



**ZAS ROBOTICS**

BUILD THE FUTURE

Navigational Robotics

Series

Robot Car-3 (Advanced  
Autonomy)

Documentation (v1.0)

## Contents

1. Robot Car-3 (Advance autonomy Edition)- Product Overview & Hardware Documentation.....	3
2. What's Included in the Car-3.....	4
3. Hardware System Architecture .....	5
4. Software Architecture.....	7
5. Software Examples, GitHub Resources for foundational Code.....	8
6. Student Learning Outcomes.....	12
7. Technical Specifications .....	13
8. Final Summary.....	14

# 1. Robot Car-3 (Advance autonomy Edition)- Product Overview & Hardware Documentation.

The **ZAS Robotics Navigational Robot Car-3 (Advanced Autonomy Edition)** is the most sophisticated platform in the ZAS Navigational Series.

It is specifically designed for:

- University students
- Engineering educators
- Robotics & AI research labs
- ML/RL experimenters
- Mechatronics and embedded systems programmes

Car-3 enables learners to understand and apply **real autonomous navigation**, using professional-grade sensors:

- **DC-370 precision encoder motors**
- **MPU6050 IMU (accelerometer + gyroscope)**
- **QMC5883 3-axis digital compass**
- **NEO-6M GPS module**
- **Magnetic rotary encoder**
- **OLED real-time telemetry display**
- **Integrated power & control system based on Arduino Nano**

## Students learn:

- Odometry
- Gyroscope drift correction
- GPS navigation
- Heading stabilisation
- Complementary, Kalman & Extended Kalman Filters
- PID speed & steering control
- Trajectory planning
- Reinforcement Learning dataset generation
- Research-level algorithm development

Car-3 provides a stable hardware foundation with **infinite algorithm possibilities**, making it ideal for:

- Autonomous navigation courses
- B.Tech/M.Tech labs
- Industry training

- Final-year academic projects
- Early SLAM & localisation research
- Reinforcement Learning experimentation

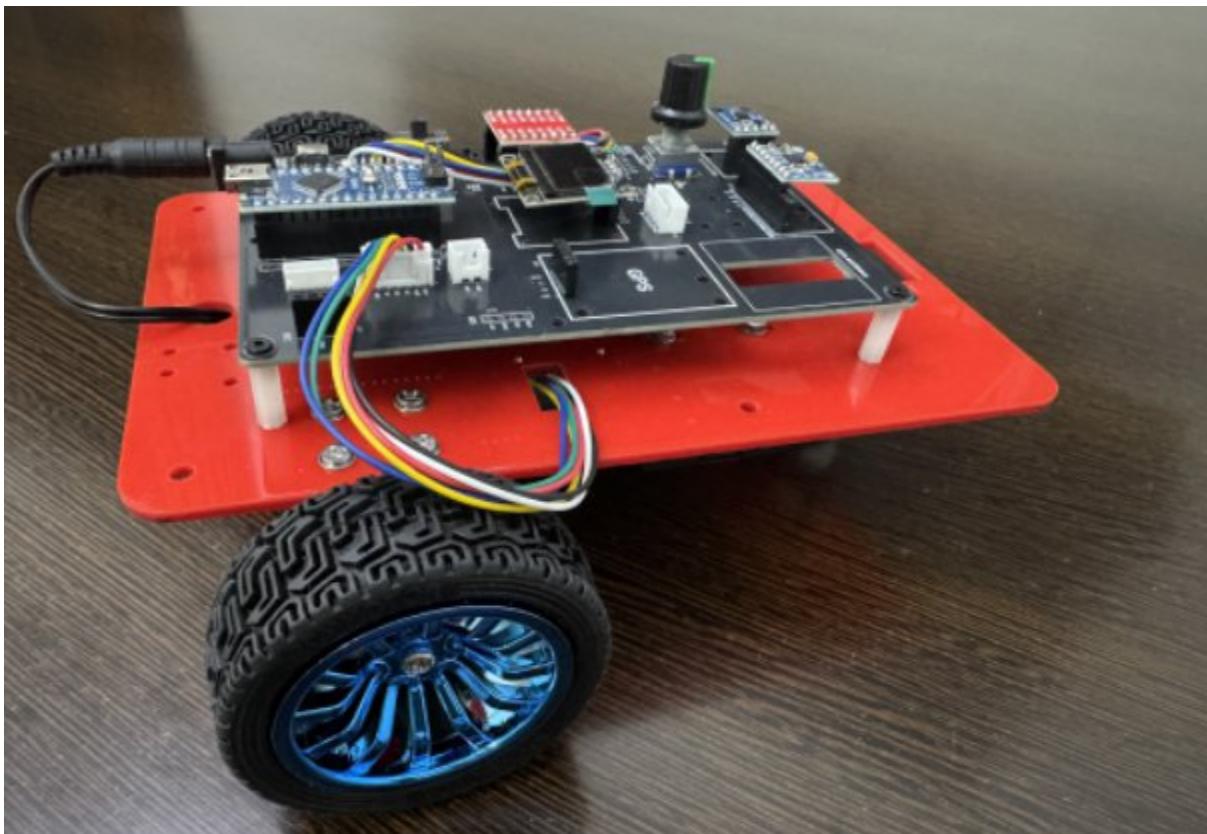


Figure 1. Robot Car-2 (Communication Edition)

## 2. What's Included in the Car-3

The kit is delivered as a **fully assembled and calibrated autonomous robot**, with all sensors and wiring neatly integrated. No soldering required.

### Core Robot Platform

- Fully Assembled Car-3 Robot
- **Arduino Nano (ATmega328P)** microcontroller
- **2× DC-370 High-Precision Encoder Motors**
- Front caster wheel
- Integrated power distribution board (5V & 3.3V regulators)

### Advanced Sensor Suite

- **MPU6050 – 6-axis Accelerometer + Gyroscope**
- **QMC5883 – 3-Axis Digital Compass**

- **NEO-6M GPS Module** (with dedicated antenna mount)
- **Rotary Encoder Module**
- **Dual Motor Encoders** (Left & Right)

## User Interface & Telemetry

- **0.96" OLED Display – Real-time robot telemetry**  
(heading, speed, IMU angle, GPS status, PID data)

## Power System

- **18650 High-Capacity Battery Holder**
- **18650 Smart Charger**
- Clean, regulated 5V/3.3V supply
- On-board reverse polarity & over-current protection

## Accessories

- Plug-and-play JST cables
- Preloaded foundational test programs
- Documentation:
- Pinouts
- Board layout
- Sample algorithms
- IMU, GPS, encoder test code
- PID and odometry examples

This is a complete ready-to-use research platform for universities, AI labs, and academic projects.

## 3. Hardware System Architecture

Car-3 is built using a **modular, research-grade architecture** suitable for real autonomy and sensor fusion experiments.

Hardware is organised into three layers.

## A. Base Chassis – High-Stability Navigation Platform

The precision 3-wheel design ensures stable motion ideal for:

- Kalman Filtering
- PID tuning
- Odometry
- EKF localisation

## **1. Dual DC-370 Encoder Motors**

Provide:

- Accurate wheel rotation measurements
- High-fidelity odometry
- Smooth speed control
- Low noise for EKF and control experiments

## **2. Rear Caster Wheel**

Provides friction-free support:

- Smooth forward/backward motion
- Stable rotation
- Reduced IMU noise

## **3. Rear-Mounted Battery Pack**

Ensures:

- Low centre of gravity
- Stable IMU readings
- Reduced mechanical vibration
- Improved compass accuracy

# **B. Multi-Sensor Navigation Suite**

## **1. MPU6050 IMU**

Used for:

- Roll, pitch, yaw
- Drift analysis
- Complementary & Kalman filter experiments

## **2. QMC5883 Compass**

Provides:

- Absolute magnetic heading
- Long-term drift correction

## **3. NEO-6M GPS**

Enables:

- Outdoor navigation
- Waypoint following
- Long-distance path tracking

- RL dataset generation

#### 4. Dual Wheel Encoders

Provide:

- Dead-reckoning odometry
- Speed calculation
- EKF velocity input

#### 5. Magnetic Rotary Encoder

Used for:

- Precise rotation experiments
- Heading correction
- Research data collection

#### 6. OLED Display

Shows:

- Heading
- IMU angle
- GPS lock
- PID output
- Encoder counts

### C. Creative Controller + Integrated Power System

#### Key Features

- Stable **regulated 5V & 3.3V** rails
- Over-current protection
- JST plug-and-play connectors
- Arduino Nano MCU
- Debug LEDs
- Clean signal routing for IMU, compass, GPS & encoders

### 4. Software Architecture

Car-3 uses a **4-layer embedded architecture**, similar to ROS2 navigation systems.

#### Layer 1 – Sensor Acquisition

Reads raw data from:

- IMU

- Compass
- GPS
- Encoders
- Rotary encoder

## Layer 2 – Sensor Fusion & State Estimation

Implements:

- Low-pass filters
- Complementary filtering
- Kalman & EKF
- Pose estimation ( $x, y, \theta$ )

## Layer 3 – Motion Control

Handles:

- PID speed control
- Steering correction
- Odometry-based motion
- Trajectory following

## Layer 4 – High-Level Behaviour

Students create:

- Autonomous driving logic
- Waypoint navigation
- RL policies
- GPS path following
- Heading-controlled behaviours

## 5. Software Examples, GitHub Resources for foundational Code

The Navigational Robot **Car-3 (Advanced Autonomy Edition)** is more than a hardware platform.

It is supported by a **complete educational software ecosystem**, designed to help students understand real autonomous systems and build their own algorithms from the ground up.

ZAS Robotics provides a structured set of foundational example programs through its official GitHub repository.

These examples offer:

- Clear and modular code structure
- Extensive inline comments
- Hardware-validated logic
- Experiment-oriented outputs
- Full compatibility with Arduino Nano

These programs are **not meant to be final solutions**, but rather **launchpads** for student innovation, university laboratory sessions, and research workflows.

## • Foundational Example Program Overview

The repository contains six core programs, each focusing on one major subsystem of the Car-3 robot.

### **01\_ControllerBoardUnitTest\_WithoutServo.ino**

#### **Purpose:**

A comprehensive diagnostic program for validating every subsystem:

- Dual encoder interrupt reading (left & right wheels)
- Motor direction + PWM control
- OLED real-time feedback
- GPS parsing
- IMU (roll/pitch) readings
- Compass azimuth calculation
- Rotary encoder position tracking

#### **Student Use Cases:**

- Validate wiring and sensor behaviour
- Verify encoder polarity
- Observe orientation drift and GPS updates
- Understand update rates and timing loops

This is the **primary hardware verification tool**—recommended before running any autonomy or PID experiments.

### **02\_GPS\_TX\_RX\_OLED.ino**

#### **Purpose:**

Dedicated GPS acquisition and OLED display test.

Shows:

- Latitude
- Longitude

- Date
- Time (UTC)
- Validity checks

**Student Use Cases:**

- Learn TinyGPS++ parsing
- Measure GPS drift outdoors
- Compare accuracy vs update frequency
- Prepare waypoint navigation experiments

## **03\_DigitalCompass\_QMC\_BearingAngle.ino**

**Purpose:**

Real-time compass calibration and bearing calculation.

Includes:

- Hard-iron and soft-iron correction
- Axis min/max collection
- Normalised readings
- True North angle output

**Student Use Cases:**

- Understand magnetic distortion
- Study drift over time
- Build compass + IMU fusion for EKF
- Test non-linear magnetometer behaviour

Serves as the **foundation for orientation filtering** and state-estimation algorithms.

## **04\_MP6050\_Roll\_Pitch\_OLED.ino**

**Purpose:**

Reads accelerometer and gyro data and computes:

- Roll
- Pitch

**Student Use Cases:**

- Study accelerometer vs gyro noise
- Visualise drift
- Understand IMU coordinate frames
- Begin complementary filter design

## **05\_Basic\_Car\_Movements.ino**

### **Purpose:**

Simple motion primitives for:

- Forward
- Reverse
- Left turn
- Right turn

### **Student Use Cases:**

- Check motor wiring
- Quickly test indoor movement
- Build complex behaviours on top
- Prepare for PID and closed-loop control

## **06\_QMC\_Raw\_Value.ino**

### **Purpose:**

Simultaneous IMU + Compass raw data logging.

Outputs CSV-formatted:

- Ax, Ay, Az
- Gx, Gy, Gz
- Mx, My, Mz

### **Student Use Cases:**

- Build noise models
- Design custom Kalman Filters
- Perform offline EKF experiments
- Create datasets for ML / RL
- Prepare research papers

An essential tool for **sensor science and data-driven robotics research**.

## Why These Example Programs Matter

Each example mirrors the workflow used in real robotics research:

Skill Area	Supported By	Why It Matters
Motor control	Basic Movements	Foundation of PID + trajectory control
IMU science	MPU examples	Orientation estimation, drift, noise
Compass fusion	QMC examples	Stable heading for EKF
GPS navigation	GPS_TX_RX	Outdoor navigation, waypoint following
Odometry	ControllerBoardTest	Precise localisation
Sensor fusion	All modules	Central topic in modern robotics
Data logging	Raw_Value scripts	ML/RL training, research papers

Students progress from **raw sensor reading → filtering → pose estimation → navigation algorithms → publishable research.**

## How Students Should Use These Codes

The code library is designed to **support learning**, not replace it.

**ZAS provides:**

- Correct baseline programs
- Sensor initialization logic
- Clean architecture
- Commented examples
- Hardware mappings and pinouts

This approach ensures **true understanding** and prevents dependence on prewritten code.

Students learn **theory through hands-on experimentation**—the defining feature of real engineering.

## 6. Student Learning Outcomes

Car-3 is engineered as a university-level autonomy platform—not a toy robot.

Students gain practical, industry-ready skills that directly map to robotics, AI, embedded systems, and autonomous navigation.

### A. Core Robotics

- Motor control
- PWM

- Encoders
- Calibration
- Odometry

## B. Sensor Fusion

- Complementary filter
- Kalman & EKF
- Pose estimation

## C. Control Engineering

- Open-loop vs closed-loop
- PID tuning
- Trajectory control

## D. AI & Research

- RL dataset collection
- ML experiments
- Imitation learning
- Noise and uncertainty modelling

## E. Engineering Skills

- Debugging hardware
- Writing custom algorithms
- Performing repeatable experiments

# 7. Technical Specifications

### **Motor System:**

- DC 370 high-torque encoder motors
- 6–12 V operation
- 300–350 RPM
- Dual-channel encoder

### **Microcontroller:**

- Arduino Nano (ATmega328P)
- 32 KB flash, 2 KB SRAM
- 6 PWM channels
- 10-bit ADC

### **Sensors:**

- MPU6050 Accelerator + Gyroscope

- QMC5883 Magnetometer
- NEO-6M GPS
- Rotary Encoder
- Dual wheel encoders
- 128×64 OLED

**Power System:**

- 18650 Li-ion cells
- Regulated 5V/3.3V outputs
- Overcurrent and reverse-polarity protection

## 8. Final Summary

The **ZAS Car-3 (Advanced Autonomy Edition)** is the highest-level robot in the ZAS Navigation Series.

It is a **research-grade, university-level autonomy platform**, designed for deep learning in:

- Sensor Fusion
- EKF Localization
- GPS Navigation
- PID Motion Control
- Reinforcement Learning
- Filtering & Noise Reduction
- Real-World Robotics Engineering

With powerful sensors, precision encoder motors, Arduino-based processing, and full software transparency, Car-3 gives students the rare opportunity to learn **real robotics—not simulations**.