Diseño de un sistema de seguridad para un prototipo β de un robot explorador en ambientes hostiles por intervención bruta de control y reubicación espacial en zona segura.

Alejandro Lupiáñez Bellido[1,a] , Israel Sánchez Domínguez[2,b]

[1] UVM – Campus Mérida, Mérida, Yucatán, México
[2] UNAM – IIMAS - Unidad- Mérida, Mérida, Yucatán, México
[a]alejandro.lupianez@my.uvm.edu.mx
[b]israel.sanchez@iimas.unam.mx.

Fecha de recepción: 8 de mayo de 2018 — Fecha de aceptación: 22 de enero de 2019

Resumen

El uso de la tecnología para la inspección de zonas de riesgo para el ser humano, es un campo que ha ido creciendo en los últimos años, del mimo modo que todas las áreas de implementación tecnológica, sin embargo, aún existen varias limitantes que no han sido del todo solventadas, como, por ejemplo, la autonomía y seguridad de los propios dispositivos de inspección. En el presente trabajo se pretende innovar presentando un diseño y verificación de un algoritmo de seguridad aplicable a robots exploradores para inspección en zonas donde haya poco espacio de maniobra, visibilidad limitada y terreno irregular, (como puede ser el caso un derrumbe estructural), mediante la construcción y caracterización de un prototipo β o dispositivo probador, cuya principal misión se centra en corroborar el algoritmo de seguridad, que deberá ser capaz de asegurar su integridad en desplomes inesperados y cambios en la inclinación del terreno. El prototipo se diseñó basado en una plataforma Arduino, y requiere de varios sensores para su funcionamiento, algunos de los cuales, pueden ser intercambiables según los requerimientos específicos del ambiente de operación del robot explorador. El prototipo demostró su operatividad, pero se detectaron fallas mecánicas y de computo que requieren depuración. Sin embargo, los resultados respecto de algoritmo de seguridad básico son alentadores para poder evolucionar el prototipo actual para la mejora sustancial del algoritmo y establecer una línea clara de evolución del proyecto y de esta forma, asentar unas bases definidas para el desarrollo de sistemas de seguridad para robots exploradores basados en la lógica propuesta.

Palabras clave: Arduino, algoritmo, seguridad, explorador, prototipo β.

Design of a security system for an β Prototype explorer robot in hostile environments, by direct control intervention for spatial relocation in a safe area.

Abstract

The use of technology for the inspection of risky areas for human beings, is a field that has been growing in recent years, as much as all areas of technological implementation, however there are still several

---

[a]alejandro.lupianez@my.uvm.edu.mx
[b]israel.sanchez@iimas.unam.mx.
Nota: Este artículo de Divulgación es parte de Ingeniería–Revista Académica de la Facultad de Ingeniería, Universidad Autónoma de Yucatán, Vol. 23, No. 1, 2019, ISSN: 2448-8364

limitations that cannot be solved, such as, for example, the autonomy and safety of the inspection devices. In the present work we try to innovate by presenting the design and verification of a security algorithm for exploratory robots that work in areas where there is little room for maneuver, limited visibility and irregular terrain, (for example when a structural collapse occurs), by means of construction and characterization of a β prototype (or tester device) whose main mission is to validate the security algorithm which should ensure its integrity in unexpected collapses and changes in the inclination of the terrain. The prototype was designed based on an Arduino platform, and requires several sensors for its operation, some of which can be interchangeable according to the specific requirements of the operating environment of the explorer robot application. The prototype demonstrated its operability, but mechanical and computing flaws were detected that require debugging. However, the results regarding the security algorithm are encouraging to be able to evolve the actual prototype in order to get a substantial improvement of the security algorithm and to set a clear line of evolution of this project. In this way it is intended to establish a clear base for the development of security systems based on the proposed logic.

Keywords: Arduino, algorithm, security, exploratory, β prototype.

Introducción

La naturaleza nos ha enseñado que es impredecible además de altamente destructiva, y sin embargo realmente lo que causa mayor daño a la vida humana son nuestras propias obras; edificios, carreteras o puentes al ser destruidos por fenómenos naturales. Por esta razón, es necesario contar con dispositivos que sean capaces de realizar tareas para las que los humanos no estamos preparados y puedan facilitar nuestro trabajo. Un ejemplo claro se vive en derrumbes estructurales, donde es necesario realizar inspecciones en espacios reducidos con riesgo de desplome del terreno, alta contaminación, etc… Para este tipo de actividades, se diseñó un sistema de seguridad aplicable a robots de inspección en zonas peligrosas, tanto para un humano, como, inclusive, para el propio robot explorador, por esto, el software diseñado, le permite protegerse en planos inclinados repentinos como escombros por derrumbe, fallas geológicas o terreno arenoso, y colocarse en una posición segura sin requerir de una rápida y precisa corrección del operador. Tras la acción de protección, las funciones del robot volverán al control del operador, siempre que se encuentre en una posición segura.

Para establecer un marco de referencia, dado un panorama de investigación en sistemas de seguridad para vehículos con proyectos, quizá, demasiado complejos o avanzados, y orientados a vehículos autónomos como el presentado en **[8]** o soluciones avanzadas de vehículos inteligentes tripulados **[9]**, se pretende establecer una línea de investigación para sistemas de seguridad simples para robots exploradores controlados a distancia (en **[10]** se presenta detallada una propuesta para esto), basada en comparaciones de las mediciones del entorno tomadas por el robot e interrupciones de las funciones normales del mismo. Para este desarrollo, es necesario contar con dispositivos que den al robot la habilidad de determinar situaciones de riesgo mediante las mencionadas mediciones. Con esto en consideración, y con el objetivo de poder probar el sistema en acción, se construyó un prototipo β simple, emulando un robot explorador final (quien será el usuario último del programa), donde se implantaron varios sensores controlados mediante una plataforma Arduino UNO **[1]**, elegida debido a su facilidad de comunicación, programación y control de los sensores.

En el presente trabajo, únicamente se evaluará el algoritmo en una versión β del dispositivo final,

de donde se obtendrá retroalimentación, tanto en software como en hardware para poder trazar una proyección estimada de las evoluciones y etapas necesarias hasta realizar una versión final de robot explorador, que cuente con la implementación del sistema de seguridad basado en el algoritmo propuesto. A continuación, se describe la metodología usada para llevar a cabo el diseño, construcción y caracterización de este prototipo β.

1. Metodología:

Para comenzar el diseño del sistema de seguridad, y poder comprobar su funcionamiento, se construyó un dispositivo robótico (prototipo β), con interfaz, componentes y programación simples, donde se cargará únicamente el control de los sensores y el algoritmo de seguridad simplificado, por el cual, el dispositivo, será capaz de iniciar/reanudar el sistema mediante acción externa (para fines de este proyecto se define esta condición como una interacción con un sensor táctil en el prototipo implementado), detectar cambios de inclinación en el plano horizontal, objetos o paredes cercanas en las direcciones de avance y retroceso, y considerar un estado de seguridad añadido actuando de apagado forzado (o stand-by) del propio sistema de seguridad (esta acción en el prototipo β, se implementará mediante un sensor de color, captando una tarjeta roja, pudiendo ser, desempeñada por otros tipo de sensores en el modelo del robot explorador final, de acuerdo a la aplicación deseada y parámetros del terreno como temperatura o humedad). Más adelante se detalla el funcionamiento del algoritmo implementado.

Es importante destacar que, en lo que concierne al prototipo β desarrollado, dado que únicamente se le cargará los sistemas de operación de los sensores y el algoritmo de seguridad, todo momento en el que el prototipo se encuentre inmóvil, será considerado como un control normal del sistema del robot explorador final, el cual será realizado por control remoto por un operario. También es conveniente añadir que se supone que, en un modelo final del explorador, el sistema de seguridad permanece ejecutándose en segundo plano hasta que se dan las condiciones necesarias para su intervención en el sistema principal de control, esto posiblemente, gracias a rutinas de interrupciones.

Debido a estas acciones, el robot usuario del algoritmo, deberá contar, al menos, con los siguientes elementos, para llevar a cabo tanto el monitoreo o medición de los parámetros de funcionamiento, como para ejecutar las acciones oportunas en base a esas mediciones:

- Sensor giroscopio/acelerómetro.
- Sensor de distancia en direcciones de avance y retroceso
- Sensor o interruptor de encendido y reanudación
- Sensor de paro de emergencia, activando el sistema de seguridad
- Unidad de potencia.
- Electrónica de potencia asociada.
- Fuentes de alimentación para el control y la tracción por separado.

Estos elementos son necesarios para el correcto seguimiento y funcionamiento del programa, de acuerdo al diagrama de flujo presentado a continuación; el cual fue diseñado para seguir de forma secuencial las instrucciones para un buen funcionamiento, poniendo especial cuidado en medir y auto conservarse. El diagrama de flujo del sistema se presenta en la Fig. 1 y está diseñado según la lógica que rige el sistema de seguridad a implementar y en base a los sensores y parámetros específicos implementados en prototipo β, los cuales serán detallados más adelante:

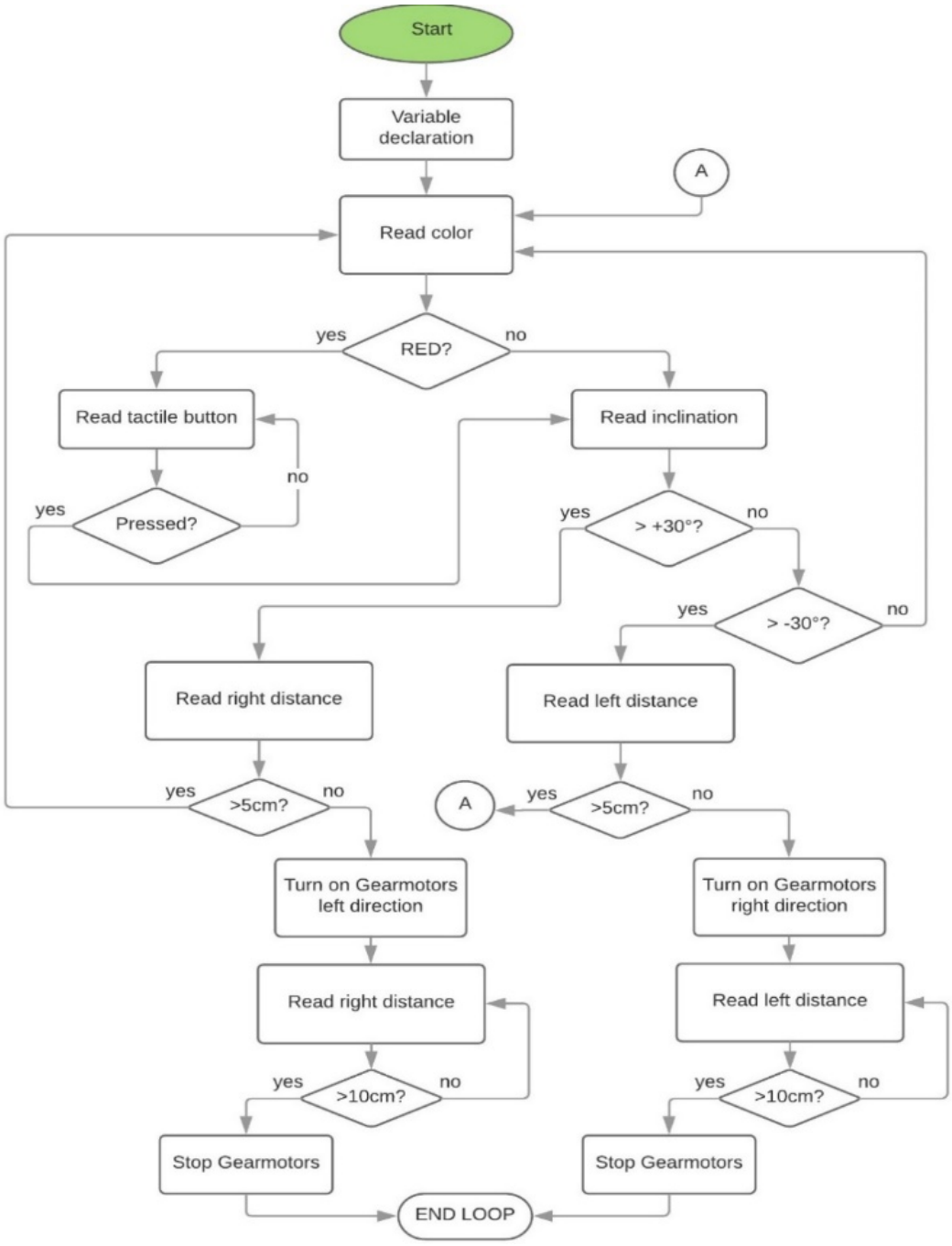

Fig. 1. Diagrama de flujo del funcionamiento básico del algoritmo de seguridad implementado.

La primera acción que toma el programa es evaluar si el estado de paro de emergencia se ha activado (para el caso del prototipo β, se evalúa si previamente se le ha mostrado una tarjeta de color rojo al sensor que evalúa el paro de emergencia). En caso de estar en estado de paro de emergencia (o stand-by del sistema de seguridad), el dispositivo quedará en estado permanente de lectura del sistema de arranque/reanudación, hasta que este sea activado (para este prototipo, seguirá en estado de espera hasta que se reactive el funcionamiento mediante el botón táctil). Esto

significaría un control manual del operador sin el respaldo del sistema de seguridad hasta que se accione su reanudación. Es decir, el paro de emergencia del sistema de seguridad funciona como un apagado temporal del mismo, mas no de las funciones normales de control del dispositivo explorador ejercido por el operador.

Si no se ha detectado el paro de emergencia o si ya se ha accionado la reanudación, el sistema comenzará a leer la inclinación del dispositivo. En base a esta lectura efectuada por el sensor (giroscopio), se identificarán tres estados mediante disyuntivas encadenadas relacionadas con el ángulo de la inclinación procesado por el sistema, así como con un ángulo de máxima inclinación permisible prefijado en la programación (para el caso de este prototipo β, esta máxima inclinación permisible será de 30 grados). Los tres estados serán:

a) Lectura de un ángulo hacia la derecha mayor al ángulo máximo permisible.
b) Lectura de un ángulo hacia la izquierda mayor al ángulo máximo permisible.
c) Lectura de un ángulo menor hacia la izquierda y hacia la derecha al ángulo máximo permisible.

Comenzando por este último caso, este supone simplemente que la inclinación (en cualquier dirección) no es suficiente para activar el sistema de seguridad del robot explorador, es por esto que, en este caso, el sistema de seguridad regresa al punto departida, iniciando el ciclo de nuevo, evaluando si se ha activado el sistema de paro de emergencia y dejando el control total del dispositivo al operador sin tomar ninguna acción sobre la tracción de las ruedas motrices. Como adelanto, no sobra añadir que una de las principales mejoras aplicables al algoritmo en futuros trabajos será eliminar la lectura sistemática del estado de emergencia, reemplazando esta secuencia por un protocolo de interrupciones con el consecuente ahorro de tiempo de computo que esto conllevaría en cada ciclo que no se encuentre activado. Esta implementación no fue posible en este prototipo debido a que la tarjeta Arduino no es compatible con este tipo de programación de interrupciones. De ser posible, los protocolos de interrupción acortarían considerablemente el tiempo de cómputo general del algoritmo completo, evitando la lectura de sensores en ciclos innecesarios. El estudio de tiempos de computo se detallará más adelante.

Para los dos primeros casos, significaría que se ha cumplido una de las condiciones para que el algoritmo de seguridad se ejecute por encima del control manual del operario. La segunda condición se relaciona con la cercanía de un obstáculo en la dirección en la que se está dando la inclinación. Es por esto que, en este momento, el sensor de distancia situado en la dirección inclinada realizará la lectura de la distancia hasta el obstáculo.

Esta lectura generará dos nuevos casos:

i. **La distancia medida al obstáculo es mayor a la "distancia crítica permitida" preestablecida en la programación**: En este caso, aun no existe problema alguno para la intervención inmediata del sistema de seguridad en el control normal del dispositivo explorador, es por esto que el sistema de seguridad regresara al punto de partida, iniciando el ciclo de nuevo y evaluando nuevamente si se ha activado el sistema de paro de emergencia. (Para el caso de este prototipo, esta distancia crítica permitida se fijó en 5 cm.)

ii. **La distancia medida al obstáculo es menor a la "distancia crítica permitida" preestablecida en la programación**: En este caso, se han cumplido las dos condiciones básicas necesarias para la intervención del sistema de seguridad, esto significa que el control normal del dispositivo explorador (dado por el operador) queda anulado temporalmente

hasta que el sistema de seguridad consiga reubicar el dispositivo en una zona segura. Con esto se consigue evitar una posible colisión con el obstáculo debido a la inclinación y a la cercanía excesiva entre este y el dispositivo. Para el caso del prototipo construido, éste enviará la señal de activación a los motores, accionándolos, hacia la dirección correspondiente (contraria a la inclinación sensada anteriormente), mediante los sistemas de potencia y control implementados.

Llegados a este punto el control normal del dispositivo está intervenido por el sistema de seguridad que, independientemente de lo que el operador haga y la velocidad de reacción a la que lo haga, logrará reubicar el prototipo en una zona segura. En el caso del prototipo β, se define esta "Zona Segura" como una "*distancia de seguridad mínima*" de 10cm para volver a restablecer el control normal al operario. Para lograr esto, una vez que se ha intervenido el sistema de control principal y accionado los motores en la dirección opuesta a la inclinación para evitar la colisión con el obstáculo, el sistema de seguridad entrará en un estado cíclico de lectura y comparación de la distancia al obstáculo con la "distancia de seguridad mínima" preestablecida para devolver el control manual al operador. En el momento en el que la distancia medida por el sensor en la dirección de inclinación, supere el límite estipulado por la "distancia de seguridad mínima", el sistema de seguridad devolverá el control normal al operador remoto del dispositivo. En el caso del prototipo construido, esto significa un alto en la señal de alimentación de los motores de tracción.

Es importante destacar que esta "distancia de seguridad mínima" prefijada para devolver el control al operador, debe ser mayor que la "distancia crítica permitida" a la que el sistema de seguridad toma acción, para poder darle al operador un margen en el que pueda corregir la situación y evitar que el sistema de seguridad se dispare de nuevo entrando en un bucle de correcciones.

Para la construcción del dispositivo de pruebas, una vez que se ha diseñado el algoritmo y se ha considerado los elementos electrónicos necesarios para su funcionamiento, se establecieron los modelos concretos de cada componente, así como con las fuentes de alimentación, sistema de potencia para tracción y la plataforma utilizada.

De manera general, los modelos elegidos para la implementación del prototipo β, fueron aquellos que, de una manera sencilla, permitían seguir la lógica del programa presente en el diagrama de flujo, permitiendo lograr un funcionamiento idóneo y simple del prototipo, estos elementos, elegidos por su simplicidad de uso y precio económico, son los siguientes:

- Plataforma: **Arduino UNO [1]**. Elegido en base a su facilidad de programación y compatibilidad con sensores básicos y de fácil disponibilidad. Para facilitar las conexiones del resto de componentes se utilizará el apoyo de una **tarjeta Shield para Arduino** acoplada justo sobre el propio Arduino.
- Sensor giroscopio/acelerómetro: **MPU-6050 [2].**
- Sensor de distancia para ambas direcciones: **Sensores ultrasónicos Hc-SR04 [3].**
- Sensor de encendido y reanudación: **TouchPad capacitivo TTP223B [4].**
- Sensor paro de emergencia del sistema de seguridad: **Sensor de color TCS230 [5].**
- Unidad de potencia: **2 motoreductores simples con llanta acoplada.**
- Fuentes de alimentación para control y tracción por separado: tanto la placa base del sistema (Arduino UNO) como la potencia de los motoreductores se

alimentan **con 2 baterías aisladas de 9V**. Una aplicada de ellas encargada para alimentar a todo el sistema de control por Arduino y la otra dedicada exclusivamente a la tracción de los motoreductores, vinculados por la electrónica de potencia.

- Electrónica de potencia: se controlaron los motores mediante un **puente doble H encapsulado L293D** controlado con las señales **PWM del Arduino.** Otros proyectos para construcciones similares utilizan este mismo driver motor por su sencillez y practicidad como en el caso **[7]**.

Como se había mencionado, en la Fig. 1 se muestra el diagrama de flujo del sistema en relación a la implementación de estos modelos de los componentes en concreto.

Para comprender estos componentes, su funcionamiento y sus valores críticos, primeramente, se realizó una batería de pruebas en cada uno de ellos, donde se obtuvieron los siguientes valores relevantes:

- El valor RAW que el Giroscopio MPU-6050 presentó para una correspondencia en los 30 grados (valor editable según operación y preestablecido en la lógica de funcionamiento) fue de 8400 en el eje longitudinal.
- El sensor Ultrasónico Hc-SR04 mide el tiempo transcurrido entre el momento en que envía un pulso sónico y el momento en que lo recibe. Aplicando las siguientes fórmulas se obtiene la distancia al obstáculo:

$$\boldsymbol{Tiempo\ (para\ 1cm) = \frac{1cm}{vel\ sonido} = \frac{1cm}{0.0343(\frac{cm}{\mu s})} = 29.15\ \mu s} \quad \textbf{(1)}$$

$$\boldsymbol{distancia(cm) = \frac{(tiempo\ medido/2)}{29.15}} \quad \textbf{(2)}$$

- En la conexión de alimentación mediante el Puente H L293D, las pruebas se centraron en probar los valores para los PWM, la velocidad de giro que se provoca en los servos, definiendo los sentidos de giro y establecer el esquema de conexiones, como se observa en la Fig. 2. Para un valor máximo de corriente en los motores, corresponde el valor 255 del PWM desde Arduino.

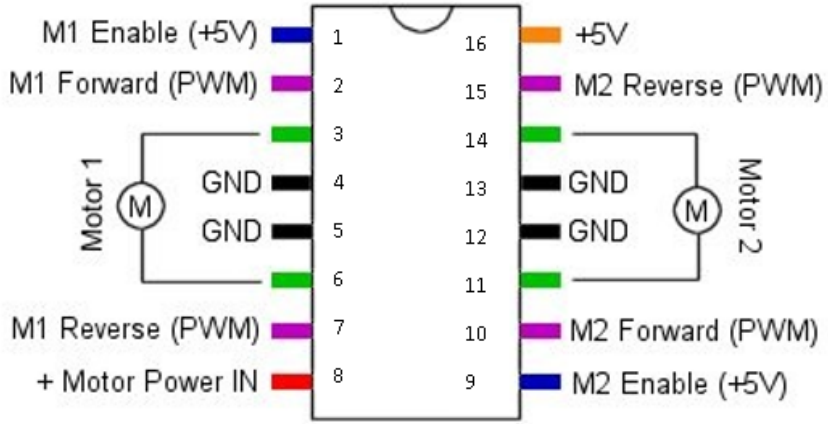

Fig. 2. Esquema de conexiones del Puente H L293D

- El **sensor capacitivo táctil TTP223B** por ser un sensor muy simple y de fácil implementación, todas las pruebas se basaron en el control del estado de una variable y en su integración a un ciclo en el que el programa quede permanentemente esperando la entrada de este sensor.

- En lo referente al **sensor de color TCS230** son las configuraciones preestablecidas para captar un porcentaje determinado de la frecuencia de los colores primarios en función a los LED´s integrados en el dispositivo (S1, S2, S3, S4). La necesidad de esto en el proyecto, es debido a que el proceso de inicialización o paro, es que el sensor lea una tarjeta de color rojo por lo que su configuración de LED's resulta (considerando un 20% de la frecuencia):

  - S0 = HIGH
  - S1 = LOW
  - S2 = LOW
  - S3 = LOW

  Por otro lado, para determinar la tarjeta roja se acotaron entre un rango de valores para el sensor de 35≥ Rojo <30.

Estos valores se obtuvieron mediante la prueba y monitoreo de la frecuencia recibida al mostrar al sensor el color de la tarjeta que debe detener el programa.

Para el diseño físico y construcción del prototipo β, se desarrolló un soporte simple donde se montaron todos los elementos que conforman el sistema. Este diseño debe considerar, principalmente, la instalación de los motoreductores, así como la ubicación de los sensores ultrasónicos de distancia.

El material empleado para dicho soporte fue material acrílico debido a su facilidad para moldearlo a la forma deseada, incluyendo ranuras y pasantes, así como la capacidad de ser trasparente, lo cual facilitaba todo el montaje electrónico. Para diseñar la forma del soporte se consideró el conjunto de los elementos que deben ir montados sobre él y el método de acople (ya fuera atornillado, encajado o pegado), además se realizó un modelo 3D en el programa SolidWorks para su mejor visualizado y fabricación.

Posterior a esto, se procedió a realizar el montaje y conexiones de todos los componentes y sensores de acuerdo a la lógica y nomenclatura de entradas y salidas propuesta en el código del programa de Arduino, como se observa en la Fig. 3

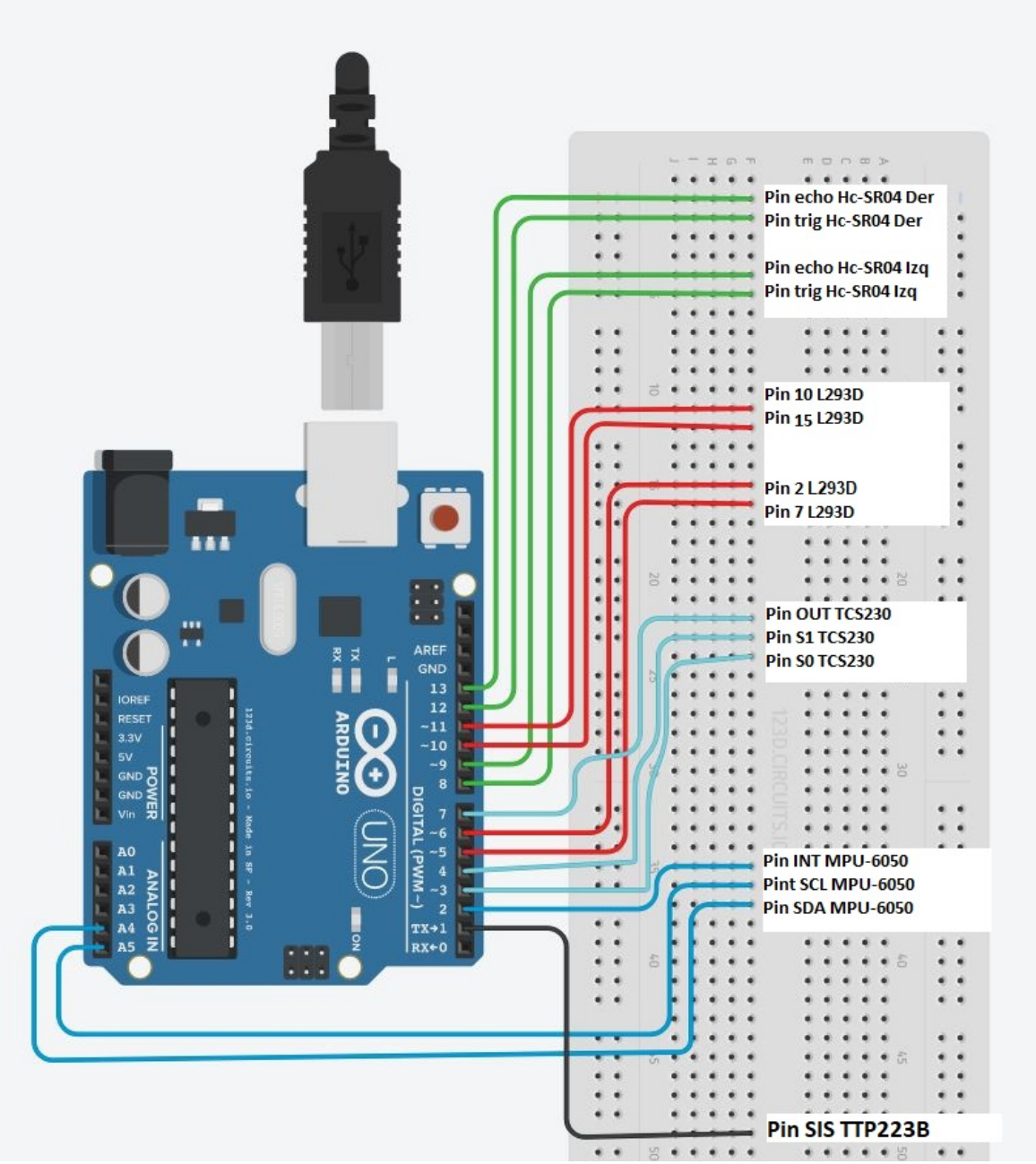

Fig. 3. Diagrama de conexiones para el prototipo β, del robot.

Realizado ya todo el montaje del prototipo β, se procedió a hacer pruebas "in vivo" con objetivo de probar la funcionalidad del algoritmo de seguridad, obtener una retroalimentación de ambos apartados (lógica del algoritmo y respuesta del hardware) así como poder descubrir fallas tanto en software como en hardware. Esto se realizó en un ambiente controlado, en donde la característica principal fue poder constatar la funcionalidad del algoritmo de seguridad en el prototipo en base a las variables principales del sistema: inclinación y distancia al obstáculo. Para

ello se diseñó un soporte pivotante sobre un eje, para someter al dispositivo a los valores de inclinación supuestos. Además, en dicha base, se implementaron paredes laterales las cuales imitarían la presencia de obstáculos con los que el dispositivo impactaría bajo las condiciones de inclinación. En el apartado de resultados, puede observarse, en la Fig.7, las condiciones a las que el dispositivo fue sometido en función de las variables sensadas; inclinación y distancia al obstáculo en la dirección de la inclinación.

En pruebas a futuro se pensará en ir incrementado el grado de dificultad de las pruebas, como, por ejemplo, suponer otro eje simultaneo de inclinación en el plano e ir analizando el comportamiento de las evoluciones del prototipo en cada una de las pruebas, para finalmente presentar un dispositivo funcional, reproducible, confiable y autónomo, tanto en funciones del software de seguridad, como en respuesta del hardware.

Por otro lado, acompañando a estas pruebas, también se realizó un estudio de tiempos de cómputo para la ejecución del algoritmo por el controlador, en este caso la plataforma Arduino. Esto fundamentado en establecer una base para estimar una proyección de posibles evoluciones del dispositivo hasta llegar al modelo final (esta proyección será detallada en la sección 3.1. Trabajo a futuro y proyección de proyecto).

Con ánimo de conocer la serie de tiempos que le lleva al Arduino procesar cada una de las acciones del proceso, se ejecutaron, en el código del programa, una serie de códigos para tomar estas mediciones de tiempo. Estas mediciones toman el tiempo trascurrido desde que inició el programa hasta el momento exacto (línea de código) en el que se toma la medición.

Los instantes exactos de estas mediciones se presentan en la Fig. 4 señalados en el diagrama de flujo por una flecha blanca. De aquí en adelante, cada medición recibe el nombre vinculado a la acción inmediata anterior al momento de la toma de tiempo, siendo, por ejemplo, la medición “Encendido de motores” el tiempo trascurrido desde el inicio del programa hasta que la acción de encender los motoreductores se haya completado computacionalmente.

Esta nomenclatura aparece más adelante en la sección resultados para referirse a cada medición de forma particular.

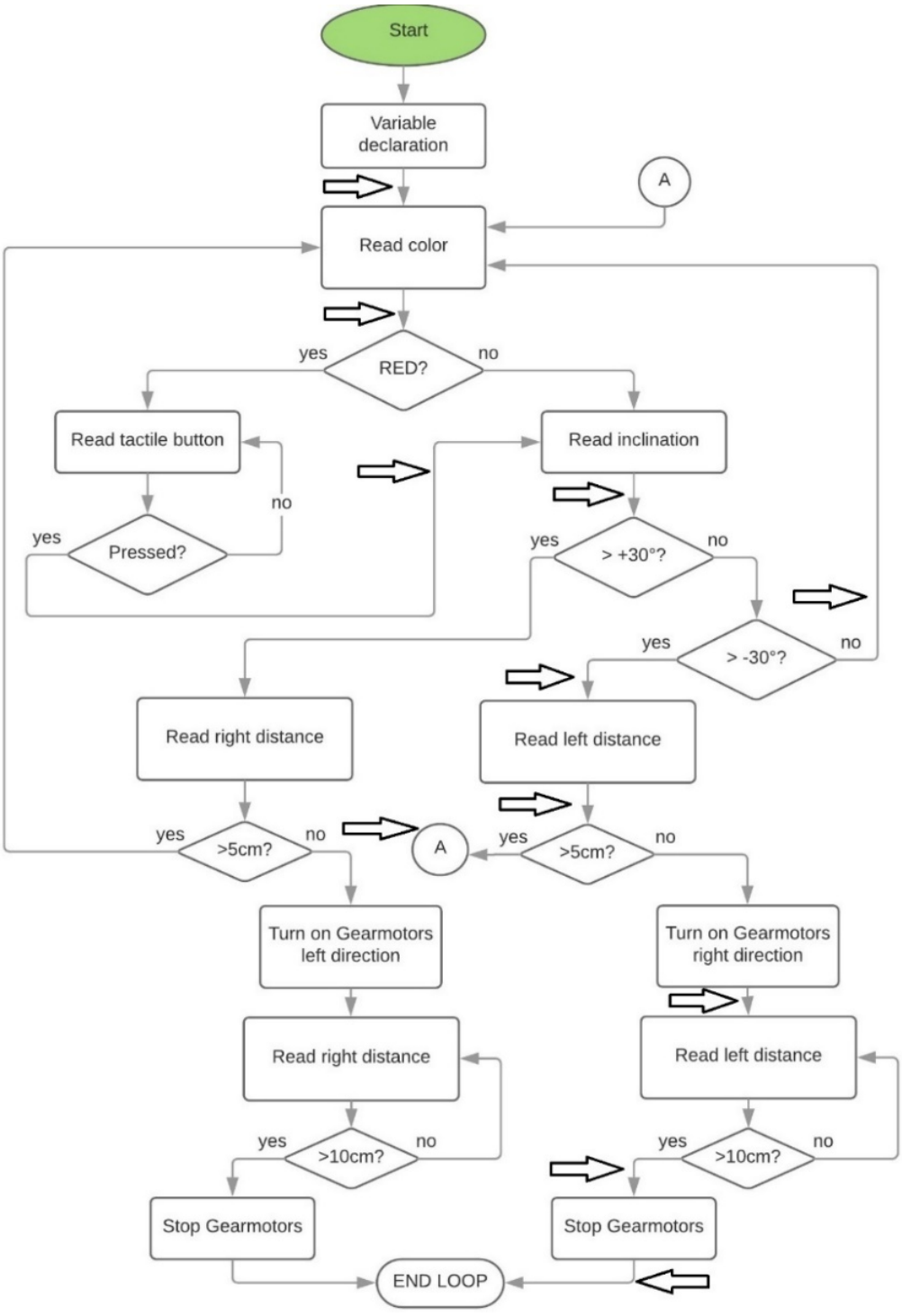

Fig. 4. Puntos de toma de tiempos en el diagrama de flujo.

Es importante aclarar, que en el estudio de tiempos se desarrollaron tres escenarios diferentes para probar los tres lazos característicos del sistema. Además, también se evaluó cada uno de ellos introduciendo un ciclo del paro de emergencia del sistema de seguridad (o stand-by).

La definición de estos lazos se expone en la siguiente tabla:

Lazo abierto más largo posible	NO se muestra tarjeta roja, inclinación NO mayor a 30ºC, inclinación MENOR a -30º, distancia medida NO mayor a 5 cm, corrección en un ciclo hasta 10 cm.
Lazo cerrado por inclinación	NO se muestra tarjeta roja, inclinación NO mayor a 30ºC, inclinación MAYOR a -30º.
Lazo cerrado por distancia	NO se muestra tarjeta roja, inclinación NO mayor a 30ºC, inclinación MENOR a -30º, distancia medida SI mayor a 5 cm.
Lazo abierto más largo posible con ciclo de paro de emergencia	SI se muestra tarjeta roja, un ciclo para la continuación, inclinación NO mayor a 30ºC, inclinación MENOR a -30º, distancia medida NO mayor a 5 cm, corrección en un ciclo hasta 10 cm.
Lazo cerrado por inclinación con ciclo de paro de emergencia	SI se muestra tarjeta roja, un ciclo para la continuación, inclinación NO mayor a 30ºC, inclinación MAYOR a -30º.
Lazo cerrado por distancia con ciclo de paro de emergencia	SI se muestra tarjeta roja, un ciclo para la continuación, inclinación NO mayor a 30ºC, inclinación MENOR a -30º, distancia medida SI mayor a 5 cm.

Fig. 5. Definición de lazos de flujo del sistema

2. Resultados:

La implementación de todos los sensores, se llevó a cabo, considerando que esta parte se puede modificar según las necesidades de las aplicaciones del robot, siendo posible sustituir sensores, modificando levemente el código de control del dispositivo. El resultado de toda la implementación de sensores realizada, así como el montaje de todos los componentes en el soporte puede apreciarse en la Fig. 6. Es remarcable que, en esta imagen, se puede apreciar la facilidad física de adición e intercambio de sensores que presenta el prototipo β.

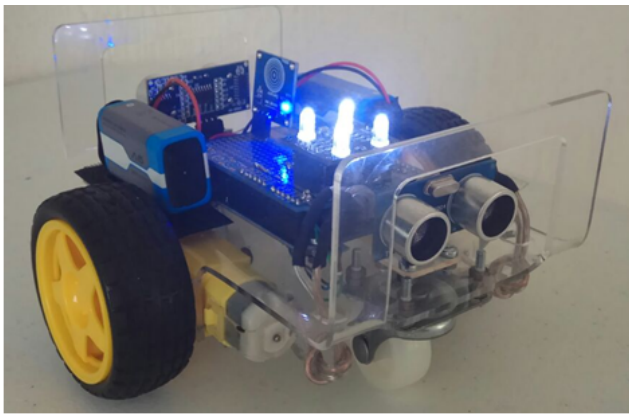

Fig. 6. Imágenes de la implementación de los sensores en el prototipo β, del robot.

Las pruebas de funcionamiento del algoritmo de seguridad, según el diagrama de flujo mostrado en la Fig. 1, demostraron el accionamiento del sistema según lo esperado.

En la Fig. 7 se muestran las gráficas de los sensores más representativos para el funcionamiento del dispositivo.

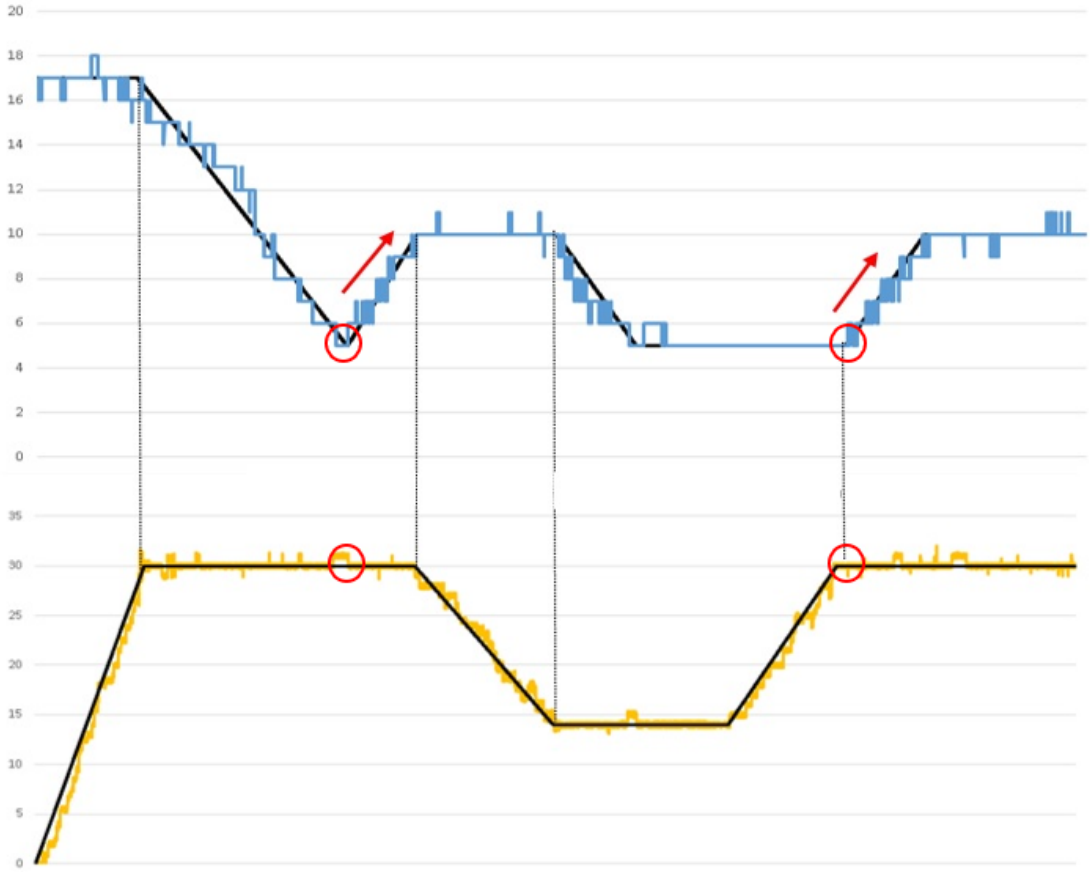

Fig. 7. Funcionamiento del código de control para los sensores en el prototipo β, del robot.

En las gráficas de la Fig. 7, se puede observar, en base a los dos parametros básicos del sistema, el correcto funcionamiento del algoritmo de seguridad mediante las pruebas

realizadas, además de poder apreciar los instantes exactos de intervención del sistema de seguridad (resaltados con flechas y circulos en rojo) mientras se cumplan las dos condiciones básicas de funcionamiento: una distancia medida de 5 cm o menor (gráfico azul, medición del HC-SR04) y una inclinación de, al menos, 30° (gráfico amarillo, medición del MPU-6050). Por el contrario, puede apreciarse que el sistema no interviene aunque la distancia medida sea 5 cm, mientras la inclinación no alcance el valor estipulado. Del mismo modo, tampoco interviene mientras la distancia no alcance el valor mínimo de 5 cm aunque la inclincación iguale o supere a los 30° de límite. Las líneas negras en ambas gráficas representan una estimación matemática ideal del movimiento real que se realizó en el ensayo, siendo, para el caso del HC-SR04, la distancia real a la que se colocó el obstáculo en al dirección de la inclinación, mientras que en la gráfica del MPU-6050, representa el grado de inclinación real que se le presentó al dispositivo.

La intervención del sistema de seguridad concluye con el posicionamiento del dispositivo en una distancia segura de 10 cm del obstáculo en la dirección de inclincación. Estos valores críticos de 30° y 5-10 cm son valores propios del prototipo β. La parte de alimentación para el robot se solucionó colocando dos baterías de 9V, a cada extremo del robot, sobre las ruedas, sin embargo la potencia de estas baterías no es la más adecuada para el robot, ya que le resta autonomía y, sobre todo, potencia de tracción. La alimentación para la plataforma Arduino no presenta mayores problemas con este tipo de configuración, pero toda la parte mecánica de tracción, se ve limitada en potencia debido a estas baterías.

En cuanto al estudio de tiempos realizado, los resultados se exponen en las siguientes tablas y gráficos:

Lazo abierto más largo posible	Tiempo
1. Variables:	0
2. Lectura de la condición de paro (color):	692
3. Lectura de la Inclinación:	724
4. Decisiones encadenadas:	726
5. Lectura de Distancia:	1418
6. Encendido de motores:	1425
7. *While* de 1 interacción lectura de Distancia:	2123
9. Apagado de motores:	**2136**

Fig. 8. Tabla y gráfico de tiempos de cómputo para el lazo abierto más largo.

Lazo abierto más largo posible con ciclo de paro-reanudación	Tiempo
1. Variables	0
2. Lectura de la condición de paro (color):	692
3. *While* de 1 interacción del paro-reanudación:	692
4. Lectura de la Inclinación:	730
5. Decisiones encadenadas:	742
6. Lectura de Distancia:	1440
7. Encendido de motores:	1447
8. *While* de 1 interacción de lectura de Distancia:	2146
9. Apagado de motores:	**2158**

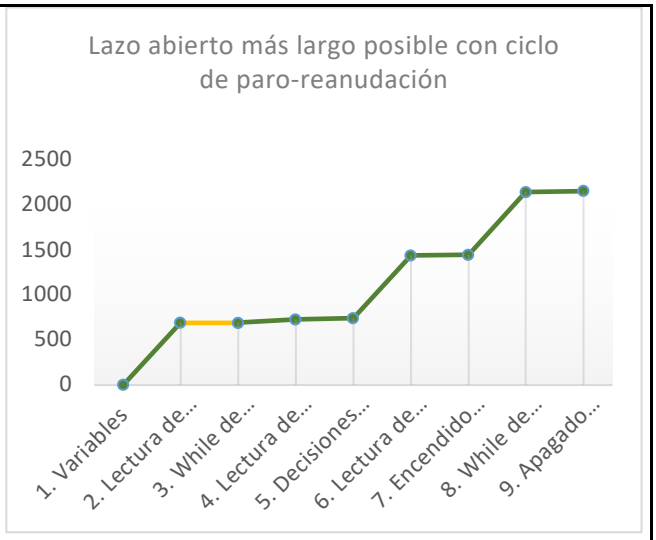

Fig. 9. Tabla y gráfico de tiempos de cómputo para el lazo abierto más largo con ciclo paro-reanudación

El detalle más destacado de la comparación de estas dos tablas es el hecho de que la acción añadida en la Fig. 9, el ciclo de paro (resaltado en naranja), no supone un aumento del tiempo en sí misma, pues la lectura al concluir esta acción refleja el mismo valor que la acción anterior, asumiendo que la acción añadida no consume tiempo de computo. Sin embargo, a partir de este punto, la toma de tiempos se desfasa, aumentando en las mediciones posteriores, resultando en un aumento promedio de 22 ms de una tabla a otra. Para fines prácticos, esto podría ser interpretado como que la acción añadida (ciclo de paro-reanudación) toma un tiempo de computo de **22ms,** aunque no se evidencie de forma directa en la gráfica. En la Fig. 10 puede observarse, con mayor detalle, cómo y cuándo se presenta este desfase medición a medición.

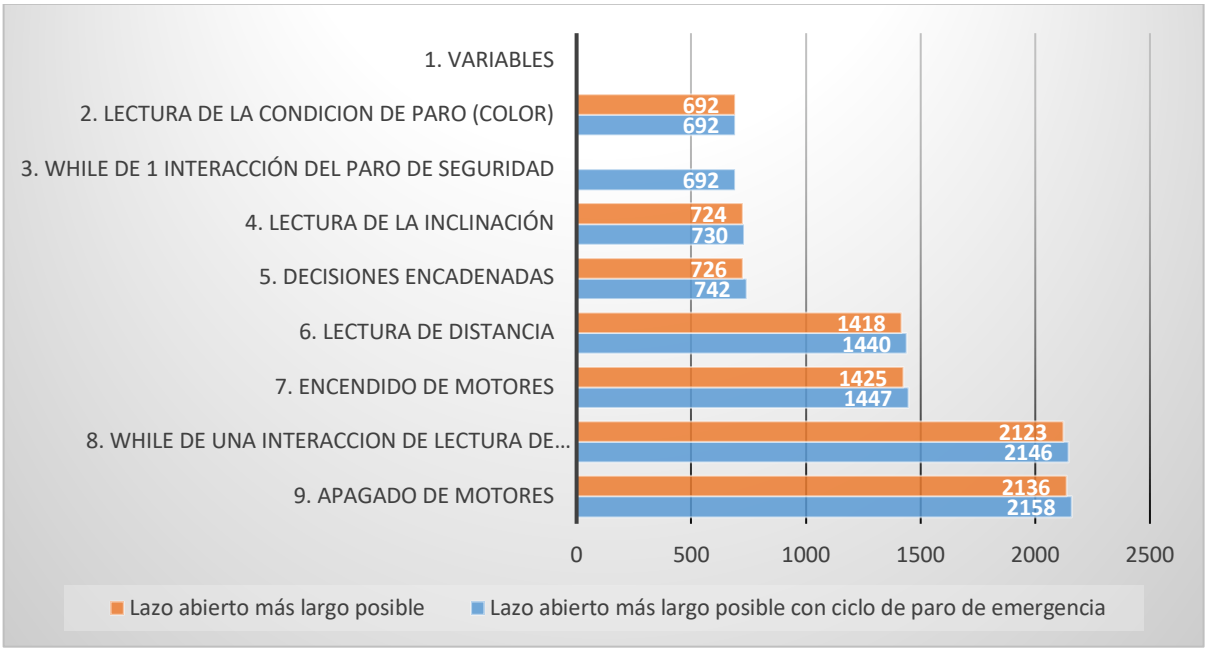

Fig. 10. Gráfico comparativo de tiempos con y sin ciclo de paro-reanudación para el lazo más largo.

También es conveniente mencionar que las acciones que mayor tiempo de computo toman son: 2. Lectura de condición de paro (color), 6. Lectura de Distancia y 8. While de 1 interacción de lectura de Distancia.

Interpretando esto, es posible entender que estas lecturas, son críticas debido a su papel relevante en los ciclos de lazo cerrado característicos del sistema. Las lecturas de tiempos realizadas sobre estos lazos representan los tiempos en los que el sistema, en un estado de inestabilidad o semi-inestabilidad, vuelve a estar listo para tomar la siguiente medición de distancia o inclinación. A continuación, se presentan las tablas con las lecturas de tiempos para estos lazos:

Lazo cerrado por inclinación	SIN ciclo de paro	CON ciclo de paro
1. Variables	0	0
2. Lectura de la condición de paro (color)	692	692
3. *While* de 1 interacción del paro de seguridad	-	692
4. Lectura de la Inclinación	724	730
5. Decisiones; inicio 2º ciclo	**726**	**733**

Fig. 11. Tabla de tiempos de cómputo para el lazo cerrado por inclinación.

El resultado de la Fig. 11 implica que 726 ms son necesarios para que el sistema, una vez ha comprendido que la inclinación actual no supone peligro alguno, vuelva a realizar otra medición de inclinación y opere en consecuencia. Esto implica que, si una inclinación repentina acontece en menos de este tiempo, el sistema llegará tarde a su detección, pudiendo tardar como máximo 726 ms en detectar la perturbación (o 733 ms si se ha cumplido la condición de paro y reanudación en ese ciclo. Es decir, el sistema tarda 726 ms en volver a estar en condiciones de actuar, tras haber sensado el ciclo anterior, algo totalmente inaceptable para el desempeño del sistema de seguridad. Para este lazo, el tiempo añadido por la acción de paro-reanudación es de 13 ms.

Lazo cerrado por distancia	SIN ciclo de paro	CON ciclo de paro
1. Variables	0	0
2. Lectura de la condición de paro (color)	692	692
3. *While* de 1 interacción del paro-reanudación	-	692
4. Lectura de la Inclinación	724	730
4. Decisiones encadenadas	726	742
5. Lectura de Distancia	1418	1440
6. Inicio 2º ciclo	**1418**	**1440**

Fig. 12. Tabla de tiempos de cómputo para el lazo cerrado por distancia.

De forma semejante al lazo anterior, el resultado de la Fig. 12 implica que 1418 ms son necesarios para que el sistema, una vez ha comprendido que la inclinación es crítica, pero distancia actual no supone peligro alguno, vuelva a realizar otra medición de distancia y opere en consecuencia. Esto implica que, si la distancia al obstáculo en la dirección de la inclinación disminuye de forma repentina en menos de este tiempo, el sistema llegará tarde a su detección, pudiendo tardar como máximo 1418 ms en detectar la perturbación y 1425 ms en reaccionar a ella como se puede observar en la Fig. 8. Es decir, para este ciclo, el sistema tarda 1440 ms en volver a estar en condiciones de actuar, tras haber evaluado el ciclo anterior, algo totalmente inaceptable para el correcto desempeño del sistema de seguridad debido a la propia naturaleza del efecto de corrección. El sistema tardará 1440 y 1447 ms en realizar estas acciones respectivamente, si se ha cumplido la condición de paro y reanudación en ese ciclo, por tanto, para este lazo, el tiempo añadido por la acción de paro-reanudación es de 22 ms.

A modo de resumen y comparación final se presenta la Fig. 13 con los tiempos finales de los tres lazos analizados, con y sin ciclo de paro-reanudación.

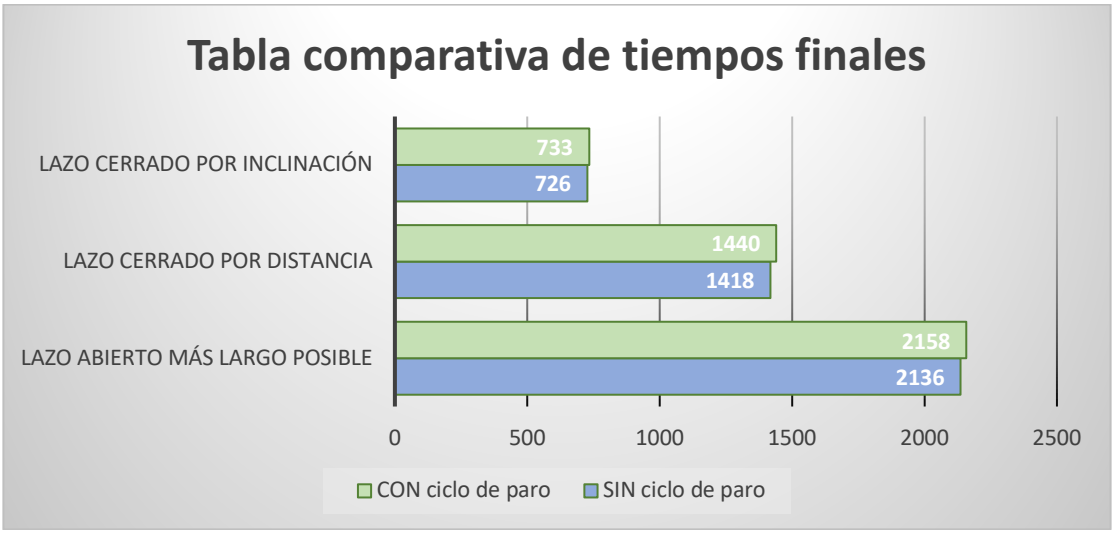

Fig. 13. Gráfico comparativo de tiempos finales de los lazos analizados.

Por último, es conveniente añadir dos consideraciones importantes para entender el sistema:

- La acción de lectura de la inclinación, únicamente toma los datos en un solo plano, por lo que, de ser necesarios datos de inclinación en otros planos o ejes de rotación, la lectura de estos datos aumentaría, presumiblemente, el tiempo de cómputo consumido por esta acción.

- Durante todo este análisis se han estado considerando tiempos únicamente de cómputo, esto quiere decir que para obtener los tiempos finales de respuesta del sistema debe considerase, también, el tiempo que

cada sensor y actuador requiere para tomar la medición de forma física.

3. Conclusiones:

Del desarrollo del presente trabajo, se pueden obtener las siguientes conclusiones:

Se diseñó, construyó y caracterizó un robot móvil, basado en una plataforma Arduino, el cual emula el funcionamiento para un modelo final. El funcionamiento de los sensores empleados, permitieron obtener respuestas conforme lo estipulado en el diagrama de flujo. Sin embargo la lógica tanto en el diagrama, principalmente en el código de control de los sensores, permitió obervar una serie de fallas, las cuales deben ser corregidas para un desempeño más adecuado del robot.

Respecto al hardware, durante las pruevas realizadas, se pudo obserbar la poca tracción o agarre de los neumaticos utilizados, dejando al robot móvil en una situación de inestabilidad nada deseable para la aplicación final del robot. Ademas, toda la lógica de programacion se basa en (o asume) que las ruedas del robot, en ningún momento, estén ancladas mecanicamente al motor, por lo que, si no tienen tracción, estas rodaran en el sentido de la pendiente. Por esto, tambien es necesario que la superficie de las ruedas consiga un elevado coeficiente de rozamiento con la supercie.

En relación a esto, se pueden definir las dos mayores problematicas del prototipo β que deberán ser corregidas en versiones posteriores del dispositivo:

- Los motores requieren una mayor potencia para generar mayor tracción en las ruedas, para que estas sean capaces de llevar al dispositivo a una posición segura venciendo la pendiente de inclinación.
- El coeficiente de friccion de las ruedas debe ser muy superior al generado por las ruedas utilizadas para evitar que estas resbalen con la superficie inclinada.

Adicionalmente a lo ya mencionado, y tomando como referencia los resultados obtenidos en la toma de tiempos de cómputo, puede concluirse que es completamente necesario un cambio en el hardware del controlador base del sistema, pues los tiempos obtenidos por la placa Arduino UNO son inaceptables para la correcta funcionalidad del dispositivo en un ambiente real, pues un tiempo de respuesta de 1440, o incluso 726 ms entre mediciones deja una ventana demasiado grande entre fallas del plano y la acción de corrección**.** Para este cambio puede proponerse una tarjeta Raspberry pi 3 debido a su mejor velocidad de procesarimento. Esta velocidad puede obserbarse con datos cuantificados en [6] comparando su rendimiento con una computadora personal.

3.1 Trabajo a futuro y proyección del proyecto:

Se planteó en este proyecto el diseño de un prototipo β para probar y evaluar el algoritmo del sistema de seguridad desarrollado, sin embargo este primer prototipo, aunque evalúa, de forma satisfactoria, la funcionalidad del software, requiere de la inmediata corrección de errores básicos de diseño que afectan su desempeño. Una vez analizadós estos errores básicos, expuestos al término de las concluciones del este proyecto, es fácil identificar cual debe ser la evolución inmediata, la cual debe disponer de un sistema de tracción y alimentación mejorado, así como unos neumáticos con mucha mayor aderencia en diversos terrenos. También será recomendable realizar un cambio de tarjeta

Arduino a un modelo superior para mejorar la capacidad y velocidad de procesamiento.

Si bien este proyecto puede servir de báse consistente de la que pártir, la corrección de estos errores, sumado a la mejora en el procesador y la mejora de tiempos de respuesta que ello ocasionará, conseguirá establecer un **prototipo β_1,** dando, con él, el primer paso para el desarrollo de un modelo final del dispositivo explorador, completamente funcional y comercial que incorpore el algoritmo del sistema de seguridad desarrollado.

Desde este proyecto, es posible establecer una proyección estimada del desarrollo de prototipos hasta llegar al modelo final, apuntando, además, los aspectos clave que debe mejorar o implementar cada evolución:

Prototipo β	• Implementación simple y evaluación del sistema de seguridad. • 1er Análisis de tiempos de cómputo. Primer prototipo.
Prototipo β_1	• Solución de errores básicos. • Cambio de controlador base para evaluar mejor de tiempos de cómputo. • 2o análisis de tiempos de cómputo.
Prototipo α	• Definición del tipo de actividad exacta a la que irá destinado el dispositivo. • Selección e implementación de hardware comercial/industrial. • Optimización software de seguridad e implementación control manual por RC.
Prototipo α_1	• Optimización interelación sistema de seguridad-control manual. • 3ero análisis de tiempos de respuesta (análisis final).
Modelo Final	• Análisis/diseño estructural y resistencia de materiales (simulaciones estructurales). • Consideraciones de manufactura. • Estudio de comercialización.

Fig. 14. Proyección y evaluación del proyecto.

4. Referencias:

[1] Manual de programación Arduino uno.
https://arduinobot.pbworks.com/f/Manual+Programacion+Arduino.pdf, last accessed 2017/11/21.

[2] MPU-6050 Homepage, https://playground.arduino.cc/Main/MPU-6050, last accessed 2017/11/21.

[3] HC-SR04 Homepage http://www.naylampmechatronics.com/blog/10_Tutorial-de-Arduino-y-sensor-ultras%C3%B3nico-HC-S.html last accessed 2017/11/21

[4] TTP223B Homepage, https://radiokot.ru/konkursCatDay2014/53/01.pdf, last accessed 2017/11/21.

[5] TCS230 Homepage, http://howtomechatronics.com/tutorials/arduino/arduino-color-sensing-tutorial-tcs230-tcs3200-color-sensor/, last accessed 2017/11/21.

[6] Cavanzo, G., Pérez, M., Villavisan, F. 2017. «Medición de eficiencia de algoritmos de visión artificial implementados en raspberry pi y ordenador personal mediante Python», Ingenium, vol. 18. n.° 35, pp. 105-119.

[7] Maity, A., Paul, A., Goswami, P., and Bhattacharya, A. 2017. Android Application Based Bluetooth Controlled Robotic Car.*International Journal of Intelligent Information Systems*. Vol. 6, No. 5, pp. 62-66. doi: 10.11648/j.ijiis.20170605.12.

[8] Zhao, P., Chen, J., Song, Y., Tao, X., Xu, T., and Mei, T. 2012. «Design of a Control System for an Autonomous Vehicle Based on Adaptive-PID», *Int J Adv Robotic Sy*, Vol. 9, pp. 44.

[9] František Duchoň, F., Hubinskýa, P., Hanzela, P., Babineca, A., and Tölgyessya, M. 2012. «Intelligent vehicles as the robotic applications», *Procedia Engineering* 48, pp. 105 – 114.

[10]. Shaokun, W., Xiao, X., Hongwei, Z. 2011. «The Wireless Remote Control Car System Based On ARM9», *International Conference on Instrumentation, Measurement, Computer, Communication and Control*.