Introducción a la programación

Entrega 18/6/2018

II & III Proyecto de programación

Star Force

# Introducción

Juan Pablo Alvarado Villalobos

2018135360

Sebastián Calderón Yock

2018161630

En el año 1984, la compañía Video Ware sacó al mercado el juego Mega Force. El año siguiente, Hudson Soft publicó al mismo para las computadoras del hogar (MSX) y Famicom. La versión norteamericana vendría hasta dos años después, cortesía de la empresa de entretenimiento Tecmo, para la consola NES de Nintendo; esta última con el nombre de Star Force. El segundo proyecto constó de recrear este juego en Python, con la ayuda de bibliotecas como Tkinter y Python. El tercer proyecto trata de construir un controlador para reproducir el juego desarrollado anteriormente, con un arduino y el módulo MPU-9250 con giroscopio, acelerómetro y magnetómetro.

Respecto a la programación, la tarea se orientó al trabajo en equipo, pues se trabajó en parejas (trío para el caso particular de este grupo), con el objetivo de fomentar el trabajo efectivo y significativo en este equipo; además de otras habilidades blandas como liderazgo, delegar tareas, responsabilidad, comunicación, entre otros. Las bibliotecas que se usaron fueron exclusivamente Tkinter y Pygame, la última para la interfaz gráfica del juego. Además de eso, se hizo uso de iteración por primera vez en un proyecto; no obstante, el uso de recursión fue imperante para al menos cuatro módulos en la parte lógica. Finalmente, se utilizaron algoritmos de ordenamiento para el manejo de datos, acorde con los temas vistos en el curso de CE-1101.

La segunda parte, el trabajo en hardware, trata de hacer un control basado en un Arduino Uno, con el cual se pueda utilizar el juego. Este se construye con varios componentes, los principales son un módulo con giroscopio, acelerómetro y magnetómetro, un potenciómetro, dos botones, un display de 7 segmentos y 6 leds. Al programa se le deben hacer los cambios necesarios para poder leer los valores del control y enviarle señales. Esta es la parte que complementa la carrera, pues el punto de ingeniería en computadores es poder trabajar con software y hardware.

Entre los principales inconvenientes observados durante la elaboración del programa, el primero de tales fue cambiar de Tkinter a Pygame, pues sus métodos, funciones integradas y modus operandi diverge notablemente; por ejemplo, en Tkinter al posicionar imágenes en la pantalla, el punto de referencia estaba en el centro de la imagen, en Pygame era de la esquina superior izquierda, provocando así problemas a la hora de validar colisiones con otros objetos. Otra dificultad estuvo en imitar el efecto 3D de Star Force, pues para esto fue necesario agrandar imágenes y desaparecerlas al alcanzar cierto tamaño ya definido. También se puede mencionar que trabajar en conjunto con todos los integrantes fue complicado, ya que en pocas ocasiones la disponibilidad de los integrantes coincidía.

Para el tercer proyecto, resultó un proceso relativamente más lento que el desarrollo de software porque es la primera vez que se trata con hardware, que naturalmente es más tedioso. La inexperiencia jugó en contra del equipo pues se requiere de trabajo fino y delicado, así como eficiente y rápido para avanzar en el tiempo debido. De la misma forma se presentaron varios errores o prácticas no recomendadas.

**Conclusiones:**

* La moduralización juega un papel clave en el desarrollo, ya que se requiere probar funciones por separado y saber cómo juntarlas de forma ordenada. También, si un módulo falla, el problema se contiene a tal módulo y corregirlo resulta más sencillo. Lo anterior es mucho más relevante al trabajar con hardware, porque el manejo de valores de entradas y salidas se facilita si son individuales.
* Se desarrollaron aún más las habilidades relacionadas con la programación a mayor escala; como, por ejemplo: múltiples módulos integrados, pruebas sistemáticas y ordenadas, uso de recursividad e iteración y sus respectivas finalizaciones, pues es muy importante no generar un bucle infinito.
* El apoyo del programa en distintas bibliotecas de Python es vital para la viabilidad del proyecto, dado que el uso de Tkinter para interfaz de ventanas y Pygame para las animaciones dentro del juego resulta notoriamente sencillo.
* Lo que visualmente se toma como animaciones y movimientos tridimensionales, puede ser simulado con distintas imágenes sistemáticas que se asemejen al efecto visual que se da; así se pueden realizar proyectos con mecánicas mucho más complejas y avanzadas.
* Una opción muy viable y práctica común entre los programadores es reciclar código de otros programas ya hechos por ellos, como módulos de ordenamiento de datos o interfaces de proyectos anteriores; asimismo estos se pueden usar como referencia para el nuevo proyecto con el fin de ahorrar tiempo.
* Se tuvo una serie de problemas con el uso de hilos en Pygame, por lo que se optó por usar iteración.
* Para eventos de choques o algo semejante del juego, como se trata de imágenes y no de objetos, lo más sencillo es utilizar funciones y ecuaciones matemáticas para validar la situación y modelar su comportamiento.
* Si se recurre a la recursividad, es importante que no sea para procesos extensos como el control del jugador, porque eventualmente se excede el límite de llamadas recursivas y el juego va a fallar.
* La mejor forma de trabajar en un programa, como el juego en cuestión, es en una aplicación para editar texto como Sublime; esto ahorra tiempo y facilita el proceso con sus funciones internas.
* Usar imágenes PNG mejoran la experiencia del usuario y su apariencia es mucho más agradable que una etiqueta en Tkinter.
* Usar variables globales es una mala práctica porque ocupan mucho espacio en memoria, su uso debe minimizarse.
* Al principio se dieron varios problemas por la legibilidad del código respecto a los integrantes del grupo, así que se documentó mejor y se utilizaron nombres más significativos para funciones y variables.
* La experiencia juega un papel muy importante, por lo que el trabajo en hardware es mejor priorizarlo a algún integrante del equipo con experiencia en soldadura y actividades semejantes.
* La discusión de ideas y la buena comunicación en el equipo son vitales, porque siempre algún miembro tiene una idea más eficiente para alguna función del programa que las de los demás.
* Se debe prestar mucha atención a la construcción de circuitos para evitar crear un fallo, un cortocircuito, quemar los componentes o perder tiempo buscando errores.
* Se tuvo problemas dirigiendo correctamente energía a los leds y el Display 7 segmentos, por lo que se hizo un repaso a la Ley de Ohm, que permitió continuar con el avance.
* Una forma de construir un circuito sobre una placa perforada es por medio de puentes de estaño entre algunos elementos, de esta forma se ahorra el uso de cables y complicarse soldando estos.
* El Arduino Uno posee una resistencia integrada, lo cual evita o disminuye el evento de quemar artilugios conectados a tal si se da un cortocircuito.
* Se debe evitar el uso de cable de red, porque este consta de una serie de hilos de cobre entrelazados y la soldadura de estos se dificulta por ser muchos.
* Se tuvo problemas con los Display 7 segmentos porque estos varían entre cátodos y ánodos, lo cual implica una ligera variación en la conexión a la fuente de poder y a tierra. Esto se resuelve probando ambas conexiones para verificar qué tipo es.
* El manejo de cables hace que el proceso de conexión resulte confuso y tedioso. La mejor manera de proseguir es utilizando módulos de cables, nodos o un cable principal que contenga todas las subconexiones.
* Ciertos componentes secundarios como resistencias, zócalos, entre otros, son muy importantes porque su uso es recurrente y permiten evitar desperdiciar componentes de mayor valor económico.
* La biblioteca Pyserial y el IDE de Arduino presentan problemas entre sí en varias ocasiones, como cuando alguna de las dos está haciendo uso del Arduino y este se llama en la otra; es importante que solo una esté funcionando.
* Soldar con un cautín de menos de 30W es un proceso más lento que con uno de mayor potencia, pero es más seguro y confiable para soldar componentes importantes, porque con uno más potente se corre el riesgo de quemarlo.
* Usar jumpers para trabajar en la protoboard es más sencillo porque su salida es un pin largo y resistente, más que un cable común.
* Cuando se trabaja con un módulo con función de giroscopio, es mejor tratar sus valores por rangos, porque este es muy variable y puede conducir a errores dentro de la ejecución de un programa.

**Recomendaciones**

* Utilizar distintas biblioteca de Python según lo requiera el programa, como Pygame, Tkinter, Pyserial, etc.
* Evitar lo más posible el uso de variables globales dentro del programa para un trabajo más eficiente.
* Si se trabaja en grupo, delegar claramente las funciones de los integrantes, planear lo que se va a trabajar, cómo y para qué de forma exacta, así no se pierde tiempo sincronizando aportes que no se relacionan correctamente.
* Utilizar archivos con formatos que faciliten su uso y mejoren la experiencia dentro del juego, como el PNG para las imágenes. Evitar el uso de archivos de formato WAV porque significan un gran espacio de memoria abarcado respecto a otros.
* Controlar al Arduino en todo momento si se trata de un control, esto es mejor con un hilo o una función por iteración.
* Para los sprites de los jugadores u otro elemento, se recomienda buscar páginas de uso gratuito donde se encuentren modelos para este fin, sin infringir los derechos del autor sobre ellos.
* Si una imagen no está en formato PNG, basta con buscar una página de edición de imágenes online; si el fondo coincide con algún color dentro de la imagen, es recomendable pintar este fonfo de algún color distintivo que sea diferente de cualquiera en la imagen y luego borrar este color con algún programa de edición.
* Utilizar código de proyectos anteriores para los nuevos, de esta forma se ahorra bastante tiempo creando pantallas y demás elementos de la interfaz.
* Usar otros programas como Atom y Sublime porque ahorran mucho tiempo para la escritura de código con sus funciones como la predicción de texto. De la misma forma, respaldar en GitHub o mínimo tener siempre una versión anterior del programa antes de editarlo.
* Recurrir a un multímetro cuando un circuito no funciona correctamente para hallar el problema con facilidad.
* No tocar la soldadura de estaño con las manos porque llena a esta de grasa y a la hora de soldar puede provocar problemas.
* Plantear el circuito que se quiere construir primero en una protoboard para verificar su funcionamiento correcto, de esta forma si no funciona basta con desconectar y reconectar de manera correcta; si está soldado conlleva otras complicaciones.
* No calentar mucho los componentes importantes como el Display o el módulo de Arduino con el cautín, de ser posible que se conecte por medio de un zócalo para poder rescatar el módulo al final del proyecto.
* Revisar las especificaciones de cada componente y verificar el voltaje con el que trabajan para conectarlo a una fuente de poder pertinente.
* Evitar lo más posible el cable UTP porque pelarlo y soldarlo se complica si no se utiliza para una sola conexión, por constar de varios cables entrelazados.
* Asignar cables de colores específicos a la conexión de un elemento determinado para que su análisis sea más sencillo.
* Cerrar al final de cada programa después de que se llame al Serial del Arduino y este se abra.
* Verificar el puerto y la velocidad con la que el Arduino está trabajando, para evitar errores de conexión y asegurarse de que el programa corra debidamente.
* Al producir un cambio en el Display es recomendable darle un tiempo de aproximadamente 2 segundos para que se refresque sin generar un problema.
* Utilizar alicate para sostener los elementos que se van a soldar, calentar estos últimos y finalmente poner una pequeña sección de estaño para que no se den problemas de soldadura excesiva.

**Análisis de resultados**

* Se utilizó la biblioteca Pygame para realizar movimientos y escalamiento de imágenes, pues es más simple y posee atributos que lo facilitan; no así con Tkinter.
* Con el fin de establecer comunicación entre el Arduino Uno y el juego, se recurrió a la biblioteca Pyserial, porque permite recibir y enviar señales de Python al Arduino y viceversa con escasas 4 líneas de código: definir el puerto, abrirlo, leer o escribir y cerrar.
* Para simular el efecto del acercamiento de la nave del jugador a elementos del juego como naves enemigas o aros, se agrandaron las imágenes de estos últimos dos elementos a un ritmo constante. La justificación respectiva es la gran inversión que supone trabajar con modelos 3D y las opciones para tal en Pygame son limitadas.
* El control del movimiento del jugador se programó con las teclas del teclado en lugar del mouse, como en el ejemplo suministrado por el asistente, porque para la segunda parte del proyecto esto se va a manejar con los valores del módulo de Arduino, lo cual es más fácil si se divide en cuatro casos individuales: arriba, abajo, izquierda y derecha (teclas de teclado).
* Luego de reiteradas pruebas, se determinó que para asegurar el funcionamiento correcto del Display 7 segmentos, se requiere de un tiempo de espera de aproximadamente 1.6 segundos, el cual fue determinado experimentalmente.
* El manejo del Arduino, como es constante en todo momento de la ejecución del juego, se hizo por medio de un hilo exclusivo para este, que funciona para desplazarse en las distintas pantallas de la programación y para controlar al jugador.
* Las buenas prácticas como la documentación interna, el uso de software de respaldo como GitHub y el registro de alteraciones al proyecto mejoran cuando se trabaja en grupo, pues esto disminuye el trabajo y permite economizar tiempo.
* El ordenamiento de los puntajes, se requiere guardar tanto puntuación como nombre en forma de tupla, y además ordenar solo con el puntaje. Para esto se desarrolló una función que sacaba el mayor puntaje de las tuplas y se repite el proceso recursivamente para generar una lista ordenada según los puntajes, luego esta lista se reduce a 5 tuplas y se sobrescribe en el archivo .txt, posteriormente se genera una string más presentable en forma de columnas para que la lea la pantalla de puntuaciones.
* Aunque el control de Arduino funciona correctamente utilizando cable UTP, se corre el riesgo de que el alambre dentro de este se quiebre por su delicadeza.
* EL programa funciona correctamente, a menos que se presente el evento en el que algún componente del control se desconecte o se pierde la conexión con el Arduino; en cuyo caso el programa se detiene. En Python esto ocurre porque no se puede establecer conexión con algún elemento que llegue a utilizar, es casi es mismo caso si una variable no se definiera o una función llame a otra que no existe.

**Fuentes consultadas**

* Python-course.eu. (2018). *GUI Programming with Python: Python Tkinter Tutorial*. [online] Available at: https://www.python-course.eu/python\_tkinter.php [Accessed 19 Apr. 2018].
* Programa de lectura y escritura en Arduino suministrado por el asistente Santiago Gamboa.
* Partes del código “Vintage Tanks como referencia”

**Bitácora (Anexo)**

**Parte 1**

**21/5:** Investigamos sobre pygame, sus métodos, elementos, funciones y formas de editar una imagen. Se descargaron imágenes genéricas para probar estas funciones y hacer el plan de acción.

**22/5:** Buscamos varios modelos de naves en internet, se editaron de forma que perdieran su fondo y se guardaran en formato PNG. Luego creamos una función para agrandar imágenes de manera que asemejara el acercamiento de estas a la pantalla.

**23/5:** Se recicló el código del proyecto 1 para la interfaz de ventanas de este proyecto.

**24/5:** Se buscaron imágenes pertinentes para cambiar los botones de la interfaz pasada para que se ajuste a la nueva, se editaron los títulos y nombres de funciones.

**26/5:** Se conectó la función de agrandar imágenes con la interfaz, luego se creó otra función que validara el evento de choque del jugador con el aro. Finalmente esta se corrigió para cuando el jugador pasara a través de estos.

**29/5:** Copiamos la programación de los aros para los enemigos.

**30/5:** Se crean las funciones para generar enemigos, moverlos y sus respectivos eventos de choque.

**31/5:** Creamos una función para que el jugador y los enemigos dispararan basada en los eventos de choque de los aros.

**Parte 2**

**14/6:** Se empiezan a conectar los componentes principales (leds, Display y módulo) al Arduino y la protoboard. Probamos si el circuito está bien.

**15/6:** Compramos componentes, reparamos la conexión de leds y Display y se asiste a varias sesiones de preguntas presenciales con los miembros del laboratorio Lutec.

**16/6:** Se conectan todos los elementos del control a la protoboard, se prueba con las funciones de lectura y escritura suministradas por el asistente, hacemos pruebas con el programa y se comienza a soldar. Se adapta la programación del juego para recibir señales del módulo de Arduino.

**17/6:** Se terminan de soldar los componentes a la placa perforada y se encajan en la carátula del control remoto.