SHARK: Stable Hoverboard-driven Autonomous Robot Kit

Nicolas F. Dias[1] Jardel D. S. Dyonisio[1] Luís F. M. Quadros[1] Gabriel A. Dorneles[1] Jardel J. P. Prado[1]
Marina Z. Rocha[1] Murilo C. Bicho[1] João F. S. S. Lemos[1] Henrique C. Bezerra[1] Cassio N. S. Teixeira[1]
Igor P. Maurell[1] Rodrigo S. Guerra[1] Paulo L. J. Drews Jr[1]

***Abstract*— This paper presents the SHARK low-cost robotic mobile base. It breaks down both the hardware and software components employed in its construction, alongside expenses breakdowns. Our results showed a cost-effective base with substantial payload capacity, capable of providing support for new teams and researchers looking to enter the mobile robotics field. All the development is open-sourced and available at: `https://github.com/butia-bots/shark_mb`.**

I. Introduction

In the past few years, there has been a notable increase in both interest and progress within the domain of domestic robotics. Concurrently, the RoboCup@Home Open Platform League has garnered attention from various teams globally. In light of this, we present the Stable Hoverboard-driven Autonomous Robot Kit (SHARK) (Figure 1), an open-source, open-hardware and low-cost robotic platform with the goal of empowering new researchers getting started in the mobile robotics field. Additionally, we aim to facilitate the development of various applications that can leverage this platform — whether in its default version or customized to meet specific requirements.

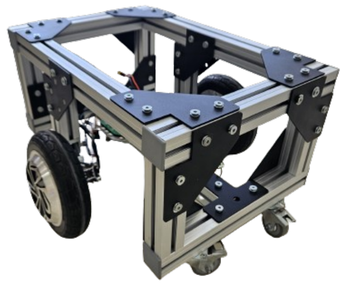


Fig. 1. SHARK platform.

In pursuit of this objective, we adapted the hardware commonly found in self-balancing scooters, commonly referred to as hoverboards. The mobile base is designed to support loads of up to 120kg and is integrated with ROS (Robot Operating System)[1], a suite of frameworks for robot development. The SHARK platform was inspired primarily by differential mobile bases that allow for mechanical simplicity, increased maneuverability, and easy adaptation, such as the PatrolBot[2] (Figure 2).

Fig. 2. Mobile robot PatrolBot. [Anisi and Thunberg, 2007]

Utilizing a hoverboard not only streamlines maintenance for the mobile base, thanks to the global availability of parts, but also addresses the challenges of traveling with batteries, since they are accessible at affordable prices from any location around the world. The financial cost for the hoverboard is also notably lower when compared to the independent acquisition of its components, mainly an ODESC-type brushless motor driver, motors, and their individual elements, including engines and batteries. A single hoverboard unit is priced at approximately $200, while the cost of its individual components amounts to around $278 if bought individually.

II. Methodology

A. Construction

To develop the platform, components from a hoverboard were used. The device contains most of the parts needed to build the robotic base, offers low cost, ease of maintenance and supports up to 120kg. A breakdown of the components and their quantities used in a hoverboard is provided in Table I.

A suspension system inspired by the PatrolBot was developed, providing improved stability, assisting in reducing wheel slippage, and allowing for more effective navigation through slightly uneven terrain. The system ensures more accurate odometry readings by keeping the wheels always in contact with the ground, providing more reliable data collection. The assembled suspension model can be seen in figure 5.

*This work was supported by the Robocup Federation

[1]Centro de Ciências Computacionais, Universidade Federal do Rio Grande, Brazil

[1]Robot Operating System `www.ros.org`.

[2]Patrolbot `https://www.generationrobots.com/media/PatrolBot-PTLB-RevA.pdf`

Item	ID	Figure
Aluminum frame	1	3
Main Board	2	3
Auxiliary board	3	3
Wheels whit Brushless Motor	4	3
Battery charger	5	4
36V battery	6	4

TABLE I
HOVERBORD COMPONENTS LIST

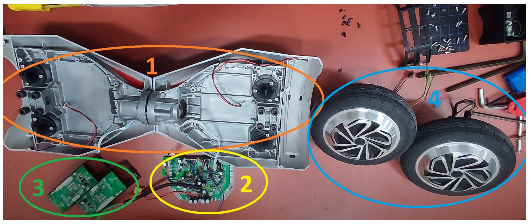


Fig. 3. Parts of a hoverboard.

The main component of the suspension system is the arm that connects the wheel to the column. The length of this arm is within the necessary limits to allow free movement and avoid collisions. In addition, the circular hole at the end of the arm must be the right diameter to fit the wheel axle, taking into account the desired distance for articulation.

Proper adjustment of the tightening is crucial to eradicate any lateral play within the joint, allowing the arm to articulate smoothly without encountering excessive resistance due to friction. This adjustment must be made before attaching the wheel, as the space between the joint and the wheel may become insufficient for the proper use of tools after complete assembly. This prior care ensures that the joint is properly adjusted before final installation, contributing to the good performance and durability of the suspension system.

As for the installation of the traction spring, it was necessary to use fasteners for the spring hook that could be securely tightened with T-nuts inserted into the profile rails. Due to the significant load imposed on the spring to support the desired payloads, 3D printing materials have proven to have extremely low mechanical strength. It is therefore important to make the fasteners out of metal to ensure the durability and proper performance of the system.

The steps described above were repeated to assemble the other end of the suspension, while positioning the motor shaft in the opposite direction. The complete suspensions are then attached to the upper part of the frame by means of steel connection plates, which also play the role of securing all the other parts of the frame.

To illustrate this process, the fixed structure has been subdivided in this example into two parts: an upper one, consisting of a rectangle where the drive wheels are attached, and a lower one housing the electronics and castors, formed by a central beam connected to four columns, both are shown in figure 6. However, due to the modular nature of the design, the base can be separated in ways that facilitate convenient transportation. This modular flexibility stands out as a feature of the design, allowing for different configurations to optimize transport and practicality in a variety of situations.

Fig. 4. Battery and charger.

Fig. 5. Suspension system.

The suspension adjustment process involves changing the position of the spring supports in order to alter the initial deformation of the spring. This change has a direct impact on the force generated by the spring, which in turn ensures that the lever effect provided by the articulated joint keeps the wheel in contact with the ground, applying the necessary pressure to ensure traction.

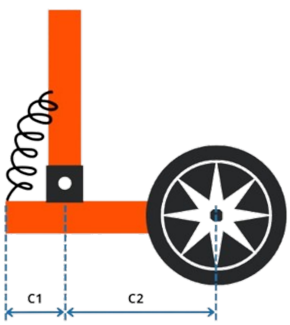
To determine the appropriate spring for the suspension, it is necessary to estimate the total weight of the robot and the space available around the spring. This information is important for calculating the force required and the maximum desired deformation of the spring, as can be seen in figure 7. In the Springs table, available on GitHub[3], it is possible to check springs and their k constants that may have adequate strength and dimensions to meet the project specifications.

B. Board configuration

The hoverboard has three boards inside, two of which are auxiliary boards that have telemetry instruments and a main board that has the power control, we only use the main board while discarding the others. Modifications to the firmware are important to adapt the board to the specific needs of the project. The firmware developed by EFeru [4] offers adjustments that allow the board to be controlled, enabling

[3]Springs table https://github.com/butia-bots/shark_mb/tree/main/Docs
[4]EFeru https://github.com/EFeru/hoverboard-firmware-hack-FOC

Fig. 6. Lower and upper parts.



$$F = \frac{\left(\frac{m \cdot g}{2}\right) \cdot C2}{C1 \cdot \sin(\theta)}$$

$$k = \frac{F}{\delta}$$

Fig. 7. Equations for determining the required force and elastic constant.

the implementation of specific functionalities and integration with the ROS environment. Only specific changes[5] to the proposed firmware were necessary to adapt it to the SHARK project.

The board of the self-balancing scooter is originally locked against modification, and to unlock it we used the STM32CubeProgrammer[6] software. After unlocking, the board's memory can be erased using the eraser option. Once this step has been completed, the board is ready to be rewritten with the adapted firmware.

To make it possible to write the new firmware using ST-LINK, jumper connectors had to be soldered to the SDW programming (3.3V, SWCLK, GND and SWDIO). It is important to note that firmware modifications are not universal and require specific microcontrollers, such as STM32 and GD32, to ensure compatibility.

The board used in SHARK has two communication cables, RX and TX, available on the left sideboard (USART 3), which only supports 3.3V, or on the right sideboard (USART 2), which only supports 5V. The choice between these options will depend on the FTDI module used. Figure 8 shows the components of the self balancing scooter board.

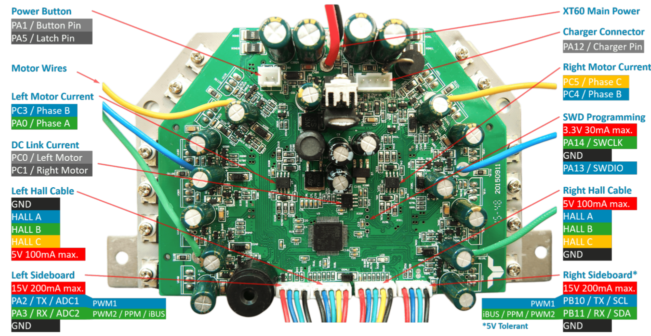


Fig. 8. Board components. Source: [Feru, 2019].

C. ROS integration

The SHARK MB is easily integrated with ROS Noetic and ROS2 Humble using the ROS package created by PaddyCube [7]. This package provides the necessary launch files to test and begin the autonomous navigation implementation.

Important topics such as '/cmd_vel' and '/odom' are available with this package. The '/cmd_vel' topic allows the transmission of speed commands to control the linear and angular movement of the robotic base. On the other hand, '/odom' provides odometry information, including position and orientation in relation to a reference frame. These topics are the base for the development of autonomous navigation systems, allowing the robot to perceive its environment and make informed decisions while navigating its surroundings.

In order to take advantage of the integration of the robotic base with the ROS middleware, a URDF[8] description (Figure 9 and 10) was developed, providing easy integration with ROS. This allows new sensors to be added or structural changes to be made quickly, providing the necessary kinematic structure for the navigation system. The virtual representation of the base can be viewed on Rviz[9], providing a tool for visualizing and analyzing configurations. The URDF model is also used and is taken into account for the creation of path routes, preventing collisions during traversal. In addition, an SDF model was created to be used in possible simulation scenarios.

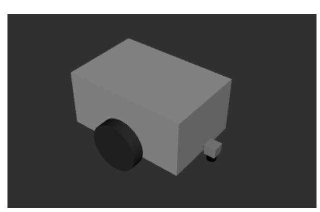


Fig. 9. URDF side view.

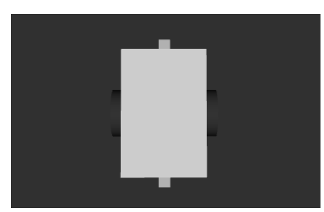


Fig. 10. URDF top view.

[5]SHARK firmware https://github.com/butia-bots/doris_base

[6]Software STM32CubeProgrammer https://www.st.com/en/development-tools/stm32cubeprog.html

[7]PaddyCube https://github.com/hoverboard-robotics/hoverboard-driver

[8]URDF http://wiki.ros.org/urdf

[9]Rviz http://wiki.ros.org/rviz

The creation of maps is an important component for enabling autonomous navigation in robots. Through the utilization of gmapping[10] package, the SHARK platform develops a comprehensive spatial representation of its environment (Figure 11), allowing path planning from one location to another. This capability allows the robot to navigate autonomously, as it can analyze the map to identify obstacles, plan optimal routes, and navigate through its surroundings.

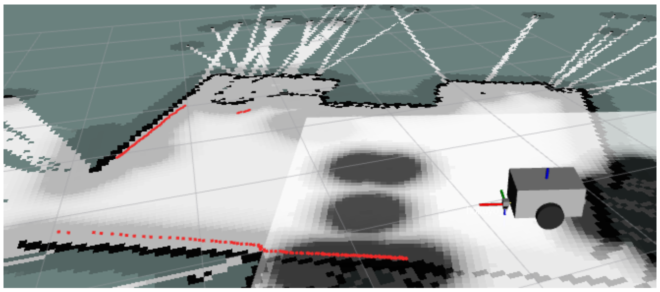


Fig. 11. SHARK MB in the environment.

Furthermore, the navigation aspect of the system is using the navigation stack[11]. This stack incorporates various algorithms and components, including but not limited to path planning, localization, and obstacle avoidance, to enable autonomous navigation. By integrating with the navigation stack, the SHARK platform can execute navigation tasks, dynamically adapting to changes in its environment while ensuring collision-free movement. Thus, the combination of environment mapping with gmapping and navigation capabilities using the ROS navigation stack empowers the SHARK platform with comprehensive autonomous navigation functionality.

Finally, the base allows a joystick to be attached[12], offering an intuitive and practical interface for controlling the robot's movement. This functionality allows direct and immediate interaction with the robot, facilitating its operation by human users and providing greater flexibility in various usage scenarios.

D. Cost Analysis

One of the strengths of the SHARK mobile base lies in its cost-effectiveness. In contrast to readily available commercial mobile robot platforms, the SHARK project prioritizes affordability, making it significantly cheaper for new researchers to acquire.

A simplified breakdown of the costs can be seen in Table II, revealing a significant cost advantage compared to commercially available mobile robot platforms. Figure 12 highlights the cost vs. payload capacity of various mobile bases. Our solution (red zone) stands out with a competitive payload capacity despite being significantly more affordable compared to other options. Notably, the SHARK design achieves this affordability by leveraging readily available hoverboard components, minimizing the number of components and utilizing open-source software.

Item	Subtotal (USD)
Structural material	1,053.65
Electronic material	686.95
Total	1,740.60

TABLE II

BREAKDOWN OF THE COSTS

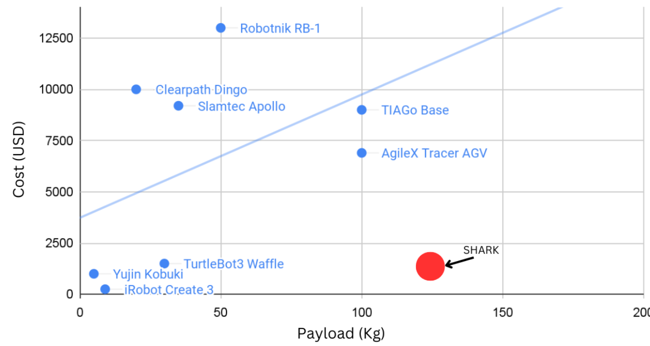


Fig. 12. Cost vs payload for some popular mobile bases.

III. CONCLUSIONS

This paper details the SHARK project, which developed a low-cost mobile robot base leveraging hoverboard technology. The project aimed to streamline entry for new teams in the RoboCup@Home category and accelerate research in related fields.

The SHARK base integrates effectively into domestic environments while offering easy adaptation for various mobile robot projects. You can explore these applications and possibilities on the project's GitHub repository[13].

The SHARK project allowed existing technologies to be incorporated into hoverboards, speeding up the development process and making it more economically viable. In addition, the maintenance of the base proved to be simple, since spare parts for hoverboards are easily found due to the widespread presence of these devices. The simplicity of the structure of the mobile base, with few components, also contributes to ease of maintenance. The SHARK project has therefore achieved its initial objectives, delivering a low-cost base with good payload capacity and providing support for new teams looking to enter the domestic robot category.

REFERENCES

[Anisi and Thunberg, 2007] Anisi, D. and Thunberg, J. (2007). Survey of patrolling algorithms for surveillance ugvs.

[Feru, 2019] Feru, E. (2019). hoverboard-firmware-hack-foc. https://github.com/EFeru/hoverboard-firmware-hack-FOC.

[10] Gmapping ROS package http://wiki.ros.org/gmapping

[11] Navigation Stack ROS package http://wiki.ros.org/navigation

[12] SHARK MB with joystick https://youtube.com/shorts/z7LHLPvpSBA

[13] SHARK MB GitHub https://github.com/butia-bots/shark_mb