

PONTIFICIA UNIVERSIDAD CATÓLICA MADRE Y MAESTRA
FACULTAD DE CIENCIAS DE LA INGENIERÍA
ESCUELA DE INGENIERÍA EN COMPUTACIÓN Y TELECOMUNICACIONES



Presentado Por:

Luis Ignacio Mateo Gómez 1012-7691

Gesbien Rafael Núñez Vargas 1013-7963

Christopher Paul - 1012-8901

Asignatura:

Programación Aplicada

ST-ISC-105-T

Profesor:

Ing. Máximo Pérez

ÍNDICE DE CONTENIDO

INTRODUCCIÓN	3
OBJETIVO GENERAL	4
OBJETIVOS ESPECÍFICOS	4
MARCO TEÓRICO	5
DESARROLLO	8
IMÁGENES	12
LIMITANTES DEL PROYECTO	13
CONCLUSIONES GENERALES	13
REFLEXIÓN Y CONCLUSIÓN DE CADA MIEMBRO	14

INTRODUCCIÓN

En el proyecto anterior de la materia de Programación Aplicada, bajo la tutela del profesor Máximo, logramos un notable avance al crear un ojo biómico funcional.

Nuestra experiencia en esta asignatura fue el resultado del trabajo colaborativo de un equipo compuesto por tres estudiantes de ingeniería de sistemas en su último año de carrera. Tuvimos la oportunidad de explorar una amplia gama de componentes eléctricos, controlados por una Raspberry Pi. Para facilitar nuestra colaboración, utilizamos la tecnología SSH para trabajar de forma remota en un router local proporcionado por el profesor.

Inicialmente, cada miembro del equipo trabajó de manera individual en la codificación de los programas necesarios para nuestros diferentes proyectos. Posteriormente, unimos fuerzas para desarrollar soluciones en Python, aprovechando diversas bibliotecas como Gpiozero, RaspiCam, entre otros. Durante este proceso, nos familiarizamos con la integración y programación de componentes como servomotores, sensores ultrasónicos, sensores de temperatura, cámaras, entre otros. Ahora, nos enfrentamos a un nuevo desafío emocionante: crear un abanico inteligente. Este dispositivo, que emplea motores DC (de corriente) de 9V y 12V, contará con una serie de características avanzadas.

En primer lugar, el abanico ofrecerá un modo manual, que permitirá al usuario elegir entre un funcionamiento fijo, un movimiento rotatorio constante y la capacidad de ajustar la velocidad de rotación a tres niveles diferentes por minuto. Además, contará con un modo inteligente, que activará el abanico al detectar movimientos mediante nuestro sensor ultrasónico, ofreciendo así una respuesta automática a la presencia de personas.

También integraremos un sistema para ajustar la velocidad del abanico en función de la temperatura ambiente, utilizando datos proporcionados por nuestro sensor de temperatura. Finalmente, implementaremos un modo de apagado inteligente, que desconectará el abanico si no se detecta presencia durante un período de tiempo determinado por el usuario.

Esta nueva empresa nos desafía a combinar nuestra experiencia previa con nuevos conocimientos y habilidades, promoviendo así un aprendizaje continuo y el desarrollo de soluciones innovadoras en el campo de la ingeniería y la programación aplicada.

OBJETIVO GENERAL

Diseñar e implementar un abanico inteligente utilizando componentes y programación aplicada, con la capacidad de detectar la presencia de personas, ajustar la velocidad de rotación en función de la temperatura ambiente y ofrecer modos de funcionamiento manual e inteligente.

OBJETIVOS ESPECÍFICOS

- Integrar un sensor ultrasónico para la detección de objetos cercanos y la activación automática del abanico.
- Implementar un sistema de control para ajustar la velocidad del abanico en respuesta a la temperatura ambiente, utilizando datos proporcionados por un sensor de temperatura.
- Desarrollar un modo manual que permita al usuario elegir entre un funcionamiento fijo y un movimiento rotatorio constante, con la capacidad de ajustar la velocidad de rotación.
- Diseñar un modo inteligente que active el abanico al detectar movimientos mediante el sensor ultrasónico y ajuste automáticamente la velocidad de rotación.
- Integrar un sistema de apagado inteligente que desconecte el abanico si no se detecta presencia durante un período de tiempo establecido por el usuario.

MARCO TEÓRICO

Raspberry Pi

El Raspberry Pi se caracteriza por su tamaño compacto, que es similar al de una tarjeta de crédito, y su bajo costo en comparación con otras computadoras. A pesar de su tamaño pequeño, el Raspberry Pi ofrece un rendimiento sorprendente gracias a su procesador ARM y a su capacidad para ejecutar sistemas operativos completos, como Raspbian (una distribución de Linux optimizada para el Raspberry Pi), Ubuntu y otros.

Una de las características más destacadas del Raspberry Pi es su versatilidad. La placa cuenta con una variedad de puertos de entrada y salida (E/S), incluyendo puertos USB, HDMI, Ethernet, y GPIO (General Purpose Input/Output), que permiten la conexión de una amplia gama de dispositivos y periféricos. Esto hace que el Raspberry Pi sea ideal para una variedad de aplicaciones, desde proyectos de bricolaje y electrónica hasta servidores domésticos, estaciones de trabajo de escritorio y sistemas embebidos.

Además de su hardware versátil, el Raspberry Pi cuenta con una comunidad activa y dedicada de usuarios y desarrolladores que contribuyen con software, tutoriales y proyectos open-source. Esta comunidad vibrante ha permitido que el Raspberry Pi se convierta en una plataforma de aprendizaje y experimentación accesible para personas de todas las edades y niveles de habilidad en todo el mundo.

FUNDACIÓN DE RASPBERRY PI

La Fundación Raspberry Pi trabaja para poner el poder de la informática y la creación digital en manos de personas de todo el mundo. Para ello, proporciona computadoras de bajo costo y alto rendimiento que la gente usa para aprender, resolver problemas y divertirse. Proporciona extensión y educación para ayudar a más personas a acceder a la informática y la creación digital; desarrolla recursos gratuitos para ayudar a las personas a aprender sobre informática y a crear cosas con computadoras y también capacita a educadores que pueden guiar a otras personas a aprender.

Cómo funcionan los motores de CC

Un motor de DC consta de las siguientes partes básicas:

- Caja protectora metálica
- Imanes permanentes

- Eje
- Comutador
- Devanados de bobina
- Terminales.

Los motores de corriente continua funcionan a partir de la interacción entre la electricidad y el magnetismo. Una corriente eléctrica que fluye a través de una bobina de alambre (el comutador) crea un campo magnético. Cuando este campo magnético se coloca en un campo magnético externo creado por los imanes permanentes, la bobina experimenta una fuerza que hace girar el motor. Esta fuerza es proporcional tanto a la corriente que fluye en las bobinas como a la fuerza del campo magnético externo.

La necesidad de un controlador de motor para motores DC

Los pines GPIO de Raspberry Pi sólo pueden proporcionar una corriente de 16 mA por pin, lo que no es suficiente para alimentar un motor de CC. Para proporcionar la corriente requerida por el motor, necesitamos utilizar un controlador de motor.

Un motor DC necesita un controlador de motor para ser controlado adecuadamente por un Raspberry Pi debido a varias razones fundamentales:

1. **Corriente y Voltaje:** Los motores DC suelen requerir corrientes y voltajes que están más allá de lo que puede proporcionar directamente un pin GPIO (General Purpose Input/Output) del Raspberry Pi. Los pines GPIO están diseñados para manejar señales de bajo voltaje y corriente, no para alimentar dispositivos de alto consumo como los motores.
2. **Protección del Raspberry Pi:** Conectar directamente un motor DC al Raspberry Pi podría dañar la placa. Los motores DC, especialmente al arrancar o detenerse, pueden inducir picos de voltaje y corriente que pueden afectar negativamente a la placa y sus componentes.
3. **Control de Velocidad y Dirección:** Un controlador de motor proporciona la capacidad de controlar la velocidad y la dirección del motor. Esto se logra mediante la modulación del ancho de pulso (PWM) para controlar la velocidad, así como la inversión de la polaridad para controlar la dirección.
4. **Aislamiento Eléctrico:** Los motores pueden generar interferencia eléctrica (ruido) que podría afectar otros componentes del circuito, especialmente dispositivos

electrónicos sensibles como un Raspberry Pi. Usar un controlador de motor puede dar algún aislamiento eléctrico entre el motor y el circuito de control.

5. **Eficiencia Energética:** Al utilizar un controlador de motor se pueden implementar técnicas de control más eficientes para maximizar el rendimiento energético del motor, lo que puede ser especialmente importante en aplicaciones donde la eficiencia energética es una consideración importante.

Raspi_MotorHAT

La biblioteca Raspi_MotorHAT es una herramienta útil para controlar motores utilizando el módulo Motor HAT (Hardware Attached on Top) (HAT significa "Hardware Attached on Top") en un Raspberry Pi. Este módulo es una placa de expansión diseñada para el Raspberry Pi que facilita el control de hasta cuatro motores DC o dos motores paso a paso a la vez. Algunas características claves de esta herramienta son:

- **Integración con Raspberry Pi:** La biblioteca Raspi_MotorHAT está diseñada para funcionar con el Raspberry Pi, aprovechando su potencia de procesamiento y capacidad de E/S para controlar motores de manera eficiente.
- **Control de motores DC:** La biblioteca permite controlar motores de corriente continua (DC) conectados al módulo Motor HAT. Esto incluye funciones para controlar la velocidad y la dirección de los motores, así como para detenerlos y arrancarlos según sea necesario.
- **Control de motores paso a paso:** Además de los motores DC, el módulo Motor HAT también puede controlar motores paso a paso. La biblioteca Raspi_MotorHAT proporciona funciones para controlar la velocidad, dirección y número de pasos de estos motores, lo que permite una amplia gama de aplicaciones, como robótica y posicionamiento preciso.
- **Interfaz sencilla:** Raspi_MotorHAT ofrece una interfaz sencilla y fácil de usar para controlar los motores conectados al módulo Motor HAT. Esto incluye métodos para inicializar el módulo, configurar los motores y controlar su funcionamiento con comandos simples en Python.
- **Soporte de la comunidad:** Al ser parte del ecosistema del Raspberry Pi, la biblioteca Raspi_MotorHAT cuenta con el respaldo de una comunidad activa de usuarios y desarrolladores. Esto significa que es fácil encontrar ayuda, recursos adicionales y ejemplos de proyectos que utilicen esta biblioteca en línea.

DESARROLLO

Definición de variables globales y de pines:

- Se definen las variables tiempo_ejecucion y tiempo_actual que controlarán el tiempo de funcionamiento.
- Se define el sensor ultrasónico ultrasonic_sensor, indicando los pines de echo y trigger y la distancia máxima y mínima que puede medir, en este caso 2 metros.
- Se inicia el controlador de motores Raspi_MotorHAT enviando su dirección.
- Se obtiene el motor DC conectado al puerto 2 del controlador y se establece su velocidad base a 150 rpm.
- Se obtiene el motor paso a paso configurado para 200 pasos por vuelta y conectado al puerto 1 del controlador.

```
from gpiozero import DistanceSensor
from Raspi_MotorHAT import Raspi_MotorHAT, Raspi_DCMotor
import time

#####
# Variable #####
tiempo_ejecucion = 3
tiempo_actual = 0
#####

#####
# Ultrasonic #####
echo_pin = 17
trigger_pin = 4
ultrasonic_sensor = DistanceSensor(
    echo=echo_pin, trigger=trigger_pin, threshold_distance=0.2, max_distance=2)
#####

#####
# Sensor Temp #####
# dht_device = adafruit_dht.DHT11(board.D10)
#####

#####
# Motor DC #####
motor_hat = Raspi_MotorHAT(addr=0x6f)
dc_motor = motor_hat.getMotor(2)
dc_motor.setSpeed(150) # Velocidad base
#####

#####
# Motor Stepper #####
# 200 pasos por vuelta, stepper motor en el puerto 1
stepper_motor = motor_hat.getStepper(200, 2)
#####
```

En resumen, se definen los dispositivos que se van a usar y las métricas necesarias para el buen funcionamiento:

- Sensor ultrasónico para detección de objetos
- Motor DC para el abanico
- Motor paso a paso para movimiento oscilante del abanico

Definición de Funciones:

Las funciones guardar_posicion_stepper y leer_posicion_stepper fueron creadas para guardar en un archivo texto llamado “posicion_stepper.txt” la posición en la cual el abanico inteligente se paró después de no detectar ningún movimiento a través del sensor ultrasónico. Por ejemplo, si se paró en una rotación en las manecillas del reloj, al volver a prenderse, el abanico inteligente rotara en contra las manecillas del reloj.

En caso de que el archivo no exista, lo que puede ocurrir en la primera corrida del código, devolverá 0.

```
1 usage
def guardar_posicion_stepper(posicion):
    with open('posicion_stepper.txt', 'w') as file:
        file.write(str(posicion))

1 usage
def leer_posicion_stepper():
    try:
        with open('posicion_stepper.txt', 'r') as file:
            posicion = int(file.read())
        return posicion
    except FileNotFoundError:
        return 0 # Si el archivo no existe, devuelve la posición inicial (0)
```

La siguiente función llamada control_abanico, es la función principal de nuestro abanico inteligente, el cual se ejecutará en un bucle infinito para el control de esta última. Esta

función, empieza leyendo la distancia del sensor ultrasónico. En caso de que esta distancia este inferior a 50 cm, es decir hay un objeto cercano, el algoritmo cumplirá las siguientes etapas:

- Enciende el motor DC del abanico.
- Lee la posición inicial del motor stepper.
- Hace girar el stepper de manera oscilante durante el tiempo elegido por el usuario en segundos. En este caso durante 15 segundos.
- Devuelve el stepper a la posición inicial.
- Deja encendido el motor DC 2 segundos más para poder identificar si hay un objeto o una persona frente al sensor ultrasónico.

```
Usage:
def control_abanico():
    try:
        while True:
            distancia = ultrasonic_sensor.distance
            tiempo_actual = 0

            if distancia < 0.5:
                tiempo_inicio = time.time()
                print("Objeto detectado a {} metros.".format(distancia))
                # Encender el motor DC (abanico)
                dc_motor.run(Raspi_MotorHAT.FORWARD)
                posicion_inicial = leer_posicion_steerer()

                while tiempo_actual < tiempo_ejecucion:
                    # Girar el stepper motor durante 15 segundos
                    # Mueve 100 pasos en sentido horario
                    stepper_motor.step(
                        100, Raspi_MotorHAT.FORWARD, Raspi_MotorHAT.DOUBLE)
                    time.sleep(1)
                    stepper_motor.step(
                        100, Raspi_MotorHAT.BACKWARD, Raspi_MotorHAT.DOUBLE)
                    tiempo_actual += 1

                    stepper_motor.step(
                        posicion_inicial, Raspi_MotorHAT.BACKWARD, Raspi_MotorHAT.DOUBLE)
                    guardar_posicion_steerer(posicion_inicial)
                    # Seguir ejecutando el motor DC durante 2 segundos más
                    time.sleep(2)
                    dc_motor.run(Raspi_MotorHAT.RELEASE)

            else:
                print("Ningún objeto detectado.")
                time.sleep(1)

    except KeyboardInterrupt:
        print("Interrupción de teclado. Deteniendo el abanico.")
        dc_motor.run(Raspi_MotorHAT.RELEASE)
```

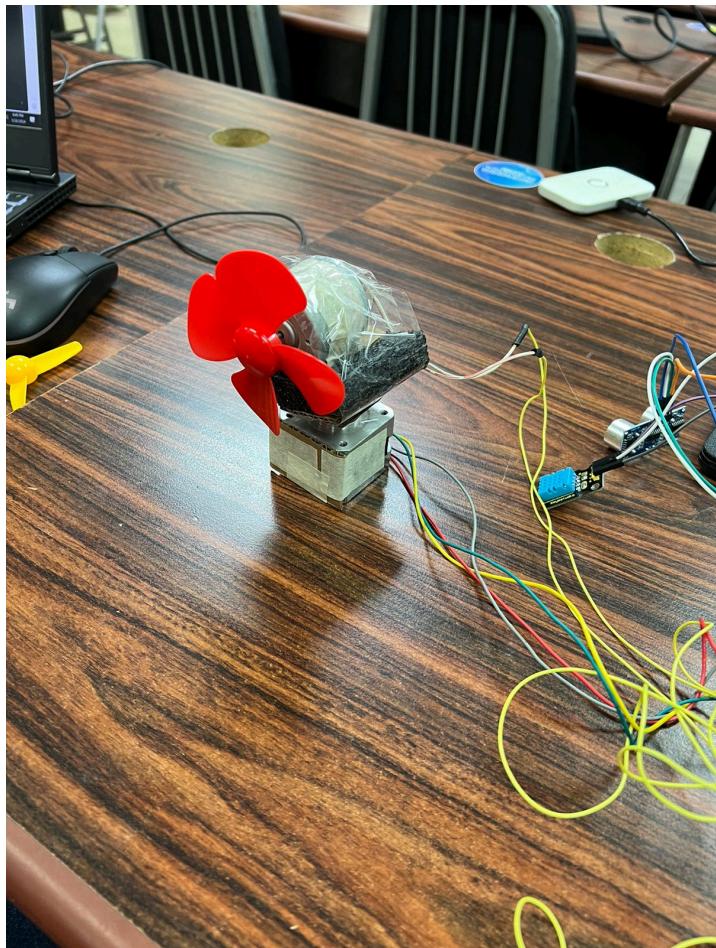
LIBRERÍAS PYTHON USADAS

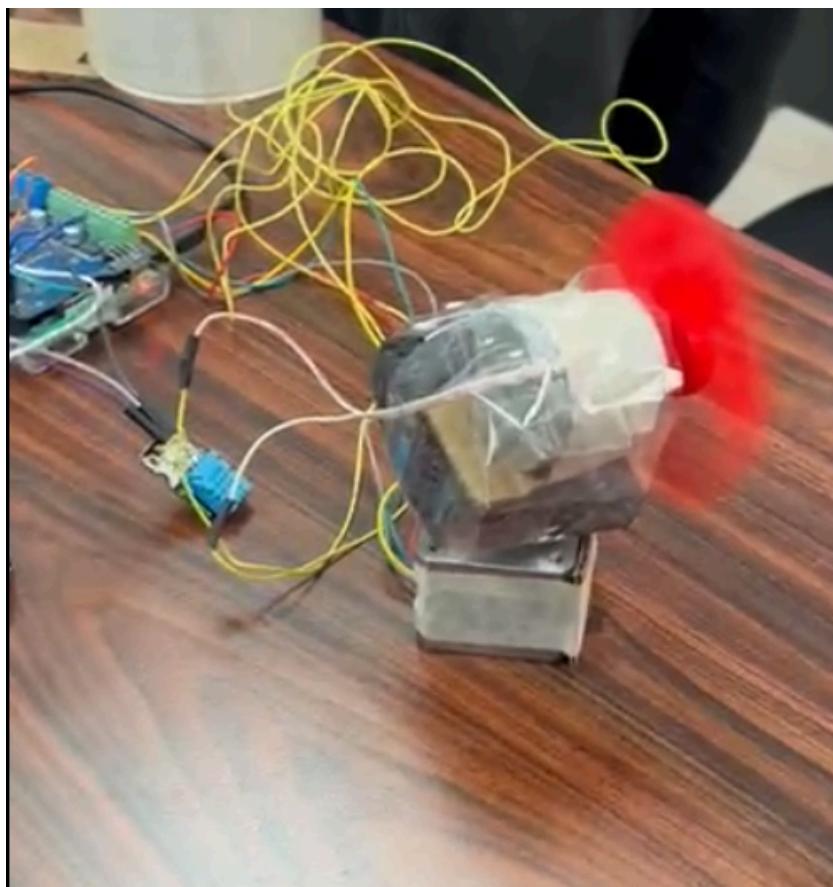
- Gpiozero
- Raspi_MotorHAT
- time

HERRAMIENTAS USADAS

- Raspberry Pi modelo 3
- Cables genéricos
- Motor CD, modelo: EG-510-9F2
- Servo motores
- Sensor Ultrasónico
- Sensor de humedad y temperatura DHT11
- Cinta adhesiva
- Tijeras y cortadores
- Hélices

IMÁGENES





LIMITANTES DEL PROYECTO

Integración de componentes: La correcta integración de los componentes electrónicos, como el sensor ultrasónico, el motor DC y el sensor de temperatura, se vio dificultada por posibles incompatibilidades en las conexiones o protocolos de comunicación.

Dificultades de integración: La integración de los componentes y la interacción entre hardware y software plantearon desafíos adicionales, como la sincronización entre el sensor ultrasónico y los motores, lo que pudo haber afectado la confiabilidad y la coherencia de las respuestas del sistema.

CONCLUSIONES GENERALES

El proyecto del abanico inteligente fue una gran experiencia que nos permitió mejorar nuestras habilidades en electrónica, programación y resolución de problemas. Fue un desafío interesante la forma de utilizar dos motores, unas hélices y un sensor ultrasónico para crear un abanico con capacidad de detección y movimiento inteligente.

Logramos el objetivo principal de incorporar la detección de objetos cercanos mediante el sensor ultrasónico y la oscilación controlada del motor paso a paso. Cabe mencionar que se añadió una funcionalidad de control de temperatura con un sensor correspondiente, pero nos vimos dificultades al combinar los componentes hardware mencionado antes con este último en el Raspberry Pi por cuestiones de combinaciones electrónicas.

El proceso de diseño e implementación del código fue una parte muy valiosa, ya que nos exigió investigar y evaluar distintas soluciones para controlar de manera coordinada los diferentes elementos del proyecto. Aprendimos a depurar y optimizar el funcionamiento mediante pruebas y modificaciones iterativas del programa.

REFLEXIÓN Y CONCLUSIÓN EN GENERAL

En general, este proyecto fue una experiencia enriquecedora que nos permitió afianzar conocimientos previos en electrónica, programación y resolución de problemas. Nos brindó también la oportunidad de desarrollar nuevas capacidades en el armado de circuitos, comunicación entre dispositivos y automatización de tareas mediante programación. Esperamos poder aplicar estas competencias adquiridas en futuros proyectos de tipo similar sea en la vida cotidiana o en un ambiente laboral.

El proyecto del abanico inteligente ha sido emocionante ya que nos ha permitido no solo consolidar nuestros conocimientos previos en programación y resolución de problemas, sino también explorar nuevas dificultades en el diseño y la implementación de soluciones creativas. En el desarrollo, tenemos obstáculos, destacando el problema de conexión entre el motor servo y el motor DC. Por estas dificultades, nos ha ayudado a mejorar nuestras habilidades técnicas.