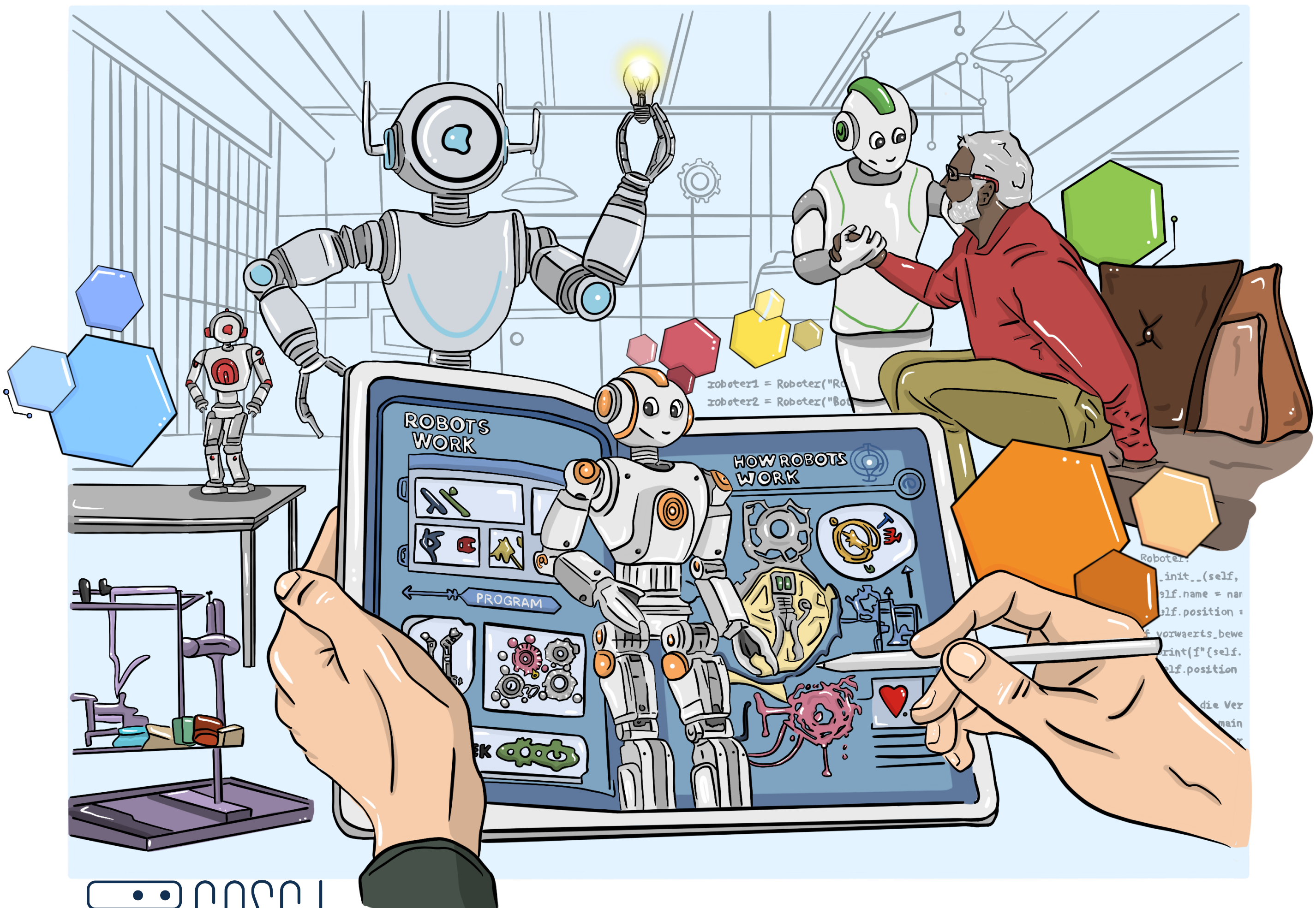
Open interactive textbook on

AI-powered and Cognition-enabled Robotics



EASE Research Center Bremen

Contents

Introduction 1

Robots have become an indispensable part of our modern world, contributing significantly to various industries and aspects of daily life. Figure 1.1 shows several examples of different robots accomplishing disparate tasks in diverse environments. Assembly and production robots are deployed in manufacturing and assembly lines making production processes more efficient and robust. Transportation robots work in warehouses for material handling and logistics. They navigate through warehouse environments, transporting goods and optimizing inventory management. Robotic vacuum cleaners have become common household appliances. They autonomously navigate living spaces, using sensors to detect obstacles and efficiently clean floors. Drones are a form of flying robots, which are used for tasks like aerial photography, surveillance, and delivery in various industries. Self-driving cars use a combination of sensors, cameras, and AI algorithms to navigate roads, make decisions, and potentially revolutionize the transportation industry. Inspection robots equipped with cameras and sensors are employed for inspecting and maintaining infrastructure, such as pipelines, bridges, and power lines, in challenging or hazardous environments. Agricultural robots are increasingly used for tasks like planting, harvesting, and monitoring crops. They aim to improve efficiency and reduce the need for manual labor.

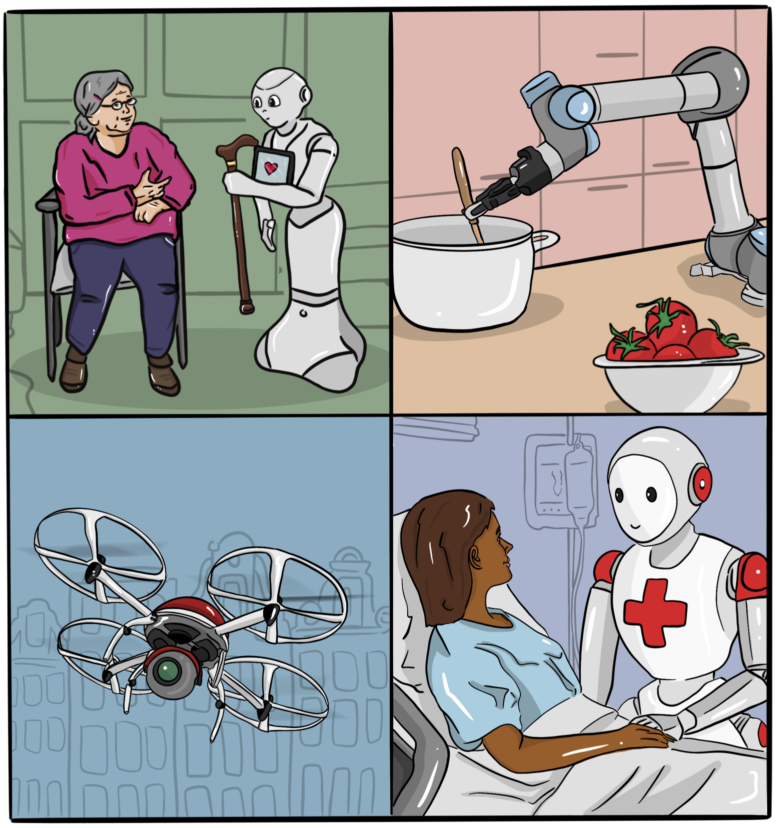


Figure 1.1: The figure shows four different robots accomplishing four different tasks in four different environments.

The practicality and success of these varied and impressive robotic applications is a result of the ingenuity, competence, and foresight of robot engineers. These experts meticulously define tasks and engineer the operational environments to preclude the necessity of executing open, unconstrained tasks in uncontrolled settings. By minimizing the complexities and unpredictability inherent to these tasks and environments, they successfully circumvent a vast amount and possibly open-ended number of individual challenges. This proactive approach obviates the need for overly intricate control programs, ensuring that the robots perform efficiently and effectively within their designated parameters.

One way to reduce complexity is to realize robot applications through single-purpose robots. Single-purpose robots are specifically designed for the respective tasks to excel in a particular function, enhancing their efficiency and reliability within a defined scope. Consider, for example, robotic vacuum cleaners or self-driving vehicles. Robotic vacuum cleaners are equipped with coverage algorithms. These algorithms enable the robot to systematically cover all reachable areas during cleaning, ensuring comprehensive coverage without redundancy. Autonomous vehicles are programmed for place-to-place navigation tasks. They use advanced navigation algorithms and sensor systems to move from one location to another without colliding with obstacles or other entities. This careful planning minimizes the risk of accidents.

If the tasks themselves cannot be sufficiently simplified, robot engineers sometimes try to further reduce complexity by structuring the environment. Structuring the working environment is crucial for supporting successful robot actions. This involves creating a controlled and optimized setting where robots can operate efficiently. Fixtures and mountings in an automobile factory, for instance, ensure that objects are positioned optimally for the robots to perform their tasks. In an automobile factory, robots are programmed to execute very fast and accurate motions with high reliability and repeatability. This precision is essential in manufacturing processes to maintain product quality and production speed.

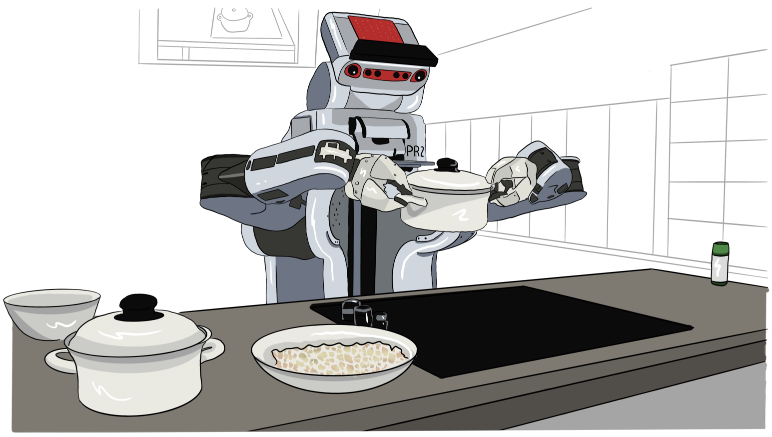


Figure 1.2: Modern mobile manipulation robot making popcorn.

Besides specialized, single-purpose robots, the realm of robotics is expanding to include general-purpose robots, such as the one illustrated in

Figure 1.2, which are becoming increasingly prevalent. These versatile robots typically feature designs inspired by the human form, providing arms and grippers for manipulation, and heads equipped with cameras that can be pointed into various directions. With their advanced motion and physical manipulation abilities, these robots are adept at performing a wide array of manipulation tasks. From setting and clearing the table to heating meals in the microwave, preparing popcorn, replenishing the coffee machine, and brewing coffee, their capabilities extend to a myriad of daily activities.

With the sensing and motion capabilities of these general purpose robots it is possible to accomplish a dynamically changing and expanding set of human-scale everyday manipulation activities in open human living environments – if their control programs manage the complexity of the necessary information processing tasks.

1.1 AI-powered and Cognition-enabled Robotics (AICOR)

In this book, we delve into the challenge of designing and implementing computer programs capable of controlling general-purpose robots. Our focus is on enabling these robots to autonomously execute a wide range of everyday manipulation tasks, ensuring they can be dynamically and intuitively tasked to perform such activities.

The control programs we envisage are to interpret naturally expressed task requests, like “bring me something to drink” or “clean up,” and proficiently carry out these tasks.

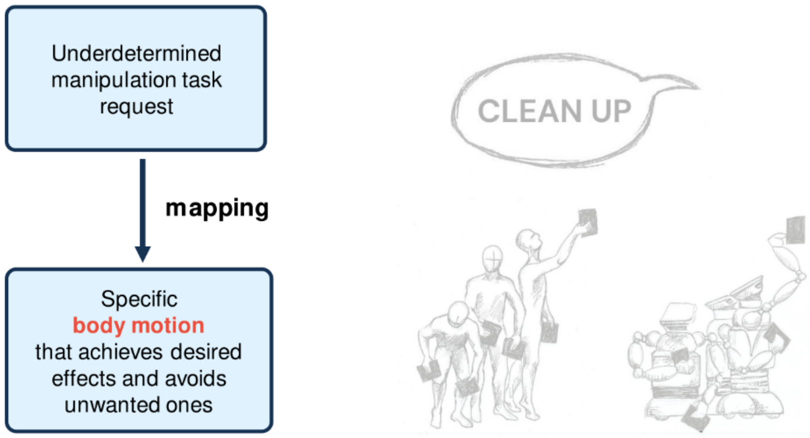


Figure 1.3: Body motion problem.

In order to do so, the control programs have to solve the **body motion problem**, which is illustrated in Figure 1.3 and defined below:

body motion problem

Given: a naturally formulated task request
infer and execute: a motion of the robot body that

- achieves the desired effects and
- avoid unwanted side effects.

Inferring the precise body movements required to fulfill an underdetermined task request represents an enormous computational challenge. This task goes beyond mere execution; it involves interpreting what somebody else wants one to do. It requires to understand how the physical world works and predicting the consequences of actions to choose action variations that will succeed. It also calls for comprehensive knowledge, commonsense, and intuitive physics reasoning. Necessary reasoning methods include informed decision making, learning from experience, prospection, action emulation, failure monitoring, diagnosis, and recovery, and planning intended courses of action based on predicted consequences of actions.

This task and how to solve it is studied and investigated in the field of AI-powered and cognition-enabled robotics (AICOR). AICOR represent a cutting-edge field where robotics are not only automated through artificial intelligence but also endowed with cognitive abilities resembling human-like understanding and decision-making. This integration aims to create robots that can interact more naturally with their environment and with humans.

AI-powered and Cognition-enabled Robotics (AICOR)

The interdisciplinary research field dedicated to the creation and advancement of such proficient robot control systems is termed "AI-powered and Cognition-enabled Robotics (AICOR)." This field synergizes cutting-edge and well-established methodologies from artificial intelligence and robotics, integrating them with principles and insights derived from models of human cognition.

Objective of AICOR

The objective of AICOR is to understand the design and the operation of robot control systems that can competently solve the body motion problem for natural and dynamically changing task requests, understand what they are doing and how as well as the consequences of their actions and translate this understanding into successful and trustworthy action.

AICOR robots hold immense promise in significantly enhancing the lives of many individuals, particularly those facing physical and cognitive challenges. Some of these individuals are confined to their beds, unable to lead independent lives, often reliant on others for assistance, and at times feeling like a burden. AICOR robots have the potential to bridge the gap between their needs and aims and their physical capabilities. By employing autonomous robots as assistive tools, these individuals could obtain what they need, precisely when they need it, autonomously, thus eliminating the need to seek help constantly. In this way, robots could markedly improve their quality of life, offering a higher level of independence and dignity.

AICOR robots capable of interpreting naturally stated tasks and translating them into successful action are applicable across a broad spectrum of domains. By assuming roles in perilous situations, such as rescue operations, these robots can be expected to minimize risks for humans. Furthermore, their integration is anticipated to yield substantial economic impact helping to sustain the workforce that is needed to secure our wellbeing. By relieving human workers from hazardous aspects of their jobs, these robots not only safeguard health but also augment productivity and quality of life.

Exploring the computational models underpinning AICOR robots not only advances our competence in designing and realizing robots but also propels progress in arguably the most profound scientific endeavor: unraveling the mysteries of the brain and mind, and deciphering the mechanisms that empower intelligent behavior.

1.2 Perspectives on robots

This section presents three key perspectives on robots:

1. The first perspective characterizes robots as software-controlled articulated electro-mechanical devices that accomplish their tasks by moving their body.
2. The second perspective is targeted at robots that are dynamically tasked with a variety of complex tasks that are to be accomplished in an open environment. In this case robots are best viewed as agents that have beliefs and goals and autonomously decide on the corse of action in order to achieve the robustness and flexibility for successful task completion
3. The third perspective considers the case in which the decision making has to be well informed in order to make the right choices. For example, in a chemical application the robot has to reason about possible chemical reactions before pouring one substance into another one. In this case it is helpful to think of the robot as an information processing or cognitive system.

1.2.1 Robots as software-controlled mechanical devices

Let us start with a definition of what we consider robots to be:

Robot

A Robot is an articulated electro-mechanical device that is operated and controlled by computer programs in order to accomplish tasks.

Robots have a physical body, which is an assembly of body parts including grippers, heads, a base, upper body, lower arms, and other components. The body parts are connected by joints, which are actuated by motors. The control program orchestrates the operation of these motors, enabling the robot to perform complex, coordinated movements: navigating by turning the base's wheels, aligning the head towards specific directions, or manipulating the arms and grippers to interact with objects. This intricate coordination allows the robot to change its posture and exert forces on its environment, thereby accomplishing tasks or, in some cases, leading to unintended side effects. The crux of the challenge for the robot's control program lies in interpreting a task request and devising a sequence of movements that ensures the achievement of the intended outcomes while at the same time mitigating any adverse effects.

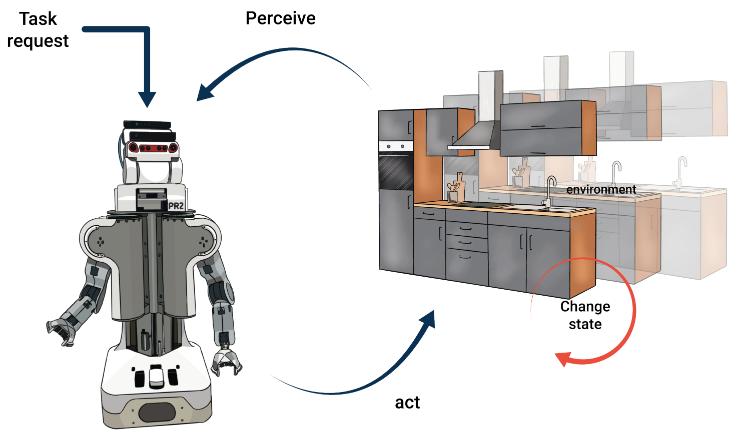


Figure 1.4: A robot agent accomplishing task requests by moving its articulated body as dictated by the robot control program by causing physical changes in the environment.

Figure 1.5 illustrates the structure of a state-of-the-art general-purpose mobile manipulation robot, highlighting several of its critical components in greater detail. The diagram focusses on components that endow the robot with its principal manipulation and perception capabilities. For manipulation, the robot's kinematic chain, which includes the shoulder, elbow, and hand joints, is pivotal. This chain facilitates the precise movement of the robot's end effector, the gripper, allowing it to attain specific poses and follows selected trajectories. The navigation base, equipped with steerable wheels, provides the robot mobility, enabling it to traverse and position itself within its operational surroundings. For perception, crucial sensors are integrated into the robot's design. Laser sensors measure distances to obstacles in their path, providing spatial awareness, while cameras capture visual data from the environment. This visual input allows the robot to process and interpret task-relevant information, playing a crucial role in its interaction with the surrounding world.

Often, the directives and information contained in a task request are not sufficient to specify an appropriate sequence of detailed robot movements. Consequently, the robot must perceive and interpret the task's context to bridge these information gaps. To do so, the robot relies on its sensors. These sensors are designed for measuring various physical parameters

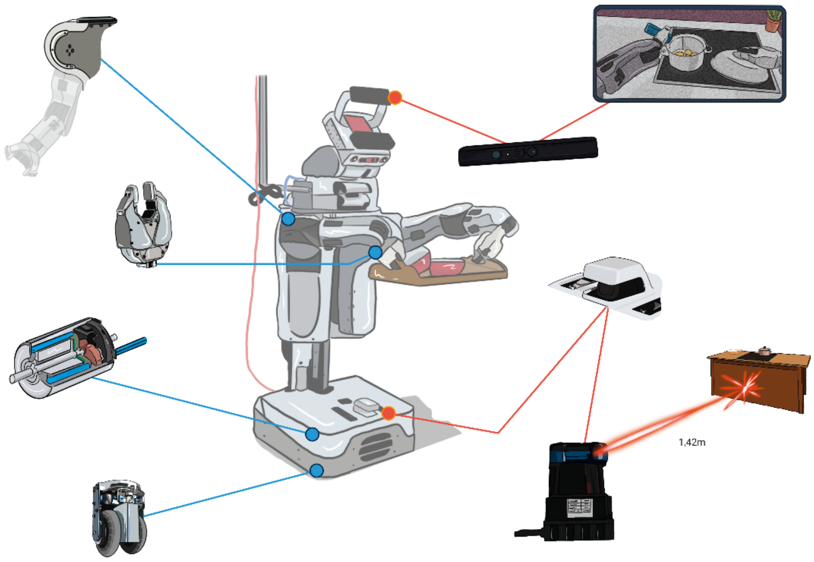


Figure 1.5: The mobile manipulation robot: a PR2 robot produced by Willow Garage.

related to both the robot's own structure and its external environment. For instance, force sensors enable the robot to measure the amount of pressure it exerts on objects, while encoders measure the extent of joint movement, even detecting if a motion is hindered or stalled. Additionally, other sensors are attuned to environmental attributes: contact sensors identify collisions between the robot and its surroundings, distance sensors ascertain the proximity of nearby objects, and cameras capture visual snapshots of the robot's environment. The data acuired by these sensors provide the control program with raw information about the robot's status and its operational context, information that is indispensable for the successful execution of tasks.

1.2.2 Robots as agents

As previously discussed, robot control systems are to solve the body motion problem. This challenge escalates when dealing with general-purpose robots, where task requirements are dynamic, open-ended, and potentially multifaceted. General-purpose robots may be called upon to execute a variety of tasks, each with its own complexity and structure. These tasks are often abstractly defined, lacking sufficient detail, thereby necessitating the acquisition of additional information during task execution to determine suitable body motions. Moreover, these robots must possess robust failure detection mechanisms and recovery protocols. These factors contribute to the complexity inherent in designing and operating general-purpose robotic systems.

For example, imagine a meal preparation robot that is able to cut slices of bread. Since this is a very specific task, a general-purpose robot should also be able to slice a cucumber, or quarter a peach. For this, the robot needs to know how cutting, slicing and quartering relate to each other, and most importantly, how the task request can be translated to appropiate body motions that achieve the desired result.

During task execution, a robot must continuously infer the most appropriate body motion, considering its current knowledge of the task, learnings from ongoing actions, and assumptions about the environment. This requires bridging the gap between the limited information provided by the task request and the detailed, context-specific information necessary for precise manipulation in varying environments. As depicted in Figure 1.6, this gap is bridged by the robot's knowledge, and its perception and reasoning capabilities.

	context-specific body motion
-	**vague task request**
=	**perception & knowledge & reasoning of the robot**

Figure 1.6: The gap between the information needed to generate the context-specific motions for table setting and the information contained in the task request has to be filled through the knowledge and the reasoning capabilities of the robot agent.

Given the unpredictability of tasks and environmental conditions, it is impractical for robot engineers to anticipate all potential reasoning tasks and actions during the design phase. Instead, robot control systems must be imbued with the ability to autonomously make decisions, showcasing adaptability, dependability, and efficiency in diverse and uncertain scenarios and contexts.

To promote autonomous decision-making, we conceptualize control programs as robot agents capable of independently executing human-scale tasks, as illustrated in Figure 1.7. Viewing robots as agents involves modeling them as entities with cognitive capabilities, where behavior is guided by desires, beliefs, and intentions. These agents strive to fulfill task requests robustly and efficiently, aligning with the preferences of the individuals they serve. They formulate and maintain beliefs about task-relevant contexts to make informed decisions and intend to act rationally, optimizing their performance based on predefined metrics.

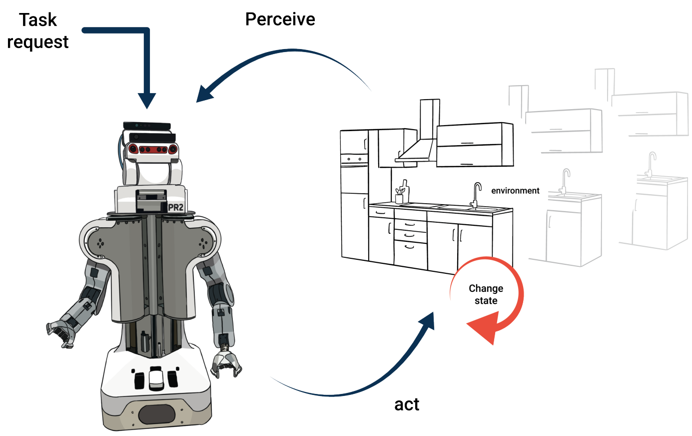


Figure 1.7: Top-level model of robot agents

In this conceptual framework* robot agents are robots that act in an environment in order to change the state of the environment to achieve goals as dictated by the task requests. The framework enables us to describe the interaction of robots and the environment they act in, how goals and tasks of the robots can be stated and the goal achievement through robot actions be measured, and how robots should select their course of action in order to maximize the impact of its actions.

In the rational robot framework a *robot agent* is conceptualized as an entity that acts in an *environment* in order to achieve its goals. The agent perceives the environment through its *sensors* and changes the state of the environment through its physical actions. The agent is controlled through a function that maps percepts from its sensors and prior knowledge into an action that the robot executes. We further conceptualize the processes with which robots decide on their course of action and how the actions change the environment as an iterative interaction between the robot and the environment it is operating in. In each iteration the robot agent

1. perceives the state of the environment,
2. decides on the next action, and
3. executes the action in order to change its environment.

The repeated execution of the steps (1.) to (3.) forms a so-called *perception-action loop*.

1.2.3 Robots as information processing entities

To better understand cognitive requirements for robot control as agents, it is insightful to draw from human cognitive capabilities. The human brain demonstrates exceptional skill in managing tasks with versatility, resilience, and creativity, especially evident in remote robot operation.

Imagine a scenario: a person connects a game controller to a robot and uses virtual reality glasses for immersion in the robot's environment (refer to Figure 1.8). The person becomes the puppeteer of the robot's movements, adeptly guiding it through various tasks from household chores to intricate manual tasks. This scenario not only showcases the potential of human-guided robotics but also highlights a key insight: successful world interaction is essentially about processing information.

video of a robot being remotely controlled to clean a living room

Through the game controller and virtual reality glasses, the person processes visual information from the robot's cameras, makes decisions, and translates these into commands, resulting in the robot's physical actions. This demonstrates the embodiment of human cognitive reasoning in a robot: humans leverage their cognitive skills to process information and make decisions, while the robot's actuators implement these decisions in the physical world. This interaction exemplifies the general, robust, flexible, and competent control humans have over robots, achieving tasks with remarkable adaptability and problem-solving capabilities evidenced through:

* To this end, we adopt the model of a *rational robot agent*, which is inspired by the original definition of a rational agent by [**russell10aima**].

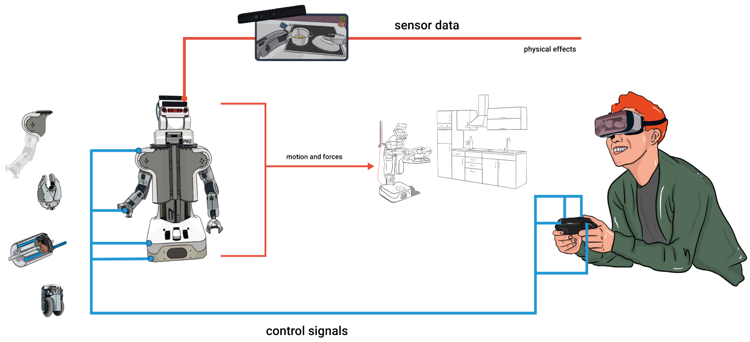


Figure 1.8: Remote control of a mobile manipulation robot with a game controller.

- Generality and Flexibility: Humans can seamlessly adapt their control strategies to work with different robots, handling various objects and tools. This adaptability extends to performing tasks in different environments, showcasing a remarkable generalization of skills.
- Competence Across Contexts: Humans can accomplish tasks in a range of contexts, including situations where additional considerations, such as the presence of a small child, come into play. This highlights the robustness and contextual awareness inherent in human control over robots.
- Handling Novelty: Humans can proficiently tackle variations of tasks with novel objects and in unknown environments. This ability to adapt to unforeseen circumstances underscores the flexibility and problem-solving acumen of human operators.
- Learning from Various Sources: Humans can learn to accomplish novel tasks through diverse sources, such as reading instructions, watching instruction videos, or interacting with a teacher. This learning process involves understanding the task, its nuances, and potential risks.
- Understanding and Communication: Humans possess a deep understanding of their actions, evident through their ability to answer questions about what they are doing, why they are doing it, and how. They can anticipate the consequences and risks of their intended actions and consider alternative courses of action.
- Dynamic Collaboration: Humans can dynamically adapt their task interpretation and preferences when jointly accomplishing a task with another human. This collaborative aspect involves a shared understanding and synchronized effort toward task completion.

In essence, the remarkable capabilities demonstrated by humans in controlling robots highlight the synergy between information processing, or better cognition, and physical action.

In this book we investigate the question of whether we can replace the human in our setting with a computer program that can perform these information processing tasks autonomously, see Figure 1.9.

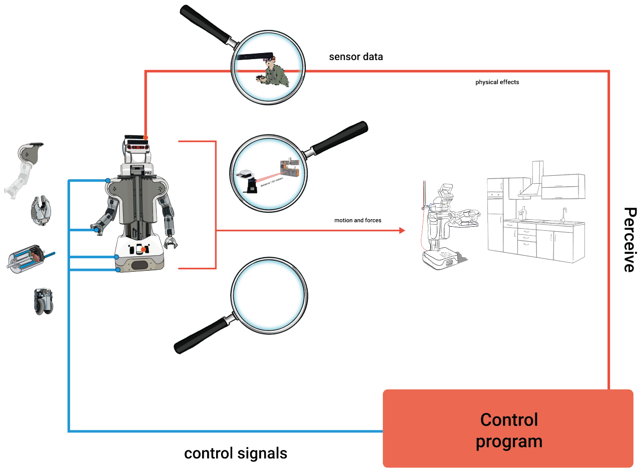


Figure 1.9: Control system model

1.3 Target capabilities of AICOR robot agents

Figure 1.10 illustrates the cognitive capabilities of the robot agents we investigate in this textbook. These include robot agents that can accomplish tasks in a generalized manner, for example, transporting any object from any place to any destination, given they have the bodily capability to do it; robot agents accomplishing complex manipulation tasks requiring them to understand how the world works in order to act successfully in it; robot agents that learn novel task variations by reading instructions and watching instruction videos, requiring to recognize and understand task-critical motion patterns; finally, robot agents accomplishing joint tasks with humans requiring them to negotiate, infer, and satisfy shared task interpretations.

The highlighted robot agents encompass a spectrum of tasks, each demonstrating varying degrees of complexity and cognitive capabilities. First, we delve into the realm of a robot agent engaged in human-scale everyday transportation tasks, such as the nuanced activities of setting and cleaning a table. This example illustrates the adaptability and dexterity required for robots to seamlessly integrate into common household activities.

Moving to the domain of meal preparation, our exploration extends to a robot agent tasked with accomplishing simple yet intricate meal preparation tasks. Here, the focus is on manipulating and altering the physical state of objects and substances, demanding a fine-tuned coordination of robotic actions and reasoning about the consequences of actions.

Taking a leap into more advanced capabilities, the book delves into robot agents that can learn novel variations of manipulation tasks by leveraging external sources, such as web pages like WikiHow and instructional videos. This underscores the robot's ability to acquire new skills and knowledge autonomously and thereby evolve it's understanding of how the world works and how to successfully act in it.

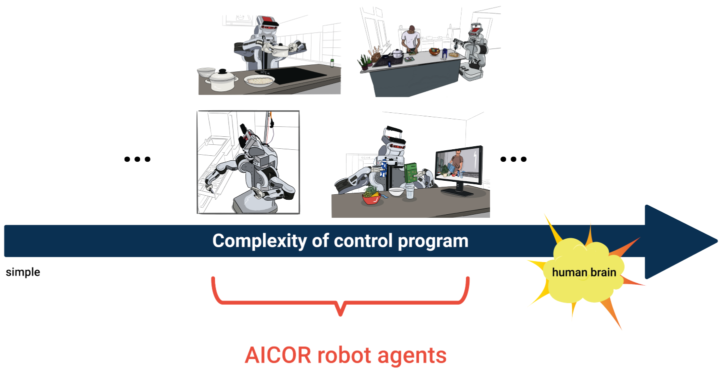


Figure 1.10: The focus of this book are robot agents, that is robot control programs that can be best understood by being attributed with beliefs, goals, and intentions and that have a substantial degree of autonomy that gives them robustness, flexibility, and goal-directedness.

The pinnacle of complexity within the book's scope lies in the exploration of robot agents capable of joint manipulation tasks alongside humans in human-scale everyday scenarios. This entails a unique set of challenges, requiring the robot agents to discern and fulfill human intentions rather than relying solely on pre-programmed or learned instructions. The emphasis here is on collaborative and adaptive behavior, showcasing the potential for robots to engage in cooperative tasks within real-world environments.

Throughout the book, we will dissect and analyze these diverse robot agents, exploring the intricacies of their control systems, learning mechanisms, and adaptive decision- making processes.

1.4 The AICOR virtual research, education, and training building

The AICOR learning environment provides a digital platform specifically designed to study, conduct research, and work in the field of AI-powered and cognition-enabled robotics. At the core of this platform is the AICOR Virtual Building (AICOR ViB), as illustrated in Figure 1.11. AICOR ViB, a digital hub, serves the dual purpose of research and education, offering virtual tours and interactive experiences within the domain of AICOR.

AICOR provides a comprehensive and immersive learning and research environment, which includes the following components:

- An **education floor**, which contains various learning resources:
  - the EASE learning hub provides
    - a collection of video lectures on selected topics in cognition-enabled robot manipulation held by leading experts in the field
    - several virtual tutorials on software components of AICOR robots

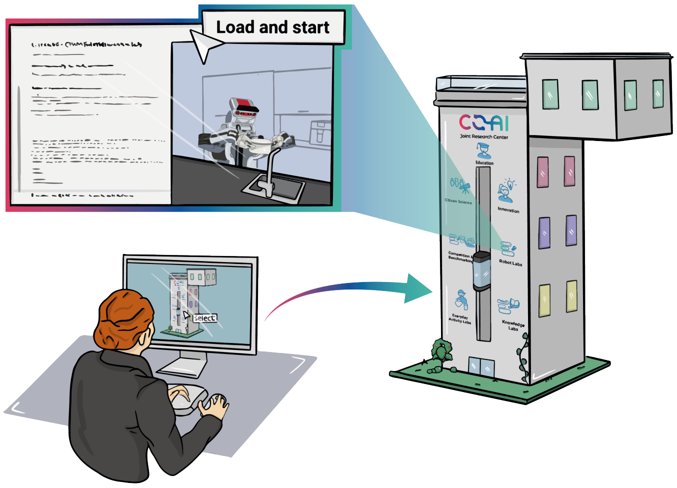


Figure 1.11: AICOR virtual research and training building.

  * information for the resident students at the University of Bremen.
- several virtual research laboratories facilitating research-oriented education and training such as the
  * household transportation task lab investigating cognition-enabled robot agents accomplishing transportation tasks in human environments
  * meal preparation lab investigating how to design and realize generalized robot plans for categories of everyday manipulation tasks, such as cutting, pouring, whisking, wiping, etc. The focus is on designing plans that can accomplish a task on any object or substance, with any tool, for any purpose, and in any context.
  * actionable knowledge graph lab/ robot skill and competence acquisition lab combines web-based abstract knowledge acquisition with embodied self-programming and learning to learn to acquire new task variations
- David Vernon's comprehensive resources with a Wiki providing a large set of excellent pointers into the field. Most notable are
  * a link to David Vernon's course on cognitive robotics
  * a link to David Vernon's course on artificial cognitive systems accompanying his textbook with the same tile

Within AICOR ViB, digital twins of actual robotics research laboratories are available. These virtual environments provide an opportunity for engaging in practical exercises that are in direct correlation with the textbook's material. The platform allows students to customize their learning experience by selecting a robotic task, choosing a robot, and defining the operational environment. This level of user interaction makes AICOR ViB a functional tool for academic pursuits in AI-powered and cognition-enabled robotics.

During a visit to a ViB laboratory, users can select a task, a robot, and an environment for the robot's operation. The selected components are integrated into a software container†, which can be downloaded and utilized on a personal computer or operated in the cloud. Access to the open-source code of the robot control systems is available in these virtual research laboratories. The laboratories, involving robots, environments, and tasks, are represented as knowledge bases, making them understandable and interpretable by machines. Additionally, experimental data from sessions are automatically recorded and can be interactively analyzed using the openEASE web-based knowledge service, enhancing the reproducibility of research. AICOR ViB thus provides a powerful infrastructure suitable for a range of academic activities, including software projects, thesis research, and participation in robotics competitions like RoboCup@Home.

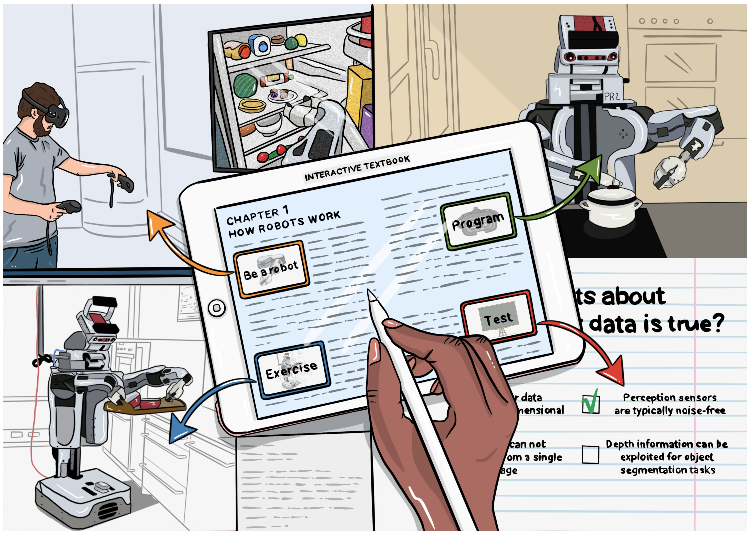


Figure 1.12: The AICOR interactive textbook.

In conjunction with studying the book chapters, different forms of interactive learning materials are accessible, as depicted in Figure 1.12. These resources include video lectures and tutorials offering detailed insights that complement the written content. Moreover, game-like environments are available where users can embody a robot avatar to undertake manipulation tasks. Users also have the opportunity to query and interact with the knowledge bases of robots and robotic experiments, facilitating the visualization of answers, data compilation for machine learning training sets, and the execution of programming exercises. These exercises provide a practical context to validate the efficacy of solutions, extending to complete robot control programs and real robotic systems.

As a student you have a digital desktop for managing all your learning activities. The desktop provides access to the courses you are taking, the exercises you have to complete, the literature you have collected, and your thesis research. The desktop is connected to learning management system, which in our case is Moodle (Modular Object-Oriented Dynamic Learning Environment). The learning management system serves as a platform for educators to create and manage courses online, providing tools to facilitate both asynchronous and synchronous learning. Features of the learning environment include the ability to post and organize course content, conduct quizzes and assessments, manage enrollments,

† This integration is facilitated through Docker technology.

and facilitate communication through forums, chats, and messaging systems.

1.5 Robot Agents that . . .

In this section we explore how robot agents accomplish different kinds of everyday activities that actually require a little amount of body movements (often pick, carry, place motions), but in large variations that require semantic domain knowledge.

For each of the following exemplary tasks and the environments they are executed in, different domain knowledge needs to be accessed.

1.5.1 . . . accomplish everyday transportation tasks

The research laboratory presents a robot agent that executes tasks including setting the table, cleaning up after eating, and loading and unloading the dishwasher. The collection of experiments show how general a robot control system can be programmed if it employs knowledge and reasoning.

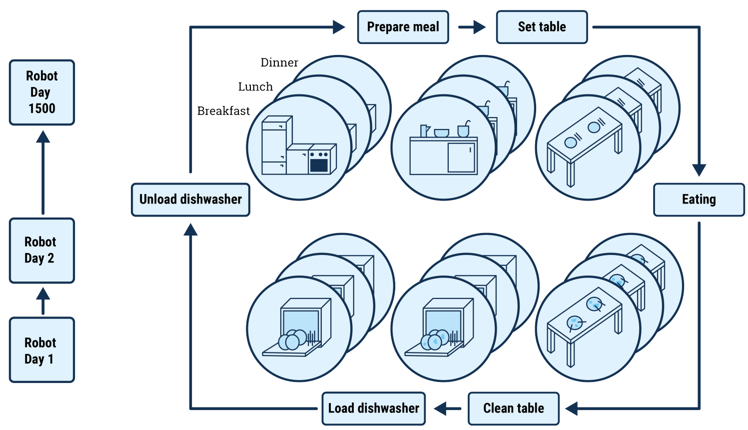


Figure 1.13: The household challenge: for a robot lifetime of robot days perform for each meal at each day set the table, clean the table, load the dishwasher, and unload it afterwards.

The control program of the robot operates based on a fundamental principle: "put things where they belong." This principle breaks down into a series of sophisticated pick and place actions. For instance, when setting the table for breakfast, the robot:

- Opens the drawer to fetch clean tableware.
- Picks up a cereal box, a cup, and a milk bottle from their respective storage spots.
- Arranges the items neatly on the table, ensuring the setup is appropriate for the meal.

After the meal, the robot:

- Clears the table, carefully handling the fragile tableware.

- Loads the dishwasher with the used items, optimizing space for efficiency.
- Cleans the table surface, preparing it for the next use.

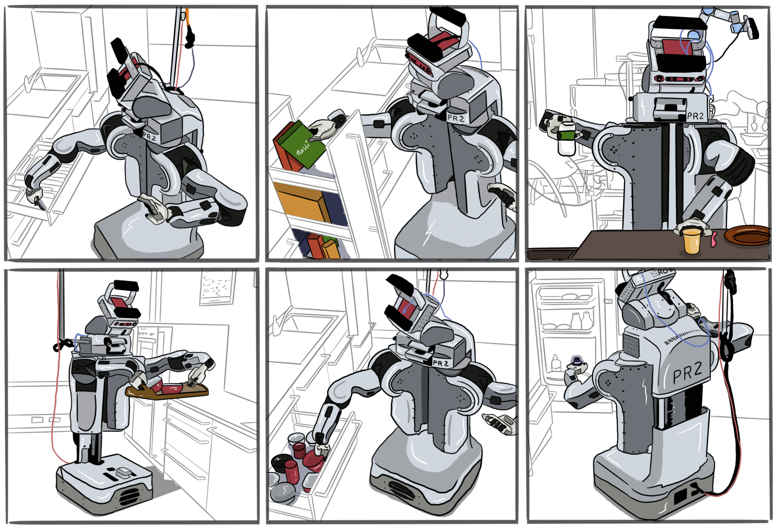


Figure 1.14: The robot agent performing a variation of pick up actions as part of the household challenge: (1) opening a drawer, (2) picking cereal, (3) placing cup and milk, (4) carrying a tray, (5) picking a bowl, (6) placing milk.

The robot is equipped with an extensive knowledge base, storing detailed information about various household routines and preferences. It understands that table settings differ between breakfast, lunch, and dinner and adjusts its actions accordingly. The robot's adaptability is highlighted by its ability to recognize and handle tableware, acknowledging its fragility and the possibility of stacking items efficiently.

For instance, in a specific kitchen setup, the robot identifies the storage locations of tableware, even if they vary from one kitchen to another. However, if placed in a new environment where the storage locations are unknown, such as a different kitchen or a storage room, the robot may require updates to its knowledge base to continue performing efficiently.

With its sophisticated task execution, adaptability, and extensive knowledge base, this robot represents a significant leap forward in household automation. However, achieving such a level of functionality and intelligence in a robot involves overcoming substantial challenges, particularly in developing a rich knowledge base and ensuring the robot's adaptability to diverse household environments.

1.5.2 . . . work in a retail store

In the domain of retail, robotics is beginning to revolutionize the shopping experience, mirroring the advancements in household robotics. The primary operations of these robots remain pick and place actions, but the overarching goal shifts to "looking for and ordering things." While single-task robots have made their presence felt in storage rooms of large logistics firms, multifunctional robots are gradually making their way to the shop floor, exemplified by shelf scanning robots used for stocktaking (as visualized in Figure 1.15).

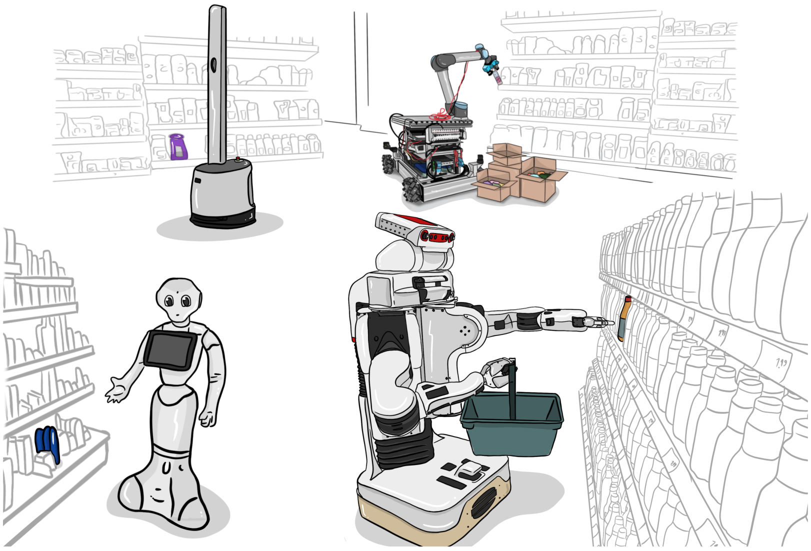


Figure 1.15: Robot agent building a model of a retail store. Left: Robot identifying objects in the store with its camera. Right, top: Perception results of the robot. Right, bottom: Object recognition of the perception system.

The stocktaking robot is designed to:

- Recognize individual shelves and their respective levels.
- Detect and read the price tags of products on each shelf level.
- Count the number of products placed consecutively. Compile and update all the gathered information into a database.

This robot autonomously builds a model of its environment to navigate and perform tasks effectively. However, its capabilities are tailored to the structured environment of retail stores, which are characterized by standardized layouts with shelves, shelf levels, and products. The identification of products is facilitated by barcodes, and the positioning of products facing the customers simplifies perception and interaction.

Despite the structured nature of retail environments, shopping or service robots face numerous challenges, especially in real-time, customer-centric settings:

- Customer Traffic: Navigating through and operating in crowded spaces.
- Customer Preferences: Understanding and adapting to individual customer needs and behaviors.
- Fast Changing Products: Keeping up with the frequent changes in product placements and new stock.
- Misplaced Products: Identifying and dealing with products that are not in their designated spots.
- Narrow Spaces: Manipulating and picking products in tightly packed shelves.

To address these challenges, shopping robots must link the perceived information (like barcodes) to customer demands. For instance, if a customer is looking for the cheapest toothpaste, the robot must identify which barcodes correspond to different toothpaste brands and determine the most cost-effective option. This requires integrating extensive product knowledge, potentially sourced from web stores and online product databases. Such integration enables these robots to assist customers effectively by helping them locate and identify products based on specific criteria.

Shopping assistant robots represent a significant step towards automating and enhancing the retail experience. By combining sophisticated perception abilities, comprehensive product knowledge, and customer interaction capabilities, these robots have the potential to transform the shopping landscape. However, the transition from structured, predictable environments like storage rooms to dynamic, customer-driven shop floors introduces a set of challenges that necessitate advanced solutions in robot design, environmental understanding, and customer service automation.

1.5.3 . . . prepare simple meals

Another robot you can find in the AICOR ViB is the popcorn making robot. For a visual introduction of the task, consider the snapshots of the cooking activity depicted in Figure 1.16. Additionally, a comprehensive demonstration of the robot performing the complete popcorn preparation task can be viewed at the provided YouTube link `https://www.youtube.com/embed/cTCJSNjTdo0?si=dED7tOiqp9lujkOV`.

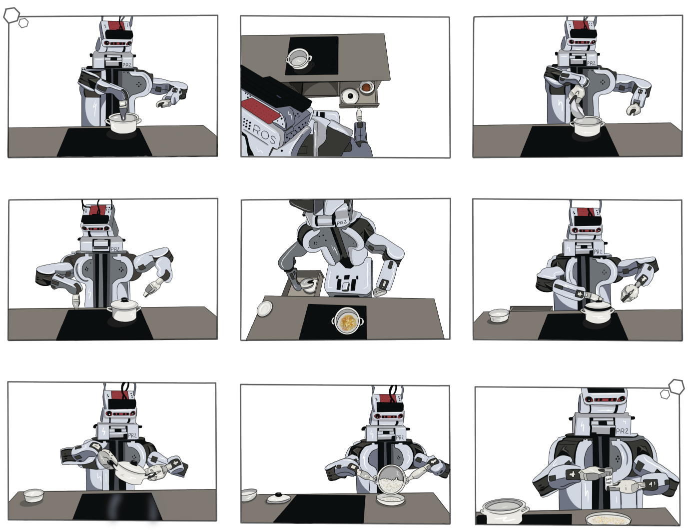


Figure 1.16: Action steps for popcorn making: (1) putting the cooking pot on the stove, (2) opening the drawer, (3) pouring the corn into the pot, (4) switching on the drawer, (5) grasping the lid, (6) putting the lid on the pot, (7) distributing the corn evenly in pot, (8) pouring the popcorn onto the plate, (9) salting the popcorn.

The AICOR ViB's popcorn-making robot represents a pinnacle of robotics, turning a simple instruction like "make popcorn" into a showcase of

VR exercise: Making popcorn in a virtual environment ("Be a robot")

advanced robotics and AI capabilities. This task, while straightforward in appearance, encompasses a wealth of complex, underlying processes that epitomize the intricacies of robotics in practical applications.

Analysis of Task Complexity

- Decomposition of the High-Level Instruction: The robot must dissect the command into actionable steps. It involves understanding the sequence of operations, such as acquiring popcorn kernels, measuring them, and setting up the cooking appliance.
- Importance of Ordering and Timing of Actions: Ensuring the correct order and timing of actions is vital. The robot must comprehend the sequence that leads to successful task completion, like not turning on the microwave prematurely.
- Necessity of Sensorimotor Coordination: Accurate sensorimotor coordination is essential. The robot navigates the kitchen, handles objects (like a popcorn packet), and monitors the cooking, adapting to the specific environment and tools.

Understanding and Interaction with the Environment

- Environmental Understanding and Self-Awareness: The robot requires comprehensive knowledge of the kitchen environment, including the locations of items and how to operate appliances.
- Procedural Knowledge and Action Sequence Dependencies: Understanding the sequence of actions and their dependencies is crucial, like knowing to place the pot on the stove before heating it.
- Sensory Feedback and Monitoring: The robot must monitor the process through sensory feedback, like recognizing the sound of popcorn popping.

Advanced System Integration

Control program exercise

- Environmental Mapping for Object Recognition: The robot uses sophisticated mapping to recognize objects.
- Task Planning Algorithms: Algorithms are used to deduce and order the steps from a high-level instruction.
- Control Systems for Precise Object Interaction: Precise interaction with objects is achieved through advanced control systems.
- Sensory Processing and Learning Mechanisms: The robot adapts to new environments or changes through advanced sensory processing and learning mechanisms.

Physical and Interactional Considerations

- Handling of Different Objects: The robot considers the physical characteristics like weight, shape, and temperature of objects for interaction.
- Task-Specific Knowledge: Knowledge of specific tasks, like where popcorn is stored or how to operate a salt grinder, is crucial.

Example Task: Popcorn Making The process involves several steps, each requiring context-dependent execution:

1. Picking and placing an empty pot on the hotplate.
2. Turning on the hot plate.
3. Handling the corn bowl and adding corn to the pot.
4. Placing the lid on the pot.
5. Shaking the pot to prevent burning.
6. Monitoring until the popcorn is ready.
7. Turning off the hot plate.
8. Transferring the popcorn to a bowl and placing the pot in a safe area.

1.5.4 ... assist in laboratories

In the realm of scientific research and testing, robots are increasingly being introduced to assist with intricate assembly tasks, such as compiling chemical test kits (as shown in Figure 1.17). These tasks demand a nuanced understanding of physics and material properties, far beyond the requirements of typical household or retail assistance robots. Laboratory assembly robots must manipulate delicate and often minuscule components, necessitating a sophisticated blend of compositional knowledge, material awareness, and functional understanding.

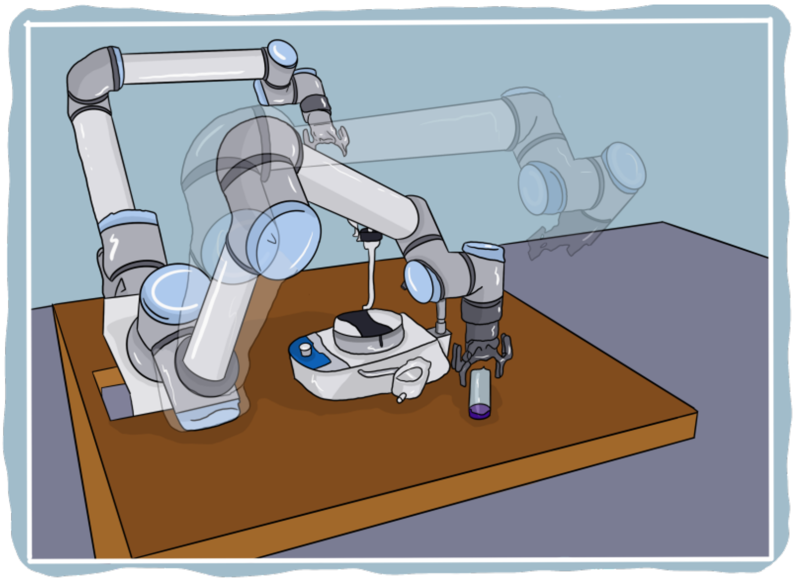


Figure 1.17: Robot agent assembling sterility test kits in a medical laboratory. The transparent arm simulates the planned body movement of the robot in order to calculate success.

To manage the intricate assembly tasks typically found in a laboratory, a robot must possess:

- Compositional Knowledge: Understanding the assembly process, akin to how humans interpret instruction sheets.
- Material Knowledge: Recognizing the properties of various materials, such as the fragility of glass or the malleability of rubber, and adapting manipulation strategies accordingly.
- Functional Understanding: Identifying the purpose and proper application of each component within the assembly, ensuring that

each part is used correctly, like placing a rubber plug on the top of a glass tube, not at the bottom or side.

Laboratory environments offer certain advantages that facilitate the successful deployment of robots:

- Structured Environments: Labs are meticulously organized, with each tool and component having a specific place and purpose.
- Minimal Human Traffic: Unlike retail or household settings, labs typically have fewer people moving around, reducing the complexity of navigation and operation.
- Limited Object Variability: The number of different objects and materials a robot must recognize and handle is relatively small, allowing for more focused and specialized knowledge bases.
- Detailed Action Sets: The tasks are well-defined with specific steps and sequences, enabling robots to follow precise instructions without requiring significant on-the-fly decision-making.
- Consistency in Tasks: There's minimal variation in the tasks performed, allowing robots to perfect specific routines without needing to adapt to new or unexpected scenarios frequently.

Laboratory assembly assistant robots exemplify the integration of advanced robotics in high-precision, high-stakes environments. These robots, equipped with detailed knowledge of materials, physics, and functional applications, are capable of handling delicate and complex tasks with precision and consistency. The structured nature of laboratory environments further contributes to their success, providing a controlled setting that maximizes the robots' efficiency and accuracy. While these robots currently operate within a relatively narrow scope of tasks, their potential to revolutionize laboratory work by enhancing precision, reducing manual errors, and increasing efficiency is profound.

1.5.5 ... are ocean scientists

Transitioning from the structured confines of human-made environments to the vast and unpredictable realm of nature, underwater robots designed for scientific research represent a pinnacle in robotics engineering. These robots are deployed in dynamic and often harsh natural environments to observe and analyze ecosystems over extended periods. Their tasks and operational challenges are fundamentally different from those encountered in controlled settings, demanding a unique set of capabilities and design considerations.

Underwater research robots must be equipped to handle the complexities of natural settings, which include:

- Advanced Sensory Capabilities: Possessing sensors that can navigate and gather data in conditions with low light and high reflection, typical of underwater environments.
- Autonomous Functioning: Operating independently for prolonged periods without the need for external control, often in areas where human intervention is not feasible.
- Self-Repair Mechanisms: Having the ability to perform diagnostics and basic repairs autonomously to ensure continued operation and return to the surface if necessary.

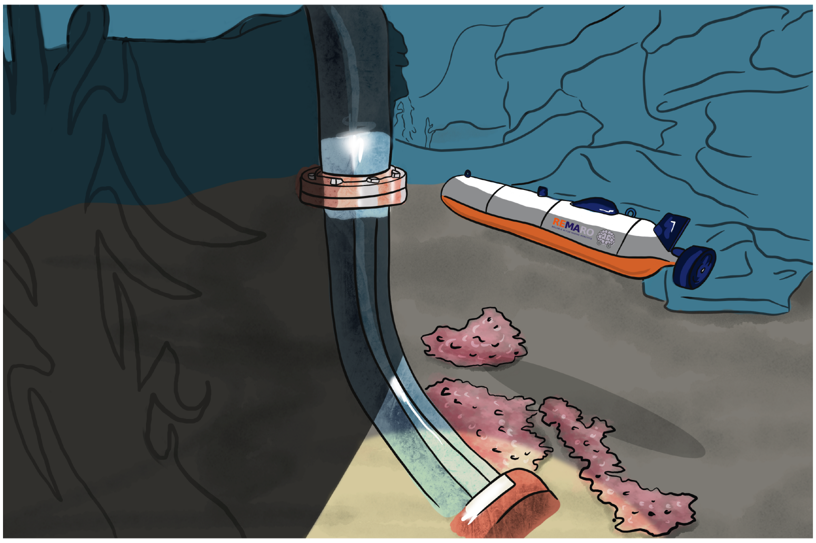


Figure 1.18: Underwater env 1

- Accurate Position Estimation: Maintaining precise navigation and positional awareness even in adverse conditions, where conventional systems like GPS are not operable.
- Adaptive Behavior Modeling: Understanding and predicting the behavior of living organisms in their natural habitat, accounting for both short-term actions and long-term patterns like breeding seasons or coral growth.

Building robots capable of conducting research in natural underwater environments poses significant challenges:

- Environmental Robustness: Designing systems that can withstand pressure, temperature, and salinity variations, along with physical obstacles and unpredictable elements.
- Energy Efficiency: Ensuring the robot can manage its energy resources efficiently, especially crucial when operating autonomously over extended periods.
- Data Processing and Transmission: Handling the collection, processing, and, where possible, transmission of vast amounts of data, often with limited bandwidth or in delayed transmission scenarios.
- Interaction with Living Organisms: Developing non-intrusive methods to observe and interact with marine life, ensuring that the robot's presence does not adversely affect the natural behavior and balance of the ecosystem.

Underwater research robots in natural environments represent an advanced frontier in robotics, where the machines are not just tools but also explorers and observers of the unknown. These robots hold the promise of unlocking mysteries of the underwater world, providing insights into complex ecological patterns and the effects of environmental changes. The design and operational challenges they face push the boundaries of current technology, driving innovation in robotics, materials science, and environmental science.

1.5.6 . . . learn to prepare meals

Elevating the capabilities of household robots and robots that prepare simple meals, the cooking assistant robot represents a significant leap in domestic robotics. Unlike the relatively structured tasks of setting and cleaning the table, cooking introduces open-ended task categories with a high degree of variability and complexity. This robot is to prepare meals, handling a wide range of ingredients, kitchen tools, and cooking techniques.

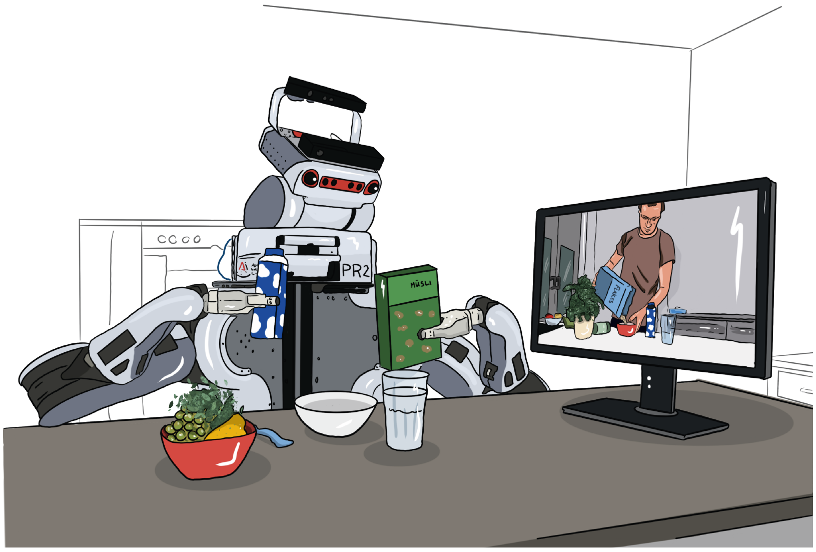


Figure 1.19: Robot agent learning to prepare meals by watching instruction videos.

The cooking robot is required to perform sophisticated cooking actions, such as:

- Cutting and Peeling: Precisely handling a variety of textures and shapes of fruits, vegetables, and other ingredients.
- Mixing and Stirring: Understanding the required consistency and applying the appropriate technique for different dishes.
- Differentiating Pouring Techniques: Recognizing when to pour ingredients into a container versus pouring through a strainer or sieve.
- Manipulating Complex Objects: Opening jars, milk cartons, cereal packs, and other packaged food items with varying levels of difficulty.

Cooking involves not only the mechanical execution of tasks but also a deep understanding of the process and sequence of actions. Instructions that are intuitive to humans often lack the explicit detail required for robotic comprehension. For example:

- Understanding Implicit Instructions: Instructions like "Add the milk to the dough, mix it, and pour it into a pan" are inherently understood by humans but require explicit contextual understanding and sequencing for a robot.

- Differentiate between Task Requests: A general-purpose robot needs to be able to relate instructions like cutting, slicing and quartering and differentiate them in the ways they influence body motions.
- Learning from Demonstrations: An effective approach for teaching cooking to robots involves learning from demonstrations, where robots observe and interpret human actions. This method allows robots to perceive executed actions and understand the nuances of task variations by comparing different demonstrations (as illustrated in Figure 1.19).

For a cooking robot to operate effectively, it must:

- Adapt to Different Kitchen Environments: Recognize and adapt to the varying layouts, storage solutions, and equipment found in different kitchens.
- Understand Recipe Variations: Interpret a wide array of recipes, accounting for the inevitable variability and occasional ambiguity in cooking instructions.
- Learn from Human Behavior: By analyzing demonstrations, the robot can accumulate knowledge about cooking techniques, ingredient handling, and the sequence of steps involved in preparing various dishes.

The cooking assistant robot is a groundbreaking advancement in household robotics, expanding the possibilities of robotic assistance in daily life. However, the complexities of cooking tasks, combined with the need for nuanced understanding and adaptability, present formidable challenges. Overcoming these hurdles requires innovative approaches to robot learning, sensory perception, and action execution, paving the way for a future where robots not only assist in household chores but also take on the role of culinary experts in our kitchens.

1.5.7 . . . accomplish tasks together with humans

Developing autonomous robots that collaborate effectively with humans in household tasks represents one of the most intricate challenges in the field of robotics. These robots are not just expected to execute tasks but also to understand, adapt, and seamlessly integrate into human routines and preferences. The ability to establish a shared understanding and coordinate actions with human partners is crucial, especially in dynamic and unpredictable home environments.

For successful collaboration, a robot must be equipped with a deep understanding of several nuanced human-centric concepts:

- Prioritization of Tasks (Importance): The robot must discern the priority of tasks, such as understanding that removing boiling water from a stove is more critical than setting the table at that particular moment.
- Understanding Human Preferences (Cooperation): The robot should recognize and respect human preferences, like acknowledging if a human prefers to prepare the salad themselves while the robot sets the table.

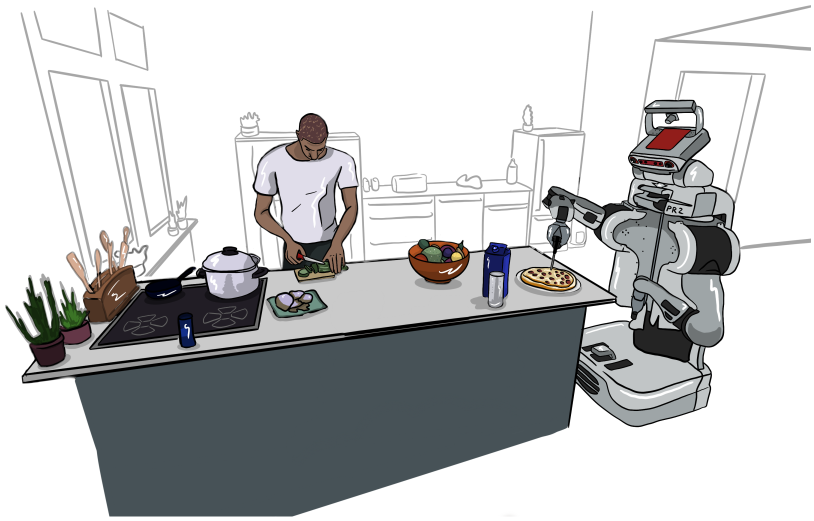


Figure 1.20: Robot agent preparing a meal together with a human. Here it is important to not only plan its own tasks but also coordinate with other agents like the human.

- Effective Communication: Ensuring clear and effective communication channels, so the robot can understand instructions from humans and, conversely, convey its intentions or needs clearly.
- Navigating Shared Spaces (Deference): The robot must be adept at sharing space with humans, avoiding obstructing pathways, and being able to pause or reroute its actions when in close proximity to humans.

In a collaborative setting, especially in tasks involving potential hazards like cooking, the robot's ability to ensure safety is paramount:

- Environmental Awareness: The robot should be constantly aware of its surroundings, able to detect the presence of humans and other obstacles to avoid collisions or unsafe interactions.
- Emergency Protocols: Implementing emergency stop mechanisms and other safety protocols to immediately halt operations if a potential risk is detected.
- Proactive Hazard Prevention: Understanding and anticipating potential dangers, such as the risk of spilling boiling water, and taking preemptive actions to prevent accidents.

Developing robots capable of such sophisticated human collaboration involves several key technical considerations:

- Advanced Sensory Systems: Equipping robots with sensors that can detect and interpret human presence, gestures, and spoken commands with high accuracy.
- Contextual Understanding and Adaptation: Enabling robots to understand the context of tasks and adapt their actions based on the dynamic preferences and behaviors of human partners.
- Real-Time Decision Making: Implementing algorithms that allow for real-time analysis and decision-making, ensuring that the

robot's actions are always aligned with the current situation and human partner's expectations.

Robots designed for human collaboration in household tasks embody the convergence of advanced robotics, artificial intelligence, and human-computer interaction disciplines. These robots hold the potential to not only assist in daily chores but also enrich human life by providing companionship, understanding, and adaptability in shared living environments. However, realizing this vision requires overcoming substantial challenges in robot design, sensory perception, context understanding, and safety assurance, paving the way for a future where humans and robots collaborate seamlessly.

1.6 Outline of the textbook

Foundations

2 The AICOR conceptual framework

In Chapter 1, we have acquired basic intuitions for understanding AI-powered and cognition-enabled robotics (AICOR), emphasizing the significant roles that autonomous robots can play in various industries and daily life. We explored the different perspectives on robots, the challenges involved in interpreting and executing task requests, and the essential cognitive capabilities required for these robots to operate effectively in dynamic environments. This introduction to AICOR set the stage for a deeper examination of the principles and mechanisms that underpin the development and operation of autonomous robotic systems.

Transitioning from this introductory view AICOR robots, Chapter 2 develops a conceptual framework that supports AICOR, providing a structured approach to understanding the interactions between users, robots, and their environments. This framework is essential for designing and implementing robots capable of performing complex tasks with high adaptability and efficiency. As Nilsson aptly stated,

"As scientists and engineers, we should continue to attempt to simplify, to organize, and to make elegant models—otherwise there are serious doubts that we would ever be able to understand enough about intelligence to design intelligent machines or to teach these design methods to students. If bridges had to be kludges, we wouldn't have a man-made bridge across the Golden Gate because complex bridge-building couldn't be understood, taught, or remembered. Successful engineering requires the frictionless case and a succession of gradually more complex models."

The creation and use of simplified, organized, and elegant models are core scientific and engineering methods to advance our understanding and capabilities in AICOR robotics.

In this chapter, we will introduce and elaborate on the User-Robot-Environment (URE) system, a comprehensive framework that encapsulates the core components and interactions essential for AICOR. We will examine the dimensions of the robot control problem, the necessary cognitive and physical capabilities of robot agents, and the iterative processes that enable robots to adapt and improve over time. By formalizing these concepts, we aim to provide a clear and systematic approach to developing autonomous robots that can navigate and manipulate their environments effectively, ultimately enhancing their functionality and impact across various domains.

To illustrate the elegance and necessity of such frameworks, consider the Golden Gate Bridge, a marvel of engineering that stands as a testament to the power of simplified and well-organized models. Just as the bridge was built through a succession of refined models, our approach to developing AICOR systems must be grounded in clear, elegant, and progressive frameworks that enable us to understand and harness the complexities of intelligent robotic behavior.

This chapter aims to:

1. **Introduce the Core Components of AICOR:** Identify and describe the six fundamental elements—User, Robot Agent, Environment, Task Request, Body Motion, and Activity Assessment—that form the basis of the AICOR framework.

2. **Explain the Interactions Within the AICOR System:** Explore how these components interact dynamically, creating a system characterized by continuous feedback and adaptation.
3. **Analyze the Dimensions of Robot Control:** Examine the factors influencing robot control, including physical and computational makeup, task complexity, and environmental context.
4. **Address the Challenges in Robotics:** Discuss key challenges such as the body motion problem, long-term autonomy, and the development of cognizant robot agents that understand and justify their actions.
5. **Detail the Control Program-centric Perspective:** Highlight the importance of the control program in orchestrating the robot's interactions with the environment to ensure accurate and efficient task completion.
6. **Model Ongoing Activities:** Describe how robots manage and adapt their actions in real-time to handle dynamic and evolving situations.
7. **Formalize the Conceptual Framework:** Introduce the rational robot agent model, formalizing the interaction between robots and their environments through perception-action loops and utility functions.

By the end of this chapter, readers will have a thorough understanding of the AICOR framework and its applications. This knowledge is essential for developing sophisticated robotic systems that can operate autonomously and effectively in dynamic environments, ultimately enhancing various aspects of human life and industry.

2.1 The User-Robot-Environment (URE) system

AI-powered and Cognition-enabled Robotics (AICOR) relies on understanding the interactions between the user, the robot, and their environment. This section explains the main components of this system, known as the user-robot-environment (URE) system. The URE system enables us to effectively study how users issue task requests, how robots interpret and execute these tasks, and how the environment influences and is modified by robotic actions. By dissecting these elements and their interactions, we provide a comprehensive framework for analyzing and designing autonomous robotic systems capable of performing complex tasks with high adaptability and efficiency. Understanding these interactions is crucial for developing sophisticated control programs that enable robots to navigate and manipulate their environments seamlessly, ultimately enhancing their functionality and effectiveness.

From the AICOR perspective, the user, robot agent, and environment form an integrated system, as depicted in Figure 2.1. The operation of this system is characterized by complex interactions and feedback loops. The user acts as the system operator, issuing commands and providing feedback based on task performance. The robot agent is the system's core, equipped with sensors to gather data, actuators to perform tasks, and a control program to interpret commands and execute actions. The environment includes all objects, obstacles, and conditions that can affect or be affected by the robot's actions.

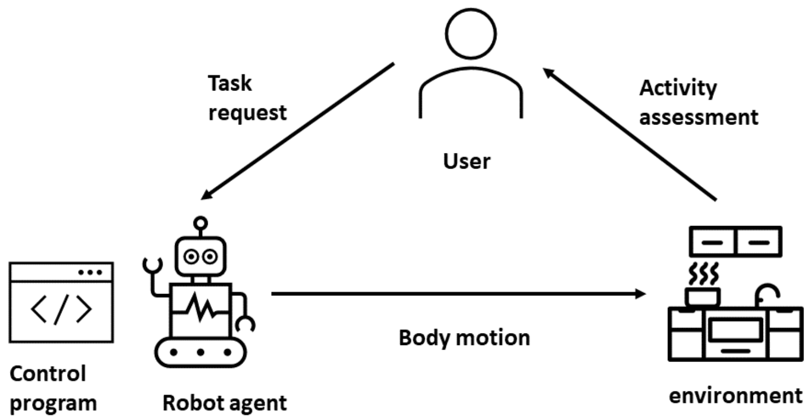


Figure 2.1: The robot agent system conceptual framework.

Our aim is to design the robot agent such that the user-robot-environment system works as it is intended to work in terms of system dynamics and feedback mechanisms, ensuring that the robot agent's actions are efficient, accurate, and reliable within a dynamic and sometimes unpredictable environment, ultimately satisfying the user's requests.

2.1.1 Entities and Interactions in the URE System

We begin building our conceptual framework by introducing the key concepts, which are the entities of the system and their interactions.

The **user** is the individual who interacts with the robot, issuing task requests that the robot is expected to fulfill. The user has the role of initiating the interaction by providing these requests. Task requests can vary in complexity, from simple directives like "bring me something to drink" to more complex instructions such as "clean up the room." The user's satisfaction with the robot's performance influences subsequent interactions and shapes the overall assessment of the robot's effectiveness.

user

The **robot agent** is an autonomous system equipped with sensors, actuators, and a control program. These components enable the robot to interpret and execute task requests. The sensors collect data about the environment and the robot's own state, while the actuators perform physical actions to interact with the environment. As the intermediary between the user and the environment, the robot agent's performance directly impacts the user's activity assessment.

robot agent

The **control program** processes the task request, plans the necessary actions, and generates the corresponding body motions.

control program

The **environment** is the physical space in which the robot operates. It includes all objects and conditions that the robot may interact with or be affected by. The environment provides the context for the robot's actions, encompassing everything from furniture and tools to other objects the robot may need to navigate around or manipulate. The state of the environment after the robot's actions is a crucial factor in the user's activity assessment.

environment

A **task request** is a directive issued by the user to the robot agent. These requests are often broad or vaguely specified, such as "bring me

task request

something to drink" or "clean up." The task request initiates the sequence of actions within the system. The clarity and specificity of the task request can influence how easily the robot agent can interpret and execute the required actions. When task requests are vague, the robot agent must utilize advanced reasoning and contextual understanding to determine the appropriate body motions.

Body Motion

Body motion refers to the physical movements of the robot, generated by the control program. These movements are necessary to accomplish the task requested by the user. Body motion is the robot agent's response to the task request and encompasses all the physical actions the robot performs to interact with the environment. Effective body motion requires **effectively-successful** control and coordination to ensure that the movements are both effective in achieving the desired physical effects and successful in fulfilling the task objectives.

activity assessment

Activity assessment is the evaluation conducted by the user to determine how well the robot has fulfilled the task request. This evaluation is based on criteria such as task effectiveness, task accuracy, efficiency, and adherence to user preferences. Activity assessment closes the loop in the interaction cycle, providing feedback on the robot's performance. Positive assessments reinforce trust and satisfaction, while negative assessments may lead to adjustments in future task requests or modifications to the robot's control program.

2.1.2 How the URE System Works

Building on the understanding of the core entities and their interactions, we now explore how these components operate in a continuous cycle within the user-robot-environment system.

The process begins with the user, who interacts with the robot agent by issuing a task request. A task request is a directive provided by the user, often broad or vaguely specified, such as "bring me something to drink" or "clean up." This request sets the system in motion, involving the interconnected components of the user, the robot agent, and the environment—the physical space in which the robot operates, including all objects and conditions that may influence or be influenced by the robot's actions.

Upon receiving the task request, the robot agent processes the request. Equipped with sensors, actuators, and a control program, the robot interprets the task request by leveraging its internal models and knowledge bases to infer the specific actions required. For instance, if the task request is "bring me something to drink," the robot must identify potential drink options, locate them within the environment, and plan a series of actions to retrieve and deliver the drink to the user. This involves understanding both the desired outcome and the steps necessary to achieve it.

The control program then translates the planned actions into specific body motions—the physical movements needed to accomplish the task. These body motions produce physical forces that interact with objects in the environment, such as opening a refrigerator door or grasping a bottle. The interaction between body motions and the environment results in physical state changes, reflecting modifications in the environment caused by the robot's actions. For example, the act of picking up a drink alters the position of the drink from its original location to being in the robot's grasp.

Throughout task execution, the robot agent dynamically adapts its actions based on real-time feedback from its sensors to handle variations or obstacles in the environment. Once the task is completed, the user conducts an activity assessment to evaluate how well the robot has fulfilled the task request. This evaluation considers factors such as task effectiveness, accuracy, efficiency, and adherence to specific preferences. The feedback from this assessment is crucial as it is fed back into the robot's control system to refine its internal models and improve future performance.

This iterative process of receiving task requests, executing them, and incorporating user feedback ensures that the robot becomes more adept at handling a variety of everyday tasks, enhancing its utility and reliability within the system.

Summary In this section, we explored how the user-robot-environment system operates through a cyclical interaction. The user initiates the process with a task request, the robot agent interprets and executes the task, and the user assesses the performance, providing feedback that refines the system for future interactions. This iterative process is essential for developing autonomous robotic systems capable of performing complex tasks with high adaptability and efficiency.

2.1.3 Detailed Example: "Bring Me Some Milk"

Having established the foundational components and interactions within the user-robot-environment system, we now illustrate how these elements come together in a practical scenario. In this detailed example, we explore the step-by-step process involved when a robot is tasked with "bringing milk from the refrigerator."

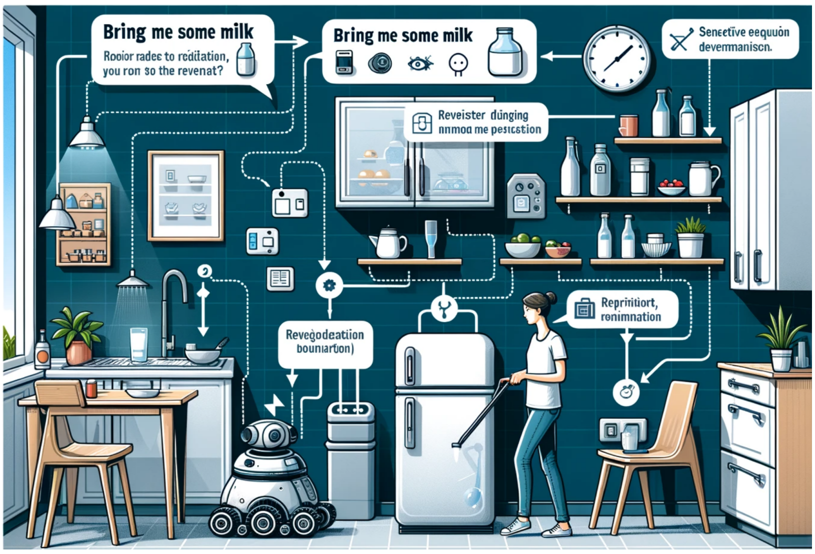


Figure 2.2: Fetching the milk in the Top-level conceptual framework.

In this scenario, the core components of the robot agent system are mapped into a specific task context. The process begins with the user, who interacts with the robot by requesting, "bring me some milk." This task request sets the system into motion, prompting the robot agent—an autonomous system equipped with sensors, actuators, and a control program—to interpret the task, devise a plan of action, and execute the necessary steps to fulfill the request.

Figure 2.3: Episode delivering milk.

The **environment** encompasses the physical space where the robot operates, including the kitchen, the refrigerator, and any obstacles that might be present between the robot's starting position and the milk's location. This environment provides the context in which the robot must navigate and perform its actions.

The **body motion** refers to the physical movements the robot must execute to complete the task. These movements include:

1. Navigating to the refrigerator
2. Opening the refrigerator door
3. Identifying and picking up the milk
4. Closing the refrigerator door
5. Returning to the user to deliver the milk

Each of these movements requires effectively-successful coordination and control.

The **activity assessment** involves the user evaluating how well the robot fulfilled the task request, considering factors such as the effectiveness and accuracy of the robot's actions, the efficiency with which it performed the task, and adherence to any specific preferences or instructions provided. The feedback from this assessment helps refine the robot's future performance, ensuring continuous improvement in its ability to execute similar tasks autonomously and effectively.

1. **User Issues Task Request:** "Bring me some milk."
2. **Robot Processes Request:**
   - Uses sensors to understand its current position and the environment layout.
   - Control program creates a plan: navigate to the kitchen, identify the refrigerator, open the refrigerator door, locate the milk, grasp the milk container, close the refrigerator door, and navigate back to the user.
3. **Execution of Body Motions:**
   - Move forward 5 meters, turn 90 degrees left, move forward another 3 meters.
   - Extend arm to grasp the refrigerator handle, pull the door open, identify and retrieve the milk, push the door closed.
   - Navigate back to the user and release the milk.
4. **User Conducts Activity Assessment:**
   - Evaluate the effectiveness and accuracy of the robot's actions.
   - Determine whether the robot brought the correct item.
   - Assess the efficiency of task completion, including smoothness and speed.
   - Consider adherence to preferences, such as avoiding unnecessary movements and handling the milk properly.

Figure 2.4: Process Breakdown.

Summary: This example illustrates how the robot agent system components interact to fulfill the task request "bring me some milk." It showcases the importance of each component and their dynamic interactions, from task perception and planning to execution and assessment.

2.2 Dimensions of the Robot Control Problem

The design of a robot's control program is not a standalone task but is – as illustrated in Figure 2.6 – profoundly influenced and sometimes even determined by three core aspects:

1. the robot's physical and computational makeup of the **robot**,
2. the spectrum of **tasks** it is expected to undertake, and
3. the **environment**al context it is set to operate in.

These aspects collectively impose specific requirements on the robot's decisional, reasoning, and control faculties that a control program should satisfy to ensure that the robot can autonomously achieve its tasks over extended periods of operation in a robust, flexible, natural, and effective manner.

The first aspect, namely the robot's physical and computational makeup, is subject to a multifaceted array of factors, pivotal among which are the following:

- *Motion Repertoire:* This factor captures the diversity of the robot's physical movements and the consequent forces it can exert upon

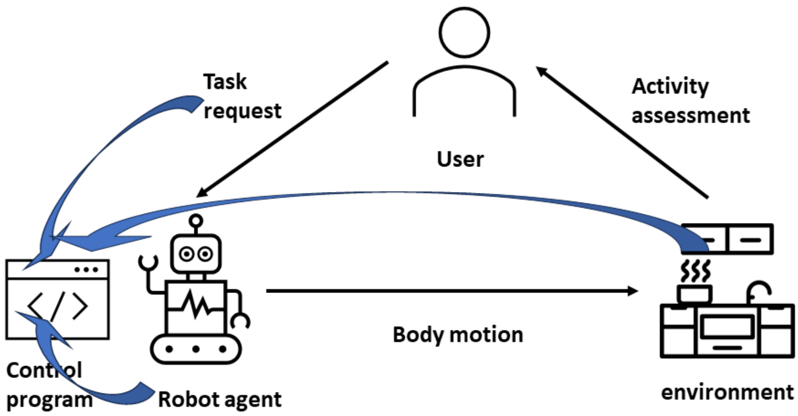


Figure 2.5: Aspects of the robot control problem that a control system should address through its design.

objects. The breadth of the motion repertoire available to a robot fundamentally shapes its interaction capabilities and operational versatility and extends the range of motions it can select from.

- *Tool Utilization:* The capacity for tool use significantly amplifies a robot's functional repertoire. Tool use, however, introduces substantial complexity into the control program, necessitating advanced cognition-enabled reasoning about altered kinematic structure and physical dynamics and potential action expansion resulting from tool integration. For example, when using a hammer the robot has to reason about a new kinematic structure where the kinematic chain for the robot's gripper is extended with the hammer and the hammer instead of the gripper becomes the end effector of the chain. In this case also the dynamics of controlling the end effector changes and new actions, namely hammering a nail into a piece of wood become feasible.
- *Sensor and Effector Reliability:* The precision and dependability of a robot's sensory and effector systems are are another key factor. Unreliability ind inaccuracy in sensing and action cause uncertainty and operational failures, compelling the need for sophisticated mechanisms for probabilistic state estimation, error detection, diagnosis, and recovery.
- *Adaptive Improvisation:* The ability to improvize, namely to use the robot body at execution time in novel ways is another factor to be considered in the design of the control program. For example, a robot might discover that it can close a door by pushing with its elbow if both grippers are in use.

Each of these factors introduces distinct challenges that cumulatively dictate the sophistication and resilience of a robot's control system. These challenges underscore the necessity for including cognition-enabled reasoning capabilities into robot control systems, ensuring adaptability and robustness in diverse operational contexts.

The second critical aspect influencing the design and functionality of robot control systems is the nature and complexity of tasks the robot is expected to perform. This dimension encompasses several intricate factors:

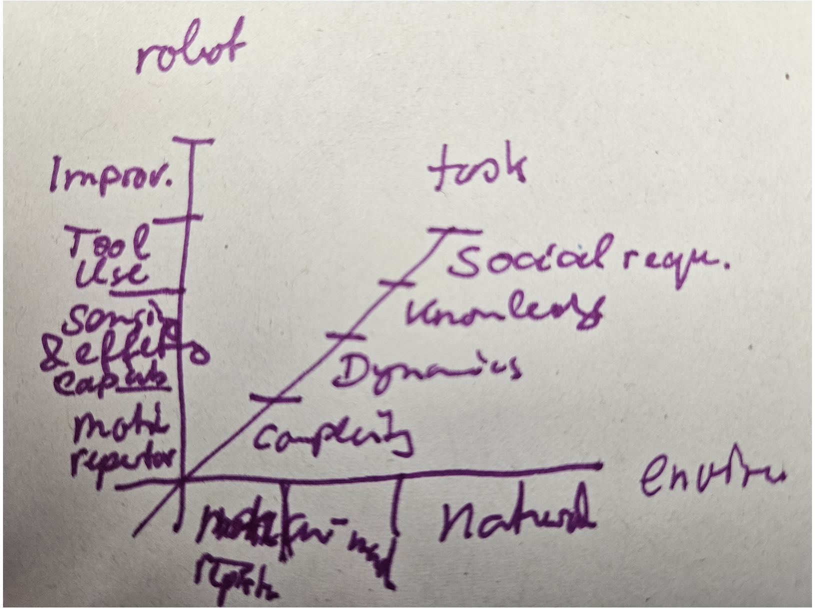


Figure 2.6: Dimensions of the robot control problem.

- *Complexity and Conjunctive Tasks:* Tasks range from simple, singular actions to complex, multifaceted ones. Robots may be required to execute multiple tasks conjunctively, necessitating advanced reasoning to manage potential interferences and synergies between simultaneous objectives.
- *Dynamic Tasking:* Robots might be requested to perform additional tasks or active tasks might be revised or cancelled during an ongoing activity, imposing a requirement for robust task management capabilities within the robot's control system. This involves real-time monitoring, adaptation, and prioritization among changing objectives.
- *Knowledge-Intensive Action:* Certain tasks demand a deep understanding of complex, domain-specific knowledge. For instance, a robot engaged in a chemical laboratory must be capable of reasoning about potential chemical reactions, understanding the properties of substances, and predicting outcomes of their interactions.
- *Resolution of Ambiguities:* Tasks may be underdetermined or ambiguous, presenting challenges that require the robot to disambiguate and refine tasks during execution. The robot must be capable of navigating uncertainties, making informed assumptions, and resolving ambiguities through logical, heuristic, or probabilistic reasoning.
- *Social Interaction and Joint Action:* Tasks involving social interaction or joint tasks with humans add another layer of complexity. These tasks require the robot to reason about different interpretations of underdetermined tasks and how to infer and negotiate a shared task and action interpretation.

Addressing these multifaceted task characteristics demands a control system that is not only technically proficient but also capable of exhibiting a degree of cognitive flexibility, situational awareness, and adaptive planning and learning.

The third pivotal aspect influencing the complexity of robotic control tasks pertains to the characteristics of the environment the robot is operating in. This dimension can be broadly categorized into man-made and natural environments, each presenting unique challenges and requirements:

- *Man-Made Environments:*
  - Functional Structures: These environments are typically designed with functionality in mind, containing elements that facilitate task execution. For instance, door handles are ergonomically designed for easy grasping and operation.
  - Interaction with Complex Appliances: Robots operating in such environments may need to interact with sophisticated devices, each having specific operational protocols. An example is a multi-purpose kitchen oven, which requires understanding of various modes, settings, and the physical processes involved.
- *Natural Environments:*
  - Lack of Structure: In contrast to man-made settings, natural environments are often less predictable and lack standardized functional structures. This unpredictability demands higher levels of adaptability and problem-solving capabilities from the robot.
  - Dynamic and Potentially Adversarial Conditions: Natural settings can change rapidly and may present adversarial conditions. For instance, weather conditions can alter the terrain, or unexpected obstacles may emerge, requiring the robot to continuously adapt its strategies and make real-time decisions.

In both cases, the environment profoundly impacts the control tasks, necessitating a control system that is not only sensitive to the surrounding context but also capable of dynamic adaptation and sophisticated decision-making processes. The complexity of the environment necessitates a multifaceted approach to the design of robot control systems, ensuring they are robust, versatile, and capable of operating efficiently in a wide range of scenarios.

Discussion

- The number of specific robot applications tailored for different combinations of robots, tasks, and environments exceeds what robot programmers can do.
- Need for generalized solutions
- Need for transferability
- Gil Pratt: "Is a Cambrian Explosion coming to robotics"

2.3 Capabilities of Robot Agents

In the previous section, we explored the dimensions of the robot control problem, identifying key challenges such as body motion planning, sensor integration, real-time decision-making, task adaptation, error detection and recovery, and long-term autonomy. Addressing these challenges is crucial for the effective functioning of AI-powered and cognition-enabled robots. In this section, we will discuss the specific capabilities that robot agents must develop to overcome these control problems. By enhancing their body motion, autonomy, lifelong learning, and cognitive abilities, robots can better navigate and manipulate their environments, thereby fulfilling their intended tasks with higher efficiency and reliability. These capabilities are not just theoretical advancements but practical necessities for tackling the complexities outlined in the previous section.

The effectiveness of the AI-powered and Cognition-enabled Robotics (AICOR) system hinges on the capabilities of the robot agents within it. These robot agents are the programmed entities that drive the system's functionality, and they must exhibit several key capabilities to ensure the system operates seamlessly.

First, they must solve the robot body motion problem: given a task request, they need to determine the effectively-successful body movements required to accomplish it. Second, they need to function as agents, which means they must act autonomously to accomplish tasks and goals. This requires autonomous decision-making that is flexible, robust, and efficient. Third, they require long-term autonomy. This means their activities should extend beyond just repetitively fulfilling task requests to include long-term task management, resource preparation for future actions, and continuous learning to improve their competence. Lastly, robot agents should possess a deep understanding of their actions: what they are doing, how and why they are doing it, what the outcomes will be, and how they can modify their actions to achieve or avoid specific effects.

In this section, we will explore these essential dimensions of robot capability in detail.

2.3.1 Body Motion Problem

Building on the key capabilities required for robot agents, we now delve into one of the most fundamental challenges they face: the body motion problem. To effectively fulfill task requests and score well in the user's activity assessment, the robot agent must solve this central challenge, which is crucial for interpreting and executing tasks competently and efficiently.

Body Motion Problem

Given: An underdetermined task request.

Infer: How to move the robot's body to achieve the desired effects while avoiding unwanted side effects.

This problem requires the robot to determine the effectively successful physical movements needed to fulfill a user's directive, even when the directive is vague or lacks specific details. The complexity arises from the need to interpret the task request, plan a series of actions, and execute them with precision in a dynamic environment.

Interpreting the Task

Explanation of the Body Motion Problem Interpreting the task is the first crucial step in solving the body motion problem. This involves understanding the user's intent and breaking down the high-level task request into specific, manageable sub-tasks. For instance, if the user requests the robot to "clean up the room," the robot must infer which objects need to be moved, where they should be placed, and how to handle any obstacles present in the environment. This requires the robot's control program to use its internal models and knowledge bases to deduce the details that the task request omits. By accurately interpreting the task, the robot can ensure that it understands the desired outcome and the necessary steps to achieve it, setting a solid foundation for effective task execution.

Planning the Actions

Once the task is interpreted, the next step is planning the actions. This involves developing a detailed plan that specifies the sequence of movements needed to complete the task. The robot's control program must consider various factors, such as the robot's capabilities, the layout of the environment, and the locations of relevant objects. For example, if the task is to "bring me some milk," the robot must plan a route from its current location to the kitchen, identify the refrigerator, and devise the steps required to retrieve and deliver the milk. The planning phase is critical as it transforms the interpreted task into a concrete series of actions that the robot can execute, ensuring that the task is approached methodically and efficiently.

Executing the Plan

Executing the plan involves translating the planned actions into effectively-successful body motions. This includes navigating the environment, manipulating objects, and interacting with various elements to accomplish the task. The robot must coordinate its sensors, actuators, and control program to work seamlessly together, ensuring that each movement is performed accurately. Additionally, the robot needs to be adaptable, adjusting its actions in real-time based on feedback from its sensors. For example, if an object is not in its expected location, the robot must modify its plan to locate and retrieve it. Effective execution requires precision, coordination, and flexibility, allowing the robot to complete tasks efficiently while avoiding any unwanted side effects.

The ultimate goal of solving the body motion problem is to achieve the desired effects of the task while avoiding any unwanted side effects. This involves:

- **Task effectiveness and precision:** Effective-successfully executing actions to achieve the specific outcomes desired by the user.
- **Efficiency:** Performing tasks in a timely manner without unnecessary or redundant movements.
- **Safety and Reliability:** Ensuring that the robot's actions do not cause harm or damage to itself, the environment, or any objects and people within it.
- **Trustworthiness:** Ensuring that the robot operates reliably and predictably, building user confidence in the robot's capabilities.
- **Transparency and Explainability:** Providing clear and understandable information about the robot's actions and decision-making processes, allowing users to comprehend how and why the robot performs certain actions.

By effectively solving the body motion problem, the robot agent can fulfill task requests accurately and efficiently, thereby scoring well in the user's activity assessment. This process is critical for the robot's ability to perform a wide range of tasks autonomously and to adapt to new and varying situations. Understanding and addressing the body motion problem is a key component in the development and implementation of AI-powered and cognition-enabled robotic systems.

2.3.2 Robot Agents and Autonomy

In AICOR, the goal is to develop robots that can fulfill vaguely stated task requests with a high level of autonomy. Achieving this requires robots to operate as if they possess beliefs, goals, and intentions, enabling them to make autonomous and informed decisions. This approach is essential because the tasks assigned to these robots are typically human-scale, abstract, and often ambiguous. Furthermore, the robots operate in environments where their knowledge may be uncertain, incomplete, or inaccurate.

Robot Agent

A robot agent in AI-powered and Cognition-enabled Robotics (AICOR) is an autonomous robot designed to interpret and execute task requests with a high degree of flexibility, robustness, and informed decision-making. These robot agents are engineered to act as if they possess beliefs, goals, and intentions, enabling them to autonomously navigate and interact with their environment to achieve specified objectives.

Autonomous robot agents must exhibit several critical characteristics to navigate these challenges successfully. Flexibility is paramount, as these robots need to interpret and adapt to broadly defined tasks, managing variations and uncertainties inherent in human environments. For instance, when given a task like "clean up the room," the robot must decide what cleaning entails, which objects need to be moved, and where they should be placed.

Robot Autonomy

Robot autonomy in AI-powered and Cognition-enabled Robotics (AICOR) refers to the capability of a robot agent to operate independently and effectively without human intervention. This includes interpreting and executing tasks, making informed decisions, adapting to dynamic and uncertain environments, and learning from experiences to improve performance over time.

Autonomous robots in AICOR exhibit flexibility, robustness, and transparency, enabling them to handle human-scale, abstract tasks that are often ambiguous and complex. They achieve this by leveraging advanced cognitive abilities to understand their goals, plan actions, detect and recover from failures, and provide clear explanations for their behavior, ensuring reliable and efficient task completion.

Robustness is another essential attribute, enabling robots to perform reliably in dynamic and unpredictable environments. This robustness involves detecting execution failures, diagnosing their causes, and recovering to continue the task successfully. Such resilience ensures that the robot can handle unexpected obstacles or changes in the environment without requiring human intervention.

The ability to make autonomous decisions is crucial for these robots to achieve their goals. They must evaluate their current state, predict the outcomes of potential actions, and choose the most appropriate course of action. This decision-making process allows the robot to act effectively even when direct instructions are incomplete or ambiguous. Alongside autonomy, robots need to make informed decisions based on data available from their sensors and prior experiences. This involves continuously updating their knowledge base and adapting their strategies to improve task performance over time.

Operating in complex environments filled with ambiguities, autonomous robot agents must interpret high-level, abstract task requests and translate them into concrete actions. This requires a deep understanding of the task's goals and the ability to infer missing details. Given that robots often have incomplete or inaccurate information about their environment, they must be capable of making decisions with uncertain knowledge. They use probabilistic reasoning to assess the likelihood of different outcomes and select actions that maximize the chances of success.

When execution failures occur, robots must detect these failures and diagnose their causes. Understanding why a task did not proceed as expected allows them to identify potential solutions. After diagnosing a failure, robots need to implement recovery strategies, which might involve adjusting their actions, re-planning their approach, or seeking additional information to resolve the issue. Autonomy also involves learning from past experiences to improve future performance. Robots must analyze the outcomes of their actions, learn from mistakes, and refine their strategies to become more effective over time.

In summary, autonomy in AICOR robot agents is characterized by flexibility, robustness, and the ability to make autonomous and informed decisions. These capabilities enable robots to handle human-scale tasks that are abstract and ambiguous, navigate uncertain environments, and continuously improve their performance. The development of such autonomous agents is critical for achieving the goals of AI-powered and cognition-enabled robotics.

2.3.3 Life-long Autonomy

In the previous sections, we explored the essential capabilities required for AICOR robot agents to interpret and execute tasks autonomously, ensuring flexibility, robustness, and efficiency. While these capabilities are fundamental, it is crucial to recognize that AICOR robot agents exist continuously in the world. Their activity cannot be considered merely as a sequence of episodes aimed solely at achieving specific task requestsindependently. Instead, these agents must accomplish tasks while simultaneously improving their ability to perform future tasks.

Life-long autonomy necessitates that robot agents prepare for the future, often at the expense of current resources, with the expectation that these investments will yield additional benefits over time. For example, a robot agent can enhance its environment to facilitate future tasks, perform actions opportunistically rather than strictly upon request, and continually learn to improve its competence and skills for upcoming challenges.

This section delves into the concept of life-long autonomy, outlining the necessary strategies and capabilities for robots to operate autonomously over extended periods, maintaining performance, reliability, and adaptability. We will discuss the importance of sustainable behaviors, continuous improvement, habitual routines, strategic planning, lifelong learning, and knowledge sharing, all of which contribute to the development of truly autonomous systems that can thrive in dynamic environments.

Life-long autonomy in AICOR robot agents involves adopting a holistic approach to decision-making that considers the long-term cumulative impact of their actions. This ensures sustainable and adaptive behaviors that contribute positively to future objectives. Key aspects of life-long autonomy include sustainable behaviors, learning from failures, developing stereotypical behaviors, strategic planning through lifepath analysis, lifelong learning and apprenticeship, and abstract understanding with knowledge sharing.

Sustainable Behaviors

To achieve life-long autonomy, robots must adopt a decision-making approach that considers the long-term consequences of their actions. This involves ensuring sustainable and adaptive behaviors that enhance the robot's performance and reliability. Keeping a consistent and predictable operational space, known as environment stabilization, is essential for maintaining high performance and reducing uncertainties.

Learning from Failures

Continuous improvement is a cornerstone of life-long autonomy. Robots need to analyze past mistakes and refine their future actions to ensure progressive improvement and resilience. This iterative learning process allows robots to adapt and enhance their capabilities over time.

Stereotypical Behavior and Entropy Reduction

Developing habitual behaviors, or stereotypical behaviors, streamlines robot operations and reduces unpredictability. These routines help robots handle repetitive tasks proficiently, optimizing resource use and freeing up resources for more complex problem-solving activities.

Lifepath Analysis

Strategic planning through lifepath analysis is crucial for aligning robot actions with long-term goals and sustainability. Robots must plan their actions considering future states and potential challenges, ensuring that current actions contribute positively to future objectives.

Lifelong Learning and Apprenticeship

Robots act as lay scientists, continuously exploring their environments to gain competence and adaptability. Acquiring habitual skills and commonsense knowledge enables robots to interpret vague task requests and adapt to new situations. Lifelong learning and apprenticeship ensure that robots remain competent and versatile.

Abstract Understanding and Knowledge Sharing

For effective life-long autonomy, robots must develop a higher-level understanding that can be shared with other robots and applied across various contexts. This generalization enhances collective intelligence and versatile problem-solving capabilities. Adapting to novel challenges is crucial for truly autonomous systems, allowing robots to generalize learning to new and unforeseen situations.

Implementation Strategies Implementing life-long autonomy in robots requires a multifaceted approach that ensures consistent performance and adaptability. Key strategies include environment stabilization, continuous learning from failures, developing habitual routines, strategic planning through lifepath analysis, fostering lifelong learning, and enhancing knowledge sharing. These strategies collectively enable robots to sustain high performance and adaptability over extended periods. Examples of such strategies are:

- **Environment Stabilization:** Consistently maintaining an operational environment to support reliable performance.
- **Analyzing and Learning from Failures:** Continuously refining strategies based on past experiences.
- **Developing Habitual Routines:** Establishing standard behaviors to optimize task execution.
- **Lifepath Analysis:** Planning actions with a long-term perspective to ensure sustainability and goal alignment.
- **Fostering Lifelong Learning:** Encouraging robots to act as learners, continuously improving their knowledge and skills.
- **Enhancing Knowledge Sharing:** Developing mechanisms for robots to share abstract knowledge, enhancing their collective capabilities.

By integrating these dimensions, robot agents can achieve a high level of long-term autonomy, ensuring performance, reliability, and adaptability over extended periods. This comprehensive approach allows robots to not only fulfill immediate task requests but also prepare for and excel in future challenges, continuously enhancing their utility and effectiveness.

2.3.4 Cognizant Robot Agents

Extending the collection of robot agent capabilities, we now turn our focus to cognizant robot agents.

Cognizant Robot Agent

Cognizant robot agents are characterized by their ability to understand what they are doing, why they are doing it, and how their actions will impact their environment. Cognizant robot agents can reason about the consequences of their intended actions, infer necessary adaptations in an effect-guided and effect-aware manner, and translate this understanding into successful actions.

Cognizant robot agents represent a significant advancement in robotics by integrating understanding, reasoning, and effect-aware decision-making into their operational framework. These agents are designed to interpret the context of their tasks, reason about the potential consequences of their actions, and adapt their behavior dynamically to achieve the desired outcomes effectively. By applying advanced AI principles and cognitive system concepts, as outlined by Ronald Brachman, cognizant robot agents can execute tasks with a thorough understanding of their objectives, methods, and the impact of their actions, ensuring more reliable and intelligent performance in complex and changing environments. This integration significantly elevates their capability to operate autonomously and intelligently, setting a new standard for robotic performance.

Learning and Improvement

Robot agents should continuously learn from their experiences and refine their actions based on feedback. For instance, if a robot repeatedly fails to grasp an object, it should analyze its actions, understand the failure, and adapt its approach for future attempts. This process of learning from failures and successes enhances the robot's competence and adaptability, allowing it to perform tasks more efficiently over time.

Reasoning and Decision-Making

Cognizant robot agents must possess robust reasoning capabilities to interpret task requests, plan actions, and make informed decisions. This involves understanding the context of tasks, predicting outcomes, and selecting the best course of action to achieve desired goals. For example, when faced with multiple potential actions, the robot should be able to evaluate each option's potential impact and choose the one most likely to succeed.

Explanation and Transparency

Cognitive robot agents should be able to explain their actions and reasoning processes. This transparency builds trust and allows users to understand why the robot took certain actions. For example, if a robot decides to clean a different area first, it should explain its reasoning, such as detecting a higher level of dirt in that area. Such explanations help users comprehend the robot's behavior and rationale, fostering a better human-robot interaction.

Commonsense Knowledge and Naive Physics

By incorporating commonsense knowledge and an understanding of naive physics, robot agents can better interpret vague task requests and interact with the physical world more effectively. For instance, knowing that liquids can spill helps the robot handle containers more carefully. This knowledge allows robots to perform tasks more safely and effectively, as they can anticipate and mitigate potential issues.

Adaptation to Novel Situations

Cognitive robots should be capable of adapting to new and unforeseen challenges. They should apply learned knowledge to novel tasks and environments, demonstrating flexibility and resilience. For instance, if a new type of object is introduced, the robot should use its understanding of similar objects to handle it appropriately. This ability to generalize and adapt is crucial for robots operating in dynamic and unpredictable environments.

Testing Cognition-Enabled Capabilities

To evaluate whether a robot agent is "cognition-enabled," we test its ability to answer questions that require cognitive capabilities and transform abstract information into successful actions. This approach aligns with principles from Bloom's taxonomy in pedagogics, which categorizes cognitive skills from basic recall to higher-order thinking. For example, the robot should accurately recall facts, understand and explain concepts, apply knowledge to perform tasks, analyze situations, synthesize information, and evaluate actions and outcomes. These testing categories ensure that robots are not only functional but also intelligent and adaptive in dynamic environments.

Example Scenarios: To illustrate the practical application of cognizant robot agents, consider the following example scenarios that demonstrate their cognitive capabilities in various situations.

- **Understanding Task Requests:** A user requests, "Bring me something to drink." The robot must identify possible drinks, locate them, and deliver one, explaining its choice if asked.
- **Problem-Solving:** Faced with a new obstacle, the robot should analyze the situation, develop a solution, and explain its approach.
- **Learning and Adapting:** The robot should learn from failures. If it spills a drink, it should adjust its grip technique and explain the improvement.

By incorporating these cognitive capabilities, our robot agents can operate more autonomously and intelligently, continuously improving their performance, reasoning about their actions, and effectively communicating with users. This approach enhances their ability to handle a wide range of tasks in dynamic and open environments, making them more reliable and versatile in real-world applications.

2.4 Robot Agents Acting in Physical Environments

In Section 2.1, we considered the user, the robot agent, and the environment as atomic entities, each playing a distinct role within the AICOR system. However, these entities are, in reality, complex systems composed of multiple interrelated components. This section delves deeper into these components, providing a more granular view of their interactions and dynamics. By exploring the intricate subsystems within the robot agent and the environment, which are illustrated in Figure 2.7, we aim to understand how these detailed interactions influence the overall functionality and performance of autonomous robotic systems. This comprehensive perspective is crucial for developing more robust, efficient, and adaptive robot agents capable of operating effectively in dynamic physical environments.

In this section, we investigate the subsystems of the robot agent, the environment, and the user in greater detail. Our aim is to understand how the robot agent generates body motions that exert physical forces on the environment, how it perceives and interprets environmental data, and how the environment itself changes in response to the robot's movements. This detailed exploration will reveal the dynamic interplay between these subsystems, providing insights into the mechanisms that enable autonomous robotic systems to interact effectively with their surroundings.

2.4.1 The Robot Agent Subsystem

The robot agent subsystem encompasses the hardware and software components responsible for executing effecively-successful body motions, as discussed in Section 2.3.1. This includes the reasoning methods that enable the robot to perform complex tasks accurately.

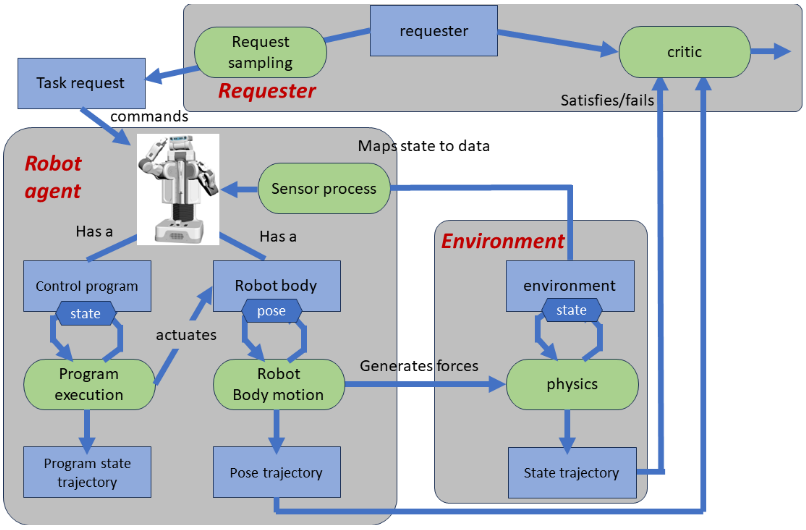


Figure 2.7: Hierarchical conceptual framework.

In this section, we investigate the subsystems of the robot agent, the environment, and the user in greater detail. Our aim is to understand how the robot agent generates body motions that exert physical forces on the environment, how it perceives and interprets environmental data, and how the environment itself changes in response to the robot's movements. This detailed exploration will reveal the dynamic interplay between these subsystems, providing insights into the mechanisms that enable autonomous robotic systems to interact effectively with their surroundings.

The robot agent, as a subsystem, consists of two main components: the robot control program and the robot body. Each component plays a vital role in enabling the robot to interact effectively with its environment. Understanding these components and their operations is crucial for developing autonomous systems capable of performing complex tasks in dynamic settings.

Components of the Robot Agent The **robot control program** acts as the brain of the robot, responsible for processing sensor data, planning actions, and generating commands for the actuators. It uses sophisticated algorithms and models to interpret task requests, make decisions, and adapt to changing conditions. The control program ensures that the robot's actions are coordinated, efficient, and goal-directed, making it a dynamic system that updates its computational state continuously.

robot control program

The **robot body** is the physical structure of the robot, equipped with essential components like sensors and actuators.

robot body

- **Sensors:** These are the sensory organs of the robot, collecting data from the environment, such as distances to objects, surface textures, and environmental conditions. Sensors like cameras, lidar, ultrasonic sensors, and touch sensors provide crucial information for the robot's perception of its surroundings.
- **Actuators:** These are the muscles of the robot, responsible for executing physical movements. Actuators include motors, servos, and hydraulic systems that control the robot's limbs, wheels, or other

movable parts. Through actuators, the robot can manipulate objects, navigate through space, and perform complex tasks, transforming the outputs of the control program into physical actions.

Operation of the Robot Agent Subsystem The operation of the robot agent subsystem involves a continuous loop of perception, computation, and action:

- **Perception:** The operation begins with perception, where the robot's sensors collect real-time data from the environment. This includes detecting distances, surface textures, and various environmental conditions, providing a comprehensive snapshot of the surroundings that is crucial for informed decision-making.
- **Computation:** Following perception, the sensor data is processed by the robot control program. This computational phase involves interpreting the sensory input, updating the control program's state, and planning the necessary actions. The control program uses algorithms and models to make decisions and adapt to changes, maintaining a sequence of program states that guide its operations.
- **Action:** Based on the updated computational state, the control program generates commands for the robot's actuators. These commands direct the motors, servos, and hydraulic systems to perform specific physical movements, ensuring that the robot's actions are aligned with the planned tasks.
- **Body Motion:** The execution of commands by the actuators results in body motions that alter the robot's pose. These movements generate physical forces that interact with the environment, causing changes in its state. The dynamic nature of the robot body allows it to perform a wide range of tasks, from navigating spaces to manipulating objects.
- **Feedback Loop:** The final stage in the operation of the robot agent subsystem is the feedback loop. The robot's sensors continuously gather data, providing feedback on the outcomes of its actions. This real-time feedback is essential for the control program to adjust computations and refine future actions, ensuring that the robot adapts to changing conditions and maintains goal-directed behavior.

By iterating through this cycle of perception, computation, and action, the robot agent can effectively perform tasks and interact dynamically with its environment. This continuous feedback loop ensures that the robot adapts to changing conditions and maintains goal-directed behavior. Understanding these components and their interactions provides a comprehensive view of how robot agents function autonomously in complex environments, continuously improving their performance through an iterative process of perception, computation, action, and feedback.

2.4.2 The Environment Subsystem

The environment in which the robot operates is a dynamic and multifaceted subsystem that encompasses all physical elements and conditions the robot interacts with. Understanding these components and their interactions is essential for enabling robots to navigate and perform tasks effectively within their surroundings.

Components of the Environment The environment consists of various objects, substances, and the robot body itself, each playing a specific role in the robot's activities.

- **Objects:** These include both man-made and natural objects. Man-made objects are often designed for specific purposes and can be composed of multiple parts. Natural objects are those found in the environment that are not designed by humans. With respect to the robot's activity, objects can be categorized into structural and static parts, which provide a stable structure and must be navigated around without altering them, obstacles that present barriers or challenges to avoid, and manipulable objects such as tools, utensils, or materials needed for a task. Additionally, devices are specific types of objects that perform physical processes, such as ovens for heating and dishwashers for cleaning, each with its own operational states and processes that the robot may need to interact with or monitor.
- **Substances:** These include various materials and elements the robot might encounter or need to handle, such as liquids, gases, or granular materials. The properties of these substances, such as viscosity, temperature, or reactivity, can influence how the robot interacts with them.
- **Robot Body:** The robot body itself is part of the environment. For instance, the gripper of the robot can be closed around an object to be picked up, and the amount of force exerted by the gripper will determine whether the object is lifted, slips, or breaks. The robot body's interactions with the environment must be effectively-successful controlled to ensure successful task execution without causing damage.

Evolution of the Environment The evolution of the environment involves understanding how it emits physical quantities that can be detected and measured by the robot's sensors and how the environment's state changes over time. Initially, the robot perceives the environment through its sensors, which gather data on positions, orientations, and the physical states of objects. This sensory input provides the robot with a real-time snapshot of its surroundings, crucial for making informed decisions.

As the robot moves and performs tasks, the state of the environment evolves dynamically. This evolution includes changes in the positions and orientations of objects, as well as their physical states, such as articulation poses and deformations. The robot must continuously update its understanding of the environment to adapt to these changes and ensure accurate task execution. The physical state of each object and the overall environment are governed by fundamental principles of physics, constrained by physical laws that dictate how objects interact and change over time.

Kinematics and dynamics play a significant role in the environment's evolution. Kinematics focuses on the motion of the robot's parts without considering the forces causing the motion, involving position, velocity, and acceleration. Dynamics, on the other hand, takes into account the forces and torques that cause motion, including the effects of mass, inertia, and Newton's laws of motion. Understanding these principles allows the robot to predict and respond to changes in the environment accurately.

In addition to kinematics and dynamics, contact mechanics and rigid body dynamics are essential for comprehending environmental interactions. Contact mechanics involves understanding friction, grip force, and surface properties to ensure successful manipulation of objects. Rigid body dynamics models robots and objects as inflexible bodies to simplify calculations and predict their motion under applied forces and torques. Compliant motion and environmental interaction further enhance the robot's ability to adapt to slight variations and forces in the environment, ensuring smooth and effectively-successful interactions. This holistic understanding of the environment's evolution enables the development of more robust, adaptive, and efficient robotic systems.

2.4.3 The User Subsystem

We take a very minimalistic view of the user subsystem, focusing on its two main components: the task request generator and the task assessment critic. Despite its simplicity, the user subsystem plays a crucial role in guiding and evaluating the robot agent's activities.

Components of the User Subsystem The user subsystem, though minimalistic, consists of two crucial components that drive the robot agent's actions and evaluate its performance: the task request generator and the task assessment critic.

- **Task Request Generator:** This component is responsible for generating task requests that the user issues to the robot agent. The task request generator formulates specific instructions or goals for the robot to achieve, initiating the robot's actions. These requests can vary in complexity, from simple commands like picking up an object to more complex tasks involving multiple steps and considerations.
- **Task Assessment Critic:** After the robot agent completes a task request, whether it succeeds, gives up, or finishes without fully satisfying the task requirements, the task assessment critic evaluates the robot's performance. This evaluation considers the body motion during the active task performance and the physical evolution of the environment. The critic assesses how well the robot agent has accomplished the given task request by examining the effectiveness, accuracy, efficiency, and overall effectiveness of the robot's actions.

Operation of the User Subsystem The user subsystem operates by continuously interacting with the robot agent through task requests and performance assessments:

- **Task Request Generation:** The task request generator issues a task request to the robot agent. This request serves as a directive for the robot, outlining the specific actions or goals it needs to achieve.

The clarity and specificity of these requests are crucial for guiding the robot's actions effectively.

- **Performance Assessment:** Upon completing the task, the task assessment critic evaluates the robot's performance. This assessment involves analyzing the robot's body motions during the task and the resulting changes in the environment. The critic examines factors such as the precision of movements, adherence to the task requirements, and the overall impact of the robot's actions on the environment.
- **Feedback Loop:** The evaluation results from the task assessment critic provide valuable feedback for the robot agent. This feedback helps refine future task executions, improve decision-making processes, and enhance the robot's overall performance. By continuously assessing and providing feedback, the user subsystem ensures that the robot agent can learn and adapt, improving its ability to perform tasks autonomously and effectively.

By understanding the minimalistic yet essential components of the user subsystem and its operation, we can appreciate how user interactions guide and refine the robot agent's activities. This subsystem plays a critical role in ensuring that the robot can respond to task requests accurately and improve its performance through continuous assessment and feedback.

2.4.4 Interaction Between Subsystems

The interaction between the robot agent and the environment is central to the robot's ability to perform tasks autonomously. This interaction can be characterized by several key processes:

- Perception: The robot uses its sensors to gather data about the environment. This includes detecting objects, identifying obstacles, and assessing environmental conditions. Perception is a continuous process, providing real-time feedback that informs the robot's actions.
- Action Planning: Based on the perceived data, the robot's control program plans a sequence of actions to accomplish the task request. This involves path planning, object manipulation strategies, and contingency planning for potential obstacles or changes in the environment.
- Execution: The control program generates commands for the actuators to execute the planned actions. This includes movements like navigating to a location, picking up an object, or avoiding an obstacle. The robot's actions modify the environment, creating a dynamic feedback loop.
- Adaptation: As the robot interacts with the environment, it continuously receives feedback through its sensors. If the environment changes or if the robot encounters unexpected obstacles, it must adapt its actions in real-time. This may involve re-planning its path, adjusting its grip on an object, or altering its strategy to achieve the desired outcome.

2.4.5 Example Scenario: Retrieving the Milk Box from the Refrigerator

Consider a robot tasked with retrieving a milk box from the refrigerator. This scenario involves various interactions between the robot agent and the environment:

- Perception: The robot uses its sensors to scan the kitchen, identifying the refrigerator and navigating towards it. Upon reaching the refrigerator, it uses more detailed sensors to detect the handle and understand the door mechanism.
- Action Planning: The control program plans the sequence of actions: opening the refrigerator door, locating the milk box inside, grasping the milk box, and closing the refrigerator door. It must consider factors such as the position of the milk box, the door's weight, and any potential obstacles inside the refrigerator.
- Execution: The robot's actuators execute the planned actions. It opens the refrigerator door using its arm, carefully maneuvers to avoid knocking over other items, and uses a gripper to pick up the milk box. The robot then retracts its arm and closes the refrigerator door.
- Adaptation: Throughout this process, the robot continuously adapts its actions based on real-time feedback. If the milk box is not in the expected position, the robot must adjust its strategy, perhaps searching different shelves or repositioning its grip.

By treating the robot agent and the environment as interconnected subsystems, we can better understand and optimize their interactions. This detailed perspective enables the development of more robust, efficient, and adaptive robot agents capable of performing complex tasks autonomously in dynamic physical environments.

2.5 The Control Program-centric Perspective of the Framework

In this section, we restructure the user-robot-environment system to create a more structured control problem for designing the robot control program as an engineered computational system. To achieve this, we decompose the overall system into the control system and the system that is controlled by the control system. This is illustrated in Figure 2.8. A significant modification in this perspective is that the robot body becomes part of the controlled system. By adopting a control program-centric perspective, we can better understand how the control program orchestrates the interactions between the robot body and the environment, ensuring that tasks are completed accurately and efficiently.

The overall system is decomposed into the control system and the controlled system. This decomposition is crucial for managing the interactions between the robot agent and the environment. By considering the robot body as part of the controlled system, we gain a clearer understanding of how to structure the control program to effectively manage these interactions.

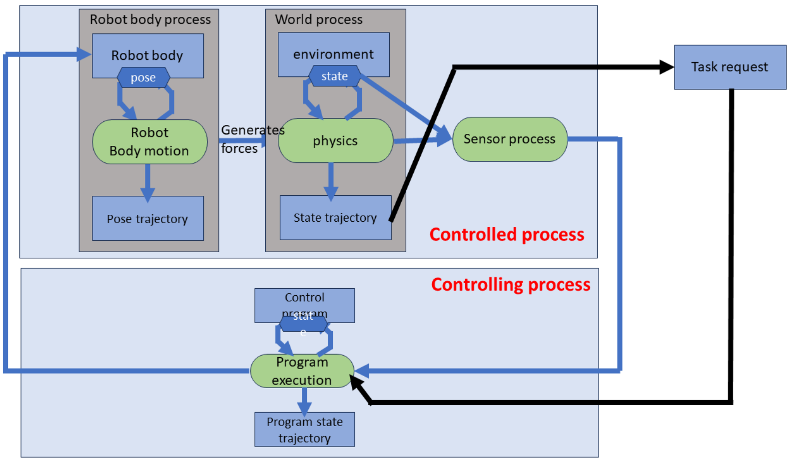


Figure 2.8: Control program-centric conceptual framework. detail control program additional components: perception, belief state, action planning and execution, and prospection.

Components of the Controlled System The controlled system consists of the robot body motion process, the world process, and the sensor process. Each component plays a critical role in the robot's ability to interact with its environment:

- **Robot Body Motion Process:** This process manages actuator signals that cause the robot body to move and change its pose. The motion generates physical forces that act on the environment, altering its state and enabling the robot to perform tasks.
- **World Process*:** The world process encompasses the entire environment, including all objects and conditions. The current state of the world is the state on which the robot acts, providing the context for its interactions and task execution.
- **Sensor Process:** This process maps the physical states of the world onto measured sensor data. The sensor process provides the control system with essential information about the environment, allowing the robot to perceive and respond to changes effectively.

Modularization of the Control Program To effectively manage the controlled system, the control program is organized in a modular and transparent manner. This structure mirrors the components of the controlled system, making it easier to understand and manage:

- **Belief State:** The belief state is the robot's internal estimate of the current world state. It serves as the foundation for the robot's actions, guiding its decision-making processes.
- **State Estimation Process:** This process robustly and accurately assesses the world state, functioning as the inverse of the sensor process. It ensures that the belief state is updated continuously and accurately, reflecting the real-time conditions of the environment.
- **Action Planning and Execution:** Using the belief state, the robot plans and executes actions. This process involves reasoning about the consequences of different action options to determine the next

* world process = environment process + robot body motion

actuation command. By doing so, the robot can perform tasks efficiently and effectively.

- **Prediction Process:** Well-structured models of the interaction between the dynamic state of the robot body and the respective physical changes in the environment facilitate the prediction process. These models enable the robot to anticipate and plan for future states, enhancing its ability to execute tasks accurately.

The continuous feedback loop between the control system and the controlled system is essential for real-time adaptation. As the robot interacts with its environment, the control program processes sensory feedback and updates the belief state accordingly. This loop ensures that the robot can adapt to changes in the environment dynamically, maintaining high performance and reliability in executing tasks.

By structuring the control program with these components, we enable the robot agent to operate autonomously and efficiently in dynamic environments. This control program-centric perspective ensures that the robot adapts to real-time changes, completing tasks accurately and efficiently while maintaining robustness and adaptability.

2.6 The URE Process View: Modelling ongoing Actions

Building on the foundational concepts of the URE system, we will now delve into the URE Process View, which employs Explicit Dynamic Action Models (EDAM) to enable robotic systems to manage and adapt ongoing actions in real-time. EDAMs, which are illustrated in Figure 2.9, enhances the flexibility, robustness, and efficiency of task execution in dynamic environments, ensuring that robots can respond flexibly to changing conditions and continuously improve their performance.

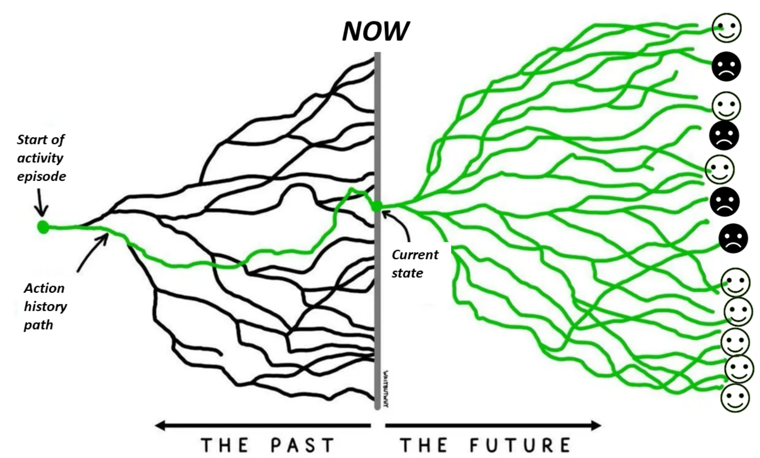


Figure 2.9: Explicit Dynamic Action Model (EDAM).

The URE Process View focuses on modeling ongoing actions to enhance the adaptability and reliability of robotic systems in dynamic environments. This approach is crucial for designing control programs that allow robots to respond to real-time changes and ensure robust task execution.

Components of EDAM An Explicit Dynamic Action Model (EDAM) consists of several key components that enhance a robot's ability to execute tasks effectively. The Current Action State represents the real-time progress of tasks, enabling robots to adapt their actions as necessary. This allows the robot to dynamically adjust its behavior based on the ongoing situation and task progress. The Distribution of Intended Body Motions shows alternative courses of action that could continue the current task, with flexibility and success probability managed by the control system. More options increase execution flexibility, and a higher number of successful continuations indicates a greater likelihood of task accomplishment. The Action Task defines the overall objective of the robot's actions, ensuring that all individual actions are aligned with the robot's mission, thus providing clear goals and direction. Finally, Action History records past actions and their outcomes, allowing the robot to learn from previous experiences. By analyzing past successes and failures, the robot can enhance its future performance, becoming more efficient and reliable.

Operation of EDAM in Task Execution EDAM empowers robots to execute tasks by continuously updating the situation context and adapting actions based on real-time perception. For example, when retrieving a milk carton from the fridge, the action model includes positional information and constraints to prevent spills, ensuring effectively-successful handling. Throughout the task, the robot monitors progress and updates the action model, aligning intended body motions with current task requirements.

Action models in EDAM expand into detailed plans that allow for predictive and adaptive planning. These plans explicitly describe body motions and their parameters, facilitating reasoning and modifications during execution. By enabling dynamic and context-aware adjustments, these plans enhance the robot's ability to perform tasks efficiently and safely. For instance, grasping poses may be selected based on foresight to facilitate subsequent actions without the need to regrasp the object, ensuring seamless and effective task completion.

Conclusion The URE Process View, through the use of EDAM, provides a detailed and dynamic representation of ongoing actions within the robot's control program. This framework ensures that robots can flexibly and robustly handle everyday activities in dynamic environments, continuously improving through real-time adjustments and learning from past experiences. This adaptability enhances the reliability and utility of robotic systems in real-world applications, making them more effective in executing complex tasks autonomously.

2.7 Formalizing the URE system

This section builds upon the previous discussion of the conceptual framework for AICOR by introducing a mathematical formalization that focuses on the control program-centric perspective of the URE system.

This formalization aims to provide a structured and precise model that enhances the development and operation of advanced robotic systems.

A mathematical formalization is a process of representing real-world phenomena, problems, or systems using mathematical concepts and structures. This involves the use of variables, equations, functions, and constraints to create a precise and unambiguous model that can be analyzed and manipulated mathematically. Mathematical formalizations are essential in various fields, including science, engineering, and computer science, as they provide a rigorous framework for understanding and solving complex problems.

In the context of AI-powered and cognition-enabled robotics (AICOR) and the User-Robot-Environment (URE) system, formalizing these interactions is crucial for developing control programs that can manage the complex interactions between users, robots, and their environments. By establishing a mathematical model, we can ensure precision, predictability, and optimization in robotic operations, thereby enhancing their ability to perform tasks autonomously and adaptively.

Components of Mathematical Formalization

The key components of this formalization include variables and parameters, equations and inequalities, functions and mappings, and constraints and conditions.

Variables and Parameters

Variables represent quantities that can change or vary within a model, such as the robot's position or the state of objects in the environment. Parameters are constants that define specific characteristics of the model, like the dimensions of the robot or fixed environmental factors. For instance, the speed of the robot might be a variable, while its maximum load capacity is a parameter.

Equations and Inequalities

Equations define relationships between variables and parameters by asserting that two expressions are equal, such as the kinematic equations governing the robot's movement. Inequalities express relational differences and constraints within the model, specifying allowable ranges for variables, like the joint angles of a robot or the boundaries within which it can operate.

Functions and Mappings

Functions describe how one variable depends on another, mapping inputs to outputs. For example, in the URE system, a function might describe how sensor inputs are translated into a perception of the environment, or how control inputs determine the robot's actions. Mappings ensure that each input is related to an appropriate output within the model, facilitating the dynamic relationship between different system components.

Constraints and Conditions

Constraints specify the operational limits within the model, such as physical boundaries for the robot's movements or safety constraints to avoid collisions. Conditions are logical statements that must be true for the system to operate correctly. In the URE system, conditions might include maintaining a safe distance from obstacles or completing tasks within a specified time frame.

Mathematical formalizations provide the foundation for analyzing and predicting the behavior of various systems and processes. In physics, Newton's Laws of Motion offer a mathematical framework to describe the motion of objects. These laws utilize variables such as force, mass, and acceleration, and employ equations like F=m*a to predict how physical systems will behave under different conditions. In the context of object manipulation, mathematical models assess the forces and stresses within structures, ensuring their safety and effectiveness. Robotics relies on mathematical formalizations for computing effectively-successful body motions, enabling robots to interact seamlessly with their environments. Additionally, fluid dynamics uses complex equations to model the movement and deformation of fluids and other substances, often employing techniques such as finite element analysis to solve these problems. In computer science, algorithms are formalized as mathematical procedures designed to solve specific problems. The efficiency and correctness of these algorithms are rigorously analyzed using mathematical techniques, ensuring they perform optimally in practical applications.

Purpose of Mathematical Formalization

The purpose of mathematical formalization in the context of the URE system is to provide a precise, predictable, and optimizable framework for robotic operations. This formalization eliminates ambiguity, allowing for exact communication of ideas and solutions, ensures predictability by enabling the modeling and anticipation of system behavior under various conditions, and facilitates optimization by identifying the best possible solutions within given constraints. These benefits are crucial for enhancing the precision, efficiency, and adaptability of robots in dynamic environments.

Formalization of the URE System

The process of formalizing the User-Robot-Environment (URE) system involves several critical steps. First, we identify the specific problem or phenomenon within the URE system that requires modeling, such as task execution or environmental interaction. Next, we define the key variables and parameters that influence the system, including the robot's position, sensor inputs, and environmental conditions. Following this, we establish mathematical relationships and equations that describe how these variables interact, creating a structured representation of the system's dynamics. Finally, we validate the model by comparing its predictions with empirical data, ensuring that the formalization accurately reflects real-world behavior and interactions within the URE system.

The process of formalizing an agent-environment system involves identifying the specific tasks and interactions, defining key variables, establishing mathematical relationships, and validating the model. This approach ensures the system's dynamics are accurately represented.

The specific problem addressed in this formalization is task execution within an environment. The goal is to model how an agent perceives its environment and selects actions to maximize its performance.

Definition of Key Variables and Parameters:

- Robot agent/control program:
  - Robot body state, Robot body motion, Percepts, and Actions
  - Robot body state: The agent's state at time t is represented by s_t.
  - Percepts: Denoted by O, representing the set of all possible observations the agent can receive.

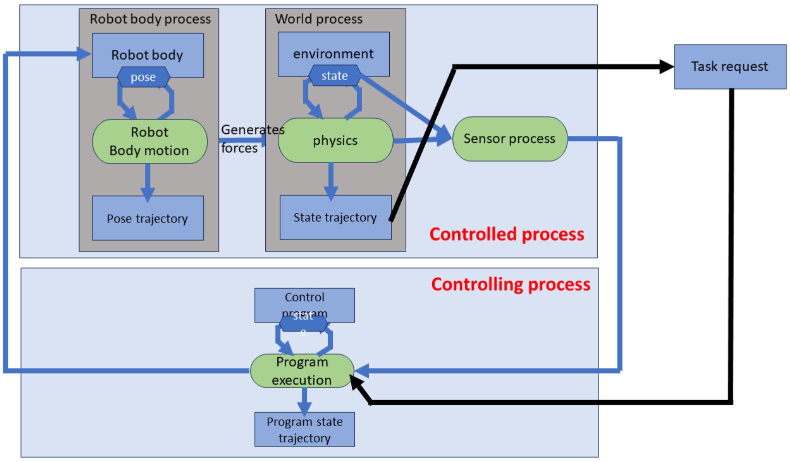


Figure 2.10: revise with the variable names and function signatures
Control program-centric conceptual framework.

- Actions: Denoted by A, representing the set of all possible actions the agent can take.

▶ Environmental States and Conditions

- States: The environment's state at time t is represented by X_t.
- Initial State: The environment's initial state X_0.
- Transition Function: $f_e : A \times X \longrightarrow X$, describing how actions influence state transitions.
- Perceptual Filtering Function: $f_p : X \longrightarrow O$, describing how states influence percepts.

▶ Establishment of Mathematical Relationships

- Agent Function
  The agent function maps sequences of percepts to actions: $f : O^T \longrightarrow A$
  For a percept sequence O_t, the action at time t is given by: $A_t = f(O_t)$
- Environment Dynamics
  The environment is defined by its states, transition function, and perceptual filter: $E = (X, f_e, f_p)$
  The state history is determined by:
  $X_0 =$ initial state
  $X_{t+1} = f_e(A_t, X_t)$
  $O_t = f_p(X_t)$

▶ Agent Program Implementation
The agent program l running on architecture M implements the agent function. The architecture M updates the agent's internal state and generates actions: $(I_{t+1}, A_t) = M(l, I_t, O_t)$
The implemented agent function is: $f(O_t) = A_t \quad \text{where} \quad (I_{t+1}, A_t) = M(l, I_t, O_t)$

▶ Performance Assessment
To evaluate the agent's performance, we define a utility function U on state histories: $U : X^T \longrightarrow \mathbb{R}$
The value of an agent function f in the environment E is: $V(f, E) = U(\text{effects}(f, E))$
For an agent program l executed by architecture M: $V(l, M, E) =$

$U(\text{effects}(f, E))$

2.8 Summary and Conclusion

Summary

This chapter provided a comprehensive overview of the conceptual framework that underpins AI-powered and cognition-enabled robotics (AICOR). We explored the intricate dynamics and interactions between key components: the user, the robot agent, and the environment. Below are the key points covered:

1. The Robot Agent System: The user, robot agent, and environment form an integrated system characterized by complex interactions and feedback loops. The robot agent operates autonomously, equipped with sensors, actuators, and a control program to interpret and execute user task requests.
2. Detailed Description of Concepts and Interactions:
   - User: Issues task requests and evaluates robot performance.
   - Robot Agent: An autonomous system that interprets tasks and interacts with the environment.
   - Environment: The physical space where the robot operates, including all objects and conditions it interacts with.
   - Task Request: Directives from the user that initiate the robot's actions.
   - Body Motion: The physical movements the robot performs to complete tasks.
   - Activity Assessment: User evaluation of the robot's performance, providing feedback for future tasks.
3. System Operation: Described the cyclical interaction process between the user, robot agent, and environment, emphasizing the continuous feedback loop that ensures task accuracy and efficiency.
4. The Body Motion Problem: Highlighted the challenge of interpreting underdetermined task requests and planning effectively-successful actions to achieve desired outcomes.
5. Long-term Autonomy: Discussed strategies for maintaining robot performance over extended periods, including environment stabilization, learning from failures, habitual behavior development, lifelong learning, and knowledge sharing among robots.
6. Cognizant Robot Agents: Introduced cognitive systems such as reinforcement learning, probabilistic reasoning, and symbolic AI that enable robots to understand and justify their actions. For example, reinforcement learning allows robots to optimize actions based on past experiences, while probabilistic reasoning helps them make decisions under uncertainty.
7. Robot Agents Acting in Physical Environments: Explored the interactions between robot agents and their environments, focusing on perception, action planning, execution, and adaptation.
8. Control Program-centric Perspective: Emphasized the importance of the control program in orchestrating the robot's interactions with the environment, ensuring accurate and efficient task completion.

9. Modeling the Ongoing Activity: Described how robots manage and adapt their actions in real-time to handle dynamic and evolving situations.
10. Dimensions of the Robot Control Problem: Analyzed the factors influencing robot control, including physical and computational makeup, task complexity, and environmental context.
11. Formalizing the Conceptual Framework: Introduced the rational robot agent model, formalizing the interaction between robots and their environments through perception-action loops and utility functions.

Conclusion

The framework for AI-powered and cognition-enabled robotics (AICOR) is essential for developing autonomous robots capable of performing complex tasks in dynamic environments. By understanding the interactions between the user, robot agent, and environment, we can design robots that are not only efficient and reliable but also adaptive and intelligent. Long-term autonomy is a critical aspect, requiring robots to stabilize their environments, learn from experiences, develop habitual behaviors, and continuously improve their capabilities. Cognizant robot agents, equipped with advanced AI and cognitive systems, represent the future of robotics, enabling machines to understand, justify, and optimize their actions.

The concepts discussed in this chapter lay the foundation for developing sophisticated robotic systems that can operate autonomously and effectively in real-world applications. As technology advances, these frameworks will become increasingly vital in creating robots that enhance our daily lives, industries, and scientific endeavors.

2.9 Exercises regrading the AICOR conceptual framework

Review Questions

- Core Concepts:
  - What are the six core concepts of the AICOR framework?
  - How do these core concepts interact within the AICOR system?
- System Operation:
  - Describe the cyclical interaction process between the user, the robot agent, and the environment.
  - How does the activity assessment influence future task requests and robot performance?
- Challenges:
  - What is the body motion problem and how does it affect the robot's task execution?
  - What strategies are proposed to achieve long-term autonomy in robotic agents?

- Cognizant Robot Agents:
  - How do cognitive capabilities enhance the performance and reliability of robotic agents?
  - Give an example of how a cognitive robot agent can explain its actions and reasoning processes.
- Control Program:
  - Why is the control program considered central to the AICOR framework?
  - Explain the role of EDAMs in modeling the ongoing activities of robotic agents.
- Dimensions of Control:
  - What are the three core aspects that influence the design of a robot's control program?
  - How do these aspects affect the robot's ability to perform tasks autonomously?
- Formal Framework:
  - Describe the rational robot agent model.
  - How do perception-action loops and utility functions contribute to the formalization of the AICOR framework?

By answering these questions, readers can assess their understanding of the chapter and identify areas that may require further review or study.

3 Logic-based Knowledge Representation & Reasoning

In Chapter 2, we transitioned from introductory knowledge to delve into the conceptual framework that supports AI-powered and cognition-enabled robotics (AICOR), providing a structured approach to understanding the interactions between users, robots, and their environments. This framework is essential for designing and implementing robots capable of performing complex tasks with high adaptability and efficiency. As Nilsson aptly stated, scientists and engineers should strive to simplify, organize, and create elegant models. Without such models, understanding intelligence enough to design intelligent machines or teach these methods would be nearly impossible. He used the example of the Golden Gate Bridge, noting that if bridge-building were a haphazard process, a structure as complex as the Golden Gate would never have been built. Successful engineering, he argued, requires the creation of simplified models that strip away complexities to focus on the core elements of a problem, making it easier to understand, teach, and build upon and then a progression of increasingly complex models. This perspective underlines the importance of creating simplified, organized, and elegant models to advance our understanding and capabilities in AI-powered and cognition-enabled robotics.

Building on this foundation, Chapter 3 explores the necessity of a rigorous, well-defined, and unambiguous mathematical apparatus for expressing solving the body motion problem and enabling robots to reason about what they are doing. Like architects, system engineers, and robotics control engineers, we require a formal framework to guide our design and implementation processes. Among the most effective tools at our disposal is logic. Genesereth and Nilsson highlight the critical role of leveraging existing logical structures:

"Anyone who attempts to develop theoretical apparatus relevant to systems that use and manipulate declaratively represented knowledge, and does so without taking into account the prior theoretical results of logicians on these topics, risks (at best) having to repeat some of the work done by the brightest minds of the twentieth century and (at worst) getting it wrong."

By integrating logic into our frameworks, we can ensure that our approaches are grounded in proven methodologies, allowing us to build intelligent systems that are both effective and reliable.

Within these logical frameworks we investigate algorithms and methods that allow robots to draw conclusions from their knowledge, make decisions, and plan actions. Our aim is to lay the foundations of robot agents that acquire and maintain an understanding of how the world works and use this understanding to successfully act the world.

Consider the scenario of a household robot tasked with performing a wide range of everyday chores, such as setting the table, cleaning up after meals, and loading the dishwasher. This robot must possess a deep understanding of its environment, the tasks it needs to perform, and the best ways to achieve these tasks efficiently.

For instance, when asked to set the table for dinner, the robot must interpret the task request by understanding what items are needed on the table for dinner, where these items are stored, and how they should be arranged based on household norms. It must then plan the actions by developing a sequence of steps to retrieve the items, navigate to the dining area, and place the items correctly, which involves spatial reasoning and the ability to handle objects carefully.

Executing the plan requires performing the actions while monitoring progress and making adjustments as needed. If an obstacle is encountered or an item is missing, the robot must adapt its plan and find alternative solutions. After completing the task, the robot should evaluate its performance, learning from any mistakes, such as placing a glass incorrectly, and improving its future performance.

Throughout the process, the robot must reason about its actions and consequences, ensuring that tasks are performed safely and efficiently. This includes not placing fragile it.

By the end of this chapter, readers will gain a comprehensive understanding of the knowledge and reasoning mechanisms essential for developing AI-powered and cognition-enabled robots. These insights will lay the groundwork for creating robots that not only perform tasks autonomously but also understand and adapt to their environment, thereby meeting the complex demands of real-world applications.

3.1 The Origins of Knowledge Representation and Reasoning

For the purpose of AICOR we take the roots of Knowledge Representation and Reasoning (KR&R) in artificial intelligence (AI) to be two foundational hypotheses that shaped the field's development: the Physical Symbol System Hypothesis and the Knowledge Representation Hypothesis. These hypotheses have profoundly influenced how AI systems are designed to represent and manipulate knowledge.

3.1.1 The Physical Symbol System Hypothesis

Proposed by Allen Newell and Herbert A. Simon in the 1970s, the Physical Symbol System Hypothesis (PSSH) posits that a physical symbol system has the necessary and sufficient means for general intelligent action. This hypothesis asserts that cognitive processes can be understood as forms of symbol manipulation, where symbols are abstract entities representing objects, concepts, or states in the world. For instance, the symbol "apple" might represent the concept of an apple. These symbols can be combined into expressions that capture complex information, such as "apple is red."

Physical Symbol System Hypothesis

A physical symbol system has the necessary and sufficient means for general intelligent action.

A physical symbol system (PSS) includes processes for creating, modifying, and interpreting these expressions using formal rules, akin to those in logic and mathematics. This framework provided a foundation for developing symbolic AI, where knowledge and reasoning are explicitly encoded using formal symbols and rules. PSSH fundamentally shifted the understanding of intelligence, emphasizing that it is not tied to a specific physical form but to the ability to process and manipulate symbols. This revolutionary idea suggested that computers, capable of manipulating symbols, could be designed to think and reason like humans. In this framework, cognitive tasks are seen as involving the manipulation of symbol structures by following formal rules, bridging the conceptual gap between human cognition and machine intelligence.

3.1.2 The Knowledge Representation Hypothesis

Building on the Physical Symbol System Hypothesis (PSSH), the Knowledge Representation Hypothesis (KRH) emphasizes the importance of **how** knowledge is structured and represented within an AI system. This hypothesis asserts that the form and organization of knowledge are crucial for enabling intelligent behavior. Effective knowledge representation allows an AI system to understand and interact with its environment, make decisions, and learn from experience.

Knowledge representation hypothesis

Any mechanically embodied intelligent process will be comprised of structural ingredients that:

- Represent (encode) knowledge about the world.
- Can be manipulated in a rational way to produce behavior that exhibits that knowledge.

The KRH suggests that any mechanically embodied intelligent process will comprise structural ingredients that represent knowledge about the world and can be manipulated rationally to produce behavior that exhibits that knowledge. This involves ensuring that knowledge representation is expressive enough to capture relevant aspects of the world and the tasks the AI system needs to perform. Additionally, the representation should allow for efficient computation, enabling the system to reason and make decisions in a timely manner. The robustness of the representation is also critical, allowing the system to handle new and unexpected situations by updating its knowledge base and incorporating new information seamlessly.

Caveat: Note, in AICOR we don't adopt the Physical Symbol System and Knowledge Representation Hypotheses to their fullest extent. Rather we consider physical symbol systems designed according to the KRH as powerful tools for realizing AICOR robots.

3.1.3 Solving the Body Motion Problem with Physical Symbol Systems

Let us now consider how PSSs can solve the body motion problem in robotics, transforming robots into systems capable of planning and executing complex physical tasks. The body motion problem involves determining the effectively-successful physical movements required for a robot to fulfill a task request while avoiding unwanted side effects. By leveraging PSSH, robots can encode, manipulate, and reason about symbols representing tasks, environments, and actions, thereby facilitating intelligent and adaptive behavior.

To apply PSSH to the body motion problem, symbols and symbol structures are used to represent various elements of the task and environment. High-level task instructions, such as "bring me a drink," are represented as symbols within the robot's control system, encapsulating essential task details for further decomposition. The robot's environment, including objects and their states, is also represented symbolically, allowing meaningful perception and interaction. Possible actions, such as "move-to," "grasp," and "lift," are encoded as symbols with associated rules, enabling the robot to decompose high-level tasks into manageable subtasks and sequence actions accordingly.

Task decomposition involves breaking down a high-level task into subtasks using symbolic rules. For instance, a request to "bring me a drink" can be decomposed into: "move to kitchen," "locate drink," "grasp drink," and "return to user." These subtasks are structured into a sequence of actions, ensuring systematic and logical task execution. During execution, real-time feedback allows the robot to adjust its actions, ensuring smooth and efficient completion despite unforeseen obstacles.

For example, in retrieving milk from a refrigerator, the process starts with the user command, "Get the milk from the refrigerator." The robot translates this into a symbolic task representation: "get-milk(refrigerator)." The robot's knowledge base includes the refrigerator's location, typical milk location, and how to operate the door. The robot plans an action sequence with symbols such as "Move(refrigerator)," "Open(refrigerator-door)," "Locate(milk)," "Grasp(milk)," "Lift(milk)," "Close(refrigerator-door)," "Move(user)," and "Deliver(milk)," creating a clear, step-by-step framework.

The robot navigates to the refrigerator, confirms its position, identifies the refrigerator, opens the door, locates the milk, grasps and lifts it, closes the door, and returns to the user with the milk. Throughout this process, sensor feedback ensures accurate adjustments. The robot can explain its actions based on its internal representations, such as "I moved to the refrigerator because that's where the milk is stored" and "I opened the refrigerator door to access the milk," demonstrating its autonomous and adaptive task execution.

Conclusion By combining the principles of the Physical Symbol System Hypothesis and the Knowledge Representation Hypothesis, a robot can effectively plan and execute complex tasks like retrieving milk from a refrigerator. The PSSH provides the framework for symbol manipulation and action sequencing, while the KRH ensures that the robot has a detailed and adaptive understanding of its environment, allowing it to perform tasks intelligently and explain its actions. This integrated approach enables robots to not only act autonomously but also adapt to dynamic conditions and justify their behavior, leading to more robust and reliable robotic systems.

3.2 Building Cognitive Robot Abilities with Logic-Based Symbol Systems

In the preceding sections, we explored the foundational hypotheses that underpin Knowledge Representation and Reasoning (KR&R) in artificial intelligence, specifically the Physical Symbol System Hypothesis (PSSH) and the Knowledge Representation Hypothesis (KRH). We examined how these hypotheses enable robots to represent, manipulate, and reason about symbols, facilitating intelligent and adaptive behavior. We also discussed the application of PSSH to solve the body motion problem, highlighting how symbolic representation allows robots to decompose complex tasks into manageable actions.

Building on these concepts, Section 3.2 delves deeper into the realization of core cognitive capabilities for solving the body motion problem. The key idea of this section is that these capabilities are implemented as a physical symbol system that is firmly built on top of a logical framework. This integration of symbolic representation with logic provides a robust foundation for developing advanced cognitive functions in robotics. By leveraging logical structures, robots can achieve a higher level of reasoning, decision-making, and adaptability, enabling them to perform complex tasks with greater precision and efficiency.

In this section, we will explore how logic-based systems enhance the cognitive abilities of robots, allowing them to interpret and execute tasks within dynamic environments. We will discuss the essential components of these systems, their implementation, and their impact on the development of intelligent robotic agents. Through this examination, we aim to provide a comprehensive understanding of how symbolic logic forms the backbone of cognitive robotics, driving innovation and advancing the field.

3.3 What is a Logic?

A logic is a systematic framework used for representing and reasoning about knowledge. It consists of a set of formal principles and rules that guide the process of drawing conclusions from given premises. In the context of artificial intelligence and robotics, logic provides the foundational structure for encoding knowledge, making decisions, and performing reasoning tasks.

Key Components of Logic: To understand how logic functions as a foundation for cognitive capabilities in AI and robotics, it is essential to explore its key components. These components provide the structure and mechanisms through which logical reasoning is performed, enabling systems to represent and manipulate knowledge effectively. The primary components of logic include syntax, semantics, and calculus, each playing a crucial role in the logical framework.

- **Syntax:** The formal structure and rules that define how symbols and expressions can be constructed. Syntax specifies the allowable combinations of symbols to form valid statements or formulas.

- **Semantics:** The meaning or interpretation of the symbols and expressions within a logical system. Semantics provides the rules for assigning truth values to statements based on their structure and the information they represent.
- **Calculus:** The formal system that prescribes how to compute the symbolic structures of a logic. It includes:
  - **Inference Rules:** The logical procedures used to derive new statements or conclusions from existing ones. Inference rules are the mechanisms by which logical reasoning is performed, enabling the system to draw valid conclusions from known facts.
  - **Axioms:** The foundational statements or assumptions within a logical system that are accepted as true without proof. Axioms serve as the starting point for logical reasoning and are used to derive further knowledge within the system.

One objective in designing logics for problem-solving is to ensure they are both correct and complete. The concepts of correctness and completeness are critical to understanding the efficacy of a logical calculus. Correctness ensures that any statement derived using the inference rules is semantically valid, meaning it holds true within the logical system's framework. This guarantees that the reasoning process does not produce false conclusions from true premises. Completeness, on the other hand, guarantees that if a statement is semantically valid (true in all models of the logic), it can be derived using the inference rules. This means the calculus is powerful enough to capture all truths within the logical system. Together, correctness and completeness ensure that the logical system is both sound and capable of deriving all true statements, making it a reliable tool for reasoning and decision-making in artificial intelligence and robotics.

If the calculus is both correct and complete, we have a decision procedure for determining whether a formula Φ is true. By enumerating the sentences that the calculus can generate, we can check if Φ is contained within this set. If Φ is included, then Φ is true. This is important because it allows a computer system to syntactically test the validity of sentences without having to understand their meaning. This capability is crucial for automated reasoning and verification, enabling systems to efficiently determine the truth of statements based purely on their syntactic form, thereby enhancing their ability to perform complex reasoning tasks.

In summary, logic is an essential tool in artificial intelligence and robotics, providing the structure and rules necessary for representing knowledge, performing reasoning tasks, and enabling intelligent behavior.

Together, these components—syntax, semantics, inference rules, and proof theory—form the foundation of a logic system. They enable the precise representation, interpretation, and manipulation of knowledge, making logic a powerful tool for developing intelligent systems capable of autonomous decision-making and problem-solving. By understanding and applying these elements, we can create robust frameworks for reasoning about complex information and solving intricate problems in various domains.

In the context of problem-solving, such as solving the body motion problem, logic is used to model real-world situations and derive solutions through logical reasoning. A **model** in logic is an interpretation that assigns meaning to the symbols and statements within a logical system, effectively representing a possible state of the world. For instance, in a model of a robot's environment, symbols could represent objects, actions, and their relationships, providing a structured way to reason about the robot's tasks.

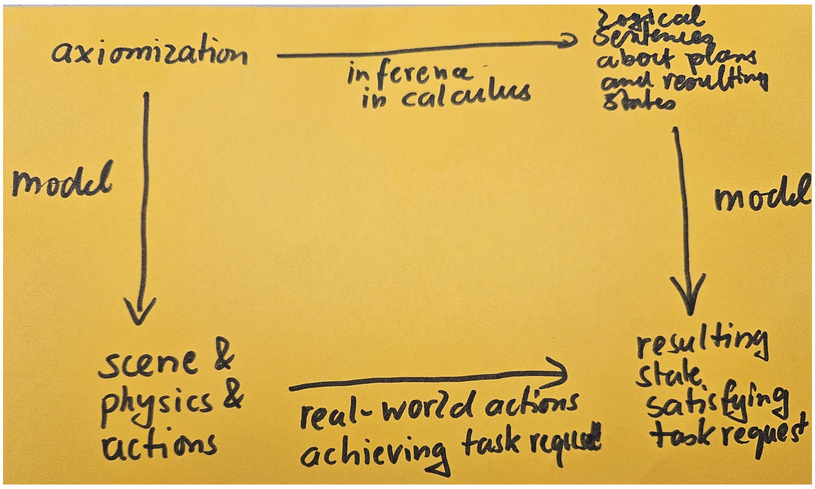


Figure 3.1: Problem-solving with logic.

Axiomatization involves defining a set of axioms or fundamental truths that form the basis of the logical system. These axioms are used to derive further statements and conclusions through the application of inference rules. By constructing a formal model with well-defined axioms, logic allows for precise and unambiguous problem representation, ensuring that derived solutions are logically sound.

For example, in robotics, axioms might include statements like "all objects in the kitchen are reachable" or "if a robot grasps an object, it can move it to a specified location." Using these axioms, the robot can plan and execute actions to solve tasks, such as fetching an item or navigating an environment. The logical framework ensures that each step in the problem-solving process is validated against the defined axioms and inference rules, leading to reliable and consistent outcomes.

In summary, logic is an essential tool in artificial intelligence and robotics, providing the structure and rules necessary for representing knowledge, performing reasoning tasks, and enabling intelligent behavior. Through the use of models and axiomatizations, logic facilitates accurate problem-solving by ensuring that conclusions are derived systematically and correctly from foundational truths.

3.4 Predicate Logic as an Instance of Logic

Predicate logic is a powerful and expressive symbolic KR&R formalism that provides a systematic way to represent and reason about knowledge. Predicate logic is used to represent and reason about propositions involving objects and their relationships. In this section, we introduce, describe, and explain the components of the predicate logic, focusing on its syntax, semantics, and calculus.

3.4.1 Syntax

The syntax of predicate logic defines the symbols and rules for constructing well-formed formulas. It includes the following components:

- Formation Rules: The formation rules for predicate logic can be summarized as follows:
  - Terms:
    - A constant is a term.
    - A variable is a term.
    - If f is an n-place function symbol and $t_1, t_2, ..., t_n$ are terms, then $f(t_1, t_2, ..., t_n)$ is a term.
  - Formulas:
    - If P is an n-place predicate symbol and $t_1, t_2, ..., t_n$ are terms, then $P(t_1, t_2, ..., t_n)$ is an atomic formula.
    - If α is a formula, then $\neg\alpha$ is a formula.
    - If α and β are formulas, then $(\alpha \wedge \beta)$, $(\alpha \vee \beta)$, $(\alpha \rightarrow \beta)$, and $(\alpha \leftrightarrow \beta)$ are formulas.
    - If α is a formula and x is a variable, then $\forall x\alpha$ and $\exists x\alpha$ are formulas.

Examples of syntax: Representing Robot Knowledge

- Vocabulary: First, we need to represent the relevant knowledge about the environment and the task:
  - Objects: *robot*, *milk*, *refrigerator*
  - Predicates:
    In(milk, refrigerator): The milk is in the refrigerator.
    At(robot, location): The robot is at a specific location.
    Holding(robot, milk): The robot is holding the milk.
    Open(refrigerator): The refrigerator is open.
    Closed(refrigerator): The refrigerator is closed.
- Representing the Initial State: We define the initial state of the world using predicates:
  In(milk,refrigerator), *At(robot,kitchen)*, *Closed(refrigerator)*
- Goal: The goal state is for the robot to be holding the milk:
  Holding(robot,milk)
- Actions and Preconditions: We define the actions the robot can take and their preconditions: Examples:
  - Open Refrigerator:
    - Preconditions: *At(robot, refrigerator)* ∧ *Closed(refrigerator)*
    - Effects: *Open(refrigerator)* ∧ ¬*Closed(refrigerator)*
  - Pick Up Milk:
    - Preconditions: *At(robot, refrigerator)* ∧ *In(milk, refrigerator)* ∧ *Open(refrigerator)*
    - Effects: *Holding(robot, milk)* ∧ ¬*In(milk, refrigerator)*

3.4.2 Semantics

The semantics of predicate logic provides meaning to the syntactically correct statements. It involves interpreting the symbols and formulas within a logical model. A model $\mathcal{M}$ for predicate logic consists of:

- **Domain:** A non-empty set D of objects.
- **Interpretation:** An assignment of meaning to the non-logical symbols:
  - Each constant is mapped to an element of D.
  - Each n-ary predicate is mapped to a subset of D^n.
  - Each n-ary function is mapped to a function from D^n to D.
- **Valuation:** An assignment of elements of D to the variables.

Formulas are evaluated as true or false based on the interpretation and valuation. For example:

- An atomic formula $P(x_1, \ldots, x_n)$ is true in $\mathcal{M}$ if the tuple $(v(x_1), \ldots, v(x_n))$ is in the interpretation of P.
- $\neg\varphi$ is true if φ is false.
- $\varphi \wedge \psi$ is true if both φ and ψ are true.
- $\forall x\, \varphi$ is true if φ is true for every possible valuation of x.
- $\exists x\, \varphi$ is true if there is at least one valuation of x for which φ is true.

3.4.3 Calculus

The calculus of predicate logic, also known as a deductive system, provides the formal rules for deriving new formulas from given formulas. Some of the key inference rules include:

- **Modus Ponens:** From φ and $\varphi \to \psi$, infer ψ.
- **Modus Tollens:** From $\neg\psi$ and $\varphi \to \psi$, infer $\neg\varphi$.
- **Generalization:** From $\varphi(x)$, infer $\forall x\, \varphi(x)$, where x is not free in any assumption.
- **Existential Instantiation:** From $\exists x\, \varphi(x)$, infer $\varphi(c)$ for some new constant c.

A **formal proof** in predicate logic is a finite sequence of well-formed formulas, where each formula is either an axiom or derived from previous formulas using inference rules. The goal is to derive a formula φ from a set of axioms Γ, denoted as $\Gamma \vdash \varphi$.

3.4.4 Conclusion

Predicate logic, with its well-defined syntax, semantics, and calculus, serves as a foundational framework for representing and reasoning about knowledge. By leveraging its formal structure, we can encode complex information, interpret it within logical models, and derive new knowledge through rigorous inference. This makes predicate logic an invaluable tool in the development of intelligent systems capable of sophisticated reasoning and problem-solving.

3.5 Using Predicate Logic to Solve the Robot Body Motion Problem

Knowledge representation and reasoning (KR&R) methods are essential for enabling robots to make informed and intelligent decisions. These methods allow robots to represent actions, their preconditions, and effects using formalisms such as situation calculus or event calculus. By reasoning about the consequences of potential actions, robots can select the most appropriate course of action to achieve their goals. This capability is crucial for ensuring that robots can effectively plan and execute tasks in various scenarios.

The symbolic nature of KR&R makes the reasoning process interpretable and explainable, which is crucial for building trust and accountability in robotic decision-making. Robots can provide explanations for their decisions, allowing humans to understand and verify the decision-making process. KR&R also enables robots to plan sequences of actions to achieve goals, taking into account knowledge about actions, preconditions, and effects. This structured and efficient decision-making capability extends to failure analysis and recovery, where robots can detect and diagnose failures, and plan recovery strategies, thereby improving their robustness and adaptability in dynamic and uncertain environments.

3.5.1 Formalization in Situation Calculus

enhance with **at(robot,refrigerator)**

- ▸ Constants and Terms
  - Constants:
    - * Milk: The object representing the milk.
    - * Refrigerator: The object representing the refrigerator.
    - * Agent: The object representing the agent (the person performing the task).
  - Actions:
    - * Open(Refrigerator): The action of opening the refrigerator.
    - * TakeOut(Agent, Milk): The action of the agent taking the milk out of the refrigerator.
    - * Close(Refrigerator): The action of closing the refrigerator.
  - Situations
    - * S0: The initial situation where the refrigerator is closed, and the milk is inside.
    - * do(a, s): The function representing the situation resulting from performing action a in situation s.
  - Fluents wrong!
    - * Inside(Milk, Refrigerator, s): True if the milk is inside the refrigerator in situation s.
    - * Holding(Agent, Milk, s): True if the agent is holding the milk in situation s.
    - * RefrigeratorOpen(Refrigerator, s): True if the refrigerator is open in situation s.
  - Fluents: needs update

  * Holds Predicate: The holds(fluent, situation) predicate will be used to describe which fluents are true in which situations.

Formalization of Actions

- Open Refrigerator:
  $\text{Poss}(\text{Open}(\text{Refrigerator}), s) \equiv \neg\text{holds}(\text{RefrigeratorOpen}(\text{Refrigerator}), s)$
  $\text{holds}(\text{RefrigeratorOpen}(\text{Refrigerator}), \text{do}(\text{Open}(\text{Refrigerator}), s)) \equiv \text{true}$
- Take Out Milk:
  $\text{Poss}(\text{TakeOut}(\text{Agent}, \text{Milk}), s) \equiv \text{holds}(\text{RefrigeratorOpen}(\text{Refrigerator}), s) \wedge \text{holds}(\text{Inside}(\text{Milk}, \text{Refrigerator}), s)$
  $\text{holds}(\text{Holding}(\text{Agent}, \text{Milk}), \text{do}(\text{TakeOut}(\text{Agent}, \text{Milk}), s)) \equiv \text{true}$
  $\text{holds}(\text{Inside}(\text{Milk}, \text{Refrigerator}), \text{do}(\text{TakeOut}(\text{Agent}, \text{Milk}), s)) \equiv \text{false}$
- Close Refrigerator:
  $\text{Poss}(\text{Close}(\text{Refrigerator}), s) \equiv \text{holds}(\text{RefrigeratorOpen}(\text{Refrigerator}), s)$
  $\text{holds}(\text{RefrigeratorOpen}(\text{Refrigerator}), \text{do}(\text{Close}(\text{Refrigerator}), s)) \equiv \text{false}$
- Initial Situation: The initial situation S_0 is defined as:
  $\text{holds}(\text{Inside}(\text{Milk}, \text{Refrigerator}), S_0) \equiv \text{true}$
  $\text{holds}(\text{RefrigeratorOpen}(\text{Refrigerator}), S_0) \equiv \text{false}$
  $\text{holds}(\text{Holding}(\text{Agent}, \text{Milk}), S_0) \equiv \text{false}$
- Task: Get the Milk Out of the Refrigerator. To formalize the entire task of getting the milk out of the refrigerator, we describe a sequence of actions leading to the desired situation.

  1. Open the refrigerator:
     $\text{do}(\text{Open}(\text{Refrigerator}), S_0)$
  2. Take the milk out:
     $\text{do}(\text{TakeOut}(\text{Agent}, \text{Milk}), \text{do}(\text{Open}(\text{Refrigerator}), S_0))$
  3. Close the refrigerator:
     $\text{do}(\text{Close}(\text{Refrigerator}), \text{do}(\text{TakeOut}(\text{Agent}, \text{Milk}), \text{do}(\text{Open}(\text{Refrigerator}), S_0)))$

- In the final situation, we want to check if the agent is holding the milk and the refrigerator is closed:
  $\text{holds}(\text{Holding}(\text{Agent}, \text{Milk}), \text{do}(\text{Close}(\text{Refrigerator}), \text{do}(\text{TakeOut}(\text{Agent}, \text{Milk}), \text{do}(\text{Open}(\text{Refrigerator}), S_0)))) \equiv \text{true}$
  $\text{holds}(\text{RefrigeratorOpen}(\text{Refrigerator}), \text{do}(\text{Close}(\text{Refrigerator}), \text{do}(\text{TakeOut}(\text{Agent}, \text{Milk}), \text{do}(\text{Open}(\text{Refrigerator}), S_0)))$ false

3.6 Modeling Issues in Logic-based Robot Agency

3.6.1 Generality and Transferability

- Cambrian Explosion coming to Robotics
- Giskard as a logical formula
- the "any" problem

3.6.2 The "At" Fluent and the “Milk”

The at fluent entails a lot of hand waving

- At the refrigerator really means for the robot to be at a place where it can reach and see the handle of the refrigerator door, open the refigerator door and then see and reach the milk.
- The “milk” really means the container that contains a milk, its shape, friction, weight, and other physical properties that might interact with the feasibility of picking it up.

3.6.3 Lazy and On-demand symbolic representations

In our discussion so far we have assumed that the symbols and symbol structures are defined before the problem-solving step and that they are supposed to provide appropriate models for all problems the robot has to solve in the future.

But what if we create the symbol structures on the fly and on-demand for the specific problems coming up.

There are two mechanisms we will introduce below that are very powerful extensions of the logic approach: 1) procedural attachments and 2) virtual reality scene graphs.

3.6.4 Egg cracking

The "egg cracking problem" is a classic example used to illustrate the challenges of commonsense and intuitive physics reasoning in artificial intelligence (AI). It was introduced by Leora Morgenstern in her paper "Mid-size Axiomatizations and Theories: Perfect Information Games" (1996). The problem describes a scenario where an agent (e.g., a robot) needs to crack an egg into a bowl without making a mess or breaking the yolk. While this task may seem trivial for humans, it requires a significant amount of commonsense knowledge and intuitive understanding of physics to be successfully executed by an AI system.

Some of the key aspects of the egg cracking problem that make it challenging for AI systems include:

- Representing and reasoning about the physical properties of objects: The agent needs to understand the physical properties of the egg (e.g., its fragility, the difference between the shell and the yolk, the fact that the yolk is more delicate than the shell). It also needs to understand the properties of the bowl (e.g., its rigidity, its ability to contain liquids).
- Understanding the dynamics of object interactions: The agent must reason about the forces involved in cracking the egg, such as the amount of force required to break the shell without damaging the yolk. It needs to understand the consequences of applying too much or too little force, and how to adjust its actions accordingly.

- Commonsense knowledge about the task: The agent requires commonsense knowledge about the typical way humans crack eggs, such as holding the egg over the bowl, tapping it against a hard surface, and separating the shell halves. It also needs to understand the purpose of the task (e.g., cracking the egg for cooking) and the desired outcome (e.g., an intact yolk in the bowl).
- Intuitive physics reasoning: The agent must have an intuitive understanding of physics concepts like gravity, friction, and the behavior of solid and liquid materials. It needs to reason about how the egg will behave when cracked, how the yolk and shell will separate, and how to control the motion of the yolk to prevent it from breaking or spilling.

The egg cracking problem highlights the need for AI systems to have a rich knowledge base that combines commonsense knowledge, intuitive physics understanding, and the ability to reason about the dynamics of object interactions. It has been used as a benchmark for testing the capabilities of AI systems in commonsense and intuitive physics reasoning, as well as for developing new techniques and approaches in these areas.

3.6.5 Making Knowledge "action-able"

References

- In Defense of Logic
- Algorithms = Logic + Control
- Ernest Davis
- Drew McDermott: Logic for Action
- DVM: a critique of pure reason

3.7 Ontologies and Description Logics

3.7.1 Concept Definitions in Predicate Logic

Predicate logic, also known as first-order logic, is a formal system used to express propositions with predicates and quantifiers. It extends propositional logic by allowing the use of variables, quantifiers, and predicates, making it more expressive and capable of representing more complex statements. Here, we will apply predicate logic to define the concepts in the User-Robot-Environment (URE) framework and describe their relationships.

Syntax and Semantics

- **Syntax:** The syntax of predicate logic defines the rules for constructing valid expressions (well-formed formulas). A well-formed formula might be an atomic formula like $P(x)$ or a more complex formula involving quantifiers and logical connectives, such as $\forall x(P(x) \rightarrow Q(x))$.

- **Semantics:** The semantics of predicate logic assigns meanings to the symbols and formulas. An interpretation provides a domain of discourse and assigns values to the variables and predicates. For example, in the domain of natural numbers, $P(x)$ might mean "x is even".

Concept Definitions in the URE Framework:

1. User: Predicate: User(x): $User(x)$ represents that x is a user.
   Example: $User(John)$ means "John is a user."
2. Robot Agent: Predicate: RobotAgent(x); $RobotAgent(x)$ represents that x is a robot agent.
   Example: $RobotAgent(R1)$ means "R1 is a robot agent."
3. Environment: Predicate: Environment(x): $Environment(x)$ represents that x is an environment.
   Example: $Environment(Kitchen)$ means "Kitchen is an environment."

Relationships Between Concepts

1. User and Task Request: Relationship: A user issues a task request to a robot.
   Logic: $\forall u \forall r \forall t (User(u) \wedge RobotAgent(r) \wedge TaskRequest(u, r, t) \rightarrow IssuedTaskRequest(u, r, t))$ "For all u, r, and t, if u is a user, r is a robot agent, and u issues task t to r, then t is an issued task request."
2. Task Request and Robot Agent: Relationship: The robot agent receives and processes the task request.
   Logic: $\forall r \forall t (RobotAgent(r) \wedge TaskRequest(u, r, t) \rightarrow ProcessesTaskRequest(r,$
   "For all r and t, if r is a robot agent and receives task request t, then r processes task request t."
3. Robot Agent and Environment: Relationship: The robot interacts with the environment.
   Logic: $\forall r \forall e (RobotAgent(r) \wedge Environment(e) \rightarrow InteractsWithEnvironment($
   "For all r and e, if r is a robot agent and e is an environment, then r interacts with e."

3.7.2 Description Logics for Concept Definition

Description logics (DLs) are a family of formal knowledge representation languages that are used to describe the knowledge about the concepts and relationships within a domain. They are particularly suited for defining and reasoning about the concepts and relationships that exist in knowledge bases.

Basic Components of Description Logics

1. Concepts: Concepts (or classes) represent sets of individuals or objects in the domain. They are analogous to unary predicates in predicate logic.
2. Roles: Roles (or properties) represent binary relationships between individuals or objects in the domain. They are analogous to binary predicates in predicate logic.

3. Individuals: Individuals (or instances) represent the basic objects in the domain. They are analogous to constants in predicate logic.

Syntax of Description Logics The syntax of description logics is composed of atomic concepts, atomic roles, and individuals, along with various constructors to build complex concepts and roles.

1. Atomic Concepts and Roles:
   - Atomic Concept: Represented by uppercase letters, such as A, B, $Person$, $Robot$.
   - Atomic Role: Represented by lowercase letters, such as r, s, $hasChild$, $worksIn$.
   - Individual: Represented by lowercase letters, such as a, b, $john$, $r1$.
2. Concept Constructors
   - Top Concept: $\top$ (universal concept) includes all individuals in the domain.
   - Bottom Concept: $\bot$ (empty concept) includes no individuals.
   - Conjunction (Intersection): $C \sqcap D$ (analogous to logical AND) includes individuals that belong to both concepts C and D.
     Example: $Person \sqcap Employee$
   - Disjunction (Union): $C \sqcup D$ (analogous to logical OR) includes individuals that belong to either concept C or concept D.
     Example: $Person \sqcup Robot$
   - Negation (Complement): $\neg C$ (analogous to logical NOT) includes individuals that do not belong to concept C.
     Example: $\neg Person$
   - Existential Quantification: $\exists r.C$ includes individuals that are related by role r to at least one individual that belongs to concept C.
     Example: $\exists hasChild.Doctor$ (individuals having at least one child who is a doctor)
   - Universal Quantification: $\forall r.C$ includes individuals that are related by role r only to individuals that belong to concept C.
     Example: $\forall hasChild.Person$ (individuals all of whose children are persons)
   - Value Restriction: $\forall r.C$ (every related individual must be in C).
     Example: $\forall hasChild.Student$ (individuals all of whose children are students)

Representing the URE Framework in Description Logic

he User-Robot-Environment (URE) framework can be represented in description logic by defining the relevant concepts, roles, and individuals. Here, we will describe each component of the URE framework and its relationships using description logic notation.

Basic Components in Description Logic

1. Concepts:
   - User: $User$

- Robot Agent: $RobotAgent$
- Environment: $Environment$
- Task Request: $TaskRequest$
- Action: $Action$
- Evaluation: $Evaluation$

2. Roles

- issuesTask: Relates a User to a TaskRequest.
- processesTask: Relates a RobotAgent to a TaskRequest.
- interactsWith: Relates a RobotAgent to an Environment.
- executesAction: Relates a RobotAgent to an Action.
- changesEnvironment: Relates a RobotAgent and an Action to an Environment.
- evaluatesTask: Relates a User, RobotAgent, TaskRequest, and Evaluation.

Description Logic Representation

1. Concept Definitions:
   $User \sqsubseteq \top$ (User is a concept that includes all individuals who are users.), $RobotAgent \sqsubseteq \top$, $Environment \sqsubseteq \top$, $TaskRequest \sqsubseteq \top$, $Action \sqsubseteq \top$, $Evaluation \sqsubseteq \top$
2. Role Definitions

- $issuesTask \sqsubseteq User \times TaskRequest$: The issuesTask role relates users to task requests.
- $processesTask \sqsubseteq RobotAgent \times TaskRequest$: The processesTask role relates robot agents to task requests.
- $interactsWith \sqsubseteq RobotAgent \times Environment$: The interactsWith role relates robot agents to environments.
- $executesAction \sqsubseteq RobotAgent \times Action$: The executesAction role relates robot agents to actions.
- $changesEnvironment \sqsubseteq RobotAgent \times Action \times Environment$: The changesEnvironment role relates robot agents and actions to environments.
- $evaluatesTask \sqsubseteq User \times RobotAgent \times TaskRequest \times Evaluation$: The evaluatesTask role relates users, robot agents, task requests, and evaluations.

3. Complex Concept Definitions

- Task Requests Issued by Users: $\exists issuesTask.TaskRequest \sqsubseteq User$
- Task Requests Processed by Robot Agents: $\exists processesTask.TaskRequest \sqsubseteq RobotAgent$
- Robot Agents Interacting with Environments: $\exists interactsWith.Environment$ $RobotAgent$
- Robot Agents Executing Actions: $\exists executesAction.Action \sqsubseteq RobotAgent$
- Robot Agents Changing Environments: $\exists changesEnvironment.Environmen$ $RobotAgent \sqcap \exists changesEnvironment.Action$
- Evaluations of Task Requests by Users: $\exists evaluatesTask.Evaluation \sqsubseteq User \sqcap \exists evaluatesTask.RobotAgent \sqcap \exists evaluatesTask.TaskRequest$

4. ABox (Assertional Box):

- $User(John)$: John is a user.
- $RobotAgent(R1)$: R1 is a robot agent.

- $Environment(Kitchen)$: Kitchen is an environment.
- $TaskRequest(BringMilk), issuesTask(John, BringMilk)$: John issues the task request 'BringMilk' to R1.
- $processesTask(R1, BringMilk)$: R1 processes the task request 'BringMilk'.
- $interactsWith(R1, Kitchen)$: R1 interacts with the Kitchen.
- $Action(MoveToFridge)$: R1 executes the action 'MoveToFridge'.
- $executesAction(R1, MoveToFridge)$: R1 changes the environment (Kitchen) through the action 'MoveToFridge'.
- $changesEnvironment(R1, MoveToFridge, Kitchen)$: John evaluates the task request 'BringMilk' performed by R1 as 'Success'.
- $evaluatesTask(John, R1, BringMilk, Success)$

Summary Using description logic, we can formally represent the concepts and relationships within the URE framework. This structured representation allows for precise modeling and reasoning about the interactions between users, robot agents, and environments, facilitating the development and analysis of autonomous robotic systems.

3.7.3 SOMA – an Ontology for AICOR

- top-level ontologies
- robot agents
- control programs

3.7.4 Domain and Task Ontologies

3.7.5 Semantic Web

- semantic web
- others can see and link to ontologies
- semantic web services

3.7.6 Knowledge Services for Robots

openEASE

EuroCore

Robotics Institute Germany

3.8 Knowledge Graphs

A knowledge graph is a structured representation of knowledge in the form of a graph, where nodes represent entities (such as people, places, and concepts), and edges represent the relationships between these entities. This graphical structure enables easy visualization and understanding of complex interconnections within a domain of knowledge.

Characterization and Explanation as Light-Weight Knowledge Bases

- **Structure and Simplicity:** Knowledge graphs are characterized by their simple, flexible structure. Unlike traditional, heavy-weight knowledge bases, which might require extensive schemas and intricate ontologies, knowledge graphs often rely on a more straightforward schema, making them easier to build, manage, and extend. This simplicity allows for quick adaptation and scalability as new information becomes available.
- **Integration and Interoperability:** One of the key strengths of knowledge graphs as light-weight knowledge bases is their ability to integrate diverse data sources. By using a common framework, such as RDF (Resource Description Framework) or property graphs, knowledge graphs can merge information from various datasets seamlessly. This interoperability enhances their utility in aggregating and querying vast amounts of data without the need for complex data warehousing solutions.
- **Flexibility in Querying:** Knowledge graphs support flexible querying, enabling users to traverse relationships and uncover insights through graph traversal algorithms or query languages like SPARQL (for RDF-based graphs) and Gremlin (for property graphs). This capability makes knowledge graphs highly effective for applications requiring dynamic and complex queries, such as recommendation systems, semantic search, and data analytics.
- **Semantic Richness:** Despite being light-weight, knowledge graphs retain a level of semantic richness that allows for meaningful data representation and inference. By incorporating semantic relationships and contextual information, knowledge graphs can enhance the understanding and utilization of data. This semantic layer enables more accurate and relevant results in various applications, from natural language processing to AI-driven analytics.
- **Ease of Maintenance and Expansion:** The light-weight nature of knowledge graphs translates to lower maintenance overhead. Adding new entities and relationships is often straightforward and does not require extensive reconfiguration. This ease of expansion is crucial for domains where knowledge evolves rapidly, such as technology, healthcare, and business intelligence.

Applications and Advantages

- Data Integration: Knowledge graphs excel in integrating disparate data sources, providing a unified view of information.
- Recommendation Systems: By analyzing relationships between entities, knowledge graphs can generate personalized recommendations.
- Semantic Search: Enhanced search capabilities arise from understanding the context and relationships within the data.
- AI and Machine Learning: Knowledge graphs provide structured data that can improve the training and performance of AI models.

In summary, knowledge graphs serve as light-weight knowledge bases by offering a flexible, scalable, and semantically rich framework for representing and querying complex data. Their ability to integrate diverse data sources, support dynamic querying, and adapt to evolving knowledge makes them invaluable tools in various fields, driving innovation and enhancing data-driven decision-making.

- discovering new entity categories and relations
- example: enabling the robot to competently separate waste

3.8.1 Knowledge Graphs as a KR&R Framework

What is a Knowledge Graph? A knowledge graph is a structured representation of knowledge that uses a graph-structured data model to represent and operate on data. It consists of nodes, edges, and labels, where nodes represent entities (such as objects, events, or concepts), edges represent the relationships between these entities, and labels provide additional context or attributes to the nodes and edges.

Components of a Knowledge Graph

- Nodes: Represent entities such as people, places, things, or concepts.
- Edges: Define the relationships between nodes.
- Labels: Provide additional information about the nodes and edges.

Characteristics and Functions Knowledge graphs are designed to integrate, unify, and link data from various sources, providing a comprehensive and interconnected view of information. They are used to:

- Store Interlinked Descriptions: Knowledge graphs store descriptions of entities and their relationships, enabling a more contextual understanding of data
- Enable Data Integration and Analytics: By linking data from different sources, knowledge graphs facilitate data integration, analytics, and sharing
- Support Reasoning and Inference: Knowledge graphs can apply reasoning to derive new knowledge from existing data, making implicit information explicit
- Enhance Search and Recommendation Systems: They are used in search engines and recommendation systems to provide more accurate and contextually relevant results

Applications Knowledge graphs have a wide range of applications, including:

- Search Engines: Used by Google, Bing, and other search engines to enhance search results with contextual information
- Question-Answering Systems: Powering systems like WolframAlpha, Siri, and Alexa to provide accurate answers to user queries
- Scientific Research: Applied in fields such as genomics and systems biology to integrate and analyze complex datasets
- Enterprise Solutions: Used in various industries for data integration, semantic search, and intelligent content recommendation

Relation to Predicate Logic Predicate logic, also known as first-order logic, is a formal system in logic that uses quantified variables over non-logical objects and allows the use of sentences to express facts about these objects. It is a powerful tool for knowledge representation and reasoning in artificial intelligence (AI).

Predicate Logic in Knowledge Graphs Knowledge graphs often utilize predicate logic to represent and reason about the relationships between entities. In a knowledge graph:

- Nodes represent entities or objects.
- Edges represent predicates or relationships between entities.
- Triples (subject, predicate, object) are used to express facts, similar to how predicate logic uses predicates to relate subjects and objects

For example, a triple in a knowledge graph might be (Alice, isMarriedTo, Bob), which corresponds to the predicate logic expression isMarriedTo(Alice, Bob).

Benefits of Using Predicate Logic

- Expressiveness: Predicate logic allows for the representation of complex relationships and rules within a knowledge graph.
- Reasoning: It supports logical inference, enabling the derivation of new knowledge from existing facts.
- Flexibility: Predicate logic can handle various types of relationships and entities, making it suitable for dynamic and evolving datasets

This logical inference demonstrates how knowledge graphs can use predicate logic to derive new information from existing data. In summary, a knowledge graph is a powerful tool for organizing and reasoning about data, leveraging the principles of predicate logic to represent and infer complex relationships between entities. This combination enhances the ability to integrate, analyze, and utilize data across various domains and applications.

- turning the object transportation robot into a recycling robot
- drone in the alps
- screws
- doors

3.8.2 Acquiring, Maintaining, and Linking Knowledge Graphs

3.8.3 Domain-specific Knowledge Graphs

3.8.4 Cutting any Fruit with any Tool for any Purpose

3.9 Scene Graphs as Virtual Knowledge Bases

3.10 Semantic Digital Twins

3.11 The Table Setting Scenario as a Logical Knowledge Base

- making knowledge action-able
  - motions
  - images
- opening the frige and KnowRob Q&A
- getting the milk out of the frige

3.12 Virtual Research Building and openEASE

- URDF
- Scene Graphs
- SOMA

3.13 Discussion of KR&R

Greek Letters with Pronunciations

Character	Name	Character	Name
α	alpha *AL-fuh*	ν	nu *NEW*
β	beta *BAY-tuh*	ξ, Ξ	xi *KSIGH*
γ, Γ	gamma *GAM-muh*	o	omicron *OM-uh-CRON*
δ, Δ	delta *DEL-tuh*	π, Π	pi *PIE*
ϵ	epsilon *EP-suh-lon*	ρ	rho *ROW*
ζ	zeta *ZAY-tuh*	σ, Σ	sigma *SIG-muh*
η	eta *AY-tuh*	τ	tau *TOW (as in cow)*
θ, Θ	theta *THAY-tuh*	υ, Υ	upsilon *OOP-suh-LON*
ι	iota *eye-OH-tuh*	ϕ, Φ	phi *FEE, or FI (as in hi)*
κ	kappa *KAP-uh*	χ	chi *KI (as in hi)*
λ, Λ	lambda *LAM-duh*	ψ, Ψ	psi *SIGH, or PSIGH*
μ	mu *MEW*	ω, Ω	omega *oh-MAY-guh*

Capitals shown are the ones that differ from Roman capitals.