



Universidad Nacional de La Matanza

Departamento de Ingeniería e Investigaciones Tecnológicas

Sistemas Operativos Avanzados

Sistema Embebido Arduino Auto Rastreable

Primer Cuatrimestre 2017

INFORME FINAL

Integrantes:

Andrade, Martin	37181057
Argento, Juan Pablo	34670631
Boullon, Daniel	36385582
Lorenz Vieta, Germán	32022001

Índice

1.Objetivo General del Proyecto	4
2.Solución Propuesta	4
3.Hardware utilizado	4
Descripción	5
Arduino UNO	5
Sensor HC-SR04	5
Sensor MPU 92/50	6
Servo SG90	6
Motor CC con reductora	6
Puente H L923D	7
Módulo Bluetooth HC-05	7
TP4056	7
Led	8
Regulador 7805	8
Software	8
Repositorio del Proyecto	8
4. Diagrama de conexión	9
5. Implementación	9
Sistema embebido	9
Librerías	9
Servo.h	9
Ultrasonic.h	9
Wire.h	10
mpu9050.h	10
Setup()	10
loop()	10
Aplicación Android	11
Actividades	11
Bluetooth	11
Sensores	11
Servicio de conexión Bluetooth	12
Clase Lienzo y dibujo con Canvas	13
DeviceListAdapter	13
6. Funcionamiento	14
Sistema embebido	14
Aplicación android	14
7. Problemáticas	15
Diseño	15
Mecánicos	15
Dirección	15
Traslación	15
Electrónicos	15
Alimentación a componentes sensibles	15
Alimentación Autónoma	15

Alimentación al Sistema General	16
Alimentación a los motores	16
Tabla de Consumos	16
Físicos	16
Determinar aceleración lineal con sensor	16
Determinar Giro con sensor	17
Determinar Coordenadas relativas	17
Filtrar información de los sensores	17
Matemático	18
Procesamiento	18
8. Bibliografía	18

1.Objetivo General del Proyecto

El objetivo del proyecto consiste en construir un dispositivo que deberá ser operado desde un dispositivo móvil conectado inalámbricamente, el cual pueda informar su posición relativa para permitir el seguimiento del mismo en el tiempo.

2.Solución Propuesta

La forma de hallar la posición de un cuerpo a partir de los datos es utilizando el acelerómetro, el procedimiento para hallar la posición con respecto al tiempo puede ser visto como una doble integración de la aceleración : $\iint a(t)dt dt$ donde $a(t)$ es la aceleración. Para estimar la posición usando un dispositivo digital, es necesario aplicar la integración numérica.

3.Hardware utilizado

Sensores

- [HC-SR04](#) x2
- [MPU 6050](#)
- [MPU 9250](#)

Actuadores:

- [Servo SG90](#)
- Motor CC con (reductora)
- Leds x6

Otros Componentes:

- [Arduino UNO](#) ([ATMega 328P](#))
- Protoboard
- [Puente H L293D](#)
- Módulo [Bluetooth HC-05](#)
- [Regulador 7805](#)
- Batería EB-L1G6LLU x2
- Auto a Friccion
- Resistencias varias para leds
- Cables varios para conexionado con el protoboard
- Interruptor

Descripción

Arduino UNO



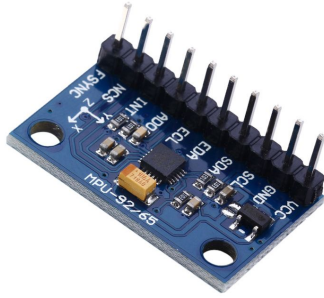
[Arduino Uno](#) es una placa electrónica basada en el microcontrolador [ATmega328P](#). Cuenta con 14 entradas/salidas digitales, de las cuales 6 se pueden utilizar como salidas PWM (Modulación por ancho de pulsos) y otras 6 son entradas analógicas. Además, incluye un resonador cerámico de 16 MHz, un conector USB, un conector de alimentación, una cabecera ICSP y un botón de reset. La placa incluye todo lo necesario para que el microcontrolador realice su tarea, basta conectarla a un ordenador con un cable USB o a la corriente eléctrica a través de un transformador.

Sensor HC-SR04



Este [sensor](#), se maneja con un trigger(disparador) de ultrasonido y un echo. El trigger envía una señal de ultrasonido, esta rebota contra el objeto más próximo y vuelve hacia el dispositivo que lo captura por el echo. Entonces este sensor mide en realidad, el tiempo que tarda en volver la señal del ultrasonido por medio del clock y realizando una operación se puede calcular la distancia, ya que la señal de ultrasonido viaja a una velocidad constante.

Sensor MPU 92/50



El sensor InvenSense [MPU-9250](#) contiene un acelerómetro MEMS, un giroscopio MEMS (MPU-6050) y un magnetómetro MEMS (brújula) en un solo chip junto con un procesador que permite el mezclado de datos.

El giroscopio cuenta con un rango de escala de ± 250 , ± 500 , ± 1.000 , ± 2.000 °/s (dps) y el acelerómetro un rango de ± 2 g, ± 4 g, ± 8 g, y ± 16 g y el magnetómetro $\pm 4800\mu T$.

Servo SG90



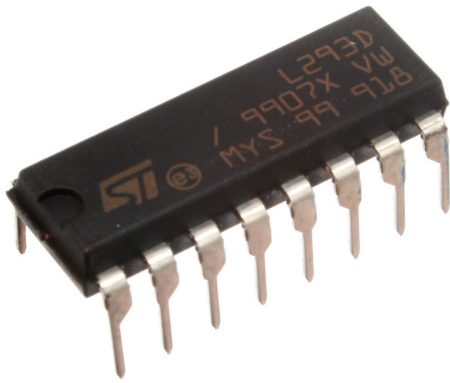
En servo miniatura de gran calidad y diminutas dimensiones cuyos requerimientos de energía son bastante bajos y se permite alimentarlo con la misma fuente de alimentación que el circuito de control.

Motor CC con reductora



Utilizamos un motor CC de lectora de CD para acelerar o desacelerar la marcha del vehículo. Su movimiento se lo trasladamos a la reductora del Auto a fricción a través de un sistema de polea. [Utilizamos un capacitor](#) para aumentar su fuerza al comenzar los movimientos

Puente H L923D



Un [Puente en H](#) es un circuito electrónico que permite a un motor CC girar en ambos sentidos un motor, pero también puede usarse para frenarlo (de manera brusca), al hacer un corto entre las bornas del motor, o incluso puede usarse para permitir que el motor frene bajo su propia inercia, cuando desconectamos el motor de la fuente que lo alimenta

Módulo Bluetooth HC-05



Es un [módulo Bluetooth](#) que tiene la particularidad de que puede ser configurado como maestro o como esclavo. Se lo implementó para la conexión con el dispositivo android bajo el protocolo RS 232 Serial, configurado como esclavo ya que la conexión se realizará con un solo dispositivo Android a la vez.

TP4056



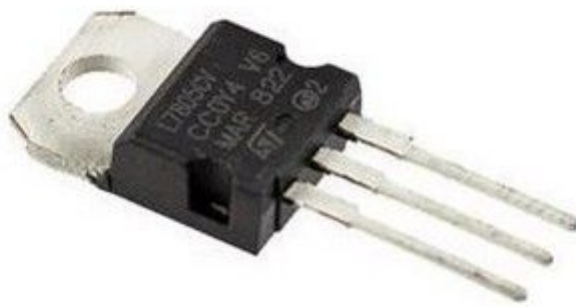
[Módulo](#) que se alimenta de una tensión que puede ser desde 4V a 8V y permite la carga de una batería (en nuestro caso, batería de Litio) aplicando una corriente de hasta 1000 mA a 4.2V.

Led



Utilizamos los leds en nuestro vehículo como actuadores que indican el encendido del vehículo, las luces delanteras y traseras. También combinamos la información del sensor lumínico del celular para que enciendan automáticamente en ambientes de poca luz. También en el caso de detectar cercanía con un obstáculo y dar marcha atrás el vehículo para evitar la colisión.

Regulador 7805



Es un [dispositivo electrónico](#) que tiene la capacidad de regular voltaje positivo de 5V a 1A de corriente. Lo utilizamos para garantizar una fuente de alimentación constante.

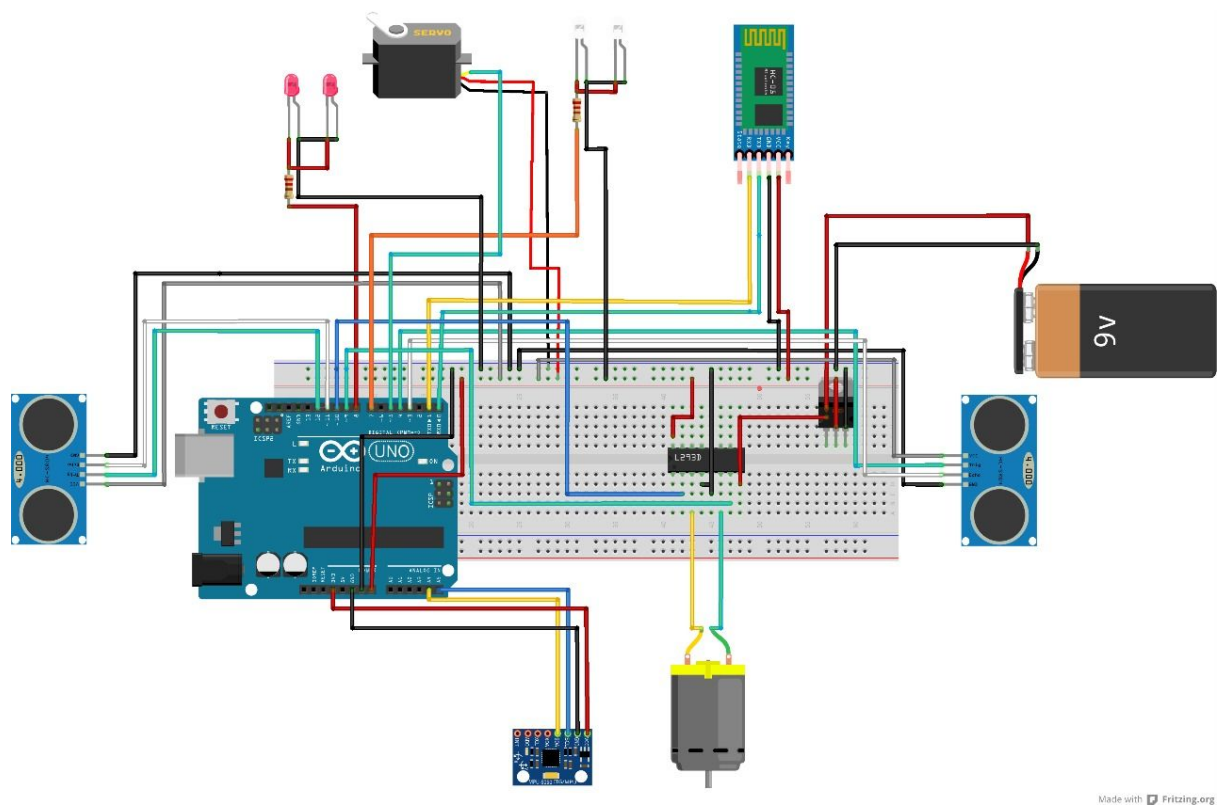
Software

- Android Studio 2.3, JRE 1.8.0
- Arduino 1.8.2

Repositorio del Proyecto

<https://github.com/germangelv/SEAAR>

4. Diagrama de conexión



Made with Fritzing.org

5. Implementación

Sistema embebido

Una vez hecho todas las conexiones pertinentes expuestas en la ilustración anterior, se pasa a implementar el [código](#) en el SE.

Librerías

Servo.h

Se debe definir el pin a utilizar (en nuestro caso elegimos el 5). Declaramos el objeto con `servoMotor` y utilizamos las funciones

`servoMotor.attach(intpinServo)`: inicia el servo para trabajar con el pin indicado.

`servoMotor.write(intangulo)`: desplaza el servo al ángulo especificado.

Ultrasonic.h

Esta librería nos simplifica los cálculos del ultrasonido, pero de igual manera son de sencilla aplicación: $\text{Distancia} = \{(\text{Tiempo entre Trig y el Echo}) * (\text{V.Sonido } 340 \text{ m/s})\} / 2$

Wire.h

Esta biblioteca le permite comunicarse con dispositivos I2C. Con esta librería seremos capaz de conectarnos con el mpu-9250.

mpu9050.h

Biblioteca para obtener los datos de los 3 sensores en crudo. Son 9 ejes, 3 del acelerómetro (ax,ay,az), 3 del giroscopio (gx, gy,gz) y 3 del magnetómetro (mx,my,mz), datos raw.

Setup()

Se inicializa la comunicacion I2C con el dispositivo analogico mpu 9250.

Luego pasa por inicializar el motor poniendo el pin y ángulos mínimo y máximos.

Se define los pines de entrada y salida al arduino.

Y finalmente se inicializa el mpu 9250.

loop()

Único proceso donde se recibe los datos enviados por el bluetooth, y se procesa la entrada que será un char que se guardará en una variable *estado*, donde controlará a través en un switch para cada estado su acción correspondiente

Acción	Estado
ADELANTE	W
REVERSA	S
PARAR	P
DOBLAR HACIA LA DERECHA	D
DOBLAR HACIA LA IZQUIERDA	I
ENDEREZAR	X
APAGAR	O
ENCENDER	F
MAX VELOCIDAD	J
ENCENDER LED DELANTERO	L
APAGAR LED DELANTERO	K

Luego se verifica que no hay obstáculos, si se está moviendo hacia delante verifica que no haya algún obstáculo enfrente, y si está en reversa verifica que no hay obstaculos detras.

Aplicación Android

Actividades

En la aplicación utilizamos un solo activity (MainActivity.class) con su layout asociado (activity_main.xml). En el cual nos conectamos a través de bluetooth al HC-05, manejamos el auto utilizando el acelerómetro y dibujamos las coordenadas (x,y) que obtenidas a partir del procesamiento de los datos obtenidos por el acelerómetro.

Bluetooth

Para poder utilizar el bluetooth declaramos los respectivos permisos en el AndroidManifest:

`"android.permission.BLUETOOTH"`

`"android.permission.BLUETOOTH_ADMIN"` Es necesaria para que la app inicie la detección de dispositivos y controle los ajustes de Bluetooth.

`"android.permission.ACCESS_COARSE_LOCATION"`

`"android.permission.ACCESS_FINE_LOCATION"`

`"android.permission.BLUETOOTH_PRIVILEGED"`

Se importan las siguientes bibliotecas en el main activity

`android.bluetooth.BluetoothAdapter;`

`android.bluetooth.BluetoothDevice;`

BluetoothAdapter representa el adaptador local de Bluetooth.

BluetoothDevice representa un dispositivo bluetooth remoto.

Sensores

Para poder utilizar los sensores tuvimos que implementar en el main activity la interfaz

SensorEventListener para el registro de cambios en el sensor.

Esta interfaz nos obliga a implementar 2 métodos:

onSensorChange() a cual nos permite implementar las acciones a realizar cuando un sensor registra un cambio.

onAccuracyChange() en la cual se implementan las acciones a realizar cuando se cambia la precisión de un sensor. En nuestro caso no lo utilizamos ya que no era de utilidad.

Para poder utilizar cada sensor tuvimos que implementar la clase SensorManager

sensorManager = (SensorManager) getSystemService(**SENSOR_SERVICE**);

y así instanciar cada sensor...

Sensor de proximidad

```
sensorAproximidad = sensorManager.getDefaultSensor(Sensor.TYPE_PROXIMITY);
```

Se utilizó este sensor para cuando detecte una proximidad en el celular menos a 5 cm, se frene el vehículo por completo.

Acelerómetro

```
acelerometro = sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
```

Se utilizó el acelerómetro

Sensor Luminosidad

```
sensorLuminosidad = sensorManager.getDefaultSensor(Sensor.TYPE_LIGHT);
```

Al utilizar esta interfaz y clase el Android Studio nos importa las siguientes bibliotecas

```
android.hardware.Sensor;
```

```
android.hardware.SensorEvent;
```

```
android.hardware.SensorEventListener;
```

```
android.hardware.SensorManager;
```

Servicio de conexión Bluetooth

Con la clase BluetoothConnectionService creamos el servicio de bluetooth que corre en background. Importamos las siguientes clases

```
android.bluetooth.BluetoothAdapter;
```

```
android.bluetooth.BluetoothDevice;
```

```
android.bluetooth.BluetoothServerSocket;
```

```
android.bluetooth.BluetoothSocket;
```

BluetoothSocket representa la interfaz de un socket de Bluetooth. Este es el punto de conexión que permite que una aplicación intercambie datos con otro dispositivo Bluetooth a través de InputStream y OutputStream.

BluetoothServerSocket representa un socket de servidor abierto que recibe solicitudes entrantes.

A fin de conectar dos dispositivos Android, un dispositivo debe abrir un socket de servidor con esta clase. Cuando un dispositivo Bluetooth remoto realice una solicitud de conexión a este dispositivo, el BluetoothServerSocket mostrará un BluetoothSocket conectado una vez que la conexión se acepte.

Dentro de esta clase BluetoothConnectionServer tenemos tres clases que se encargan de establecer la conexión con el HC-05 o con algún otro dispositivo. Estas clases son

AcceptThread, ConnectThread y ConnectedThread, las cuales extienden de la clase Thread.

AcceptThread va a estar corriendo escuchando conexiones entrantes, corre hasta que una conexión es aceptada. El HC-05 está diseñado para estar parado acá, en el acceptThread, esperando que algún dispositivo se quiera conectar. En este caso nuestro celular.

Los dos dispositivos van a estar con el acceptThread corriendo, hasta que el connectThread de alguno de ellos arranque, en este caso nuestro cel, y agarre el socket y se conecte a él. Este thread se ejecuta al intentar realizar una conexión de salida con el otro dispositivo. Funciona hasta que la conexión tiene éxito o falla

El ConnectedThread que es el responsable de mantener la conexión Bluetooth, enviando data y recibiendo informacion entrante a través de los input/output streams respectivamente.

Clase Lienzo y dibujo con Canvas

La clase Lienzo es una clase que extiende de view, dicha view es utilizada en activity_main.xml para poder graficar en ella media el método “dibujar” declarado en la clase. El cual grafica un punto en base a las coordenadas obtenidas por el acelerómetro del arduino.

```
public void dibujar(float x, float y){  
    drawCanvas.drawPoint(x,y,drawPaint);  
    invalidate();  
}
```

El método invalidate nos permite pintar el lienzo, e ir “refrescando” cada vez que recibe una nueva coordenada

En dicha clase se implementaron las siguientes librerías que hacen posible la utilización de clases y métodos para dibujar.

```
android.graphics.Bitmap;  
android.graphics.Canvas;  
android.graphics.Paint;
```

Con el método setupDrawing() configuramos el área sobre la que pintar y el pincel, en nuestro caso drawpaint, como el color del mismo y el grosor del trazo por ejemplo.

DeviceListAdapter

La clase DeviceListAdapter, es un Adapter. Un objeto Adaptador actúa como puente entre un AdapterView (en nuestro caso, device_adapter_view.xml) y los datos de una Vista (en nuestro caso la ListView de los dispositivos Bluetooth). El adaptador permite el acceso a los elementos de datos, éste también es responsable de crear una vista para cada elemento en la colección de datos.

Para poder trabajar con los Adapter que veremos en esta entrega, tenemos que realizar los siguientes pasos:

1. Definir el estilo de nuestra lista a través de un layout (XML), en nuestro caso `device_adapter_view.xml`
2. Crear una clase (Adapter) que extienda de `ArrayAdapter<?>`, en donde ? puede ser un objeto o un tipo de dato. En nuestro caso `BluetoothDevice` que son los dispositivos remotos que se van detectando.
3. Agregar un Tag o elemento `GridView` o `ListView`(como en nuestro caso) según sea el caso al layout de la actividad en la cual deseamos incluir el listado; si deseamos que nuestro listado tenga forma de lista: Debemos utilizar `ListView`; si deseamos que nuestro listado sean cuadrículas: Debemos utilizar `GridView`.
4. Invocar al Adapter desde una Activity.

6. Funcionamiento

Sistema embebido

El sistema embebido debe responder a las órdenes enviadas desde la aplicación Android (avance, detención, retroceso, giro) al mismo tiempo que le envía a la misma los datos como:

- Datos sobre su desplazamiento en forma de valores de x e y.
- Presencia de obstáculo delante o detrás del vehículo.

Aplicación android

La aplicación es la encargada de establecer la conexión entre el dispositivo Android y el sistema embebido vía Bluetooth, debiendo seleccionar a éste último de entre los dispositivos que aparezcan dentro del alcance.

Al momento de establecerse la conexión, la aplicación comienza a recibir la información referente al desplazamiento del vehículo y, a medida que éste se desplaza, grafica la trayectoria recorrida por el mismo tomando como referencia la posición en la que se produjo la conexión como punto inicial.

Sumado a ésto, desde la aplicación se le envían las órdenes correspondientes al encendido del vehículo, avance, retroceso, giro o detención utilizando la inclinación del dispositivo (mediante la implementación del acelerómetro del mismo).

Como función adicional, la aplicación detecta la disminución de la luz ambiente mediante el sensor lumínico del dispositivo (si es que éste dispone de uno) y, llegado a cierto nivel de luz, procede a enviar la orden al vehículo de que encienda las luces delanteras.

7. Problemáticas

Diseño

En un principio tomamos el diseño de prototipos tales como el [Charly F1 1.2](#) para la carcasa y el giro pero encontramos la dificultad de no poder centrar las ruedas de direccionamiento. Posteriormente nos basamos en el [Auto RC](#) y creamos la estructura, pero luego al tener que utilizar la protoboard y no poder diseñar un circuito o shield para la arduino uno, lo descartamos.

Por último nos basamos para la entrega en el prototipo del [Auto RC con conexión Bluetooth](#) y la mecánica del giro de las ruedas en el [Auto F1](#)

Mecánicos

Dirección

El sistema de dirección fue parte de la problemática del Diseño en todo momento dado que posteriormente solo con un giro suave podríamos obtener información precisa del giróscopo del IMU. Para ello nos basamos en el [Auto F1](#). Utilizamos el Servo para centrar las ruedas y girarlas y el método descrito en la referencia para trasladar el movimiento del Servo a las ruedas.

Traslación

Para la traslación utilizamos un motor CC de lectora de CD con su reductora. Dado el tamaño de la misma optamos por utilizar la reductora que incorpora el Auto a Friccion con un diseño de polea para transmitirle el giro del mismo

Electrónicos

Alimentación a componentes sensibles

Dada la necesidad de alimentación de los sensores, la arduino uno y los leds se utilizó un regulador 7805 para que los 5V de alimentación a menos de 1 amperio sea estable.

Alimentación Autónoma

Para la alimentación autónoma tomamos la decisión de usar baterías de litio de celular, específicamente dos EB-L1G6LLU dado que ofrecen en serie 2100mA a 7.4V y en paralelo 4200mA a 3.7V.

Alimentación al Sistema General

Para cargar cada batería de litio utilizamos un [TP4056](#) y un circuito serie- paralelo con un switch que nos permite desconectar los componentes sensibles, los motores para cargar individualmente o en conjunto ambas baterías y también nos permite desconectar los [TP4056](#) para alimentar en serie con ambas baterías todos los componentes

Alimentación a los motores

Para la alimentación de los motores utilizamos un [Puente H](#) conectado directamente a la alimentación de las baterías que nos permite utilizar su máxima potencia ($\approx 8.4V$, $\approx 2A$) y administrar la misma desde la arduino con pines PWM

Tabla de Consumos

La siguiente tabla representa los consumos estimados de todos los componentes bajo las peores condiciones (máximo consumo de motores y máximo procesamiento del [ATmega328P](#))

Cant.	Componente	Voltaje	Amperaje	Potencia	Total
1	Arduino UNO	5V	46mA	0.23W	0.23W
1	Bluetooth HC-05	5V	50mA	0,25W	0,25W
1	Puente H L293D	5V	30mA	0,15W	0,15W
1	Motor CC	6V	600mA	3,6W	3,6W
1	Servo SG90	5V	650mA	3,25W	3,25W
1	MPU	3.3V	4mA	0,013W	0,013W
2	Ultrasonido HCSR04	5V	15mA	0,075W	0,15W
6	Led	3V	20mA	0,06W	0,36W
Total					7,77W
2	Batería EB-L1G6LLU	3,7	2100mA	7,77	15,54W
Margen de consumo en el peor caso					7,77W

Físicos

Determinar aceleración lineal con sensor

Para determinar la aceleración lineal del auto utilizamos un [MPU 6050](#) y posteriormente el [MPU 9250](#) dado que incorpora un acelerómetro para tal fin.

Los problemas que hemos tenido aquí son que las mediciones del acelerómetro tiene ruido y errores de todo tipo, estas reducen la precisión de los resultados. Algunos son producto de

las condiciones ambientales (como la temperatura, campos magnéticos, etc.) mientras que otros se deben al muestreo y procesado de la señal.

Determinar Giro con sensor

Para determinar el giro del auto se utilizó la información del giróscopo incorporado en el [MPU 6050](#) y posteriormente el [MPU 9250](#).

A diferencia del acelerómetro, da las medidas con mucha precisión. Pero al realizar los cálculos del ángulo es inevitable que se produzca un pequeño error, que con el tiempo va acumulándose hasta que se defasa el ángulo. Este error es el denominado drift.

Error de la inclinación

La orientación afecta a la salida del acelerómetro y consecuentemente a la estimación de la posición basadas en esos datos.

Determinar Coordenadas relativas

Para la determinación de las coordenadas relativas investigamos sobre los métodos de [integración numérica](#) y recomendamos con datos viables la utilización del [método de trapecios](#) sobre [simpson](#) según la [Tesis del cuadricóptero](#) y la [Tesis sobre Localización peatonal](#)

Filtrar información de los sensores

Cuando se estableció el proyecto, nos indicaron el requisito de representar en el dispositivo móvil la posición relativa del auto. Cuando establecimos los objetivos empezamos a priorizar las tareas más complejas intentamos determinar con exactitud la aceleración lineal del auto que integrada dos veces nos [ofrecería la posición relativa](#).

Para ello utilizamos la información del sensor [MPU 6050](#) y comenzamos a investigar cómo funciona el mismo. La información RAW que ofrece el [MPU 6050](#) y el [MPU 9250](#) por defecto en el acelerómetro tiene varios factores de error. Estos son el [ruido](#) que se elimina utilizando el [filtro de Kalman](#), factores de [corrección de la gravedad](#) que se solucionan calibrando y compensando en tiempo real la información del sensor, mezclando su información con la del giróscopo según las [Tesis del cuadricóptero](#), la [Tesis sobre Localización peatonal](#) y la [Tesis de análisis ergonómico de riesgo laboral](#)

Según la experiencia de estudio desarrollada durante las etapas del [proyecto del área de detección de movimiento](#) usando sensores inerciales es recomendable la utilización de filtro especial de carácter estadístico, llamado [filtro de Kalman](#) o filtro de [Kalman Extendido](#), lamentablemente el uso del filtro comprende [uso de matemática de alto nivel](#) lo que hubiese demandado [mucho tiempo de aprendizaje](#) y desarrollo, algo complicado dadas las diferentes ramas de este proyecto y el tiempo limitado para su realización.

No se pudo obtener correctamente los datos de la posición en el sistema embebido y, debido a ésto, no es posible dibujar la trayectoria recorrida por el vehículo en la aplicación.

Matemático

Al intentar a realizar la doble integral de forma numérica nos surgió el inconveniente de que método usar. Por trapezoide o por Simpson.

La forma trapezoidal proporciona más errores que Simpson. Pero Simpson el número de puntos N tiene que ser impar para la regla de Simpson, lo cual es un claro inconveniente con datos de muestreo.

Por eso se aconseja utilizar trapezoidal una vez obtenido los datos filtrado de la aceleración.

Procesamiento

La complejidad de los cálculos matemáticos, de los filtros para mejorar la precisión de los datos y de las integrales en tiempo real para obtener la posición implican una capacidad de procesamiento superior a la que posee [ATmega328P](#) (16Mhz) una solución posible sería utilizar un procesador Cortex ARM que apoye estos cálculos según la librería de [Kris Winner](#).

8. Bibliografía

Bibliografía que se utilizó para la construcción de la Aplicación Android

Sitio Android developer, sección de Bluetooth

<https://developer.android.com/guide/topics/connectivity/bluetooth.html?hl=es-419>

Tutorial para habilitar y deshabilitar Bluetooth

<https://www.youtube.com/watch?v=y8R2C86BIUc>

Tutorial para habilitar descubribilidad del dispositivo

https://www.youtube.com/watch?v=QVD_8PrvG4Q&t=2s

Tutorial para buscar dispositivos

https://www.youtube.com/watch?v=hv_-tX1VwXE

Tutorial sobre cómo emparejar los dispositivos

<https://www.youtube.com/watch?v=YJ0JQXcNNTA&t=10s>

Tutorial para enviar y recibir datos a través de bluetooth

https://www.youtube.com/watch?v=Fz_GT7VGGaQ&t=3s

Tutorial que usamos para aprender a dibujar en canvas

<https://www.youtube.com/watch?v=hPYy5fidpgs>

Tutorial con el que aprendimos el manejo de sensores

<https://www.youtube.com/watch?v=-yDx9Bjs970>

Bibliografía que se utilizó para el arduino

Datasheet Arduino Uno

<https://www.farnell.com/datasheets/1682209.pdf>

Datasheet Microcontrolador ATmega 328P

http://www.atmel.com/Images/Atmel-42735-8-bit-AVR-Microcontroller-ATmega328-328P_Datasheet.pdf

Datasheet Regulador 7805

<https://www.sparkfun.com/datasheets/Components/LM7805.pdf>

Datasheet Sensor Ultrasonico HC-SR04

<http://www.micropik.com/PDF/HCSR04.pdf>

https://docs.google.com/document/d/1Y-yZnNhMYy7rwhAgyL_pfa39RsB-x2qR4vP8saG73rE/edit

Datasheet Modulo Bluetooth HC-05

<http://www.electronicaestudio.com/docs/istd016A.pdf>

<http://www.electronicoscaldas.com/modulos-rf/452-modulo-bluetooth-hc-05.html>

Datasheet Sensor MPU 6050

<https://www.invensense.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf>

Datasheet Sensor MPU 9250

<https://www.invensense.com/wp-content/uploads/2015/02/PS-MPU-9250A-01-v1.1.pdf>

Datasheet Servo SG90

<http://www.micropik.com/PDF/SG90Servo.pdf>

<https://www.minitronica.com/producto/tower-pro-sg90/>

Datasheet Puente H L293D

<http://www.ti.com/lit/ds/symlink/l293.pdf>

Datasheet TP4056

<https://dlnmh9ip6v2uc.cloudfront.net/datasheets/Prototyping/TP4056.pdf>

Prototipo Auto RC con conexión Bluetooth

<http://www.instructables.com/id/Arduino-Bluetooth-RC-Car-Android-Controlled/>

Prototipo Charly F1 1.2

<https://www.youtube.com/watch?v=2V7IKw5eCoU>

Prototipo Auto RC para basarnos en dirección y tracción

<https://www.youtube.com/watch?v=PHdtGlrSu4o>

Prototipo de Sistema de Dirección para Auto F1

<https://www.youtube.com/watch?v=mTw6DAnpZ9k>

Calcular desplazamiento usando acelerometro y giroscopio (MPU6050)

<https://stackoverflow.com/questions/26476467/calculating-displacement-using-accelerometer-and-gyroscope-mpu6050>

Formas simples para calibrar magnetometro po Kriss Winner

<https://github.com/kriswiner/MPU6050/wiki/Simple-and-Effective-Magnetometer-Calibration>

Utilizando Datos Serie del IMU CurieIMU para Arduino/Genuino 101

<https://www.arduino.cc/en/Tutorial/Genuino101CurieIMURawImuDataSerial>

Explicación del Método del Filtro de Kalman

http://www.convict.lu/htm/rob/imperfect_data_in_a_noisy_world.htm

http://www.bccr.fi.cr/investigacioneseconomicas/metodoscuantitativos/Filtro_de_Kalman.pdf

Clases que explican el funcionamiento del Filtro de Kalman

<https://www.youtube.com/watch?v=-7fsC2fCw-8>

Explicacion simple del Filtro de Kalman

<http://www.bzarg.com/p/how-a-kalman-filter-works-in-pictures/>

Filtro de Kalman extendido

<http://bionanouni.wdfiles.com/local--files/teaching-mt227-horario-2009ii/Clase13-01.pdf>

Filtro de Kalman aproximado unidimensional

<https://github.com/bachagas/Kalman>

Ejemplo Filtro de Kalman

<http://iaci.unq.edu.ar/materias/control2/web/tp/Kalman.pdf>

Filtro de Kalman en Java para procesar desde el celular

https://imagej.nih.gov/ij/plugins/download/Kalman_Filter.java

Métodos de Integración y Derivación Numérica

<http://forum.arduino.cc/index.php?topic=278844.0>

<https://previa.uclm.es/profesorado/evieira/ asignatura/meccomp/book/sistemas/integracion/integracion.pdf>

Método de Simpson en C

<https://elrincondelc.com/foros/viewtopic.php?t=17731>

<http://www.dailyfreecode.com/code/simpsons-13-rule-2390.aspx>

Determinar la orientación con arduino y el IMU MPU-6050

<https://www.luisllamas.es/arduino-orientacion-imu-mpu-6050/>

HC-SR04 Arduino – Tutorial Con Código Incluido

<http://www.educachip.com/hc-sr04-arduino-tutorial/>

Librería de Jeff Rowberg para MPU 6050

<https://github.com/jrowberg/i2cdevlib/tree/master/Arduino/MPU6050>

Librería de Kris Winer para MPU 9250

<https://github.com/kriswiner/MPU9250>

Explicación Sensores Acelerómetro Giróscopo

<http://www.prometec.net/imu-mpu6050/>

Manejo de potencia para motores con el integrado L293D

http://robots-argentina.com.ar/MotorCC_L293D.htm

Tesis sobre Localización peatonal para ambientes y la problemática de los mismos

https://www.cl.cam.ac.uk/research/dtg/www/files/publications/public/abr28/ojw28_thesis.pdf

Condensador Motor CC

<http://smf.edaboard.com/topic-11084301.0.html>

Crear Reductora con Motor CC

<https://www.youtube.com/watch?v=xKupqDPYvtY&t=73s>

Usar arduino con los IMU de 9dof MPU-9150 y MPU-9250

<https://www.luisllamas.es/usar-arduino-con-los-imu-de-9dof-mpu-9150-y-mpu-9250/>

Exposicion de defectos y correccion de los mismos

<https://www.youtube.com/watch?v=C7JQ7Rpnw2k>

Tesis sobre Implementación de un dron cuadricóptero con Arduino

<http://academica-e.unavarra.es/bitstream/handle/2454/19208/TFG%20Jose%20Etxeberria.de>

Análisis ergonómico de riesgo laboral

<http://dspace.ups.edu.ec/bitstream/123456789/7508/1/UPS-CT004422.pdf>

Librería para MPU 9250 de Kris Winner

<https://github.com/kriswiner/MPU9250>