**Straight Line Accuracy: Testing Robot Motion Methods**

**Introduction:**

One common challenge in robotics and autonomous vehicles is maintaining a straight path of motion. A variety of factors can affect a robot’s ability to drive straight, including:

1. The type of terrain
2. Uneven surfaces, gravel, or small obstacles that can cause wheels to slip
3. Differences in power applied to each wheel
4. Inconsistencies in motor performance
5. Inaccuracies or limitations in sensor feedback

To improve straight-line motion, robots often rely on sensors and control techniques that help detect and correct deviations from the intended path. Some of the key tools used include:

1. **Wheel encoders** – to track how far each wheel has turned
2. **Inertial Measurement Units (IMUs)** – to measure acceleration and angular drift
3. **Compasses** – to detect changes in heading
4. **Cameras** – to assess orientation relative to the surrounding environment

In this activity, you will experiment with different methods of driving a robot forward and compare how accurately each approach maintains a straight path over a one-meter distance.

**🧠 Objective:**

Test and compare the accuracy of three different methods for driving a robot forward ***one meter***.

Observe how timing-based control compares to encoder feedback and full PID-based motion control.

**📏 Instructions:**

1. On the floor or table, **mark a start line** and measure exactly **1 meter** straight ahead. Mark the **1-meter goal line** clearly with tape.
2. For each method below, run **3 trials** and record the **distance traveled** by the robot in centimeters. Measure from the front of the robot to the goal line.
3. After completing all trials, compare the results and answer the analysis questions.

**🧪 Method 1: Time-Based Drive**

**Code behavior:** Drive at a fixed speed for a fixed time (e.g. 3 second at half-speed).

|  |  |  |
| --- | --- | --- |
| **Trial** | **Distance Reached (cm)** | **Notes (e.g. veering, overshoot)** |
| 1 |  |  |
| 2 |  |  |
| 3 |  |  |
| **Average** |  |  |

**🧪 Method 2: Encoder-Based Drive**

**Code behavior:** Drive until either the left or right encoders reach 6000 ticks.

|  |  |  |
| --- | --- | --- |
| **Trial** | **Distance Reached (cm)** | **Notes (e.g. left/right drift, slowdown)** |
| 1 |  |  |
| 2 |  |  |
| 3 |  |  |
| **Average** |  |  |

**🧪 Method 3: Full Motion Controller**

**Code behavior:** Use the PID controller to drive for 6000 ticks

|  |  |  |
| --- | --- | --- |
| **Trial** | **Distance Reached (cm)** | **Notes (e.g. stability, course correction)** |
| 1 |  |  |
| 2 |  |  |
| 3 |  |  |
| **Average** |  |  |

**📝 Summary Questions:**

1. **Which method was the most accurate (closest to 100 cm)?**
2. **Which method was the most consistent (least variation between trials)?**
3. **What kind of issues did you observe (drifting, overshooting, stopping short)?**
4. **Why do you think Method 3 was more/less reliable than the others?**

**🧪 Lab 4: Determine the Input for One Meter**

**Objective:** Choose one of three movement methods above that resulted in the straightest path and most predictable travel distances. Use that method to find the value needed to travel as close to **1 meter** as possible.

**✅ Steps:**

1. Choose one method:  
   ☐ Method 1 – Time-Based Drive  
   ☐ Method 2 – Encoder Ticks  
   ☐ Method 3 – Full Motion Controller
2. Based on your previous data, **estimate what input** (time, ticks, etc.) would result in a 1-meter movement.
3. Test your input and record how far the robot actually travels.
4. Adjust your input **if needed** and repeat until you're as close to a meter as possible

**📋 Data Table:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attempt** | **Input Used (time/ticks/etc.)** | **Distance Reached (cm)** | **Adjustment Made?** | **Notes** |
| 1 |  |  | ☐ Yes ☐ No |  |
| 2 |  |  | ☐ Yes ☐ No |  |
| 3 |  |  | ☐ Yes ☐ No |  |
| 4 |  |  | ☐ Yes ☐ No |  |