UNIVERSIDAD RICARDO PALMA

FACULTAD DE INGENIERÍA

ESCUELA PROFESIONAL DE INGENIERÍA ELECTRÓNICA

DETECCIÓN DEL ESTADO ETÍLICO EN CONDUCTORES A PARTIR DE UN ALGORITMO DIFUSO IMPLEMENTADO EN RASPBERRY PI 3

TESIS

PARA OPTAR EL TÍTULO PROFESIONAL DE INGENIERO ELECTRÓNICO

PRESENTADA POR

Bach. Ferreyra Ramos, José Luis

Bach. Martinez Amaya, Enrique Jesús

Asesor: Dr. Ing. Huamaní Navarrete, Pedro Freddy

LIMA-PERÚ

2021

DEDICATORIA

Dedico esta tesis a Dios y a mis padres José y Carmen, quienes gracias a su esfuerzo, consejos y apoyo a lo largo de mis años de estudio, pude cumplir con las diferentes metas que me trace a lo largo de mi camino.

José Luis Ferreyra Ramos

DEDICATORIA

A Dios, por brindarme la confianza y visión de poder alcanzar mis objetivos a pesar de cualquier adversidad que se pueda presentar en el camino.

A mis padres Enrique y Elena, por brindarme la oportunidad de desarrollar y demostrar las grandes virtudes que me fueron inculcadas por ellos gracias a su gran visión de instruir a un hijo preparado para afrontar los grandes retos de esta vida.

A mis tías Doris M, Rosario M. y María G. las cuales de manera insaciable siempre me mantenían enfocado en el desarrollo de este proyecto de tesis, llevando mi motivación por titularme a un nivel más alto del que me pude imaginar.

A mis hermanos Jessica, Magaly, Rous, Richard y Fernando, por siempre estar observándome en cada paso que doy, lo cual me fue de gran ayuda para mantenerme firme en mis objetivos.

Enrique Jesús Martínez Amaya

AGRADECIMIENTOS

Agradezco a mi alma mater, por haberme brindado los conocimientos de esta maravillosa carrera.

Al Dr. Ing. Pedro Huamaní, por su confianza, paciencia y gran motivación que nos brindó en toda nuestra etapa universitaria.

Al Ing. Yefman Cahuana amigo personal que nos brindó la asesoría automotriz necesaria para poder implementar nuestro prototipo en un vehículo.

ÍNDICE GENERAL

ÍNIDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

El presente proyecto describe el diseño, implementación y configuración de un algoritmo difuso para detectar el posible estado de ebriedad en conductores de vehículos, y así evitar que manejen en este estado. Para ello, se utilizaron sensores comerciales encontrados en el mercado local, como el sensor de alcohol MQ-3 y un flujómetro; así como también se utilizó un microcomputador Raspberry Pi 3, una pantalla HDMI, un módulo RFID, un conmutador de arranque y un relé automotriz, aplicando para esto una estrategia de recopilación y análisis de data haciendo uso de un algoritmo difuso con el cual se pudo entrenar para dar respuestas correctas según sus datos correspondientes a la hora de inferir; de igual forma, se logró la interacción entre el proyecto y el usuario mediante una pantalla HMI, utilizando los protocolos de comunicación mediante Wifi se logró sincronizar el Raspberry Pi 3 a un servidor en MySQL el cual validó el código de acceso del módulo RFID y además permitió registrar todos los eventos de condición de arranque; luego, se definieron las condiciones iniciales del actuador como es el caso del relé automotriz con la cual se llevó a cabo el arranque del motor. Posteriormente, se realizó una rutina de funcionamiento en el proyecto, el cual inició con la autorización de la lectura de los sensores mediante el módulo RFID, inmediatamente después la inferencia difusa que logró decidir la condición etílica del usuario. Y, finalmente la activación del actuador permitiendo el arranque del vehículo. Las pruebas del algoritmo difuso involucraron el uso de 5 tipos de bebidas alcohólicas muy consumidas por la población peruana (vino, pisco, ron, cerveza y whisky), así como un muestreo de 30 personas de las cuales el 54% dio como resultado el “no arranque” del vehículo, y el 46% el “si arranque” del mismo.

Palabras claves: Sensor MQ3, Flujómetro, Raspberry Pi 3, Algoritmo Difuso, MySQL.

ABSTRACT

This project describes the design, implementation and configuration of a fuzzy algorithm to detect the possible state of drunkenness in vehicle drivers, and thus prevent them from driving in this state. For this, commercial sensors found in the local market were used, such as the MQ-3 alcohol sensor and a flowmeter; as well as a Raspberry Pi 3 microcomputer, an HDMI screen, an RFID module, a start switch and an automotive relay were used, applying for this a strategy of data collection and analysis using a fuzzy algorithm with which it was possible to train to give correct answers according to their corresponding data when inferring; In the same way, the interaction between the project and the user was achieved through an HMI screen, using the communication protocols through Wifi, the Raspberry Pi 3 was synchronized to a MySQL server which validated the access code of the RFID module and also allowed logging of all boot condition events; then, the initial conditions of the actuator were defined as is the case of the automotive relay with which the engine start was carried out. Subsequently, an operation routine was carried out in the project, which began with the authorization of the reading of the sensors using the RFID module, immediately after the diffuse inference that managed to decide the user's ethyl condition. And finally, actuator activation allowing the vehicle to start. The tests of the fuzzy algorithm involved the use of 5 types of alcoholic beverages widely consumed by the Peruvian population (wine, pisco, ron, beer, and whiskey), as well as a sample of 30 people of which 54% resulted in "don’t start”the vehicle, and 46% the “start” of ther car.

Keywords: Sensor MQ3, flowmeter, Raspberry Pi 3, Sensor MQ3, Fuzzy Algorithm, MySQL.

INTRODUCCIÓN

En la actualidad, debido a la gran cantidad de accidentes automovilísticos con un estimado de 88,168 (Consejo Nacional de Seguridad Vial, 2017), se ha buscado implementar medidas de seguridad en los vehículos para ayudar a validar periódicamente el estado de los conductores antes de ingresar a una carretera. Según la revisión bibliográfica, existen numerosos trabajos de investigación que logran dicho propósito, tal como el de M. Casanova (2014) donde utiliza sensores de presión y alcohol para determinar la ebriedad según un valor determinado y posible de ser modificado según las normas de tránsito en Quito-Ecuador. O, el de Cesar Guañuna (2014) que se basa en 3 tipos de sensores para realizar sus distintos análisis como el nivel de alcohol, el peso y la posición. En vista de ello, para este proyecto de tesis se planteó la implementación de un prototipo electrónico para detectar el estado etílico en conductores, basado en un algoritmo difuso. Tal algoritmo fue diseñado e implementado algorítmicamente en un micro ordenador Raspberry Pi 3. En consecuencia, con el análisis del grupo de reglas difusas planteadas, se puede controlar la habilitación de un vehículo mediante un relé automotriz, complementado con una alarma visual en una pantalla gráfica. De esta manera, la presente investigación se enfoca en el estudio de la lógica difusa para la detección del estado de ebriedad en los conductores, con la finalidad de utilizar las dos variables principales: el peso y los gramos de alcohol en la sangre de un muestreo de 30 personas en las que se involucra tanto hombres como mujeres. Asimismo, esta investigación se justifica debido a la gran cantidad de accidentes producidos por conductores en estado de ebriedad, lo cual se puede disminuir al contar con el prototipo electrónico planteado. De esta manera, en las siguientes secciones se describen los principales aspectos, como la formulación del problema, la importancia, las limitaciones del estudio y los objetivos. Así como también, se muestran los términos teóricos importantes antes de presentar los antecedentes, y se describe la metodología de investigación haciendo énfasis en las técnicas de recolección y de procesamiento de datos. Asimismo, el diagrama electrónico de detección de conductores en estado de ebriedad e inhabilitación de la marcha vehicular propuesto en este proyecto de tesis, el cual está basado en el uso de un flujómetro comunicado con un sensor MQ-3 y una tarjeta RFID para suministrar a través del microcontrolador ATMEGA, las dos variables principales para el algoritmo difuso quien se encontrará implementado en un micro ordenador

Raspberry Pi 3. A su vez, dicho Raspberry Pi 3 permitirá o inhabilitará el arranque vehicular a partir de un relé automotriz, como respuesta a las leyes difusas de inferencia planteadas.

CAPÍTULO I. PLANTEAMIENTO Y DELIMITACIÓN DEL PROBLEMA

1.1. Formulación y definición del problema

Una gran cantidad de accidentes automovilísticos, a nivel nacional, se debe principalmente a la ingesta de alcohol por parte de los conductores. Por lo cual, en los últimos años, la Policía Nacional del Perú ha reforzado el control disponiendo de efectivos policiales los fines de semana en distritos de la capital, donde frecuentemente se realizan fiestas y se ingiere gran volumen de alcohol. Sin embargo, esto no es suficiente para brindar una seguridad óptima a todos los transeúntes. Por tal razón, el problema aún persiste y por ello es necesario tratar de atenuarlo con un prototipo electrónico que permita detectar el estado etílico en conductores. No obstante, esta posible solución presenta limitaciones debido a que no podrá ser instalado obligatoriamente en todos los automóviles particulares.

De esta manera, la formulación del problema general será:

¿Es posible implementar algorítmicamente un algoritmo difuso en un Raspberry Pi 3, con la finalidad de inhabilitar la marcha de un vehículo en el caso que su conductor se encuentre en estado etílico?

Y, de los problemas específicos:

a) ¿Es posible diseñar e implementar un algoritmo difuso en un micro ordenador Raspberry Pi 3 haciendo uso del lenguaje de programación Python?

b) ¿De qué manera se diseñará e implementará un circuito electrónico que permita la adquisición del grado de alcohol en la sangre, de manera confiable, utilizando un sensor de flujo y uno de alcoholemia?

c) ¿Cómo implementar una conexión eléctrica utilizando un relé automotriz, para inhabilitar el arranque de un vehículo en el caso de detectar un estado etílico en conductores?

1.2. Importancia y Justificación del estudio

En referencia a la lucha por reducir la cantidad excesiva de accidentes vehiculares, se tomó en cuenta la importancia de aumentar el control de conductores que a diario transportan al público que se moviliza por una ciudad, particularmente las unidades

de Taxi. Por lo tanto, este proyecto se justifica debido a la gran capacidad que tiene la electrónica en poder acondicionar sensores que faciliten el análisis de diferentes sustancias, además desarrollar e implementar algorítmicamente la teoría de lógica difusa en un dispositivo de hardware Raspberry Pi 3. De esta manera, sería viable aplicar correctamente la inferencia en el análisis del posible estado etílico del conductor de un automóvil, utilizando recursos económicos al alcance de todos los interesados.

1.3. Limitaciones del Estudio

Se limita al uso de dos variables para la inferencia del algoritmo difuso con respecto al estado de ebriedad del conductor, restringiendo la salida del algoritmo difuso a un dato discreto que corresponde a la conexión física hacia un relé automotriz habilitando o inhabilitando la marcha del automóvil. Asimismo, se utilizaron sensores comerciales para la medida del alcohol y flujo, así como también se hizo uso de un dispositivo RFID (IDentificación por Radio Frecuencia) para la obtención de información referente a cada conductor, tales como el peso, nombres y apellidos contenidos en una base de datos. Por otro lado, se utilizó un grupo de muestra de 30 personas, de ambos sexos y con edades fluctuando entre 18 y 60 años, y como también con un peso corporal oscilando entre los 45 y los 110 Kg.

Igualmente, este proyecto de tesis está limitado para aquellas unidades vehiculares donde sea viable la instalación del relé automotriz, así como también se realizó el análisis, diseño y prueba considerando la ingesta de solo cinco tipos de licores (cerveza, vino, pisco, ron y whisky) en una proporción suficiente para activar el sensor.

1.4. Objetivos

1.4.1 General

Detectar el estado etílico en conductores utilizando una inferencia a partir de un algoritmo difuso que será diseñado e implementado en un Raspberry Pi 3, para inhabilitar la marcha de un vehículo.

1.4.2 Específicos

a) Diseñar e implementar un algoritmo difuso en un micro ordenador Raspberry Pi 3 utilizando el lenguaje de programación Python.

b) Diseñar e implementar un circuito electrónico que permita la adquisición del grado de alcohol en la sangre, de manera confiable, utilizando un sensor de flujo y de alcoholemia.

c) Implementar una conexión eléctrica utilizando un relé automotriz, para inhabilitar el arranque de un vehículo en el caso de detectar un estado etílico en conductores.

CAPÍTULO II. MARCO TEÓRICO

2.1. Marco Histórico

Debido al gran ingenio que revolucionó la forma de transporte y comunicación con la aparición del automóvil, se produjeron muchas situaciones favorables como a la vez desfavorables para las personas de aquella época; por lo cual, se vio indispensable la creación y establecimiento de normas para la organización y seguridad de las carreteras.

Es así que, en 1908, mediante el Congreso Internacional de Carreteras realizado en Roma, se definieron patrones básicos sobre los signos de tráfico. Para 1909, nueve gobiernos europeos acordaron el uso de estos signos, los cuales indican “niveles de grado de cruce de ferrocarril”, “golpe”, “curva” e “intersección”. Además, el aumento paralelo de signos de tránsito y automóviles entre los años 1926 y 1949 condujo al desarrollo del sistema de señalización europea.

Luego, en 1968 se firmó por todos los países europeos la Convención de Viena, la cual es un tratado sobre la circulación por carretera teniendo como objetivo la normalización de normas de tránsito en sus países participantes, con fines de mejorar el tránsito y la seguridad en las carreteras. Como resultado de este tratado en Europa Occidental, en gran parte de Asia y África, un total de 52 países se comprometieron con dicho tratado.

Estados Unidos de igual manera desarrolló su propio sistema de señalización, el cual fue adoptado por varias naciones, pero con pequeñas variaciones propias de cada país. Este sistema está implementado en toda América y casi en la totalidad de Oceanía y algunos países africanos (Anónimo, 2017)

En la actualidad, cada año los accidentes vehiculares cobran a nivel mundial la vida de 1.2 millones de personas, tratándose así de la primera causa de mortalidad entre personas de 15 y 29 años, y de la novena en cifras mundiales (Organización panamericana de la salud, 2018). En el Perú, según el Sistema Informático de Denuncias Policiales (SIDPOL), el promedio de accidentes vehiculares según los últimos años es de 118,044, mientras que en Lima se presenta 58,007 accidentes que corresponde al 49.7% del total. De esta manera, los accidentes automovilísticos pueden tener diversas causas; sin embargo, es alarmante observar que en cuarto lugar

se encuentre la Ebriedad del Conductor con un 6.6%, lo cual a la vez demuestra la falta de conciencia que los conductores han adoptado en los últimos años.

2.2. Investigaciones Relacionadas con el tema

María Casanova Vásquez (2014), en la tesis titulada: Diseño, construcción e instalación de un alcoholímetro electrónico con dispositivos de bloqueo de un vehículo, señala:

> Se llevó a cabo la implementación del sistema en un vehículo Mazda BT-50. Donde el objetivo principal fue proteger y evitar que el vehículo se encienda si el conductor excede el nivel de alcohol permitido por las leyes de tránsito vigentes. Los elementos que componen el alcoholímetro son sensores MQ-3, de presión, pantalla LCD, módulo de sonido, componentes internos, microcontroladores modificadores de voltaje, resistencias y relés. Se realizaron pruebas en el Centro de Contraventores de Tránsito de Quito, obteniendo parámetros de funcionamiento del alcoholímetro, que sirvieron para construir el módulo de control, que va alojado en un lugar visible para el conductor y permite realizar la prueba de alcocheck cuando vaya a encender su vehículo, o en caso de que el sistema lo solicite. Asimismo, las pruebas realizadas permitieron comparar las variaciones del mismo con respecto al alcoholímetro utilizado en la Policía de Tránsito, otorgando un límite de 0.3 gr/l de alcohol en la sangre, pero posible de ser programado para diferentes niveles.

César Guañuna Pozo (2014), en la tesis titulada: Diseño e implementación de sistemas de encendido para automóvil mediante bloqueo por alcocheck con dispositivos de peso y posición de conductor, señala:

> Se comenzó con el estudio del material de bibliografía concerniente a índices de consumo de alcohol y los efectos que estos conllevan si se juntan con la conducción, se estudió también al sistema de encendido del vehículo Skoda Fabia 2.0 y los diferentes sensores que actúan en el circuito de bloqueo. Para la elaboración del circuito se adquirió el sensor de alcohol MQ-3, el de posición por ultrasonido SRF05 y el interruptor de peso. Ya con todos los elementos principales a disposición se diseñó el circuito de bloqueo tomando en cuenta

parámetros como el suministro de voltaje por medio de la batería del vehículo y la carga que ejerce el circuito de bloqueo en el sistema de generación de energía del automóvil. Después del diseño se construyó el circuito en la placa electrónica. Posteriormente se colocó el circuito de bloqueo y los sensores de alcohol, peso y posición en lugares estratégicos dentro del habitáculo del vehículo con el objetivo de no interferir en la estética del mismo. Finalmente, ya con todos los dispositivos correctamente instalados se realizó la medición y el análisis de los sensores cuando el conductor se encuentra sobrio y cuando ha ingerido algún tipo de bebida alcohólica. Además, se determinó que para aprobar la prueba del alcocheck el conductor no debe sobrepasar el consumo de 4 cervezas de 330cc y 4.2° de alcohol.

A. Ahmad, S. Shafi, A. Kumaravel (2013), en la tesis titulada: VehNode: Wireless Sensor Network Platform for automobile pollution control, señala:

> Se planteó una solución a su problemática mediante la introducción de VehNode, una plataforma de monitoreo de contaminación vehicular que es capaz de medir diferentes tipos de concentraciones de contaminantes contenidas en el humo producido por el vehículo e informa el estado automáticamente cuando sea necesario a las agencias involucradas. Asimismo, ellos aseguran la existencia de la plataforma de red de sensores inalámbricos para el control de la contaminación de automóviles, centrándonos en una fácil accesibilidad de los datos en tiempo real a través de la Web. Los datos en tiempo real estarán disponibles para tres grupos principales de usuarios: Propietario del Vehículo, Departamento de Tráfico y agencias ambientales nacionales.

S. Kumar, A. Jasuja (2017) en su proyecto titulado: Air quality monitoring system based on IoT using Raspberry Pi, señala:

> Se presentó un sistema de monitoreo de la calidad del aire que incluye varios parámetros: PM 2.5, monóxido de carbono, dióxido de carbono, temperatura, humedad y presión del aire. Además, el Internet de las cosas que converge con

la computación en la nube ofrece una técnica novedosa para una mejor gestión de los datos provenientes de diferentes sensores, recopilados y transmitidos por una minicomputadora basada en ARM de bajo consumo y bajo costo, la Raspberry Pi 3. El sistema se prueba en Delhi y las mediciones se comparan con los datos proporcionados por la autoridad de control del entorno local y se presentan en forma de tabla. Los valores de los parámetros medidos se muestran en IBM Bluemix Cloud.

H. Chiang, Y. Chen, B. Wu, T. Lee (2014), en su proyecto titulado: Embedded Driver-Assistance System Using Multiple Sensors for Safe Overtaking Maneuver, señala:

> Que en este documento se desarrolló una etapa de fusión de datos basada en un algoritmo de advertencia de colisión en el que los vehículos adelantados y otros vehículos en el carril vecino, se contabilizan para evitar colisiones. El sistema emplea control difuso en la dirección y la automatización de la velocidad, para emular las tareas de conducción realizadas por los seres humanos. La aplicabilidad del sistema propuesto se examinó en un entorno de carretera real, y un conjunto de resultados experimentales demuestran la viabilidad de utilizar las estrategias de coordinación involucradas mientras se realizan varias maniobras de conducción.

2.3. Estructura teórica y científica que sustenta el estudio

2.3.1. Tabla de Alcoholemia:

El pasado 09 de junio del año 2002, se publicó la Ley N° 27753 que tiene como contenido la modificación de los artículos 111°, 124° y 274° del Código Penal referidos al homicidio culposo, lesiones culposas y conducción en estado de ebriedad o drogadicción y el artículo 135° del código procesal penal, sobre mandato de detención. (República del Perú, 2018)

En esta modificación se anexó una tabla de alcoholemia cuyos valores son referenciales y forman parte de dicha ley. Asimismo, se solicitó sea expuesta

obligatoriamente en lugares visibles donde se expendan bebidas alcohólicas. Ver la figura N° 1.

TABLA DE ALCOHOLEMIA

1er. Período: 0.1 a 0.5 g/l: subclínico. No existen síntomas o signos clínicos, pero las pruebas psicométricas muestran una prolongación en los tiempos de respuesta al estímulo y posibilidad de accidentes. No tiene relevancia administrativa ni penal.
2do. Período: 0.5 a 1.5 g/l: ebriedad. Euforia, verborragia y excitación, pero con disminución de la atención y pérdida de la eficiencia en actos más o menos complejos y dificultad en mantener la postura. Aquí está muy aumentada la posibilidad de accidentes de tránsito, por disminución de los reflejos y el campo visual.
3er. Período: 1.5 a 2.5 g/l: ebriedad absoluta. Excitación, confusión, agresividad, alteraciones de la percepción y pérdida de control.
4to. Período: 2.5 a 3.5 g/l: grave alteración de la conciencia. Estupor, coma, apatía, falta de respuesta a los estímulos, marcada descoordinación muscular, relajación de los esfínteres.
5to. Período: niveles mayores de 3.5 g/l: Coma. Hay riesgo de muerte por el coma y el para respiratorio con afección neumonológica, bradicardia con vaso dilatación periférica y afección intestinal.

Figura N° 1: Tabla de alcoholemia
Fuente: Lpderecho (2018)

2.3.2. Pantalla HDMI:

Es un dispositivo cuya función es el desplaye de video haciendo uso de la tecnología HDMI, que por sus siglas en ingles se le conoce como Interfaz multimedia de alta definición, la cual puede transmitir audio y video por medio de un único cable sin necesidad de compresión (S. Mueller. 2015). A continuación, en la figura N° 2, se muestra la Pantalla HDMI modelo TFT de 3.5” que fue empleada en este proyecto de tesis.

Figura N° 2: Pantalla HDMI TFT 3.5”
Fuente: Robotshop (2020)

2.3.3. Acondicionamiento de señales:

Es una técnica que convierte una señal electrónica en otro tipo de señal diferente. Principalmente, su finalidad es hacer que una señal de difícil lectura a través de un sensor o instrumento convencional, sea posible leerlo de manera más sencilla. (Ramón Pallas, 2003)

2.3.4. Algoritmo difuso:

Es un tipo de control que hace uso de la teoría de lógica difusa. En ese caso el control puede ser del tipo multi-variable permitiendo que, a su vez, un grupo de entradas y salidas, sean posibles controlarlos. Asimismo, son tres definiciones importantes que se encuentran asociadas a esta teoría. A continuación, se describen:

- Variable Lingüística: es la variable utilizada como entrada o salida, al algoritmo difuso. Se diferencia de una variable convencional porque su valor asociado no es un número.
- Reglas difusas: son un conjunto de leyes que establecen la relación entre las variables lingüísticas de entrada y salida de un algoritmo difuso.
- Conjunto difuso: es el valor asignado a cada una de las variables lingüísticas de un algoritmo difuso.

Seguidamente, en la figura 3, se muestra una representación general de los bloques operacionales de un algoritmo difuso. Donde el bloque fuzzificación convierte las variables físicas de entrada del sistema en variables difusas mediante las funciones de pertenencia. Luego, en el bloque defuzzificación, las variables difusas obtenidas del sistema de inferencia son convertidas en variables de reales de salida. Por otro lado, el bloque Base de Conocimientos define las reglas lingüísticas del control y la manipulación de la información difusa referente a las funciones de pertenencia de los conjuntos difusos. Y, finalmente, el bloque sistema de inferencia se apoya en las reglas del controlador y la inferencia difusa, para calcular las salidas difusas asociadas a las variables reales de salida. (Pedro Ponce, 2010)

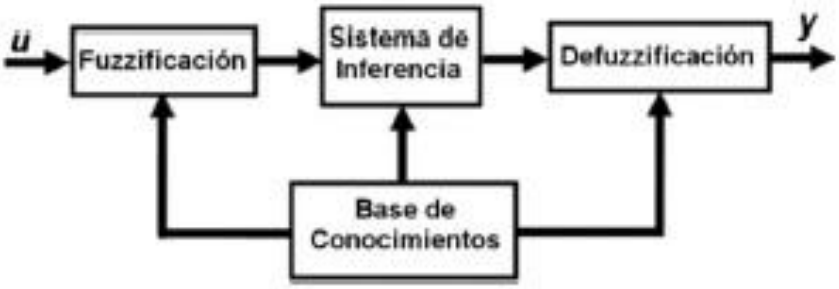

Figura N° 3: Diagrama de bloques algoritmo difuso
Fuente: Conycit(2015)

2.4. Hardware del proyecto.

2.4.1 Raspberry Pi 3.

Es un dispositivo de diseño embebido y programable para estudiantes. Permite el diseño y desarrollo de diferentes proyectos incluido como uno de los principales el aplicado al internet del todo. Contiene procesador ARM cortex A53 de cuatro núcleos a 1.2GHz de 64 bits. Además, el Raspberry Pi 3 consta de Bluetooth 4.1 de bajo consumo, wifi 802.11n integrado, 1GB de RAM, socket Ethernet 10/100, salida HDMI, conector GPIO, ranura microSD, entre otras características. (Mike McGrath 2016). A continuación, en la figura 4 se muestra la fotografía del dispositivo Raspberry Pi 3 versión B utilizado en este proyecto de tesis.

Figura N° 4: Dispositivo Raspberry Pi 3
Fuente: Raspberrypi.org(2015)

2.4.2 Arduino uno.

Arduino es una plataforma de hardware y software libre, basada en un microcontrolador ATMEGA328p embebido en una placa facilitando el manejo de entradas y salidas, analógicas y digitales, con un entorno de desarrollo basado en un lenguaje de programación sencillo para prototipos electrónicos. (Claudio Peña, 2020). A continuación, en la figura 5 se muestra la fotografía del dispositivo Arduino Uno utilizado en este proyecto de tesis.

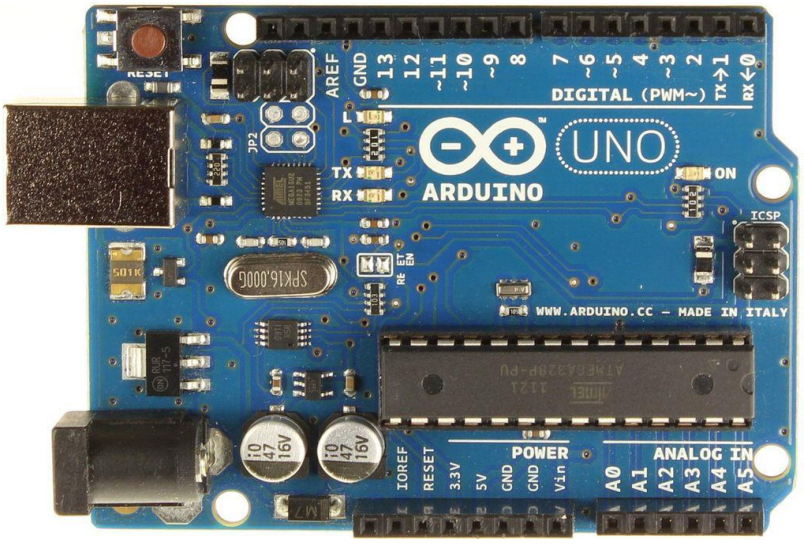

Figura N° 5: Dispositivo Arduino Uno
Fuente: Electrocrea (2020)

2.4.3 R.F.I.D.

Según (Daniel Hunt y otros. 2007), R.F.I.D es una tecnología que por sus siglas en inglés significa identificación por radiofrecuencia, la cual es utilizada para transmitir e identificar un código único pregrabado de forma inalámbrica en cortas distancias entre personas y objetos. A continuación, en la figura 6 se muestra la fotografía del dispositivo R.F.I.D genérico utilizado en este proyecto de tesis.

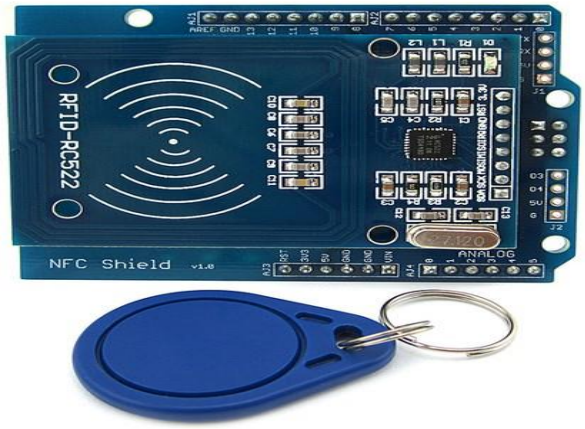

Figura N° 6: Dispositivo R.F.I.D. Lector y tag en forma de llavero
Fuente: Geekbuying

2.4.4 Sensor MQ-3.

Es un tipo de sensor adecuado para la detección de alcohol en el aire, con un rango de operación de 20 a 500 ppm, y se traduce a un cierto nivel de voltaje. (Leonel Corona y Otros, 2014).

Para que el sensor MQ-3 tenga un nivel de voltaje estable se implementa en su placa un divisor de tensión entre la resistencia interna del sensor y una resistencia externa en este caso de 10K ohms, esto se realiza con la finalidad de calibrar la sensibilidad del sensor; adicionalmente, utiliza un regulador de voltaje 7805 no regulable el cual brinda un voltaje de salida 5 v constante, esto para mayor protección según especificaciones técnicas del fabricante, tal como se muestra en la figura N° 7. Asimismo, en la figura N° 8 se muestra una fotografía del sensor MQ-3 utilizado en este proyecto de tesis.

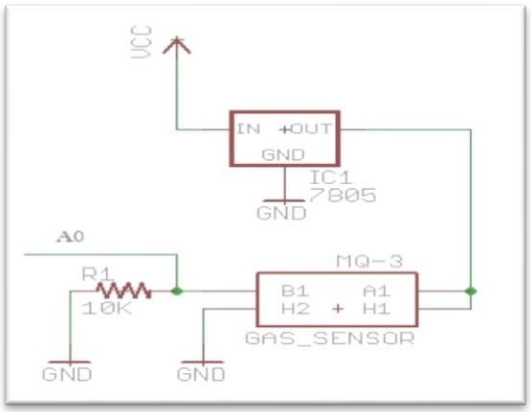

Figura N° 7: Circuito acondicionador del sensor MQ-3
Fuente: Tecexcirc (2016).

Figura N° 8: Sensor MQ3
Fuente: Naylampmechatronics (2018)

2.4.5 Flujómetro.

También conocido como caudalímetro, es un tipo de sensor que detecta la velocidad o flujo con el que se mueve un líquido o gas en un conducto o sistema, su principio de funcionamiento tiene diferentes modalidades según sea la materia con el que se está operando. Para el caso de elementos neumáticos se suele utilizar flujómetros basados en la presión diferencial siendo que estos pueden trabajar a mayores niveles de caudal y temperaturas (Ramón Pallas. 2003). A continuación, la figura N° 9 muestra la fotografía del flujómetro modelo YF – S201 utilizado en este proyecto de tesis.

Figura N° 9: Flujómetro modelo YF – S201
Fuente: LuisLlamas (2018)

2.4.6 Relé Automotriz.

Es un tipo de relé especial para vehículos, el cual se define como un control remoto para el arranque. El principio de funcionamiento de este dispositivo se basa en la configuración de 4 terminales, de los cuales 2 forman el circuito de control conformado por una bobina y los otros dos conforman el circuito actuador o de un conmutador de dos estados (abierto y cerrado). (José Orozco. 2014). A continuación, la figura N° 10 muestra una fotografía del relé automotriz modelo 40 – AR - B utilizado en este proyecto de tesis.

Figura N° 10: Relé modelo 40 – AR – B Audiopipe
Fuente: Kemik (2020)

2.4.7 Conmutador de arranque.

La llave de arranque, conocida también como conmutador de arranque o llave de contacto, dispone de un mecanismo interno el cual impide accionar el arranque una vez que el motor térmico se ha puesto en funcionamiento, como consecuencia de un primer accionamiento de arranque desde la llave. Con esto se consigue proteger al mecanismo de arrastre y al inducido del motor de arranque. De no disponer de este sistema, el motor de arranque podría alcanzar unas elevadas revoluciones que causarían averías graves debido a las altas fuerzas de inercia del inducido. (Mariano Sánchez, 2014, p. 96). A continuación, la figura N° 11 muestra una fotografía del conmutador de arranque universal utilizado en este proyecto de tesis.

Figura N° 11: Conmutador de Arranque modelo universal
Fuente: Amazon

2.5. Software del proyecto.

2.5.1 Matlab.

"Es un software de computación científica para optimizar la resolución de problemas de ingeniería; además, este software es muy útil en el desarrollo de matrices, ecuaciones y gráficos en un lenguaje de alto nivel." (Moore, 2007, p. 3)

2.5.2 Python.

Es un lenguaje de programación interpretado cuya filosofía hace hincapié en una sintaxis que favorezca un código legible. Además, también se le considera como un lenguaje de programación multi-paradigma, ya que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional. (Sebastien Chazallet. 2016)

2.5.3 MySQL.

MySQL es un sistema de gestión de base de datos relacional desarrollado como software libre, basado en lenguaje de consulta estructurado (SQL). En donde los datos son organizados y almacenados en tablas de valores. (Natsys 2014)

CAPÍTULO III. DESARROLLO DEL PROYECTO

En el presente capítulo se describe el desarrollo del proyecto, el cual está compuesto de 3 etapas, iniciando con el desarrollo del hardware electrónico, desarrollo del algoritmo difuso y base de datos en MySQL, y finalmente la fase de integración de hardware y software.

3.1. Desarrollo del hardware electrónico.

La implementación del proyecto requirió de circuitos electrónicos y diagramas automotrices, para los cuales fue necesario el desarrollo de hardware con el cual se pudo brindar la estabilidad eléctrica de los controladores (Raspberry PI 3, Atmega328p), sensores (MQ-3, Flujómetro) y actuadores (Relé Automotriz).

3.1.1 Regulador de Voltaje

La alimentación en vehículos es suministrada por una batería eléctrica cuyo voltaje es 12/24 V, razón por la cual fue necesario utilizar un regulador de voltaje a 5V con el cual fue posible cumplir con las especificaciones de alimentación, principalmente del microcontrolador Raspberry Pi 3. Este a su vez se encargó de alimentar a sus periféricos mediante sus puertos seriales USB.

El circuito regulador fue diseñado en base a un integrado LM317, el cual por recomendación del fabricante (ON SEMICONDUCTOR, 2012, Datasheet LM317), incluyó un potenciómetro para la regulación del voltaje conectado en el pin C o de ajuste y una resistencia fija cuyo valor típico es de 220 ohm entre el pin de salida y el de ajuste, tal como se muestra en la figura N° 12, esto para tener una mejor calibración del nivel de salida deseado (5 VDC). Adicionalmente, se añaden al circuito diodos de protección para los terminales del regulador y condensadores, cuya finalidad es estabilizar la salida.

Para el diseño del circuito regulador de voltaje LM317 se hizo uso del software Proteus, como se muestra en la figura N° 13. Este software brindó las capacidades de poder diseñar un diagrama electrónico y someterlo a diversas pruebas, las cuales logran evitar posibles errores al finalizar su construcción. De esta manera, gracias a estas virtudes se pudo realizar las simulaciones correspondientes con las que se validó su correcto funcionamiento.

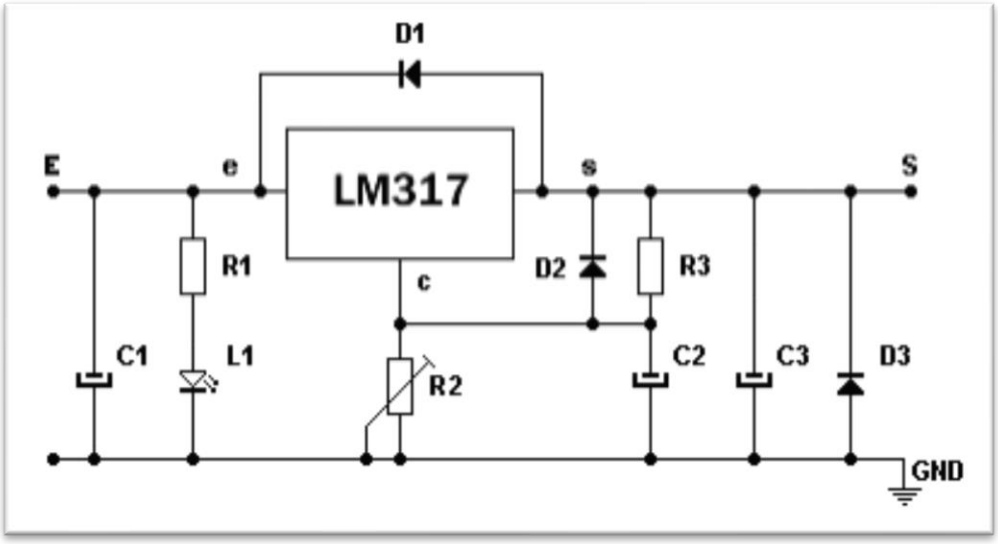

Figura N° 12: Regulador de voltaje propuesto por el fabricante
Fuente: ON SEMICONDUCTOR. (2012)

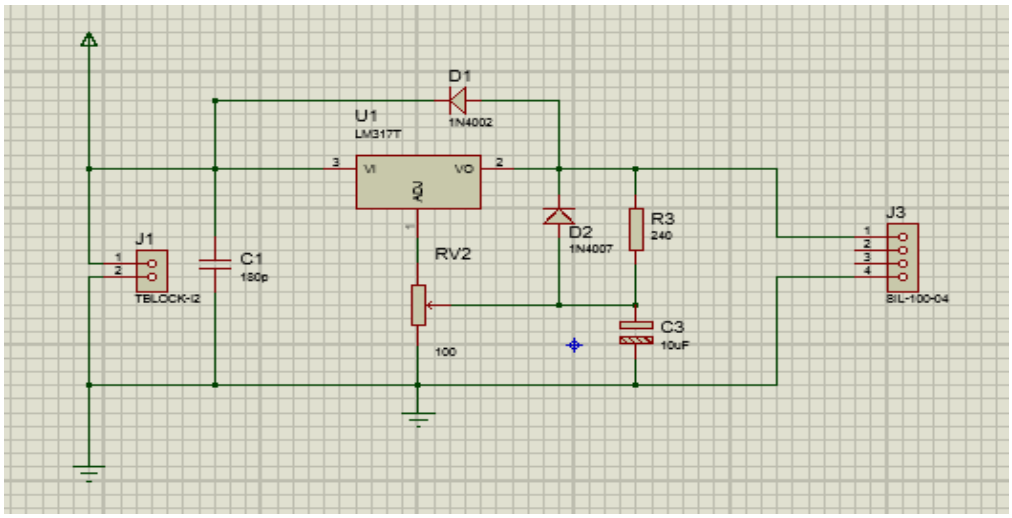

Figura N° 13: Circuito regulador de voltaje diseñado en Proteus.
Fuente: Propia.

3.1.2 Diagramas de arranque Electromecánicos

Para poder llevar a cabo un bloqueo exitoso del arranque vehicular, se tomó en cuenta dos diferentes diagramas según el tipo de vehículo, los cuales son:

a) Esquema de arranque en automóviles mecánicos

En los automóviles mecánicos, el sistema de arranque requiere del accionar tanto del interruptor de encendido, como también del interruptor

de arranque el cual se encuentra en el pedal de embrague como se puede ver en la figura N° 14. Para llevar a cabo el bloqueo del arranque vehicular, se modificó la línea de tierra que llega al relé de arranque, reemplazando el interruptor de arranque por un relé de bloqueo como se observa en la figura N° 15, el cual fue accionado únicamente cuando el algoritmo difuso infirió que el conductor estuvo acto para conducir.

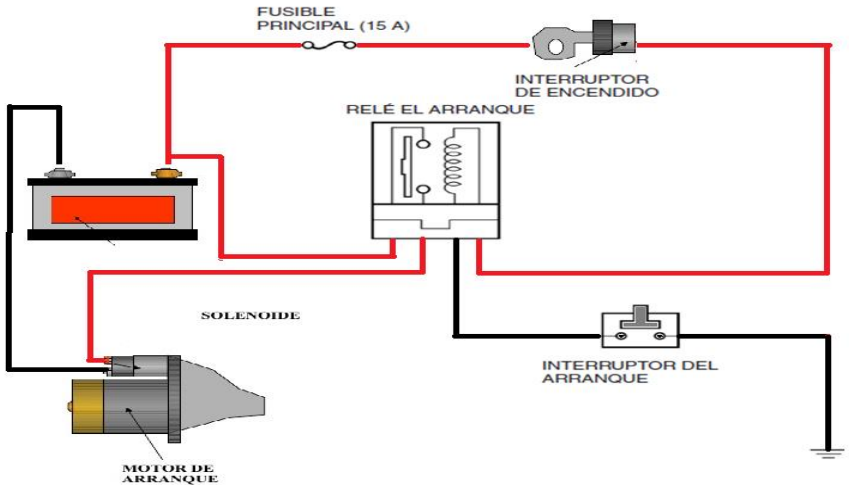

Figura N° 14: Diagrama automotriz original de vehículo mecánico.
Fuente: Propia.

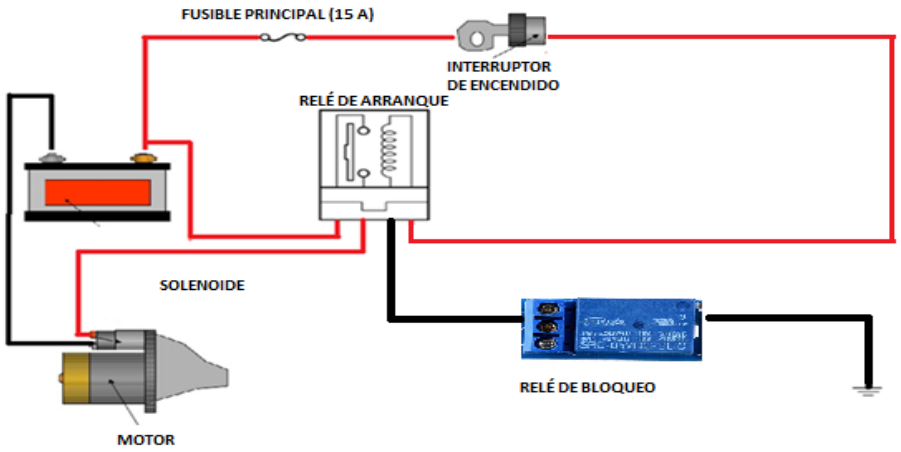

Figura N° 15: Diagrama automotriz modificado de vehículo mecánico.
Fuente: Propia.

b) Esquema de arranque en automóviles automáticos

En los vehículos automáticos, el sistema de arranque solo necesita del accionar del interruptor de encendido, esto debido a que la bobina del relé

de arranque se encuentra conectado de manera directa a la tierra de la batería, a diferencia de los vehículos mecánicos. Para llevar a cabo el bloqueo de arranque vehicular, se modificó la línea de señal que se dirige a la bobina del relé de arranque, colocando de manera serial un relé de bloqueo para evitar que el automóvil encienda, tal como se muestra en la figura N° 16.

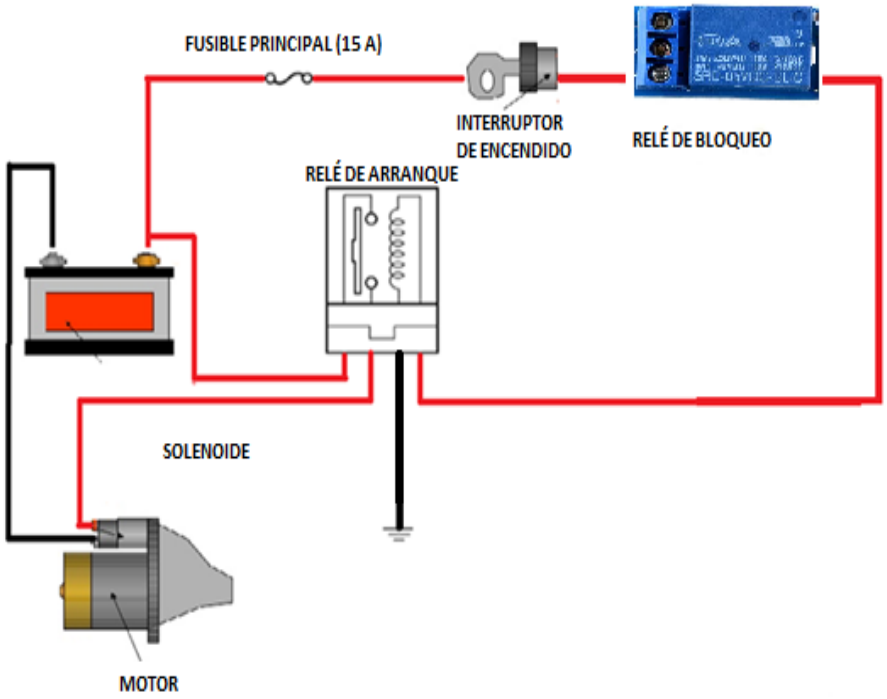

Figura N° 16: Diagrama automotriz modificado de vehículo automático.
Fuente: Propia.

3.2. Desarrollo del algoritmo difuso y base de datos en MySQL

3.2.1 Diseño del algoritmo difuso

Para el diseño del algoritmo difuso se decidió aplicar el método Mamdani, debido a que brinda ventajas importantes como: una función de membresía o de pertenencia de salida, facilidad de interpretación de las reglas, incorporación de conocimientos y experiencias, posibilidad de hacer uso de los sistemas MISO (entrada múltiple, salida única) y MIMO (entrada múltiple, salida múltiple), a diferencia del método Takagi-Sugeno que es de igual de conocido al anterior pero proporcionando: ausencia de una función de membresía de

salida, reglas basadas en funciones lineales lo cual fue poco interpretable, y opción de uso MISO (entrada múltiple, salida única). Debido a estas diferencias, se eligió el primer método mencionado con el cual se obtuvo una respuesta óptima en el control de arranque del vehículo. De esta manera, en el algoritmo difuso propuesto se hizo uso de dos variables lingüísticas de entradas: Alcohol y Peso. Asimismo, cada una de estas variables está compuestas por 5 funciones de membresía (2 trapezoides y 3 triángulos), y a cada función se le asignaron parámetros adecuados para facilitar el diseño del algoritmo difuso. Seguidamente, se describen las variables lingüísticas de entrada.

- Variable lingüística: Alcohol

  Esta variable lingüística es la encargada de establecer límites en los niveles de alcohol que debe tener una persona al conducir un vehículo, estos límites están establecidos de acuerdo con el reglamento del límite máximo de alcohol en la sangre que puede tener un conductor en las carreteras peruanas. Y la unidad corresponde a gramos/litro de sangre (g/l).

  Bajo esta información, se definieron los conjuntos difusos que están asignados con una función de membresía cada uno, las cuales son: MB (muy bajo), BJ (bajo), CL (casi en límite), L (límite) y AL (alto). Asimismo, se hizo uso del Fuzzy Logic Toolbox de Matlab, con el cual se pudo modelar comportamientos complejos del proyecto de tesis, utilizando reglas lógicas simples y a su vez implementadas en un método de inferencia. Además, se permitió la representación gráfica, tal como se muestra en la figura N° 17.

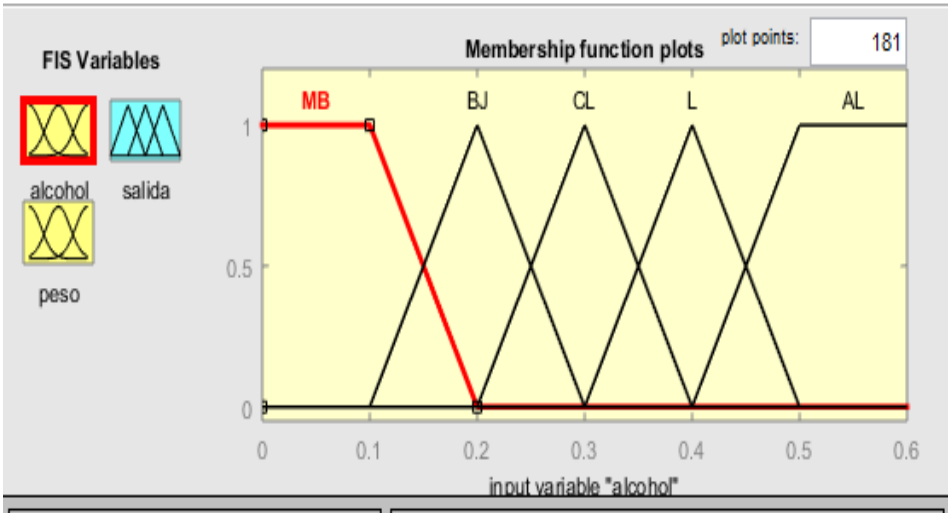

Figura N° 17: Variable de nivel de alcohol medido en la sangre elaborada en matlab.
Fuente: Propia.

Asimismo, los tipos y parámetros de cada función de membresía para la variable de entrada ALCOHOL, se detallan en la tabla N° 1:

Tabla N° 1: Tipos y parámetros de las funciones de membresía para la variable de entrada alcohol.

Funciones de Membresía	**Tipo**	**Parámetros**
MB	Trapecio	[0 0 0.1 0.2]
BJ	Triángulo	[0.1 0.2 0.3]
CL	Triángulo	[0.2 0.3 0.4]
L	Triángulo	[0.3 0.4 0.5]
AL	Trapecio	[0.4 0.5 0.6 0.6]

Fuente: Propia.

Además, es importante aclarar que las funciones de pertenencia o membresía elegidas fueron las triangulares y trapezoidales, tal como se muestra en la figura 17. Esto se debió a su sencillez y facilidad de cálculos cuando fueron programadas algorítmicamente en el sistema embebido Raspberry Pi 3. Por la

misma razón, solamente se establecieron cinco funciones de membresía; así como también, los parámetros fueron seleccionados en base a las recomendaciones bibliográficas referentes a la teoría de lógica difusa.

- Variable lingüística: Peso

Esta variable lingüística se encarga de establecer límites de peso que una persona podría conducir un vehículo, por lo que fue de gran importancia en el proyecto al igual que la de alcohol, debido al fundamento de metabolismo de alcohol en la sangre. Según Lee Goldman, Andrew I. Schafer (2016) sustenta que:

> El alcohol de las bebidas alcohólicas contiene etanol, que actúa como hipnótico, este se absorbe rápidamente mediante la circulación sanguínea del estómago y el intestino. La enzima principal del metabolismo es el ***alcohol deshidrogenasa gástrica,*** el cual metaboliza en el hígado y transforma las moléculas de etanol en un compuesto más tóxico llamado acetaldehído. El metabolismo es proporcional al peso corporal de la persona y a una diversidad de otros factores, pero una diversidad de otros factores también puede afectar como pueden ser los alimentos en el estómago o la adaptación de esta glándula debido al continuo consumo de alcohol. (Lee Goldman, Andrew I. Schafer, 2016, pp. 33)

La unidad de esta variable corresponde a kilogramos (Kg).

Entonces, en base al análisis se procedió a definir los conjuntos difusos asignándoles una función de membresía a cada uno, las cuales son: VL (very low), L (low), I (intermedium), IH (inter-high), HG (high). Asimismo, se hizo uso del Fuzzy Logic Toolbox de Matlab, con el cual se pudo modelar comportamientos complejos del proyecto de tesis, utilizando reglas lógicas simples y a su vez implementadas en un método de inferencia. Además, se permitió la representación gráfica, tal como se muestran en la figura N° 18.

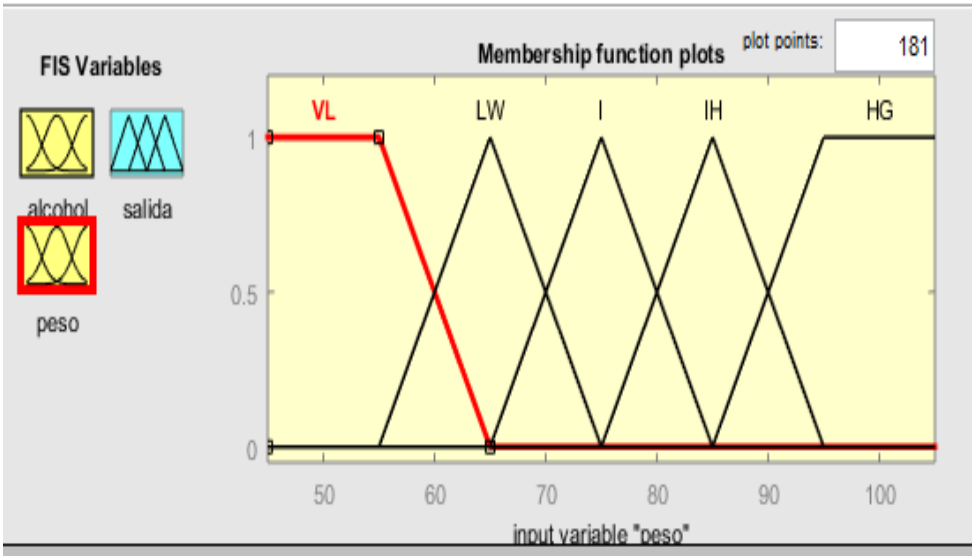

Figura N° 18: Variable de peso promedio en los conductores de vehículos en Perú.
Fuente: Propia.

Asimismo, los tipos y parámetros de cada función de membresía para la variable de entrada PESO, se detallan en la tabla N° 2:

Tabla N° 2: Tipos y parámetros de funciones de membresía para la variable de entrada PESO.

Funciones de Membresía	**Tipo**	**Parámetros**
MB	Trapecio	[45 45 55 65]
BJ	Triángulo	[55 65 75]
I	Triángulo	[65 75 85]
IH	Triángulo	[75 85 95]
HG	Trapecio	[85 95 105 105]

Fuente: Propia.

Además, es importante aclarar que las funciones de pertenencia o membresía elegidas fueron las triangulares y trapezoidales, tal como se muestra en la figura 18. Esto se debió a su sencillez y facilidad de cálculos cuando fueron programadas algorítmicamente en el sistema embebido Raspberry Pi 3. Por la misma razón, solamente se establecieron cinco funciones de membresía; así

como también, los parámetros fueron seleccionados en base a las recomendaciones bibliográficas referentes a la teoría de lógica difusa.

- Variable lingüística de Salida

Es una variable lingüística, la cual fue tomada por el relé de bloqueo para modificar el estado del proceso. Esta fue diseñada con dos funciones de membresía triangulares: ’Encendido’ y ‘Apagado’, las cuales definieron los dos estados del actuador vehicular. Los parámetros fueron asignados de manera referencial, debido a que solamente se buscó representar una superficie de control, con la cual finalmente se pudo definir los rangos de funcionamiento del actuador final. Asimismo, se hizo uso del Fuzzy Logic Toolbox de Matlab, con el cual se pudo modelar comportamientos complejos del proyecto de tesis, utilizando reglas lógicas simples y a su vez implementadas en un método de inferencia. Además, permitió la representación gráfica tal como se muestra en la figura N° 19.

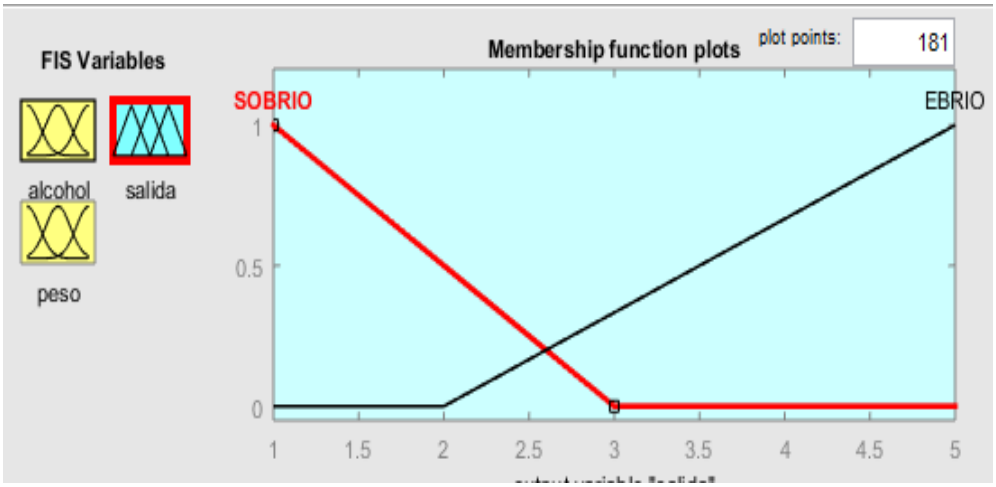

Figura N° 19: Variable de salida del algoritmo Difuso.
Fuente: Propia.

Asimismo, los tipos y parámetros de cada función de membresía para la variable de salida, se detallan en la tabla N° 3:

Tabla N° 3: Tipos y parámetros de funciones de membresía para la variable de salida.

Funciones de Membresía	Tipo	Parámetros
SOBRIO	Triángulo	[1 1 3]
EBRIO	Triángulo	[2 5 5]

Fuente: Propia.

- Reglas difusas

Las reglas difusas son representaciones de conocimientos, basados en conceptos teóricos o en testimonios de expertos en algún tema a tratar. Para el proyecto se hizo uso de las reglas basadas en el modelo difuso Mamdani, ya que brindaba como ventajas ser un modelo muy intuitivo y adaptable a la incorporación de conocimientos y experiencia. Las reglas utilizadas fueron:

1. If (alcohol is MB) and (peso is VL) then (salida is EBRIO) (1)
2. If (alcohol is MB) and (peso is LW) then (salida is SOBRIO) (1)
3. If (alcohol is MB) and (peso is I) then (salida is SOBRIO) (1)
4. If (alcohol is MB) and (peso is IH) then (salida is SOBRIO) (1)
5. If (alcohol is MB) and (peso is HG) then (salida is SOBRIO) (1)
6. If (alcohol is BJ) and (peso is VL) then (salida is EBRIO) (1)
7. If (alcohol is BJ) and (peso is LW) then (salida is EBRIO) (1)
8. If (alcohol is BJ) and (peso is I) then (salida is SOBRIO) (1)
9. If (alcohol is BJ) and (peso is IH) then (salida is SOBRIO) (1)
10. If (alcohol is BJ) and (peso is HG) then (salida is SOBRIO) (1)
11. If (alcohol is CL) and (peso is VL) then (salida is EBRIO) (1)
12. If (alcohol is CL) and (peso is LW) then (salida is EBRIO) (1)
13. If (alcohol is CL) and (peso is I) then (salida is EBRIO) (1)
14. If (alcohol is CL) and (peso is IH) then (salida is SOBRIO) (1)
15. If (alcohol is CL) and (peso is HG) then (salida is SOBRIO) (1)
16. If (alcohol is L) and (peso is VL) then (salida is EBRIO) (1)
17. If (alcohol is L) and (peso is LW) then (salida is EBRIO) (1)
18. If (alcohol is L) and (peso is I) then (salida is EBRIO) (1)
19. If (alcohol is L) and (peso is IH) then (salida is EBRIO) (1)
20. If (alcohol is L) and (peso is HG) then (salida is SOBRIO) (1)
21. If (alcohol is AL) and (peso is VL) then (salida is EBRIO) (1)
22. If (alcohol is AL) and (peso is LW) then (salida es EBRIO) (1)
23. If (alcohol is AL) and (peso is I) then (salida is EBRIO) (1)

24. If (alcohol is AL) and (peso is IH) then (salida is EBRIO) (1)
25. If (alcohol is AL) and (peso is HG) then (salida is EBRIO) (1)

De igual forma, en la tabla N° 4, se muestra la Memoria Asociativa Difusa (Fuzzy Associative Memory) correspondiente a las 25 reglas difusas establecidas.

Tabla N° 4: Memoria Asociativa Difusa.

		VARIABLE LINGÜÍSTICA PESO				
		VL	**LW**	**I**	**IH**	**HG**
VARIABLE LINGÜÍSTICA ALCOHOL	**MB**	EBRIO	SOBRIO	SOBRIO	SOBRIO	SOBRIO
	BJ	EBRIO	EBRIO	SOBRIO	SOBRIO	SOBRIO
	CL	EBRIO	EBRIO	EBRIO	SOBRIO	SOBRIO
	L	EBRIO	EBRIO	EBRIO	EBRIO	EBRIO
	AL	EBRIO	EBRIO	EBRIO	EBRIO	EBRIO

Fuente: Propia.

- Diagrama de superficie

Asimismo, con apoyo del Toolbox Fuzzy Logic del Matlab, herramienta Surface del menú View del Editor FIS (Fuzzy Inference System), fue posible representar el diagrama de superficie correspondiente al algoritmo difuso implementado en este proyecto de tesis. Ver la figura N° 20.

Y, tal como se puede observar en la figura anterior, al variar los valores de las entradas lingüísticas “peso” y “alcohol”, la variable de salida irá aumentando

o disminuyendo. Por ejemplo, si nos ubicamos en el punto de intersección correspondiente a una variable de peso igual a 95 y una variable de alcohol igual a 0.1, se observa que la salida responde con un valor menor a 2 lo cual indica que el conductor se encuentra en condiciones óptimas para conducir.

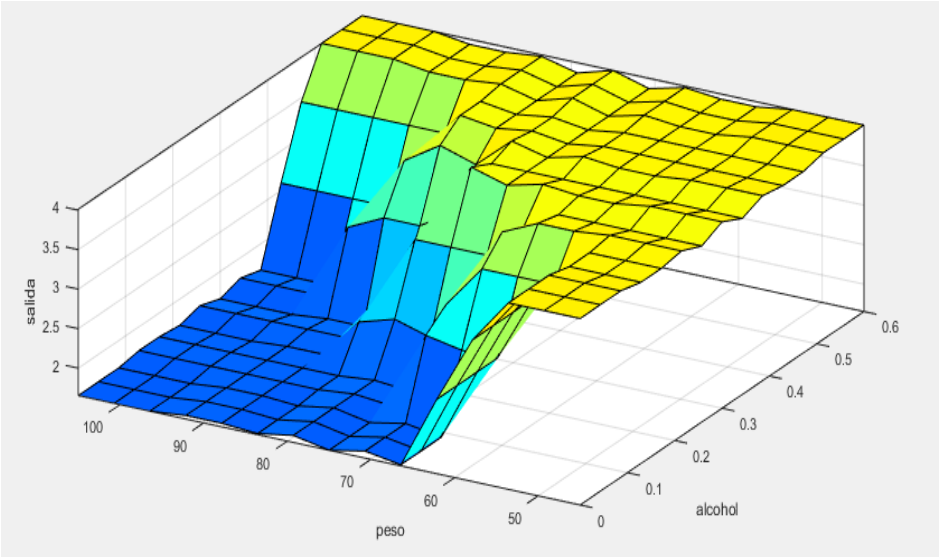

Figura N° 20: Diagrama de Superficie.
Fuente: Propia.

3.2.2 Desarrollo de la base de datos en MySQL

Para el diseño e implementación de la base de datos se utilizó el gestor de base de datos MySQL, con este fin se desarrolló un sistema relacional entre dos tablas, las cuales fueron definidas como tabla de usuarios y tabla de registros. En la primera tabla se definieron los datos de los participantes (muestra de personas) tales como el nombre, peso, tag RFID asignado, además de un código de identificación de uso exclusivo para la base de datos conocida como ID. En la segunda tabla de datos denominada registros, se ingresaron los datos del nivel de alcohol censado y la hora de ocurrencia del suceso. Las tablas fueron programadas en SQL, tal como se muestra en la figura N° 21.

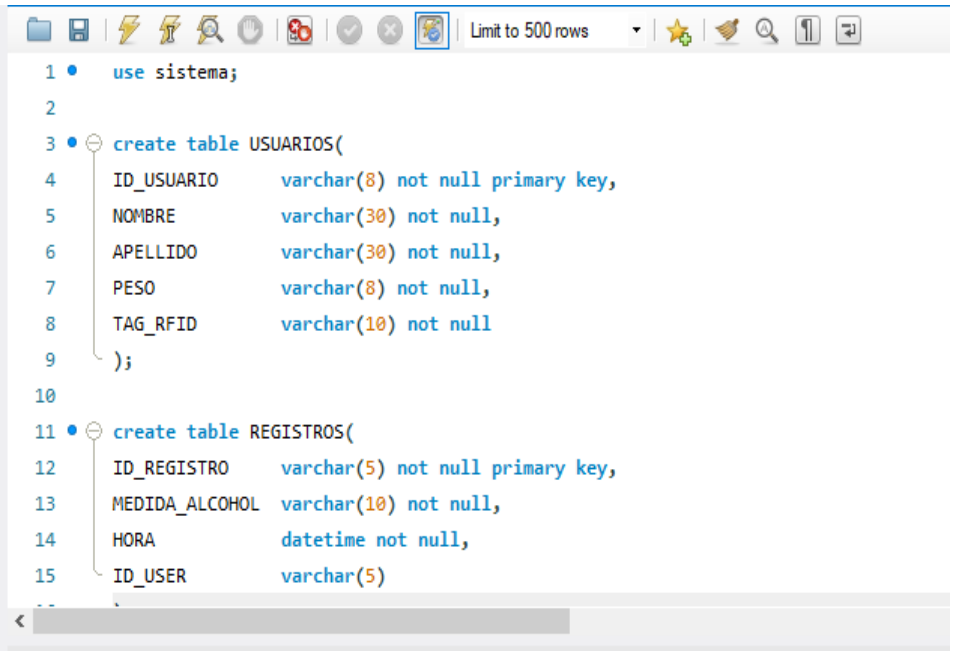

Figura N° 21: Programación de tablas en MySQL
Fuente: Propia.

Además, el gestor brinda las posibilidades de aplicar un campo relacional por lo que se agregó un ID en la tabla de registros (igual que en la primera) formando una correlación, con esto fue posible administrar de maneras más eficiente y sencilla la información de diversas tablas, tal como se puede observar en la figura N° 22.

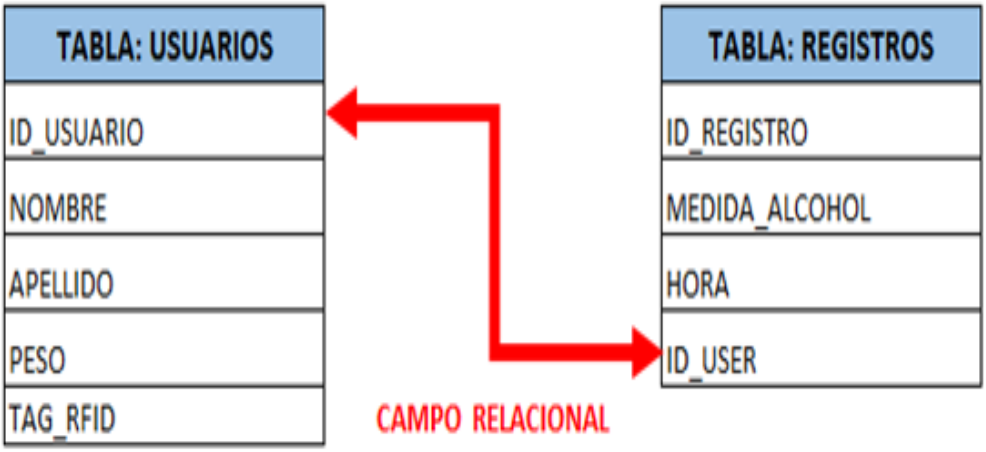

Figura N° 22: Diagrama relacional de la tabla de la base de datos
Fuente: Propia.

3.2.3 Desarrollo e implementación de la interfaz de usuario

La implementación de la interfaz de usuario se llevó acabo en una pantalla HDMI conectada al hardware Raspberry Pi 3. Y, en cuanto al diseño y la reproducción de los mensajes se desarrollaron con el lenguaje de programación Python, el cual ejecuta un mensaje diferente según sea la etapa o estado en la que se encuentre el usuario. Tales mensajes se explican a continuación. Asimismo, parte de la programación desarrollada en lenguaje Python es mostrada a través de una captura de pantalla, ver la figura N° 23 y anexo 4. Mientras que la figura N° 24 muestra una representación general del diagrama de flujo de todos los sucesos ocurridos en cuanto a la interfaz de usuario.

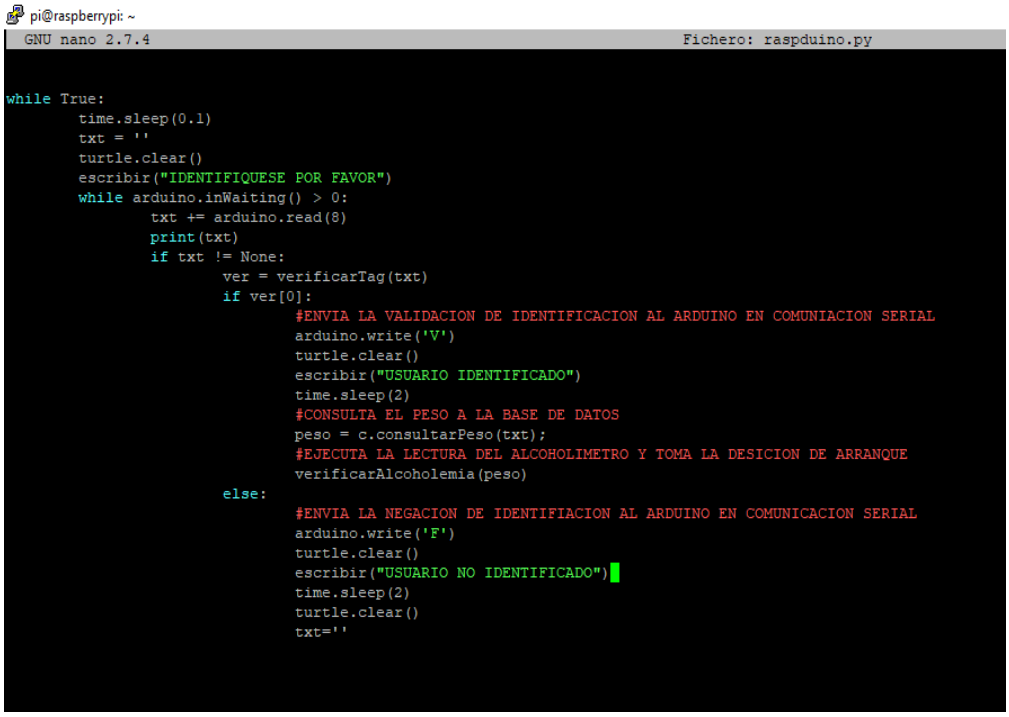

Figura N° 23: Captura del ciclo principal del código del programa
Fuente: Propia.

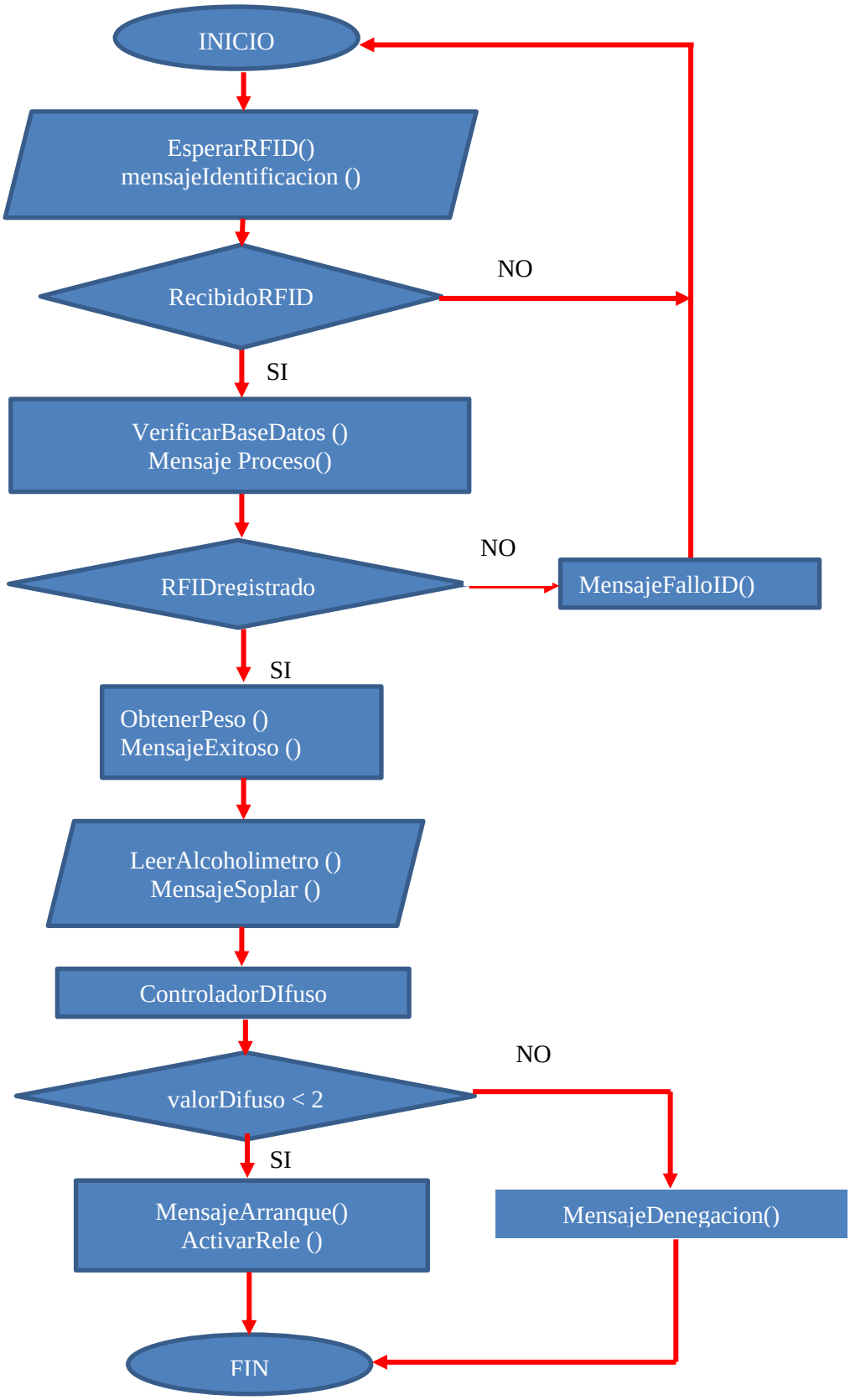

Figura N° 24: Diagrama de flujo del programa principal
Fuente: Propia.

De esta manera, inicialmente se proyecta un mensaje de identificación como se muestra en la figura N° 25, el cual persistirá hasta que el conductor del vehículo se identifique con su tag RFID.

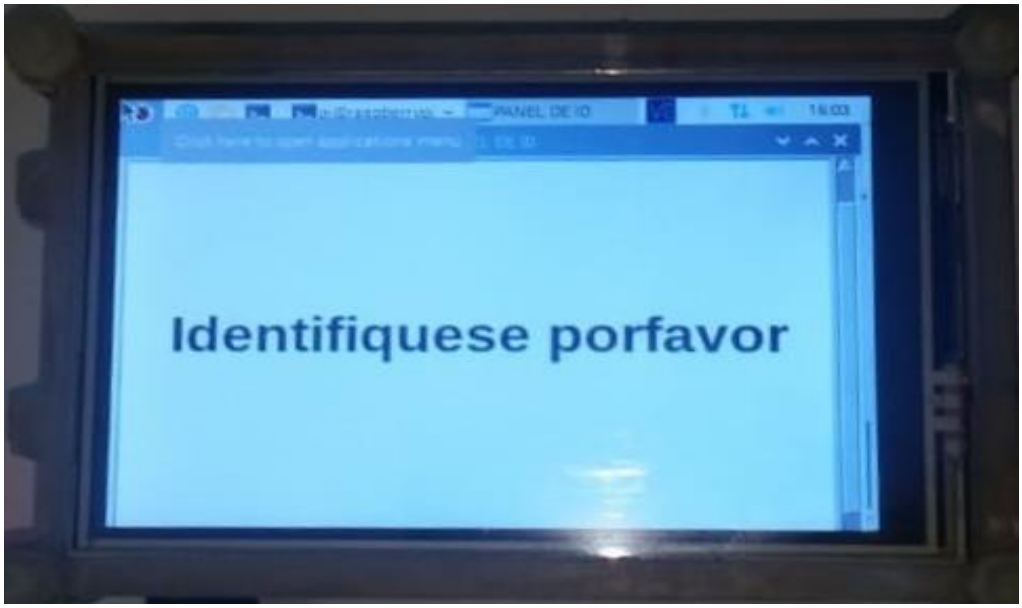

Figura N° 25: Pantalla de inicio de sistema
Fuente: Propia.

Inmediatamente después de que el usuario utilizó su tag RFID, el sistema consulta a la base de datos, cuyo modulo en python se detalla en el anexo 3; para luego tomarse un tiempo y procesar la data recibida, tal como se muestra en la figura N° 26.

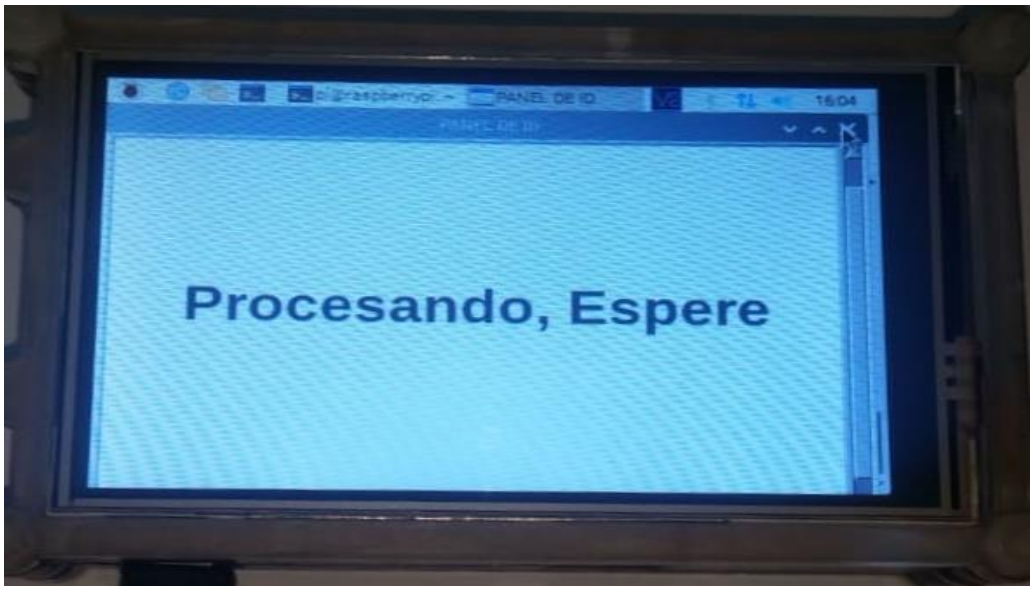

Figura N° 26: Pantalla de procesamiento del RFID.

Fuente: Propia.

Una vez terminado el procesamiento del tag RFID el sistema presenta dos diferentes escenarios: uno exitoso y otro de alerta. El caso exitoso corresponde a un usuario identificado en la base de datos existente. Ver la figura N° 27.

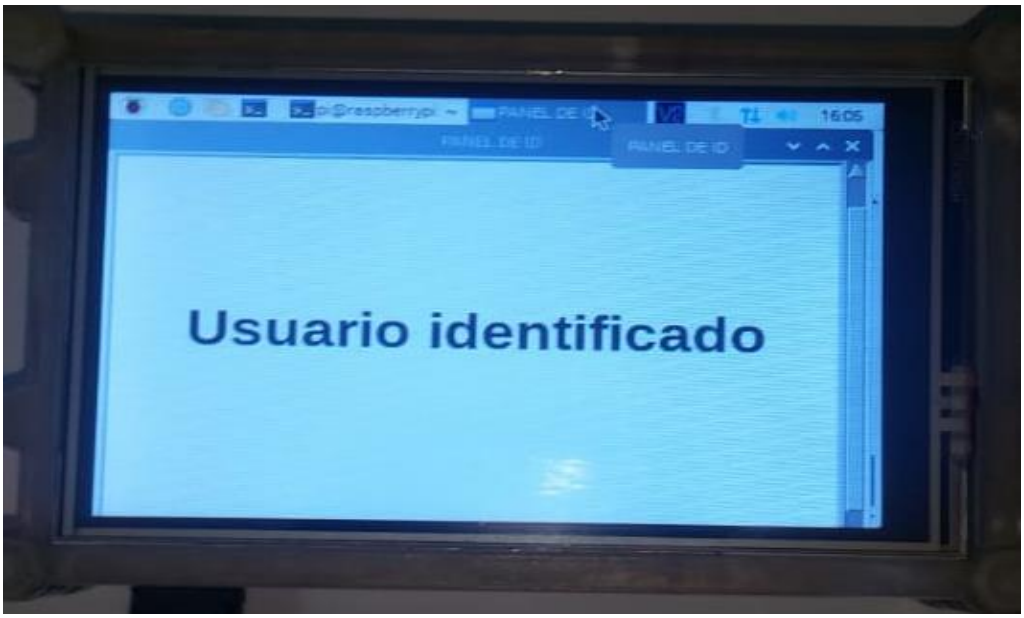

Figura N° 27: Panel de usuario identificado.
Fuente: Propia.

En caso contrario se muestra un mensaje de alerta denegando el acceso al usuario. Ver la figura N° 28.

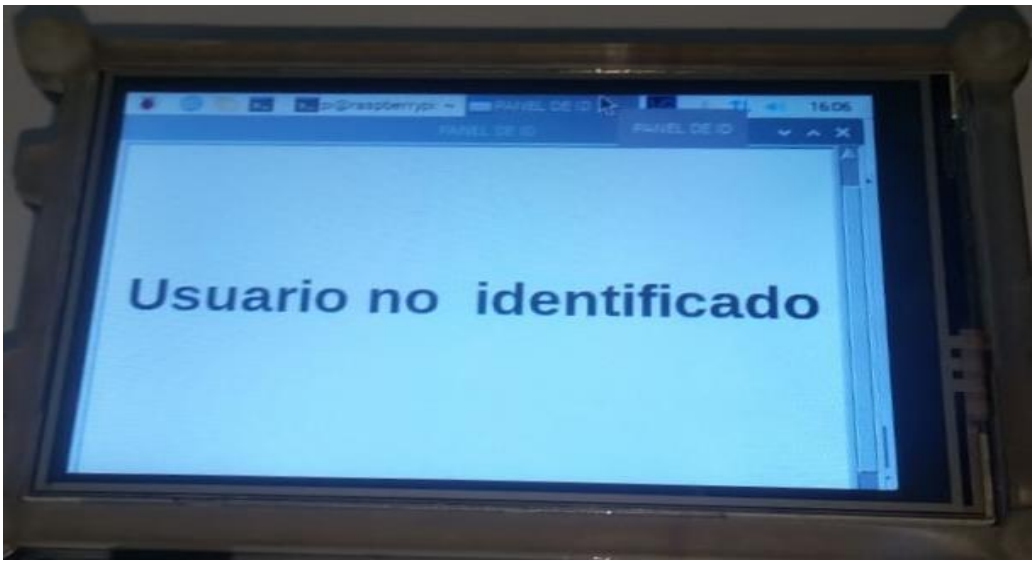

Figura N° 28: Pantalla de usuario no identificado.
Fuente: Propia.

Para el escenario exitoso se muestra un mensaje, en la figura N° 29, pidiendo al usuario que sople sobre el sensor MQ-3.

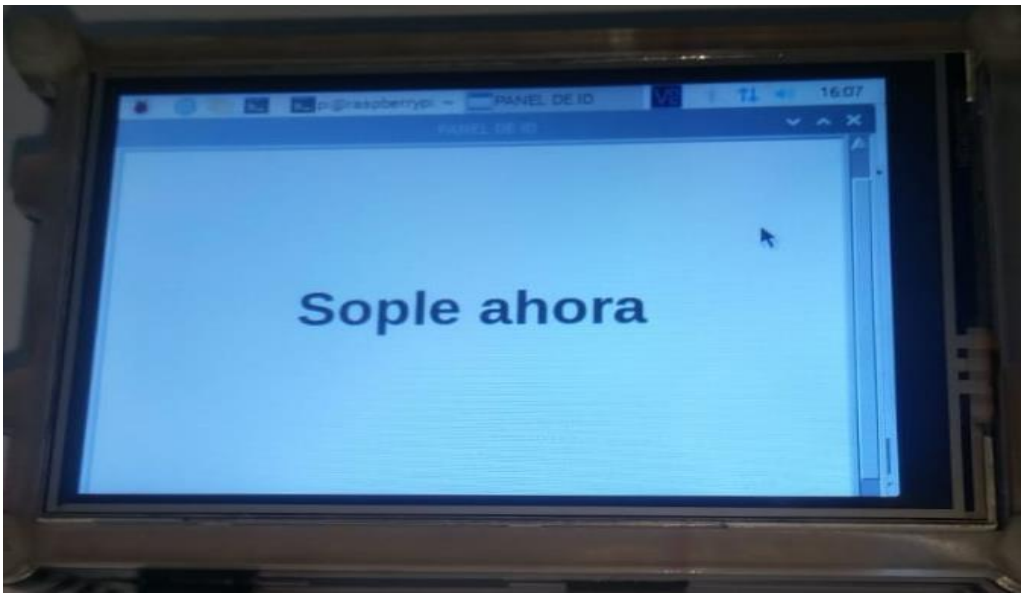

Figura N° 29: Pantalla pidiendo al usuario soplar en el sensor.
Fuente: Propia.

En esta instancia se presentaron dos casos. El primero, cuando el usuario pasó la prueba de alcoholemia implementada en el algoritmo difuso, por lo cual se observa el mensaje mostrado en la figura N° 30.

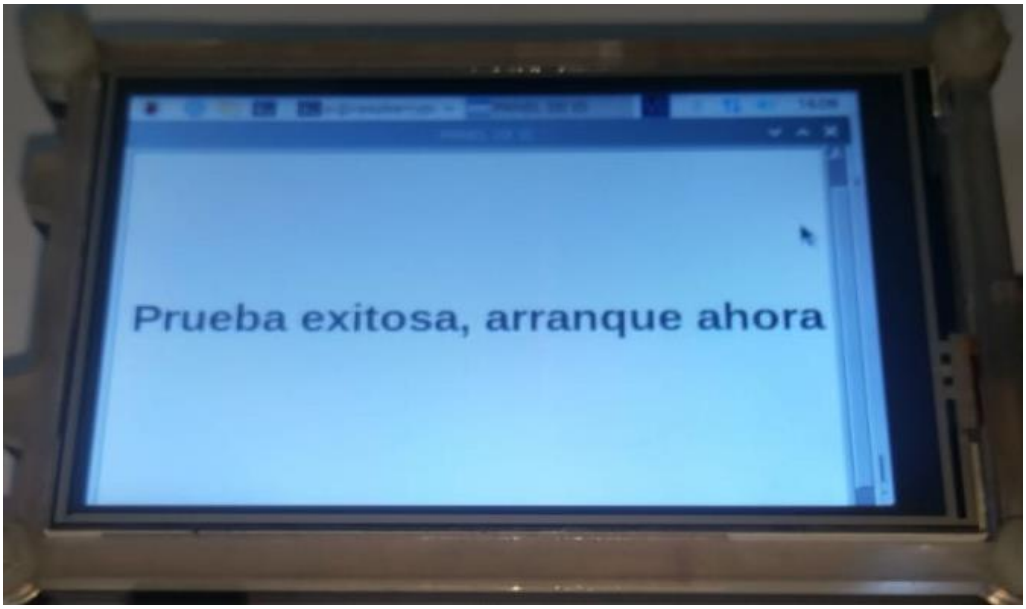

Figura N° 30: Pantalla de prueba exitosa.
Fuente: Propia.

El segundo caso fue cuando falló en la prueba de alcoholemia, por lo que se observa el mensaje mostrado en la figura N° 31.

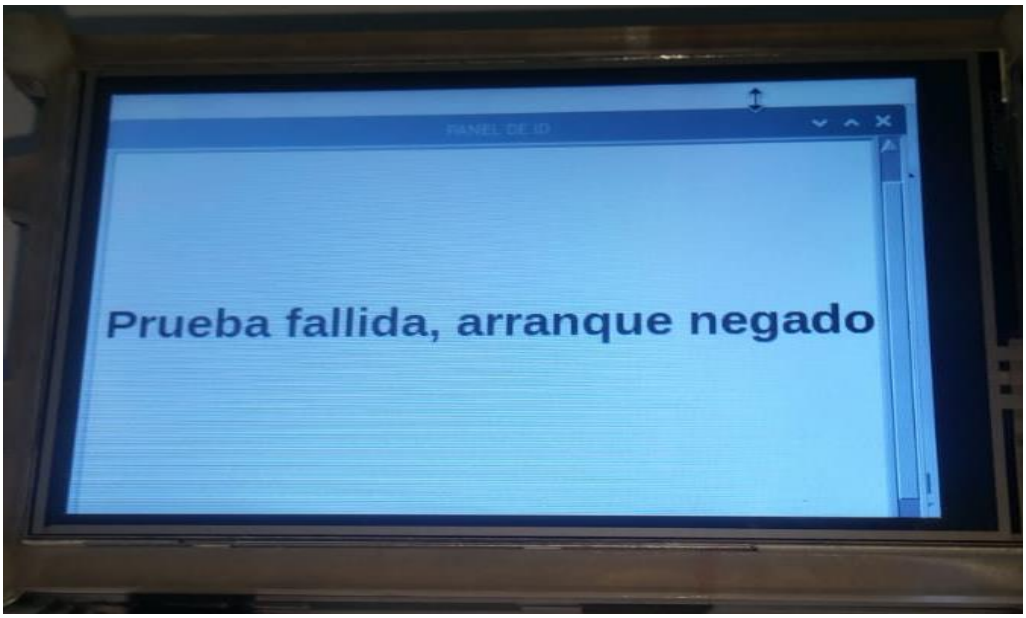

Figura N° 31: Imagen de prueba fallida.
Fuente: Propia.

3.3. Integración de Hardware y Software.

Para la integración del proyecto se optó en primera instancia plantear de manera lógica cada etapa, de las cuales la primera y más importante fue la lectura de las variables lingüísticas como son el alcohol y el peso. Para este ciclo se realizó un análisis minucioso acerca de los niveles de lectura de cada componente utilizado, esto debido a que, como fue en el caso del censado de alcohol, se presentaron pequeñas contingencias; por ello, se acondicionó al sensor de alcohol (MQ-3) el uso de los sensores flujómetro con la finalidad de evitar que este sensor de alcohol capte lecturas erróneas, tales como la medición de niveles de alcohol a causa de perfumes, alcohol medicinal, aromatizantes, etc. Por lo tanto, este acondicionamiento consistió en la configuración de la programación, para que el censado utilizando el MQ-3 se accione según el nivel del flujo que capte el flujómetro, cuyas características se detallan en el anexo 2. Además, se buscó la manera de tener a los sensores integrados en un solo accesorio, sujetándolos con dos objetos en forma de pilares, para permitir su eficiente uso. Y, finalmente se diseñó un componente adicional que se colocó en la entrada del sensor de flujo, con la finalidad de ser utilizado una sola vez por cada

conductor al que se le realizó las pruebas, evitando así residuos de partículas de alcohol impregnadas en el flujómetro y como también por tema de salubridad. Todo ello se observa en la figura N° 32.

Figura N° 32: Accesorios para la lectura de nivel de alcohol.
Fuente: Propia.

Además, para la lectura del peso de cada conductor también se presentó una contingencia, la cual consistió en validar la información de cada *tag* correspondiente a la tarjeta RFID que llevaría cada conductor, por lo que se optó por implementar un archivo bloc dentro del Raspberry Pi 3, para proporcionar estos datos cada vez que un conductor haga uso de algún vehículo en donde se encuentre implementado el proyecto de tesis propuesto.

Seguidamente, una vez pasada la fase de lectura de variables, se procedió a plantear la respuesta del algoritmo después de haber sido analizada e inferida la información mediante el control difuso, cuyas características se detallan en el anexo 1. Por lo que se decidió enviar una señal de respuesta de 5 VDC hacia el relé de bloque para permitir el arranque vehicular. Además, se diseñó una interacción de manera gráfica con el conductor a través de una pantalla HDMI incorporada en el algoritmo principal Raspberry Pi 3, tal como se muestra en la figura N° 33.

Seguidamente, se buscó la manera de implementar todo el proyecto dentro de un vehículo, por lo que se tuvo que tener en concideración tanto el nivel de voltaje de la batería vehicular como el nivel de voltaje con el que se alimenta el Raspberry Pi 3. Asi que, dado que las baterías de los vehículos por lo general son de 12 VDC, y el

Raspberry Pi 3 como máximo recibe 5 VDC de alimentacion, se diseñó un relugador de voltaje para este valor, con lo cual fue posible su funcionamiento. Este a su vez logró alimentar al Atmega328p junto con todos sus perifericos como lo son: el flujómetro, el sensor MQ-3, lector RFID y el relé de bloqueo, tal como se muestra en la figura N° 34.

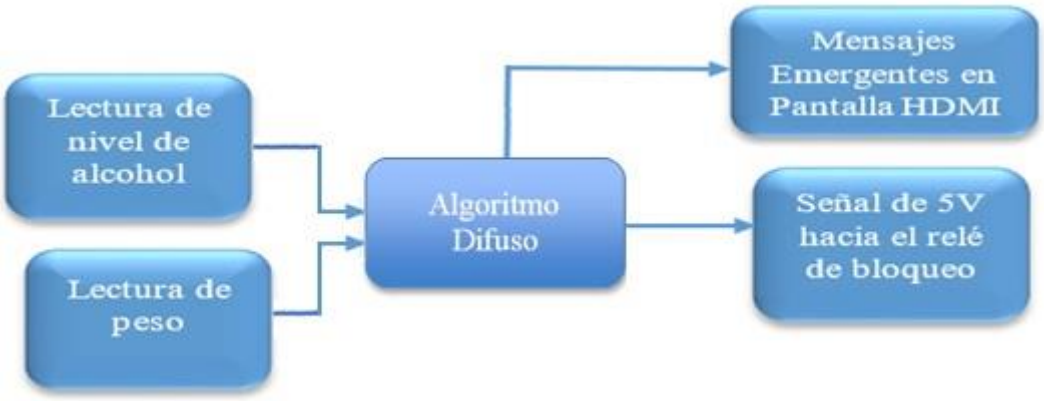

Figura N° 33: Diagrama de bloques general del proyecto de tesis.
Fuente: Propia.

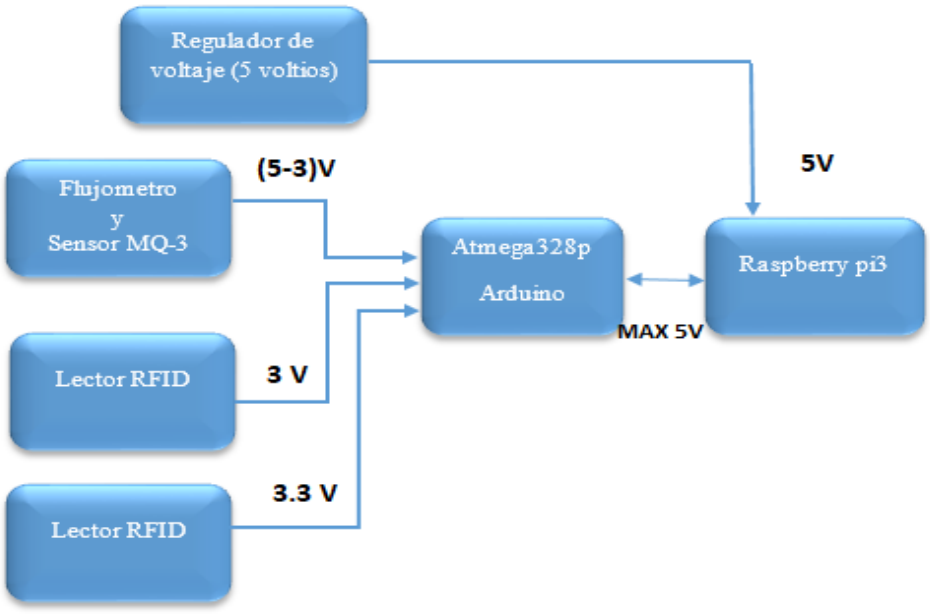

Figura N° 34: Diagrama de bloques de las conexiones de entrada del Raspberry Pi 3.
Fuente: Propia.

Finalmente, se muestra un diagrama pictórico, el cual buscó representar de manera completa y compresible la integración del hardware del proyecto, mostrando la

apariencia física de cada elemento que formó parte del diseño, el cual estuvo compuesto desde los sensores de lectura de datos hasta la pantalla HDMI (interfaz multimedia de alta definición), la cual entregaba información al usuario acerca del proceso que se llevaba a cabo mediante su interacción con el programa. Ver la figura N° 35.

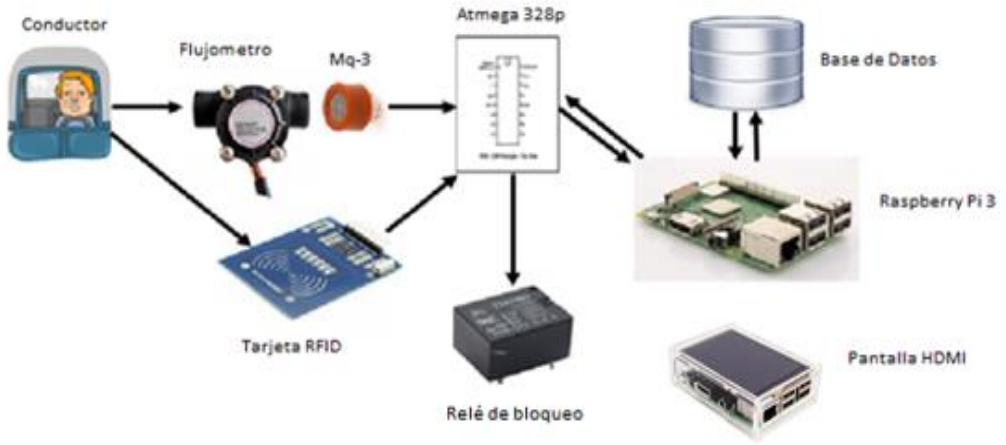

Figura N° 35: Diagrama pictórico del proyecto.
Fuente: Propia.

Asimismo, la figura N° 36 muestra la fotografía del montaje completo del prototipo propuesto en este proyecto de tesis, tanto en el salón del vehículo de marcha mecánica, como en la parte delantera del mismo. Además, como se puede observar, el sensor MQ-3 ha sido instalado en la parte delantera del vehículo junto al flujómetro teniendo en cuenta que los dos forman un arreglo para tener un censado confiable, el lector de RFID para facilitar la validación del conductor fue colocado en la parte derecha del timón, por otro lado la pantalla fue colocada encima del tablero del conductor para mayor visualización al momento de interacción entre el usuario y el sistema; finalmente el relé ha sido acondicionado al pedal de embrague del vehículo puesto a la facilidad y que además, este permite bloquear la línea de tierra hacia el relé de arranque.

Figura N° 36: Fotografía de la propuesta de tesis montado en un vehículo.
Fuente: Propia.

CAPÍTULO IV. PRUEBAS Y RESULTADOS

En el presente capítulo se muestran los resultados obtenidos en las pruebas realizadas con el prototipo desarrollado. Para ello, este capítulo fue subdivido en 4 secciones, de las cuales 3 están dedicadas a mostrar y analizar los resultados obtenidos para 3 casos diferentes (combinaciones de alcohol en la sangre y peso), mientras que la sección restante muestra todos los resultados obtenidos del conjunto de prueba en una tabla resumen.

4.1. Resultados de la primera prueba.

La primera prueba fue realizada con un individuo de 85 kg, el cual ingirió una lata de cerveza de contenido 355 ml. La prueba fue tomada pasados 30 minutos de la ingesta de la bebida, con motivos de tener una lectura más precisa. Por lo tanto, como resultado se obtuvo en el alcoholímetro un valor de 0.19 g/l. Este resultado permitió que el sujeto arranque el vehículo. Como prueba de ello se muestra el mensaje de prueba exitosa en la pantalla LCD y el encendido de un foco (el cual simula el arranque del vehículo). Ver Figura N° 37.

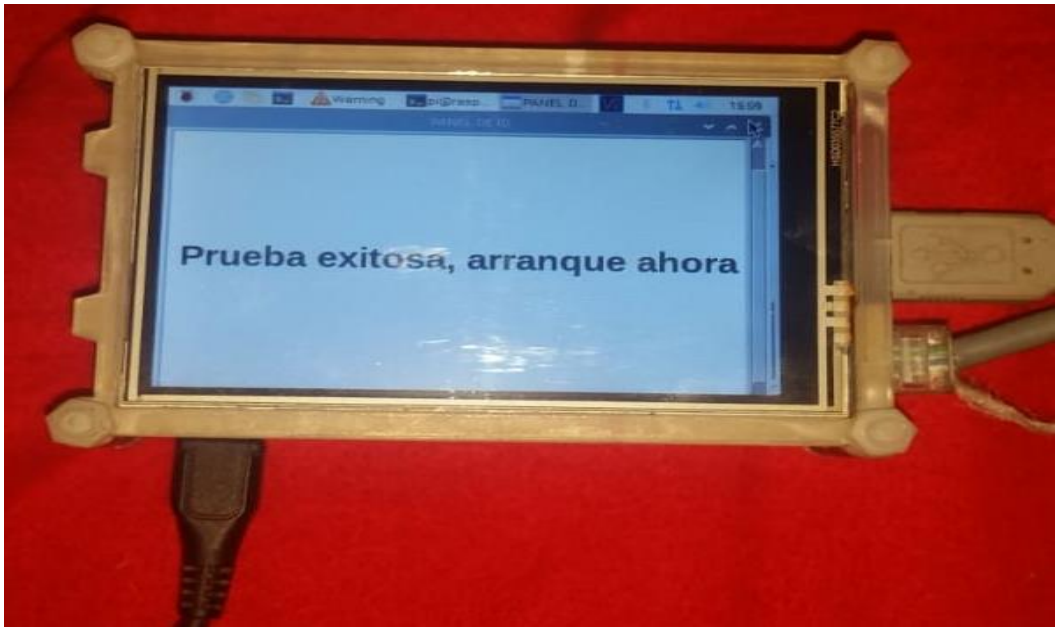

Figura N° 37: Pantalla de resultado de la prueba
Fuente: Propia.

Tal como se describió anteriormente, el resultado fue registrado en la base de datos del estudio, razón por la que fue posible analizar el desempeño del algoritmo difuso. Entonces, en primer lugar, el peso del participante se encontró dentro del conjunto

difuso I (“Intermedium”) y el grado de alcohol en la sangre dentro del conjunto MB (“Muy Bajo”), los cuales fueron analizados con el conjunto de reglas difusas establecidas y arrojaron un resultado correspondiente al conjunto de salida “Sobrio”. De esta manera, el valor de salida del algoritmo difuso obtenido por el prototipo y registrado en la base de datos fue igual a 1.1, valor que en efecto corresponde al conjunto “Sobrio”. Por lo tanto, se comprobó que el prototipo obtuvo una respuesta de acuerdo a la experiencia con la que se estableció el algoritmo difuso. En la figura N° 38 se muestra los valores de las variables Peso y Alcohol, así como el ID del usuario y el respectivo valor difuso. Tal representación corresponde a una captura de pantalla del programa MySQL WorkBench.

	ID_USUARIO	PESO	MEDIDA_ALCOHOL	VALOR_DIFUSO
▶	47859898	85	0.19	1.1

Figura N° 38: Valores en la base de datos para la primera prueba
Fuente: Propia.

Paralelamente, se realizaron las comprobaciones con apoyo del Toolbox Fuzzy Logic del Matlab. Por lo cual, en la figura N° 39 se observan tres columnas, de las cuales las dos primeras representan tanto el nivel de alcohol en la sangre con un 0.19 g/l así como el peso de la persona que en esta prueba correspondió a 85 kg; es así que, estos valores generan una serie de conjuntos difusos de color amarillo que mediante la inferencia dada por las reglas difusas se pudo obtener uno o más conjuntos difusos de salida de color azul (operación de implicación). Y, mediante el método de agregación se procedió a la unión estos últimos conjuntos permitiendo la fusificación. Posteriormente, utilizando el método del centroide, se logró realizar la defusificación obteniendo finalmente un valor de salida igual a 1.66.

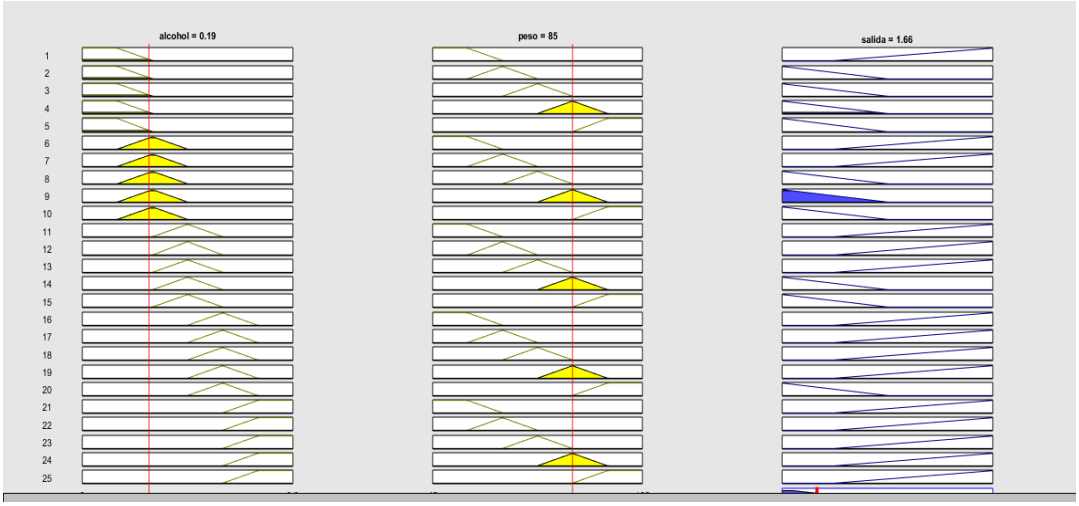

Figura N° 39: Rule Viewer Fuzzy from test 1.
Fuente: Propia.

4.2. Resultados de la segunda prueba.

La segunda prueba fue realizada con un individuo de 62 kg, el cual ingirió una copa de vino cuyo contenido aproximadamente fue de 90 ml. La prueba fue tomada pasados 30 minutos de la ingesta de la bebida, con motivos de tener una lectura más precisa. Por lo tanto, como resultado se obtuvo en el alcoholímetro un valor de 0.3 g/l. Este resultado no permitió que el sujeto arranque el vehículo. Como prueba de ello se muestra el mensaje de prueba fallida en la pantalla LCD y la ausencia del encendido de un foco (el cual simula el NO arranque del vehículo). Ver la figura N° 40.

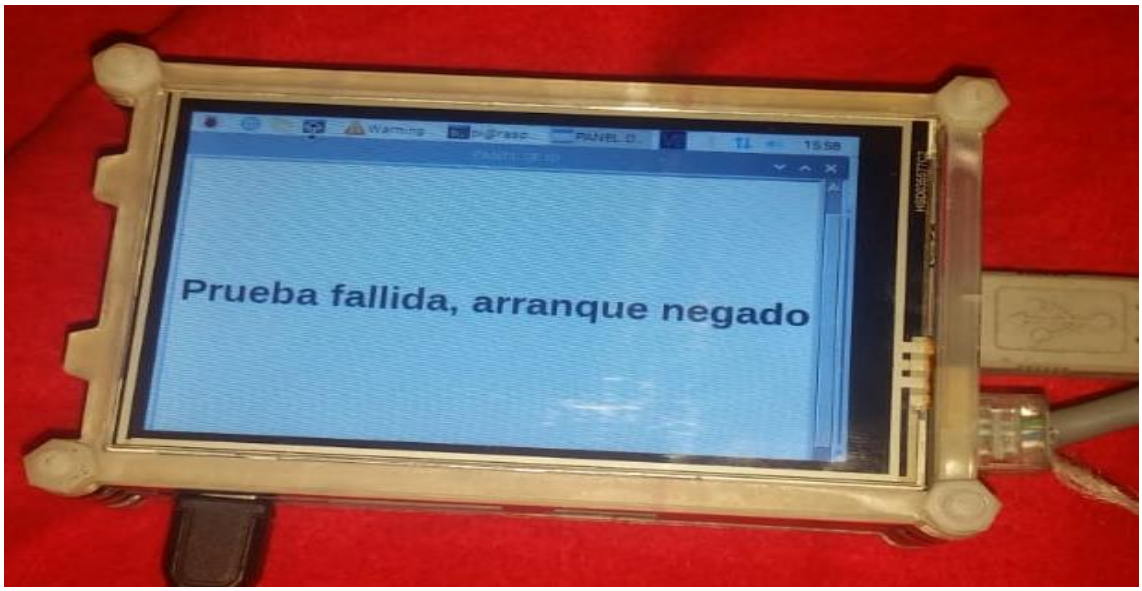

Figura N° 40: Resultado negativo obtenido de esta prueba.
Fuente: Propia.

Tal como se describió anteriormente, el resultado fue registrado en la base de datos del estudio, razón por la que fue posible analizar el desempeño del algoritmo difuso. Entonces, en primer lugar, el peso del participante se encontró dentro de los conjuntos difusos VL (“Very Low”) y LW (“Low”) mientras que el grado de alcohol en la sangre dentro del conjunto CL (“Casi en límite”), los cuales fueron analizados con el conjunto de reglas difusas establecidas y arrojaron un resultado correspondiente al conjunto de salida “Ebrio”. De esta manera, el valor de salida del algoritmo difuso obtenido por el prototipo y registrado en la base de datos fue igual a 3.57, valor que en efecto corresponde al conjunto “Ebrio”. Por lo tanto, se comprobó que el prototipo obtuvo una vez más una respuesta de acuerdo a la experiencia con la que se estableció el algoritmo difuso. En la figura N° 41 se muestra los valores de las variables Peso y Alcohol, así como el ID del usuario y el respectivo valor difuso. Tal representación corresponde a una captura de pantalla del programa MySQL Workbench.

	ID_USUARIO	PESO	MEDIDA_ALCOHOL	VALOR_DIFUSO
▶	21451183	62	0.3	3.57

Figura N° 41: Valores en la base de datos para la segunda prueba.
Fuente: Propia.

Paralelamente, se realizaron las comprobaciones con apoyo del Toolbox Fuzzy Logic del Matlab. Por lo cual, en la figura N° 42 se observan tres columnas, de las cuales las dos primeras representan tanto el nivel de alcohol en la sangre con un 0.3 g/l así como el peso de la persona que en esta prueba correspondió a 62 kg; es así que, estos valores generan una serie de conjuntos difusos de color amarillo que mediante la inferencia dada por las reglas difusas se pudo obtener uno o más conjuntos difusos de salida de color azul (operación de implicación). Y, mediante el método de agregación se procedió a la unión estos últimos conjuntos permitiendo la fusificación. Posteriormente, utilizando el método del centroide, se logró realizar la defusificación obteniendo finalmente un valor de salida igual a 3.94.

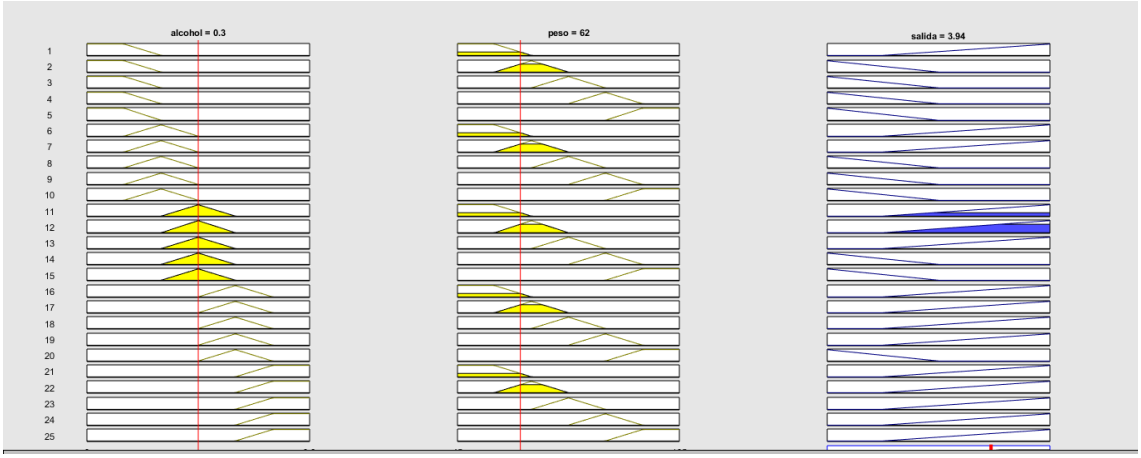

Figura N° 42: Rule Viewer Fuzzy from test 2.
Fuente: Propia.

4.3. Resultados de la tercera prueba.

La tercera prueba fue realizada con un individuo de 95 kg, el cual ingirió un vaso de Whisky cuyo contenido aproximadamente fue de 44 ml. La prueba fue tomada pasados 30 minutos de la ingesta de la bebida, con motivos de tener una lectura más precisa. Por lo tanto, como resultado se obtuvo en el alcoholímetro un valor de 0.42 g/l. Este resultado no permitió que el sujeto arranque el vehículo. Como prueba de ello se muestra el mensaje de prueba fallida en la pantalla LCD y la ausencia del encendido de un foco (el cual simula el NO arranque del vehículo). Ver la figura N° 43.

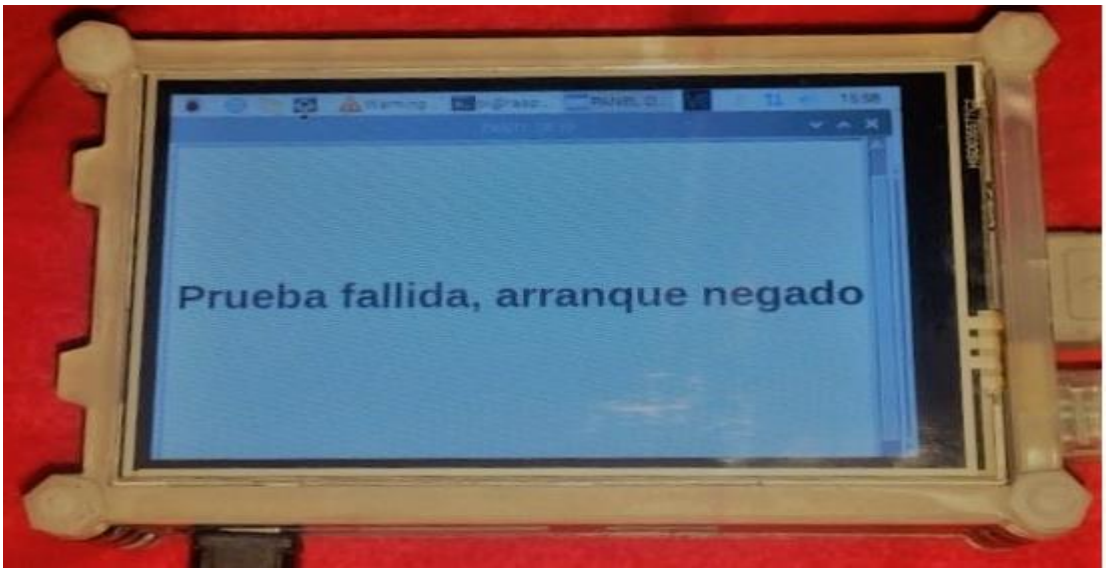

Figura N° 43: Pantalla que indica el bloqueo del motor.
Fuente: Propia.

Tal como se describió anteriormente, el resultado fue registrado en la base de datos del estudio, razón por la que fue posible analizar el desempeño del algoritmo difuso. Entonces, en primer lugar, el peso del participante se encontró dentro de los conjuntos difusos IH (“Inter-High”) y H (“High”) mientras que el grado de alcohol en la sangre dentro de los conjuntos AL (“Al límite”) y A (“Alto”), los cuales fueron analizados con el conjunto de reglas difusas establecidas y arrojaron un resultado correspondiente al conjunto de salida “Ebrio”. De esta manera, el valor de salida del algoritmo difuso obtenido por el prototipo y registrado en la base de datos fue igual a 2.87, valor que en efecto corresponde al conjunto “Ebrio”. Por lo tanto, se comprobó que el prototipo obtuvo una vez más una respuesta de acuerdo a la experiencia con la que se estableció el algoritmo difuso. En la figura N° 44 se muestra los valores de las variables Peso y Alcohol, así como el ID del usuario y el respectivo valor difuso. Tal representación corresponde a una captura de pantalla del programa MySQL Workbench.

	ID_USUARIO	PESO	MEDIDA_ALCOHOL	VALOR_DIFUSO
▶	08833443	95	0.42	2.87

Figura N° 44: Valores registrados en la base de datos para el participante
Fuente: Propia.

Paralelamente, se realizaron las comprobaciones con apoyo del Toolbox Fuzzy Logic del Matlab. Por lo cual, en la figura N° 45 se observan tres columnas, de las cuales las dos primeras representan tanto el nivel de alcohol en la sangre con un 0.42 g/l así como el peso de la persona que en esta prueba correspondió a 95 kg; es así que, estos valores generan una serie de conjuntos difusos de color amarillo que mediante la inferencia dada por las reglas difusas se pudo obtener uno o más conjuntos difusos de salida de color azul (operación de implicación). Y, mediante el método de agregación se procedió a la unión estos últimos conjuntos permitiendo la fusificación. Posteriormente, utilizando el método del centroide, se logró realizar la defusificación obteniendo finalmente un valor de salida igual a 2.37.

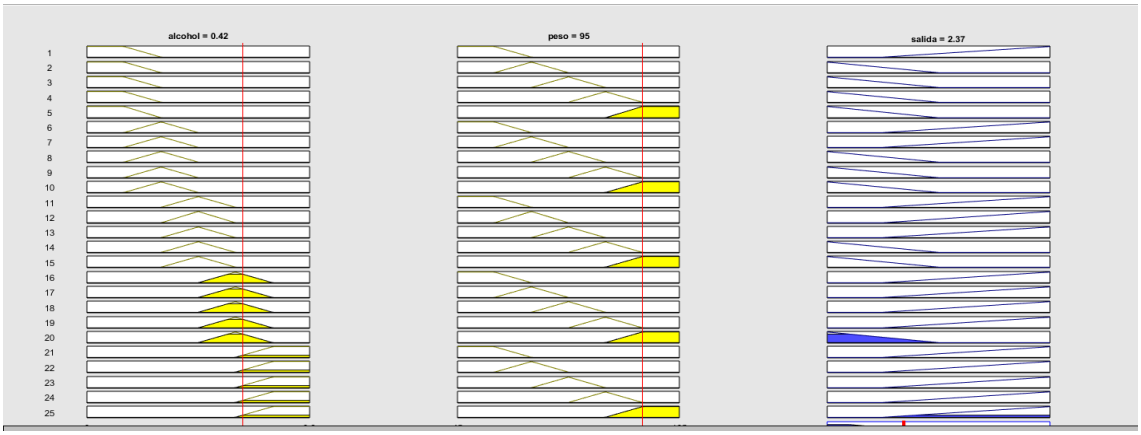

Figura N° 45: Rule Viewer Fuzzy from test 3.
Fuente: Propia.

4.4. Resultados de la muestra.

Tal como fue mencionado en el primer capítulo de este proyecto de tesis, se ha empleado una muestra de 30 personas, de ambos sexos, con edades comprendidas entre 18 y 65 años de edad, así como con un peso fluctuando entre los 45 y 105 Kg. Asimismo, se ha considerado la ingesta de 5 tipos de licores los cuales corresponden a Cerveza (una lata de 355 ml), Vino (una copa de 90 ml), Pisco (una copa de 45 ml), Ron (una copa de 44 ml) y Whisky (un vaso de 44 ml). Entonces, con el fin de mostrar todos los valores recaudados en las pruebas realizadas, se organizó la siguiente tabla resumen.

Tabla N° 5: Tabla general de muestras realizadas.

Persona	Peso	Bebida Alcohólica	Nivel de Alcohol	Valor de Matlab	Valor de Raspberry	ACCIÓN DEL RELÉ
1	85	Cerveza	0.19	1.66	1.1	ARRANCAR
2	62	Vino	0.3	3.94	3.57	NO ARRANCAR
3	95	Pisco	0.42	2.37	2.87	NO ARRANCAR
4	83	Ron	0.22	2.37	2.81	NO ARRANCAR
5	110	Whisky	0.18	1.68	1.2	ARRANCAR
6	79	Pisco	0.14	1.73	1.34	ARRANCAR
7	77	Vino	0.23	2.63	2.87	NO ARRANCAR
8	62	Cerveza	0.35	3.84	3.57	NO ARRANCAR
9	59	Ron	0.58	3.9	3.71	NO ARRANCAR
10	56	Whisky	0.64	4	5	NO ARRANCAR
11	54	Pisco	0.44	3.9	3.71	NO ARRANCAR
12	62	Vino	0.41	3,9	3.57	NO ARRANCAR
13	74	Cerveza	0.18	2.08	2.8	NO ARRANCAR
14	82	Ron	0.23	2.63	2.83	NO ARRANCAR
15	58	Whisky	0.2	3.9	3.57	NO ARRANCAR
16	69	Pisco	0.27	3.35	3.11	NO ARRANCAR
17	96	Vino	0.09	1.65	1	ARRANCAR
18	52	Cerveza	0.4	4.01	3.57	NO ARRANCAR
19	86	Ron	0.24	1.73	1.4	ARRANCAR
20	71	Whisky	0.42	3.9	3.57	NO ARRANCAR
21	65	Pisco	0.31	4	3.57	NO ARRANCAR
22	81	Vino	0.17	1.73	1.4	ARRANCAR
23	66	Cerveza	0.31	4	3.57	NO ARRANCAR
24	77	Ron	0.29	3.62	3.19	NO ARRANCAR

25	59	Whisky	0.61	19	5	NO ARRANCAR
26	62	Pisco	0.33	3.94	3.57	NO ARRANCAR
27	58	Vino	0.44	3.9	3.57	NO ARRANCAR
28	71	Cerveza	0.23	2.84	3	NO ARRANCAR
29	65	Ron	0.37	3.94	3.57	NO ARRANCAR
30	84	Cerveza	0.12	1.68	1.2	ARRANCAR

Fuente: Propia.

De la tabla anterior, se observa que para los casos de las personas 10 y 25 se obtuvieron valores iguales a 5 con el algoritmo desarrollado en el Raspberry Pi 3. Esto indica que el dato obtenido de alcohol en la sangre se encontró fuera del rango de valores contemplados en el respectivo conjunto difuso. Por lo cual, para ambos casos dichos valores resultaron mayores a 0.6, razón por la cual, y debido a que en el prototipo estos valores fueron tomados como demasiado altos, dichos valores se redondearon a un valor de 5. Inhabilitando así el arranque del motor.

A continuación, se realizó un gráfico de torta (ver figura N° 46) en el cual se representan los porcentajes de pruebas con el prototipo desarrollado, permitiendo y evitando el arranque del automóvil. Entonces, como se puede observar, se obtuvo mayor cantidad de escenarios con prohibición de arranque, debido a que estos escenarios fueron los más probables y los que resultaron más comunes durante el desarrollo del proyecto de tesis.

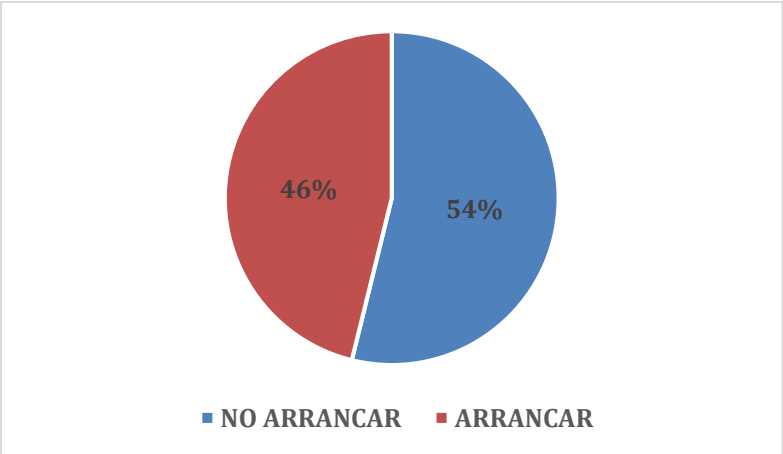

Figura N° 46: Gráfico de pastel para las decisiones tomadas por el prototipo desarrollado para cada una de las pruebas.
Fuente: Elaboración propia

Adicionalmente, se presenta otro gráfico de pastel (Ver figura N° 47) para mostrar las cantidades de licores usados. Siendo el de mayor consumo la cerveza y el de menor el whisky.

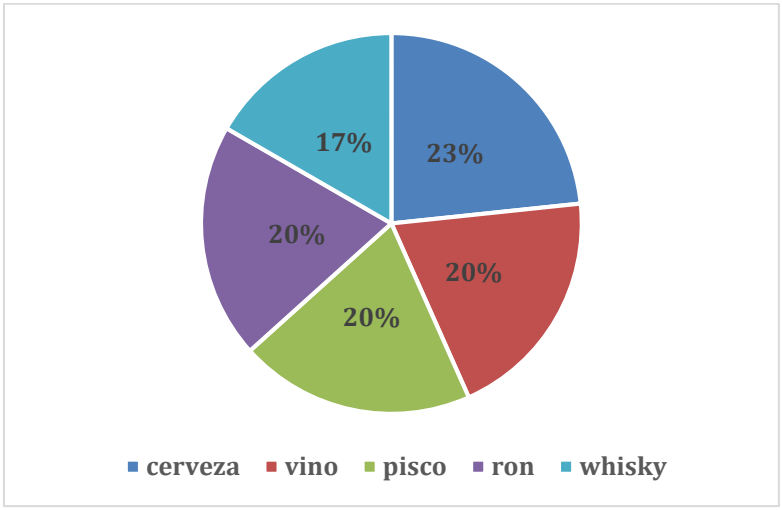

Figura N° 47: Gráfico de pastel representativo de las cantidades de licores usados.
Fuentes: Elaboración propia

Otro gráfico considerado de utilidad es el mostrado en la figura N° 48, donde se muestran los casos estudiados representados en un gráfico de barras. Además, se observan los dos escenarios posibles por tipo de licor utilizado, siendo las barras de color azul los casos en cuyo arranque fue permitido y de forma contraria las barras rojas los casos de denegación del arranque.

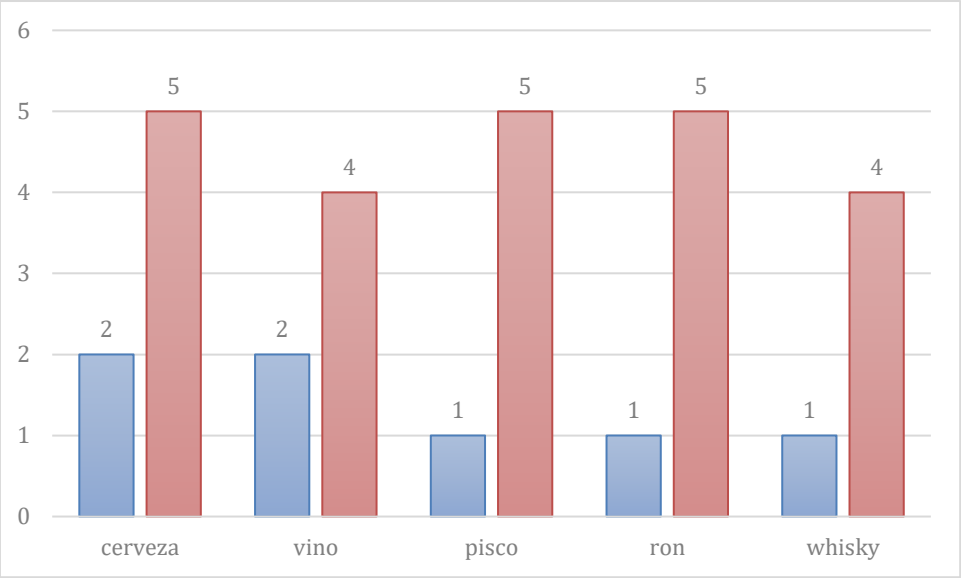

Figura N° 48: Gráfico de barras para las pruebas realizadas según el tipo de alcohol
Fuente: Elaboración propia

Respecto a la fiabilidad del prototipo propuesto, se pudo notar que este cumple su función o el comportamiento es el esperado en todas las pruebas. Esto debido a que el valor umbral seleccionado para la inferencia en el conjunto “EBRIO” se toma en el inicio de dicho conjunto, asignándose el valor de 2 (según los parámetros referenciados del conjunto) por lo que cualquier valor superior al umbral activará el relé. De esta forma, el algoritmo difuso resultó ser altamente fiable.

CONCLUSIONES

1. Se diseñó un algoritmo difuso basado en el modelo Mandani, el cual tiene 2 variables lingüísticas de entrada y 25 reglas difusas. Todo ello se simuló, inicialmente, en el software Matlab. Luego, fue trasladado al lenguaje de programación Python el cual tiene tipado dinámico lo que hace al código más ágil, permitiendo de esta manera la implementación en el micro ordenador Raspberry PI 3 tal como se observa en la sección 3.2 del tercer capítulo.

2. Asimismo, se diseñó un arreglo de dos sensores haciendo uso del MQ-3 y del Flujómetro YF-S201, tal como se muestra en la sección 3.3 del tercer capítulo. La lectura del MQ-3 se condicionó al nivel representado en valores análogos del flujómetro de tal manera que se descartaran mediciones falsas en el MQ3, tales como medidas del ambiente o mal ejecutadas. De esta forma, sólo se activó la medición cuando se comprobó que el flujo es proveniente de un soplido.

3. Y de igual forma, se logró intervenir las líneas de señal del relé de arranque del automóvil, con lo que se pudo agregar un relé adicional de bloqueo, tal como se muestra en la sección 3.1.2 del capítulo 3. Esto permitió condicionar el paso de señal de la llave de arranque siempre que la función de membresía obtenida en la variable lingüística de salida, sea de SOBRIO. Esto se observa en la figura N°19.

RECOMENDACIONES

Se recomienda analizar los tiempos de procesamiento de diferentes lenguajes de programación a la hora de simular un algoritmo difuso, puesto que en esta etapa el código principal toma un intervalo de tiempo de espera, en donde el algoritmo difuso comienza a trabajar y proporciona un resultado. Este procedimiento de ser muy alto afectaría la eficiencia del proyecto.

Además, se recomienda hacer un análisis profundo de sistemas de telemetría que permitan llevar un control y monitoreo eficiente y de bajo costo para la implementación de proyectos que requieran de un seguimiento constante, haciendo uso de sistemas GSM o de sistemas CLOUD donde se puede encontrar diferentes plataformas de alto rendimiento como lo son: AWS, Google, Azure, Ubidots, Atmosphera, etc.

Así también se recomienda mantener el uso de un respaldo local que contenga información solicitada en el funcionamiento, con la finalidad que al perder la conectividad inalámbrica, y no sea posible solicitar esta data a algún servidor, la información de respaldo almacenada dentro del Raspberry Pi 3, entre en acción y mantenga funcionando el proyecto.

REFERENCIAS BIBLIOGRÁFICAS

Ahmad, A. Shafi, S. y Kumaravel, A (2013). VehNode: Wireless Sensor Network Platform for automobile pollution control. Recuperado de: https://ieeexplore.ieee.org/document/6558235

Anónimo, (2017). Historia y origen de las señales de tráfico. URL: http://www.sitographics.com/conceptos/temas/historia/Traffics_sign_history.html.

Amazon. Recuperado el 22 de abril de 2020, de: https://www.amazon.com/-/es/Interruptor-encendido-universal-posiciones-corriente/dp/B07FBTZXVS

Casanova, M. (2014). Diseño, construcción e instalación de un alcoholímetro electrónico con dispositivo de bloqueo de un vehículo. URL: http://dspace.espoch.edu.ec/bitstream/123456789/3732/1/65T00138.pdf

Chazallet S. (2016). *Python 3: los fundamentos del lenguaje*. Ediciones ENI.
Consejo Nacional de Seguridad Vial. (2017) Cuadro estadístico de accidentes de tránsito. URL: https://www.mtc.gob.pe/cnsv/estadistica.html

Chiang, H. Chen, Y. y Wu, B. (2014). "Embedded Driver-Assistance System Using Multiple Sensors for Safe Overtaking Maneuver". Recuperado de: https://ieeexplore.ieee.org/document/6352826

Conicyt (2015). Recuperado el 25 de febrero de 2020, de https://scielo.conicyt.cl/scielo.php?script=sci_arttext&pid=S0718-33052017000100070

Consejo Nacional de Seguridad Vial. (2017) Cuadro estadístico de accidentes de tránsito.Recuperado el 10 de diciembre de 2019, de https://www.mtc.gob.pe/cnsv/estadistica.html

Corona, L. Abarca, G y Mares, J. (2014). Otros sensores. *Sensores y Actuadores* (pp 263 – 265) México D.F. Editorial Patria.

Electrocrea (2020). Recuperado el 15 de marzo de 2020 de https://electrocrea.com/products/arduino-uno

Geekbuying. Recuperado el 15 de marzo de 2020, de http://overguide.com/archives/2140/get-a-good-nights-sleep-on-a-bean-bag-chair-bed

Guañuna, C. (2014). Diseño e implementación de sistema de encendido para automóvil mediante bloqueo por alcocheck con dispositivos de peso y posición del conductor. URL: http://repositorio.ute.edu.ec/bitstream/123456789/4813/1/56875_1.pdf

Hunt D. Puglia A. y Puglia M. (2007). *RFID: A guide to Radio Frequency Identification.* John Wiley & Sons.

Kemik (2020). Recuperado el 22 de abril de 2020, de: https://www.kemik.gt/comprar/relay-audiopipe-40a-12v/

Kumar, S. y Jasuja, A. (2017) "Air queality monitoring system based on lo T using Raspberry pi". Recuperado de: https://ieeexplore.ieee.org/document/8230005

LuisLlamas (2018). Recuperado el 22 de abril de 2020, de https://www.ebay.es/itm/1-30L-min-1-2-Hall-Effect-Flowmeter-Water-Flow-Sensor-Control-For-Arduino-/262134268800

Lpderecho (2018). Recuperado el 19 de diciembre de 2019, de https://lpderecho.pe/ebriedad-drogadiccion-agravante-feminicidio-vs-alteracion-conciencia-eximente-responsabilidad/

McGrath, M. (2016*). Raspberry Pi 3 in easy step*. In Easy Steps.

Moore, H. (2007). *Matlab para Ingenieros*, Utah, USA: Pearson Educación

Mueller, S. (2015). *Upgrading and Repairing PCs.* USA. Que publishing.

Natsys (2014). *Todo sobre MySQL: Libro ideal para ingresar en el mundo de la base de datos MySQL*. Natsys.

Naylampmechatronics(2018) Recupera el 10 de abril de 2020 de https://naylampmechatronics.com/sensores-gas/72-sensor-de-alcohol-mq3.html

OnSemiconductor. (2012). Recuperado el 22 de abril de 2020, de: https://pdf1.alldatasheet.com/datasheet-pdf/view/11662/ONSEMI/LM317.html

Organización Panamericana de la salud, (2018). Todas las muertes por accidentes de tránsito son prevenibles. URL: https://www.paho.org/per/index.php?option=com_content&view=article&id=3957:seguridad-vial&Itemid=900

Orozco, J. (2014). *Manual de computadoras y módulos automotrices.* México. Electrónica y Servicio.

Pallas, R. (2003). Sensores y acondicionadores de señal. España, Marcombo Boixareu editores

Peña,C (2020). *Descubriendo Arduino*. Red Users.

Pedro Ponce C (2010). *Inteligencia artificial con aplicaciones a la Ingeniería*. Editorial Alfaomega.

Raspberrypi.org (2015).Recuperado el 5 de marzo de 2020, de https://www.amazon.com/-/es/Element14-Raspberry-Pi-Placa-base/dp/B07BDR5PDW

Republica del Peru (2018) Ley Nº 27753. Modificaciones de Articulos 111, 124 y 274 del Codigo Penal. Recuperado el 19 de diciembre de 2019, de https://static.legis.pe/wp-content/uploads/2019/03/Ley27753%20Legis.pe_.pdf?fbclid=IwAR3rA5Pv5Rb735w1WT9hBDo_yc9I-G_2M6BjKG4b8rbThlQLQzDp372xfts

RobotShop (2020), Recuperado el 25 de febrero de 2020, de https://www.robotshop.com/us/es/pantalla-tactil-tft-lcd-320x480-35-para-raspberry-pi.html

Sánchez, M. (2014). Mantenimiento del sistema de arranque del motor del vehiculo. Recuperado de: https://eltrasteroloco.files.wordpress.com/2017/04/mantenimiento-del-sistema-de-arranque-del-motor-del-vehc3adculo-tm.pdf.

Tecexcirc (2016).Recuperado el 24 de marzo de 2020, de https://tecexcirc.wordpress.com/2016/08/13/alcohol-sensor-mq-3-interfacing-with-arduino/

ANEXOS

Anexo 1: Código en Python del Algoritmo difuso.

```
def minimo(a, b):
    a = float(a)
    b = float(b)
    if a < b:
        return a
    else:
        return b

def maximo(a, b):
    a = float(a)
    b = float(b)
    if a >= b:
        return a
    else:
        return b

def buscar(a):
    vec = a
    l = len(vec)
    temp = []
    for i in range(0, l):
        if vec[i] != 0:
            temp.append(i)
    return temp

def comparar(vec):
    l2 = len(vec)
    m = int(l2 / 41)
    s = vec[0:41]
    if m == 1:
        return s
    else:
        for v in range(1, m):
            for j in range(0, 41):
                s[j] = maximo(float(s[j]), float(vec[j+v*41]))
        return s

def conjunto_a(a):
    a = float(a)
    if 0 <= a <= 0.1:
        ua11 = 1
        ua12 = 0
        ua13 = 0
        ua14 = 0
        ua15 = 0
    elif 0.1 < a <= 0.2:
        ua11 = -a/0.1 + 2
```

```
        ua12 = a/0.1 - 1
        ua13 = 0
        ua14 = 0
        ua15 = 0
    elif 0.2 < a <= 0.3:
        ua11 = 0
        ua12 = -a/0.1 + 3
        ua13 = a/0.1 - 2
        ua14 = 0
        ua15 = 0
    elif 0.3 < a <= 0.4:
        ua11 = 0
        ua12 = 0
        ua13 = -a/0.1 + 4
        ua14 = a/0.1 - 3
        ua15 = 0
    elif 0.4 < a <= 0.5:
        ua11 = 0
        ua12 = 0
        ua13 = 0
        ua14 = -a/0.1 + 5
        ua15 = a/0.1 - 4
    elif 0.5 < a <= 0.6:
        ua11 = 0
        ua12 = 0
        ua13 = 0
        ua14 = 0
        ua15 = 1
    else:
        ua11 = 0
        ua12 = 0
        ua13 = 0
        ua14 = 0
        ua15 = 0
    return[ua11, ua12, ua13, ua14, ua15]

def conjunto_p(p):
    p = float(p)

    if 45 <= p <= 55:
        up11 = 1
        up12 = 0
        up13 = 0
        up14 = 0
        up15 = 0
    elif 55 < p <= 65:
        up11 = -p / 10 + 6.5
        up12 = p / 10 - 5.5
        up13 = 0
```

```
        up14 = 0
        up15 = 0
    elif 65 < p <= 75:
        up11 = 0
        up12 = -p / 10 + 7.5
        up13 = p / 10 - 6.5
        up14 = 0
        up15 = 0
    elif 75 < p <= 85:
        up11 = 0
        up12 = 0
        up13 = -p / 10 + 8.5
        up14 = p / 10 - 7.5
        up15 = 0
    elif 85 < p <= 95:
        up11 = 0
        up12 = 0
        up13 = 0
        up14 = -p / 10 + 9.5
        up15 = p / 10 - 8.5
    elif 95 < p <= 105:
        up11 = 0
        up12 = 0
        up13 = 0
        up14 = 0
        up15 = 1
    else:
        up11 = 0
        up12 = 0
        up13 = 0
        up14 = 0
        up15 = 0
    return [up11, up12, up13, up14, up15]

def reglas(va, vp):
    a = va
    p = vp
    ua = Conjunto_a(a)
    up = Conjunto_p(p)
    u = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
    if 0 <= a <= 0.2:
        if 45 <= p <= 65:
            u[0] = minimo(ua[0], up[0])
        if 55 <= p <= 75:
            u[1] = minimo(ua[0], up[1])
        if 65 <= p <= 85:
            u[2] = minimo(ua[0], up[2])
        if 75 <= p <= 95:
            u[3] = minimo(ua[0], up[3])
```

```
    if 85 <= p <= 105:
        u[4] = minimo(ua[0], up[4])
if 0.1 <= a <= 0.3:
    if 45 <= p <= 65:
        u[5] = minimo(ua[1], up[0])
    if 55 <= p <= 75:
        u[6] = minimo(ua[1], up[1])
    if 65 <= p <= 85:
        u[7] = minimo(ua[1], up[2])
    if 75 <= p <= 95:
        u[8] = minimo(ua[1], up[3])
    if 85 <= p <= 105:
        u[9] = minimo(ua[1], up[4])
if 0.2 <= a <= 0.4:
    if 45 <= p <= 65:
        u[10] = minimo(ua[2], up[0])
    if 55 <= p <= 75:
        u[11] = minimo(ua[2], up[1])
    if 65 <= p <= 85:
        u[12] = minimo(ua[2], up[2])
    if 75 <= p <= 95:
        u[13] = minimo(ua[2], up[3])
    if 85 <= p <= 105:
        u[14] = minimo(ua[2], up[4])
if 0.3 <= a <= 0.5:
    if 45 <= p <= 65:
        u[15] = minimo(ua[3], up[0])
    if 55 <= p <= 75:
        u[16] = minimo(ua[3], up[1])
    if 65 <= p <= 85:
        u[17] = minimo(ua[3], up[2])
    if 75 <= p <= 95:
        u[18] = minimo(ua[3], up[3])
    if 85 <= p <= 105:
        u[19] = minimo(ua[3], up[4])
if 0.4 <= a <= 0.6:
    if 45 <= p <= 65:
        u[20] = minimo(ua[4], up[0])
    if 55 <= p <= 75:
        u[21] = minimo(ua[4], up[1])
    if 65 <= p <= 85:
        u[22] = minimo(ua[4], up[2])
    if 75 <= p <= 95:
        u[23] = minimo(ua[4], up[3])
    if 85 <= p <= 105:
        u[24] = minimo(ua[4], up[4])
return u
```

```
def Encender(v_a, v_p)
    s = []
    v = Reglas(v_a, v_p)
    vec = buscar(v)
    l = len(vec)
    reglas = [1, 2, 3, 4, 7, 8, 9, 13, 14, 19]
    for i in range(0, l):
        indice = vec[i]

        for reg in range(0, 10):
            if vec[i] == reglas[reg]:
                est = True
            else:
                est = False

            if est:
                x = 3 - 2 * v[indice]
                for j in range(10, 51):
                    m = j / 10
                    if 1 <= m <= x:
                        sobri.append("{0:.3f}".format(v[indice]))
                    elif x < m <= 3:
                        sobri.append("{0:.3f}".format(-m/3 + 1))
                    elif 3 < m <= 5:
                        sobri.append("{0:.3f}".format(0))

    if sobri == []:
        for i in range(0, 41):
            s.append(0)
        return s

    s = comparar(sobri)
    return s

def Apagar(v_a, v_p):
    v = Reglas(v_a, v_p)
    vec = buscar(v)
    l = len(vec)
    reglas = [0, 5, 6, 10, 11, 12, 15, 16, 17, 18, 20, 21, 22, 23, 24]

    for i in range(0, l):
        indice = vec[i]

        for reg in range(0, 15):
            if vec[i] == reglas[reg]:
                est = True
            else:
                est = False
```

```
            if est:
                x = 2 + 3 * v[indice]
                for j in range(10, 51):
                    m = j / 10
                    if 1 <= m <= 2:
                        ebri.append(0)
                    elif 2 < m <= x:
                        ebri.append("{0:.3f}".format(v[indice]))
                    elif x < m <= 5:
                        ebri.append("{0:.3f}".format((m / 3) - 2/3))

        if ebri == []:
            for i in range(0, 41):
                e.append(0)
            return e

        e = comparar(ebri)

        return e

    def defuzificar(a, p):
        va = a
        vp = p

        s = Encender(va, vp)
        e = Apagar(va, vp)
        l = len(s)

        de = []
        den = 0

        for i in range(0, l):
            de.append(maximo(e[i], s[i]))
            den = den + de[i]

        x = []
        for j in range(10, 51):
            x.append(j / 10)

        t = de[0] * x[0]
        for j in range(1, len(x)):
            t = t + de[j] * x[j]

        if den > 0:
            r = t / den
        else:
            r = 0

        return r
```

Anexo 2: Código en arduino para la lectura del RFID y alcoholímetro

```
#include <SPI.h>
#include <MFRC522.h>

#define RST_PIN  9    //Pin 9 para el reset del RC522
#define SS_PIN  10   //Pin 10 para el SS (SDA) del RC522
MFRC522 mfrc522(SS_PIN, RST_PIN); ///Creamos el objeto para el RC522

#define PIN_RELE  6// pin al rele

volatile long numPulsos;

void contarPulsos(){
  numPulsos++;
}

void setup() {
  Serial.begin(9600); //Iniciamos La comunicacion serial
  SPI.begin();        //Iniciamos el Bus SPI
  mfrc522.PCD_Init(); // Iniciamos el MFRC522
  pinMode(2,INPUT);//Sensor de caudal
  pinMode(6,OUTPUT);//rele
  attachInterrupt(0,contarPulsos,RISING);//
  interrupts();
}

int flag = 0;
char v;
byte ActualUID[4]; //almacenará el código del Tag leído

void loop(){

  enviarCodigo();
  while(flag == 1){
    v = Serial.read();
    if( v == 'V'){
      double a = leerSensor();
      if(a >= 0){
        Serial.print(a);
      }
    }else if( v == 'F'){
      flag = 0;
    }else if(v == 'R'){
      flag = 0;
      digitalWrite(PIN_RELE,HIGH);
      delay(1000);
      digitalWrite(PIN_RELE,LOW);
    }
```

```
  }
}

// Lee el codigo de la tarjeta;
void leerCodigo() {
    String txt;
    // Revisamos si hay nuevas tarjetas  presentes
    if ( mfrc522.PICC_IsNewCardPresent()){
      //Sfeleccionamos una tarjeta
          if(mfrc522.PICC_ReadCardSerial()) {
               // Enviamos serialemente su UID
              for (byte i = 0; i < mfrc522.uid.size; i++){
                  ActualUID[i]= mfrc522.uid.uidByte[i];
              }
              //Terminacion de lectura de la tarjeta
               mfrc522.PICC_HaltA();
          }
    }
}

// Lee el sensor MQ 3 + flujometro:
double leerSensor(){
  //numPulsos=0;
  double temp = 0;

  for(int i=0;i<50;i++){

    double lectura = analogRead(A0);
    double volt = lectura*5/1023;
    // Ecuacion para calibrar el mq3
    double expo = (volt - 0.1535) / 0.3958;
    double dato = pow(2.71828, expo);

    if(temp < dato){
      temp = dato;
    }
    delay(100);
  }
  return temp;
  /*if(numPulsos > 250){
    return temp;
  }else{
    return -2;
  }*/
}



// Metodos para el RFID
```

```
boolean compararCodigo(byte array1[])
{
  byte  array2[]={0,0,0,0};
  if(array1[0] != array2[0])return(false);
  if(array1[1] != array2[1])return(false);
  if(array1[2] != array2[2])return(false);
  if(array1[3] != array2[3])return(false);
  return(true);
}

void enviarCodigo(){
    limpiarVector();
    leerCodigo();
    if(!compararCodigo(ActualUID)){
      for(byte i = 0; i < 4; i++){
        if(ActualUID[i] <= 15){
          Serial.print(0);
          Serial.print(ActualUID[i],HEX);
        }else{
          Serial.print(ActualUID[i],HEX);
        }
      }
      flag = 1;
    }
}
```

Anexo 3: Modulo en Python para realizar la consulta con la base de datos

```
import pymysql as sql

#VARIABLE QUE ALMACENA EL CODIGO DE LA PERSONA INGRESANTE
res = ""

#VARIABLES QUE ALMACENAN LOS PARAMETROS DE CONEXION CON
LA BD
host = "" #HOST O IP DEL SERVER
user = "" #USUARIO
pwd = "" # CONTRASEÃ‘A
db = "" # NOMBRE DE LA BASE DE DATOS


#CONSULTA A UN ARCHIVO TXT POR UN CODIGO
def consultarA(cod):

    doc = open('path','r')
    lista = []
    f = True
    while f:
        linea = doc.readline().strip()

        if linea != "":
            lista.append(linea)
        else:
            f = False
    doc.close()

    for i in range (len(lista)):

        if lista[i] == cod:
            return True

    return False

#CONSULTA A LA BASE DE DATOS POR UN CODIGO
def consultarB(cod):

    con = sql.connector(host,user,pwd,db)

    cursor = con.cursor()
    query= "SELECT ID FROM PERSONAS WHERE TAG =
\"{0}\";".format(cod)
    cursor.execute(query)

    res = cursor.fetchall()
```

```
        if res != None:
                con.close()
                return True

        con.close()
        return False

#CONSULTA A LA BASE DE DATOS EL NUMERO DE EVENTO EN LA
TABLA DE INGRESO
def consultaE():

        con = sql.connector(host,user,pwd,db)
        cursor = con.cursor()

        query= "SELECT COUNT(EVENTOS) FROM INGRESOS;"
        cursor.execute(query)

        h = cursor.fetchall()
        con.close()

        return h

#VERIFICA SI EL CODIGO SE ENCONTRO O NO
def verificar(logA,logB):

        if logA or logB:
                return "V"
        else:
                return "F"
#REGISTRA UN INGRESO EN LA BASE DE DATOS
def Escribir(hora,valor_alcolimetro):

        con = sql.connector(host,user,pwd,db)
        cursor = con.cursor()
        evento = int(consultaE()) + 9999
        query= "INSERT INTO TABLE ingresos
values(\"{0}\",\"{1}\",\"{2}\",\"{3}\");".format(evento,res,hora,valor_alcoholimetro)
;
        evento += 1;
        try:
                cursor.execute(sql)
                con.commit()
        except:
                con.rollback()

        con.close()
```

Anexo 4: Modulo principal del programa

```
from datetime import date
from datetime import datetime
import Consultas as cs
import difuso as d
import turtle
import serial

arduino = serial.Serial("/dev/ttyACM1", 9600, timeout=1)

turtle.setup(450,150,0,0)
title("PANEL DE ID")
screensize(450,150)
hideturtle()

def escribir(texto):
	goto(0,0)
	write(texto,False,"center",
		("arial",24,"bold"))

def verificarAlcoholemia(peso):
	turtle.clear()
	escribir(Sople ahora)
	#Se lee el valor del alocholimetro proveniente del arduino
	alc = arduino.read(4)
	time.sleep(0.5)
	hora = datetime.now()
	cs.escribir(hora,alc)
	rpta = d.defuzificar(peso,alc)
	if rpta > 1.0:
		turtle.clear()
		escribir("Prueba exitosa,arranque ahora)
		time.sleep(2)
	else:
		turtle.clear()
		escribir("Prueba fallida,arranque negado)
		time.sleep(2)


while True:
	time.sleep(0.1)
	txt = ''
	turtle.clear()
	escribir("Identifiquese por favor")
	while arduino.inWaiting() > 0:
		txt += arduino.read(8)
		print(txt)
```

```
if txt != None:
        escribir("Procesando, espere")
        if ver[0]:
                #Envia la validacion de id al arduino
                arduino.writer('V')
                escribir("USUARIO IDENTIFICADO")
                time.sleep(2)
                #Consulta peso en la base de datos
                peso = c.consultarPeso(txt)
                #Ejecuta lectura de alcoholimetro
                verificarAlcoholemia(peso)
        else:
                arduino.write('F')
                turtle.clear()
                escribir("USUARIO NO IDENTIFICADO")
                time.sleep(2)
                turtle.clear()
                txt = ''
```