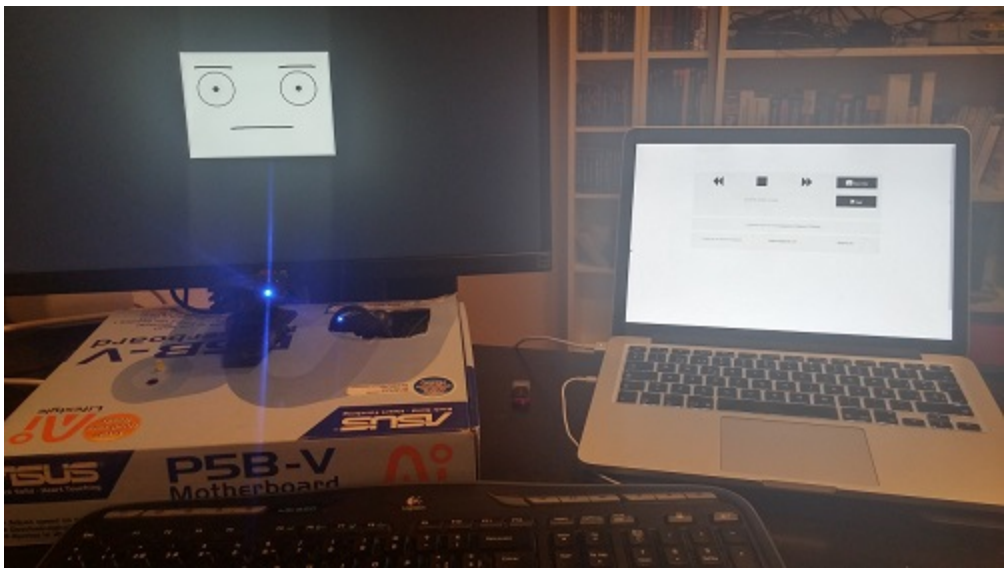
Tarea del tema 7: Proyecto - MiCo: Robot emocional



Diego de los Reyes

Enunciado

Proyecto: creación de una aplicación Se trata de crear un proyecto completo con la Raspberry. Puede ser hardware, software, integración, mezcla de todo...

Objetivos

Al final de la realización de este proyecto se pretende haber alcanzado los siguientes objetivos:

- **Demostrar conocimientos:** Se pretende demostrar conocimientos de todos los temas del curso.
- **Desarrollar un robot emocional:** Que sea capaz de expresar emociones según lo que ocurre a su alrededor.
- **Escalabilidad:** Tanto a nivel hardware como a nivel software, se pretende que el sistema pueda crecer fácilmente.
- **Modularidad:** Se busca desarrollar pequeños módulos, tanto a nivel hardware como software, cada uno con una funcionalidad completa y definida, con la menor dependencia posible entre ellos.
- **Abstracción entre hardware y software:** Para minimizar la dependencia entre hardware y software, esto es, que los programas ejecutados puedan funcionar sin conocer detalles específicos del hardware, como los pines donde están conectados lo componentes o los nombres de dispositivos de la webcam o de la placa Arduino.
- **Divertirse y aprender:** Por supuesto, el objetivo final tanto de este proyecto como de todo el curso, es aprender Raspberry Pi y las posibilidades que ofrece esta plataforma.

Descripción del proyecto

En este proyecto, que tiene como base la práctica 6, vamos a fabricar a MiCo (Mi Compañero) un robot emocional, con las siguientes características:

- Expresión facial de emociones, mediante animaciones en Python, según la personalidad configurable y lo que ocurre alrededor del robot.
- Webcam como cámara de fotos, con movimiento.
- Recepción de órdenes mediante email.
- Envío de fotos mediante gmail.
- Detección de la cantidad de luz.

Se utilizará una **Raspberry Pi** como cerebro del robot, a la que se conectará, además de un teclado y ratón USB, un hub USB con una webcam (el EyeToy de PS2) y una placa Arduino, a la que se conectarán los componentes electrónicos, esto es, el servomotor y el LDR.

Montaje y configuración

El montaje parte de las prácticas anteriores. No obstante, vamos a recordar los pasos efectuados hasta ahora y a completar con los nuevos pasos.

Paso 1: Preparación de la Raspberry Pi

Conectamos un el siguiente material:

- Cargador de móvil para alimentarla.
- Cable HDMI para conectar a un monitor.
- Monitor.
- Cámara web EyeToy de PlayStation 2.
- Teclado y ratón inalámbricos USB.
- HUB USB.
- Tarjeta SD con Raspbian instalado, tal y como se explica en el tema 3.

Paso 2: Conexión y programación de la webcam

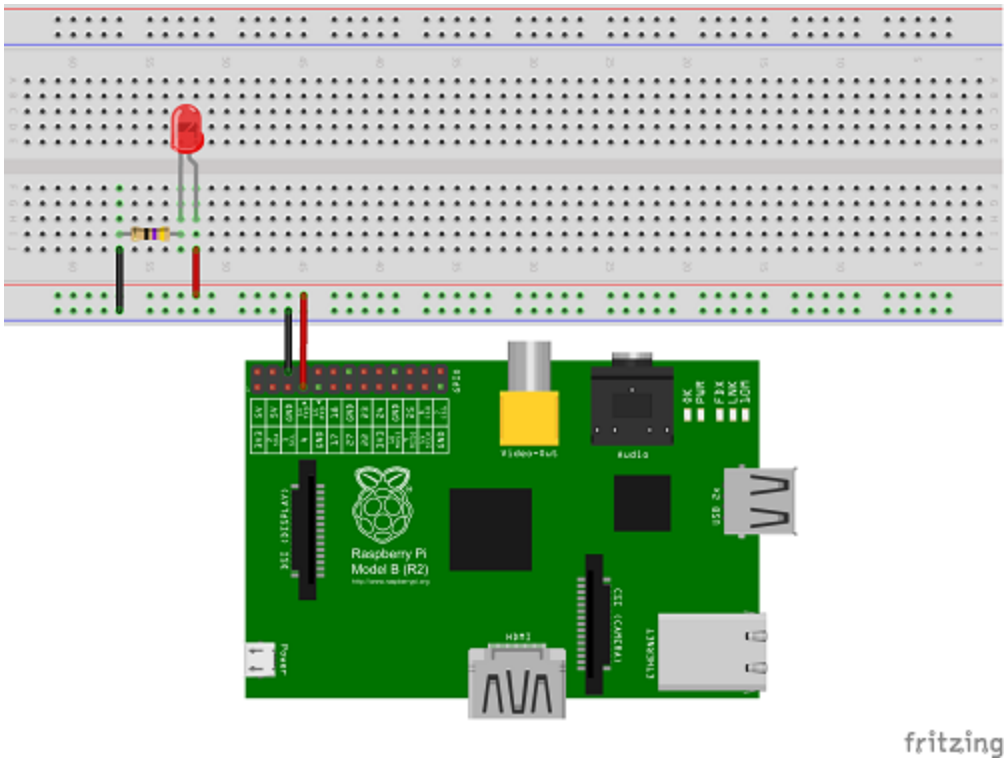
En la práctica 5 escribimos un código con el que sacamos fotos empleando EyeToy de PS2. En esta práctica, aislamos esta funcionalidad en un archivo independiente, **py/webcam.py**.

Paso 3: Programación del envío y la recepción de emails mediante Gmail

En prácticas anteriores utilizamos la API que ofrece Google para manejar cuentas de Gmail para la recepción de órdenes mediante email. En esta práctica, hemos añadido el envío de emails con archivos adjuntos y hemos aislado toda comunicación con Gmail en el archivo **py/gmail.py**.

Paso 4: Conexión del led al puerto GPIO y programación

Como vimos en la práctica 6, conectamos al puerto 7 GPIO el led con una resistencia de 47 ohmios, según el siguiente esquema (puede verse a mayor resolución en la carpeta **doc/fritzing**):



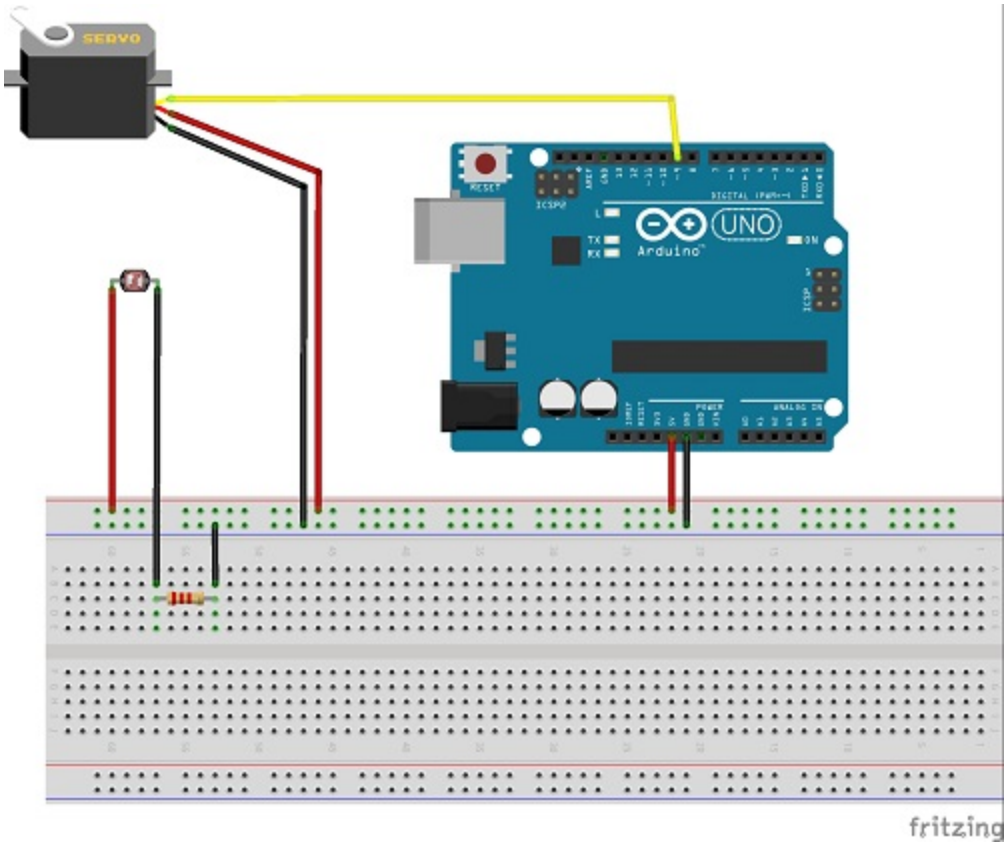
La funcionalidad del manejo de este led y de cualquier otro componente que se fuera a conectar a los puertos GPIO de la Raspberry Pi la vamos a aislar en el archivo **raspberry.py**.

Paso 5: Preparación de Arduino

En primer lugar, vamos a preparar el material necesario para Arduino:

- Placa Arduino, con cable USB y fuente de alimentación.
- Servomotor de 180º
- LDR
- Placa protoboard
- Cables

A continuación, realizamos el montaje, según el siguiente esquema (puede verse a mayor resolución en la carpeta **doc/fritzing**):



Y programamos el firmware que reciba por serial las órdenes necesarias para mover la webcam. El código se encuentra en **arduino/mico/mico.ino**.

Paso 7: Comunicación con Arduino

En la práctica anterior, programamos la comunicación con Arduino. En esta práctica, aislamos esta comunicación en el archivo **py/arduino.py**, al que añadimos la nueva funcionalidad de leer sensores, que son, la cantidad de luz que mide el sensor LDR y la posición del servo que mueve la webcam.

Paso 8: Desarrollo de la web que simplifica el envío de comandos por mail

Partimos de la web desarrollada para la práctica 6. A esta web le hemos dado un lavado de cara, haciendo una interfaz de usuario más amigable. Las funcionalidades quedan intactas. El código fuente se puede encontrar en la carpeta **web**.

Paso 9: Programación de expresiones mediante PyGame

El archivo **py/expresiones.py**, desarrollado desde cero para esta práctica, se apoya en la librería pygame para dibujar un rostro capaz de expresarse para nuestro robot emocional. Esto permite al robot expresar emociones, variando la expresión de su cara, y mover las pupilas. Se puede ver una prueba de su funcionamiento en el siguiente vídeo:

https://www.youtube.com/watch?v=kOC2v3MShDI

Paso 10: Creación del archivo de configuración

Este archivo, ubicado en **py/config.cfg**, sirve opciones de configuración al resto de módulos, para aislar configuraciones específicas del usuario que vaya a manejar a MiCo y de la Raspberry Pi donde lo instale, como el usuario y contraseña de gmail de su MiCo, su usuario de gmail donde recibirá las fotos que envíe, la configuración de los dispositivos externos (puerto donde está conectado Arduino y puerto GPIO del led conectado a la Rasbperry Pi). Además, contiene otras entradas para configurar la personalidad del robot, en función de las cuales tardará más o menos en enfadarse cuando le mandes muchas órdenes o tendrá más o menos fobia a la luz, por ejemplo.

Paso 11: Montaje final en la carcasa

Quitamos la protoboard y metemos todos los componentes dentro de una caja para tenerlo más organizado. El resultado puede verse en estas fotos:

Interior:

Exterior:

Paso 12: Orquestación de todos los componentes

El archivo **py/main.py**, que contiene el bucle principal de ejecución, cuyo único objetivo es iniciar a MiCo.

El archivo **py/mico.py** y sus funciones wake, live y sleep, que despertarán y dormirán al robot según corresponda, e inicializarán el resto de programas. Éstas son las únicas tres funciones que se utilizan desde el exterior.

El programa que ejecuta mico.py se encarga de utilizar todos los módulos anteriores. Además, tiene programados una serie de comportamientos, cuyo resultado muestra una u otra emoción. Los comportamientos programados son:

- Pasión por la fotografía:

  https://www.youtube.com/watch?v=ZVJFOEATSpI

- Recepción de órdenes

  https://www.youtube.com/watch?v=kdUxwpPqvGk

- Enfado si recibe muchas órdenes, hasta poder enfermar, y aburrimiento si no recibe ninguna

  https://www.youtube.com/watch?v=i2V8g_ISST8

- Fobia a la luz y aturdimiento si hay mucha

  https://www.youtube.com/watch?v=0Z3jndfd9MY

- Vergüenza si comete un error

  https://www.youtube.com/watch?v=dCtxA4LEmPE

También tiene una función que se encarga de borrar la memoria a corto plazo del robot, para, por ejemplo, olvidarse de que se siente enfadado o avergonzado por algo que ha hecho mal.

En la siguiente playlist de Youtube pueden verse todos los vídeos:

https://www.youtube.com/playlist?list=PLSWDLiNLOIcJZHwqFf2CGX3aUfiJCe-ti

Manual de uso

MiCo, Mi Compañero, es un robot asistente con emociones y personalidad propias. Este robot te permitirá ver de manera remota lo que sucede en la habitación donde se encuentre gracias a su cámara, que podrás mover para mirar hacia los lados.

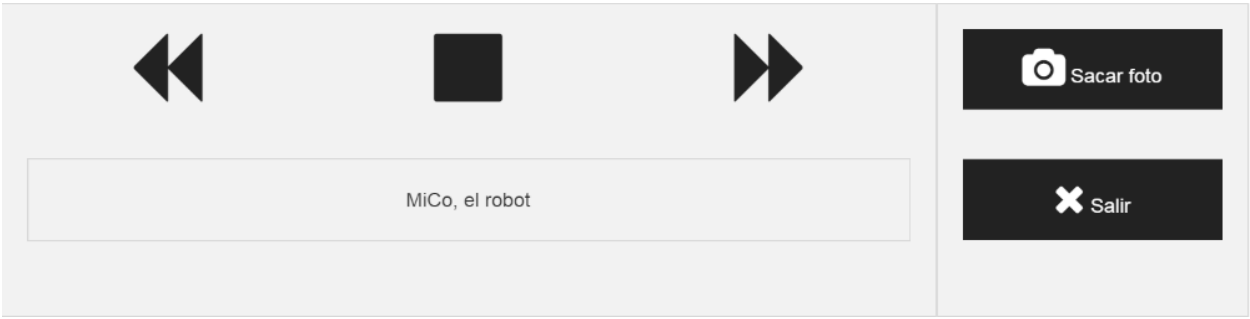
Para ponerlo en funcionamiento sigue los siguientes pasos:

1. **Conexión:** Enchufa MiCo a la corriente eléctrica, conecta su cable HDMI a una pantalla y su toma ethernet a una clavija RJ45. Enciende el interruptor y verás cómo arranca el sistema operativo.
2. **Configuración:** A continuación, crea una cuenta Gmail para tu MiCo. Una vez creada, busca el archivo mico/config.cfg y configura su cuenta recién creada, añadiendo la tuya como mailto para recibir en ella los correos que te envíe.
3. **Inicio:** Para iniciar mico, abre un terminal, ve al directorio donde se encuentre el código y escribe:

   sudo python main.py

   MiCo se ejecutará entonces a pantalla completa.

4. **Salir:** Puedes salir del programa pulsando la tecla “q” o enviando la orden “Salir” desde la interfaz.
5. **Uso:** Puedes manejar a tu MiCo a través de su web. Configura el archivo **mico.php** y sube el código web a un servidor. Accede a su url y dispondrás de una interfaz para manejarlo, desde donde podrás mover la webcam, sacar una foto (que recibirás en el correo configurado) y salir, deteniendo el funcionamiento de tu MiCo.



Un ejemplo de web puede verse aquí:

http://innovaia.es/mico/

Conclusiones

Si revisamos los objetivos, podemos comprobar que:

- **Demostrar conocimientos:** Se ha realizado un proyecto completo, integrando software y hardware y utilizando técnicas nuevas de programación en Python, como separar el código en diferentes ficheros y un archivo de configuración. Se ha integrado el módulo pygame para realizar la expresión de la cara, y se ha añadido un nuevo sensor, LDR a Arduino, cuyo valor se puede leer desde la Raspberry Pi. Aunque siempre se puede hacer mucho más, se consideran cubiertos los objetivos de la práctica y del curso con este proyecto.
- **Desarrollar un robot emocional:** En los vídeos vistos a lo largo de esta memoria podemos comprobar cómo el robot es capaz de expresar diferentes emociones según diferentes estímulos, de modo que hemos conseguido desarrollar un robot emocional básico, pero funcional.
- **Escalabilidad:** Partiendo de la práctica anterior, hemos logrado aumentar su funcionalidad. La refactorización de código ayuda a que cada módulo pueda añadir nuevas funciones con facilidad y que la adición de nuevos módulos sea sencilla.
- **Modularidad:** A nivel de software, Python nos ha permitido de una manera fácil crear módulos independientes que realizan funciones concretas, como el módulo que maneja Gmail, el que maneja Arduino o el que maneja las expresiones. Cada componente conoce sólo aquello que necesita conocer y desconoce al resto de componentes. Por ejemplo, el módulo webcam no sabe de la existencia del módulo gmail, y sin embargo, el programa principal los sabe utilizar en conjunto, llamando sólo a las funciones principales que ofrece, sin tener que conocer los detalles de su implementación. A nivel hardware, añadir una nueva placa Arduino sería tan sencillo como programar su firmware, los drivers en Python y pedir al programa principal (mico.py) que lo use.
- **Abstracción entre hardware y software:** Ningún módulo que maneje hardware conoce los dispositivos, pines o puertos que necesite para funcionar. Sólo sabe dónde buscarlos. Y esto es gracias al uso del archivo de configuración, el único que sabe exactamente la configuración interna del hardware (además de guardar otras variables configurables, como la cuenta de gmail). Gracias a esto, el software es completamente independiente del hardware. Por otro lado, actualmente sólo existe un archivo de configuración, porque el sistema es pequeño, pero no sería difícil añadir más archivos de configuración (por ejemplo, uno por módulo o conjunto de módulos de una misma familia) si el sistema empezase a crecer.
- **Divertirse y aprender:** Principalmente, me quedo con que en este curso he aprendido un lenguaje totalmente desconocido, Python, que es sumamente sencillo y funciona muy bien en la Rasbperry Pi, y con la facilidad de comunicación entre Raspberry Pi y Arduino, cuya sinergia consigue unos resultados increíbles de una forma más o menos sencilla. Arduino maneja muy bien la electrónica y Raspberry tiene un gran cerebro para ejecutar software y añadir muchos componentes, como el mismo Arduino o una Webcam. Las horas invertidas en realizar este proyecto han sido más que gratificantes, se ven resultados fácilmente y, aunque también ha habido momentos de bloqueo, el resultado final es más que satisfactorio, así que, por supuesto, he conseguido divertirme y aprender..

Además de todo lo anterior, quería comentar que en la práctica 6 escribí en las conclusiones que el led conectado a la Raspberry Pi sería más correcto conectarlo a Arduino para que éste lo manejara. Sin embargo, esto no lo veo ahora tan correcto. Sí que es cierto que la potencia electrónica debería ir en Arduino y la potencia de control software en Raspberry Pi, pero este último también tiene potencia para mover cosas pequeñas, y este led debe encenderse ante la llegada de órdenes, ya vengan de Arduino, de Gmail o de otra fuente diferente, con lo que debe funcionar con independencia de la existencia de Arduino, razón por la cual no lo he movido de sitio, porque pienso que, en este caso, estaba equivocado y sí que está bien ubicado aquí.

Por otro lado, la interfaz gráfica está desarrollada en Python con la librería Pygame. Las dimensiones de las caras son las mismas que para Scratch, 480x360 px. Esto es porque inicialmente consideré utilizar Scratch para realizar dicha interfaz, ya que he trabajado con esta herramienta y me parece muy sencilla de utilizar. Probé la comunicación por sockets entre Python y Scratch en la Raspberry, pero no conseguía sincronizar los mensajes, así que empecé a buscar alternativas, y encontré Pygame, que parecía bastante sencillo, así que me decidí a usar una librería nueva que no hubiera probado, y, la verdad, estoy contento con el resultado. Pienso que para este proyecto es más acertado usar Pygame que Scratch. No obstante, seguiré investigando la comunicación Scratch - Python, ya que puede ser interesante para otros proyectos.

El motor de emociones es bastante simple; el robot tiene programados una serie de comportamientos que sienten lo que sucede en el entorno, y, en base a esto, el robot expresa emociones. Este sistema es muy básico porque no existe aprendizaje, no existe personalidad (sí que hay una serie de parámetros de personalidad, pero muy rudimentarios), pero sí que muestra las posibilidades de hacia dónde puede crecer este proyecto, metiendo técnicas de Inteligencia Artificial y aprendizaje.

Sobre el manual de usuario, en un producto cerrado, Raspbian debería arrancar directamente MiCo, mostrando al usuario un menú de configuración inicial básico. Además, la web desde donde controlar a tu robot, debería ser una única web, pública, donde los usuarios pudieran registrarse, asociar a su robot y manejarlo.

Por último, este proyecto puede usarse para varias finalidades. El límite es la imaginación:

- **Enseñar emociones básicas:** Tanto a niños como a personas con problemas para reconocerlas.
- **Vigilancia:** Instalando el sistema y pidiéndole que te envíe fotos de la sala donde se encuentre.
- **Niñera:** Partiendo de la vigilancia, se puede controlar a un niño y transmitirle expresiones y, con ampliaciones, incluso transmitir otros mensajes personalizados, o desarrollar habilidades que les cuenten cuentos, por ejemplo.
- **Telepresencia:** Gracias a la cámara, si se amplía el módulo que la maneja para que también grabe vídeo, se podría realizar un sistema de telepresencia con un avatar que exprese las emociones de su interlocutor.