TABLE II
CHALLENGES IN DEVELOPING ROBOT SOFTWARE ARCHITECTURE FOR THE AGRICULTURAL APPLICATIONS

| Challenges | Description |
|---|---|
| Sensor data processing | Agricultural robots rely heavily on sensor data to make decisions. The data may also account for different time frames. However, the data can be noisy, incomplete, or even incorrect, leading to wrong decisions. Developing software architecture that can handle such uncertainties and still make accurate decisions is a challenge. |
| Heterogeneity | Agricultural robots are used for a wide range of tasks, and therefore, the software architecture must accommodate the heterogeneous nature of the tasks. |
| Resource constraints | Agricultural robots often operate in remote areas and have limited resources, such as memory, processing power, and communication bandwidth. The software architecture must be designed to optimize the use of these resources. |
| Real-time processing | Agricultural robots need to be able to process data in real-time or near real-time in order to make decisions and take actions. This requires efficient and fast software architecture, which can be challenging to design. |
| Unpredictable environments | Agricultural environments can be unpredictable and change rapidly, which can make it difficult to design software architecture that can adapt quickly to changing conditions. |
| Energy efficiency | Agricultural robots often operate in remote areas, where access to power is limited. Developing software architecture that is energy-efficient and can operate for long periods on a limited power supply is a challenge. |
| Data management | Agricultural robots generate large amounts of data, which must be managed and processed in real-time. This requires efficient data management and processing systems, which can be challenging to design and implement. |
| Integration with existing systems | Agricultural robots must be able to integrate with existing agricultural systems, such as irrigation and fertilization systems. This requires careful design of software architecture to ensure seamless integration and compatibility. |
| Scalability | The software architecture must be scalable to accommodate the increasing demand for agricultural robots in the future. |
| Safety | Agricultural robots often work in close proximity to humans and livestock. Ensuring the safety of both humans and animals is a critical challenge in developing software architecture for agricultural robots. |
| Interoperability | Agricultural robots often work in collaboration with other robots and systems, and therefore, the software architecture must be designed to ensure interoperability. |
| Robustness | Agricultural robots operate in harsh environments, where they may encounter physical obstacles or suffer from mechanical failures and the software architecture must be robust enough to withstand these conditions. |
| Adaptability | Agricultural robots operate in dynamic environments, and the software architecture must be adaptable to changing conditions and requirements. |
| Cost-effective | Agricultural robots must be cost-effective, and the software architecture must be designed to optimize the cost of development and maintenance. |

development of robotic systems with human-like cognitive abilities, such as perception, reasoning, and learning. Examples, subsumption architecture [6] is a control architecture that emphasizes reactive behavior and task decomposition. It is composed of multiple layers, each of which is responsible for a different task or behavior. Higher layers can suppress the output of lower layers, allowing the robot to perform complex behaviors in a modular fashion; ACT-R architecture is a cognitive architecture that emphasizes decision-making, planning, and goal-directed behavior. It is based on the idea of production systems, which are rules that specify a condition–action pair. ACT-R models human cognition and is often used to predict human behavior in various tasks.

*2) Reactive Architecture Versus Deliberative Architecture:* Reactive architecture [7] focuses on the development of robotic systems that can quickly and appropriately respond to changes in the environment or stimuli, while deliberative architecture focuses on the development of robotic systems that can plan and reason about their actions and goals. Example, behavior Tree architecture is a reactive architecture that emphasizes task decomposition and decision-making. It is composed of a tree structure, where each node represents a task or behavior. The tree is executed in a top-down manner, allowing the robot to make decisions based on the current state of the environment; goal-oriented action planning is a deliberative architecture that emphasizes planning and goal-directed behavior. It is based on the idea of a plan library, which is a set of predefined plans that can be used to achieve specific goals. The robot selects the appropriate plan based on the current goal and the state of the environment.

*3) Layered Architecture Versus Hybrid Architecture:* Layered architecture [8] organizes the robotic system into distinct layers that provide different levels of abstraction and functionality, while hybrid architecture combines elements of different architectures to achieve specific goals or requirements. Example, model-view-controller (MVC) architecture [9], [10] is a layered architecture that separates the user interface (view), application logic (controller), and data (model) into separate components. This separation allows for greater modularity and flexibility, making it easier to modify or replace individual components; service-oriented architecture is a hybrid architecture that combines elements of both centralized and distributed architectures. It is based on the idea of services, which are self-contained, modular components that can be accessed remotely. Services can be combined in different ways to create complex systems, making it easier to adapt to changing requirements.

*4) Sense–Plan–Act Architecture Versus Behavior-Based Architecture:* Sense–plan–act architecture [11] divides the robotic system into three stages: sensing the environment, planning a course of action, and executing the plan, while behavior-based architecture focuses on the development of autonomous agents that can perform tasks through the interaction of simple behaviors. Example, hybrid deliberative/reactive architecture is a sense–plan–act architecture that combines elements of both deliberative and reactive architectures. It is composed of multiple layers, each of which is responsible for a different task or behavior. Higher layers can suppress the output of lower layers, allowing the robot to perform complex behaviors in a modular fashion; subsumption architecture (behavior-based architecture) is a behavior-based architecture that emphasizes reactive behavior and task decomposition. It is composed of multiple layers, each of which is responsible for a different task or behavior. Higher layers can suppress the output of lower