Interoceptive Robotic Agents for Shared Control and Learning in Collaborative Construction Work

Xiaoshan Zhou[1], Carol C. Menassa[2], and Vineet R. Kamat[3]

[1]Ph.D. Candidate, Dept. of Civil and Environmental Engineering, Univ. of Michigan, Ann Arbor, MI 48109, USA.
[2]Professor, Dept. of Civil and Environmental Engineering, Univ. of Michigan, Ann Arbor, MI 48109, USA.
[3]Professor, Dept. of Civil and Environmental Engineering, Univ. of Michigan, Ann Arbor, MI 48109, USA.

xszhou@umich.edu, menassa@umich.edu vkamat@umich.edu

Abstract -

Advancing the adaptivity of autonomous mobile robots (AMRs), particularly empowering them to align with human expertise and learn from past experiences to infer optimal decisions in response to changing task demands and dynamic environment, is critical for their application in modular construction automation and reducing operational carbon footprints. Although a few learning paradigms based on human demonstrations have been introduced, robots remain rigid in structured tasks and lack internal adaptivity and authentic learning capabilities. This research explores a fundamental approach to equipping robots with an internal drive for self-reflection and conscious learning within a shared control setting, where human supervisors provide occasional corrective inputs and interventions. This approach is centered on the concept of interoception, which we factorize as "cognitive dissonance" within the robotic cognitive architecture. When significant decision divergence with human counterparts occurs, the robotic agent will actively seek decision rationale from semantic inputs, extracts knowledge from human scaffolding, encodes it into a memory system, and retrieves this "cached" knowledge for future decision-making in similar scenarios. This paper explores this new perspective to help build adaptive motion planning in AMRs. First, we propose integrating the legacy of heuristic costs from grid/graph-based path planning algorithms with a hypergraph model that caches declarative and procedural knowledge extracted from human semantic inputs. Second, we design a velocity-replay module that uses few-shot contrastive learning with a encoder-decoder architecture and compare its performance against reinforcement learning with human feedback and imitation learning. Two simulation demonstrations were conducted using the Issac Sim platform to showcase multi-robot synchronization and stacking collaboration. In addition, the proposed shared control paradigm was tested in a Gazebo environment in ROS, where human inputs were provided via relatively noisy brainwave-based control, compared to conventional peripheral devices. The results demonstrate how this approach better mitigates human input errors while facilitating autonomy for task completion compared to existing dominant shared control methods. The insights from this study resonate with recent advances in neuroscience and reinforcement learning, and pave the way toward artificial general intelligence in AMRs, fostering their progression from complexity to competence in construction automation.

Keywords -

Autonomous Mobile Robot; Shared Control; Motion Planning; Learning; Memory

1 Introduction

Automation in construction can significantly contribute to achieving sustainability goals [1]. In particular, modular construction using offsite prefabrication and onsite assembly have the potential to address climate action by reducing operational carbon footprints compared to traditional onsite methods such as concrete mixing and curing [2]. In additional to potential environmental benefits, automation can improve worker safety by mitigating exposure to hazardous conditions. For example, automated robotic machinery used for onsite tasks, such as assembly, reduces risks like falls from scaffolding, inhalation of welding fumes, and electrocution [3]. Mobile robots, including forklifts, cranes, and dozers, have been increasingly used to streamline offsite and onsite construction processes [4]. However, the integration of mobile robotics in construction introduces new challenges. Without effective coordination of multi-robot path planning that manages sequential task dependencies and precise velocity control to adapt to dynamic nature of construction workflows, risks such as struck-by hazards increase [5].

In addition, autonomous mobile robots (AMRs) reshape the power dynamics between humans and machines. These robots can operate with programmed autonomy, performing tasks such as delivering materials across a precast factory or a site to support fabrication or assembly processes [3]. Although they remain under the supervision of human workers, their autonomy shifts the relationship from subservience—where robots follow human instructions without intrinsic understanding—to augmentation, where robots act more like symbiotic co-workers with spe-

cialized capabilities. This shift, however, can provoke apprehension among industry practitioners, often fueled by fear of the unknown [6]. Concerns about the emergence of superintelligent systems, often referred to as singularity, suggest a future where robots could become uncontrollable and irreversible in their autonomy. Such fears are compounded by the opaque reasoning processes of robots and the asymmetry in knowledge between humans and machines, leading to uncertainty about whether these systems are truly beneficial and trustworthy.

To address these challenges, this research advocates for advancing the cognitive capabilities of AMRs to align with human intelligence. A critical focus is to learn from humans and learn the way how humans learn. In fact, many principles of robotics have been inspired by how humans perceive, reason, and act. For instance, robotic sensing mimics human sensory organs: cameras (RGB, depth, or infrared) parallel human vision, force sensors replicate touch, microphones simulate hearing, and joint encoders mirror proprioception. State-of-the-art robots have developed highly dexterous manipulation skills comparable to those of humans. However, a key distinction between humans and robots lies in adaptability, self-reflection, and learning. Human workers can consciously think, learn from past experiences, and modify their behaviors to adapt to changes. In contrast, robots are mechanical and purpose-built, lacking genuine situational awareness and the ability to abstract knowledge, let alone reuse it. More fundamentally, robotics research has primarily focused on replicating exteroception (sensing the external environment), while overlooking interoception (perceiving internal states). Interoception, however, plays a criticla role in human cognition, facilitating individuals to maintain homistasis, extrapolate mental schemas from observations, and make decisions based on hierarchical prioritization [7]. Inspired by this, a logical next step in robotics reseach should explore equipping mobile robots with interoceptive-like capabilities.

Interoception in mobile robots is a novel concept, and its conceptualization and implementation require a formal definition of the phenotypic characteristics—particularly the internal states—of mobile robots. Unlike human interoception, we are not suggesting that robots experience genuine emotions, pain, or self-consciousness. Instead, we define ”cognitive dissonance” as an abstract internal cognitive state variable robotic agents possess and factorize it based on the intensity of human inputs to refine robotic autonomy in shared control paradigms. This is proposed along with a new shared control architecture as illustrated in Figure 1: Rather than allowing human interventions to override robotic decisions, our proposed shared control integrates human inputs as a tributary that converges with autonomous decisions to produce a collaborative decision-making output forwarded to the robot's actuators. Simultaneously, this architecture inherently introduces a potential state of conflict between the robot's internal beliefs and its executed actions. Such conflicts create an internal drive for the robotic agent to reflect on its decisions, seeking insights from human feedback to identify potential biases or missed factors causing misalignment with human expectations. Through this reflective process, the robot engages in conscious-like learning and can gradually evolve into a meta-level understanding that reconciles discrepancies with its human counterparts.

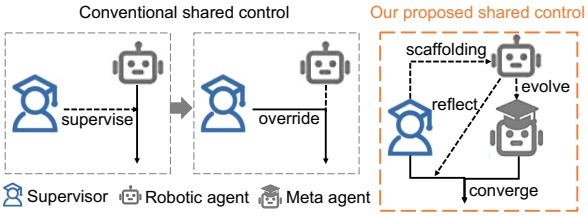

Figure 1. Shared control architectures

This framework is particularly designed for spatio-temporal-sensitive motion planning situations in multi-robot systems that navigate shared spaces and must dynamically adapt to evolving task sequences. For example, robots delivering panels must complete their task before façade installation can proceed. Precise velocity control is requrid thoughtout the process to avoid collisions and minimize downtime. To achieve this, the knowledge required by the robotic agents to make efficient motion planning decisions can be categorized into declarative knowledge and procedural knowledge. Declarative knowledge includes those facts (e.g., terrain features that were previously unavailable from the spatial grid map, traffic regulations with safety margins around obstacles, and path preferences provided by human supervisors) that inform future path-planning decisions. In our approach, this knowledge is acquired through human feedback that is prompted by an intuitive Graphical User Interface (GUI). The GUI facilitates the formalization of human knowledge while providing contextual dependency. This feedback is then encoded using natural language processing (NLP) techniques, including tokenization, tagging, and logical reasoning, and finally constructed as a knowledge graph (scaffolding). This design aligns with Lev Vygotsky's theory of the Zone of Proximal Development, which highlights bridging the gap gradually between what the learner (robotic agent) can do independently and what it can do with guidance from a more knowledgeable other [8].

Procedural knowledge pertains to how the mobile robot accurately replicates velocity profiles. Unlike dominant methods such as reinforcement learning (RL) and imitation learning—which start from a “blank slate”, rely on

random exploration in large state spaces, or require explicitly designed rewards (a challenging task)—our approach formalizes learning as a structured process of schema representation and modification. This process is regulated by goal-oriented behavior, avoiding the habituation tendencies of RL that often fail to adapt quickly to reward contingencies and thus sacrifice flexibility [9]. Our approach conceptualizes learning as a sequential production of components comprising a human-corrected velocity repertoire, implemented using an encoder-decoder architecture based on few-shot contrastive learning and Gated Recurrent Unit (GRU) to capture temporal dependencies. These designs lay the groundwork for factorizing the ontology of mobile robot motion planning and appeal to diverse optimization principles for routing decisions. An illustration of the robotic interoception concept, along with its relationship to modern topics in RL, is shown in Figure 2.

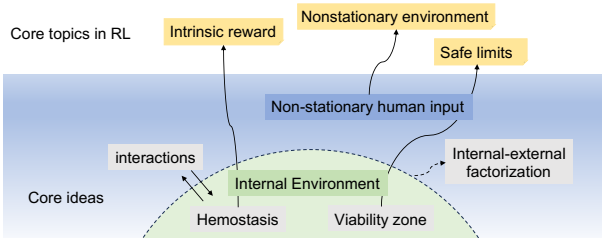

Figure 2. Overview of the interocepltive framework

In practice, the workflow of the proposed approach is illustrated in Figure 3. While acknowledging the multifaceted nature of interoceptive robotics, this paper narrows its focus to a detailed methodology for motion planning in Section 2, followed by three illustrative examples in Section 3.

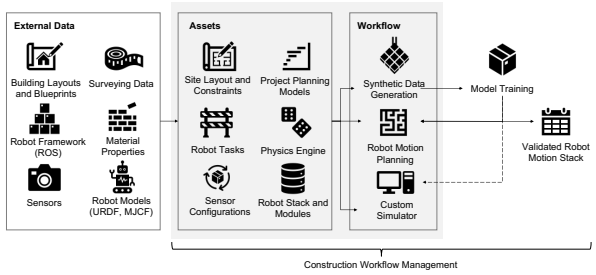

Figure 3. Simulation-enabled workflow for on-site robot motion planning

2 Methodology and Experimentation

The overarching pipeline of the designed methodology is shown in Figure 4. The motion planning consists of two components: path planning and velocity planning.

2.1 Path Planning

The path planning is defined to find a continuous path from a start location to a goal location throughout the subset of spatial configuration space consisting of traversable space. The space is created as a discretized approximation of connectivity using an occupancy grid, which can also be seen as a graph in which vertices are individual configurations and edges represent the transitions between pairs of configurations. Paths are found by searching the space using A* [10] (see Figure 5 for an abstracted formulation of the physical spatial environment and paths).

Given the designated paths, the tracking of the AMRs' positions is available (see Figure 5) with a hypothesized constant velocity. In practice, however, the human supervisors occasionally adjust the velocity to accommodate ground or floor conditions, task hierarchies, and interaction dynamics with other entities. This research proposes a shared control paradigm based on a Bayesian updating framework, leverageing robotic autonomy (scaled in absolute terms) as the prior and velocities from human agency (scaled as increments) as the likelihood. Using the integral of velocity curves (often approximated with sample-based approaches, e.g., Monte Carlo) at a pre-defined sampling rate along the designated path, the distances of the AMRs can be tracked in real time, with proximity thresholds set to trigger automated alerts.

In the proposed shared control paradigm, the factorized cognitive dissonance of AMRs' reasoning and action—as parameterized by the intensity of human inputs—is demonstrated using a contour plot (see Figure 6). This provides an intuitive representation for retrospective documentation of rationale behind human interventions. The texts inputs from human supervisors are processed with NLP open information extraction (OpenIE) focusing particularly on two attributes—terrain features and task sequences—as declarative knowledge mapped to the space vertices. Additional attributes are preserved as episodic knowledge, which is however beyond the scope of this article. The source code for this simulation is available at `https://github.com/XiaoshanZhou624/A-star-with-GUI-for-human-inputs`.

Task sequence attributes partially overlap with positional information from the spatial space but require an additional task space to define the availability state of the objects at the target workspace where robots offload or load and the occupancy state of critical pathways shared by robots approaching the workspace. We propose to use hypergraphs to model this new configuration space that requires additional complexity. A hypergraph is an extension of a graph where hyperedges (or hyperarcs) are not restricted to connecting only two vertices [11], thus allowing us to accommodate more dependence conditions. A graph-based model also facilitates to modify the heuristic

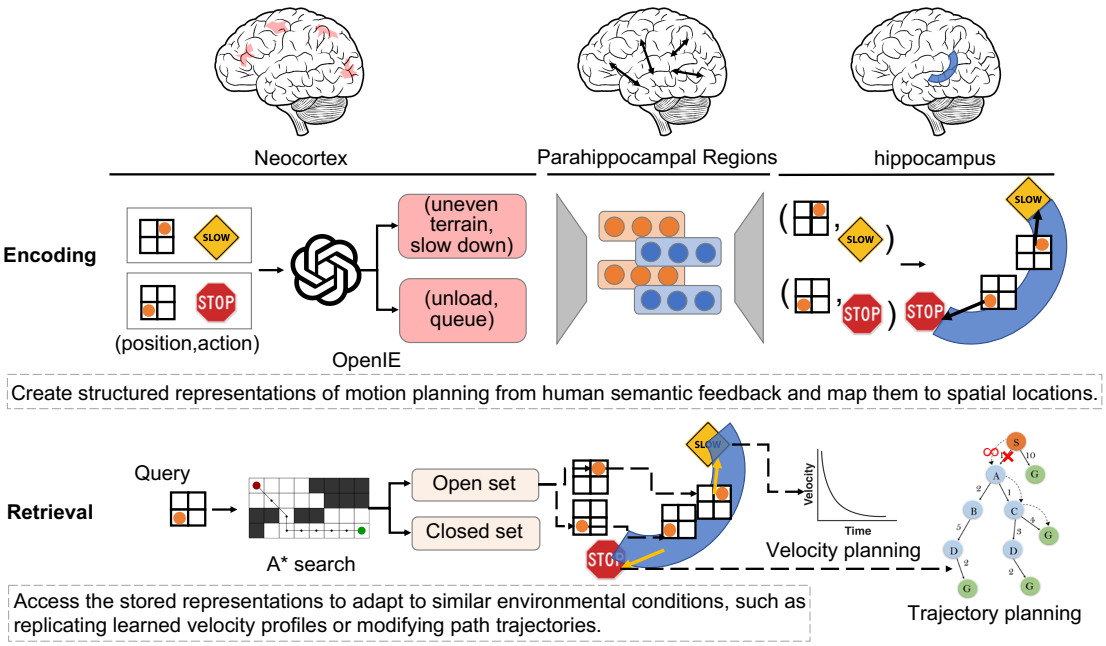

Figure 4. Methodology overview for knowledge encoding and retrieval in path and velocity planning

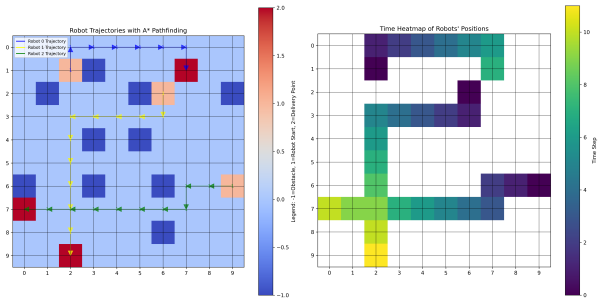

Figure 5. (left) Trajectories with A* pathfinding in a grid. (right) Heatmap of AMRs position tracking

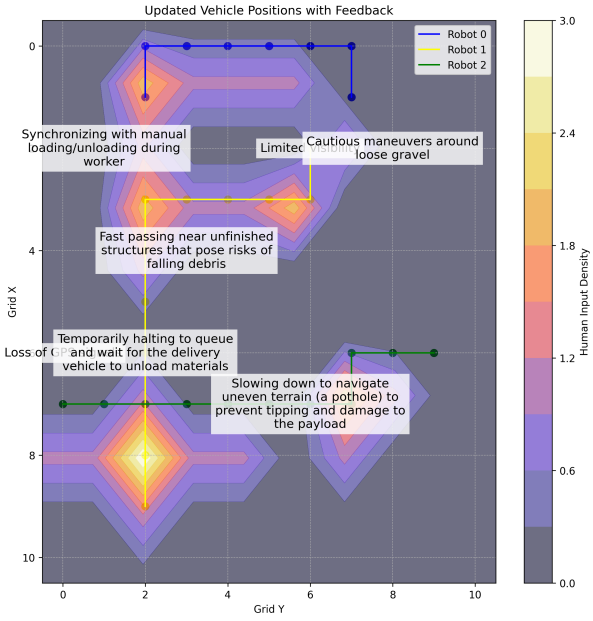

Figure 6. Contour plot of human intervention intensity and GUI for human semantic inputs in rationale

costs in A*, thus automatically influencing path planning in the algorithmic logic that determines which nodes are first expanded as leaf nodes and added to the closed set. The task space states define the preconditions for admissible moves to the explored nodes. A visual depiction of the hypergraph is shown in Figure 7, and the pseudocode for the modified A* algorithm is provided in Figure 8.

2.2 Velocity Planning

As the spatial configuration expands in the hypergraph model, terrain attributes can be coded to trigger the replay of human-refined velocity profiles. We propose an encoder-decoder network architecture, as shown in Figure 9. Given only one trial of data, we designed a sliding time window to segment the velocity profiles into smaller segments. Leveraging the similarity contrast principles of few-shot learning, we construct positive pairs from consecutive windows and negative pairs from random windows. A Gaussian filter is applied to smooth the data within each window. Subsequently, these pairs are processed through

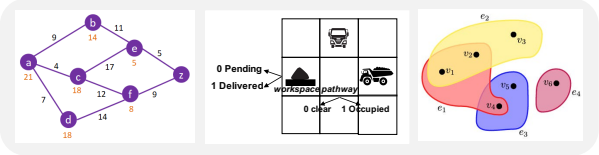

Figure 7. Hypergraph attribute visualization

Algorithm 1 Hypergraph A* with State-Dependent Costs

Input:
$H = (V, E)$: A hypergraph where:
- V: Set of nodes $v = (p, s)$, with p as position and s as binary state.
- E: Set of hyperedges $e \subseteq V$, where the cost of each node $v \in e$ depends on the s of other nodes in e, represented by $c(v, e, \text{states})$.

start: Starting node.
goal: Goal node.
Output: The optimal path from *start* to *goal*.
Procedure:

Initialize $open\_set \leftarrow$ PriorityQueue() ▷ Stores nodes to be explored, ordered by f_score
$g\_score[v], f\_score[v] \leftarrow \infty$ for all $v \in V$ ▷ Initialize cost scores
$g\_score[start] \leftarrow 0$ ▷ Cost to reach the start node is zero
$f\_score[start] \leftarrow$ `heuristic_cost`$(start, goal, H)$ ▷ Estimate total cost from start to goal
$open\_set.push(start, f\_score[start])$ ▷ Add start node to the open set
while $open\_set \neq \emptyset$ **do** ▷ Main loop: Process nodes until the open set is empty
  $current \leftarrow open\_set.pop()$ ▷ Retrieve node with lowest f_score
  if $current == goal$ **then**
    return `reconstruct_path`$(current)$ ▷ Return the reconstructed optimal path
    Break ▷ Terminate the loop as the goal is reached
  end if
  for $neighbor \in neighbors(current)$ **do** ▷ Explore each neighbor of the current node
    if $neighbor \notin e$ for all $e \in E$ **then** ▷ Check if neighbor is part of any hyperedge
      $h\_cost \leftarrow$ `heuristic_cost`$(neighbor, goal, \emptyset)$ ▷ No hyperedge states to consider
    else
      $connected\_states \leftarrow \{n.s \mid n \in e.connected\_nodes\}$ ▷ Retrieve nodes' states in hyperedge
      $h\_cost \leftarrow$ `heuristic_cost`$(neighbor, goal, connected\_states, e)$ ▷ Update heuristic
    end if
    $tentative\_g \leftarrow g\_score[current] +$ `edge_cost`$(current, neighbor)$
    $tentative\_f \leftarrow tentative\_g + h\_cost$ ▷ Compute total estimated cost
    if $tentative\_g < g\_score[neighbor]$ **then** ▷ Update path if better cost is found
      $g\_score[neighbor] \leftarrow tentative\_g$
      $f\_score[neighbor] \leftarrow tentative\_f$
      $open\_set.push(neighbor, f\_score[neighbor])$ ▷ Re-add neighbor to the open set
    end if
  end for
end while

Figure 8. Pseudocode for hypergraph A* with state-dependent costs

a Siamese Network [12] with GRU modules to extract embeddings for each window, using the Euclidean distance as the contrastive loss for network training.

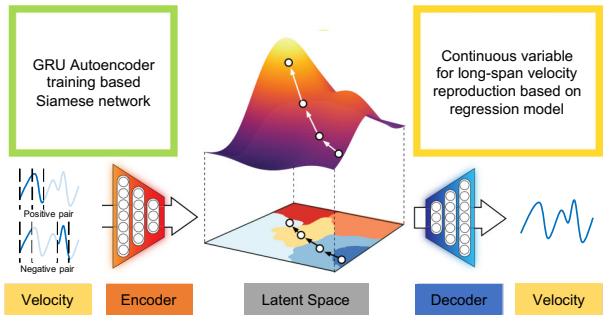

Figure 9. Technical overview of the encoder-decoder architecture for velocity replay

During retrieval, a regression model is designed and trained using Mean Squared Error as the loss function to map the embeddings back to velocity values (see Figure 13 for a demonstration). These values can then be directly broadcasted and subscribed to by robotic servo motor nodes in the Robot Operating System (ROS) environment. Detailed algorithm designs are available in the open-source code repository at `https://github.com/XiaoshanZhou624/velocity-replay`.

3 Demonstration in Simulation

3.1 Multi-Robot Coordination for Pick-and-Place

This section provides two examples of multi-robot coordination in motion planning and adaption to workpiece properties. The scenes and simulations were set up in Nvidia Omniverse Issac Sim (version 4.1.0).

As shown in Figure 10, two Jetbots navigate the workspace using a differential drive controller with physically accurate articulated joints. The task sequence dependencies are: after the first Jetbot delivers material to the workspace, it must reverse to clear the pathway. Simultaneously, the second Jetbot must wait and avoid entering the workspace prematurely to prevent blocking the pathway. A cube was added to the scene as an abstracted form of the material. To ensure realistic interactions, physical properties such as force and rigidity were applied to enable the first Jetbot to push the material toward the Franka Panda manipulator for handover. A Franka Panda manipulator was positioned in the workspace to pick up the object and rotate to hand it over to the second Jetbot. The hypergraph vertices' preconditions were incorporated into the program logic by parameterizing tasks or programming states to transition between subtasks, such as the Franka's handover or allowing the Jetbot to resume movement instead of being blocked by non-steppable areas influenced by A* heuristic costs. A video of this simulation is available at `https://youtu.be/RwKJ0zggWfE`. In real-world applications, precise localization is often achieved using land-marked AprilTags for accurate positioning of robots and manipulators. As demostrated in the video, when the goal location of the transport robot overlaps with the handover location of the manipulator's end-effector, collisions can occur. This highlights the need for research on precise spatial alignment, timing calculations, and accurate velocity control to synchronize robotic motion. These considerations are particularly crucial in large automated machinery, such as forklifts, or when dropping large-scale material panels that require significant space for maneuvering, reorientation, or flipping before placement.

Although block stacking tasks have been widely studied in robotic manipulation and have demonstrate high dexterity, real-world placement present greater complexity. Unlike uniform blocks with consistent shapes, sizes, and weights that can be easily aligned in horizontal or vertical stacks, construction materials exhibit considerable variability. For instance, pieces of lumber often differ in di-

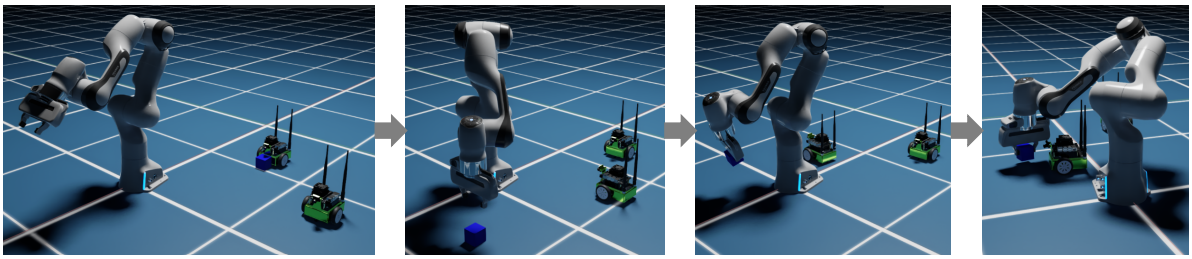

Figure 10. Robot motion planning for material delivery and handover coordination

mensions, while palletized materials may contain different items, such as tiles and shingles, each varying in weight. Moreover, some materials may need repositioning during unloading to facilitate subsequent tasks. For example, glass panels may require flipping to expose the correct side for façade installation. Proper planning must account for balanced weight distribution during transport to mitigate the risk of tipping caused by uneven loads affecting inertia and centrifugal forces, particularly for wheeled robots. An example is provided in Figure 11. This demonstration was adapted from a UR 10 Bin Stacking application containing a conveyor belt, a pallet where the bins should be stacked, and a UR 10 robot with a suction gripper. Bins arriving right-side up were flipped at a designated station to ensure proper stacking upside down. The placement alternated between sides of the pallet to balance weights. A supplementary video demonstrating the robot's motion planning is available at `https://youtu.be/yHbuoG7d05A`. Future work will integrate sensors for weight tracking and incorporate this data into the the manipulator's motion planning to maintain load balance while minimizing travel distance. In addition, sim-to-real discrepancies will be analyzed by examining how stiffness and damping parameters in the simulation setup correlate with the accuracy of joint properties in URDF robotic models and real-world variations in loads and terrain friction.

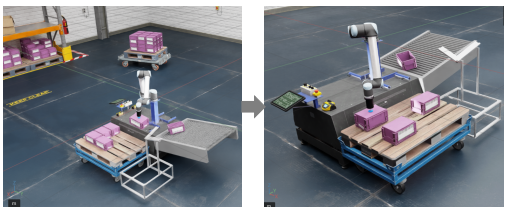

Figure 11. UR10 bin stacking and motion planning with weight-balanced placement

3.2 Velocity Replay

The velocity embeddings obtained after training are visualized in Figure 12, with dimensionality reduced using t-Distributed Stochastic Neighbor Embedding (t-SNE). These embeddings are stored as compressed vectors of the original velocity profiles. During retrieval, they are mapped back to velocity, as shown in Figure 13, where regression models reconstruct multiple possible original profiles.

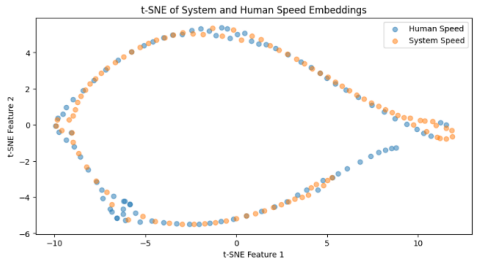

Figure 12. t-SNE visualization of velocity embeddings after contrastive Siamese Network learning

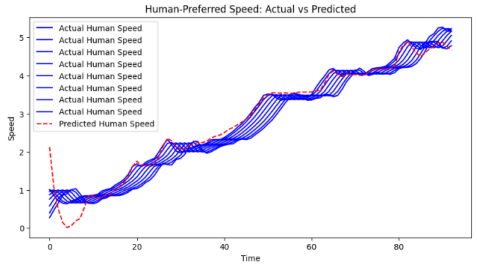

Figure 13. Illustration of mapping embeddings back to velocity using the regression model

The core objective here is to contrast our proposed method with currently dominant approaches—Reinforcement Learning with Human Feedback (RLHF) and Imitation Learning (IL). The learning mechanisms of the three methods and their performance comparisons are shown in Figure 14. Mostly importantly, IL functions similarly to behavioral cloning, allowing for a near-perfect replay of velocity profiles with high learning efficiency. However, as a supervised learning method, IL does not involve reasoning or prediction beyond what it has seen during

end-to-end training. Once the environment changes (e.g., unexpected obstacles, disruptions, or external forces), IL cannot adapt. RLHF uses human input to shape the reward function, guiding the robotic agent's actions. With extensive training episodes, the agent is likely to develop a well-represented internal world model that informs its decisions. However, the training process can be highly time-consuming. Given the semi-structured nature of construction sites, robots are expected to follow designated paths while remaining aware of unexpected proximity intrusions. In our approach, a dedicated module for velocity embeddings enables compact storage and efficient replay. This approach offers greater flexibility and better aligns with the ontology of human learning.

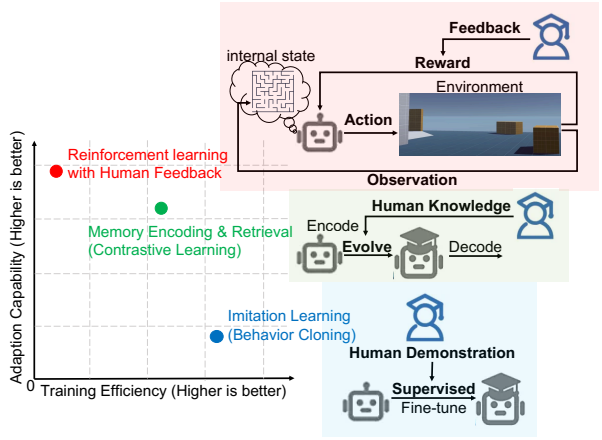

Figure 14. Comparison of RLHF, IL, and our contrastive learning approach using an encoder-decoder model

3.3 Shared Control Comparisons with EEG-Based Brainwave Input

We present a case study that simulates the possibility of incorrect human inputs in shared control. These inaccuracies can arise from two factors: (1) humans making incorrect decisions and (2) errors in detecting their intentions, such as missed detections or false alarms. In this simulation, we introduce potential detection errors by interpreting human inputs from brain signals measured via scalp electroencephalography (EEG). We trained a Bayesian inference framework to interpret EEG signals related to velocity adjustment intentions (acceleration or deceleration). Details on the EEG signal methodology can be found in the authors' other publications. Although the methodological framework accounts for uncertainty by using probabilistic reasoning, inherent noise in the signal spectrum and human inability to maintain consistent control can lead to fluctuations and inaccuracies in detected inputs.

Figure 15. Simulation platforms: (left) Gazebo in third-person perspective; (right) Unity in first-person perspective

To evaluate our approach, we tested in two scenarios (as shown in Figure 15). First, we set up a TurtleBot in Gazebo to follow another TurtleBot moving at a predefined velocity profile. Human users can intervene the following Turtlebot through three shared control paradigms: overriding, weighted pooling, and the proposed Bayesian updating method. EEG signal was streamed via WebSocket from Emotiv, processed into velocity commands in real time, and broadcast using the `cmd_vel` topic in ROS, which was subscribed to by the following TurtleBot. We measured how shared control can reconcile fluctuations in human inputs while achieving a balanced integration with robotic autonomy. As evidenced in Figure 16, our Bayesian updating method can smooth fluctuation in human inputs, always preserve the dual-agent decision-making rationale, and allows the robotic agent to follow preferred velocity patterns without requiring continuous human intervention.

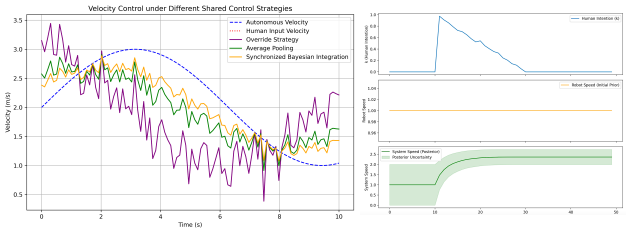

Figure 16. Shared control performance comparisons

Second, we tested how well a robot in Unity resumes motion after stopping due to autonomous obstacle avoidance detected by LiDAR monitoring. Although collisions were prevented, the key question was whether and how the robot could navigate past a narrow passage threshold and continue moving without getting stuck. In Unity, autonomous driving was enabled using the NavMeshAgent component, and the EEG data stream was subscribed to via the TCP protocol. Task completion times were measured across multiple trials for the three shared control modes and are presented in Table 1. In overriding mode, the human required significantly more time to complete the maneuver. In weighted (average) pooling mode, the hu-

man had to exert strong mental effort to insert control and influence the robot's motion. However, in our proposed Bayesian updating approach, a small human input acted as a minor disturbance but could help the robot escape a local deadlock. After a brief intervention, the human can hand over control to the robot to resume autonomous mode and complete the task efficiently.

Table 1. Task Completion Time

Shared Control Mode	Task Completion Time (s)
Overriding	> 30
Weighted Pooling	10–25
Bayesian Updating	< 10

4 Conclusion

Mobile robots are envisioned as learning agents within a shared control paradigm, where we define an interoceptive cognitive architecture that enables them to develop an internal drive for self-reflection based on feedback from collaborative decision-making human counterparts. The knowledge acquired through this shared control process is abstracted, structured, and encoded, allowing for more flexible retrieval and faster adaptation in future similar situations compared to conventional RLHF and IL frameworks. The proposed shared control framework also helps counterbalance potential errors in human inputs by leveraging dual-agent intelligence and empowers robotic autonomy to automatically follow human-preferred velocity patterns compared to coventional overriding and weighted pooling shared control. This proposed approach is expected to advance robotized construction automation by shfiting from rigid, pre-programmed behaviors to greater autonomy and adaptive intelligence, enabling robots to coordinate better with human coworkers and multi-robot systems. Furthermore, this research creates opportunities to handle more heterogeneous and complex tasks within the modular construction vision, thus contributing to safer and more sustainable, decarbonized construction practices.

5 Acknowledgement

The authors would like to acknowledge the financial support received for this research from the US National Science Foundation (NSF) Grant SCC-IRG 2124857. Any opinions in this paper are those of the authors and do not necessarily represent those of the NSF.

References

[1] Z. Zheng, F. Wang, G. Gong, H. Yang, and D. Han. Intelligent technologies for construction machinery using data-driven methods. *Automation in Construction*, 147:104711, 2023.

[2] S. K. Yevu, E. K. Owusu, A. P. C. Chan, S. M. E. Sepasgozar, and V. R. Kamat. Digital twin-enabled prefabrication supply chain for smart construction and carbon emissions evaluation in building projects. *Journal of Building Engineering*, 78:107598, 2023.

[3] C. Liu, J. Wu, X. Jiang, Y. Gu, L. Xie, and Z. Huang. Automatic assembly of prefabricated components based on vision-guided robot. *Automation in Construction*, 162:105385, 2024.

[4] K. You, L. Ding, C. Zhou, Q. Dou, X. Wang, and B. Hu. 5g-based earthwork monitoring system for an unmanned bulldozer. *Automation in Construction*, 131:103891, 2021.

[5] D. Kim, S. Lee, and V. R. Kamat. Proximity prediction of mobile objects to prevent contact-driven accidents in co-robotic construction. *Journal of Computing in Civil Engineering*, 34(4):04020022, 2020.

[6] C.-J. Liang, T.-H. Le, Y. Ham, B. R. K. Mantha, M. H. Cheng, and J. J. Lin. Ethics of artificial intelligence and robotics in the architecture, engineering, and construction industry. *Automation in Construction*, 162:105369, 2024.

[7] Giovanni Pezzulo, Francesco Rigoli, and Karl Friston. Active inference, homeostatic regulation and adaptive behavioural control. *Progress in Neurobiology*, 134:17–35, 2015. doi:10.1016/j.pneurobio.2015.09.001.

[8] B. Eun. The zone of proximal development as an overarching concept: A framework for synthesizing vygotsky's theories. *Educational Philosophy and Theory*, 51(1):18–30, 2019.

[9] K. Khetarpal, M. Riemer, I. Rish, and D. Precup. Towards continual reinforcement learning: A review and perspectives. *Journal of Artificial Intelligence Research*, 75:1401–1476, 2022.

[10] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[11] C. Berge. *Hypergraphs: Combinatorics of Finite Sets*. Elsevier Science, 1984.

[12] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah. Signature verification using a "siamese" time delay neural network. In *Proceedings of the 7th International Conference on Neural Information Processing Systems*, pages 737–744, Denver, Colorado, 1993.