General instructions on using this template and submitting a manuscript to *Science Robotics*

Use this template to speed the processing of your paper and the completion of the manuscript's record in our system. Our goal is to automatically identify and extract title, authors, abstract, and other component parts of your paper and to enrich it with reference links and an accurate layout.

If you are using LaTeX, please convert your paper into a Word docx. If this is not possible, please use our LaTeX template and guidelines (available here) and upload a PDF version of your paper. Some conversion approaches are available here: http://www.tug.org/utilities/texconv/textopc.html

Please use the actual template starting on the next page, which includes more specific formatting instructions.

You can submit your paper at http://cts.sciencemag.org/. Additional instructions are available at https://www.science.org/content/page/scirobotics-instructions-research-articles

So that we can extract parts of your paper (even if you do not use this template), begin each section with the specific words listed below, some of which are followed by a colon. Do not use paragraph breaks in the title, author list, or abstract. The author list, corresponding author email(s), and affiliation(s) should be checked carefully because they will be published as listed in the manuscript.

Title: No more than 100 characters and spaces, lacking jargon, punctuation (apart from commas) and abbreviations where possible.
Authors:
Affiliations:
Abstract: 250 words or less.
One Sentence Summary: No more than 125 characters and spaces.
Main Text:
References and Notes: Only a single numbered list should be provided for all references cited in the main text and in the supplementary materials. Please include the title of the article in the reference.
Acknowledgments: Split into general, Funding, Author contributions, Competing interests, and Data and materials availability, as described in the template below.
Supplementary Materials: Include a list, noting which references are only cited in the SM.
Fig. #. (Begin each figure caption with a label, "**Fig. 1.**", for example, as a new paragraph.)
Table #. (Begin each table caption with a label "**Table 1.**", for example, as a new paragraph.)

Please use the .docx format (all versions after Word 2007 for PC and Word 2011 for Mac) and include page numbers in your submitted file. We also encourage use of line numbers. Supplementary Materials (comprising Materials and Methods, figures, and tables) should be in a separate file.

When you are ready to submit, please delete this page with all its contents.

Title: Self-adaptive embodied neuromorphic intelligence using in-memory computing hardware

Authors:

Zai-zheng Yang[1]†, Hang Zhao[1]†, Yichen Zhao[1], Xing-Jian Yangdong[1], Yuekun Yang[1], Chen Pan[2], Cong Wang[1]*, Shi-Jun Liang[1]*, Feng Miao[1]*

Affiliations:

[1] Institute of Brain-inspired Intelligence, National Laboratory of Solid State Microstructures, School of Physics, Collaborative Innovation Center of Advanced Microstructures, Nanjing University, Nanjing, China.
[2] Institute of Interdisciplinary Physical Sciences, School of Science, Nanjing University of Science and Technology, Nanjing, China.
[3]For large groups, use the name of the group or consortium and include a full list of the authors and affiliations at the end of the main manuscript or in the Supplementary Materials.

*Corresponding author. Email: cong@nju.edu.cn(C.W.); sjliang@nju.edu.cn(S.J.L.); miao@nju.edu.cn(F.M.)
†These authors contributed equally to this work.

Abstract: Embodied artificial intelligence (AI) holds promise as a crucial step toward achieving artificial general intelligence, enabling intelligent agents to perceive, interact with, and learn from the information in the physical world. However, current digital computing paradigm struggles with the von Neumann bottleneck, particularly when processing the massive influx of analogue sensory data, resulting in significant energy consumption and latency. To address these challenges, we propose an approach using analogue in-memory computing (AIMC), which processes analogue sensory signals directly within memory. This method significantly reduces the data conversion burden and enables parallel neuromorphic computing. In this work, we present an embodied in-memory neuromorphic computing system consisting of perception, planning, memory, and motion modules. Our system demonstrates a dramatic reduction in perception latency (from milliseconds to microseconds) compared with traditional digital schemes. Through in-situ learning in the analogue domain, our system can autonomously adapt to environmental noise and hardware variations. The processed environmental information is stored in the AIMC hardware, enabling the agent to make informed decisions and accomplish specified task in complex and dynamic real-world scenarios. This work highlights the potential of AIMC to achieve high-speed and energy-efficient embodied intelligence with real-time interaction and decision-making capabilities in dynamic environments.

One-Sentence Summary: Provide a one-sentence summary (containing no more than 125 characters and spaces) capturing the most important point of the paper.

Main Text:

INTRODUCTION

Embodied artificial intelligence, which is equipped with physical form and capable of learning through real-world interactions, is considered a key step toward achieving artificial general intelligence. Embodied intelligent agents are capable of perceiving, memorizing, planning, and interacting with the physical world. These agents perform intelligent tasks based on multimodal sensory signals collected from the environment, including visual, auditory, and tactile signals. However, the environmental information is produced in the analogue domain, while the traditional processors operate in a digital manner. This separation gives rise to frequent conversions between the analogue sensory signals and the digital data, resulting in significant delays in perception-motor responses and high energy consumption. Moreover, when von Neumann processors process the collected information, frequent data transfers between memory and processing units worsen the latency and energy efficiency. These issues will be exacerbated with the increase in sensor data volume and algorithm complexity, causing significant challenges in the implementation of embodied intelligence where ultralow response latency and energy consumption are required.

Analogue in-memory computing (AIMC) technology emerges as a promising solution to address these challenges. The sensor signals can be processed by AIMC hardware directly within the analogue domain, significantly alleviating the data conversion burdens between the analogue and digital domain. Moreover, AIMC enables parallel neuromorphic computing within the memory, overcoming the memory wall of von Neumann architecture. Hence, AIMC possesses immense potential for implementing embodied intelligence with rapid response and high energy efficiency.

In this study, we demonstrate an AIMC-based embodied intelligent agent, which consists of AIMC-based perception, planning, and memory modules, as well as a motor system. We show that the AIMC-based perception module achieves a microsecond-level recognition latency due to the elimination of analogue-to-digital conversions of sensor signals, while the digital scheme undergoes a millisecond-level recognition latency. Moreover, through in-situ training within the analogue domain, the AIMC-based perception module is capable of adapting to the complex physical world, recovering from the performance degradation caused by environmental noise and hardware variations (*e.g.,* sensor-to-sensor variations). We also show that the collected environmental information can be stored within the AIMC hardware, and then used for the decision-making process of the AIMC-based planning module. Through the cooperation of the AIMC-based functional modules and the motor system, the AIMC-based embodied agent is capable of exploring, planning, and finally achieving specified objectives through interacting with complex and varying environments.

RESULTS

Embodied neuromorphic computing based on AIMC hardware

Humans perceive their surrounding environment, store memories, plan actions, and control motions with remarkably low energy consumption, due to the energy-efficient computational capabilities of their cerebrums. The cerebrum (Fig. 1A) consists of the frontal lobe, occipital lobe, parietal lobe, and temporal lobe. The frontal lobe is in charge of planning and motion control, while other lobes process sensory signals of various modalities, including vision, touch, and hearing. Additionally, the hippocampus, located within the temporal lobe, functions as the center for memory processing. These regions are constituted by neurons interconnected through synapses, whose synaptic structures govern both the information storage and computing. AIMC hardware (ReRAM, PCM, Flash memory, *etc.*) effectively replicates synaptic functions by offering programmable physical properties (*e.g.,* conductance, electric charge quantity) and supporting in-situ computing by physical principles (*e.g.,* Ohm's aw and Kirchhoff's law). AIMC crossbar arrays are able to mimic the dense connections between neuron layers, enabling parallel and synchronous neuromorphic computing (Fig. 1B). Therefore, AIMC hardware holds the potential to achieve embodied intelligence by imitating the cerebrum's functional modules, including perception, long-term memory, and planning.

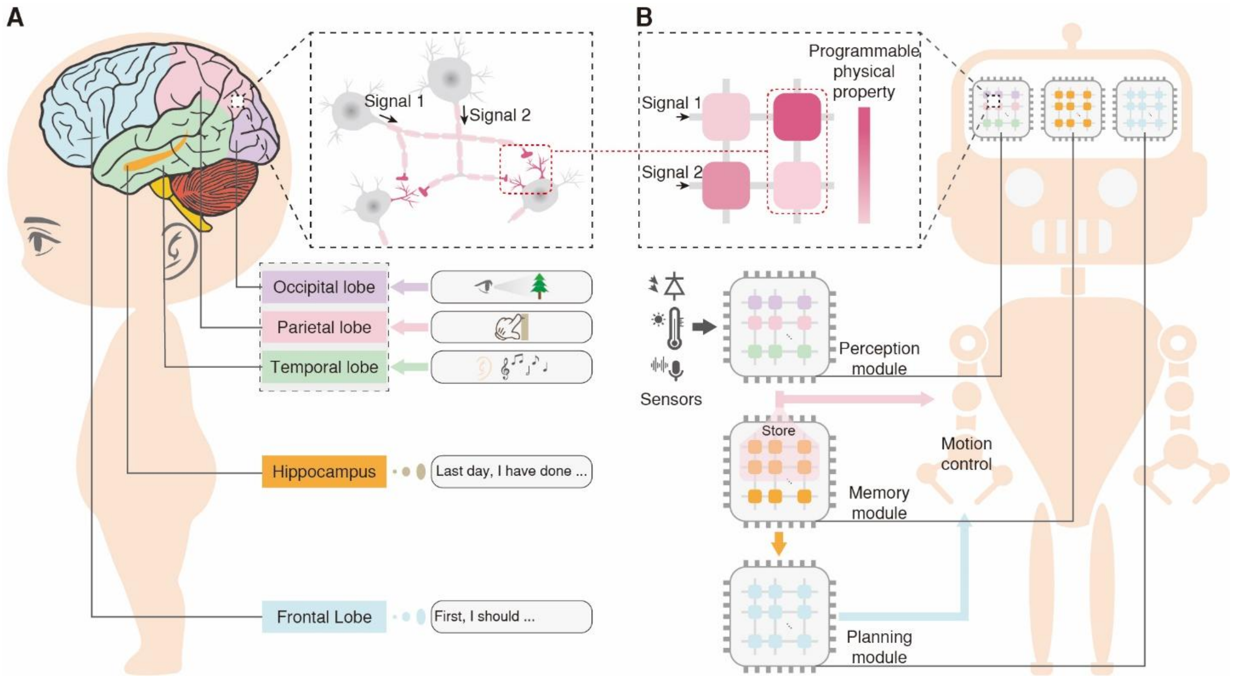
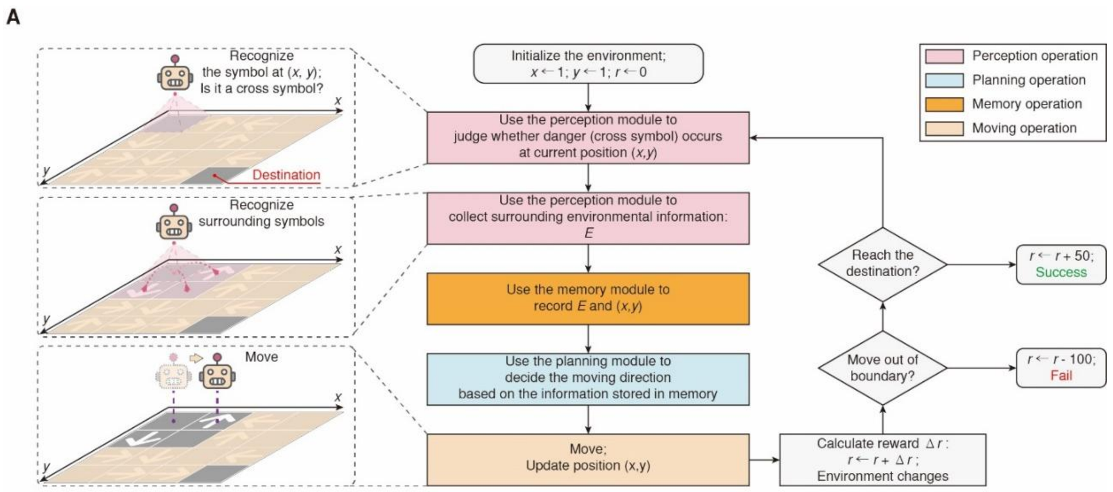


Fig. 1. Embodied neuromorphic computing based on AIMC hardware. (**A**) The functional regions of human cerebrum. The frontal lobe governs planning and decision-making; the hippocampus is in charge of organizing, storing and retrieving memories; the temporal lobe, parietal lobe, and occipital lobe process the auditory signals, tactile signals, and visual signals, respectively. These regions are constituted by neurons and synapses. (**B**) The schematic diagram of an AIMC-based embodied neuromorphic computing system. The synapses connecting two neuron layers can be implemented by an AIMC array, which leverages physical laws to parallelly process signals within the analogue domain. Thus, AIMC hardware can be used to imitate the human cerebrum's functions, including perception, long-term memory, and planning.

We design a toy environment for evaluating the performance of embodied neuromorphic intelligence. The environment is a 4×4 grid map where each grid cell contains a symbol (Fig. 2A), and the agent is assigned to move from the initial position (1, 1) to the destination (4, 4). At the arrival in each grid cell, the agent firstly recognizes the symbol at current position by its sensors and perception module. Once a cross symbol appears, the agent is supposed to escape from current position, as the occurrence of cross symbols represents "danger". Otherwise, the agent will be punished for its failure in recognition of dangers. After confirming the safety of current position, the agent explores its surroundings to collect environmental information. Specifically, the symbols around will be sequentially displayed at the agent's current position for recognition, and the recognition results are used to update the memory module. Afterwards, the agent uses its planning module to decide the moving direction based on the collected environmental information. As the agent moves, it receives a reward signal (Δr) from the environment (Detailed reward designs are explained in Methods). The agent needs to arrive at the destination while maximizing the accumulated reward, which requires the seamless cooperations of its functional modules.

We built an AIMC-based embodied intelligent agent (Fig. 2B), which comprises a sensor module, a motor system, and AIMC-based modules for perception, long-term memory, and planning. The agent uses these modules to accomplish the aforementioned task through interacting with the environment, which is a grid map displayed on a screen. The environmental information (*i.e.,* the symbol displayed on the screen) is sensed by a series of photodiodes, whose produced photocurrents are used as the inputs to the AIMC-based perception module. The processed perception results are then stored in the memory module, whose readout signals are used by the planning module for decision making (*i.e.,* deciding the moving direction in which the motor system moves the sensor module). As the moving finishes, the environment changes and the agent proceeds with the next perceiving-planning-moving cycle until successfully reaching the destination or failure.

A


Recognize the symbol at (x, y); Is it a cross symbol?
Destination
Recognize surrounding symbols
Move
Initialize the environment; x ← 1; y ← 1; r ← 0
Use the perception module to judge whether danger (cross symbol) occurs at current position (x,y)
Use the perception module to collect surrounding environmental information: E
Use the memory module to record E and (x,y)
Use the planning module to decide the moving direction based on the information stored in memory
Move; Update position (x,y)
Calculate reward Δr: r ← r + Δr; Environment changes
Move out of boundary?
r ← r - 100; Fail
Reach the destination?
r ← r + 50; Success
Perception operation
Planning operation
Memory operation
Moving operation

B

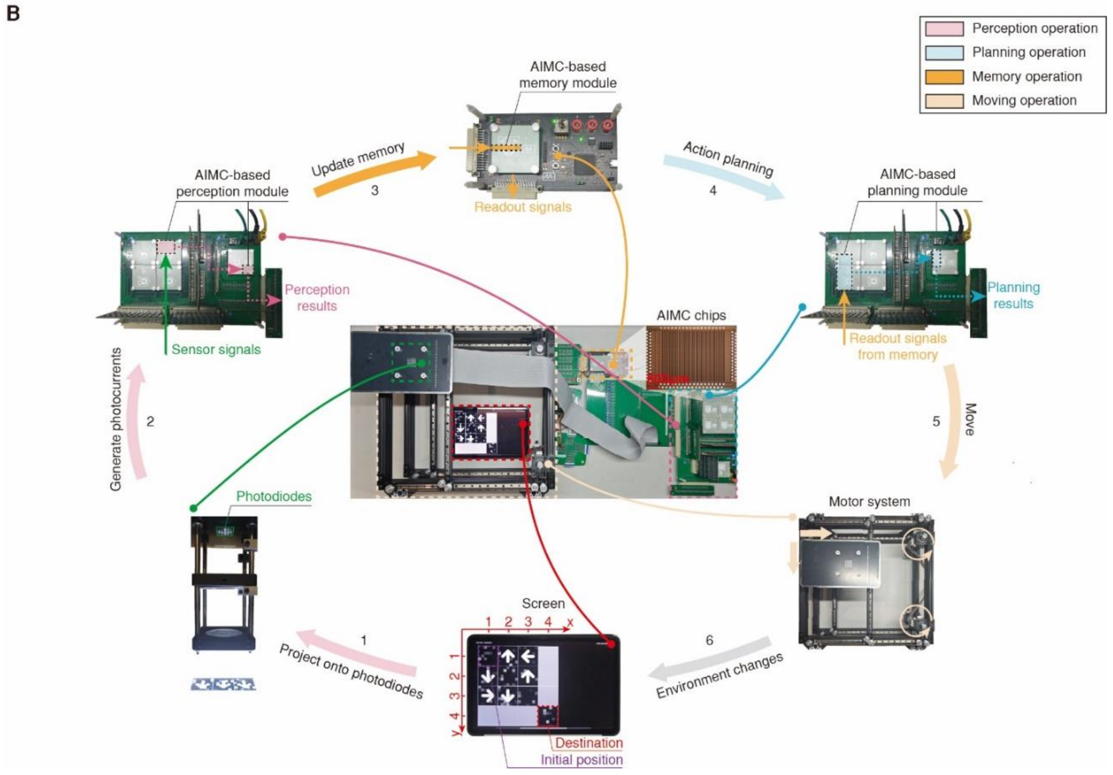


Fig. 2. The AIMC-based embodied intelligent agent and the toy environment. (**A**) The toy environment for evaluating the performance of embodied neuromorphic intelligence. (**B**) The assembled AIMC-based embodied intelligent agent and its interactions with the environment.

Self-adaptive AIMC-based perception module

We demonstrate that the AIMC-based perception module can adapt to varying and noisy environment. Figure 3A depicts the schematic diagram of the sensor module and the AIMC-based perception neural network. A symbol pattern displayed on the screen is projected through optical lenses onto a 5×5 photodiode array, whose produced photocurrents are amplified and then serve as the visual sensory signals of the agent. These signals are used as the inputs to a two-layer perception neural network, whose outputs represent the recognition results. However, the sensor signals produced in real world generally deviates from the simulations (Examples are presented in Supplementary Fig. S2), due to the environmental noise (*e.g.,* the noise pixels displayed on the screen) and hardware nonidealities (*e.g.,* sensor-to-sensor variations). As a result, the perception neural network might encounter performance degradations. We adopted an in-situ training approach to recover the performance of the perception module. This approach relies on a two-phase forward propagation process, where the gradients of parameters are calculated through perturbing the hardware neural network and measuring the effects on outputs. According to the obtained gradients, the neural network parameters thereby can be tuned to adapt to the sensor signals (More details are provided in Methods). The recognition accuracy recovers after the in-situ training (Figs. 3B and 3C), indicating that the perception module has adapted to the physical world.

We show that the AIMC-based perception module enables ultralow recognition latency, which is significant for the rapid reaction of embodied intelligent agent to environmental signals. Since the sensor signals are directly processed within the analogue domain by the perception neural network, they need not be converted to the digital domain. Furthermore, the programmed AIMC arrays within the perception module are able to parallelly perform the vector-matrix multiplication (VMM) operation on input signals. This one-shot VMM operation manner leads to a low processing latency of microsecond (μs) levels (Fig. 3D). By contrast, the digital scheme exhibits a high latency of 990 μs (Fig. 3E), including 979 μs for digitizing the 25-channel sensor signals and 11 μs for performing the VMM operations on the sensor data. Notably, AIMC technology has also been demonstrated to enable high-speed and reliable motor control. Therefore, AIMC has the potential to implement embodied intelligence with fast reaction speed and high energy efficiency.

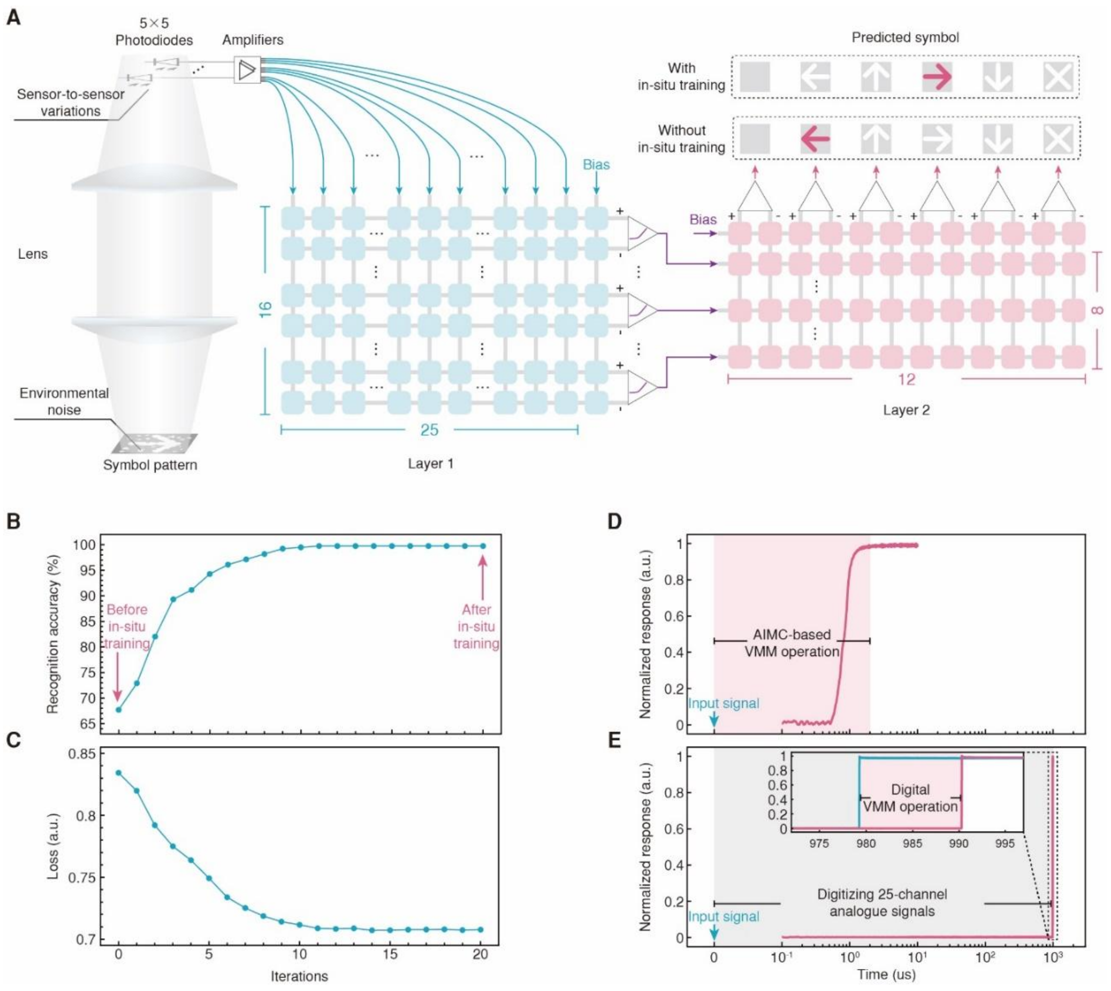


Fig. 3. Self-adaptive AIMC-based perception module. (**A**) The schematic diagram of the AIMC-based perception module. The symbol pattern displayed on the screen is projected onto 5×5 photodiodes. The photocurrents produced by the photodiodes are amplified and used as the inputs to an AIMC-based perception neural network. Each neural network parameter is represented by the difference of the weights stored within a pair of AIMC cells. The outputs of the perception neural network correspond to 6 symbol categories. Owing to the environmental noise and sensor-to-sensor variations, the perception module might fail in symbol recognition. (**B**) The evaluated recognition performance during in-situ training. After in-situ training, the perception neural network recovers from performance degradation. The recognition accuracy is evaluated on 300 random symbols. (**C**) The loss curve of in-situ training. (**D**) The AIMC-based perception module directly performs VMM operation on the analogue sensor signals with a processing latency of 2 µs. (**E**) By contrast, the digital scheme consumes 979 µs and 11 µs for digitizing the 25-channel analogue signals and performing VMM operation on the digitized data, respectively.

Implementations of long-term memory and AIMC-based planning modules

AIMC hardware can also be used to implement the cerebrum's functions of long-term memory and planning, which are critical for the embodied intelligent agent to make correct decisions. Figure 4A shows an example of perceiving-planning-moving cycle. At each step of the journey to the destination, the agent is able to explore the symbols surrounding its current position, and other unvisited areas (the shade areas in Fig. 4A) are temporarily invisible to the agent. The information collected by the perception module is encoded and then is used to update the weights of the memory module, which employs a 1×20 AIMC array to store the 20 encoded values (including the 4×4 map information and the 4 coordinate values). The in-memory planning module decides the moving policy using a planning neural network, which is a 20×24×5 sized dueling Q-network and is trained through the Dueling Double Deep Q-learning algorithm (Details are provided in Methods). We use AIMC hardware to implement the 20×24×4 sized part of the trained Q-network (Fig. 4B), which receives the readout signals of the memory module as inputs. The 4 outputs of the AIMC-based planning neural network represent the estimated advantage functions of the 4 moving directions. The generated policy is to select the moving direction as the one with the maximum estimated advantage value (*i.e.,* the moving direction that is predicted to receive maximum future reward). As shown in Fig. 4C, through reinforcement learning approaches, the planning neural network gradually learns to generate the optimal moving policy navigating to the destination.

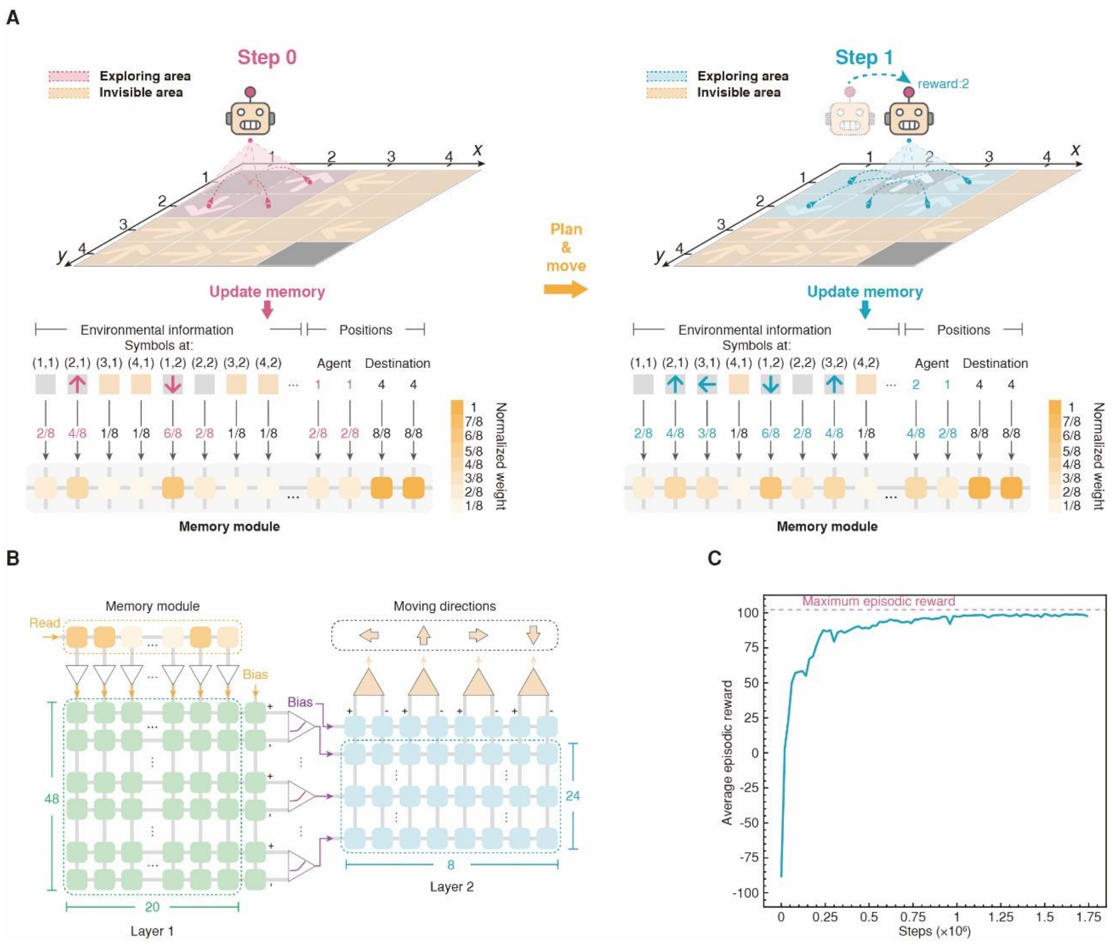


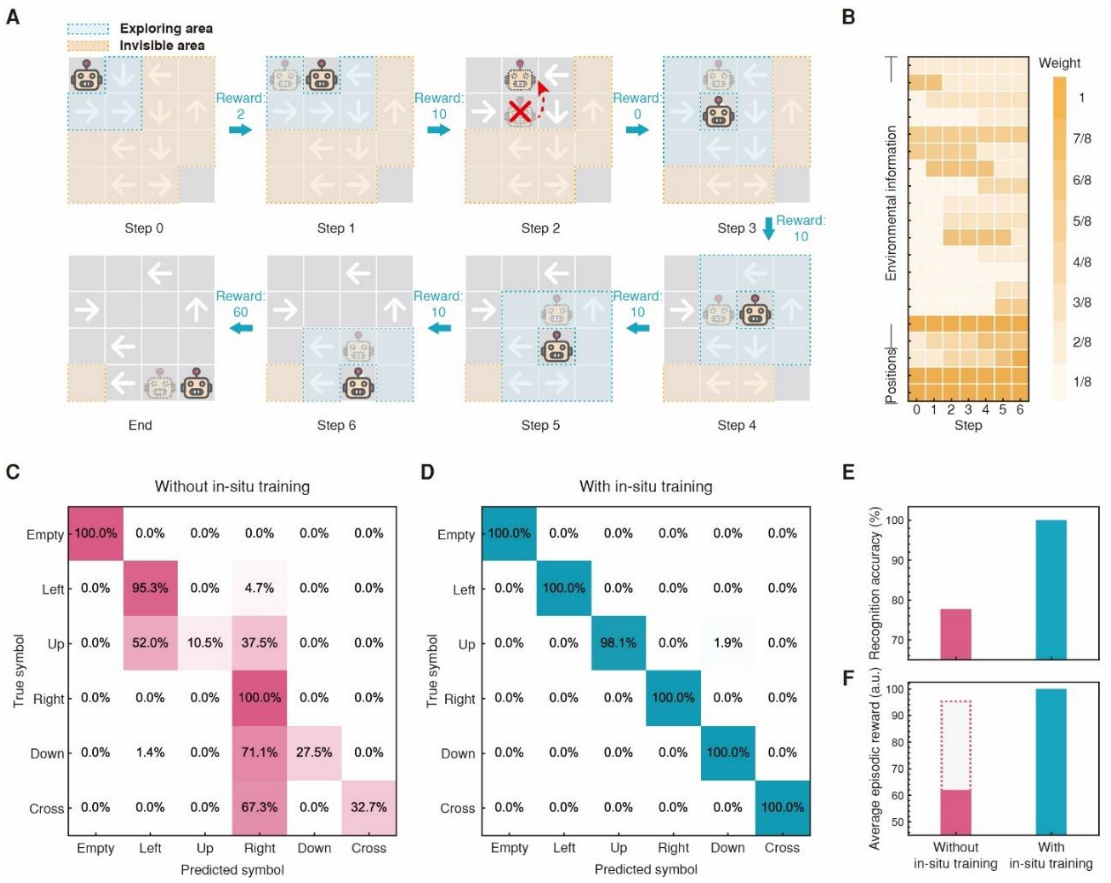
Fig. 4. Implementations of memory module and AIMC-based planning module. (**A**) A perceiving-planning-moving cycle. The agent explores the environment by recognizing the surrounding symbols, and stores the collected information within the memory module. The planning module decides the moving policy based on the information recorded in the memory module. After movement, the agent receives a reward signal from the environment. (**B**) AIMC-based implementation of the planning neural network. The readout signals of the memory module are used as the inputs to the AIMC-based planning neural network. The 4 outputs of the planning neural network correspond to the 4 moving directions. (**C**) The learning curve of the planning neural network. The average episodic reward is evaluated by interacting with random environments for 5000 steps. Through using

the deep Q-learning approach, the trained planning neural network is able to find the optimal path with the maximum episodic reward.

Overall system performance

We demonstrate that the in-memory embodied agent is able to successfully perform the assigned task in the physical world, through the cooperations between its modules of perception, memory, planning, and motion. Figure 5A depicts the agent's moving trajectory and received rewards in a trial. Despite the displayed symbol patterns being perturbed by noise pixels, the agent is able to accurately collect the environmental information using the adaptive perception module, and then stores the collected information in the memory module (Fig. 5B). Additionally, the agent is capable of avoiding the potential danger, which is represented by the displayed cross symbol and randomly occurs in the journey. As shown in the step 2 in Fig. 5A, the agent leaves from its current position when a cross symbol appears. Owing to the ultralow recognition latency of in-memory perception module, the agent can react to dangers rapidly. Through alternating exploration, planning, and moving, the agent reaches the destination in the optimal path that maximizes the received rewards.

We evaluate the performance of the embodied in-memory neuromorphic computing system through conducting 100 trials in random environments. Before in-situ learning, the perception module fails to recognize a substantial portion of symbols due to the gap between simulation and real-world conditions (Fig. 5C). Its unsatisfactory recognition capability gives rise to incorrect environmental information collection, misleading the planning module and reducing the risk avoidance capabilities. As a result, the agent receives an average summed reward of 61.6, where a reward reduction of 33.7 arises from the failures in risk avoidance (the dashed area in Fig. 5F). Through in-situ learning, the perception module gets adapted to the real environment. The trained agent achieves a recognition accuracy of 99.8% during 100 trials (Fig. 5D and 5E) and a success rate of 100% in avoiding dangers (*i.e.,* recognizing the cross symbol). Finally, the agent succeeds in accomplishing the assigned tasks in the real world, achieving an average summed reward of 99.6 (Fig. 5F).



	Empty	Left	Up	Right	Down	Cross
Empty	100.0%	0.0%	0.0%	0.0%	0.0%	0.0%
Left	0.0%	95.3%	0.0%	4.7%	0.0%	0.0%
Up	0.0%	52.0%	10.5%	37.5%	0.0%	0.0%
Right	0.0%	0.0%	0.0%	100.0%	0.0%	0.0%
Down	0.0%	1.4%	0.0%	71.1%	27.5%	0.0%
Cross	0.0%	0.0%	0.0%	67.3%	0.0%	32.7%

	Empty	Left	Up	Right	Down	Cross
Empty	100.0%	0.0%	0.0%	0.0%	0.0%	0.0%
Left	0.0%	100.0%	0.0%	0.0%	0.0%	0.0%
Up	0.0%	0.0%	98.1%	0.0%	1.9%	0.0%
Right	0.0%	0.0%	0.0%	100.0%	0.0%	0.0%
Down	0.0%	0.0%	0.0%	0.0%	100.0%	0.0%
Cross	0.0%	0.0%	0.0%	0.0%	0.0%	100.0%

Fig. 5. Performance of the AIMC-based embodied intelligent agent. (**A**) An example of the agent's trajectories from initial position to destination. The agent interacts with the environment through perceiving, planning, and moving. At the 2*nd* step, the agent leaves from its current position when recognizing the appeared cross symbol, which represents the occurrence of dangers. (**B**) The information stored within the memory module. The weights of 1×20 AIMC array in the memory module are updated according to the information collected at each step of the trajectory. (**C**) Confusion matrix of the recognition results, when the perception module is not adapted through in-situ training. (**D**) Confusion matrix of the recognition results after in-situ training. (**E**) Comparison between agent's recognition accuracies before and after environment adaptation. (**F**) Comparison between the episodic rewards before and after environment adaptation.

DISCUSSION

We have demonstrated an embodied intelligent agent that employs analogue in-memory computing technology to implement the perception, planning, and memory functions of human cerebrum. We show that the AIMC-based system significantly outperforms conventional digital approaches in terms of perception latency. Furthermore, we demonstrate that the in-memory perception module can adapt to real-world variations through in-situ training, ensuring robust recognition performance despite environmental noise and hardware variations. Through seamless cooperation between the perception, memory, planning, and motion control modules, the embodied intelligent agent successfully explores the dynamic environments, makes real-time decisions, and excellently accomplishes its designated tasks.

MATERIALS AND METHODS

Analogue in-memory computing chips

The AIMC chips are fabricated with 180nm CMOS technology. Each chip consists of 34×32 AIMC cells. The structure of the AIMC cell is similar to that of current mirrors, where the output current is proportional to the input current. Within each AIMC cell, there are a series of metal-oxide-semiconductor field-effect-transistors (MOSFETs), which possess gates of varying dimensions. These MOSFETs operate with a common source MOSFET, which is responsible for receiving the input current, denoted as I_{in}. The output current from the *ith* MOSFET is represented as I_i. The scaling from I_{in} to I_i is decided by the ratio of the gate dimensions between the *ith* MOSFET and the source MOSFET.

$$I_i = \frac{\frac{W_i}{L_i}}{\frac{W_{source}}{L_{source}}} I_{in} = C_i I_{in} \tag{1}$$

W and L denote the width and length of the MOSFET's gate, respectively. Each AIMC cell contains eight output MOSFETs, with each one being controlled by a static random-access memory (SRAM) cell. The *ith* MOSFET replicates the input current scaled by a factor of 2^{i-4}, which corresponds to the *ith* bit in an 8-bit integer. The total output current I_{out} from each cell is proportional to its input current I_{in}, with the scaling factor relating to the stored weight *w* in the cell.

$$I_{out} = I_{in} \times \frac{w}{8} \tag{2}$$

AIMC cells within the same row share the same input current. The output currents from AIMC cells within the same column are aggregated using Kirchoff's law. The AIMC chip performs VMM operations in parallel by applying input currents to it and collecting the output signals.

PCB systems of the in-memory embodied intelligent agent

We assembled an AIMC-based embodied intelligent system, which comprises a sensor module, a memory module, a neuromorphic computing accelerator, and a motor system.

The sensor module consists of an optical cage system, photodiodes, and amplifiers. The optical cage system is used to project the symbol displayed on a screen to the photodiodes, which are used for sensing the light intensities. The photocurrents generated from 5×5 photodiodes are amplified by the amplifiers, and serve as the inputs to the perception neural network, which is implemented by the neuromorphic computing accelerator.

The memory module contains a AIMC chip, which is used to store the environmental information collected by the perception module. Specifically, a 1×20 area is responsible for storing the collected information. The 20 outputs of the memory module are used as the inputs to the planning neural network, which is also implemented by the neuromorphic computing accelerator.

The AIMC-based neuromorphic computing accelerator is composed of AIMC chips, voltage-controlled current sources (VCCSs), transimpedance amplifiers (TIAs), nonlinear activation units, ADCs and a microcontroller. The AIMC-based accelerator is able to implement a two-layer analogue hardware neural network, where the maximum size of the first layer is 64×32 and the maximum size of the second layer is 32×16. We use the AIMC-based accelerator to simultaneously implement a 25×8×6 sized perception neural network and a 20×24×4 sized planning neural network. The outputs of these neural networks decide the behaviors of the motor system.

The motor system is a two-axis motorized stage, to which the sensor module is fixed. Two stepper motors work together to move the stage by driving the belts. The four behaviors correspond to moving the stage along the X/Y axis at a constant distance.

Environment settings

The embodied agent is trained to accomplish a task through interacting with the environment, which is a 4×4 grid displayed on the screen. There are random symbols within the grid cells, including the empty symbol, the arrow symbols of four directions, and the cross symbol. The agent is initially placed at the top left corner of the grid, and aims to reach the lower right corner. Note that there is an optimal path to the destination for each randomly generated grid. During the journey to the destination, the agent is supposed to correctly recognize the surrounding symbols, which are perturbed by random noise pixels. Besides, the agent should make the correct moving decision according to the collected environmental information to maximize the summed reward. The reward rules are designed as follows:

(1) If the agent moves beyond the grid boundary, the corresponding reward is -100, and the game ends.
(2) If the agent moves closer to the destination than ever, the corresponding reward is 10.
(3) If the agent moves in the direction of an arrow symbol, the reward is 0; furthermore, the arrow symbol is "used" and will be replaced with an empty symbol. Conversely, if the agent moves in the opposite direction of an arrow symbol, the reward is -20. In other situations, the reward is -8.
(4) If the agent correctly recognizes the appeared cross symbol and performs the avoidance action, the reward is 0; otherwise, the reward is -50.
(5) If the agent arrives at the destination, the reward is 50.

Hardware implementation of the perception module

The perception neural network is a two-layer analogue hardware neural network, which is of size 25×8×6. The 25×8 sized weights of the first layer correspond to a 25×16 area of a AIMC chip, where a pair of AIMC cells represents a weight value. Besides, the 8-channel bias values are programmed into a 1×16 AIMC cell array. The 25-channel photocurrents generated from the photodiodes are amplified and used as the inputs to the perception neural network. The difference of the output currents from each pair of columns is amplified by a differential TIA, then undergoes activation by a nonlinear activation unit, and finally is transformed to current by a VCCS. The 8-channel output currents from the first layer serve as the inputs to the second layer. Similarly, the weights and bias values of the second layer are programmed into an 8×12 area and a 1×12 area of another AIMC chip, respectively. The 6-channel outputs of the second layer represent the outputs of the perception neural network. A detailed illustration of the hardware perception neural network is shown in Supplementary Fig. S6.

Latency comparison between the in-memory perception module and the digital scheme

We compared the in-memory perception module and the digital counterpart in terms of the latency of processing the sensor signals. The latency is defined as the period from the time when sensor signals are produced until the completion of the VMM operation on these signals. The latency of AIMC-based VMM operation is measured by applying pulse signals to the AIMC array and probing the output response. The digital processing latency is measured based on a STM32F407 microcontroller unit (MCU), which is configured with a clock frequency of 168 MHz. An on-chip ADC of the MCU is used to digitize 25-channel sensory signals into floating point numbers. Afterwards, the MCU performs the VMM operation on the 25 digital values and the 25×8 weight matrix.

Hardware implementation of the memory module

The memory module is used to store the collected environmental information, including the 4×4 map, the agent's position (*i.e.,* x and y coordinates), and the destination's position. These 20-channel values are encoded and programmed into a 1×20 area of a AIMC chip, which receives a constant input current and generates 20-channel output currents. The environmental information encoding is described as follows. The 4×4 map is recorded as 16 values, with each value corresponding to a cell in the grid. The potential statuses of a grid cell and the respective encoded values are listed as below: unknown: 1/8; empty symbol: 2/8; left arrow symbol: 3/8; up arrow symbol: 4/8; right arrow symbol: 5/8; down arrow symbol: 6/8; destination: 1. Additionally, the position information is encoded through normalizing the coordinate values. Initially, the statuses of all grid cells are unknown except for the starting point and the destination, and the agent's position is (1, 1) while the destination's position is (4, 4). The 1×20 AIMC cells are programmed according to the encoded 20 values, and the value 1 means programming the corresponding AIMC cell to produce an output current of 5μA. The 20-channel output currents serve as the inputs to the planning neural network.

Hardware implementation of the planning module

The planning neural network is a 20×24×5 sized two-layer neural network, whose outputs contains the estimated value function and the advantage functions corresponding to the 4 moving directions. AIMC hardware is used to implement the 20×24×4 part of the planning neural network, which outputs the estimated advantage functions. Similar to the perception neural network, a 21×48 AIMC array and a 25×8 AIMC array are used to implement the first and second layer of the planning neural network, respectively. A detailed illustration of the hardware planning neural network is shown in Supplementary Fig. S6. The final 4 outputs represent the estimated advantage values of the 4 moving behaviors (*i.e.,* moving left, up, right, and down), among which the one with the maximum advantage value is selected as the planned action.

Training the perception neural network

The analogue hardware perception neural network is trained to correctly recognize the 6 types of symbols. We measured the input-output characteristic curve of the nonlinear activation unit (Supplementary Fig. S3) and used it as the activation function during software training. The cross entropy between the one-hot labels and the scaled output voltages is selected as the loss function. The trained weights were programmed into the AIMC chips used for implementing the perception neural network. Afterwards, we adopted an in-situ training approach to address the performance degradation caused by the disparities between simulation and reality, including the environmental noise and the variations in the device performance of photodiodes.

In-situ training approach

During in-situ training, we use a perturbation-based approach to obtain the gradients of the neural network parameters. This approach relies on a two-phase forward propagation process. In the normal phase, a symbol as well as random noise pixels is displayed on the screen, and then is projected onto the 5×5 photodiodes, whose produced photocurrents serve as the inputs to the hardware perception neural network. In the perturbation phase, we inject a small current into the output currents of AIMC chips, while the inputs of the neural network remain the same. The gradients can be calculated based on the injected current and the difference between the neural network's outputs in the two phases. For example, denote the first column in the weight matrix (including the bias) of the first layer as $W_1^{(1)}$, its gradients can be estimated through injecting a small current (denoted as Δi) to the difference output current of the first column pair and calculating the difference between the loss values in the two phases:

$$g_1^{(1)} = \frac{L_{normal} - L_{perturbation}}{\Delta i} \cdot \overrightarrow{I_{in}} \quad (3)$$

where L_{normal} and $L_{perturbation}$ denote the respective loss values in the normal phase and the perturbation phase, and $\overrightarrow{I_{in}}$ denotes the input currents of the first layer. During in-situ training, we recorded a set of full-precision weights, and these weights were updated according to the calculated gradients using the batch gradient descent approach. The updated weights were rounded to integers and then were reprogrammed into the corresponding AIMC arrays. Note that the batch size was 256 in experiment. The parameters of the trained perception neural network are shown in Supplementary Fig. S4.

During in-situ training, the output currents of each layer are perturbed by injecting a small current, which influences the final outputs. Through measuring the perturbation's effect on the loss function, we can estimate the gradient of the corresponding perturbed current. For example, supposing the change of the loss value is $\Delta Loss_1$ when injecting a current (denoted as Δi_1) to the 8th output channel of the first layer, then the gradient of the corresponding output current is estimated by $\Delta Loss_1/\Delta i_1$. Furthermore, the gradients of the corresponding weights (represented by $W_8^{(1)}$) can be calculated by $I_1 \Delta Loss_1/\Delta i_1$, where I_1 denotes the input currents of the first layer.

Training the planning neural network

The planning neural network is trained to make the optimal decision according to the collected environmental information, which is stored within the memory module. We trained the planning neural network through reinforcement learning approaches, specifically the dueling double deep Q-learning algorithm, which is described as follows:

Dueling double deep Q-learning.

```
Initialize dueling Q-network $Q$ with random parameters $\theta$. The Q-network outputs the value function $V(s;\theta)$
and the advantage functions $A(s,a;\theta)$, the Q-value is calculated by: $Q(s,a;\theta) = V(s,\theta) +$
$\left(A(s,a;\ \theta) - \frac{1}{N_a}\sum_{a'} A(s,a';\theta)\right)$, where $N_a$ denotes the dimension of the action space.
Initialize target dueling Q-network $Q'$ with parameters $\theta'$, $\theta' \leftarrow \theta$.
Initialize replay buffer $D$ and collect $N_{random}$ transitions through interacting with the environment using
random actions.
Reset the memory module; randomly reset the environment.
episode ← 0.
While episode < M do
  For t from 1 to T do
```

```
        Collect information from environment, update the memory module with the observed state $s_t$.
        Select a random action $a_t$ with probability $\varepsilon$
        otherwise select $a_t = argmax_a Q(s_t, a; \theta)$.
        Execute the action $a_t$ and receive reward $r_t$, collect information from environment and update the memory
        module with the observed state $s_{t+1}$.
        Store transition $(s_t, a_t, r_t, s_{t+1})$ in $D$.
        If episode terminates, reset the memory module and the environment, *episode* ← *episode*+1.
    End For
    For $x$ from 1 to N do
        Sample random minibatch of transitions $(s_i, a_i, r_i, s_{i+1})$ from $D$.
        Define $a_{i+1}^{max} = argmax_a Q(s_{i+1}, a; \theta)$.
        $y_i = \begin{cases} r_i & \text{if episode terminates at step } i+1 \\ r_i + \gamma Q'(s_{i+1}, a_{i+1}^{max}; \theta') & \text{otherwise} \end{cases}$
        Calculate the gradients of $\theta$ according to the squared temporal difference error $(y_i - Q(s_i, a_i; \theta))^2$.
        Update $\theta$.
        Update the parameters of the target Q-network, $\theta' \leftarrow \tau\theta + (1-\tau)\theta'$
    End For
End While
```

Supplementary Materials

Fig. S1. The optical photograph of the in-memory embodied intelligent system.

Fig. S2. Comparison of sensor signals in simulation and real world.

Fig. S3. The nonlinear activation unit.

Fig. S4. Parameters of the perception neural network.

Fig. S5. The trained parameters of the 20×24×4 section of the planning neural network.

Fig. S6. The programmed weights of AIMC chips for implementing the perception and planning neural networks.

Movie S1. The performance of the AIMC-based embodied agent on navigation tasks.

References and Notes

1. There is only one reference list for all sources cited in the main text, figure and table legends, and supplementary materials. Do not include a second reference list in the supplementary materials section. References cited only in the supplementary materials section are not counted toward length guidelines.
2. Each reference should have a separate number and should be on a separate line ending in a period. For details of correct reference style, with examples, see https://www.science.org/content/page/scirobotics-instructions-research-articles.

    Example: A. Person, B. Being, Article title: Then subheading. *Credible Journal* **#Volume**, #pg–#pg (#Year). doi:#here
3. You can use an automatically numbered list in MS Word. The reference list should be in the order in which references are cited: first through the main text, then through figure and table captions, and last through Supplementary Materials.
4. Please do not include any extraneous language such as explanatory notes as part of a reference to a given source. *Science Robotics* requires that manuscripts do not include end notes; if information is important enough to include, please put into main text.

Acknowledgments:

Funding

National Key R&D Program of China 2023YFF1203600
National Key R&D Program of China 2023YFF0718400
National Natural Science Foundation of China 62305155
National Natural Science Foundation of China 62122036
National Natural Science Foundation of China 12322407
National Natural Science Foundation of China 62034004
National Natural Science Foundation of China 12074176
Leading-edge Technology Program of Jiangsu Natural Science Foundation BK20232004
Fundamental Research Funds for the Central Universities 02042103031
AIQ foundation
Xiaomi Young Scholar Foundation

Author contributions: Each author's contribution(s) to the paper should be listed [we encourage you to follow the CRediT model]. Each CRediT role should have its own line, and there should not be any punctuation in the initials.

Examples:
Conceptualization: Z.Z.Y., C.W.
Methodology: Z.Z.Y., H.Z., Y.C.Z.
Investigation: Z.Z.Y., H.Z., C.W., Y.Y., C.P.
Visualization: Z.Z.Y., H.Z.
Funding acquisition: C.W., S.J.L., F.M.
Project administration: S.J.L., F.M.
Supervision: S.J.L., F.M.
Writing – original draft: Z.Z.Y., H.Z., C.W., S.J.L., F.M.
Writing – review & editing: Z.Z.Y., C.W., S.J.L., F.M.

Competing interests: Authors declare that they have no competing interests.

Data and materials availability: All data and code are available in the main text or the supplementary materials, and from the corresponding authors upon reasonable request.