RADemics

Integration of Python with Raspberry Pi and Arduino for AI Based Physical Automation

Sneha. R, Shrishailappa Patil, Shaik Mabasha

SRI RAMAKRISHNA COLLEGE OF ARTS AND SCIENCE FOR WOMEN, WALCHAND INSTITUTE OF TECHNOLOGY, VNR VIGNANA JYOTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY

Integration of Python with Raspberry Pi and Arduino for AI Based Physical Automation

[1]Sneha. R, Assistant Professor, Computer Science, Sri Ramakrishna College of arts and Science for Women, Mail id: sneharajesh642@gmail.com , Mobile number: 824 800 2831.

[2]Shrishailappa Patil, Professor, Computer, Walchand Institute of Technology, Solapur, Mail id: stpatil72@gmail.com , Mobile number: 93615 95146.

[3]Shaik Mabasha, Assistant Professor, Dept. of CSE (AIML&IoT), VNR Vignana Jyothi Institute of Engineering and Technology, Hyderabad,500090. Mail ID: mabaskvyec@gmail.com, Mobile No: 824 800 2831.

Abstract

The convergence of artificial intelligence (AI) with embedded hardware platforms has catalyzed significant advancements in physical automation, enabling intelligent and adaptive control in diverse applications. This chapter presents a comprehensive exploration of the integration of Python programming with Raspberry Pi and Arduino microcontroller architectures to develop efficient AI-driven automation systems. The hybrid framework leverages Python's versatility to orchestrate real-time data acquisition, AI inference, and hardware actuation, addressing the inherent computational and communication challenges of edge-based automation. Emphasis is placed on designing modular architectures that facilitate seamless interfacing, low-latency decision-making, and robust error handling, ensuring system reliability and scalability. Case studies encompassing smart home automation, industrial robotics, and environmental monitoring illustrate practical implementations and performance considerations. The chapter discusses optimization strategies for deploying lightweight AI models on resource-constrained devices and the use of multi-threaded, event-driven programming paradigms to enhance system responsiveness. The findings provide critical insights for researchers and practitioners aiming to harness Python's rich ecosystem in developing next-generation intelligent physical automation solutions.

Keywords: Artificial Intelligence, Embedded Systems, Python Programming, Raspberry Pi, Arduino, Physical Automation

Introduction

The rapid advancement of embedded computing and artificial intelligence (AI) technologies has transformed the landscape of physical automation across industrial, commercial, and domestic environments [1]. Intelligent automation systems are increasingly expected to perform complex tasks such as perception, decision-making, and real-time control autonomously [2]. This shift has been driven by the need for greater operational efficiency, precision, and adaptability in dynamic conditions. Central to this evolution is the integration of flexible programming frameworks with robust hardware platforms capable of running AI algorithms at the edge, near the source of data generation [3]. Among such platforms, the Raspberry Pi and Arduino ecosystems have garnered

significant attention due to their affordability, accessibility, and extensive community support. Python, a versatile and widely adopted programming language, serves as an effective interface for coordinating AI inference, sensor data acquisition, and actuator control in these embedded systems [4]. This chapter explores the methodologies and design principles that facilitate the seamless integration of Python with Raspberry Pi and Arduino, focusing on the realization of intelligent physical automation systems [5].

The architecture of modern automation systems demands the coordination of diverse hardware components and software modules, often requiring heterogeneous processing capabilities [6]. Raspberry Pi, as a single-board computer running a full-fledged Linux operating system, provides computational power sufficient for running AI models, handling network communications, and performing complex data processing tasks [7]. In contrast, Arduino microcontrollers excel at low-level sensor interfacing and precise timing control, albeit with limited processing resources. Integrating these platforms effectively necessitates the use of efficient communication protocols and middleware that bridge high-level AI-driven decision-making with real-time actuation commands [8]. Python plays a pivotal role by offering extensive libraries for hardware abstraction, serial communication, and AI model deployment, thereby enabling developers to build scalable and modular automation workflows [9]. This integration addresses critical challenges such as minimizing latency, ensuring synchronization, and maintaining robustness across system components [10].

Deploying AI models on embedded hardware introduces constraints related to processing power, memory, and energy consumption [11]. Consequently, AI algorithms must be optimized for edge deployment without sacrificing accuracy or responsiveness [12]. Frameworks like TensorFlow Lite and PyTorch Mobile have made significant strides in facilitating lightweight model execution on Raspberry Pi, allowing real-time tasks such as object detection, anomaly detection, and predictive maintenance [13]. Python's rich ecosystem supports these frameworks and provides additional tools for data preprocessing, feature extraction, and post-inference control logic implementation. The combination of these capabilities enables automation systems to make intelligent decisions locally, reducing dependence on cloud services and improving privacy and security [14]. Hybrid system designs incorporating Arduino for real-time sensor data acquisition and Raspberry Pi for AI inference leverage the strengths of each platform, achieving a balanced trade-off between responsiveness and computational complexity [15].