Robotics Club Summer Project

Three Wheel Drive System with Odometry

Report by Arittra Malhotra

June 23, 2025

1 Introduction

We intend to build a three-wheel drive system, which has various applications such as mobile robotics, autonomous platforms, and omnidirectional transport systems. To control the robot's movement manually, we are using a remote controller and receiver setup, allowing wireless input for direction and speed. For smooth and accurate motion, we implement a PID (Proportional-Integral-Derivative) controller, which helps fine-tune motor responses and minimize error during movement.

In the second phase of the project, we plan to integrate an odometry system, which will track the robot's position and orientation in real time using data from wheel encoders or motor feedback. To support omnidirectional movement, we will be using omniwheels, enabling the robot to move laterally and rotate simultaneously without changing its orientation.

2 Materials and Methods

Part One: Three-Wheel Drive System

For the drive system, we employed the following components:

- **Omniwheels:** Three omniwheels were used to enable omnidirectional movement, allowing the robot to move in any direction without changing its orientation.
- **DC Motors with 2-Channel Encoders:** Each wheel is driven by a DC motor equipped with a 2-channel encoder to provide feedback for closed-loop control.
- **Sabertooth 2x25 Motor Driver:** This dual-channel motor driver was used to drive two of the motors efficiently, with high current capability and smooth acceleration control.
- **Kangaroo Motion Controller:** Used in conjunction with the motor driver to implement PID control, enabling precise speed and position regulation based on encoder feedback.
- **Arduino Mega:** Served as the central microcontroller to interface between the receiver, sensors, and motion controller.
- **RC Controller and Receiver:** A standard 6-channel RC controller and receiver were used to provide manual control over the robot's movement wirelessly.
- **11.1V LiPo Battery:** A 3-cell 11.1V lithium-polymer battery was used as the main power source, offering high current output for motors and electronics.

Part Two: Odometry System

In the second phase of the project, we integrated a dedicated odometry system for tracking position and orientation:

- **Dead Wheels (Omniwheels):** Three passive omniwheels were used as "dead wheels" that do not contribute to motion but rotate freely to measure displacement.
- **Rotary Encoders:** Each dead wheel was coupled with a high-resolution rotary encoder to measure linear movement along specific axes.
- **Encoder Mounting Mechanism:** Custom mounting hardware was designed to securely attach the dead wheels and encoders to the robot chassis in fixed orientations, ensuring accurate motion capture.

3 Progress

Part One: Three-Wheel Drive System

- We began by preparing the robot base and securely attaching the three DC motors to it.
- Initially, we learned how to control a single motor using the **Cytron MDD10A motor driver** with an **Arduino Uno**.(We

used the sabertooth2x25 motor driver for the final project).

- We then integrated an **RC controller and receiver** to wirelessly control the motor via the driver.
- Next, we implemented **PID (Proportional-Integral-Derivative) control** to achieve smooth and stable motion. At this stage, a single motor setup was sufficient, so the Arduino Uno continued to be used.
- Upon scaling up to control all three motors, we transitioned to an **Arduino Mega** to handle the additional PWM outputs and encoder inputs.
- We initially tuned the PID constants (K_p, K_i, and K_d) manually for each motor.
- Later, we integrated **Kangaroo motion controllers** to enable built-in PID control with encoder feedback. We learned to tune the controllers both manually and using the **DEScribe software**.
- During the testing phase, we initially used a regulated bench power supply, but later replaced it with a **3-cell 11.1V LiPo battery** for mobile, untethered operation.
- After completing the hardware and control system integration, we developed the **final control code**, which implements the **inverse kinematics for a 3-wheel omni configuration**. The code also translates **RC joystick inputs into directional velocity commands** (e.g., forward, strafe, and rotation), which are then mapped to the appropriate motor speeds.
- With all components tested and integrated, **Part One of the project was successfully completed**.

Part Two: Odometry System

- We began by assembling the **rotary encoders** and **three passive dead wheels (omniwheels)**.
- To mount the encoders securely, we **laser-cut custom mounting hardware** to ensure accurate alignment with the chassis.
- Currently, we are designing the **CAD model for a new base** that will accommodate the dead wheels and ensure consistent ground contact.
- Simultaneously, we are working on understanding and implementing the **odometry logic** for tracking the robot's position and orientation using three encoders placed at 120° intervals.
- This includes developing the code to convert encoder tick data into global displacement and rotational change using trigonometric relationships.

4 Discussion

The successful implementation of the three-wheel omni drive system confirmed the feasibility of using omnidirectional motion in robotic platforms. During testing, the robot was able to move in all directions—including forward, sideways, and rotationally—with smooth transitions and acceptable response times. The integration of PID control significantly improved the stability and accuracy of movement.

The kinematic model used in our final control code effectively translated joystick input from the RC controller into motor velocities by resolving desired motion vectors into individual wheel commands.

In Part Two, the implementation of the dead-wheel odometry system is ongoing. The choice of using three passive omniwheels for dead wheels, spaced at 120°, was based on the fact that they would be easier to join with the existing base. The ongoing code development aims to translate encoder tick counts into global displacement using inverse kinematics and transformation matrices.

Acknowledgements

I would like to thank our project mentors , Nitye and Mahesh for their continued guidance and help.