Designing a Simple Robot with NI LabVIEW

Overview

NIRo, a demonstration platform built by engineers at National Instruments, is a small unmanned ground vehicle that uses off-the-shelf components purchased from a local hobby store. The control system is implemented on an NI Single-Board RIO embedded platform that includes an integrated FPGA and real-time processor.

Parts Overview

NIRo requires several key hardware components to effectively navigate an environment and avoid obstacles:

Embedded controller – for sensor data acquisition, decision making, and motor control

Infrared sensors – for detecting objects within a certain distance on both sides and rear of the robot platform

Ultrasonic sensors – for detecting objects in front of the robot because they cover a larger area

H-bridges – for directing the current from the battery to the motors in the correct direction to move the motors forward or backward

Figure 1 shows where each hardware component is located on NIRo. Table 1 provides a detailed parts list.

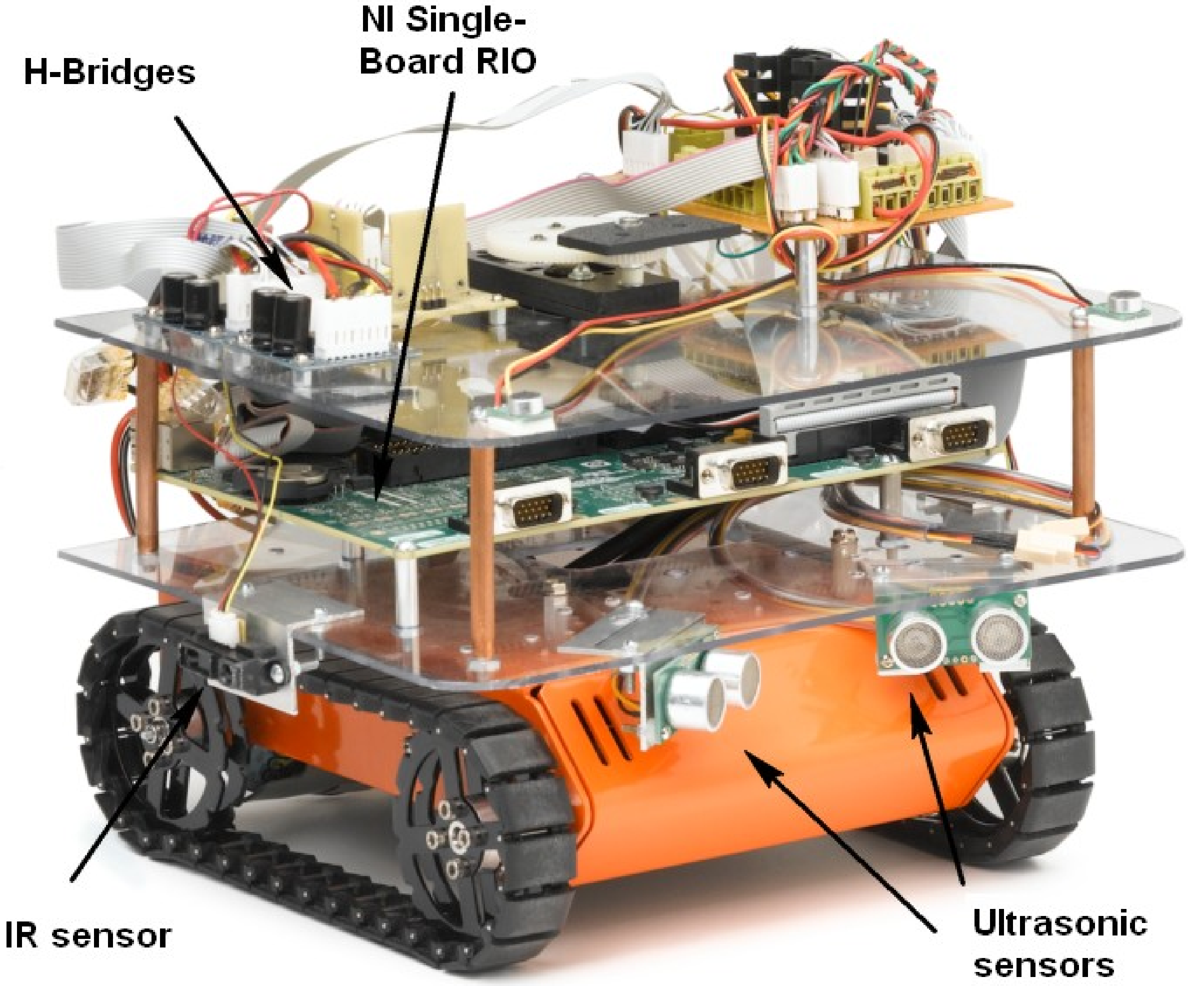

Figure 1. Diagram of NIRo's Hardware

Robot Part	Vendor	Model
Embedded controller	National Instruments	sbRIO-9632
Platform	Robotics Connection	Traxster TM
IR range sensor	Sharp	GP2D12
Ultrasonic	Devantech	SRF05

H-bridge	Solutions Cubed	Simple H-bridge

Table 1. Parts List

System Overview

NIRo’s software architecture uses both low-level and high-level control. High-level tasks, such as obstacle avoidance and interpretation of sensor data, are executed on the embedded real-time processor. The obstacle avoidance algorithm uses the data from the IR and ultrasonic sensors to make decisions about how to navigate around its environment.

NIRo also has low-level control of the motors, which is implemented on the FPGA. Depending on the outcome of the obstacle avoidance algorithm, the FPGA outputs a pulse-width modulation (PWM) signal over digital I/O lines to control the motors. The data is transferred between the real-time processor and the FPGA using FPGA interface functions built into LabVIEW.

The main software components that make up NIRo’s software architecture include:

- Obstacle avoidance algorithm
- Motor control (PWM)

NIRo’s software architecture is displayed in Figure 2. The IR sensors are connected to the built-in analog channels on the NI Single-Board RIO device, and the ultrasonic sensors are connected to the built-in digital lines. The motors are connected to the H-bridges, which are then connected to the NI Single-Board RIO device using digital I/O.

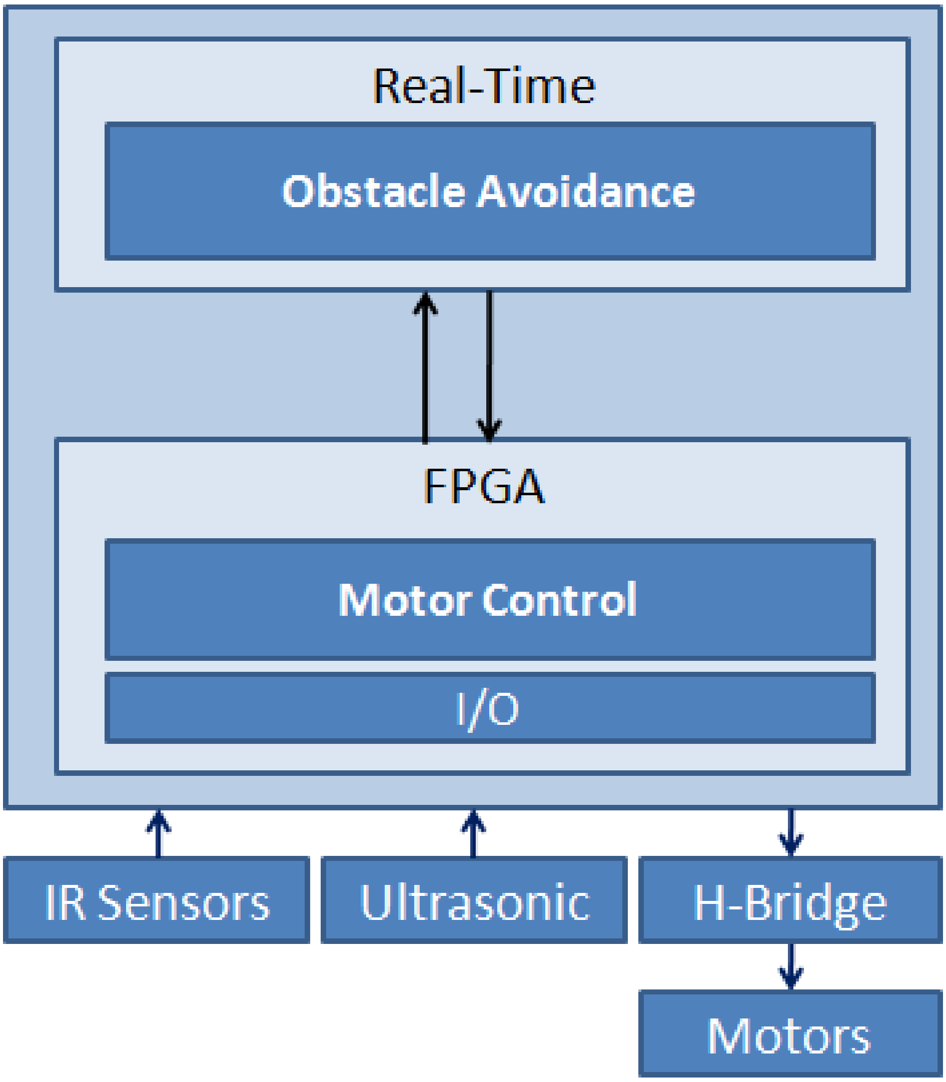

Figure 2. NIRo's Software Architecture Implemented on the NI Single-Board RIO Platform

Obstacle Avoidance

For obstacle avoidance, NIRo monitors the values from the IR and ultrasonic sensors. If the values coming from the sensors indicate that something is close to him, NIRo turns at a rate proportional to the distance the obstacle is away from him. In other words, the closer the obstacle is to NIRo, the faster he turns away from it.

This is illustrated by the control block diagram in Figure 3. There is a desired minimum distance between objects and NIRo. The robot wants to turn away from any object that is closer than this minimum distance. The proportional difference between the desired minimum distance and actual distance of objects and the actual distance measured by the sensors is used on the robot.

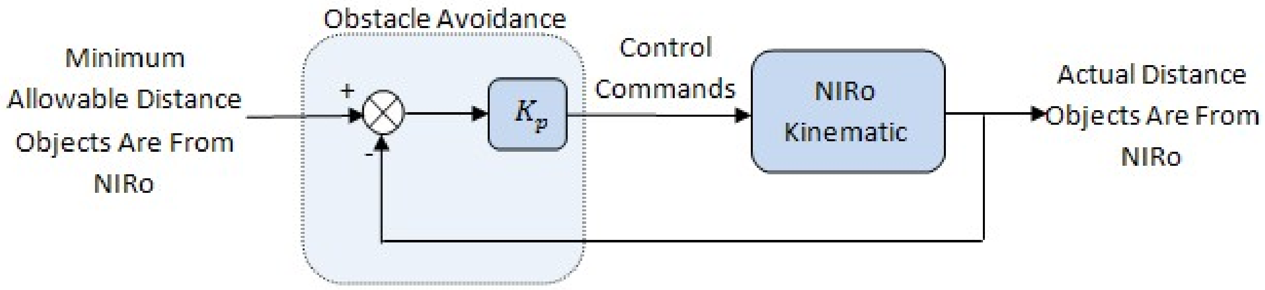
Obstacle Avoidance
Minimum Allowable Distance Objects Are From NIRo
+
-
K_p
Control Commands
NIRo Kinematic
Actual Distance Objects Are From NIRo

Figure 3. Control Block Diagram for Obstacle Avoidance

The implemented code for obstacle avoidance is shown in Figure 4. Notice that, similar to Figure 3, the actual distance objects are from NIRo is compared to the minimum desired distance these objects should be away from NIRo. Also notice that the algorithm within the case structure in Figure 4 is similar to the obstacle avoidance block in Figure 3. The output from the PID.vi is then used to send a command to the motors, which causes NIRo to turn away from the nearby obstacle.

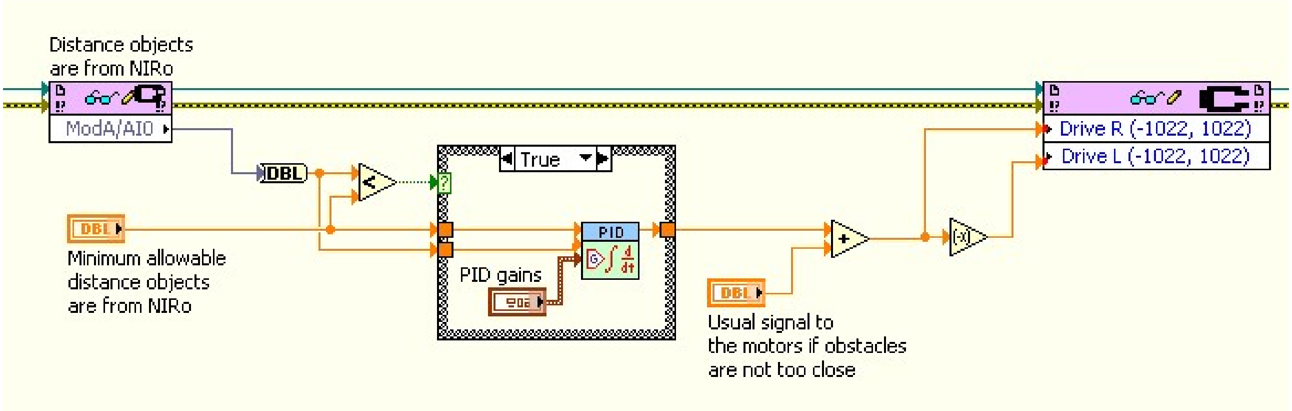
Distance objects are from NIRo
ModA/AI0
DBL
True
PID
PID gains
DBL
Minimum allowable distance objects are from NIRo
DBL
Usual signal to the motors if obstacles are not too close
Drive R (-1022, 1022)
Drive L (-1022, 1022)

Figure 4. LabVIEW Code for Obstacle Avoidance

Motor Control (PWM)

The output from the obstacle avoidance algorithm controls the duty cycle of the PWM signal for each of the motors. For example, if NIRo does not detect any obstacles in his path, he sends data to the FPG, which tells the motors to move forward. This is achieved by sending the same duty cycle for both motors. If NIRo needs to turn, the duty cycle is different for each of the motors.

The duty cycle commands are being sent from the real-time processor to the FPGA. On the FPGA, the duty cycle commands are transferred into digital commands for the H-bridges. Each H-bridge takes four digital signals, and the combination of these digital signals directs the current from the battery to the motor. Depending on the digital signals sent to the H-bridge, a motor turns forward or backward or does not move at all.

Summary

Although NIRo is a small-scale ground robot, he uses many of the same sensors, algorithms, and techniques common in larger and more complex robots. With LabVIEW Real-Time, LabVIEW FPGA, and NI Single-Board RIO, developers can easily integrate hardware and software and rapidly design, develop, and deploy algorithms. High-level decision-making algorithms and low-level motor control can all be implemented using a single platform. By integrating commercially available off-the-shelf hardware and using a high-level graphical programming language, you greatly reduce the complexities of developing an autonomous robot.