# 📁 Magnetic Tracking Robot — Nabit-Inspired

**Full Design Document Folder**

---

## 1. Project Summary

The Magnetic Tracking Robot is a Nabit-inspired robotic system designed to detect and follow magnetic fields using Hall effect sensors. It moves precisely by adjusting its position according to differences in magnetic strength detected on its left and right sides.
 This project matters because it demonstrates how robots can use **magnetic sensing** to navigate or align with targets, which can be applied in **industrial automation, precision delivery, or medical robotics** where magnetic tracking is safer and more reliable than cameras or GPS.

---

## 2. Problem Statement

In many environments, optical or GPS-based tracking systems fail due to interference, lighting, or obstacles. This robot solves the problem by using **magnetic field detection** to maintain orientation and position even when visual or wireless signals are weak or blocked.

---

## 3. Use Case

A possible use case is a **guided delivery robot** that aligns with magnetic paths embedded in the floor or walls. It can also be used in **industrial lines** where robots need to position precisely with magnets or magnetic tags for loading/unloading tasks.
 Users simply place two magnets (left/right) to define the target zone. The robot then aligns automatically between them using magnetic feedback.

---

## 4. Component Details & Behavior

| Component | Description |
| --- | --- |

| | |
|---|---|
| **Magnets** | Two fixed Neodymium magnets placed at left and right for consistent magnetic field generation. |
| **Hall Sensors** | Two sensors (left/right) detect magnetic field strength and send signals to the Arduino UNO. |
| **Arduino UNO** | Reads both sensors, compares field strength, and sends control signals to the motor. Runs a **PID loop** for precision control. |
| **DC Motor + Encoder** | Provides motion and feedback. The encoder measures motor position and sends data back to the Arduino for correction. |
| **Wheel Hub** | Single driven wheel with small caster to stabilize movement. |
| **Power System** | Battery powers Arduino and motor (with shared ground). Separate supply recommended for motor stability. |

---

## 5. How It Works (Control Logic Summary)

1. **Sensing:** Both Hall sensors measure magnetic field strength.

2. **Comparison:** Arduino calculates the difference between left and right readings.

3. **Control:** Arduino adjusts the motor output to balance readings.

4. **Feedback:** Encoder sends position updates so the Arduino can fine-tune motion.

5. **Result:** The robot aligns itself between the two magnets, maintaining equal magnetic strength on both sides.

## 6. Prototype Interactions (3+ Events)

1. **Sensor Change Event:**

   - If the left Hall sensor detects a stronger field than the right, Arduino triggers a **clockwise rotation**.

   - Interaction: *Sensor → Arduino → Motor*

2. **Encoder Feedback Event:**

   - When encoder tick count shows deviation from desired angle, Arduino adjusts PWM duty to correct the position.

   - Interaction: *Encoder → Arduino → Motor Control*

3. **Sensor Equilibrium Event:**

   - When both Hall sensors read within a certain tolerance, Arduino stops or stabilizes the motor to hold position.

   - Interaction: *Sensors → Arduino → Motor Stop/Hold*

## 7. Diagrams (Included)

You have two diagrams that illustrate the system:

- **Version 1:** Full labeled design with all connections, components, and notes (your detailed blueprint).

- **Version 2:** Simplified schematic for clarity and visual explanation.

*(Both images are referenced as core documentation visuals.)*

---

## 8. Future Improvements

- Add a **PID controller** in Arduino code for smoother balancing.

- Use **stronger magnets** or **adjustable sensitivity** Hall sensors for finer detection.

- Include an **OLED display** to visualize magnetic strength or motor correction in real time.

- Develop **autonomous calibration** (robot aligns itself automatically at startup).

## 9. Timeline (Estimate)

| Stage | Description | Duration |
| --- | --- | --- |
| Phase 1 | Research & Design | 1 month |
| Phase 2 | Prototype Assembly | 3 months |
| Phase 3 | Electronics & Arduino Programming | 2 months |
| Phase 4 | Testing & Feedback Loop Tuning | 2 months |
| Phase 5 | Final Adjustments & Report | 2 months |
| **Total** | **≈10 months** | |