Coursework 2 Final Report

Title:

Date: 12/05/2024

Word count: 2000 (excluding references and figures)

Summary

This project revolves around the creation and implementation of a fixed sensing device which uses various sensors to protect a room from intruders and alert the owner. This report contains:

- Background and motivation
- Related work
- Design
- Implementation
- Testing
- Critical Reflection

Background and motivation

My project will be a portable room protection device which will detect intruders breaking into the protected room alongside potentially inferring some information about the intrusion such as whether it was forceful or sneaky. In addition to protecting the room security wise, the system could also detect unwanted disasters such as a fire present within the room. In theory, this device would be able to notify the user of the events that have been detected within the protected room alongside some deterrent methods to alert the intruder that their presence has been detected.

There were 275,919 police reported burglary offences in England and Wales between 2022/2023 [1] with an average cost of each burglary estimated at £2,856 [2]. This highlights a rough cost of £788million (rounded) purely due to burglaries during 2022/2023 alone for England and Wales. Despite this showcasing a massive financial implication of theft, a much more important implication of this issue is the human emotion side. In a study investigating the human effect of burglaries, more than a quarter (27.7%) of people admitted that it took them between one to five months, while almost one in five (19.1%) people stated that it took or would take anywhere from six months to a year to feel reassured of their safety at home after being the victim of a burglary [2].

It is evident that the need for good security in the modern day is paramount and this is why I intend to contribute to the solution.

Related work

Mukhopadhyay, Anchal, and Kar (2018) explore how the use of sensors and machine learning algorithms can be utilised to detect the presence of a harmful intruder, with a high level of accuracy of 86% with 2 second signal length [3]. This paper uses a wireless sensor network system, which

differs from this report's wired sensor system, however, we can gain insights from this paper to assist with our development.

Aboelela and Khan (2013) implement a system which is designed to detect different types of intruders such as human or animal [4]. By utilising neural networks and sensors, this research yields excellent intrusion accuracy.

We can also apply the Benford et al (2005) "expected, sensed, and desired" [5] framework to this project. In terms of "expected", intruders are expected to move, emit sounds, change light patterns, and increase temperature within a room. For "sensed", using a range of sensors, we will attempt to sense the "expected" aspects of an intruder by making deductions from raw sensor data. We "desire" the ability to accurately detect intrusion even within areas of noise and disruption, such as rooms with busy hallways outside.

Design

The first physical interaction between the user and the device will be the 4-digit passcode set upon device activation, and the re-entered 4-digit passcode for deactivation, as seen in *Figure 1*. Then, the user will be asked to twist the rotary sensor to set the desired event detection sensitivity, as seen in *Figure 2*. Furthermore, a system to inform the user of events via text message (SMS) will be required, such as the design within *Figure 3*. This design trumps an application or an internet-based notification service, as a headless portable device will be tricky to connect to internet wherever you set it and present "seamful" aspects into the user experience. It may even be the case there isn't any Wi-Fi to connect to where the device is being set, and if there is, the device won't function properly if the connection is unreliable where it is crucial that events be delivered as soon as they are detected.

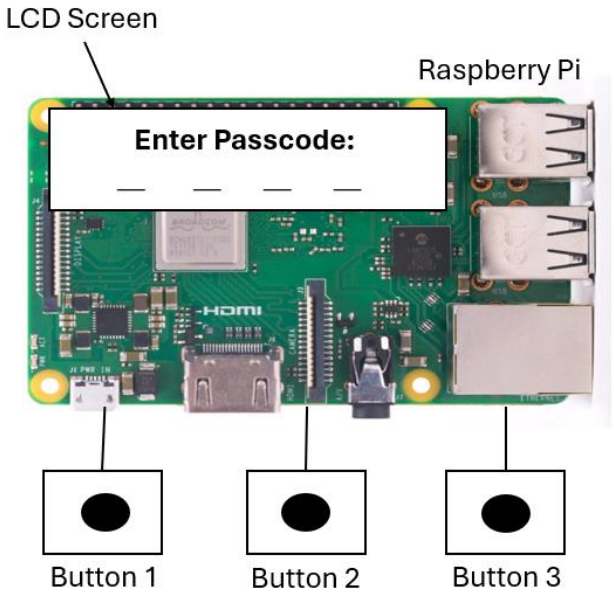

Figure 1: Designed physical interaction between user and device for activating and deactivating the security device, displayed on the GrovePi LCD screen.

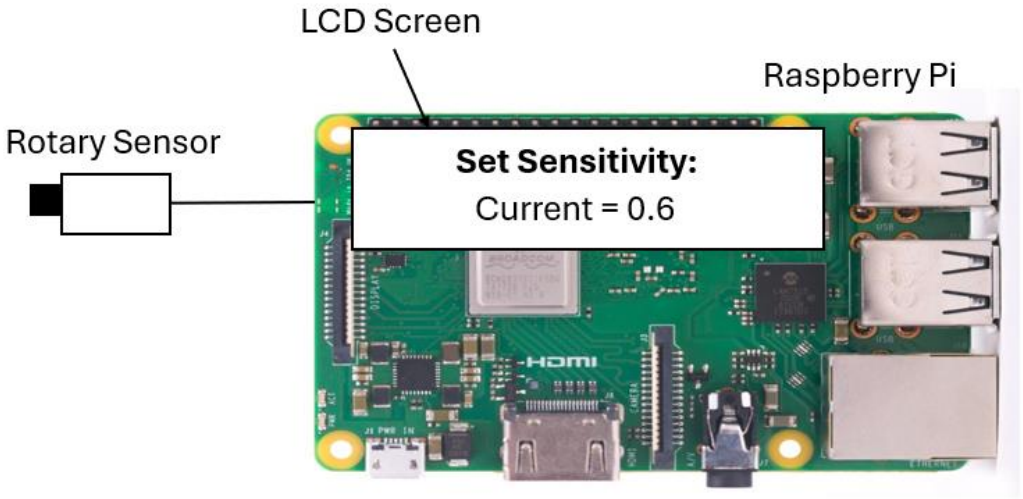

Figure 2: Designed physical interaction between user and device for setting the event detection sensitivity using the rotary sensor, displayed on the GrovePi LCD screen.

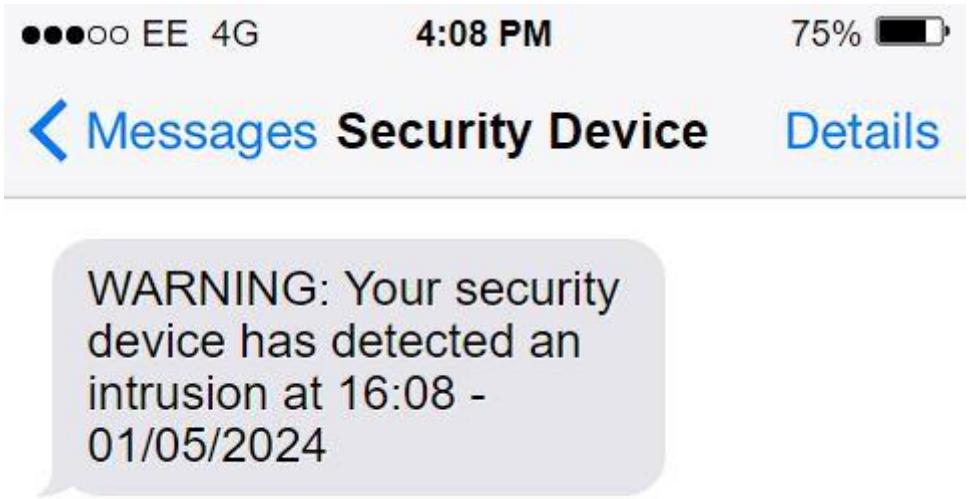

Figure 3: Designed interaction between user and device, where the system notifies the user of an intrusion via SMS.

Figure 4 shows the design of the device and the technologies used. As seen in the diagram, the device operates using a Raspberry Pi, and a GrovePi board, used to implement sensor usage. Three buttons are included on the design, each to implement the 4-digit passcode process which is used to activate and deactivate the device. The LCD screen assists with this process, visualising and guiding the user through. The rotary sensor allows the user to adjust the sensitivity of event triggers based on the environment the device is placed in. The PIR sensor, sound sensor, and light sensor are all used to infer what's happening in the device's surroundings and detect an event. For example, the PIR can be used to detect motion, the sound sensor can detect any unusual sounds in the room such as sudden sound spikes or a noisier environment, and the light sensor can detect whether a light has been turned on, a door has been opened, or if a burglar is using a flashlight to navigate around the dark room. *Figure 5* is a flowchart which depicts the designed flow of the system.

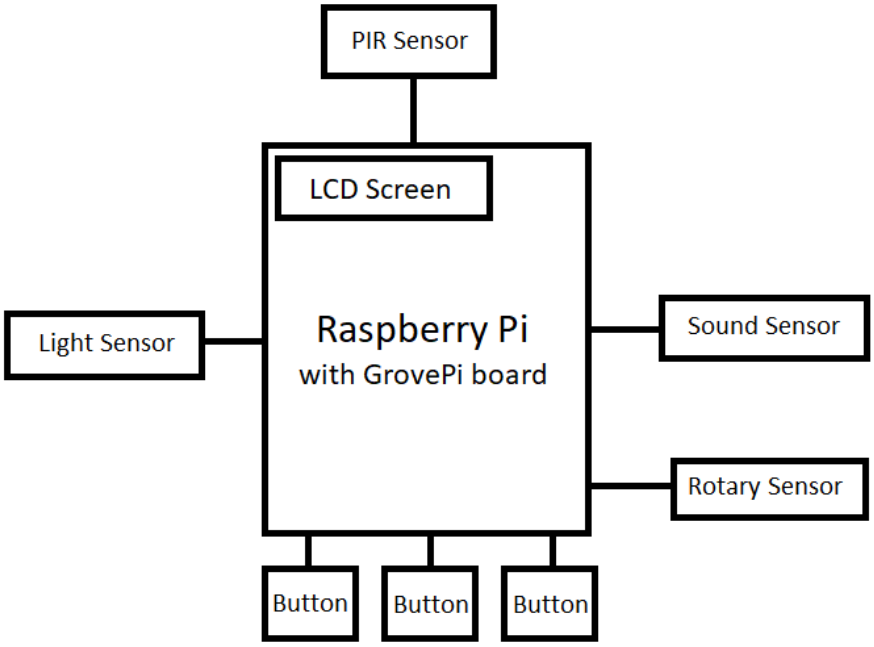

Figure 4: The design of the security device, including the use of technologies and physical connection to the Raspberry Pi.

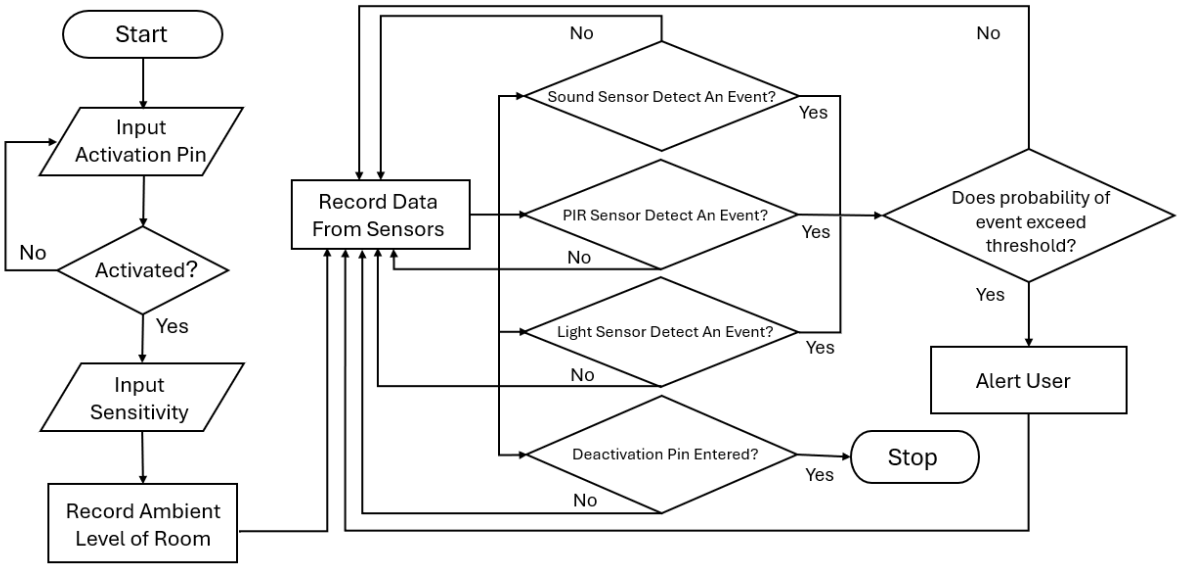

Figure 5: Flowchart for the security device system.

The sound and light sensor will use adaptive thresholds to allow for event detection. This allows us to collect the rooms ambient level so that anything out of the ordinary can be observed and acted upon.

I would also like to investigate that if one of the sensors breaks/malfunctions, the device can still operate.

Implementation

Figure 6 shows what the final prototype set up looks like. Looking back at the originally designed device in *Figure 4*, this final prototype matches it exactly, and design assumptions were correct.

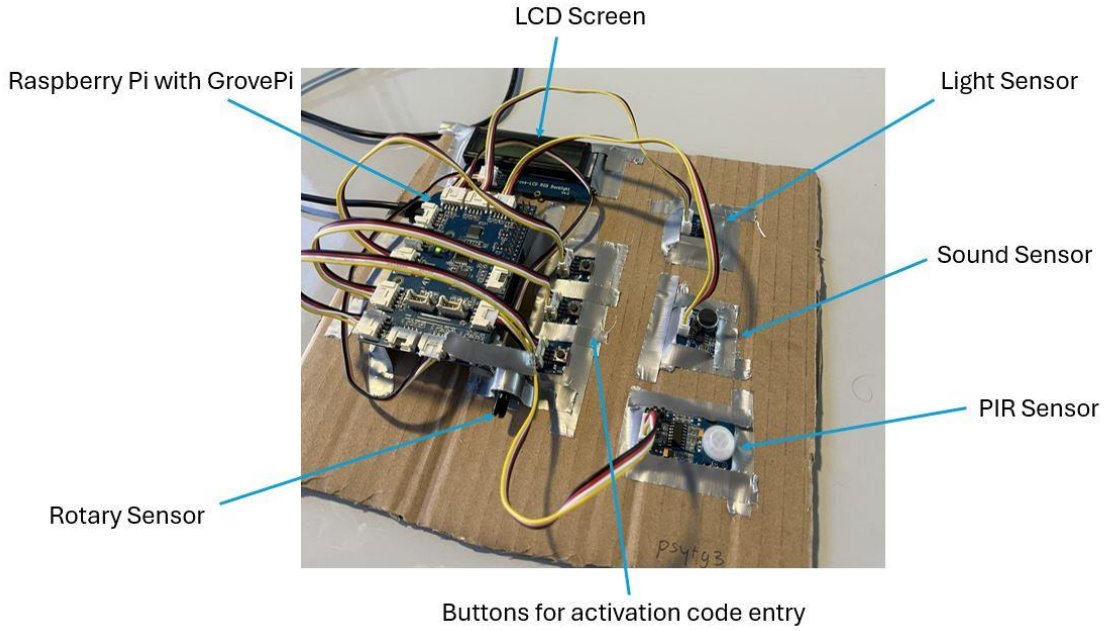

Figure 6: Labelled portable room security device.

In reference to the proposal, the idea of "detecting unwanted disasters such as fires" was scrapped, as this cannot be tested without burning down the CS building. Also, the idea of "potentially inferring some information about the intrusion such as whether it was forceful or sneaky" was scrapped to save space in the report.

When collecting data for testing, the pods in the CS building were used and various types of room entries were recorded with the device at different positions within the room. For example, *Figure 7* shows the first 10 rows of a silent entry, with the device furthest from the door. Six datasets were recorded. For each device location (near/far from door), 3 recording were taken based on noise level of entry. There was silent, normal, and loud.

	A	B	C
1	light	sound	pir
2	770	315	0
3	768	315	0
4	770	318	0
5	768	313	0
6	770	316	0
7	768	317	0
8	768	318	0
9	768	321	0
10	768	316	0

Figure 7: First 10 rows of silent entry data recording, with device furthest from the door.

The original plan was to utilise the rotary sensor to allow the user to set the event detection sensitivity themselves. However, after testing, it was discovered that there was an optimal configuration for 100% accuracy for all recorded datasets, so it was removed.

To implement the ambient level collection, upon device activation, the user will be asked to immediately leave the room. Sound/light levels exceeding the ambient level and threshold_trigger will trigger the sensors individual event variable. As seen in *Figure 5*, when one of these variables/sensors is triggered, the probability of an event will be calculated. The function to determine probability can be seen in *Figure 8.*

```
def calculate_probability(light, sound, pir):
    light = int(light)
    sound = int(sound)
    pir = int(pir)
    probability = (light * weights["light"]) + (sound * weights["sound"]) + (pir * weights["pir"])
    return probability
```

Figure 8: Function to calculate probability of an event occurring.

The SMS feature discussed in the interaction design was scrapped, simply since it was too complicated for this project and would require purchases. Instead, where an SMS or notification would be sent, a print statement outputs “EVENT DETECTED” to the device’s terminal, and the device’s LCD screen turns from green to red to implement “deterrent methods to alert the intruder that their presence has been detected” as discussed in the proposal.

Testing

The first draft of the algorithm was tested with the initially chosen weighting seen in *Figure 9* with various levels of set sensitivity threshold (from the rotary) with the results seen in *Figure 10*.

```
# Probability weights
weights = {"light": 0.3, "sound":0.3, "pir":0.4}
```

Figure 9: Probability weighting used to calculate event probability using the function seen in Figure 8.

Sensitivity Threshold	True Positives	False Positives	False Negatives	True Negatives	Truth Accuracy	Negatives Accuracy	Overall Accuracy
0.1	104	14	0	545	0.881355932	1	0.940677966
0.2	104	14	0	545	0.881355932	1	0.940677966
0.3	104	14	0	545	0.881355932	1	0.940677966
0.4	104	0	0	559	1	1	1
0.5	22	0	82	559	1	0.872074883	0.936037441
0.6	22	0	82	559	1	0.872074883	0.936037441
0.7	22	0	82	559	1	0.872074883	0.936037441
0.8	0	0	104	559	n/a	0.843137255	n/a
0.9	0	0	104	559	n/a	0.843137255	n/a

Figure 10: Table showing accuracy for each sensitivity threshold using the probability weighting in Figure 9.

The truth and negative accuracy columns were calculated using the following equation:

$$Accuracy = \frac{Number\ of\ Correct\ Predictions}{Total\ Number\ of\ Predictions}$$

Overall accuracy was calculated by weighting each accuracy using the following equation:

$$Overall\ Accuracy = (0.5 * True\ Accuracy) + (0.5 * Negative\ Accuracy)$$

The best accuracy we obtained for the first testing was 1.0 using the data in *Figure 9* and *Figure 10*, which is great. We see that each other sensitivity in the table introduces false positives or false negatives, which we'd like to avoid. Therefore, for our silent entry, we have found our optimal settings right away. After testing each type of entry (silent/normal/loud) for each of my datasets, I have found the best configuration for optimal accuracy with no false positives or negatives as seen in *Figure 11*. All the recorded datasets happen to have the same optimal configuration for 100% accuracy.

Furthest From Door					
Entry Type	Optimal Light Weight	Optimal Sound Weight	Optimal PIR Weight	Optimal Sensitivty	Accuracy
Silent	0.3	0.3	0.4	0.4	1
Normal	0.3	0.3	0.4	0.4	1
Loud	0.3	0.3	0.4	0.4	1
Next To Door					
Entry Type	Optimal Light Weight	Optimal Sound Weight	Optimal PIR Weight	Optimal Sensitivty	Accuracy
Silent	0.3	0.3	0.4	0.4	1
Normal	0.3	0.3	0.4	0.4	1
Loud	0.3	0.3	0.4	0.4	1

Figure 11: Table showing optimal configurations for each type of entry.

For each dataset, the entry occurred at 362 iterations (18.1 seconds), and each iteration occurred every 0.05 seconds. Using the following formula, we can work out the detection delay for each dataset:

$$Total\ Time = time.sleep(x) * Number\ of\ Iterations$$

The detection delay for each dataset can be seen in *Figure 12* and is visualised in Figure 13, where we can see an overall downwards curve in delay as the device is placed further away from the entry point and the entry noise volume increases.

Furthest From Door			
Entry Type	Entry Time (s)	Event Detection Time (s)	Delay (s)
Silent	18.1	27.95	9.85
Normal	18.1	22.35	4.25
Loud	18.1	21.75	3.65
Next To Door			
Entry Type	Entry Time (s)	Event Detection Time (s)	Delay (s)
Silent	18.1	22	3.9
Normal	18.1	19.3	1.2
Loud	18.1	19.4	1.3

Figure 12: Table showing event detection latency for each dataset.

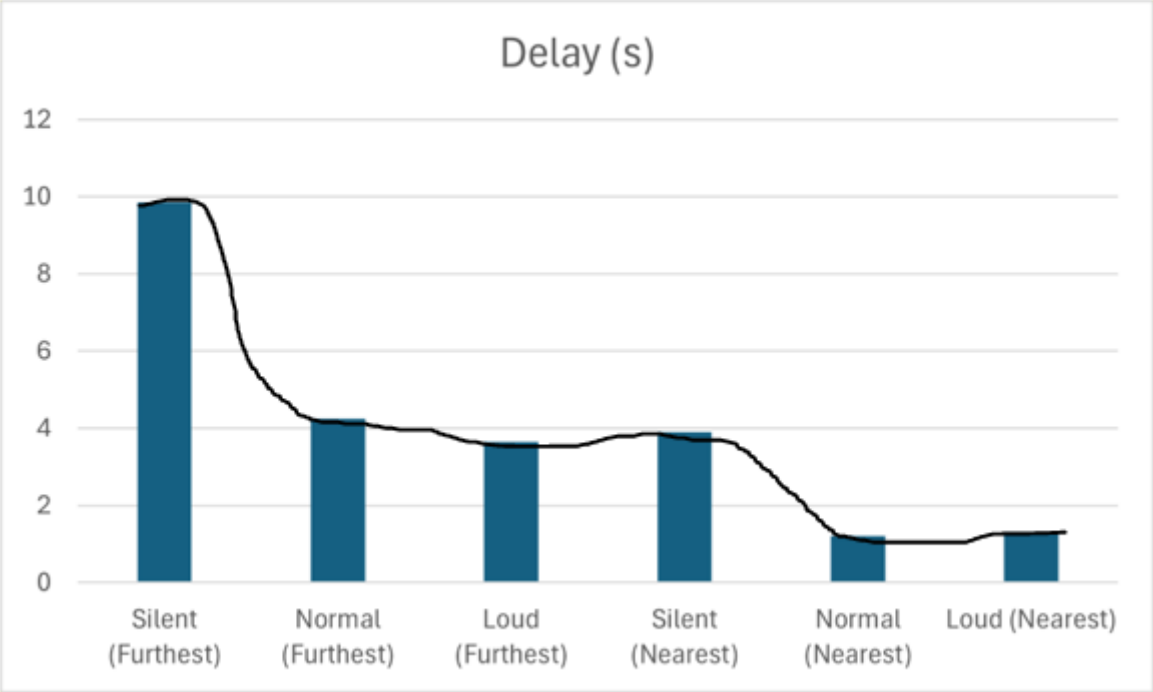

Figure 13: Graph showing delays for each dataset.

This data proves the device is working as expected. The closer the device is to the entry point, and/or the louder the entry, the faster the event will be detected. This data shows us that having the device nearer to the entry point is optimal for performance.

After carrying out final testing in the CS kitchen, mixed reality lab, and C60, the value of 30 for threshold_trigger was best suited overall, particularly in environments with external noise. This value picked up all intrusion attempts, whilst ignoring hallway noise and eliminating resulting false positives. Furthermore, this final testing proved that when one sensor broke/malfunctioned, the device was still operational, as seen in *Figure 14*. However, when the PIR is broken, the light sensor must be triggered, so the device is best kept in a dark room to maximise these chances.

Light	Sound	PIR	Outcome
broken	functional	functional	device works properly
functional	broken	functional	device works properly
functional	functional	broken	device works properly

Figure 14: Testing of device when one sensor is broken.

Critical reflection

One major flaw with this project is the lack of varied recorded data from different rooms. When developing the algorithm, this device had been exclusively tested on the CS building pods and might as well be called a “portable CS pod protection device”. I couldn’t collect much data as for one lab none of the Raspberry Pis were working for me, and for the next one I was ill. However, after easter, during final testing, I got the opportunity to test the device in various other locations and expand its range of accurate protection.

When comparing the device with literature, we see that our accuracy is far better than the accuracy obtained in the study by Mukhopadhyay, Anchal, and Kar (2018). Their highest accuracy obtained was 86%, ours was 100%. However, they were detecting intruders using seismic sensors, which is very different to what was used for this study, so is perhaps incomparable. One issue in terms of literature is that due to the device designed for this paper being so niche, there isn't any literature to compare it with. Therefore, it is being compared with intrusions detection systems using sensors in much different contexts than are relevant to our own, making it harder to make accurate comparisons in terms of performance.

Ethically, there are a few areas which may have implications. Analysing these implications through Greenfield's principles, initially we see an issue with the device defaulting to harmlessness. If the device fails, it needs to fail gracefully, and not send the user false intrusions warnings which may affect the user's psychological safety and potentially cause major disruptions to their day. In addition to this, the device violates both the "be self-disclosing" and "be deniable" principles. However, for this device to function for its intended purpose, it needs to violate these principles as it doesn't want to allow the intruder to query ownership, how it works, and its capabilities, or even worse, allow the intruder to just "opt out" and steal whatever he wants. There's also the issue of the device being misused, such as unauthorised surveillance, stalking, or harassment. This would violate the principle of "be conservative of face", as we want this device to protect society and not allow the device to be used for harm, humiliation, and general anti-social behaviour.

References

1. Crime in England and Wales (2024) Crime in England and Wales: Appendix tables, table A4 Office for National Statistics. Available at: https://www.ons.gov.uk/peoplepopulationandcommunity/crimeandjustice/datasets/crimeinenglandandwalesappendixtables (Accessed: 01 May 2024)

2. The human effect of a burglary (2023) ADT. Available at: https://www.adt.co.uk/blog/the-human-effect-of-a-burglary (Accessed: 01 May 2024)

3. B. Mukhopadhyay, S. Anchal and S. Kar, "Detection of an Intruder and Prediction of His State of Motion by Using Seismic Sensor," in IEEE Sensors Journal, vol. 18, no. 2, pp. 703-712, 15 Jan.15, 2018, doi: 10.1109/JSEN.2017.2776127.

4. E. H. Aboelela and A. H. Khan, "Wireless sensors and neural networks for intruders' detection and classification," The International Conference on Information Network 2012, Bali, Indonesia, 2012, pp. 138-143, doi: 10.1109/ICOIN.2012.6164365.

5. Benford, S., Schnädelbach, H., Koleva, B., Anastasi, R., Greenhalgh, C., Rodden, T., Green, J., Ghali, A., Pridmore, T., Gaver, B., Boucher, A., Walker, B., Pennington, S., Schmidt, A., Gellersen, H., Steed, A., 2005. Expected, sensed, and desired: A framework for designing sensing-based interaction. ACM Trans. Comput.-Hum. Interact. 12, 3–30. https://doi.org/10.1145/1057237.1057239

Appendix A – instructions

[Put instructions related to running / operating your coursework's code etc. here]

1. Power device on.
2. Connect to the device via SSH.
3. Copy the device_code.py file on to the device over SSH.
4. Run the code with “python device_code.py”
5. Enter activation code using the buttons and the LCD screen on the device.
6. Leave the room as shown on the LCD screen and lock doors within 30 seconds (can be reconfigured in code file).
7. Device will then calculate the ambient levels of the room.
8. Once ambient levels are collected, any intrusions will be detected and printed to the terminal.
9. To deactivate, walk up to the device and re-enter the activation code, and the program will end.

[Appendix B – NOTE: You need to submit your coursework as zip file to Moodle – including this report and source code, test data files etc.]