LABORATORY EXPERIMENT 4

KINEMATICS AND DIFFERENTIAL MOTION FOR MOBILE ROBOTS

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. RESULTS

In this experiment, the robot was programmed to move forward a distance of 1 meter and to perform precise 90° turns at designated points. Encoder feedback was used throughout the motion to monitor and correct deviations. As a result, the positional error during straight-line movement was consistently maintained within 3.8 cm, ensuring relatively accurate linear travel. Similarly, during rotational maneuvers, the angular error was limited to within 7°, indicating that the robot was able to achieve reliable and repeatable turns with minimal drift or overshoot.

VI. DISCUSSION

The integration of wheel encoders significantly enhanced the robot's movement precision by enabling real-time adjustments to individual wheel velocities. Although minor issues such as encoder slip and measurement noise were encountered, they were effectively mitigated by using a lower-value pull-up resistor and adding a capacitor to suppress signal bounce. These measures improved the reliability of encoder readings. Furthermore, implementing a differential drive model allowed for accurate estimation of the robot's pose over time, contributing to more consistent and predictable navigation.

VII. CONCLUSION

The objectives of the experiment were successfully achieved. The robot demonstrated the ability to move forward and execute turns while maintaining errors within the specified tolerance limits. Additionally, simulations conducted in Webots validated the robot's high path-tracking accuracy. For future improvements, the implementation of Proportional-Integral-Derivative controllers could be explored to achieve even smoother and more robust motion control. Further development could also focus on adapting the system to operate effectively on non-planar surfaces or uneven terrains, expanding its applicability to more complex 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[] = {PB1, PA10, PA9, PA8, PB9, PB8, PB7, PB6};
volatile int counter_isr = 0;


void moveForward() {
  digitalWrite(PB1, HIGH);  digitalWrite(PA10, LOW); // Motor 1
  digitalWrite(PA9, HIGH);  digitalWrite(PA8, LOW);  // Motor 2
  digitalWrite(PB9, HIGH);  digitalWrite(PB8, LOW);  // Motor 3
  digitalWrite(PB7, HIGH);  digitalWrite(PB6, 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++;

}


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() {

        while(counter_isr < 91){ //pan move aprox 1m
                moveForward();
        }
        stopAllMotors();

}
```