

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

Fecha de Entrega:

09/03/2024

1. Informaciones generales	3
1.1 Descripción General	3
1.2 Objetivos	3
1.3 Alcance del proyecto	3
2. Especificaciones del desarrollo	3
2.0 Materiales	3
2.1 Funcionalidad	4
3. Reflexión Final	9
4. Anexo de Imágenes	10

1. Informaciones generales

1.1 Descripción General

Este proyecto consiste en poder controlar un servo motor horizontal y vertical mediante la interfaz de usuario. El usuario puede ajustar la posición de los servos utilizando las teclas de flecha derecha/izquierda para el servo horizontal y flecha arriba/abajo para el servo vertical. Además, el usuario puede modificar la velocidad de giro de los servos mediante las teclas '+' y '-'. El programa permite realizar capturas de imagen con una cámara cuando se presiona la tecla 'Enter', y se guarda la captura con un nombre de archivo que incluye la fecha, hora y las posiciones actuales de los servos. Esto como fin de tener una cámara que se pueda mover como un ojo humano, ya sea para tomar fotos en posiciones específicas.

1.2 Objetivos

1. Poder controlar los servos horizontal y vertical con las teclas de flecha.
2. Poder ajustar la velocidad de giro de los servos.
3. Hacer capturas de imágenes utilizando una cámara al presionar la tecla 'Enter', en diferentes posiciones.
4. Proporcionar una interfaz de usuario simple utilizando la biblioteca.

1.3 Alcance del proyecto

El proyecto se basa en el control básico de servos a través de una interfaz de usuario en la consola. No incluye funciones avanzadas para procesar imágenes ni configuraciones detalladas de la cámara. El código funciona en un ciclo continuo, lo que permite al usuario modificar las posiciones de los servos y tomar capturas de imagen cuando lo desee. Además, la velocidad de giro de los servos se puede ajustar según las preferencias del usuario.

2. Especificaciones del desarrollo

2.0 Materiales

- Raspberry Pi 4
- Cables
- 2 servo motores
- Camara Pi

2.1 Funcionalidad

Se importan las librerías necesarias como curses, gpiozero y datetime para la interfaz, control de servos y generar nombres de archivos respectivamente.

```
1 import curses
2 import os
3 from gpiozero import AngularServo
4 from gpiozero.pins.pigpio import PiGPIOFactory
5 from datetime import datetime
6 import argparse
```

Se inicializan los servos conectados a los pines 2 y 3, indicando el mínimo y máximo ángulo.

```
#Inicializando el Servo
pin_servo_horizontal = 2
pin_servo_vertical = 3
pigpio_factory = PiGPIOFactory()

servo_horizontal = AngularServo(pin_servo_horizontal, min_angle=0, max_angle=180, pin_factory=pigpio_factory)
servo_vertical = AngularServo(pin_servo_vertical, min_angle=0, max_angle=180, pin_factory=pigpio_factory)
```

- La función main contiene la interfaz con curses y toda la lógica del programa.
- Se imprimen instrucciones iniciales sobre los comandos.
- Se definen las variables de ángulo y velocidad inicial.

```

def main(stdscr):
    # Imprimiendo instrucciones.
    stdscr.clear()
    curses.curs_set(0)
    stdscr.addstr("Presiona '→' giro a la derecha.\n")
    stdscr.addstr("Presiona '←' giro a la izquierda.\n")
    stdscr.addstr("Presiona '+' aumentar velocidad de giro.\n")
    stdscr.addstr("Presiona '-' disminuir velocidad de giro.\n")
    stdscr.addstr("Presiona 'Enter' realizar captura.\n")
    stdscr.addstr("Presiona 'q' para salir.\n")

    # Variables de angulo y velocidad correspondientes.
    angulo_horizontal = 90
    angulo_vertical = 90

    constante_velocidad = args.velocidad

    #Inicializando el servo a angulo declarado.
    servo_horizontal.angle = angulo_horizontal
    servo_vertical.angle = angulo_vertical

```

- El bucle while capture continuamente la tecla presionada.
- Según la tecla se ajusta el ángulo de cada servo o la velocidad.
- Al presionar enter se toma una foto con rpicam, generando un nombre con fecha, hora y ángulos actuales.

```

19
20     while True:
21         # Captura de tecla
22
23         key = stdscr.getch()
24
25         if key == ord('q'):
26             break
27         elif key == curses.KEY_RIGHT:
28             angulo_horizontal -= constante_velocidad
29             angulo_horizontal = max(0, angulo_horizontal)
30             servo_horizontal.angle = angulo_horizontal
31         elif key == curses.KEY_LEFT:
32             angulo_horizontal += constante_velocidad
33             angulo_horizontal = min(180, angulo_horizontal)
34             servo_horizontal.angle = angulo_horizontal
35         elif key == curses.KEY_UP:
36             angulo_vertical -= constante_velocidad
37             angulo_vertical = max(0, angulo_vertical)
38             servo_vertical.angle = angulo_vertical
39         elif key == curses.KEY_DOWN:
40             angulo_vertical += constante_velocidad
41             angulo_vertical = min(180, angulo_vertical)
42             servo_vertical.angle = angulo_vertical
43         elif key == ord('+'):
44             if constante_velocidad + 25 > 180:
45                 print("No se puede aumentar mas de velocidad.")
46             else:
47                 constante_velocidad += 25
48                 print("Velocidad aumento a " + str(constante_velocidad) + "\n")
49         elif key == ord('-'):
50             if constante_velocidad - 25 < 0:
51                 print("No se puede disminuir mas de velocidad.\n")
52             else:
53                 constante_velocidad -= 25
54                 print("Velocidad disminuyo a " + str(constante_velocidad) + "\n")
55         elif key == curses.KEY_ENTER or key == 10 or key == 13:
56             # Obtener la fecha y hora actual
57             fecha_hora_actual = datetime.now()
58
59             # Formatear la fecha y hora en el formato deseado
60             fecha_hora_formateada = fecha_hora_actual.strftime('%Y-%m-%d_%H-%M-%S')
61             file_name = f"captura_{fecha_hora_formateada}_a_{angulo_vertical}_v_{angulo_horizontal}.jpg"
62
63             print("Captura Realizada. Guardada como JPG con el nombre: " + file_name)
64
65             command = f"rpicam-jpeg -t 1 -n -o {file_name} > /dev/null 2>&1"
66             os.system(command)

```

El argumento de línea de comandos define la velocidad inicial.

```

3 if __name__ == "__main__":
4     parser = argparse.ArgumentParser(description="velocidad")
5     parser.add_argument("--velocidad", type=int, default=25, help="velocidad")
6     args = parser.parse_args()

```

Curses.wrapper envuelve la función main para mostrar la interfaz.

```

92
93 curses.wrapper(main)

```

Código Completo:

```

import curses
import os
from gpiozero import AngularServo
from gpiozero.pins.pigpio import PiGPIOFactory
from datetime import datetime
import argparse

```

```

#Inicializando el Servo
pin_servo_horizontal = 2
pin_servo_vertical = 3
pigpio_factory = PiGPIOFactory()

servo_horizontal = AngularServo(pin_servo_horizontal, min_angle=0, max_angle=180,
pin_factory=pigpio_factory)
servo_vertical = AngularServo(pin_servo_vertical, min_angle=0, max_angle=180,
pin_factory=pigpio_factory)

def main(stdscr):
    # Imprimiendo instrucciones.
    stdscr.clear()
    curses.curs_set(0)
    stdscr.addstr("Presiona '→' giro a la derecha.\n")
    stdscr.addstr("Presiona '←' giro a la izquierda.\n")
    stdscr.addstr("Presiona '+' aumentar velocidad de giro.\n")
    stdscr.addstr("Presiona '-' disminuir velocidad de giro.\n")
    stdscr.addstr("Presiona 'Enter' realizar captura.\n")
    stdscr.addstr("Presiona 'q' para salir.\n")

    # Variables de angulo y velocidad correspondientes.
    angulo_horizontal = 90
    angulo_vertical = 90

    constante_velocidad = args.velocidad

#Inicializando el servo a angulo declarado.
servo_horizontal.angle = angulo_horizontal
servo_vertical.angle = angulo_vertical

while True:
    # Captura de tecla
    """
    '→' giro a la derecha.
    '←' giro a la izquierda.
    '+' aumentar velocidad
    '-' disminuir velocidad
    'Enter' realizar captura
    'q' para salir
    """
    key = stdscr.getch()

    if key == ord('q'):
        break

```

```

elif key == curses.KEY_RIGHT:
    angulo_horizontal -= constante_velocidad
    angulo_horizontal = max(0, angulo_horizontal)
    servo_horizontal.angle = angulo_horizontal
elif key == curses.KEY_LEFT:
    angulo_horizontal += constante_velocidad
    angulo_horizontal = min(180, angulo_horizontal)
    servo_horizontal.angle = angulo_horizontal
elif key == curses.KEY_UP:
    angulo_vertical -= constante_velocidad
    angulo_vertical = max(0, angulo_vertical)
    servo_vertical.angle = angulo_vertical
elif key == curses.KEY_DOWN:
    angulo_vertical += constante_velocidad
    angulo_vertical = min(180, angulo_vertical)
    servo_vertical.angle = angulo_vertical
elif key == ord('+'):
    if constante_velocidad + 25 > 180:
        print("No se puede aumentar mas de velocidad.")
    else:
        constante_velocidad += 25
        print("Velocidad aumento a " + str(constante_velocidad) + "\n")
elif key == ord('-'):
    if constante_velocidad - 25 < 0:
        print("No se puede disminuir mas de velocidad.\n")
    else:
        constante_velocidad -= 25
        print("Velocidad disminuyo a " + str(constante_velocidad) + "\n")
elif key == curses.KEY_ENTER or key == 10 or key == 13:
    # Obtener la fecha y hora actual
    fecha_hora_actual = datetime.now()

    # Formatear la fecha y hora en el formato deseado
    fecha_hora_formateada = fecha_hora_actual.strftime('%Y-%m-%d_%H-%M-%S')
    file_name =
f"captura_{fecha_hora_formateada}_a_{angulo_vertical}_v_{angulo_horizontal}_h.jpg"

print( "Captura Realizada. Guardada como JPG con el nombre: " + file_name)

command = f"rpicam-jpeg -t 1 -n -o {file_name} >/dev/null 2>&1"
os.system(command)

if __name__ == "__main__":
    parser = argparse.ArgumentParser(description="velocidad")
    parser.add_argument("--velocidad", type=int, default=25, help="velocidad")
    args = parser.parse_args()

curses.wrapper(main)

```

3. Reflexión Final

Luis Mateo: Siendo este el primer proyecto realizado en la materia, mi reflexión sería que ha sido desafiante, dado a que no la parte física de este tuvimos dificultades desde un principio, pero en el apartado del manejo de dispositivos, librerías, estuvo bien. Ha sido un buen primer proyecto para comenzar de lleno a aplicar lo que aprendimos en contextos más reales.

Christopher Paul: Este proyecto de programación aplicada si bien es un prototipo inicial, demuestra el potencial de automatizar el proceso de captura de imágenes. Esto podría extenderse a aplicaciones más complejas. Nos sirvió de base para desarrollar proyectos más amplios que integren visión por computadora, robótica, IoT. Por ejemplo, automatizando la detección y seguimiento de objetos. También podemos concluir que es un buen ejemplo de cómo construir prototipos funcionales con componentes hardware como una primera validación de concepto. Permite iterar y escalar la complejidad de manera incremental dependiendo de los componentes que tenemos a nuestra disposición en el transcurso de esta materia.

Gesbien Nuñez: Siendo este nuestro primer proyecto realizado en esta materia, lo mas desafiante fue diseñar la parte física que sostiene a los servos. Más allá de nuestra implementación del código, este proyecto nos demandó creatividad para poder solucionar el problema de poder sostener firmemente los servos y que la cámara no tambaleara cada vez que se movía. Para mi ha sido un muy buen primer proyecto ya que nos demandó desde un principio no solo pensar en el código sino que también tuvimos que tener en cuenta el factor fisico que en muchos casos es de vital importancia y muchas veces no le tomamos mucha importancia.

4. Anexo de Imágenes



















