Automatic Error Correction of GPT-Based Robot Motion Generation by Partial Affordance of Tool

Takahiro Suzuki, Yuta Ando and Manabu Hashimoto
Graduate School of Engineering, Chukyo University, Aichi, Japan

Keywords: Robot Motion Generation, Automatic Error Correction, Part Affordance, Functional Consistency, Robot Motion Template, GPT.

Abstract: In this research, we proposed a technique that, given a simple instruction such as "Please make a cup of coffee" as would commonly be used when one human gives another human an instruction, determines an appropriate robot motion sequence and the tools to be used for that task and generates a motion trajectory for a robot to execute the task. The proposed method uses a large language model (GPT) to determine robot motion sequences and tools to be used. However, GPT may select tools that do not exist in the scene or are not appropriate. To correct this error, our research focuses on function and functional consistency. An everyday object has a role assigned to each region of that object, such as "scoop" or "contain". There are also constraints such as the fact that a ladle must have scoop and grasp functions. The proposed method judges whether the tools in the scene are inconsistent with these constraints, and automatically corrects the tools as necessary. Experimental results confirmed that the proposed method was able to generate motion sequences a from simple instruction and that the proposed method automatically corrects errors in GPT outputs.

1 INTRODUCTION

Research targeting the development of life-support robots is becoming increasingly active in recent years thanks to major advances in machine learning. The ultimate objective of this research is to achieve a robot that can act appropriately based on simple text such as "Please make a cup of coffee" that would naturally be used to instruct a fellow human being to perform a certain task.

To this end, a number of image processing technologies are being developed and steadily approaching a practical level. These include technologies for recognizing tools such as YOLO(Redmon et al., 2016) and SSD(Liu et al., 2016) technologies(Xu et al., 2021; Qin et al., 2020; Ardon et al., 2020; Liang and Boularias, 2023) for understanding how tools are used, and technologies for determining grasping parameters such as Fast Graspability(Domae et al., 2014) and so on(Song et al., 2020; Zhang et al., 2021).

On the other hand, achieving natural interaction between humans and robots will require technology that enables the robot itself to break down a broad task given by a human and generate a specific sequence of motions and technology that determines the appropriate tool to be used for each motion. These technologies, however, must be highly versatile in use, and as a result, they have not been extensively studied(Yamada et al., 2020; Takata et al., 2022; Inagawa et al., 2020). These methods require detailed instructions from a human at the motion level.

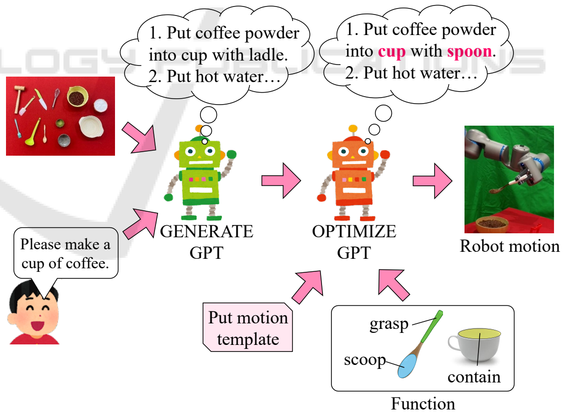


Figure 1: Overview of proposed method. Generate robot motions from scene information and task-level instruction given by a human. Determine sequence of motions and tools to be used by GPT. Additionally, use information on tool functions and a newly proposed motion template to optimize the GPT tools. Finally, create robot motions from that information.

Suzuki, T., Ando, Y. and Hashimoto, M.
Automatic Error Correction of GPT-Based Robot Motion Generation by Partial Affordance of Tool.
DOI: 10.5220/0012375000003660

In *Proceedings of the 19th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2024) - Volume 4: VISAPP*, pages 314-323
ISBN: 978-989-758-679-8; ISSN: 2184-4321

Here, we take as an example the task of "making coffee." This task consists of the four subtasks of (1) put coffee powder into cup, (2) put hot water into cup, (3) put sugar into cup, and (4) stir. In addition, subtask (1) "put coffee powder into cup" can be further divided into smaller motions such as (1a) pick a spoon, (1b) scoop coffee powder from bowl, (1c) put scooped coffee powder into cup, and (1d) place spoon down.

Now, on examining the above relationships, it can be seen that generating specific robot motions from a simple instruction given by a human would require that the hierarchical relationship of task $\rightarrow$ subtask $\rightarrow$ motions be understood and that those motions be put into the correct order.

In addition, subtask (1) "put coffee powder into cup" requires that the tools to be used be correctly determined, as in "use a spoon" for scooping and use a "cup" for containing the scooped coffee powder.

Based on the above, this research proposes a method for generating a specific motion trajectory for a robot from a task-level instruction given by a human by appropriately determining the robot's motion sequence for executing that task and the tools to be used.

The main flow of this method is shown in Figure1. Here, to determine a motion sequence at the subtask level from an instruction given by a human and to determine the tools to be used, we use Generative Pretrained Transformer (GPT), a large language model having wide applicability to the content of many fields that has been extensively researched in recent years (GENERATE GPT).

However, since GPT outputs the motion sequence and used tools as a character string, that in itself cannot generate robot motions. We therefore propose a motion template to connect motions (character string) at the subtask level with actual robot motions.

In addition, GPT currently under development cannot necessarily determine the appropriate tools to be used. For example, for the motion of "put coffee powder," GPT may actually propose that the coffee be scooped into a natsume (a small lacquered container used in a Japanese tea ceremony) or poured into a bowl. GPT may also propose tools that do not exist in the scene. To solve this problem, we use functionality obtained from the results of sensing a scene, where "function" refers to the role of each part of a tool. Specifically, the method judges whether the tools to be used as determined by GPT are functionally consistent, and if they are not, determines the optimal tools from other tools (OPTIMIZE GPT).

The contributions of this research are as follows.

- Achieved the ability of generating an actual motion trajectory for a robot from a simple task-level instruction.
- Proposed a motion template for connecting the instruction given by a human with the robot's motion trajectory and used tools.
- Made it possible to set functional constraints on the tools used in the motion template and to judge whether the tools set by GPT are appropriate from a functional perspective.
- Enabled GPT to select optimal tools for the motions under consideration by searching candidates for tools based on function.

2 PROBLEM FORMULATION

In this section, we formulate the problem to be solved by this research and define the assumptions made in solving that problem. We also explain the basic ideas behind solving the problem.

2.1 Definition

Given the input of a scene's RGB image and point cloud and a task-level instruction such as "Please make some hot coffee," the goal of this research is to determine an appropriate robot motion sequence for executing that task and the tools for doing so and to generate a motion trajectory for a robot hand to execute the task. The motion trajectory is defined in terms of the position and orientation (six degrees of freedom) of the robot hand in 3D space.

2.2 Assumptions

In this research, the following two assumptions are made.

- The subtask-level motions that the robot can make are set beforehand.
- The correspondence between the ingredient contained in a tool and the name of that tool is known.

In the first assumption, the subtask-level motions that the robot can make such as "pick," "place," and "put" are set beforehand. In other words, the robot cannot perform any motions other than the ones specifically set beforehand.

In the second assumption, in the case, for example, that coffee powder is contained in a bowl, that correspondence is set beforehand so that coffee powder comes to mind given a bowl. In this regard, recent progress in generic object recognition technology has made it possible to recognize a bowl and coffee separately. Nevertheless, to reduce the load associated with generating teacher data and reduce the number of classes to be identified, this research recognizes only tools.

2.3 Ideas Behind Generating Robot Motions from a Language Instruction

Generating robot motions from a task-level language instruction requires that the meaning expressed by the character string in the language instruction be understood, that the tools to be used and the motion sequence be determined from that meaning, and that the motion trajectory of the robot hand in 3D space be generated. Here, to understand all sorts of language instructions and determine the motion sequence and tools to be used, we use GPT, a large language model, the research of which has been making remarkable progress in recent years. Specifically, this means inputting the language instruction and information on the objects in the target scene into GPT to determine the motion sequence that enumerates the motions at the subtask level and the tools to be used.

However, actual robot motions cannot be generated simply on the basis of enumerating the motion sequence and tools to be used. We therefore propose the concept of a motion template to connect subtask-level motions and actual motions. This is the first idea characterizing this research. Details are described in section 3.3, but the role of a motion template is to assign a correspondence between the name of the subtask-level motion and the motion trajectory, the motion-source/motion-target tools, and the tool used for that motion.

The second novel idea of this research is to determine whether the tools determined by GPT are appropriate based on information on the functions. Here, ”function” is the role played by each tool, as described in detail in section 3.3. In this research, our method determines whether a tool determined by GPT is appropriate from the perspective of information on tool functionality, and if inappropriate, feeds back that information to GPT to determine another tool.

The third novel idea of this research is to enumerate tools considered to be good candidates from the perspective of information on tool functionality in addition to the tool determined by GPT and to then have GPT select the most optimal tool from those tools. Assuming, for example, that ”ladle” has been selected as the tool for scooping up coffee powder. In this case, since a ladle possesses a ”scoop” function, it can scoop up something, but on the other hand, it is not generally used as a tool for scooping up coffee powder. The method therefore searches for all tools having a scoop function (such as spoon and ladle) from the input scene and feeds back search results to GPT. GPT then judges whether a ladle is the most optimal tool for scooping up coffee powder from among the tools returned from the search, and if not, selects the most optimal tool from other candidates.

3 METHOD

In this section, we describe our method for generating robot motion from a language instruction.

3.1 Overview

The flow of the proposed method is shown in Figure2. This method consists of two modules. The first module determines a subtask-level motion sequence and used tools from the input RGB image of a scene and a task-level language instruction. The second module determines whether the determined tools are optimal, and generates a motion trajectory for the robot hand.

3.2 Determining Motion Sequence and Tools

The green frame in Figure2 outlines this module, which determines a subtask-level motion sequence and the tools to be used at that time from a scene's RGB image and a task-level language instruction.

This module begins by recognizing the names of the tools present in the scene using a generic object recognition method like YOLO(Redmon et al., 2016). Since it is assumed here that a correspondence between objects and ingredients is already known, ingredients present in the scene can also be recognized.

Next, the module inputs scene information, a task-level language instruction like ”make hot coffee,” and the types of subtask-level motions that the robot can perform (put, stir, etc.) into GPT. It also submits to GPT a query requesting ”a motion sequence for executing that task using available motions and a list of tools to be used at that time” and outputs a motion sequence and used tools.

3.3 Correct Tools and Generate Motion Trajectory

This module, shown by the purple frame in Figure2, determines whether the determined tools are optimal

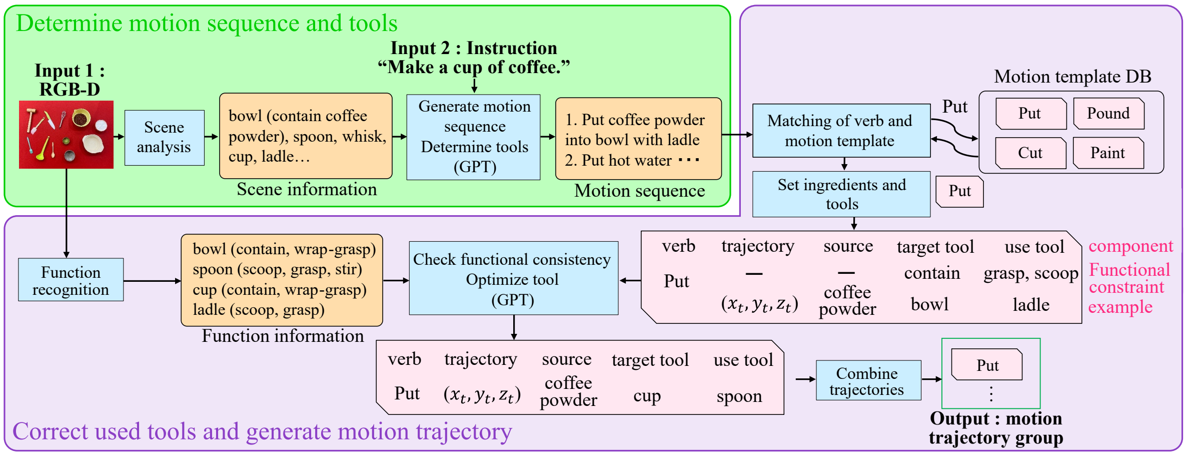


Figure 2: Flow of proposed method. Input RGB-D data and task-level instruction, output motion trajectory group. This method consists of a module for generating a motion sequence and tools to be used from input data (green frame) and a module that corrects those tools and generates a group of motion trajectories (purple frame).

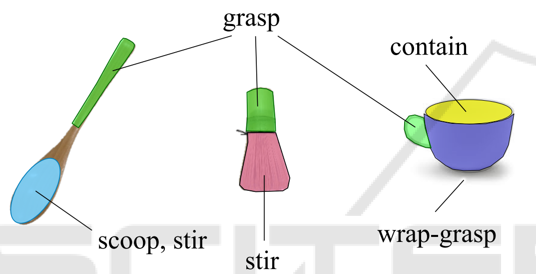


Figure 3: Tool-function information. Each part of a tool has a specific role. In the case of a spoon, the handle has the role of being grasped by a human or robot (grasp function) and the concave part has the role of scooping up something (scoop function) and the role of stirring (stir function).

using a motion template, and generates a motion trajectory for the robot hand. We first explain the tool-function information and motion templates that are used here and then describe the method for correcting tools and generating a motion trajectory for the robot hand using the above.

3.3.1 Tool-Function Information

Everyday objects (tools) such as spoons and cup have roles like ”scoop” and ”contain” assigned to each of their parts. Such a role is called a function or affordance, which is a concept proposed by Myer et al(Myers et al., 2015). Figure3 shows the functions assigned to a spoon and cup.

A designer who creates an everyday object envisions the motion used for that item and reflects functionality in that item. For example, a whisk is created envisioning a motion that stirs some kind of liquid. In other words, a human grasps the whisk in the domain of the grasp function and stirs the liquid in the domain of the stir function, which shows that function and

(a) Overview of a motion template

verb	trajectory	source	target tool	use tool

(b) Constraints set in a motion template (Example of put)

put	—	—	contain	grasp, scoop

(c) Example of tools and ingredient set in a motion template (put)

put	(x_t, y_t, z_t)	coffee-powder	cup	spoon

Figure 4: Proposed motion templates. A motion template consists of a verb, motion trajectory, motion source, tool targeted by motion, and used tool (a). Functional constraints are set on tools (b).

motion are highly related. A spoon, meanwhile, must have scoop, stir and grasp functions, which reflects the close relationship between tool and function. To enable robots to use tools, a number of methods for recognizing the affordances of everyday objects are being researched(Do et al., 2018; Minh et al., 2020).

3.3.2 Motion Templates

Figure4 shows an overview of the proposed motion template. A motion template associates the name of a motion and a motion trajectory. This motion template has been proposed by Tsusaka et al(Tsusaka et al., 2012). Applying this method, for example, to the ”put” motion of scooping up something and pouring it into another tool, if the domain of the scoop function is given a certain movement, the motion can be successfully executed regardless of the size or shape of the tool. In addition, tool-function information can be used here to determine the point at which the robot should grasp that tool. In other words, given

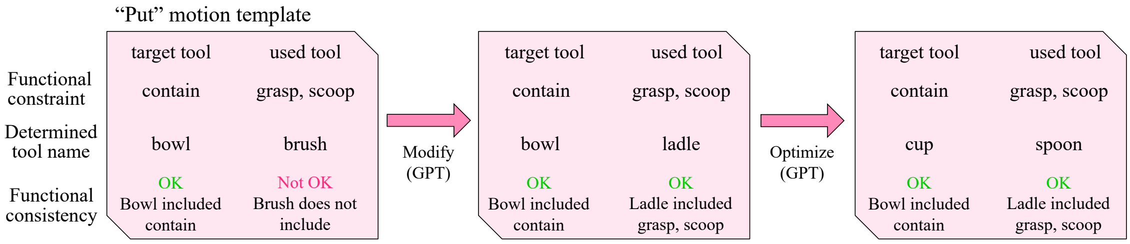


Figure 5: Modification of tools used in a motion template. Shown here is an example of a "put" motion targeting coffee powder for the case of a cup, spoon, ladle, etc. present as other tools. Since a brush has no scoop function and lacks functional consistency as a result, GPT changes this to a tool having a scoop function (indicated by Modify in the figure). Finally, GPT selects an optimal tool from among tools having the same function (Optimize).

the movement of the scoop function, the movement of the grasp point for producing that movement (i.e., the motion trajectory of the robot hand) can be generated.

However, an actual "put" motion cannot be achieved solely on the basis of that information. Also needed is information on what is to be scooped, where it should be poured, and what tool should be used to do the scooping. For this reason, we add to the motion template the components "source" (the target of scooping in the case of a put motion), "target tool" (the destination of pouring the target of scooping in the case of a put motion), and "used tool" (the tool used when executing the motion). We propose a novel motion template with these additional terms. Then, once an appropriate ingredient or tool has been set for each of those items, the motion becomes possible.

In addition to the above, the tools set in "target tool" and "used tool" must satisfy functional constraints. For a "put" motion, the "target tool" must be able to contain something. In short, a tool having a contain function must be set here. The "used tool," meanwhile, must have a grasp function so that the robot can grasp the tool and a scoop function for scooping up something. In this way, the tools set in "target tool" and "used tool" must have specific functions. This is a condition that differs for each type of motion, so the functions that "target tool" and "used tool" must have can be set beforehand for each type of motion template.

3.3.3 Generation of Robot Motion Using Tool-Function Information and Motion Template

The purple frame in Figure2 shows an overview of this module. This module optimizes the tools to be used based on the character string output by the previous processing step expressing a motion sequence and the tools to be used as well as on tool-function information. It also outputs a motion trajectory that the robot can perform.

The tool-function recognition process removes the plane surface of the desk, eliminates noise, etc. with respect to the scene's point cloud data and extracts a point cloud for each tool. Here, we use PointNet++(Qi et al., 2017) for this model. This processing creates a database that associates tool names and functions.

In motion template matching, the process extracts a verb from character string and selects a motion template corresponding to that verb. Furthermore, as described in section 2.2, essential subtask-level motions are set beforehand in the motion template database. Ingredient and tools corresponding to source, target tool, and used tool are also extracted from the character string. Here, for the "put" motion, the process outputs a character string in the form "Put [source name] into [target tool name] with [used tool name] by GPT for generating a motion sequence and determining the tools to be used.

To check the functional consistency of tools, this module judges whether the tools set in target tool and used tool are functionally correct as shown by the Modify section in Figure5. As described in section 3.3.2, the tools set in target tool and used tool must each have a specific function, which differs for each type of motion. The module judges whether a set tool has this function using a function database. For example, let's assume that GPT determines that a brush is to be used for the put motion as shown in Figure5. A brush, however, has no scoop function and is therefore not functionally consistent. In such a case, names of tools having that function will be extracted from the function database and fed back to GPT to extract an appropriate tool from those names. At this time, GPT selects an optimal tool from the tools presented. This process is repeated until a functionally appropriate tool is set.

In tool optimization, the module judges whether the tools set in target tool and used tool are optimal tools as shown by the Optimize section in Figure5. For the case of "put coffee powder," for example, the put motion template is used. The functions that the

Table 1: Motion templates used in the experiment and functional constraints of the tools.

Motion	Target tool	Use tool
Put	Contain	Grasp, Scoop
Stir	Contain	Grasp, Stir
Paint	-	Grasp, Paint
Pound	Contain	Grasp, Pound
Cut	Contain	Grasp, Cut

used tool should have in this template are the grasp function and scoop function. As a result, either ”ladle” or ”spoon” is acceptable for the used tool. However, ”spoon” is generally used. In this way, whenever there are multiple tools in the scene having the same function as the functional constraint set in the motion template, the names of those tools are fed back to GPT to extract from those names an optimal tool. That is, GPT selects an optimal tool to be used at the time of that motion from the presented tools.

At this point in time, the ingredient and tool names corresponding to source, target tool, and used tool are known, so the trajectories of the motion template will be transformed using information on the locations of that ingredient and tools and added to the motion trajectory group. This series of processes is performed only for each character string showing the motion sequence and used tools output by the previous processing step. Finally, these motion trajectories are combined according to the motion sequence.

4 EVALUATION EXPERIMENTS

This section describes experiments to evaluate the proposed method. These consist of an experiment in determining robot motion sequence and tools to be used using virtual data, an experiment in determining robot motion sequence and tools to be used using tools obtained from an actual scene and tool-function information, an experiment in correcting used tools from an intentionally incorrect recipe, and an experiment in generating motion using an actual robot.

4.1 Determining Robot Motion Sequence and Used Tools Using Virtual Data

4.1.1 Setup

This experiment was performed to evaluate whether a robot motion sequence and used tools could be correctly determined by the proposed technique.

The scene used here is assumed to include spoon, natsume, chawan, whisk, chagama, ladle, bowl, hammer, knife, rice scoop, pot and cutting board as tools and matcha powder, hot water, water, curry powder, rice, salt coffee powder, sugar, pineapple and chocolate bar as ingredients. In addition, used functions are cut, contain, scoop, grasp, wrap-grasp, stir, pound, and paint. This experiment uses virtual data, so it is assumed that the functions, tools, etc. have been correctly recognized from the scene. The motion templates used here are shown in Table 1 and the tasks given by a human are ”Make hot matcha,” ”Make hot coffee,” ”Make curry rice,” ”Make pineapple juice,” and ”Make liquid chocolate.” GPT-4, GPT-3.5 Turbo was used as a GPT engine.

In the experiment, the proposed technique was used to determine a robot motion sequence and used tools from tool and ingredient information in the scene and from an instruction given by a human. Ten trials were performed for each task. Here, a human judged and evaluated whether the technique could determine a motion sequence and used tools that could successfully execute the task given by a human.

4.1.2 Experimental Result

The results in Table2 confirm that the proposed method can be used to correct the tool to be used. For the task of making hot coffee, GPT selected a suboptimal tool. As an example, whisk was selected when stirring coffee powder. However, the proposed method corrected it into a spoon with the same functionality. For the task of making curry rice, GPT generated a sequence that made it impossible to make curry. For example, a recipe was generated in which curry powder and water are put in a pot and rice is put in a bowl. In addition, a suboptimal tool was selected four times. With the proposed method, these were corrected into optimal tools. However, it was not possible to correct the recipe to make curry. This is because GPT optimizes the tools for each sequence. In other words, GPT does not consider the relationship between successive sequences. In actual cooking, the tools to be used must be determined while also taking into consideration the sequences that come before and after. Therefore, to solve this error, we need a system that optimizes the tools while also considering the content of successive sequences.

Figure6 shows the communication between the system and GPT in text form. These sentences are generated automatically. GPT and the system can interact with each other to ensure that they correct any errors to achieve the optimal tool for the task.

Table 2: Errors in the selection of tools to be used by GPT and their correction by the proposed method.GPT may not be able to select the optimal tool to be used.With the proposed method, these are corrected into optimal tools.

Product	Number of successful tools determined by GPT	Erroneously determined tools by GPT		Correction by proposed method	
		Number of times	Example	Number of successes	Example
Hot matcha	7	3	1. Impossible to make it: 2 2. Selecting suboptimal tools: 1	1	Successfully corrected error No.2
Hot coffee	9	1	1. Selecting suboptimal tools: 1	1	Successfully corrected all error
Curry rice	5	5	1. Impossible to make it: 1 2. Selecting suboptimal tools: 4	4	Successfully corrected error No.2
pineapple juice	2	8	1. Impossible to make it: 2 2. Selecting suboptimal tools: 6	6	Successfully corrected error No.2
Liquid chocolate	3	7	1. Impossible to make it: 2 2. Selecting suboptimal tools: 5	5	Successfully corrected error No.2

Table 3: Experimental result using actual scenes.

Product	Number of successful tools determined by GPT	Erroneously determined tools by GPT		Correction by proposed method	
		Number of times	Example	Number of successes	Example
Hot matcha	10	0	-	-	-
Hot coffee	7	3	1. Selecting suboptimal tools: 3	3	Successfully corrected all errors
Curry rice	3	7	1. Impossible to make curry: 3 2. Selecting suboptimal tools: 4	4	Successfully corrected error No.2

4.2 Determining Robot Motion Sequence and Used Tools Using Obtained from Scene and Tool-Function Information

4.2.1 Setup

This experiment was performed to evaluate whether a robot motion sequence and used tools could be correctly determined using the results of recognizing tools and their functions from an actual scene.

In the experiment, the Intel RealSense Depth Camera SR305 was used as a sensor. GPT-4, GPT-3.5 Turbo was used as a GPT engine. The tools used here were a spoon, hammer, ladle, brush, whisk, *chawan* (a Japanese tea bowl), bowl, *natsume*, *chagama* (a tea kettle used in a Japanese tea ceremony), pot, and dish. The ingredients were water, salt, sugar, coffee powder, curry powder, and rice (only for the task of preparing curry) contained in a bowl, matcha powder contained in a *natsume*, hot water contained in a bowl, and hot water contained in a *chagama* (for the tasks of preparing matcha). These items were randomly placed in the target scene. Recognized functions are cut, contain, scoop, grasp, wrap-grasp, stir, pound, and paint. The motion templates used here were the same as those of the experiment in section 4.1. In this experiment, a human gave three tasks: ”Make hot matcha,” ”Make hot coffee,” and ”Make curry rice.” The experimental procedure consisted of recognizing tools and functions from the scene’s RGB image and point cloud and determining the robot motion sequence and used tools from the above and the instruction given by the human. Ten trials were performed for each task. A human judged and evaluated whether the method could determine a motion sequence and used tools that could successfully execute the task given by the human.

Training for YOLO used in tool recognition targeted the tools described above. Training data con-

GPT initially generated motion sequences and tools

```
GPT: 1.  Put chocolate bar into bowl with spoon.
GPT: 2.  Put hot water into pot with ladle.
GPT: 3.  Stir chocolate bar in bowl with spoon.
GPT: 4.  Pound chocolate bar in bowl with hammer.
```

Check consistency and result of correcting tools (sequence 1)
・・・ Consistency : OK, correcting required

```
INPUT:  Put chocolate bar into bowl with spoon.
  SYSTEM: Functional consistency of target tool -> OK (Bowl includes contain function)
  SYSTEM: Consistency of used tool -> OK (Spoon includes scoop, grasp functions)
  SYSTEM: You have selected the Bowl. Is it the best selection?
  SYSTEM: Other tools with the same functionality are chawan, natsume, and bowl.
  GPT:    The best tool to put the chocolate bar into is a bowl.
  SYSTEM: You have selected the spoon. Is it the best selection?
  SYSTEM: Other tools with the same functionality are spoon and ladle.
  GPT:    A ladle is generally used to put liquid chocolate into a bowl.
OUTPUT: Put chocolate bar into bowl with ladle.
```

Check consistency and result of correcting tools (sequence 4)
・・・ Consistency : OK, correcting not required

```
INPUT:  Pound chocolate bar in bowl with hammer.
  SYSTEM: Functional consistency of target tool -> OK (Bowl includes contain function)
  SYSTEM: Consistency of used tool -> OK (Hammer includes pound, grasp functions)
  SYSTEM: You have selected the bowl. Is it the best selection?
  SYSTEM: Other tools with the same functionality are chawan, natsume and bowl.
  GPT:    Bowl.
OUTPUT: Pound chocolate bar in bowl with hammer.
```

Final output

```
1. Put chocolate-board into bowl with ladle.
2. Put hot-water into pot with ladle.
3. Stir chocolate-board in bowl with spoon.
4. Pound chocolate-board in bowl with hammer.
```

Figure 6: Experimental result using virtual data. Only sequence 1 and 4 are described in the communication. This is the communication in text form.These sentences are generated automatically.The motion sequence and tools initially generated by GPT are not optimal.In sequence 1, it is not appropriate to use a spoon, but we can see that it is corrected to ladle by the proposed method.

sisted of 8892 images with those objects randomly placed and annotations indicating the corresponding positions and types of those objects. Additionally, in the training for PointNet++ used in function recognition, actual objects were scanned using a 3D scanner to create 3D models. A human then added annotations indicating tool function to those models. Finally, 2.5D data observed from multiple viewpoints were created from these 3D models. Here, 745 pairs of this training data and teacher data were created and used for training purposes.

4.2.2 Experimental Result

Experiment results are shown in Table3. The view of this table is the same as Table2.

The results in Table3 confirm that the proposed method can be used to correct the tool to be used. The same trend as in section 4.1.2 was observed when actual scenes were used. The proposed method was particularly effective in the task of “make hot coffee.” We confirmed that object recognition and function recognition were highly accurate in all scenes. Therefore, the tool should be determined while considering the sequence before and after.

GPT initially generated motion sequences and tools (intentionally incorrect)

```
GPT: 1.  Put matcha powder into brush.
GPT: 2.  Put hot water into brush with brush.
GPT: 3.  Stir matcha powder in chagama with ladle.
```

Check consistency and result of correcting tools (sequence 1)

```
INPUT:  Put matcha powder into brush.
  SYSTEM: Functional consistency of target tool -> NG (Brush does not include
          contain function)
  SYSTEM: Consistency of used tool -> NG (No tool that includes scoop, grasp functions)
  SYSTEM: Which tool should you put matcha powder into?
  GPT:    Chawan.
  SYSTEM: What tool is generally used to put matcha powder into a chawan?
  GPT:    The tool commonly used to put matcha powder into a chawan is a spoon.
  SYSTEM: Functional consistency of target tool -> OK (Chawan includes contain function)
  SYSTEM: Consistency of used tool -> OK (Spoon includes scoop, grasp functions)
  SYSTEM: You have selected the chawan. Is it the best selection?
  SYSTEM: Other tools with the same functionality are chawan, natsume and bowl.
  GPT:    The best tool to put matcha powder into is the chawan.
  SYSTEM: You have selected the spoon. Is it the best selection?
  SYSTEM: Other tools with the same functionality are spoon and ladle.
  GPT:    A spoon is generally used to put matcha powder into a chawan.
OUTPUT: Put matcha powder into chawan with spoon.
```

Check consistency and result of correcting tools (sequence 2)

```
INPUT:  Put hot water into brush with brush.
  SYSTEM: Functional consistency of target tool -> NG (Brush does not include
          contain function)
  SYSTEM: Consistency of used tool -> NG (Brush does not include scoop, grasp functions)
  SYSTEM: Which tool should you put hot water into?
  GPT:    Bowl.
  SYSTEM: What tool is generally used to put matcha powder into bowl?
  GPT:    Ladle.
  SYSTEM: Functional consistency of target tool -> OK (Bowl includes contain function)
  SYSTEM: Consistency of used tool -> OK (Ladle includes scoop, grasp functions)
  SYSTEM: You have selected the bowl. Is it the best selection?
  SYSTEM: Other tools with the same functionality are chawan, natsume and bowl.
  GPT:    The best tool to stir matcha powder into is the chawan.
  SYSTEM: You have selected the ladle. Is it the best selection?
  SYSTEM: Other tools with the same functionality are spoon and ladle.
  GPT:    A ladle is generally used to put water into a chawan.
OUTPUT: Put hot water into chawan with ladle.
```

Final output

```
1. Put matcha powder into chawan with spoon.
2. Put hot water into chawan with ladle.
3. Stir matcha powder in chawan with whisk.
```

Figure 7: Communication between the system and GPT. The motion sequences and tools are intentionally generated incorrectly. Only sequence 1 and 2 are described in the communication. In sequence 2, tools that do not lack functional consistency are selected after which the best tool is finally selected.

4.3 Correction of Used Tools from an Intentionally Incorrect Recipe

4.3.1 Setup

For the case that GPT makes a mistake in tools to be used, this experiment was performed to test whether those tools could be corrected using tool-function information and GPT.

Experimental conditions were the same as those in section 4.2. Here, instead of a mistake made by GPT, used tools incorrectly stated by a human (e.g., ”Put coffee powder into a ladle with a brush.”) were specified and the proposed method attempted to correct the used tools. Ten trials were performed for each task. A human judged and evaluated whether the method could determine used tools that could successfully execute the task given by a human.

4.3.2 Experimental Result

The experimental result showed that the tools used were successfully corrected 8 out of 10 times. Figure7 shows the actual communication in which the tools to be used are corrected. In this experiment, humans intentionally specify the incorrect tool to use. In Sequence1, the tool is judged to lack functional consistency. A chawan and a spoon were selected as alternative tools, and they were judged to be optimal compared to the other tools. Sequence2 is also determined to lack functional consistency. A bowl and ladle were then selected as alternative tools. In the optimization, a chawan was determined to be optimal compared to the bowls.

One of the reasons for this failure was the erroneous determination of the tools into which the ingredients were put. As mentioned in section 4.1.2, this is because GPT optimizes the tools for each sequence.

4.4 Correction of Used Tools from an Intentionally Incorrect Recipe

4.4.1 Setup

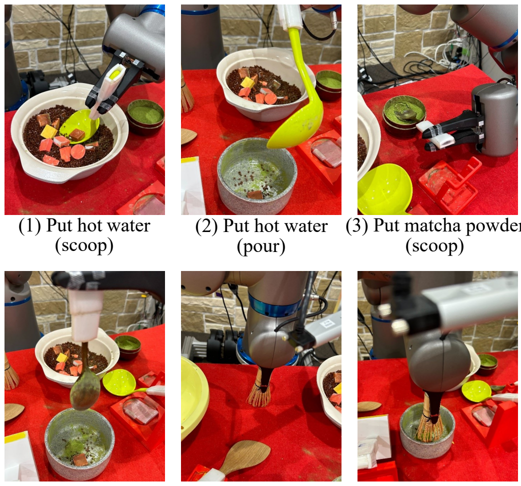
This experiment was performed to evaluate whether an actual robot could move according to an instruction given by a human using the robot motion trajectory determined by the proposed method. In the experiment, the Intel SR305 was used as a sensor and Torobo from Tokyo Robotics Inc. was used as the robot. GPT-4, GPT-3.5 Turbo was used as a GPT engine. The tools used here were bowl, whisk, spatula, brush, hammer, chawan, ladle, and spoon, and the ingredients were coffee powder, curry powder, hot water, water, sugar, and salt contained in a bowl and matcha powder contained in a natsume. These items were randomly placed in the target scene. The recognized functions were the same as those used up to now. The motion templates were the same as those of the experiment in section 4.1. The task given by the human was "Make hot matcha." The experimental procedure consisted of recognizing tools and functions from the scene's RGB image and point cloud and determining the robot motion sequence and used tools from the above and the instruction given by the human. A motion trajectory was generated and actual robot motion was performed. Ten trials were performed. A human judged and evaluated whether the task given by the human could be successfully executed.

Training conditions for YOLO and PointNet++ were the same as those in section 4.2.

4.4.2 Experimental Result

In the experiment, the robot succeeded in making hot matcha 8 out of 10 times. Figure8 shows the robot in actual operation.

The proposed method successfully determined the motion sequences and tools used 10 out of 10 times.



(1) Put hot water (scoop) (2) Put hot water (pour) (3) Put matcha powder (scoop)

(4) Put matcha powder (pour) (5) Stir hot water (pick) (6) Stir hot water (stir)

Figure 8: Robot operating state. There are a snapshots of some of the robot's movements. For easy visualization, brown beads are used instead of hot water. The robot is making hot matcha based on the motion sequences and the tools to be used generated by the proposed method.

However, two errors occurred when generating robot paths. This is because the robot's range of motion is small and the robot collides with other objects. Therefore, the motion range of the robot must also be considered when determining the motion trajectory in the motion template. It is also effective to consider the range of motion and modify the trajectory to a motion trajectory suitable for the robot.

5 CONCLUSIONS

In this research, we proposed a method that, given a simple instruction such as "Please make a cup of coffee" as would commonly be used when one human gives another human an instruction, determines an appropriate robot motion sequence and the tools to be used for that task and generates a motion trajectory for a robot to execute the task.

The proposed method uses GPT to determine the robot's motion sequence and tools to be used from a given instruction. In addition, we proposed a method that judges whether the tools determined by GPT are optimal based on the consistency of tool-function information and that provides feedback for selecting other tools if not optimal.

The experimental results confirm that the proposed method can be used to correct the tool to be used. However, GPT generated a sequence that made it impossible to make product. An example of a task to make curry, a recipe was generated in which curry powder and water are put in a pot and rice is put in a bowl. Therefore, to solve this error, we need a system that optimizes the tools while also considering the content of successive sequences.

In future work, we propose a method for optimizing the motion sequence for performing the task, and correcting the tools to be used to take into account the motion sequence before and after.

ACKNOWLEDGEMENTS

This paper is based on results obtained from a project, JPNP20006, commissioned by the New Energy and Industrial Technology Development Organization (NEDO).

REFERENCES

Ardon, P., Pairet, E., Petrick, R., Ramamoorthy, S., and Lohan, K. (2020). Self-assessment of grasp affordance transfer. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9385–9392.

Do, T., Nguyen, A., and Reid, I. (2018). Affordancenet: An end-to-end deep learning approach for object affordance detection. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, pages 5882–5889.

Domae, Y., Okuda, H., Taguchi, Y., Sumi, K., and Hirai, T. (2014). Fast graspability evaluation on single depth maps for bin picking with general grippers. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, pages 1997–2004.

Inagawa, M., Takei, T., and Imanishi, E. (2020). Japanese recipe interpretation for motion process generation of cooking robot. In *Proceedings of IEEE/SICE International Symposium on System Integration (SII)*.

Liang, J. and Boularias, A. (2023). Learning category-level manipulation tasks from point clouds with dynamic graph cnns. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA))*, pages 1807–1813.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C., and Berg, A. (2016). Ssd: Single shot multibox detector. In *Proceedings of European conference on computer vision (ECCV)*, pages 21–37.

Minh, C., Gilani, S., Islam, S., and Suter, D. (2020). Learning affordance segmentation: An investigative study. In *Proceedings of International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, pages 2870–2877.

Myers, A., Teo, C., Fermüller, C., and Aloimonos, Y. (2015). Affordance detection of tool parts from geometric features. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, pages 1374–1381.

Qi, C. R., Yi, L., Su, H., and Guibas, L. J. (2017). Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Proceedings of Advances in Neural Information Processing Systems (NIPS)*.

Qin, Z., Fang, K., Zhu, Y., Fei-Fei, L., and Savarese, S. (2020). Keto:learning keypoint representations for tool manipulation. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, pages 7278–7285.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once:unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 779–788.

Song, S., Zeng, A., Lee, J., and Funkhouser, T. (2020). Grasping in the wild: Learning 6dof closed-loop grasping from low-cost demonstrations. In *IEEE Robotics and Automation Letters*, volume 5, pages 4978–4985.

Takata, K., Kiyokawa, T., Ramirez-Alpizar, I. G., Yamanobe, N., Wan, W., and Harada, K. (2022). Efficient task/motion planning for a dual-arm robot from language instructions and cooking images. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 12058–12065.

Tsusaka, Y., Okazaki, Y., komatsu, M., and Yokokohji, Y. (2012). Development of in-situ robot motion modification method by hand-guiding instruction. In *Transactions of the JSME (in Japanese)*, pages 461–467.

Xu, R., Chu, F.-J., Tang, C., Liu, W., and Vela, P. (2021). An affordance keypoint detection network for robot manipulation. In *IEEE Robotics and Automation Letters*, volume 6, pages 2870–2877.

Yamada, T., Murata, S., Arie, H., and Ogata, T. (2020). Representation learning of logic words by an rnn: from word sequences to robot actions. In *Frontiers in neurorobotics*, volume 11.

Zhang, X., Koyama, K., Domae, Y., Wan, W., and Harada, K. (2021). A topological solution of entanglement for complex-shaped parts in robotic bin-picking. In *Proceedings of IEEE International Conference on Automation Science and Engineering (CASE)*, pages 461–467.