IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

ANDROID CONTROL ROBOT

[1]P. Boopathi, [2].M. Sakthi Sanjay, [3] Dr. M. Jaithoon BiBi
[3]Assistant Professor
Department of Computer Science with Cognitive Systems,
Sri Ramakrishna College of Arts and Science,
Coimbatore.

INTRODUCTION:

The integration of mobile technology and robotics has paved the way for innovative applications in various domains, ranging from industrial automation to entertainment. This journal presents the design and implementation of an Android-controlled robot, utilizing a combination of Java for the Android application and C++ for the robot's firmware. The project aims to provide a comprehensive guide for enthusiasts and researchers interested in developing similar systems, offering insights into the integration of these two programming languages for effective robot control.

The introduction sets the stage by discussing the increasing significance of mobile-controlled robotic systems in modern society. It outlines the motivation behind the project, highlighting the need for seamless communication between Android devices and robots, and introduces the objectives and scope of the journal.

Background and Related Work: This section provides a comprehensive overview of existing literature and related projects in the field of Android-controlled robotics. It discusses the various approaches, technologies, and challenges associated with such systems, laying the groundwork for the proposed solution.

System Architecture: The system architecture section describes the overall design of the Android-controlled robot, including hardware components, software modules, and communication protocols. It illustrates how Java and C++ are integrated to facilitate bidirectional communication between the Android application and the robot.

Android Application Development: This section delves into the development of the Android application responsible for controlling the robot. It covers aspects such as user interface design, Bluetooth connectivity, and real-time data transmission, showcasing the implementation details using Java programming language.

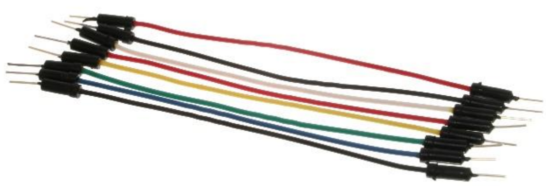
Robot Firmware Implementation: Here, the focus shifts to the firmware development for the robot using C++. It discusses the design of control algorithms, sensor interfacing, motor control, and communication protocols required to receive commands from the Android application and execute them on the robot.

Integration and Testing: Integration and testing are crucial phases in the development process. This section elaborates on how the Android application and the robot firmware are integrated and validated through rigorous testing procedures, ensuring the reliability and robustness of the system.

- Results and Discussion: The results section presents the performance evaluation of the Android-controlled robot, including latency analysis, response time, and overall user experience. It discusses the strengths, limitations, and potential improvements of the proposed solution.

Conclusion and Future Work: The journal concludes by summarizing the key findings and contributions of the project. It also outlines potential avenues for future research and development in the field of Android-controlled robotics, such as incorporating machine learning techniques for autonomous navigation or enhancing the user interface for better user experience.

JUMPER WIRE:



Jumper wires are essential components in electronics and robotics projects. They are used to create connections between various components on a breadboard, circuit board, or between different modules. Jumper wires typically consist of a flexible insulated wire with metal pins or sockets at each end, allowing for easy insertion into breadboard holes or connection points on electronic components.

These wires come in various lengths, colors, and configurations to accommodate different project requirements. Common types include:

1. Male-to-Male (M-M): Both ends of the wire have pins, suitable for connecting two female headers or sockets.
2. Male-to-Female (M-F): One end has pins, while the other end has sockets, enabling connections between male headers and female headers or sockets.
3. Female-to-Female (F-F): Both ends have sockets, used for extending or connecting female headers or sockets.

In this project we are using Male-to-Male jumper wire

RESISTOR:

resistor is an electronic component used to limit the flow of electric current in a circuit. Resistors are crucial in electronics for controlling voltage and current levels, protecting components from damage, and shaping signal waveforms.

- Resistance **Value**: The resistance of the resistor is 180 ohms, meaning it impedes the flow of electric current by 180 ohms when a voltage is applied across it.

- Color **Code**: Resistors often use a color code to denote their resistance value. Each color corresponds to a digit, and the resistance value is determined by the sequence of colors on the resistor body. For example, a resistor with a resistance of 180 ohms might be color-coded with bands in brown (1), gray (8), and brown (0), with an additional gold (±5%) or silver (±10%) band indicating the tolerance.

 Power **Rating**: Resistors also have a power rating, which indicates how much power they can safely dissipate without overheating or getting damaged. For a 180-ohm resistor, the power rating specifies the maximum power (in watts) the resistor can handle without exceeding its temperature limits.

 Applications: 180-ohm resistors find applications in various electronic circuits, such as LED current limiting, voltage dividers, biasing circuits, and signal conditioning.

 Availability: 180-ohm resistors are readily available in through-hole and surface-mount packages, making them suitable for both breadboard prototyping and PCB (Printed Circuit Board) assembly.

USB CABLE:

1. **Power Supply**: The USB cable can provide power to the robot's control system, such as microcontrollers, sensors, and other electronic components. This is especially useful during development and testing phases when the robot is connected to a computer or a USB power source.
2. **Data Communication**: The USB cable facilitates bidirectional data communication between the Android device and the robot's control system. Through USB communication, commands generated by the Android application can be sent to the robot for execution, while sensor data and feedback from the robot can be transmitted back to the Android device for monitoring and processing.
3. **Programming and Debugging**: During the development process, the USB cable may be used for programming and debugging the firmware running on the robot's microcontroller or embedded system. It allows developers to upload code changes, troubleshoot issues, and monitor the robot's behavior in real-time using development tools and software.
4. **Firmware Updates**: The USB connection enables firmware updates to be applied to the robot's control system, allowing for enhancements, bug fixes, and new features to be deployed without requiring physical access to the hardware.
5. **Charging**: If the robot incorporates a rechargeable battery, the USB cable can also serve as a means for charging the battery when connected to a power source, ensuring uninterrupted operation.

Overall, the USB cable plays a critical role in facilitating power supply, data communication, programming, debugging, firmware updates, and charging in an Android-controlled robot project using Java and C++. It serves as the primary interface between the Android application running on the mobile device and the embedded control system of the robot.

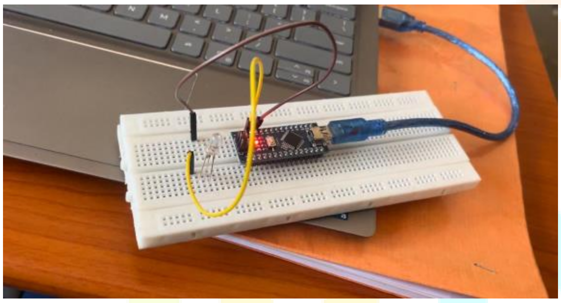
TYPES OF USB CONNECTOR:

1. **Power Supply**: The USB cable can provide power to the robot's control system, such as microcontrollers, sensors, and other electronic components. This is especially useful during development and testing phases when the robot is connected to a computer or a USB power source.
2. **Data Communication**: The USB cable facilitates bidirectional data communication between the Android device and the robot's control system. Through USB communication, commands generated by the Android application can be sent to the robot for execution, while sensor data and feedback from the robot can be transmitted back to the Android device for monitoring and processing.
3. **Programming and Debugging**: During the development process, the USB cable may be used for programming and debugging the

firmware running on the robot's microcontroller or embedded system. It allows developers to upload code changes, troubleshoot issues, and monitor the robot's behavior in real-time using development tools and software.

4. **Firmware Updates**: The USB connection enables firmware updates to be applied to the robot's control system, allowing for enhancements, bug fixes, and new features to be deployed without requiring physical access to the hardware.
5. **Charging**: If the robot incorporates a rechargeable battery, the USB cable can also serve as a means for charging the battery when connected to a power source, ensuring uninterrupted operation.

Overall, the USB cable plays a critical role in facilitating power supply, data communication, programming, debugging, firmware updates, and charging in an Android-controlled robot project using Java and C++. It serves as the primary interface between the Android application running on the mobile device and the embedded control system of the robot.



This is the breadboard connected with Arduino nano this two-device connected with another device with a help of a usb connector. And jumper plays a main role in this device.

Software tools Arduino:

Arduino development typically involves a combination of software tools for writing, compiling, and uploading code to Arduino microcontroller boards. Here are some essential software tools commonly used in Arduino development:

Arduino IDE (Integrated Development Environment): The Arduino IDE is the official software tool for Arduino development. It provides a simple and user-friendly interface for writing, compiling, and uploading Arduino sketches (programs) to Arduino boards. The IDE includes a text editor with syntax highlighting, a compiler, and a serial monitor for debugging and communication with the Arduino board.

Arduino CLI (Command Line Interface): Arduino CLI is a command-line tool that provides a way to compile, upload, and manage Arduino sketches from the terminal or command prompt. It offers more flexibility and automation options compared to the Arduino IDE, making it suitable for integration into automated build systems and development workflows.

Visual Studio Code with Arduino Extension: Visual Studio Code (VS Code) is a popular open-source code editor with a rich ecosystem of extensions. The Arduino Extension for VS Code integrates Arduino development capabilities directly into the editor, providing features such as code completion, syntax highlighting, and project management. It offers a more modern and customizable development environment compared to the Arduino IDE.

Platform IO: Platform IO is an open-source ecosystem for IoT development that supports multiple development platforms, including Arduino. It provides a unified development environment with support for various microcontroller boards, libraries, and frameworks. Platform IO offers advanced features such as multi-platform development, library management, and debugging tools, making it suitable for complex Arduino projects.

Atmel Studio: Atmel Studio is an integrated development platform specifically designed for programming Atmel microcontrollers, including those used in Arduino boards. While it's more commonly used for professional development with Atmel's AVR microcontrollers, it can also be configured to work with Arduino-compatible boards, offering advanced debugging features and optimization options.

Eclipse IDE with Arduino Plugin: Eclipse is a versatile open-source IDE widely used for software development in various programming languages. The Arduino Plugin for Eclipse extends its capabilities to support Arduino development, providing features such as code editing, project management, and debugging. It's suitable for developers familiar with Eclipse who prefer a more customizable development environment.

Existing systems:

Bluetooth Control: One common approach is to use Bluetooth communication between the Android device and the Arduino Nano board. Android applications can be developed to send commands over Bluetooth to the Arduino, which then interprets these commands to control the robot's movements, sensors, or other functionalities.

HC-05/HC-06 Bluetooth Module: These modules are commonly used for Bluetooth communication between Arduino and Android devices. There are existing Arduino libraries and Android applications that facilitate Bluetooth communication and provide an interface for controlling the robot.

Wi-Fi Control: Another option is to use Wi-Fi communication for controlling the robot. Wi-Fi modules such as the ESP8266 or ESP32 can be used with the Arduino Nano to enable wireless communication. Android applications can send commands to the Arduino Nano over Wi-Fi using protocols such as TCP/IP or UDP.

ESP8266/ESP32: These modules are widely used for Wi-Fi communication and can be programmed using the Arduino IDE. There are existing libraries and examples available for integrating these modules with Arduino Nano and developing Android applications for Wi-Fi control.

IoT Platforms: Some IoT platforms provide frameworks and tools for building Android-controlled robots with Arduino Nano boards. These platforms often include cloud connectivity, mobile app development frameworks, and device management capabilities.

Blynk: Blynk is an IoT platform that provides a drag-and-drop interface for building mobile apps to control Arduino-based projects. It supports a wide range of Arduino boards, including the Arduino Nano, and provides libraries for integrating with various communication protocols such as Bluetooth and Wi-Fi.

Custom Android Applications: Developers can also create custom Android applications using programming languages such as Java or Kotlin to communicate with the Arduino Nano board. This approach offers flexibility and customization but requires more development effort.

Android Development: Android developers can use Android Studio to develop custom applications that communicate with the Arduino Nano over Bluetooth, Wi-Fi, or other communication interfaces. Libraries such as Android or Retrofit can be used to simplify communication tasks.

Proposed system:

For a proposed system for Android-controlled robots using an Arduino Nano board, let's outline the key components and functionalities:

Hardware Components:

Arduino Nano: The Arduino Nano serves as the central control unit of the robot, interfacing with various sensors, actuators, and communication modules.

Motor Drivers: Motor drivers control the movement of the robot's motors, allowing it to navigate its environment. Depending on the robot's configuration, motor drivers such as L298N or L293D can be used.

Sensors: Sensors such as ultrasonic sensors, infrared sensors, or gyroscopes can be integrated to provide feedback about the robot's environment.

Wireless Module: A communication module, such as a Bluetooth module (e.g., HC-05/HC-06) or Wi-Fi module (e.g., ESP8266/ESP32), enables wireless communication between the Arduino Nano and the Android device.

Android Application:

Develop an Android application to act as the user interface for controlling the robot.

The application should have an intuitive interface with controls for driving the robot (forward, backward, left, right), as well as options for additional functionalities like sensor readings, camera feed (if applicable), and settings.

Implement Bluetooth or Wi-Fi communication protocols in the Android application to establish a connection with the Arduino Nano board.

Incorporate features for sending control commands (e.g., motor speed and direction) from the Android device to the Arduino Nano and receiving feedback data from the robot's sensors.

Arduino Sketch:

Write an Arduino sketch to run on the Arduino Nano board, defining the behavior of the robot based on the commands received from the Android application.

Implement code to interpret incoming commands from the Android device and control the robot's motors accordingly.

Integrate sensor reading functionalities to gather data from onboard sensors and transmit relevant information back to the Android device for display or further processing.

Integration and Testing:

Integrate the hardware components, Arduino sketch, and Android application.

Test the system thoroughly to ensure proper communication between the Android device and the Arduino Nano, as well as reliable control and feedback mechanisms.

Conduct real-world testing to evaluate the robot's performance in various scenarios and environments, making adjustments as necessary to optimize functionality and user experience.

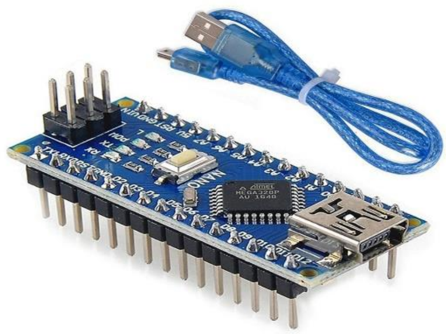
Documentation and User Guide:

Provide comprehensive documentation outlining the system architecture, hardware setup, software implementation, and usage instructions for both the Android application and the Arduino Nano.

Create a user guide to assist users in setting up and operating the Android-controlled robot, including troubleshooting tips and best practices.

Arduino Nano boards are compact, versatile microcontroller platforms that are widely used in electronics projects due to their small size, affordability, and ease of use. Here are some common uses of Arduino Nano boards.

Arduino Nano board:



Prototyping: Arduino Nano boards are often used for prototyping electronic circuits and systems. Their small form factor and breadboard-friendly design make them ideal for experimenting with sensors, actuators, and other electronic components.

DIY Electronics Projects: Hobbyists and makers frequently use Arduino Nano boards to create a wide range of DIY electronics projects. These projects can include LED displays, temperature and humidity sensors, motion-activated lights, and more.

Robotics: Arduino Nano boards are popular choices for building robotic projects, including small wheeled robots, robotic arms, and drones. They can control motors, servos, and sensors, making them suitable for a variety of robotic applications.

Home Automation: Arduino Nano boards can be used to create home automation systems that control lights, appliances, and other devices. By interfacing with relays, sensors, and wireless modules, Arduino Nano-based home automation systems can provide remote monitoring and control capabilities.

IoT (Internet of Things): Arduino Nano boards can be integrated into IoT projects to monitor and control physical devices over the internet. They can communicate with cloud platforms and other IoT devices using protocols such as MQTT, HTTP, and WebSocket.

Data Logging and Monitoring: Arduino Nano boards can be used to collect data from sensors and log it to an SD card or transmit it wirelessly to a computer or cloud server. This data can then be analyzed and visualized to gain insights into environmental conditions, energy usage, and more.

Educational Projects: Arduino Nano boards are commonly used in educational settings to teach students about electronics, programming, and robotics. They provide a hands-on learning experience that helps students understand concepts such as circuits, sensors, and control systems.

Wearable Electronics: Due to their small size and low power consumption, Arduino Nano boards are well-suited for wearable electronics projects. These projects can include smart watches, fitness trackers, and other wearable devices that collect data and interact with the user.

CONCLUTION:

Arduino Nano boards offer a compact, affordable, and versatile platform for a wide range of electronics projects. Their small form factor and breadboard-friendly design make them ideal for prototyping and experimentation, while their capabilities extend to more complex applications such as robotics, home automation, IoT, and wearable electronics.

Arduino Nano boards empower hobbyists, makers, educators, and professionals to bring their creative ideas to life, providing a user-friendly platform for learning, exploring, and innovating in the field of electronics and embedded systems. With a vast ecosystem of sensors, actuators, shields, and accessories, Arduino Nano boards offer endless possibilities for innovation and creativity.

Whether used in educational settings to teach students about electronics and programming, in DIY projects to create custom electronic devices, or in commercial applications to prototype new products, Arduino Nano boards continue to be a popular choice among electronics enthusiasts worldwide.

In summary, Arduino Nano boards play a crucial role in democratizing electronics and empowering individuals to turn their ideas into reality, contributing to the vibrant maker community and fostering innovation in the field of embedded systems.

REFERENCE:

- Arduino. (n.d.). Arduino Nano. Retrieved from https://www.arduino.cc/en/Guide/Arduino Nano

 Make: DIY Projects and Ideas for Makers. (n.d.). Arduino Nano - Overview and Applications. Retrieved from https://makezine.com/2009/10/06/arduino-nano-overview-and-appl/

 Instructables. (n.d.). Arduino Nano Projects. Retrieved from https://www.instructables.com/circuits/howto/arduino+nano/

 Exploring Arduino. (2013). Turner, J. Wiley.

 Arduino Project Handbook: 25 Practical Projects to Get You Started. (2016). McGowan, M., & Schembri, B. No Starch Press.

Dr. M.Jaithoon Bibi, Assistant Professor, Sri Ramakrishna College of Arts and Science, Coimbatore. She received her M.Phil in the field of Data Mining from Sri Ramakrishna College of arts and Science for Women, Coimbatore. Completed PhD in the field of Data mining from PSGR Krishnammal College for women, Coimbatore. Her area of interest is data analysis and predictive models to reveal patterns and trends in data from exiting data sources. Her teaching experience is 4 year. She has published 11 research papers in international journals, Scopus, Web of Science and conferences in the area of data mining. She attended various international level conferences, seminars, workshops and technical symposium. She is a member of IAENG(International Association of Engineering. She have authored a books on Infrastructure Management MCQ's, API Creation: For Artificial Intelligence and Data Story telling with Tableau Public.