UNIVERSIDAD TÉCNICA DE MANABÍ
Fundada en 1952

INFORMÁTICA Y SISTEMAS
Revista de Tecnologías de la Informática y las Comunicaciones

Diseño e implementación de software y hardware de un prototipo de vehículo autónomo recolector de objetos basado en tecnología Arduino

Design and implementation of software and hardware of a prototype autonomous object collector vehicle based on Arduino technology

Autores

*Sánchez Rogel Kevin Damián

Ibarra Pincay Melanie Cristina

Triviño Valdez Diego Lenin

Universidad Técnica de Manabí, Facultad de Ciencias Informáticas, Portoviejo, Ecuador.

* Autor para correspondencia

Comó citar el artículo: Sánchez Rogel, K.D., Ibarra Pincay, M.C. y Triviño Valdez, D.L. (2022). Diseño e implementación de software y hardware de un prototipo de vehículo autónomo recolector de objetos basado en tecnología Arduino. *Informática y Sistemas Revista de Tecnologías de la Informática y las Comunicaciones. 6*(2), 78–90. https://doi.org/10.33936/isrtic.v6i2.4399

Enviado: 09/02/2022
Aceptado: 22/06/2022
Publicado: 01/07/2022

Resumen

En este artículo se presenta el diseño e implementación de software y hardware para elaborar un prototipo de vehículo autónomo recolector de objetos. Los componentes electrónicos necesarios se conectan y comunican con un Arduino Mega 2560, y estos son alimentados con baterías de ion de Litio, para definir el amperaje y voltaje necesario de estas baterías se realizaron cálculos teóricos del consumo de cada componente. En cuanto al algoritmo desarrollado, permite el movimiento del vehículo y del brazo robótico de forma manual y remota, por medio de un mando Wireless de PS2. Además de esto, gracias a la integración de sensores, se incorporó funcionalidades de detección de distancia, con esto se puede reconocer si hay un objeto o no delante del vehículo, para que así, el vehículo pueda recoger objetos de forma automática y almacenarlos en un contenedor. Con lo expuesto, se denota la particularidad de la implementación, el prototipo tiene más de una forma de uso, teniendo un control manual o automático y de forma remota. También se realizaron distintas pruebas de funcionalidad del prototipo respecto a la distancia entre el vehículo recolector y el objeto, obteniendo datos e interpretándolos. De esta forma y en conjunto a las funcionalidades programadas para controlar el vehículo, el brazo robótico y el control automático han dado como resultado un prototipo funcional de vehículo recolector de objetos controlado remotamente, que presenta una probabilidad de éxito inversamente proporcional, a mayor distancia se encuentra el objeto respecto al vehículo, menor será la probabilidad de éxito para recolectarlo. La metodología a seguir para la implementación de un vehículo autónomo recolector de objetos con Arduino Mega 2560, consiste en: 1) diseño del hardware, 2) implementación del hardware, 3) diseño del software y algoritmo, 4) implementación del software y algoritmo, 5) pruebas técnicas de funcionamiento. Este prototipo se puede tomar como base para la creación de equipos autónomos recolectores de objetos, para distintas áreas de aplicación. De aquí pueden partir soluciones para mejorar o desarrollar dispositivos que complementen una necesidad específica en el ámbito científico, investigativo o industrial

Palabras clave: Desplazamiento vehículo, Arduino Mega, Vehículo autónomo, Recolector objetos.

Abstract

This article presents the design and implementation of software and hardware to develop a prototype of an autonomous object collector vehicle. The necessary electronic components are connected and communicate with an Arduino Mega 2560, and these are powered by Lithium-ion batteries, to define the necessary amperage and voltage of these batteries, theoretical calculations of the consumption of each component were made. Regarding the developed algorithm, it allows the movement of the vehicle and the robotic arm manually and remotely, by means of a Wireless PS2 controller. In addition to this, thanks to the integration of sensors, distance detection functionalities were incorporated, with this it is possible to recognize whether or not there is an object in front of the vehicle, so that the vehicle can pick up objects automatically and store them in a container. With the above, the particularity of the implementation is denoted, the prototype has more than one way of use, having manual or automatic control and remotely. Different prototype functionality tests were also

INFORMÁTICA Y SISTEMAS
Revista de Tecnologías de la Informática y las Comunicaciones

Facultad de Ciencias Informáticas
Universidad Técnica de Manabí
Av. Urbina y Che Guevara, Portoviejo, Ecuador
revista.iys@utm.edu.ec

carried out regarding the distance between the collection vehicle and the object, obtaining and interpreting data. In this way and together with the functionalities programmed to control the vehicle, the robotic arm and the automatic control have resulted in a functional prototype of a remotely controlled object collection vehicle, which presents an inversely proportional probability of success, the greater the distance If you find the object relative to the vehicle, the lower the chance of success in collecting it. The methodology to follow for the implementation of an autonomous object collector vehicle with Arduino Mega 2560, consists of: 1) hardware design, 2) hardware implementation, 3) software and algorithm design, 4) software and algorithm implementation, 5) technical performance tests. This prototype can be taken as a basis for the creation of autonomous object-collecting equipment for different application areas. Solutions can start from here to improve or develop devices that complement a specific need in the scientific, research or industrial field.

Keywords: Vehicle displacement, Arduino Mega, Autonomous vehicle, Object collector.

1. Introducción

Al día de hoy, los microcontroladores son cada vez más necesarios dentro de nuestra vida cotidiana, desde calentar comida en un microondas o realizar una transacción en un cajero automático, hasta automatizar procesos como la navegación de autos y aviones. Su amplio uso, ha hecho que se vayan desarrollando placas e interfaces electrónicas en gran cantidad, adecuadas para cumplir con las necesidades y requerimientos tecnológicos de las industrias que tienen sistemas que son controlados y operados por personas (Tapia Ayala & Manzano Yupa, 2013). Cabe indicar, la existencia de una variedad de proyectos y tesis que implementan infraestructuras de hardware que tienen como pieza indispensable estas placas impresas con microprocesador, tales como; sistemas de control de posición de un objeto, prototipo de sistema inteligente de prevención de accidentes de tránsito, desarrollo de vehículo autónomo, construcción de un robot explorador en zonas de riesgo, entre otros Casquete Quiñonez & Moreira Maridueña, 2018).

Ahora bien, existen situaciones en donde en el ámbito investigativo, científico, industrial o alguna otra área, requieren explorar y recolectar muestras de suelo (Ministerio del Ambiente de Perú, 2014), objetos, desechos tóxicos, entre otros elementos que se encuentran en un espacio inaccesible o peligroso para el ser humano, o simplemente donde se desempeñe un trabajo de recolección constante de objetos en un área.

En este contexto, el presente artículo datará el diseño e implementación de software y hardware de un prototipo de vehículo autónomo recolector de objetos basado en tecnología Arduino, el cuál desempeñará la función de recorrer y recolectar objetos en un espacio, con ayuda de un brazo robótico, controlado remotamente y con un modo manual y automático de uso. Este prototipo se desarrollará con el controlador ATmega2560, la tecnología Arduino y su software, que nos permite implementar funcionalidades a un hardware o dispositivo, es decir, crear objetos interactivos, programar instrucciones de forma sencilla, además de facilitar la conexión e implementación de hardware a las placas Arduino (Torrico Santos, 2016).

Esto aportará de tal manera que, se pueda tomar como base este prototipo para la creación de equipos autónomos recolectores de objetos, para distintas áreas de aplicación. De aquí pueden partir soluciones para mejorar o desarrollar dispositivos que complementen una necesidad específica como en el área de la salud e investigación científica.

2.Materiales y Métodos

La metodología a seguir para la implementación de un vehículo autónomo recolector de objetos con Arduino Mega 2560, consiste en: 1) diseño del hardware, 2) implementación del hardware, 3) diseño del software y algoritmo, 4) implementación del software y algoritmo, 5) pruebas técnicas de funcionamiento.

Se planea tener como cerebro del carro dicho microcontrolador que será programado con la función de procesar la información que reciba de los componentes electrónicos conectados, como, por ejemplo, de los sensores, y a su vez, poder enviarle órdenes, como a los motores de las ruedas: avanzar, retroceder, girar a la derecha o izquierda, también controlar cada movimiento de las partes del brazo robótico y activar función de recoger objetos automáticamente mediante los sensores o realizar esta función con un control manual.

Para poder controlar los motores de las ruedas del vehículo, el Shield Driver L293D será acoplado al Arduino. El microcontrolador, tendrá la programación para el correcto funcionamiento del vehículo. De igual manera, se hace uso del módulo PCA- 9685 para controlar los 4 servo motores que conforman el brazo robótico, además del control directo del carro y del brazo robótico que se logrará empleando mando de PS2 con su respectivo receptor Wireless.

Se implementarán los componentes necesarios (sensor ultrasónico e infrarrojo), para lograr que el vehículo pueda de forma automática encontrar un objeto, posicionarse en frente de él a una distancia de 9cm a 10cm y el brazo robótico pueda recolectarlo por sí solo. Para tener una correcta distribución de voltaje se usará módulos Step Down para reducir el voltaje de la fuente de alimentación (Peña Uriarte, 2019) que son pilas 18650 de 3.7V y 2000 mAh.

A continuación, se enlistará los materiales necesarios para llevar a efecto esta implementación:

e-ISNN 2550-6730 | Vol. 6, Núm. 2 (78-90): Julio-Diciembre, 2022 | DOI: 10.33936/isrtic.v6i2.4399

Tabla 1. Equipos y materiales usados para el desarrollo e Implementación del prototipo.

Equipos	Materiales
Arduino MEGA 2560	Tornillos y tuercas
Driver L293D para motores	Baterías 18650 2000mAh
Sensor Ultrasónico HC-SR04	Jumpers para conexiones
Servo Motor SG90	Soporte para Sensor Ultrasónico
Sensor Infrarrojo	Chasis 2 pisos 4WD
Motores DC 3V A 6V	18650 Portapilas Batería Litio
Protoboard	Ruedas de 66mm
Switch on/off	

3.Diseño del Hardware del prototipo

3.1 Diagrama de conexiones

Para comprender mejor el hardware utilizado y el diseño de las conexiones para el vehículo autónomo recolector de objetos, observemos el diagrama de la Figura 1.

Las conexiones no tienen un código de color uniforme, aunque en general se ha respetado el uso del rojo como señal de voltaje positivo (VCC o +5V), y el negro el de masa (GND). Lo que sí que se corresponde es exactamente con el color de los cables utilizados en el prototipo real (Navarro Cosme, 2016).

De acuerdo al diagrama anterior contamos con el siguiente hardware:

Arduino Mega 2560:

Por medio de esta placa microcontroladora, conectaremos el sensor ultrasónico, sensor infrarrojo, receptor PS2, módulo PCA9685 para controlar los servos y el Motor Shield L293D para el manejo de los motores de las ruedas, se leerán los datos de ellos a través de los pines y se podrán programar las respectivas instrucciones para el funcionamiento del vehículo Campo Jorge et al., 2017), brazo robótico y sistema de recolección de objetos.

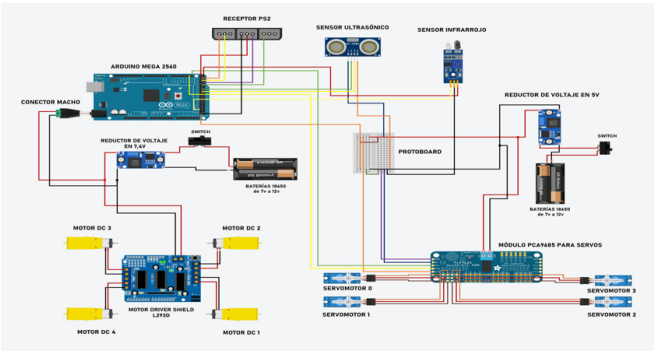


Figura 1. Diagrama de conexiones del Arduino Mega 2560 y los componentes electrónicos

Motor Shield L293D:

Este shield o escudo nos permitirá facilitar el control de hasta cuatro motores DC, gracias a sus dos drivers puente H L293D, se puede controlar la velocidad y dirección de giro de los motores, que serán usados para darle movimiento a las ruedas (Circuit Best, 2021). Este escudo estará montado en el Arduino Mega 2560 y tendrá conectado 4 motores DC. Cabe indicar que para que la energía proporcionada a este escudo sea independiente de la de Arduino, quitamos el jumper de configuración que incluye la placa (Lady Ada, 2012).

Módulo PCA9685:

A través de este módulo, conectaremos los 4 servomotores que serán usados para el funcionamiento del brazo robótico y es el que nos permitirá posteriormente controlar cada servo, en cuanto a su posición. Este módulo genera señales PWM, las cuales son empleadas por los servomotores para realizar sus movimientos (Earl, 2012), estas señales se transformarán a grados para tratar a los servos con mayor facilidad. Cuenta con seis pines, GND, 0E, SCL, SDA, VCC, V+. En la conexión de V+ se provee energía externa. La conexión entre los pines del módulo y el Arduino Mega se muestran en la siguiente tabla:

Tabla 2. Configuración de pines módulo PCA-9685

PCA9685	Arduino Mega
GND	GND
0E	GND
SCL	21
SDA	20
VCC	5V
V+	5V*

Reductores de voltaje LM2590:

El módulo convertidor DC-DC Step-Down (Buck) LM2596 es capaz de regular un voltaje de entrada mayor y convertirlo en uno menor a su salida, voltaje de entrada: 4.75 a 30V, voltaje de salida: 1.25 a 26V y con corriente promedio de salida de 2A. (Semiconducter Componentes Industries, s.f.) Es usado para reducir el voltaje que se dirige hacia la placa Arduino Mega (Reducimos a 7.4V) y la placa PCA9685 para servos (reducimos a 5V).

Sensor Ultrasónico HC-SR04:

Permite medir distancia mediante ultrasonido desde 2cm hasta 450cm, lo usaremos para saber a qué distancia se encuentra el objeto a recoger, la cuál será de 9 cm a 10 cm. Este sensor funciona emitiendo ondas de ultrasonido hacia delante, cuando estas ondas chocan con un objeto, estas rebotan hacia el sensor, y él las capta (ElecFreaks, 2013), el tiempo que se demora en regresar la onda, es transformado en distancia por software. Este sensor HC-SR04 cuenta con dos pines, Trig: el cual genera el

INFORMÁTICA Y SISTEMAS
Revista de Tecnologías de la Informática y las Comunicaciones

Facultad de Ciencias Informáticas
Universidad Técnica de Manabí
Av. Urbina y Che Guevara, Portoviejo, Ecuador
revista.iys@utm.edu.ec

Tabla 3. Configuración de pines HC-SR04

HC-SR04	Arduino Mega
GND	GND
Echo	34
Trig	32
VCC	5V

pulso y Echo quien lo recibe. Conexión:

Sensor Infrarrojo HW-201:

Tiene un par de tubos de transmisión y recepción de infrarrojos. El tubo de transmisión emite luz infrarroja de una cierta frecuencia y esta nos servirá para detectar a un máximo de 11cm si hay o no un objeto al frente, en la posición conveniente para que el brazo robótico pueda recolectar el objeto. Los pines son 3 y están conectados a Arduino Mega de la siguiente manera:

Tabla 4. Configuración de pines HW-201

HW-201	Arduino Mega
OUT	50
GND	GND
VCC	5V

Receptor Wireless de PS2:

Logra la conexión de un mando de Play Station 2 con el Arduino Mega, para así usarlo y poder dar órdenes de movimiento y otras funcionalidades al vehículo recolector de objetos. En cuanto a la conducción manual se tendrá de una forma fluida sin in- terrupciones, la conexión del receptor al Arduino Mega se realiza por los siguientes pines (Macho, s.f.a):

Tabla 5. Configuración de pines receptor PS2

PS2 Controller	Arduino Mega
data	22
command	24
vibration	No conectado
GND	GND
VCC	5V
attention	26
clock	28
unknown	No conectado
acknokedge	No conectado

4.Implementación del Hardware del prototipo

4.1Estructura del vehículo

Con el hardware previamente mencionado, se implementó la estructura para el vehículo recolector de objetos. Lo que se ha ido realizando, dando como resultado el esquema del Arduino, es la colocación de las placas Arduino Mega, el Driver Motor Shield L293D, el módulo PCA9685, y a través de ellos conectar, los servomotores, motores DC, sensor ultrasónico, sensor infrarrojo y receptor del mando de PS2 para controlar al carro, en conjunto con las piezas que se requieren para el funcionamiento del vehículo recolector Arduino, que en este caso son el chasis, las ruedas, interruptores, porta pilas y pilas de Li-ion de 3.7 voltios cada una, protoboard, módulos reductores de voltaje, jumpers de conexión, contenedor para objetos y estructura del brazo robótico.

Se muestra a continuación imágenes de cómo queda armado el vehículo recolector de objetos Arduino:

Vista general del prototipo:

En la posterior figura presenciamos la estructura del vehículo, el chasis se compone de dos láminas de acrílico, lo que permite tener dos pisos de espacio para ubicar los componentes, además consta de 4 ruedas, brazo robótico y contenedor de objetos. El prototipo pesa aproximadamente 0.6Kg y tiene una altura total con el brazo robótico extendido hacia arriba de 30cm.

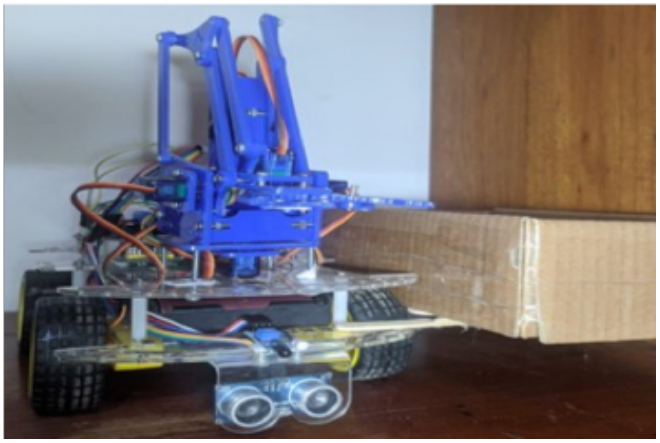


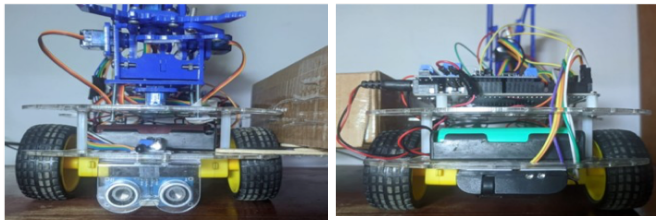
Figura 2. Vista general del vehículo

Vista frontal y trasera del prototipo:

En la parte frontal del vehículo podemos ver que está montado el sensor infrarrojo y el sensor ultrasónico, en el chasis del carro

e-ISNN 2550-6730 | Vol. 6, Núm. 2 (78-90): Julio-Diciembre, 2022 | DOI: 10.33936/isrtic.v6i2.4399

además de dos baterías 18650 para alimentar el brazo robótico (Figura 3.a). En la parte de atrás del vehículo, encontramos el receptor Wireless de PS2, para poder controlarlo remotamente, también un conjunto de 3 baterías 18650 para alimentar el Arduino y el Motor Shield (Figura 3.b).

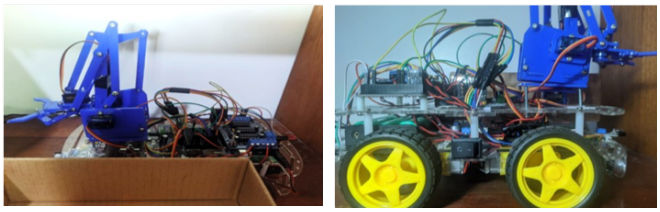


a. Vista frontal del vehículo **b**. Vista trasera del vehículo

Figura 3. Vista frontal y trasera del prototipo

Vista lateral izquierda y derecha del prototipo:

A la izquierda, podemos observar el con- tenedor de los objetos, que puede adaptarse otros más grandes o pequeños, también una vista del brazo robótico en su posición de descanso y el switch off/on de los servos (Figura 4.a). A la derecha, podemos presenciar que el vehículo consta de dos pisos, en el superior tenemos montado el Arduino Mega, el motor shield, protoboard, módulo PCA9685 y brazo robótico. En el piso inferior ubicamos los 2 módulos reductores de voltaje, dos portapilas con sus respectivas baterías y un switch on/off para el Arduino y los motores (Figura 4.b).



a. Vista frontal del vehículo **b**. Vista trasera del vehículo

Figura 4. Vista lateral izquierda y derecha del prototipo

4.2 Especificaciones de energía y alimentación

Este vehículo es alimentado por dos fuentes de energía independientes:

La primera fuente: consta de un set de tres pilas 18650 Li-ion a 3.7V y 1800mAh, conectadas en serie para que el voltaje de las 3 pilas se sume, dando alrededor de 12.30V con la carga completa y 11.10V con la carga media. Este conjunto de baterías es conectado a un switch on/off, y luego conectado a un módulo Step Down, para reducir el voltaje de entrada y obtener un voltaje de salida de 7.4V, esto con el fin de no sobre calentar el Arduino Mega, ya que a partir de 7V, el excedente de voltaje lo convierte en calor. Luego por la misma salida del módulo se hizo una conexión en paralelo, obteniendo otro cable de la salida positiva y otro cable de la salida negativa, para alimentar al Motor Shield L293D, el cual tendrá el jumper de configuración quitado, para que la energía que entre al Shield, no pase al Arduino y sea exclusivamente para los motores DC, eliminando así el efecto negativo del ruido de los motores hacia el Arduino. Cabe indicar que la conexión de los cables positivo y negativo deben ser ubicados correctamente en el Motor Shield, ya que este no tiene protección contra la polaridad inversa (Lady Ada, 2012).

A continuación, podemos ver un diagrama de las conexiones previamente detalladas:

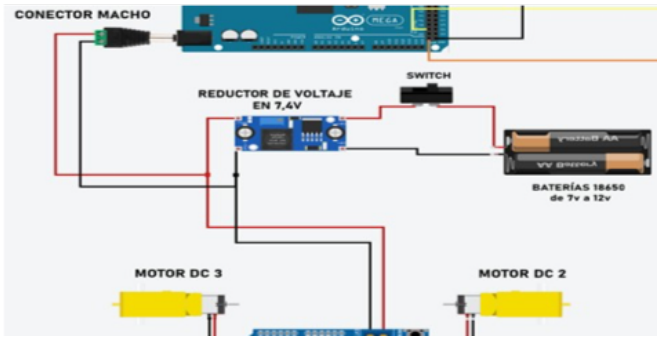


Figura 5. Conexiones de la fuente de energía del Arduino y Motor Shield

La segunda fuente: está compuesta por un set de dos pilas 18650 Li-ion a 3.7V y 2000mAh, conectadas en serie para que el voltaje de las 2 pilas se sume, dando alrededor de 8.20V con la carga completa y 7.4V con la carga media. Este conjunto de baterías es conectado a un switch on/off, y luego conectado a un módulo Step Down, para reducir el voltaje de entrada, y obtener un voltaje de salida de 5.05V, ya que esta es la cantidad adecuada de voltaje para alimentar los servomotores, ya que si se sobrepasa este voltaje, se podrían quemar, esta alimentación se la hace en las borneras del módulo PCA9685, para que la alimentación sea independiente al resto de componentes, este módulo es el encargado de distribuir los 5V a cada servomotor conectado. Además, en las salidas del módulo Step Down se hace una conexión en paralelo para suministrar al protoboard de 5V y así poder usar esta alimentación para los distintos sensores que la requieran.

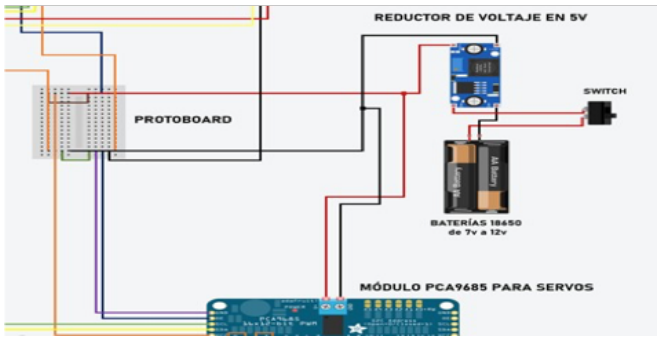


Figura 6. Conexiones de la fuente de energía para el módulo PCA9685 y el protoboard

INFORMÁTICA Y SISTEMAS
Revista de Tecnologías de la Informática y las Comunicaciones

Facultad de Ciencias Informáticas
Universidad Técnica de Manabí
Av. Urbina y Che Guevara, Portoviejo, Ecuador
revista.iys@utm.edu.ec

4.3 Autonomía energética

A continuación, se analizará el consumo de los componentes y se aproximará de forma teórica el tiempo de funcionamiento constante del vehículo recolector de objetos.

Fuente de energía I: Tiene un voltaje promedio de 11.10V y una capacidad de corriente de 1800mAh. Alimenta los siguientes componentes:

Tabla 6. Tabla de consumo de corriente de la fuente de 1800mAh

Componentes	Cantidad	Consumo de mA por unidad	Consumo de mA total
Arduino Mega	1	96mA	96mA
Receptor PS2	1	15mA	15mA
Sensor IR	1	20mA	20mA
Motor Shield L293D	1	20mA	20mA
Motor DC	4	150mA	600mA
		Corriente total	**751mA**

Consideramos los valores de mA de los componentes estando en uso, por lo tanto, estos suman un total de consumo de corriente de 751mA, ahora para saber cuánto tiempo pueden estar funcionando teniendo una fuente de energía que nos proporciona 1800mAh, debemos realizar el siguiente cálculo:

Dividimos los mAh (miliamperios por hora) proporcionados por la fuente de alimentación sobre el mA (miliamperio) total que consumen todos los componentes (Macho, s.f.b).

Obtenemos el resultado en horas, para convertirlo a minutos, multiplicamos el resultado por 60.

2,397 * 60 = 143,82min ≈ 144 minutos

Por lo que, en un uso constante de estos componentes, las baterías durarían aproximadamente 2 horas con 40 minutos antes de descargarse. El vehículo podría estar en movimiento constante por ese tiempo.

Fuente de energía II: Tiene un voltaje promedio de 7.4V y una capacidad de corriente de 2000mAh. Alimenta los siguientes componentes:

Consideramos los valores de mA de los componentes estando en uso, por lo tanto, estos suman un total de consumo de corriente de 2026mA, ahora para saber cuánto tiempo pueden estar funcionando teniendo una fuente de energía que nos proporciona 2000mAh, debemos realizar el siguiente cálculo:

Tabla 7. Tabla de consumo de corriente de la fuente de 2000mAh

Componentes	Cantidad	Consumo de mA por unidad	Consumo de mA total
Servomotor SG90	4	500mA	2000mA
Sensor Ultrasónico	1	15mA	15mA
Módulo PCA9685	1	10mA	10mA
		Corriente total	**2026mA**

Dividimos los mAh (miliamperios por hora) proporcionados por la fuente de alimentación sobre el mA (miliamperio) total que consumen todos los componentes “(Macho, s.f.b)”.

Obtenemos el resultado en horas, para convertirlo a minutos, multiplicamos el resultado por 60.

0,987 * 60 = 59,23min ≈ 59 minutos

Por lo que, en un uso constante de estos componentes, las baterías durarían aproximadamente 2 horas con 40 minutos antes de descargarse. El vehículo podría estar en movimiento constante por ese tiempo.

Por lo que, en un uso constante de estos componentes, las baterías durarían aproximadamente 59 minutos antes de descargarse. Cabe indicar que, el brazo robótico solo llega a consumir 500mA (cada servomotor) cuando está en uso, por cada recolecta de objeto, los servomotores funcionan alrededor de 20 segundos, entonces podemos calcular la cantidad de objetos que puede recoger hasta que se agote la batería (convertimos los 59 min en segundos para el cálculo): 3540s/20s=177, teniendo un total de 177 recolectas de objetos hasta acabar la batería de 2000mAh.

5.Diseño del Software y algoritmo del prototipo

El entorno de desarrollo para Arduino está constituido por un editor de texto para escribir el código, un área de mensajes, una consola de texto, una barra de herramientas con botones para las funciones comunes, y una serie de menús (Molina Cruz et al., 2019) como

e-ISNN 2550-6730 | Vol. 6, Núm. 2 (78-90): Julio-Diciembre, 2022 | DOI: 10.33936/isrtic.v6i2.4399

se indica en la figura 7. Este software permite la conexión con el hardware de Arduino para cargar los programas y comunicarse con ellos. Ocuparemos la versión 1.8.16 del ARDUINO IDE.

Arduino utiliza para escribir la programación lo que denomina "sketch"(programa), los programas son escritos en el editor de texto.

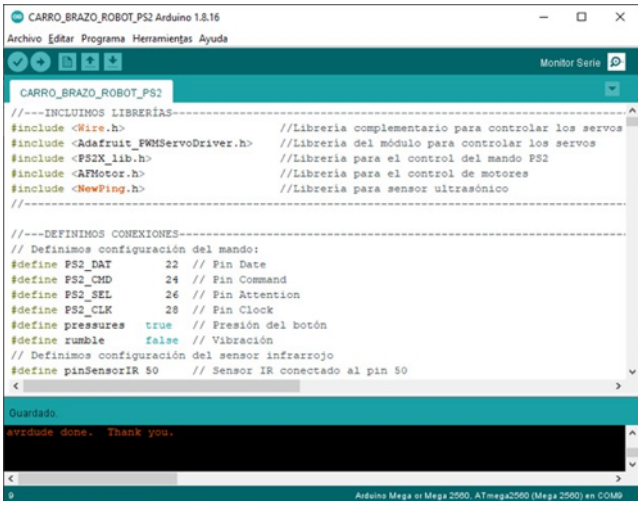


Figura 7. Arduino IDE v1.8.16

5.1Programación de la placa Arduino Mega 2560

El programa que se ha desarrollado y almacenado en el Arduino Mega 2560 realiza las siguientes tareas:

Poner en movimiento el vehículo, controlar el avance, retroceso, giros a izquierda, giros a la derecha y detenerse, este control es de forma manual por medio del mando PS2.

Controlar el movimiento de las 4 partes del brazo robótico, las cuales son la base, hombro, codo y pinza, este control es de forma manual por medio del mando PS2.

Detectar distancia por medio del sensor ultrasónico, para así decidir como posicionarse respecto al objeto, en la opción de recolección de objetos automática.

Detectar si hay o no un objeto al frente del vehículo, en el ángulo correcto para el desplazamiento del brazo robótico, por medio del sensor infrarrojo.

Mueve el brazo robótico de forma automática cuando se requiere tomar un objeto y ubicarlo en el contenedor.

Controla la velocidad de los motores, para así ir más despacio cuando se posicione el carro en modo automático o ir más rápido cuando se use el modo manual.

Convertir los datos recibidos por los servomotores para tratarlos como ángulos y poder mover el brazo robótico en las posiciones requeridas.

Realiza la conexión y configuración con el receptor de PS2, para posteriormente poder conectar un mando controlador y enviar órdenes de forma remota.

6. Implementación del Software y algoritmo del prototipo

6.1 Descripción del código desarrollado en Arduino IDE

En este apartado se detallará el código desarrollado para el funcionamiento del vehículo recolector de objetos.

Librerías: Para el correcto funcionamiento de módulos y sensores se requiere la instalación de las si- guientes librerías.

include <Wire.h>//Librería complementario para controlar los servos

include <AdafruitPWMServoDriver.h>//Librería del módulo para controlar los servos include <PS2Xlib.h>//Librería para el control del mando PS2

include <AFMotor.h>//Librería para el control de motores include <NewPing.h>//Librería para sensor ultrasónico

Definiciones: Definimos en qué pines se encuentran configurados los componentes, tales como el re- ceptor PS2, sensor infrarrojo, sensor ultrasónico, también se define alguna configuración de estos.

// Definimos configuración del mando:

define PS2-DAT 22 // Pin Date define PS2-CMD 24 // Pin Command define PS2-SEL 26 // Pin Attention define PS2-CLK 28 // Pin Clock

define pressures true // Presión del botón define rumble false // Vibración

// Definimos configuración del sensor infrarrojo

define pinSensorIR 50 // Sensor IR conectado al pin 50

// Definimos configuración del sensor ultrasónico:

define TRIGGER-PIN 32 // Arduino pin tied to trigger pin on the ultrasonic sensor. define ECHO-PIN 34 // Arduino pin tied to echo pin on the ultrasonic sensor.

define MAX-DISTANCE 200 // Maximum distance we want to ping for (in centimeters).

Instancia de objetos: Creamos los objetos necesarios a utilizar, como el control PS2, los servos, cada motor DC y el sensor ultrasónico.

// Instanciamos un objeto de la clase PS2X para el control PS2 PS2X ps2x;

// Instanciamos un objeto servos, para el control de los servomotores

Adafruit-PWMServoDriver servos = Adafruit-PWMServoDriver();

// Inicializar motors pin

AF-DCMotor Motor1(1, MOTOR12-1KHZ); AF-DCMotor Motor2(2, MOTOR12-1KHZ); AF-DCMotor Motor3(3, MOTOR34-1KHZ); AF-DCMotor Motor4(4, MOTOR34-

1KHZ);

// Instanciamos objeto para sensor ultrasónico

NewPing sonar(TRIGGER-PIN, ECHO-PIN, MAX-DISTANCE);

Declaración de variables: Se declaran algunas variables a usar en el programa.

// Variables para el control de velocidad de los motores:

int velocidad = 200; // Definimos velocidad de las ruedas rango de 0-255

int velocidad-giro = 230; // Definimos velocidad de las ruedas rango de 0-255

// Variables para definir el rango de trabajo de los servos (duty) y controlarlos en grados: int pos0 = 145;

int pos180 = 472;

// Variables para configurar mando PS2, error validará si hay conexión o no, y vibrate si deseamos vibración:

int error = 0; byte vibrate = 0;

// Variables para definir las posiciones predeterminadas de los servos y salto int angulo1 = 106; // Empezamos en el centro base

int angulo2 = 90; // Empezamos en el centro hombro int angulo3 = 102; // Empezamos en el centro codo int angulo4 = 136; // Empezamos en 136 grados pinza

int salto = 4; // Los servos se moverán de 4 en 4 grados

// Variables para identificar las partes del brazo robótico respecto a sus servos const int base = 0; // Base, servo 0

const int hombro = 1; // Hombro, servo 1 const int codo = 2; // Codo, servo 2 const int pinza = 3; // Pinza, servo 3

// Variables para controlar las pausas y tiempo para realizar ciertas tareas, similar a delay int periodo = 50; // Controlamos a qué velocidad se moverá el brazo en recolección unsigned long TiempoAhora = 0; // Esta variable almacenará tiempo

// Variables para corregir el efecto rebote de los botones/ pulsadores del mando PS2 byte press-r1 = 0;

// Variable para saber si se ha recogido un objeto o no bool estadoRecoger = 0;

Setup: Ahora, en el setup se realiza la configuración de algunos pines, se inicia el puesto serial a una velocidad de 5700 baudios para que el receptor PS2 funcione correctamente y configuramos el mismo.

void setup()

pinMode(pinSensorIR, INPUT); //Indicamos que el pin del sensor IR será de ENTRADA Serial.begin(57600); //Indicamos que la comunicación será a una velocidad de 57600 para que el mando PS2 funcione correctamente

delay(300); //Delay adicional para dar tiempo al módulo ps2 inalámbrico antes de configurarlo. error = ps2x.config-gamepad(PS2-CLK, PS2-CMD, PS2-SEL, PS2-DAT, pressures, rumble);

if (error == 0) else

Serial.println(.Error, mando no encontrado"); servos.begin(); // Inicializamos los servos

servos.setPWMFreq(50); //Definimos que los servos trabajarán a una frecuencia de 50Hz

Loop: Lo siguientes es el Loop(), donde se desarrolla la lógica de todo el funcionamiento del vehículo recolector de objetos, al ser extenso, solo se describirá la función que realiza el proceso de recolecta automática:.

void comenzarAutomatico(): Esta función es la que desempeña todo el proceso de recolecta automática, del brazo robótico en conjunto con los movimientos del vehículo y el uso de sensores para posicionarse correctamente. La descripción del algoritmo, una vez llamada la función es la siguiente:

1. El estado del botón R1, pasa a apagado, esto controla que la pulsación del botón haya sido solo una, para no llamar a la función más de una vez por equivocación al mismo tiempo.

2. El estado de la variable estadoRecoger pasa a falso, por lo que recién se llama a la función y aún no se ha recolectado ningún objeto.

3. Mientras estadoRecoger ==0, es decir mientras no se haya recolectado ningún objeto, se realizan las siguientes instrucciones:

4. Obtenemos la distancia en centímetros, proporcionada por el sensor ultrasónico (distancia), leemos también el estado del sensor infrarrojo (estadoSensorIR).

5. Si la distancia de un objeto está en el rango de 9 cm a 10 cm, entonces:

a) Si el sensor infrarrojo ha detectado un objeto, entonces detén el vehículo, recoge el ob- jeto por medio de la función recogerObjeto(), y actualiza el estado estadoRecoger ==1, indicando que un objeto ya ha sido recogido.

b) Si el sensor infrarrojo no ha detectado ningún objeto,

INFORMÁTICA Y SISTEMAS
Revista de Tecnologías de la Información y las Comunicaciones

entonces, la distancia es correcta pero el objeto no está bien posicionado al frente del sensor IR, por lo tanto, el vehículo girará de izquierda a derecha hasta que el objeto sea reconocido por el sensor IR.

6. Si la distancia del objeto reconocido está entre 2cm a 8cm, entonces:

a) Si el sensor IR ha detectado un objeto, reduce la velocidad de los motores DC, y retrocede el vehículo hasta que la distancia esté en un rango de 9cm a 10cm y se detienen los motores.

b) Si el sensor IR no detecta ningún objeto, el vehículo procede a girar de izquierda a derecha con el fin de alinear el carro hasta que el sensor IR detecte al objeto que se encuentra cerca.

7. Si la distancia del objeto reconocido está en un rango de 11 a 22 cm, entonces:

a) Reduce la velocidad de los motores DC, y avanza el vehículo hasta que la distancia detectada sea menor que 11 cm, luego se detienen los motores.

8. Si la distancia es 0, significa que un objeto está muy cerca o muy lejos, o no es detectado, por lo tanto:

a) Si el sensor infrarrojo si detecta un objeto cerca, entonces gira el vehículo de izquierda a derecha hasta que el sensor ultrasónico detecte una distancia de hasta 22cm y detén el vehículo.

b) Si el sensor infrarrojo no detecta objetos, entonces significa que no hay nada alrededor, por lo tanto, debo hacer un giro de izquierda a derecha para asegurarse de que realmente no exista nada a una distancia de 22 cm, luego, si la distancia sigue siendo cero entonces mueve el vehículo de izquierda a derecha hasta que la distancia sea distinta de cero.

9. Si la distancia es mayor o igual a 23 cm, entonces:

a) Si el sensor infrarrojo detecta un objeto al frente, gira el vehículo hasta que la distancia sea menor a 23 cm.

b) Si el sensor IR no detecta ningún objeto, revisa alrededor 2 veces por medio de giros de izquierda a derecha si es que hay un objeto, y detente si el sensor IR encuentra algo. Luego de hacer esta revisión, si la distancia obtenida por el sensor ultrasónico sigue siendo mayor a 22 cm, entonces avanza el vehículo aproximadamente unos 20 centímetros y detén los motores.

10. Una vez que se rompa este ciclo cuando un objeto finalmente haya sido recogido, configura la velocidad de recorrido y de giro del vehículo a la predeterminada.

6.2 Programación de los botones del mando PS2

Los siguientes botones fueron programados con las funcionalidades necesarias para operar el vehículo recolector de objetos, ver Figura 8.

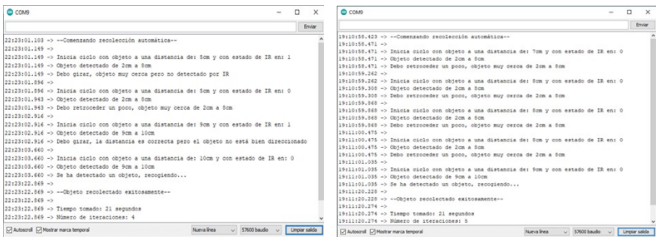
Figura 8. Disposición de acciones a botones del mando PS2.

7. Pruebas técnicas de funcionamiento

Para la obtención de resultados se realizaron pruebas técnicas del funcionamiento. Con el prototipo se llevó a cabo la evaluación de tres escenarios: cuando el objeto se encuentra muy cerca, muy lejos y a una distancia adecuada, respecto a la posición del vehículo. Para cada escenario se realizaron un total de 60 pruebas, en el entorno de prueba, el objeto se ubicó según el caso a evaluar, 30 de los test son con el sensor infrarrojo detectando un objeto y 30 con el sensor infrarrojo sin detectar nada, las baterías se encontraban en una carga media, el brazo robótico estaba en posición de descanso y el objeto a recoger tiene 6 cm de ancho en su base y 2 cm de ancho en la parte superior, con un alto de 7.5 cm.

7.1. Escenario 1: 1 objetos que están muy cerca (2 cm a 8 cm)

Los datos fueron monitoreados a través del Monitor Serie del Arduino IDE y fueron recopilados en una hoja de cálculo, como se ve a continuación, ver Figura 9 (los datos completos en el anexo 1):



a. Ver video de la prueba en el anexo 4

b. Ver video de la prueba en el anexo 5

Figura 9. Visualización del Monitor Serie - Escenario 1

De las 60 pruebas, 50 fueron exitosas y 10 fallidas, los datos recopilados fueron los siguientes:

e-ISNN 2550-6730 | Vol. 6, Núm. 2 (78-90): Julio-Diciembre, 2022 | DOI: 10.33936/isrtic.v6i2.4399

Tabla 8. Datos del número de iteraciones del algoritmo

Valor mínimo de iteraciones	Valor máximo de iteraciones	Valor promedio de iteraciones	Valor más repetitivo de iteraciones
2	6	3.22	2

Tabla 9. Datos del tiempo que toma recoger objetos

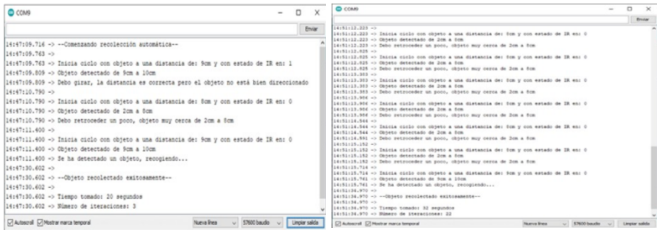
Valor mínimo de tiempo	Valor máximo de tiempo	Valor promedio de tiempo	Valor más repetitivo de tiempo
19s	24s	20.72s	20s

7.1.1. Observaciones y conclusiones:

Con los datos y observaciones se concluye que, si el objeto se encuentra de 2 cm a 8 cm, existe una probabilidad del 83 % de que el objeto sea recogido con éxito, además de tener un promedio de aproximadamente 3 iteraciones y demorar en promedio 21 segundos en recogerlo. El 17 % de probabilidad restante es de que el vehículo no logre posicionarse bien y retroceda mucho además de que exista algún problema con la pinza del brazo robótico. Cabe resaltar que en los casos de éxito el valor mínimo de pasos para recoger el objeto es de 2 iteraciones y el máximo de 6, y en cuestión de tiempo el mínimo fue de 19s y máximo de 24s.

7.2. Escenario 2: Recogiendo objetos que están en la distancia adecuada (9 cm a 10 cm)

Los datos fueron monitoreados a través del Monitor Serie del Arduino IDE y fueron recopilados en una hoja de cálculo, como se ve a continuación, ver Figura 10 (los datos completos en el anexo 2):



a. Ver video de la prueba en el anexo 6

b. Ver video de la prueba en el anexo 7

Figura 10. Visualización del Monitor Serie - Escenario 2

De las 60 pruebas, 55 fueron exitosas y 5 fallidas, los datos recopilados fueron los siguientes:

Tabla 10. Datos del número de iteraciones del algoritmo

Valor mínimo de iteraciones	Valor máximo de iteraciones	Valor promedio de iteraciones	Valor más repetitivo de iteraciones
1	26	2.47	1

Tabla 11. Datos del tiempo que toma recoger objetos

Valor mínimo de tiempo	Valor máximo de tiempo	Valor promedio de tiempo	Valor más repetitivo de tiempo
19s	34s	20.02s	19s

7.2.1. Observaciones y conclusiones:

Con los datos y observaciones se concluye que, si el objeto se encuentra de 9 cm a 10 cm, existe una probabilidad del 92 % de que el objeto sea recogido con éxito, además de tener un promedio de aproximadamente 2 iteraciones y demorar en promedio 20 segundos en recogerlo. El 8 % de probabilidad restante es de que el vehículo no logre posicionarse bien, falle en recolectar o se quede girando indefinidamente. Cabe resaltar que en los casos de éxito el valor mínimo de pasos para recoger el objeto es de 2 iteraciones y el máximo de 6, y en cuestión de tiempo el mínimo fue de 19s y máximo de 24s.

7.3. Escenario 3: Recogiendo objetos que están lejos (mayor a 11 cm)

Los datos fueron monitoreados a través del Monitor Serie del Arduino IDE y fueron recopilados en una hoja de cálculo, como se ve a continuación (los datos completos en el anexo 3):

De las 60 pruebas, 36 fueron exitosas y 24 fallidas, los datos recopilados fueron los siguientes:

7.3.1. Observaciones y conclusiones:

Con los datos y observaciones se concluye que, si el objeto se encuentra de 11 cm a 22 cm, no se tiene dificultad para recoger, pero si está aún más lejos, se dificulta. Considerando todas las pruebas, existe una probabilidad del 60 % de que el objeto sea recogido con éxito, además de tener un promedio de aproximadamente 8 iteraciones y demorar un máximo de 58

e-ISNN 2550-6730 | **Vol. 6, Núm. 2 (78-90): Julio-Diciembre, 2022** | **DOI: 10.33936/isrtic.v6i2.4399**

INFORMÁTICA
Facultad de Ciencias Informáticas
Universidad Técnica de Manabí
Av. Urbina y Che Guevara, Portoviejo, Ecuador
revista.iys@utm.edu.ec

a. Ver video de la prueba en el anexo 8

b. Ver video de la prueba en el anexo 9

Figura 11. Visualización del Monitor Serie - Escenario 3

Tabla 12. Datos del número de iteraciones del algoritmo

Valor mínimo de tiempo	Valor máximo de tiempo	Valor promedio de tiempo	Valor más repetitivo de tiempo
20s	58s	38.18s	20s

Tabla 13. Datos del tiempo que toma recoger objetos

Valor mínimo de iteraciones	Valor máximo de iteraciones	Valor promedio de iteraciones	Valor más repetitivo de itera ciones
2	27	8.45	4

segundos en recogerlo. El 40 % de probabilidad restante es de que el vehículo no logre encontrar objetos, ya que hace cambios de trayectoria por las lecturas de distancia de otros elementos que tenga cerca, y se puede quedar girando en un solo lugar por algún tiempo. Como vemos, es muy probable que el objeto no sea recogido, no hay buen desempeño.

8. Resultados y Discusión

Los resultados obtenidos de este desarrollo e implementación de vehículo autónomo recolector de objetos se respaldan con las pruebas técnicas ejecutadas. El prototipo, tal y como fue diseñado e implementado en software y hardware, logra tener la autonomía para desempeñar un trabajo diario, pero siempre y cuando no se lo use con constancia, en cuanto a la funcionalidad de recoger objetos de forma manual, va muy bien, el control de forma remota no es complicado y tiene un alcance de 12 metros. La funcionalidad de recolecta automática, tiene buen rendimiento con objetos que se encuentran cerca y comienza a tener menos probabilidad de éxito cuando están muy lejos. Analizando este resultado, conviene usar el vehículo manualmente hasta estar cerca de un objeto y luego usar la opción de recolecta automática. Cabe resaltar que se encontraron las siguientes limitaciones en el prototipo:

Al vehículo recolector de objetos se le dificulta recoger elementos que se encuentren muy lejos de él, a aproximadamente a partir de 90 cm.

Debido a la potencia de los servomotores, tiene limitado recoger objetos pesados y muy grandes, por lo que, un objeto con estas dimensiones puede funcionar bien: 6 cm de ancho en su base y 2 cm de ancho en la parte superior, con un alto de 7.5 cm.

Objetos que estén atrás del vehículo, no los recogerá, al menos que se gire el carro con el control manual.

Dicho esto, se podría modificar y actualizar los componentes que integran este prototipo, tales como los servomotores, se podrían mejorarlos para que el brazo pueda recoger objetos más pesados, así mismo adicionar más sensores de distancia e infrarrojo para mejorar el algoritmo de recolecta automática, y el prototipo eleve la probabilidad exitosa de recolecta en los objetos que se encuentren más lejos. Con estas agregaciones, también se debería aumentar el amperaje de las baterías, ya que los servomotores, consumen bastante corriente. De igual manera se pueden expandir funcionalidades como tener una cámara y así realizar tareas de exploración, transporte y recolección.

9. Conclusiones

El prototipo del vehículo recolector de objetos, fue implementado de tal forma que el hardware desempeñe bien su trabajo, gracias a su alimentación adecuada y las conexiones realizadas, pero cabe indicar que el prototipo puede ser mejorado con la incorporación de más sensores infrarrojos, para que pueda detectar de mejor forma los objetos y orientar el brazo robótico a la dirección donde el sensor esté detectando algo.

En cuanto al software, programado en el Arduino IDE, logra ser totalmente funcional para el control del vehículo y brazo robótico de forma manual remota por medio de un mando PS2 Wireless, y el algoritmo desarrollado para la recolecta automática de objetos, es buena a pesar de tener solo un sensor IR y un sensor ultrasónico, y funciona correctamente cuando el objeto se encuentra no mayor a 30cm, cuando la distancia es superior, el vehículo tiende a bajar su probabilidad de recoger con éxito un objeto, además de demorar más y tener más iteraciones (realizar más pasos).

Gracias al uso del Arduino Mega 2560, se pudieron realizar conexiones de módulos sin problemas de falta de pines, ya que también tiene montado un Motor Shield L293D. En cuanto a la autonomía energética, puede ser usado constantemente por casi 3 horas, a buena velocidad considerando el peso del vehículo de aproximadamente 0.6 Kg.

Por otra parte, esta construcción del prototipo, que tiene como particularidad ser versátil en su uso, por poder ser controlado remotamente de manera manual o automática, contribuirá como base para la creación de equipos autónomos recolectores de objetos, para distintas áreas de aplicación. De aquí pueden

e-ISNN 2550-6730 | Vol. 6, Núm. 2 (78-90): Julio-Diciembre, 2022 | DOI: 10.33936/isrtic.v6i2.4399

partir soluciones para mejorar o desarrollar dispositivos que complementen una necesidad específica en el ámbito científico, investigativo o industrial.El procedimiento presentado para su implementación sirve como referencia para futuros proyectos que puedan incluir mejoras a este prototipo, para que tenga mayor eficiencia en encontrar objetos en distintos tipos de ambiente.

Anexos

A.1. Anexo 1: Consolidado de pruebas – Recogiendo objetos que están muy cerca.

N° Test	Valor distancia inicial (cm)	Valor de IR inicial (0=Detecta objeto; 1=No detecta objeto)	Número de Iteraciones	Tiempo (segundos)	¿Prueba exitosa?	Observación
1	2	0	4	21	Si	Sin observación
2	2	0	3	21	No	Pinza sotó objeto
3	2	0	4	21	Si	Sin observación
4	2	0	4	21	No	No se posiciono bien
5	2	0	3	21	Si	Sin observación
6	3	0	2	20	Si	Sin observación
7	3	0	3	20	Si	Sin observación
8	3	0	2	20	Si	Sin observación
9	3	0	2	20	Si	Sin observación
10	3	0	2	20	Si	Sin observación
11	4	0	3	20	No	Pinza sotó objeto
12	4	0	2	20	Si	Sin observación
13	4	0	3	20	No	Pinza sotó objeto
14	4	0	3	20	Si	Sin observación
15	5	0	2	20	Si	Sin observación
16	5	0	2	20	Si	Sin observación
17	5	0	2	20	Si	Sin observación
18	5	0	2	20	No	Pinza sotó objeto
19	5	0	4	21	Si	Sin observación
20	6	0	2	20	Si	Sin observación
21	6	0	2	20	Si	Sin observación
22	6	0	2	20	Si	Sin observación
23	7	0	5	21	Si	Sin observación
24	7	0	2	20	Si	Sin observación
25	7	0	2	20	Si	Sin observación
26	7	0	2	20	Si	Sin observación
27	8	0	2	19	Si	Sin observación
28	8	0	2	19	Si	Sin observación
29	8	0	2	19	Si	Sin observación
30	8	0	2	19	Si	Sin observación
31	2	1	6	24	Si	Sin observación
32	2	1	5	22	Si	Sin observación
33	2	1	No	No	No	Se quedó girando
34	2	1	6	23	Si	Sin observación
35	3	1	4	21	Si	Sin observación
36	3	1	5	21	No	Tumbó el objeto
37	3	1	5	22	Si	Sin observación
38	3	1	5	23	Si	Sin observación
39	3	1	4	22	Si	Sin observación
40	3	1	5	24	Si	Sin observación
41	4	1	5	23	Si	Sin observación
42	4	1	4	21	Si	Sin observación
43	4	1	6	23	Si	Sin observación
44	4	1	3	21	Si	Sin observación
45	5	1	4	21	Si	Sin observación
46	5	1	3	21	Si	Sin observación
47	5	1	4	21	Si	Sin observación
48	5	1	3	21	Si	Sin observación
49	5	1	4	21	Si	Sin observación
50	6	1	4	21	Si	Sin observación
51	6	1	3	20	Si	Sin observación
52	6	1	4	21	Si	Sin observación
53	7	1	3	20	Si	Sin observación
54	7	1	3	20	Si	Sin observación
55	7	1	3	20	Si	Sin observación
56	7	1	3	20	Si	Sin observación
57	8	1	2	20	Si	Sin observación
58	8	1	No	No	No	Se quedó girando
59	8	1	3	21	Si	Sin observación
60	8	1	3	20	No	Pinza sotó objeto

Figura 12. Consolidado de pruebas – Recogiendo objetos que están muy cerca

A.2. Anexo 2: Consolidado de pruebas – Recogiendo objetos que están a una distancia ideal.

N° Test	Valor distancia inicial (cm)	Valor de IR inicial (0=Detecta objeto; 1=No detecta objeto)	Número de Iteraciones	Tiempo (segundos)	¿Prueba exitosa?	Observación
1	9	0	1	19	Si	Sin observación
2	9	0	1	19	Si	Sin observación
3	9	0	1	19	Si	Sin observación
4	9	0	1	19	Si	Sin observación
5	9	0	1	19	Si	Sin observación
6	9	0	1	19	Si	Sin observación
7	9	0	1	19	Si	Sin observación
8	9	0	1	19	Si	Sin observación
9	9	0	1	19	Si	Sin observación
10	9	0	1	19	Si	Sin observación
11	9	0	1	19	Si	Sin observación
12	9	0	1	19	Si	Sin observación
13	9	0	1	19	Si	Sin observación
14	9	0	1	19	Si	Sin observación
15	9	0	1	19	Si	Sin observación
16	10	0	1	19	No	Pinza sotó objeto
17	10	0	1	19	Si	Sin observación
18	10	0	1	19	Si	Sin observación
19	10	0	1	19	Si	Sin observación
20	10	0	1	19	Si	Sin observación
21	10	0	1	19	Si	Sin observación
22	10	0	1	19	Si	Sin observación
23	10	0	1	19	Si	Sin observación
24	10	0	1	19	Si	Sin observación
25	10	0	1	19	Si	Sin observación
26	10	0	1	19	Si	Sin observación
27	10	0	1	19	Si	Sin observación
28	10	0	1	19	Si	Sin observación
29	10	0	1	19	Si	Sin observación
30	10	0	1	19	Si	Sin observación
31	9	1	2	20	Si	Sin observación
32	9	1	3	20	Si	Sin observación
33	9	1	2	19	Si	Sin observación
34	9	1	3	20	Si	Sin observación
35	9	1	22	32	Si	No retrocedió rápido
36	9	1	5	22	Si	Sin observación
37	9	1	3	20	Si	Sin observación
38	9	1	No	No	No	Se quedó girando
39	9	1	2	20	Si	Sin observación
40	9	1	No	No	No	Se quedó girando
41	9	1	3	20	Si	Sin observación
42	9	1	2	20	Si	Sin observación
43	9	1	2	20	Si	Sin observación
44	9	1	2	20	Si	Sin observación
45	9	1	2	20	Si	Sin observación
46	10	1	No	No	No	Se quedó girando
47	10	1	2	20	Si	Sin observación
48	10	1	3	20	Si	Sin observación
49	10	1	2	20	No	Pinza sotó objeto
50	10	1	2	20	Si	Sin observación
51	10	1	2	20	Si	Sin observación
52	10	1	2	20	Si	Sin observación
53	10	1	2	20	Si	Sin observación
54	10	1	4	21	Si	Sin observación
55	10	1	3	21	Si	Sin observación
56	10	1	2	20	Si	Sin observación
57	10	1	3	21	Si	Sin observación
58	10	1	26	34	Si	Se demoró en retroceder
59	10	1	2	20	Si	Sin observación
60	10	1	2	20	Si	Sin observación

Figura 13. Consolidado de pruebas – Recogiendo objetos que están a una distancia ideal

A.3. Anexo 3: Consolidado de pruebas – Recogiendo objetos que están muy lejos.

N° Test	Valor distancia inicial (cm)	Valor de IR inicial (0=Detecta objeto; 1=No detecta objeto)	Número de Iteraciones	Tiempo (segundos)	¿Prueba exitosa?	Observación
1	54	1	7	33	Si	Sin Observación
2	130	1	9	31	No	Pinza tumbó objeto
3	37	1	6	25	No	No encontró objeto
4	12	1	4	21	Si	Sin Observación
5	13	1	2	20	Si	Sin Observación
6	14	1	3	20	Si	Sin Observación
7	15	1	3	20	Si	Sin Observación
8	16	1	4	20	Si	Sin Observación
9	17	1	4	20	Si	Sin Observación
10	18	1	3	21	Si	Sin Observación
11	19	1	3	21	Si	Sin Observación
12	20	1	3	20	Si	Sin Observación
13	21	1	5	22	Si	Sin Observación
14	22	1	5	22	Si	Sin Observación
15	36	1	No	No	No	Se quedó girando
16	40	1	53	51	Si	Sin Observación
17	49	1	55	52	Si	Sin Observación
18	45	1	57	50	Si	Sin Observación
19	67	1	No	No	No	Se quedó girando
20	78	1	No	No	No	Se quedó girando
21	120	1	No	No	No	Se quedó girando
22	115	1	No	No	No	Se quedó girando
23	37	1	No	No	No	Se quedó girando
24	37	1	3	42	Si	Sin Observación
25	32	1	6	38	Si	Sin Observación
26	47	1	12	47	Si	Sin Observación
27	60	1	No	No	No	Se quedó girando
28	57	1	38	52	Si	Sin Observación
29	78	1	No	No	No	Se quedó girando
30	94	1	No	No	No	Se quedó girando
31	124	1	No	No	No	Se quedó girando
32	143	1	No	No	No	Se quedó girando
33	28	1	30	33	Si	Sin Observación
34	14	1	3	20	Si	Sin Observación
35	15	1	3	20	Si	Sin Observación
36	16	1	4	21	Si	Sin Observación
37	96	1	No	No	No	Se quedó girando
38	38	1	4	21	Si	Sin Observación
39	39	1	4	21	Si	Sin Observación
40	40	1	25	56	Si	Sin Observación
41	23	1	3	20	Si	Sin Observación
42	42	1	8	28	Si	Sin Observación
43	57	1	20	58	Si	Sin Observación
44	78	1	No	No	No	Se quedó girando
45	97	1	20	54	Si	Sin Observación
46	47	1	33	51	Si	Sin Observación
47	130	1	No	No	No	Se quedó girando
48	93	1	20	50	Si	Sin Observación
49	88	1	27	46	Si	Sin Observación
50	35	1	3	20	Si	Sin Observación
51	36	1	4	21	Si	Sin Observación
52	49	1	No	No	No	Se quedó girando
53	45	1	No	No	No	Se quedó girando
54	67	1	No	No	No	Se quedó girando
55	78	1	No	No	No	Se quedó girando
56	94	1	38	52	Si	Sin Observación
57	135	1	No	No	No	Se quedó girando
58	93	1	No	No	No	Se quedó girando
59	32	1	No	No	No	Se quedó girando
60	68	1	No	No	No	Se quedó girando

Figura 14. Consolidado de pruebas – Recogiendo objetos que están muy lejos.

A.4. Anexo 4: Video 1 - Test objeto a 5cm y sensor IR en 1

https://youtu.be/oVxLaNBtGaY

A.5. Anexo 5: Video 2 - Test objeto a 7cm y sensor IR en 0

https://youtu.be/QH8kORIpV80

A.6. Anexo 6: Video 3 - Test objeto a 9cm y sensor IR en 1

https://youtu.be/Z3DTINCNd_g

A.7. Anexo 7: Video 4 - Test objeto a 10cm y sensor IR en 1

https://youtu.be/t7nyGARxkYc

A.8. Anexo 8: Video 5 - Test objeto a 54cm y sensor IR en 1

https://youtu.be/rN0-gZszOOg

A.9. Anexo 9: Video 6 - Test objeto a 110cm y sensor IR en 1

https://youtu.be/axZ4KRDxois

Referencias

Campo Jorge, D., Sastre Vásquez, M., Gamas Faulkner, I., & Samaniego González, E. (2017). Diseño y construcción de un prototipo robótico de un automóvil para transporte

e-ISNN 2550-6730 | **Vol. 6, Núm. 2 (78-90): Julio-Diciembre, 2022** | **DOI: 10.33936/isrtic.v6i2.4399**

INFORMÁTICA Y SISTEMAS
Revista de Tecnologías de la Informática y las Comunicaciones
Facultad de Ciencias Informáticas
Universidad Técnica de Manabí
Av. Urbina y Che Guevara, Portoviejo, Ecuador
revistas@umanabi.edu.ec

personal y económico con la tecnología Arduino. *Revista De Iniciación Científica*, *3*(1), 110–129. https://revistas.utp.ac.pa/index.php/ric/article/view/1705

Casquete Quiñonez, F. X., & Moreira Maridueña, P. E. (2018). *Diseño de vehículo de exploración para el tendido de cables UTP basados en tecnología arduino y controlado por un dispositivo celular o tablet con sistemas operativo android* [Tesis de Grado, Universidad de Guayaquil, Facultad de Ciencias Matemáticas y Físicas]. http://repositorio.ug.edu.ec/handle/redug/26723

Circuit Best (23 de junio de 2021). *Arduino Bluetooth controlled car with Front Back Lights using Arduino UNO, L293D Motor Driver, HC-05.* Recuperado en diciembre de 2021 de https://circuitbest.com/arduino-bluetooth-controlled-car-with-front-back-lights-using-arduino-uno-l293d-motor-driver-hc-05/

Earl, B. (16 de cotubre de 2012). *Adafruit PCA9685 16-Channel Servo Driver. Adafruit Learning System.* Recuperado en diciembre de 2021 de https://learn.adafruit.com/16-channel-pwm-servo-driver

ElecFreaks (2013). *Ultrasonic Ranging Module HC-SR04.* Recuperado en diciembre de 2021 de https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf

Lady Ada (27 de agosto de 2012). *Adafruit Motor Shield. Adafruit Learning System.* Recuperado en noviembre de 2021 de https://learn.adafruit.com/adafruit-motor-shield"

Macho, J. C. (s.f.a). *Controlando el Rover con un mando PS2.* Prometec. Recuperado en diciembre de 2021 de https://www.prometec.net/controlador-ps2/

Macho, J. C. (s.f.b). *¿Cuánto consume Arduino?.* Prometec. Recuperado en diciembre de 2021 de https://www.prometec.net/consumos-arduino/

Ministerio del Ambiente de Perú (2014). *Guía para muestreo de suelos* [Archivo PDF]. https://www.minam.gob.pe/calidadambiental/wp-content/uploads/sites/22/2013/10/GUIA-PARA-EL-MUESTREO-DE-SUELOS-final.pdf

Molina Cruz, D. A., Cedeño Ferrin, J. A., Marcillo Parrales, K., Marcillo Parrales, A., Ortiz Hernandez, M., Mero Lino, E., & Merchán García, F. (2019). *Módulo con controladores lógicos programables para la enseñanza-aprendizaje de electrónica.* Editorial Área de Innovación y Desarrollo,S.L.

Navarro Cosme, T. (2016). *Desarrollo de un prototipo de vehículo autónomo semi-inteligente basado en Arduino* (Tesis de Grado, Universitat Politècnica de València). https://riunet.upv.es/handle/10251/71526

Peña Uriarte, D. (2019). Diseño e implementación de coche teledirigido mediante Arduino [Tesis de Grado, Universidad del País Vasco]. https://addi.ehu.es/bitstream/handle/10810/32243/TFG_PeñaUriarte.pdf

Semiconducter Componentes Industries (s.f.). *LM2596 3.0 A, Step-Down Switching Regulator.* Recuperdado en diciembre de 2021 de https://www.onsemi.com/products/power-management/dc-dc-power-conversion/converters/LM2596

Tapia Ayala, C. H., & Manzano Yupa, H. M. (2013). *Evaluación de la plataforma Arduino e implementación de un sistema de control de posición horizontal* [Tesis de grado, Universidad Politécnica Salesiana]. https://dspace.ups.edu.ec/bitstream/123456789/5522/1/UPS-GT000511.pdf

Torrico Santos, V. (2016). *Diseño e implementación de un vehículo a escala controlado remotamente* [Tesis de Grado, Universidad Carlos III de Madrid]. https://e-archivo.uc3m.es/handle/10016/26653

e-ISNN 2550-6730 | **Vol. 6, Núm. 2 (78-90): Julio-Diciembre, 2022** | **DOI: 10.33936/isrtic.v6i2.4399**