Engineering 6 (2020) 221–226

Contents lists available at ScienceDirect

Engineering

journal homepage: www.elsevier.com/locate/eng

Views & Comments

The General-Purpose Intelligent Agent

Cewu Lu [a,#], Shiquan Wang [b,#]

[a] Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China
[b] Flexiv Ltd., Shanghai 201105, China

1. Introduction

In our daily life, we expect smart devices to help us with different kinds of tasks. For this reason, it is necessary to design a variety of robots and related algorithms specifically for different tasks and scenarios. However, such a methodology results in smart devices with limited intelligence that fall far below our expectations and fail to adapt to different real-life task scenarios. In contrast, if we regard humans as "intelligent agents," such an agent, given appropriate tools, can perform different manipulation tasks in various scenarios. Inspired by this idea, we perceive a need for a general-purpose intelligent agent (GIA), such that most tasks can be executed by the same agent, which will significantly accelerate the development of intelligent industries and increase the convenience of human lifestyles. We also believe that, in order to approach the goal of artificial general intelligence [1], a powerful and standard GIA is necessary.

This paper discusses the architecture of the GIA, starting from two perspectives. The first perspective is a GIA that is humanlike: We hope that the GIA can have the full range of humanlike perception [2], decision-making [3], and knowledge and learning [4]. We also hope to achieve a GIA with the ability to adapt existing knowledge and experiences to new situations, and then drive the body to implement all kinds of manipulations correctly, thus eventually accomplishing tasks. The second perspective is a GIA that is computer-like: We hope that the design of the GIA architecture can refer to the idea of computer design [5]—for example, where clear protocols are defined among the modules, the module itself can be upgraded, and the user can describe the task through high-level semantic programming language without putting effort into the bottom-level design of the system. After fully considering these two perspectives, we propose a primitive flow model to extract the manipulation-primitive commonalities across tasks and targeting objects in order to complete as many tasks as possible. Based on the primitive flow model, this paper suggests a GIA that is equipped with five basic modules: an execution module, a perception module, a task compiler, a knowledge engine, and a central GIA processor. In this paper, we will discuss how these modules can be functionally compared with human beings (i.e., in terms of being intelligent), while simultaneously aligning with the concepts of computer design (i.e., in terms of being workable). We expect a GIA with this architecture to have three characteristics:

(1) Transferability: For most tasks, the user only needs to provide a simple task description; there is no need to change the agent or redesign the bottom-level common hardware, protocols, or operating system.

(2) Scalability: Every basic module is relatively independent and can be upgraded, like the central processing unit (CPU) and random access memory (RAM) in general computers.

(3) Knowledge expansion: Knowledge can be gained and experience can be shared among numerous GIA units. With the accumulation of completed tasks within a group of GIAs, the knowledge system will be expanded, and the time required to learn new tasks will be significantly reduced.

We further discuss how to quantify the intelligence capability of an agent. It is our opinion that task transferability is the most important index for evaluating the intelligence capability of an agent. Therefore, we propose the agent–human cost ratio (AHCR) index, which is the ratio of the time required by an agent to learn a new task to that required by a human to learn the same task. This indicator can well evaluate the comprehensive ability of an agent in terms of general intelligence. It must be noted that the concept of the GIA presented in this paper is not intended to be that of an agent that can complete all human manipulation tasks; rather, it is the concept of an agent that can pursue as many tasks as possible that can be completed by a unified and standard agent. In certain scenarios, special robots can be more efficient than a GIA according to the need, such as floor-sweeping robots. This is similar to the relationship between general computers and special computers; although the vast majority of people use general computers with a general-purpose architecture, computers with a special architecture are still required to run certain special calculations.

In the remainder of this paper, we will introduce the idea of the primitive flow model, the basic architecture of the GIA, the benchmark design, the community building, and the impact of the GIA on intelligent science. We will then discuss the applications of the GIA in various industries. Finally, we draw a conclusion.

[#] These authors contributed equally to this work.

https://doi.org/10.1016/j.eng.2019.12.013

2. The primitive flow model

Many manipulation tasks need to be completed in a variety of industries and in daily life. The core of the GIA is to describe these nearly infinite tasks using a simple model. In this paper, we propose the idea of the primitive flow model. Basically, any human manipulation can be broken down into multiple steps; for example, the task of unscrewing a screw can be divided into ① grabbing the screwdriver, ② moving the screwdriver to align with the screw, ③ twisting the screwdriver, and ④ moving the screwdriver away from the screw.

Following this thinking, we explore how to break down a complicated manipulation task into several micro-tasks. To clarify the decomposition of one task, we define "general manipulation that cannot be further decomposed" as the manipulation primitive; examples include grasping, twisting, inserting, and kneading. By "general," we mean that the manipulation primitives are required to have a wide range of transferring commonalities, which are cross-task and cross-object.

For example, there is a strong commonality between twisting open the cap of a chemical reagent bottle (e.g., in the pharmaceutical industry) and twisting open the cap of a salad dressing bottle (e.g., in the household industry). Moreover, it can be assumed that a human's manipulation-primitive set is finite. Therefore, after the manipulation-primitive set is defined, any human manipulation task can be parsed as a primitive flow—that is, as a sequence of manipulation primitives. Next, we will present an in-depth discussion about the complete definition of manipulation primitives and the construction of primitive flows.

2.1. Three elements of a manipulation primitive

A complete definition of a manipulation primitive includes three components: the manipulation-primitive type P (e.g., grasping, twisting, and inserting); the situation of the targeting object O (including manipulated objects and tools), which indicates the visual, force, and sound semantics; and the set of the final state S of the targeting object after the task has been accomplished. There are many possibilities for the object's final state in some tasks; for example, breaking glass may result in the common broken state of normal glass or in the granular crushed state of tempered glass. Hence, we consider S as a set, and after the successful execution of the task on the object O by the manipulation-primitive type P, the final state changes to be a member of set S. These components are called the three elements of the adaptive manipulation-primitive triplet $\{P, O, S\}$.

When the agent acquires the triplet $\{P, O, S\}$, it activates the knowledge of the manipulation-primitive type P, extracts the state and property of object O (i.e., appearance, force sense, materials, functionality, and affordance), and executes adaptive manipulation to achieve the state S. This process also relies on a powerful knowledge engine, which will be introduced later. As explained above, we conceptually define the set S, however, from a practical standpoint, if S is too large to enumerate, we will design (or train) a discrimination function to output a binary state indicating whether the primitive is completed or not.

2.2. Primitive-based task parsing

After confirming the manipulation-primitive set, it becomes an open problem of how to parse a targeted task into a series of primitives—namely, a primitives flow. In addition, the environment of the GIA is always dynamically changing. Thus, a dynamic task-parser function $Z()$ is needed to sequentially select the manipulation primitive dynamically according to the environmental changes.

Discussion: The manipulation-primitive flow model can simplify the description of various complex manipulation tasks. Considering the limited types of human manipulation primitives, any task can be converted into a primitive sequence. Manipulations under the same primitive operation have strong commonality even when addressing different objects, which makes it possible to extract common knowledge to achieve powerful task transferability.

3. The architecture of the GIA

As shown in Fig. 1, we now discuss the basic modules of the GIA: the execution module, perception module, task compiler, knowledge engine, and central GIA processor. These modules work together to complete a variety of tasks under a unified protocol, with no need to redesign the software and hardware architecture.

Taking the primitive flow model as the task-understanding principle, we introduce the five basic modules below.

Execution module. This part mainly consists of the mobile platform and the manipulation platform.

(1) Mobile platform. By the spatial movement of the mobile platform, the manipulated object is located in the working space of a manipulation platform. Common mobile platforms include systems that are wheeled, four-legged, two-legged, or with slide rails, the mobility limitations and flexibility of which are reduced in turn, while the control difficulty is increased in turn. The question of how to improve the control stability, flexibility, and efficiency is still an important research topic in the field of robotics.

(2) Manipulation platform. By applying the force of different positions and dimensions to the objects, the ideal result is achieved, as planned. Common manipulation platforms include robotic arms of the parallel or series type and their end-effectors, as well as driving methods such as the electric, pneumatic, and hydraulic drive. How to balance the accuracy and flexibility of

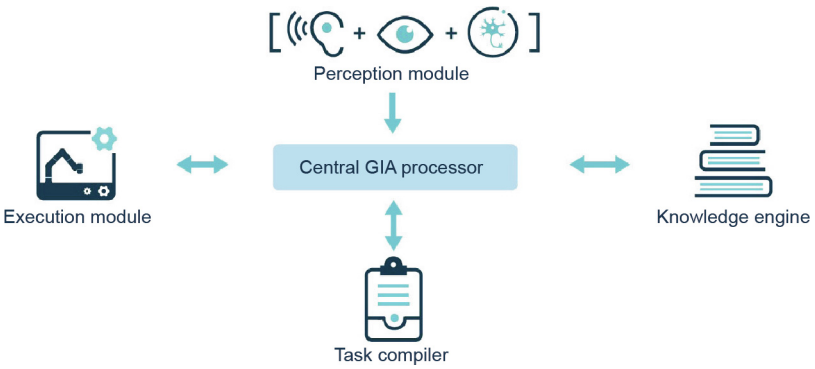

Fig. 1. The proposed GIA architecture with five basic modules.

the manipulating force, together with the speed, stability, load, and energy efficiency of the manipulation platform, is a key issue for the further evolution of the execution module.

Perception module. The perception module receives external signals, which includes collecting visual signals, force signals, and sound signals, and simultaneously analyzes the vision, force, and auditory semantics in the environment. Among them, the analysis of visual and auditory semantics requires the application of computer vision and speech-recognition technology.

Technology prospect: Visual and auditory signal acquisition is a relatively mature technique in which visual information may include accurate three-dimensional (3D) and thermal information. At present, computer vision technology based on deep learning has limited capability to recognize objects that are not present in the training set. How to fuse interactions to environments and improve the recognition of new objects is also an open academic issue. The feedback information of the force sensor may include low-frequency six-dimensional force (3D contact force and 3D contact torque) signals, high-frequency tactile signals (i.e., the vibration state of the contacting surface, which can be used to identify the physical and manipulation-state characteristics of the contacted object), and tactile spatial information (i.e., the spatial distribution of force signals, typically obtained by sensors in the form of arrays and/or class-like image features). At present, research on the design and analysis methods of force sensors is still evolving. How to obtain more accurate and comprehensive force information at lower cost and how to integrate force information with other sensing dimensions are also open academic subjects at present.

Task compiler. The task compiler is the interface for users to assign tasks to the GIA. We propose to develop a task-oriented high-level programming language to translate task descriptions from a user into a primitive flow in real-world applications. To be specific, the task compiler is responsible for compiling a task into an executable dynamic task-parser function $Z()$. The function can select the current optimal manipulation primitive from a pre-built manipulation-primitive library based on the information given by the perception module and the knowledge from the knowledge engine. Unlike traditional computer programming languages, users can implement a task description besides by program coding, and can also through other various methods such as teaching by demonstration or natural languages.

Open questions: A new scientific problem will be how a task (especially a complex manipulation task) can be described in order to generate (or learn) a dynamic task-parser function $Z()$ of the manipulation primitive, which machines can execute. Demonstration is intuitive, and has been studied for a long time [6], but it is usually robotic-specific and cost-prohibitive to record every possible successful path for a complex task. Natural language is promising, as natural language processing (NLP) techniques have made considerable progress lately [7]; however, the open-set feature of a natural language prevents the language from being learned effectively, and types of closed-set vocabulary that can be used for robotics instruction are rarely explored. A closed-set vocabulary is also needed for program coding.

Furthermore, it is necessary to think about "machine teaching," and not just about machine learning. Most researchers focus on machine learning because machine teaching is relatively simple due to the only requirement of sample labeling; nevertheless, a more challenging problem is how to teach a machine to understand the task.

Technology prospect: For tasks with fixed steps (e.g., industrial scenarios), we can program the manipulation-primitive sequence directly. For tasks requiring a dynamic decision-making process, in which decisions on what kind of manipulation primitives should be used are needed for each step, a relatively more achievable method is to observe human decision-making behavior and adopt imitation learning. In this process, "future prediction" is also a key technical point. Humans do not need large-scale trial and error to select a primitive; rather, we rely on basic predictions of the future, which greatly enhances the possibility of successful decision-making.

Knowledge engine. Under a unified protocol, different users can edit the knowledge of objects and manipulations, and can expand the knowledge base. For example, microwave oven manufacturers can edit their manipulation knowledge, and this knowledge will be uploaded to the central GIA knowledge engine in the cloud by the same protocol. The GIAs analyze the task and environment, and then access the corresponding knowledge in the base or the most similar knowledge set, to enable decision-making.

Open questions: ① How can the object knowledge protocol be defined for object-manipulation? ② How can human–machine (or human-in-loop) interaction be used to capture dynamic task-parsing knowledge for function $Z()$? ③ When there is no matching knowledge in the knowledge base, how can similar knowledge be relied on for reasoning?

Technology prospect: We want to build a comprehensive knowledge base, including:

(1) A knowledge engine based on object attributes. This would collect a large amount of 3D object information and edit object attributes, including the functionality, affordance, and physical attributes of every part of the object. At present, no object database exists with all the annotation information. The current largest object dataset, ShapeNet [8] annotates some attributes with an incomplete coverage of both the objects and the attributes. However, the problem of how to establish an effective protocol for annotating object attributes is crucial. For example, when setting up space for object attributes, the understanding of the object will be further generalized. When encountering unknown objects, it is still possible to understand the object well after obtaining its attributes through physical interactions, and these understandings can then be applied to the manipulation. Such settings cannot be met with ShapeNet.

(2) An adaptive manipulation-primitive knowledge engine. For each specific manipulation primitive, there is a certain commonality in dealing with different objects, and this commonality is cross-task. We hope to extract common manipulation knowledge under the same manipulation primitive in order to achieve manipulation tasks very well for different objects. Specifically, for the ith primitive, given the set of operation objects O (possibly including objects, tools, etc.) and the final state S, M_i outputs an executable instruction agent action u to complete the manipulation primitive, which can be mathematically written as $u = M_i(O, S)$. The construction of the function M_i needs to combine real-time force, vision, and other perception information and related knowledge in the engine. It should be noted that the manipulation of O strongly relies on object attribute knowledge. Taking grasping as an example, we need to know which part of the targeting object can be grasped. The concept of the manipulation primitive has been developed in two streams: explicitly (i.e., rule-based) [9] and implicitly (i.e., data-driven) [10]. Traditionally, the explicit stream has never attempted to unite all the everyday manipulations and focus on several specific small tasks, such as grasping, moving, and sweeping [11]. It is difficult to combine the primitives defined by rules into a fairly complex task. In contrast, the implicit stream aims to learn manipulation patterns entirely through data. Given the complexity of the task topology, the appropriate amount of required data could be prohibitive to collect. Our idea is closely related to the explicit stream, as we assign each primitive a human-recognizable name.

(3) A task-parsing knowledge engine. When we set up and improve the manipulation-primitive knowledge engine, we need

to determine the combined steps of the manipulation primitive according to the tasks, including many tasks with unfixed steps. At present, human behaviors are recorded by massive videos; thus, we can analyze the sequence of the manipulation primitive of these behaviors under various environments [12] and manipulated objects to obtain the task knowledge engine [13]. This can greatly improve the efficiency of the users using the task compiler to describe the task. Task parsing is challenging because the task can often be achieved through different routines. In other words, the task topology of a regular task is complex, and the model parameters cannot capture such a huge space. This is the main reason why the data-driven approach is infeasible for such a problem [14]. However, our manipulation-primitive knowledge engine will give a strong priority to the task-parsing problem.

Central GIA processor. After parsing the task of the task compiler, the central GIA processor reads the sensor module signal and invokes knowledge in the knowledge engine to make a comprehensive decision and issue instructions to the execution module. This module is the central computing module of the whole system, which quickly accesses the cloud knowledge engine and performs real-time decision-making. This module puts forward certain requirements for the computing design.

This architecture has three desirable characteristics:

(1) Task-oriented semantic editing. Users can issue task descriptions to the GIA through the task compiler by means of the high-level programming language for tasks, and do not need to design the underlying structure. This is similar to a situation in which when we use a high-level computer programming language; there is no need to design or handle various details such as the computer CPU, memory, and so on.

(2) Scalable modules under the standard protocol. The perception module, execution module, and central GIA processor have a unified operating protocol and communication protocol among the modules, under which hardware or software can be independently upgraded.

(3) Knowledge and experience can grow. After establishing certain editing protocols for objects and knowledge, the knowledge engine will be open to all. Users can submit knowledge modules of objects, manipulation primitives, and task knowledge under unified specifications to achieve never-ending knowledge growth.

Discussion: We propose that the GIA should not only be analogous to humans in terms of its intelligent function, but also be analogous to the design of a computer (Table 1):

Analogous to humans. The execution module is similar to human hands and feet; the perception module can be compared to the eyes, ears, and tactile nerves, which are used to perceive information from the environment; the task compiler when understanding the task description is similar to the area in the brain that understands language; the knowledge engine can be compared to the brain memory area; and the central GIA processor's function of comprehensive processing can be compared to the specific part of the brain that is responsible for logic.

Analogous to computers. A computer cannot directly affect the external physical world, so the execution module does not have a corresponding computer analog. Similarly, the perception module is analogous to the computer input devices, such as keyboards and mouse. The task compiler is analogous to a compiler in a computer, which allows various programs to be executed on a central GIA processor (analogous to a CPU), while the knowledge engine can be analogous to computer memory. Table 1 lists the human and computer analogs for the GIA.

Table 1
Comparison of the GIA's five modules with human organs (function) and computer modules (design).

GIA modules	Comparable human organs	Comparable computer modules
Execution module	Hands, feet	—
Perception module	Eyes, ears, tactile nerves	Keyboard, mouse
Task compiler	Brain language section	Compiler
Knowledge engine	Brain memory section	Memory
Central GIA processor	Brain logic section	CPU

4. Benchmark

A quantified index is needed to measure the performance of the GIA. Intelligent transferability, which mainly refers to the cost of learning a new task, is the direct measurable standard for general intelligence level; transferability can also be regarded as a key bottleneck in the development of artificial intelligence (AI) technology based on deep learning [15]. Therefore, we propose the AHCR as the measurement index. The AHCR is defined as follows:

For task A, we have:

T_{agent}: the average time needed from the user obtaining task A's descriptions to the agents acquiring the whole skill of task A;

T_{human}: the average time needed from a human learner obtaining task A's descriptions to the learner acquiring the whole skill of task A.

Thus, we can compute the AHCR of task A as follows:

$AHCR = T_{agent}/T_{human}$

T_{agent} is also called "the overall cost of agent teaching and learning." Here, average time is used considering people's time-consumption bias. The main insight of the AHCR is that for the time ratio of an agent's task learning to a human's task learning, the value of the ratio is hoped to be lesser than or close to 1.

This index can measure an agent's general intelligence capability comprehensively. Teaching and learning cost are two important factors. ① Teaching cost largely depends on the task complier's maturity. If a considerable amount of time must be spent on task description, due to a poor task complier (e.g., too much hard-coding time), the overall teaching time will be increased. In an ideal situation, we can use natural language or body language to describe the task, and the agent can clearly understand it. ② Learning cost is the time needed for agent learning, such as model training. It should be noted that, as estimated, most of the tasks require a human-in-the-loop for users to re-code according to feedbacks, which tests the agents' active learning effectiveness. In addition, the more powerful a GIA's knowledge engine and reasoning/conducting ability are, based on learning cost, the less time is needed.

Scalable task zoo. Since there will be some bias in using a single task to evaluate an agent's transferability, it is necessary to set up a task zoo with enough diversity and representativeness to avoid measuring bias. We will use average AHCR to set a measurement standard. We expect that, with the task quantity rising, the overall learning duration will decrease. Under this circumstance, a new academic question will emerge: What kinds of tasks should be the first and should come later in order to ensure that the learning cost is minimized? For example, humans learn new things starting from easier learning and moving to more difficult learning.

5. Community and experience sharing

The development of the GIA strongly relies on experience sharing, which means that a considerable number of academic and business communities, along with effective organization, are needed to teach the GIA various tasks. We expect the advent of the following communities:

- Perception model communities, in which members submit and check various kinds of perception models, which are used for different object detectors;

- Task zoo communities (where we expect the task zoo to be scalable), in which all members can submit new tasks, and an approval committee is needed to check whether the task submitted is approved or not;
- Task compiler communities, which encourage the development of multiple task-oriented task-programming languages, and where, after competitions, a few of the most outstanding languages win and are widely used;
- Knowledge engine communities, in which members submit and check various kinds of manipulation primitives, objects, and knowledge.

6. Impact of the GIA on intelligent science

In our opinion, the GIA goes far beyond the idea of AI-empowered robots, and will contribute enormously to the development of intelligent science. The key bottleneck faced by the mainstream deep learning method is that untrained data performance significantly drops for new tasks—that is, there is low transferability. The GIA can enhance machines' transferability on tasks.

First, in terms of an understanding of world concepts, agents need a deep understanding of the concepts of their environment (especially the operational objects). Based on the primitive set with generalization ability, the agent can have a deeper understanding of the physical world by interacting with objects, and can better represent various concepts of objects, such as functionality, operation modes, and materials. For example, we understand the concept of scissors by judging whether an object can cut things through interactions. As Fig. 2 shows, a human can conclude that what an object is, even though no similar pictures have been provided in the training set. In contrast, a traditional learning algorithm cannot achieve this result without relevant information being present in the pre-training.

The GIA provides an interactive foundation with the outside world that allows the agent to gain additional object property understandings; this differs from the traditional deep learning system. Based on property perception, the system can also gain a good understanding of unseen objects. With the assistance of manipulation-primitive construction, the research will shift from pattern recognition to object understanding, enabling a good generalization ability for unseen objects.

Scalable swarm intelligence. Even though this concept has been put forward for over 20 years, it is difficult for swarm intelligence to function across tasks [16]. The GIA provides a good opportunity for scalable swarm intelligence due to its foundation of the manipulation primitive, as the manipulation primitive processes strong universality in different tasks. With the mass manipulations created and documented by GIA communities, this universality will be further generated to support task transfer. Furthermore, as GIAs have a common protocol, it will be much easier to share experience and achieve stronger transferability among GIAs. In addition, empowered by universality, each manipulation primitive can achieve a more effective structured indexing of numerous intelligence spaces, and can construct reasonable connections among common semantics.

Fig. 2. Scissors cannot be recognized by a vision deep learning model if this type of scissors was not present in the training set. However, humans can recognize the object as "scissors" based on its functionality through manipulation.

7. Applications

The GIA will significantly improve working efficiency, and is expected to bring huge changes into the manufacturing, medical treatment, catering, agriculture, and household product industries among others. For example, it is currently impossible to redesign hardware, software, or learning models to address every new task in each specified home environment, which is a key obstruction to household product development. Considering the different difficulty levels of various fields, the GIA's actual application will occur in sequential steps. Table 2 roughly summarizes four levels of GIA application.

We consider that a high intelligence level is needed to determine what manipulation primitive to use each time, such as when repairing objects. With the help of the knowledge engine, the objects of manipulation can perform adaptively to the change of the environment. For this reason, we can establish a distribution of different fields of application across the four levels of difficulty (Fig. 3). However, it cannot be said that a specific application will be at only one level; rather, the applications are cross-level. Among the considered GIA applications, industry GIA is at a relatively simple level, since after confirmation of the task, the object and environment of manipulation do not vary much, and the manipulation process is known. In contrast, household GIA involves interaction with humans (i.e., uncertain objects) as well as unknown processes (e.g., caring for the aged), which increases the complexity.

Thus, we suggest that the industry GIA application should be realized first, since it will be possible to implement this type of application relatively soon. This application can then function as a basic prototype for future common GIAs, based upon which subsequent GIAs in other industries can upgrade, eventually achieving cross-industry universal GIAs.

Execution module. Most of the workers who focus on assembling are fixed in position in front of the station. The execution module of such a GIA can be a pair of robot arms, with seven degrees of freedom (DOFs) to match a human's flexibility. When assembling complicated parts, a human usually relies on a sense of touch and force to "feel the way" toward successful assembling, which is a process of applying forces and moments in different dimensions (6 DOFs in total) on the parts. Therefore, the arms must be able to exert different combinations of forces and moments dexterously on the parts. Thus, the robot arms should have high 6 DOFs force-control performance.

Perception module. For an arm with a high force-control performance, high-quality force/moment sensors are required. These

Table 2
Four levels of difficulty for GIA application.

Level	Condition of object and environment	Process step
1	Both the environment and the objects of manipulation vary slightly	Known process step
2	Either the manipulation environment or the objects of manipulation varies greatly	Known process step
3	Both the environment and the object of manipulation vary greatly	Known process step
4	Both the environment and the object of manipulation vary greatly	Unknown process step

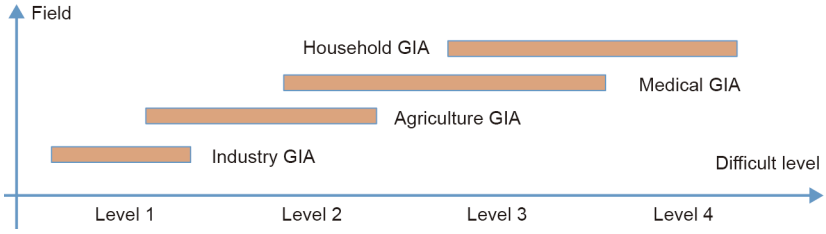

Fig. 3. Distribution of different GIA applications on different levels of difficulty.

could be embedded in the arm joints or on the end-effectors. It is also necessary for the GIA to imitate the human senses of touch and vision, which identify objects, localize the positions and orientations of each object, and identify the assembling features on each object. These sense are always required in order to support sophisticated hand–eye coordination. A possible way to achieve 3D vision is a RGB-D camera.

Task compiler. In production lines, well-documented standard operation procedures (SOPs) and working instructions are necessary in order for human workers to perform their tasks. Thus, the most straightforward task compiler is one that can parse the SOPs and convert them into sequences of correct primitives, and learn the objects from pictures in the SOPs so as to link the objects for each primitive. Another, even simpler, compiler could be a programming pad that allows the field engineering to edit the primitive flow, since the set of primitives is comparatively narrow compared with GIAs for other applications.

Knowledge engine. Here, we propose a set of primitives required for normal assembling procedures, although these may not be a complete set for every assembly task. This set includes:

(1) Inserting and/or gripping part A and sliding its male/female feature to the female/male feature of part B, where possible features are connectors, shafts/holes, and screws/bores;

(2) Mating and/or pressing part A on top of part B and moving around the surface of part A so that the features of part A and part B can be mated together with tight tolerance, where possible features are part geometries that fit with each other, with a transitional or press fit;

(3) Screwing, turning, and installing part A into part B, where features include screw threads and pipe threads.

Industry GIA. Compared with the capability of GIAs required for a broader scope of applications, the GIAs applied in industrial automation will operate in a relatively structured environment with well-defined procedures; thus, they will be less difficult to implement than the GIAs in other applications.

At present, various types of robots and customized equipment have been utilized in modern production lines to manufacture products efficiently, with related technology that has evolved over tens of years. Mature automation works include soldering, painting, and material loading. However, there are still procedures that can only be done by humans, which require more sophisticated material handling that must adapt to the compliance of the parts and non-perfect tolerance control of the components, such as complex assembling. Here, we propose a possible form of industry GIA that is capable of these types of tasks.

8. Conclusion

In this article, we discussed the new concept of the GIA in order to pursue general transferability among tasks. A core primitive flow model and GIA architecture were proposed. We believe that the GIA will significantly promote intelligent science.

References

[1] Shapiro SC, editor. Encyclopedia of artificial intelligence. New York: John Wiley; 1992.
[2] Wiss JF, Parmelee RA. Human perception of transient vibrations. J Struct Div 1974;100(4):773–87.
[3] Odum HT. Environmental accounting: emergy and environmental decision making. New York: Wiley; 1996.
[4] Studer R, Benjamins VR, Fensel D. Knowledge engineering: principles and methods. Data Knowl Eng 1998;25(1–2):161–97.
[5] Hayes JP. Computer architecture and organization. New York: McGraw-Hill; 2002.
[6] Mandlekar A, Zhu Y, Garg A, Booher J, Spero M, Tung A, et al. RoboTurk: a crowdsourcing platform for robotic skill learning through imitation. In: Proceedings of the 2nd Conference on Robot Learning; 2018 Oct 29–31; Zürich, Switzerland; 2018. p. 879–93.
[7] Devlin J, Chang MW, Lee K, Toutanova K. Bert: pre-training of deep bidirectional transformers for language understanding. 2018. arXiv:1810.04805.
[8] Chang AX, Funkhouser T, Guibas L, Hanrahan P, Huang Q, Li Z, et al. ShapeNet: an information-rich 3D model repository. 2015. arXiv:1512.03012.
[9] Morales A, Prats M, Sanz P, Pobil AP. An experiment in the use of manipulation primitives and tactile perception for reactive grasping. In: Proceedings of the 2007 Robotics: Science and Systems; 2007 Jun 27–30; Atlanta, GA, USA; 2007.
[10] Dasari S, Ebert F, Tian S, Nair S, Bucher B, Schmeckpeper K, et al. RoboNet: large-scale multi-robot learning. In: Proceedings of the 3rd Conference on Robot Learning; 2019 Oct 30–Nov 1; Osaka, Japan; 2019.
[11] Fang K, Zhu Y, Garg A, Kurenkoy A, Mehta V, Li FF, et al. Learning task-oriented grasping for tool manipulation from simulated self-supervision. 2018. arXiv:1806.09266.
[12] Li YL, Xu L, Liu X, Huang X, Xu Y, Chen M, et al. HAKE: human activity knowledge engine. 2019. arXiv:1904.06539.
[13] Li YL, Zhou S, Huang X, Xu L, Ma Z, Fang HS, et al. Transferable interactiveness knowledge for human–object interaction detection. In: Proceedings of the 2019 IEEE Conference on Computer Vision and Pattern Recognition; 2019 Jun 16–20; Long Beach, CA, USA; 2019.
[14] Xu D, Nair S, Zhu Y, Gao J, Garg A, Li FF, et al. Neural task programming: learning to generalize across hierarchical tasks. In: Proceedings of the 2018 IEEE International Conference on Robotics and Automation; 2018 May 21–25; Brisbane, QLD, Australia; 2018.
[15] LeCun Y, Bengio Y, Hinton G. Deep learning. Nature 2015;521:436–44.
[16] Kennedy J. Swarm intelligence. In: Zomaya AY, editor. Handbook of nature-inspired and innovative computing. Boston: Springer; 2006. p. 187–219.