ExoskeletonXEEG

Ain Shams University
College of Engineering
Mechatronics Department

GRADUATION PROJECT PROPOSAL

Project tite		Exoskeleton x EEG	
Supervisor		DR. Mohamed Ibrahim MohamedHassan Awad DR. Mohamed Mahmoud Hamdy Mohamed abdel Hafiz	
Number of student	7	Student Qualification	Bachelor's degree

Team Members

Name	Email	Phone	CV
Eyad Hatem	eyad.elgendy@techademics.org	01115468026	Eyad Hatem CV
Mahmoud Magdy	Mahmoud.mag78@gmail.com	01158273074	Mahmoud MagdyCV
Habiba Mohammed	HABIBA.MO.Salem@gmail.com	01202270777	Habiba MohamedCV
Selvia Talaat	selviatalat52@gmail.com	01224245487	Selvia Talaat CV
Youssef Osama	Youssef.osama.2607@gmail.com	01157063640	youssef osama CV
Asmaa Mohamed	**asmaasokar19@gmail.com**	**01127915788**	Asmaa mohamed CV

Student skills required

- ROS
- Integrating MATLAB/Simulink with Ros
- For EEG sensor
  Signal Processing: Understanding how to filter and analyze EEG signals, including techniques for noise reduction and artifact removal.

- Programming: Proficiency in programming languages such as Python, MATLAB, or C/C++ for data acquisition and analysis.

- Electronics: Knowledge of circuit design and electronics, especially related to interfacing sensors with microcontrollers or embedded systems.

- Mechatronics Design: Familiarity with mechatronics principles to integrate EEG sensors into robotic or automated systems.

- Machine Learning: Basic understanding of machine learning techniques for pattern recognition and classification of EEG data.

Introduction

The ExoMotus orthotically fits to the lower limbs and part of the upper body and is intended to enable individuals with lower limb disabilities to perform and restore routine ambulatory functions such as standing, sitting and
walking, which increases the patient's physical activity level.
ExoMotus User weith up to 100kg and height ranges between 150 cm and 190 cm.

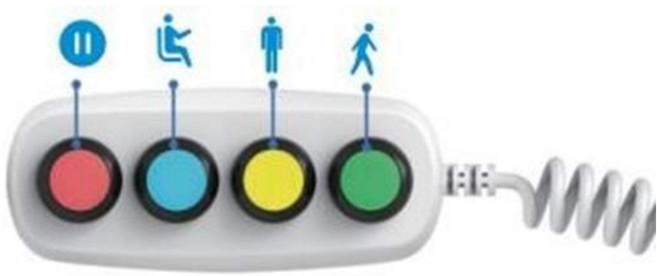
ExoMotus consists of 3 main components:

- ExoSkeleton
- Backpack
- Hand ControL

Objective and description

The ExoMotus currently relies on a remote control system, which has four buttons:

- Green Button: Makes the exoskeleton walk.
- Red Button: Stops the exoskeleton.
- Blue Button: Makes the exoskeleton sit.
- Yellow Button: Allows the exoskeletonto stand.



This system aims to transition the exoskeleton from state to state
We intend to change the control system to make it easier and smoother for individuals with disabilities or those wearing the device. We will use EEG sensors to achieve this goal, as follows:

1. Brain Signal Detection: EEG sensors are placed on the scalp to capture brain signals.
2. Signal Processing: These signals are processed to identify specific patterns associated with different mental commands or intentions.
3. Command Interpretation: The system translates these patterns into commands that control the exoskeleton.
4. Exoskeleton Movement: The exoskeleton responds to the commands and executes the desired actions.

AND We will use impedance control to develop the system and make it suitable for individuals with reduced mobility, not just for those with disabilities. This development aims to enhance the device's responsiveness and flexibility, providing a better experience for users

Tools and methods

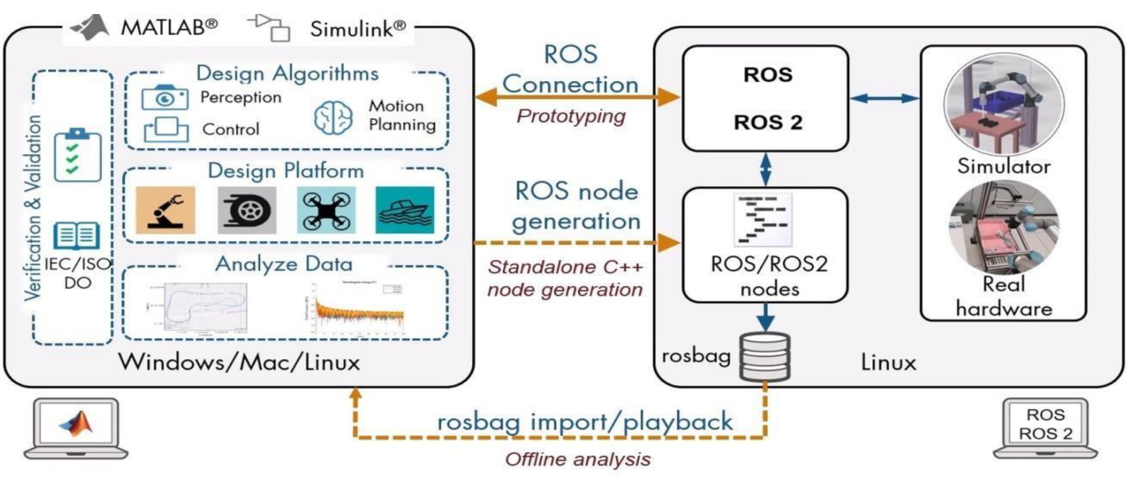
1-Integrating MATLABSIMUlINK with ROS

1- ROS Toolbox Overview

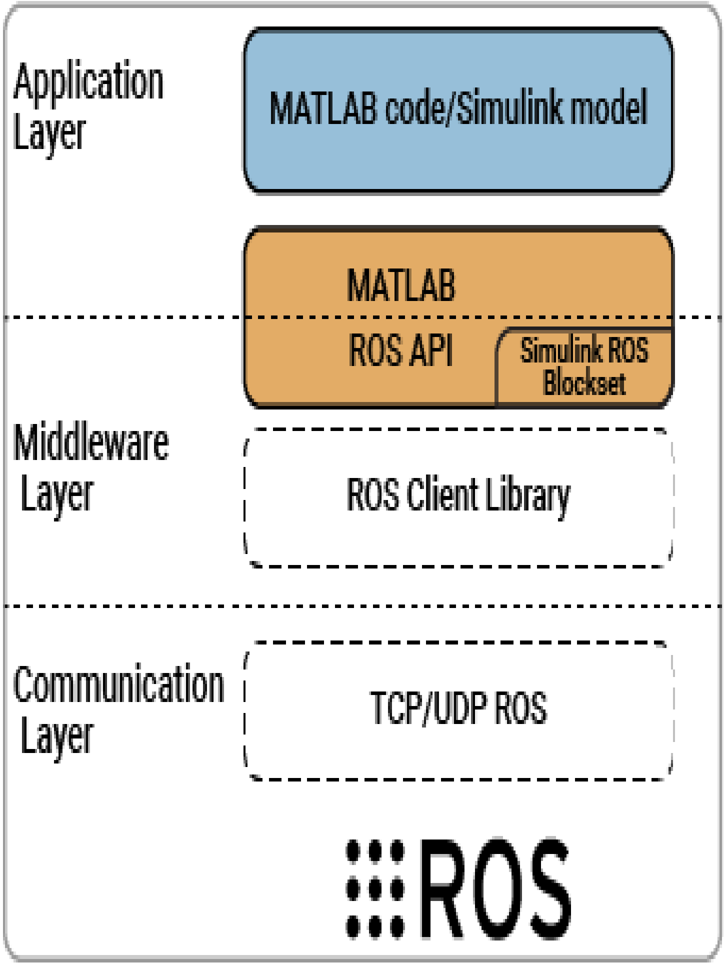
- The ROS Toolbox provides an interface that connects MATLAB® and Simulink® with the Robot Operating System (ROS and ROS 2). It allows users to design networks of ROS nodes and integrate MATLAB or Simulink-generated ROS nodes with existing ROS networks.

Key features include:

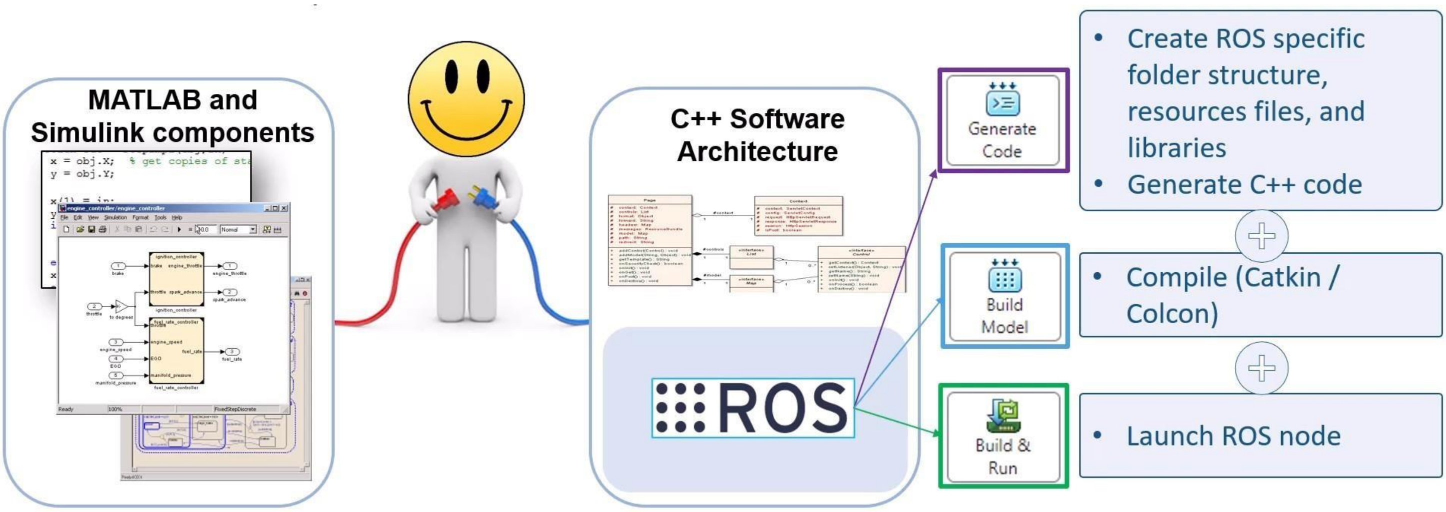
- **Data Visualization and Analysis**: Users can visualize and analyze ROS data by recording, importing, and playing back rosbag files, as well as connecting to a live ROS network to access messages.
- **Verification of ROS Nodes**: The toolbox supports desktop simulation and connections to external simulators like Gazebo and physical hardware for verifying ROS nodes.
- **Code Generation**: It enables C++ and CUDA® code generation through MATLAB Coder™, Simulink Coder, and GPU Coder™, allowing automatic generation of ROS nodes from MATLAB scripts or Simulink models for deployment on simulated or physical hardware.
- **Simulink External Mode**: This feature allows users to view messages and modify parameters while the model is running on hardware.



2- ROS Integration in MATLAB and Simulink



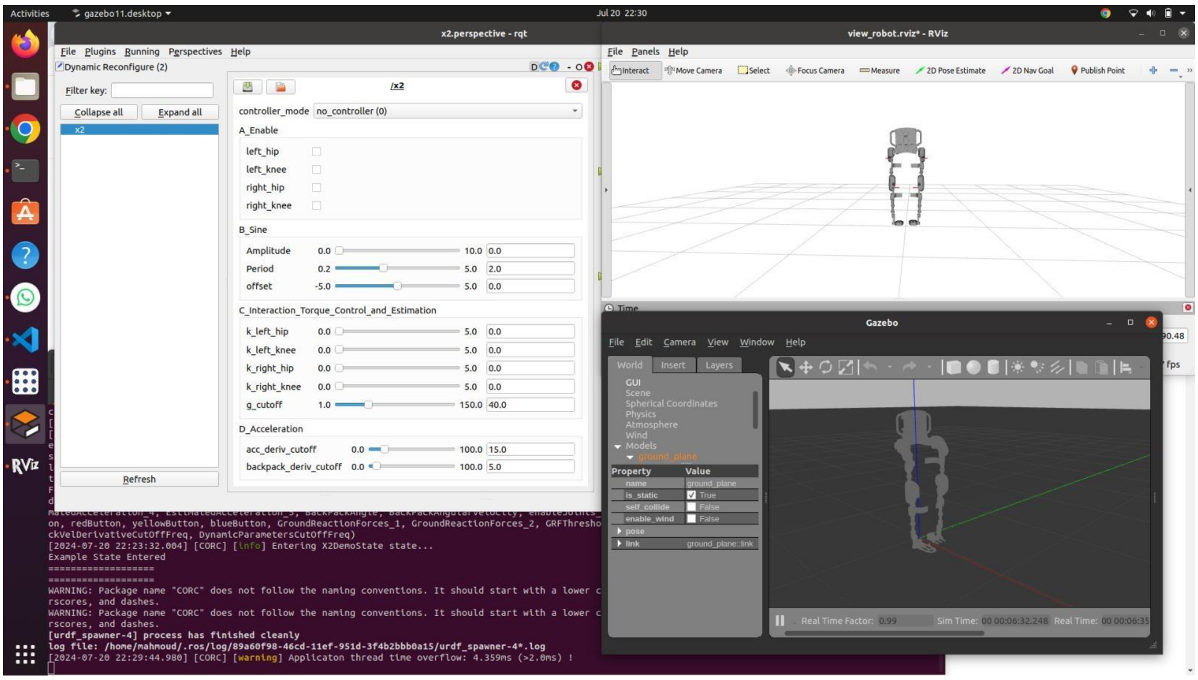
- **Application Layer**: MATLAB and Simulink provide APIs and blocksets that allow users to integrate custom code and models. This enables the effective use of ROS capabilities within these environments.
- **Middleware Layer:** Based on DDS, the middleware layer serves as a bridge between the application layer and the communication layer. It includes client libraries and APIs to facilitate communication with the ROS system.
- **Communication Layer:** The communication layer utilizes DDS as the underlying protocol, managing data exchange between nodes in a distributed system.



2-Using Ros

- Robot Operating System (ROS) is an open-source middleware framework designed to simplify robotic application development. It offers tools, libraries, and conventions for building and managing robotic software in a clustered computing environment.

  <u>Key Advantages of ROS</u>:
- **Modularity and Reusability**: Developers can create reusable components called "nodes," enabling focus on specific tasks without needing to understand the whole system.
- **Rich Ecosystem**: ROS has a vast collection of libraries, tools, and packages contributed globally, facilitating development through pre-built functionalities for navigation, perception, control, and simulation.
- **Interoperability**: Supporting multiple programming languages (primarily C++ and Python), ROS allows easy integration of components and communication between heterogeneous systems.
- **Simulation Tools**: Tools like Gazebo enable developers to simulate robots in a 3D environment, reducing risks and costs by allowing algorithm testing without physical robots.
- **Community Support and Collaboration**: An active community provides extensive forums, documentation, and educational resources, helping both new and experienced developers share knowledge and collaborate on projects.
- **Support for Hardware Abstraction**: ROS enables applications to work across various hardware interfaces with minimal modification to the codebase.
- **Real-time Capabilities**: Although not inherently a real-time system, ROS can be integrated with real-time computing solutions for applications requiring strict timing, such as autonomous vehicles and industrial robots.



3-EEG sensor

what is EEG sensor and its evolution

- EEG was mainly used as a diagnostic tool in neurology and psychiatry. For example, to detect abnormal brain activity or to characterize epileptic seizures. For such applications, EEG systems were usually stationary and optimized for high signal quality using conductive gels and gold or silver electrodes.

- Driven by miniaturization and advances in wireless technology, portable EEG-amplifiers were developed that allow now for mobile recordings outside the laboratory. For example, 32-channel recordings are now possible with EEG amplifiers that can be attached to the user’s head and carried around.

- To reduce preparation time, dry electrodes that typically use metallic spike arrays54 and solid gel55 electrodes were introduced that do not require gels or fluids. However, these electrodes require direct contact with the scalp and often come with lower signal quality.56 Another alternative to the conventional wet electrodes uses saline soaked sponges (eg, R-net by Brain Products, waveguard net by ANT Neuro, or GT Cap Gelfree-S3 by Greentek) that also reduce preparation time and require no hair washing after application.

How EEG works

- Brainwave Detection: EEG sensors are place brain signals.

- Signal Processing: These signals are proces patterns associated with different mental c Command Interpretation: The system transl commands that control the exoskeleton.

- Exoskeleton Movement: The exoskeleton res executing the desired actions

4-Impedance control

- Impedance Control is a control strategy used in robotics to manage the interaction between a robot and its environment. It focuses on regulating the dynamic relationship between the force exerted by the robot and its motion. This method is particularly useful in tasks requiring compliance and adaptability, such as in robotic manipulation and human-robot interaction.

1. Impedance

- In the context of robotics, impedance refers to the relationship between the motion of the robot and the forces it experiences. It can be thought of as a combination of mass (inertia), damping (resistance to motion), and stiffness (elasticity). Impedance can be adjusted to achieve desired interactions with the environment.

2. Force and Position Control

- Impedance control blends aspects of force and position control. Instead of simply moving to a position (position control) or applying a force (force control), it allows a robot to adjust its position in response to external forces. This enables the robot to be compliant when needed (e.g., when encountering unexpected obstacles) while still being capable of precise movements.

3. Dynamic Interaction

- This control strategy is particularly advantageous in dynamic environments where robots must adapt to changing conditions, such as when collaborating with humans or interacting with soft or deformable objects.

Advantages

- Flexibility: Adapts to different tasks and environments.
- Safety: Reduces the risk of collisions and damage during interaction with humans or fragile objects.
- Natural Interaction: Enables more intuitive human-robot collaboration

5-NVIDIA Jetson Xavier: Overview and Capabilities

The NVIDIA Jetson Xavier is a robust AI computing platform tailored for autonomous machines and embedded applications, making it ideal for developers, researchers, and institutions in fields like robotics, drones, and smart cameras.

Key Specifications:

Processing Power: Features an 8-core ARM CPU and a 512-core Volta GPU, enabling over 11 trillion operations per second (TOPS).

Memory and Storage: Includes 32 GB LPDDR4x RAM and 32 GB eMMC storage, expandable via microSD.

Connectivity: Supports multiple interfaces such as USB 3.1, HDMI 2.0, PCIe, allowing integration with diverse sensors.

Power Efficiency: Operates at a maximum of 30 watts, suitable for power-sensitive applications.

Software Support: Runs on the JetPack SDK, compatible with frameworks like TensorFlow and PyTorch.

Capabilities:

AI and Deep Learning: Executes complex deep learning models for computer vision, speech, and natural language processing.

Robotics: Supports advanced capabilities like SLAM, obstacle detection, and path planning.

Multi-Sensor Fusion: Processes data from various sensors, enabling efficient and safe operation of autonomous systems.

Real-Time Performance: Handles real-time processing needs, crucial for responsive applications.

Advantages:

Scalability: Flexible for projects ranging from small robotics to large autonomous vehicles.

Performance per Watt: Delivers high performance while maintaining low power consumption, ideal for edge computing.

Future-Proofing: Regular updates from NVIDIA ensure compatibility with evolving AI technologies and frameworks.

Overall, Jetson Xavier stands out as a powerful and flexible platform for deploying advanced AI applications, making it a valuable asset in various technological advancements.

Time plan

Milestone 1 MATLAB Fundamentals
Milestone 2 Simulink Basics
Milestone3 Introduction to ROS
Milestone 4 Advanced ROS Concepts
Milestone 5 Simulation Framework
Milestone 6 Integration with ROS
Milestone 7 Review and Presentation
Milestone 8 Control Theory Fundamentals
Milestone 9 Control Algorithm Development
Milestone 10 Integration of Control Algorithms
Milestone 11 Documentation and Review of Control Algorithms
Milestone 12 Introduction to Machine Learning
Milestone 13 Machine Learning Concepts
Milestone 14 EEG Technology
Milestone 15 EEG data analysis
Milestone 16 Control strategies with EEG
Milestone 17 Comprehensive System Testing
Milestone 18 Final Documentation
Milestone 19 Presentation Preparation
Milestone 20 Final Project Presentation