COGAR 2nd Assignment

Mamoru Ota, Group E
Documentation: https://cogar-assignment-group-e.readthedocs.io/en/latest/

I. INTRODUCTION

AUTONOMOUS TIAGo robots deliver sushi orders in a restaurant environment. This report analyzes two core modules - **Perception System** and **Orchestration Manager** - in terms of the cognitive architecture concepts introduced in the course: perception, attention, action selection, and memory. The analysis focuses on how these cognitive functions are practically instantiated within a working robotic system.

II. PERCEPTION

The **Perception System** supplies task-relevant information to the robot by simulating sensory input, rather than modelling physical hardware such as RGB-D cameras or LiDAR sensors. Instead, it leverages probabilistic functions to emulate the uncertainty and noise inherent in real-world environments. These simulations yield Boolean indicators (e.g., "obstacle_detected," "plate_located") that serve as abstracted sensory outputs. This approach simplifies perception while preserving the informational structure needed to support decision-making and goal-directed behaviour.

From a cognitive perspective, this module implements perceptual abstraction - the transformation of continuous, noisy environmental data into discrete, symbolic representations. This is a foundational step in symbol grounding, whereby sensory input acquires semantic meaning within the architecture. Without such grounding, subsequent modules (e.g., reasoning or action selection) would be unable to meaningfully interpret sensory data or associate it with behavioural outcomes.

Moreover, the system exhibits task-modulated perception: it dynamically switches between different perceptual routines based on the robot's current operational phase (e.g., navigating to a table, grasping a dish). This reflects the principle that perception in intelligent agents should be context-sensitive rather than uniform. Such mechanisms are consistent with cognitive architectures like ACT-R and SOAR, in which perception is shaped by task goals and only relevant environmental features are processed at any given time.

This targeted allocation of perceptual resources enhances both computational efficiency and behavioural relevance. As emphasized by Kotseruba and Tsotsos in their review of 40 cognitive architectures [1], intelligent systems operating in real time must filter sensory information according to current goals to avoid being overwhelmed by irrelevant data. By prioritizing only task-relevant input, the **Perception System** enables the robot to maintain real-time responsiveness while conserving processing power.

III. ATTENTION

The architecture employs goal-directed attention primarily through the **Orchestration Manager**, which functions as the central hub for coordinating robot behavior in a dynamic environment. Rather than indiscriminately processing all incoming sensory data or messages, this component implements selective filtering mechanisms to prioritize only information that is immediately relevant to task execution. This approach mirrors top-down attentional control observed in human cognition, where internal goals and task context determine what stimuli are attended to.

One core example is its ability to ignore duplicate orders using an internal cache (self.last_order_message). This prevents redundant task execution and conserves computational resources, particularly in high-frequency message environments like ROS.

Another key attentional behavior involves conditional message processing. The **Orchestration Manager** continuously listens to multiple topics - /orders, /availability, and /position - but only takes action when a viable task-robot match can be made. Specifically, it checks that:

- there is at least one unassigned order in the queue,
- at least one robot is currently marked as "available," and
- positional data exists for those robots.

When these conditions are met, it computes a ranking of candidate robots based on their Euclidean distance to the serving area using a function like compute_distance. This ranking is used to prioritize the closest available robot, ensuring fast and efficient delivery. By dynamically adjusting attention based on both internal state (e.g., task queue) and external updates (e.g., robot location), the system demonstrates context-sensitive filtering that supports real-time responsiveness.

This form of conditional message gating aligns with the Selective Tuning Model [2], in which large streams of potentially distracting input are pruned early in the processing pipeline. Only data relevant to the current goals and situational constraints are allowed to influence action selection.

From a cognitive architecture standpoint, this dual-level attentional control (filtering inputs and ranking options) is essential for scalable and robust behavior. It minimizes cognitive load on the decision-making subsystem by ensuring that only semantically significant and contextually relevant information is processed further.

IV. Action Selection

Action selection in this architecture is handled by the **Orchestration Manager**, which acts as a centralized decision-making unit responsible for assigning tasks (i.e., sushi delivery orders) to available TIAGo robots. This module integrates symbolic rule-based logic with real-time situational awareness to ensure efficient and context-sensitive action selection.

The process begins with an internal check for eligibility: the system proceeds only if there is at least one unassigned order in the queue and at least one robot marked as "available" in the status dictionary. Once these preconditions are satisfied, the manager evaluates all feasible ⟨robot, order⟩ pairs using a simple but effective cost function. This function incorporates two primary components:

- Distance cost: The Euclidean distance between the robot's current location and the service area. This approximates travel time and minimizes energy usage and latency.
- Task urgency: Encoded implicitly through the position of the order in the priority-based queue. Orders with higher priority (e.g., cleaning tasks) appear earlier in the queue, reflecting greater urgency.

The system then selects the robot with the lowest total cost to fulfill the task. The decision is operationalized by publishing a task message to the robot via the /order_TIAGo ROS topic. Concurrently, the selected robot's status is updated to "occupied" in the system's internal availability dictionary, and the order is removed from the queue.

Currently, the system publishes error messages to the /error_messages topic when task failures occur (e.g., grasping or navigation issues). However, it does not yet include a mechanism for requeuing failed tasks or triggering a new assignment based on these error signals. As such, reactive replanning is not fully implemented in the current version, though the infrastructure (e.g., message channels and status tracking) could support its future integration.

From a cognitive architecture perspective, this approach reflects a hybrid of deliberative reasoning (utility-based task assignment) and reactive adaptation (on-the-fly error handling). It mirrors the conflict resolution phase in models like ACT-R and SOAR, where multiple production rules may match current conditions, and the system selects the one with the highest utility [3]. In this case, utility is modeled through spatial and temporal efficiency.

V. Memory

Memory in this architecture is primarily implemented through the **Orchestration Manager**, which plays a central role in fulfilling key short-term memory functions within the system.

Specifically, the **Orchestration Manager** operates as a form of working memory, by dynamically tracking the current goal tuple ⟨table, dish, robot⟩, along with relevant contextual information such as the grasping status of the robot and its availability. This volatile memory is continuously and automatically updated in response to real-time events and changes in the environment as the robot executes its assigned tasks. Such updates enable the system to make decisions that are sensitive to the current context, thereby supporting fluid and adaptive behavior.

In addition, the system utilizes incoming ROS message queues (e.g., /orders, /position) as a kind of sensory buffer. These message queues temporarily hold short-term perceptual data - such as newly received orders or robot position updates - which are then sampled and interpreted by higher-level modules for further processing.

This layered memory implementation can be loosely compared to the Atkinson–Shiffrin model of memory, in which short-term sensory traces are first registered and then passed into working memory, where they may be actively processed or retained depending on the task demands. When necessary, these traces are interpreted in the context of pre-encoded task rules, guiding the robot's behavior. As Sun (2004) emphasizes, the combination of symbolic representations and transient, temporary memory structures is essential for developing cognitive agents capable of reasoning flexibly and adapting effectively to the uncertainties of real-world environments [3].

VI. Conclusion

By focusing on two key modules, this architecture illustrates how core cognitive functions - perceptual abstraction, goal-directed attention, utility-based action selection, and multi-layered memory - can be realized in a simplified robotic system. The architecture demonstrates that even lightweight, modular designs can support real-time, adaptive behavior in dynamic environments, making them a viable basis for further development in both applied robotics and cognitive modeling.

References

[1] Kotseruba, Iuliia, and John K. Tsotsos. "40 years of cognitive architectures: core cognitive abilities and practical applications." Artificial Intelligence Review 53.1 (2020): 17-94.

[2] Tsotsos, John K., et al. "Modeling visual attention via selective tuning." Artificial intelligence 78.1-2 (1995): 507-545.

[3] Sun, Ron. "Desiderata for cognitive architectures." Philosophical psychology 17.3 (2004): 341-373.