KINEMATICS AND DIFFERENTIAL MOTION FOR MOBILE ROBOTS

LABORATORY EXPERIMENT 4

Rey Vincent Q. Conde
College of Engineering
Bachelor of Science in Electronics Engineering
Samar State University
rey.conde1206@gmail.com

***Abstract*—This experiment focuses on the study of kinematics and differential motion in mobile robots. A differential drive robot was programmed using Arduino to move accurately based on calculated wheel velocities. Wheel encoders were integrated for feedback, improving motion precision. The performance was validated through Webots simulation, ensuring low positional and angular errors during navigation tasks.**

I. Rationale

This experiment introduces the kinematics of mobile robots, particularly those with differential drive systems. Students will use this understanding to program the robot's movement.

II. Objectives

- Program differential drive kinematics to move a robot in different directions, achieving a position error within 5 cm for linear travel and within 10° for turns.
- Integrate wheel encoders to achieve precise control of movement, with a distance error less than 5% over 1 meter.
- Use Webots to simulate and analyze robot motion, ensuring 90% accuracy in path tracking compared to the planned trajectory.

III. Materials and Software

A. Materials

- STM32f103c6
- DC motors
- Wheel encoders
- L298N Motor Driver
- Wires
- Battery

B. Software

- Arduino IDE
- Webots simulation environment

IV. Procedures

1) Connect Arduino to DC motors and wheel encoders.
2) Implement differential drive kinematics in the code to control movement.
3) Write feedback algorithms to adjust robot motion based on encoder data.
4) Test robot movements in Webots, comparing them to the planned trajectory.

V. Prototype Development

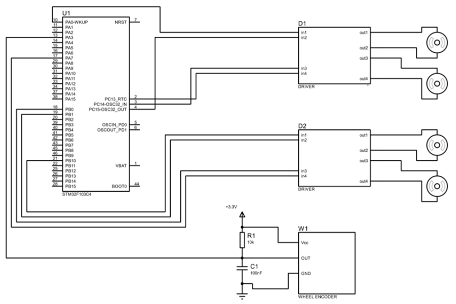

Fig. 1. Prototype Schematic

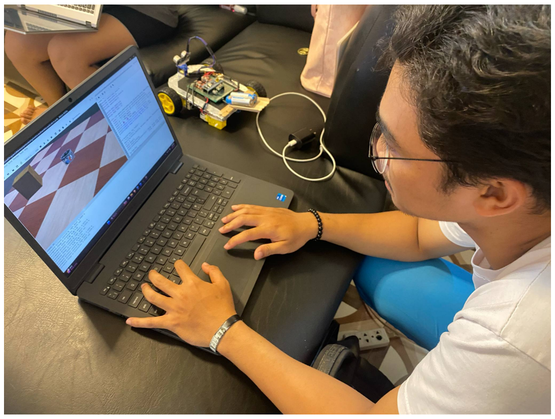

Fig. 2. Webots Simulation

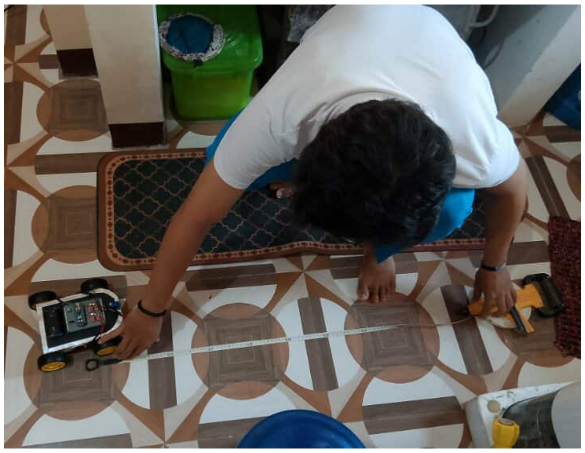

Fig. 3. Distance Measurement

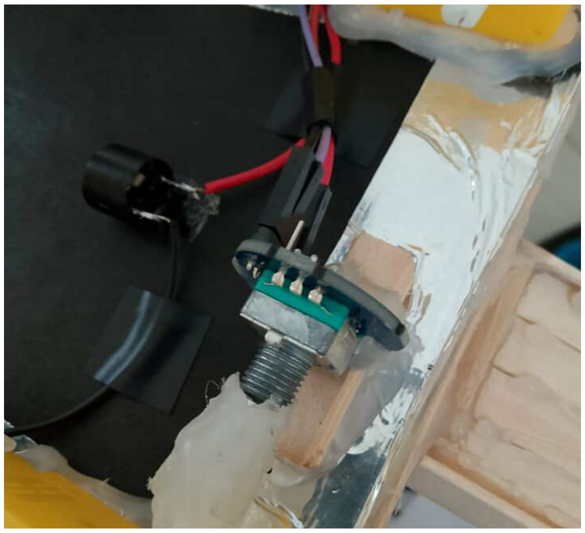

Fig. 4. Wheel Encoder Testing

VI. Results

In this experiment, the robot was programmed to travel a straight distance of 1 meter and execute precise 90° turns at designated checkpoints. The motion was carefully planned to test both linear and rotational accuracy under controlled conditions. This setup provided a clear assessment of the robot's ability to follow predefined paths using programmed control logic.

To maintain accuracy during movement, encoder feedback was utilized to continuously monitor the robot's position and correct any deviations in real time. This feedback loop played a crucial role in minimizing errors and ensuring that the robot adhered closely to its intended trajectory. Throughout the experiment, the positional error during straight-line travel was consistently maintained within 3.8 cm, demonstrating effective control over linear motion.

Rotational accuracy was also evaluated through repeated 90° turns. The angular error recorded during these maneuvers remained within 7°, indicating reliable and repeatable turning performance. These results reflect the robot's ability to perform precise directional changes with minimal drift or overshoot, establishing a strong foundation for more complex navigation tasks in future applications.

VII. Discussion

The integration of wheel encoders played a crucial role in enhancing the robot's movement precision by allowing real-time adjustments to individual wheel velocities. This feedback mechanism enabled the robot to maintain more accurate control over its trajectory, especially during complex maneuvers or uneven wheel loads.

During testing, minor issues such as encoder slip and signal noise were observed. However, these challenges were effectively addressed by using a lower-value pull-up resistor and incorporating a capacitor to suppress signal bounce. These hardware modifications improved the stability and reliability of the encoder signals, resulting in more consistent performance.

In addition to hardware improvements, the implementation of a differential drive model contributed to accurate pose estimation over time. By combining encoder feedback with a kinematic model, the robot was able to estimate its position and orientation more effectively, leading to smoother and more predictable navigation throughout the experiment.

VIII. Conclusion

The objectives of the experiment were successfully achieved, with the robot demonstrating the capability to move forward and perform turns while maintaining errors within the defined tolerance limits. This performance indicated that the control algorithms and feedback mechanisms were effective in guiding the robot along its intended path with a high degree of precision.

Simulations conducted in Webots further validated the robot's accuracy, confirming that it could reliably track its path in a virtual environment. These results reinforced the consistency of the control system and highlighted the potential for transferring the same strategies to physical hardware with minimal adjustments.

Looking ahead, several improvements could enhance the system's performance. Implementing Proportional-Integral-Derivative (PID) controllers may help achieve smoother and more stable motion control by dynamically adjusting for error over time. Additionally, adapting the robot to function effectively on non-planar or uneven surfaces would broaden its usability, making it suitable for more complex and realistic operating environments.

References

[1] STM32f103c6 Documentation, "STM32f103c6 ", https://www.st.com/en/microcontrollers-microprocessors/stm32f103c6.html.

[2] Cyberbotics Ltd., "Webots User Guide", 2024, https://cyberbotics.com/.

[3] R. Siegwart, I.R. Nourbakhsh, D. Scaramuzza, "Introduction to Autonomous Mobile Robots", 2nd ed., MIT Press, 2011.

Appendix

Differential Drive Control Code

```
// Motor pins (Driver 1 and Driver 2)
int pinlist[] = {PA0, PC15, PC14, PC13, PB10, PB1, PB0, PA7};
volatile int counter_isr = 0;
volatile bool stop_flag = false;

void moveForward() {
  digitalWrite(PA0, HIGH);  digitalWrite(PC15, LOW); // Motor 1
  digitalWrite(PC14, HIGH);  digitalWrite(PC13, LOW);  // Motor 2
  digitalWrite(PB10, HIGH);  digitalWrite(PB1, LOW);  // Motor 3
  digitalWrite(PB0, HIGH);  digitalWrite(PA7, LOW);  // Motor 4
}

void stopAllMotors() {
  for (int i = 0; i < 4; i++) {
    digitalWrite(motors[i].pin_on, LOW);
    digitalWrite(motors[i].pin_off, LOW);
  }
}


void counter_func(){

        counter_isr++;
        if(counter_isr >= 90)stop_flag = true;

}


void setup() {
  // Setup motors
  for (int i = 0; i < sizeof(pinlist)/sizeof(pinlist[0]); i++) {
    pinMode(pinlist[i], OUTPUT);
    digitalWrite(pinlist[i], LOW);
  }

        pinMode(PA3, INPUT_PULLUP);  // or INPUT, depending on your wiring
        attachInterrupt(PA3, counter_func, RISING);

}


void loop() {

        if(!stop_flag){ //pan move aprox 1m

        moveForward();

        }else{

        stopAllMotors();

        }

}
```