Date of publication 04. 07. 2025, date of current version 04, 07, 2025.
Digital Object Identifier 10.1109/ACCESS.2025. Doi Number

ThermoScout: Thermoelectric-powered Surveillance Robot (July 2025)

Surabhi M[1], Dishita Reddy Garudadri[2], Kshama Bhat K[3] and Reshmi M[4]
[1,2,3,4]R.V. College of Engineering, Bengaluru, 560059 INDIA
[1]Computer Science Engineering Department, R.V. College of Engineering, Bengaluru, 560059 INDIA
[2]Computer Science Engineering Department, R.V. College of Engineering, Bengaluru, 560059 INDIA
[3]Electronics and Communication Engineering Department, R.V. College of Engineering, Bengaluru, 560059 INDIA
[4]Biotechnology Department, R.V. College of Engineering, Bengaluru, 560059 INDIA
Corresponding author: Surabhi.M (e-mail: surabhim.cs24@rvce.edu.in).
This work was supported in part by R.V. College of Engineering under Experiential Learning (Theme: Energy).

ABSTRACT

A thermoelectric self-powered surveillance robot is presented as an autonomous platform that eliminates dependence on external power sources by harvesting waste heat through the Seebeck effect. Thermoelectric generators (TEGs) convert thermal gradients into electrical energy to enable continuous operation in remote and industrial environments. The system integrates a Raspberry Pi 4B for processing and employs the Roboflow platform to develop custom AI models for real-time fire and knife detection. Autonomous navigation is achieved using ultrasonic sensors for obstacle detection and distance measurement. A servo-controlled camera enhances visual coverage, while an alert mechanism with buzzers and LED indicators provides immediate notifications upon threat identification. Energy optimization is accomplished through four TEG modules combined with heat sinks, boost converters for voltage regulation and rechargeable battery storage to ensure consistent power delivery. This approach demonstrates the feasibility of sustainable, energy-autonomous surveillance in applications where up to 70% of energy is typically lost as waste heat. Key features include thermoelectric energy harvesting, AI-based threat detection and fully autonomous patrolling, offering a scalable solution for smart security in industrial and remote settings

INDEX TERMS

Thermoelectric generators, autonomous surveillance, waste heat recovery, Raspberry Pi, AI-based threat detection, energy harvesting, ultrasonic navigation, renewable power systems.

I. INTRODUCTION

Energy-autonomous robotic systems are gaining increasing attention as sustainable solutions for surveillance in remote and industrial environments. Conventional surveillance platforms often rely on grid power or frequent battery replacements, limiting their operational continuity and scalability. Meanwhile, a significant portion of thermal energy generated by industrial equipment and internal combustion engines is dissipated as waste heat, representing an untapped energy source.

Recent advances in thermoelectric generators (TEGs) have enabled efficient conversion of thermal gradients into usable electrical energy through the Seebeck effect. Coupled with embedded AI platforms, such as the Raspberry Pi, and lightweight machine learning frameworks, it is now feasible to implement intelligent monitoring systems capable of autonomous operation without external power dependencies. This work presents the design and implementation of a thermoelectric self-powered surveillance robot that integrates waste heat recovery, autonomous navigation, and real-time threat detection. The system leverages custom-trained AI models for fire and weapon identification, combined with ultrasonic obstacle avoidance, to provide continuous patrolling in challenging environments. The approach demonstrates a scalable and sustainable method for enhancing security infrastructure while reducing energy waste and operational overhead. The proposed solution highlights the potential for integrating renewable energy harvesting with smart security applications to create resilient, low-maintenance surveillance systems.

II. METHODOLOGY

Overview

The development of the ThermoScout surveillance robot followed a structured, phase-wise methodology aimed at achieving full system integration of thermoelectric energy harvesting, autonomous mobility and AI-based threat detection. The methodology can be broadly categorized into five key stages: system design, hardware development, software implementation, testing and final integration.

A. System Design

The project began with a detailed system design phase, where the hardware and software requirements were defined. The design focused on selecting energy-efficient components such as TEG modules, Raspberry Pi 4B, DC geared motors and suitable sensors. A modular, multi-level chassis design was planned to accommodate thermal, mechanical and electronic subsystems efficiently.

B. Hardware Development

The hardware was assembled on a 4-wheeled mobile chassis. Four TEC1-12706 thermoelectric generators were mounted on heat sinks to harvest waste heat and connected in series for higher voltage output. A boost converter was used to step up the generated voltage, which was then stored in rechargeable Li-ion batteries. Power was distributed via buck converters to supply regulated voltage to the Raspberry Pi and other modules. Additional components such as the ultrasonic sensor, servo motor and motor driver (L298N) were installed for obstacle detection and motion control.

C. Software Implementation

The control system was programmed using Python, executed on the Raspberry Pi OS. The RPi.GPIO library was used for managing GPIO pins, while OpenCV handled real-time video feed processing. Custom object detection models (fire and knife recognition) were trained using the Roboflow platform and deployed on the Raspberry Pi in TensorFlow Lite format for real-time, offline inference.

D. Testing and Calibration

Each subsystem was individually tested and calibrated. TEG performance was measured under controlled temperature differences to validate voltage output. Motor control and obstacle avoidance were verified via sensor input. AI model inference was tested with live video feeds to ensure reliable threat detection. Calibration included fine-tuning the servo motor angles and obstacle detection thresholds.

E. Final Integration

After validating individual subsystems, a full integration of hardware and software was performed. The robot was tested in a simulated environment to ensure coordination between energy harvesting, motion, sensing, and AI-based response. The integration confirmed that the robot could operate autonomously using harvested energy, while detecting and responding to threats in real-time.

System Design and Architecture

The ThermoScout system is designed as an autonomous surveillance robot capable of functioning in off-grid environments by harvesting thermal energy and integrating AI-based threat detection. The design emphasizes modularity, energy efficiency, and real-time responsiveness. The architecture comprises three major subsystems: power generation and management, mobility and sensing, and AI-driven perception and control.

A. Hardware Architecture

1) Power Module

The power subsystem is centred around Thermoelectric Generators (TEGs) that convert temperature differences into electrical energy based on the Seebeck effect. Four TEC1-12706 modules are connected in series to maximize voltage output. A boost converter (MT3608) elevates the low TEG voltage to charge two 3.7V Li-ion 18650 batteries. To ensure voltage stability for the 6V buck converter that supplies the motor driver and actuators.

2) Mobility System

The locomotion system consists of four DC geared motors, each coupled to a wheel and controlled via an L298N motor driver module. The Raspberry Pi controls the direction and speed of the motors through GPIO pins, enabling forward, reverse and turning motions.

3) Sensing System

An ultrasonic sensor (HC-SR04) is mounted on a servo motor to dynamically scan the surrounding environment. This configuration allows for real-time obstacle detection and avoidance by adjusting the robot's direction. Additionally, a Raspberry Pi camera module is mounted at the front for continuous video capture, forming the input for AI-based object detection.

B. Software Architecture

1) Control System

The system is programmed using Python 3 on Raspberry Pi OS. The RPi.GPIO library is utilized to interface the Raspberry Pi with hardware components including motors, sensors and actuators. Movement decisions are based on real-time distance data from the ultrasonic sensor.

2) AI-Based Threat Detection

Custom-trained object detection models for fire and knife recognition are developed using the Roboflow platform. These models are deployed locally on the Raspberry Pi, allowing offline, real-time inference. The OpenCV library handles video feed processing, bounding box rendering and alert visualization.

3) Remote Access and Debugging

The Raspberry Pi is operated in headless mode using VNC Viewer, which enables remote desktop access for code deployment, debugging, and monitoring system performance.

C. Data Flow and Functional Overview

The complete system workflow can be summarized as follows:

- Thermal Input → TEGs generate voltage → Boost converter steps up voltage → Batteries store and regulate power via buck converters.
- Camera Input → Raspberry Pi processes video → AI model identifies threats → Alerts are generated (via buzzer and LEDs).
- Ultrasonic Sensor Data → Raspberry Pi evaluates distance → Motor driver adjusts movement accordingly to avoid obstacles.

Hardware Components

Sensors

- Ultrasonic Sensor: Enables real-time obstacle detection and distance analyzation during autonomous navigation.
- Raspberry Pi Camera Module: Captures real-time surveillance video and supports AI-based image processing for fire and threat detection.

Microcontroller and Processing Unit

- Raspberry Pi 4B: Provides advanced AI model processing, image analysis, and overall system control with enhanced computational capabilities.

Power System

- Thermoelectric Generators (TEGs) (4 units): Harvest waste heat energy via the Seebeck effect, converting temperature differentials into electrical power.
- MT3608 Boost Converter: Boosts the low voltage generated by the TEGs to meet the power requirements of system components.
- Rechargeable Li-ion Batteries (2 units): Store harvested energy and supply stable power to the rover's subsystems.
- Heat Sinks (4 units): Maintain optimal temperature differentials across TEG modules to maximize power generation efficiency.

Locomotion System

- DC Gear Motors (4 units): Enable autonomous movement and navigation across varied terrains.
- Motor Driver Module: Controls motor speed, direction, and movement coordination.
- Wheels (4 units): Provide stable mobility and manoeuvrability.
- Servo Motor: Controls camera positioning to support dynamic scanning and target tracking.

Storage and Connectivity

- MicroSD Card: Stores AI models, image processing algorithms, and system data required for operation.

Software and Programming

- Python Programming: Used to implement AI models, image processing pipelines, and system control logic on the Raspberry Pi platform.
- Roboflow: Supports dataset management, image annotation, and optimization of custom object detection models for fire and knife threat recognition.
- VNC Viewer: Enables remote desktop access and real-time monitoring during development and testing.
- Raspberry Pi OS: Provides the operating system platform necessary to run all software components and manage AI inference tasks

Power Optimization in Thermoelectric Energy Harvesting

Power optimization in the thermoelectric modules was achieved through a combination of strategic placement, thermal management, and electrical conditioning techniques. TEG units were mounted with high-conductivity thermal paste to ensure efficient heat transfer from the hot surfaces, while large aluminum heat sinks were attached to the cold side to maximize the temperature gradient across each module. Electrical output from the TEGs was first boosted using an MT3608 boost converter and then regulated through a buck converter to provide stable power suitable for driving the motors via the motor driver module. Multiple TEG modules were connected in series to increase voltage output while maintaining consistency under varying load conditions. These measures ensured that the harvested energy remained sufficient and stable to power the locomotion system, enabling autonomous movement without reliance on external power sources.

Testing and Validation

1. Testing Process

Testing was performed to evaluate the functionality and reliability of all major subsystems of the thermoelectric self-powered surveillance robot. The thermoelectric energy harvesting capability was assessed by measuring the output voltage of the TEG modules across controlled temperature differentials to confirm their performance under realistic conditions. Power system stability tests involved monitoring the boost converter output and battery charging behaviour under various load scenarios to ensure consistent voltage delivery. Locomotion and motor control were tested by operating the robot in multiple environments to observe responsiveness and manoeuvrability. Ultrasonic obstacle detection was evaluated through repeated trials in which the robot autonomously detected and avoided obstacles. Image acquisition and AI threat detection testing involved capturing live video streams and analysing the accuracy of the custom-trained models in detecting and classifying fire and knife threats in real time. An overall integration test was conducted to verify that all hardware components and processing modules functioned together without conflicts.

2. Validation

Validation confirmed that the thermoelectric modules generated sufficient voltage to support continuous system operation and that the power delivery remained stable under dynamic load conditions. The robot demonstrated precise motor control and reliable locomotion across different surfaces. Ultrasonic sensors consistently provided accurate distance measurements, enabling effective obstacle avoidance. The AI-based image processing pipeline successfully detected, classified and localized fire and knife threats with consistent performance during live operation. The integrated system operated stably, with all subsystems communicating effectively and maintaining reliable functionality, demonstrating the feasibility of a fully energy-autonomous, AI-enabled surveillance platform.

Challenges and Limitations

The development and implementation of the *ThermoScout* surveillance robot involved several technical and practical challenges. These challenges influenced design decisions and highlight areas for future optimization.

A. Challenges Encountered

1) Low Voltage Output from TEGs

Thermoelectric Generators (TEGs) produced very low voltage under moderate temperature gradients (~0.19 V per module at $\Delta T = 30°C$), which was insufficient to directly power the system components.

Solution: A boost converter (MT3608) was integrated to step up the voltage, and energy was stored in Li-ion 18650 batteries for consistent power supply.

2) Power Distribution Instability

Simultaneous operation of motors, sensors and the Raspberry Pi caused voltage fluctuations and occasional resets during peak load.

Solution: Implemented buck converters to regulate voltage output separately for the Raspberry Pi (5V) and the motor driver (6V), ensuring stable system operation.

3) Remote System Access

Initially, the Raspberry Pi failed to connect in headless mode, hindering software debugging and deployment. **Solution**: VNC Viewer was used to establish remote desktop access, enabling code updates and real-time monitoring without the need for a dedicated display.

4) Sensor Calibration and Accuracy

The ultrasonic sensor readings were inconsistent due to environmental noise and poor positioning. **Solution**: The sensor was mounted on a servo motor to enable directional scanning, improving obstacle detection coverage and accuracy.

5) AI Model Deployment on Edge Device

Running real-time AI models on the Raspberry Pi led to latency and processing delays due to limited computational resources.

Solution: Optimized the model by training lightweight versions using Roboflow and testing it for better performance on embedded systems.

B. Limitations

- Limited Energy Harvesting Capacity: The efficiency of TEGs is highly dependent on the temperature gradient, which limits power generation under normal room conditions.
- Surface Constraints: The current prototype is suitable for smooth indoor surfaces; its mobility is restricted on uneven terrain or outdoors.
- Partial Threat Detection Scope: AI detection is limited to fire and knife threats. Additional training and datasets are required for broader threat identification.
- Short-Term Testing Environment: The system was tested primarily in a controlled lab environment. Extended field testing under real heat sources and remote conditions is yet to be performed.
- No Wireless Communication Module: Currently, the robot operates offline. Integration with IoT platforms or GSM/Wi-Fi modules is a potential future enhancement.

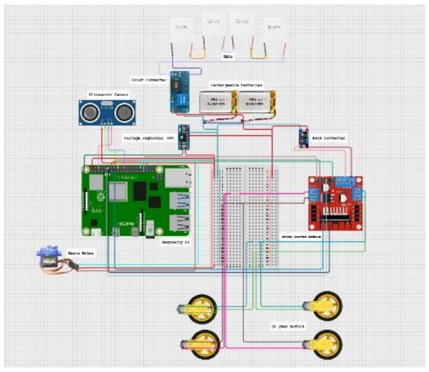


Fig 1(a). Circuit Diagram

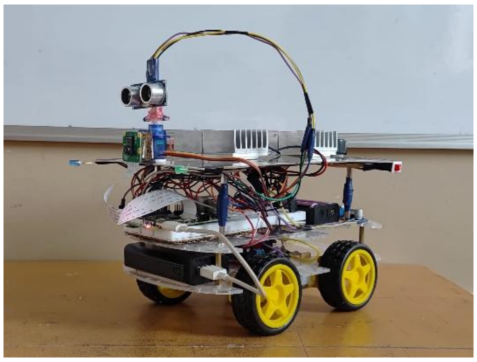


Fig 1(b). ThermoScout Prototype

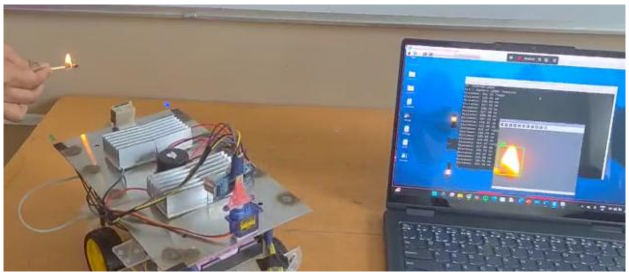


Fig 2(a). Fire Detection on Model

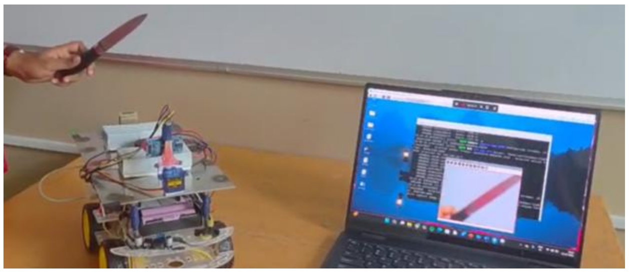
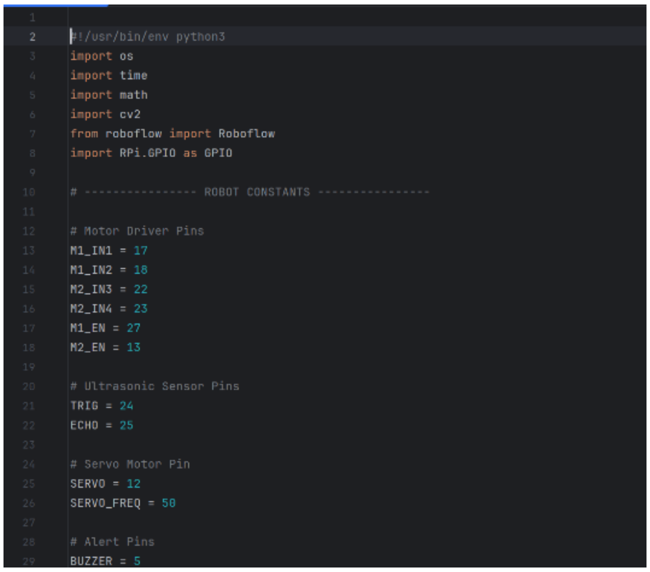
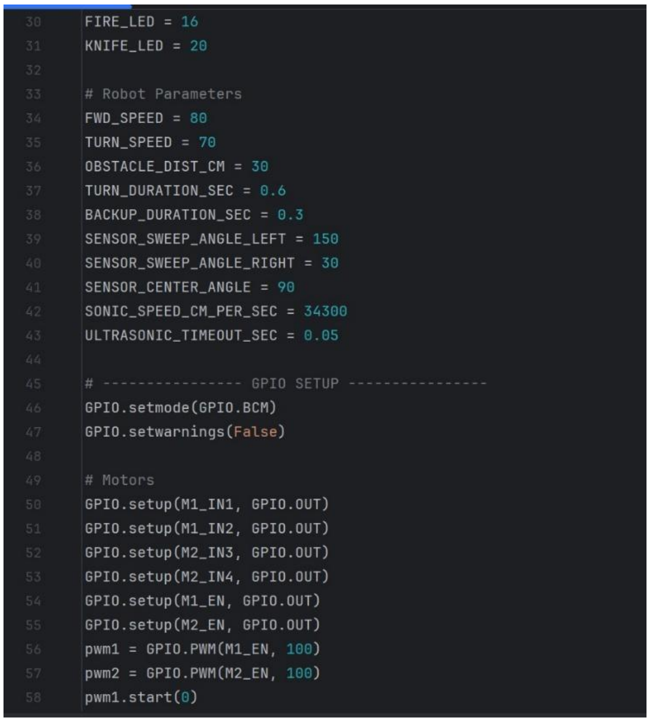


Fig 2(b). Knife Detection on Model

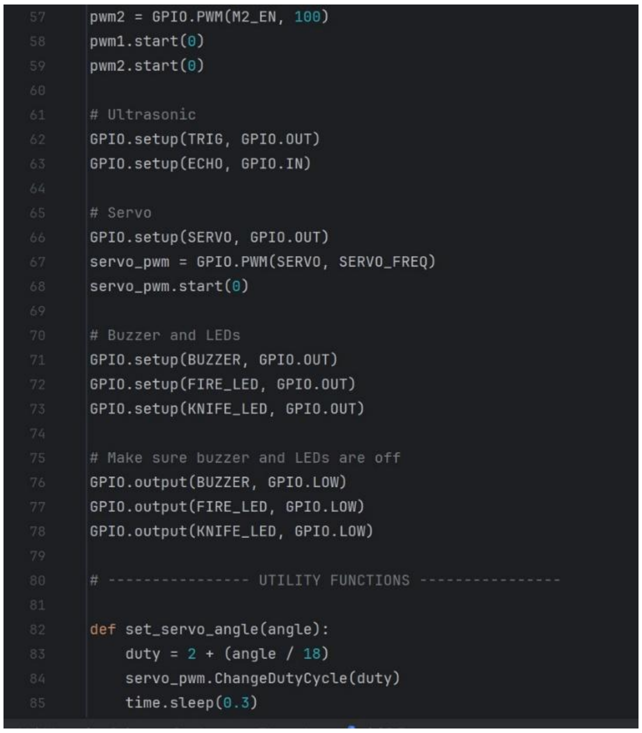
CODE SNIPPET



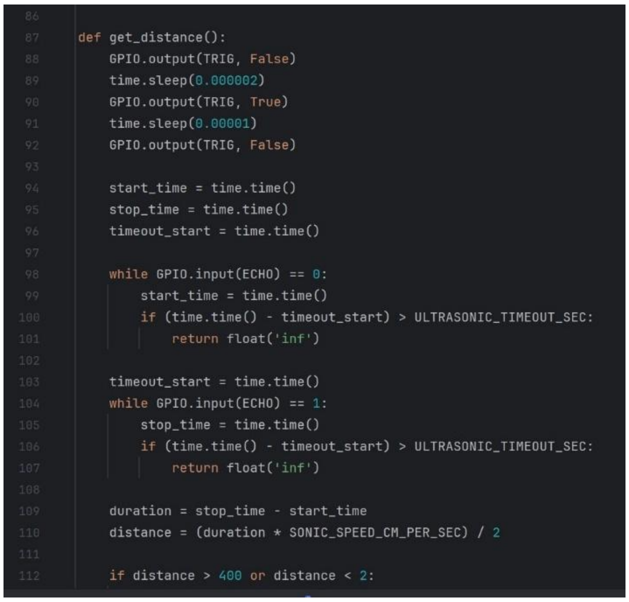
```python

#!/usr/bin/env python3
import os
import time
import math
import cv2
from roboflow import Roboflow
import RPi.GPIO as GPIO

# --------------- ROBOT CONSTANTS ---------------

# Motor Driver Pins
M1_IN1 = 17
M1_IN2 = 18
M2_IN3 = 22
M2_IN4 = 23
M1_EN = 27
M2_EN = 13

# Ultrasonic Sensor Pins
TRIG = 24
ECHO = 25

# Servo Motor Pin
SERVO = 12
SERVO_FREQ = 50

# Alert Pins
BUZZER = 5
```



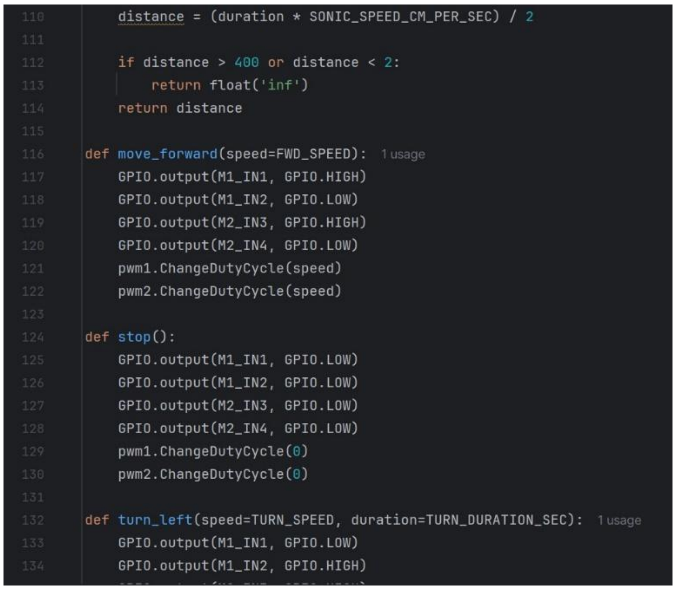
```python
FIRE_LED = 16
KNIFE_LED = 20

# Robot Parameters
FWD_SPEED = 80
TURN_SPEED = 70
OBSTACLE_DIST_CM = 30
TURN_DURATION_SEC = 0.6
BACKUP_DURATION_SEC = 0.3
SENSOR_SWEEP_ANGLE_LEFT = 150
SENSOR_SWEEP_ANGLE_RIGHT = 30
SENSOR_CENTER_ANGLE = 90
SONIC_SPEED_CM_PER_SEC = 34300
ULTRASONIC_TIMEOUT_SEC = 0.05

# --------------- GPIO SETUP ---------------
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

# Motors
GPIO.setup(M1_IN1, GPIO.OUT)
GPIO.setup(M1_IN2, GPIO.OUT)
GPIO.setup(M2_IN3, GPIO.OUT)
GPIO.setup(M2_IN4, GPIO.OUT)
GPIO.setup(M1_EN, GPIO.OUT)
GPIO.setup(M2_EN, GPIO.OUT)
pwm1 = GPIO.PWM(M1_EN, 100)
pwm2 = GPIO.PWM(M2_EN, 100)
pwm1.start(0)
```



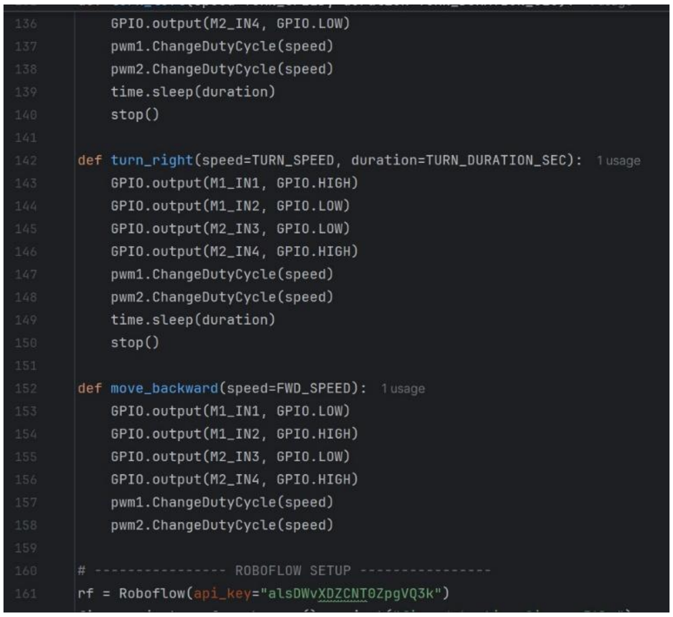
```python
pwm2 = GPIO.PWM(M2_EN, 100)
pwm1.start(0)
pwm2.start(0)

# Ultrasonic
GPIO.setup(TRIG, GPIO.OUT)
GPIO.setup(ECHO, GPIO.IN)

# Servo
GPIO.setup(SERVO, GPIO.OUT)
servo_pwm = GPIO.PWM(SERVO, SERVO_FREQ)
servo_pwm.start(0)

# Buzzer and LEDs
GPIO.setup(BUZZER, GPIO.OUT)
GPIO.setup(FIRE_LED, GPIO.OUT)
GPIO.setup(KNIFE_LED, GPIO.OUT)

# Make sure buzzer and LEDs are off
GPIO.output(BUZZER, GPIO.LOW)
GPIO.output(FIRE_LED, GPIO.LOW)
GPIO.output(KNIFE_LED, GPIO.LOW)

# --------------- UTILITY FUNCTIONS ---------------

def set_servo_angle(angle):
    duty = 2 + (angle / 18)
    servo_pwm.ChangeDutyCycle(duty)
    time.sleep(0.3)
```



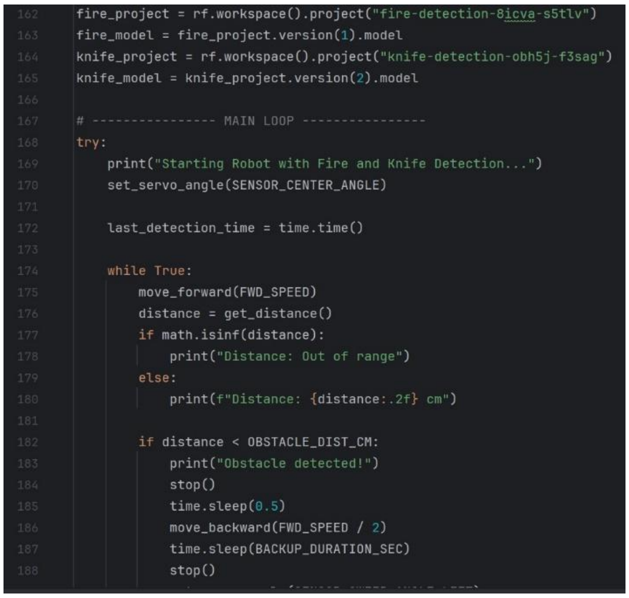
```python

def get_distance():
    GPIO.output(TRIG, False)
    time.sleep(0.000002)
    GPIO.output(TRIG, True)
    time.sleep(0.00001)
    GPIO.output(TRIG, False)

    start_time = time.time()
    stop_time = time.time()
    timeout_start = time.time()

    while GPIO.input(ECHO) == 0:
        start_time = time.time()
        if (time.time() - timeout_start) > ULTRASONIC_TIMEOUT_SEC:
            return float('inf')

    timeout_start = time.time()
    while GPIO.input(ECHO) == 1:
        stop_time = time.time()
        if (time.time() - timeout_start) > ULTRASONIC_TIMEOUT_SEC:
            return float('inf')

    duration = stop_time - start_time
    distance = (duration * SONIC_SPEED_CM_PER_SEC) / 2

    if distance > 400 or distance < 2:
```



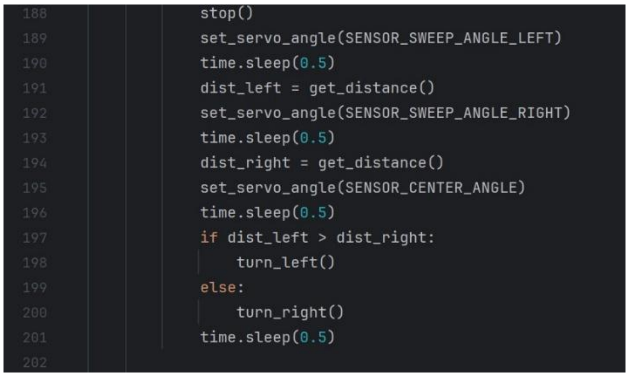
```python
    distance = (duration * SONIC_SPEED_CM_PER_SEC) / 2

    if distance > 400 or distance < 2:
        return float('inf')
    return distance

def move_forward(speed=FWD_SPEED):  1 usage
    GPIO.output(M1_IN1, GPIO.HIGH)
    GPIO.output(M1_IN2, GPIO.LOW)
    GPIO.output(M2_IN3, GPIO.HIGH)
    GPIO.output(M2_IN4, GPIO.LOW)
    pwm1.ChangeDutyCycle(speed)
    pwm2.ChangeDutyCycle(speed)

def stop():
    GPIO.output(M1_IN1, GPIO.LOW)
    GPIO.output(M1_IN2, GPIO.LOW)
    GPIO.output(M2_IN3, GPIO.LOW)
    GPIO.output(M2_IN4, GPIO.LOW)
    pwm1.ChangeDutyCycle(0)
    pwm2.ChangeDutyCycle(0)

def turn_left(speed=TURN_SPEED, duration=TURN_DURATION_SEC):  1 usage
    GPIO.output(M1_IN1, GPIO.LOW)
    GPIO.output(M1_IN2, GPIO.HIGH)
```



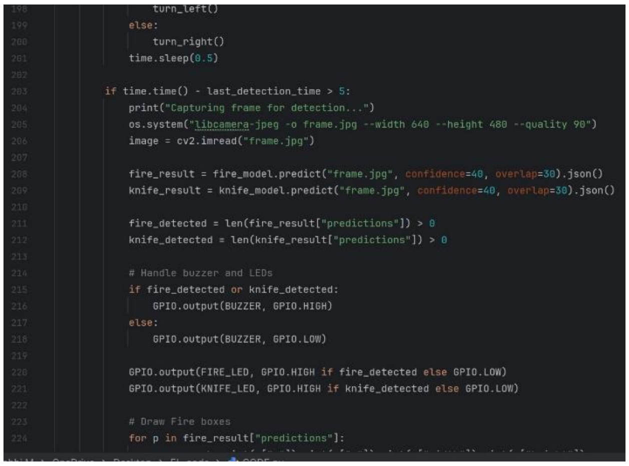
```python
    GPIO.output(M2_IN4, GPIO.LOW)
    pwm1.ChangeDutyCycle(speed)
    pwm2.ChangeDutyCycle(speed)
    time.sleep(duration)
    stop()

def turn_right(speed=TURN_SPEED, duration=TURN_DURATION_SEC):  1 usage
    GPIO.output(M1_IN1, GPIO.HIGH)
    GPIO.output(M1_IN2, GPIO.LOW)
    GPIO.output(M2_IN3, GPIO.LOW)
    GPIO.output(M2_IN4, GPIO.HIGH)
    pwm1.ChangeDutyCycle(speed)
    pwm2.ChangeDutyCycle(speed)
    time.sleep(duration)
    stop()

def move_backward(speed=FWD_SPEED):  1 usage
    GPIO.output(M1_IN1, GPIO.LOW)
    GPIO.output(M1_IN2, GPIO.HIGH)
    GPIO.output(M2_IN3, GPIO.LOW)
    GPIO.output(M2_IN4, GPIO.HIGH)
    pwm1.ChangeDutyCycle(speed)
    pwm2.ChangeDutyCycle(speed)

# ---------------- ROBOFLOW SETUP ----------------
rf = Roboflow(api_key="alsDWvXDZCNT0ZpgVQ3k")
```



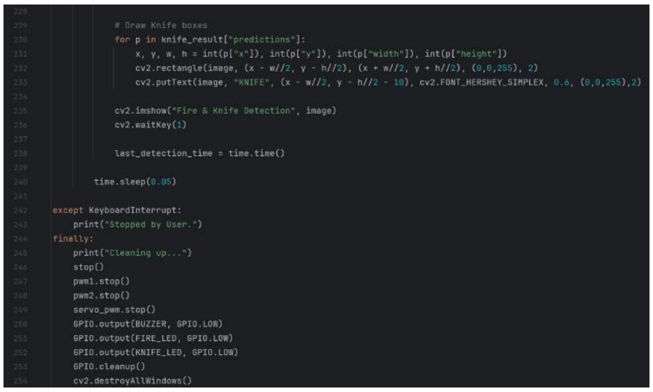
```python
fire_project = rf.workspace().project("fire-detection-8icva-s5tlv")
fire_model = fire_project.version(1).model
knife_project = rf.workspace().project("knife-detection-obh5j-f3sag")
knife_model = knife_project.version(2).model

# ---------------- MAIN LOOP ----------------
try:
    print("Starting Robot with Fire and Knife Detection...")
    set_servo_angle(SENSOR_CENTER_ANGLE)

    last_detection_time = time.time()

    while True:
        move_forward(FWD_SPEED)
        distance = get_distance()
        if math.isinf(distance):
            print("Distance: Out of range")
        else:
            print(f"Distance: {distance:.2f} cm")

        if distance < OBSTACLE_DIST_CM:
            print("Obstacle detected!")
            stop()
            time.sleep(0.5)
            move_backward(FWD_SPEED / 2)
            time.sleep(BACKUP_DURATION_SEC)
            stop()
```



```python
            stop()
            set_servo_angle(SENSOR_SWEEP_ANGLE_LEFT)
            time.sleep(0.5)
            dist_left = get_distance()
            set_servo_angle(SENSOR_SWEEP_ANGLE_RIGHT)
            time.sleep(0.5)
            dist_right = get_distance()
            set_servo_angle(SENSOR_CENTER_ANGLE)
            time.sleep(0.5)
            if dist_left > dist_right:
                turn_left()
            else:
                turn_right()
            time.sleep(0.5)
```



```python
                turn_left()
            else:
                turn_right()
            time.sleep(0.5)

        if time.time() - last_detection_time > 5:
            print("Capturing frame for detection...")
            os.system("libcamera-jpeg -o frame.jpg --width 640 --height 480 --quality 90")
            image = cv2.imread("frame.jpg")

            fire_result = fire_model.predict("frame.jpg", confidence=40, overlap=30).json()
            knife_result = knife_model.predict("frame.jpg", confidence=40, overlap=30).json()

            fire_detected = len(fire_result["predictions"]) > 0
            knife_detected = len(knife_result["predictions"]) > 0

            # Handle buzzer and LEDs
            if fire_detected or knife_detected:
                GPIO.output(BUZZER, GPIO.HIGH)
            else:
                GPIO.output(BUZZER, GPIO.LOW)

            GPIO.output(FIRE_LED, GPIO.HIGH if fire_detected else GPIO.LOW)
            GPIO.output(KNIFE_LED, GPIO.HIGH if knife_detected else GPIO.LOW)

            # Draw Fire boxes
            for p in fire_result["predictions"]:
```



```
            # Draw Knife boxes
            for p in knife_result["predictions"]:
                x, y, w, h = int(p["x"]), int(p["y"]), int(p["width"]), int(p["height"])
                cv2.rectangle(image, (x - w//2, y - h//2), (x + w//2, y + h//2), (0,0,255), 2)
                cv2.putText(image, "KNIFE", (x - w//2, y - h//2 - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.6, (0,0,255),2)

            cv2.imshow("Fire & Knife Detection", image)
            cv2.waitKey(1)

            last_detection_time = time.time()

        time.sleep(0.05)

except KeyboardInterrupt:
    print("Stopped by User.")
finally:
    print("Cleaning up...")
    stop()
    pwm1.stop()
    pwm2.stop()
    servo_pwm.stop()
    GPIO.output(BUZZER, GPIO.LOW)
    GPIO.output(FIRE_LED, GPIO.LOW)
    GPIO.output(KNIFE_LED, GPIO.LOW)
    GPIO.cleanup()
    cv2.destroyAllWindows()
```

III. MATH

A. EQUATIONS

1. Ohm's Law for Circuit Design
The relationship between voltage, current and resistance is defined as:

$$V = I \, X \, R$$

where:

V = Voltage across the circuit (V)

I = Current flowing through the circuit (A)

R = Resistance of the circuit (Ω)

2. TEG Output Voltage from Temperature Difference
The open-circuit voltage of a TEG is approximately:

$$V_{TEG} = S \, x \, \Delta T$$

where:

V_{TEG} = Output voltage of the TEG (V)

S = Seebeck coefficient (V/°C) → typically 0.05 V/°C for TEC1-12706

ΔT = Temperature difference across the TEG (°C)

3. Power generated by TEG
The power generated is:

$$P_{TEG} = V_{TEG} \, x \, I_{TEG}$$

where:

P_{TEG} = Power output (W)

V_{TEG} = Output voltage (V)

I_{TEG} = Output current (A)

4. Energy Consumption of System

$$E = V_{supply} \, x \, I_{system} \times t$$

where:

E = Energy consumed (J or Wh)

V_{supply} = Voltage supplied (V)

I_{system} = Current consumed (A)

t = Time (s or h)

5. Battery Runtime

$$t_{runtime} = I_{drawn} / C_{battery}$$

where:

$t_{runtime}$ = Battery runtime (h)

$C_{battery}$ = Battery capacity (Ah)

I_{drawn} = Current drawn (A)

6. Boost Converter Output Power

$$P_{boost} = \eta \times P_{TEG}$$

where:

P_{boost} = Output power after boost converter (W)

η = Efficiency (typically 0.85)

P_{TEG} = Input power to converter (W)

7. Ultrasonic Distance Measurement

$$D = (v \times t) / 2$$

where:

D = Distance (m or cm)

v = Speed of sound (~343 m/s)

t = Time for echo (s)

IV. UNITS

1. Voltage:
Primary: Volts (V)
Secondary: Millivolts (mV) or microvolts (μV) for smaller measurements, if needed.

2. Current:
Primary: Amperes (A)
Secondary: Milliamperes (mA) or microamperes (μA) for smaller measurements, if needed.

3. Temperature:
Primary: Celsius (°C)
Secondary: Kelvin (K) for scientific purposes, if needed.

4. Power:
Primary: Watts (W)

5. Energy:
Primary: Joules (J)
Secondary: Watt-hours (Wh)

6. Resistance:
Primary: Ohms (Ω)
Secondary: Kiloohms (kΩ)

7. Distance (Ultrasonic):
Primary: Centimeters (cm)
Secondary: Millimeters (mm)

8. Time:
Primary: Seconds (s)
Secondary: Milliseconds (ms)

V. SOME COMMON MISTAKES

1. **Incorrect Power Supply Distribution**

A common mistake in thermoelectric-powered systems is underestimating the power requirements of the Raspberry Pi, motors, and camera modules. When all components share a single unstable power source, the Pi may reboot or freeze as soon as motors activate or inference begins.

Solution: Use dedicated buck-boost converters rated for sufficient current output, and isolate motor power from sensitive electronics. Always measure voltage under load to verify stability before connecting devices.

2. **Improper TEG Placement and Heat Dissipation**

Thermoelectric generators often fail to produce usable voltage if the temperature gradient is insufficient or heat sinks are too small. Placing TEGs without proper contact or thermal paste significantly reduces efficiency.

Solution: Mount TEGs with good thermal interfaces, large aluminium heat sinks, or active cooling fans to maintain a high ΔT across the module. Verify temperature difference using a thermometer during operation.

3. Loose Connections in Motor Driver Wiring

Loose or poor-quality jumper wires connected to the motor driver can cause intermittent motor operation, sudden stops, or erratic movement when the robot is in motion.

Solution: Use secure screw terminals or soldered connections for motor outputs. Confirm all grounds share a common reference point to prevent floating voltages.

4. Camera Resource Conflicts

Running video capture and AI inference simultaneously on the Raspberry Pi without optimization often leads to CPU overload, memory exhaustion or crashing processes.

Solution: Optimize inference pipelines by lowering image resolution and frame rate, selecting lightweight models and avoiding multiple concurrent camera processes.

5. Incorrect Ultrasonic Sensor Timing

Ultrasonic sensors may return inconsistent or false readings if echo pulse timing is misconfigured or if trigger pulses are sent too frequently without adequate delay.

Solution: Include appropriate delays (for example, 60 ms) between measurements and implement averaging over multiple readings to improve stability.

6. Underestimating Thermal Losses in TEG Wiring

Using thin wires or long connections between the TEG modules and the boost converter can introduce significant voltage drops, reducing available power.

Solution: Use short, thick-gauge wires to connect TEG outputs and verify voltage at the converter input terminals under load conditions.

7. Lack of Robust Error Handling in Code

Neglecting to include error handling can cause the system to freeze or crash if a sensor disconnects, a process fails, or the AI model becomes unresponsive.

Solution: Implement exception handling, status checks, and watchdog timers to automatically restart or log failures without requiring a manual reset.

8. Incorrect Boost Converter Adjustment

Setting the boost converter output voltage too high creates a risk of damaging the Raspberry Pi or causing unstable operation.

Solution: Adjust the boost converter output carefully using a Multimeter, and target a regulated voltage around 5.1 V to power the Pi safely.

VI. GRAPHICS PRESENTATION

TEG

Temp on Cold Side = 27°C

Temp on Hot Side	Temp Diff	Voltage generated
57°C	30°C	0.19 V
77°C	50°C	0.38 V
60°C	33°C	0.22 V

Based on the graph in Figure 3., it is observed that the voltage generated by the thermoelectric modules increases proportionally with the temperature difference across the TEG surfaces. This behavior is clearly visible when comparing the output at 30 °C and 50 °C temperature differentials, where the voltage rises from approximately 0.19 V to 0.38 V. The trend demonstrates that maintaining a higher temperature gradient results in a higher electrical output, although the increase is not strictly linear across all data points. This slight deviation may be attributed to variations in heat sink performance and minor inconsistencies in contact resistance during measurements. All readings were taken using the same set of TEG modules under consistent load conditions, with the hot side temperature regulated and thermal paste applied to optimize heat transfer.

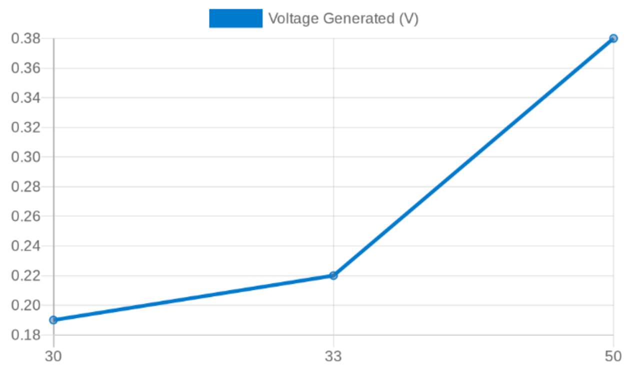


FIGURE 3. TEG Output Voltage vs. Temperature Difference Graph

The graph shows the relationship between the temperature difference across the thermoelectric generator modules and the corresponding voltage output measured experimentally. As observed, the generated voltage increases with a rise in the temperature difference. The curve demonstrates a generally proportional trend, where at a temperature difference of 30 °C the output voltage is approximately 0.19 V, increasing to about 0.38 V at a 50 °C differential. This indicates that maintaining a higher temperature gradient across the TEG surfaces significantly improves the voltage generation capability. All measurements were conducted under identical load conditions, with consistent heat sink placement and contact pressure, ensuring reliable comparison among data points. This behaviour validates that effective thermal management strategies, such as improving heat dissipation and maximizing the hot side temperature, directly impact the efficiency and feasibility of thermoelectric energy harvesting for powering embedded systems.

VII. CONCLUSION

In conclusion, the implementation of the self-navigating surveillance robot using TEGs to harvest energy and image processing approaches to detect threats provides a substantial step forward in autonomous security systems. The incorporation of AI models for fire and knife detection increases the robot's ability to spot potential threats in real-time and make monitored environments safer. The presented system's autonomous operation with continuous surveillance and maximization of threat alerts indicates that it is suitable for practical use in different sectors, including industrial, commercial, and residential security. Future work will focus on improving the robot's navigation algorithms and threat detection models to ensure reliability in diverse operating conditions.

APPENDIX

Appendix A – Threat Detection Models

The threat detection capability of the system is implemented through two distinct models developed using Roboflow. The fire detection model is trained to identify fire hazards in real time, using a diverse dataset of images capturing flames in various environments to maximize accuracy and reduce false detections. The knife detection model is designed to recognize knives and sharp objects under different orientations and lighting conditions, improving robustness and ensuring reliable identification of threats during live surveillance.

Appendix B – Navigation and Camera Systems

The navigation system uses autonomous control relying on ultrasonic sensors to detect and avoid obstacles in real time, enabling safe movement across diverse terrains. For visual monitoring and image acquisition, the system integrates the Raspberry Pi Camera Module v2 equipped with a Sony IMX219 8-megapixel CMOS sensor. This camera supports multiple video resolutions including 1080p30, 720p60, and VGA90, and connects to the Raspberry Pi via a 15 cm CSI ribbon cable for efficient data transfer and high-quality image capture

Appendix C – Processing and Power Specifications

The core processing is handled by the Raspberry Pi 4 Model B, a system-on-chip platform featuring a quad-core ARM Cortex-A72 64-bit processor clocked at 1.5 GHz and a minimum of 2 GB RAM to accommodate advanced computations and data processing. The power supply is provided by a 3.7 V lithium-ion battery pack, which can be recharged either through thermoelectric generators or conventional AC power sources, ensuring continuous operation and energy autonomy of the surveillance robot.

ACKNOWLEDGMENT

The authors express their gratitude to the Computer Science Department, Department of Electrical and Electronics Engineering, CAD Lab, and Physics Lab of RVCE for providing the necessary facilities for the construction and implementation of the system. The authors also thank the faculty of RVCE and external partners involved in the development of the self-navigating surveillance robot. Special thanks to the technical staff of RVCE for their help with implementing the TEG power system and integrating the image processing models. Their advice and resources were invaluable in the successful completion of this project.

REFERENCES

1) Win, S.L.Y.; Chiang, Y.-C.; Huang, T.-L.; Lai, C.-M. Thermoelectric Generator Applications in Buildings: A Review. Sustainability 2024, 16, 7585. https://doi.org/10.3390/su16177585

2) Ridwan,M.;Gasulla,M.; Reverter, F. Principle and Applications of Thermoelectric Generators: A Review. Sensors 2025, 25, 2484. https://doi.org/10.3390/s25082484

3) Guoneng Li Sijie Dong, Yiqi Fan, Qiangsheng Li , Wenwen Guo, Youqu Zheng , Yuanjun Tang, A review on micro combustion powered thermoelectric generator: History, state-of-the-art and challenges to commercialization, https://doi.org/10.1016/j.rser.2024.114897

4) Jaziri, N.; Boughamoura, A.; Müller, J.; Mezghani, B.; Tounsi, F.; Ismail, M. A comprehensive review of Thermoelectric Generators: Technologies and Common Applications. *Energy Reports* 2020, *6*, 264–287. https://doi.org/10.1016/j.egyr.2019.12.011

5) Debnath, R.; Bhowmik, M.K. A Comprehensive Survey on Computer Vision Based Concepts, Methodologies, Analysis and Applications for Automatic Gun/Knife Detection. *J. Vis. Commun. Image R.* 2021, *78*, 103165. https://doi.org/10.1016/j.jvcir.2021.103165

6) Sucuoglu, H. S. (2025). Development of Real-Time Fire Detection Robotic System with Hybrid-Cascade Machine Learning Detection Structure. Processes, 13(6), 1712. https://doi.org/10.3390/pr13061712

7) Lee, C.-H., Lee, W.-H., & Kim, S.-M. (2023). Development of IoT-Based Real-Time Fire Detection System Using Raspberry Pi and Fisheye Camera. Applied Sciences, 13(15), 8568. https://doi.org/10.3390/app13158568

8) Ding, N., Wang, X., Xian, X. et al. Photovoltaic, thermoelectric and electromagnetic generation technologies applied in power systems for mobile unmanned systems. Sci. China Technol. Sci. 66, 599–629 (2023). https://doi.org/10.1007/s11431-022-2159-8

9) Ahmed, S.M., Shiva, A.D.S., Gunjan, V.K. (2022). Domestic Smart Fire Fighting Robot with Multisensory Fire Detection and Warning System Using Python IDE. In: Gunjan, V.K., Zurada, J.M. (eds) Proceedings of the 2nd International Conference on Recent Trends in Machine Learning, IoT, Smart Cities and Applications. Lecture Notes in Networks and Systems, vol 237. Springer, Singapore. https://doi.org/10.1007/978-981-16-6407-6_5

10) Yanagisawa, R., Koike, S., Nawae, T., Tsujii, N., Wang, Y., Mori, T., ... & Nomura, M. (2023). Planar-type silicon thermoelectric generator with phononic nanostructures for 100 {\mu} W energy harvesting. arXiv preprint arXiv:2307.11382.

11) Tappura, K., Juntunen, T., Jaakkola, K., Ruoho, M., Tittonen, I., Ritasalo, R., & Pudas, M. (2020). Large-area implementation and critical evaluation of the material and fabrication aspects of a thin-film thermoelectric generator based on aluminum-doped zinc oxide. Renewable Energy, 147, 1292-1298.

12) Li, B., Guo, X., Li, T., & Li, Y. T. (2023). A thermoelectric generation system using waste heat

13) recovery from petrochemical pipeline to power wireless sensor. Energy Sources, Part A: Recovery, Utilization, and Environmental Effects, 45(3), 7694-7704.

14) Jadon, A., Omama, M., Varshney, A., Ansari, M. S., & Sharma, R. (2019). FireNet: a specialized lightweight fire & smoke detection model for real-time IoT applications. arXiv preprint arXiv:1905.11922.

SURABHI M received her Senior Secondary School Certificate (SSSC) from the Central Board for Secondary Education (CBSE) in 2024 and her Matriculation
Certificate or Secondary School Leaving Certificate (SSLC) from the Central Board for Secondary Education (CBSE) in 2022. She completed her secondary education at Sri Chaitanya Techno School, Electronic City, Bangalore, Karnataka and pursued her senior secondary education at Sri Chaitanya school, Electronic city Bangalore, Karnataka. Currently, she is pursuing her Bachelor of Engineering (B.E.) degree in Computer Science Engineering at R. V. College of Engineering, Bangalore, India.

KSHAMA BHAT K received her Senior Secondary School Certificate (SSSC) from the Central Board for Secondary Education (CBSE) in 2024 and her Secondary School Leaving Certificate (SSLC) from the Central Board for Secondary Education (CBSE) in 2022. She completed her secondary education at Chinmaya Vidyalaya, Badiadka, Kasaragod, Kerala and pursued her senior secondary education at Chinmaya Vidyalaya, Vidyanagar, Kasaragod, Kerala. Currently, she is pursuing her Bachelor of Engineering (B.E.) degree in Electronics and Communication Engineering at R. V. College of Engineering, Bangalore, India.

DISHITA REDDY GARUDADRI received her Senior Secondary School Certificate (SSSC) from the Central Board for Secondary Education (CBSE) in 2024 and her Matriculation Certificate or Secondary School Leaving Certificate (SSLC) from the Council for the Indian School Certificate Examinations (CISCE) in 2022. She completed her secondary education at Vibgyor High School, Haralur Road, Bangalore, Karnataka, and pursued her senior secondary education at Delhi Public School, Bangalore East, Karnataka. Currently, she is pursuing her Bachelor of Engineering (B.E.) degree in Computer Science Engineering at R. V. College of Engineering, Bangalore, India.

RESHMI M received her Senior Secondary School Certificate (SSSC) from the Central Board for Secondary Education (CBSE) in 2024 and her Secondary School Leaving Certificate (SSLC) from the Central Board for Secondary Education (CBSE) in 2022. She completed her secondary education at Sri taralabalu central school, Davanagere, Karnataka and pursued her senior secondary education at Sir M Visvesvaraya pu College, Davanagere, Karnataka. Currently, she is pursuing her Bachelor of Engineering (B.E.) degree in Biotechnology at R. V. College of Engineering, Bangalore, India.