

Myriapod Robot: Modular Robot Platform with Exotic Locomotion

Juan Torres Camacho, Ethan Thomas, and
Axel Maysonet

Dept. of Electrical Engineering and Computer
Science, University of Central Florida,
Orlando, Florida, 32816-2450

Abstract — *This paper presents a novel approach to advancing mobile robotics through the design and construction of a prototype myriapod robot platform. Inspired by centipedes, this robot features multiple segments, each equipped with its own set of legs for efficient navigation across varied terrain. The project aims to demonstrate the platform's modularity and flexibility by allowing seamless addition of sensors and devices to individual segments. Utilizing a custom-designed PCB and 3D-printed body, the robot achieves coordinated movement via a metachronal wave pattern. With wireless teleoperation capabilities, this versatile platform shows promise for applications in search and rescue, inspection, and manufacturing, contributing significantly to the evolution of mobile robotics.*

Index Terms — Mobile robots, centipede, metachronal wave, Controller Area Network (CAN), Bluetooth

I. INTRODUCTION

Humans have long been intrigued by replicating the abilities of creatures in nature. From maze-solving robot mice to stair-climbing robot dogs, the quest to mimic animals' mobility has captivated us. Alternative mobility robots offer features and capabilities beyond traditional mobile robots, such as navigating challenging terrain and swimming in oceans, opening doors to endless applications.

Among nature's marvels, myriapods, with their numerous legs, stand out for their exceptional maneuverability on land. Centipedes, a type of myriapod, utilize their many legs for stability and control, contrasting with animals sporting just two or four legs. However, designing bipedal or quadrupedal robots capable of both speed and stability poses immense challenges, demanding complex computations to remain upright. Instead, focusing on centipede-inspired robots offers mechanical solutions for balance and movement, minimizing computational requirements.

Centipedes exhibit a distinct body structure with a head for control and a segmented trunk housing legs for efficient locomotion. Our robotic centipede mimics this design closely, featuring a head for logic and a modular trunk with segmented body parts, each accommodating a pair of legs. This modularity allows users to expand the robot's capabilities by adding segments, enabling the integration of additional functionalities like sensors or logic units.

Our robotic centipede aims to provide an affordable, durable, and expandable mobile robot platform for diverse applications, from search and rescue missions to industrial surveying and potential space exploration. Unlike costly alternatives, our solution prioritizes affordability by utilizing inexpensive computer components without compromising expandability. Unlike more restrictive options, such as Spot from Boston Dynamics [1], the robotic centipede offers limitless possibilities for functionality enhancement through modular expansion.

In comparison to other alternative mobility robots like aerial drones or treaded robots, our centipede robot excels in versatility and maneuverability. While drones and treaded robots are specialized for specific tasks, the robotic centipede offers a broader range of capabilities without being weighed down by additional equipment or limited by terrain. With ongoing advancements, our centipede robot aims to demonstrate its superiority and further enhance its robustness in various applications.

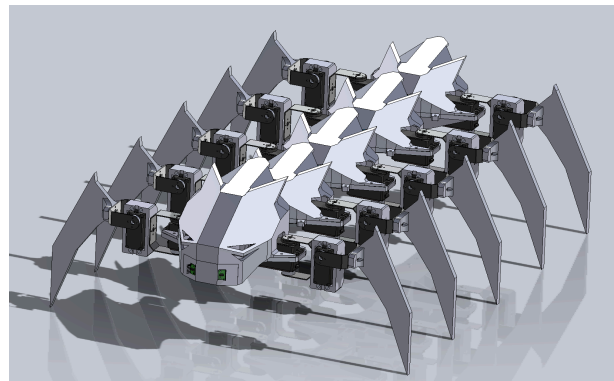


Fig. 1. Myriapod CAD assembly

II. HARDWARE COMPONENTS

The myriapod robot is constructed from carefully selected physical components, collectively chosen by our team, to ensure seamless communication among them, thereby facilitating the optimal functionality of the final robot assembly.

A. Microcontroller

The STM32 G431RB was chosen for its ability to fulfill the requirements of the myriapod robot project efficiently. Its functionality aligns well with the project's distributed computing system, allowing each segment to have its own microcontroller for communication and command execution. With multiple simultaneously available communication interfaces like I2C, SPI, UART, and CAN, it supports a wide range of peripherals and simplifies integration with other devices. Ample GPIO pins and timers cater to leg servo control, while its CORDIC math coprocessor significantly reduces the computational overhead of the trigonometric operations needed to generate the metachronal waves used to control the leg servos. Supported by comprehensive development resources like STM32CubeMX and availability of development boards, the STM32G431RB emerges as the optimal choice, balancing features, price, and development support effectively.

B. Distance Sensors

The STMicroelectronics VL53L7CX time-of-flight (ToF) sensor was selected as the ideal choice for obstacle avoidance in the myriapod robot project. Despite its slightly higher cost, its exceptionally wide field of view offers nearly three times the coverage compared to competing sensors, reducing the number of sensors needed and simplifying PCB design. Additionally, its cost-effectiveness remains compelling, especially considering its comprehensive development support, including development kits, integration-ready boards, and well-documented software libraries. The sensor's wide field of view, combined with its cost efficiency and robust support resources, makes the VL53L7CX the optimal solution for achieving effective obstacle avoidance in the project.

C. Leg Servos

We chose the ReadyToSky TD-8160 servo for our centipede's legs because of its optimal size and impressive 60 kg/cm torque capacity, which are crucial for handling larger designs and increased loads. Its compatibility with our planned lithium polymer battery, operating within a range of 6 - 8.4V, ensures seamless integration with our chosen battery. Additionally, the servo's exceptional IP67 rating guarantees resilience against water and dust, enhancing its suitability for outdoor environments. These features make the ReadyToSky TD-8160 servo a reliable choice, ensuring the effective implementation and sustained operation of our centipede project.

D. Turning Servo

The MG996R servo motor was chosen for the robotic centipede turning task due to its comprehensive advantages over similar servo motors. Despite slight differences in specifications among brands, the MG996R stood out for several reasons. Firstly, its dimensions, slightly smaller than competitors, facilitated a more efficient design integration, crucial for maintaining the centipede's appearance. Secondly, while its operating speed fell between alternatives, its weight of 55g was lighter, aiding in overall agility. Thirdly, its lower stall torque met requirements without unnecessary strength, thus saving on costs. Additionally, its operating voltage range of 4.8V to 7.2V provided flexibility and cost efficiency. With a competitive unit price of \$7.50, the MG996R emerged as the most cost-effective choice, effectively balancing performance and affordability.

E. Battery

We've chosen the Zee LiPo battery for its exceptional value and high-quality performance. Not only does this brand offer reliable batteries at an affordable price point, but it also garners positive reviews and feedback from users, indicating its dependability over time. Crucially, it meets all our project's criteria, including size, weight, cost-effectiveness, and the ability to deliver the necessary voltage and current. With these qualities, the Zee LiPo battery ensures smooth integration into our project, providing the power and reliability we need for its successful operation without compromising on quality or budget.

III. HARDWARE DESIGN

The myriapod robot's design employs a distributed system utilizing discrete microcontrollers and a single Controller Area Network (CAN) bus for communication. Each segment comprises a power distribution board, G431RB microcontroller, VD230 CAN transceiver, and four leg servo motors. This setup offers flexibility, simplifying both the mechanical and electrical connections between each segment and allows the robot to function with just the head segment or with additional body segments for movement. Centralizing the robot's logic in the head segment would complicate mechanical connections and limit expansion potential.

The VD230 CAN transceiver on each segment facilitates bidirectional communication between the segment and the CAN bus, allowing movement

commands from the head segment and data transmission from segments back to the head. While the transceivers don't support the faster FD-CAN standard, classic CAN's 1 Mbps bandwidth is sufficient for the project. The CAN network follows a bus topology with 120 Ω resistors at each end, easily configured via jumpers on development boards and segment PCBs. This distributed architecture enhances flexibility, simplifies connections, and enables efficient communication across the myriapod robot.

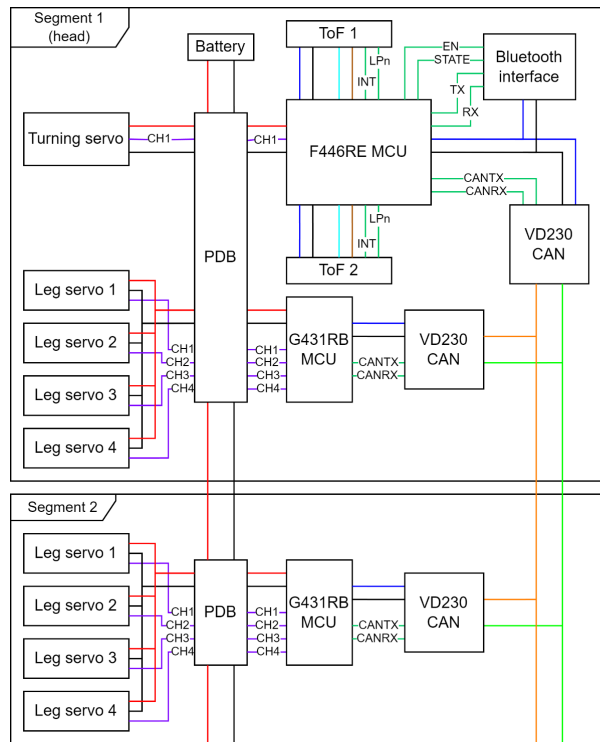


Fig. 2. High level hardware design

The head segment of the myriapod robot includes additional components such as a battery, F446RE MCU, two time-of-flight sensors, and a turning servo unit. During early development with the distance sensors, it was discovered that the firmware for the sensors must be loaded from the MCU to each sensor on startup. Due to the G431RB's relatively small 128 KB flash, fitting the 84 KB sensor firmware on the MCU was a challenge, even without the motor control code. As a result we added a second MCU, the STM32F446RE to the head segment with sufficient flash memory to control these sensors and orchestrate communication between the robot and the user. Unfortunately, the F446RE lacks the CORDIC math co-processor of the G431RB, so it cannot replace the MCUs on the body segments.

IV. MECHANICAL DESIGN

The segment electronics' distributed nature necessitates a similar approach in the mechanical design of the robot. While the head and tail segments have slight variations to accommodate additional functions, all body segments are identical in form and function, simplifying the design. They utilize the same leg mechanism to maintain uniformity. It should be noted that all credit for the robot's mechanical design, including the CAD assembly and simulation (Fig. 1, 3-5), belongs to Diego Santiago.

A. Head and Tail Segments

Removing the top covers of the centipede reveals the hardware configuration of each segment. The ST Nucleo development boards seen in Fig. 3 will be replaced by our custom PCBs. In the head segment, the secondary MCU (F446RE) manages the ToF sensors and Bluetooth interface, mounted above the primary MCU (G431RB). Based on mechanical considerations, the battery is mounted at the rear, facilitated by an additional tray in the rear segment, as shown in Fig. 3.

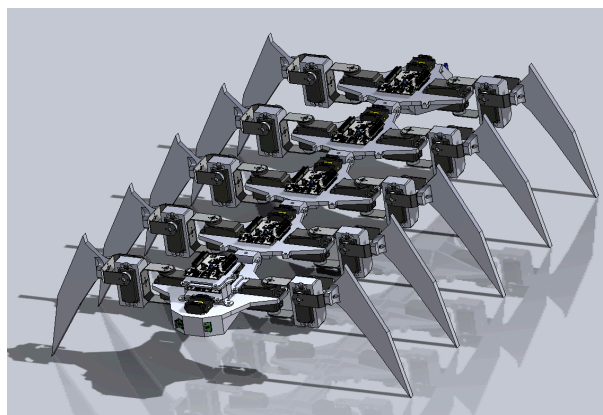


Fig. 3. Myriapod CAD assembly (no cover)

B. Simulation

In the design phase, static analysis was conducted to fortify potential weak points in the myriapod frame. Displacement and stress simulations (Fig. 4 and 5) revealed minimal displacement, well within ABS plastic's elastic range. Despite apparent displacement in Fig. 4, the largest recorded displacement was less than 4% of 1 mm, remaining within acceptable limits. Fig. 5's stress analysis showed predominantly solid blue color, indicating effective distribution of internal forces by sternites. These underside support plate structures efficiently manage forces from segment mass, servos, and legs in a stationary pose, ensuring structural integrity of the robot.

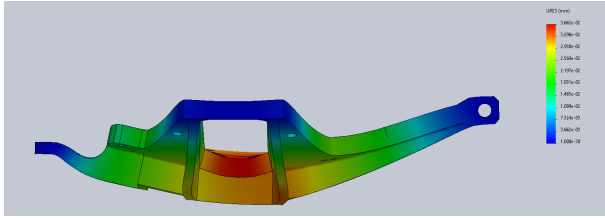


Fig. 4. Displacement analysis

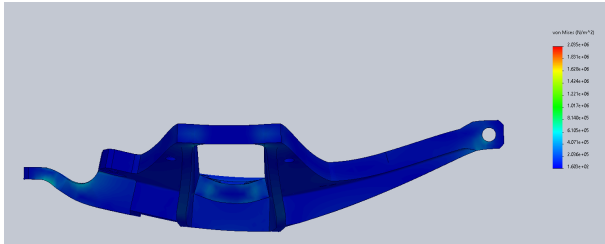


Fig. 5. Stress analysis

V. SOFTWARE DESIGN

The software in the myriapod robot's head and body segments is tailored to meet the overall robot requirements efficiently. It focuses on inter-segment communication via CAN bus, servo control, and peripheral communication using I2C or SPI. The head segment manages time-of-flight sensors for obstacle avoidance and controls the turning servo. Code is kept concise, utilizing existing libraries for reliability and simplified debugging. Interrupt-based, hardware-focused code maximizes responsiveness without the need for a real-time operating system, minimizing overhead. Interrupts are triggered by hardware events such as the reception of a CAN message or a timer elapsing, reducing CPU usage and increasing reliability. This approach ensures that tasks are triggered as soon as their stimulus is received and uses the nested vector interrupt controller (NVIC) on each MCU as a preemptible scheduler to handle simultaneous tasks [2].

A. Segment Communication Protocol

The CAN communication protocol requires a common "language" to ensure compatibility among devices on the bus. Each command corresponds to specific software requirements such as servo control, metachronal wave locomotion, and interfacing with any connected peripherals. Hardware abstraction layers handle frame formation and transmission, leaving programmers to determine frame field values. While a full discussion of the CAN frame format is beyond the scope of this paper, the arbitration (identifier), RTR, and DLC fields in each frame are of particular interest in

this project. The identifier denotes the segment and servo number in hexadecimal (e.g., 0x12 for Segment 1 Servo 2). The classic CAN protocol restricts data lengths to 8 bytes or fewer. To accommodate this limit, the language design uses the data length stored in the DLC field to identify the command in the message rather than dedicating a data byte to it. The remote transmission request (RTR) bit indicates whether the given frame is a data or remote frame. Data frames are the most common and are used to send data from one node to another. On the other hand, remote frames do not contain any data and are typically used to request data from another node. As designed, the head segment uses data frames to command the body segments and remote frames to request sensor data and for the heartbeat system.

The heartbeat system is responsible for periodically verifying that the head segment MCU can communicate with all body segment MCUs. First, the head segment MCU sends a remote frame to the body segment it is checking. Then, that segment sends a 1-byte data frame back to the head segment with its CAN ID. If the ID the head segment checked matches the ID in the 1-byte response frame, the check passes. As designed, the system only provides a visual indication to the user that communication is working and does not take any action if a check fails.

The language design ensures efficient parsing, leveraging buffered queues in both the G431RB and F446RE CAN interfaces. During initialization, hardware-level filters efficiently accept pertinent messages, preventing the CPU from wasting cycles processing messages not addressed for its segment. By employing bitmask comparisons in filters, precise addressing of segments and servos within CAN messages is enabled. As configured, the CAN ID of any incoming message must match the upper 7 bits of the segment to pass the filters. The remaining 4 bits in this field are the sub-identifier and are used to specify for which servo or peripheral the command is intended. An additional filter is configured to always accept messages from a designated "broadcast" ID. This ID is used to identify messages that should be received by all segments simultaneously, including time-sensitive operations. This protocol, capable of accommodating up to 128 segments each with up to 16 servos or other addressable components, provides extensive configurability for future expansions and experimentation. The parsing sequence shown in Fig. 6 shows how CAN messages are filtered by the MCU's CAN interface before being processed and executed. It should be noted that the VD230 CAN transceivers and the CAN interfaces on the MCUs are separate entities.

The former connects the latter to the CAN bus, whereas the latter is used to connect the MCU to the transceiver.

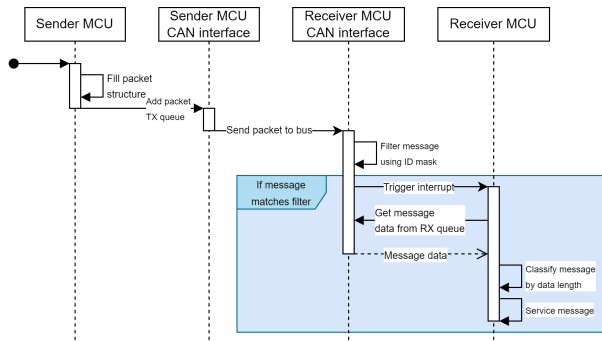


Fig. 6. Sequence diagram for CAN communication software

B. Servo Control

The servo control software subsystem facilitates precise control of the myriapod's servo motors, enabling communication between the segment MCUs and each servo. Servos, typical in remote control hobby applications, react to pulse-width modulation (PWM) signals to adjust their angles. Notably, servos react to the pulse length rather than the PWM duty cycle, allowing for flexibility in PWM signal frequency within a range that remains interpretable by the servo.

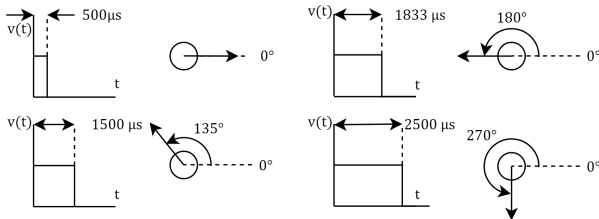


Fig. 7. Relationship between pulse duration and servo angle

The servo control subsystem establishes precise servo motor control by correlating pulse duration to servo angle, as depicted in Fig. 7. Servo movement within a defined range is achieved by mapping input angle units to corresponding pulse durations. The four servo motors on each body segment are controlled by a single timer with multiple capture/compare channels to ensure that the generated signals share a common timebase and simplify the control logic. While the servos have well-tuned integrated control systems, their high top speed can cause the current through the motor to spike, straining the power distribution system and potentially causing hardware errors. To address current spikes during servo movement, a linear slew mechanism was implemented to gradually change the PWM signals

instead of immediately sending the desired position. Additionally, precautions are taken to prevent servo collisions and minimize current spikes, particularly during startup, where a sequential motor movement sequence is adopted to avoid concurrent motion until all servos reach a predefined "home" position.

C. Metachronal Wave Locomotion

The myriapod robot employs metachronal waves for locomotion, characterized by identical waves with a phase shift between them [3]. Quadrature sine waves illustrate this principle, showing how multiple waves are evenly spread across a single period of the common function. This concept extends to various applications, such as three-phase electricity, where waves are spaced 120° apart to cover the 360° range of the sine function.

This strategy ensures mechanical stability by providing sufficient contact points with the ground. Given that a plane must be defined by at least 3 points, we can use (1) to determine the phase shift necessary between each segment for a centipede with N segments to be stable. This equation abstracts the raw phase shift value by relating it to L , the number of segments required for all phases to be present on at least one segment. This quantity can be found by dividing the angle of a complete rotation (e.g., 360°) by the chosen phase shift. Increasing the length of the metachronal sequence enhances stability, benefiting from the proximity of legs to grounded ones, but requires additional segments for the robot to physically support it. The implementation complexity of the metachronal wave depends on how the common function is computed, with options ranging from trigonometric functions to lookup tables. Leveraging MCU capabilities like the CORDIC coprocessor significantly reduces the time required to compute the metachronal wave for all four leg servos, leaving more time for other tasks and increasing the theoretical maximum speed of the robot.

$$\frac{2N}{L} \geq 3 \quad (1)$$

Efficient metachronal wave locomotion for the myriapod robot necessitates optimizing joint motion. While prioritizing hip joint range may seem logical, it leads to wasted energy due to misalignment of the reactive forces on the legs of the centipede. When solely employing the horizontal joint, reactive forces diverge from the intended path, undermining efficiency as the legs on opposing sides counteract each other's motions. A better approach involves integrating the knee joint to parallelize reactive forces, enhancing propulsion efficiency by leveraging torque from both joints. This method, illustrated in Fig. 8, ensures smoother

locomotion and maximizes energy utilization. Likewise, turning mechanics involve scaling movement angles based on commanded turn angles to ensure uniformity across segments. However, the limited contact points with the ground necessitate gradual propagation of turn commands through the body, allowing for smooth transitions between straight and turn-biased movements. Fine-tuning these parameters during prototyping will optimize turning effectiveness and overall locomotion efficiency.

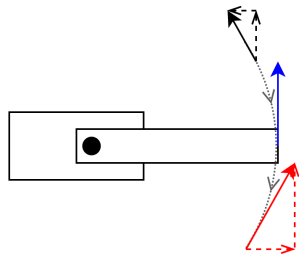


Fig. 8. Simplified force vector for naive metachronal wave

D. Startup and Synchronization Sequence

Ensuring synchronized movement among segments is crucial for efficient operation of the myriapod robot, as desynchronization could lead to collisions between legs and potential damage. While each segment is aware of its position within the robot and the motors under its control when powered on, the motors can move freely when unpowered. As a result, the motors could move unexpectedly if they are commanded to move without homing them, presenting a potential safety risk to both the user and the robot itself if not addressed. In order to mitigate this risk, all segments are sequentially commanded to move to a predefined home position after power-on to prevent leg collisions and indicate proper functionality. This process is essential for each joint of every leg to ensure correct alignment and readiness for operation.

The homing sequence requires two steps to safely move all legs from an unknown position in their range of motion to the home position. Initially, in an unspecified configuration denoted by gray cones in Fig. 8, all legs could potentially collide if commanded to move without homing. Step 1 of the homing process moves all legs towards the front of the centipede, with individual segments homed sequentially to minimize current spikes and ensure smooth movement. This step prioritizes the front legs, which are typically unobstructed during power-on. Step 2 completes the homing sequence by moving all legs back to a neutral position. This sequence was initially implemented with the assumption that the centipede should stand up as a result of this process. However, the stance at the end of Step 1 in Fig. 9 put significant strain on the frame. As a

result, this sequence should be only performed with the body touching the ground.

This synchronization process could be improved by upgrading leg motors for better accuracy and adding encoders to the motors so the body segment MCUs can query the motor's current position before commanding it to move. These enhancements aim to optimize the homing process, ensuring efficient and reliable operation of the robotic centipede.

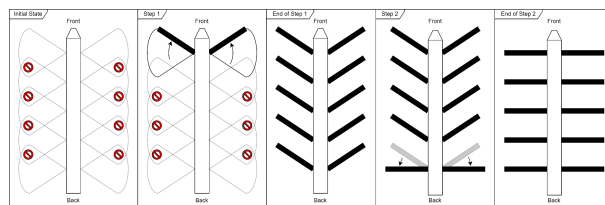


Fig. 9. Mechanical synchronization and homing sequence

VI. REMOTE CONTROL INTERFACE

A. Mobile application

The graphical user interface (GUI) for teleoperating the robot and its controlling application are shown in Fig. 10. The main screen features a directional pad (D-pad) and lever speed controls, essential for teleoperated movement. Utilizing a D-pad over raw text-based commands enables the user to easily control the centipede in real time. Each press of the D-pad sends a serial command over bluetooth to its receiver in the head segment. Each command is represented by a string of characters like "WAVE 30 45" and is received by the Bluetooth module and translated into one or more CAN messages by the F446RE on the head segment PCB. These messages are received and executed by the G431RB MCUs on each body segment. "30" corresponds to the desired speed value and "45" corresponds to the desired turning angle. Each direction of the D-pad has its own instructions, such as Up, which goes forward at a set speed with no angle set, Left/Right commands the robot to go at a specified angle and Down stops the robot. Additionally, the lever to the right of the D-pad allows the user to control the robot's speed. All of the D-pad commands can be set and modified in the settings page in the menu bar in the top right of the user interface. Also located in the menu bar are shortcuts to other pages such as 'Bluetooth', which is used to connect to a device and 'Data' used to display incoming data received from the centipede to the app. The data that would be displayed on this screen are items like status codes, temperature or light values, and anything else we might need.

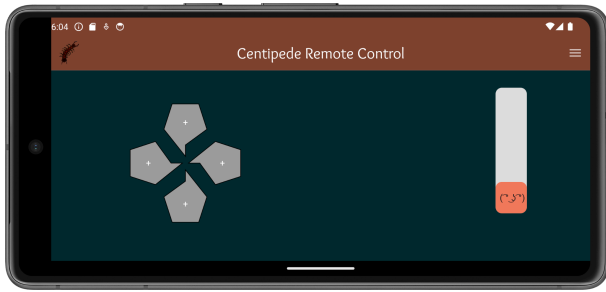


Fig. 10 Mobile app user interface

B. Serial Communication Protocol

Similar to the CAN protocol for intra-robot communication, the myriapod uses a serial protocol over Bluetooth to interface with the outside world. Given the text-based nature of this protocol, these commands were designed to be human readable and easy to craft in software, enabling custom applications to be made to control the robot. Each serial command corresponds to one or more CAN messages on the robot, meaning that low level functions like setting a single servo to a specific angle can be achieved without directly connecting to the robot's CAN bus. The "language" used by the robot for Bluetooth communication includes responses such as "OK" or "FAILED" to allow the programmer to easily see whether the command executed successfully as well as any data returned by the command. Currently, four commands (SET, WAVE, GET, and HOME) have been implemented to control the robot and any connected peripherals from the mobile app or any serial terminal via Bluetooth.

VII. PCB DESIGN

The printed circuit boards (PCBs) for the robotic centipede were carefully designed for smooth functionality and reliable performance. This section focuses on the schematic design, highlighting the selected microcontroller units (MCUs), power distribution, and key connections. Detailed datasheets from component vendors guided our component placement and trace routing strategy. Careful consideration of each component ensures optimal performance and functionality.

We designed two separate PCBs to replace the development hardware and combine most of the electrical components on each segment into one physical device. The first PCB is dedicated to the head segment (seen in Fig. 11) and serves as the central hub for components such as ToF sensors and the Bluetooth module. On the other hand, we have the body segment PCB (seen in Fig. 12), with five units distributed along

the centipede's segments. Despite labeling the PCBs as for the head and body, the head segment of the robot features both PCBs to enable legs to be attached to the head segment. These PCBs are stacked on top of each other, similar to the development boards on the 3D model in Fig. 3. Both PCB designs include serial wire debug (SWD) headers for programming and debugging the MCUs as well as two assignable LEDs and a reset button to assist in development. Additionally, both the 5 V and 3.3 V voltage rails feature status LEDs and test pads to simplify troubleshooting of any issues that may occur.

The head segment PCB features headers for the two ToF sensors, the Bluetooth module, and three servo motors. Unlike the body segment PCBs, the communication interfaces on the head segment PCB are intended for use with their specific peripherals and have been routed to dedicated connectors for those peripherals. While the two ToF sensors share a single I2C interface, a second I2C interface is exposed to facilitate connection with an external OLED display used to visualize the output of the ToF sensors to assist in debugging the obstacle avoidance software. Finally, while the original design only requires that the head segment controls a single servo to control the robot's turning angle, two additional headers were placed to support motorizing the neck joint between the head segment and the first body segment.

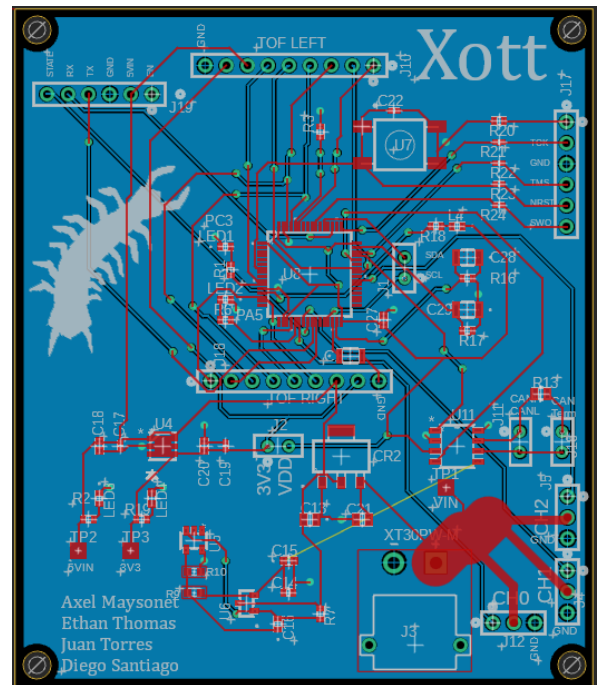


Fig. 11. Head segment PCB layout

Given the large power requirements of the leg servos, additional care was taken to ensure their operation would not interfere with the MCU in the body segments. We used decoupling capacitors placed close to the MCU to stabilize the voltage going to it and minimize interference from other components. An LD39050 voltage regulator maintains a stable 3.3V voltage rail for the main MCU and is consistent with ST's chosen regulator for their development board. Modifications were made to accommodate the centipede robot's relatively high current draw, namely the inclusion of 40 mil power traces going to each servo and a massive 250 mil trace to connect them to the power input and output ports. A power output port is passively connected to the input, enabling these PCBs to easily be daisy chained from a single power source. The MCU and other sensitive components were positioned as far away from the raw, high current components to reduce potential interference.

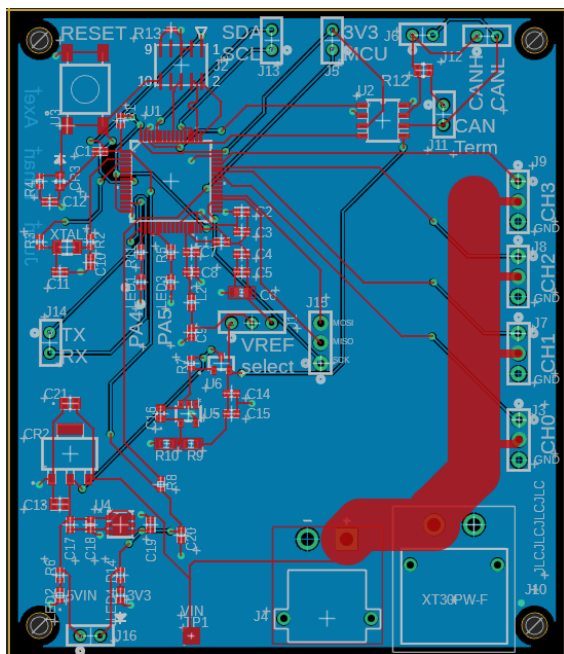


Fig 12. Body segment PCB layout

Each body segment PCB also features I2C, full-duplex SPI, and UART interfaces as well as the CAN interface necessary for the robot's core functionality. This suite of simultaneously available interfaces is a core part of this project's modularity and flexibility as a robot platform as these interfaces provide hardware-level compatibility with a wide array of sensors and peripherals.

VIII. CONCLUSION

The myriapod robot, drawing inspiration from the centipede, offers a strong legged robotic platform to be used as a foundation for more specialized tasks. This innovative structure allows the robot to navigate diverse terrains more effectively than conventional wheeled or tracked counterparts. The primary objective of this project was to demonstrate the myriapod robot's ability to autonomously traverse flat and uneven terrain, showcasing its modular capability. Each segment boasts the potential for dedicated sensors or devices, facilitating adaptability, and scalability. The envisioned quick-disconnect system aims to enhance the ease of use, offering a practical solution for adding or modifying segments after construction and in the field. The wireless teleoperation feature, controlled via a smartphone or other Bluetooth-capable device, adds a layer of versatility, making the robot suitable for both teleoperated and autonomous robotic applications.

Essential to the success of this project is the coordination among the centipede robot's segments, achieved through a metachronal wave, a synchronized, wave-like motion utilized by centipedes, realized by the microcontrollers on each segment. The implementation of a custom-designed PCB and 3D printed bodies for each segment, orchestrated by the head segment, enables the robot to execute forward, left, and right movements. This not only showcases the technical capabilities of the project but also emphasizes the potential for its practical application in real-world scenarios.

ACKNOWLEDGEMENT

The authors wish to acknowledge the assistance and support of Diego Santiago and Paulo Camasmie throughout the development of this project.

REFERENCES

- [1] Boston Dynamics, "Equip the right payload for the job", BostonDynamics.com, <https://bostondynamics.com/products/spot/payload/>
- [2] STMicroelectronics, "STM32 Cortex[®]-M4 MCUs and MPUs Programming Manual", https://www.st.com/resource/en/programming_manual/pm0214-stm32-cortexm4-mcus-and-mpus-programming-manual-stmicroelectronics.pdf
- [3] A. Matulich, "Metachronal waves of legs" *Numerical Analysis Better Left Unsaid*, <https://www.nablu.com/2022/12/metachronal-waves-of-legs.html>