Tele-Operated Micro:bit & Arduino Vehicle Prototype

Members

1. **Aangir Doshi (BT2024078)**
2. **Thrissha Arcot (BT2024024)**
3. **Abhyudaya Singh (BT2024180)**
4. **Ayush Patel (BT2024171)**
5. **Utkarsh Rastogi (BT2024119)**
6. **Shive Bhat (BT2024067)**
7. **Vraj Vashi (BT2024062)**

Course: Computer Architecture
Prof: Karthikeyan Vaidyanathan

Abstract

This report details the design and implementation of a low-cost remote-driving vehicle using two BBC micro:bit v2 boards for tilt-based control and an Arduino Mega for motor actuation and obstacle avoidance. One micro:bit reads accelerometer data to derive left/right motor speeds, sends these wirelessly to a second micro:bit, which relays them over UART to the Arduino. Four HC-SR04 ultrasonic sensors mounted on the vehicle provide real-time obstacle detection, enabling the Arduino to override remote commands when necessary. The system demonstrates a complete sensor-to-actuator loop and serves as a foundation for more advanced teleoperation projects.

---

1. Introduction

Remote operation of vehicles promises safer, more efficient transport of goods and services, especially where human presence is hazardous or impractical. This prototype explores the core pipeline—human input → wireless link → processing → actuation → safety override—

with off-the-shelf microcontrollers and sensors. The aim is to validate system logic and provide a hands-on learning platform for teleoperation fundamentals.

---

2. System Overview

- **Control Node (Handheld):** micro:bit v2 with accelerometer, running Python code on the micro:bit Python Editor.
- **Relay Node (Vehicle Receiver):** second micro:bit v2 receiving radio packets, forwarding via UART to Arduino Mega.
- **Processing & Actuation Node:** Arduino Mega reads commands and sonar sensors, drives two DC motors via Adafruit Motor Shield.
- **Safety Layer:** Four HC-SR04 ultrasonic sensors detect obstacles front/left/right/back and trigger immediate overrides.

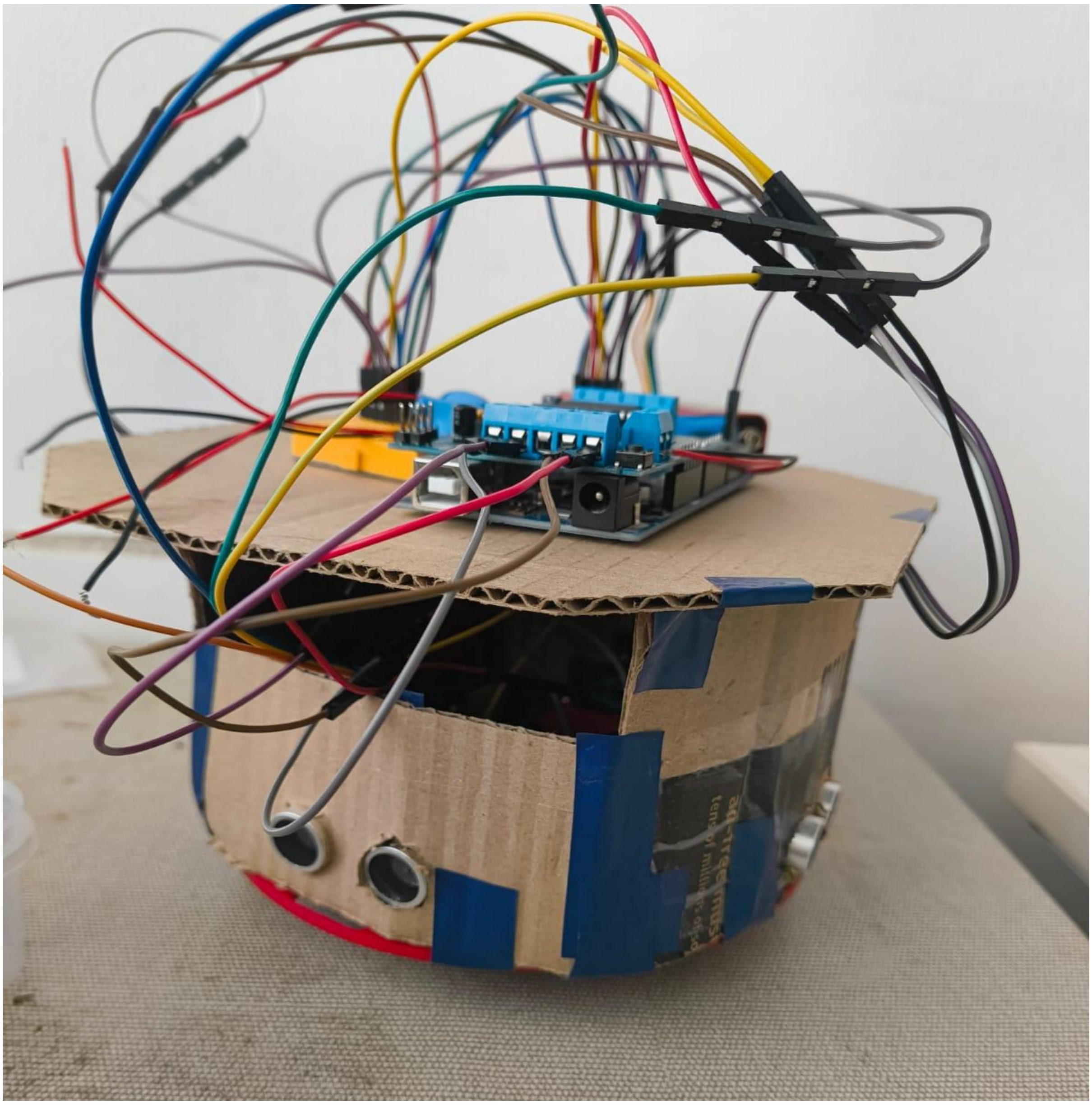
---

3. Hardware Components

1. **BBC micro:bit v2 (×2)**
   - In-built 3-axis accelerometer
   - 2.4 GHz radio (group 1, power 7)
2. **Arduino Mega 2560**
   - UART interface (Serial1 at 9600 baud)
   - Digital I/O for sonar triggers/echoes
3. **Adafruit Motor Shield v1** (AFMotor library)
   - Two DC motors wired to ports M1 (left) and M2 (right)
4. **HC-SR04 Ultrasonic Sensors (×4)**
   - Front: TRIG 28 / ECHO 29
   - Left: TRIG 24 / ECHO 25
   - Back: TRIG 22 / ECHO 23
   - Right: TRIG 26 / ECHO 27
5. **Power Supply**
   - Two 9V batteries for the motors . microbits powered by the use battery packs , Arduino powered by connecting to laptop.

4. Wiring Description

- **Tilt-Controller micro:bit → Radio:** powered via USB battery pack.
- **Receiver micro:bit P0 → Arduino Mega pin 19 (Serial1 RX):** single-wire UART connection.
- **Motor Shield → Arduino Mega:** stacked on headers.
- **Ultrasonic Sensors:** each sensor's TRIG pin to a distinct digital pin, ECHO pins to corresponding inputs.
- **Motors:** connected to Shield ports M1 and M2 for left and right wheels.
- **Ground Commoning:** all devices share a common GND.



5. Software Implementation

5.1 Tilt-Remote Code (micro:bit)

1. **Configuration Constants:**
   - Dead-zone ±400, zone thresholds ±800
   - SPEED_LOW = 70, SPEED_HIGH = 100
2. **Radio Setup:**

   python

   ```
   radio.on()
   radio.config(group=1, power=7)
   ```
3. **zone_speed(val) Function:** maps raw accelerometer tilt (–1024…+1024) to discrete speeds {0, ±70, ±100}.
4. **Main Loop (every 10 ms):**
   - Read accelerometer.get_x(), .get_y()
   - Apply dead-zone filtering
   - Compute xs, ys via zone_speed
   - Determine LSpeed & RSpeed through if/else blocks handling straight, pivots, and arcs
   - radio.send("{},{}".format(LSpeed, RSpeed))

5.2 Relay Code (micro:bit)

1. **Radio Receive & UART Forward:**

   python

   ```
   packet = radio.receive()
   if packet:
   x_str, y_str = packet.split(",")
   uart.write(f"{x_str} {y_str}\n")
   ```
2. **UART initialized on pin0 at 9600 baud**

5.3 Arduino Logic

1. **Setup:**
   - Serial1.begin(9600)
   - Motor shield initialized (motor_L, motor_R)
   - Sonar pins pinMode(trig, OUTPUT) / pinMode(echo, INPUT)
2. **readUltrasonicCM(trig, echo)** triggers sensor, measures pulse, converts to cm.
3. **Main Loop (~100 Hz):**
   - Read all four distances
   - If Serial1.available(): parse cmdLeftSpeed, cmdRightSpeed
   - **Obstacle Avoidance:**
     - If front < 20 cm → pivot or reverse based on side/back clearance
     - Else, if turning would drive into too-close side, straighten or pivot opposite
     - If reversing into obstacle, zero speeds
   - **Motor Drive:**

     cpp

     ```
     motor_L.run( (L>=0)? FORWARD : BACKWARD );

     motor_L.setSpeed(abs(L));
     ```
   - delay(5);

---

6. Testing & Observations

- **Functional Check:** vehicle responds to tilt: forward/back and left/right arcs.
- **Safety Test:** sonar-triggered overrides successfully prevented collisions in static obstacle course.
- **Feedback:** found controls intuitive; minor jitter when tilt near dead-zone thresholds.

---

7. Contributions

- **Hardware Integration:** Aangir Doshi, Thrissha Arcot, Vraj Vashi, Abhyudaaya Singh
- **Tilt-Control Algorithm:** Ayush Patel, Utkarsh Rastogi, Shive Bhat.

- **Obstacle Avoidance Logic:** Aangir Doshi, Abhyudaya Singh, Thrissha Arcot, AI models.

Guidance from seniors: Nitheezkant R, Ishan Jha.

---

8. Conclusion

This project demonstrates a simple yet complete teleoperated vehicle pipeline—from human tilt input through wireless transmission, microcontroller processing, to motor actuation with real-time safety overrides. It serves as a robust foundation for future enhancements, such as richer feedback (video streaming) or lower-latency links (5G modules).

Appendix

- **Full Code Listings:** see attached tilt_remote.py , relay_node.py , vehical_controller.ino .
- **Component Datasheets:** HC-SR04, AFMotor library documentation.