**-Desarrollo del proyecto-**

**OBJETIVO GENERAL:**

* Desarrollar un robot que implemente la evasión de obstáculos

**OBJETIVOS ESPECÍFICOS:**

* Recabar información acerca de los dispositivos y componentes a utilizar en el desarrollo del proyecto
* Buscar información de las características electrónicas para conocer su funcionamiento y sus límites operativos
* Elección de la plataforma de hardware más apropiada y accesible
* Diseño y desarrollo del prototipo.
* Desarrollo del algoritmo que le permita detectar los objetos en su camino al avanzar y que dispare
* La acción de retroceder y evadir obstáculos.
* Realizar pruebas de funcionamiento

**ESTADO DEL ARTE**

De acuerdo con un artículo publicado en la revista de la Universidad Nacional Autónoma de México por medio de Facultad de Ingeniería se presenta una síntesis de los métodos de anticolisión que se utilizan para evadir obstáculos con robots móviles, en particular, los métodos de planeación de rutas y métodos basados en campos potenciales artificiales.

Dicha síntesis propone que existen dos tipos de comportamiento: el movimiento guiado por una ruta planificada y la reacción para la evasión del objeto.

El que un robot móvil cuente con la capacidad de planear trayectorias óptimas libres de colisión y reaccionar ante la presencia de objetos inesperados en el camino. La aplicación de estas dos técnicas en un sólo robot da como resultado la creación de sistemas híbridos, los cuales usan las dos técnicas, de planeación de rutas y de evasión de obstáculos, para superar los problemas que se presentan en el transcurso de la conquista de la meta.

**CARRO INTELIGENTE**

**-ÁREAS DE LA I.A. APLICADAS AL PROYECTO-**

El presente trabajo describirá como resolvimos un problema para personas que tengan alguna discapacidad visual mediante un carro inteligente. Lo cual consiste en un carro en el que la persona que contenga problemas visuales trate de conducir un carro para encontrar el destino al cual desea llegar, para ello debe ir pasando por calles y avenidas las cuales tienen objetivos (Carros Personas) y este carro deberá de advertirle al conductor mediante un alarma o que el mismo carro tenga el control del carro y escoger una opción viable para andar en los lugares del destino.

**INTRODUCCION**

En los últimos años la inteligencia artificial ha evolucionado en gran medida hasta el punto que se han construido maquinas inteligentes que inclusive tienen la capacidad de aprender. Abarca un sinnúmero de campos y conceptos entre los cuales se encuentra el concepto de agente, se dice que estos constituyen el próximo avance más significativo en el desarrollo de sistemas y pueden ser considerados como la nueva revolución en el software. La característica quizá más importante de un agente inteligente es la autonomía, quizá la mayoría de personas se preguntarán si maquinas con inteligencia artificial llegarán a superar la mente humana, quizá estén dotadas de mucha inteligencia pero pensamos que será muy difícil superar la inteligencia de los humanos.

**DESCRIPCIÓN DEL PROBLEMA**

**Meta:** El conductor con discapacidad visual deberá llegar a su destino final de la manera más segura, escogiendo la dirección más segura y estable, además deberá evitar chocar con los obstáculos que se encuentre en su camino.

**Inicio**: El conductor tendrá un punto de partida cualquiera, tendrá varias rutas para llegar a su destino, y este deberá tener la capacidad para llegar al destino sorteando los obstáculos colocados en su trayecto. El camino debe ser el más viable para garantizar la seguridad.

**Obstáculos:** Los obstáculos serán casas, finales de calles, los automóviles entre otras cosas que las cuales estarán a lo largo de todo el trayecto que recorre el conductor. Estos obstáculos permanecen estáticos.

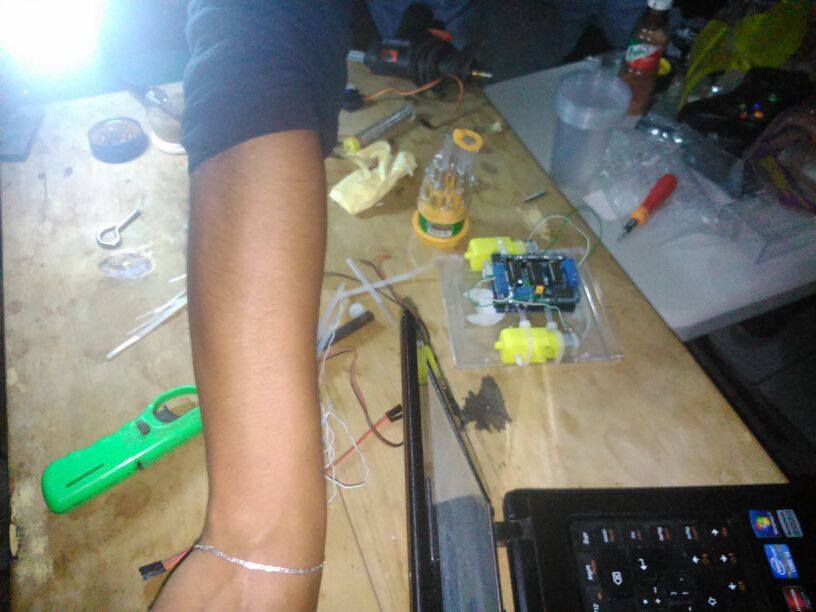
**¿El porqué de un robot evitador de obstáculos?**

Pues en IA un robot es una maquina la cual puede tomar decisiones autónomamente esto enfocado a un robot que es capaz de tomar decisión propia de que dirección tomar cuando evita un obstáculo.

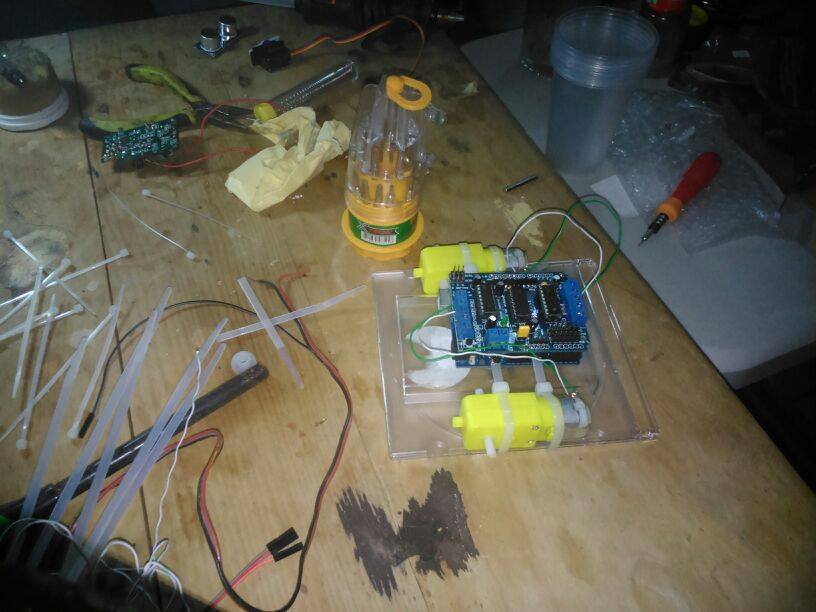
**DESARROLLO**

Este robot está basado en la plataforma Arduino que tiene empotrado un micro controlador Atmega 328 el cual alberga el algoritmo que le da la inteligencia necesaria para que pueda realizar las distintas tarea como en este caso la evasión de obstáculos, se ocuparon otros componentes tales como un controlador de motores o doble puente h, el cual le da a los dos moto reductores el control necesario para que pueda funcionar adecuadamente además se adquirieron otros componentes tales como un servomotor y .

Un sensor ultrasónico el cual tiene la función de escanear el obstáculo regresándole una respuesta emitiendo una salida ultrasónica y receptando la respuesta, como un sonar, el servomotor funge como torreta la cual panea el terreno y va enviando información al robot la cual es procesada por el algoritmo para tomar una decisión si es que llega a presentarse un obstáculo sobre el (todo esto en cuestión de segundos).

**Proceso de desarrollo del armazón del robot:**

Se utilizó como desarrollo de este proyecto el libro construyendo robots para aficionados de editorial MARACOMBO el cual explica paso a paso el proceso de ensamblaje. Se ocupó una cajita de disco como plataforma de soporte para los motores y la placa de control de los motores y el Arduino.

**Etapa final del desarrollo del proyecto:**

Para finalizar el ensamblaje se colocarón unas llantas de modelismo muy utiles en robotica y parte escencial del proyecto, se empotro el sensor ultrasonico encima del servomotor el cual ara la función de torreta y paneo del ambiente a su alrededor.

**FUNCIONAMIENTO**

La parte escencial de nuestro proyecto es el servomotor y el sensor ultrasonico el cual va enviando información al algoritmo como entrada de datos de que es lo que va detectando, una vez que el robot se encuentra con un obstaculo este toma la decisión si moverse a la izquierda o a la derecha o tomar otro rumbo si es que se encuentra encerrado en un laberinto.

Puente H: El puente h es un circuito analogico/Digital basado en pulsos, el cual dependiendo de la tabla de verdad gira a la izquierda o a la derecha, en el caso del proyecto del robot se empotro una shield conocida como adafruit la cual contiene 2 puentes h para una cantidad de 4 motores, de corriente continua o 2 motores a pasos y 2 servmotores. La justificación por la cual se ocupo esta placa es su fácil configuración y adaptación a distintos prototipos o robots.

Arduino: El arduino es la parte escencial del proyecto es donde esta resguardado el microcontrolador ATMEGA 328 y el cual aloja el programa, este programa a us vez tiene el control tanto de entradas como salidas de los distintos pines ocupados en el proyecto.

**Funcionamiento del Algoritmo**

**#include <AFMotor.h>//Librería para la comunicación con nuestros motores**

**#include <Servo.h> //Libreria para la comunicacion con nuestro servomotor**

**#include <NewPing.h>//Libreria para las comunicacion con los pings que se ocuparan**

**#define TRIG\_PIN A4 //Pina A4 pone el trig del sensor ultrasonico sobre la shield que controla los motores**

**#define ECHO\_PIN A5 //Pina A5 pone el echo del sensor ultrasonico sobre la shield que controla los motores**

**#define MAX\_DISTANCE 200 // coloca el valor máximo de reconocimiento del sensor aproximadamente 200 cm.**

**#define MAX\_SPEED 180 //coloca la velocidad de los motores a un intervalo de 180 rpm aunque lo máximo son 256rpm**

**#define MAX\_SPEED\_OFFSET 10 //esto permite que el intervalo de tiempo entre que pare un motor y otro sea de 10milisegundos**

**#define COLL\_DIST 10 // coloca la distancia minima a la cual el robot debe de girar**

**#define TURN\_DIST COLL\_DIST+10 // coloca la distancia minima a la cual el robot debe de parar y tomar una decisión**

**NewPing sonar(TRIG\_PIN, ECHO\_PIN, MAX\_DISTANCE); //coloca los sensores en el metodo sonar de la libreria sonar la cual recibira de parametro los estatus anteriores**

**AF\_DCMotor motor1(1, MOTOR12\_1KHZ); // crea el motor #1, utilizando la salida M1 de la shield controladora, y le da un valor de 1khz a la frecuencia del pwm**

**AF\_DCMotor motor2(4, MOTOR12\_1KHZ); // crea el motor #2, utilizando la salida M4 de la shield controladora, y le da un valor de 1khz a la frecuencia del pwm**

**Servo myservo; // crea un objeto servo de la clase instanciada**

**int pos = 0; // variable posición a continuación se colocan una serie de variables que seran utiliadas en el resto del programa**

**int maxDist = 0;**

**int maxAngle = 0;**

**int maxRight = 0;**

**int maxLeft = 0;**

**int maxFront = 0;**

**int course = 0;**

**int curDist = 0;**

**String motorSet = "";**

**int speedSet = 0;**

**//-------------------------------------------- SETUP LOOP ----------------------------------------------------------------------------**

**void setup() {**

**myservo.attach(9); // colocal el servo sobre el pin 9 del arduino (Servo\_2 del motorshield)**

**myservo.write(90); // le dice al servo la posición de 90 grados**

**delay(2000); // espera alrededor 200 milisegudnos**

**checkPath(); // rutina o método necesario para que pueda empezar a recorrer el camino**

**motorSet = "FORWARD"; // coloca el motor con la variable forward la cual nos sirve para que avanze**

**myservo.write(90); // se asegura de que el servo aun siga panenando**

**moveForward(); // hace que el robot siga moviendose adelante**

**}**

**//------------------------------------------------------------------------------------------------------------------------------------**

**//---------------------------------------------MAIN LOOP ------------------------------------------------------------------------------**

**//el main loop es el ciclo infinito el cual se repetira n mil veces dentro de la ejecución del programa**

**void loop() {**

**checkForward(); // revisa si el robot esta avanzando constantemente hacia adelante, esto ocaciona que el consumo energetico en los motores sea mayor**

**checkPath(); //pone al sensor ultrasonico a escanear de posibles obstaculos**

**}**

**//-------------------------------------------------------------------------------------------------------------------------------------**

**//------------método checkpath para el servomotor.**

**void checkPath() {**

**int curLeft = 0;**

**int curFront = 0;**

**int curRight = 0;**

**int curDist = 0;**

**myservo.write(144); //coloca el servo a 54 grados**

**delay(120); //espera 120 milisegundos para que el servo busque su posición**

**for(pos = 144; pos >= 36; pos-=18) // ciclo importante del programa el cual realiza el paneo desde 144 grados a 36**

**{**

**myservo.write(pos); // le dice al servo que vaya a la posición almacenado en la variables pos**

**delay(90); // espera 90 segundos para obtener la posición**

**checkForward(); // checa si el motor si aun se esta moviendo adelante**

**curDist = readPing(); // obtene la distancia de algun objeto si es que lo hubiera**

**if (curDist < COLL\_DIST) { //si la distancia es menor antes de que colisione**

**checkCourse(); // funcion que checa el curo y lo recorre**

**break; //salta de este ciclo**

**}**

**if (curDist < TURN\_DIST) { // si la distancia es menor gira**

**changePath(); // corre la rutina o funcion chancepath**

**}**

**if (curDist > curDist) {maxAngle = pos;}**

**if (pos > 90 && curDist > curLeft) { curLeft = curDist;}**

**if (pos == 90 && curDist > curFront) {curFront = curDist;}**

**if (pos < 90 && curDist > curRight) {curRight = curDist;}**

**}**

**maxLeft = curLeft;**

**maxRight = curRight;**

**maxFront = curFront;**

**}**

**//-------------------------------------------------------------------------------------------------------------------------------------**

**void setCourse() { // coloca la dirección de viaje basada en el mapeo**

**if (maxAngle < 90) {turnRight();}**

**if (maxAngle > 90) {turnLeft();}**

**maxLeft = 0;**

**maxRight = 0;**

**maxFront = 0;**

**}**

**//-------------------------------------------------------------------------------------------------------------------------------------**

**void checkCourse() {**

**moveBackward();**

**delay(500);**

**moveStop();**

**setCourse();**

**}**

**//-------------------------------------------------------------------------------------------------------------------------------------**

**void changePath() {**

**if (pos < 90) {veerLeft();}**

**if (pos > 90) {veerRight();}**

**}**

**//-------------------------------------------------------------------------------------------------------------------------------------**

**int readPing() { //lee la distancia del sensor ultrasonico.**

**delay(70);**

**unsigned int uS = sonar.ping();**

**int cm = uS/US\_ROUNDTRIP\_CM;**

**return cm;**

**}**

**//-------------------------------------------------------------------------------------------------------------------------------------**

**void checkForward() { if (motorSet=="FORWARD") {motor1.run(FORWARD); motor2.run(FORWARD); } } // se asegura que los motores vayan adelante**

**//-------------------------------------------------------------------------------------------------------------------------------------**

**void checkBackward() { if (motorSet=="BACKWARD") {motor1.run(BACKWARD); motor2.run(BACKWARD); } } // se asegura que los motores vayan hacia atras**

**//-------------------------------------------------------------------------------------------------------------------------------------**

**// In some cases, the Motor Drive Shield may just stop if the supply voltage is too low (due to using only four NiMH AA cells).**

**// The above functions simply remind the Shield that if it's supposed to go forward, then make sure it is going forward and vice versa.**

**//-------------------------------------------------------------------------------------------------------------------------------------**

**void moveStop() {motor1.run(RELEASE); motor2.run(RELEASE);} // para los motores**

**//-------------------------------------------------------------------------------------------------------------------------------------**

**void moveForward() {**

**motorSet = "FORWARD";**

**motor1.run(FORWARD); // hace que se mueva adelante el motor 1**

**motor2.run(FORWARD); // hace que se mueva adelante el motor 2**

**for (speedSet = 0; speedSet < MAX\_SPEED; speedSet +=2)**

**{**

**motor1.setSpeed(speedSet+MAX\_SPEED\_OFFSET);**

**motor2.setSpeed(speedSet);**

**delay(5);**

**}**

**}**

**//-------------------------------------------------------------------------------------------------------------------------------------**

**void moveBackward() {**

**motorSet = "BACKWARD";**

**motor1.run(BACKWARD); // turn it on going forward**

**motor2.run(BACKWARD); // turn it on going forward**

**for (speedSet = 0; speedSet < MAX\_SPEED; speedSet +=2) // slowly bring the speed up to avoid loading down the batteries too quickly**

**{**

**motor1.setSpeed(speedSet+MAX\_SPEED\_OFFSET);**

**motor2.setSpeed(speedSet);**

**delay(5);**

**}**

**}**

**//-------------------------------------------------------------------------------------------------------------------------------------**

**void turnRight() {**

**motorSet = "RIGHT";**

**motor1.run(FORWARD); // hace que el motor uno vaya hacia adelante**

**motor2.run(BACKWARD); // hace que el motor dos vaya hacia atras**

**delay(400); // corre los motores alrededor de 400 milisegundos**

**motorSet = "FORWARD";**

**motor1.run(FORWARD); // hace que ambos motores vayan hacia adelante**

**motor2.run(FORWARD);**

**}**

**//-------------------------------------------------------------------------------------------------------------------------------------**

**void turnLeft() {**

**motorSet = "LEFT";**

**motor1.run(BACKWARD); //**

**motor2.run(FORWARD); // prende el motor 2 hace que vaya hacia adelante**

**delay(400); // corre el motor y le da un delay de 400 milisegundos**

**motorSet = "FORWARD";**

**motor1.run(FORWARD);**

**motor2.run(FORWARD);**

**}**

**//-------------------------------------------------------------------------------------------------------------------------------------**

**void veerRight() {motor2.run(BACKWARD); delay(400); motor2.run(FORWARD);} //verifica el motor derecho vaya hacia adelante alrededor de 400 ms**

**//-------------------------------------------------------------------------------------------------------------------------------------**

**void veerLeft() {motor1.run(BACKWARD); delay(400); motor1.run(FORWARD);} // veering el motor izquier vaya hacia atras alrededor de 400 ms**

**//-------------------------------------------------------------------------------------------------------------------------------------**

**CONCLUSIÓN:**

Se utilizó un enfoque heurístico para dar una propuesta de solución que permita ayudar a personas invidentes en el desenvolvimiento de sus actividades diarias para esto se implementó un sensor ultrasónico montado sobre un moto reductor el cual le permite hacer una barrido de detección el cual a su vez el moto reductor se montó sobre un dispositivo móvil cuyo movimiento se vería controlado por el algoritmo el cual toma en cuenta la señal recibida por el sensor y la dirección que tiene el moto reductor como un disparador para determinar y activar el movimiento de los motores de esta forma se puede guiar al dispositivo para que pueda evadir el obstáculo.