Instituto Tecnológico de Costa Rica.

Escuela de Ingeniería en Computadores.

Taller de Programación

Proyecto Programado 1.

Proyecto Androide de Comportamiento Objetivo

Profesor: Jeff Schmidt Peralta.

Estudiante: Marco Herrera Valverde.

Fecha de entrega: 16 de abril de 2017

Tabla de contenidos

Proyecto Androide de Comportamiento Objetivo (P.A.C.O)

Introducción

P.A.C.O es el primer proyecto programado del curso de Taller de programación y consiste básicamente en la creación de un robot virtual. Este androide responde a una serie de comandos sencillos, y a partir de ellos ejecuta funciones específicas; para así, obtener un resultado definido. El proyecto combina el manejo de imágenes, archivos de sonido y de texto; fue diseñado en Python 3.6.1 con ayuda de Tkinter para la implementación de una interfaz gráfica. En adición, se utilizó la librería externa llamada Python Imaging Library (PIL), para facilitar el uso de imágenes al crear animaciones, y Winsound para la reproducción de archivos de sonido. Cabe destacar que las funciones creadas son en su mayoría implementadas mediante recursividad de cola.

Descripción del problema

Para este proyecto no existe un problema específico y puntual de investigación, más allá del hecho de como crear el robot, lo que genera una serie de problemas menores, por ejemplo: ¿cómo crear una animación?, ¿cómo guardar el estado final del robot?, ¿cómo reproducir un sonido?, entre otros. Por lo tanto se puede considerar el problema general como: “determinar cómo crear un robot virtual que cumpla las especificaciones requeridas”.

Dificultades encontradas

A continuación se mencionan las dificultades encontradas a lo largo del desarrollo del proyecto y la solución utilizada.

- Cargar una imagen png para usar transparencia / Python Imaging Library permite imagenes png
- Crear un entry de más de una línea / Se usa un entry que se mueve sobre un Label.
- Crear la batería sin usar imágenes / Se dibuja el contorno y se crea un rectángulo que cambia de tamaño y color según el nivel de la batería.
- Crear una animación / Se redefine la imagen que se carga en el contenedor y se mueve la posición, lo que produce el efecto animado.

- La animación no funciona / hay que tener seguimiento del PhotoImage entonces al convertir la imagen de PIL a PhotoImage se hace con una referencia al contenedor.
- Llamar un comando a partir de un string / se soluciona con un diccionario.
- Obtener el “n” del comando power(n) / se hace un split con los parentesis y se obtienen todas la partes, para luego validarlas.
- No usar imágenes diferentes para derecha e izquierda / el método transpose de PIL permite cambiar la dirección de la imagen con el argumento FLIP_LEFT_RIGHT.
- Al escribir un comando antes que se termine de ejecutar otro, se interfieren / se deshabilita la entrada de texto mientras se ejecutan los comandos .
- Si se está revisando la batería siempre, no es eficiente / se llama una función que revise la batería, sólo en ciertas ocasiones, como cuando se ejecuta un comando.
- Reproducir música sin parar ni interferir con el programa / winsound tiene los argumentos SND_ASYC y SND_LOOP que permite solucionarlo.
- Leer partes de un archivo de texto / se coloca el cursor antes de lo que se quiere leer, con el método seek() y luego se lee la cantidad deseada de caracteres con read(#).
- Cuando se guarda el estado y la batería es de 2 dígitos o 1 se modifican las posiciones de todos los “seek” y “read” que se usan para cargar el estado / si son dos dígitos, se guarda 0## y si es 1 se guarda 00#, así no afecta las demás posiciones.
- Escribir el texto de la ventana help / se escribe en un documento de texto y luego se lee, ya que es un texto muy extenso, y facilita el trabajo.
- Ejecutar la función color_fondo sin otro botón / se encontró que un entry tiene la opción de *validatecommand* lo cual llama una función, pero debe retornar True o False.
- La posición de las lágrimas en la función cry cambia dependiendo de la dirección del robot / se hacen casos completamente apartes según la dirección.
- Entre muchos otros problemas de menor grado que surgieron al desarrollar el proyecto. Pero tuvieron una solución más rápida y sencilla.

Análisis de resultados

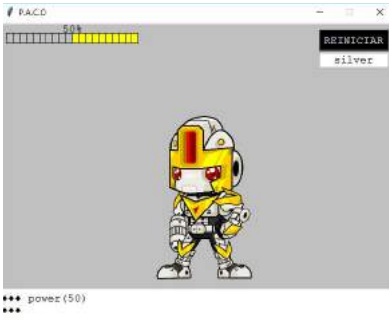
Mediante las condiciones de prueba y las pruebas realizadas, los resultados obtenidos con el proyecto del robot virtual son aceptables y cumplen con las expectativas del diseñador, ya que logra ejecutar todas los comandos permitidos de manera correcta. A continuación se muestran una serie de imágenes que representan los resultados obtenidos al ejecutar algunas funciones.

Goahead

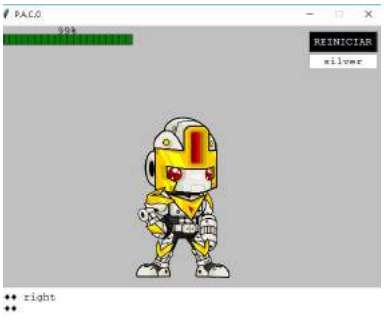
Built

Dance

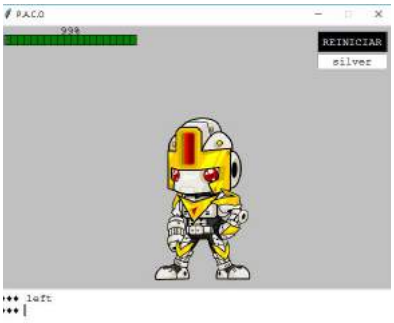
Power(n)



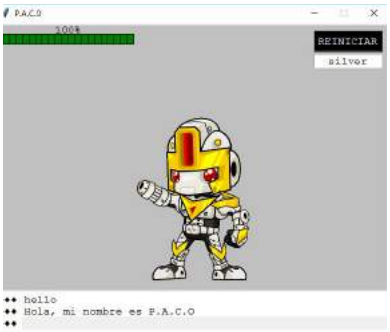
Right



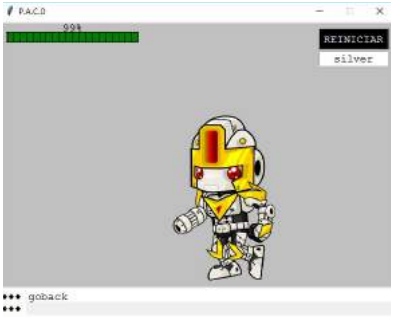
Left



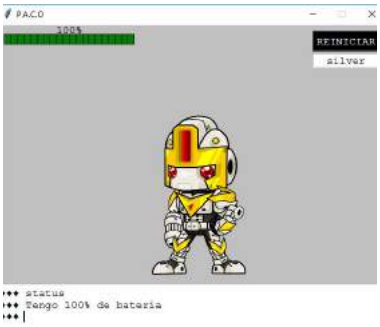
Hello



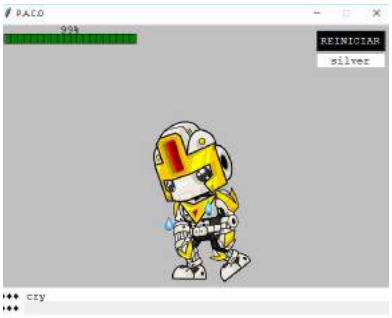
Goback



Status



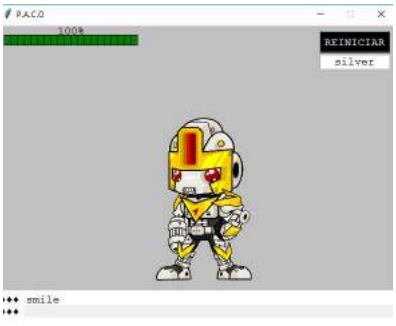
Cry



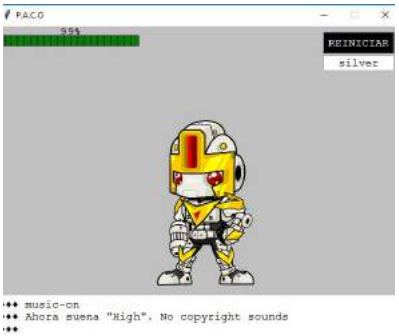
Shoot

Fly

Smile



Music-on



Bitácora de actividades

Esta bitácora contiene las principales actividades realizadas durante el proyecto, pero no todas ya que algunas funciones serán muy pequeñas, sin embargo el tiempo de ellas se encuentra considerado en otras funciones. Ya que una actividad de 10 o 20 minutos extendería mucho la bitácora.

Cuadro 1. Bitácora de actividades para el desarrollo del proyecto.

Actividad	Tipo de actividad	Fecha (2017)	Duración
Análisis del proyecto y los objetivos a alcanzar	Análisis de requerimientos	6 de abril	22:00 - 23:30 1.5 horas
Desarrollo del shell	Investigación y diseño	7 de abril	17:30 - 20:30 Total = 3 horas
Verificar comandos	Investigación	8 de abril	17:00 - 18:00 Total = 1 hora
Crear Animación “goahead”	Investigación y programación	8 de abril	20:00 - 23:30 Total = 3.5 horas
Crear Animación “goback” y “smile”	Programación	8 de abril	23:10 - 00:10 Total = 1 hora
Crear representación de batería	Investigación y programación	9 de abril	12:00 - 13:30 Total = 1.5 horas
Cambiar el estado de la batería	Investigación y programación	9 de abril	15:00 - 17:00 Total = 2 horas
Girar una imagen	Investigación	9 de abril	17:00 - 18:00 Total = 1 hora
Crear animaciones “left” y “right”	Programación	9 de abril	18:00 - 19:00 Total = 1 hora
Crear animación de “dance” y “power(n)”	Investigación y programación	9 de abril	19:30 - 21:00 Total = 1.5 horas
Comprimir imagenes, deshabilitar shell y otros.	Investigación y programación	9 de abril	21:30 - 22:30 Total = 1 hora

Intento fallido de abrir archivos de texto	Investigación y programación	9 de abril	23:00 - 1:00 Total = 2 horas
Leer y escribir archivos de texto	Programación	10 de abril	12:30 - 13:30 Total = 1 hora
Mejora batería	Diseño	10 de abril	13:30 - 14:00 Total = 0.5 horas
Crear animación de muerte	Programación	10 de abril	14:00 - 16:00 Total = 2 horas
Solucionar problemas al iniciar y guardar estado.	Investigación y programación	10 de abril	16:30 - 18:30 Total = 2 horas
Investigar funciones de Winsound	Investigación	11 de abril	11:00 - 12:00 Total = 1 hora
Implementar la funcion “music-on” y “music-off”	Programación	11 de abril	12:30 - 13:00 Total = 0.5 horas
Crear ventana “help”	Programación y diseño	11 de abril	22:30-00:30 Total = 2 horas
Implementar función “status”, “hello” y “built”	Programación	13 de abril	16:00 - 16:30 Total = 0.5 horas
Crear animación de “cry”	Investigación y programación	13 de abril	16:30 - 19:00 Total = 2.5 horas
Crear animación “shoot”	Programación	13 de abril	20:30 - 21:30 Total = 1 hora
Crear “Reiniciar”, “color_fondo” y solucionar algunos errores.	Investigación, programación y diseño	13 de abril	22:30 - 1:30 Total = 3 horas
Crear función “fly”	Programación	14 de abril	16:00 - 18:30 Total = 2.5 horas
Investigar, modificar, crear y comprimir archivos de imagen y sonido	Investigación y diseño	14 de abril	18:30 - 21:30 Total = 3 horas
Ordenar código y hacer la documentación interna	Documentación interna	14 de abril	22:00 - 1:00 Total = 3 horas

Ordenar código y hacer la documentación interna	Documentación interna	15 de abril	11:00 - 12:30 Total = 1.5 horas
Prueba personal del código	Pruebas	15 de abril	14:00 - 14:30 Total = 0.5 horas
Prueba ajena del código	Pruebas	15 de abril	14:30 - 15:00 Total = 0.5 horas
Correcciones al código	Programación	15 de abril	15:00 - 15:30 Total = 0.5 horas
Documentacion externa	Documentación externa	15 de abril	17:00 - 19:30 Total = 2.5 horas

Estadística de tiempos

Actividad	**Tiempo**
Análisis de requerimientos	1.5 horas
Diseño de la aplicación	6 horas
Investigación de funciones	16 horas
Programación	18.5 horas
Documentación interna	4.5 horas
Pruebas	1 hora
Elaboración del documento	2.5 horas
Total	**50 horas**

Conclusión personal

En conclusión, el robot posee las caracterìsticas necesarias para ejecutar una serie de comandos indicados por el usuario y de esa manera interactuar con él. Por otra parte, se puede concluir que un proyecto como este requiere de mucho tiempo tanto de investigación, como diseño y programación, ya que no es un trabajo sencillo de llevar a cabo.

Anexos

En cuanto al desarrollo de este proyecto, se considera que es importante dar mérito y crédito a quien lo merece. Por lo tanto es importante mencionar lo siguiente:

Se utilizó el libro digital “An introduction to Tkinter” del sitio web Effbot.org.
http://effbot.org/tkinterbook/
Para crear audio a partir de texto se utilizó el servicio online de “Text 2 speech”
http://www.text2speech.org/
Para comprimir imágenes se utilizó el servicio online de “Compress Png”
http://compresspng.com/
El “sprite” del robot fue creado por el usuario “pzUH” y descargado de la página OpenGameArt.org https://opengameart.org/content/the-robot-free-sprite
Por último, el título de la canción utilizada es “High” creada por “JPB” para “NoCopyrightSounds” o “NCS” que es una disquera que produce música de distintos artistas, y puede ser utilizada libremente en videos, videojuegos, entre otros.