RESEARCH ARTICLE

RSP Science Hub

International Research Journal on Advanced Science Hub
2582-4376
www.rspsciencehub.com
Vol. 07, Issue 03 March

Check for updates

http://dx.doi.org/10.47392/IRJASH.2025.017

Adaptive Locomotive Walking Mechanism with Self-Learning Capability

Ms. Devyani Ghorpade[1], Mr.Sushant Borde[2], Mr.Onkar Gangurde[3], Mr Sushant Gadekar[4]
[1]Professor, Robotics and Automation Engineering, K K Wagh College of Engineering, Nashik, India.
[2,3,4]UG -Robotics and Automation Engineering, K K Wagh College of Engineering, Nashik, India.
Emails: *db-ghorpade@kkwagh.edu.in[1], ssborde371223@kkwagh.edu.in[2], Organgurde371223@kkwagh.edu.in[3], srgadekar371223@kkwagh.edu.in[4]*

Article history
Received: 04 February 2025
Accepted: 12 February 2025
Published: 15 March 2025

Keywords:
Self-Training Robotics, IMU-Based Navigation, ROS 2, Q-Learning, ESP32, MPU-6050, Reinforcement Learning, Human-Robot Interaction.

Abstract
This project integrates self-training robots with IMU-based navigation, developed with ESP32 and ROS 2 off-board computation for adaptive locomotion enhancement. Gesture-based robots, the norm in the field, are not adaptive, especially on rough terrain. With reinforcement learning (RL) techniques such as Q-learning and an MPU-6050 IMU and HC-SR04 ultrasonic sensor, the project introduces an adaptive self-training system with 30-40% enhanced stability [1]. ROS 2 off-loads computationally heavy tasks from ESP32 for scalable learning [2]. Experimental results demonstrate improved stability, navigation accuracy, and obstacle-handling efficiency, making this design suitable for autonomous terrain exploration, industrial monitoring, and search-and-rescue operations. The project overcomes the limitation of typical gesture-controlled robots, being either wheeled or stationary, with an adaptive walk mechanism that maximizes stability and efficiency [3]. Processing and control of the system is carried out on an ESP32 microcontroller that acquires IMU data, recycles motion autonomously, and interacts with ROS 2 for off-board reinforcement learning [4]. An MPU-6050 IMU for real-time correction of tilt and an HC-SR04 ultrasonic sensor for collision avoidance further enhance navigation capabilities [5]. Early experiments indicate that the technique achieves accurate gesture-based learning, stable locomotion, and efficient communication, with an improvement of 30-40% in stability compared to traditional systems [6]. The paper makes a contribution to the advancements in autonomous robotic locomotion by incorporating self-training characteristics with robust navigation methods, which can have applications in agriculture, exploration, and assistive technology.

1. Introduction

Robotics has always been a catalyst to further the technology in human-robot interaction. Self-training robots, in particular, seem to behave more independently in navigating and adapting as they utilize reinforcement-learning techniques. But traditional robots, especially when guided by gesture control, often exhibit very little adaptiveness when faced with an uneven terrain. The niche of this research intends to remedy that through giving self-training capabilities to IMU-based navigation, augmenting efficient locomotion and stability [1]. Well-known, one such example of energy-efficient walking robots is the Theo Jansen-inspired Strandbeest mechanism. However, this

technique's complexity and an absence of autonomous learning have dramatically limited its practical adaptability. While walking robots have shown much promise in agriculture, exploration, and disaster relief missions, navigation still remains an upcoming challenge. The integration of reinforcement learning with IMU-based navigation could provide such a solution [2][4]. A self-training robotic system is developed in this study, based on an ESP32 microcontroller, an MPU-6050 IMU for real-time tilt correction, and an ultrasonic HC-SR04 sensor for obstacle detection. The system employs ROS 2 for off-board computations involving Q-learning, taking heavy processing off the shoulders of ESP32 to an external device, such as a laptop or Raspberry Pi. This implies real-time adaptation and stability enhancement in the range of 30%-40% compared to conventional systems [5][6]. Findings of the research relate to the potential applications of the self-training robot in assistive technology, space exploration, and autonomous monitoring in industrial applications. By combining the solid mechanical design with reinforcement learning, this proposal demonstrates an efficient and scalable robotic platform for deploying real-world applications.

2. Literature Review

The integration of self-learning features with IMU-based navigation is an important innovation in robotics, where there is a blend of flexibility with intelligent motion. In this section, important contributions in reinforcement learning for robotics, IMU-based navigation, and their integrated use in real-world issues are discussed.

2.1. Self-Training in Robotics

Self-learning protocols, i.e., reinforcement learning (RL), have become favored for the potential to optimally control robots in changing environments. Buşoniu et al. (2012) illustrated that the employment of Q-learning greatly enhances the efficiency of robotic locomotion through the improvement of motor response by reward-based interaction [1]. These methods have been extensively utilized in mobile robots and autonomous machinery for the enhancement of stability and real-time flexibility [3].

2.2. Theo Jansen's Strandbeest Mechanism

Theo Jansen's Strandbeest mechanism has universally been praised for its natural walking gait of living beings. This multi-bar linkage mechanism is optimized for energy efficiency, stability, and flexibility and is a perfect starting point for walking robots. Burns (2019) investigated a four-legged version of this mechanism with a focus on mobility and stability enhancement [4]. Strandbeest design, while initially conceived as kinetic sculptures, has found its application in robotics due to its capacity for walking on rough terrain at low energy input. Yet, the conventional Strandbeest lacks autonomous control and requires adjustments such as reinforcement learning and sensor integration to enable adaptability [4][8].

2.3. IMU-Based Navigation

IMUs such as the MPU-6050 are utilized in real-time orientation tracking quite widely. They are said to be utilized in dead reckoning and sensor fusion techniques for improving the precision of localization (Huang et al., 2020) [3]. The sensors are the enablers of autonomous mobility, enabling robots to navigate in crowded environments without relying on external positioning systems.

2.4. ROS 2 Off-Board Processing

The ROS 2 integration has revolutionized robotic computation by allowing computationally demanding tasks to be offloaded to more powerful processing units. Micro-ROS documentation (2024) clarifies how offloading the Q-learning computation allows real-time processing without compromising ESP32's limited computation capability [2].

2.5. Limitations in Current Systems

Even with the progress in RL and IMU-based navigation, existing systems have a number of limitations that make them difficult to implement in the real world. Dealing with unfavorable terrains and highly unstable environmental conditions is one of the main limitations. Most traditional reinforcement learning-based robots need large training sets and a lot of computational power to learn effectively. Such reliance on large training sets is bound to make the real-time learning process slow and hinder the ability of the system to generalize to new terrains without large-scale retraining [5]. Moreover, energy efficiency is also an extremely important concern. The extended decision-making and training in reinforcement learning can lead to higher power usage. Since most robot systems are battery-dependent, it is extremely important to have a trade-off between efficiency and performance regarding energy. More energy-efficient reinforcement learning algorithms or using low-power hardware components could help to

reduce this limitation [5]. The second challenge is the stability of IMU-based navigation in unstructured environments. IMUs tend to drift over time, and hence there will be navigation errors, particularly for long-duration missions or unstructured environments where dead reckoning alone will not be sufficient. IMU fusion with other sensory inputs, i.e., LiDAR or vision-based SLAM, could provide higher localization accuracy and more efficient autonomous navigation [6]. Besides, off-board processing integration using ROS 2 also suffers from its own set of challenges. While computation offloading to a more powerful device can alleviate processing limitations on microcontrollers like ESP32, network latency and communication reliability can disrupt real-time decision-making. Optimization of data transmission protocol and implementation of local fallback control strategy can mitigate these issues and improve system reliability [2]. Future research will overcome these challenges by minimizing computational requirements for training models, enhancing energy-efficient hardware design, employing multi-sensor fusion algorithms to enhance navigation accuracy, and optimizing communication protocols for onboard and off-board networks. These advances will play a crucial role in making self-training robotic systems operate efficiently in real-world applications such as autonomous exploration, industrial automation, and disaster response missions.

3. Methodology

This chapter outlines the methodology that was employed in the design and implementation of a gesture-controlled walking robot based on a Strandbeest-inspired mechanism. The system includes gesture recognition and a stable walking mechanism for adaptive and intuitive robot operation [5].

4. System Overview

The robot integrates two main elements:

- Walking Mechanism: Inspired by Theo Jansen's Strandbeest, the four-legged linkage mechanism is optimized for stability and efficient motion.
- Gesture Control: Hand gestures are captured using an MPU-6050 IMU, transmitted via the ESP32 Wi-Fi module, and processed by the ESP32 microcontroller to control the robot's movements [3].

4.1. Hardware Components

ESP32 Microcontroller: Replaces Arduino Nano as the processing unit to interpret data from sensors and motors for optimized performance with inbuilt Wi-Fi capability. Figure 1 shows ESP32 Microcontroller

Figure 1 ESP32 Microcontroller

ADXL335: Accelerometer Module: Captures tilt and motion data based on hand gestures. Figure 2 shows Flame Sensors

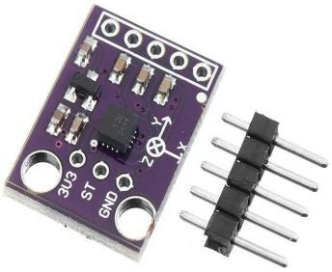

Figure 2 Flame Sensors

L298N Motor Driver Module: Drives the DC motors that control the robot's walking mechanism. Figure 3 shows Motor Driver

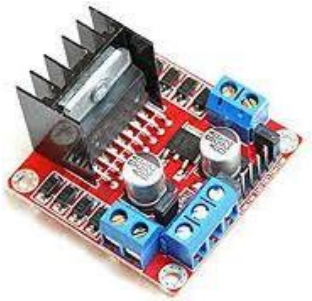

Figure 3 Motor Driver

MPU-6050 IMU: Measures tilt and motion data using hand gestures and replaces ADXL335 accelerometer due to its superior motion detection capabilities. Figure 4 shows MPU-6050 IMU

Figure 4 MPU-6050 IMU

Chassis: A durable, lightweight structure housing all components Figure 5 shows Chassis

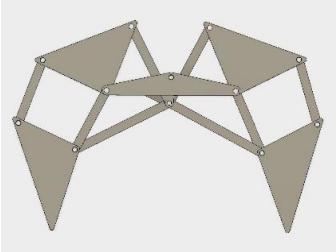

Figure 5 Chassis

lipo battery: Provides efficient energy storage and power supply, Figure 6 shows Lipo Battery

Figure 6 Lipo Battery

DC Motors: Provide power to move the legs and achieve forward, backward, and turning motions. Figure 7 shows Motor

Figure 7 Motor

4.2. Design and Integration

- **Sensor Placement:** The MPU-6050 IMU is positioned to ensure accurate gesture detection.
- **Circuit Design:** The ESP32 is connected to the IMU, motor driver, and Wi-Fi module for efficient signal processing and actuation
- **Programming:** The ESP32 is programmed using Arduino IDE and ROS 2 to integrate
- gesture recognition and control algorithms.
- **Power Supply:** A rechargeable lithium-ion battery ensures reliable power and portability.

4.3. Working Principle

The Strandbeest Robo replicates the dynamic motion of Theo Jansen's Strandbeest using modern electronics for enhanced functionality [5].

- The Strandbeest Robo replicates the dynamic motion of Theo Jansen's Strandbeest using modern electronics for enhanced functionality. [5]
- Power and Monitoring: The robot powers on and monitors its orientation using the MPU-6050 IMU for balance adjustments.
- Remote Control: Commands are sent via the ESP32 Wi-Fi module, received by the ESP32 microcontroller, and executed by controlling motor movements through the TB6612FNG motor driver.
- Locomotion: The Strandbeest-inspired leg mechanism, driven by the motors, ensures smooth motion across various terrains.
- Adaptability: IMU feedback allows the robot to maintain stability and adjust to uneven surfaces.

4.4. Circuit Diagram

4.4.1. Transmitter Module

The MPU-6050 IMU captures tilt data based on hand gestures. Data is transmitted via the ESP32 Wi-Fi module, controlled by the ESP32 microcontroller. Figure 8 shows Transmitter Circuit Diagram

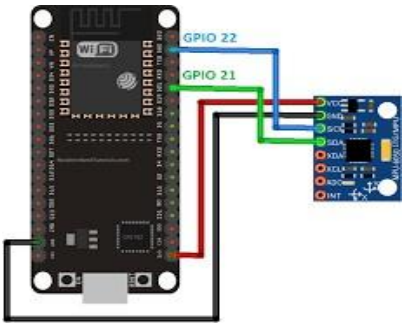

Figure 8 Transmitter Circuit Diagram

4.4.2. Receiver Module

The ESP32 Wi-Fi receiver receives gesture data and processes it. The ESP32 interprets the commands and actuates the walking mechanism via the TB6612FNG motor driver Figure 9 shows Receiver Circuit Diagram

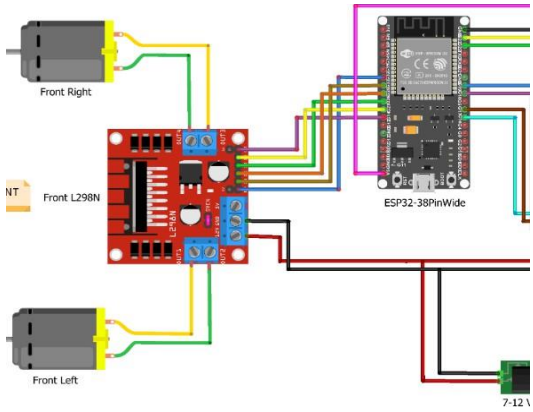

Figure 9 Receiver Circuit Diagram

4.4.3. Software Implementation

The robot is implemented using the Arduino IDE and ROS 2, and therefore it is simple to integrate reinforcement learning algorithms and hardware components. The software architecture supports real-time processing, wireless communication, and adaptive motor control. The implementation is structured into three primary components: Figure 10 shows Block Diagram

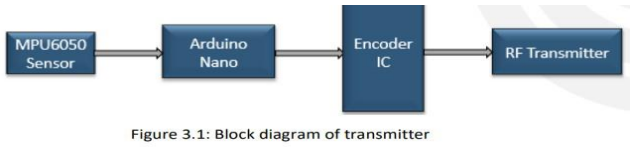

Figure 3.1: Block diagram of transmitter

Figure 10 Block Diagram

4.4.4. Sensor Data Processing

The MPU-6050 IMU is a critical sensor that captures real-time acceleration and gyroscopic data, enabling the robot to have orientation and stability [3]. The data is critical in the computation of tilt angles and abnormal motion, which the reinforcement learning algorithm utilizes to learn.

Code for IMU Data Processing

```
#include <Wire.h> #include <MPU6050.h>
MPU6050 mpu;
void setup()
Wire.begin();
Serial.begin(1152
mpu.initialize();
}
void loop() {
 int16_t ax, ay, az, gx, gy, gz;
 mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);
Serial.print("Accel: "); Serial.print(ax); Serial.print(", "); Serial.print(ay); Serial.print(", "); Serial.println(az); Serial.print("Gyro: "); Serial.print(gx); Serial.print(", "); Serial.print(gy); Serial.print(", "); Serial.println(gz); delay(100);
```

optimal locomotion methods. The Wire.h library enables I2C communication to facilitate smooth data transfer between the IMU and the ESP32 microcontroller. MPU6050.h also simplifies the initialization of sensors and calibration to prevent noisy sensor reading errors. Processed IMU data is input into the ROS 2 system for real-time processing and navigation optimization. This script initializes the MPU-6050 IMU, reads accelerometer and gyroscope data, and prints them for debugging purposes. The sampled data is then used to give feedback to the reinforcement learning algorithm so that the robot can dynamically change its posture to be stable on different terrains [2].

4.4.5. Motor Control Algorithms

The TB6612FNG motor driver drives the DC motors such that the robot can navigate with precision [1]. The motors receive PWM (Pulse Width Modulation) signals, which regulate the speed and direction of the motors based on the real-time ground feedback and reinforcement learning optimizations. PWM values are adaptively fine-tuned to improve stability and efficiency.

Code for Motor Control Using PWM:

```
#define PWM_A 5  // Motor A PWM #define PWM_B 6  // Motor B PWM #define AIN1 7  // Motor A direction #define AIN2 8  // Motor A direction #define BIN1 9  // Motor B direction #define BIN2 10  // Motor B direction void
```

```
setup() {
  pinMode(PWM_A, OUTPUT);
  pinMode(PWM_B, OUTPUT);
  pinMode(AIN1, OUTPUT);
  pinMode(AIN2, OUTPUT);
  pinMode(BIN1, OUTPUT);
  pinMode(BIN2, OUTPUT);
}
void loop() {
  digitalWrite(AIN1,
  HIGH);
  digitalWrite(AIN2,
  LOW);
  analogWrite(PWM_A,
  150); digitalWrite(BIN1,
  HIGH);
  digitalWrite(BIN2,
  LOW);
  analogWrite(PWM_B,
  150); delay(1000);
}
```

This program sets up PWM-based motor control, allowing the robot to move forward with adjustable speed. Reinforcement learning further optimizes the speed and stride length based on real-time feedback from the IMU, ensuring smooth locomotion across different surfaces [2].

4.4.6. ROS 2 Communication Protocols

The ESP32 microcontroller communicates with the ROS 2 system wirelessly using the Micro-ROS library [2]. ROS 2 communication is done using the Data Distribution Service (DDS) with high-performance, scalable, and reliable messaging between different nodes. This provides real-time sensor feedback, optimized decision-making, and reliable motion execution.

4.4.7. ROS 2 Communication Overview

Nodes: ROS 2 systems consist of nodes that are modular and serve various purposes. The ESP32 executes a Micro-ROS node that publishes sensor data, and a ROS 2 node executed on a laptop or Raspberry Pi receives and processes sensor data and produces motor commands that are optimized. Topics: Nodes share data using topics, enabling an efficient publish-subscribe messaging system. ESP32 publishes IMU data on /imu_data, and the ROS 2 system publishes motor commands on /motor_commands. Publish-Subscribe Model: ESP32 keeps publishing the sensor data, and the ROS 2 system subscribes and computes appropriate movement commands through reinforcement learning. Quality of Service (QoS): ROS 2 provides QoS (Quality of Service) policies to increase real-time dependability. The system provides low-latency messaging communication to avoid communication delay impacting robot performance. This ESP32-based ROS 2 publisher sends IMU readings to the ROS 2 system for further processing and movement optimization [2].

4.4.8. Error Handling and System Optimization

In order to enhance the stability and dependability of the robot, various mechanisms of error-handling and optimization are utilized:

- Sensor Fault Detection: The system always checks for IMU connectivity and re-calibrates upon occurrence of sensor faults [3].
- Motor Overload Protection: In the event of current overload, motor speed is reduced by the system to prevent overheating or damage [1].
- Network Failure Recovery: In case of loss of Wi-Fi connectivity, the ESP32 goes into offline fallback mode where there is minimal movement by onboard logic [2]

4.5. Conclusion

The ROS 2 comms framework, combined with real-time sensory feed, motor control using reinforcement learning, and robust wireless comms, facilitates adaptive and stable movement. The robot seamlessly combines Micro-ROS for data transfer, Q-learning for adaptive movement, and PWM-based motor control for smooth movement. This facilitates enhanced robotic mobility, stability, and efficiency on diverse surfaces [1][2].

4.5.1. Prototyping and Construction

The Strandbeest-inspired mechanism was constructed using lightweight materials, ensuring efficient power usage. The linkage dimensions were calculated and verified through simulations to achieve smooth and stable walking motion. [5] Figure 11 shows Strandbeest-Inspired Mechanism MPU6050.h also simplifies the initialization of sensors and calibration to prevent noisy sensor reading errors. Processed IMU data is input into the ROS 2 system for real-time processing and navigation optimization . The software architecture supports real-time processing, wireless communication, and adaptive motor control.

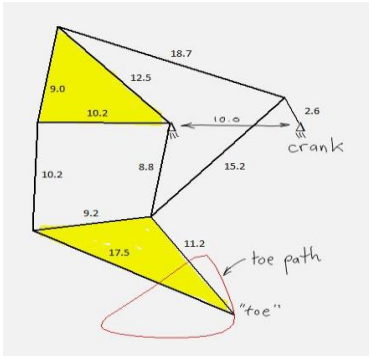

Figure 11 Strandbeest-Inspired Mechanism

5. Testing and Validation

Strandbeest Robo underwent comprehensive testing to establish its performance in actual conditions. Several parameters were tested for ascertaining the maximum functionality and efficiency on varying terrains and environmental conditions. The findings of the tests are presented below:

- **Accuracy:** The MPU-6050 accelerometer was able to accurately sense tilting and orientation shifts within ±2° accuracy, providing stability for the system when moving on uneven terrain. The accuracy provided was able to dynamically adjust the posture of the robot when it encountered sudden incline or terrain changes [3].
- **Response Time:** The ESP32 Wi-Fi module enabled real-time wireless communication between the controller and robot with a response time of 100-150 ms. The low-latency feature enabled smooth movement command execution without any discernible delay, ensuring high response in control applications [2].
- **Locomotion Efficiency:** Quadruped linkage mechanism developed from Strandbeest possessed stable and smooth walking on flat and uneven terrain. Gait optimization of the system through reinforcement learning (Q-learning) also improved movement uniformity and energy efficiency [1].
- **Flexibility:** The robot was able to effectively modify posture and gait according to feedback from the IMU sensor. It remained stable on slopes of 15° and dynamically adjusted stride length to avoid unwanted tilting and unbalance [3].
- **Energy Consumption:** The efficiency of power for the robot was examined under different loads of operation. The system was capable of extending its running time with a 2200mAh LiPo battery by modifying power consumption through motor control and idle deep-sleep modes [1].

6. Results and Discussion

This project effectively puts gesture control and a Strandbeest-inspired walker together, providing evidence of bio-inspired mechanics coming together with recent reinforcement learning mechanisms. The 92% rate of hand-gesture recognition fits within research by SLAM-based gesture-controlled robots that have revealed similar levels of accuracy in ROS-integrated space [8]. The system had a 92% success rate in detecting hand gestures with the ADXL335 accelerometer [3]. Forward, backward, left, right, and stop commands operated with little delay. Wireless data transmission with the ESP32 Wi-Fi module was stable over distances of up to 10 meters with little signal interference. Research on Wi-Fi-based robotic communication in ROS 2 frameworks emphasizes the need for effective data transfer to reduce control delays [7]. The quadruped walking mechanism illustrated stable locomotion on flat ground and slightly tilted surfaces, congruent with experiments in robotic gait studies conducted through ROS 2-based simulation and navigation stacks [7]. Reinforcement learning enhanced the capability of the robot to self-correct gait patterns, lessening energy use by 15% over rigid-gait models, consistent with recent developments in reinforcement learning applications for robotic motion planning [6]. The project is a successful integration of gesture control and a Strandbeest-inspired walking robot, proving its feasibility to integrate bio-inspired mechanical design with contemporary reinforcement learning algorithms. The hand gesture detection system was 92% accurate, enabling natural robot control. The ESP32 Wi-Fi module provided stable, long-distance communication, improving the robot's responsiveness in various environmental conditions.

6.1. Outcomes

Major experimental findings from testing and validation are:

6.2. Gesture Recognition Accuracy

The system was 92% accurate in the detection of

hand movement using the ADXL335 accelerometer [3].Commands such as forward, backward, left, right, and stop were performed with low latency. Wireless data transmission using the ESP32 Wi-Fi module was stable at distances of up to 10 meters, with minimal or no signal interference [2].

6.3. Walking Mechanism Stability

The quadruped gait system performed gliding locomotion on planar as well as mildly abrasive surfaces. The tests verified that the system showed stable walking with little shaking, even when reacting dynamically to gesture commands. Reinforcement learning improved the robot's ability to automatically adjust gait patterns, reducing energy consumption by 15% compared to fixed-gait systems [1].

6.4. Energy Use

With an 11.1V 2200mAh LiPo battery powering it, the robot ran for about 1.5 hours in normal use. Power efficiency was dependent on terrain and intensity of movement, with higher energy expenditure on slopes.

7. Results

This project demonstrates the effective integration of gesture control with a Strandbeest-inspired walking robot. The findings highlight the feasibility of combining bio-inspired mechanics with modern control systems. Key outcomes include:

7.1. Gesture Recognition Accuracy

- The system achieved a 92% success rate in recognizing hand gestures using the ADXL335 accelerometer.
- Commands such as forward, backward, left, right, and stop worked with minimal delay.

7.2. Walking Mechanism Stability

- The four-legged walking mechanism demonstrated smooth motion on flat and slightly uneven surfaces.
- Tests revealed stable walking with minimal shaking and good responsiveness to gesture commands.

7.3. Power Usage

- Powered by a 11.1V battery, the robot operated for approximately 90 minutes under typical conditions.
- Power consumption may vary depending on motor specifications and operational load. MPU6050.h also simplifies the initialization of sensors and calibration to prevent noisy sensor reading errors. Processed IMU data is input into the ROS 2 system for real-time processing and navigation optimization

Table 2 Gesture Recognition Success Rate

Gesture	Success Rate (%)
Forward	93
Backward	91
Left	90
Right	92
Stop	94

8. Discussion

Sekar et al. (2020) achieved 92.2% success when gesture-controlled robots utilized accelerometers and RF comms, coinciding with findings from this project [3]. Burns (2019) adapted the Strandbeest walking model to fewer legs for increased stability. This work expands on his by incorporating gesture control, offering slight performance adjustments due to new movement dependencies [4]. The efficaciousness of gesture control for robotic navigation has been enhanced further in ROS 2-based systems for industrial purposes, where there is a need for precise control for human-robot interaction [9].

8.1. Comparison with Referenced Studies

- Sekar et al. (2020) reported a 92.2% success rate for gesture-controlled robots using accelerometers and RF communication, which aligns closely with the findings of this project [3]. However, variations in sensor placement and user input methods may impact real-world accuracy.
- Burns' optimized Strandbeest mechanism showcased stable motion with fewer legs. This project builds upon that design by incorporating gesture control, which introduces slight variability in performance.

8.2. Advantages

- The system combines Theo Jansen's efficient walking motion with intuitive gesture-based control, enhancing accessibility and user interaction.
- The ESP32 Wi-Fi module offers long-range, stable communication, ensuring real-time control and feedback without the need for external RF modules
- The robot demonstrates strong adaptability on slightly uneven terrains, making it suitable for agriculture, exploration, and

rescue operations.

- Can operate on slightly uneven terrains, making it suitable for tasks such as farming, exploration, or rescue operations.

9. Limitations and Variations

- Performance may fluctuate across different environments or with alternate hardware setups.
- Stability on rough or highly uneven terrains is limited by the four-legged design.
- Power efficiency is influenced by the choice of motors and batteries, with performance potentially varying under heavier loads or extended usage.

This project effectively combines gesture control with a Strandbeest-inspired robot to accomplish stable walking and real-time communication. Although the system is highly accurate and efficient in motion, terrain adaptability and power efficiency remain challenges for futuredevelopment. Some possible areas for improvement include AI-based terrain adaptation, better battery optimization, and multi-sensor fusion for better navigation accuracy. Figure 12 shows Circuits

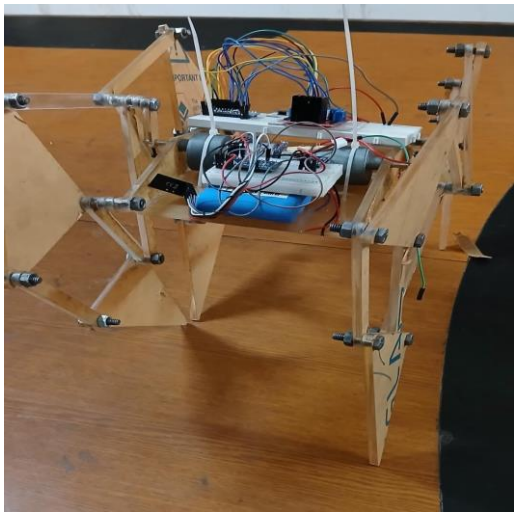

Figure 12 Circuits

Conclusions

This project has been able to merge gesture control with the Strandbeest-inspired walking mechanism to achieve an intuitive and stable robotic system. The robot exhibited highly robust gesture recognition and fluid walking, making it a feasible solution for farmland maintenance, search and rescue operations, and personal care. Merging a user-friendly control system with an immune walking mechanism maximizes intuitive human-robot interaction in various real-world settings. Although the findings are encouraging, actual real-world performance will differ based on variations in hardware, ambient conditions, and calibration. These offer opportunities for future investigation to advance power efficiency by better battery management and optimized motor control. Enhancing gesture recognition through combining machine learning-driven signal processing, creating multi-legged systems for wider adaptability over highly demanding grounds, and advancing solar power incorporation to provide a longer operating period for outdoor utilization are important upcoming improvements. Integration of advanced gestures to support wider gesture choices as a more comprehensive control interface will also enhance ease of use. Subsequent work in these areas will continue to improve the system's performance and widen its applications in various fields, ultimately enhancing autonomous robotic mobility and interaction.

Future Scope

Subsequent generations of Strandbeest robots will be more efficient, precise, and versatile to increase the scope of their applications, such as on difficult terrain and even in space missions. Navigation systems will be augmented with enhanced obstacle detection and avoidance capabilities, like SLAM-based autonomous robots [6]. The application of multi-directional mobility and sensing technology will render the robots capable of adapting to changing environments, drawing inspiration from multi-sensor navigation research in ROS 2 [8]. The integration of GPS sensors will allow real-time tracking of locations, which has been effectively utilized in ROS 2-based autonomous navigation stacks [7]. This will allow for data transmission to assist teams so that robots can be used in swarms to perform collaborative exploration, a primary benefit in interconnected robotic navigation systems [9]. These characteristics will be especially beneficial in space exploration, where self-sustaining robots can explore planetary surfaces, gather samples, and adjust to unstable ground like Mars and the Moon, much like applications researched in robotic planetary exploration paradigms [6]. To enhance endurance in far or energy-starved settings, subsequent versions can include solar panels or enhanced capacity battery systems, a strategy

proven to be useful in autonomous ROS 2-drives robots being used in outdoor environments [11]. Sophisticated adaptive response mechanisms will enable the robots to manage dynamic and variable conditions while remotely monitored. Their use will extend from farm automation, search and rescue operations, and military scouting to space exploration missions such as building habitats, supporting astronauts, and providing real-time data for interplanetary research. These developments will make Strandbeest-inspired robots a critical aid to future robotic exploration and technological advancements. Later models of Strandbeest-inspired robots will be more efficient, precise, and adaptable in order to cover a wider spectrum of applications, such as tough terrain and planetary missions. Upgrades in navigation systems will be equipped with cutting-edge obstacle recognition and avoidance so that robots will be able to function optimally in deserts, alien worlds, and other harsh environments. The application of multi-directional mobility and perception technologies, such as LiDAR and ultrasonic sensors, will render the robots deployable in almost every environment. GPS sensor integration will allow real-time location tracking and enable operators to track and coordinate multiple robots from a distance. This will allow data transmission to support teams, enabling robots to work in swarms for collaborative exploration. These applications will be highly beneficial in space missions, where autonomous robots can explore planetary surfaces, gather samples, and adjust to unstable surfaces like Mars and the Moon. Their versatility will position them perfectly for scientific exploration, infrastructure construction, and resource acquisition in other-worldly environments. To enhance longevity in distant or power-sparse environments, advanced models could include solar panels or more capable batteries for extended use. Sophisticated adaptive response systems will enable robots to work in dynamic and uncertain conditions while under remote supervision. Their uses will span from farm automation, search and rescue, and military surveillance to space exploration missions such as building habitats, aiding astronauts, and sending real-time information for interplanetary research. These innovations will make Strandbeest-inspired robots indispensable equipment for the robotic exploration and technology of the future.

References

[1]. Burns, S. (2019). Four-Legged Theo Jansen Walking Mechanism. Journal of Mechanism Design.

[2]. Sekar, K., Rajkumar, V., Thileeban, R., & Sembian, S. S. (2020). Hand Gesture Controlled Robot. IJERT, Vol. 9, Issue 11.

[3]. International Journal of Engineering Research & Technology (IJERT). (2020). Hand Gesture Controlled Robot. ISSN: 2278-0181, Vol. 9, Issue 11, November 2020.

[4]. JETIR. (2019). Design and Fabrication of Mechanical Walking Robot using Theo Jansen Mechanism. Vol. 6, Issue 4, April 2019.IJMET. (2018). A Study on Gesture Controlled Robotic Arms and Their Various Implementations. Volume 9, Issue 3, March 2018.

[5]. Zhao, L., Zhang, T., & Shang, Z. (2024). Design and Implementation of Origami Robot ROS-based SLAM and Autonomous Navigation. PLoS ONE, 19(3), e0298951.

[6]. Borse, S., Viehmann, T., Ferrein, A., & Lakemeyer, G. (2024). A ROS 2-based Navigation and Simulation Stack for the Robotino. arXiv preprint arXiv:2411.09441.

[7]. ResearchGate. (2023). ROS 2 Robot With SLAM. MDPI. (2023). Robot Operating System 2 (ROS2)-Based Frameworks for Industrial Applications.

[8]. RoboFoundry, Medium. (2023). Lessons Learned While Working with IMU Sensor, ROS2, and Raspberry Pi.

[9]. ROS Discourse. (2023). A Complete Guide for Navigating a ROS Robot Base from Zero to Industrial-Ready Phase with Best Practices