**Flow of the Programs (Laptop & Teensy)**

These two programs work together to control a robot in both **manual mode** (via a PS4 controller) and **autonomous mode** (following waypoints). The laptop (Python) visualizes the robot's movement, while the Teensy (C++) handles motor control, odometry, and communication with the IMU (BNO055) and encoders.

**Step-by-Step Flow**

**1. Initialization (Both Programs)**

* **Laptop (Python)**
  + Initializes a Pygame window (2000x1000) for visualizing the robot's movement.
  + Opens a serial connection with the Teensy (COM3, 115200 baud rate).
  + Defines colors, initializes the trail list, and prepares to receive serial data.
* **Teensy (C++)**
  + Initializes:
    - **Encoders (for wheel rotations)**
    - **BNO055 IMU (for orientation)**
    - **PS4 Controller (via USBHost for manual mode)**
  + Configures **motors & PID control** for precise movement.
  + Starts a **timer** to continuously update PID control and odometry.

**2. Main Loop Execution**

**Manual Mode (PS4 Controller)**

**Trigger:** The user moves the right/left joystick.

* The **Teensy reads joystick values** (X, Y for translation, LeftX for rotation).
* It **maps joystick values** to speed setpoints for the three motors.
* **PID Control**:
  + Reads encoder values to determine actual motor speed.
  + Computes the error and adjusts motor PWM accordingly.
* **Odometry**:
  + Updates the robot’s position using encoders + BNO055 IMU.
  + Sends the updated global **X, Y, Theta** back to the laptop via **Serial**.

**Autonomous Mode (Path Following)**

**Trigger:** The laptop receives position data and auto\_flag == 1.

* The **Laptop (Python) reads Serial Data** (auto\_flag, x, y, a) and determines the robot’s position.
* It **divides the path** into **2 cm steps** and calculates intermediate waypoints.
* Sends the **next step (x, y, angle)** back to the Teensy.
* The **Teensy receives these coordinates**, updates x and y as motor targets, and moves towards them.

**3. Real-time Visualization (Laptop)**

* The **laptop draws the movement trail** of the robot in **Pygame**.
* Displays the **current robot position** in text format.
* Updates the screen every frame to reflect new movement.

**Example Use Case**

**Use Case: Navigating to a Target Position (200, 100)**

**Step 1: Start in Manual Mode**

1. The user moves the **PS4 joystick** to manually control the robot.
2. The Teensy:
   * Reads joystick values.
   * Calculates motor speeds via PID.
   * Moves the robot based on joystick input.
   * Sends updated (X, Y, Theta) back to the laptop.
3. The laptop updates the **Pygame visualization**.

**Step 2: Switch to Autonomous Mode**

1. The robot reaches a **starting position** (e.g., X = 50, Y = 20).
2. The laptop receives **auto\_flag = 1**, signaling the switch to **autonomous mode**.
3. The laptop calculates a **step-by-step path** to (200, 100) in **2 cm increments**.
4. It sends the first waypoint to the **Teensy**.

**Step 3: Robot Moves to Target**

1. The Teensy reads the new target (X = 52, Y = 22).
2. Updates x, y motor setpoints and moves towards the next waypoint.
3. Once it reaches (200, 100), it **stops** and waits.