***Line Following Buggy with Simultaneous Localisation and Mapping Capabilities***

The vehicle uses an onboard computer to convert the images and data received from the camera attached on the vehicle into 3d structures which is later processed into a map. This explains the concept of SLAM. As of now the vehicle uses a line following system for movement and actuation.

***How it moves?***

The robot uses a simple yet effective line-following mechanism for navigation. It is equipped with an onboard infrared (IR) sensor that continuously monitors the surface beneath it. The working principle is based on the reflective property of surfaces: when the IR sensor emits infrared light, it detects the amount of light that is reflected back. If the surface is white or reflective, the IR light bounces back and is detected by the sensor, indicating that the robot is not on the black line. In such a case, the robot recognizes that it has deviated from its path and adjusts its direction accordingly. On the other hand, if the surface is black, it absorbs the infrared light, and little to no light is reflected back to the sensor. This signals the robot that it is on the correct path, prompting it to continue moving forward.

For movement, the robot uses two DC motors that are connected to the rear wheels, enabling forward and backward motion. Steering is managed by a servo motor connected to the front wheel assembly. What makes this setup unique is that the steering mechanism isn’t fixed to a specific path; instead, it actively scans the area. The IR sensor is mounted on a servo that rotates it to the left, right, and forward, helping the robot determine the direction where the black line continues. Based on this input, the servo adjusts the steering to keep the robot aligned with the path.

This approach not only allows the robot to follow a predetermined track, but also gives it a degree of adaptability to find the correct route even at intersections or sharp turns.

***Components Summary***

JETSON NANO – On board Computer (Controller)

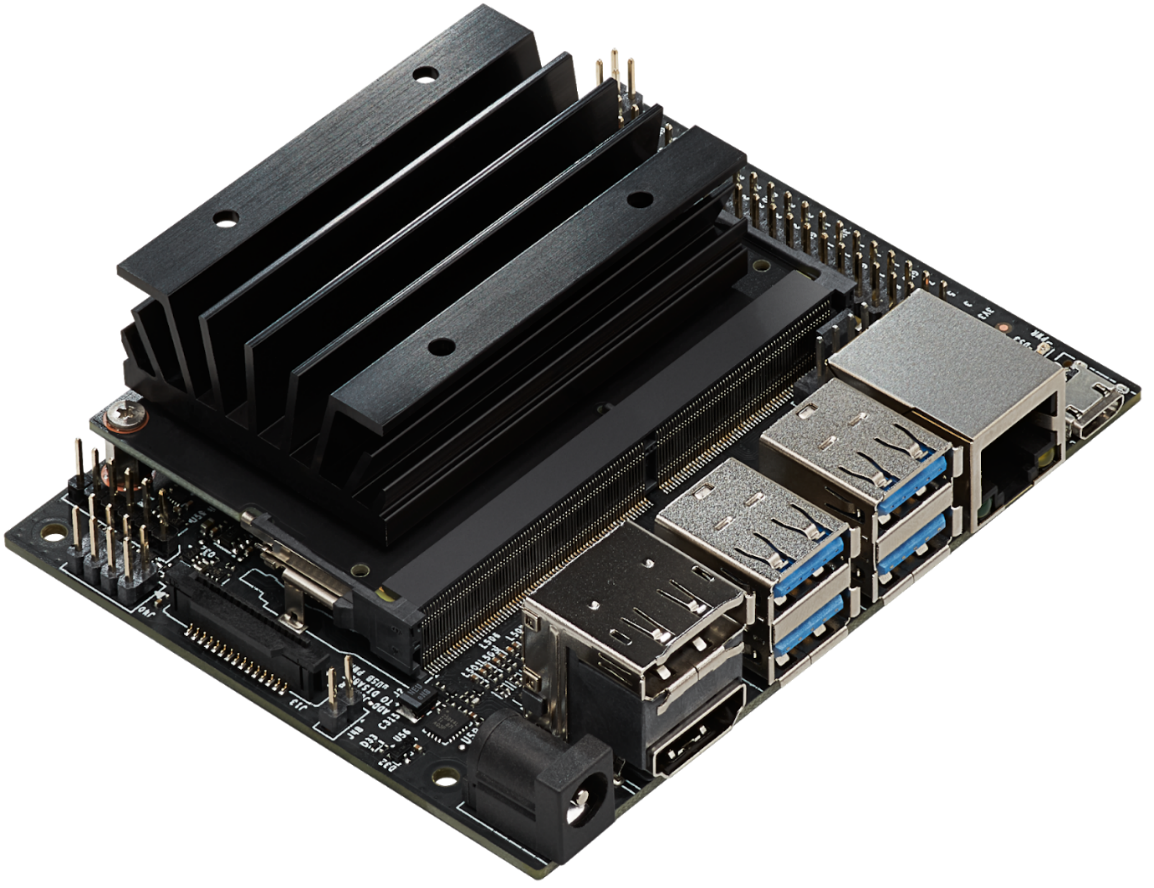
SG90 SERVO MOTOR – Actuators

IR Module – Sensing

Xbox Kinect v1 – Camera

ESP32 – Controller or acts a bridge between actuators and jetson nano.

***Jetson Nano***

******

Technical Specifications and Features

Processor:

* CPU: Quad-core ARM Cortex-A57 @ 1.43 GHz.
* GPU: 128-core NVIDIA Maxwell architecture GPU.
* Floating Point Performance: Up to 472 GFLOPS.

Memory and Storage:

* RAM: 4GB 64-bit LPDDR4, 25.6 GB/s bandwidth.
* Storage: microSD card slot for main storage.

Wireless & Connectivity (via expansion or USB):

* Ethernet: Gigabit Ethernet port.
* Wi-Fi and Bluetooth: Not onboard by default, but supported via USB dongle or M.2 module (Jetson Nano 2GB includes Wi-Fi).

Display and Camera:

* Display Output: HDMI 2.0 and DisplayPort 1.2.
* Camera Interface: MIPI CSI-2 (15-pin connector for Raspberry Pi Camera Module v2).

I/O and Expansion:

* USB: 4 × USB 3.0 ports (original model), 1 × USB 2.0 micro-B (device mode), 1 × USB 2.0 (2GB version).
* GPIO: 40-pin expansion header (Raspberry Pi compatible layout).
* Other I/O: I²C, SPI, UART, PWM, and GPIOs.

Power Supply:

* Power Input: 5V via micro-USB or barrel jack (5V⎓4A recommended for full performance).
* Power Modes: Configurable 5W or 10W modes.

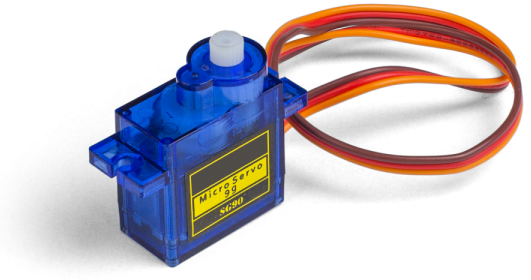
Software and Development:

* OS Support: Ubuntu-based JetPack SDK with Linux4Tegra.
* Frameworks: Support for TensorFlow, PyTorch, OpenCV, Keras, Caffe, ROS, and more.
* AI SDKs: Includes NVIDIA DeepStream, CUDA Toolkit, cuDNN, TensorRT.

References

* [Jetson Nano Specifications and Datasheet](https://developer.nvidia.com/embedded/jetson-nano)
* [Jetson Operating System](https://developer.nvidia.com/embedded/develop/software)

***SG90 Servo***

******

Technical Specifications and Features

Motor Type: SG90 is a 9g micro servo motor with analog control.  
Material: Gear set made from plastic (typically nylon).  
Weight and Dimensions: Weighs about 9g; dimensions approx. 22.5 × 11.8 × 31 mm.  
Operating Voltage: 4.8V to 6.0V (typically 5V).  
Stall Torque:

* At 4.8V: ~1.8 kg·cm
* At 6.0V: ~2.2 kg·cm  
  Operating Speed:
* At 4.8V: ~0.1 s/60°
* At 6.0V: ~0.08 s/60°  
  Rotation Range: ~180°, controlled via PWM signal.  
  PWM Control: Standard 50Hz PWM (20 ms period), where:
* 1 ms pulse ≈ 0°
* 1.5 ms pulse ≈ 90° (neutral)
* 2 ms pulse ≈ 180°  
  Connector: 3-pin female connector (Brown = GND, Red = VCC, Orange = PWM signal).

**References**

* [SG90 Actuator Datasheet](https://www.friendlywire.com/projects/ne555-servo-safe/SG90-datasheet.pdf)
* [SG90 Working](https://www.electronics-lab.com/project/using-sg90-servo-motor-arduino/)

***IR Sensor (LM393)***

***A blue and white electronic device

AI-generated content may be incorrect.***

Technical Specifications and Features

Sensor Type: Infrared reflective sensor module based on LM393 comparator.  
Functionality: Detects objects or surface contrast by comparing reflected IR light intensity.

**Components**

* Infrared LED (Emitter)
* Photodiode or Phototransistor (Receiver)
* LM393 Comparator IC
* Trimpot (Potentiometer for sensitivity adjustment)
* Indicator LEDs (Power and Output)

**Operating Voltage**

* 3.3V to 5V DC (compatible with microcontrollers like Arduino, Raspberry Pi, ESP32)

**Current Consumption**

* Typically 10–20 mA (may vary slightly depending on module)

**Detection Range**

* Approximately 2 mm to 30 mm
* Most accurate for high contrast surfaces (e.g., black line on white background)

**Output**

* Digital Signal (High/Low logic level)
  + Output Low (0): Object detected (IR reflected)
  + Output High (1): No object detected (no IR reflection)

**Interface**

* Typically 3-pin or 4-pin header:
  + VCC (Power Input)
  + GND (Ground)
  + OUT (Digital Output)
  + EN (Optional Enable Pin in some variants)

**Tuning**

* Sensitivity adjustable via onboard trimpot
* Turning the potentiometer changes the comparator threshold, allowing fine-tuning for different distances or surface reflectivity

**Applications**

* Line-following robots
* Obstacle and edge detection
* Proximity sensors for automation
* Motor speed measurement (with encoder wheel)

**Limitations**

* Limited to short-range detection
* Susceptible to interference from strong ambient infrared sources like sunlight
* Works best in controlled lighting environments

**References**

* [IR Sensor Datasheet](https://adiy.in/shop/ir-sensor-module-with-pot/?srsltid=AfmBOor4Jx69rnmTFdGppe6GSkGGFQ6U-aAsoYLNl0ANHYaVmts_Brbl)
* [IR Sensor Schematics](https://www.elprocus.com/infrared-ir-sensor-circuit-and-working/)

***XBOX Kinect v1***



**Technical Specifications and Features**

Sensor Type: RGB-D motion sensing input device developed by Microsoft for Xbox 360 and later used in PC applications.  
Purpose: Captures depth and color data to enable body tracking, 3D scanning, and gesture recognition.

**Components**

* RGB Camera (Color)
* Depth Sensor (IR projector + IR camera)
* Multi-array Microphone (4 microphones)
* Tilt Motor for vertical adjustment
* Accelerometer

**Operating Voltage**

* Requires 12V DC input (via proprietary connector or adapter)
* USB for data connection to host (USB 2.0)

**Power Consumption**

* Approximately 2.5W–5W

**Camera Specifications**

* RGB Camera:
  + Resolution: 640×480 @ 30 FPS (default), up to 1280×1024 via unofficial mods
  + Field of View: ~57° horizontal, ~43° vertical
* Depth Camera:
  + Resolution: 320×240 @ 30 FPS
  + Depth Range: ~0.8 m to 4.0 m
  + Technology: Structured light using IR dot projector

**Microphone Array**

* 4 microphones with beamforming and ambient noise suppression
* Enables voice commands and sound localization

**Motorized Tilt**

* Automatic tilt adjustment of up to ±27°
* Controlled via software commands over USB

**Interface**

* Proprietary connector (Xbox 360) or USB adapter for PC use
* USB 2.0 for data
* Separate power connection (12V)

**Compatibility**

* Xbox 360 (native)
* PC (with Kinect for Windows SDK or OpenNI/NiTE)
* Compatible with Windows, Linux (via OpenNI/libfreenect), and ROS

**Software & SDK**

* Microsoft Kinect SDK (Windows)
* OpenNI & NITE (open-source alternative)
* OpenCV, PCL, and Open3D support for point cloud and image processing

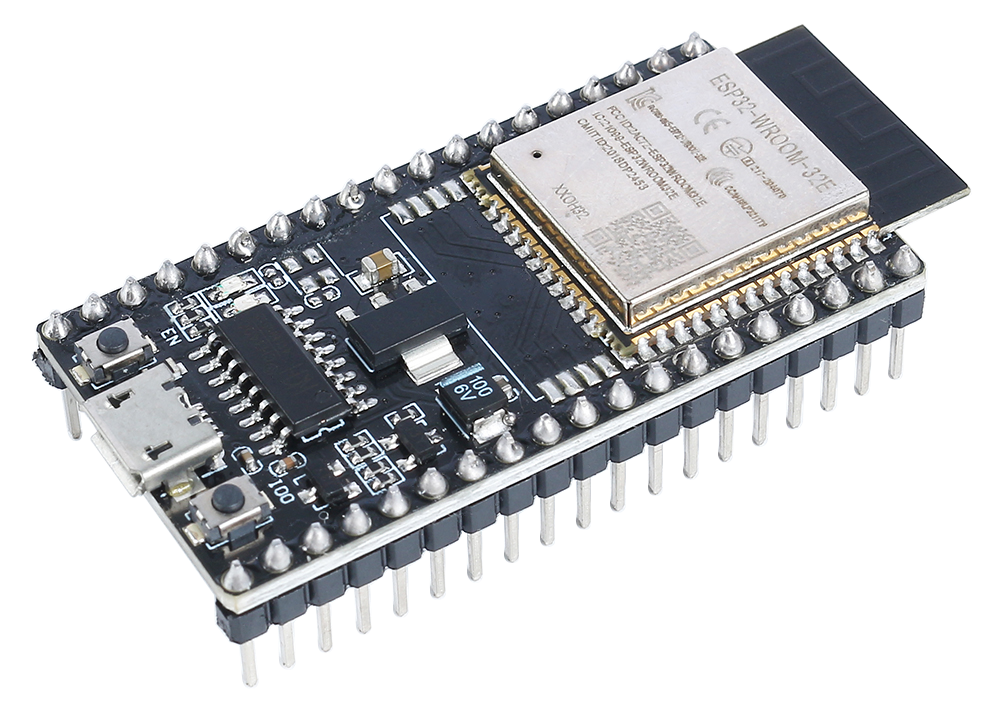
**Applications**

* Gesture and body tracking
* 3D scanning and point cloud generation
* Robotics and SLAM
* Voice recognition and control
* AR/VR prototyping

**References**

* [Develop with Xbox Kinect v1](https://learn.microsoft.com/en-us/archive/msdn-magazine/2012/june/kinect-starting-to-develop-with-kinect)
* [What is Kinect?](https://docs.hidale.com/hardware/sensors/kinect-v1)

***ESP32***



**Technical Specifications and Features**

Microcontroller: ESP32 — a low-power system-on-chip (SoC) microcontroller with integrated Wi-Fi and Bluetooth, developed by Espressif Systems. It is widely used in IoT, robotics, and embedded systems projects.

**Processor**

* Dual-core Xtensa® 32-bit LX6 microprocessor (can also be configured as single-core)
* Clock Speed: up to 240 MHz
* Performance: up to 600 DMIPS
* Ultra-low-power co-processor for sensor monitoring during sleep

**Memory**

* SRAM: 520 KB
* ROM: 448 KB
* External Flash Support: Up to 16 MB (typically 4MB in Dev Boards)

**Connectivity**

* Wi-Fi: IEEE 802.11 b/g/n
* Bluetooth: v4.2 BR/EDR and BLE
* Ethernet MAC Interface
* SPI, I²C, I²S, UART, CAN

**GPIO**

* Total GPIO Pins: 34
* Most pins support PWM, ADC, DAC, SPI, I²C, UART
* 12-bit SAR ADC (up to 18 channels)
* 2 × 8-bit DAC
* Capacitive Touch: 10 inputs
* Hall Sensor and Temperature Sensor built-in

**Timers and PWM**

* 4 × 64-bit Timers
* 2 × 32-bit Timers
* PWM for up to 16 channels (LEDC)

**Power Supply**

* Operating Voltage: 2.2 V to 3.6 V (typically 3.3V)
* Deep Sleep Current: <5 μA
* Power Modes: Active, Modem Sleep, Light Sleep, Deep Sleep, Hibernation

**Storage**

* SPI Flash (external): 4 MB to 16 MB (depending on board)
* Optional microSD card support via SPI

**Security**

* Hardware acceleration for encryption (AES, SHA-2, RSA, ECC, etc.)
* Secure boot and Flash encryption
* Random Number Generator

**Development and Programming**

* Programming Interfaces: USB-UART, JTAG
* Programming Languages: C/C++ (Arduino IDE, ESP-IDF), MicroPython, Lua
* Tools & SDKs:
  + Arduino Core for ESP32
  + Espressif IDF (official SDK)
  + PlatformIO

**Dimensions (Common Dev Boards)**

* ESP32 DevKit v1: ~51mm × 25mm
* NodeMCU-32S: ~48mm × 25mm

**Applications**

* IoT Devices and Home Automation
* Wireless Sensor Networks
* Wearables
* Robotics and Drones
* Smart Agriculture
* Voice Assistants (ESP32-LyraT)

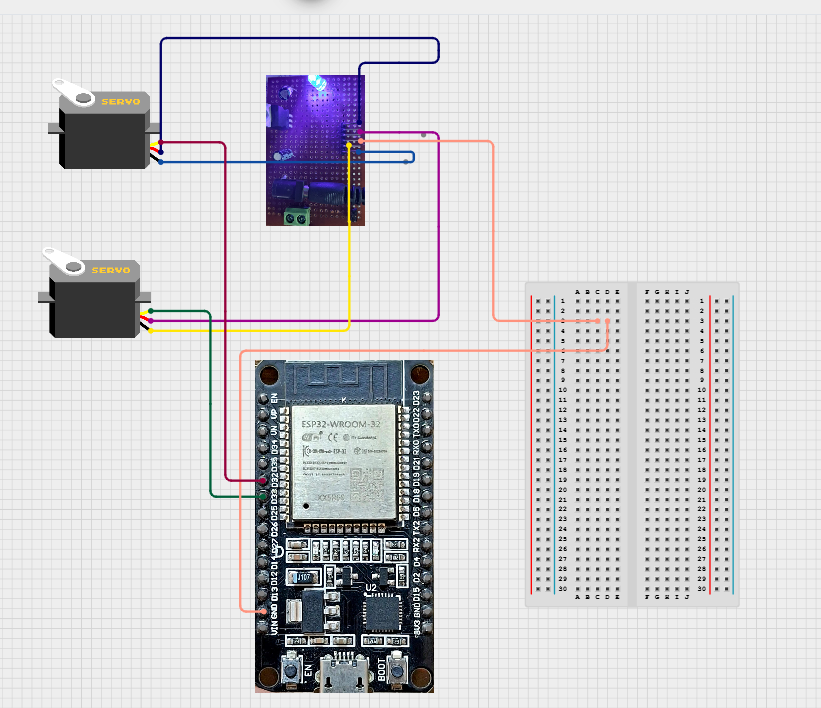
**Limitations**

* 3.3V logic (not 5V tolerant)
* Some GPIOs have specific boot functions — care needed during design
* High-frequency operations may require attention to power and grounding

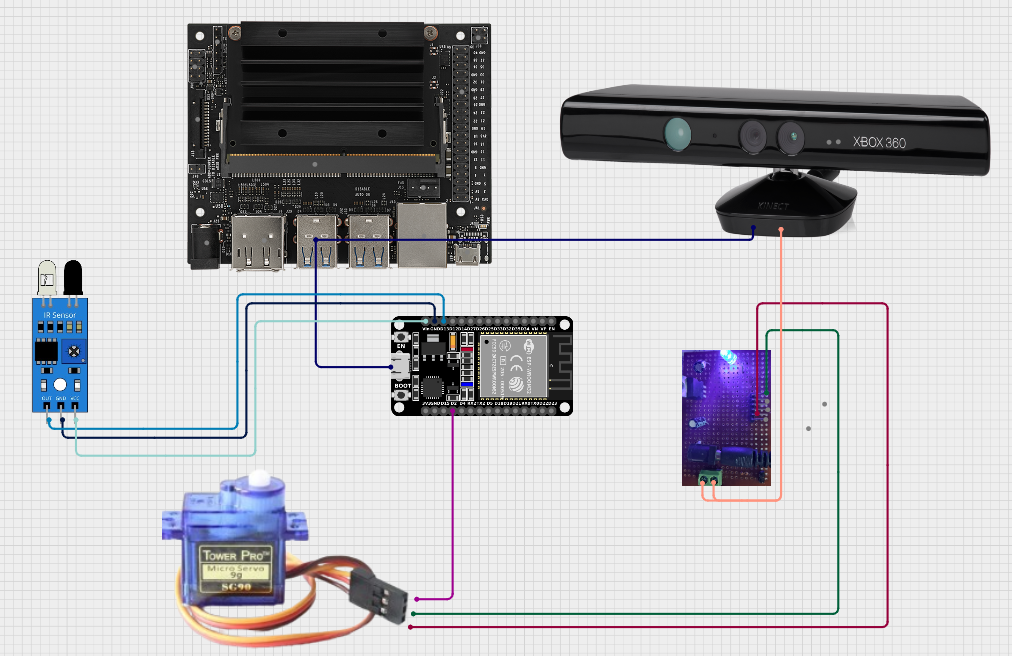
**References**

* [Espressif Official Datasheets and Technical Reference Manual](https://docs.espressif.com/projects/esp-dev-kits/en/latest/esp32/esp32-devkitm-1/user_guide.html#getting-started)
* [ESP32 DevKit v1 Schematic](https://dl.espressif.com/dl/schematics/esp32_devkitc_v4-sch.pdf)
* [Arduino-ESP32 GitHub Repository](https://github.com/espressif/arduino-esp32)

***CONNECTION DIAGRAM***

******

This is the connection laid between the *ESP32* two servos (used one instead of two in the project), the servos are powered through a buck convertor and a common ground is established between the buck convertor and *ESP32* microcontroller. The buck convertor is used to convert 12v to 5v.



The above diagram shows the entire connection between the *Jetson* board and the actuation system which consists of the IR sensor, *SG90* Servo. The IR senses and control the movement of the actuator while the *Kinect* effectively maps the space.

***Link to the codebase***

[GitHub (working)](https://github.com/adithya1770/nitk/tree/working)

[GitHub (final)](https://github.com/adithya1770/nitk/tree/final)