evaluación del algoritmo con asfalto

Los datos se los recolecto en el parque Metropolitano, en la calle Ladrón de Guevara y calle Alfonso Perrier, el terreno analizado en este apartado es asfalto, ver Figura 30. Al comprobar el

desempeño del algoritmo se obtuvo 30 datos acertados, teniendo una eficiencia del 100% para clasificar el asfalto como se puede ver en la Tabla 30.

Figura 30. Tipo de terreno asfalto

Tabla 30

Evaluación del algoritmo de clasificación para el asfalto

Muestras	Terreno	Predicción	Porcentaje
30	Asfalto	Correcto	100%

5.5.3. Evaluación del algoritmo con césped

Los datos se los recolecto en el parque Metropolitano, parque de Guapulo, parque La Carolina y en el estadio de la Universidad Politécnica Nacional, el terreno analizado en este apartado es césped, ver Figura 31. Al comprobar el desempeño del algoritmo se obtuvo 30 datos acertados, teniendo una eficiencia del 100% para clasificar el césped como se puede ver en la Tabla 31.

Figura 31. Tipo de terreno césped

Tabla 31

Evaluación del algoritmo de clasificación para el césped

Muestras	Terreno	Predicción	Porcentaje
30	Césped	Correcto	100%

5.5.4. Evaluación del algoritmo con grava

Los datos se los recolecto en el parque Metropolitano, el terreno analizado en este apartado es grava, ver Figura 32. Al comprobar el desempeño del algoritmo para la grava se obtuvo 26 datos acertados y 4 erróneos de los cuales tres corresponden al césped y otro al adoquín, teniendo una eficiencia del 86.6% para clasificar este tipo de terreno, en la Tabla 32 se muestra las predicciones correctas y erróneas con sus porcentajes.

Figura 32. Tipo de terreno grava

Tabla 32

Evaluación del algoritmo de clasificación para la grava

Muestras	Terreno	Predicción	Porcentaje
26	Grava	Correcto	86.6%
3	Césped	Incorrecto	10%
1	Adoquín	Incorrecto	3.3%

5.5.5. Evaluación del algoritmo en campo

La última prueba que se realizó para comprobar la eficiencia del clasificador fue una de campo, la cual consistió en tomar muestras de los distintos tipos de terrenos alrededor de un campus, el lugar que se eligió para esto fue el campus de la Escuela Politécnica Nacional debido a que cuenta con los tipos de terrenos que se desean analizar, los datos obtenidos del GPS se utilizaran para georreferenciar en un mapa como se puede apreciar en la Figura 33, para luego comparar los

diferentes tipos de terreno en cada tramo con los datos obtenidos. En la Figura 33 se observa los distintos terrenos que reconoció el clasificador.

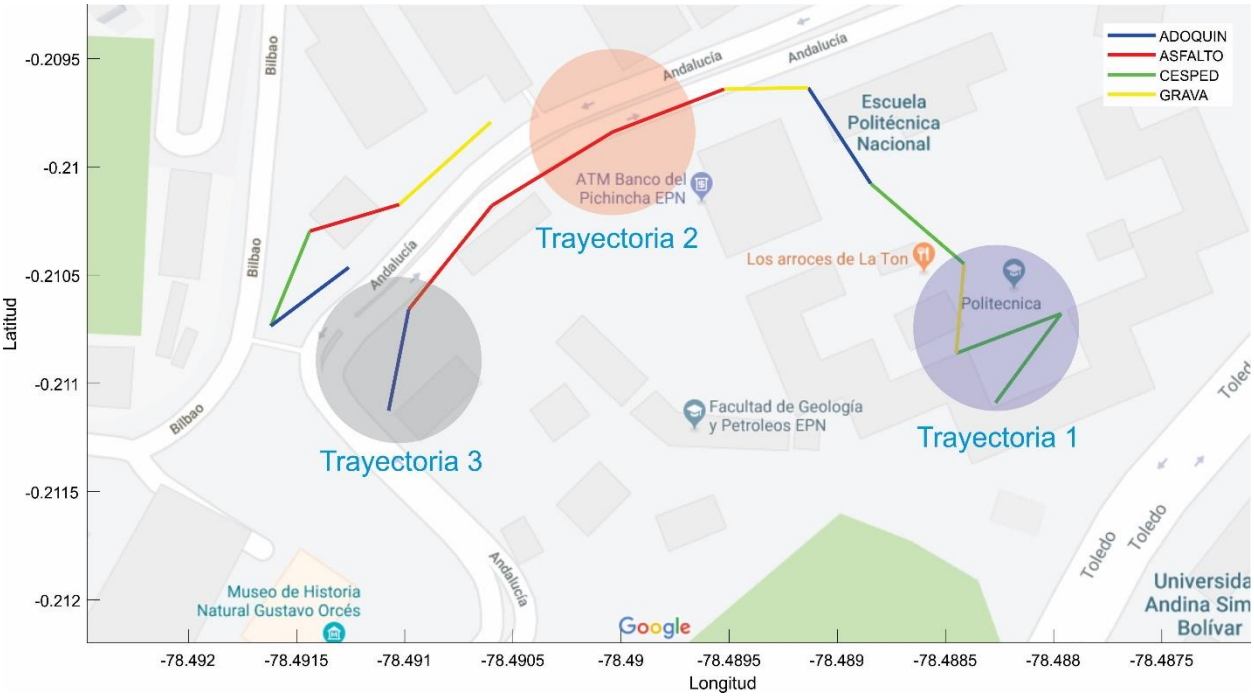


Figura 33. Terrenos reconocidos por el clasificador

Del total de las muestras que se recogió tres dieron un error en la clasificación, esto debido al mal estado de los terrenos, para el resto de datos el algoritmo clasifico cada uno de los terrenos sin errores.

En la trayectoria 1 de la Figura 33 se observa que el sistema clasifico los terrenos como césped, en la Figura 34 se confirma que el tipo de suelo clasificado es correcto.

Figura 34. Tipo de terreno de la trayectoria 1

En la trayectoria 2 de la Figura 33 se observa que el sistema clasifico los terrenos como asfalto, en la Figura 35 se confirma que el tipo de suelo clasificado es correcto.

Figura 35. Tipo de terreno de la trayectoria 2

En la trayectoria 3 de la Figura 33 se observa que el sistema clasifico el terreno como adoquín, en la Figura 36 se confirma que el tipo de suelo clasificado es correcto.

Figura 36. Tipo de terreno de la trayectoria 3

CAPÍTULO 6

CONCLUSIONES Y RECOMENDACIONES

6.1. Conclusiones

Se logró cumplir con el objetivo principal del proyecto el cual consistía en desarrollar un sistema de clasificación de terrenos empleado técnicas de aprendizaje de máquina, mediante un análisis cuantitativo se determinó que el modelo de calificador que mejor se adapta a este proyecto es Linear SVM, debido a que presenta una mayor exactitud con respecto a los otros dos modelos que se compararon, presentando un valor de 93.2% de precisión.

Después de elegir el modelo de clasificación que mejor se adaptó al sistema, se procedió a realizar las pruebas correspondientes mediante técnicas de aprendizaje supervisado, obteniendo como resultado diferentes valores de desempeño para cada tipo terreno, estos valores fueron de un 93.3% para el adoquín, un 100% para el asfalto, 100% para el césped y 86.6% para grava, lo que demuestra que el sistema de clasificación tiene una buena eficiencia.

Se comprobó mediante las pruebas realizadas en el capítulo 5 que el algoritmo Linear SVM tiene un buen desempeño al momento de clasificar los terrenos entrenados, pero presenta fallas cuando el suelo se encuentra en mal estado, un ejemplo de esto es el césped y el asfalto ya que en el primero al estar en mal estado se estaría tomando datos de la tierra siendo una superficie diferente, lo mismo sucede con el asfalto pues al estar este desgastado cambia su característica de superficie suave a una más áspero.

La interfaz gráfica desarrollada es de fácil manejo, además permite una vista de resultados clara y concisa. Haciéndola amigable para el usuario.

6.2. Recomendaciones

El desempeño del clasificador para el césped fue el más bajo, esto debido a que su superficie cambia dependiendo el tamaño del pasto, por eso es recomendable que los datos sean tomados en una superficie uniforme para obtener mejores resultados en la clasificación.

Para complementar este trabajo de investigación, se recomienda utilizar otros tipos de terreno para el entrenamiento del modelo de clasificación y así obtener un sistema más completo, permitiendo clasificar más de los cuatro tipos de terrenos que se seleccionaron en este proyecto.

En este proyecto se utilizó SVM como técnica de clasificación ya que es muy utilizado para resolver este tipo de problemas, se propone utilizar otros modelos y formas de clasificación como redes neuronales, vecinos más cercanos entre otros y así poder comparar los resultados obtenidos.

BIBLIOGRAFÍA

Ellutta, P., Manduchi, R., Matthies, L., Owens, K., & Rankin, A., (2000). Terrain perception for DEMO III. In: Proc. Intelligent Vehicles Symposium, 2000. IV 2000. Proceedings of the IEEE. IEEE, pp. 326 - 331.

Brooks, C., Iagnemma, K., & Dubowsky, S., (2005). Vibration-based terrain analysis for mobile robots. In: Proc. Proceedings of the 2005 IEEE International Conference on Robotics and Automation. IEEE, pp. 3415 - 3420.

Castano, R., Manduchi, R., & Fox, J., (2001), Classification experiments on real-world texture. 1 - 20.

DuPont, E., Collins, E., Coyle, E., & Roberts, R., (2008), Terrain classification using vibration sensors: theory and methods. New Research on Mobile Robotics. 1 - 41.

DuPont, E., Moore C., & Roberts, R., (2008), Terrain Classification for Mobile Robots Traveling at Various Speeds: An Eigenspace Manifold Approach. 2008 IEEE International Conference on Robotics and Automation. IEEE, pp. 3284–3289.

Jitpakdee, R., & Maneewarn, T., (2008). Neural Networks Terrain Classification using Inertial Measurement Unit for an Autonomous Vehicle. SICE Annual Conference 2008. 554 – 558.

Khaleghian, S., & Taheri, S., (2017). Terrain classification using intelligent tire. Journal of Terramechanics 71. 15–24.

Larson, A.C., Voyles, R.M., & Demir, G.K., (2004). Terrain classification through weakly-structured vehicle/terrain interaction. In: Proc. 2004 IEEE International Conference on Robotics and Automation. ICRA'04. IEEE, pp. 218–224.

Lee, S.Y., & Kwak, D.M., (2011). A terrain classification method for UGV autonomous navigation based on SURF. In: Proc. 2011 8th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI). IEEE, pp. 303–306.

Manduchi, R., Castano, A., Talukder, A., & Matthies, L., (2005). Obstacle detection and terrain classification for autonomous off-road navigation. Autonomous Robots 18, 81 - 102.

Parka, B., Kimb, J., & Lee, J., (2012). Terrain feature extraction and classification for mobile robots utilizing contact sensors on rough terrain. Procedia Engineering 41, 846 - 853.

Ojeda, L., Borenstein, J., Witus, G., & Karlsen, R., (2006). Terrain characterization and classification with a mobile robot. J. Field Robot. 23 (2), 103–122.

Talukder, A., Manduchi, R., Castano, R., Owens, K., Matthies, L., Castano, A., & Hogg, R., (2002). Autonomous terrain characterization and modelling for dynamic control of unmanned vehicles. In: Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems, 2002. IEEE, pp. 708–713.

Ward, C., & Iagnemma, K., (2009). Speed-independent vibration-based terrain classification for passenger vehicles. Vehicle System Dynamics, Vol. 47, No. 9. IEEE, pp. 1095 – 1113

Weiss, C., Frohlich, H., & Zell, A., (2006). Vibration-based Terrain Classification Using Support Vector Machines. Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, pp. 4429 – 4434

Espín, D., & Villamarín, A., (2014). Diseño e implementación de un prototipo de unidad de medición fasorial (PMU – Phasor Measurement Unit) para el monitoreo, control y protección de sistemas eléctricos. Universidad de las Fuerzas Armadas – ESPE. Carrera de Ingeniería en Electromecánica.

Saltos, M., (2014). Análisis de señales sísmicas del volcán Cotopaxi mediante las transformadas de Wavelet y Fourier. Universidad de las Fuerzas Armadas – ESPE. Carrera de Ingeniería en Electrónica y Telecomunicaciones.

Oppenheim, A., & Schafer, R., (2009). Tratamiento de señales en tiempo discreto. España: Prentice Hall. 543-694.

Meyer-Baese, U,. (2007). Digital Signal Processing with Field Programmable Gate Arrays. Berlin: Springer-Verlag. 343-400

Miyara, F. (2004). Conversores D/A y A/D.

Betancourt, G., (2005). Las máquinas de soporte vectorial (SVMS). Scientia Et Technica, XI (27), 67-72.

Carmona, E., (2016). Tutorial sobre máquinas de vectores soporte (SVM). Dpto. de Inteligencia Artificial, ETS de Ingeniería Informática, Universidad Nacional de Educación a Distancia (UNED), C/Juan del Rosal, 16, 28040-Madrid (Spain).

MathWorks. (s.f.). Productos y Servicios. Recuperado el 24 de diciembre de 2018, de https://la.mathworks.com/company.html?s_tid=hp_ff_a_company

MathWorks. (2018). Soporte. Recuperado el 24 de diciembre de 2018, de https://la.mathworks.com/help/matlab/ref/filter.html

Refaeilzadeh, P., Tang, L., Lui, H. (2008). Cross-Validation. Arizona State University

James, G., Witten, D., Hastie, T., Tibshirani, R. (2013). An Introduction to Statistical Learning with Applications in R.