**Explanation of States**

1. **State 0.** “Dead”
   1. This is the state that the system is initialized into.
   2. A read LED shines.
   3. Motors are off.
   4. When the GUI is started, everything is greyed out, and the only option is to ‘initialize’ to begin.
2. State 1. “Primed/Armed/Stopped”
   1. This state is the ‘home base’ for the system.
   2. After ‘initialization,’ this state is entered.
   3. The motors are off.
   4. A yellow light shines.
   5. You can enter this state from all other states by clicking the ‘Stop’ button.
   6. When an object is detected, the system automatically enters this state.
   7. The system stays in this state until a button is clicked.
3. State 2. “Turn Left”
   1. To enter this state, click the ‘Turn Left’ button in the GUI.
   2. This state is *timed* and operates the motors.
   3. The pins connected to the h-bridge are set to cause ‘left’ angular rotation of the robot.
   4. After a period of time, the state is exited and returns to the previous state.
4. State 3. “Turn Right”
   1. To enter this state, click the ‘Turn Right’ button in the GUI.
   2. This state is *timed* and operates the motors.
   3. The pins connected to the h-bridge are set to cause ‘right’ angular rotation of the robot.
   4. After a period of time, the state is exited and returns to the previous state.
5. State 4. “Move Forward”
   1. To enter this state, click the ‘Forward’ button in the GUI.
   2. The h-bridge pins are set to rotate both motors in the same direction to move the robot forward.
   3. The ultrasonic sensor frequently records measurements; if measurements are less than a threshold, the system moves to the ‘Stop’ state to shut off the motors. Object avoidance is implemented in this state only.
   4. To manually exit this state, click any other button in the GUI.
6. State 5. “Move Backward” / “Reverse”
   1. To enter this state, click the ‘Reverse’ button in the GUI.
   2. The h-bridge pins are set to rotate both motors in the same direction to move the robot backward.
   3. To exit this state, click any other button in the GUI.

Graphical user interface, table

Description automatically generated with medium confidence

**Commands**

The ‘Manual Control’ section of the GUI is responsible for sending commands that move between states in the Arduino. Clicking a button switches to the state number shown on each button.

**4**

**5**

**3**

**2**

**1**

**0**

The radio buttons for speed only update a PWM pin value and do not cause a change of state.

Graphical user interface, table

Description automatically generated with medium confidence**Code In Action**

The GUI contains three main sections:

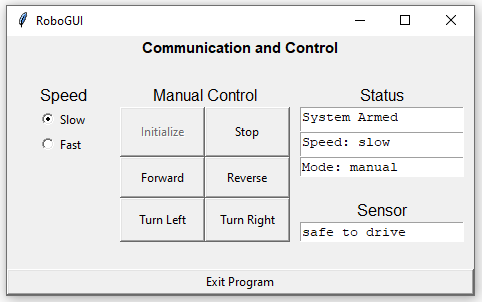
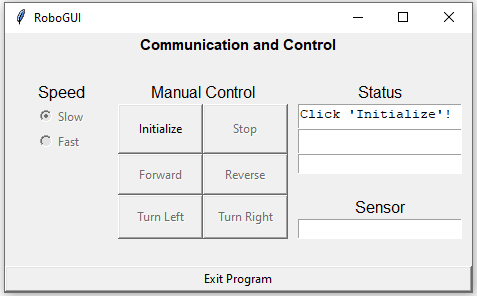
* Speed setting (left)
* Manual Control (center)
* Statuses (right)

The speed setting can be switched between ‘slow’ and ‘fast,’ which alters the PWM value sent to the motors.

The manual control buttons send commands via serial connection to the Arduino to switch between states.

The ‘Status’ and ‘Sensor’ section are used purely as means to display information to the user and are automatically updated based on system configurations and events that occur.

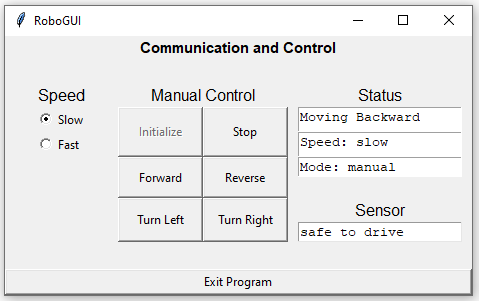
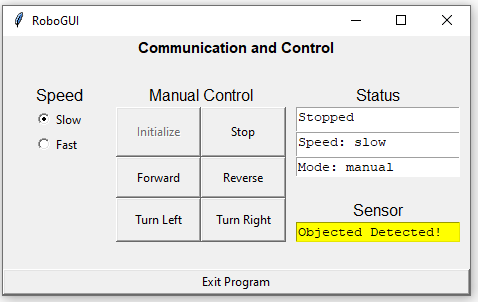
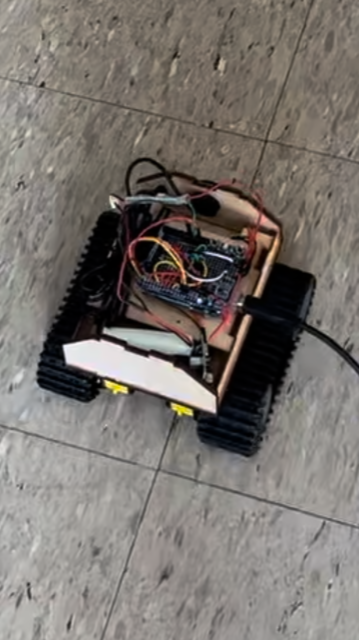
When the GUI is opened, all functionality is blocked until ‘Initialize’ is clicked. Then, regular operation begins.



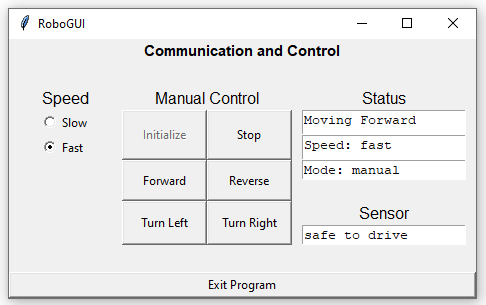
Graphical user interface, table

Description automatically generated with medium confidenceClicking different buttons in the ‘Manual Control’ section sends commands via serial connection to the Arduino, which switches the system case and updates the digital output pins that connect to the h-bridge for motor control. The GUI updates accordingly.

When the robot encroaches upon an object, the ultrasonic sensor detects it and stops the robot. The GUI updates.



The user is in control and can click any button in the GUI to free the robot from its ‘stopped’ state. Note that if the user tries to go forward, the robot will again see the object and quickly stop. Hence, backing up (reverse) is advised.



The speed of the robot can be adjusted. Click ‘slow’ or ‘fast’ sends a command to the Arduino via serial comms and sets the PWM value for the motor (the Pin that goes to the PWM input on the h-bridge).