**Instituto Tecnológico de Costa Rica**

**Ingeniería en Computadores**

**Taller de Programación**

Project II

Formula E CE TEC

Energy Saving and Telemetry Part I

**Estudiantes:**

**José Fernando Morales Vargas**

**Carné: 2019024270**

**Alejandro José Quesada Calderón**

**Carné: 2019150208**

**Profesor: Milton Villegas Lemus**

**Grupo 2**

**I Semestre**

**2019**

# Introducción

**Parte I:**

El propósito de este Proyecto era implementar un circuito de control por WiFi y telemetría en un carro a control remoto común. Por medio de este proyecto, los estudiantes entrarían en contacto con la combinación de electrónica y programación que caracteriza a la Ingeniería en Computadores.

Como herramientas para el desarrollo del proyecto se utilizó el IDE de Arduino 1.8.9, Tinkercad (Herramienta web de Autodesk, sin versión) como apoyo pedagógico, y los programas WiFiClient y TelemetryLog provistos por el asistente Santiago Gamboa para la prueba de los comandos programados.

Los componentes utilizados en el proyecto fueron los siguientes: NodeMCU v3 Lolin, driver de motor L298N, registro de corrimiento SN74LS164, regulador de voltaje 7085(5V), transistor 2N4125, resistencias: una 100kΩ, una 68k Ω, una 10k Ω, dos 1k Ω, 6 300Ω; tres pares de leds de color rojo, amarillo, blanco y un carro a control remoto usado. Para proveer de energía al circuito se utilizaron 6 baterías Panasonic Alkaline, y una batería de 9V genérica

El circuito elaborado es capaz de ser implementado en otra carrocería, los contactos directos con la misma son únicamente los cables del motor de dirección. Los demás contactos con la carrocería se pueden desprender y adaptar.

El carro recibe señales enviadas a través de WiFi por medio del NodeMCU, este las interpreta y ejecuta los comandos. Varían entre el funcionamiento normal del carro como movimiento o el cambio de dirección(dir); control de las luces (lf, lb, ll, lr) y telemetría, detectar el nivel de batería actual y si el carro se encuentra en un lugar con mucha o poca luz(sense). Se comandos especiales que modelan comportamientos particulares, Circle, Infinite, ZigZag y Especial.

A nivel de programación, se logró modelar por completo las instrucciones requeridas, sin embargo, se considera que hay mejoras que se pueden hacer para que las mismas sean más precisas en su comportamiento.

En relación al aspecto estético, se considera que el mismo puede ser mejorado. Aún no tiene un acceso fácil al circuito, sino que se debe levantar la carrocería del carro para poder revisar/cambiar algo en el circuito.

El carro requiere señales de alta potencia para mover el motor de tracción, puesto que el mismo no es muy potente.

Se logró implementar el circuito por completo en el carro a control remoto, sin embargo, de la esquemática originalmente provista se realizaron algunos cambios de conexiones del SN74LS164 y el L298N, además del intercambio del transistor 2N3206 por un 2N4125.

**Parte II:**

En el trabajo pasado, se requería que el equipo de trabajo realizara modificaciones a un carro a control remoto, para que este se asemejara más a un carro de verdad (incluyendo en sus funcionalidades) y que además pudiera ser controlado por internet. Para esa etapa, se utilizaron diversos componentes electrónicos (todos mencionados en la documentación de la parte I), entre los que destaca el NodeMCU v3 de Lolin, el cual posibilita esta comunicación inalámbrica.

Fue necesario mencionar el producto de la fase anterior a la presente, para explicar dicho trabajo, puesto que, al ser dos partes del mismo proyecto, el producto final de la primera se necesita para el desarrollo de la etapa final. Esta etapa, consiste en crear una versión mejorada del control remoto “TelemetryLog” de Python; incluir las mismas funciones de este, y, además, generar una interfaz gráfica en la que el usuario del auto pueda ver el estado del carro. En sí, la mejora del control del carro sólo es parte de lo que debía realizarse, ya que también debe añadirse a toda esta interfaz una ventana de posiciones en las que se guarda el rendimiento de los pilotos, una para llevar un registro de los autos de la escudería, y la información sobre los patrocinadores, junto con el logo de la empresa.

Para estas tareas, se está usando como herramientas la biblioteca Tkinter de Python, con dicho lenguaje en su versión 3.7.2, y GitHub (en su versión para escritorio) como repositorio remoto/ herramienta para control de versiones. De la biblioteca Tkinter se utilizan los métodos para generar ventanas, y en estas se trabaja con lienzos en los que se insertan objetos, en conjunto con hilos y funciones iterativas para el manejo de estos hilos. En profundidad sobre la ventana de test drive, se están controlando los eventos del carro con binds en las teclas, que consisten en los eventos *KeyPress y KeyRelease*, cada tecla de las utilizadas se controla en una función y dependiendo de su aplicación, invocan un hilo, o simplemente cambian el valor de banderas que mantienen activos otros hilos. Algunos ejemplos de esto son la tecla W (aceleración) y la X (apagado de las direccionales); el primer ejemplo invoca un hilo al presionarse, y el segundo funciona para cambiar las banderas mencionadas. Estas banderas se trabajan con variables no-locales y globales.

Para comunicar el módulo de software con el hardware aún se está utilizando el archivo WiFiClient.py dado por Santiago, sin embargo, se ha descartado el uso de TelemetryLog como parte del módulo, simplemente se utilizó como fuente de información y para modificar sus funciones (e implementar estas al código desarrollado por los integrantes del grupo).

Para la parte gráfica del Test Drive, se decidió simular el Heads-Up Display de un videojuego, por esto existe tanto una visión en primera persona, como una perspectiva aérea del carro en la que se puede ver el estado de este (en términos de luces y direcciones). Como decisión de desarrollo, el valor de la potencia se está dando de forma porcentual para hacerlo más legible.

***\*\*\*\*Aún se debe añadir el qué se hizo y con qué de la parte de pilotos de Morales\*\*\****

# Conclusiones:

**Parte I:**

·         Tinkercad Circuits es una herramienta web útil para simular un circuito o un proyecto pequeño sin tener que descargar un programa.

·         La página de Arduino Reference fue de gran ayuda para la parte de software de este proyecto, casi indispensable.

·         Foros relacionados al proyecto en StackOverflow y GitHub fueron de gran utilidad para darse ideas de cómo debía ser el código para el NodeMCU.

·         Utilizar imágenes de Google sobre los diagramas de los componentes individuales fue de gran ayuda para conectar el circuito.

·         Investigar en las datasheets u hojas de datos de cada componente fue vital para entender el comportamiento de estos.

·         Utilizar el L298N por sí sólo fue complicado, tanto al realizar las conexiones como al comprender su funcionamiento.

·         Las líneas que denotan el cátodo del diodo rectificador se borraron fácilmente, lo cual dificultó el uso seguro de estos.

·         Tuvimos que rearmar el circuito en la protoboard solo para compactar su diseño.

·         El diagrama del circuito debió ser corregido en conexiones importantes del 74LS164, esto llevó a un *troubleshooting* extenso.

·         Investigar sobre el arreglo de 4 diodos (o puente rectificador) fue útil para comprender la utilización de estos en el circuito.

·        Fue complicado reunirse para trabajar con los horarios especiales de transporte público en Semana Santa.

·         Desarmar el circuito repetidas veces consumió mucho tiempo, pero fue necesario para el proceso de aprendizaje.

·         El auto utilizado en particular incluía un sistema para colocar LEDs en la parte frontal, por lo cual la integración del control de las luces frontales tiene un mejor aspecto estético.

·         Utilizar el Monitor Serie para depurar el código y solucionar problemas fue una medida de gran utilidad.

·         Investigar sobre la función shiftOut() fue necesario para comprender el manejo de datos en el 74LS164.

·         El investigar sobre las funciones de categoría *analog and digital I/O* fue especialmente útil para el control de los motores.

·         La modelación matemática del comportamiento de la batería fue indispensable para la telemetría de esta.

·         Fue complicado dedicarle periodos largos al proyecto una vez que volvimos a semanas lectivas y a los parciales.

·         Los ejemplos buscados en YouTube para utilizar una LDR con Arduino como sensor de luz fueron muy útiles para la telemetría.

·         La investigación que se le dedicó a la función sprintf() de C en Arduino fue muy importante para comprender el funcionamiento del comando *sense*.

·         Aprender a manejar el código por medio de GitHub Desktop ayudó bastante, tanto en orden como en seguridad, y en el mantenimiento y la legibilidad de este.

·         No se tuvo en cuenta la posibilidad de dañar algún componente, por lo que al dañar los diodos sufrimos un atraso en el flujo de trabajo.

·         Tener herramientas como un removedor de soldadura o unas manos ayudantes a nuestra disponibilidad facilitó la tarea de soldar el circuito significativamente.

·         Reforzar los conocimientos establecidos en el taller de Arduino acerca del pwm, el uso de voids, la declaración de pines y las lecturas digitales (analogRead()) fue bastante necesario para la programación del NodeMCU.

·         Haber instalado un tema oscuro para el IDE de Arduino fue bueno para la etapa de desarrollo de código.

·         Investigar sobre el comportamiento de los operadores lógicos en C y su sintaxis en este lenguaje fue mandatorio para manejar las luces correctamente.

·         Aprender a manejar el estrés luego de situaciones críticas en este proyecto fue útil para aprovechar el tiempo de trabajo.

·         Investigar acerca del pin D8 y su función como pin durante el boot fue útil para descubrir un error muy importante en el diagrama del circuito.

·         El código brindado por el asistente Santiago G. para el TelemetryLog fue de gran utilidad para probar el comportamiento del carro y el código de este.

·         Utilizar un carro de calidad cuestionable y de varios años después de su compra nos generó problemas que se pueden evitar consiguiendo uno de mejor calidad.

·         El multímetro es una herramienta poderosa para la detección de problemas si se sabe cómo utilizarlo de manera correcta.

·         Los motores integrados en un carro a control remoto pueden ser insuficientes para movilizar el carro con un circuito similar al implementado. El peso de la circuitería y las baterías los pone bajo particular estrés.

·         Se debe tener especial cuidado al soldar diodos y transistores, puesto que los mismos son especialmente sensibles al calor.

·         No todos los pines digitales del NodeMCU están disponible para todo tipo de tarea. Previo a planear un proyecto se debe comprobar que los mismos no causen un conflicto (tal como el circuito detector de luz en el pin D8).

·         Los corredores de registro se pueden utilizar para controlar componentes que no necesitan mayor alimentación, y que solo requieren señales altas o bajas para ser controlados.

·         Los diodos rectificadores son herramientas importantes para mantener la integridad de nuestros componentes al evitar inversiones de corriente.

·         Planear cómo se van a colocar los componentes antes de soldar es un paso vital que nunca se debe evitar.

·         Si la circuitería es algo complicada, al soldar en placa preperforada es más sencillo hacer las conexiones por cables que por puentes de soldadura.

·         Utilizar bases en vez de soldar los componentes importantes aumenta la posibilidad de reutilizar los componentes importantes.

**Parte II:**

* Definir los roles y tareas para los integrantes del proyecto en las fechas iniciales del periodo de trabajo fue útil para la organización de este.
* Organizar el repositorio en GitHub antes de empezar el trabajo digital ayudó con la modularización del proyecto.
* Los conocimientos adquiridos en el taller de Git aumentaron el uso de todas las herramientas prestadas por Git al usuario.
* La utilización de Tkinter como herramienta para interfaz gráfica fue buena opción considerando su manipulación en trabajos anteriores.
* Generar las ventanas necesarias y que se pueda comunicar entre ellas antes de trabajarlas individualmente ayuda a organizar el trabajo con la interfaz gráfica.
* Estudiar los códigos prestados por el asistente del curso fue necesario para entender su funcionamiento.
* Del estudio de estos códigos se pueden descubrir
* La herramienta ScrolledText de Tkinter es muy útil si se desea incluir un texto espacioso en una ventana compacta.
* Colocar imágenes de prueba para que tomen el espacio del objeto final en la ventana About fue útil para avanzar sin dejar la ventana completamente sin manipular.
* La función geometry() fue útil para manejar las dimensiones de la ventanas con más facilidad, igualmente para que este parámetro sea más legible.
* Importar el WiFiClient.py del asistente del curso es mandatorio para el funcionamiento de este proyecto.
* Definir un solo comando de retroceso, y asignarle este a todos los botones de retorno, simplificó el código y lo optimizó.
* Aprender sobre la expresión lambda y su utilidad para “funciones anónimas” fue necesario para que se pueda utilizar un botón con argumentos.
* Avanzar paralelamente la parte estética y la lógica de la ventana Test Drive fomentó la creatividad y mantuvo una buena estructuración de dicha ventana.
* El uso de imágenes en formato PNG fue necesario para poderlas manejar con fondos transparentes y mantener una buena calidad de imagen.
* Revisar bien las funciones que se toman del código TelemetryLog o WiFiClient es conveniente, pues algunas tienen implementaciones lógicas que ya no son necesarias y se pueden modificar.
* Utilizar eventos de “keybinding” para el control del carro fue una práctica bastante útil, aún más el haberla implementado con funciones separadas según el tipo de evento (Press/Release).
* Manejar los eventos en conjunto con variables globales (o también, con no-locales) facilita el control de casos especiales/específicos.
* Consultar en internet para verificar la sintaxis o aplicación de ciertos métodos, fue útil debido al tiempo que pasó desde la última vez que se utilizó Tkinter en el curso.
* La función canvas.create\_text() fue de gran utilidad para mantener un canvas limpio, dado que este elimina la necesidad de un label para contener el texto.
* Consultar con el asistente del curso sobre divergencias entre los desarrolladores y las especificaciones del proyecto fue necesario a lo largo del desarrollo, para asegurar un cumplimiento de dichas especificaciones.
* La validación de casos especiales en los eventos fue más complicada de lo que parecía en un inicio al planearse.
* Mejorar los conocimientos necesarios para aplicar la lógica correcta en Tkinter fue necesario debido al poco manejo de esta herramienta que se tenía.
* Tener una rama en Git para cada integrante del equipo ayudó a la hora de evitar conflictos, ya que así se manejan archivos y espacios virtualmente distintos.
* Trabajar con mayor prioridad en la parte funcional de Test Drive antes de la gráfica (sin dejar esta última de lado) fue importante para un decente manejo del tiempo de trabajo.
* Utilizar una versión online de Photoshop fue necesario para adaptar las imágenes al formato que se necesitó en el desarrollo, sin tener que descargar dicha aplicación.
* Detener el avance de las tareas cuando la lógica de las direccionales se implementó sin errores (pero diferente del comportamiento solicitado), fue bueno para evitar que las equivocaciones permanecieran.
* La utilización de iteración dentro de las funciones invocadas por los hilos facilitó la validación de casos con las variables nonlocal/global.
* Tomar el módulo de test drive y mejorarlo en un archivo ajeno al repositorio en el archivo NewGUI (y posteriormente DebuggedGUIv2) fue crucial para mejorar la funcionalidad de este.
* Revisar la funcionalidad del código de Arduino para el NodeMCU fue necesario para avanzar en las tareas asignadas.
* Manejar las respuestas del NodeMCU en Python con la lista “Log” de WiFiClient fue útil, ya que validar estas con el largo de sus elementos facilitó la implementación de los sensores del auto.
* Documentar los cambios realizados al NodeMCU se considera una buena práctica para evitar problemas con los encargados del curso.
* Se aprendió (por parte de uno de los integrantes del grupo) que el uso de mainloop en todas las ventanas, así como de globales se corrige con uso de *nonlocal* y al implementar las imágenes y botones como propiedades de la ventana a trabajar.

# Recomendaciones:

**Parte I:**

·         Consultar *Tinkercad Circuits* para simular sectores del circuito si se desea probar una parte dudosa en las conexiones, requiere de una cuenta de usuario, pero de este modo no se arriesga ningún componente del circuito a ser eventualmente dañado.

 Consultar la documentación de Arduino Reference en internet (arduino.cc/reference/en/) para comprender mejor las funciones a utilizar, al igual que el lenguaje de programación.

·         Buscar información en foros GitHub sobre la biblioteca ESP8266 para saber instalarla correctamente en el Arduino IDE.

·         Buscar en StackOverflow y GitHub sobre problemas que tenga con el código del circuito en caso de que no funcione de la forma esperada, generalmente ayuda a solucionar el problema más rápidamente.

·         Darse el tiempo de buscar ilustraciones esquemáticas de los componentes del circuito, toma un poco de tiempo, pero acelera y facilita la comprensión de las conexiones que se deben hacer.

·         Buscar las hojas de datos de todos los componentes para tenerlas a mano, y utilizarlas para comprender el funcionamiento de los componentes.

·         Utilizar las hojas de datos para verificar si las conexiones del circuito no van a generar problemas que no se evidencian a simple vista.

·         Conseguir un controlador de motores L298N integrado en una placa, este le ahorrará tiempo y problemas, a cambio de un poco más de recurso económico y alimentación en el circuito.

·         Utilizar un marcador blanco o corrector líquido para marcar el cátodo de los diodos, pues estos pierden su propia marca en poco tiempo. Esto le ahorra costos en caso de dañar componentes, y tiempo.

·         Idear el circuito en la protoboard antes de comenzar a ensamblarlo, así puede tener un circuito ordenado desde el inicio y no tener que recurrir a desarmarlo diversas veces.

·         Llevar el diagrama del circuito brindado a un profesor para darle una revisión a fondo en búsqueda de errores potenciales, así se puede verificar si más adelante algún problema con el circuito es culpa del desarrollador, o un fallo en el diagrama.

·         Buscar una explicación en línea de por qué se utilizan los diodos en un arreglo de 4, esto para comprender mejor su utilización en el circuito, ya que en el diagrama no es tan obvio como en un ejemplo dedicado.

·         Investigar detenidamente el comportamiento del integrado 74LS164, toma más tiempo que los demás componentes, pero ayudará a comprender el porqué de las conexiones del circuito, y su funcionamiento.

·         Generar un plan de reunión con buenas bases entre los integrantes del equipo si se van a dedicar al proyecto en días feriados, para no perder tiempo buscando horarios de buses o vías que no estén cerradas y comunicarse con más facilidad.

·         Desarmar el circuito parcial o completamente si no encuentra alguna falla de hardware, así puede dedicarle más tiempo y precisión, y adquiere un mejor conocimiento sobre las partes y sus conexiones.

·         Conseguir un auto que tenga iluminación incluida en el mismo, así se ahorra tiempo preparando la parte estética, y no tiene que gastar en materiales extra como silicón o goma líquida para fijar los diodos.

·         Si el código está teniendo un comportamiento inesperado relacionado a algún valor, generar una línea que imprima el valor en el Monitor Serie puede ser de gran ayuda para solucionar el problema.

·         Investigar en Arduino Reference o en YouTube sobre la función shiftOut(), al igual que las otras funciones, toma algunos minutos de lectura, pero ayudará a obtener un buen funcionamiento en el 74LS164.

·         Tratar las tareas en el orden correcto, no intente programar los comandos de movimiento sin antes haber manejado con eficacia los motores por sí mismos, hacerlo en el orden recomendado le facilitará la tarea.

·         Buscar la manera de describir el comportamiento de la batería con una modelación matemática, esto evitará que deba recurrir al uso de fuerza bruta y condicionales anidados en su código de telemetría.

·         Manejar bien el tiempo una vez que se deba trabajar el proyecto en tiempo lectivo, no lo deje para los últimos días, pero no arriesgue otras clases por terminarlo con demasiado tiempo de sobra. Así habrá un balance entre el progreso y su situación académica.

·         Buscar ejemplos de uso de LDRs en Arduino como sensor de luz, para comprender mejor cómo se puede programar la telemetría.

·         Investigar sobre el comportamiento de la función sprintf() utilizada en el comando ***sense;*** de este modo comprenderá por qué deben definirse variables que la función usará como valores de retorno, y facilitará la comprensión en general del comando sense; al igual que le ayudará a encontrar su propia solución a la telemetría.

·         No complicarse con el sensor de luz: Es suficiente con usar un digitalRead() en el pin correcto para saber si hay o no luz en el ambiente, así no pierde tiempo que puede dedicar a otros problemas del código.

·         Utilizar GitHub Desktop desde el primer día en el que trabaje con el código, sólo debe instalarse y crear un repositorio, sin embargo, esta herramienta es extremadamente útil para llevar un registro de su progreso en la codificación.

·         Comprar componentes de repuesto para las partes del circuito, en caso de que alguna se dañe, así no perderá tiempo buscando los componentes cuando ya esté enfocado en el circuito y podrá avanzar más rápidamente.

·         Conseguir herramientas como un removedor de soldadura o una tercera mano le facilitarán significantemente la tarea de soldadura, aunque requiere una notable inversión económica si no dispone de estas.

·         Reforzar las bases que se enseñaron en el Taller de Arduino con el Asistente puede ayudarle a comprender el *pulse-width modulation*, la conexión de pines y las funciones utilizadas para el control de entradas y salidas analógicas y digitales.

·         Investigar sobre los operadores lógicos en C (& y |) para el manejo de bytes en el registro facilita la tarea de controlar las luces, a la hora de declarar el valor que llevará la variable *data* utilizada en el código.

·         No dejar de lado su ritual de control de estrés, o buscar su propia manera de liberarlo si aún no conoce alguno. Con un poco de investigación, le ayudará a aprovechar más el tiempo que le dedique a su proyecto.

·         Si hay algún sector del circuito que esté generando problemas al inicializar el NodeMCU (como en este caso, la LDR y el pin D8), investigar sobre las funciones en segundo plano que cumplen estos pines puede ayudarle a encontrar la causa del problema.

·         Utilizar el código de TelemetryLog brindado por el asistente le facilitará la tarea de probar sus comandos con el carro, sólo debe copiar el código de GitHub a un archivo local, y así puede saber con rapidez si el código es funcional o no.

·         Conseguir un auto que muestre estar en buenas condiciones y tener una buena calidad en sus componentes, principalmente los motores, ya que hay que tener en cuenta que estos no están diseñados para soportar el peso de los componentes a utilizar, y podrían generarle problemas de movimiento.

·         Descargar el tema oscuro para el Arduino IDE si trabajará en ambientes oscuros, sólo requiere descargar un archivo .zip y seguir una corta guía, le ayudará a mantenerse enfocado en el código y, a la larga, evitar el daño en los ojos por exposición a la luz artificial en ambientes poco iluminados.

·       Si desconoce sobre un tema a nivel técnico, solicitar la ayuda de un amigo puede disminuir la curva de aprendizaje. De negarse a la posibilidad de que le ayuden, puede ser que aumente la cantidad de tiempo que lleva una tarea de manera innecesaria.

·       Utilizar registros de corrimiento para control de luces es una forma de aumentar la cantidad de pines disponibles en un microcontrolador para otras funciones.

·       Si hay componentes que se pueden ver afectados por una inversión de la corriente, se recomienda crear un circuito protector con diodos, los mismos son baratos, y procurar proteger los componentes evita tener que comprar reemplazos a futuro.

**Parte II:**

* Empezar el trabajo por la declaración de tareas y roles, así se ahorra tiempo y evita desorden, de lo contrario podría sufrir conflictos y otras complicaciones entre los integrantes.
* No empezar a trabajar los módulos de software sin haber organizado un repositorio. Hacer caso omiso puede llevar a pérdida de proceso y a una mala modularización.
* Utilizar los conocimientos del Taller de Git (si se le da uno en clase) para aprovechar las herramientas que brinda GitHub al usuario.
* Si anteriormente utilizó Tkinter y se familiariza con la biblioteca, apéguese a esta en vez de buscar alternativas, se ahorra tiempo en investigación y se tiene mejor manejados los conocimientos.
* Al trabajar en la interfaz, generar todas las ventanas especificadas y poder comunicarlas antes de dedicarles tiempo individualmente le ayudará a llevar el trabajo mejor organizado. No hacerlo podría hacer que omita alguna especificación del trabajo.
* Si el asistente de curso presta los archivos WiFiClient y TelemetryLog para el manejo de la comunicación, es altamente útil estudiar estos, así puede decidir qué funciones se importan a su código o qué utilizar de estos, igual si decidiera modificarlos. Omitir esto complica mucho las tareas del proyecto.
* Si desea introducir un texto amplio en alguna ventana, la herramienta ScrolledText puede serle útil. No utilizar esta o alguna similar, implica que deberá cambiar el tamaño o geometría de dicha ventana.
* Si no va a trabajar profundamente una ventana, pero esta contiene imágenes que requiere colocar estratégicamente (como el caso de la de créditos), utilice imágenes de prueba para calcular estos espacios, así evita dejar la ventana sin trabajar del todo mientras avanza alguna otra tarea.
* Si desea declarar alguna dimensión definida para una ventana, utilizar la función geometry(“XxY”) para especificar estas dimensiones es una mejor opción que declararlas por separado, así le ayudará a comprender más qué se está definiendo.
* Debe importar el WiFiClient.py o utilizar este en el mismo módulo de su programa para comunicarse correctamente con el hardware, de lo contrario, deberá generar su propio método de conexión y le dificultará mucho el desarrollo del trabajo.
* Si va a colocar un botón de retroceso en todas o varias de sus ventanas, utilizar una misma función como comando para todas estas es la mejor opción, de lo contrario necesitará uno por ventana, esto le tomará más tiempo y recursos.
* Para manejar comandos con argumentos se debe utilizar la expresión *lambda*, de lo contrario, no podrá reutilizar segmentos de su código como comandos con argumentos.
* No avanzar demasiado la parte lógica de la interfaz de manejo dejando de lado la estética (o viceversa), este modus operandi le llevará a una mala estructuración de la ventana, o hasta olvidar funciones que debe cumplir.
* Si desea generar una ventana limpia y atractiva al usuario, utilice el formato PNG para sus imágenes, así mantendrá una buena calidad y podrá implementarlas con fondos transparentes. Para esto quizá necesite investigar sobre editores gráficos (depende de su conocimiento sobre el tema)
* Si toma funciones de los códigos prestados por el asistente, WiFiClient y TelemetryLog, evalúe si toda la lógica implementada es útil para sus propósitos, así puede descartar lo que no sea necesario. No hacer esto puede llevar a incluir código que no es necesario, por ende, gastar más recursos computacionales.
* Si desea controlar el hardware por medio del teclado, utilizar los bindings KeyPress/Release es una excelente opción, y también es bueno trabajar cada evento/mapeo con una función por separado.
* Para tener un mejor control de la inicialización y detención de un proceso, se facilita con el uso de variables globales o no locales (en función de banderas). Esto le ahorrará tiempo, y volverá el código menos propenso a fallar por casos especiales.
* Por más que confíe en sus conocimientos sobre la herramienta de interfaz que esté utilizando, siempre es bueno verificar la implementación de algunos métodos dudosos en páginas o libros de referencia, como effbot.org y su artículo sobre Tkinter.
* Para tener un canvas bien ordenado y atractivo, si busca implementar texto en este, puede evitar la necesidad de un label con la función créate\_text(), eso genera el texto directamente sobre el canvas, tomando menos espacio.
* Si tiene alguna duda sobre las especificaciones del proyecto, o cree que está implementando algo de manera errónea, siempre es buena idea consultar con el asistente del curso, así puede evitar llegar a discusiones con los encargados, y eventualmente perder puntos en la rúbrica del proyecto.
* Sentarse a investigar más sobre la herramienta en uso (por ejemplo, Tkinter) si no manejaba bien esta antes de iniciar sus tareas.
* Si está trabajando con otra persona o varias, tener una rama en el repositorio para cada integrante le ayudará a evitar conflictos entre cargas de cambios al repositorio remoto, y a la larga le ayudará a llevar un mejor control del avance.
* Dar prioridad al trabajo en la lógica/funcionalidad del código antes de la parte gráfica es un buen camino que seguir si desea tener un buen manejo del tiempo, sin embargo, no es bueno dejar de lado la parte gráfica completamente.
* Para adaptar las imágenes que desee usar al formato recomendado, puede utilizar editores en línea, tales como Online Photoshop (ver fuentes consultadas), así evita tener que descargar programas de este tipo.
* Si se encuentra con que una de sus aplicaciones no falla, sin embargo, esta no se comporta de la manera esperada, es mejor detener el avance en las demás tareas y reparar esta, ya que, si continúa con el trabajo, puede que olvide solucionar ese problema y termine arriesgando puntos en las evaluaciones de su producto.
* Ya que se permite el uso de la iteración, es bueno dar uso de estos métodos en el manejo de hilos que deben ejecutarse siempre que algún evento se esté dando; facilita la validación de casos al igual que detener estos mismos hilos con las condiciones implantadas en los loops.
* Si ve necesario cambiar algo muy significativo (o varias líneas de código) en alguna función o ventana específica, es una buena idea tomar esta función y manipularla en un archivo por aparte, luego cuando se sepa que los cambios realizados funcionan, implementarlos al repositorio en Git; esto puede evitarle pérdidas de código o progreso y tener que revertir los cambios con un commit anterior.
* Revisar la capacidad de ser operacional y el estado del código para el NodeMCU si va a hacer utilidad de este, ya que puede que este no cumpla con sus necesidades y le lleve a atrasarse con otras tareas.
* Para manejar las respuestas que Python obtiene de los comandos enviados al Node, es óptimo utilizar la función get\_log() del WiFiClient, pues esta retorna la respuesta que se requiere manipular en una lista, esto implica que se puede tratar con índices y posiciones, y le ahorrará problemas el apegarse a este método.
* Si realiza algún cambio que vea necesario al código de Arduino o al hardware en sí, es mejor documentar este, para así evitar tener problemas con los encargados a evaluarle. No seguir este paso puede llevarle a perder puntos en la calificación.
* Si desconoce de algún tema o de las implementaciones generales de una herramienta como Tkinter y otro integrante de su equipo es más diestro en ese campo, solicítele asesoramiento para asegurarse que todos están al tanto de qué prácticas conforman el conjunto de *buenas* prácticas, y cuáles deben evitarse.

**Análisis de Resultados:**

Con base en las especificaciones del proyecto dadas por el profesor Milton Villegas Lemus,

**Diagrama de Módulos:**

**Parte I:** Corresponde al software del NodeMCU

**Diagrama del circuito:**

****

**Parte II:**

**Diagrama de la Arquitectura:**

# Plan de pruebas:

**Parte I:**

|  |  |  |
| --- | --- | --- |
| ***Prueba*** | ***Sí*** | ***No*** |
| Ningún cable está lo suficientemente expuesto como para poder provocar un corto |  |  |
| El circuito permite paso de corriente |  |  |
| El circuito mantiene voltaje correcto en riel de 5V y riel de 6-7.5V |  |  |
| Inicia el NodeMCU |  |  |
| Se conecta al WiFi |  |  |
| Recibe señales y emite una respuesta |  |  |
| Prende luces delanteras |  |  |
| Prende luces rojas traseras |  |  |
| Prende luz amarilla izquierda |  |  |
| Prende luz amarilla derecha |  |  |
| Apaga luces según comando (apaga solo las deseadas) |  |  |
| Circuito sensor de luz cambia señal dependiendo del nivel de luz |  |  |
| Circuito de sensor de batería identifica correctamente el nivel de batería |  |  |
| Motor de tracción se mueven para adelante y atrás según señal |  |  |
| Motores de dirección se mueven a la derecha, izquierda o centro según señal |  |  |
| Ejecuta comando ZigZag |  |  |
| Ejecuta comando Infinite |  |  |
| Ejecuta comando Circle y se mueve acorde al valor enviado |  |  |
| Ejecuta movimiento Especial |  |  |

**Parte II:**  
-Interfaz de Test Drive (función de controlar al carro y dar feedback gráfico sin haber implementado la lógica de pilotos)

|  |  |  |
| --- | --- | --- |
| ***Prueba*** | ***Sí*** | ***No*** |
| Las ventanas básicas se comunican entre sí |  |  |
| La ventana Test Drive se abre correctamente |  |  |
| Las funciones dentro de la ventana entran en los sectores correctos del código (probado con prints) |  |  |
| Las funciones que envían comandos al auto lo hacen efectivamente |  |  |
| Los cambios en el auto se ven reflejados correctamente en la interfaz |  |  |
| La ventana recupera los datos de los sensores de batería y luz del auto y lo refleja en la interfaz |  |  |
|  |  |  |

\***Introducir las partes faltantes del plan de la parte II\***

**Diagrama de Módulos**

**Bitácora**

**Parte I:**

**Viernes, 12/04/2019:**

**-Entrada de Ambos**

- Administrativa: Es el último día de clases antes de salir a semana santa, por lo que debemos movilizarnos para definir un plan de trabajo para los próximos días y también qué materiales a utilizar.

En la noche, luego de haber trabajado por un rato en el circuito del proyecto, realizamos un cambio en el plan de trabajo: se acordó que nos íbamos a reunir en el domicilio del compañero Alejandro (Grecia, Alajuela) con el objetivo de terminar la parte técnica del proyecto antes del martes, o el mismo día de ser posible.

-Técnica: Hoy obtuvimos los materiales necesarios para comenzar la construcción del hardware del proyecto, gracias al asistente Santiago Gamboa, obtuvimos un “kit” de componentes que nos da la mayoría de los materiales necesarios; también conseguimos el carro a control remoto requerido para la realización del proyecto.

Nos reunimos en el apartamento de Alejandro luego de clases para iniciar el proyecto, repasamos las especificaciones de este que nos brindó el asistente Santi, y le dimos uso al diagrama del circuito iniciado para comenzar la plasmación del mismo con los componentes electrónicos obtenidos.

**Sábado, 13/04/2019:**

**-Entrada de Ambos**

- Administrativa: Hoy por la mañana, partimos desde Cartago hacia Grecia, con el objetivo de cumplir con el cambio de planes que definimos la noche anterior. Por este motivo, no trabajamos en el proyecto hasta horas de la tarde del día de hoy.

- Técnica*:* Luego de obtener algunos materiales extra de la casa del compañero José (una protoboard, algunos cables de bus y SATA, resistencias, etc.) llegamos a Grecia, y nos dedicamos a continuar con el desarrollo del circuito. Hasta ahora, tuvimos que investigar acerca de los componentes utilizados en el mismo, puesto que el arreglo de pines que se le da a cada componente en el diagrama del archivo .pdf del proyecto es un poco confuso. Gracias a algunas imágenes de Google, los datasheets de los componentes, entre otras medidas, logramos comprender la lógica del circuito, y nos dispusimos a terminarlo. Para conectar el controlador del motor L298 requerimos comprar un paquete de jumpers macho-hembra, pues tuvimos la idea de que utilizar estos para conectar el controlador a la protoboard sería más sencillo a la hora de hacer pruebas con el circuito completo.

En el tiempo de trabajo de hoy logramos hacer un acomodo preliminar de los componentes, se nos complicó el trabajo a la hora de conectar los componentes de varios pines, tanto el L298N, como el Registro, al igual que transistores y demás componentes, que a simple vista no se pueden distinguir entre una orientación correcta o una errónea. Para todos estos componentes utilizamos las imágenes de Google, ya que, en el diagrama dado, sólo se especifican los pines que deben llevar cierta conexión, y no en qué parte del circuito se encuentran dichos pines, así que recurrimos a otras fuentes para poder encontrar el pin necesario para cada caso.

**Domingo, 14/04/2019:**

**-Entrada de Ambos**

- Administrativa: Ayer quedamos de acuerdo en reunirnos en mi casa por la mañana, pues el compañero José no planea quedarse a dormir aquí estos días de trabajo, así que toma un bus de ida en la noche hacia San José, y uno de vuelta cuando nos volvemos a reunir. Por algunas complicaciones de la ruta del bus y la hora, no pudimos vernos para el trabajo hasta horas de la tarde, pero al llegar nos enfocamos en el mismo.

- Técnica*:* Aún no hemos armado el circuito completo según el diagrama que Santi nos brindó. Hemos tenido problemas con el espacio, puesto que, aunque disponemos de una protoboard con espacio suficiente para todo el circuito, estamos teniendo en cuenta que no podemos hacer un arreglo de componentes muy voluminoso si queremos que quepa con facilidad dentro del armazón del carro.

A la hora de acomodar algún componente, se nos ha dificultado también volver a conectar los pines en su respectivo orden, consume bastante tiempo. El controlador L298N lo estamos conectando a la protoboard por medio de jumpers, como especificamos en una entrada anterior; cabe destacar que estos mismos jumpers nos daban problemas pues no se sostenían al componente, y decidimos mantenerlos en su posición, fijándolos con silicón caliente, así si debíamos mover el componente, no tardaríamos un tiempo significativo en buscar el jumper correcto en caso de que se desconectara. Intentamos darle un color significativo a cada pin, para recordar su función, por ejemplo, los pines del L298N que van al NodeMCU en las posiciones del D0 al D4, llevan un mismo patrón de colores, y son los únicos 4 pines del controlador que tienen ese color de jumpers. No fue tan fácil para los demás pines pues los colores que tienen nuestros jumpers son muy variados, y no pudimos definir algún significado especial para el resto, tuvimos que guiarnos por posiciones de los pines y las imágenes de Google que fueron de gran ayuda.

Más tarde, José se fue a su casa, nos faltaba conectar la parte del circuito que corresponde a los LEDs, así que acordamos que yo me quedaría el circuito y realizaría esa parte. Tuve problemas con el posicionamiento del circuito, ya que los LEDs requerían una conexión común al regulador de 5V, y tuvimos dos opciones, ponerlos cerca de este o cerca del Registro de corrimiento, decidí ponerlo cerca del registro y utilizar jumpers y cable UTP para hacer llegar esa línea positiva hacia los diodos. El circuito quedó un poco difícil de entender a primera vista, ya que, en poco espacio, se acomodó el registro, la resistencia que corresponde a cada LED, y el mismo LED, así que probablemente lo reacomodemos en el futuro.

**Lunes, 15/04/2019:**

**-Entrada de Ambos**

-Administrativa: Hoy acordamos vernos igual que el día de ayer, tuvimos el mismo problema con los buses, pero al fin y al cabo logramos reunirnos. Planeamos ir a comprar algunas cosas que vamos a necesitar más adelante.

Fuimos por las tiendas buscando soldadura, conseguimos 3 metros previendo que seguramente necesitaremos bastante, compramos más jumpers, la placa perforada (conseguimos dos por si se llegara a necesitar otra o a dañarse alguna), en mi casa conseguimos una bolsa de *zip-ties*, y también recurrimos a comprar una protoboard extra. Intentamos conseguir LEDs de más, pues en el kit no venían muchos en caso de que alguno se dañara, pero sólo encontramos uno de un tamaño diferente, y nos conformamos con eso.

- Técnica: Requerimos comprar la protoboard extra, pues estábamos teniendo problemas con el circuito una vez que sólo faltaba conectar los motores, originalmente teníamos ideado hacer el circuito bastante compacto desde el inicio, sin embargo, conforme fuimos teniendo complicaciones, tuvimos que ir separando cada componente para que, a la hora de hacer *troubleshooting*, no necesitáramos estar desconectando las piezas para llegar a otras más pequeñas.

Cuando conectamos los motores, notamos que debíamos ser bastante cuidadosos con los diodos, investigué un poco sobre el arreglo de 4 diodos para rectificar la corriente, pues en el diagrama no es muy comprensible este arreglo. No obstante, colocamos los diodos en la protoboard de manera que se asemejara al orden que tienen en el diagrama, así podríamos saber qué diodo pertenece a cada conexión sin tener que revisar los cables. Más adelante, intentamos probar la conexión del registro a los leds, con el código de Arduino, pero desde el inicio estábamos teniendo problemas con este sector del circuito, pues se esperaba que el registro mantuviera todos los LEDs apagados hasta recibir un comando que los encendiera, pero, sin importar cómo conectemos el circuito, el registro no actúa de la forma esperada, lo más cercano a controlar los LEDs que percibimos, es que al reiniciar el NodeMCU, el registro apaga o enciende algunos diodos aleatoriamente.

- Administrativa: Ha sido bastante frustrante no poder avanzar con el proyecto por algún error en el circuito. Es difícil detectar conexiones erróneas cuando ya todo está armado; aun así, queremos comprobar que todo está bien para poder avanzar con la programación del Node y así evaluar los comportamientos del carro.

Decidimos que no nos reuniremos en los próximos tres días (martes, miércoles y jueves) para tomar un descanso de este ambiente, y disfrutar un poco de la semana Santa. El compañero José se llevó el circuito hoy para su casa, en caso de que él quiera hacer algún reacomodo en estos días.

**Martes, 16/04/2019:**

**-Entrada de Alejandro Quesada:**

­-Administrativa: El día de hoy, como habíamos acordado, no nos reunimos para trabajar. Sin embargo, como el compañero José se había llevado el circuito, él lo desarmó varias veces y lo reconectó siguiendo los lineamientos del diagrama de Santi. Yo por mi parte, busqué algunos videos en los que se utilizaran Arduinos para solucionar problemas simples, con el objetivo de aprender más a fondo acerca del funcionamiento del proyecto, y lo que se espera que suceda en cada caso al enviar comandos. También investigué un poco sobre la herramienta *Tinkercad*, José me recomendó utilizarla, pues en ella podía probar sectores del circuito en un simulador, sin tener necesariamente el circuito conmigo. Parece una herramienta bastante útil; inicialmente no encontré cómo utilizar Tinkercad para la simulación de circuitos, hasta que luego comprendí que en general es una herramienta de modelado 3d y la herramienta *Tinkercad Circuits* es una subdivisión de esta.

- Técnica: Algo que se corrigió, fue un error encontrado por el compañero Kevin Masís, quien notó que los pines correspondientes al *Voltage Supply* y *Voltage Logic Supply* estaban intercambiados en el diagrama, lo que se convirtió en un problema menos a la hora de revisar las conexiones.

**Entrada de José Morales:**

-Administrativa: Hemos acordado que martes, miércoles y jueves serían días de descanso. Me he sentido algo exhausto a nivel mental por la cantidad de horas que se han puesto en el proyecto. Viajar todos los días e bus, tan seguido, tanta distancia es algo cansado.

A pesar del acuerdo de no trabajar, no he podido dejarlo por todo de lado. la mayoría del día la pasé descansando, sin embargo, en la noche revisé un poco más el circuito para evaluar por qué no nos funcionaba.

-Técnica:  Al estar concentrados primero en el funcionamiento del shift register, he revisado mayor documentación. me pareció algo raro que no nos funcionara a pesar de estar aparentemente bien conectado. Decidí probar si por casualidad algún pin del NodeMCU se había quemado, usando el ejemplo del “blink” en una protoboard aparte. Confirmé que todos los pines funcionaran.

**Miércoles, 17/04/2019:**

**-Entrada de Alejandro Quesada:**

- Administrativa: Al igual que el día de ayer, acordamos no reunirnos hoy, además, no trabajaríamos en el proyecto del todo, consideramos este como el día de verdadero descanso del proyecto, yo me tomé el día para salir, aunque José, quien se había quedado en casa, decidió usar parte de su tiempo para revisar si el circuito seguía teniendo algún error.

- Técnica: Luego de desarmar y recolocar el circuito en la protoboard varias veces, aún no se logra descubrir qué puede ser lo que ocasiona que las luces sólo cambien de estado al reiniciar el NodeMCU, es un comportamiento que no entendemos, suponemos que está relacionado al Registro de Corrimiento, pero es lo más que hemos entendido del problema hasta el momento. El NodeMCU también estaba teniendo problemas para conectarse a la red Wi-Fi, consultamos con Santi y no pudo ayudarnos a encontrar la solución, por el momento nos dijo que para cargar código y para encender el Node inicialmente, que lo hagamos fuera del circuito. Según un reporte del compañero José, el circuito (luego de ser rearmado distintas veces) sigue teniendo cortos en el sector de los motores, sospechamos que puede ser causa de los diodos, o del L298N, sin embargo, no hemos llegado al fondo del problema. Otro problema que tuvimos fue respecto a la batería de 9V, esta se descargó por debajo de la corriente que el NodeMCU necesita (según una corta investigación del compañero José, son 300mA). Debido a las complicaciones que podemos tener en los próximos días para conseguir materiales de repuesto (por la situación de la Semana Santa) José habló con el asistente Santiago, quien le dijo que intentará conseguirnos una batería de reemplazo para que podamos usarla cuando vayamos a colocar el circuito en la placa perforada. Por ahora, estamos usando la computadora como fuente de alimentación, aunque José tuvo la idea posteriormente de utilizar una batería portátil para el Node mientras solucionamos el problema; pensamos que sería una buena idea conseguir baterías recargables, aunque por el momento es algo complicado. En un futuro, sin embargo, consideramos que sería una buena inversión para los proyectos.

Tuvimos que investigar brevemente acerca del transistor que controla la fotorresistencia, pues unos compañeros destacaron que en el diagrama no especifica cuáles pines del transistor son el Emitter, Base, y Collector, buscamos nuestro modelo y verificamos que estuviera bien colocado según las necesidades del circuito.

- Administrativa: Hemos tenido algunos problemas personales respecto al lugar de trabajo, José me comunicó que se siente incómodo trabajando en mi casa, y no está de acuerdo con quedarse en dicho lugar para aprovechar más el tiempo, pues según lo que me comunicó, le genera disconformidad el pensar simplemente en la idea, así que tendremos que replantear el esquema de trabajo que llevamos hasta ahora.

**Entrada de José Morales:**

-Administrativa: Al igual que los días anteriores, decidimos no reunirnos. Seguí probando el circuito entre tiempos de descanso.

-Técnica: Decidí rearmar el circuito desde 0. Al probar el circuito noté que la corriente no estaba llegando bien a los leds, o al menos eso aparentaba. Dependiendo de la posición del L298N, podía ser que encendieran, resetearan el registro de corrimiento, entre otras anomalías. El mismo comportamiento se daba anteriormente. Al tocar los diodos directamente, me dí cuenta que los mismos estaban algo calientes, por lo que inmediatamente desconecté esa sección del circuito y decidí no realizar pruebas en él hasta después

**Jueves, 18/04/2019:**

**-Entrada de Alejandro Quesada:**

Administrativa: Respecto a los cambios en el esquema de trabajo, intentamos buscar espacios alternos para reunirnos y trabajar en el proyecto, le sugerí al compañero José la posibilidad de trabajar en su casa, e inclusive que aceptaran que me quede a dormir, debido a que siento que soy más capaz de adaptarme al ambiente de su domicilio que él al mío, y no me sentiría incómodo. Sin embargo, sus padres rechazaron la propuesta en su totalidad, por lo que su casa quedó descartada como lugar de trabajo. Hoy también fue un día de los tres que declaramos como “descanso del proyecto” aunque no fueron totalmente lo que esperábamos; no obstante, mantuvimos firme la aclaración de no reunirnos en estos tres días. Estuvimos hablando sobre qué hacer el día de mañana, que será el día que sí nos reunamos para seguir trabajando en el proyecto. Me di la tarea de consultar en la agencia de buses TUAN sobre su horario para esos próximos días, pues sería necesario utilizar su servicio, tanto para que el compañero viniera a mi casa, como para ir de vuelta a la suya. Descartamos que el servicio fuera un problema, pues sí van a tener horarios disponibles para los días viernes, sábado y domingo.

Le sugerí al compañero otra posibilidad: que viniéramos de vuelta a mi apartamento en Cartago, pues, medité sobre la situación y es de, todos los lugares, en el que he visto a José desarrollarse más cómodamente y con tranquilidad, por lo que hablé con mi familia para que nos puedan llevar allá el día de mañana. Acordamos entonces, reunirnos en mi casa para trabajar un poco el circuito.

Le comenté a José que me estaba sintiendo mal emocionalmente, que, aunque en términos académicos nos conviene más volver a Cartago para seguir trabajando en el proyecto, que la decisión me pone triste y tenso porque quería pasar la mayor cantidad de tiempo posible con mi familia, suponiendo que luego, en la siguiente mitad del semestre se me complicaría mucho verlos. Se mantuvo firme la decisión de volver a Cartago; además de eso, José me recomendó hablar de la situación con mi familia, pues él ve que me conviene conseguir ayuda psicológica para manejar el estrés, la ansiedad y otros problemas, que, si continúo posponiendo esa búsqueda de ayuda, me voy a ver muy afectado psicológicamente.

**Entrada de José Morales:**

-Administrativa: He conversado con él compañero Alejandro sobre el esquema de trabajo que hemos manejado. Soy una persona con ansiedad, por lo cual, entornos algo ruidosos pueden dejarme en un estado en el que no soy capaz de avanzar, y me vuelvo más un estorbo que una colaboración. He comentado esto con el compañero Alejandro puesto que días anteriores había resultado frustrante para mí no poder concentrarme a la hora de trabajar los días anteriores.

-Técnica: He repetido la construcción del circuito varias veces, sin embargo, en ninguna he logrado siquiera que funcione el registro de corrimiento. He consultado a algunas personas, sin embargo, a partir de una sola foto o video fue difícil para ellos dar con el problema

**Viernes, 19/04/2019:**

**-Entrada de Alejandro Quesada:**

Administrativa: Hoy logramos reunirnos un poco más temprano que los días anteriores, aunque no trabajamos por mucho tiempo. Mi mamá nos dio almuerzo al momento que José llegó a la casa, ya había arreglado la situación con mi hermana para que nos hiciera el favor de llevarnos a Cartago en la tarde, así que un rato después de almorzar, guardamos las cosas que necesitamos llevarnos.

El plan cambió un poco, no nos vendríamos a Cartago ambos el día de hoy, sino que aprovecharíamos el viaje que accedió a realizar mi hermana, para dejar a José en su casa; posteriormente me vendrían a dejar a mi apartamento, yo me dejaré el circuito y nos reuniremos el sábado.

Al llegar a Cartago, acomodé mis cosas y me relajé un corto periodo, luego, empecé a trabajar un poco en el código para el manejo de los motores.

-Técnica: Consulté con ejemplos en línea sobre las funciones que se espera que utilicemos (analogWrite, digitalWrite, reads, prints, entre otros) para manejar mejor los temas, sus entradas, las restricciones que tienen; la página de arduino.cc o Arduino Reference fue muy útil para este propósito. También le pedí un poco de guía a Alejandro Vásquez, él me dijo que ya habían logrado solucionar algunos problemas que tuvieron con esa misma parte y me aconsejó. Aprendí completamente a generar el código para los motores, me dispuse a generar mi versión de este, y cuando ya estaba completo, traje el circuito y el carro para probarlo. A la primera ejecución, tuve unos problemas, puesto que antes de hoy, no habíamos trabajado con el NodeMCU, por lo que tuve que descargar la biblioteca de ESP8266 y añadirla a las bibliotecas del IDE de Arduino. Creo que habría sido buena práctica si ambos hubiéramos instalado todo el mismo día, y no sólo mi compañero.

Luego de preparar todo para subir el código, esperé que cargara, y al cargarse en el NodeMCU, llegó el momento de probar el comando. Envié ***pwm:0;*** tal como se especificaba, pero no sucedió nada. Empecé a analizar el circuito detenidamente, pues estaba bastante seguro de que el código funcionaba como debía; en el circuito noté que las tierras de los diodos no estaban conectadas, más bien estaba cerrándose todo el arreglo de diodos a la línea positiva de 5V. Cuando conecté los diodos a tierra, creí haber solucionado el problema, hasta que me llevé la sorpresa de percibir un extraño olor a quemado proveniente del circuito. Me asusté y desconecté inmediatamente las fuentes de alimentación, no pude ver qué componente exactamente se había calentado tanto, pues en ningún momento liberaron humo. No obstante, sabía que había sido algún punto en el circuito de los motores, pues sucedió justo después de cerrar el arreglo de diodos; al pensar esto, tomé los 4 diodos para revisar que estuvieran en la polaridad correcta. Esto fue un problema grave y consumió bastante tiempo, pues a nuestros diodos se les había borrado la marca que indicaba el lado negativo de los mismos. Tuve que recurrir a una guía en línea que describía el proceso para comprobar dos cosas: primero, que la orientación de los diodos fuera la correcta, y segundo, que no se hubiera dañado el componente semiconductor que se encuentra dentro de la cápsula del diodo. Con esto, noté que dos diodos estaban en la orientación errónea, y que uno de ellos se había dañado. Dejé de operar con el circuito, le comuniqué a mi compañero de la situación, y me dediqué a investigar un poco del siguiente paso en la programación: los operadores lógicos para el manejo de bytes en el código del registro de corrimiento.

Hoy también empezamos a utilizar GitHub para el manejo en conjunto de nuestro código, investigamos un poco de cómo debe utilizarse, cómo crear el repositorio y cómo controlarlo entre nosotros, aunque ya el compañero José conocía más del tema. Descargamos GitHub Desktop y empezamos a dar nuestros propios *commits* de cambios al archivo en el repositorio dedicado para este proyecto. Tal como se esperaba de esta herramienta, está mostrando ser de bastante utilidad para llevar un registro de nuestro progreso con el archivo de código .ino, y seguiremos utilizándola para comunicarnos cuando no podamos trabajar en el mismo lugar.

**Entrada de José Morales:**

-Administrativa:

-Técnica:

**Sábado, 20/04/2019:**

**-Entrada de Alejandro Quesada:**

- Administrativa: El compañero José y yo empezamos a discutir sobre el problema que tuve con los diodos rectificadores, ya que él fue el último que había armado el circuito, y estaba muy seguro de que lo había dejado todo bien conectado. Insistí en que tuve que corregir la conexión a tierra, y que fue así como noté que los diodos estaban en la orientación incorrecta. Mientras José venía de camino a mi apartamento, hablé con otros compañeros para ver si alguno que viviera cerca era capaz de brindarnos diodos extra, porque los necesitábamos para probar el circuito.

Mientras esperábamos la llegada del compañero Brian Wagemans (quien se ofreció a ayudarnos a verificar las conexiones de nuestro circuito como una tercera opinión, además de brindarnos componentes extra que él y su compañero de equipo no necesitaron), desconectamos los motores por completo del circuito, para así probar otros sectores que también requerían evaluación, como los LEDs y el registro de corrimiento.

Estuvimos un rato verificando que las conexiones estuvieran en el lugar correcto. Luego de un tiempo, algunos compañeros destacaron que habían encontrado un error en el diagrama de Santi, y que el pin del CLR en el registro no debía ir a tierra, al contrario, este debía estar conectado a los 5V en positivo. Realizamos este cambio, pero probando con el código que teníamos, el registro seguía sin comportarse de la manera esperada, ahora, en vez de encender o apagar luces aleatoriamente, las iniciaba todas en un estado encendido, y al ir reiniciando el NodeMCU, el registro por alguna razón apagaba los LEDs uno por uno. Cuando llegó Brian con los componentes extra, intercambiamos nuestro registro 74LS164 por un repuesto que nos brindó Brian, también reemplazamos los diodos y verificamos que estuvieran en la orientación correcta. Brian nos brindó otros componentes que les habían sobrado, pues ellos (él y su compañero de equipo) compraron componentes de repuesto para casi todo el kit, debido a que no querían atrasarse en caso de dañar alguno. Aunque ello requiere un mayor recurso económico, puede ser una táctica bastante útil para trabajar en periodos como semana Santa, donde no todos los locales estarán a la disposición del estudiante.

Más adelante, Brian se fue; José y yo nos quedamos para seguir haciendo pruebas y evaluar el circuito. No logramos llegar a ningún avance, lo desarmamos, no llegamos a completarlo por el nivel de exhaustividad que sentíamos, y en el ámbito administrativo, tuvimos una serie de problemas que nos impidieron avanzar. José me hizo saber que había algunas conductas de mí que, con previo aviso, me había pedido cambiar por el bien del equipo de trabajo y la mantención de nuestros roles, que ambos somos personas con ciertos problemas emocionales y dificultades de adaptación, así que lo mejor para ambos era mantenernos serenos. Yo no cumplí con esta parte, y el compañero me hizo saber que yo en serio necesitaba mejorar en ese aspecto para seguir trabajando en conjunto, y no bloqueando el progreso de ambos.

En la noche, José habló con un amigo suyo, quien estaba dispuesto a ayudarnos con el troubleshooting, así que decidimos cambiar el esquema de trabajo. Acordamos que José trabajaría unos días la parte física del proyecto, A mí persona le correspondía generar el código para hacer que el carro funcione completamente, tal como lo acordamos.

**Entrada de José Morales:**

-Administrativa:

He hablado con el compañero Alejandro sobre las complicaciones que hemos tenido a lo largo del proceso del proyecto. Entre algunos problemas que conversamos fueron la duplicidad de tareas, inseguridad a la hora de proceder, entre otras cosas.

Hablando con Daniel Rojas, un amigo que cursa la carrera de Electrónica, el me dijo que podía ayudarme con el circuito. Le comenté que muchos de los  problemas se estaban dando probablemente por falta de conocimiento de nosotros, por lo cual, le dije que no era con el ensamblado del circuito en lo que quería que me ayudara, sino en revisar que en el proceso todo lo estuviera haciendo bien, o indicarme qué secciones estaba haciendo mal

-Técnica:

Hemos reemplazado el diodo dañado. Consultamos a más compañeros por posibles errores en nuestro circuito sin resultado positivo. Estoy seguro de que hemos cometido errores a nivel de circuito, puesto que viendo el código y habiendo probado todo desde el serial monitor, el NodeMCU si está enviando las señales. Es frustrante no saber que se está haciendo mal, en cierta parte, me gustaría que semana santa hubiese sido semana lectiva, buscar ayuda técnica es especialmente difícil cuando todas las posibles personas que pueden brindarla están de vacaciones.

**Domingo, 21/04/2019:**

**-Entrada de Alejandro Quesada**

-Hoy por la mañana, José y Daniel lograron solucionar el problema del registro de corrimiento. Lograron darse cuenta de que todo estaba funcionando, ya que le solicitamos a Alejandro Soto que nos brindara una versión de su código que nos funcionara para probar que todos los componentes del circuito funcionaban/estaban bien conectados, y no que fuera un error de que estaban dañados. Luego de comprobar que todos los componentes del circuito funcionaban como debían, descartamos el código de Alejandro, para evitar problemas futuros con similitud de versiones.

El amigo de José le dio algunos consejos, nos corrigió el uso de tanto silicón para sostener los jumpers que conectan el L298N al resto del circuito, por ejemplo; nos dijo que utilizar una liga de hule era una mejor y más simple opción.

Yo no iba a trabajar mucho en el código el día de hoy, debido a que el examen de taller de programación estaba programado para el día de mañana; sin embargo, el asistente Santiago conversó con el profesor Milton acerca de la situación, y acordaron posponer el examen para que tuviéramos tiempo de hacer el proyecto.

Debido a esto, en la tarde me dispuse a iniciar con las tareas faltantes del código, esto me iba a ayudar a comprender el orden y el trabajo de cada pin en el circuito, así como la utilización de los conocimientos aprendidos en el sitio de Arduino y en otras páginas como StackOverflow y Github.

Busqué ejemplos de cómo utilizar una fotorresistencia o *light-dependant resistor* para controlar la iluminación de un diodo, esto me dio una idea del comportamiento del circuito que corresponde a la LDR, y así poder aplicarlo en el código fuente para este proyecto. Discutí con Santiago cuál era la solución correcta a la telemetría, debido a que me generó confusión la especificación del proyecto, en la que dice que el valor de retorno del LDR es interpretado por el programador; no obstante, el compañero Alejandro Soto me mencionó que a ellos Santi les había dicho que lo mejor era que lo hicieran retornar un 1 o un 0, pues si no, tendrían problemas en la evaluación del código, pues este se revisaría con un script y si tiene valores distintos, perderían puntos. Esto fue lo que me llevó a consultar con Santi, yo había generado mi propia versión del comando para las luces, pero lo devolví a fin de cuentas a un valor entero de 0 o 1.

Para comprender el comportamiento del comando ***sense;*** fue necesario investigar sobre la función ***sprintf()*** de C, gracias a esa investigación breve comprendí cuál era el sentido de que se encontrara en el código, y así pude generar mi solución al comando. Inicialmente, había planteado que el valor ***light*** que se invoca en los argumentos de sprintf fuera el valor de retorno de una función, la cual, dependiendo de la lectura digital del pin D8 en el NodeMCU, retornaba un string que determinaba si el nivel de luz era “bajo” o “alto”.

Hoy la mitad de la telemetría estaba lista, así que el avance siguiente se realizará mañana. Por hoy, subí los cambios al repositorio de GitHub, y presté atención a las correcciones y consejos que José me dijo que había recibido de su amigo Daniel, ya que también es necesario que yo conozca y me lleve con los componentes y la circuitería.

**Entrada de José Morales:**

-Administrativa: Me he dirigido a la casa de mi amigo desde la madrugada. Como mencioné anteriormente, viajar es algo estresante, sin embargo, que he sentido motivado puesto que con algo de ayuda técnica veo plausible finalizar el proyecto.

-Técnica: Ha sido un día especialmente provechoso. En el proceso de prototipado, mi amigo Daniel me indicó alguno de los errores que había estado cometiendo, ya fuera por distracción o desconocimiento. Habían errores de puenteo de positivos y negativos, algunas conexiones del registro del corrimiento estaban mal hechas. El compañero me enseñó cómo comprobar de manera correcta si las señales están siendo enviadas desde el NodeMCU con el Multímetro, y como en general, testear que todo esté en el orden que debe. También me dió consejos de como posicionar las cosas en la proto para mantener el orden. Habiendo finalizado el circuito, lo probamos con un código proporcionado por el compañero Alejandro Soto, para verificar que en todos los días que habíamos hecho pruebas no se hubiera dañado ningún componente. Todo funcionó por completo excepto los motores. Luego de evaluar un poco más, Daniel notó que el L298N no estaba haciendo contacto con las bases de jumpers que habíamos hecho, probablemente por el mismo silicón que habíamos usado. Le quitamos el silicón a la base, y el compañero le puso una liga para presionar más fuertemente los jumpers. probamos de nuevo y todo funcionó. Decidí que sería bueno entonces empezar a soldar, puesto que Daniel me podría aconsejar también durante el proceso para evitar malas prácticas. Me dió un taller rápido de cómo soldar cosas de manera consecutiva, y efectivamente desde ese taller me siento con mayor confianza a la hora de soldar. El registro de corrimiento y las bases del NodeMCU quedaron soldadas. Elabore un croquis para saber dónde posicionar cada pieza. Al ser ya las once de la noche, me he retirado de la casa de mi compañero, sin embargo, le he pedido una “tercera mano” para terminar el proceso de soldado.

**Lunes, 22/04/2019:**

**-Entrada de Alejandro Quesada**

-Técnica: José me dijo que le está siendo muy de utilidad estar en la casa de Daniel, ya que, luego de verificar que todas las conexiones estaban bien, empezó a soldar el circuito a la placa perforada, y para esta tarea, Daniel le prestó unas pinzas con lupa que se conocen como “Tercera Mano” o “Manos ayudantes”, al igual que un desoldador para absorber estaño mal aplicado en caso de que sea necesario. Estas herramientas no se incluyeron en el kit que se nos vendió, imagino que por razones económicas, pero siento que sería de gran utilidad tener herramientas de ese tipo bajo nuestra propiedad, para proyectos futuros.

-José me comunicó que a la hora de soldar le surgió un problema, llevaba un tiempo ya soldando otros componentes, pero no sabía que para el transistor había que tener un poco más de cuidado, pues este soporta una menor exposición al calor. Al intentar soldar este componente a la placa, dejó de funcionar correctamente el material semiconductor dentro del transistor, ya que la corriente pasaba en ambas direcciones, y perdió su funcionamiento como “diodo”. Por suerte, parte de los componentes extra que Brian W. nos cedió era un transistor del mismo tipo, por lo que no generó mayor problema esta situación, lo que sí fue complicado es que se dio luego de que José se fuera del domicilio de Daniel, por lo que ya no tenía el desoldador a mano, y tendremos que esperar un tiempo para avanzar en ese aspecto del circuito.

Discutimos sobre la parte estética del carro, si usaríamos los LEDs que conseguimos, o dejaríamos las luces frontales que venían incorporadas en la carcasa del carro; José sugirió que utilizáramos los que vienen incorporados en el carro, ya que estos tienen el mismo nivel de luminosidad (y nosotros, al haber dañado uno de los LEDs blancos que teníamos en el kit, sólo disponíamos de dos LEDs diferentes). También planteamos cómo instalaremos las luces traseras y las direccionales, debido a que las únicas luces que el carro nos facilita para colocar son las frontales, ideé que podíamos lijar la capa interior de pintura de la carcasa, ya que, en la sección de luces traseras, esta sí brinda un material translúcido de color rojo, así que no sería necesario perforar completamente la superficie, pero para las luces direccionales, sí se necesitaría. Definimos que, en caso de ser necesario, perforaremos el carro con el cautín para pasar los diodos, sellar con silicón y así darle un aspecto no tan estético, pero sí práctico.

Sobre el código: Le solicité a José que me brindara con un dato necesario para la telemetría de la batería, necesitaba saber en cuánto está la lectura analógica del pin A0 al momento en el que las baterías llegan a su carga máxima, y me dijo que está alrededor del 745 en pwm. Este dato será útil para programar el sensor de nivel de batería más adelante.

**Entrada de José Morales:**

-Administrativa: He estado en mi casa todo el día, metido en mi cuarto, soldando.

-Técnica: Tuve un problema soldando la sección de la fotoresistencia, al parecer calenté demasiado el pin base y emisor del transistor, y he descompuesto el diodo interno del transistor. Tenemos un repuesto, no es un 3906, sino un 4125, el mismo provisto por el compañero Brian, sin embargo, creo que entre todo el desorden de soldado y desoldado, se quemaron las donas y debía re-soldar el circuito completo. Es tan fatal que definitivamente creo necesitar una herramienta para quitar soldadura, puesto que no tengo algo parecido a mi alcance.

**Martes, 23/04/2019:**

**-Entrada de Alejandro Quesada**

-Hoy, el compañero José continúa dejando el carro preparado para las pruebas del código, mientras que yo estoy generando las soluciones a los problemas de programación que habíamos tenido anteriormente.

José me notificó que debía cambiar el comportamiento de los operadores lógicos en el código que tenía generado para las luces, ya que, teníamos los LEDs en desorden, y para controlar con más facilidad el comportamiento de estas, al igual que para darle más legibilidad al código, pondría cada juego de LEDs apareado, en orden de frontales, traseras, y laterales. Corregí en el código el comportamiento tal como se me solicitó, comprendí mejor el uso de los operadores lógicos en C, y cómo se manipulaba el byte de envío al registro, dependiendo del argumento **MSBFIRST** o **LSBFIRST,** pues según estos argumentos, la idea lógica de acomodar el byte según los pines a los que estaba conectado cada diodo podía verse alterada. Concluimos que debía utilizarse LSB (least significative bit) para nuestro propósito, pues de este modo, el bit más significativo corresponde a la salida QA del registro de corrimiento, haciendo más comprensible cada operación lógica con los dos bytes, que es lo que queríamos lograr.

-Me dio curiosidad que el IDE de Arduino no tuviera un tema oscuro integrado, así que busqué en internet si existía alguna forma de importar un tema creado por alguien más. Encontré en GitHub un archivo comprimido (.zip) que contenía las carpetas necesarias para instalar el tema oscuro, junto a videos de guías para el mismo. Luego de instalarlo, me sentí mejor a la hora de utilizar el IDE y siento que es bastante beneficioso haber instalado este tema.

**Entrada de José Morales:**

-Administrativa: Todo el día he buscado un removedor de soldadura, sin embargo, no parece haber ninguno disponible para pedir prestado, sin embargo, mi amigo Daniel me ha comunicado que me puede prestar el suyo, el cual se encontraba en casa de sus padres en Escazú. al ser relativamente cerca, tan pronto llegué a mi casa, me subí en mi bicicleta y fui por el removedor. Apenas he regresado a mi casa he procedido a arreglar la sección del circuito descompuesta.

-Técnica: He desoldado la parte del circuito de la fotoresistencia para retirar el transistor viejo y colocar el nuevo. He vuelto a soldar todo a una distancia considerable de donde estaba anteriormente el circuito, y he dejado un espacio entre los pines del transistor para evitar sobrecalentarlo como la vez anterior. Todo parece funcionar, sin embargo, hay anomalías. El Node parece no encender si está conectado a la fotoresistencia. Decidí desoldar del pin D8 el cable que conectaba al circuito, para revisarlo posteriormente.

**Miércoles, 24/04/2019:**

**-Entrada de Alejandro Quesada.**

-Estamos manteniendo el mismo esquema de trabajo, José está encargándose de terminar el aspecto físico lo antes posible, yo me estoy tomando mi tiempo para aprender sobre el código, mientras él me explica los cambios que ha llevado a cabo en el hardware para que yo comprenda cada movida realizada.

-Técnica: Por ahora, se cambió el valor de retorno de ***sense;*** específicamente el que la función sprintf() invoca en el valor de la luz: Anteriormente era una función que retornaba un String, pero toda esa sección del código se comentó, y se añadió una línea en la que se cambió la variable ***light*** a un digitalRead del pin D8. Esto simplifica la tarea de la telemetría, pues con esta función, sólo puede haber un valor de 1 o 0 en el pin. También, discutimos sobre los comportamientos esperados para los movimientos especiales, yo definí el código para los movimientos de comando ***Circle, ZigZag, e Infinite***, aunque sólo con un tiempo preliminar, puesto que esta parte del código se está manejando por medio de delays en milisegundos, y hasta no tener el carro listo completamente para realizar las pruebas de estos movimientos, no podrán definirse con la exactitud necesaria.

También logramos avanzar con la programación de la telemetría de la carga para la batería, le solicité ayuda al compañero Brian W vía WhatsApp para comprender el problema, ya que tanto José como otros compañeros me advirtieron que era más *tricky* de lo que parecía. Yo había buscado sobre ejemplos de medida con ecuaciones de la Ley de Ohm utilizando el pin analógico de un Arduino, pero no me fueron de mucha utilidad, porque la fuente de alimentación que vamos a medir es mayor a 5V, por lo que se debe calcular con métodos alternos a simples cálculos de corriente. Utilicé un valor que José me había brindado previamente, que la batería ronda los 745 en pwm al estar completamente cargada. Brian me dio la idea de utilizar este valor para hacer una función lineal que se aproximara al comportamiento de la batería, utilizamos una regla de 3 para obtener el valor de pwm en el que la batería debería estar para que se entienda como un 80% de carga. Luego de utilizar la regla de 3, encontramos que, a como 745 es 100, 596 es aproximadamente 80, con esto logramos modelar una recta, y utilizamos la herramienta en línea de geogebra.org para calcular la gráfica y la función algebraica, con esto despejamos el valor de X en la función, y logramos definir una operación matemática, la cual, al tomar el analogRead del pin A0 como uno de sus valores, encuentra el valor porcentual en el que se coloca la recta, en otras palabras, la x. Debido a la poca legibilidad y la complejidad de este proceso, se describió a este con un conjunto de comentarios en el código .ino antes de definir la línea de código, para así darle una buena documentación.

**Entrada de José Morales:**

-Administrativa: Mi horario es algo incómodo para trabajar el proyecto, sin embargo, tan pronto he llegado a mi casa me he propuesto arreglar el problema de la fotoresistencia. me he dormido a altas horas de la noche, sin embargo, he identificado el problema. Soldé lo que restaba del circuito y el mismo está prácticamente listo

-Técnica: Luego de un arduo proceso de investigación me he dado cuenta de que, por diseño, el circuito no funciona si hay luz - eso es, si la fotoresistencia es expuesta a luz. Inicialmente creí que había sido yo el que había hecho algo mal, sin embargo, pude librarme de esa autoacusación. Al fin y al cabo, el problema era que el pin D8, el que originalmente tenía conectado la fotoresistencia, bloquea el proceso de booteo del NodeMCU si al inicio lee un HIGH de entrada  - si la fotoresistencia recibe luz, deja pasar corriente que es detectada como un HIGH por el Node. confirmé el problema tapando la fotoresistencia y verificando que todo prendía sin problemas. Le comuniqué a Santiago el problema, mientras, como solución, decidí que sería el único pin del Node “desprendible”, soldándole un cable con cabeza de jumper en vez de soldarlo de forma directa

**Jueves, 25/04/2019:**

**-Entrada de Alejandro Quesada.**

-El día de hoy no se trabajó mucho en el proyecto, José se desempeñó en la parte estética del carro, mientras yo hice algunas correcciones en el código para los comandos de movimiento. Queda pendiente realizar el comando especial, pero si todo lo que hemos hecho hasta el momento sale bien, eso no generará mayor problema. Como aspecto a destacar, debido a los últimos días que he pasado trabajando en el código, siento que GitHub Desktop es una herramienta casi indispensable para tener un manejo organizado del código, es bastante útil.

**Viernes, 26/04/2019:**

**-Entrada de Alejandro Quesada**

-El día de hoy, José terminó de conectar el circuito de motores a la placa perforada por medio de las soldaduras requeridas, yo no trabajé en el código debido a que tenía clases en la mañana y en la tarde ambos teníamos la defensa de nuestro proyecto 1. A mí no me fue tan bien en esta, pero aprendí de los errores corregidos por el profesor, y espero que, con nuestro avance en este proyecto, note la mejoría que hemos tenido.

-No trabajamos más en el proyecto debido a que tenemos un examen de matemática general mañana por la mañana.

**Lunes 29/04/2019:**

**-Entrada de Alejandro Quesada**

-El día de hoy, mi compañero José me notificó que lo único que falta para que tengamos el carro preparado para las pruebas del código es colocar los LEDs en su posición respectiva, yo estoy a la espera de que podamos realizar dichas pruebas, ya que el código está prácticamente listo y sólo nos queda evaluar los tiempos de delay para los comandos de movimiento.

-No trabajamos mucho en el proyecto debido a que tenemos examen de taller de programación el día de mañana, y debemos prepararnos lo más posible para este.

**Martes, 30/04/2019:**

**-Entrada de Alejandro Quesada**

- Hoy, el carro está prácticamente terminado, sólo falta posicionar la fotorresistencia en un lugar estratégico y aislar los contactos en el carro para evitar que haya algún corto en el circuito interno, luego de las clases, José y yo nos reunimos en el laboratorio K1-206 para realizar las pruebas del código. Procedimos a colocar el carro en una base elevada hecha por nosotros mismos (un par de rollos de celotape) para mantener el auto fijo en una posición, comprobamos todos los comandos del código, y en efecto, todo funcionaba correctamente. Cuando estábamos a punto de probar si el carro efectuaba los movimientos correctamente, topamos con una barrera inesperada: al poner el carro en el suelo, no fue suficiente, ni con la potencia máxima (comando de 1023), para que el carro se moviera una vez puesto en el suelo. Conversamos sobre la situación con Santi, nos aconsejó que utilicemos más baterías para darle más fuerza al motor, y que en caso de que no logre hacer que el carro avance, lo veríamos más adelante. Estamos esperando que esto funcione, pero en caso de que no salga bien, que al menos el profesor Milton comprenda que es una situación fuera de nuestro control; programamos y conectamos el carro correctamente, pero es más un problema de la calidad del carro y el peso para el que estaba diseñado, que de desarrollo en el proyecto.

Quedó pendiente evaluar un error en un comando, el ***pwm:0;*** este comando daba el comportamiento esperado, detener el carro, aunque tenía un problema peculiar: en el TelemetryLog, el comando retornaba un error de **utf-8 codec**, haciéndonos ver que había algún problema con el código. Mañana investigaremos la fuente de este error, mientras intentamos solucionar el problema de las baterías.

**Entrada de José Morales:**

-Administrativa: Hemos trabajado en un laboratorio del edificio K1 por la noche para evaluar el programa que el compañero Alejandro había desarrollado. Todo funcionaba a nivel de programación, sin embargo, nos dimos cuenta que el peso del circuito sobre el carro era demasiado como para que el mismo fuera capaz de operar bajo condiciones normales. Hemos decidido que uno avanzará el trabajo escrito, mientras el otro trata de resolver el problema de la potencia.

-Técnica: Todo el circuito se encuentra funcional. tuvimos un pequeño susto porque en un momento salió humo durante las pruebas, sin embargo verificamos que se dió un corto en la resistencia de un led por un contacto no deseado con otro cable. Verificamos de nuevo las instrucciones y todo funcionó. probé los componentes según las enseñanzas de mi amigo Daniel y todo se encontraba en perfecto estado.

**Miércoles 01/05/2019:**

**-Entrada de Alejandro Quesada**

-Hoy estamos trabajando en los detalles finales del proyecto: José está buscando cómo colocar más baterías en el carro (hasta 2 más según nos indicó Santi), mientras que yo consulté con el compañero Alejandro Soto sobre el error que estábamos teniendo con el comando de pwm:0;  Alejandro me ayudó a escanear el código del WiFiClient.py, y con este nos dimos cuenta que el error estaba en un mensaje de retorno que tenía nuestro código, en el que se usaba un carácter no-ASCII para notificarle al usuario que estaba dando un valor erróneo. Se corrigió el mensaje, y el código se comporta de la manera esperada.

**Entrada de José Morales:**

-Técnica: He arreglado el problema de la batería de una forma particular. corté un jumper hembra y otro macho, compré cinta aislante y tomé dos baterías restantes para ensamblar un paquete de baterías en serie. gracias a que previamente las conexiones de las baterías del carro ya estaban en cabezas de jumpers, agregarlo no fue difícil (de hecho, para ese tipo de adaptaciones fue que pensé en agregar las cabezas de los jumpers).

Cabe notar que nuestros problemas de baterías pueden rastrearse a que las mismas se descargan relativamente rápido, y no tenemos mucho dinero como para estar comprando cada vez que muestran signos de debilidad. Alejandro y yo hemos conversado sobre comprar baterías recargables, sin embargo, por el precio de las mismas, se vuelven algo inaccesibles en el presente, a pesar de que a largo plazo sean más efectivas en cuanto a costo/beneficio se refiere

En cuanto a lo demás, he comprobado el código del compañero Alejandro por completo. el mismo es 100% funcional, sin embargo, dividí ciertas tareas en módulos aparte para que fuera más fácil entender directamente lo que hace el código. También corregí algunas de las trayectorias basado en las pruebas con el carro completamente ensamblado, aunque en general, como se mencionó antes, el código funciona por sí solo sin problema alguno.

**--------------------------------------------------------------------------------------------------------------**

**Parte II:**

**Alejandro Quesada.**

**Viernes, 17/05/2019.**

**Administrativa:** El día de hoy, los integrantes del equipo nos sentamos a definir los roles que tendríamos a lo largo del desarrollo de esta fase del proyecto, el plan al hacer esto es que podamos tener un plan de trabajo bien definido desde el inicio y tengamos la posibilidad de rectificar la calidad de nuestro trabajo con el profesor encargado del proyecto.

**Técnica:** Al definir los roles, acordamos que me corresponderían las tareas relacionadas al *Front-End developer*: Documentación del código, verificación de calidad (en general) y ser *Back-End support.* Mientras se asignaron estos roles, entre mi compañero y yo también acordamos que su rol sería el *Back-End developer*: correspondiente a la coordinación del trabajo y dar soporte en la documentación. Cabe destacar que la definición de estos roles es el paso inicial del trabajo y pueden llegar a cambiar según avancemos en el desarrollo del trabajo.

**Sábado, 18/05/2019.**

**Administrativa:** El compañero José se encargó de la organización inicial del trabajo antes de comenzar el desarrollo.

**Técnica:** Hoy se creó el repositorio en GitHub en el cual se contendrá todo lo necesario para el funcionamiento del proyecto a medida que avancemos en este. El repositorio consiste de tres ramas, la principal, y una rama para cada desarrollador, inicialmente, estas subramas son una copia de la principal en la que cada uno tendrá su espacio para trabajar y guardar las modificaciones del repositorio; más adelante (cuando sea necesario) utilizaremos la función *merge* de git para actualizar el trabajo de cada uno respecto a la rama del otro integrante del equipo. Para esto y durante el resto del trabajo, estamos utilizando la versión de escritorio de Git, *GitHub Desktop*, la consideramos sumamente útil ya que agiliza el trabajo y simplifica el entendimiento de cómo funciona Git a comparación de la consola base de esta herramienta. Igualmente, los conocimientos adquiridos en el Taller de Git dado en tiempo lectivo fue de ayuda para comprender mejor el funcionamiento de esta herramienta.

Luego de que José creara el repositorio, me dediqué a crear la base de la interfaz para este proyecto. Para esta tarea, utilicé la biblioteca Tkinter de Python, el entorno de desarrollo integrado de Python y la versión 3.7.2 del lenguaje. Gracias a que realizamos un proyecto con estas mismas herramientas anteriormente, pensé que sería fácil adaptar los conocimientos previos a las necesidades de este trabajo.

Hasta el punto de desarrollo actual, logré que existan las ventanas solicitadas en el pdf del proyecto, y que pueda haber una comunicación entre ellas, básicamente basándome en comandos de retorno que destruyen una ventana y muestran la ventana “destino”.

**Extensión de la Administrativa:** Hoy logré completar esta tarea de generar el código base para la interfaz y lo subí al repositorio de GitHub; creí que iba a ser una tarea bastante sencilla, pero noté que (por el tiempo que ha pasado desde que realizamos el primer proyecto, y lo poco que logré completar este mismo) tuve que investigar un poco sobre la sintaxis de Tkinter (sus funciones y comandos). Para esto fue útil volver a consultar el sitio effbot.org/tkinterbook.

**Lunes, 20/05/2019**

**Administrativa:** El día de hoy no hubo avances **significativos** en el proyecto de mi parte. Esta semana estaré ocupado por quices, tareas y otras evaluaciones, así que probablemente no pueda avanzar en el proyecto hasta el fin de semana.

**Técnica:** Modifiqué el archivo del código que se tenía para las ventanas, a este le añadí una versión muy preliminar de los contenidos de la ventana de about o créditos, para la configuración de esta utilicé una función de creación de imágenes que nos había cedido el asistente Santiago para el primer proyecto, y al haber utilizado su TelemetryLog en el proyecto pasado, encontré una herramienta interesante para el display de texto, que es el ScrolledText; esta permite insertar un texto amplio en un espacio pequeño, lo cual será útil para aprovechar más el espacio disponible en las ventanas. Para la ventana about, también se creó una imagen de prueba que funge simplemente como *place-holder* para cuando incluyamos las imágenes de los autores del trabajo. Sobre el trabajo en general, también se modificaron las dimensiones de las ventanas de la interfaz. Para esto fue útil una herramienta de la cual desconocía en el primer proyecto, la window.geometry, esta permite insertar un string con las dimensiones de la ventana de la forma (“1280x720”) lo que la hace más fácil de configurar, y también vuelve más legible este parámetro, dado que es más notable que corresponde a una relación de aspecto.

**Sábado, 25/05/2019**

**General:** Hoy no trabajé en profundidad en mis tareas del trabajo. Los compañeros de clase David Solís y Miguel Mesén (quienes conforman otro equipo) van a reunirse hoy para avanzar en su propio trabajo; me comuniqué con ellos para aprovechar dicha reunión y estar presente en caso de que necesitara ayuda con el desarrollo de mi propio trabajo, ya que hasta la fecha, David ha avanzado bastante justo en lo que me corresponde desarrollar, por lo cual tenerlo a la mano en caso de necesitar sus fuentes de investigación o un poco de orientación con la lógica operacional del Test Drive.

**Técnica:** Avancé la parte de la interfaz que corresponde a la ventana Test Drive, pero para lograrlo, necesitaba tener los archivos Python dados por Santi en el segundo proyecto, el TelemetryLog y el WiFiClient, ya que estos son cruciales para comunicar la parte programada con el hardware en el proyecto. Al copiar estos archivos y ponerlos dentro del mismo dominio del código para la interfaz, fui capaz de importar algunos métodos de estos, como la función send.

Investigué sobre cómo utilizar la misma función para devolverse de una ventana a la principal, utilizando esta ventana como argumentos, ya que para simplificar el código había intentado simplemente enviar el comando de la ventana como un argumento; pero al intentarlo noté que había un error de código. Luego de investigar, aprendí que esto se puede realizar por medio de expresión lambda, al consultar en YouTube sobre cómo cambia la funcionalidad de un comando o de un proceso (en comparación con situaciones que no requieran utilizar esta expresión).

Luego de realizar este cambio en la utilización de un botón de retorno, me dediqué a buscar imágenes para llevar a cabo el desarrollo de la ventana Test Drive justo como quería realizarla. Para esto, era necesario utilizar imágenes en formato png o Portable Network Graphics, dado que, si se quiere utilizar una imagen con fondo transparente, deben crearse justo encima del canvas (o sea, sin utilizar un label para colocar la imagen sobre este) y también que la imagen por sí misma no posea un color de fondo.

**Domingo, 26/05/2019**

**Administrativa:** Aún me encuentro en el domicilio del compañero David, cada uno está enfocado en sus propias tareas y sólo nos consultamos en caso de necesitar despejar alguna duda sobre el proyecto.

**Técnica:** Aún me encuentro trabajando en dos enfoques principales de la ventana Test Drive: programar los aspectos funcionales para los eventos en el teclado, y también ir adaptando la interfaz gráfica a lo que se desea obtener al final de esta etapa. Respecto a este último punto, estuve buscando imágenes que coincidan más con lo que buscamos para el trabajo; es por esto por lo que algunas imágenes hoy fueron eliminadas y otras fueron añadidas a la carpeta en el repositorio.

También, estuve trabajando en la parte lógica específicamente para los comandos de dirección en el vehículo. Para esto, modifiqué la función *send*() de TelemetryLog, y esta versión modificada la incluí en el código de nuestro trabajo. Esto porque es más conveniente utilizar simplemente esta función para hacer llegar un comando al NodeMCU, que llamar a la función del archivo cada vez que fuera necesario.

Para el manejo de eventos en este proceso, se está utilizando un *keybinding* en la ventana Test Drive, estos binds se asignan a funciones que se activan cuando existe un evento en el teclado; para esta tarea en específico se está trabajando con KeyPress y KeyRelease, por lo que las funciones sólo se activan con eventos de presión de una tecla, y la liberación de esta misma.

Para controlar los eventos con más facilidad, se añadieron variables globales para manejar estos eventos; hasta ahora con los que se ha trabajado es con los eventos de movimiento. Para estos eventos, no basta con llevar un control de las teclas, ya que puede haber varias generando eventos al mismo tiempo; por lo cual recurrí a la implementación de hilos y así ser capaz de ejecutar subprocesos paralelamente sin depender de la ejecución secuencial del código. Para ayudarme con estos eventos, consulté de nuevo la página effbot.org esta vez en el directorio de “Events and Bindings” https://effbot.org/tkinterbook/tkinter-events-and-bindings.htm

También se añadió texto al canvas con un formato que no había utilizado anteriormente, consiste en generar el código sobre el canvas sin la necesidad de utilizar un label como contenedor para el texto. En el canvas de la ventana principal, se añadieron textos que serán place-holders para la información de los pilotos, cuando eventualmente se incluya ese módulo.

Consulté con el asistente del curso, Santiago Gamboa, sobre algunos aspectos de las especificaciones del proyecto que no me quedaban del todo claro, por el momento lo que logré aclarar es sobre las luces direccionales, creí que estas debían encenderse automáticamente cuando el carro toma una dirección; Santiago me explicó que estas no debían complicarse tanto, simplemente debía asignársele un botón a las luces para que estas pudieran encenderse independientemente. Este cambio se está implementando actualmente en la lógica del trabajo.

**Lunes, 27/05/2019**

**Administrativa:**  Hoy fui capaz de trabajar en el proyecto hasta horas de la tarde porque tenía clases en la mañana.

**Técnica:** Estuve teniendo problemas para validar casos especiales en la utilización de las teclas para el control del carro: estas estaban funcionando parcialmente, debido a que si se ejecutan una vez se comportan de la forma esperada, pero mantener la tecla presionada estaba generando múltiples eventos y alterando el funcionamiento del código, para solucionar este problema tuve que dedicarle bastante tiempo a replantear lo ya programado antes de seguir con los próximos pasos; tuve que añadir nuevas variables para forzar al sistema a trabajar con una única presión de la tecla: ahora se comportan de modo que para repetir un proceso, se debe liberar la tecla primero. Luego de corregir estos errores, procedí a centrarme en los eventos faltantes, ya que por el momento la funcionalidad del Test Drive (respecto al control del hardware) está cerca de completarse; sin embargo, aún no he implementado la operación con imágenes para dar la retroalimentación al usuario y que este sea capaz de notar los elementos del auto que están en funcionamiento a medida que lo vaya manipulando.

Hoy el compañero José y yo actualizamos nuestras ramas para que estén las 3 sincronizadas, hicimos un merge a la rama principal y cada uno actualizó su rama correspondientemente. Estar trabajando cada uno con su rama ha sido útil para llevar los archivos organizados y evitar conflictos.

Continué trabajando en la lógica del programa (respecto al Test Drive), en la noche logré que las direccionales funcionen completamente en términos operacionales, sin embargo, aún está pendiente la parte estética/gráfica de estas funciones.

Estuve editando la imagen que cumple como *layout* del auto (o sky view) pues esta tenía un aspecto extraño de tener una sombra a un lado del carro, para todas estas ediciones estuve utilizando una versión online no oficial de Photoshop y me ha sido de gran ayuda para preparar las imágenes para esta ventana.

**Martes, 28/05/2019**

**Administrativa:** Hoy y los próximos días se me dificultará trabajar en el proyecto debido a las clases y asignaciones que les corresponden. Específicamente hoy, logré dedicarle tiempo hasta en la tarde.

**Técnica:** Hoy requerí detener el avance del proyecto para corregir una implementación errónea en la lógica del código respecto a la aceleración y desaceleración del auto: como en los días pasados había tenido conflictos con las luces direccionales y su comportamiento, supuse que la aceleración del auto debía acercarse del mismo modo a la de un auto de verdad; tal que, si se libera el acelerador, el auto debe ir disminuyendo la velocidad poco a poco. Consulté sobre la situación con Santiago, quien me orientó y me explicó el comportamiento esperado para este aspecto (cito textualmente): “Cada vez que yo presiono el acelerador, él manda pwm:algo; eso lo ejecuta una vez, cuando yo suelto la tecla, el carro deja de recibir pwm:algo y ahí se queda”.

Debido a la explicación anterior, tuve que hacer un cambio mayor en la lógica de movimiento, tenía un hilo que se activaba cuando ni la W ni la S (Que son parte de nuestras 4 teclas de movimiento, W-A-S-D) se están presionando, y este aproximaba la velocidad hasta que fuera 0. La corrección dejó un nuevo comportamiento, que cumple con las especificaciones dadas por Santi.

Luego de corregir este problema, trabajé en las imágenes para las direccionales y las luces frontales con el objetivo de que estas aparezcan en pantalla, y al dejarlas preparadas, se guardaron en el repositorio.

Santi más tarde me recomendó que adaptara mi lógica para que no mande valores de pwm pequeños (aproximados a 0 entre 400 y -400) para evitar que el auto se quedara estático emitiendo pitidos, y adapté los hilos, de forma que dentro del while principal del hilo, existe un while secundario que controla este rango de valores, y no es hasta que se supera la condición de operación de este loop que se empiezan a enviar los valores al auto. Como llevo poco tiempo practicando con estos loops, las condiciones parecen no estar comportándose explícitamente como se escriben, ya que si le determino al while que se mantenga activo mientras la potencia sea menor que 1000, el hilo termina superando este while y enviando un comando de más. Estos errores en las condicionales de los loops deben corregirse más adelante, por ahora la prioridad es llegar todo a un estado funcional.

**Jueves, 30/05/2019**

**Administrativa:** Estamos en los últimos días de clases, aproveché la tarde y me fui a trabajar al LAIMI 1 para recrear la ventana Test Drive.

**Técnica:** Hoy me dispuse a verificar si el código estaba siendo 100% funcional, para esto tomé el código ya existente para la ventana Test Drive, lo copié en un nuevo archivo de Python, y lo trabajé por fuera del repositorio; esto con el objetivo de depurar el código obtenido hasta el momento, y mejorar la funcionalidad del archivo. Este nuevo archivo se llamó *NewGUI*, básicamente una versión mejorada de la interfaz gráfica. Se avanzó en funciones de movimiento, encendido y apagado de luces para el auto, y estas son completamente operacionales. Respecto a la estética sólo faltan dos aspectos: la aparición de las luces traseras y de las ruedas delanteras al virar. Aún no existe un hilo o función que se encargue de operar las luces traseras, pero este se trabajará más adelante.

También se corrigió un problema con la implementación del parpadeo de las luces direccionales: estas se habían programado para funcionar exclusivamente, por lo que, si el usuario presionaba la tecla que enciende una luz e inicia el loop iterativo de su parpadeo, y posteriormente encendía la otra luz, la más reciente cancelaba la primera en activarse (con tal de que solo existiera una parpadeando a la vez). Por sugerencia y orientación de Santiago, esto se corrigió para que las luces sean independientes, y no sólo puedan encenderse sin estar moviendo el carro, si no que también puedan activarse simultáneamente. Para esto, decidí que las luces mantendrían el mismo binding en el teclado, la Z para encender la izquierda, la C para encender la derecha, y la X para apagar cualquiera de ambas que esté activa al momento de presionar dicha tecla.

Todos los cambios que se hicieron el día de hoy se estuvieron trabajando en el archivo NewGUI.py, sin embargo, por la tarde se tomó el código de este archivo, y se creó un nuevo archivo en el que se trabajó este código, nuevamente para mejorarlo, todos los cambios que se le hicieron al código se guardaron en este archivo con el nombre de DebuggedGUIv2. Hasta ahora, ha sido bastante útil trabajar los hilos con iteración, si se compara a la recursividad implementada en el primer proyecto, es bastante más simple.

Además, se revisó un problema menor que existía con el código NodeMCU, dado que este no se había utilizado hasta la fecha, por lo cual no se encontraba dentro de una carpeta con su mismo nombre (lo cual fue un problema debido que el Arduino IDE requiere que se encuentren los archivos en este formato). Se añadió el archivo a una carpeta con el mismo nombre y se solucionó el problema.

**Viernes, 31/05/2019**

**Administrativa:** Hoy fue el último día de clases, pude dedicarle más tiempo al trabajo por esta razón. Mi compañero (y los estándares de entrega que definimos anteriormente) requerían que la funcionalidad del Test Drive estuviera preparada para hace un par de días, por motivos de atraso aún falta que funcionen completamente.

Me quedé trabajando con el código, en conjunto con el auto, hasta las 5 de la mañana del sábado.

**Técnica:** La ventana de Test Drive aún se está trabajando en un archivo ajeno al principal, el DebuggedGUIv2, las tareas faltantes para la funcionalidad mínima del Test Drive son las lecturas de los sensores del auto: la fotorresistencia y la batería. Los datos de estos sensores debían ser incluidos en esta ventana, pero aún no había trabajado en la forma de leer estos. Hoy me dediqué a estudiar los códigos brindados por Santiago, gracias a una recomendación del compañero Kevin Masís, busqué una función en el WiFiClient que se encarga de guardar los comandos (junto con la respuesta que reciben del NodeMCU) en una lista, esto facilita bastante la manipulación de la respuesta del comando *sense;* y llevar esto a cabo es necesario para obtener los datos de los sensores. Para esto, validé con el largo de la respuesta obtenida, que si superaba una cantidad de caracteres (15) sería la respuesta del comando sense; sin embargo, por las modificaciones realizadas al código base en la etapa anterior del proyecto, varias respuestas podían llegar a tener esta dimensión, por lo cual me di la tarea de modificar el archivo NodeMCU.ino y darle mensajes de retorno más cortos al resto de comandos, con el fin de que solamente el comando sense fuera tomado en cuenta para esta tarea. También, en este archivo, reemplacé la lógica de sensor de la batería, ya que esta no era la más precisa. Para esto, consulté la página Arduino Reference sobre maneras de convertir un valor de entrada (en este caso PWM) en otro margen de salida (como el deseado, de 0 a 100). Encontré el comando map, y lo implementé en el código de Arduino para el NodeMCU (este cambio fue documentado por consulta con Santiago, quien recomendó que toda modificación a este debía documentarse).

Una vez que el código en el NodeMCU estaba listo y cumplía con lo que necesitaba para manipular el comando *sense;* dentro del código de Python generé una variable en la que se guardaría la respuesta obtenida por el comando; modifiqué la función get\_log para que retornara la respuesta sin la palabra result, así, esta respuesta se enviaría como argumento para dos hilos: uno que controla lo recibido por el LDR; y el otro del sensor de baterías. Estos hilos funcionan con los índices del string obtenido en la respuesta, mientras que el índice del valor retornado por la fotorresistencia tiene una posición fija ([-4]), los dígitos del sensor de batería podían variar dependiendo de su cantidad (1 si es de 0-9, 2 de 10-99, y 3 si lee 100) por lo cual esta requirió más validación. Estas funciones van en conjunto con otro hilo que se ejecuta desde que la ventana se abre, este lo que hace es enviar el comando sense cada cierta cantidad de segundos, utilizando un delay (time.sleep()) para poder realizarlo.

Los aspectos faltantes del test drive corresponden a la obtención de los datos de los pilotos, del test drive se podrá migrar el sensor de batería para usarlo como el estado actual del carro.

**Sábado, 01/06/2019**

**Administrativa:** Hoy me levanté tarde porque me quedé hasta la mañana trabajando en el Test Drive, por lo cual no realicé tareas del proyecto hasta la tarde.

**Técnica:** La mayoría de los aspectos que me corresponden sobre el código que me corresponden están listos, por lo que hoy me dispuse a trabajar en la documentación. Me faltaba añadir detalles a mi bitácora, ya que en esta la parte técnica estaba poco desarrollada, para esto me ha sido muy útil la posibilidad de añadir descripciones a cada commit realizado al repositorio, ya que en este me basé para ir trazando cada avance y verificar que no dejara ningún detalle de lado. Mi compañero José se está encargando de integrar sus métodos para el manejo de pilotos con la interfaz que había creado hasta el momento, hay algunas tareas pendientes sobre la parte programada, pero iré avanzando la documentación mientras él se encarga de integrar ambas partes.

Luego de un rato, José me estuvo corrigiendo sobre algunas malas prácticas que tenía, me hizo saber que no era necesario tener todas las variables declaradas como globales, si no utilizarlas dentro de cada función como no-locales (*nonlocal variables*). Otro error bastante importante es que yo utilizo mainloop() en cada ventana para que los aspectos de estas ventanas funcionen, mi compañero me informó que la forma correcta de cargar elementos a una ventana es crearlos como propiedad del canvas de esa ventana, lo que quita la necesidad de generar un loop para cada ventana. Me dijo que lo malo de esto es que el código puede generar bugs y se pone en riesgo el funcionamiento del trabajo. Estas fallas están siendo corregidas, se está mejorando el código mientras avanzamos en áreas diferentes.

**Domingo, 02/06/2019**

**Administrativa:** Me fui en la mañana al apartamento de un compañero (Brian W) por un motivo de que trabajo mejor en compañía que estando sólo en mi departamento.

**Técnica:** Aún me encuentro trabajando en la documentación de mi parte del trabajo. Más adelante debo trabajar en algunas partes faltantes de la parte programada, y cuando terminemos el proyecto, trabajaremos en conjunto para terminar el documento administrativo de este.

**Fuentes Consultadas:**

**Parte I:**

Arduino. (2019). *Round Function*. Retrieved Abril 24, 2019, from Foros Arduino: https://forum.arduino.cc/index.php?topic=86779.0

*Arduino Reference*. (2019, Abril 13). Retrieved from https://www.arduino.cc/reference/en/

*Bit-wise operators in C*. (n.d.). Retrieved Abril 19, 2019, from fres2fresh: https://fresh2refresh.com/c-programming/c-operators-expressions/c-bit-wise-operators/

*BitWise operators in C*. (n.d.). Retrieved Abril 19, 2019, from https://fresh2refresh.com/c-programming/c-operators-expressions/c-bit-wise-operators/

Blagšič, M. (n.d.). *ESP8266 WiFi Module Explain and Connection*. Retrieved Abril 14, 2019, from instructables: https://www.instructables.com/id/ESP8266-Wi-fi-module-explain-and-connection/

*C++ error: swtich case jump to case label [f-permissive]*. (2017, Agosto 26). Retrieved Abril 22, 2019, from StackOverflow: https://es.stackoverflow.com/questions/97571/error-en-switch-case-error-jump-to-case-label-fpermisive/97574

*Calculador de resistencias*. (n.d.). Retrieved Abril 12, 2019, from https://www.digikey.com/es/resources/conversion-calculators/conversion-calculator-resistor-color-code-4-band

*Dark Theme for Arduino IDE: 5 easy steps*. (n.d.). Retrieved Abril 23, 2019, from instructables: https://www.instructables.com/id/Dark-Theme-for-Arduino-IDE/

*Datasheet de 74LS164*. (n.d.). Retrieved Abril 13, 2019, from http://www.alldatasheet.com/view.jsp?Searchword=74ls164&gclid=Cj0KCQjw7sDlBRC9ARIsAD-pDFpaSW\_o0V649hznoE1orpOvvOu-yFUiKRVQd8pwtwJTggFOd0MisSMaArW\_EALw\_wcB

*Datasheet L298N*. (2019, Abril 13). Retrieved from http://www.alldatasheet.com/view.jsp?Searchword=L298n%20datasheet&gclid=CjwKCAjwkcblBRB\_EiwAFmfyy2tsqZYDV3\_LhqyfA66ZJTW2ZeLo4c8Q4CO-I0ViAx3ZQtRfmH5XUBoCTZUQAvD\_BwE

*Ejemplos Básicos de Arduino para un taller introductorio*. (n.d.). Retrieved Abril 18, 2019, from GitHub: https://github.com/gsegura96/EjemplosArduino

Hernández, L. d. (n.d.). *Tutorial NodeMCU*. Retrieved Abril 12, 2019, from https://programarfacil.com/podcast/nodemcu-tutorial-paso-a-paso/

Labs, C. (2015, Abril 10). *Los Diodos y el Puente Rectificador*. Retrieved Abril 20, 2019, from Youtube: https://www.youtube.com/watch?v=zgTqmL1G7G8

*LEDs and bit-shifting, a Shift Register tutorial*. (2017, 17 Marzo). Retrieved Abril 20, 2019, from Mikroe: https://www.mikroe.com/blog/leds-bit-shifting-shift-register-tutorial

Marín, G. (n.d.). *Programación del Node con IDE*. Retrieved Abril 13, 2019, from https://programarfacil.com/esp8266/como-programar-nodemcu-ide-arduino/

*NodeMCU blue led*. (2017, Abril 11). Retrieved Abril 13, 2019, from https://www.esp8266.com/viewtopic.php?f=6&t=14474

P, P. (2019, Marzo 8). *Beginner’s Guide to ESP8266 en Git*. Retrieved Abril 14, 2019, from GitHub: https://tttapa.github.io/ESP8266/Chap01%20-%20ESP8266.html

Piensa3D. (2017, Marzo 6). *Cómo utilizar un sensor de luz LDR Arduino*. Retrieved Abril 20, 2019, from Youtube: https://www.youtube.com/watch?v=h9wGZssIBOM

Progamarfacil. (2016, Noviembre 29). *Cómo medir la carga de una pila con Arduino*. Retrieved Abril 20, 2019, from Youtube: https://www.youtube.com/watch?v=mBp5NaivDGk

*Shift Register: Parallel and Serial Shift Register Basic Electronics Tutorials*. (n.d.). Retrieved Abril 18, 2019, from electronics-tutorials: https://www.electronics-tutorials.ws/sequential/seq\_5.html

taifur. (n.d.). *Seven Pro Tips for EsP8266: 9 steps (with Pictures)*. Retrieved Abril 14, 2019, from instructables: https://www.instructables.com/id/ESP8266-Pro-Tips/

*Tinkercad Circuits*. (n.d.). Retrieved Abril 16, 2019, from Tinkercad: https://www.tinkercad.com/circuits

TutosIngeniería. (2015, Noviembre 30). *Medir tensiones mayores a 5V en Arduino*. Retrieved Abril 20, 2019, from Youtube: https://www.youtube.com/watch?v=nW30cV6\_I5o

*Unable to load the code from arduino ide to esp8266*. (n.d.). Retrieved Abril 14, 2019, from GitHub: https://github.com/esp8266/Arduino/issues/2801

**Parte II:**

*Arduino Reference. (2019, Mayo, 31). Retrieved from* [*https://www.arduino.cc/reference/en/*](https://www.arduino.cc/reference/en/)

*Free Phototool, Retrieved May 28th, 2019 from* [*https://www.freephototool.com/*](https://www.freephototool.com/)

*Tkinter Events and Bindings, Retrieved May 26th, 2019 from* [*https://effbot.org/tkinterbook/tkinter-events-and-bindings.htm*](https://effbot.org/tkinterbook/tkinter-events-and-bindings.htm)

*How to pass arguments to a tkinter command, retrieved May 27th , 2019, from* [*https://www.delftstack.com/howto/python-tkinter/how-to-pass-arguments-to-tkinter-button-command/*](https://www.delftstack.com/howto/python-tkinter/how-to-pass-arguments-to-tkinter-button-command/)

*Lambda Expressions, consulted May 27th , 2019 at Socratica,* [*https://www.youtube.com/watch?v=25ovCm9jKfA&t*](https://www.youtube.com/watch?v=25ovCm9jKfA&t)

*Global, local, nonlocal variables, Programiz, retrieved June 1st, 2019 from* <https://www.programiz.com/python-programming/global-local-nonlocal-variables>