Shivali Sawant, 2026, 14:2
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology
An Open Access Journal

Implementation of Process Control Blocks in Robotics Systems for Efficient Task Management

Shivali Sawant[1], Veeksha Shetty[2], Gaurang Parmar[3], Jainish Shah[4], Nandkishor Narkhede[5]

[1,2]UG Program in Electronics & Computer Science Shah & Anchor Kutchhi Engineering College University of Mumbai, Mumbai, India.

[3,4,5]Shah & Anchor Kutchhi Engineering College University of Mumbai, Mumbai, India

Abstract- This paper presents a case study on the implemen- tation of Process Control Blocks (PCBs) in robotic systems for efficient task management. Modern robotic systems must simul- taneously manage multiple processes including sensor data col- lection, motor control, and inter-system communication. Without structured process management, resource conflicts can degrade system performance and delay critical tasks. This study analyzes how PCBs enable operating systems to organize, schedule, and switch between processes effectively in robotic environments. Through analytical investigation and review of existing literature, the study demonstrates that PCB-based scheduling significantly improves robot reliability and real-time responsiveness. Schedul- ing algorithms such as Priority Scheduling and Round Robin are evaluated in the context of robotic operations.

Keywords: Process Control Block, Robotics, Task Schedul- ing, Operating Systems, Real-Time Systems, Robot Operating System (ROS).

I. INTRODUCTION

Robotics is a multidisciplinary field combining mechanical systems, electronics, and computer science to create intelligent machines capable of autonomous operation. Robots are widely deployed in industries, healthcare, manufacturing, and space exploration [1].

Modern robotic systems perform multiple concurrent tasks such as collecting sensor data, controlling motors, and commu- nicating with external devices. Managing these tasks efficiently requires sophisticated operating system support. A Process Control Block (PCB) is a fundamental data structure used by an operating system to store all information about a running process [2].

In robotic contexts, PCBs are critical for coordinating different tasks so that the robot operates smoothly without resource conflicts. Industries rely on robotic automation to increase productivity, reduce operational costs, and improve accuracy. Robots can perform repetitive tasks continuously without fatigue, making them ideal for large-scale production systems.

A key requirement of robotic systems is real-time deci- sion making. Robots must continuously process sensor data and respond quickly to environmental changes. For example, an autonomous warehouse robot must detect obstacles, plan routes, and control motors simultaneously. Efficient process management through PCBs is therefore a fundamental com- ponent of robotic operating systems.

II. PROBLEM STATEMENT

Robotic systems execute numerous processes simultane- ously, including environmental sensing, information process- ing, movement control, and inter-system communication. Without proper process management, these tasks may conflict with each other and degrade system performance.

Multiple processes compete for shared resources such as CPU time, memory, and I/O devices. If not managed properly, critical tasks may be delayed. For instance, delayed motor control commands can result in incorrect movement or colli- sions. Similarly, delayed sensor processing may prevent timely obstacle detection.

Therefore, the challenge is to design an efficient PCB system that allows robots to manage multiple tasks effectively, ensuring that each process receives the necessary resources at the correct time.

III. OBJECTIVES OF THE STUDY

1. To analyze the importance of Process Control Blocks in robotics systems.
2. To study how PCBs manage multiple processes in robotic operations.
3. To evaluate the role of process scheduling in improving robot efficiency.
4. To understand how operating systems support robotics applications.

IV. LITERATURE REVIEW

Several researchers have studied robotics systems and op- erating system structures. The literature review highlights key contributions and identifies gaps that this study addresses.

Quigley et al. [1] introduced ROS (Robot Operating Sys- tem), an open-source framework that simplifies robotics soft- ware development using a node-based modular architecture. However, improved real-time scheduling mechanisms were identified as a research gap.

Silberschatz et al. [2] provided a comprehensive theoret- ical treatment of operating system concepts, demonstrating that PCBs efficiently store process information. The direct application of these concepts to robotic systems remained understudied. Burns and Wellings [3] investigated real-time scheduling algorithms and demonstrated improvements in system respon- siveness. Thrun et al. [4] showed that probabilistic methods improve robot perception accuracy, while Siciliano and Khatib [5] presented a comprehensive handbook integrating sensors, controllers, and actuators in robotic systems.

The collective literature confirms that robotic systems re- quire efficient OS support. Scheduling algorithms such as FCFS, Round Robin, and Priority Scheduling are critical for ensuring high-priority tasks execute without delay.

TABLE I
SUMMARY OF LITERATURE REVIEW

Author & Year	Title	Key Findings	Research Gap
Quigley et al., 2009	ROS: Open-Source Robot OS	Simplifies robotics software development	Need improved real-time scheduling
Silberschatz et al., 2018	Operating System Concepts	PCBs store process information efficiently	Application in robotics needs study
Burns & Wellings, 2017	Real-Time Systems	Improves system responsiveness	Simpler real-time systems needed
Thrun et al., 2005	Probabilistic Robotics	Improves robot perception accuracy	Need efficient process management
Siciliano & Khatib, 2016	Handbook of Robotics	Integrates sensors, controllers, actuators	Need better process coordination

V. RESEARCH METHODOLOGY

A. Type of Research

This study employs an analytical and case-study-based research methodology, drawing on secondary data sources including research papers, robotics textbooks, and official robotics system documentation.

B. Tools Used

The study utilized:

- Python programming language for simulation examples
- Robotics simulation tools for process modeling
- Academic research databases for literature analysis

C. Method of Analysis

TABLE II
ROBOTIC PROCESS PRIORITY TABLE

Process	Task	Priority
P1	Sensor Data Processing	High
P2	Motor Control	High
P3	Communication	Medium

The study analyzes how robotic processes are stored and managed using PCBs, and how scheduling techniques improve overall system performance. The analysis focuses on three representative robotic

processes with assigned priorities and execution times.

VI. CASE DESCRIPTION AND DATA ANALYSIS

Consider a robotic system performing three main concurrent tasks, as described in Table II. Each task is represented as a process with an associated priority level.

Each process is stored in a PCB containing information required by the operating system. Sensor data processing (P1) collects environmental data from cameras, ultrasonic sensors, or infrared sensors. Motor control (P2) translates control com- mands into physical movement by sending signals to actuators. Communication (P3) enables interaction with control centers, cloud servers, or collaborative robots.

Process Control Blocks store essential information includ- ing:

1. Process ID
2. Process State (Ready / Running / Blocked)
3. Program Counter
4. CPU Registers
5. Memory Allocation
6. Priority Level
7. I/O Status Information

The OS scheduler uses this information to determine which process executes at any given time, performing context switches as required.

Table III presents the scheduling parameters for all robotic tasks, including data logging as an additional low-priority background task.

TABLE III
ROBOTIC TASK SCHEDULING PARAMETERS

Process	Priority	Execution Time
Sensor Processing	High	5 ms
Motor Control	High	3 ms
Communication	Medium	7 ms
Data Logging	Low	10 ms

Under a Priority Scheduling algorithm, Motor Control exe- cutes first (High priority, 3 ms), followed by Sensor Processing (High priority, 5 ms), then Communication (Medium, 7 ms), and finally Data Logging (Low, 10 ms). This order ensures time-critical tasks complete within their deadlines.

Context switching, managed through PCBs, allows the CPU to:

- Save the state of the currently running process
- Load the next scheduled process from its PCB
- Execute the newly loaded process
- Save its state and restore the previous process when required

This mechanism is fundamental to enabling concurrent robotic operations without data loss or corruption.

VII. DISCUSSION

The analysis demonstrates that Process Control Blocks play a significant role in robotics systems. By organizing process information into structured data blocks, operating systems can efficiently perform process switching and scheduling.

In robotics applications, real-time performance is critical. Tasks such as motor control must be executed within strict time limits to ensure accurate movement. PCB-based scheduling ensures these tasks receive higher priority compared to less critical processes such as data logging.

Compared to traditional computing systems, robotic systems require faster response times, making efficient process man- agement extremely important. The implementation of PCBs significantly improves the reliability and efficiency of robotic systems.

Modern robotic operating systems such as ROS use mod- ular architectures where different processes run as indepen- dent nodes. These nodes communicate using message-passing mechanisms, which further enhances system flexibility and scalability [1].

VIII. CONCLUSION

This paper examined the role of Process Control Blocks in robotics systems. The study found that PCBs help organize process information and improve task scheduling in robotic operations, enabling robots to perform multiple tasks simulta- neously without delays.

Through efficient scheduling and process tracking, PCBs enable robots to perform multiple operations without system conflicts. This capability is essential for modern robotic sys- tems that rely on real-time data processing and precise motor control.

As robotics technology continues to advance, the role of OS mechanisms such as PCBs will become increasingly significant in supporting complex robotic applications. Future work should focus on implementing and validating PCB-based scheduling in real robotic hardware and integrating artificial intelligence with robotic process management.

IX. LIMITATIONS AND FUTURE SCOPE

A. Limitations

- The study is primarily based on theoretical analysis.
- Limited access to real robotic systems for experimental validation.

B. Future Scope

- Implement PCB-based scheduling in real robotic systems.
- Study advanced real-time operating systems used in robotics.
- Integrate artificial intelligence with robotic process man- agement.

REFERENCES

1. M. Quigley et al., "ROS: An Open-Source Robot Operating System," in Proc. ICRA Workshop on Open Source Software, 2009.
2. A. Silberschatz, P. B. Galvin, and G. Gagne, Operating System Concepts, 10th ed. Hoboken, NJ, USA: Wiley, 2018.
3. A. Burns and A. Wellings, Real-Time Systems and Programming Lan- guages, 4th ed. Harlow, UK: Addison-Wesley, 2009.
4. S. Thrun, W. Burgard, and D. Fox, Probabilistic Robotics. Cambridge, MA, USA: MIT Press, 2005.
5. B. Siciliano and O. Khatib, Eds., Springer Handbook of Robotics. Berlin, Germany: Springer, 2016.
6. Robot Operating System Documentation. [Online]. Available: