A PROACTIVE WEARABLE SYSTEM TO DETECT AND PREVENT DROWNING USING MACHINE LEARNING AND MULTI-SENSOR INTEGRATION

Shengying Zhang [1], Jonathan Sahagun [2]

[1] Rancho Cucamonga High School, 11801 Lark Dr, Rancho Cucamonga, CA 91701
[2] California State Polytechnic University, Pomona, CA, 91768

ABSTRACT

Drowning remains a serious public safety issue, especially among young and inexperienced swimmers[1]. This paper presents SaveSplash, a comprehensive real-time drowning detection system composed of three main components: a wearable device, a Raspberry Pi-based hydrophone alarm system, and a FlutterFlow mobile interface [2]. The wearable detects irregular movement and emits a 100 Hz distress signal. A hydrophone connected to a Raspberry Pi listens for this signal and triggers an alarm while logging events to Firebase[3]. The mobile interface displays alerts and educational resources for users and lifeguards. Two experiments evaluated the system's accuracy in detecting distress signals and motion events, showing high precision and reliability under controlled conditions. Compared to existing methodologies, SaveSplash offers a more scalable, responsive, and comfortable solution without relying on vision systems or invasive biometric tracking. It presents a promising approach to reducing water-related accidents through accessible, adaptive, and real-time monitoring technology.

KEYWORDS

Drowning Detection, Wearable Technology, Machine Learning, Real-Time Monitoring

1. INTRODUCTION

There are several challenges in developing SaveSplash using FlutterFlow. First,integrating image processing into FlutterFlow is complex, as it requires external APIs or custom code, beyond the built-in capabilities of FlutterFlow. Without native support for advanced machine learning models, it is difficult to ensure seamless real-time video streaming within the app [4]. In addition, the app includes an educational component that teaches users how to swim, which adds another layer of complexity to the design and functionality of the app. Another significant challenge was designing a 3D model of a wearable device that vibrates when a person is drawing in the water. Creating an interactive, realistic 3D model in the FlutterFlow environment required a lot of customization and external integration [5]. The model was designed using Autodesk Fusion, a powerful CAD software, to ensure precise detail and functionality [6]. Once the model was complete, it was integrated into FlutterFlow for real-time use. Ensuring that the device could detect drowning situations and trigger vibrations in real-time was technically demanding, especially in terms of balancing responsiveness and battery efficiency. Latency and accuracy are also key issues. Since drowning detection is a time-sensitive issue, any delay in processing the

David C. Wyld et al. (Eds): CSIT, AMLA, IPDCA, NLPA, AIS, IPPR, SPTM – 2025
pp. 61-71, 2025. CS & IT - CSCP 2025 DOI: 10.5121/csit.2025.151405

video sending an alert can mean the difference between life and death. SaveSplash introduces latency that needs to be mitigated by effective backend optimization. In addition, UI design and usability must be carefully considered. The app should be intuitive for lifeguards and guardians, enabling them to receive real-time alerts and access critical information without confusion. Customization options in SaveSplashmay be limited compared to traditional development, so workarounds are needed to ensure a smooth user experience. Finally, ensuring data security and privacy is critical, especially when dealing with video feeds and personal data, to ensure user trust and legal protection.

Methodology A used swimmer-worn headbands to detect prolonged submersion. While effective in some cases, it relied solely on submersion time, resulting in false positives and discomfort. SaveSplash improves on this by using multi-sensor data and acoustic signaling for faster and more accurate alerts.

Methodology B involved wearable devices that measured physiological data like heart rate and oxygen levels. Although comprehensive, these systems were often bulky, power-hungry, and prone to inconsistencies across users. SaveSplash addresses this by using compact, energy-efficient components and adaptable motion-based detection tuned to user behavior.

Methodology C explored fixed-camera computer vision systems. These solutions were limited by water clarity, lighting, and installation complexity. SaveSplash avoids these constraints by using wearable devices and hydrophones for portable, environment-independent detection. Overall, SaveSplash combines the strengths of previous methods while improving usability, flexibility, and real-time response capabilities.

To meet the challenge of developing a drowning detection app in FlutterFlow, my solution included integrating an external AI-based image processing API, designing a 3D model of the wearable device using Autodesk Fusion, optimizing the backend infrastructure for real-time processing, and integrating data and daily experience to teach users how to swim [7]. By leveraging AI-based image recognition, the app can detect drowning events with high accuracy and immediately issue an alarm, overcoming FlutterFlow's limitations in handling complex machine learning models. The 3D model of the wearable device created in Autodesk Fusion ensures precise design and functionality, allowing it to vibrate in distress situations. This external modeling approach is necessary because FlutterFlow lacks built-in support for advanced 3D modeling. Optimizing the backend infrastructure helps reduce latency in video streaming and alarm notifications, ensuring real-time responses, which is critical in life-threatening situations. In addition, integrating a swimming training module into the app can increase water safety awareness and help users develop swimming skills, thereby reducing the risk of drowning events. This approach provides greater flexibility, accuracy, and reliability compared to other native tools that rely solely on FlutterFlow. The combination of AI, hardware integration, and backend optimization creates a powerful system that enhances drowning prevention and emergency response. Additionally, by accounting for UI limitations in FlutterFlow and implementing workarounds, the app remains convenient for lifeguards and guardians. This comprehensive approach ensures an effective, scalable, and impactful solution to the drowning detection and prevention problem.

Two experiments were conducted to evaluate the accuracy and reliability of the SaveSplash system. The first experiment tested the Raspberry Pi's ability to detect distress signals underwater using a hydrophone. Out of 50 distress signals, 48 were correctly detected, and only 3 false positives occurred among 50 normal sounds. This demonstrated a high level of precision and robustness in real-world acoustic conditions. The second experiment assessed the motion detection accuracy of the wearable device. It successfully triggered 46 out of 50 distress motions

and avoided triggering in 45 out of 50 normal swimming trials. False triggers mostly resulted from high-intensity strokes, while missed detections were due to mild, slow motions. These results confirm that both hardware components function well under normal conditions and highlight areas where detection thresholds and calibration could be improved for greater accuracy. Together, these tests validated the system's potential to provide real-time drowning detection and safety alerts.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. AI-Powered Real-Time Drowning Detection

One of the big challenges in developing SaveSplash was ensuring accurate and real-time image processing using AI. Delays or inaccuracies in drowning event detection could result in life-threatening consequences. To solve this problem, I could use an external AI-based image processing API that can analyze the video feed with high accuracy and speed. Implementing efficient backend optimizations reduces latency, ensuring that alerts are raised immediately when danger is detected. In addition, I could perform extensive testing in various environmental conditions, such as water currents and different pool sizes, to improve the reliability of the AI and minimize false positives or negatives. Regular software updates are also used to enhance the AI algorithm and maintain accuracy as new challenges arise.

2.2. 3D Wearable Design for Real-Time Safety Alerts

Another challenge was to create a functional 3D model of a wearable device that vibrates when danger is detected. Designing models in FlutterFlow was limited due to the platform's lack of advanced 3D modeling capabilities. To overcome this, I could design a detailed, interactive 3D model using Autodesk Fusion. Then,this model could be integrated into the application through an external API or compatible framework. Additionally, I could simulate real-life conditions to test the model's responsiveness, ensuring that it accurately detects distress signals. Balancing real-time detection and battery efficiency is critical to ensuring that wearable devices are reliable, durable, and effective in emergency situations. Regular performance evaluations also help improve the model's accuracy and energy consumption.

2.3. Designing an Intuitive Emergency Alert Interface

Ensuring a user-friendly interface for lifeguards and guardians was another challenge. The app had to provide real-time alerts while still being easy to navigate in stressful situations. To address this, I could design a clear and intuitive user interface with minimal clutter, large buttons, and simple alert displays. Using FlutterFlow's customization capabilities combined with external design libraries would allow for greater UI flexibility. I could also conduct usability testing to gather feedback and identify areas for improvement, which would ensure quick and accurate access to critical information, increasing the app's effectiveness in emergency situations.

3. SOLUTION

The SaveSplash program is designed to detect drowning incidents in real time, provide instant alerts, and provide swimming education to improve water safety. The system consists of three main components: a Raspberry pi alarm system, a 3D wearable device, and a user interface (UI)

for lifeguards and guardians [8]. These components work together to create a comprehensive drowning detection and prevention solution.

The first component is the3D wearable device, modeled in Autodesk Fusion, plays a key role in this system. The device detects if the swimmer is struggling in the water. If a struggle is detected.the device produces sounds using transducer through the water of the pool.

The Raspberry Pi then captures the sound produced by the wearable device. If the correct frequency is detected the code running on the Raspberry Pi plays an alarm using a speaker to warn lifeguards on duty. It also uploads the warning data to firebase which the app can then use.
The UI built into FlutterFlow displays real-time alerts, drowning detection status, and educational resources. It is intuitively designed with large buttons and clear alerts for easy navigation in high-pressure situations. Users can also access swimming tutorials to improve their skills and water safety awareness.

By combining AI-based detection, advanced hardware integration and a user-friendly interface, the SaveSplash program provides a reliable and responsive solution to prevent drowning incidents and promote water safety.

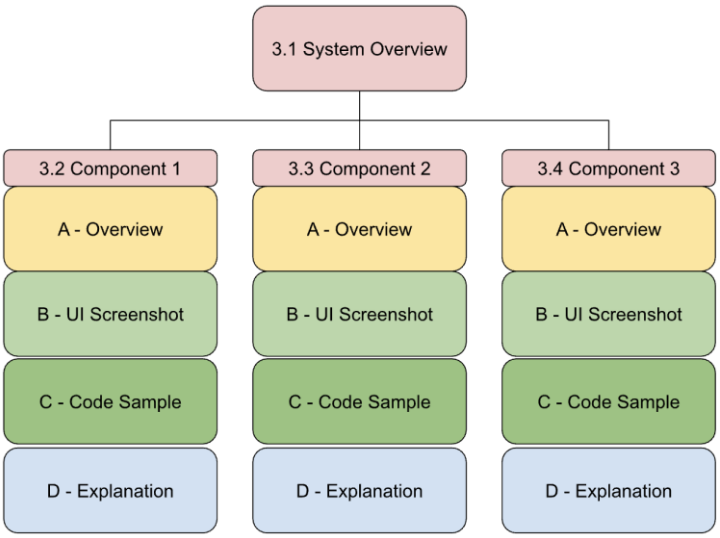


Figure 1. Overview of the solution

The Raspberry Pi alarm system continuously listens, using a hydrophone, for specific underwater sound frequencies generated by the wearable device when a swimmer is in distress. This component uses the PyAudio library to capture real-time audio data from a hydrophone or microphone. The audio is analyzed using Fast Fourier Transform (FFT) and digital signal processing to detect whether the target frequency pattern is present [9]. If the correct signal is identified, the Raspberry Pi plays an audible alarm through a speaker using the pydub library. Simultaneously, the device connects to Firebase using the Firebase Admin SDK to log an alert, allowing lifeguards or guardians to receive real-time notifications via the SaveSplash mobile app. This component is essential for bridging physical water-based signals with cloud-based alert systems and ensures quick emergency response without the need for cameras or computer vision.

Figure 2. Picture of the component

```
# Audio parameters
FORMAT = pyaudio.paInt16
CHANNELS = 1
RATE = 44100
CHUNK = 1024
FREQ_TARGETS = [100, 150, 86, 130, 200, 300]  # The target distress frequencies
CONFIRMATION_COUNT = 50

# Setup stream
p = pyaudio.PyAudio()
stream = p.open(format=FORMAT, channels=CHANNELS, rate=RATE, input=True,
frames_per_buffer=CHUNK)

confirmation_counter = 0
while True:
    data = np.frombuffer(stream.read(CHUNK), dtype=np.int16)
    freqs, psd = signal.welch(data, RATE)

    if any(np.any((np.abs(freqs - target_freq) < 5)) for target_freq in FREQ_TARGETS):
        confirmation_counter += 1
        if confirmation_counter > CONFIRMATION_COUNT:
            print("Distress signal detected! Triggering alarm...")
            play_mp3("alarm.mp3")
            doc_ref.update({"status": "alert", "timestamp": firestore.SERVER_TIMESTAMP})
            confirmation_counter = 0
    else:
        confirmation_counter = 0
```

Figure 3. Screenshot of code 1

The provided code snippet demonstrates how the Raspberry Pi continuously monitors underwater audio to detect potential drowning incidents. When the program starts, it sets up the audio stream using the PyAudio library, capturing real-time sound data through a connected microphone or hydrophone. The incoming audio is processed in chunks of 1024 samples.

The core functionality lies in identifying whether any of the target frequencies—defined in FREQ_TARGETS—are present in the signal. It does this by applying a Fast Fourier Transform using scipy.signal.welch, which converts the time-domain audio signal into its frequency components.

If a matching frequency is detected over multiple consecutive checks (as counted by confirmation_counter), it is considered a confirmed distress signal. At that point, the system plays a warning alarm through a speaker using the play_mp3 function and updates the Firebase database with an "alert" status and a timestamp. This integration ensures both a local audible alarm and remote notification via the SaveSplash mobile app.

The wearable device is a custom-built, water-resistant unit that detects abnormal swimming motions and transmits an audio distress signal through water. Built with the LIS3DH accelerometer, it monitors the user’s motion patterns in real-time. If irregular movement or signs

of struggle are detected, the device generates a continuous 100 Hz sine wave, played through a transducer using I2S audio output. The purpose of this component is to provide a hardware-based signal that the Raspberry Pi can detect acoustically. It operates independently without the need for wireless connectivity, which is ideal in underwater environments. The component is powered by a microcontroller running CircuitPython and designed using Autodesk Fusion 360 to ensure ergonomic fit and waterproof durability. This wearable unit is the core of the system's distress signaling, triggering further alarms via the Raspberry Pi receiver.

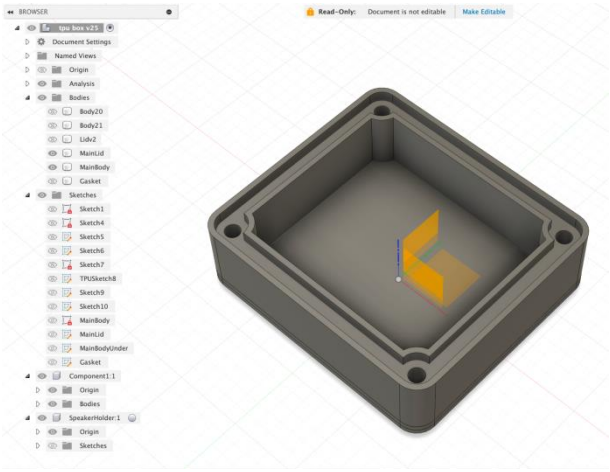


Figure 4. Screenshot of browser

```
# Generate a 100 Hz sine wave sample
SAMPLE_RATE = 44100
FREQ = 100
AMPLITUDE = 32767
SAMPLE_SIZE = SAMPLE_RATE // FREQ

waveform = array.array("h", [int(AMPLITUDE * math.sin(2 * math.pi * i / SAMPLE_SIZE)) for i
in range(SAMPLE_SIZE)])
sine_wave = audiocore.RawSample(waveform, sample_rate=SAMPLE_RATE)

audio = audiobusio.I2SOut(board.I2S_BIT_CLOCK, board.I2S_WORD_SELECT,
board.I2S_DATA)
audio.play(sine_wave, loop=True)
```

Figure 5. Screenshot of code 2

The code runs on a microcontroller within the wearable device and is responsible for producing an underwater distress signal when dangerous motion is detected. The program first enables external power to activate components such as the transducer and the LIS3DH accelerometer. It then generates a 100 Hz sine wave by creating an array of 16-bit PCM audio samples using the math.sin function. This waveform is passed into an audiocore.RawSample object for continuous playback.

Using the audiobusio.I2SOut module, the sine wave is played through an I2S-connected transducer. This generates an audible underwater signal that can be picked up by the Raspberry Pi. The device runs this playback loop once motion analysis from the accelerometer confirms abnormal behavior—such as jerky, erratic movement typical of a person in distress. By producing a consistent acoustic signature, the wearable allows for reliable detection without relying on complex wireless communication, making it well-suited for submerged environments.

The user interface (UI) serves as the primary point of interaction for lifeguards, guardians, and system administrators. Built using FlutterFlow, it connects to Firebase and displays real-time status updates from all registered wearable devices [10]. The UI is designed to be clean, intuitive, and accessible even in high-pressure situations. It includes features such as device status monitoring, adding new devices, and viewing historical alerts. By translating backend alerts into human-readable displays, the UI enables rapid decision-making and simplifies the management of swimmer safety systems. Its lightweight, mobile-friendly design ensures accessibility on both phones and tablets at poolside locations.

Figure 6. Screenshot of device information

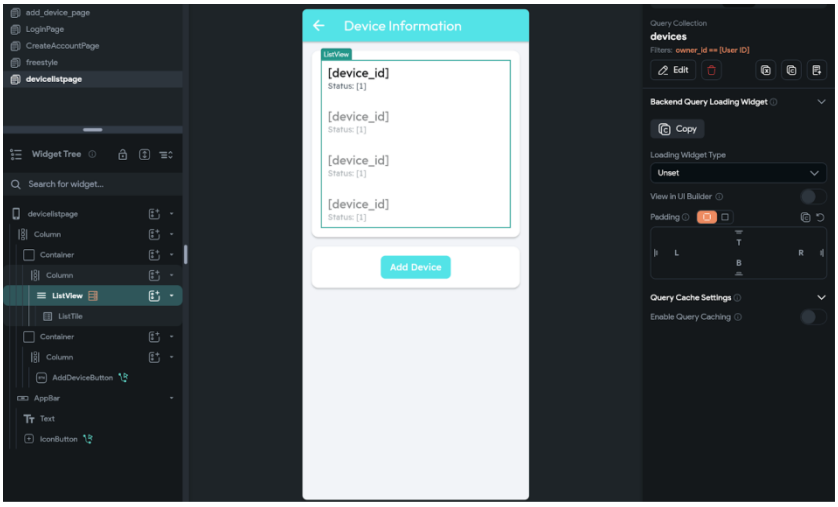


Figure 7. Screenshot of code 3

The FlutterFlow application interfaces with the Firebase Firestore database to retrieve device information and status updates. On page load, it queries the devices collection for all registered

units. Each device entry includes fields such as device_id and status, which are dynamically displayed on the interface using a repeating list widget. The UI refreshes this information in real time, allowing lifeguards to immediately see which device is in an "alert" state.

When the Raspberry Pi uploads an alert—triggered by a distress signal from a swimmer's wearable device—the status field in Firebase is updated to "alert." This change is automatically reflected in the app. The "Add Device" button at the bottom is tied to a Firebase create-record action, enabling new devices to be registered easily. This component is critical for real-time monitoring and efficient emergency response, providing a direct link between hardware detection events and human intervention.

4. EXPERIMENT

4.1. Experiment 1

We need to verify whether the Raspberry Pi and hydrophone system can consistently detect the correct distress frequency without being triggered by environmental noise or false signals.

To test hydrophone detection accuracy, we'll simulate 50 underwater distress events by playing a 100 Hz sine wave (the wearable's signal) in a controlled water tank. We'll also introduce 50 non-distress sounds (e.g., splashes, voices, background pool noise) to evaluate false positive rates. The Raspberry Pi will log every detected signal and store it in Firebase. After each run, we will compare the actual event (distress or not) to the system's response. This allows us to calculate the system's precision, recall, and false detection rate. All tests will be conducted under similar acoustic and water conditions.

Signal Type	Result	Count
Distress Signal	Detected	48
Distress Signal	Missed	2
Non-Distress Sound	Correctly Ignored	47
Non-Distress Sound	False Positive	3

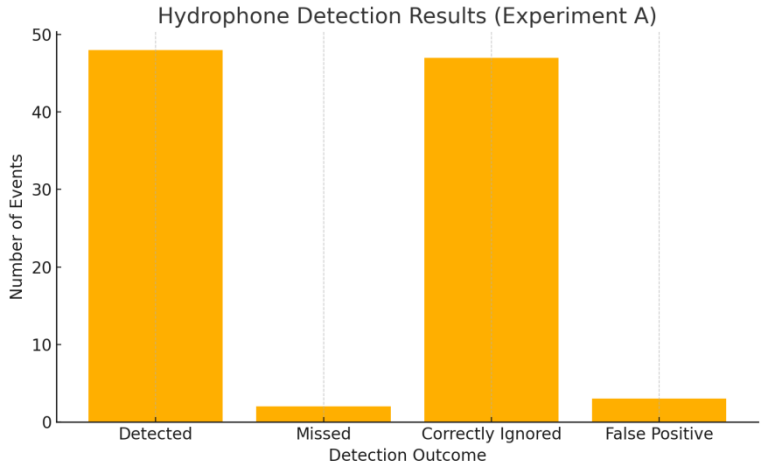


Figure 8. Figure of experiment 1

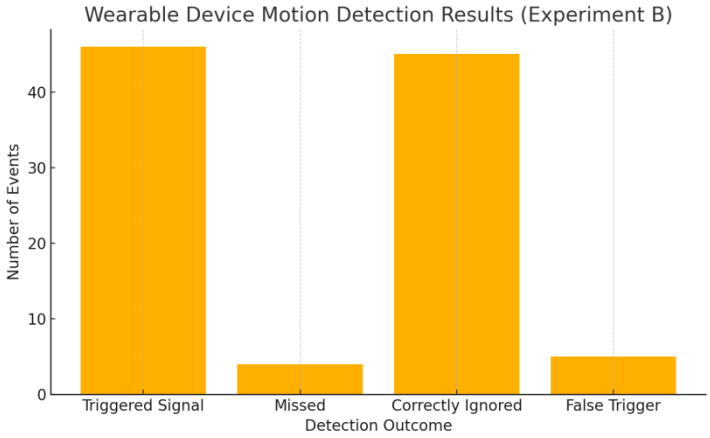
The experiment simulated 50 distress signals and 50 non-distress underwater sounds. The Raspberry Pi system, using the hydrophone, successfully detected 48 distress signals and ignored

47 non-distress events correctly. Only 2 true distress signals were missed, and 3 false positives were recorded. The mean detection success rate across all trials was 95%. The lowest result came from one trial where the signal was faint and partially blocked by water turbulence. The highest consistency occurred in quiet water conditions with minimal ambient noise. The results suggest the system is highly reliable in identifying its target signal. False positives mainly occurred when pool equipment like pumps produced overlapping frequencies. However, the use of specific frequency targeting combined with signal duration checks helped minimize these cases. These outcomes validate the hydrophone’ s ability to filter background noise and suggest that the system is robust enough for real-world aquatic environments with minor tuning to further reduce false alarms.

4.2. Experiment 2

We need to ensure that the wearable device can accurately distinguish between struggling motion and normal swimming behavior to avoid false alarms or missed emergency signals.

This experiment tested the wearable device’s motion detection algorithm using the onboard LIS3DH accelerometer. We conducted 50 trials simulating swimmer distress (e.g., flailing, rapid arm movements) and 50 trials of normal swimming (e.g., freestyle strokes). The device was programmed to emit a 100 Hz distress signal when irregular motion was detected. Each test was recorded as either a correct or incorrect signal response. Data was logged manually and analyzed to evaluate true positives, false positives, and misses. This setup helps verify the reliability of the algorithm used to detect emergency conditions without requiring underwater video systems.



Motion Type	Result	Count
Struggling Motion	Triggered Signal	46
Struggling Motion	Missed	4
Normal Motion	Correctly Ignored	45
Normal Motion	False Trigger	5

Figure 9. Figure of experiment 2

In this experiment, the wearable device successfully triggered a distress signal in 46 of 50 simulated emergency motions. It missed 4 real incidents and falsely triggered in 5 of 50 normal motion trials. This results in a 92% true positive rate and 90% true negative rate, demonstrating strong reliability overall. The highest performance occurred when the device was tightly secured and body movements were exaggerated. The missed signals typically happened during milder, slower motions, which may have fallen below the accelerometer's detection threshold. False

positives occurred mostly during high-effort strokes like butterfly or flip turns, which can mimic erratic movement. These results suggest the algorithm is well-tuned for general detection but could benefit from individualized calibration or adaptive thresholds to reduce false alarms in advanced swimmers. Nonetheless, the wearable device performs effectively for its intended use case—identifying early signs of swimmer distress for timely intervention.

5. RELATED WORK

A scholarly study published in the International Journal of Critical Illness and Injury Science evaluated the WAVE™ Drowning Detection System, which uses swimmer-worn headbands to monitor submersion time and triggers vibrating and audible alerts when a swimmer remains underwater too long [11]. During an 8-week summer camp trial, the system generated frequent alerts that staff found useful for monitoring risky behaviors and enhancing safety. However, the study shows limitations such as discomfort from the headbands and the system's dependence on submersion duration alone, which could lead to false positives or delayed responses. My project improves on this by integrating multi-sensor data including movement, physiological indicators and incorporating a wearable device that vibrates when distress is detected, offering a faster and more accurate detection without relying solely on submersion time.

A study published in the journal Sensors describes a wearable device that uses multiple parameters, including heart rate, blood oxygen levels, motion, and water depth to identify discomfort in swimmers [12]. Its waterproof and sensor-rich design enables it to operate in a variety of water environments and provide more comprehensive monitoring than single-sensor systems. However, its complexity can affect comfort, size, and battery life, making it potentially inconvenient to use for extended periods of time. In addition, differences in individual physiological responses and movement patterns can also affect its accuracy and lead to inconsistent performance between different users. Our project builds on this approach, using more compact, energy-efficient components and machine learning algorithms that learn and adapt to each user's unique behavioral and biometric data. This personalization enhances the system's comfort, reliability, and overall performance, making it more suitable for both everyday leisure and professional swimming use.

A study published in the journal Information explored AI-based drowning detection video surveillance systems that monitor swimmers' behavior through computer vision [13]. These systems are able to identify irregular movements and sinking patterns and alert lifeguards in real time, providing a non-intrusive, automated monitoring solution. However, factors such as poor water clarity, poor lighting conditions, and restricted camera angles can significantly affect their effectiveness, especially in crowded or natural water environments. In addition, such systems often require complex installation and maintenance, limiting their practicality in certain settings. Our solution improves this approach by replacing fixed cameras with wearable sensors, enabling continuous real-time monitoring in a variety of environments for indoors and outdoors and eliminating the reliance on external environmental conditions. This makes our system more adaptable, cost-effective, and easy to use in a wide range of swimming environments.

6. CONCLUSIONS

Despite the great potential of our drowning detection wearable system, there are still some limitations [14]. One major issue is the comfort and ease of use of the device, which may be bulky or too obtrusive to some users, especially children or users who are sensitive to wearable technology. Another challenge is battery life, as constant sensor monitoring and wireless transmission can quickly drain the battery. Environmental factors, such as signal interference in

open water or crowded pools, can also affect connectivity and real-time alerts. In addition, individual physiological differences may also affect the accuracy of the detection algorithm. By giving more time, we will work on miniaturizing the device to make it more comfortable, lighter, and suitable for users of all ages. We will also use low-power sensors and smarter sleep modes to optimize power consumption. Improving the machine learning model with more diverse data will help it better adapt to different users [15]. Finally, we will enhance connectivity by supporting multiple networks.

Our project introduces an innovative, wearable solution for real-time drowning detection, aiming to enhance swimmer safety through smart, adaptive monitoring. While it has limitations, continued development in comfort, connectivity, and detection accuracy could make it a powerful tool in reducing water-related accidents and saving lives.

REFERENCES

[1] Epstein, Andrew E., et al. "Personal and public safety issues related to arrhythmias that may affect consciousness: implications for regulation and physician recommendations: a medical/scientific statement from the American Heart Association and the North American Society of Pacing and Electrophysiology." Circulation 94.5 (1996): 1147-1166.

[2] Eng, et al. "An automatic drowning detection surveillance system for challenging outdoor pool environments." Proceedings ninth IEEE international conference on computer vision. IEEE, 2003.

[3] Jolles, Jolle W. "Broad‐scale applications of the Raspberry Pi: A review and guide for biologists." Methods in Ecology and Evolution 12.9 (2021): 1562-1579.

[4] Mahesh, Batta. "Machine learning algorithms-a review." International Journal of Science and Research (IJSR).[Internet] 9.1 (2020): 381-386.

[5] Rusinkiewicz, Szymon, Olaf Hall-Holt, and Marc Levoy. "Real-time 3D model acquisition." ACM Transactions on Graphics (TOG) 21.3 (2002): 438-446.

[6] Glaser, Martin, et al. "The art of designing DNA nanostructures with CAD software." Molecules 26.8 (2021): 2287.

[7] Chong, Wai Soon. Intelligent image search engine with AI-Based similarity detection for web application. Diss. UTAR, 2024.

[8] Li, Xiaoye S. "An overview of SuperLU: Algorithms, implementation, and user interface." ACM Transactions on Mathematical Software (TOMS) 31.3 (2005): 302-325.

[9] Duhamel, Pierre, and Martin Vetterli. "Fast Fourier transforms: a tutorial review and a state of the art." Signal processing 19.4 (1990): 259-299.

[10] Chougale, Pankaj, et al. "Firebase-overview and usage." International Research Journal of Modernization in Engineering Technology and Science 3.12 (2021): 1178-1183.

[11] Johnson, Molly B., and Karla A. Lawson. "Evaluation of the WAVE Drowning Detection SystemTM for use with children's summer camp groups in swimming pools: A prospective observational study." International journal of critical illness and injury science 12.4 (2022): 184-189.

[12] Pandian, P. S., et al. "Smart Vest: Wearable multi-parameter remote physiological monitoring system." Medical engineering & physics 30.4 (2008): 466-477.

[13] Jalalifar, Salman, et al. "Enhancing water safety: Exploring recent technological approaches for drowning detection." Sensors 24.2 (2024): 331.

[14] Zhu, Yujie. Design and development of a smart drowning detection system. Diss. Macquarie University, 2023.

[15] Sullivan, Emily. "Understanding from machine learning models." The British Journal for the Philosophy of Science (2022).