Proyecto

Coche teledirigido con RaspberryPi

DEF

(Documento Especificación Funcional)

Antonio Duce Gimeno

ALU.50301

Índice de contenido

Índice de ilustraciones

Introducción

La finalidad principal del proyecto es estudiar la capacidad para controlar elementos físicos desde aplicaciones móviles, aplicaciones Web y aplicaciones de escritorio. Para esta finalidad usaremos un dispositivo (RaspberryPi) que nos permitirá interactuar con el medio físico mediante los puertos GPIO.

Inicialmente el estudio lo vamos a centrar en la capacidad de controlar los movimientos de un coche teledirigido, al cual dotaremos de diversos sensores (Video, Proximidad) que nos permitirán tener un conocimiento del entorno en el que estamos trabajando .

Estudiaremos de esta misma forma las distintas formas que tenemos de comunicarnos con el dispositivo que vamos a crear, transmitir ordenes y recibir información.

Podemos englobar el proyecto dentro de una categoría de vehículo autónomo, para ello construiremos un prototipo de coche teledirigido que sea capaz de reconocer el entorno en el que circula.

Requisitos funcionales

- *Construir un dispositivo que tenga capacidad de movimiento*
- *El dispositivo creado tiene que ser capaz de obtener información del medio que le rodea*
- *Construir un sistema multi-plataforma que nos permita comunicarnos con el dispositivo*
- *El sistema tiene que ser capaz de comunicarse con el dispositivo para obtener y mandar información*
- *El sistema tiene que ser capaz de establecer una conexión con el dispositivo por medio de una red PAN*
- *El sistema tiene que ser capaz de establecer una conexión con el dispositivo por medio de una red LAN*
- *El sistema tiene que ser capaz de establecer la conexión con el dispositivo por medio de una red WAN.*
- *El sistema tiene que ser capaz de controlar los movimientos del dispositivo creado*
- *El dispositivo tiene que ser capaz de mandar información del medio que le rodea la sistema controlador.*
- *El dispositivo tiene que ser capaz de detectar obstáculos*
- *El dispositivo tiene que se capaz de calcular trayectorias para esquivar obstáculos.*

Estudio General

Dentro de este campo, controlar elementos físicos desde un ordenador tenemos una gran variedad de proyectos desde casas domóticas, hasta coches autónomos sin conductor, pasando por multitud de proyectos industriales, todos ellos se basan en el mismo principio, obtener un control telemático del medio físico.

En este apartado pondremos varios ejemplos de uso y estudiaremos un poco mas o fondo como están desarrollados estos proyectos.

Vehículos Autónomos

Otro gran precedente dentro de este campo son los coche autónomos, coches que pueden conducir de forma autónoma sin necesidad de un conductor, se define este hecho como la capacidad que tiene un sistema de imitar las acciones humanas de manejo y control de un vehículo, por lo tanto un vehículo autónomo es capaz de percibir el medio que le rodea mediante diversos sensores y técnicas de navegación y actuar en consecuencia.

Últimamente se ha hablado mucho de un proyecto el cual parece que es el el principal representante de este campo, pero realmente aunque es una innovación aplicar automatismos para conseguir que un sistema sea capaz de tomar la gran cantidad de decisiones que implica conducir, ya existían varios proyectos prototipos sobre este tema.

Por ejemplo en la Exposición universal de 1939 se presento un coche eléctrico que conducía gracias a un circuito integrado en el pavimento.

En 1980 DARPA (Defense Advanced Research Projects Agency) hizo que un vehículo condujese mas de 600 metros por terreno abrupto fuera de mapa, gracias a un sistema de guía láser.

En 1994 los vehículos construidos por Daimler-Benz y Ernst Dickmans , condujeron por una autopista de 3 carriles en París mas de 1000 kilómetros con trafico intenso a una velocidad de 130km/h solo con pequeñas intervenciones humanas.

En 1995 El equipo Dickmans aplico un sistema de visión computerizada a su coche y permitió realizar un viaje Munich- Copenague ida y vuelta a mas de 175Km/h, con un 95% de conducción autónoma.

Dentro de este tema se puede destacar también el gran proyecto de Google de coche autónomo derivado de un proyecto ganadar de DARPA Grand challenge 2005 (Stanley) coche autónomo creado por el equipo de Sebastian Thrun ingerniero de Google y Director de Stanford Artificial Intelligence Laboratory.

Ilustración 1: Google Driverless Car.

El dotar de inteligencia a los automóviles es una linea que se ha ido reforzando con el paso del tiempo gracias a los avances en electrónica y computación, en realidad actualmente muchos de los coches actuales ya implementan sistemas inteligentes. El principal objetivo de dotar de esta inteligencia a los automóviles se basa en erradicar el error humano consiguiendo de estar forma vehículos capaces de circular de forma autónoma.

Aunque la tecnología ha avanzado mucho en este aspecto, aun dista mucho el objetivo de tener vehículos autónomos circulando libremente. Por este motivo actualmente el desarrollo de vehiculos inteligentes de forma comercial esta centrado en los ***sistemas avanzados de asistencia a la conducción (ADAS),*** estos sistemas están basados en sensores de entorno (Ultrasonidos, Radar) Cámaras de vídeo que detectan objetos y son capaces de detectar las inmediaciones del vehiculo, ayudando al conductor a hacer la conducción mas cómoda y segura.

Entre los sistemas ADAS existentes encontramos las siguientes:

- **Sistema de navegación a bordo (In-vehicle navigation system)**. Con GPS y TMC para ofrecer posicionamiento en carretera e información del tráfico

- **Control de crucero adaptativo (Adaptative cruise control).** Adapta automáticamente la velocidad del vehículo para mantener una distancia seguridad con los vehículos que le anteceden.

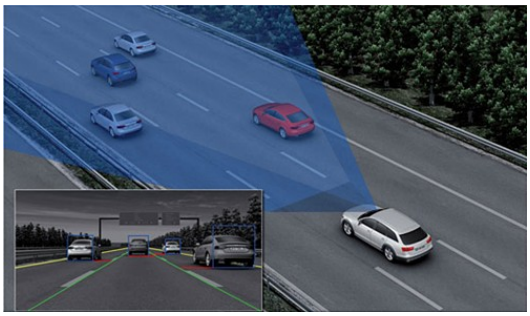


Ilustración 2: Control de crucero adaptativo

Ilustración 3: Navegador de a bordo

- **Asistencia de cambio de carril (Lane change assistance).** Son sistemas que cubren el ángulo muerto de los vehículos avisando de la presencia de otros en sus inmediaciones.
- **Sistema de advertencia de abandono de carril (Lane departure warning system).** dvierte al conductor cuando el vehículo comienza a salirse de su carril en autopistas y carreteras principales.

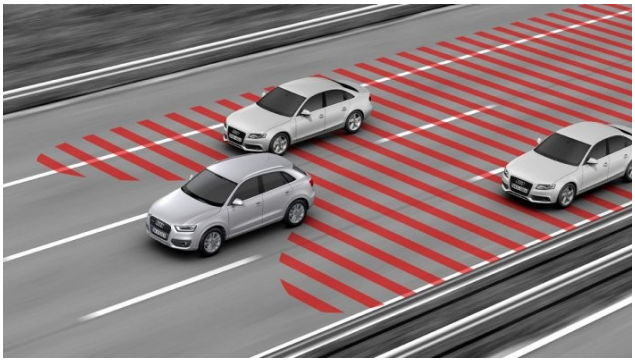
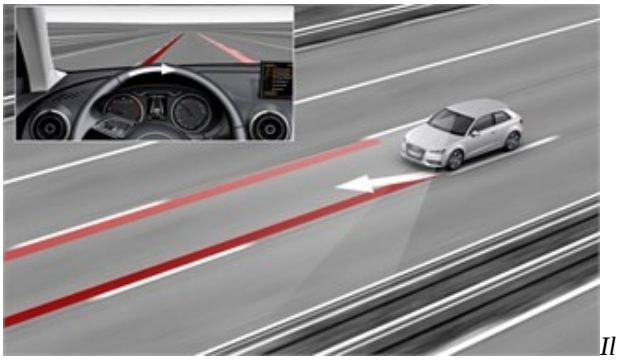


Ilustración 4: Asistencia de cambio de carril



Il ustración 5: Advertencia de abandono de carril

- ***Sistema anticolisión (Collision avoidance system).*** *Detecta obstáculos u otros vehículos detenidos en la vía y actúa sobre el freno del vehículo en caso de prever colisión.*
- ***Adaptación de velocidad inteligente (Intelligent speed adaptation).*** *Este sistema monitoriza la velocidad límite de la carretera y advierte al conductor o actúa sobre el vehículo en caso de que este sobrepase dicho límite.*

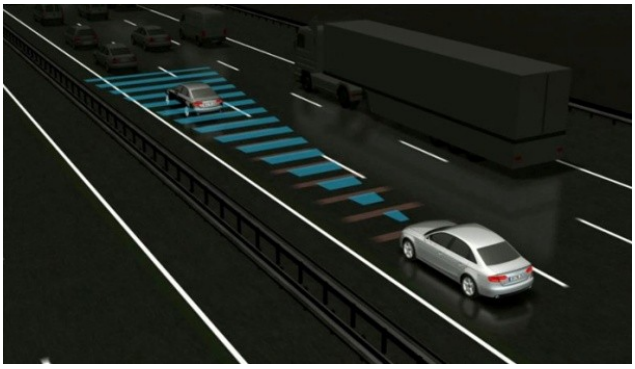


Ilustración 6: Sistema anti-colisión

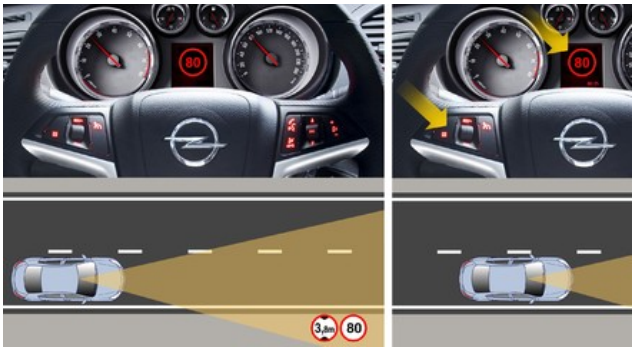


Ilustración 7: Adaptación de velocidad inteligente

- ***Reconocimiento de señales de tráfico (Traffic sign recognition).*** *Capta las señales de tráfico y advierte al conductor sobre su presencia en la carretera.*

- ***Ayuda en aparcamiento (Park assist).*** *Es un sistema que cuando se activa indica al conductor mediante alertas la cercanía a los obstáculos en la maniobra de aparcamiento.*

Ilustración 8: Reconocimiento de señales

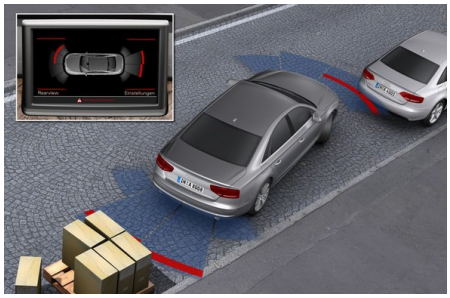


Ilustración 9: Ayuda en aparcamiento

- ***Control de luces adaptativo (Adaptive light control).*** *Actúa sobre las luces de del vehículo en respuesta a las condiciones de visibilidad, dirección, suspensión, velocidad del vehículo, o presencia de otros vehículos.*

- ***Visión nocturna (Automotive night visión).*** *Dispone de una pantalla donde se monitoriza la carretera en visión nocturna. Sirve para mejorar la percepción del conductor en la oscuridad.*

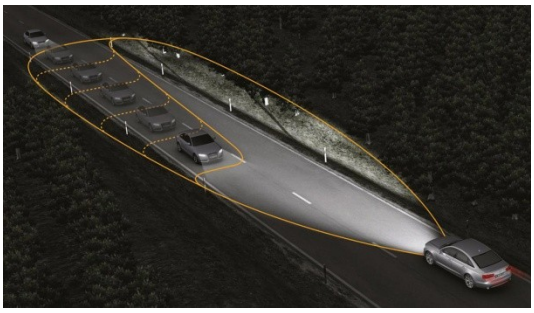


Ilustración 10: Control de luces adaptativo

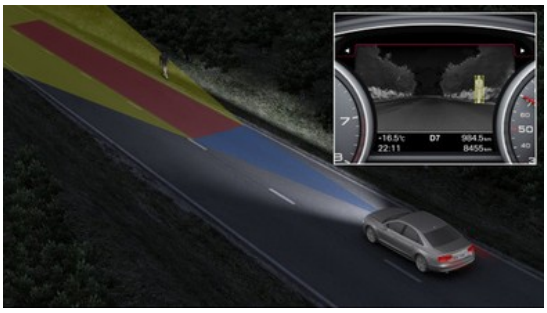


Ilustración 11: Visión nocturna

- ***Control de descenso (Hill descend control).*** *Permite un descenso de pendientes suave y controlado en terrenos irregulares sin que el conductor tenga que pisar el freno.*

- ***Detección de somnolencia en el conductor (Driver drowsiness detection).*** *Dispone de una cámara que monitoriza la cara del conductor y detecta cuando éste está somnoliento.*

- ***Sistemas de comunicación Vehicular (Vehicular communication systems).*** *Son nodos que se sitúan en las carreteras y que se comunican con los vehículos transmitiéndoles información sobre el tráfico o advertencias de seguridad.*

En conclusión vemos como cada vez se van integrado mas automatismos o sistemas ADAS que van tomando mas protagonismo en la conducción en busca de la meta de la conducción segura y autónoma que ira tomando relevancia según vayamos aceptando estas nuevas mejoras.

Se puede observar que dentro del ámbito de la conducción autónoma no existe ningún estándar puesto que aun esta en vías de desarrollo una solución única y cada modelo esta basado en sus propios sistemas, no cabe duda que este es un gran proyecto que tarde o temprano terminara por salir a luz y estandarizarse puesto que los controles de seguridad necesarios para conseguir que un coche conduzca sin intervención humana son gigantescos.

Drones

Vehículo aéreo no tripulado (VANT) o Dron es una aeronave autónoma o controlada a distancia, Inicialmente usados con fines militares se popularizaron en la población civil.

Históricamente los VANT eran siempre vehículos pilotados remotamente , pero se ve una cierta evolución a tener un control sobre el vehículo desde una ubicación remota o dándoles instrucciones de vuelo pre programadas dándoles un complejo sistema de automatización dinámica.

Se pudo observar un punto de inflexión por el cual se popularizaron entre la población civil con la aparición de los primeros drones de Parrot AR.Drone, que aunque no fue ni mucho menos el primero, si que lo popularizo este tipo de vehículos.

Percepción del entorno

Uno de los fundamentos básicos de la conducción es la percepción del entorno, este hecho no es diferente en un coche autónomo puesto que es necesario que este de forma independiente sea capaz de reconocer el terreno y los posibles obstáculos que le rodean.

Esta percepción se puede realizar con muchas y variadas técnicas que pasamos a relatar un poco mas a fondo a continuación

Visión artificial

Sin duda esta es una de las técnicas mas importantes dentro de la automatización de un vehículo, podriamos definirla como la técnica dentro de la inteligencia artificial capaz de "entender" o "detectar" los objetos en una imagen.

Como características básicas dentro de la visión artificial podriamos destacar la siguientes:

- *Detección y localización de objetos (caras , coches..).*
- *Registro y evaluación de resultados*
- *Seguimiento de objetos entre distintas imagenes.*
- *Mapeo de un escenario para conseguir un modelo tridimensional , permitirá moverse por el escenario capturado.*

Todos estos objetivos se consiguen por distintas técnicas unas mas complejas que otras , como reconocimiento de patrones, geometría y proyección, aprendizaje estático, teoría de grafos...

Uno de los aspectos mas importantes de la visión artificial es sin duda el reconocimiento de objetos , patrones o identificación de figuras y formas, este reconocimiento puede ir desde ejemplos simples (reconocer los elementos rojos de un fotografía) hasta la detección y reconocimiento de personas.

Detección de objetos

En este campo tenemos muchas alternativas de estudio como la detección de objetos por patrones, colores o formas pero dato que en el mundo real la variación es demasiado grande tenemos que recurrir a cálculos mas avanzados como la detección de bordes.

Para obtener la forma de un imagen es necesario obtener los bordes de esta, se podria definir como la transición entre dos regiones significativamente distintas, esto nos proporciona una información muy importante sobre los objetos que existen en la imagen.

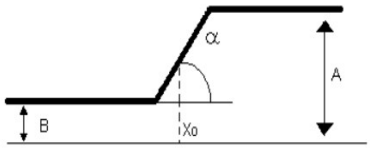


Ilustración 12: Modelo de borde ideal

En la imagen anterior se muestra un modelo unidimensional y continuo de un borde, este modelo representa la transición entre una región Rango B y otra completamente diferente Rango A.

El resultado de aplicar una detección de bordes a una imagen puede reducir significativamente la cantidad de datos a ser procesados y así descartar información no relevante para los cálculos, existen múltiples algoritmos para conseguir este objetivo, aunque encontramos implementaciones creadas sobre estas funciones, explicamos brevemente algunas de ellas:

- ***Detector de bordes Canny.*** *Esta función utilizada en gran cantidad de software de procesamiento gráfico debe su nombre a su autor (John F. Canny), el algoritmos consiste en seleccionar los píxeles candidatos a pertenecer a un contorno mediante dos umbrales, uno alto y uno bajo, si un pixel tiene un gradiente superior a el umbral alto es considerado como borde si por el contrario es meno que el umbral bajo es descartado, si el pixel esta entre los dos umbrales solo se acepta si esta en contacto con un pixel de umbral alto.*

- ***Transformadas de Hough.*** *Con este calculo es posible detectar todo tipo de figuras que sea posible expresarlas mediante un formula matemática, deben su nombre al su inventor (Paul Hough), para este calculo se usa cuya dimensión es igual al numero de parámetros desconocidos del problema.*

Calculo de distancias

El calculo de distancias mediante una cámara es relativamente sencillo puesto que disponemos de dos de los datos mas relevantes:

- *Altura a la que tenemos la cámara respecto al suelo*

- *Ángulo de inclinación de la cámara.*

Con estos datos podemos obtener otros datos interesantes y útiles mediante el uso de sencillas formulas de trigonometría.

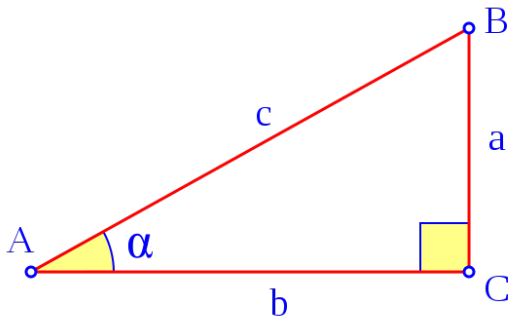


Ilustración 13: Esquema de trigonometría

- ***Calculo de la hipotenusa:*** $c = \frac{b}{\cos\alpha}$
- ***Calculo de la altura:*** $a = c \cdot sen\alpha$
- ***Calculo de la distancia:*** $a = \frac{b \cdot sen\alpha}{\cos\alpha} \rightarrow b = \frac{a \cdot \cos\alpha}{sen\alpha}$

En este punto se nos plantea una incógnita como trasladamos estos datos a una medida física, es decir como traducimos la distancia en píxeles a centímetros, este calculo es también sencillo, puesto que conocemos la altura tanto en el medio real como en el medio físico podemos hacer una regla de tres por la cual obtendremos la distancia aproximada con el objeto.

El problema principal de este método es que necesitamos puntos de referencia con los cuales calcular las distancias,

Visión Artificial Estéreo

La visión estero es el modelo mas usado en la visón artificial para la proyección 3d, y calcular distancias y posiciones en el mundo real.

Una definición mas exacta es la visión binocular donde gracias dos imágenes y mediante una serie de cálculos conseguimos una única imagen.

Cuando trabajamos con cámaras estéreo es necesario tener en cuenta y conocer una serie de datos de importantes:

- *Debemos conocer la distancia entre las dos cámaras así como su ángulo de rotación.*
- *Debemos conocer la distorsión racial de las lentes de las cámaras.*
- *Debemos conocer las distancias focales de las cámaras.*

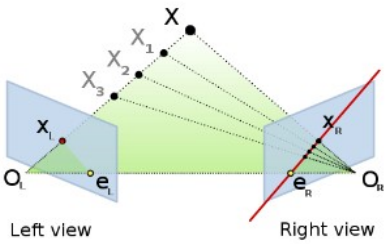


Ilustración 14: Visión Estéreo

Estos datos podemos obtenerlos mediante una serie de cálculos automatizados , usando los algoritmos de Zhang[1] *, los cuales mediante un proceso de hacer varias fotografías a un cuadro de ajedrez en distintas posiciones espaciales nos permite obtener estos datos.*

1 Dada la cantidad de información sobre este tema es preferible no detallar en exceso este punto. Y hacer referencia a sus estudios en la bibliografia.

Geometría de Estereoscópica

Como hemos definido anteriormente para poder hacer uso de la visión estereoscópica necesitamos dos cámaras cuyos ejes ópticos sean paralelos y separadas espacialmente en uno de los ejes, esta separación la definiremos como linea base y es una de las variables que se usaran para el calculo de la profundidad.

Tenemos que tener en cuenta que los ejes ópticos de las dos cámaras solo debe estar separados en una dimensión y nos basaremos en este dato para realizar el resto de los cálculos.

Según el siguiente gráfico

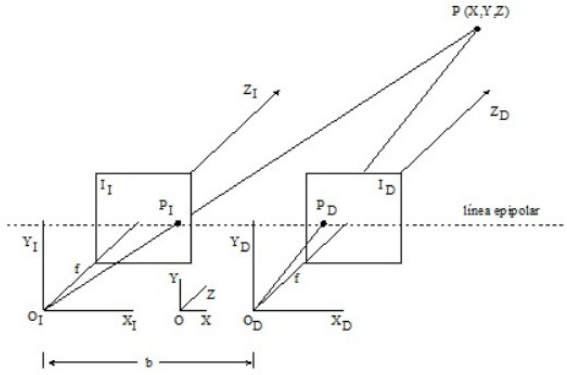


Ilustración 15: Gráfico geometría epipolar

podemos definir que O es el origen de nuestras imágenes, f es la distancia focal y b es la linea base. En cada una de las cámaras estableceremos un punto de coordenadas 3D (X,Y,Z) en el mundo real que se ven representadas por los puntos Pi(X1,Y1,Z1) y Pd(X2,Y2,Z2) que son las proyecciones del puntos real en cada una de las imágenes.

Consideramos también las lineas Poi y Pod para la linea trazada desde el centro focal de las cámaras izquierda y derecha respectivamente, estas dos lineas definen el plano epipolar.

Con este sistema definimos que la diferencia obtenida de restar los dos puntos según el eje de separación de las cámaras X1-X2 es la disparidad *d* entre las dos imágenes.

Utilizando las siguientes ecuaciones.

$$O_i: \frac{\frac{b}{2}+X}{Z}=\frac{X_i}{f} \quad O_d: \frac{\frac{b}{2}-X}{Z}=\frac{X_d}{f} \quad X_i=\frac{f}{Z}\cdot\left(X+\frac{b}{2}\right) \quad X_d=\frac{f}{Z}\cdot\left(X-\frac{b}{2}\right) \quad d=X_i-X_d=\frac{f\cdot b}{Z} \rightarrow Z=\frac{f\cdot b}{d}$$

Según esta formula maestra podemos calcular la distancia a un punto en el espacio desde dos imágenes tomadas en paralelo.

LIDAR

Es el acronimo de (Light Detection and Ranging) es una tecnología que permite medir la distancia desde un emisor a un objeto gracias a un emisor laser pulsado, la distancia al objeto es determinada por el retraso entre la emisión y la recepción del pulso reflejado.

Aunque es mas usado en el campo de la aeronáutica existen distintos proyectos de coches autónomos e implementan este sistema para reconocimiento del terreno y la distancia entre los objetos, también conocida por ser la tecnología usada en los controles de velocidad.

Existen distintas técnicas de uso de este sistema que nos permite hacer un reconocimiento del terreno cada una con sus beneficios y problemas.

- *Lineal: El haz láser es desviado por un espejo en un sentido, da unas medidas uniformes, pero existen puntos ciego al tener un barrido en una única dirección.*
- *ZigZag: Es una evolución del anterior en este caso el movimiento del espejo es en dos sentidos , pero existen descompensaciones de la cantidad de puntos obtenidos, en los extremos, teniendo menos puntos ciegos pero con zonas con menor resolución.*
- *Fibra óptica: En es caso el sistema monta múltiples espejos mas pequeños con múltiples haces lo que nos da mayor resolución pero un ángulo menor de escaneado .*

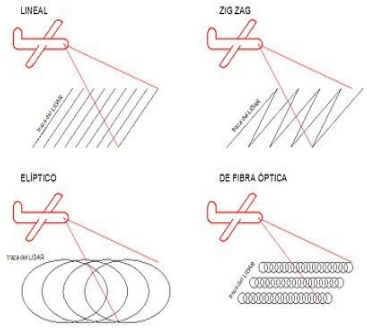


Ilustración 16: Tipos de LIDAR

- *Eliptico en este caso el haz es desviado por dos espejos produciendo un escaneado eliptico que nos da lecturas del mismo punto desde distintos ángulos, lo que nos beneficia en algunos aspectos pero complica el procesamiento al tener distintas lecturas.*

GPS

Esta tecnología hoy por hoy no es desconocida, este sistema de localización permite ubicar con una precisión de centímetros, según la técnica usada, la ubicación de un objeto a nivel mundial.

Esta localización se realiza gracias a un red de satélites en orbita que ubican el objeto gracias a la trilateracion, dándonos la ubicación en el globo con una precisión de metros.

Ilustración 17: Cálculo de la posición por triangulación

Ilustración 18: Red Navstar de satelites GPS

Su funcionamiento es relativamente sencillo pues se basa en le calculo del tiempo transcurrido entre la recepción de las cuatro señales de los satélites y la posición de estos, el dispositivo gps utilizara estos datos para triangular su ubicación respecto a los satélites.

El mensaje que nos mandan (efemeride) *dando la posición precisa de los mismos. Esta información se cambia frecuentemente, siendo actualizada por las estaciones de seguimiento de la Tierra. Los parámetros orbitales de los satélites se van actualizando a medida que su movimiento se ve alterado por la atracción del Sol y la Luna, la diferencia de gravedad entre distintas zonas de la corteza terrestre, viento solar, etc. Un período de cambio típico sería de 4 horas. .*

RADAR

acrónimo de Radio detection and ranging, es un sistema que usa hondas electromagnéticas para medir distancias, altitudes y velocidades a objetos estáticos o en movimiento.

Su funcionamiento no difiere mucho del resto de tecnologías, siendo en realidad la precursora de ellas, básicamente calcula la distancia midiendo el tiempo entre la emisión y la recepción del eco en la misma posición del emisor, de este eco se puede obtener gran cantidad de información, el uso de hondas electromagnéticas de distintas frecuencias nos puede aportar distintos tipos de información.

Radio Control

Básicamente podríamos definir como el sistema por el cual podemos controlar remotamente un objeto de forma inalámbrica, esta comunicación puede ser por cualquier medio de comunicación existente, comúnmente se usan emisoras RF de control remoto.

En el campo del radio control entra en escena tres actores fundamentales electrónica , electricidad y mecánica, que son los actores fundamentales de toda automatización que queramos lograr.

Actualmente existen multitud de vehículos comerciales de todos tipo , coches , aviones, barcos helicópteros.... radio controlados.

Implementación

Elementos necesarios

Para el desarrollo de este proyecto vamos a necesitar varios elementos hardware con los cuales montaremos el dispositivo que queremos controlar remotamente.

Estudiaremos las diversas alternativas que existen actualmente en el mercado y seleccionaremos aquellas que nos ofrezcan mejores resultados.

Inicialmente y como hardware de partida tendremos el siguiente:

RaspberryPi (Modelo B, B+)

Módulo Bluetooth USB (V2, V4)

Módulo Wifi USB (B/G)

Driver Motores

En este caso estudiaremos diversos tipos de control

I2C Adafruit PCA9685

I2C PiBorg Reverse PIC16F1824

Puente H L298N

I2C Port Expander MCP23017

Servo Motores Rotación Continua (

En este caso hemos seleccionado los siguientes tipos

SM-S4303R

FS90R

Motores DC

En este caso no seleccionamos ningún modelo especifico pero indicamos la necesidad de un sistema de moto reducción para limitar la RPM y aumentar el PAR motor,

Sensores de proximidad

Modelo HC-SR04

Cámara USB

No seleccionamos modelo especifico, pero debe ser capaz de comprimir por hardware el fuljo de video.

Descripción Software Utilizado

IDES Desarrollo

Librerías de desarrollo

Sistema Operativo

Preparación del entorno

Descripción del hardware utilizado

Controlador del sistema

Como elemento principal de sistema y encargado de controlar todos los dispositivos y donde residira la lógica del proyecto usaremos dispositivos de tipo raspberryPI, en este caso nos hemos decidido por el modelo B/B+ puesto que nos ofrece una serie de características que consideramos necesarios.

Pasmos a describir rápidamente este tipo de dispositivo.

RaspberryPI B/B+

Es un ordenador de bajo coste basado en el procesador ARM11, aunque nació como un ordenador de bajo coste para facilitar la enseñanza de informática en colegios, se ha convertido en un precedente a nivel mundial y ha abierto un nuevo mundo de proyectos hasta hace poco inabordables por coste o dificultad.

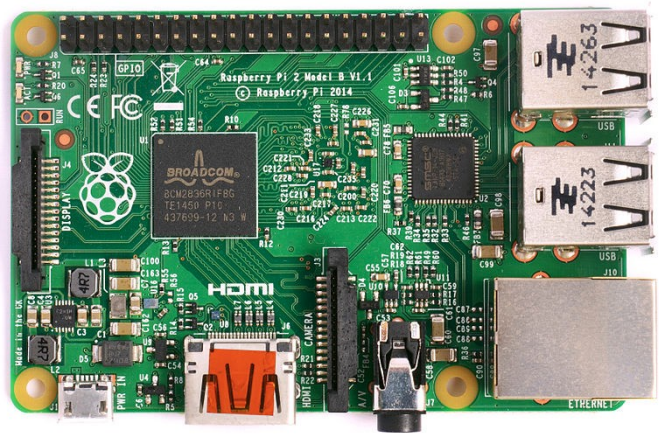


Ilustración 19: RaspberryPi B+

Las características principales por las que elegimos este dispositivo son las siguientes:

4 puerto USB 2.0

Procesador de ARM11 1176JZF-S a 700 MHz overclokeable de forma sencilla hasta 1Ghz

512 Mb de memoria Ram

Conexión de red mediante RJ45

Lector de tarjetas SD/MSD

Alimentación de bajo consumo 5V/2A

SO basado en linux

Controlador de Motores

I2C

I2c (Inter-Integrated Circuit) es un bus de comunicaciones serie usado principalmente para comunicación entre microcontroladores y periféricos .

La principal característica de I2C es que hace uso de dos lineas para transmitir la información, una de ellas transporta los datos (SDA) y la otra transporta la señal de reloj (SCL) . Es necesaria una tercera linea utilizada única y exclusivamente como masa (GND).

Los dispositivos conectados por I2C tiene una dirección única que los identifica inequívocamente el dispositivo en la serie de periféricos , estos se conectan en cascada permitiendo conectar varios dispositivos y que todos funcionen al mismo tiempo.

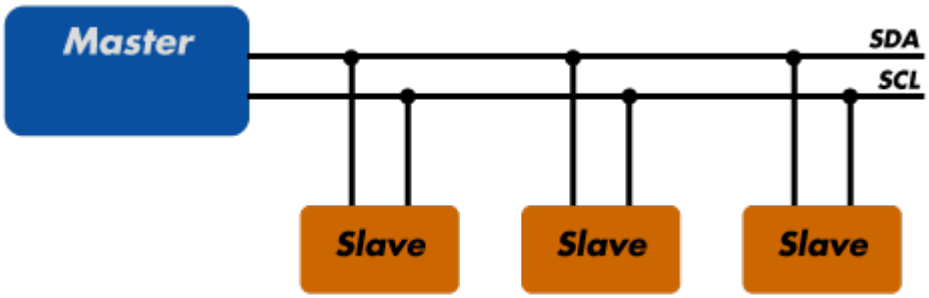


Ilustración 20: Esquema de conexión I2C

*Habiendo varios dispositivos conectados sobre el bus, es lógico que para establecer una comunicación a través de él se deba respetar un protocolo. Digamos, en primer lugar, lo más importante: existen dispositivos **maestros** y dispositivos **esclavos**. Sólo los dispositivos maestros pueden iniciar una comunicación*

*La condición inicial, de **bus libre**, es cuando ambas señales están en estado lógico alto. En este estado cualquier dispositivo maestro puede ocuparlo, estableciendo la condición de **inicio** (start).*

Esta condición se presenta cuando un dispositivo maestro pone en estado bajo la línea de datos (SDA), pero dejando en alto la línea de reloj (SCL).

El primer byte que se transmite luego de la condición de inicio contiene siete bits que componen la dirección del dispositivo que se desea seleccionar, y un octavo bit que corresponde a la operación que se quiere realizar con él (lectura o escritura).

Si el dispositivo cuya dirección corresponde a la que se indica en los siete bits (A0-A6) está presente en el bus, éste contesta con un bit en bajo, ubicado inmediatamente luego del octavo bit que ha enviado el dispositivo maestro. Este bit de ***reconocimiento*** *(ACK) en bajo le indica al dispositivo maestro que el esclavo* ***reconoce*** *la solicitud y está en condiciones de comunicarse. Aquí la comunicación se establece en firme y comienza el intercambio de información entre los dispositivos.*

I2C Adafruit PCA9685

Este es uno de los controladores de motor que usaremos en nuestras pruebas , fabricado por adafruit (https://www.adafruit.com) se nos ofrece esta placa en un kit de sencillo ensamblado.

Inicialmente este dispositivo es usado para controlar servomotores o leds RGB.

Cada una de las 16 entradas independientes que tiene el dispositivo nos permite controla un motor distinto e independiente de lo demás.

Contradictoriamente solo podremos controlar un máximo de 4 led RGB dado que cada color del led sera controlado por un canal distinto.

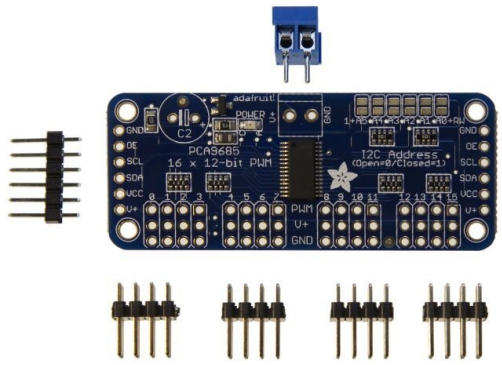


Ilustración 21: 16-channel, 12-bit PWM Fm+ I2C-bus LED controller

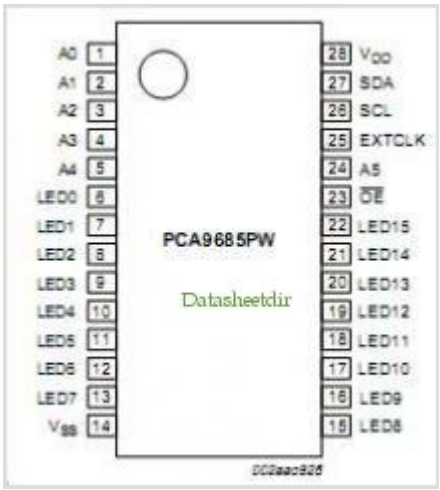


Ilustración 22: PCA9685

Esta placa nos ofrece las siguientes capacidades:

Controlador PWM sobre I2C , lo que nos permite no tener que estar controlando los pulsos ni mandar información constante a los dispositivos controlados.

Capacidad de una doble alimentación para alimentar el microcontrolador a 3.3V y un segunda entrada de potencia que permite alimentar los dispositivos controlados de forma independiente.

Permite configurar la dirección de controlador anteriormente indicada, ofreciendo la posibilidad de configura nuestro sistema en distintas direcciones.

Permite frecuencias de hasta 1,6Khz lo que nos ofrece una alta tasa de transferencia.

Permite el apagado directo de todos los dispositivos de forma automatica escribiendo en una única posición de memoria.

I2C PiBorg Reverse PIC16F1824

Este controladora de motores al no permite manejar 2 motores DC o un motor de pasos, al igual que la anterior funciona mediante el protocolo I2C.

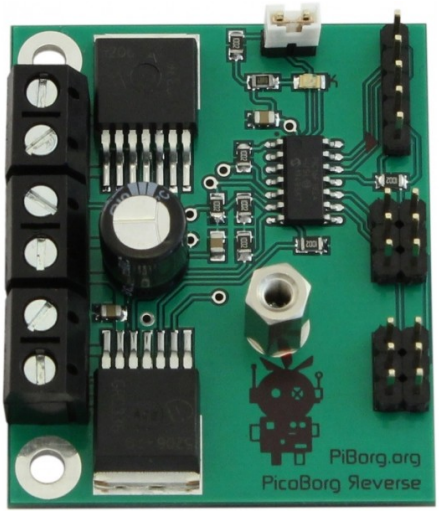


Ilustración 23: I2C PiBorg Reverse

Aunque con menores capacidades en principio es un placa completamente valida puesto que no tenemos necesidad de mas de dos motores para controlar el coche.

Esta controladora nos ofrece la posibilidad de controlar automáticamente un motor de pasos , característica que puede ser de utilidad a la hora de controlar exactamente el giro del motor.

Puente H L298N

Este elemento al igual que el anterior nor permite controlar dos motores DC o un motor de pasos, pero a diferencia del anterior no es controlado por I2C.

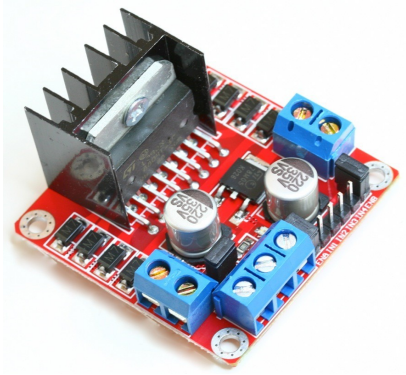


Ilustración 24: H Bridge L298N

Este elemento es un componente mucho mas económico que los dos anteriores, pero como inconveniente nos aporta la dificultad de control del interfaz, teniendo que controlar la potencia y el estado de cada uno de los pines constantemente.

Por otro lado tiene la problemática del control de temperatura puesto que este tipo de elemento tiende a calentarse en exceso impidiendo un uso prolongado del interfaz.

I2C Port Expander MCP23017

Como ultimo elemento dentro de los componentes I2C testearemos las capacidades de este micro controlador que nos ofrece una capacidad de expansión de puertos.

Con este controlador por bus I2C tenemos la posibilidad de ampliar el numero de puertos I/O a nuestro dispositivo RaspberryPI.

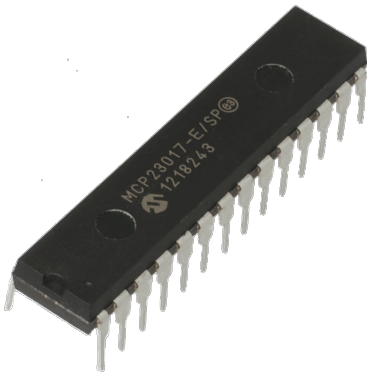


Ilustración 25: Controlador MCP23017

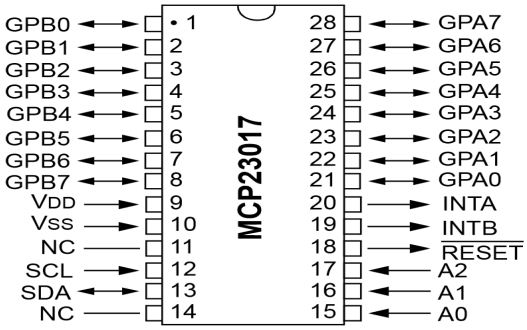


Ilustración 26: Esquema MCP23017

Este microcontrolador nos ofrece las siguientes características:

Ampliación de 16 puertos I/O

Nos permite aislar los puertos GPIO de la raspberryPi

Controlado por medio del protocolo I2C

Fácil configuración mediante pins de la dirección de memoria utilizada.

Parada de emergencia de todas las salidas mediante la escritura en una única dirección de memoria.

Permite la conexión en paralelo de hasta 9 dispositivos en cascada permitiendo aumentar el numero de puertos disponibles de forma sencilla.

Implantación

En este punto vamos a definir que elementos vamos a usar y como vamos a implementar la solución a nuestro proyecto.

Visión artificial y detección de objetos

Unos de los puntos mas importantes del proyecto es este pues vamos a crear un sistema que sea capaz de conducir de forma autónoma esquivando obstáculos que pongamos en su camino.

Partiremos de la implementación de la librería de procesamiento de imágenes OpenCV, esta librería es usada en multitud de proyectos, incluidos algunos proyectos de conducción autónoma.

Actualmente esta librería nos da la capacidad de obtener imágenes de una cámara procesarlas y obtener información de ellas.

Detección de obstáculos.

Aunque existen múltiples formas de abordar este tema nos centraremos inicialmente un hecho simple, vamos a partir de que todos los obstáculos que nos encontraremos tiene algo en común, aunque no es un dato muy realista nos servirá para avanzar en la implantación del proyecto y continuar desde este punto hacia objetivos mas complejos.

Para ello definiremos que todos los obstáculos que vamos a encontrar tendrán un circulo redondo en el centro del obstáculo que podamos identificar mediante visión artificial.

Esto nos permitirá de forma sencilla detectar si tenemos algún obstáculo en nuestro campo de visión, el siguiente punto que abordaremos mas adelante sera el calculo de la distancia hasta dicho obstáculo.

Diagrama de procesamiento

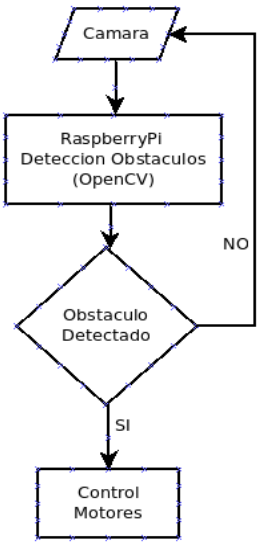


Ilustración 27: Etapa inicial proceso

Como hemos definido anteriormente vamos a actuar sobre obstáculos con una forma determinada, aun así de forma genérica podemos construir un procedimiento que iremos evolucionando a lo largo del proceso.

Inicialmente partiremos del siguiente diagrama :

Como Inicio del proceso nuestro sistema buscara obstáculos definidos por una imagen con un circulo negro, para ello usaremos las librerías OpenCV y mas concretamente la función ***HoughCircle.***

La función HoughCircles es una evolución de la función HouhLines la cual realiza varios cálculos para detectar lineas rectas dentro de una imagen, en nuestro la función nos retornara un vector con los círculos detectados indicando su centro y su radio.

Podemos observar por los siguientes ejemplos como funcionara el sistema.



Tabla 1: Detección Círculos con OpenCV

En este ejemplo vemos como por medio de la cámara capturamos una imagen , buscamos posibles círculos que se encuentren en la imagen y los marcamos dibujando el circulo sobre esta.

Hemos coloreado los las zonas donde no se encuentran los círculos para destacar las posibles rutas que podremos seguir.

```
public Mat detectarCirculos()
    {
        capturarImagen();
        Mat circulosDetectados = new Mat();

        Imgproc.erode(m_ultimaImagenGris, m_ultimaImagenGris,new Mat());
        Imgproc.dilate(m_ultimaImagenGris, m_ultimaImagenGris,new Mat());
        Imgproc.Canny(m_ultimaImagenGris, m_ultimaImagenGris, 5, 70);
        Imgproc.GaussianBlur( m_ultimaImagenGris, m_ultimaImagenGris, new Size(9, 9), 2, 2 );

        Imgproc.HoughCircles(m_ultimaImagenGris, circulosDetectados, Imgproc.CV_HOUGH_GRADIENT, 1, 1 ,200,100,0,0);
        System.out.println(circles);

        for (int i = 0; i < circulosDetectados.cols(); i++)
        {
            double[] circuloBucle = circulosDetectados.get(0, i);
            Point pCentroBuble = new Point(circuloBucle[0], circuloBucle[1]);
            Imgproc.circle(m_ultimaImagen, pCentroBuble, (int) circuloBucle[2]- 10 , new Scalar(0, 255, 0),10);
        }
        return m_ultimaImagen;
    }
```

Realmente no es necesario realizar ninguna modificación en la imagen que seria muy costosa en tiempo de procesamiento, simplemente nos bastaría con la información contenida en la variable circulosDetectados para continuar con el procesamiento.

Como se puede observar se realiza un procesamiento previo de la imagen, este procesamiento no es necesario pero para evitar posibles fallos en la detección es mejor hacerlo.

Detección de la distancia al objeto

Para el calculo de las distancias haremos uso de de uno o varios sensores que sean capaz de calcular las distancias hasta los obstáculos detectados.

Para ellos haremos uso de la librería Pi4J y de un componente hardware HC-SR04, detallado anteriormente en la sección de hardware.

Ilustración 28: HC-SR04

Una vez que hemos detectado que tenemos un obstáculo delante nuestro ejecutaremos la rutina para que nos diga la distancia hasta dicho obstáculo.

La formula que aplicaremos sera la siguiente $D_{cm} = T_{\mu s} \cdot ((340_{m/s} \cdot 100/10^{-6})/2)$ *que es el proporcional al tiempo que tarda el sonido en recorrer la distancia desde el emisor al obstáculo y volver, podríamos simplificar la formula al tratarse de constantes* $D_{cm} = T_{\mu s} \cdot 0{,}017$.

Calculo de la distancia

El sistema se basa en el uso de los puertos GPIO de la RaspberryPi los cuales hemos definido anteriormente en la sección hardware, para manejar este sensor es necesario hacer uso de una librería externa Pi4J la cual nos da acceso a una serie de métodos con los cuales podremos controlar los Puertos GPIO Leer y escribir información de ellos así como activarlos y desactivarlos.

El diagrama de montaje es relativamente sencillo, necesitaremos dadas las características de los puertos GPIO dos resistencias que pondremos en según el siguiente esquema.

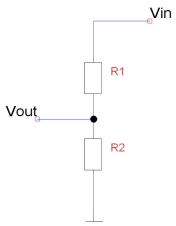


Ilustración 29: Circuito salida HC-SR04

El calculo de la magnitud de las resistencias siguen el siguiente patrón

$$V_{sal} = V_{ent} \cdot \frac{R_2}{R_1 + R_2} \Rightarrow \frac{V_{sal}}{V_{ent}} \cdot \frac{R_2}{R_1 + R_2}$$

$$\frac{3{,}3_v}{5_v} \cdot \frac{R_2}{R_1 + R_2} \Rightarrow 0{,}66 \cdot R_1 = 0{,}34 \cdot R_2 \Rightarrow R_2 = 1{,}94 \cdot R1$$

así pues podemos concluir que aunque si que es importante el valor de las resistencias estas cumple unos características que relacionaran los dos valores, para nuestro caso usaremos resistencias de 1K y de 2K.

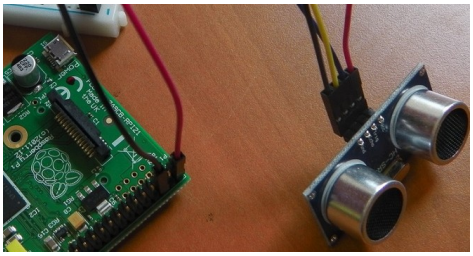


Ilustración 31: HC-SR04 Conectado a RaspberryPi

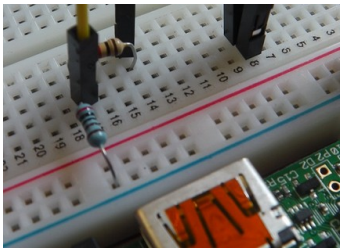


Ilustración 30: Detalle resistencias en la conexión de salida

Diagrama de procesamiento

Partiendo del diagrama anterior completaremos el proceso aplicando una nueva comprobación al proceso, esta es la distancia y según los datos que tengamos Actuaremos de una forma u otra.

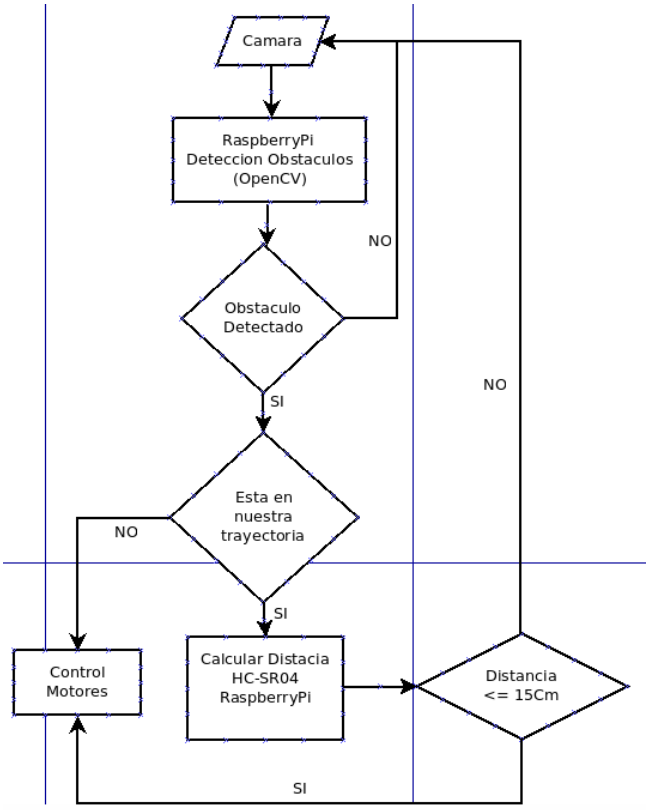


Ilustración 32: Diagrama de proceso con cálculo de distancia

Como podemos observar completamos el diagrama anterior con unos nuevos pasos en los que decidimos que vamos a hacer en caso de que nos encontremos un obstáculo.

Como punto mas importante de todo este proceso es saber a que distancia estamos del obstáculo, definimos un limite de parada de 15cm aunque este sera configurable en puntos posteriores.

Como primer paso solo actuaremos de dos formas inicialmente continuar el recorrido y parada de motores.

El calculo de la distancia como hemos indicado anteriormente lo obtenemos por medio de la siguiente formula $D_{cm} = T_{\mu s} \cdot 0{,}017$ *pero la pregunta es como obtenemos los datos para usar la formula, los obtendremos mediante el siguiente procedimiento.*

Este es una versión simplificada del que usaremos en la versión final, puesto que en este no implementamos toda la funcionalidad de timeouts ni seguridad.

```
public double CalcularDistancia()
    {
        final GpioController gpio = GpioFactory.getInstance();
        final GpioPinDigitalOutput trigPin = gpio.provisionDigitalOutputPin(RaspiPin.GPIO_00, "Trig", PinState.LOW);
        final GpioPinDigitalInput echoPin = gpio.provisionDigitalInputPin(RaspiPin.GPIO_02, "Echo");
        double tiempoInicio = 0d, TiempoFin = 0d, distanciaObstaculo = 0d;
        trigPin.high();
        try
        {
            Thread.sleep(0, 10000);
            System.out.println(".");
        }
        catch (Exception ex)
        {
            ex.printStackTrace();
        }
        trigPin.low();
        while (echoPin.isLow())
        {
            tiempoInicio = System.nanoTime();
        }

        while (echoPin.isHigh())
        {
            TiempoFin = System.nanoTime();
        }

        if (tiempoInicio > 0 && TiempoFin > 0)
        {
            double duracionPulso = (TiempoFin - tiempoInicio) / 1000d; // en MicroSegundos
            distanciaObstaculo = duracionPulso * 0.017;
        }
        return distanciaObstaculo;
    }
```

Control de Movimiento

Para el control del movimiento de nuestro vehículo crearemos un programa que nos permita movernos con libertad sobre el plano mandando un serie de ordenes básicas desde el control de de entorno.

Dentro de este punto tenemos varias alternativas las cuales tienen puntos buenos y malos y cada una tiene sus peculiaridades de implantación, nos hemos decidido por la alternativa de implantar un Puente H L298N.

Al igual que nos ha pasado con los sensores en el punto anterior tendemos que recurrir a una librería externa para el manejo del GPIO y con este el manejo de los interfaces de control de movimiento.

Puente H L298N

Ya hemos hablado de este componente en un punto anterior referente al Hardware utilizado, este componente se nos presenta como uno de los interfaces mas simples que podemos usar, se trata de un componente estándar y económico que nos permite controlar dos motores DC de forma sencilla.

Encontramos varios problemas en este interfaz :

- *Control de PWM: Este interfaz proporciona una funcionalidad básica de PWM (Modulación por ancho de pulsos) pero esta funcionalidad esta supeditada al elemento superior de control, es decir si nuestro controlador, en este caso RaspberryPi, no dispone de Control de I/O por PWM tendremos que implementarlo por Software lo cual conducirá a una perdida de rendimiento que puede afectar a todo el sistema.*

- *Sobrecalentamiento : Este tipo de unidades es conocida por su calentamiento extremo y consecuentemente su perdida de funcionalidad.*

- *Regulador de tensión : Este interfaz proporciona un regulador de tensión auxiliar de 5v, suficiente para alimentar nuestro interfaz de control, pero esta es poco fiable cuando se calienta en exceso produciendo cortes de corriente que pueden llegar a dañar el controlador del sistema*

Control de Giro

El control de giro que implementaremos en este proyecto es un sistema de dirección tipo oruga, este tipo de control de dirección se basa en el mismo sistema con el que se mueven la excavadoras que no disponen de ruedas direccionales.

Para girar a la derecha reducen la velocidad de rotación de las ruedas derechas, las paran o incluso invierten el sentido según la velocidad que quieran de giro.

Para girar a la izquierda hacen exactamente lo mismo que en el caso anterior pero invirtiendo el lado.

Diagrama de procesamiento

Como en los puntos anteriores vamos a mostrar los cambios implementados en el proceso desde este punto.

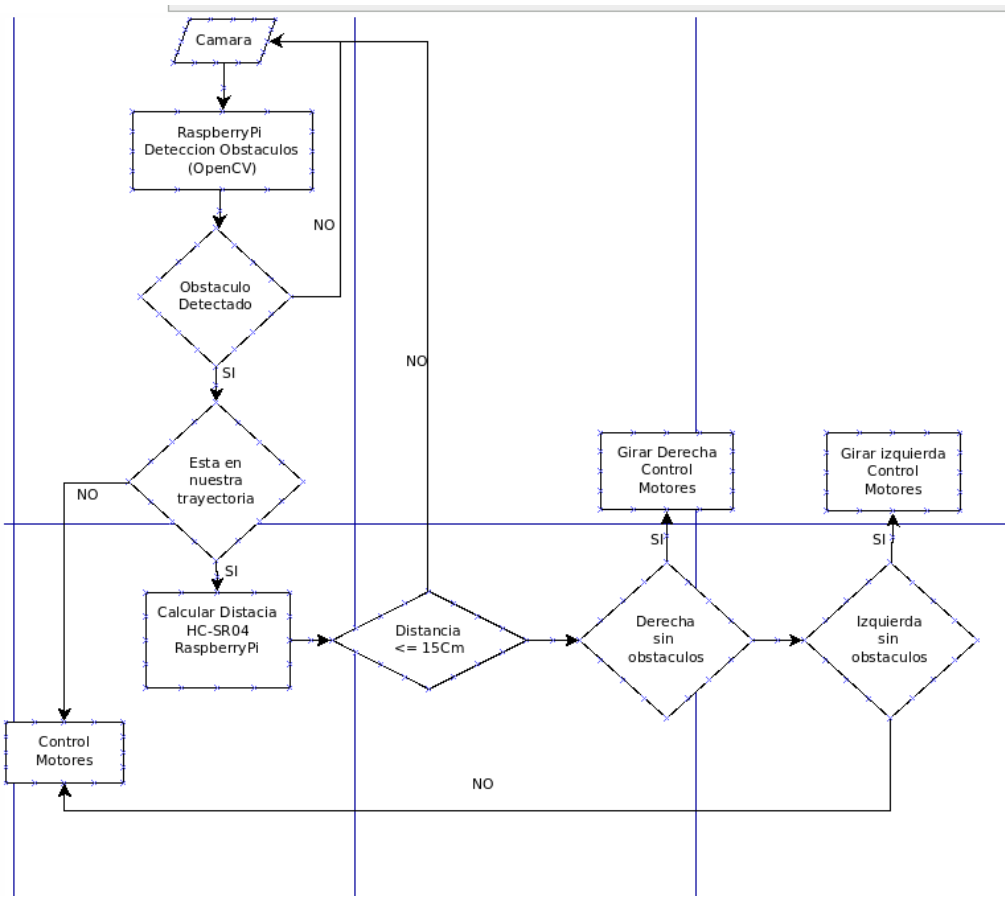


Ilustración 33: Control de giro

Como podemos ver implementamos en este punto un control de dirección básico con giro a la derecha y a la izquierda en caso de encontrar un obstáculo en nuestra ruta actual.

Ante el caso de no poder encontrar un ruta valida, tanto en la ruta actual como en la derecha y en la izquierda detendremos la marcha en pasos posteriores encontraremos una solución a este problema.

Control sentido de la marcha, giro completo

Análisis Coste Proyecto

Evolución del proyecto

En este punto destacamos las posibles mejoras de este proyecto pues consideramos que en este momento no tenemos tiempo suficiente como para desarrollar todas las funcionalidades que hemos encontrado.

Sensor de distancia

Dentro del proyecto se ha implementado un sensor de distancia por ultrasonidos orientado a largo alcance y con rangos de 3cm a 3m posicionado en la parte dentera del vehículo, pero esto no lo consideramos suficiente puesto que evidentemente tenemos puntos ciegos en la parte trasera del vehículo y que según el ángulo en el que se encuentren los obstáculos pueden producir falsos negativos.

Existen otros sensores por ejemplo Sharp GP2Y0A02YK0F que se trata de un sensor de rango largo por infrarrojos con detección continua de proximidad, este sensor trabaja en rangos de 15cm a 1.50m.

Seria posible eliminar los puntos muertos de los sensores instalando mas sensores que complementen el sistema actual.

Visión electrónica

Dentro de este campo existen muchas posibles mejoras, la cuales detallaremos a continuación, que no se han podido implementar por falta de tiempo

Ángulos muertos cámaras

La solución implementada incluye un cámara en ella parte delantera del coche que nos permite tener constancia de los obstáculos que tenemos enfrente nuestro, pero que pasa si el obstáculo o el peligro aparece por los lados y si vamos marcha atrás.

Por lo tanto parece evidente e interesante pensar en la posibilidad de ampliar el sistema de cámaras por lo menos delante y detrás del vehículo las cuales nos den información del entorno en el que nos movemos.

En otro punto tenemos el tema de ángulos muertos, este tema se podría solucionar implementando un sistema de visión panorámica controlada por un motor o implementando un sistema de visión con lentes ojo de pez que nos de un ángulo de visión mayor.

Detección de objetos genéricos

Este es otro punto que desgraciadamente hemos tenido que dejar en el tintero por motivos evidentes, aunque también se ha investigado dentro del proceso de análisis no hemos podido llega a implementar todas las funcionalidades que nos hubiera gustado.

Este paso es interesante pues existen diversas formas de avanzar desde la detección de formas genéricas y detección de bordes a la posibilidad de construir un sistema experto que sea capaz e aprender y recordar diversos obstáculos que se ha encontrado anteriormente y actuar en consecuencia.

Visión 3d

Aunque se ha hecho un estudio bastante avanzado sobre este tema dentro del proyecto no se ha conseguido el objetivo de hacer un modelo funcional de esta tecnología que consideramos suficientemente importante como para ser la protagonista principal de un proyecto general entero.

Esta mejora permitiría junto con la ayuda de diversos sensores establecer un mapa tridimensional con el cual seriamos capaces de movernos y calcular trayectorias así como obtener resultados mas que interesantes, como mapas en 3d , proyecciones de objetos …

Dentro de este tema también tendremos que de capaces de gestionar el procesamiento de todos los cálculos necesarios para conseguir una proyección 3D desde dos imágenes planas, para lo cual la plataforma elegida en este caso, RaspberryPi, se nos queda un poco escasa de potencia.

Sensores de posición

Hemos hablando mucho de sistemas de sensores de proximidad y distancia de objetos pero nos quedamos cortos en un punto importante, ¿Cual es el sentido de muestra marcha? , ¿En que posición estamos sobre el plano? y lo mas importante de todo ¿Nos estamos moviendo?.

Acelerómetros

Parece interesante en este punto hablar de los acelerómetros los cuales nos darán información básica a las tres preguntas que nos hemos planteado y que creemos que a la hora de implementar un sistema real de conducción autónoma es suficientemente importante como para tenerlo encenta.

GPS

Otro punto a destacar dentro de posibles mejoras en el sistema es la implementación de un sistema de posicionamiento.

La ley mas básica del desplazamiento es cual es mi origen y cual es mi destino sin saber estas dos variables no es posible implementar un sistema de conducción autónoma verdaderamente eficaz.

Mejoras en el sistema de control de motores

Dentro de este punto no hemos podido estudiar todos los sistemas de control disponibles y hemos orientado el proyecto hacia una solución mas sencilla de implementar, pero dentro de este punto existen varias mejoras a tener en cuenta.

Control de dirección

Dentro del proyecto hemos implementado un sistema de dirección básica basado en el movimiento denominado como oruga hacer rotar solo los motores de un lado para permitir girar hacia el lado contrario, pero aunque este sistema es eficaz no es lo suficientemente bueno y puede ser mejorado.

Se podría integrar un sistema de dirección controlado por uno o varios servomotores que actuaran sobre las ruedas direccionales que nos permitirá girar de una forma eficaz y controlar de esta forma el ángulo de giro.

Control de motores

Hemos implementado el sistema con un Puente H , que nos da ofrece la capacidad de controlar dos motores DC pero en según que situaciones puede que este sistema se quede corto y queramos implementar mas motores.

Además tenemos la problemática del control de potencia por PWM que en el caso de los puente H tiene que ser controlado por software, existen diversas placas controladoras que permiten un control de PWM propio con el cual nos olvidamos del procesamiento de este.

Bibliografia

- *Especificacion tecnica bus I2C*
  http://www.semiconductors.philips.com/acrobat_download/literature/9398/39340011.pdf
- *Wikipedia http://es.wikipedia.org/wiki/I%C2%B2C*
- *https://www.adafruit.com/datasheets/PCA9685.pdf*
- *http://www.adafruit.com/products/815*
- *https://www.piborg.org/picoborgrev/specs*
- *http://ww1.microchip.com/downloads/en/DeviceDoc/21952b.pdf*
- *http://www.adafruit.com/product/732*
- *http://wiringpi.com/extensions/i2c-mcp23008-mcp23017/*
- *http://es.wikipedia.org/wiki/Raspberry_Pi*
- *http://www.raspberrypi.org/*
- *http://pi4j.com/*
- *http://es.wikipedia.org/wiki/Dom%C3%B3tica*
- *http://www.simondomotica.es/sistemas/index.html*
- *http://www.knx.org/es/*
-