



Trabajo Práctico Obligatorio: Informe

Materia: Informática II - Ing. Electrónica - U.T.N-F.R.B.A.

Docente titular: Ing. Nahuel Gonzalez

Ayudante de primera: Ing. Lisandro Sugezky

Ayudantes de segunda: Nicolás Campos

Federico Bua

Jefe de cátedra: Ing. Marcelo Giura

Integrantes:

Coiro, Mateo

Golob, Lautaro

Liaño, Lucas

Lovallo, Alejo

Repetto, Facundo



Estabilizador de cámara Gimbal

El propósito de este documento es resumir de manera informativa y práctica los aspectos técnicos del proyecto integrador.

- [Estabilizador de cámara Gimbal](#)
- [Objetivo](#)
- [Descripción](#)
- [Marco Teórico](#)
 - [Actuadores:](#)
 - [Control:](#)
 - [Funcionamiento:](#)
 - [Filtros:](#)
- [Recursos Complementarios](#)
 - [Impresión 3D](#)
 - [Módulos externos](#)
 - [Placa complementaria](#)
 - [Materiales utilizados para las placas](#)
- [Problemas y Soluciones](#)
 - [Máquinas de estados:](#)
 - [Interfaz gráfica:](#)
 - [I2C:](#)
 - [Traspaso del código a C++:](#)
 - [Sistema de Control:](#)
 - [Filtros:](#)

[Conclusiones](#)



Objetivo

Realizar un proyecto integrador que capitalice los conocimientos aprendidos durante la materia de Ingeniería Electrónica de segundo año, Informática II.

Los integrantes del equipo de trabajo hemos decidido realizar un estabilizador de cámara de tres ejes, más conocido como Gimbal en la industria. Será desarrollado específicamente para una cámara GoPro Hero 4.

Descripción

La idea fuerza del proyecto surgió gracias a las nuevas tecnologías utilizadas para la transmisión de espectáculos en grandes espacios abiertos, tales como recitales o encuentros deportivos en grandes estadios. El método utilizado inicialmente se basó en la utilización de cables tensados y rieles en los cuales mediante un soporte, los elementos de filmación serían comandados remotamente. Sin embargo, lejos de ser una solución, formuló un problema aún mayor; a una determinada altura, los factores climáticos suponen un factor crítico a tener en cuenta que modifica la precisión con la cual se puede manipular la herramienta de filmación.

Este inconveniente fue lo que motivó nuevos avances en materia de transmisiones. Referido al problema inicial se propuso soportes acoplados en los extremos de las cámaras para poder solventar el problema del viento y condiciones climáticas adversas.

A su vez, esta corriente tecnológica impulsó avances en la transmisión de deportes extremos e individuales, para poder capturar experiencias que hasta el momento no podían ser alcanzadas. A pesar de que no se trataba de deportes en los cuales la altura fuese un factor primordial, la velocidad, los cambios bruscos de movimientos y repentinos presentaban el mismo problema descrito para eventos masivos. La solución, sin embargo, esta vez fue diferente. Mediante la conjunción de elementos que se detallarán a lo largo del documento, tanto en software y hardware, se logró el elemento que nuestro proyecto intentará replicar, un estabilizador de cámara de tres ejes mediante motores y algoritmos de control. Utilizaremos las cámaras para los cuales fueron diseñados específicamente este elemento en primer lugar, las cuales pertenecen a la empresa goPro. En la actualidad es posible utilizarlas en diferentes cámaras siempre y cuando se tenga en consideración el peso y el tamaño de la cámara.

Marco Teórico

Centraremos nuestro análisis en tres ejes fundamentales para la comprensión de los fundamentos y complejidades del proyecto: Actuadores, Control-Filtros:

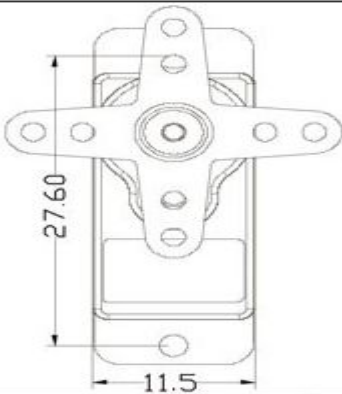
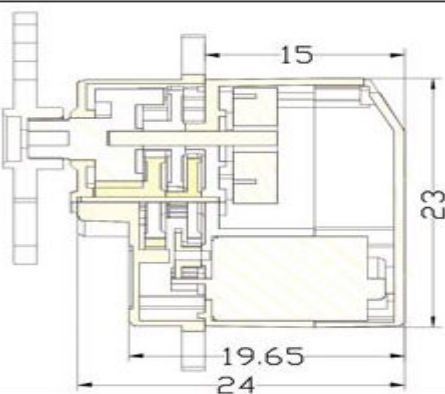


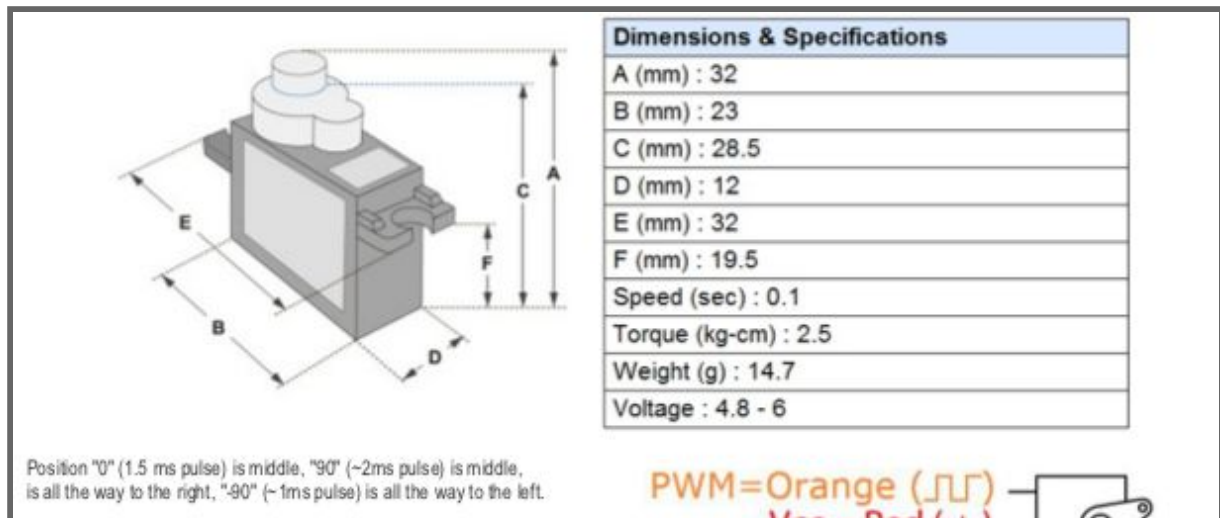
Actuadores:

Utilizamos dos tipos de servomotores. Por un lado uno de los más comunes y usados en la industria, el SG90, y por el otro el EMAX ES08MD. La elección de los motores estuvo fundada en; En primer lugar, el SG90 por ser uno de los motores más utilizados y accesibles en el mercado, y por otro lado, el EMAX ES08MD por ser un servomotor de control digital que permite tener una mayor resolución a la hora de seleccionar la posición del motor. Dejamos documentado las fotos de los motores y sus correspondientes especificaciones técnicas:

EMAX ES08MD



Mini SERVO (ES08MD) SERIES			
SERVO ES08Md SPECIFICATIONS			
			
Operating Voltage:		4.8-6.0V	
STD Direction:		Counter Clockwise / Pulse Traveling 1500 to 1900usec	
Stall Torque:	4.8V	1.6 Kgf.cm(21.0 oz.in)	
Operating Speed:	4.8V	0.12 Sec/60° at no load	
Stall Torque:	6.0V	2.0 Kgf.cm(28.0 oz.in)	
Operating Speed:	6.0V	0.10 Sec/60° at no load	
Size:		23X11.5X24 mm	
Weight:		13 g	
Plug Available		FUT ;JR	
Other		Digital; Metal	



Control:

Se utilizó un mecanismo de control ampliamente conocido y utilizado en sistemas de control industrial por sus siglas: P.I.D (Controlador Proporcional, Integral y Derivativo). El mismo se basa de un sistema de realimentación, el cual calcula la desviación o error entre un valor medio y un valor deseado. El controlador puede proveer una acción de control adaptada a los requerimientos del proceso en específico. La respuesta del controlador puede describirse en términos de respuesta del control ante un error, el grado el cual el controlador sobrepasa el punto de ajuste, y el grado de oscilación del sistema. Nótese que el uso del PID para control no garantiza un control óptimo del sistema o la estabilidad del mismo.

Funcionamiento:

Para el correcto funcionamiento se necesita al menos:

- Sensor
- Controlador
- Actuador

A su vez, los tres componentes de un controlador PID son:

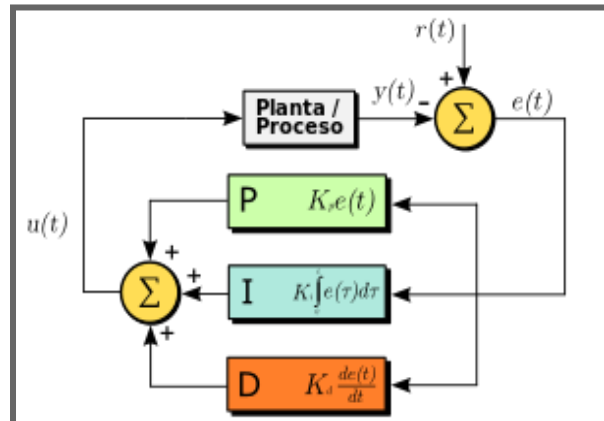
- Parte Proporcional --- constante proporcional.
- Acción Integral ----- tiempo Integral.
- Acción Derivativa ---- tiempo derivativo.

El sensor proporciona una señal analógica o digital al controlador, la cual representa el *punto actual* en el que se encuentra el proceso o sistema.

El controlador recibe una señal externa que representa el valor que se desea alcanzar, en

nuestro caso el **setPoint**, el cual es de la misma naturaleza y tiene el mismo rango de valores que la señal que proporciona el sensor.

El controlador resta la señal de punto actual a la señal de punto de consigna, obteniendo así la señal de error, que determina en cada instante la diferencia que hay entre el valor deseado (consigna) y el valor medido. La señal de error es utilizada por cada uno de los tres componentes del controlador PID. Las tres señales sumadas, componen la señal de salida que el controlador va a utilizar para gobernar al actuador. La señal resultante de la suma de estas tres se llama **variable manipulada** y no se aplica directamente sobre el actuador, sino que debe ser transformada para ser compatible con el actuador utilizado.



Se pretenderá lograr que el bucle de control corrija eficazmente, y, en el mínimo tiempo posible los efectos de las perturbaciones.

Filtros:

Debido a la implementación de la I.M.U (Inertial measurement unit), es necesario aplicar filtros para la generación de ángulos de euler. En este caso en particular, utilizamos un filtro complementario que fusiona el ángulo generado a partir de la integración de la velocidad angular en un período AT junto con el ángulo generado a partir de cálculos trigonométricos sobre el vector gravedad, medido desde el acelerómetro. Por otro lado, aplicamos un filtro de media móvil exponencial aplicado como pasa-bajo. Esto se debe a que los datos tomados de la IMU tienen fluctuaciones de alta frecuencia.

Proceso

El proceso para la organización y realización del proyecto fue diagramado por el cuerpo docente a cargo de la materia y posteriormente nuestro equipo lo modificó de acuerdo a las necesidades de nuestro proyecto en cuestión:

1. Presentación de la idea fuerza.
2. Establecimiento de limitaciones y alcance del proyecto coordinadas con el cuerpo docente:
 - Desarrollo completo del modelo mecánico, tanto diseño como simulación e impresión del chasis del gimbal.



- Desarrollo del firmware que contará con funciones que posibiliten la rotación del soporte en torno a sus propios ejes y un modo de uso que permitirá controlar el sistema para mantener constante la actitud del soporte central de uno de los tres ejes.
- Desarrollo de una interfaz gráfica que pueda controlar y visualizar la actitud del dispositivo.

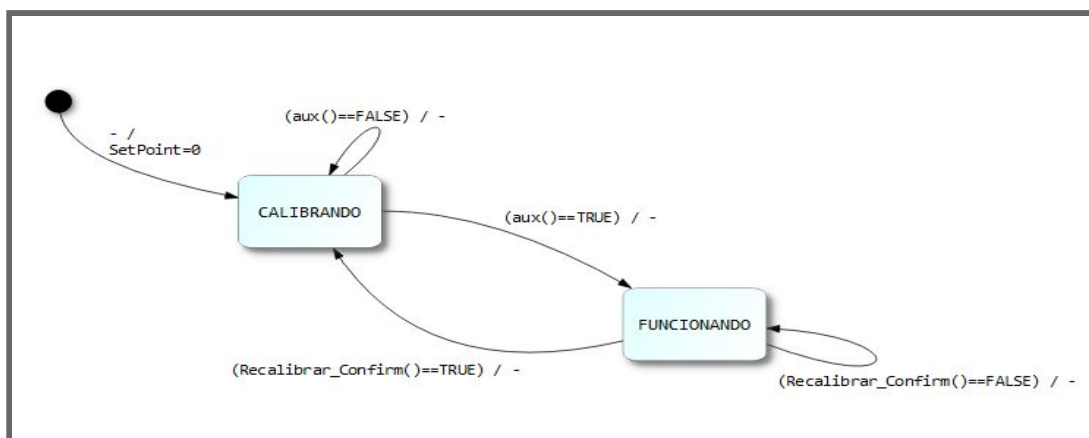
3. Lógica del proyecto mediante máquinas de estado.

Inicialmente se diagramó una lógica que contemplaba tres máquinas de estados

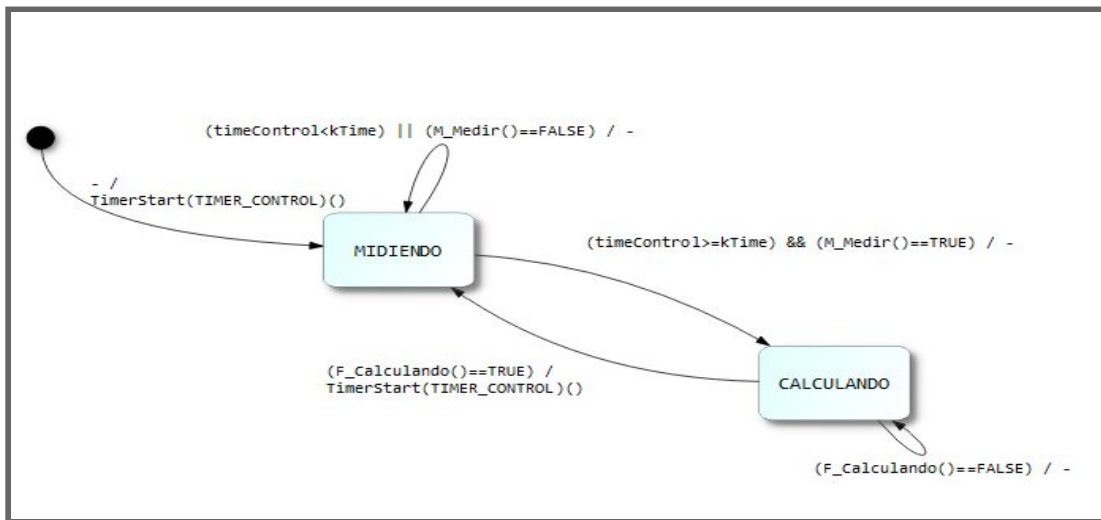
- Máquina general
- Máquina de control
- Máquina de transmisión de datos

Máquina General: La máquina general se encarga de las tareas principales que debe cumplir el sistema. La misma cuenta, a esta altura del proyecto, con dos estados FUNCIONANDO y CALIBRANDO. La máquina va a iniciar en el estado CALIBRANDO y va a ejecutar las rutinas pertinentes a la calibración. Una vez haya logrado calibrarse, función que actualmente no contempla fallos, genera una transición hacia FUNCIONANDO.

En el estado funcionando, el programa va a ejecutar la máquina de control. Luego va a generar una transición hacia calibrando si se cumplen los requisitos de Recalibrar_confirm() = aux. Esto va a suceder ya sea porque se solicita una recalibración vía comunicación Serie o porque se ha congelado el micro.

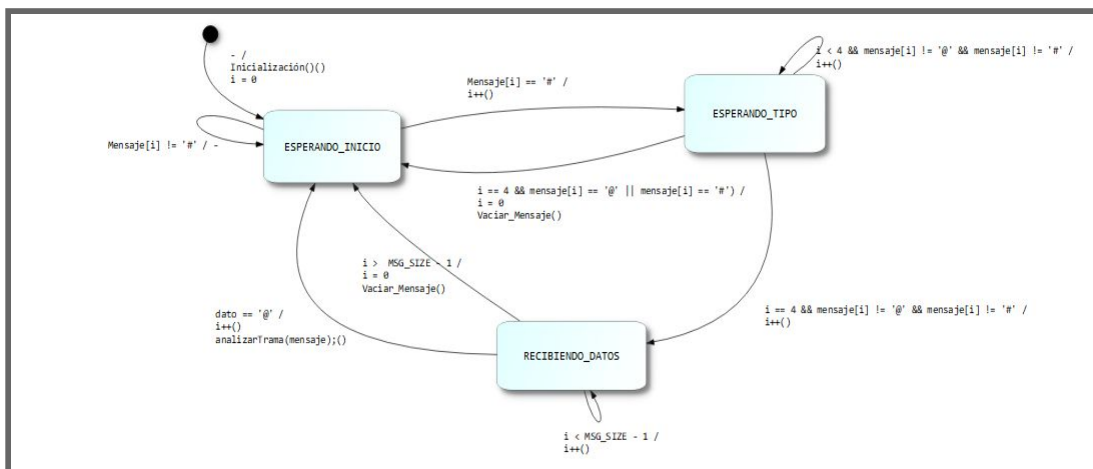


Máquina Control: La máquina de control se encarga de alternar entre mediciones y cálculo de datos. La transición es temporizada y depende de una constante de control $kTime$. Dentro del estado MIDIENDO se realizan cuantas mediciones sean posibles de datos y se procesan obteniendo ángulos de Euler. Cuando se genera la transición al estado CALCULANDO, se ejecutan los cálculos del sistema de control, obteniendo como resultado una salida "output" que debe incrementarse en la variable "actual" (correspondiente a la función `F_Ejecutando()`).



Máquina transferencia de datos: Esta máquina cuenta con 3 estados que se ocupan del correcto desempeño de la recepción de mensajes vía puerto serie. Estos tres estados son ESPERANDO_INICIO, ESPERANDO_TIPOS y RECIBIENDO_DATOS.

El estado inicial es ESPERANDO_INICIO. Allí se espera la llegada de un carácter '#'. Cuando eso ocurre, empieza la recepción de la trama y estado pasa a ser ESPERANDO_TIPO. En este nuevo estado se espera la llegada de 3 caracteres que definen el tipo de mensaje. Si la cantidad de caracteres que llegan hasta la llegada de un '@' es distinta a 3 unidades, entonces se descarta la trama y se vuelve al estado ESPERANDO_INICIO. Cuando la cantidad de caracteres recibidos supere 3 unidades sin la llegada de un '@', entonces el estado se modifica a RECIBIENDO_DATOS. Mientras la máquina se encuentre dentro de RECIBIENDO_DATOS, se esperará la llegada de un nuevo carácter y a continuación un '@'. Si no se cumplen dichas condiciones, se descarta la trama. Si se cumplieran las condiciones, se llama a la función `AnalizarTrama()`, quien verifica el tipo de dato y ejecuta acciones conforme al tipo de mensaje.



4. Modelizado del funcionamiento mediante la utilización de la librería de la cátedra conjuntamente con los periféricos incluidos en la placa.
5. Presentación de la interfaz gráfica del proyecto
6. Presentación de avances preliminares.
7. Presentación Final.

Se estableció una forma de trabajo en un principio conjunta y que posteriormente se dividiría por áreas para cumplir con los tiempos solicitados por la cátedra. A su vez, desde un primer momento incorporamos a nuestro modo de trabajo dos herramientas necesarias y sumamente útiles para cualquier tipo de proyecto;

- Documentación mediante **Doxygen**.
- Software de gestión de versiones de código, en nuestro caso mediante **GitHub**.

Recursos Complementarios

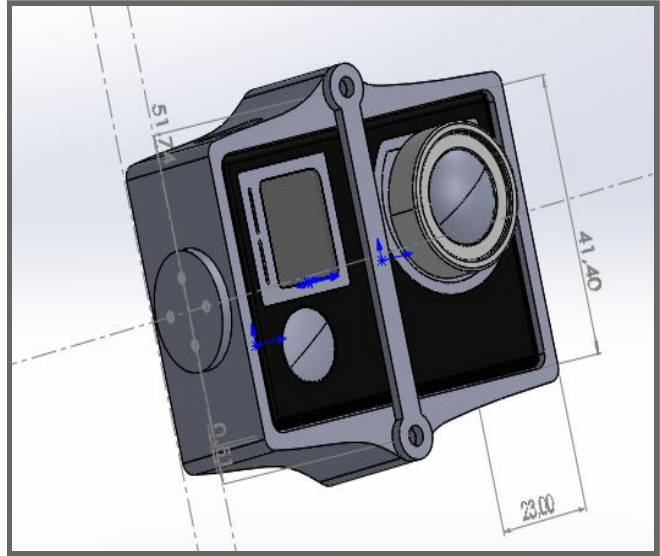
Impresión 3D

Era desde un primer momento, un aspecto a tener en cuenta desde que tomamos la decisión de avanzar en el proyecto. El estabilizador que se produce a nivel industrial utiliza materiales que eran perfectamente replicables a través de la impresión 3D. Por lo tanto, diseñamos cuatro piezas para que la cámara que utilizaremos, una GoPro Hero4 pueda ser utilizada en el proyecto.

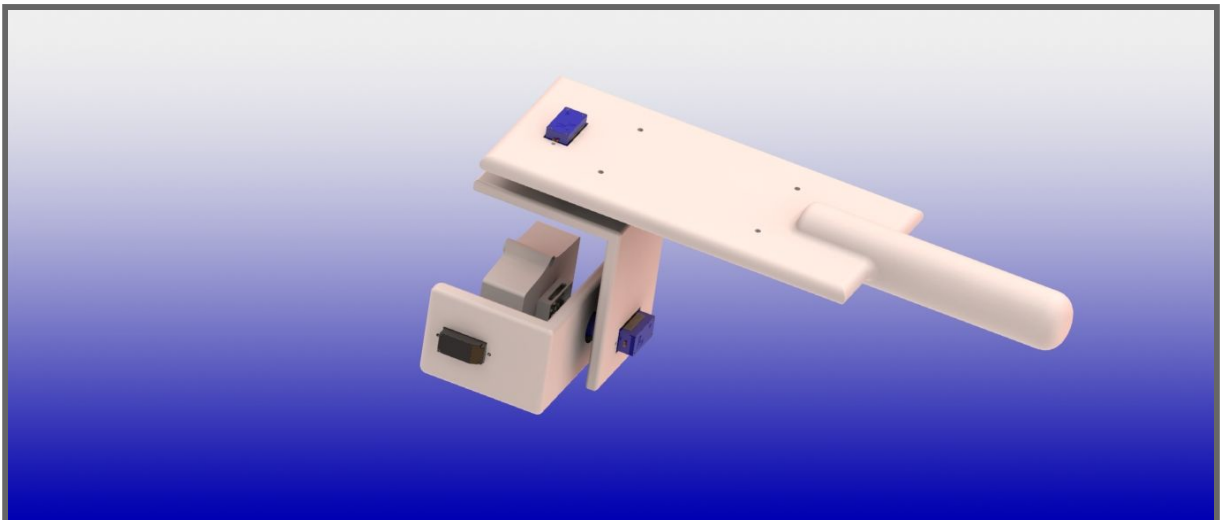
Utilizamos el programa SolidWorks para el diseño en 3D de las piezas y para poder realizar el renderizado para que la impresora 3D comprenda el modelo realizado el software Cura.

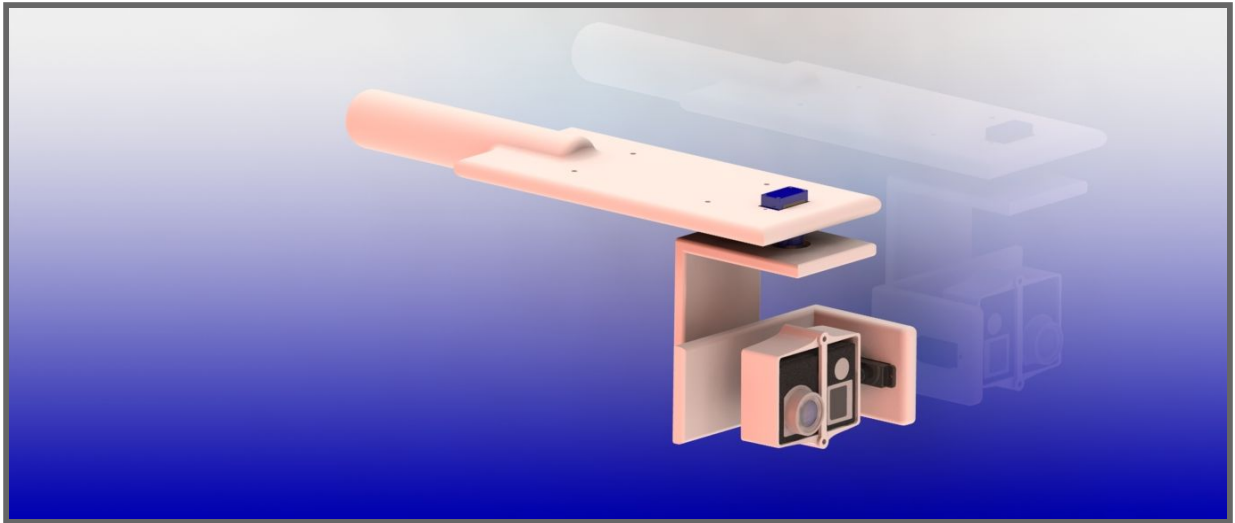
Descripción de las piezas realizadas:

- Caja a medida para que colocar la cámara: el propósito de esta pieza es darle un alojamiento a la cámara.
- Tabique para sostener la cámara: por cuestiones de diseño, la realización de una tapa para que la cámara no se caiga era difícil de alcanzar por lo que se implementó un tabique en el el frente de la caja, a la cual se agujereó en el medio de la misma tanto en la parte superior como inferior de la misma.



- Dos soportes los cuales forman ángulos perpendiculares entre sí y permiten que la sujeción de los servomotores para luego generar rotaciones.
- Mango para la manipulación del dispositivo y a su vez cumple la función de sujetar la placa a la base del mango.





Módulos externos

Para realizar mediciones correspondientes recurrimos a uno de las unidades de medición inercial más conocidas como lo es el MPU-9250. El mismo es un integrado que cuenta con un giroscopio, un acelerómetro y un magnetómetro. También tiene interfáz para comunicación mediante tanto I2C como SPI.

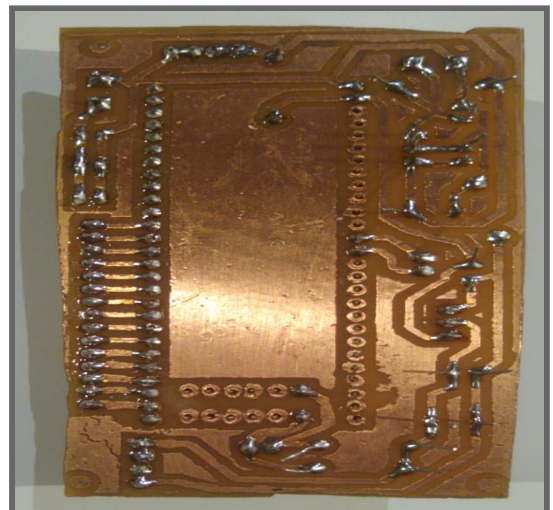
Por otro lado, para generar una comunicación inalámbrica con Qt optamos por utilizar el módulo HC-06. Este es un módulo muy utilizado en el mundo Arduino capaz de comunicarse mediante Bluetooth con distintos dispositivos. Para conectarnos con él, solo tendremos que escribir en una UART, una vez hayamos configurado el módulo.

Placa complementaria

Los motivos de la realización de una placa complementaria se encuentran en la necesidad de poder alimentar los servomotores de una manera práctica que permita el movimiento de los mismos, y la única manera de realizarlo era con el diseño y confección de una placa que pudiese estar montada sobre el gimbal, y como la placa provista por la cátedra no cumplía con nuestros requisitos decidimos realizar una nueva.

Materiales utilizados para las placas

- Dos Borneras de 2
- Un Diodo 1N5819
- Un Diodo 1N4007
- Dos Capacitores 0,1uF
- Dos Capacitores 0.22 uF





- Dos Capacitores 10mF x 16V
- Un Regulador LM7805
- Un Regulador LM1117T3.3
- Dos Resistencias 4,7Kohm
- Varias Tiras de pines hembra

Problemas y Soluciones

Máquinas de estados:

Al comienzo del año nuestro primordial objetivo fue consolidar cual iba a ser la lógica general del funcionamiento del dispositivo en cuestión. Para ello utilizamos como funciones de referencia, toda la biblioteca que proporciona la cátedra.

El principal problema que hubo con las máquinas de estado fue que en un primer momento se plantearon desde un nivel de aplicación bastante abstraído del funcionamiento de los drivers, sin considerar el comportamiento de las interrupciones. También tenemos en cuenta que para el comienzo del año no contábamos con los conocimientos acerca de las interrupciones.

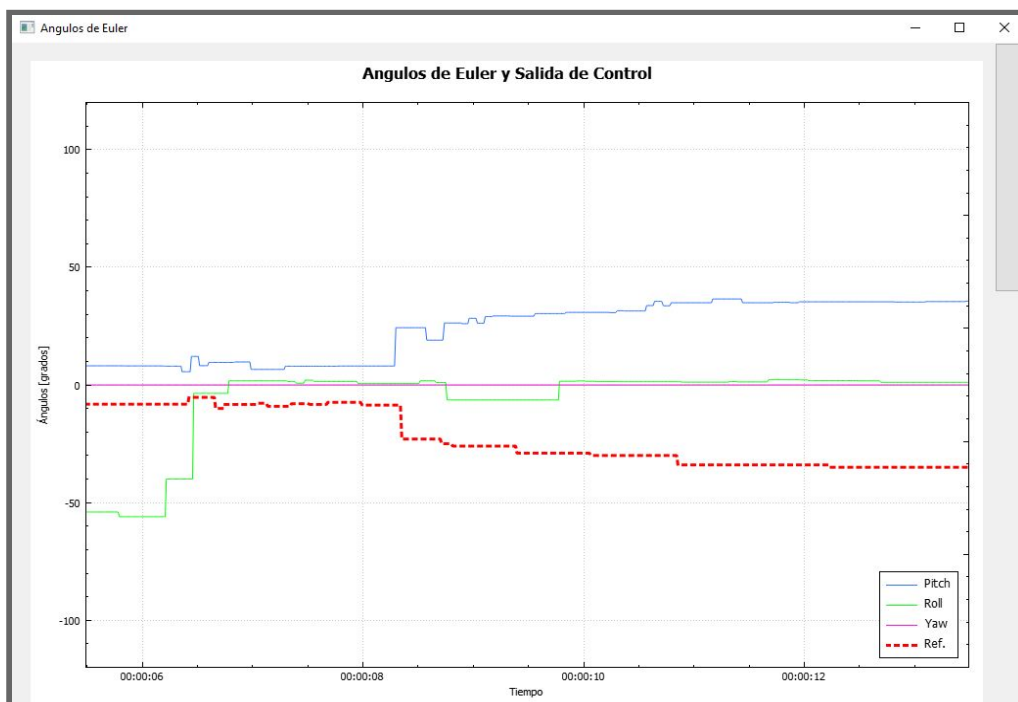
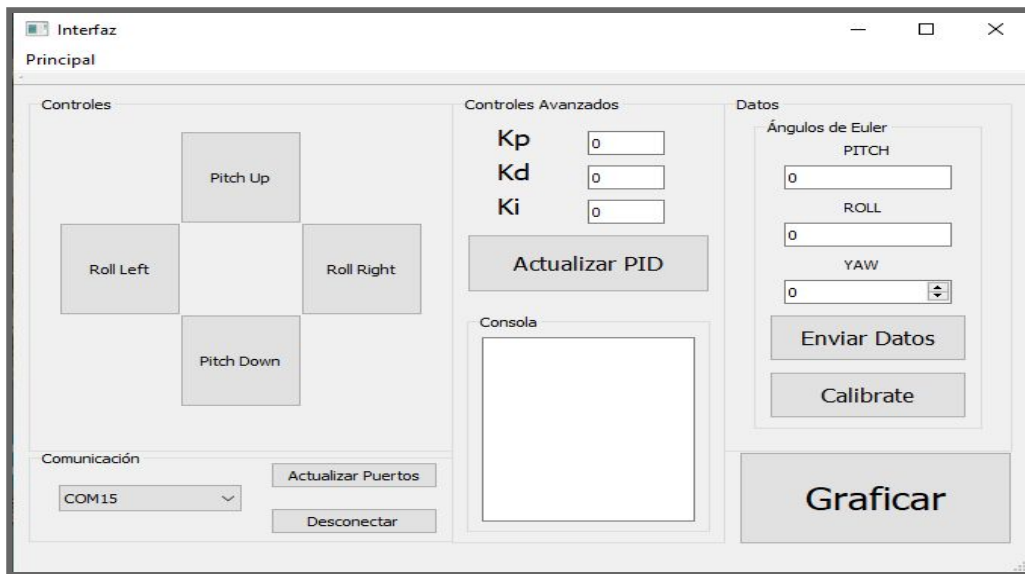
La solución en este caso fue re-hacer parte del funcionamiento de las máquinas abstrayendo aún más la capa de aplicación, pero incluyendo más herramientas en la capa de primitivas.

Interfaz gráfica:

Para el desarrollo de la interfaz gráfica se intentó desarrollar una aplicación que utilizara alguna librería gráfica como lo son Allegro o OpenGL. Nos encontramos con que estas librerías, en particular OpenGL, son bastante más complejas de lo que esperábamos. En ambos casos resultaba altamente dificultoso vincular las aplicaciones de Qt con las pantallas que generaban ambas herramientas.

También se trató de agregar una aplicación externa, previamente realizada por uno de los integrantes, hecha en Java. Aquí el problema que encontramos era compatibilizar el método para acceder a la misma información, dado que no eran distintos procesos pero tampoco compartían las mismas variables globales. Qt no compatible con otros lenguajes ó entornos de trabajo.

En este caso, la solución que encontramos para representar gráficamente la actitud de la cámara fue la utilización de una librería propia de Qt (qcustomplot) para generar gráficos complejos en tiempo real. Aparte se agregó un feature para manipular las constantes de control desde la aplicación en la computadora.





I2C:

Referido al periférico I2C, podemos decir que fue una de nuestras mayores complicaciones, dado que nos costó entender que para poder solucionar los problemas que teníamos debíamos leer reiteradas veces el datasheet. Esto sumado a que para poder evaluar el comportamiento del periférico es necesario la utilización de instrumental como un osciloscopio, del cual no disponemos en nuestras casas, y también fue otro instrumento que debimos aprender a utilizar. Esta fue sin dudas, la etapa del proyecto más ardua y tediosa. Agradecemos el esfuerzo de los ayudantes para con nosotros y su paciencia y dedicación para poder sobrepasar esta etapa.

Traspaso del código a C++:

Traspassar el firmware en MCU del lenguaje C al lenguaje C++ resultó ser más complicado de lo esperado. Para manipular la IMU, optamos con adquirir un código en c++ que contenía un objeto capaz de realizar muchas de las tareas necesarias. Si bien no terminamos utilizando la totalidad del objeto, dado que algunas de las rutinas contenían errores, de todas formas intentamos agregar este objeto en el firmware.

Para realizarlo, tuvimos que migrar el proyecto de C a C++. Esto implica que ahora es necesario que haya más de un compilador y un linker vinculados trabajando sobre el mismo código. Esto nos generó confusiones, dado que dependiendo cómo se declaran ciertos tipos de datos, defines o instrucciones pre-compilador, van a generar reacciones en algunas partes del código o no. Por tanto ciertos bugs que creíamos relacionados a la lógica del programa, terminaron estando vinculados con estos problemas de tener más de un compilador corriendo al mismo tiempo.

Sistema de Control:

Diseñar e implementar un sistema de control no es nada trivial. A pesar de no tener conocimientos reales de control, intentamos implementar uno de los algoritmos más utilizados comercialmente como se mencionó anteriormente.

Nuestra ignorancia en esta área provocó que nos atrasemos demasiado en el desarrollo del proyecto. Resultó que nuestros actuadores (servomotores) resultan bastante difícil de modelar, o al menos así lo vemos nosotros, y generan interacciones no deseadas en el sistema.

Tuvimos para esta etapa la ayuda de uno de los grupos de investigación de la facultad, relacionado a drones, con lo cual han implementado varios sistemas de control distintos. Sin embargo, sólo fue una guía, completamente necesaria de todos modos, para poder realizarlo.



Filtros:

Los filtros implementados fueron, en gran medida, diseñados por nosotros. Inicialmente, como se indicó en la sección de C++, se trató de utilizar un objeto diseñado por un profesional estadounidense. Este objeto utilizaba tanto las mediciones del acelerómetro, el giroscopio como del magnetómetro. Debido a que las mediciones obtenidas por el magnetómetro eran anormales, se buscó la forma de migrar el filtro. Por dicho motivo pasamos de utilizar el filtro Madgwick a utilizar un filtro complementario, junto con un filtro pasa-bajo, ambos diseñados por nosotros, o al menos en gran medida.

Los resultados obtenidos por los primeros intentos de filtros eran demasiado ruidosos (± 10 grados), por lo que no resultaban adecuados para nuestro dispositivo.

Luego de múltiples intentos, y una pequeña ayuda externa, logramos identificar dónde estaba el problema, en uno de los objetos al intentar realizar una pre-calibración. Por tomar como válido este cálculo externo, generamos datos erróneos los cuales generaban ruido en la señal de retroalimentación.

Una vez encontrado el problema, lo único que tuvimos que hacer fue re-hacer las funciones de calibración del objeto. Actualmente la señal resultante tiene un error de ± 1 grado, lo cual tomamos como válido para nuestro fin.

Conclusiones

A pesar de que el proyecto está orientado por la cátedra a que se evalúe solamente el software y no se contemple el hardware, el producto final al que se intentó llegar significó un desafío enorme para todo el equipo de trabajo. Tuvimos que lidiar con la inexperiencia de casi todos los integrantes en cuanto a hardware, diseño, armado y soldado de placas, protocolos de comunicación entre periféricos como lo fue I2C y sistemas de control. Con lo cual el tiempo empleado en tan sólo la comprensión de esos conocimientos fue muy grande comparado con el tiempo empleado en aplicarlos.

Sin embargo, al tratarse de una temática libre, la dificultad del proyecto fue alta desde un principio y a nuestro criterio, quizás no proyectamos los posibles inconvenientes y zonas de conflicto que podrían llegar a suceder mientras escalaba el proyecto. Fue desde un primer momento, una propuesta ambiciosa.

Si nos limitamos a evaluar de manera general si cumplimos con el objetivo inicial propuesto, podemos aseverar que se han alcanzado las condiciones mínimas del proyecto. Si lo detallamos, entonces la conclusión es que no estamos completamente satisfechos con los resultados. Es posible la realización de mejoras, tales como:



- Componentes, ya que los utilizados no son los más adecuados para el proyecto en cuestión.
- Filtros, ya que es la primera vez que realizamos este tipo de sistema.
- Control, por el mismo motivo detallado en filtros, consideramos que en esta área es dónde más mejoras podrían realizarse.

A pesar de todos los inconvenientes previamente mencionados a lo largo del documento, consideramos que la experiencia es en definitiva beneficiosa para el alumnado sobre todo en un ítem no mencionado, el trabajo en equipo. Destacamos el compromiso, organización y dedicación por parte de todos para con el proyecto y el entusiasmo y decisión con la que se llevó adelante todo el proceso desde el momento en que se decidió una idea y cuál era el camino a seguir.



Título:

Estabilizador de cámara GIMBAL

Ciclo Lectivo 2019

Curso N°

R2003

Grupo N°

3

Integrantes

Apellido y nombres	Legajo	Calificación individual	Fecha
Coiro, Mateo	1672538		16/12/2019
Golob, Lautaro	1672587		16/12/2019
Liaño, Lucas	1674389		16/12/2019
Lovallo, Alejo	1673658		16/12/2019
Repetto, Facundo	1674705		16/12/2019

Calificación Grupal:

Fecha: 16/12/2019

Profesor:

Ing. Nahuel Gonzalez

Auxiliar/es docente:

**Ing. Lisandro Sugezky
Sr. Nicolás Campos
Sr. Federico Bua**

Observaciones primera entrega:

Observaciones segunda entrega:

