

SIGVerse シミュレーター一般公開用
サンプルアプリケーションの開発
成果物

お好み焼き協調料理エージェント
チュートリアル

平成 22 年 2 月

株式会社 とめ研究所

目次

1. はじめに.....	3
1.1. お好み焼き協調料理の目的.....	3
1.2. お好み焼き協調料理の概念.....	4
1.3. お好み焼き協調料理の手順.....	6
1.3. お好み焼き協調料理のゴール.....	10
2. お好み焼き協調料理の起動.....	11
2.1. お好み焼き協調料理の起動(サーバ側).....	12
2.2. お好み焼き GUI の起動(クライアント側).....	13
2.3. お好み焼き CUI の起動(クライアント側).....	14
2.4. SIGViewer の起動.....	15
3. お好み焼き協調料理の操作.....	18
3.1. お好み焼き協調料理の操作.....	19
3.2. 知覚に関する操作.....	25
3.3. 力学に関する操作.....	27
3.4. 対話に関する操作.....	34
4. お好み焼き協調料理の設定.....	36
4.1. 状態遷移情報.....	36
4.2. 全ての登場人物.....	37
4.3. 状態遷移条件.....	37
4.4. 状態遷移処理.....	39
4.5. お好み焼き協調料理の設定.....	40
4.2. 知覚に関する設定.....	93
4.3. 力学に関する設定.....	98
4.1. 対話に関する設定.....	102
5. お好み焼き協調料理の改造.....	104
5.1. お好み焼き協調料理の機能.....	104
5.2. SIGVerse の提供する API.....	105
5.2. お好み焼き協調料理の構造.....	106
5.3. お好み焼き協調料理の処理.....	114
5.4. お好み焼き協調料理のコンパイル.....	121
5.5. お好み焼き協調料理の知覚の改造.....	123
5.6. お好み焼き協調料理の力学の改造.....	125
5.7. お好み焼き協調料理の対話の改造.....	128
A1. お好み焼き協調料理のインストール.....	130
A1.1. お好み焼き協調料理のインストール(サーバ側).....	130
A1.2. お好み焼き GUI のインストール(クライアント側).....	133
A1.3. お好み焼き CUI のインストール(クライアント側).....	134
A1.4. SIGViewer のインストール(クライアント側).....	134

A2. トラブルシューティング.....	135
A2.1. 起動に関するトラブルシューティング.....	135
A2.2. 操作に関するトラブルシューティング.....	138
A2.3. 設定に関するトラブルシューティング.....	138
A2.4. 改造に関するトラブルシューティング.....	140
A3. 状態遷移に関する記述一覧.....	142
A3.1. 登場人物一覧.....	142
A3.2. 状態遷移条件.....	143
A3.3. 状態遷移処理.....	144
A4. お好み焼き協調料理ソフトウェアの俯瞰.....	146
A4.1. お好み焼き協調料理のディレクトリ構造.....	146
A4.2. お好み焼き協調料理ソフトウェア一覧.....	147
A4.3. お好み焼き協調料理の主たる操作.....	147

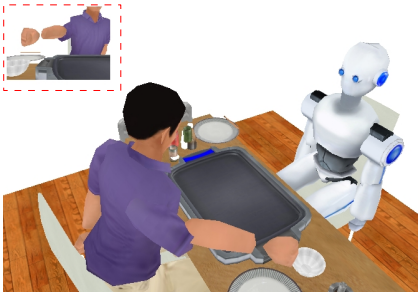
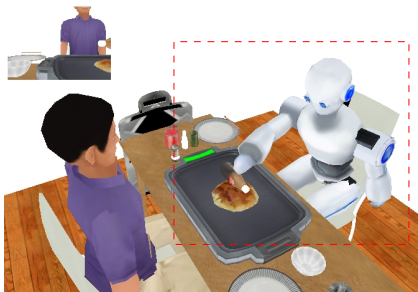


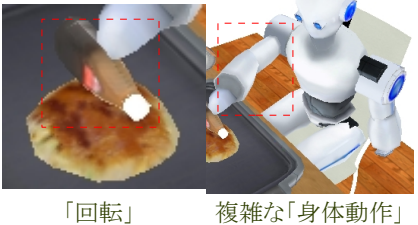

1. はじめに

Contents

- ・はじめに
 - －お好み焼き協調料理の目的
 - －お好み焼き協調料理の概念
 - －お好み焼き協調料理の手順
 - －お好み焼き協調料理のゴール

The Okonomiyaki cooperation dish is an application of one of the applications it using the SIGVerse platform of the sample. The SIGVerse platform has three excellent abilities. It is a simulation ability of "Perception", "Mechanics" and "Conversation". The Okonomiyaki cooperation dish has achieved the Okonomiyaki cooperation dish by using three abilities of those SIGVerse as "Externals change", "Bodily movement/Movement/Rotation" and "Avator and Robot conversation/Agent mutual conversation".

お好み焼き協調料理は、SIGVerse プラットフォーム上で動くサンプルアプリケーションです。その SIGVerse プラットフォームは三つの優れた能力をもちます。それは「知覚」「力学」「対話」のシミュレーション能力です。お好み焼き協調料理は SIGVerse の三つの能力それぞれを「見た目変化」「身体動作/移動/回転」「アバタ・ロボット対話/エージェント相互対話」として実現しています。

知覚	力学	対話
 <p>ロボットはアバタの行動を見えています</p>	 <p>ロボットはソースをお好み焼きに塗ります</p>	 <p>ロボットは「生地を混ぜてね」と助言します</p>
 <p>次の行動を補助します</p>	 <p>「回転」 「移動」 複雑な「身体動作」</p>	 <p>ロボットはアバタが戸惑う場合、進んでアバタを補助します</p>

1.1. お好み焼き協調料理の目的

You learn the usage of SIGVerse to this Okonomiyaki cooperation dish according to the procedure of "Operation", "Setting", and "Remodel". You can be well informed of the usage of SIGVerse when it finished reading this tutorial, and make a new application before long. This tutorial aims at the help.

貴方は、このお好み焼き協調料理を「操作」「設定」「改造」を通じて学びます。貴方はこのチュートリアルを読み終えた時、SIGVerse の使い方を熟知し、やがて新しいアプリケーションを作成できるでしょう。このチュートリアルは、その手助けを目的としています。

1.2. お好み焼き協調料理の概念

The okonomiyaki cooperation dish is assumption done in a general okonomiyaki shop. Moreover, the okonomiyaki made by the above-mentioned cooking method is called "Kansai" okonomiyaki. This "Kansai" okonomiyaki needs various cooking utensils besides pork and the ingredients, for instance, the iron plate and the source. The state changes like this okonomiyaki on SIGVerse, and the unit of the thing that rushes into action (It is moved) is called "Agent". For instance, Hera who uses it to turn okonomiyaki inside out is one of the agents, too. Agent's all lists that appear with the okonomiyaki cooperation dish are recorded as follows.

お好み焼き協調料理は、一般的なお好み焼き屋で行う前提です。また、[下記](#)調理法で作られるお好み焼きは「関西」お好み焼きと言います。この「関西」お好み焼きは豚肉や生地他に、様々な調理器具、例えば鉄板やソースを必要とします。SIGVerse上ではこのお好み焼きの様に状態が変化したり、行動を起こす(動かされたりする)物の単位を「エージェント」と呼びます。例えば、お好み焼きを裏返す為に用いるヘラもエージェントの一つです。以下にお好み焼き協調料理で登場する全てのエージェントの一覧を記します。

Character's name 登場人物名	Distinguished name 分類名	Predominant role 主な役割
Avator アバタ	User 利用者	It is a leading part of the okonomiyaki cooperation dish, and it serves as user's taking the place on SIGVerse virtual space. お好み焼き協調料理の主役であり、SIGVerse 仮想空間上で利用者の代わりに務めます
Robot ロボット	Robot ロボット	It operates autonomous, and the dish of Avator is assisted and it advises. 自律的に動作し、アバタの料理の補助やアドバイスなどを行います
Okonomiyaki お好み焼き	Material 材料	It is an object of the dish, and a dish baked mixing a favorite tool material with the cloth. 料理の対象であり、好きな具材を生地に混ぜて焼き上げる料理です
Pork 豚肉		It is one of the materials of okonomiyaki, and an indispensable material to the Kansai okonomiyaki. お好み焼きの材料の一つであり、関西お好み焼きに必須の材料です
Sauce ソース	Seasoning 調味料	It is one of the seasonings put on okonomiyaki, and a source for peculiar okonomiyaki to Japan. お好み焼きにかける調味料の一つであり、日本特有のお好み焼き用のソースです
Seaweed(Nori) 海苔		It is one of the seasonings put on okonomiyaki, and a seasoning of which the origin is peculiar seaweeds to Japan. お好み焼きにかける調味料の一つであり、日本特有の海草を元とした調味料です
Katsubushi 鰹節		It is a seasoning that one of the seasonings put on okonomiyaki it, ferments a peculiar fish(bonito) to Japan, and is dry. お好み焼きにかける調味料の一つであり、日本特有の魚を発酵させて乾燥させた調味料です
Oil 油		In the oil used when okonomiyaki is burnt on the iron plate, it paints it with the tool called a brush. お好み焼きを鉄板の上で焼く時に使う油で、刷毛と言う道具で塗ります
Hera ヘラ	Cooking utensil 調理器具	It turns inside out, and the cooking utensil of the divided fork of okonomiyaki. お好み焼きを裏返したり、切り分けるフォークのような調理器具です
Teppan 鉄板		It is a cooking utensil of the board of the plastron to bake okonomiyaki. お好み焼きを焼き上げる鉄製の板の調理器具です
Bowl ボウル		It is a container where the cloth before okonomiyaki is burnt is put. お好み焼きを焼く前の生地を入れておく容器です

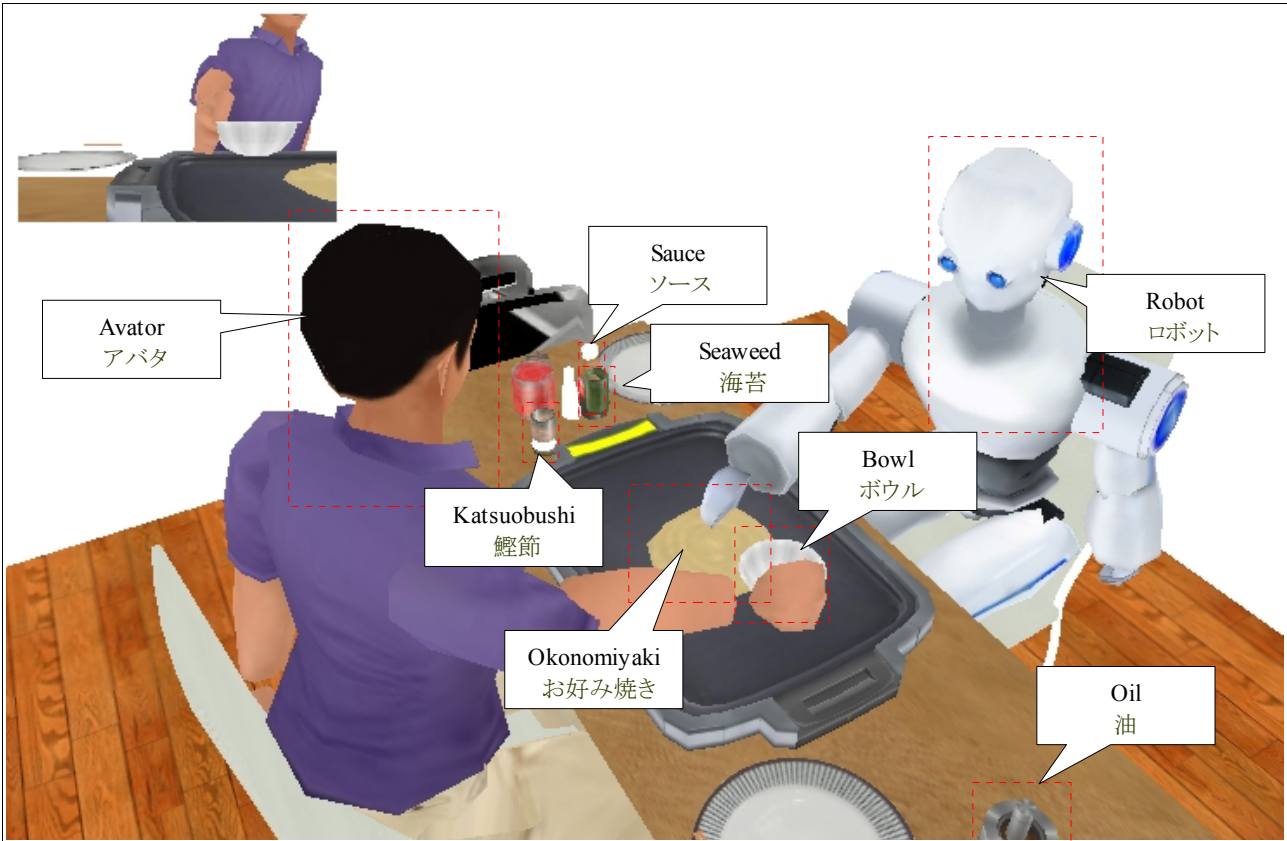
A detailed explanation of okonomiyaki is published in Wikipedia. Please refer to that when you want to know more detailed information.

お好み焼きの詳しい説明は、Wikipedia にも掲載されています。より詳しい情報を知りたい場合は、そちらを参考にしてください。

Website Ex.	Okonomiyaki on Wikipedia
-------------	--

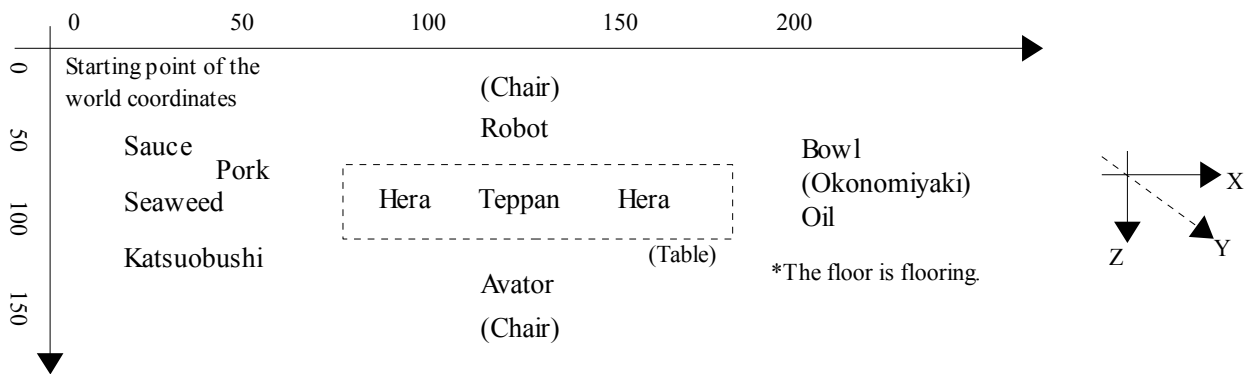
The sample of Figure of appearing "Agent" is recorded as follows. These agents accompany all the operation such as "Movement". The movement such as chairs is not accompanied oppositely is not treated as an agent.

以下に登場する「エージェント」の図例を記します。これらのエージェントは全て「移動」などの動作を伴います。逆に椅子など移動を伴わないものはエージェントとして扱いません。



The sample of Figure of appearing "Agent" is recorded as follows. These agents accompany all the operation such as "Movement". Because the chair etc. do not move oppositely at all, it is not treated as an agent.

SIGVerse は世界座標を元とした仮想空間を作成します。その仮想空間は三次元空間です。以下に全てのエージェントの配置の図例と一覧を記します。



Agent's name	x	y	z
Avator	100	50	160
Robot	100	50	30
Okonomiyaki	100	70	100

Agent's name	x	y	z
Teppan	100	70	100
Oil	170	70	100
Bowl	150	70	100

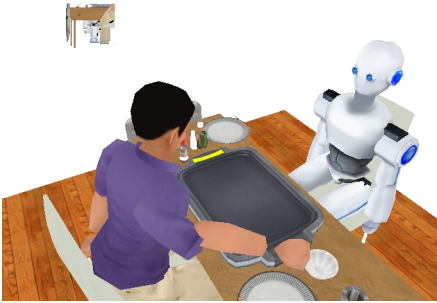







Agent's name	x	y	z
Sauce	40	70	100
Seaweed	50	75	100
Katsuobushi	55	70	110










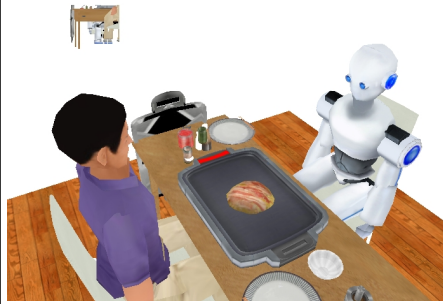
*This arrangement is a rough standard. この配置は大体の目安です










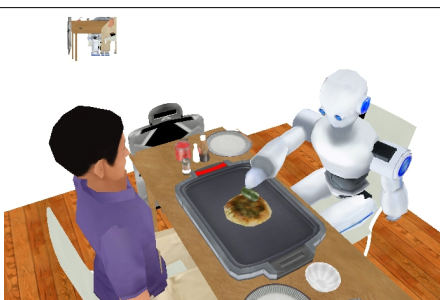
1.3. お好み焼き協調料理の手順









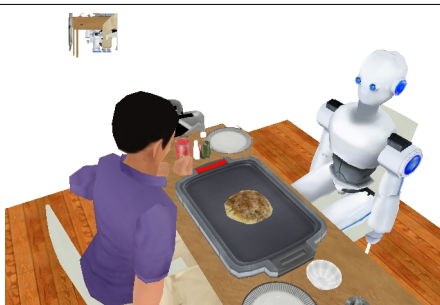

Okonomiyaki is a peculiar dish to Japan. This dish is done according to "Ingredients is mixed", "Ingredients is burnt on the iron plate", "It turns inside out and burn" and "Season with the seasoning" procedures. The procedure of the dish of Okonomiyaki is written in the following.





お好み焼きは日本独特の料理です。この料理は好きな具材を生地に混ぜて鉄板の上で焼きます。この料理は「生地を混ぜる」「生地を鉄板の上で焼く」「更に裏返して焼く」「調味料で味付けする」手順で行います。下記にお好み焼きの料理の手順を記します。

料理の手順	動作サンプル	
	アバター	ロボット
<p>Mix dough</p> <p>“Avator or Robot mixes a favorite tool with the ingredients.” アバター又はロボットは具材を生地に混ぜます。</p> <p>For instance, the lobster, the cheese, and the cabbage are mixed. The meaning of the Okonomiyaki is to use a favorite tool.</p>		
<p>Take oil</p> <p>“To paint oil on the iron plate (Teppan), Avator or Robot takes oil.” アバター又はロボットは鉄板の上に油を塗る為に、油を手元に引き寄せます。</p>		
<p>Put oil on Teppan</p> <p>“Avator or Robot paint oil on the iron plate(Teppan).” アバター又はロボットは鉄板の上に油を塗ります。</p> <p>*Oil is painted with the brush. *油は刷毛で塗られます。</p>		
<p>Check gas level</p> <p>“Avator or Robot confirms the remainder of the gas level of the iron plate.” アバター又はロボットは鉄板の下のガスコンロのガスの残量を確認します。</p> <p>*Avator or Robot confirms it by watching, only watching.</p>		

料理の手順	動作サンプル	
	アバター	ロボット
<p>Fire up Teppan</p> <p>“Avator or Robot ignites the iron plate. (using gas)” アバタ又はロボットは鉄板のガスコンロに火をつけます。</p> <p>*There is one kind of iron plate about the gas stove.</p>		
<p>Put ingredients on the Teppan</p> <p>“Avator or Robot puts the ingredients of Okonomiyaki on the iron plate. “ アバタ又はロボットは鉄板に生地をのせます。</p> <p>*The ingredients is a mix of flour and water. It looks like the pancake.</p>		
<p>Take pork</p> <p>“Avator or Robot takes pork. “ アバタ又はロボットは豚肉を手元に引き寄せます。</p> <p>This pork is always necessary for Okonomiyaki. And pork is burnt with the ingredients. Okonomiyaki is cooked so.</p>		
<p>Put pork</p> <p>“Avator or Robot puts pork on Okonomiyaki. “ アバタ又はロボットは豚肉をお好み焼きの上へのせます。</p>		
<p>Check condition of roasted Okonomiyaki</p> <p>“Avator or Robot confirms the state (condition of roasted) of Okonomiyaki. “ アバタ又はロボットはお好み焼きの焼け具合を確認します。</p>		

料理の手順	動作サンプル	
	アバター	ロボット
<p>Flip Okonomiyaki</p> <p>“Avator or Robot turns Okonomiyaki inside out.” アバター又はロボットはお好み焼きを裏返します。</p>		
<p>Take sauce</p> <p>“Avator or Robot takes the sauce.” アバター又はロボットはソースを手元に引き寄せます。</p> <p>Avator or Robot season Okonomiyaki by using the sauce.</p>		
<p>Put sauce on Okonomiyaki</p> <p>“Avator or Robot paints the sauce on Okonomiyaki.” アバター又はロボットはソースをお好み焼きに塗ります。</p> <p>Sauce like a Japanese original soy sauce.</p>		
<p>Take seaweed(Nori)</p> <p>“Avator or Robot takes seaweed.” アバター又はロボットは海苔を手元に引き寄せます。 Seaweed is a Japanese original seasoning. It one kind of is about seaweeds. The externals are greens. And seaweed is a powder.</p> <p>*Seaweed is put in the container.</p>		
<p>Put seaweed on Okonomiyaki</p> <p>“Avator or Robot puts seaweed on Okonomiyaki.” アバター又はロボットは海苔をお好み焼きに振りかけます。</p>		



料理の手順	動作サンプル	
	アバター	ロボット
<p>Take Katsuobushi</p> <p>“Avator or Robot takes the Katsuobushi.” アバター又はロボットは鰹節を手元に引き寄せます。</p> <p>Katsuobushi is a dried bonito. This is a Japanese original seasoning, and the powder. it is incased.</p>		
<p>Put Katsuobushi on Okonomiyaki</p> <p>“Avator or Robot puts the Katsuobushi on Okonomiyaki.” アバター又はロボットは鰹節をお好み焼きに振りかけます。</p>		
<p>Cut Okonomiyaki</p> <p>“Avator or Robot cuts Okonomiyaki.” アバター又はロボットはお好み焼きを切り分けます。</p> <p>Avator or Robot cuts Okonomiyaki for easiness to eat. This is the same as the pancake.</p>		
<p>Put Okonomiyaki on dish</p> <p>“Avator or Robot puts Okonomiyaki on the plate(dish).” アバター又はロボットはお好み焼きを皿にのせます。</p>		
<p>Reduce the heat of Teppan to low</p> <p>“Avator or Robot weakens thermal power of the iron plate.” アバター又はロボットは「必要に応じて」鉄板のガスコンロの火を弱めます。</p>		

料理の手順	動作サンプル	
	アバター	ロボット
Increase the heat of Teppan to high “Avator or Robot strengthens thermal power of the iron plate.” アバタ又はロボットは「必要に応じて」鉄板のガスコンロの火を強めます。		
Put out the fire of Teppan “Avator or Robot stops thermal power of the iron plate.” アバタ又はロボットは鉄板のガスコンロの火を止めます。		

1.3. お好み焼き協調料理のゴール

The goal of the Okonomiyaki cooperation dish is to be able to cook without scorching Okonomiyaki. If Avator or Robot cooks alone, Okonomiyaki is burned. It is necessary to cook two people for that cooperatively. The Okonomiyaki cooperation dish can set the behavior of the Robot at that time. For instance, it is "Assist positively", "On negative lines assistance", "Only advice" and "Everything is left".

お好み焼き協調料理のゴールは、お好み焼きをこがさずに料理することです。アバター一人で料理するとお好み焼きはこげる為、アバタはロボットと二人で協調して料理する必要があります。お好み焼き協調料理では、その時のロボットの振る舞いを「積極的に補助」「消極的に補助」「アドバイスのみ」「全て任せる」から選択できます。

State of Okonomiyaki	It is burned (The dish is a failure).	It is completed (The dish is a success).
Sample of figure *Please look at Okonomiyaki on the iron plate well.		

You will learn the behavior of this robot, understand contents of the Okonomiyaki cooperation dish, and learn the goodness of SIGVerse. You must learn the Okonomiyaki cooperation dish by using this tutorial, and remember how to make the application of SIGVerse. At that time, SIGVerse will bring you various findings.

貴方はこのロボットの振る舞いを学び、お好み焼き協調料理の中身を理解し、そして SIGVerse の良さを知ってください。貴方はこのチュートリアルを利用して、SIGVerse のアプリケーションの作り方を覚えてください。その時、SIGVerse は様々な知見を貴方にもたらずでしょう。

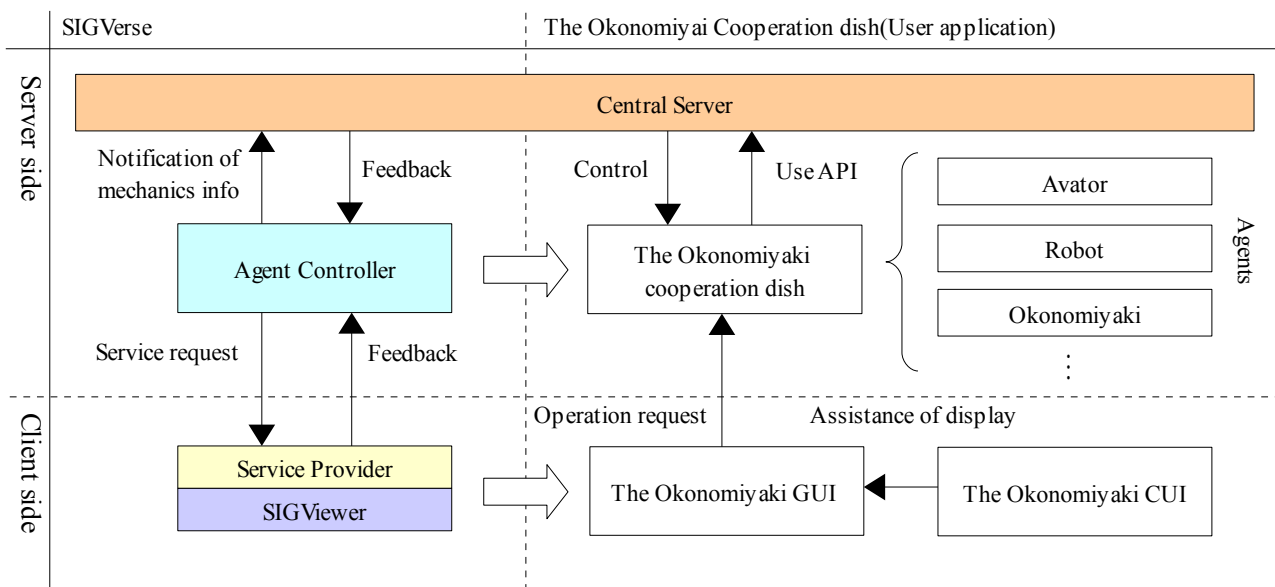
2. お好み焼き協調料理の起動

Contents

- ・お好み焼き協調料理の起動
 - －お好み焼き協調料理の起動 (サーバ側)
 - －お好み焼き GUI の起動 (クライアント側)
 - －お好み焼き CUI の起動 (クライアント側)
 - －シグビューワの起動

This paragraph explains the method of starting the Okonomiyaki cooperation dish. The Okonomiyaki cooperation dish becomes a location of a user application original on SIGVerse. Therefore, it depends on the composition of the software of SIGVerse. SIGVerse consists of "Central server" and "Service provider". The Okonomiyaki cooperation dish application moves on "Central server" of that, and confirms the operation by "SIGViewer" that is the service provider. In addition, the Okonomiyaki cooperation dish application consists of "Okonomiyaki CUI" to assist "Okonomiyaki GUI" to tell the user's operation and the display. The list of those samples of Figure and the appearing software is recorded as follows.

本章はお好み焼き協調料理の起動方法を説明します。この為にあらず SIGVerse の全体の構成を簡単に説明します。SIGVerse は「セントラルサーバ」「サービスプロバイダ」から成り立ちます。お好み焼き協調料理アプリケーションは、その「セントラルサーバ」上で動き、サービスプロバイダの一つである「SIGViewer」でその動作を確認します。更にお好み焼き協調料理アプリケーションは、利用者の操作を伝える「お好み焼き GUI」と、表示を補助する「お好み焼き CUI」からなります。以下にそれらの例を記します。



Name of software	Operating layer	Role of software
Central Server セントラルサーバ	Server side サーバサイド	Central Server controls the conversation with the calculation of mechanics with the main of the SIGVerse server. The user application moves on this software. セントラルサーバは力学の計算と対話を制御する SIGVerse の本体サーバです。ユーザアプリケーションは、このソフトウェア上で動作します。
The Okonomiyaki cooperation dish お好み焼き協調料理	Server side サーバサイド	This is a user application that controls the Okonomiyaki cooperation dish. Avator and Robot are managed by the unit called "Agent". これはお好み焼き協調料理を制御するユーザアプリケーションです。アバターやロボットは「エージェント」と呼ばれる単位で管理されます。
Service Provider(SIGViewer) サービスプロバイダ (SIGViewer)	Client side クライアントサイド	This controls perception. For instance, the list is made though Robot is seen. Moreover, Viewer of the simulation operates, too. This function is offered with software named SIGViewer.

Name of software	Operating layer	Role of software
		エージェントの視覚情報や音声情報を提供します。例えば、ロボットが見たもののリストを作成します。また、シミュレーションの動作の Viewer としての役割も持ちます。
The Okonomiyaki GUI お好み焼き GUI	Client side クライアントサイド	The Okonomiyaki GUI offers the operation of Avator. Initialization and beginning the scenario of the Okonomiyaki cooperation dish are done besides. お好み焼き GUI は、アバタの操作を提供します。それ以外にお好み焼き協調料理のシナリオの初期化と開始を行います。
The Okonomiyaki CUI お好み焼き CUI	Client side クライアントサイド	OkonomiyakiCUI is superimpose and outputs the voice of the remark of Robot. This corresponds to the assistance of the display. お好み焼き CUI は、スーパーインポーズやロボットの発言の音声出力を行います。これは表示の補助に相当します。

2.1. お好み焼き協調料理の起動(サーバ側)

The Okonomiyaki cooperation dish is started. This starts on the server for the user application. Central Server operates on Linux. Therefore, it starts from the SSH client etc. The procedure is written as follows.

*In this material, SIGVerse is made assumption that has already been installed in the HOME directory.

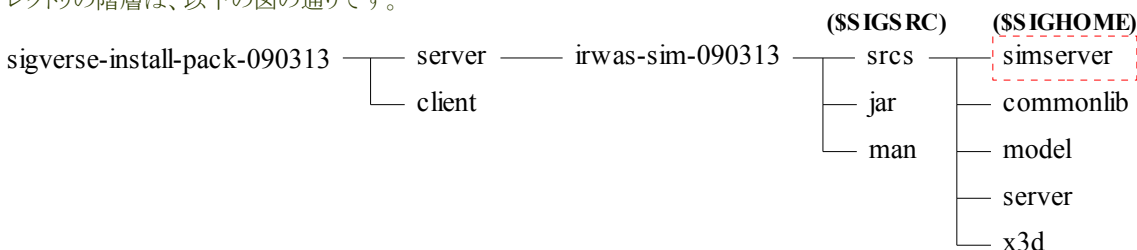
お好み焼き協調料理は、セントラルサーバ上で動作するユーザアプリケーションの為にサーバ上で起動します。セントラルサーバは Linux 上で動き、全ての操作は SSH クライアントなどから行います。以下にその手順を記します。

※この資料では、SIGVerse は既に HOME ディレクトリ(ユーザカレントディレクトリ)にインストールされている前提とします。

2.1.1. セントラルサーバディレクトリに移動

Central Server starts when the simserver.sh shell script is executed. It moves to the directory with simserver.sh for that. The hierarchy of the directory is as shown in the following figures.

セントラルサーバは simserver.sh シェルスクリプトを実行して起動します。その為に simserver.sh があるディレクトリに移動します。ディレクトリの階層は、以下の図の通りです。



Operation Ex. `cd $HOME/sigverse-install-pack-090313/server/irwas-sim-090313/srcs/simserver`

Moreover, this simserver directory is omitted and the srcs directory on "\$SIGHOME" and simserver is omitted with "\$SIGSRC" by the following sentences.

また、この simserver ディレクトリを以下の文章で「\$SIGHOME」、simserver 上の srcs ディレクトリを「\$SIGSRC」と省略します。

2.1.2. セントラルサーバの起動

Simserver.sh is executed. Simserver.sh needs one or more arguments. It is port number specified by "-p". For instance, please specify 9000 for a port number. And this port number is very important. Please remember.

simserver.sh を実行します。simserver.sh は最低一つの引数を必要とします。それは「-p」で指定するポート番号です。ポート番号は例えば 9000 を指定してください。そしてこのポート番号はとても重要です。覚えておいて下さい。

Operation Ex. `./simserver.sh -p 9000`

2.1.3. セントラルサーバの起動の確認

The start of the Okonomiyaki cooperation dish is confirmed. It sees and confirms the list of the process by the ps command. Or, it confirms it by the log when simserver.sh is started. When the list of the following process was displayed or the log was displayed, it started normally.

お好み焼き協調理の起動を確認します。それは ps コマンドでプロセスの一覧を見て確認します。または、simserver.sh を起動した場合のログで確認します。下記のプロセスの一覧が表示されているか、ログが表示された場合は、正常に起動しました。

Operation Ex. <code>ps aux grep \$USER</code>	
Confirmation by log of simserver.sh	Confirmation by display of ps command
<pre>[*****@***** ~]\$./start_serv.sh 9000 [SYS] reading ./conf/nii_newman.x3d... [SYS] Controller attached to "Avator_000" [SYS] Controller attached to "Bowl_000" [SYS] Controller attached to "Sauce_000" [SYS] Controller attached to "Teppan_000" [SYS] Controller attached to "Nori_000" [SYS] Controller attached to "Okonomi_000" [SYS] Controller attached to "Robot_000" : confirm</pre>	<pre>/bin/sh -x ./simserver.sh -p 9000 ./simserver -p 9000 ./model/runac -h 127.0.0.1 -p 9000 -n Abura_000 -l command/Command.so -s ./model/runac -h 127.0.0.1 -p 9000 -n Avator_000 -l command/Command.so -s ./model/runac -h 127.0.0.1 -p 9000 -n Bowl_000 -l command/Command.so -s ./model/runac -h 127.0.0.1 -p 9000 -n Nori_000 -l command/Command.so -s ./model/runac -h 127.0.0.1 -p 9000 -n Okonomi_000 -l command/Command.so -s ./model/runac -h 127.0.0.1 -p 9000 -n Robot_000 -l command/Command.so -s ./model/runac -h 127.0.0.1 -p 9000 -n Sauce_000 -l command/Command.so -s ./model/runac -h 127.0.0.1 -p 9000 -n Teppan_000 -l command/Command.so -s : confirm</pre>

Please read the troubleshooting when it doesn't start and the error is output.

*Moreover the log and the display omit part.

もし、起動しない場合やエラーが出力される場合は、トラブルシューティングをご覧ください。
※また、ログの表示は一部を省略しています。

2.2. お好み焼き GUI の起動(クライアント側)

Okonomiyaki GUI is started. Okonomiyaki GUI operates on the client side. You make it to assumption in which Okonomiyaki GUI has already been installed. Please read "Install of the Okonomiyaki cooperation dish" about the installation method. The procedure of the start is written as follows.

お好み焼き GUI を起動します。お好み焼き GUI はクライアントサイドで動作します。貴方は既にお好み焼き GUI をインストールしている前提とします。インストールの方法は、導入章をご覧ください。以下に起動の手順を記します。

2.2.1. お好み焼き GUI の起動

Okonomiyaki "GUI" starts with the software of general Windows in a similar way. Please start from the start menu of Windows.

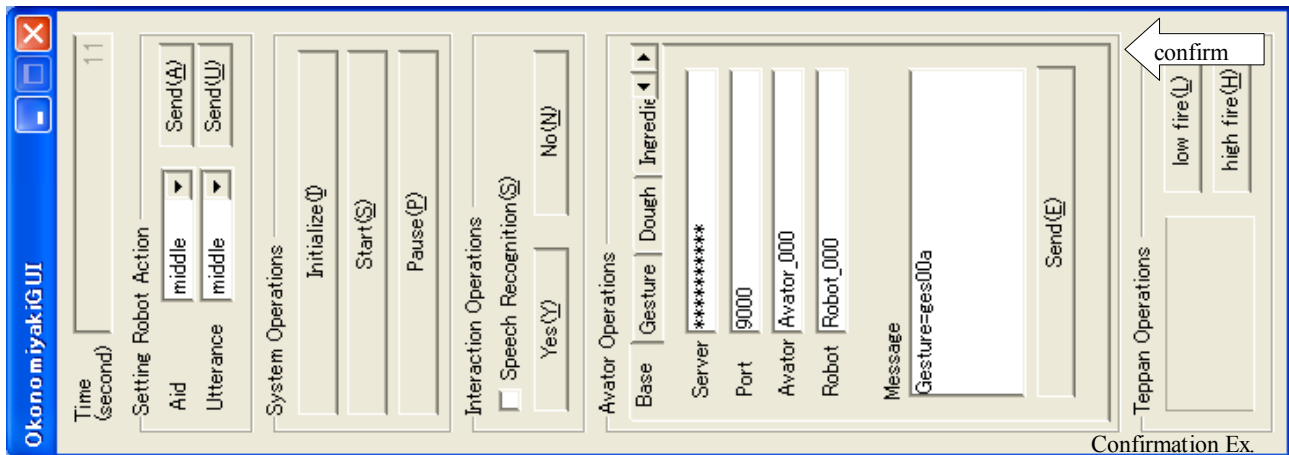
お好み焼き GUI は一般的な Windows のソフトウェアと同じ方法で起動します。Windows のスタートメニューから起動してください。

Operation Ex.	
---------------	--

2.2.2. お好み焼き GUI の起動の確認

Okonomiyaki “GUI” start is confirmed. The Window application of the figure below is displayed normally.

お好み焼き GUI の起動を確認します。正常の場合、下図の Window アプリケーションが表示されます。



2.3. お好み焼き CUI の起動(クライアント側)

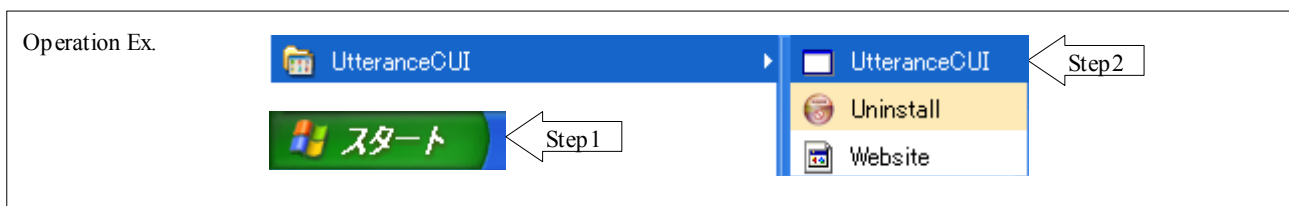
Okonomiyaki CUI is started. Okonomiyaki CUI operates on the client side. You make it to assumption in which Okonomiyaki CUI has already been installed. Please read "Install of the Okonomiyaki cooperation dish" about the installation method. The procedure of the start is written as follows.

お好み焼き CUI を起動します。お好み焼き CUI はクライアントサイドで動作します。貴方は既にお好み焼き CUI をインストールしている前提とします。インストールの方法は、「Install of the Okonomiyaki cooperation dish」をご覧ください。以下に起動の手順を記します。

2.3.1. お好み焼き CUI の起動

Okonomiyaki “CUI” starts with the software of general Windows in a similar way. Please start from the start menu of Windows.

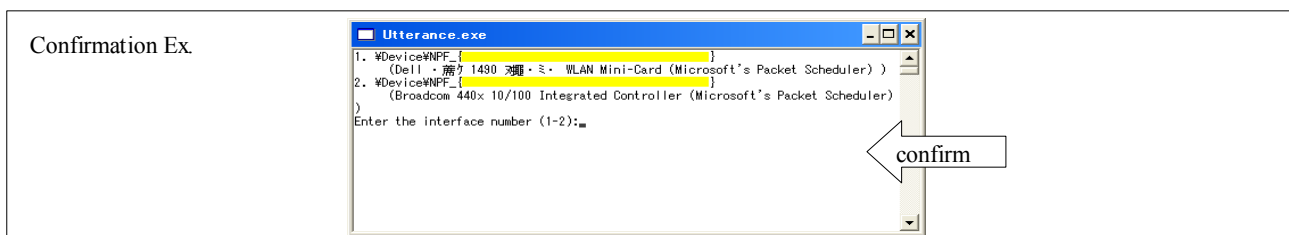
お好み焼き CUI は一般的な Windows のソフトウェアと同じ方法で起動します。Windows のスタートメニューから起動してください。



2.3.2. お好み焼き CUI の起動の確認

Okonomiyaki “CUI” start is confirmed. The Window console of the figure below is displayed normally.

お好み焼き CUI の起動を確認します。正常の場合、下図の Window コンソールが表示されます。



2.4. SIGViewer の起動

SIGViewer is a service provider. Therefore, it operates on the client side. Moreover, it connects to the server so that SIGViewer may have the role as the service provider. This paragraph explains the setting of the connection destination and the start method. The setting and the start are done according to the following procedures.

SIGViewer はサービスプロバイダです。その為、クライアントサイドで動作します。また、SIGViewer はサービスプロバイダとしての役割を持つ為、サーバに接続します。この項は、その接続先の設定と起動方法を説明します。以下の手順で設定と起動を行います。

2.4.1. SIGViewer の設定

The configuration file is edited and SIGViewer is set. The configuration file is in "C:\Program Files\SIGViewer\release" when the directory of default is selected at the installation of SIGViewer. Please open configuration file "startup.cfg" that is here with the text editor etc. The example of the content of the configuration file is recorded in the following.

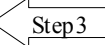
SIGViewer の設定は、設定ファイルを編集して行います。その設定ファイルは、SIGViewer のインストールの時にデフォルトのディレクトリを選択した場合は「C:\Program Files\SIGViewer\release」にあります。ここにある設定ファイル「startup.cfg」をテキストエディタなどで開いてください。下記に設定ファイルの内容の例を記します。

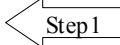
```
# -----
# Host name and port number of simulation server
# シミュレーションサーバーのホスト名とポート番号
# -----
#SERVER=localhost
SERVER=192.168.3.108
PORT=9999
:
```

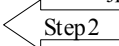
Please look at the upper part of this configuration file. SERVER and PORT are written. You should change this SERVER to "Internet Protocol address" of Central Server or "Domain name". Moreover it is necessary to change PORT to the port number specified when Central Server is started. Please rewrite it in correct SERVER and PORT.

設定ファイルの上の部分を見てください。SERVERとPORTが書かれています。貴方はこのSERVERをセントラルサーバの「IPアドレス」または「ドメイン名」に変更する必要があります。また、PORTはセントラルサーバを起動した時に指定したポート番号に変更する必要があります。正しいSERVERとPORTに書き換えてください。

```
# -----
# Host name and port number of simulation server
# シミュレーションサーバーのホスト名とポート番号
# -----
#SERVER=localhost
#SERVER=192.168.3.108
SERVER=socio.iir.nii.ac.jp
PORT=9000
:
```

Please save it. 

 Step 1

 Step 2

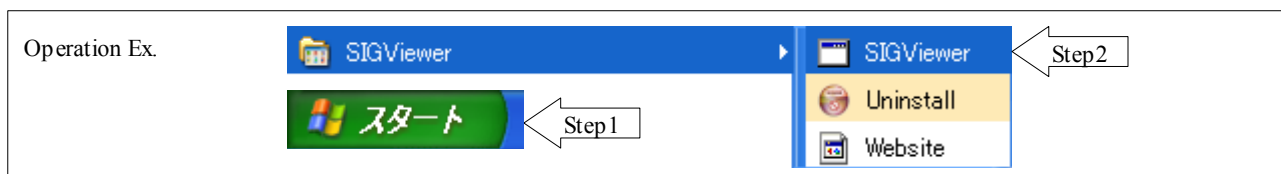
Please save the configuration file. The setting of SIGViewer ends by this. Please read the manual of SIGVerse when you want to change other settings.

設定ファイルを保存してください。これでSIGViewerの設定は終わりです。それ以外の設定を変えたい場合は、SIGVerseのマニュアルをご覧ください。

2.4.2. SIGViewer の起動

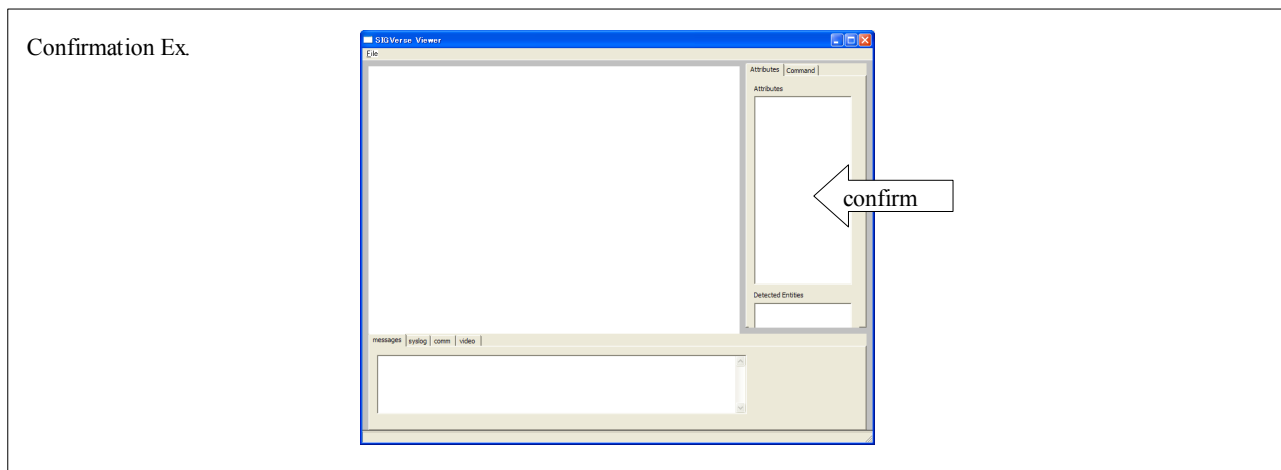
SIGViewer starts with the software of general Windows in a similar way. Please start from the start menu of Windows.

SIGViewer は一般的な Windows のソフトウェアと同じ方法で起動します。Windows のスタートメニューから起動してください。



When the Window application of the figure below was displayed, SIGViewer was normally started. Please read the troubleshooting when it is not started or the Window application shuts immediately even if it starts.

下図の Windows アプリケーションが表示された場合、SIGViewer は正常に起動されました。もし、起動されない場合、または起動しても直ぐに Windows アプリケーションが閉じる場合は、トラブルシューティングをご覧ください。



2.4.3. SIGViewer とセントラルサーバの接続

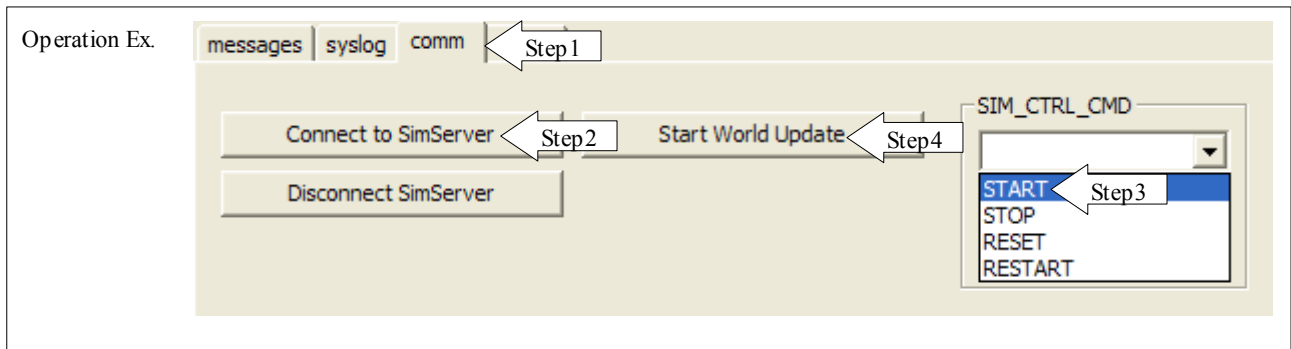
The connection with Central Server is the following procedure. Please operate it SIGViewer as follows.

- Step1. The Comm tag is selected.
- Step2. The Connect to SimServer button is selected.
- Step3. START is selected from among SIM_CTRL_CMD.
- Step4. Start World Up date button is selected.

*It waits until the processing of SIGViewer ends for a little while.

セントラルサーバとの接続は次の手順です。SIGViewer に以下の操作をしてください。

- ステップ1. Comm タグを選択します。
 - ステップ2. Connect to SimServer ボタンを押下します。
 - ステップ3. SIM_CTRL_CMD の中から START を選択します。
 - ステップ4. Start World Update ボタンを押下します。
- ※少しの間、SIGViewer の処理が終わるまで待ちます。



When the screen of Avator and Robot of the figure below is displayed, SIGViewer is normally started.

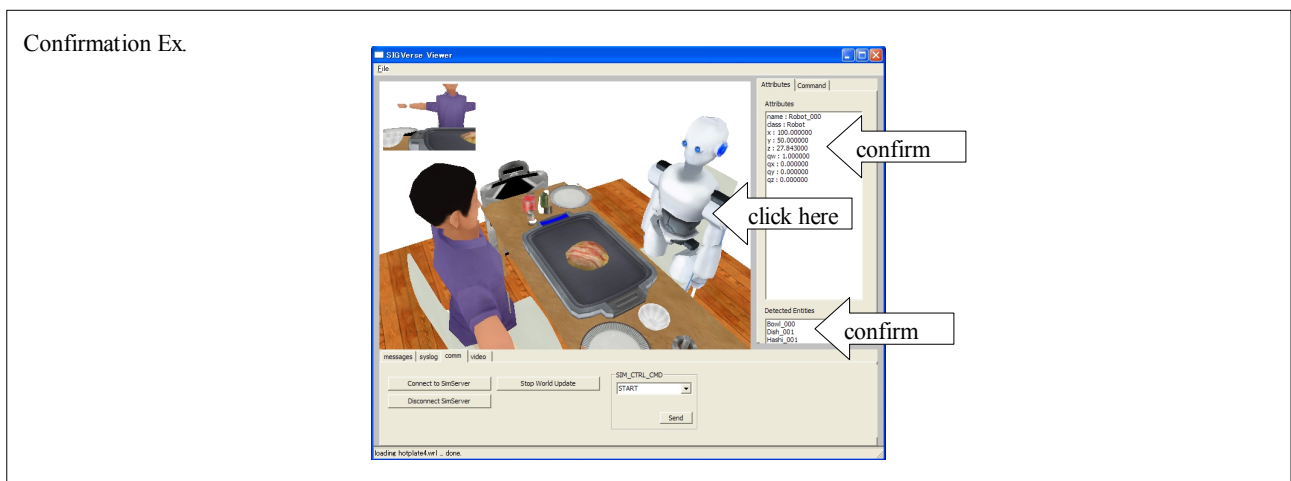
下図のアバタとロボットの画面が表示された場合、SIGViewerは正常に起動されています。



2.4.4. SIGViewerの起動の確認

Please left-click Robot on SIGViewer with the mouse. SIGViewer operates normally when the value is displayed in "Attributes" and "Detect Entities" in the part in the right of SIGViewer.

SIGViewer上のロボットをマウスで左クリックしてください。SIGViewer右の部分の「Attributes」と「Detect Entities」に値が表示された場合、SIGViewerは正常に動作しています。



3. お好み焼き協調料理の操作

Contents

- ・お好み焼き協調料理の操作
 - －お好み焼き協調料理の操作
 - －知覚に関する動作
 - －力学に関する操作
 - －対話に関する操作

In this paragraph, it explains the operation of three ability of SIGVerse operation of the Okonomiyaki cooperation dish and "Perception", "Mechanics", and "Conversation". In the Okonomiyaki cooperation dish, Okonomiyaki-GUI is chiefly used. In three abilities of SIGVerse, SIGViewer is chiefly used.

Especially, please confirm "Supplementary level" that shows the degree that Robot helps in the Okonomiyaki cooperation dish. If it lowers kindly if a supplementary level is improved, Robot becomes unkind. As for the Okonomiyaki GUI, the setting is possible.

この章では、お好み焼き協調料理の操作とSIGVerseの三つの能力「知覚」「力学」「対話」の操作について説明します。お好み焼き協調料理では、主にお好み焼きGUIを使います。SIGVerseの三つの能力では、主にSIGViewerを使います。特に、お好み焼き協調料理では、ロボットの手伝う度合いを表す「補助度」について確認してください。ロボットは、補助度を高めれば親切に、低めれば不親切になります。お好み焼きGUIはその設定が可能です。

補助度	
お好み焼き GUI	「高」
<p>お好み焼き GUIで補助度を設定します</p>	<p>「高」</p> <p>Please mix dough</p> <p>ロボットはアドバイスと補助をします</p>
	<p>「中」</p> <p>I mix dough</p> <p>アバタが戸惑う場合は補助します</p>
<p>補助度には「低」「中」「高」「全」の四種類があります</p>	<p>「全」</p> <p>Thanks for you take oil</p> <p>ロボットは全て料理します</p>
	<p>「低」</p> <p>Please mix dough</p> <p>ロボットはアドバイスだけです</p>

3.1. お好み焼き協調料理の操作

This paragraph explains the manner of operation of the Okonomiyaki cooperation dish. All the Okonomiyaki cooperation dishes are operated with Okonomiyaki GUI. The manner of operation is written in the following.

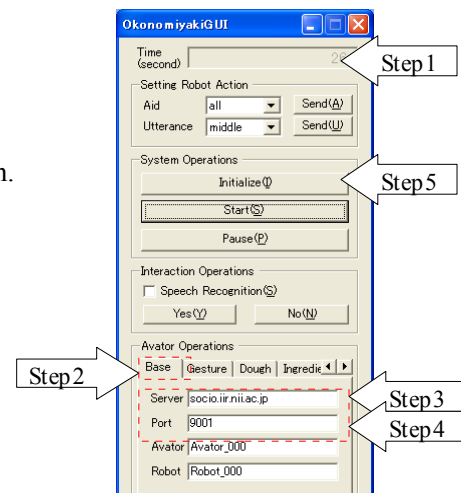
この項は、お好み焼き協調料理の操作方法を説明します。お好み焼き協調料理はお好み焼き GUI で全て操作します。下記にその操作方法を記します。

3.1.1. セントラルサーバに接続する操作

You use Okonomiyaki GUI and operate Avator. And Okonomiyaki GUI transmits your operation to Central Server. Therefore, Okonomiyaki GUI needs information on Central Server. Please specify information on Central Server according to the following procedure.

貴方はお好み焼き GUI を用いてアバタを操作します。そしてお好み焼き GUI は貴方の操作をセントラルサーバに送信します。その為、お好み焼き GUI はセントラルサーバの情報を必要とします。下記の手順でセントラルサーバの情報を指定してください。

- Step1. Okonomiyaki GUI is started.
 - Step2. Base tag is selected.
 - Step3. IP or the domain name is input to the column of Server.
 - Step4. Port number of Central Server is input to the port column.
 - Step5. Initialize button is selected.
- Step1. お好み焼き GUI を起動します。
 - Step2. Base タグを選択します。
 - Step3. IP アドレスまたはドメイン名を Server 欄に入力します。
 - Step4. セントラルサーバのポート番号を Port 欄に入力します。
 - Step5. Initialize ボタンを押下します。



Okonomiyaki GUI has the screen where the processing result is displayed. When the following characters are displayed on the screen, Okonomiyaki GUI can be connected with Central Server.

お好み焼き GUI は処理結果を表示する画面を持ちます。その画面に以下の文字が表示されていた場合、お好み焼き GUI はセントラルサーバに接続できています。

Confirmation Ex.

Target Agent Name:Avator_000, Command: Clear=Avator_000

Target Agent Name:Avator_000, Command: Clear=Robot_000

Target Agent Name:Avator_000, Command: Clear=Okonomi_000

Target Agent Name:Avator_000, Command: Clear=Abura_000

Target Agent Name:Avator_000, Command: Clear=Sauce_000

Target Agent Name:Avator_000, Command: Clear=Katsubushi_000

⋮

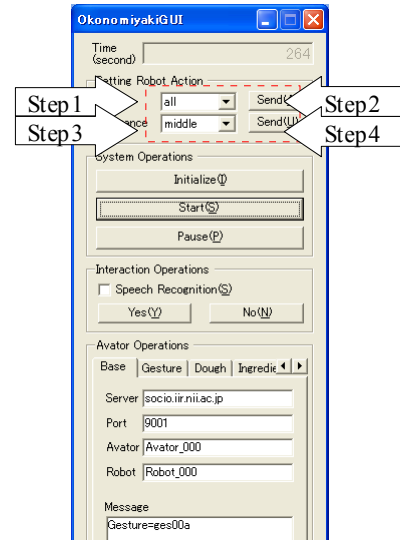
confirm

3.1.2. ロボットの補助度の操作

Robot has a supplementary level and the remark level. A supplementary level is a help degree of Robot. The remark level is a chat degree of Robot. Robot doesn't talk when the remark level is made "low". Those elements are specified according to the following procedures.

ロボットは、補助度と発言度を持ちます。補助度はロボットの手伝い度合いです。発言度はロボットのおしゃべり度合いです。発言度を「low」にした場合、ロボットはしゃべりません。それらの度合いは、以下の手順で指定します。

- When you specify a supplementary level
 - Step1. Supplementary level is selected from among the Aid column.
 - Step2. A right Send button is selected.
- When you specify the remark level
 - Step3. Remark level is selected from among the Utterance column.
 - Step4. A right Send button is selected.
- 補助度を指定する場合
 - Step1. Aid 欄の中から補助度を選択します。
 - Step2. 右側の Send ボタンを押下します。
- 発言度を指定する場合
 - Step3. Utterance 欄の中から補助度を選択します。
 - Step4. 右側の Send ボタンを押下します。



Okonomiyaki GUI has the screen where the processing result is displayed. When the following characters are displayed on the screen, a supplementary level and the remark level are correctly sent.

お好み焼き GUI は処理結果を表示する画面を持ちます。その画面に以下の文字が表示されていた場合、補助度と発言度は正しく送られています。

Confirmation Ex

Target Agent Name:Avator_000, Command:	Aid=3@Robot_000
Target Agent Name:Avator_000, Command:	Uttr=2@Robot_000
Target Agent Name:Avator_000, Command:	Aid=2@Robot_000
Target Agent Name:Avator_000, Command:	Uttr=1@Robot_000
Target Agent Name:Avator_000, Command:	Aid=1@Robot_000
Target Agent Name:Avator_000, Command:	Aid=0@Robot_000
:	:

← confirm

The behavior of a supplementary level of Robot is written as follows. Is the remark "low (off)" or else(on).

以下にロボットの補助度の振る舞いを記します。発言は「low(なし)」かそれ以外(あり)です。

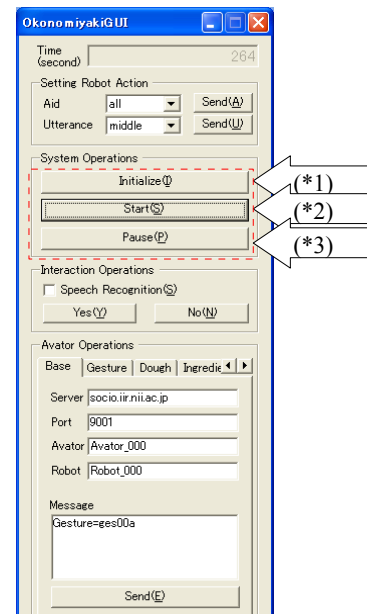
補助度	ロボットの振る舞い
「高」	ロボットはお好み焼き料理の補助をします。それ以外にアドバイスをしますが、アバタが戸惑う場合は、その料理を代わりに行います。
「中」	ロボットはお好み焼き料理の補助をします。それ以外にアドバイスをしますが、アバタが戸惑う場合でも、そのままアドバイスを続けます。
「低」	ロボットはアドバイスだけです。
「全」	ロボットは全て料理します。アバタに手伝われた場合は、お礼を言います。

3.1.3. お好み焼き協調料理の状態の操作

The Okonomiyaki cooperation dish can do initialization, beginning, and the temporary stop. Initialization is returned to the first state. Beginning begins the movement of Robot. The temporary stop stops the movement of Robot temporarily. These operations are specified with Okonomiyaki GUI. The operation method is written as follows.

お好み焼き協調料理は、初期化と開始と一時停止を行えます。初期化は最初の状態に戻します。開始はロボットの動作を開始します。一時停止はロボットの動作を一時停止します。これらの操作はお好み焼き GUI で指定します。以下にその操作の方法を記します。

- When initializing it...(*1)
All the Okonomiyaki cooperation dishes are returned to the first state. Robot becomes a halt condition.
- When beginning...(*2)
The movement of Robot begins.
- When stopping temporarily...(*3)
The movement of Robot is stopped temporarily.
- 初期化する場合・・・(*1)
全てのお好み焼き協調料理を最初の状態に戻します。ロボットは停止状態になります。
- 開始する場合・・・(*2)
ロボットの動作を開始します。
- 一時停止する場合・・・(*3)
ロボットの動作を一時停止します。



Okonomiyaki GUI has the screen where the processing result is displayed. When the following characters are displayed on the screen, various processing is normally executed.

お好み焼き GUI は処理結果を表示する画面を持ちます。その画面に以下の文字が表示されていた場合、各種の処理は正常に実行されています。

Confirmation Ex.

Target Agent Name:Avator_000, Command: Attt=1@Robot_000	(*2)
Target Agent Name:Avator_000, Command: Attt=0@Robot_000	(*3)
Target Agent Name:Avator_000, Command: Clear=Avator_000	(*1)
Target Agent Name:Avator_000, Command: Clear=Robot_000	
⋮	

*Moreover, please must be it an initial state or Robot moving or does Robot stop, and confirm SIGViewer.

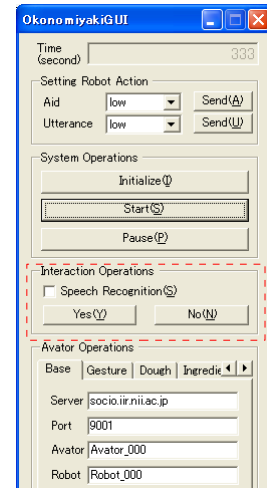
※また、初期状態であるか、ロボットが動いているか、ロボットが止まっているかは、SIGViewer でも確認してください。

3.1.4. ロボットとアバタの音声対話

This function is provided for the future. For instance, it is time when it uses voice recognition. Please understand as the sample for that time. This is not used now.

この機能は、将来の為にあります。例えば音声認識を用いる場合です。その時の為のサンプルと理解してください。現在、これは使用していません。

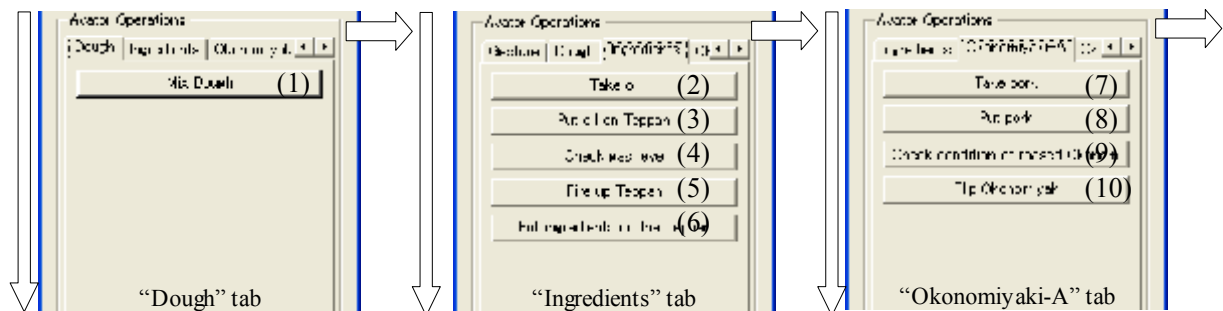
- **Speech Recognition**
It is a sample that uses the SAPI voice recognition engine.
SAPI 音声認識エンジンを利用するサンプルです。
- **Yes**
It is a sample that sends it to Central Server when the result of voice recognition is "Yes".
音声認識の結果が「はい」の場合、それをセントラルサーバに送るサンプルです。
- **No**
It is a sample that sends it to Central Server when the result of voice recognition is "No".
音声認識の結果が「いいえ」の場合、それをセントラルサーバに送るサンプルです。



3.1.5. お好み焼き協調料理の操作






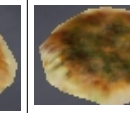
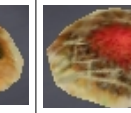

In this paragraph, it explains the operation of the dish that uses Okonomiyaki GUI. Please look at "Introduction" about the procedure of the dish. The one procedure corresponds to one button on Okonomiyaki GUI. These buttons are arranged in "Dough", "Ingredients", "Okonomiyaki-A", "Okonomiyaki-B", "Consummation" in the center part on the screen, and the "Setout" tab. Okonomiyaki is cooked in this order of the tab.

この項では、お好み焼き GUI を用いた料理の操作を説明します。料理の手順は「Introduction」を見てください。その一つの手順がお好み焼き GUI 上の一つのボタンに該当します。これらのボタンは画面中央部の「Dough」、「Ingredients」、「Okonomiyaki-A」、「Okonomiyaki-B」、「Consummation」、「Setout」タブに配置されています。このタブ順にお好み焼き料理を行います。



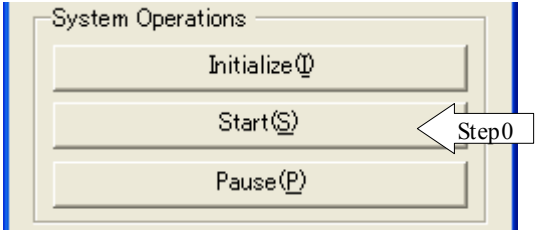
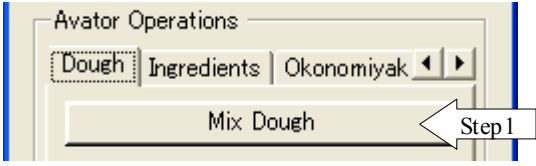
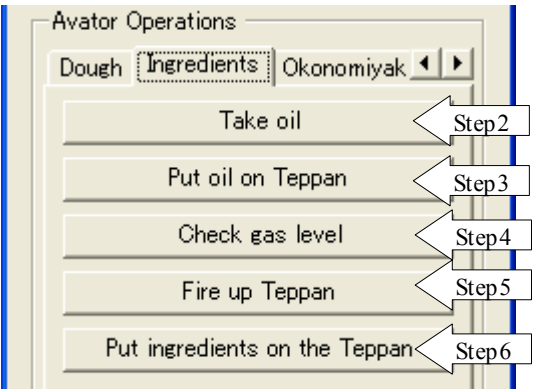
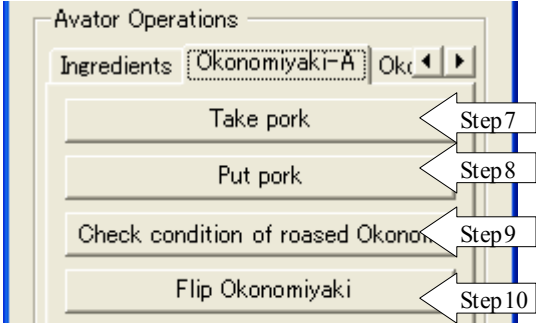
These tabs are related to the state of Okonomiyaki. The sample of Figure is written as follows.

これらのタブはお好み焼きの状態と関連しています。以下に図例を記します。

Dough	Ingredients	Okonomiyaki-A	Okonomiyaki-B	Consummation Setout	(Other) Burned		
							

The correspondence of the list and the dish procedure of each tab and the button is written as follows. You usually cook Okonomiyaki according to this procedure.

以下に、各タブとボタンの一覧と料理手順の対応を記します。貴方は、通常、この手順でお好み焼きを料理します。

Name of tab	Operation of dish	Example of figure
(Start)	Step0. Robot begins to move Step0. 「Start」…ロボットは動き始めます	
Dough	Step1. Avator mix dough Step1. アバタは生地を混ぜます	
Ingredients	Step2. Avator take oil Step3. Avator put oil on Teppan Step4. Avator check gas level Step5. Avator fire up Teppan Step6. Avator put ingredients on the Teppan Step2. アバタは油を手元に引き寄せます Step3. アバタは鉄板に油をひきます Step4. アバタはガスコンロのガス残量をみます Step5. アバタは鉄板のガスコンロに火をつけます Step6. アバタは鉄板に生地をひきます	
Okonomiyaki-A	Step7. Avator take pork Step8. Avator put pork Step9. Avator check condition of roasted Okonomiyaki Step10. Avator flip Okonomiyaki Step7. アバタは豚肉を手元に寄せます Step8. アバタは豚肉をお好み焼きにのせます Step9. アバタはお好み焼きの焼け加減をみます Step10. アバタはお好み焼きを裏返します	

Name of tab	Operation of dish	Example of figure
Okonomiyaki-B	Step11. Avator take sauce Step12. Avator put sauce on Okonomiyaki Step13. Avator take seaweed(Nori) Step14. Avator put seaweed on Okonomiyaki Step15. Avator take Katsubushi Step16. Avator put Katsubushi on Okonomiyaki Step11. アバタはソースを手元に引き寄せます Step12. アバタはソースをお好み焼きに塗ります Step13. アバタは海苔を手元に引き寄せます Step14. アバタは海苔をお好み焼きに振りかけます Step15. アバタは鰹節を手元に引き寄せます Step16. アバタは鰹節をお好み焼きに振りかけます	
Consummation	Step17. Avator cut Okonomiyaki Step17. アバタはお好み焼きを切り分けます	
Setout	Step18. Avator put Okonomiyaki on dish Step19. Avator put out the fire pf Teppan Step18. アバタはお好み焼きを皿にのせます Step19. アバタは鉄板のガスコンロの火を消します	
(Teppan)	StepA. Avator reduce the heat of Teppan to low StepB. Avator increase the heat of Teppan to high StepA. アバタは鉄板のガスコンロの火を弱めます StepB. アバタは鉄板のガスコンロの火を強めます	

3.1.6. お好み焼き協調料理のこつ

There is order in the dish of Okonomiyaki. Therefore, Okonomiyaki cannot be done even if it puts it on the iron plate without mixing the ingredients. The operation that should be done is written as follows.

お好み焼きの料理には順番があります。例えば、生地を混ぜずに鉄板の上ののせても、お好み焼きは出来ません。以下にお好み焼き GUI のタブ毎に必ず行わなくてはならない操作を記します。

Dough	Ingredients	Okonomiyaki-A	Okonomiyaki-B	Consummation	Setout
Mix Dough 生地を混ぜる	Put Ingredients on Teppan 鉄板に生地をのせる	Flip Okonomiyaki お好み焼きを裏返す	Put Katsubushi on Okonomiyaki 鰹節を振りかける	Cut Okonomiyaki お好み焼きを切り分ける	Put Okonomiyaki on Dish お好み焼きを皿によせる

3.2. 知覚に関する操作

The Okonomiyaki cooperation dish uses three abilities of SIGVerse. In this paragraph, it explains the operation of perception. This operation is done with SIGViewer. The operation that relates to perception list and method are written in the following.

お好み焼き協調料理は、SIGVerse の三つの能力を利用しています。この項では、知覚の操作を説明します。この操作は SIGViewer で行います。以下に知覚の操作の一覧と、操作の方法を記します。

3.2.1. お好み焼き協調料理の知覚に関する処理

The following commands are input to SIGViewer and the ability of the externals change of the Okonomiyaki cooperation dish using the ability of the perception of SIGVerse is operated.

SIGVerse の知覚の能力を使用した、お好み焼き協調料理の見た目変更の能力は、コマンドを SIGViewer に入力して操作します。

SIGVerse	The Okonomiyaki cooperation dish	Example of command of operation
Perception	External changes	Visual

3.2.2. 見た目の変更の操作

The externals change function changes externals of Okonomiyaki according to the following procedures.

見た目変更機能は、以下の手順でお好み焼きの見た目を変更します。これはお好み焼きの見た目を変える例です。

Step0. SIGViewer and Okonomiyaki GUI are started.

Step1. "Initialize" is selected with Okonomiyaki GUI.

Step2. "Commnad tab" of SIGViewer is selected.

Step3. It is input to "Selection for the transmission" column of SIGViewer as "Okonomi_000".

Step4. It is input to "SendMessage argument" column of SIGViewer as "Visual=2".

Step5. "Execution" button of SIGViewer is selected.

Step0. SIGViewer とお好み焼き GUI を起動します

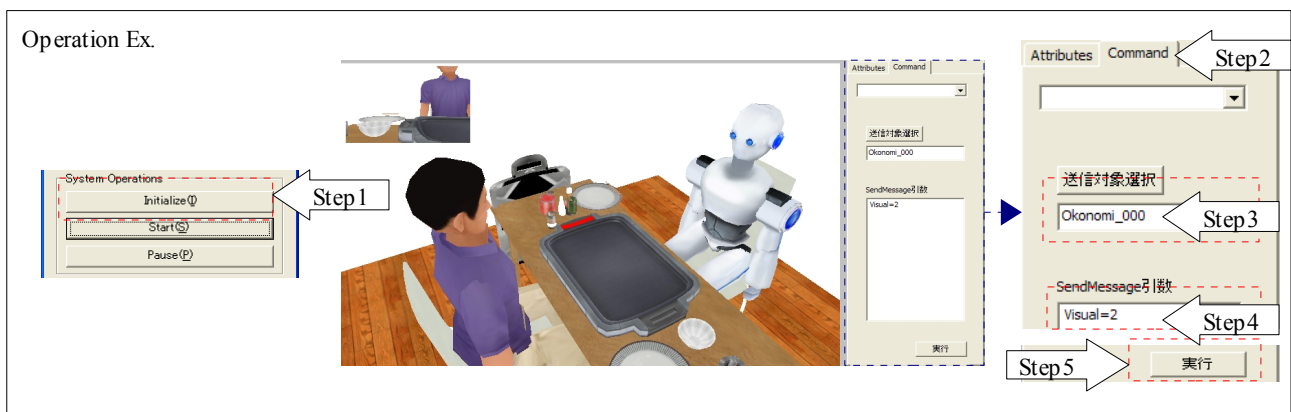
Step1. Okonomiyaki GUI で「Initialize」を押下します

Step2. SIGViewer の「Commnad タブ」を選択します

Step3. SIGViewer の「送信対象選択」欄に「Okonomi_000」と入力します

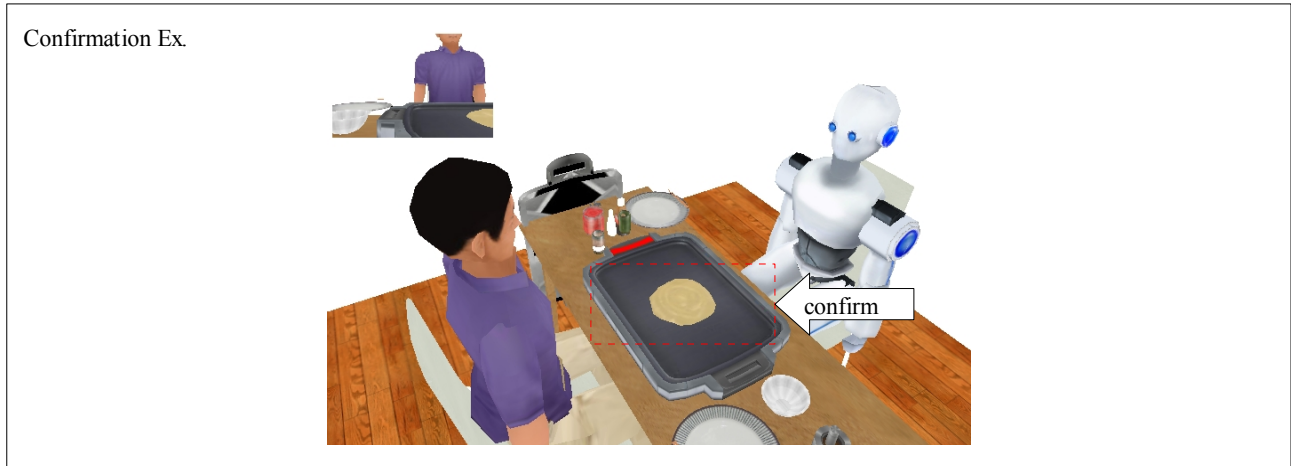
Step4. SIGViewer の「SendMessage 引数」欄に「Visual=2」と入力します

Step5. SIGViewer の「実行」ボタンを押下します



Whether externals of Okonomiyaki changed into two (Ingredients) is confirmed. The "Initialize" button of Okonomiyaki GUI is selected several times when not changing.

お好み焼きの見た目が2(生地)に変化したか確認します。もし、変化しない場合は、何度かお好み焼き GUIの「Initialize」ボタンを押下します。



There are nine kinds of externals of Okonomiyaki. The nine kinds are written as follows. There is a magnitude correlation, and when the current externals are five, the number of this externals is not revokable, and revokable in externals of 6, 7, 8 and 9 in externals of 1, 2, 3 and 4. Please note it.

お好み焼きの見た目は、9種類あります。以下にその9種類を記します。この見た目の番号は、大小関係があり、今の見た目が5の場合、1, 2, 3, 4の見た目には変更できず、6, 7, 8, 9の見た目には変更できます。注意してください。

Number of externals	Explanation of externals	Example of inputting command	Example of figure
1(Default)	It doesn't see it. 見えない状態	Visual=1	
2	State of ingredients 生地の状態	Visual=2	
3	Pork was put. 豚肉をのせられた状態	Visual=3	
4	It is burnt moderately. 程度に焼けた状態	Visual=4	
5	Sauce was put. ソースがかけられた状態	Visual=5	
6	Seaweed was put. 海苔がかけられた状態	Visual=6	
7	Katsuobushi was put. 鰹節がかけられた状態	Visual=7	
8	State of completion できあがりの状態	Visual=8	
9	It was burned. こげた状態	Visual=9	

3.3. 力学に関する操作

The Okonomiyaki cooperation dish uses three abilities of SIGVerse. In this paragraph, it explains the operation of mechanics. This operation is done with SIGViewer. The list of the operation of mechanics and the operation method are written as follows.

お好み焼き協調料理は、SIGVerse の三つの能力を利用しています。この項では、力学の操作を説明します。この操作は SIGViewer で行います。以下に力学の操作の一覧と、操作の方法を記します。

3.3.1. お好み焼き協調料理の力学に関する処理

The function of the Okonomiyaki cooperation dish and the command using the ability of the mechanics of SIGVerse are input to SIGViewer and it operates it. The list is written as follows.

SIGVerse の力学の能力を使用した、お好み焼き協調料理の機能とそのコマンドを SIGViewer に入力して操作します。以下にその一覧を記します。

SIGVerse	The Okonomiyaki cooperation dish	Example of command of operation
Mechanics	Rotation	Angle
	Movement	Move
	Bodily movement	Gesture

3.3.2. 回転の操作

The rotation function rotates the specified one according to the following procedures.

回転機能は、以下の手順で指定のものを回転します。これはお好み焼きを Y 軸に 180 度回転させる例です。

Step0. SIGViewer is started.

Step1. "Commnad tab" of SIGViewer is selected.

Step2. It is input to "Selection for the transmission" column of SIGViewer as "Teppan_000".

Step3. It is input to "SendMessage argument" column of SIGViewer as "Angle=0:1:0:0:180:10".

Step4. "Execution" button of SIGViewer is selected.

Step0. SIGViewer を起動します

Step1. SIGViewer の「Commnad タブ」を選択します

Step2. SIGViewer の「送信対象選択」欄に「Teppan_000」と入力します

Step3. SIGViewer の「SendMessage 引数」欄に「Angle=0:1:0:0:180:10」と入力します

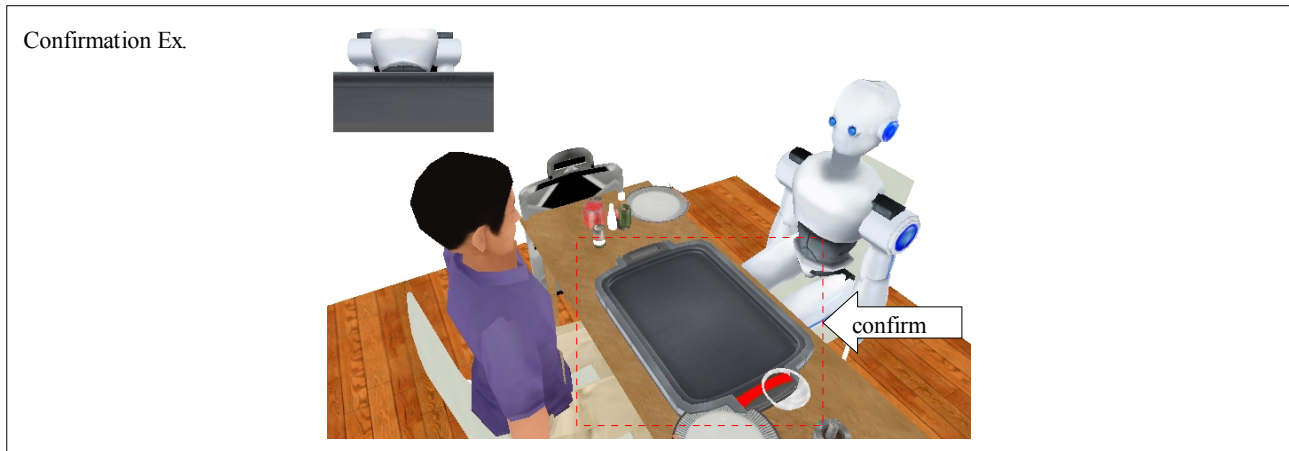
Step4. SIGViewer の「実行」ボタンを押下します

Operation Ex.

The diagram illustrates the operation steps for rotating the Okonomiyaki cooperation dish in SIGViewer. It shows a 3D scene with a person and a robot at a table with a teppan. To the right, a screenshot of the SIGViewer interface is shown with four steps highlighted: Step 1 (Attributes/Command tab), Step 2 (Selection for the transmission: Okonomi_000), Step 3 (SendMessage argument: Visual=2), and Step 4 (Execution button).

Whether the iron plate rotated in the direction of Y 180 times is confirmed. The one other than the iron plate rotate similarly, too.

鉄板が Y 方向に 180 度回転したか確認します。鉄板以外のものも同様に回転します。



The demand of the rotation is specified by the following formats. Rotation is not done at specification not correct.

回転の要求は、以下の書式で指定します。もし、正しくない指定の場合、回転は行われません。

Argument	Explanation of argument	Possible to input
1 st argument	The rotation in the direction of "X" is specified. In one, it rotates, and 0 is a meaning that doesn't rotate. Y 方向への回転を指定します。1 は回転、0 は回転しない意味です。	1, 0
2 nd argument	The rotation in the direction of "Y" is specified. In one, it rotates, and 0 is a meaning that doesn't rotate. Y 方向への回転を指定します。1 は回転、0 は回転しない意味です。	1, 0
3 rd argument	The rotation in the direction of "Z" is specified. In one, it rotates, and 0 is a meaning that doesn't rotate. Z 方向への回転を指定します。1 は回転、0 は回転しない意味です。	1, 0
4 th atgment	It means the beginning angle. 開始の角度を意味します。	0-360
5 th argument	It means the end angle. 終了の角度を意味します。	0~360
6 th argument	It means the amount of the angle that rotates at a time. *It doesn't rotate gradually when 360 is specified, and it rotates the angling directly. 一度に回転する角度の量を意味します。 ※360 を指定した場合、段階的に回転せず、直接その角度に回転します。	-360~360

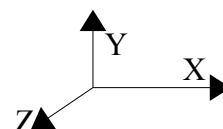
For instance, it is input when rotating in the direction of X 180 times, "At Angle = 1:0:0: 0:180:10". This is a rotation that turns Okonomiyaki inside out.

例えば、X 方向に 180 度回転する場合は、「Angle=1:0:0:0:180:10」と入力します。これはお好み焼きを裏返す回転です。

Command Ex.	Angle=1:0:0:0:180:10
-------------	----------------------

Moreover, SIGVerse does height and has the direction of Y. Please note it.

また、SIGVerse は、Y 方向を高さとしています。注意してください。



3.3.3. 移動の操作

The mobile function moves the specified one according to the following procedures.

移動機能は、以下の手順で指定のものを移動します。これは油を X 座標 100,Y 座標 70,Z 座標 100 に移動する例です。

Step0. SIGViewer is started.

Step1. "Commnad tab" of SIGViewer is selected.

Step2. It is input to "Selection for the transmission" column of SIGViewer as "Abura_000".

Step3. It is input to "SendMessage argument" column of SIGViewer as "Move=100:70:100".

Step4. "Execution" button of SIGViewer is selected.

Step0. SIGViewer を起動します

Step1. SIGViewer の「Commnad タブ」を選択します

Step2. SIGViewer の「送信対象選択」欄に「Abura_000」と入力します

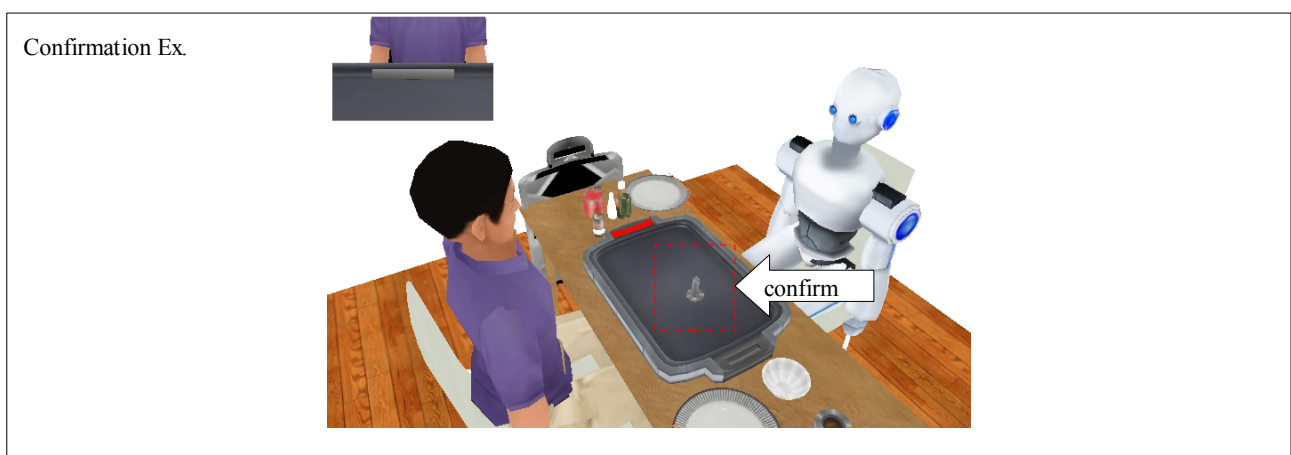
Step3. SIGViewer の「SendMessage 引数」欄に「Move=100:70:100」と入力します

Step4. SIGViewer の「実行」ボタンを押下します



Whether oil moved to x:100, y:70 and z:100 position is confirmed. The one other than oil move similarly, too.

油が X:100, Y:70, Z:100 位置に移動したか確認します。油以外のものも同様に移動します。



The demand of the movement is specified by the following formats. The movement is not done at specification not correct.

移動の要求は、以下の書式で指定します。もし、正しくない指定の場合、移動は行われません。

Argument	Explanation of argument	Possible to input
1 st argument	It is a value of X coordinates. X 座標の値です。	Numerical value including minus sign (Ex. -100~100)
2 nd argument	It is a value of X coordinates. Y 座標の値です。	Numerical value including minus sign (Ex. -100~100)
3 rd argument	It is a value of X coordinates. Z 座標の値です。	Numerical value including minus sign (Ex. -100~100)

For instance, it is input when moving to starting point (0,0,0), "Move = 0:0:0".

例えば、原点(0,0,0)に移動する場合は、「Move=0:0:0」と入力します

Command Ex.	Move=0:0:0
-------------	------------

The starting point in the Okonomiyaki cooperation dish is near behind the right of Robot. This place is a range of 200×200(x={0-200}, z={0-200}).



お好み焼き協調理上、原点はロボットの右の後ろの付近にあります。この場所は 200×200 の範囲内です。

3.3.4. 身体動作の操作

The bodily movement function is done according to the following procedures.

身体動作機能は、以下の手順で行います。これはロボットのボウルに左手を伸ばす身体動作の例です。

Step0. SIGViewer is started.

Step1. "Commnad tab" of SIGViewer is selected.

Step2. It is input to "Selection for the transmission" of SIGViewer as "Robot_000". (or "Avator_000")

Step3. It is input to "SendMessage argument" column of SIGViewer as "Gesture=ges92r".

Step4. "Execution" button of SIGViewer is selected.

Step0. SIGViewer を起動します

Step1. SIGViewer の「Commnad タブ」を選択します

Step2. SIGViewer の「送信対象選択」欄に「Robot_000」と入力します(または「Avator_000」)

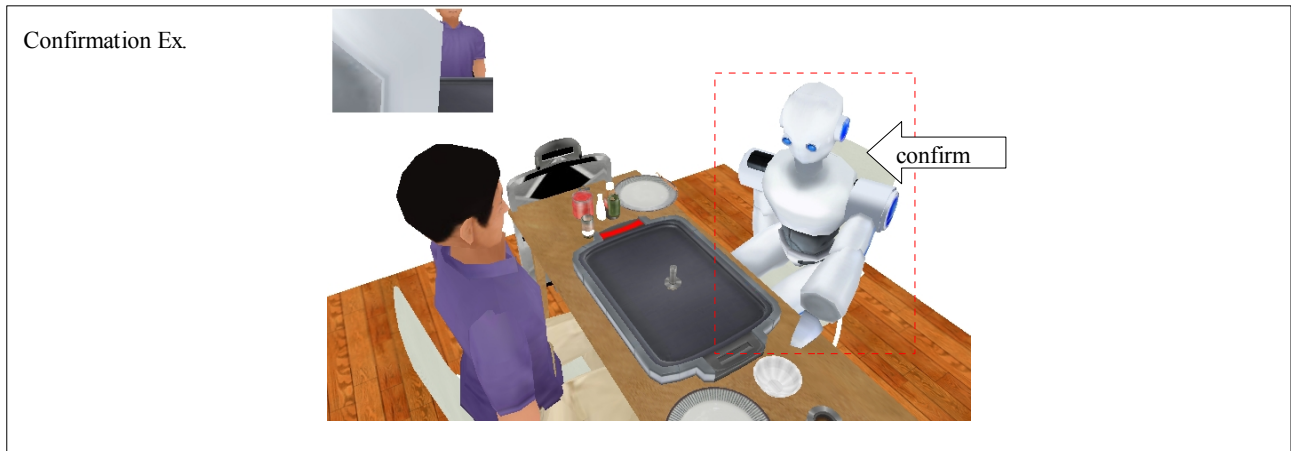
Step3. SIGViewer の「SendMessage 引数」欄に「Gesture=ges92r」と入力します

Step4. SIGViewer の「実行」ボタンを押下します

Operation Ex.

Whether Robot extended the left hand to the ingredients of Okonomiyaki is confirmed. As for the bodily movement, only Robot and Avator are possible.

ロボットがお好み焼きの生地に左手を伸ばしたか確認します。身体動作は、ロボットとアバタのみ可能です。



The demand of the bodily movement is specified by the following formats. The bodily movement is not done at specification not correct.

身体動作の要求は、以下の書式で指定します。もし、正しくない指定の場合、身体動作は行われません。

Argument	Explanation of argument	Possible to input
1 st argument	The name of the bodily movement is specified. 身体動作の名前を指定します。	Please look at the table below.

The bodily movement is closely related to the dish. The table of the correspondence of the dish and the bodily movement is written as follows.

身体動作は、料理と密接に関係しています。以下に料理と身体動作の対応の表を記します。

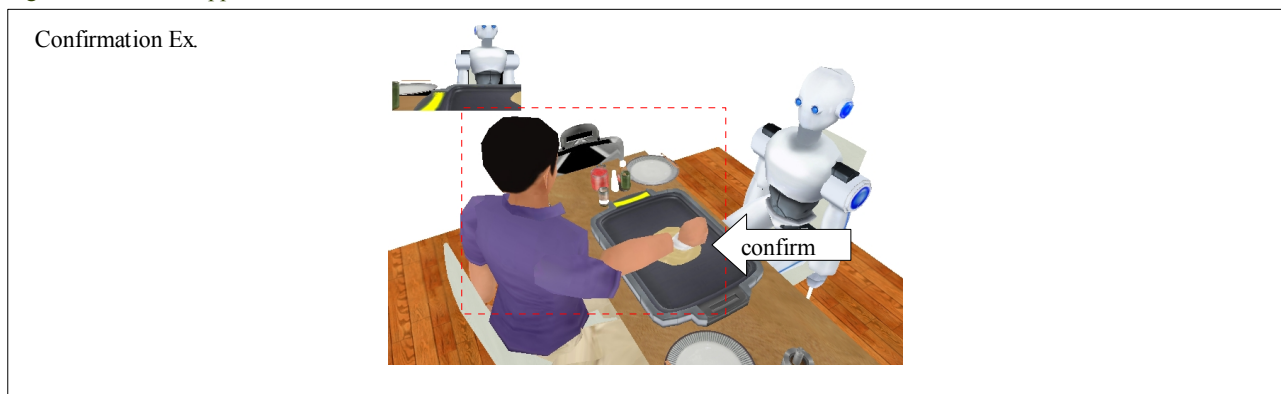
Procedure of dish	Content of bodily movement	Name of bodily movement	
		For Avator	For Robot
Initial state 初期状態	It sits on a chair. 椅子に座ります。	ges00a	ges00r
Mix dough 生地を混ぜます	It reaches the bowl and ingredients is mixed. ボウルに手を伸ばして生地を混ぜます。	ges92a	ges92r
Take oil 油を手元に引き寄せます	It reaches oil. 油に手を伸ばします。	ges82a	ges82r
	Oil is moved on the iron plate. 油を鉄板の上に移動させます。	ges83a	ges83r
Put oil on Teppan 油を鉄板にひきます	Oil is moved to the right side. 油を右方向に移動させます。	ges84a	ges84r
	Oil is moved to the left side. 油を左方向に移動させます。	ges85a	ges85r
	Oil is returned to former position. 油を元の位置に戻します。	ges86a	ges86r
Check gas level ガスコンロのガスの残量をみます	The residual quantity meter of the gas is confirmed by watching. ガスの残量計を目視で確認します。	ges42a	ges42r
Fire up Teppan 鉄板のガスコンロに火をつけます	It reaches the switch of the iron plate and it ignites it. 鉄板のスイッチに手を伸ばして点火します。	ges01a	ges01r

Procedure of dish	Content of bodily movement	Name of bodily movement	
		For Avator	For Robot
Put ingredients on the Teppan 生地を鉄板にのせます	It reaches the container that the ingredients enters. 生地の入った入れ物に手を伸ばします	ges92a	ges92r
	The container that the ingredients enters is moved on the iron plate. 生地の入った入れ物を鉄板の上に移動します	ges93a	ges93r
	The container that the ingredients enters is inclined and poured. 生地の入った入れ物を傾けて流し込みます	ges94a	ges94r
	It returns it based on the inclination of the container. 入れ物の傾きを元に戻します	ges95a	ges95r
	The container is returned to former position. 入れ物をものの位置に戻します	ges96a	ges96r
Take pork 豚肉を手元に引き寄せます	It reaches pork. 豚肉に手を伸ばします	ges05a	ges05r
	Pork is put on the ingredients. 豚肉を生地の上ののせます	ges08r	ges08r
Put Pork 豚肉をお好み焼きにのせます	Pork is pressed against Okonomiyaki a little. 豚肉をお好み焼きに少し押し付けます	ges10a	ges10r
	It is repeated. それを繰り返します	ges08a	ges08r
Check condition of roasted Okonmiyaki お好み焼きの焼け加減をみます	The state of Okonomiyaki is confirmed by watching. お好み焼きの状態を目視で確認します	ges43a	ges43r
Flip Okonomiyaki お好み焼きを裏返します	It reaches Hera. へらに手を伸ばします	ges12a	ges12r
	Hera's direction is changed. へらの方向を変えます	ges13a	ges13r
	Hera is lifted and Okonomiyaki is turned inside out. お好み焼きをへらを持ち上げて裏返します	ges14a	ges14r
	It returns it based on Hera's position. へらの位置を元に戻します	ges15a	ges15r
	It returns it based on Hera's direction. へらの方向を元に戻します	ges16a	ges16r
Take sauce ソースを手元に引き寄せます	It reaches the sauce. ソースに手を伸ばします	ges72a	ges72r
	The sauce is moved on the iron plate. ソースを鉄板の上に移動します	ges73a	ges73r
Put sauce on Okonomiyaki ソースをお好み焼きに塗ります	The sauce is inclined. ソースを傾けます	ges74a	ges74r
	It returns it based on the inclination of the sauce. ソースの傾きを元に戻します	ges75a	ges75r
	It returns it based on the position of the sauce. ソースの位置を元に戻します	ges76a	ges76r
Take seaweed(Nori) 海苔を手元に引き寄せます	It reaches the seaweed. 海苔に手を伸ばします	ges62a	ges62r
	The seaweed is moved on the iron plate. 海苔を鉄板の上に移動します	ges63a	ges63r

Procedure of dish	Content of bodily movement	Name of bodily movement	
		For Avator	For Robot
Put seaweed on Okonomiyaki 海苔をお好み焼きに振りかけます	The seaweed is inclined. 海苔を傾けます	ges64a	ges64r
	It returns it based on the inclination of the sauce. 海苔の傾きを元に戻します	ges65a	ges65r
	It returns it based on the position of the seaweed. 海苔の位置を元に戻します	ges66a	ges66r
Take Katsuobushi 鰹節を手元に引き寄せます	It reaches the Katsuobushi. 鰹節に手を伸ばします	ges52a	ges52r
	The Katsuobushi is moved on the iron plate. 鰹節を鉄板の上に移動します	ges53a	ges53r
Put Katsuobushi on Okonomiyaki 鰹節をお好み焼きに振りかけます	The Katsuobushi is inclined. 鰹節を傾けます	ges54a	ges54r
	It returns it based on the inclination of the Katsuobushi. 鰹節の傾きを元に戻します	ges55a	ges55r
	It returns it based on the position of the Katsuobushi. 鰹節の位置を元に戻します	ges56a	ges56r
Cut Okonomiyaki お好み焼きを切り分けます	—	—	—
Put Okonomiyaki on dish お好み焼きを皿にのせます	—	—	—
Reduce the heat of Teppan to low 鉄板のガスコンロの火を弱めます	It reaches the switch of the iron plate, and thermal power is weakened. 鉄板のスイッチに手を伸ばして、火力を弱めます。	ges01a	ges01r
Increase the heat of Teppan to high 鉄板のガスコンロの火を強めます	It reaches the switch of the iron plate, and thermal power is strengthened. 鉄板のスイッチに手を伸ばして、火力を強めます。	ges01a	ges01r
Put out the fire of Teppan 鉄板のガスコンロの火を消します	It reaches the switch of the iron plate, and thermal power is stopped. 鉄板のスイッチに手を伸ばして、火力をとめます。	ges01a	ges01r

The bodily movement changes the position of the one besides the movement of the body, and is rotated. Please confirm them. For instance, it is "Put ingredients on the Teppan".

身体動作は、身体の動きの他に、ものの位置を変えたり、回転をさせられます。それらも確認してください。例えば、それは「Put ingredients on the Teppan」です。



3.4. 対話に関する操作

The Okonomiyaki cooperation dish uses three abilities of SIGVerse. In this paragraph, it explains the operation of the conversation. This operation is done with SIGViewer. The list of the operation of the conversation and the operation method are written as follows.

お好み焼き協調料理は、SIGVerse の三つの能力を利用しています。この項では、対話の操作を説明します。この操作は SIGViewer で行います。以下に対話の操作の一覧と、操作の方法を記します。

3.4.1. お好み焼き協調料理の対話に関する処理

The operation of the Okonomiyaki cooperation dish using the ability of the conversation of SIGVerse inputs a certain objection to SIGViewer. This is the same as talking to Robot.

SIGVerse の対話の能力を使用した、お好み焼き協調料理の操作は、ある文言を SIGViewer に入力します。これはロボットに話しかけるのと同じです。

SIGVerse	The Okonomiyaki cooperation dish	Example of command of operation
Conversation	Avator and Robot conversation	Please look at the table below.
	Agent mutual conversation	*This is processing of internal. Therefore, it is not possible to operate it directly.

3.4.2. アバタとロボットの対話の操作

The conversation function of Avator and Robot is confirmed according to the following procedures.

アバタとロボットの対話機能は、以下の手順で確認します。これは油をとる様にロボットに話しかける例です。

Step0. SIGViewer and Okonomiyaki GUI are started.

Step1. "Initialize" is selected with Okonomiyaki GUI.

Step2. A supplementary level of Robot is set to "All" with Okonomiyaki GUI.

Step3. "Start" is selected with Okonomiyaki GUI.

Step4. "Commnad tab" of SIGViewer is selected.

Step5. It is input to "Selection for the transmission" column of SIGViewer as "Robot_000".

Step6. It is input to "SendMessage argument" column of SIGViewer as "TakeOil".

Step7. "Execution" button of SIGViewer is selected.

Step0. SIGViewer とお好み焼き GUI を起動します

Step1. お好み焼き GUI で「Initialize」を押下します

Step2. お好み焼き GUI でロボットの補助度を「全(all)」に設定します

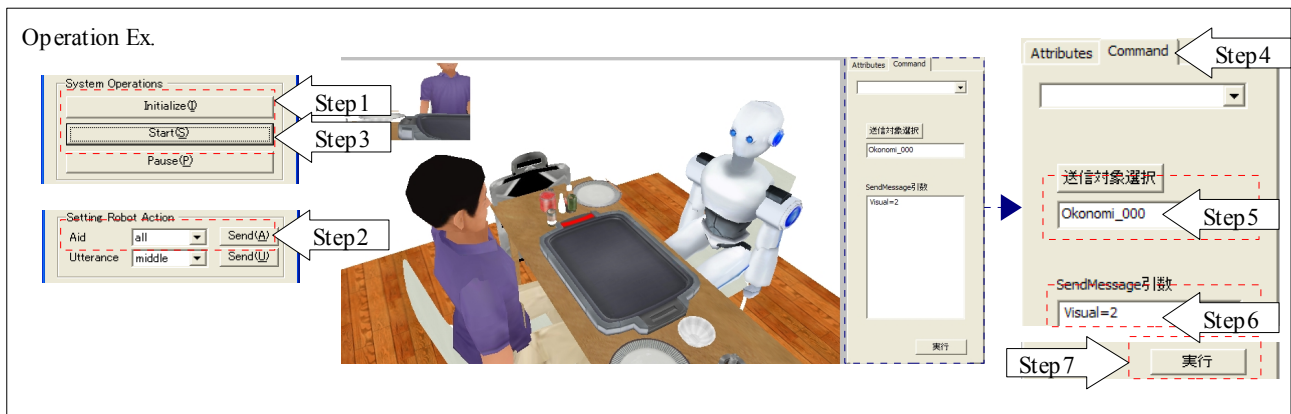
Step3. お好み焼き GUI で「Start」を押下します

Step4. SIGViewer の「Commnad タブ」を選択します

Step5. SIGViewer の「送信対象選択」欄に「Robot_000」と入力します

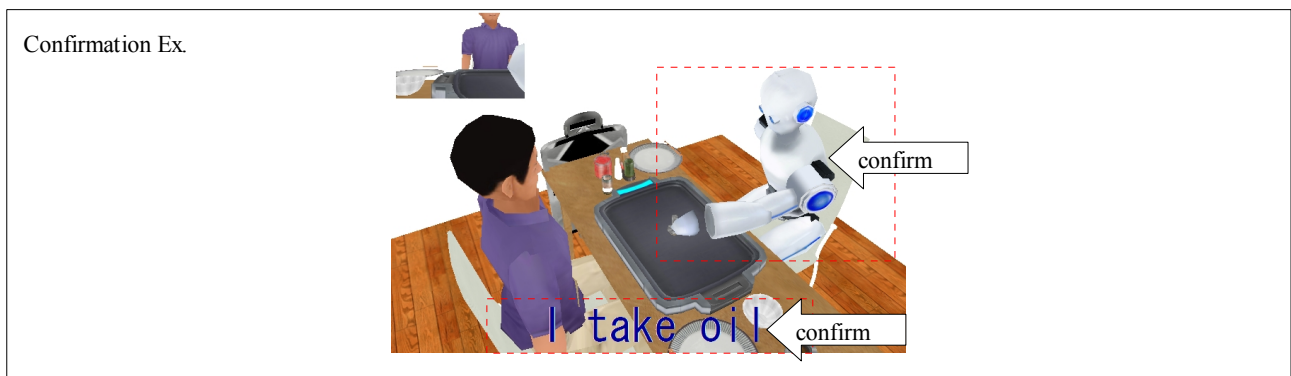
Step6. SIGViewer の「SendMessage 引数」欄に「TakeOil」と入力します

Step7. SIGViewer の「実行」ボタンを押下します



Whether it reached it to make remarks that Robot is "I take oil", and to take oil is confirmed.

ロボットが「I take oil」と発言し、油をとる為に手を伸ばしたか、確認します。



Robot catches the procedure of all dishes of Okonomiyaki. Those lists are written as follows.

ロボットは、お好み焼きの全ての料理の手順を聞き取ります。以下にそれらのリストを記します。

Procedure of dish	Dialog said to robot
Mix dough	MixDough
Take oil	TakeOil
Put oil on Teppan	PutOilOnTeppan
Check gas level	CheckGasLevel
Fire up Teppan	FireUpTeppan
Put ingredients on the Teppan	PutIngredientsOnTheTeppan
Take pork	TakePork
Put pork	PutPork
Check condition of roasted Okonomiyaki	CheckConditionOfRoastedOkonomiyaki
Flip Okonomiyaki	FlipOkonomiyaki
Take sauce	TakeSauce

Procedure of dish	Dialog said to robot
Put sauce on Okonomiyaki	PutSauceOnOkonomiyaki
Take seaweed(Nori)	TakeSeaweed
Put seaweed on Okonomiyaki	PutSeaweedOnOkonomiyaki
Take Katsuobushi	TakeKatsuobushi
Put Katsuobushi on Okonomiyaki	PutKatsuobushiOnOkonomiyaki
Cut Okonomiyaki	CutOkonomiyaki
Put Okonomiyaki on dish	PutOkonomiyakiOnDish
Put out the fire of Teppan	PutOutTheFireOfTeppan
Reduce the heat of Teppan to low	ReduceTheHeatOfTeppanToLow
Increase the heat of Teppan to high	IncreaseTheHeatOfTeppanToHigh

*When a supplementary level is only "All", Robot can receive the demand now.

※現在、ロボットは補助度が「全て」の時のみ、要求を受け取れます。

4. お好み焼き協調料理の設定

Contents

- ・お好み焼き協調料理の設定
 - －お好み焼き協調料理の設定
 - －知覚に関する設定
 - －力学に関する設定
 - －対話に関する設定

The Okonomiyaki cooperation dish has achieved "Externals change", "Rotation", "Movement", "Bodily movement", "Conversation of Avator and Robot", and "Agent mutual conversation" by using three abilities of SIGVerse. However, all those functions actually operate according to the definition file of CSV. In this paragraph, it explains the setting method of those definition files of CSV. And this setting is called "CSV programming" in the Okonomiyaki cooperation dish.

お好み焼き協調料理は、SIGVerse の三つの能力を利用して、「見た目変更」「回転」「移動」「身体動作」「アバタとエージェントの対話」「エージェント相互対話」を実現しています。ですが、実際にはそれらの全ての機能は、CSV の定義ファイルに従い動作しています。この項では、それら CSV の定義ファイルの設定方法について説明します。そして、この設定はお好み焼き協調料理上、「CSV プログラミング」と呼びます。

4.1. 状態遷移情報

The basis of the Okonomiyaki cooperation dish is a state transition with good CSV and compatibility. Therefore, one procedure of the Okonomiyaki dish corresponds in one state. For instance, "Mix dough" becomes "State to mix the dough", and "Take Oil" becomes "State to take oil". The Okonomiyaki cooperation dish sequentially changes these states next, and makes Okonomiyaki. The definition of this state transition is called "State transition information".

お好み焼き協調料理は、CSV と相性の良い状態遷移を基本としています。その為、お好み焼き料理の一つの手順が一つの状態に該当します。例えば、「Mix dough」は「生地を混ぜている状態」となり、「Take Oil」は「油をとっている状態」となります。お好み焼き協調料理は、これらの状態を順次に遷移させ、お好み焼きを作ります。この状態遷移の定義を、「状態遷移情報」と呼びます。

Procedure of dish	"State transition information"							
	before\after	0	1	2	3	4	5	6
0. Initial	0	ε	1	ε	ε	ε	ε	ε
1. Mix dough	1	ε	ε	1	ε	ε	1	ε
2. Take oil	2	ε	ε	ε	1	ε	1	ε
3. Put oil on the Teppan	3	ε	ε	ε	ε	1	1	ε
4. Check gas level	4	ε	ε	ε	ε	ε	1	ε
5. Fire up Teppan	5	ε	ε	ε	ε	ε	ε	1
6. Put ingredients on the Teppan	6	ε	ε	ε	ε	ε	ε	ε

*This is one example.

This is a state transition table. The line shows the previous state, and the row shows the following state. "ε" shows the impropriety of the state transition, and "one (numerical value)" shows possible of the state transition. In the okonomiyaki cooperation dish, the procedure of the dish of the left is expressed in the state transition table in this manner.

これは状態遷移表です。行は前の状態を表し、列は次の状態を表します。ε は状態遷移の不可を表し、1 (数値) は状態遷移の可能を表します。お好み焼き協調料理では、左の料理の手順をこの様に状態遷移表で表現します。

4.2. 全ての登場人物

In the Okonomiyaki cooperation dish, Avator and Robot appear. The above-mentioned state transition information is made by this unit. For instance, it is "State transition information on Avator", and "State transition information on Robot. ". And the unit is called "Agent". All agents who appear in the following with the Okonomiyaki cooperation dish are recorded.

お好み焼き協調理では、アバタとロボットが登場します。上記の状態遷移情報はこの単位で作成します。例えば、「アバタの状態遷移情報」、「ロボットの状態遷移情報」です。そして、その単位を「エージェント」と呼びます。下記にお好み焼き協調理で登場する全てのエージェントを記します。

Example of figure	All charactes(Agents)		
	Number in figure	Agent's Name	Name of agent who uses it on SIGViewer
	1	Avator	Avator_000
	2	Robot	Robot_000
	3	Oil	Abura_000
	4	Bowl	Bowl_000
	5	Iron plate	Teppan_000
	6	Okonomiyaki	Okonomi_000
	7	Sauce	Sauce_000
	8	Seaweed	Nori_000
	9	Katsuobushi	Katsuobushi_000

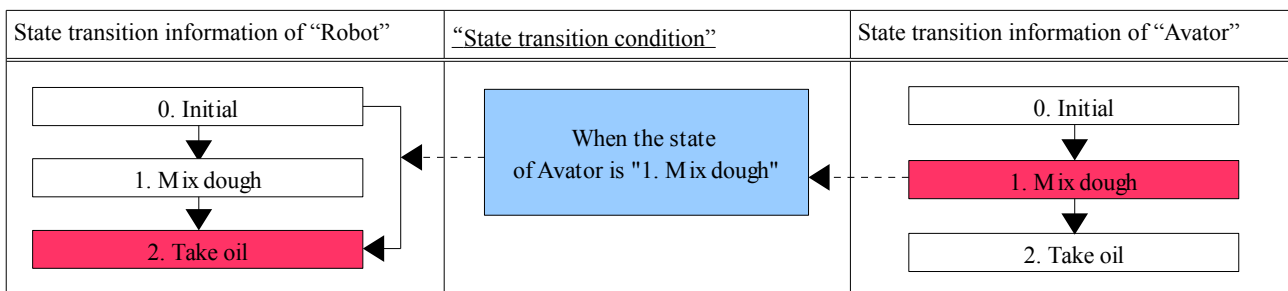
For instance, "Okonomi_000" is specified on SIGViewer when Okonomiyaki is turned inside out, and the Angle command is transmitted.

例えば、お好み焼きを裏返す場合、SIGViewer 上に「Okonomi_000」を指定し、Angle コマンドを送信します。

4.3. 状態遷移条件

In the Okonomiyaki cooperation dish, Robot sees the state of Avator and selects the action (state). In this case, how do you do? In that case, it imposes conditions on the transition of Robot state. For instance, Robot should change to "2. Take oil" when Avator is "1. Mix dough". The state transition of Robot is defined as "0. Initial" → "2. Take oil". However, condition "The state of Avator is 1. It is Mix dough" is added to this state transition. Robot can change on this condition in the "2. Take oil" state when Avator is "1. Mix dough". This condition is called "State transition condition" in the Okonomiyaki cooperation dish.

お好み焼き協調理では、ロボットはアバタの状態を見て行動(状態)を選択します。この場合はどうするのでしょうか。その場合は、ロボットの状態の遷移に条件を付けます。例えば、ロボットは、アバタが「1. Mix dough」の状態の場合、「2. Take oil」に遷移する必要があります。ロボットの状態遷移を「0. Initial」→「2. Take oil」と定義します。ただし、この状態遷移に条件「アバタの状態が 1.Mix dough である場合」を追加します。この条件により、ロボットは、アバタが「1. Mix dough」の状態である場合に、「2. Take oil」状態に遷移出来ます。この条件をお好み焼き協調理上、「状態遷移条件」と呼びます。



For instance, Robot assumes that it does "When Avator is '1. Mix dough', '2. Take oil is done'" and "'4. Check gas level' when Avator is '3. Put oil on the Teppan'". In this case, state transition information on Robot becomes the following tables.

例えば、ロボットは、「アバタが 1. Mix dough の場合、2. Take oil を行う」、「アバタが 3. Put oil on the Teppan の場合、4. Check gas level」を行うとします。この場合、ロボットの状態遷移情報は以下の表になります。

Procedure of dish	State transition information Of Robot							
	before\after	0	1	2	3	4	5	6
0. Initial	0	ε	ε	1	ε	ε	ε	ε
1. Mix dough	1	ε	ε	ε	ε	ε	ε	ε
2. Take oil	2	ε	ε	ε	ε	1	ε	ε
3. Put oil on the Teppan	3	ε	ε	ε	ε	ε	ε	ε
4. Check gas level	4	ε	ε	ε	ε	ε	ε	ε
	5	ε	ε	ε	ε	ε	ε	ε
	6	ε	ε	ε	ε	ε	ε	ε

The condition of "The state of Avator is 1" is put on this "0. Initial" → "2. Take oil". The condition of "The state of Avator is 3" is put on this "2. Take oil" → "4. Check gas level". This "The state of Avator is 3" is expressed as "State=3@Avator_000" in the Okonomiyaki cooperation dish. The example of the state transition condition is written as follows.

この「0. Initial」→「2. Take oil」に「アバタの状態が 1 である場合」の条件を付けます。「2. Take oil」→「4. Check gas level」に「アバタの状態が 3 である場合」の条件も付けます。この「アバタの状態が 3 である場合」は、お好み焼き協調理上、「State=3@Avator_000」と表現します。以下にその状態遷移条件の例を記します。

Procedure of dish	"State transition condition Of Robot"							
	before\after	0	1	2	3	4	5	6
0. Initial It set condition.	0	ε	ε	State=1@Avator_000	ε	ε	ε	ε
1. Mix dough	1	ε	ε	ε	ε	ε	ε	ε
2. Take oil	2	ε	ε	ε	ε	State=3@Avator_000	ε	ε
3. Put oil on the Teppan	3	ε	ε	ε	ε	ε	ε	ε
4. Check gas level It set condition.	4	ε	ε	ε	ε	ε	ε	ε
	5	ε	ε	ε	ε	ε	ε	ε
	6	ε	ε	ε	ε	ε	ε	ε

*This "State transition information" and "State transition condition" are defined with **two CSV files**. These two are not same information.

※この「状態遷移情報」と「状態遷移条件」は二つの CSV ファイルで定義します。これら二つは同じ情報ではありません。

In the Okonomiyaki cooperation dish, Robot assists the dish of Avator by using this "State transition information" and "State transition condition".

お好み焼き協調理では、ロボットはこの「状態遷移情報」と「状態遷移条件」を用いてアバタの料理を補助します。

4.4. 状態遷移処理

In the Okonomiyaki cooperation dish, Avator or Robot does various actions along with the state transition. For instance, it is "Oil is taken", and "Okonomiyaki is turned inside out. ", etc. Three abilities of SIGVerse are used, and in the CSV programming, "Rotation", "Movement", and "Bodily movement" are done, and cooked at the state transition. It is called "State transition processing" in the Okonomiyaki cooperation dish.

For instance, Robot executes quite the same command as "Move" input by operating the situation and "Oil is taken" "3.2. Operation of mechanics". As a result, oil can be taken. The state transition processing example is written as follows.

お好み焼き協調料理では、アバタまたはロボットは、状態遷移に伴い、色々な行動をします。例えば、「油をとる」、「お好み焼きを裏返す」などです。CSVプログラミングでは、SIGVerseの三つの能力を利用して、状態遷移の時に「回転」、「移動」、「身体動作」を行い、料理をします。それをお好み焼き協調料理上、「状態遷移処理」と呼びます。

例えば、ロボットは「油をとる」場合、「3. Operation of the Okonomiyaki cooperation dish」の操作で入力した「Move」と全く同じコマンドを実行します。それにより油をとれます。以下に、状態遷移処理の例を記します。

Procedure of dish	"State transition processing" of Robot		
	State	State transition processing	Content of processing
0. Initial ←	0	—	(none)
1. Mix dough ←	1	Gesture=ges92r, State=00X2@Okonomi_000	It reaches the ingredients. Okonomiyaki is put into the mixing state. 生地 に手を伸ばします お好み焼きを混ぜた状態にします
2. Take oil ←	2	Gesture=ges82r, Move=100:80:100@Abura_000, State=1@Abura_000	It reaches oil. Oil is moved on the iron plate. Oil is made used. 油 に手を伸ばします 油を鉄板の上に移動します 油を使用済みの状態にします
3. Put oil on the Teppan ←	3	Gesture=ges84r, Move=90:80:100@Abura_000,	The hand is moved on the iron plate. Oil is moved right and left. 鉄板 の上で左右に手を動かします 油を左右に移動します
4. Check gas level ←	4	Gesture=ges08r	The gas remainder amount meter is seen. ガス残量計 を見ます
5. Fire up Teppan ←	5	Gesture=ges01r, Attr=2@Tepan_000	It reaches the switch of the gas. Thermal power of Teppan is strengthened. ガス のスイッチに手を伸ばします 鉄板の火力を強めます
6. Put ingredients on the Teppan ←	6	Gesture=ges72r,Gesture=ges74r, Move=100:80:100@Bowl_000, Angle=0:0:1:0:90:10@Bowl_000, State=00A@Okonomi_000	The ingredients is put on the iron plate. The bowl is moved on the iron plate. The bowl is inclined. It puts it into the state to burn Okonomiyaki. 生地 を鉄板にのせます ボウルを鉄板の上に移動します ボウルを傾けます お好み焼きを焼いている状態にします

*This is one example.

A lot of "@" is written in the state transition processing of the right. This is a meaning said that the command will be transmitted to the agent since "@". As for Move=100:80:100@Abura_000, Robot transmits "Move=100:80:100" to oil.

右の状態遷移処理には、沢山の「@」が書かれています。これは「@」以降のエージェントにコマンドを送信する、という意味です。Move=100:80:100@Abura_000は、ロボットが油に「Move=100:80:100」を送信します。

*A part of setting is omitted.

The Okonomiyaki cooperation dish has been achieved by these "State transition information", "State transition condition", and "State transition processing".

お好み焼き協調料理は、これら「状態遷移情報」、「状態遷移条件」、「状態遷移処理」で実現されています。

4.5. お好み焼き協調料理の設定

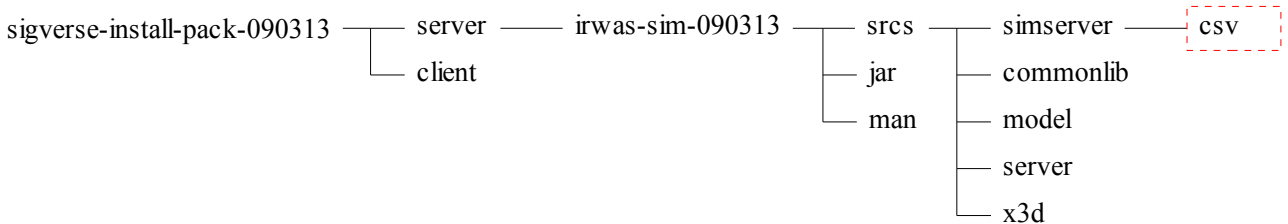
In this paragraph, it explains a concrete definition of the Okonomiyaki cooperation dish and the content. These definitions are done to the definition file on Central Server. Central Server is Linux system. Please prepare a general SSH client.

本項では、お好み焼き協調料理の具体的な定義とその内容について説明します。これらの定義はセントラルサーバ上の定義ファイルに対して行います。セントラルサーバは Linux システムです。一般的な SSH クライアントを用意してください。

4.5.1. CSV プログラミングの環境

Various definition files of the Okonomiyaki cooperation dish are preserved in directory "csv" under "simserver" that Central Server starts. It is in the SIGVerse directory. The sample of Figure is written as follows.

お好み焼き協調料理の各種の定義ファイルは、セントラルサーバが起動する「simserver」の下のディレクトリ「csv」に保存されています。それは SIGVerse のディレクトリの中です。以下に図例を記します。



The list of the definition file that moves to "simserver/csv" and is preserved is displayed. Please operate as follows, and confirm the definition file.

simserver/csv に移動して保存されている定義ファイルの一覧を表示します。以下の操作を行い、定義ファイルを確認してください。

```
Operation Ex.      cd $HOME/sigverse-install-pack-090313/server/irwas-sim-090313/srcs/simserver/csv
                  ls
```

```
Confirmation Ex.
Abura_000_attribute.csv  Abura_000_condition.csv  Abura_000_state.csv      Abura_000_transition.csv
Avator_000_attribute.csv Avator_000_condition.csv Avator_000_state.csv     Avator_000_transition.csv
Okonomi_000_attribute.csv Okonomi_000_condition.csv Okonomi_000_state.csv    Okonomi_000_transition.csv
                        :
```

These definition files of each agent are defined a little as four. It explains the content set as the role of these definition files as follows. This is very important information. These four definition files are always necessary.

これらの定義ファイルは、エージェント毎に四つと少し定義されています。以下にこれらの定義ファイルの役割と設定する内容を説明します。これはとても重要な情報です。これら四つの定義ファイルは、必ず必要です。

Name of file	Type name	Role of definition	Content of definition
*_state.csv	state	State transition information 状態遷移情報	Each agent's state transition "information" is defined by CSV. 各エージェントの状態遷移「情報」を CSV 形式で定義します。
*_condition.csv	condition	State transition condition 状態遷移条件	Each agent's state transition "condition" is defined by CSV. 各エージェントの状態遷移「条件」を CSV 形式で定義します。
*_transition.csv	transition	State transition processing 状態遷移処理	Each agent's state transition "processing" is defined by CSV. 各エージェントの状態遷移「処理」を CSV 形式で定義します。
*_attribute.csv	attriute	Agent attribute information エージェント属性情報	Each agent's attribute is defined by CSV. 各エージェントの属性を CSV 形式で定義します。

There are two definition files besides these four definition files. This is used if it is necessary. The file is "Definition file of externals" and "Supplementary file for the conversation." Either file is also optional.

これら四つの定義ファイル以外に、二つの定義ファイルがあります。これは必要な場合に使用します。そのファイルは「見た目の定義ファイル」と「対話用補助ファイル」です。どちらのファイルもオプションです。

Name of file	Type name	Role of definition	Content of definition
*_visual.csv	visual	Externals definition 見た目定義	Each agent's externals are defined. 各エージェントの見た目を定義します
*_texts.csv	texts	Supplementary definition for conversation 対話用補助定義	It is a definition of assistance that ties to the state the natural language. 自然言語を状態と結びつける補助の定義です。

All these definition files are described by Comma Separated Value(CSV). It is possible to inspect it with general spreadsheet wear because it describes it by Comma Separated Value(CSV). The sample of Figure of the edit of state transition information is written as follows.

これらの全ての定義ファイルは、CSV形式で記述します。CSV形式で記述する為、一般的な表計算ソフトウェアで閲覧可能です。

4.5.2. 全ての登場人物の設定

All Agent is described in "agents.csv" of "simserver/csv" and one agent is described in one line. When the state changes, Robot confirms other agents' states. It becomes set of the agent who confirms the state at that time. Moreover, it becomes the agent of the object of initialization to which agent's special state transition adjusts. Please describe all agent's appearing names. The definition of the Okonomiyaki cooperation dish as the description example is written as follows.

全ての Agent は、「simserver/csv」の「agents.csv」に1エージェントを1行に記述します。ロボットは、状態遷移時に他のエージェントの状態を確認します。その時に状態を確認する対象のエージェントの設定になります。また、エージェントの特殊な状態遷移の適応を行う初期化の対象のエージェントにもなります。登場する全てのエージェントの名前を記述してください。以下に記述例としてのお好み焼き協調料理の定義を記します。

Definition Ex.	Avator_000 Robot_000 Okonomi_000 Abura_000 Sauce_000 Katsuobushi_000 Nori_000 Teppan_000 Bowl_000	} 1 agent difinition(1 row)
----------------	---	-----------------------------

4.5.3. 補助度毎の定義

All agents can define the state transition of each supplementary level. For instance, when a supplementary level is "Entire (=3)", "Test_state3.csv", "Test_condition3.csv", and "Test_transition3.csv" can be made. Information and the naming convention that can make each supplementary level are written as follows.

全てのエージェントは、補助度ごとに状態遷移を定義できます。例えば、補助度が「全て(=3)」の場合、「Test_state3.csv」、「Test_condition3.csv」、「Test_transition3.csv」を作成できます。以下に補助度ごとに作成可能な情報と、命名規則を記します。

Name of file	Type name	Naming convention	Definition example
*_state.csv	state	"Agent name" + "_state" + "Supplementary level" + ".csv"	Robot_000_state2.csv
*_condition.csv	condition	"Agent name" + "_condition" + "Supplementary level" + ".csv"	Robot_000_condition2.csv
*_transition.csv	transition	"Agent name" + "_transition" + "Supplementary level" + ".csv"	Robot_000_transition2.csv

*When the corresponding file is not found, files except a supplementary level are used.

※もし該当するファイルがない場合、「補助度」を除いたファイル名を使用します

4.5.4. 属性の設定

"Direction immediately after the start", "Flag whether it is Avator or Robot", and "An initial value of a supplementary level and the remark level", etc. are described in "*_attribute.csv". The definition format and example of the description are written as follows.

「*_attribute.csv」には「起動直後の方向」、「アバタかロボットであるかのフラグ」、「補助度や発言度の初期の値」などを記述します。以下に定義の書式と記述例を記します。

Item name	Number of col	Content of item	Type of item	Input example
Initial state	1	An initial "state" name is input. 初期の状態の名前を入力します。	Alphanumeric character in 48 bytes or less	00X
Avator flag	2	1 is input for "Avator". Besides 0 is input. The agent gives priority to "Existing state of things" when there are the state transition ahead plurals when specifying it for Avator. アバタの場合は1、それ以外は0を指定します。アバタに指定した場合、エージェントは状態遷移先が複数ある時、「現在の状態」を優先します。	0 or 1	0
Robot flag	3	1 is input for "Robot". Besides 0 is input. The agent gives priority to "Excluding existing state of things" when there are the state transition ahead plurals when specifying it for the Robot. ロボットの場合は1、それ以外は0を指定します。ロボットに指定した場合、エージェントは状態遷移先が複数ある時、「現在の状態以外」を優先します。	0 or 1	0
Passive flag	4	1 is input for "Passive". Besides 0 is input. The agent doesn't do an autonomous state transition at all when specifying it for Passive. Passive の場合は1、それ以外は0を指定します。Passive に指定した場合、エージェントは自律的な状態遷移を一切行いません。	0 or 1	1
Attribute	5	The agent has an attribute free only by one. An initial value of the attribute is input. エージェントは一つだけ自由な属性を持ちます。その属性の初期の値を入力します。	Alphanumeric character in 48 bytes or less	0
Aid	6	An initial supplementary level is input. 初期の補助度を入力します。	Numerical value	1
Uttr	7	An initial remark level is input. 初期の発言度を入力します。	Numerical value	2
Y-Angle	8	The angle in an initial direction of Y is input in the form of 360 degrees. 初期の Y 方向の角度を 360 度の形式で入力します。	Numerical value (0-360)	180
Req	9	It doesn't use it in the Okonomiyaki cooperation dish now. It is for the future. 現在、お好み焼き協調料理では使いません。将来の為です。	Numerical value	1
X-Angle	10	The angle in an initial direction of Y is input in the form of 360 degrees. 初期の X 方向の角度を 360 度の形式で入力します。	Numerical value (0-360)	180
Z-Angle	11	The angle in an initial direction of Y is input in the form of 360 degrees. 初期の Z 方向の角度を 360 度の形式で入力します。	Numerical value (0-360)	180

For instance, Avator=1 is set to Avator_000_attribute.csv of "Avator_000" that is Avator. The example of setting Avator is written as follows.

例えば、アバタである「Avator_000」の Avator_000_attribute.csv には、Avator=1 を設定します。以下にアバタの設定例を記します。

Agent's name	Initial state	Avator flag	Robot flag	Passive flag	Initial Attr	Initial Aid	Initial Uttr	Initial AngleY	Initial req(opt)	Initial AngleX	Initial AngleZ
Avator_000	0	1	0	0	0	0	1	180	—	—	—
Definition Ex. state,avator,robot,passive,attr,aid,uttr,angle 0,1,0,1,0,0,180											
*The first line is a line of the comment.											

The first line of "*_attribute.csv" is a line of the comment. This line is not used.
Moreover, All agent's attributes are written as follows.

「*_attribute.csv」の先頭の行は、コメントの行です。この行は使われません。以下に、全てのエージェントの属性を記します。

Agent's name	Initial state	Avator flag	Robot flag	Passive flag	Initial Attr	Initial Aid	Initial Uttr	Initial AngleY	Initial req(opt)	Initial AngleX	Initial AngleZ
Avator_000	0	1	0	0	0	0	0	180	—	—	—
Robot_000	0	0	1	0	0	0	1	0	0	0	0
Abura_000	0	0	0	1	0	0	0	0	—	—	—
Bowl_000	0	0	0	1	0	0	0	0	—	—	—
Teppan_000	0	0	0	0	2	0	0	0	—	—	—
Okonomi_000	00X1	0	0	0	0	0	0	0	—	—	—
Sauce_000	0	0	0	1	0	0	0	0	—	—	—
Nori_000	0	0	0	1	0	0	0	0	—	—	—
Katsuobushi_000	0	0	0	1	0	0	0	0	—	—	—

Passive Agent (受動的なエージェント)とは?

The agent of the Okonomiyaki cooperation dish is greatly classified into two. It is "Active agent" and "Passive agent." An active agent is another agent it or an agent who depends on the passage of time at the state transition. A passive agent doesn't depend for the passage of another agent and time at the state transition.

For instance, Okonomiyaki scorches when it is burnt at long time and it withers. It is "Active agent." because it depends at time when changing to this "Scorching state". Similarly, because "State of the temperature" changes in the time passage, the iron plate is "Active agent." Oppositely, "Abura (Oil)" and "Seaweed" that the state doesn't change even if time passes are "Passive agent."

お好み焼き協調理のエージェントは、大きく二つに分類されます。「能動的なエージェント」と「受動的なエージェント」です。能動的なエージェントは、状態遷移の時に、他エージェントまたは時間経過に依存するエージェントです。受動的なエージェントは、状態遷移の時に、他エージェントや時間経過の依存がありません。

例えば、お好み焼きは、長い時間焼かれると焦げます。この「焦げる状態」に遷移する場合、時間に依存している為、「能動的なエージェント」です。同様に、鉄板は時間の経過で「温度の状態」が変化するために、「能動的なエージェント」です。逆に、時間が経過しても状態が変化しない「Abura (Oil)」、「Seaweed」は「受動的なエージェント」です。

Active Agents(Passive=0)		Passive Agents (Passive=1)	
It depends on other agents' states.	It depends at time.	There is no dependence.	
<div style="border: 1px solid black; padding: 5px; width: 100px; margin: 5px auto;">Avator_000</div> <div style="border: 1px solid black; padding: 5px; width: 100px; margin: 5px auto;">Robot_000</div>	<div style="border: 1px solid black; padding: 5px; width: 100px; margin: 5px auto;">Teppan_000</div> <div style="border: 1px solid black; padding: 5px; width: 100px; margin: 5px auto;">Okonomi_000</div>	<div style="border: 1px solid black; padding: 5px; width: 100px; margin: 5px auto;">Abura_000</div> <div style="border: 1px solid black; padding: 5px; width: 100px; margin: 5px auto;">Bowl_000</div>	<div style="border: 1px solid black; padding: 5px; width: 100px; margin: 5px auto;">Sauce_000</div> <div style="border: 1px solid black; padding: 5px; width: 100px; margin: 5px auto;">Nori_000</div> <div style="border: 1px solid black; padding: 5px; width: 100px; margin: 5px auto;">Katsuobushi_000</div>

4.6.1. 状態遷移情報の定義

Agent's state is defined in `*_state.csv`. For instance, it is a procedure of the Okonomiyaki cooperation dish for Avator_000. This state is a meaning said, "Phased procedure". All agent's "List of the state" and "Definition of the state" are written as follows. This is a specification. Please actually confirm it with the text editor such as vi.

「*_state.csv」には、エージェントの状態を定義します。例えば、Avator_000 の場合は、お好み焼き協調料理の手順です。この状態は、「段階的な手順」と言う意味です。以下に全てのエージェントの「状態の一覧」と「状態の定義」を記します。これは仕様です。実際に vi などのテキストエディタで確認してください。

4.6.1.1. アバタの状態の一覧

Content in state 状態の内容	Name in state 状態の名前	Content in state 状態の内容	Name in state 状態の名前
The ingredients is mixed. 生地を混ぜている状態	1	The source is put on Okonomiyaki. ソースをお好み焼きに塗っている状態	13
Oil is drawn on the iron plate. 油を手元に引き寄せている状態	2	Seaweed is taken. 海苔を手元に引き寄せている状態	14
Oil is painted on the iron plate. 油を鉄板にひいている状態	3	Seaweed is put on Okonomiyaki. 海苔をお好み焼きに振りかけている状態	15
The residual quantity of the gas of the iron plate is confirmed. 鉄板のコンロのガス残量を確認している状態	4	The dried bonito(Katsuobushi) is taken. 鰹節を手元に引き寄せている状態	16
The fire is set fire to the iron plate. 鉄板のガスコンロに火を付けている状態	5	The dried bonito(Katsuobushi) is put on Okonomiyaki. 鰹節をお好み焼きに振りかけている状態	17
The ingredients is put on the iron plate. 生地を鉄板の上のせている状態	6	Okonomiyaki is divided. お好み焼きを切り分けている状態	18
Pork is taken. 豚肉を手元に引き寄せている状態	7	Okonomiyaki is piled up in the plate. お好み焼きを皿に盛っている状態	19
Pork is put on the ingredients. 豚肉を生地の上のせている状態	8	The fire of the iron plate is weakened. 鉄板のガスコンロの火を弱めている状態	20
The combustion addition and subtraction of Okonomiyaki is seen. お好み焼きの焼け加減を見ている状態	9	The fire of the iron plate is strengthened. 鉄板のガスコンロの火を強めている状態	21
Okonomiyaki is turned inside out. お好み焼きを裏返している状態	10	The fire of the iron plate is put out. 鉄板のガスコンロの火を消している状態	22
The source is taken. ソースを手元に引き寄せている状態	12		

4.6.1.2. アバタの状態遷移情報の定義

before\after	0	1	2	3	4	5	6	7	8	9	10	12	13	14	15	16	17	18	19	20	21	22	
0	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	
1	ε	0	10	ε	10	30	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	
2	ε	ε	0	30	10	30	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	
3	ε	ε	ε	0	10	30	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	
4	ε	ε	ε	ε	0	30	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	
5	ε	ε	ε	ε	ε	0	30	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	10	10	10
6	ε	ε	ε	ε	ε	ε	0	60	ε	30	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	10	10	10

4.6.1.3. ロボットの状態の一覧

"List of the state" of Robot and "Definition of the state" are written as follows. "Definition of the state" of Robot is the same as Avator.

以下にロボットの「状態の一覧」と「状態の定義」を記します。ロボットの「状態の定義」はアバタと同じです。

Content in state 状態の内容	Name in state 状態の名前	Content in state 状態の内容	Name in state 状態の名前
The ingredients is mixed. 生地を混ぜている状態	1	The source is put on Okonomiyaki. ソースをお好み焼きに塗っている状態	13
Oil is drawn on the iron plate. 油を手元に引き寄せている状態	2	Seaweed is taken. 海苔を手元に引き寄せている状態	14
Oil is painted on the iron plate. 油を鉄板にひいている状態	3	Seaweed is put on Okonomiyaki. 海苔をお好み焼きに振りかけている状態	15
The residual quantity of the gas of the iron plate is confirmed. 鉄板のコンロのガス残量を確認している状態	4	The dried bonito(Katsuobushi) is taken. 鰹節を手元に引き寄せている状態	16
The fire is set fire to the iron plate. 鉄板ノガスコンロに火を付けている状態	5	The dried bonito(Katsuobushi) is put on Okonomiyaki. 鰹節をお好み焼きに振りかけて状態	17
The ingredients is put on the iron plate. 生地を鉄板の上ののせている状態	6	Okonomiyaki is divided. お好み焼きを切り分けている状態	18
Pork is taken. 豚肉を手元に引き寄せている状態	7	Okonomiyaki is piled up in the plate. お好み焼きを皿に盛っている状態	19
Pork is put on the ingredients. 豚肉を生地の上ののせている状態	8	The fire of the iron plate is weakened. 鉄板のガスコンロの火を弱めている状態	20
The combustion addition and subtraction of Okonomiyaki is seen. お好み焼きの焼け加減を見ている状態	9	The fire of the iron plate is strengthened. 鉄板のガスコンロの火を強めている状態	21
Okonomiyaki is turned inside out. お好み焼きを裏返している状態	10	The fire of the iron plate is put out. 鉄板のガスコンロの火を消している状態	22
The source is taken. ソースを手元に引き寄せている状態	12		

4.6.1.4. ロボットの補助度

Information on the state transition of Robot is different according to assistance. The purpose of this is for a supplementary level to do only assistance (advice) by the situation and "low" remark. Moreover, when a supplementary level is "middle", assistance (advice) by the remark and the dish are assisted. Moreover, when a supplementary level is "high", the assistance of assistance (advice) by the remark and the dish and the assistance after advice are done. It is "All" that Robot all cooks now. Robot has information on three kinds of state transitions in these for a different state transition. The sample of Figure is written as follows.

ロボットの状態遷移の情報は、補助により異なります。これは、補助度が「低」の場合、発言による補助(アドバイス)のみを行う為です。また、補助度が「中」の場合、発言による補助(アドバイス)と料理の補助を行います。また、補助度が「高い」場合、発言による補助(アドバイス)と、料理の補助と、アドバイスの後の補助を行います。後は全てロボットが料理する「全て」です。これらは異なる状態遷移の為、ロボットは3種類の状態遷移の情報を持ちます。以下に図例を記します。

Supplementary level	low	middle	high	all	
Assistance by remark 発言による補助	√	√	√ + ○	√ + □	○ The dish is helped when there is no reaction even if advising 助言に反応しない場合、補助します
Assistance of dish 料理の補助		√	√	√	□ The reward is said when helped. 手伝われた場合、お礼を言います

4.6.1.5. ロボットの状態遷移情報の定義(補助度:低)

Robot sees the state of Avator and does suitable "Assistance by the remark".

ロボットは、アバタの状態を見て適した「発言による補助」を行います。

before/after	0	1	2	3	4	5	6	7	8	9	10	12	13	14	15	16	17	18	19	20	21	22
0	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
1	ε	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
2	ε	ε	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
3	ε	ε	ε	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
4	ε	ε	ε	ε	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
5	ε	ε	ε	ε	ε	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
6	ε	ε	ε	ε	ε	ε	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
7	ε	ε	ε	ε	ε	ε	ε	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
8	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
9	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
12	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε
13	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	ε	ε	ε	ε	ε	ε	ε
14	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	ε	ε	ε	ε	ε	ε
15	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	ε	ε	ε	ε	ε
16	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	ε	ε	ε	ε
17	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	ε	ε	ε
18	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	ε	ε
19	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10
20	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
21	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
22	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε

4.6.1.7. ロボットの状態遷移情報の定義(補助度:中)

Robot sees the state of Avator and does suitable "Assistance by the remark" and "Assistance of the dish".

ロボットは、アバタの状態を見て適した「発言による補助」と「料理の補助」を行います。

before/after	0	1	2	3	4	5	6	7	8	9	10	12	13	14	15	16	17	18	19	20	21	22
0	0	90	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
1	ε	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
2	ε	ε	0	90	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
3	ε	ε	ε	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
4	ε	ε	ε	ε	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
5	ε	ε	ε	ε	ε	0	90	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
6	ε	ε	ε	ε	ε	ε	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	10	10
7	ε	ε	ε	ε	ε	ε	ε	0	90	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	10	10
8	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	10	10
9	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	90	10	ε	ε	ε	ε	ε	ε	ε	ε	10	10

before/after	0	1	2	3	4	5	6	7	8	9	10	12	13	14	15	16	17	18	19	20	21	22
10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	ε	ε	ε	ε	ε	ε	10	10	ε
12	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	90	10	ε	ε	ε	ε	ε	10	10	ε
13	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	ε	ε	ε	ε	10	10	ε
14	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	90	10	ε	ε	ε	10	10	ε
15	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	ε	ε	10	10	ε
16	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	90	10	ε	10	10	ε
17	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	10	10	ε
18	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	90	10	10	10
19	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	10	10
20	ε	ε	ε	ε	ε	ε	0	0	0	0	0	0	0	0	0	0	0	10	0	0	10	10
21	ε	ε	ε	ε	ε	ε	0	0	0	0	0	0	0	0	0	0	0	10	0	10	0	10
22	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε

4.6.1.8. ロボットの状態遷移情報の定義(補助度:高)

Robot sees the state of Avator and does suitable "Assistance by the remark" and "Assistance of the dish". In addition, when Avator doesn't do anything for a fixed time, the dish is done.

ロボットは、アバタの状態を見て適した「発言による補助」と「料理の補助」を行います。更にアバタが一定時間、何もしない場合は、その料理を行います。

before/after	0	1	2	3	4	5	6	7	8	9	10	12	13	14	15	16	17	18	19	20	21	22
0	0	90	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
1	ε	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
2	ε	ε	0	90	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
3	ε	ε	ε	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
4	ε	ε	ε	ε	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
5	ε	ε	ε	ε	ε	0	90	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
6	ε	ε	ε	ε	ε	ε	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	10	10	ε
7	ε	ε	ε	ε	ε	ε	ε	0	90	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	10	10	ε
8	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	10	10	ε
9	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	90	10	ε	ε	ε	ε	ε	ε	ε	10	10	ε
10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	ε	ε	ε	ε	ε	ε	10	10	ε
12	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	90	10	ε	ε	ε	ε	ε	10	10	ε
13	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	ε	ε	ε	ε	10	10	ε
14	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	90	10	ε	ε	ε	10	10	ε
15	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	ε	ε	10	10	ε
16	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	90	10	ε	10	10	ε
17	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	10	10	ε
18	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	90	10	10	10
19	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	10	10
20	ε	ε	ε	ε	ε	ε	0	0	0	0	0	0	0	0	0	0	0	10	0	0	10	10
21	ε	ε	ε	ε	ε	ε	0	0	0	0	0	0	0	0	0	0	0	10	0	10	0	10
22	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε

4.6.1.9. ロボットの状態遷移情報の定義(補助度:全)

Robot cooks without seeing the state of Avator. If Avator helps, the reward is said.

ロボットは、アバタの状態を見ずに、料理します。もしアバタが手伝う場合は、お礼を言います。

before/after	0	1	2	3	4	5	6	7	8	9	10	12	13	14	15	16	17	18	19	20	21	22
0	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
1	ε	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
2	ε	ε	0	30	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
3	ε	ε	ε	0	10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
4	ε	ε	ε	ε	0	30	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
5	ε	ε	ε	ε	ε	0	0	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
6	ε	ε	ε	ε	ε	ε	0	60	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
7	ε	ε	ε	ε	ε	ε	ε	0	0	30	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
8	ε	ε	ε	ε	ε	ε	ε	ε	0	30	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
9	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	ε	ε	ε	ε	ε	ε	ε	10	10	ε
10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	10	ε	10	10	60	ε	ε	ε	ε
12	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	10	ε	10	ε	60	ε	ε	ε	ε
13	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	10	ε	60	ε	ε	ε	ε
14	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	10	ε	60	ε	ε	ε	ε
15	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	60	ε	ε	ε	ε
16	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	ε	ε	ε	ε
17	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	60	ε	ε	ε	ε
18	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	10	ε	ε	10
19	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	ε	ε	10
20	ε	ε	ε	ε	ε	0	ε	ε	ε	0	0	ε	ε	ε	ε	0	0	ε	ε	0	10	10
21	ε	ε	ε	ε	ε	0	ε	ε	ε	0	0	ε	ε	ε	ε	0	0	ε	ε	10	0	10
22	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0

4.6.1.10. お好み焼きの状態の一覧

The state of Okonomiyaki is made in the class of three variables. It is a state of combustion in A respect, a state of combustion in B respect, and a class in other states. The composition is written as follows.

お好み焼きの状態は、三つの変数の組で作成しています。それは、A面の焼き状態、B面の焼き状態、その他の状態の組です。以下にその構成を記します。

Okonomiyaki [a, b, d] a : 0 / 1 / 2 ≡ A respect is a raw. / It grills up. / It is burned. b : 0 / 1 / 2 ≡ B respect is a raw. / It grills up. / It is burned. d : a / b / X _n ≡ A respect is on. / B respect is on. / other state.	X ₁ : Before mixing okonomiyaki(inredients + tools) X ₂ : After mixing okonomiyaki X ₃ : Both sides of okonomiyaki are burnt. X ₄ : Seasoning is completed. X ₅ : Division is completed. X ₆ : Dishing up is completed.
---	--

Content in state 状態の内容	Name in state 状態の名前	Content in state 状態の内容	Name in state 状態の名前
State before ingredients mixes with tool 生地と具が混ざる前の状態	00X1	One side of Okonomiyaki scorched further. お好み焼きの更に片面が焦げた状態	21B

Content in state 状態の内容	Name in state 状態の名前	Content in state 状態の内容	Name in state 状態の名前
State of seed after ingredients and tool are mixed 生地と具を混ぜた後のタネ状態	00X2	One side scorches and one side is burnt in addition moderately. 片面が焦げ更に片面は適度に焼けている状態	12B
The ingredients is put on the iron plate and one side burns. タネを鉄板の上にひき片面を焼いている状態	00A	One side scorches and, in addition, one side scorches. 片面が焦げ更に片面が焦げている状態	22B
The ingredients is put on the iron plate and one side is burnt. タネを鉄板の上にひき片面が焼けた状態	01A	The seasoning was put and seasoning was completed. 調味料をかけた味付けが完了した状態	abX4
The ingredients was put on the iron plate and one side scorched. タネを鉄板の上にひき片面が焦げた状態	02A	The division of Okonomiyaki was completed. お好み焼きの切り分けが完了した状態	abX5
Okonomiyaki was turned inside out by using Hera. ヘラを用いてお好み焼きを裏返した状態	01B	It is a state that finished being piled up as for Okonomiyaki. お好み焼きを盛り終えた状態	abX6
One side of Okonomiyaki is burnt further. お好み焼きの更に片面が焼けた状態	11B		

4.6.1.11. お好み焼きの状態遷移情報の定義

before\after	00X1	00X2	00A	01A	02A	01B	02B	11B	12B	21B	22B	abX3	abX4	abX5	abX6
00X1	0	0	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
00X2	ε	0	0	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
00A	ε	ε	0	300	600	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
01A	ε	ε	ε	0	300	0	ε	ε	ε	ε	ε	ε	ε	ε	ε
02A	ε	ε	ε	ε	0	ε	0	ε	ε	ε	ε	ε	ε	ε	ε
01B	ε	ε	ε	ε	ε	0	ε	300	ε	600	ε	ε	ε	ε	ε
02B	ε	ε	ε	ε	ε	ε	0	ε	300	ε	600	ε	ε	ε	ε
11B	ε	ε	ε	ε	ε	ε	ε	0	ε	300	ε	0	ε	ε	ε
12B	ε	ε	ε	ε	ε	ε	ε	ε	0	ε	300	0	ε	ε	ε
21B	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	300	0	ε	ε	ε
22B	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	0	ε	ε	ε
abX3	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	0	0	ε
abX4	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	0	ε
abX5	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0	0
abX6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0

4.6.1.12. 鉄板の状態の一覧

The iron plate has the state that depends on the temperature of the iron plate. Besides, it has "Attribute" of thermal power. For instance, "State of heat" of the iron plate rises when the fixed time passes when having strong caloric force in "Attribute". The state transition is defined by using "Attribute" when two or more variables are necessary in this manner. The list of the state of the iron plate and the definition of the state are written as follows.

鉄板は鉄板の温度に依存した状態を持ちます。それ以外に火力という「属性」を持ちます。例えば、「属性」で火力が強い場合、一定時間が経過すると鉄板の「熱の状態」は高くなります。この様に複数の変数が必要な場合、「属性」を用いて状態遷移を定義します。以下に鉄板の状態の一覧と、状態の定義を記します。

Content in state 状態の内容	Name in state 状態の名前	Content in state 状態の内容	Name in state 状態の名前
State of "Extinction" in which iron plate is not ignited 鉄板が点火されていない「消」状態	0	Suitable "It was suitable" state for burning Okonomiyaki. お好み焼きを焼くのに適した「適」状態	3
It is "It is cold" state with a cold iron plate though heats. 加熱しているものの鉄板が冷たい「冷」状態	1	"Overheating" state to reach a high temperature considerably かなり高温に達している「過」状態	4
"It is low" state that is the temperature of the heated extent. 加熱してある程度の温度である「低」状態	2		

4.6.1.13. 鉄板の属性の一覧

Content in attribute 属性の内容	Name in attribute 属性の名前	Content in attribute 属性の内容	Name in attribute 属性の名前
Thermal power is "Low flame" 属性・火力が「弱火」	1	Thermal power is "High heat. " 属性・火力が「強火」	2

4.6.1.14. 鉄板の状態遷移情報の定義

before\after	0	1	2	3	4
0	0	0	ε	ε	ε
1	0	0	0	ε	ε
2	ε	0	0	0	ε
3	ε	ε	0	0	0
4	ε	ε	ε	0	0

4.6.1.15. ソース・海苔・鰹節・油・ボウルの状態の一覧

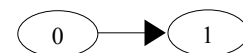
The source, seaweed, Katsuobushi, Abura and bowl has two kinds of of states of "It is possible to use it" and "It is not possible to use it". For instance, it is not possible to use it when already used. The list of the state and the definition of the state are written as follows.

ソース・海苔・鰹節・油・ボウルは「使える」と「使えない」の2種類の状態を持ちます。例えば、既に利用されている場合は、使えません。以下に状態の一覧と、状態の定義を記します。

Content in state 状態の内容	Name in state 状態の名前	Content in state 状態の内容	Name in state 状態の名前
Thing of object is full. 対象の物が満の状態	0	Thing of object is empty. 対象の物が空の状態	1

4.6.1.16. ソース・海苔・鰹節・油・ボウルの状態遷移情報の定義

before\after	0	1
0	0	0
1	ε	0



The state is only two.
And it is one-way.

4.7.1. 状態遷移条件の定義

The state transition condition defines other agents' state and attributes, etc. at the state transition. This definition is written in "*_condition.csv". The state transition condition has various kinds. The list of the state transition condition is written as follows.

状態遷移条件は、状態遷移の時に他のエージェントの状態、属性などを定義します。この定義は「*_condition.csv」に書きます。状態遷移条件は、色々な種類を持ちます。以下に状態遷移条件の一覧を記します。

Name of condition 条件の名前	Content of condition 条件の内容	Description example 記述の例
State	When my state is specified, the state transition is permitted. 自分の状態が指定された状態の場合、状態遷移を許可します	State=2
	When other agents' states are specified, the state transition is permitted. 他のエージェントの状態が指定された状態の場合、状態遷移を許可します	State=2@Avator_000
	When either state of two agents is specified, the transition is permitted. 2つのエージェントのどちらかの状態が指定された状態である場合に遷移を許可します	State=2@Avator_000 Robot_000
	The transition is permitted for less than value that the state specified with State. <u>(*In this case, only "Numerical value" of the state name is effective.)</u> 状態が State で指定した値未満の場合に遷移を許可します ※この場合、状態名は「数値」のみが有効です	State=2>Avator_000
	When states are larger than the values specified with State, the transition is permitted. <u>(*In this case, only "Numerical value" of the state name is effective.)</u> 状態が State で指定した値より大きい場合に遷移を許可します ※この場合、状態名は「数値」のみが有効です	State=0<Avator_000
Time	After it changes in the state, the transition is permitted for less than specified time. <u>(*This time is set to 0 at the state transition.)</u> その状態に遷移してから指定時間未満の場合に遷移を許可します ※この時間は、状態遷移の度に0に設定されます	Time=100>Okonomi_000
	When it is larger than specified time after it changes in the state, the transition is permitted. <u>(*This time is set to 0 at the state transition.)</u> その状態に遷移してから指定時間より大きい場合に遷移を許可します ※この時間は、状態遷移の度に0に設定されます	Time=500<Okonomi_000
Attr	When own attribute value is a specified value, the transition is permitted. 自分自身の属性値が指定値の場合に遷移を許可します	Attr=1
	When the attribute value of a specified agent is a specified value, the transition is permitted. 指定エージェントの属性値が指定値の場合に遷移を許可します	Attr=2@Teppan_000
	The transition is permitted, except when own attribute is a specified value. 自分自身の属性が指定値以外の場合に遷移を許可します	NotAttr=2
Aid	When own supplementary level is a specified value, the transition is permitted. 自分自身の補助度が指定値の場合に遷移を許可します	Aid=2
	The transition is permitted, except when own supplementary level is a specified value. 自分自身の補助度が指定値以外の場合に遷移を許可します	NotAid=2
Uttr	When own remark level is a specified value, the transition is permitted. 自分自身の発言度が指定値の場合に遷移を許可します	Uttr=1

Name of condition 条件の名前	Content of condition 条件の内容	Description example 記述の例
	The transition is permitted, except when own remark level is a specified value. 自分自身の発言度が指定値以外の場合に遷移を許可します	NotUtr=2

4.7.1.1. アバタの状態遷移条件の定義

Avator doesn't have the condition for the state transition. The purpose of this is for Avator to execute the user's operation as it is. The condition of the state transition of Avator is written as follows.

アバタは状態遷移に関して、条件を持ちません。これはアバタは利用者の操作をそのまま実行する為です。以下にアバタの状態遷移の条件を記します。

before\after	0	1	2	3	4	5	6	7	8	9	10	12	13	14	15	16	17	18	19	20	21	22
0	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
1	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
2	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
3	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
4	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
5	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
7	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
8	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
9	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
12	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
13	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
14	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
15	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
16	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
17	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
18	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
19	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
20	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
21	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
22	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε

Definition Ex.

```

"",0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22
0,,,,,,,,,State=1@Avator_000,,,,,,,,,
1,,,,Attr=2@Teppan_000,,,,,,,,,
2,,,,,,,,,Time=10<Okonomi_000,,,,,,,,,
3,,,,,,,,,
4,,,,,,,,,
5,,,,,,,,,
  ⋮
    
```

} 48 alphanumeric characters
 } 48 英数字以内
 } When there is a condition →It changes when satisfy ing it.
 } Empty value →It changes unconditionally.
 } 条件がある場合 →条件を満たせば遷移します
 } 空文字 →無条件に遷移します

When the condition is specified, "State transition condition" is input to "*_condition.csv" as shown in the table above. It changes unconditionally when "Empty character" is input. The alphanumeric character within 48 characters can be specified for the name in the state.

「*_condition.csv」には、条件を指定する場合、上の表の様に「状態遷移条件」を入力します。「空の文字」を入力した場合、無条件で遷移します。状態の名前は48文字以内の英数字が指定できます。

4.7.1.2. ロボットの状態遷移条件の定義(補助度:低)

The state transition of Robot depends on the state of Avator and the iron plate. The state transition condition of Robot is written as follows. "a" in the table is state sets of Avator, and "i" is state sets of iron plates. The state of Okonomiyaki omits the variable identifier. It is an agreement, and "<" is a small becoming "=".

ロボットの状態遷移は、アバタと鉄板の状態に依存します。以下にロボットの状態遷移条件を記します。表中の a はアバタの状態集合、i は鉄板の状態集合です。お好み焼きの状態は変数名を省略しています。「=」は一致、「<」は小なりです。

before\after	0	1	2	3	4	5	6	7	8	9	10	12	13	14	15	16	17	18	19	20	21	22
0	ε	a=1	ε	ε	ε	ε	ε	a=6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
1	ε	ε	ε	ε	ε	ε	ε	a=6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
2	ε	ε	ε	ε	a=3	ε	ε	a=6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
3	ε	ε	ε	ε	ε	ε	ε	a=6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
4	ε	ε	ε	ε	ε	ε	ε	a=6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
5	ε	ε	ε	ε	ε	ε	ε	a=6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε
6	ε	ε	ε	ε	ε	ε	ε	a=6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
7	ε	ε	ε	ε	ε	ε	ε	ε	ε	a=8	ε	a=10	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε
8	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	a=10	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε
9	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	a=10	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε
10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	a=10	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε
12	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	a=13	ε	ε	ε	ε	ε	i=4	i=2	ε
13	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε
14	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	a=15	ε	ε	ε	i=4	i=2	ε
15	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε
16	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	a=17	ε	i=4	i=2	ε
17	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε
18	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	a=19
19	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε
20	ε	ε	ε	ε	ε	a<6	ε	a<8	ε	a<10	ε	a<13	ε	a<15	ε	a<17	ε	a<19	ε	i=4	i=2	ε
21	ε	ε	ε	ε	ε	a<6	ε	a<8	ε	a<10	ε	a<13	ε	a<15	ε	a<17	ε	a<19	ε	i=4	i=2	ε
22	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε

4.7.1.3. ロボットの状態遷移条件の定義(補助度:中)

before\after	0	1	2	3	4	5	6	7	8	9	10	12	13	14	15	16	17	18	19	20	21	22
0	ε	a=1	ε	ε	ε	ε	ε	a=6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
1	ε	ε	ε	ε	ε	ε	ε	a=6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
2	ε	ε	ε	ε	a=3	ε	ε	a=6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
3	ε	ε	ε	ε	ε	ε	ε	a=6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
4	ε	ε	ε	ε	ε	ε	ε	a=6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε

before\after	0	1	2	3	4	5	6	7	8	9	10	12	13	14	15	16	17	18	19	20	21	22	
5	ε	ε	ε	ε	ε	ε	ε	a=6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε	
6	ε	ε	ε	ε	ε	ε	ε	a=6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
7	ε	ε	ε	ε	ε	ε	ε	ε	ε	a=8	ε	a=10	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε	
8	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	a=10	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε	
9	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	a=10	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε	
10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	a=10	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε	
12	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	a=13	ε	ε	ε	ε	ε	i=4	i=2	ε	
13	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε	
14	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	a=15	ε	ε	ε	i=4	i=2	ε	
15	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε	
16	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	a=17	ε	i=4	i=2	ε	
17	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε	
18	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	a=19	
19	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε	
20	ε	ε	ε	ε	ε	a<6	ε	a<8	ε	a<10	ε	a<13	ε	a<15	ε	a<17	ε	a<19	ε	i=4	i=2	ε	
21	ε	ε	ε	ε	ε	a<6	ε	a<8	ε	a<10	ε	a<13	ε	a<15	ε	a<17	ε	a<19	ε	i=4	i=2	ε	
22	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	

4.7.1.4. ロボットの状態遷移条件の定義(補助度:高)

before\after	0	1	2	3	4	5	6	7	8	9	10	12	13	14	15	16	17	18	19	20	21	22
0	ε	a=1	ε	ε	ε	ε	ε	a=6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
1	ε	ε	ε	ε	ε	ε	ε	a=6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
2	ε	ε	ε	ε	a=3	ε	ε	a=6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
3	ε	ε	ε	ε	ε	ε	ε	a=6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
4	ε	ε	ε	ε	ε	ε	ε	a=6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
5	ε	ε	ε	ε	ε	ε	ε	a=6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε
6	ε	ε	ε	ε	ε	ε	ε	a=6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
7	ε	ε	ε	ε	ε	ε	ε	ε	ε	a=8	ε	a=10	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε
8	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	a=10	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε
9	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	a=10	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε
10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	a=10	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε
12	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	a=13	ε	ε	ε	ε	ε	i=4	i=2	ε
13	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε
14	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	a=15	ε	ε	ε	i=4	i=2	ε
15	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε
16	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	a=17	ε	i=4	i=2	ε
17	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε
18	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	a=19
19	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε
20	ε	ε	ε	ε	ε	a<6	ε	a<8	ε	a<10	ε	a<13	ε	a<15	ε	a<17	ε	a<19	ε	i=4	i=2	ε
21	ε	ε	ε	ε	ε	a<6	ε	a<8	ε	a<10	ε	a<13	ε	a<15	ε	a<17	ε	a<19	ε	i=4	i=2	ε

before\after	0	1	2	3	4	5	6	7	8	9	10	12	13	14	15	16	17	18	19	20	21	22	
22	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε

4.7.1.5. ロボットの状態遷移条件の定義(補助度:全)

before\after	0	1	2	3	4	5	6	7	8	9	10	12	13	14	15	16	17	18	19	20	21	22	
0	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
1	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
2	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
3	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
4	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
5	ε	ε	ε	ε	ε	ε	i=3	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε	
6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε	
7	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε	
8	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε	
9	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	01A	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε	
10	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε	
12	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε	
13	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε	
14	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε	
15	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε	
16	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	21B	ε	ε	i=4	i<3	ε	
17	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	ε	
18	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	abX6	
19	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i=4	i=2	abX6	
20	ε	ε	ε	ε	ε	00X2	ε	ε	ε	00A	01A	ε	ε	ε	ε	01B	11B	ε	ε	i=4	i=2	abX6	
21	ε	ε	ε	ε	ε	00X2	ε	ε	ε	00A	01A	ε	ε	ε	ε	01B	11B	ε	ε	i=4	i=2	abX6	
22	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε

4.7.1.6. お好み焼きの状態遷移条件の定義

The state transition of Okonomiyaki is chiefly a state transition by the passage of time. Therefore, the state transition condition concerning time is defined. In "i" in the table, the state set and "t" of the iron plate are passage second just behind own state transition numbers. "u" is Avator or Robot is specified either and agree. "=" is an agreement. ">" is a large becoming. "&" is AND condition.

お好み焼きの状態遷移は、主に時間経過による状態遷移です。その為、時間に関する状態遷移条件を定義します。尚、表中の「i」は鉄板の状態集合、「t」は自身の状態遷移の直後からの経過秒数です。「u」は「アバタまたはロボットのいずれかの指定状態に一致する」です。「=」は一致です。「>」は大なりです。「&」はAND条件です。

before\after	00X1	00X2	00A	01A	02A	01B	02B	11B	12B	21B	22B	abX3	abX4	abX5	abX6
00X1	ε	u=1	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
00X2	ε	ε	u=6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
00A	ε	ε	ε	i>2 & t>40	i>2 & t>60	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
01A	ε	ε	ε	ε	i>2 & t>40	u=10	ε	ε	ε	ε	ε	ε	ε	ε	ε

before\after	00X1	00X2	00A	01A	02A	01B	02B	11B	12B	21B	22B	abX3	abX4	abX5	abX6
02A	ε	ε	ε	ε	ε	ε	u=10	ε	ε	ε	ε	ε	ε	ε	ε
01B	ε	ε	ε	ε	ε	ε	ε	i>2 & t>40	ε	ε	ε	ε	ε	ε	ε
02B	ε	ε	ε	ε	ε	ε	ε	ε	i>2 & t>40	ε	i>2 & t>60	ε	ε	ε	ε
11B	ε	ε	ε	ε	ε	ε	ε	ε	ε	i>2 & t>60	ε	u=17	ε	ε	ε
12B	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i>2 & t>60	u=17	ε	ε	ε
21B	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	i>2 & t>60	u=17	ε	ε	ε
22B	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	u=17	ε	ε	ε
abX3	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	u=18	ε	ε
abX4	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
abX5	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	u=19
abX6	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε

4.7.1.7. 鉄板の状態遷移条件の定義

The state transition of the iron plate is done from "Attribute" that shows thermal power being chiefly set by the passage of time and oneself. For instance, it changes in the state of heating 30 seconds after thermal power is set to the high heat. The state transition condition of the iron plate is written as follows. Moreover, "t" in the table is an elapsed time after the state changes of oneself. "r" is Attribute. "u" is Avator or Robot is specified either and agree. "=" is an agreement. ">" is a large becoming. "&" is AND condition.

鉄板の状態遷移は主に時間経過と、自身に設定されている火力を示す「属性」から行います。例えば、火力を強火に設定してから30秒後に加熱状態に遷移するなどです。以下に鉄板の状態遷移条件を記します。又、表中の「t」は自身の状態遷移後の経過時間です。「r」は「属性」です。「u」は「アバタまたはロボットのいずれかの指定の状態に一致する」です。「=」は一致です。「>」は大なりです。「&」はAND条件です。

before\after	0	1	2	3	4
0	ε	u=5	ε	ε	ε
1	r=1	ε	r=2 & t>10	ε	ε
2	ε	r=1 & t>10	ε	r=2 & t>10	ε
3	ε	ε	r=1 & t>10	ε	r=2 & t>10
4	ε	ε	ε	r=1 & t>10	ε

4.7.1.8. ソース・海苔・鰹節・油・ボウルの状態遷移条件の定義

These agents are "Passive agent" all. Therefore, the state doesn't change oneself. Therefore, the condition of the state transition is not necessary. The condition of the state transition is written as follows.

これらのエージェントは全て「受動的なエージェント」です。その為、自分自身で状態遷移しません。よって、状態遷移の条件は必要ありません。以下に状態遷移の条件を記します。

before\after	0	1
0	ϵ	ϵ
1	ϵ	ϵ

4.8.1. 状態遷移処理の定義

The state transition processing describes the processing of "Immediately after changing in the state". It corresponds to the agent's behavior. The list of the processing that can be specified is written as follows. This definition is written in "*_transition.csv".

状態遷移処理は、「その状態に遷移した直後」の処理を記述します。それはエージェントの行動に該当します。以下に指定できる処理の一覧を記します。この定義は「*_transition.csv」に書きます。

Name of processing 処理の名前	Content of processing 処理の内容	Description example 記述の例
State	It changes in the state to specify other agents' states. <u>However, when other agents do not meet the requirement of the transition of the state, the demand is disregarded.</u> 他のエージェントの状態を指定した状態に遷移します。但し、他のエージェントが状態の遷移の条件を満たさない場合、その要求は無視されます。	State=1@Abura_000 (Oil is made used.) (油を使用済みにする)
Attr	It sets it to the attribute in which other agents' attributes are specified. For instance, when thermal power of the iron plate is set, it uses it. 他のエージェントの属性を指定した属性に設定します。例えば、鉄板の火力を設定する場合に使用します。	Attr=2@Teppan_000 (Thermal power of the iron plate is made "It is strong".) (鉄板の火力を「強」にする)
Move	It moves to coordinates that specify other agents. Coordinates are described by the X:Y:Z form. 他のエージェントを指定した座標に移動します。座標は X:Y:Z 形式で記述します。	Move=100:80:100@Sauce_000 (The source is moved on the iron plate.) (ソースを鉄板の上に移動する)
	It is moved while passing coordinates that specify other agents. Coordinates are described by "X:Y:Z X:Y:Z form". Many coordinates can be specified. 他のエージェントを指定した座標を経由しながら移動させます。座標は「X:Y:Z X:Y:Z 形式」で記述します。座標はいくつでも指定できます。	Move=100:70:100 100:80:100@Sauce_000 (The source is moved on the iron plate and lifted.) (ソースを鉄板の上に移動して持ち上げる)
Angle	It rotates to the direction that specifies other agents. It describes it by "Direction (0,1) of X: direction (0,1) of Y: direction (0,1) of Z: the beginning angle: the end angle: increment angle" form. This is the same as the form operated with Operation. 他のエージェントを指定した向きに回転します。「X 方向 (0,1):Y 方向(0,1):Z 方向(0,1):開始角度:終了角度:増分角度」形式で記述します。これは Operation で操作した形式と同じです。	Angle=1:0:0:180:10@Okonomi_000 (Okonomiyaki is turned inside out.) (お好み焼きを裏返す)
Visual	Other agents' externals are changed. For instance, it puts some sauce on in Okonomiyaki. 他のエージェントの見た目を変えます。例えばお好み焼きにソースをかけた場合などです。	Visual=5@Okonomi_000 (Externals of Okonomiyaki are changed into "Externals on which the source is put".) (お好み焼きの見た目を、「ソースがかけられた見た目」に変える。)
Motion	Specification is operated. Please see "Definition of bodily movement(gesture)" about details. This is specification of the bodily movement. (指定の動作を行います。詳細は、「Definition of bodily movement(gesture)」をみてください。これは身体動作の指定	Motion=ges92r (It reaches the bowl.) (ボウルに手を伸ばす)

Name of processing 処理の名前	Content of processing 処理の内容	Description example 記述の例
	です.)	
Utterance	It makes remarks on a specified objection. <u>However please specify space for " _".</u> 指定の文言を発言します。但し、スペースは「_」に指定してください。	Utterance=Hello ("Hello" Make remarks.) (「Hello」と発言する)
	It makes remarks on the specified objection repeating after the passage of specified unit time. And, all Transition after this specification is disregarded. 指定の文言を発言します。指定した文言は、指定ユニット時間経過の後、繰り返して発言します。そして、この指定の後の Transition は全て無視されます。	Utterance=Hello&20 (Make remarks being repeat in every "Hello" and 20 unit times.) (「Hello」と20 ユニット時間間隔で繰り返して発言します)
	It makes remarks on a specified objection only first time to the state transition. 状態遷移した初回のみ、指定の文言を発言します。	Utterance=Hello&first (It is made remarks "Hello" only once in the state.) (その状態で一度のみ「Hello」と発言します。)
	It waits during the time of a specified unit. It makes remarks on a specified objection when the time of a specified unit passes, and Transition of continuation is executed. 指定ユニット時間の間、待ちます。指定ユニット時間が過ぎた場合、指定の文言を発言して、続きの Transition を実行します。	UttrAndAction=GoodBye&10 (When 10 unit time passes, it is made remarks, "Goodbye".) (10 ユニット時間が経過した場合、「さようなら」と発言します。)
	When it meets a specified requirement, it makes remarks. Transition afterwards is not done. 指定の条件を満たす場合、発言します。その後の Transition は行いません。(アバタが1(油をとった)の場合、お礼を言います。自分はその後の Transition (油をとる)を行いません。)	UttrAndReject=Thanks&1@Avator (When Avator's state is 1 "Oil was taken", the reward is said. I do not do Transition afterwards "Oil is taken".)
Reject	Processing is not executed for the state transition that has already been done. 既に行われた状態遷移の場合、処理を実行しません。	Reject=AlwasState

4.8.1.1. アバタの状態遷移処理の定義

The processing of the state transition of Avator is a concrete chiefly procedure to cook the Okonomiyaki cooperation. It consists of "Bodily movement" and "Other agents' state transitions". These actions are the same as the instruction input with Operation. The list of the processing of the state transition of Avator is written as follows.

アバタの状態遷移の処理は、主にお好み焼き協調理の具体的な手順です。それは、「身体動作」、「他のエージェントの状態遷移」から成り立ちます。これらの行動は Operation で入力した命令と同じです。以下にアバタの状態遷移の処理の一覧を記します。

Name in state 状態の名前	Description of state transition processing 状態遷移処理の記述	Content of state transition processing 状態遷移処理の内容
1 Mix dough	Motion=ges92a, Motion=ges00a, State=00X1@Okonomi_000	Operation that mixes Okonomiyaki is done. The state of Okonomiyaki is made "Mixing state". お好み焼きを混ぜる動作をします お好み焼きの状態を「混ぜた状態」にします
2 Take oil	Motion=ges82a, Motion=ges00a, State=1@Abura_000	Operation that moves oil on the iron plate is done. Oil is made "Used state". 油を「使用済の状態」にします 油を鉄板の上に移動する動作をします
3 Put oil on Teppan	Motion=ges84a, Motion=ges85a, Motion=ges86a, Motion=ges00a	Oil is moved right and left on the iron plate, and the returned operation is done.

Name in state 状態の名前	Description of state transition processing 状態遷移処理の記述	Content of state transition processing 状態遷移処理の内容
		油を鉄板の上で左右に動かして、元に戻す動作をします
4	Check gas level	Motion=ges42a,Motion=ges00a Operation that sees the gas remainder amount meter of the iron plate is done. 鉄板のガス残量計を見る動作をします
5	Fire up Teppan	Motion=ges01a,Motion=ges00a, Attr=2@Teppan_000 State=1@Teppan_000 Operation that feels after the switch of the iron plate is done. The iron plate is made "Cold state (ignited state)". Thermal power of the iron plate is made "It is strong". 鉄板のスイッチを触る動作をします 鉄板を「冷たい状態(点火した状態)」にします 鉄板の火力を「強い」にします
6	Put ingredients on the Teppan	Motion=ges92a,Motion=ges93a, Motion=ges94a,Motion=ges95a, Motion=ges96a,Motion=ges00a, State=00A@Okonomi_000 The bowl is moved on the iron plate, and the inclined operation is done. Externals of Okonomiyaki are made "Externals of the ingredients". It returns based on the inclination of the bowl, and the operation that moves to former place is done. The state of Okonomiyaki is made "State to burn A respect". ボウルを鉄板の上に移動し、傾ける動作をします お好み焼きの見た目を「生地の見たい」にします ボウルの傾きを元に戻し、元の場所へ移動する動作をします お好み焼きの状態を「A面を焼いている状態」にします
7	Take pork	Motion=ges72a,Motion=ges10a Operation that moves pork on Okonomiyaki is done. 豚肉をお好み焼きの上に移動する動作をします
8	Put pork	Motion=ges05a,Motion=ges00a, Visual=3@Okonomi_000 Operation that puts pork on Okonomiyaki is done. Externals of Okonomiyaki are made "State that pork is put". 豚肉をお好み焼きの上のせる動作をします。 お好み焼きの見た目を「豚肉がのせられた状態」にします
9	Check condition of roasted Okonomiyaki	Motion=ges43a,Motion=ges00a Operation that sees Okonomiyaki is done. お好み焼きを見る動作をします
10	Flip Okonomiyaki	Motion=ges12a,Motion=ges00a, State=abB@Okonomi_000 , Angle=1:0:0:180:20@Okonomi_000 Operation that turns Okonomiyaki inside out is done. The state of Okonomiyaki is made "State to burn B respect". Okonomiyaki is rotated to X axis 180 times (Turn it inside out). お好み焼きを裏返す動作をします お好み焼きの状態を「B面を焼いている状態」にします お好み焼きをX軸に180度回転(裏返す)します
12	Take sauce	Motion=ges72a,Motion=ges73a, State=1@Sauce_000 Operation that moves the sauce on the iron plate is done. The sauce is made "Used state". ソースを鉄板の上に移動する動作をします ソースを「使用済の状態」にします。
13	Put sauce on Okonomiyaki	Motion=ges74a,Motion=ges75a, Motion=ges76a,Motion=ges00a The sauce is inclined, and the operation painted on Okonomiyaki is done. Externals of Okonomiyaki are made "State that the sauce is painted". It returns based on the inclination of the sauce, and the operation returned to former position is done. ソースを傾けて、お好み焼きに塗る動作をします お好み焼きの見た目を「ソースが塗られた状態」にします ソースの傾きを元に戻して、元の位置に戻す動作をします
14	Take seaweed(Nori)	Motion=ges62a,Motion=ges63a, State=1@Nori_000 Operation that moves the seaweed on the iron plate is done. The seaweed is made "Used state". 海苔を鉄板の上に移動する動作をします 海苔を「使用済の状態」にします

Name in state 状態の名前	Description of state transition processing 状態遷移処理の記述	Content of state transition processing 状態遷移処理の内容
15 Put seaweed on Okonomiyaki	Motion=ges64a,Motion=ges65a, Motion=ges66a,Motion=ges00a	The seaweed is inclined, and the operation painted on Okonomiyaki is done. Externals of Okonomiyaki are made "State that the seaweed is painted". It returns based on the inclination of the seaweed, and the operation returned to former position is done. 海苔を傾けて、お好み焼きに塗る動作をします お好み焼きの見た目を「海苔が塗られた状態」にします 海苔の傾きを元に戻して、元の位置に戻す動作をします
16 Take Katsuobushi	Motion=ges52a,Motion=ges53a, State=1@Katsuobushi_000	Operation that moves the Katsuobushi on the iron plate is done. The Katsuobushi is made "Used state". 鰹節を鉄板の上に移動する動作をします 鰹節を「使用済の状態」にします
17 Put Katsuobushi on Okonomiyaki		The Katsuobushi is inclined, and the operation painted on Okonomiyaki is done. Externals of Okonomiyaki are made "State that the Katsuobushi is painted". It returns based on the inclination of the Katsuobushi, and the operation returned to former position is done. 鰹節を傾けて、お好み焼きに塗る動作をします お好み焼きの見た目を「鰹節が塗られた状態」にします 鰹節の傾きを元に戻して、元の位置に戻す動作をします
18 Cut Okonomiyaki	—	—
19 Put Okonomiyaki on dish	—	—
20 Reduce the heat of Teppan to low	Motion=ges01a,Motion=ges00a, Attr=1@Teppan_000	Operation that feels after the switch of the iron plate is done. Thermal power of the iron plate is weakened. 鉄板のスイッチを触る動作をします 鉄板の火力を弱くします
21 Increase the heat of Teppan to high	Motion=ges01a,Motion=ges00a, Attr=2@Teppan_000	Operation that feels after the switch of the iron plate is done. Thermal power of the iron plate is made strong. 鉄板のスイッチを触る動作をします 鉄板の火力を強くします
22 Put out the fire of Teppan	Motion=ges01a,Motion=ges00a, Attr=2@Teppan_000	Operation that feels after the switch of the iron plate is done. Thermal power of the iron plate is lost (Fire is extinguished). 鉄板のスイッチを触る動作をします 鉄板の火力を失くします(消火します)

*Motion=ges00a is returned to an initial bodily movement and the state to sit on a chair.

*The thing can be moved, and be rotated by the bodily movement specified with Motion.

※ 手順 Motion=ges00a は初期の動作、つまり椅子に座った状態に戻します

※Motion で指定した身体動作で物を移動したり、回転したり出来ます。

Definition Ex. 0,"Motion=ges00a,State=00X1@Okonomi_000"
 1,"Motion=ges92a,Motion=ges00a,State=00X2@Okonomi_000"
 2,"Motion=ges82a,Motion=ges83a,State=1@Abura_000"
 3,"Motion=ges84a,Motion=ges85a,Motion=ges86a,Motion=ges00a,State=3@Avator_000" } 1 state difinition

⋮

48 alphanumeric characters It inputs it by Comma Separated Value. Everything from " to " becomes it in a row.
 48 英数字以内 CSV 形式で入力します。「"」から「"」までが1列になります。

4.8.1.2. ロボットの状態遷移処理の定義(補助度:低)

The state transition changes into Robot depending on a supplementary level. The condition of the state transition changes along with it, too. This is the same as information on the state transition. The condition of the state transition of each supplementary level of Robot is written as follows.

Processing (low) of the state transition of Robot only advises on the dish of Okonomiyaki. Processing (low) of the state transition of Robot is written as follows.

ロボットは補助度により状態遷移が変わります。それに伴い状態遷移の条件も変わります。これは状態遷移の情報と同じです。ロボットの状態遷移の処理(low)は、お好み焼きの料理のアドバイスのみ行います。以下にロボットの状態遷移の処理(low)を記します。

Name in state 状態の名前	Description of state transition processing 状態遷移処理の記述	Content of state transition processing 状態遷移処理の内容
1 Mix dough	Utterance=Please_mix_dough&20	It makes remarks at intervals of "Please mix dough" and 20 unit time. "Please mix dough"と20 ユニット時間の間隔で発言します
2 Take oil	Utterance=Please_take_oil&20	It makes remarks at intervals of "Please take oil" and 20 unit time. "Please take oil"と20 ユニット時間の間隔で発言します
3 Put oil on Teppan	Utterance=Please_put_oil_on_Teppan&20	It makes remarks at intervals of "Please put oil on Teppan" and 20 unit time. "Please put oil on Teppan"と20 ユニット時間の間隔で発言します
4 Check gas level	Utterance=Please_check_gas_level&20	It makes remarks at intervals of "Please check gas level" and 20 unit time. "Please check gas level"と20 ユニット時間の間隔で発言します
Fire up Teppan	Utterance=Please_fire_up_Teppan&20	It makes remarks at intervals of "Please fire up Teppan" and 20 unit time. "Please fire up Teppan"と20 ユニット時間の間隔で発言します
6 Put ingredients on the Teppan	Utterance=Please_put_ingredients_on_the_Teppan&20	It makes remarks at intervals of "Please put ingredients on the Teppan" and 20 unit time. "Please put ingredients on the Teppan"と20 ユニット時間の間隔で発言します
7 Take pork	Utterance=Please_take_pork&20	It makes remarks at intervals of "Please take pork" and 20 unit time. "Please take pork"と20 ユニット時間の間隔で発言します
8 Put pork	Utterance=Please_put_pork&20	It makes remarks at intervals of "Please put pork" and 20 unit time. "Please put pork"と20 ユニット時間の間隔で発言します
9 Check condition of roasted Okonomiyaki	Utterance=Please_check_condition_of_roasted_Okonomiyaki&20	It makes remarks at intervals of "Please check condition of roasted Okonomiyaki" and 20 unit time. "Please check condition of roasted Okonomiyaki"と20 ユニット時間の間隔で発言します
10 Flip Okonomiyaki	Utterance=Please_flip_Okonomiyaki&20	It makes remarks at intervals of "Please flip Okonomiyaki" and 20 unit time. "Please flip Okonomiyaki"と20 ユニット時間の間隔で発言します

Name in state 状態の名前		Description of state transition processing 状態遷移処理の記述	Content of state transition processing 状態遷移処理の内容
12	Take sauce	Utterance=Please_take_sauce&20	It makes remarks at intervals of "Please take sauce" and 20 unit time. "Please take sauce"と20 ユニット時間の間隔で発言します
13	Put sauce on Okonomiyaki	Utterance=Please_put_sauce_on_Okonomiyaki&20	It makes remarks at intervals of "Please put sauce on Okonomiyaki" and 20 unit time. "Please put sauce on Okonomiyaki"と20 ユニット時間の間隔で発言します
14	Take seaweed(Nori)	Utterance=Please_take_seaweed(Nori)&20	It makes remarks at intervals of "Please take seaweed(Nori)" and 20 unit time. "Please take seaweed(Nori)"と20 ユニット時間の間隔で発言します
15	Put seaweed on Okonomiyaki	Utterance=Please_put_seaweed_on_Okonomiyaki&20	It makes remarks at intervals of "Please put seaweed on Okonomiyaki" and 20 unit time. "Please mix dough"と20 ユニット時間の間隔で発言します
16	Take Katsuobushi	Utterance=Please_take_Katsuobushi&20	It makes remarks at intervals of "Please take Katsuobushi" and 20 unit time. "Please take Katsuobushi"と20 ユニット時間の間隔で発言します
17	Put Katsuobushi on Okonomiyaki	Utterance=Please_put_Katsuobushi_on_Okonomiyaki&20	It makes remarks at intervals of "Please put Katsuobushi on Okonomiyaki" and 20 unit time. "Please put Katsuobushi on Okonomiyaki"と20 ユニット時間の間隔で発言します
18	Cut Okonomiyaki	Utterance=Please_cut_Okonomiyaki&20	It makes remarks at intervals of "Please cut Okonomiyaki" and 20 unit time. "Please cut Okonomiyaki"と20 ユニット時間の間隔で発言します
19	Put Okonomiyaki on dish	Utterance=Please_put_Okonomiyaki_on_dish&20	It makes remarks at intervals of "Please put Okonomiyaki on dish" and 20 unit time. "Please put Okonomiyaki on dish"と20 ユニット時間の間隔で発言します
20	Reduce the heat of Teppan to low	Utterance=Please_reduce_the_heat_of_Teppan_to_low&20	It makes remarks at intervals of "Please reduce the heat of Teppan" and 20 unit time. "Please reduce the heat of Teppan"と20 ユニット時間の間隔で発言します
21	Increase the heat of Teppan to high	Utterance=Please_increase_the_heat_of_Teppan_to_high&20	It makes remarks at intervals of "Please increase the heat of Teppan" and 20 unit time. "Please increase the heat of Teppan"と20 ユニット時間の間隔で発言します
22	Put out the fire of Teppan	Utterance=Please_put_out_the_fire_of_Teppan&20	It makes remarks at intervals of "Please put out the fire of Teppan" and 20 unit time. "Please put out the fire of Teppan"と20 ユニット時間の間隔で発言します

4.8.1.3. ロボットの状態遷移処理の定義(補助度:中)

Processing (middle) of the state transition of Robot does "Advice" of the dish of Okonomiyaki and "Assistance".

ロボットの状態遷移の処理(middle)は、お好み焼きの料理の「アドバイス」と「補助」を行います。

Name in state 状態の名前	Description of state transition processing 状態遷移処理の記述	Content of state transition processing 状態遷移処理の内容
1 Mix dough	Utterance=Please_mix_dough&20	It makes remarks at intervals of "Please mix dough" and 20 unit time. "Please mix dough"と20 ユニット時間の間隔で発言します
2 Take oil	Utterance=I_take_oil, Motion=ges82r,Motion=ges83r, State=1@Abura_000	It is made remarks "I take oil". Operation that moves oil on the iron plate is done. Oil is made "Used state". "I take oil"と発言します。 油を「使用済の状態」にします 油を鉄板の上に移動する動作をします
3 Put oil on Teppan	Utterance=Please_put_oil_on_Teppan&20	It makes remarks at intervals of "Please put oil on Teppan" and 20 unit time. "Please put oil on Teppan"と20 ユニット時間の間隔で発言します
4 Check gas level	Utterance=I_check_gas_level, Motion=ges42r,Motion=ges00r	It is made remarks "I check gas level". Operation that sees the gas remainder amount meter of the iron plate is done. "I check gas level"と発言します。 鉄板のガス残量計を見る動作をします
5 Fire up Teppan	Utterance=I_fire_up_Teppan, Motion=ges01r,Motion=ges00r, Attr=2@Teppan_000&first , State=1@Teppan_000	It is made remarks "I take oil". Operation that feels after the switch of the iron plate is done. The iron plate is made "Cold state (ignited state)". Thermal power of the iron plate is made "It is strong". "I fire up Teppan"と発言します。 鉄板のスイッチを触る動作をします 鉄板を「冷たい状態(点火した状態)」にします 鉄板の火力を「強い」にします
6 Put ingredients on the Teppan	Utterance=Please_put_ingredients_on_the_Teppan&20	It makes remarks at intervals of "Please put ingredients on the Teppan" and 20 unit time. "Please put ingredients on the Teppan"と20 ユニット時間の間隔で発言します
7 Take pork	Utterance=I_take_pork, Motion=ges05r,Motion=ges08r	It is made remarks "I take pork". Operation that moves pork on Okonomiyaki is done. "I take pork"と発言します。 豚肉をお好み焼きの上に移動する動作をします
8 Put pork	Utterance=Please_put_pork&20	It makes remarks at intervals of "Please put pork" and 20 unit time. "Please put pork"と20 ユニット時間の間隔で発言します
9 Check condition of roasted Okonomiyaki	Utterance=I_check_condition_of_roased_Okonomiyaki, Motion=ges43r,Motion=ges00r	It is made remarks "I check condition of roasted Okonomiyaki". Operation that sees Okonomiyaki is done. "I check condition of roasted Okonomiyaki"と発言

Name in state 状態の名前	Description of state transition processing 状態遷移処理の記述	Content of state transition processing 状態遷移処理の内容
		します。 お好み焼きを見る動作をします
10	Flip Okonomiyaki	Utterance=Please_flip_Okonomiyaki&20 It makes remarks at intervals of "Please flip Okonomiyaki" and 20 unit time. "Please flip Okonomiyaki"と20 ユニット時間の間隔で発言します
12	Take sauce	Utterance=I_take_sauce, Motion=ges72r,Motion=ges73r, State=1@Sauce_000 It is made remarks "I take sauce". Operation that moves the sauce on the iron plate is done. The sauce is made "Used state". "I take sauce"と発言します。 ソースを鉄板の上に移動する動作をします ソースを「使用済の状態」にします。
13	Put sauce on Okonomiyaki	Utterance=Please_put_sauce_on_Okonomiyaki&20 It makes remarks at intervals of "Please put sauce on Okonomiyaki" and 20 unit time. "Please put sauce on Okonomiyaki"と20 ユニット時間の間隔で発言します
14	Take seaweed(Nori)	Utterance=I_take_seaweed(Nori), Motion=ges62r,Motion=ges63r, State=1@Nori_000 It is made remarks "I take seaweed(Nori)". Operation that moves the seaweed on the iron plate is done. The seaweed is made "Used state". "I take seaweed(Nori)"と発言します。 海苔を鉄板の上に移動する動作をします 海苔を「使用済の状態」にします
15	Put seaweed on Okonomiyaki	Utterance=Please_put_seaweed_on_Okonomiyaki&20 It makes remarks at intervals of "Please put seaweed on Okonomiyaki" and 20 unit time. "Please mix dough"と20 ユニット時間の間隔で発言します
16	Take Katsuobushi	Utterance=I_take_Katsuobushi, Motion=ges52r,Motion=ges53r, State=1@Katsuobushi_000 It is made remarks "I take Katsuobushi". Operation that moves the Katsuobushi on the iron plate is done. The Katsuobushi is made "Used state". "I take Katsuobushi"と発言します。 鰹節を鉄板の上に移動する動作をします 鰹節を「使用済の状態」にします
17	Put Katsuobushi on Okonomiyaki	Utterance=Please_put_Katsuobushi_on_Okonomiyaki&20 It makes remarks at intervals of "Please put Katsuobushi on Okonomiyaki" and 20 unit time. "Please put Katsuobushi on Okonomiyaki"と20 ユニット時間の間隔で発言します
18	Cut Okonomiyaki	Utterance=I_cut_Okonomiyaki, Motion=ges12r,Motion=ges00r, State=abX6@Okonomiyaki_000 —
19	Put Okonomiyaki on dish	Utterance=Please_put_Okonomiyaki_on_dish&20 —
20	Reduce the heat of Teppan to low	Utterance=I_reduce_the_heat_of_Teppan_to_low, Motion=ges01r,Motion=ges00r, Attr=1@Teppan_000 It is made remarks "I reduce the heat of Teppan to low". Operation that feels after the switch of the iron plate is done. Thermal power of the iron plate is weakened. "I reduce the heat of Teppan to low"と発言します。 鉄板のスイッチを触る動作をします 鉄板の火力を弱くします

Name in state 状態の名前	Description of state transition processing 状態遷移処理の記述	Content of state transition processing 状態遷移処理の内容
21 Increase the heat of Teppan to high	Utterance=I_increase_the_heat_of_Teppan_to_high,Motion=ges01r,Motion=ges00r,Attr=2@Teppan_000	It is made remarks "I increase the heat of Teppan to high". Operation that feels after the switch of the iron plate is done. Thermal power of the iron plate is made strong. "I increase the heat of Teppan to high"と発言します。 鉄板のスイッチを触る動作をします 鉄板の火力を強くします
22 Put out the fire of Teppan	Utterance=I_put_out_the_fire_of_Teppan,Motion=ges01r,Motion=ges00r,State=0@Teppan_000	It is made remarks "I put out the fire of Teppan". Operation that feels after the switch of the iron plate is done. Thermal power of the iron plate is lost (Fire is extinguished). "I put out the fire of Teppan"と発言します。 鉄板のスイッチを触る動作をします 鉄板の火力を失くします(消火します)

4.8.1.4. ロボットの状態遷移処理の定義(補助度:高)

Processing (high) of the state transition of Robot does "Advice" of the dish of Okonomiyaki and "Assistance". In addition, when Avator doesn't react even if advising, it cooks.

ロボットの状態遷移の処理 (high) は、お好み焼きの料理の「アドバイス」と「補助」を行います。更にアドバイスをしてもアバタの反応がない場合、料理します。

Name in state 状態の名前	Description of state transition processing 状態遷移処理の記述	Content of state transition processing 状態遷移処理の内容
1 Mix dough	Utterance=Please_mix_dough&first, UttrAndAction=I_mix_dough&10, Motion=ges92r,Motion=ges00r, State=00X2@Okonomi_000	It is made remarks "Please mix dough". The following processing is executed when there is no reaction even if ten unit time passes. It is made remarks "I mix dough". Operation that mixes Okonomiyaki is done. The state of Okonomiyaki is made "Mixing state". 「Please mix dough」と発言します 10 ユニット時間の間反応がない場合、以下の処理を実行します 「I mix dough」と発言します お好み焼きを混ぜる動作をします お好み焼きの状態を「混ぜた状態」にします
2 Take oil	Utterance=I_take_oil, Motion=ges82r,Motion=ges83r, State=1@Abura_000	It is made remarks "I take oil". Operation that moves oil on the iron plate is done. Oil is made "Used state". 「I take oil」と発言します 油を「使用済の状態」にします 油を鉄板の上に移動する動作をします
3 Put oil on Teppan	Utterance=Please_put_oil_on_Teppan &first, UttrAndAction=I_put_oil_on_Teppan &10, Motion=ges84r,Motion=ges85r,Motion=ges86r,Motion=ges00r	It is made remarks "Please put oil on Teppan". The following processing is executed when there is no reaction even if ten unit time passes. It is made remarks "I put oil on Teppan". Oil is moved right and left on the iron plate, and the returned operation is done. 「Please put oil on Teppan」と発言します

Name in state 状態の名前	Description of state transition processing 状態遷移処理の記述	Content of state transition processing 状態遷移処理の内容
		10 ユニット時間の間反応がない場合、以下の処理を実行します 「I out oil on Teppan」と発言します。 油を鉄板の上で左右に動かして、元に戻す動作をします
4	Check gas level Utterance=I_check_gas_level, Motion=ges42r,Motion=ges00r	It is made remarks "I check gas level". Operation that sees the gas remainder amount meter of the iron plate is done. 「I check gas level」と発言します 鉄板のガス残量計を見る動作をします
5	Fire up Teppan Utterance=I_fire_up_Teppan&first,Motion=ges01r,Motion=ges00r, Attr=2@Teppan_000&first, State=1@Teppan_000	It is made remarks "I fire up Teppan". Operation that feels after the switch of the iron plate is done. The iron plate is made "Cold state (ignited state)". Thermal power of the iron plate is made "It is strong". 「I check gas level」と発言します 鉄板のスイッチを触る動作をします 鉄板を「冷たい状態(点火した状態)」にします 鉄板の火力を「強い」にします
6	Put ingredients on the Teppan Reject=AlwaysState, Utterance=Please_put_ingredients_on_the_Teppan&first, UtrAndAction=I_put_ingredients_on_the_Teppan&10, Motion=ges92r,Motion=ges93r,Motion=ges94r,Motion=ges95r,Motion=ges96r, Motion=ges00r, State=00A@Okonomi_000	It is made remarks "Please put ingredients on the Teppan". The following processing is executed when there is no reaction even if ten unit time passes. It is made remarks "I put ingredients on the Teppan". The bowl is moved on the Teppan, the inclined operation is done. Externals of Okonomiyaki are made "Externals of the ingredients". It returns based on the inclination of the bowl, and the operation that moves to former place is done. The state of Okonomiyaki is made "State to burn A respect". 「Please put ingredients on the Teppan」と発言します 10 ユニット時間の間反応がない場合、以下の処理を実行します 「I put ingredients on the Teppan」と発言します ボウルを鉄板の上に移動し、傾ける動作をします お好み焼きの見た目を「生地の見え目」にします ボウルの傾きを元に戻し、元の場所に移動する動作をします お好み焼きの状態を「A面を焼いている状態」にします
7	Take pork Utterance=I_take_pork, Motion=ges05r,Motion=ges08r	It is made remarks "I take pork". Operation that moves pork on Okonomiyaki is done. 「I take pork」と発言します 豚肉をお好み焼きの上に移動する動作をします
8	Put pork Reject=AlwaysState, Utterance=Please_put_pork&first, UtrAndAction=I_put_pork&10, Motion=ges10r,Motion=ges08r,Motion=ges00r, Visual=3@Okonomi_000	It is made remarks "Please put pork". The following processing is executed when there is no reaction even if ten unit time passes. It is made remarks "I put pork". Operation that puts pork on Okonomiyaki is done. Externals of Okonomiyaki are made "State that pork is put". 「Please put pork」と発言します 10 ユニット時間の間反応がない場合、以下の処理を実行します 「I put prok」と発言します 豚肉をお好み焼きの上のせる動作をします。

Name in state 状態の名前	Description of state transition processing 状態遷移処理の記述	Content of state transition processing 状態遷移処理の内容
		お好み焼きの見た目を「豚肉がのせられた状態」にします
9	Check condition of roased Okonomiyaki Utterance=I_check_condition_of_roased Okonomiyaki, Motion=ges43r,Motion=ges00r	It is made remarks "I check condition of roased Okonomiyaki". Operation that sees Okonomiyaki is done. 「I check condition of roased Okonomiyaki」と発言します お好み焼きを見る動作をします
10	Flip Okonomiyaki Reject=AlwaysState, Utterance=Please_flip Okonomiyaki& first, UttrAndAction=I_flip Okonomiyaki& 10, Motion=ges12r,Motion=ges00r, State=abB@Okonomi_000, Angle=1:0:0:180:10@Okonomi_000	It is made remarks "Please flip Okonomiyaki". The following processing is executed when there is no reaction even if ten unit time passes. It is made remarks "I flip Okonomiyaki". Operation that turns Okonomiyaki inside out is done. The state of Okonomiyaki is made "State to burn B respect". Okonomiyaki is rotated to X axis 180 times (Turn it inside out). 「Please flip Okonomiyaki」と発言します 10 ユニット時間の間反応がない場合、以下の処理を実行します 「I flip Okonomiyaki」と発言します お好み焼きを裏返す動作をします お好み焼きの状態を「B面を焼いている状態」にします お好み焼きを X 軸に 180 度回転(裏返す)します
12	Take sauce Utterance=I_take_sauce, Motion=ges72r,Motion=ges73r, State=1@Sauce_000	It is made remarks "I take sauce". Operation that moves the sauce on the iron plate is done. The sauce is made "Used state". 「I take sauce」と発言します ソースを鉄板の上に移動する動作をします ソースを「使用済の状態」にします。
13	Put sauce on Okonomiyaki Reject=AlwaysState, Utterance=Please_put_sauce_on Okonomiyaki&first, UttrAndAction=I_put_sauce_on Okonomiyaki&10, Motion=ges74r,Motion=ges75r,Motion=ges76r,Motion=ges00r, State=13@Robot_000	It is made remarks "Please put sauce on Okonomiyaki". The following processing is executed when there is no reaction even if ten unit time passes. It is made remarks "I put sauce on Okonomiyaki". The sauce is inclined, and the operation painted on Okonomiyaki is done. Externals of Okonomiyaki are made "State that the sauce is painted". It returns based on the inclination of the sauce, and the operation returned to former position is done. 「Please put sauce on Okonomiyaki」と発言します 10 ユニット時間の間反応がない場合、以下の処理を実行します 「I put sauce on Okonomiyaki」と発言します ソースを傾けて、お好み焼きに塗る動作をします お好み焼きの見た目を「ソースが塗られた状態」にします ソースの傾きを元に戻して、元の位置に戻す動作をします
14	Take seaweed(Nori) Utterance=I_take_seaweed(Nori), Motion=ges62r,Motion=ges63r, State=1@Nori_000	It is made remarks "I take seaweed(Nori)". Operation that moves the seaweed on the iron plate is done. The seaweed is made "Used state". 「I take seaweed(Nori)」と発言します 海苔を鉄板の上に移動する動作をします 海苔を「使用済の状態」にします
15	Put seaweed on Okonomiyaki Reject=AlwaysState, Utterance=Please_put_seaweed_on Okonomiyaki	It is made remarks "Please put seaweed on Okonomiyaki". The following processing is executed when there is no reaction

Name in state 状態の名前	Description of state transition processing 状態遷移処理の記述	Content of state transition processing 状態遷移処理の内容
	onomiyaki&first, UtrrAndAction=I_put_seaweed_on_Okonomiyaki&10, Motion=ges64r,Motion=ges65r,Motion=ges66r,Motion=ges00r	even if ten unit time passes. It is made remarks "I put seaweed on Okonomiyaki". The seaweed is inclined, and the operation painted on Okonomiyaki is done. Externals of Okonomiyaki are made "State that the seaweed is painted". It returns based on the inclination of the seaweed, and the operation returned to former position is done. 「Please put seaweed on Okonomiyaki」と発言します 10 ユニット時間の間反応がない場合、以下の処理を実行します 「I put seaweed on Okonomiyaki」と発言します 海苔を傾けて、お好み焼きに塗る動作をします お好み焼きの見た目を「海苔が塗られた状態」にします 海苔の傾きを元に戻して、元の位置に戻す動作をします
16	Take Katsuobushi Utterance=I_take_Katsuobushi, Motion=ges52r,Motion=ges53r, State=1@Katsuobushi_000	It is made remarks "I take Katsuobushi". Operation that moves the Katsuobushi on the iron plate is done. The Katsuobushi is made "Used state". 「I take Katsuobushi」と発言します 鰹節を鉄板の上に移動する動作をします 鰹節を「使用済の状態」にします
17	Put Katsuobushi on Okonomiyaki Reject=AlwaysState, Utterance=Please_put_Katsuobushi_on_Okonomiyaki&first, UtrrAndAction=I_put_Katsuobushi_on_Okonomiyaki&10,Motion=ges54r,Motion=ges55r,Motion=ges56r,Motion=ges00r, State=abX4@Okonomi_000	It is made remarks "Please put Katsuobushi on Okonomiyaki". The following processing is executed when there is no reaction even if ten unit time passes. It is made remarks "I put Katsuobushi on Okonomiyaki". The Katsuobushi is inclined, and the operation painted on Okonomiyaki is done. Externals of Okonomiyaki are made "State that the Katsuobushi is painted". It returns based on the inclination of the Katsuobushi, and the operation returned to former position is done. 「Please put Kasuobushi on Okonomiyaki」と発言します 10 ユニット時間の間反応がない場合、以下の処理を実行します 「I put Katsuobushi on Okonomiyaki」と発言します 鰹節を傾けて、お好み焼きに塗る動作をします お好み焼きの見た目を「鰹節が塗られた状態」にします 鰹節の傾きを元に戻して、元の位置に戻す動作をします
18	Cut Okonomiyaki Utterance=I_cut_Okonomiyaki, Motion=ges12r,Motion=ges00r, State=abX6@Okonomi_000	—
19	Put Okonomiyaki on dish Reject=AlwaysState, Utterance=Please_put_Okonomiyaki_on_dish&first, UtrrAndAction=I_put_Okonomiyaki_on_dish&10, Motion=ges12r,Motion=ges00r, State=abX6@Okonomi_000	—
20	Reduce the heat of Teppan to low Utterance=I_reduce_the_heat_of_Teppan_to_low, Motion=ges01r,Motion=ges00r,	It is made remarks "I reduce the heat of Teppan to low". Operation that feels after the switch of the iron plate is done. Thermal power of the iron plate is weakened.

Name in state 状態の名前	Description of state transition processing 状態遷移処理の記述	Content of state transition processing 状態遷移処理の内容
	Attr=1@Teppan_000	「I reduce the heat of Teppan to low」と発言します 鉄板のスイッチを触る動作をします 鉄板の火力を弱くします
21	Increase the heat of Teppan to high Utterance=I_increase_the_heat_of_Teppan_to_high, Motion=ges01r,Motion=ges00r, Attr=2@Teppan_000	It is made remarks "I increase the heat of Teppan to high". Operation that feels after the switch of the iron plate is done. Thermal power of the iron plate is made strong. 「I increase the heat of Teppan to high」と発言します 鉄板のスイッチを触る動作をします 鉄板の火力を強くします
22	Put out the fire of Teppan Utterance=I_put_out_the_fire_of_Teppan, Motion=ges01r,Motion=ges00r, State=0@Teppan_000	It is made remarks "I put out the fire of Teppan". Operation that feels after the switch of the iron plate is done. Thermal power of the iron plate is lost (Fire is extinguished). 「I put out the fire of Teppan」と発言します 鉄板のスイッチを触る動作をします 鉄板の火力を失くします(消火します)

4.8.1.4. ロボットの状態遷移処理の定義(補助度:全)

Processing (all) of the state transition of the robot cooks all Okonomiyaki. When Avator helps, the reward is said.

ロボットの状態遷移の処理(all)は、全てのお好み焼きの料理を行います。もし、アバタが手伝う場合、お礼を言います。

Name in state 状態の名前	Description of state transition processing 状態遷移処理の記述	Content of state transition processing 状態遷移処理の内容
1	Mix dough <u>UttrAndReject=Thanks_for_you_mix_dough&1@Avator_000,</u> Utterance=I_mix_dough, Motion=ges92r,Motion=ges00r, State=00X2@Okonomi_000	When the state of Avator is "1", it is made remarks "Thanks for you mix dough". The following processing is not done. It is made remarks "I mix dough". Operation that mixes Okonomiyaki is done. The state of Okonomiyaki is made "Mixing state". もし、アバタの状態が「1」の場合、「Thanks for you mix dough」と発言します。以下の処理は行いません。 「I mix dough」と発言します お好み焼きを混ぜる動作をします お好み焼きの状態を「混ぜた状態」にします
2	Take oil <u>UttrAndReject=Thanks_for_you_take_oil&2@Avator_000,</u> Utterance=I_take_oil, Motion=ges82r,Motion=ges83r, State=1@Abura_000	When the state of Avator is "2", it is made remarks "Thanks for you take oil". The following processing is not done. It is made remarks "I take oil". Operation that moves oil on the iron plate is done. Oil is made "Used state". もし、アバタの状態が「2」の場合、「Thanks for you take oil」と発言します。以下の処理は行いません。 「I take oil」と発言します 油を「使用済の状態」にします 油を鉄板の上に移動する動作をします
3	Put oil on Teppan <u>UttrAndReject=Thanks_for_you_put_oil_on_Teppan&3@Avator_000,</u> Utterance=I_put_oil_on_Teppan, Motion=ges84r,Motion=ges85r,Motion	When the state of Avator is "3", it is made remarks "Thanks for you put oil on Teppan". The following processing is not done. It is made remarks "I put oil on Teppan".

Name in state 状態の名前	Description of state transition processing 状態遷移処理の記述	Content of state transition processing 状態遷移処理の内容	
	=ges86r,Motion=ges00r	Oil is moved right and left on the iron plate, and the returned operation is done. もし、アバタの状態が「1」の場合、「Thanks for you put oil on Teppan」と発言します。以下の処理は行いません。 「I put oil on Teppan」と発言します。 油を鉄板の上で左右に動かして、元に戻す動作をします	
4	Check gas level	UtrrAndReject=Thanks_for_you_check_gas_level&4@Avator_000 , Utterance=I_check_gas_level, Motion=ges42r,Motion=ges00r	When the state of Avator is "4", it is made remarks "Thanks for you check gas level". The following processing is not done. It is made remarks "I check gas level". Operation that sees the gas remainder amount meter of the iron plate is done. もし、アバタの状態が「1」の場合、「Thanks for you check gas level」と発言します。以下の処理は行いません。 「I check gas level」と発言します 鉄板のガス残量計を見る動作をします
5	Fire up Teppan	UtrrAndReject=Thanks_for_you_fire_up_Teppan&5@Avator_000 , Utterance=I_fire_up_Teppan, Motion=ges01r,Motion=ges00r, Attr=2@Teppan_000&first , State=1@Teppan_000	When the state of Avator is "5", it is made remarks "Thanks for you fire up Teppan". The following processing is not done. It is made remarks "I fire up Teppan". Operation that feels after the switch of the iron plate is done. The iron plate is made "Cold state (ignited state)". Thermal power of the iron plate is made "It is strong". もし、アバタの状態が「1」の場合、「Thanks for you fire up Teppan」と発言します。以下の処理は行いません。 「I fire up Teppan」と発言します 鉄板のスイッチを触る動作をします 鉄板を「冷たい状態(点火した状態)」にします 鉄板の火力を「強い」にします
6	Put ingredients on the Teppan	UtrrAndReject=Thanks_for_you_put_ingredients_on_the_Teppan&6@Avator_000 , Utterance=I_put_ingredients_on_the_Teppan, Motion=ges92r,Motion=ges93r,Motion=ges94r, Motion=ges95r,Motion=ges96r, Motion=ges00r, State=00A@Okonomi_000	When the state of Avator is "6", it is made remarks "Thanks for you put ingredients on Teppan". The following processing is not done. It is made remarks "I put ingredients on the Teppan". The bowl is moved on the Teppan, the inclined operation is done. Externals of Okonomiyaki are made "Externals of the ingredients". It returns based on the inclination of the bowl, and the operation that moves to former place is done. The state of Okonomiyaki is made "State to burn A respect". もし、アバタの状態が「1」の場合、「Thanks for you put ingredients on the Teppan」と発言します。以下の処理は行いません。 「I put ingredients on the Teppan」と発言します ボウルを鉄板の上に移動し、傾ける動作をします お好み焼きの見た目を「生地の見え目」にします ボウルの傾きを元に戻し、元の場所に移動する動作をします

Name in state 状態の名前	Description of state transition processing 状態遷移処理の記述	Content of state transition processing 状態遷移処理の内容
		お好み焼きの状態を「A面を焼いている状態」にします
7	Take pork UttrAndReject=Thanks_for_you_take_pork&7@Avator_000 , Utterance=I_take_pork, Motion=ges05r,Motion=ges08r	When the state of Avator is "7", it is made remarks "Thanks for you take pork". The following processing is not done. It is made remarks "I take pork". Operation that moves pork on Okonomiyaki is done. もし、アバタの状態が「1」の場合、「Thanks for you take pork」と発言します。以下の処理は行いません。 「I take pork」と発言します 豚肉をお好み焼きの上に移動する動作をします
8	Put pork UttrAndReject=Thanks_for_you_put_pork&8@Avator_000 , Utterance=I_put_pork, Motion=ges10r,Motion=ges08r,Motion=ges00r, Visual=3@Okonomi_000	When the state of Avator is "8", it is made remarks "Thanks for you put pork". The following processing is not done. It is made remarks "I put pork". Operation that puts pork on Okonomiyaki is done. Externals of Okonomiyaki are made "State that pork is put". もし、アバタの状態が「1」の場合、「Thanks for you put pork」と発言します。以下の処理は行いません。 「I put prok」と発言します 豚肉をお好み焼きの上のせる動作をします。 お好み焼きの見た目を「豚肉がのせられた状態」にします
9	Check condition of roased Okonomiyaki UttrAndReject=Thanks_for_you_check_condition_of_roased_Okonomiyaki&9@Avator_000 , Utterance=I_check_condition_of_roased_Okonomiyaki, Motion=ges43r,Motion=ges00r	When the state of Avator is "9", it is made remarks "Thanks for you check condition of roased Okonomiyaki". The following processing is not done. It is made remarks "I check condition of roased Okonomiyaki". Operation that sees Okonomiyaki is done. もし、アバタの状態が「1」の場合、「Thanks for you check condition of roased Okonomiyaki」と発言します。以下の処理は行いません。 「I check condition of roased Okonomiyaki」と発言します お好み焼きを見る動作をします
10	Flip Okonomiyaki UttrAndReject=Thanks_for_you_flip_Okonomiyaki&10@Avator_000 , Utterance=I_flip_Okonomiyaki, Motion=ges12r,Motion=ges00r, State=abB@Okonomi_000 , Angle=1:0:0:180:10@Okonomi_000	When the state of Avator is "10", it is made remarks "Thanks for you flip Okonomiyaki". The following processing is not done. It is made remarks "I flip Okonomiyaki". Operation that turns Okonomiyaki inside out is done. The state of Okonomiyaki is made "State to burn B respect". Okonomiyaki is rotated to X axis 180 times (Turn it inside out). もし、アバタの状態が「1」の場合、「Thanks for you flip Okonomiyaki」と発言します。以下の処理は行いません。 「I flip Okonomiyaki」と発言します お好み焼きを裏返す動作をします お好み焼きの状態を「B面を焼いている状態」にします お好み焼きを X 軸に 180 度回転(裏返す)します
12	Take sauce UttrAndReject=Thanks_for_you_take	When the state of Avator is "12", it is made remarks "Thanks for

Name in state 状態の名前	Description of state transition processing 状態遷移処理の記述	Content of state transition processing 状態遷移処理の内容
	<p>sauce&12@Avator_000, Utterance=I_take_sauce, Motion=ges72r,Motion=ges73r, State=1@Sauce_000</p>	<p>you take sauce". The following processing is not done.</p> <p>It is made remarks "I take sauce". Operation that moves the sauce on the iron plate is done. The sauce is made "Used state".</p> <p>もし、アバタの状態が「1」の場合、「Thanks for you take sauce」と発言します。以下の処理は行いません。</p> <p>「I take sauce」と発言します ソースを鉄板の上に移動する動作をします ソースを「使用済の状態」にします。</p>
13 Put sauce on Okonomiyaki	<p>UtrrAndReject=Thanks_for_you_put_sauce_on_Okonomiyaki&13@Avator_000, Utterance=I_put_sauce_on_Okonomiyaki, Motion=ges74r,Motion=ges75r,Motion=ges76r,Motion=ges00r, State=13@Robot_000</p>	<p>When the state of Avator is "13", it is made remarks "Thanks for you put sauce on Okonomiyaki". The following processing is not done.</p> <p>It is made remarks "I put sauce on Okonomiyaki". The sauce is inclined, and the operation painted on Okonomiyaki is done. Externals of Okonomiyaki are made "State that the sauce is painted". It returns based on the inclination of the sauce, and the operation returned to former position is done.</p> <p>もし、アバタの状態が「1」の場合、「Thanks for you put sauce」と発言します。以下の処理は行いません。</p> <p>「I put sauce on Okonomiyaki」と発言します ソースを傾けて、お好み焼きに塗る動作をします お好み焼きの見た目を「ソースが塗られた状態」にします ソースの傾きを元に戻して、元の位置に戻す動作をします</p>
14 Take seaweed(Nori)	<p>UtrrAndReject=Thanks_for_you_take_seaweed(Nori)&14@Avator_000, Utterance=I_take_seaweed(Nori), Motion=ges62r,Motion=ges63r, State=1@Nori_000</p>	<p>When the state of Avator is "14", it is made remarks "Thanks for you take seaweed(Nori)". The following processing is not done.</p> <p>It is made remarks "I take seaweed(Nori)". Operation that moves the seaweed on the iron plate is done. The seaweed is made "Used state".</p> <p>もし、アバタの状態が「1」の場合、「Thanks for you take seaweed(Nori)」と発言します。以下の処理は行いません。</p> <p>「I take seaweed(Nori)」と発言します 海苔を鉄板の上に移動する動作をします 海苔を「使用済の状態」にします</p>
15 Put seaweed on Okonomiyaki	<p>UtrrAndReject=Thanks_for_you_put_seaweed_on_Okonomiyaki&15@Avator_000, Utterance=I_put_seaweed_on_Okonomiyaki, Motion=ges64r,Motion=ges65r,Motion=ges66r,Motion=ges00r</p>	<p>When the state of Avator is "15", it is made remarks "Thanks for you put seaweed on Okonomiyaki". The following processing is not done.</p> <p>It is made remarks "I put seaweed on Okonomiyaki". The seaweed is inclined, and the operation painted on Okonomiyaki is done. Externals of Okonomiyaki are made "State that the seaweed is painted". It returns based on the inclination of the seaweed, and the operation returned to former position is done.</p>

Name in state 状態の名前	Description of state transition processing 状態遷移処理の記述	Content of state transition processing 状態遷移処理の内容
		もし、アバタの状態が「1」の場合、「Thanks for you put seaweed on Okonomiyaki」と発言します。以下の処理は行いません。 「I put seaweed on Okonomiyaki」と発言します 海苔を傾けて、お好み焼きに塗る動作をします お好み焼きの見た目を「海苔が塗られた状態」にします 海苔の傾きを元に戻して、元の位置に戻す動作をします
16	Take Katsuobushi UttrAndReject=Thanks_for_you_take_Katsuobushi&16@Avator_000 , Utterance=I_take_Katsuobushi, Motion=ges52r,Motion=ges53r, State=1@Katsuobushi_000	When the state of Avator is "16", it is made remarks "Thanks for you take Katsuobushi". The following processing is not done. It is made remarks "I take Katsuobushi". Operation that moves the Katsuobushi on the iron plate is done. The Katsuobushi is made "Used state". もし、アバタの状態が「1」の場合、「Thanks for you take Katsuobushi」と発言します。以下の処理は行いません。 「I take Katsuobushi」と発言します 鰹節を鉄板の上に移動する動作をします 鰹節を「使用済の状態」にします
17	Put Katsuobushi on Okonomiyaki UttrAndReject=Thanks_for_you_put_Katsuobushi_on_Okonomiyaki&17@Avator_000 , Utterance=I_put_Katsuobushi_on_Okonomiyaki, Motion=ges54r,Motion=ges55r,Motion=ges56r, Motion=ges00r, State=abX4@Okonomi_000	When the state of Avator is "17", it is made remarks "Thanks for you put Katsuobushi". The following processing is not done. It is made remarks "I put Katsuobushi on Okonomiyaki". The Katsuobushi is inclined, and the operation painted on Okonomiyaki is done. Externals of Okonomiyaki are made "State that the Katsuobushi is painted". It returns based on the inclination of the Katsuobushi, and the operation returned to former position is done. もし、アバタの状態が「1」の場合、「Thanks for you put Katsuobushi on Okonomiyaki」と発言します。以下の処理は行いません。 「I put Katsuobushi on Okonomiyaki」と発言します 鰹節を傾けて、お好み焼きに塗る動作をします お好み焼きの見た目を「鰹節が塗られた状態」にします 鰹節の傾きを元に戻して、元の位置に戻す動作をします
18	Cut Okonomiyaki UttrAndReject=Thanks_for_you_cut_Okonomiyaki&18@Avator_000 , Utterance=I_cut_Okonomiyaki, Motion=ges12r,Motion=ges00r, State=abX6@Okonomi_000	—
19	Put Okonomiyaki on dish UttrAndReject=Thanks_for_you_put_Okonomiyaki_on_dish&19@Avator_000 , Utterance=I_put_Okonomiyaki_on_dish, Motion=ges12r,Motion=ges00r, State=abX6@Okonomi_000	—
20	Reduce the heat of Teppan to low UttrAndReject=Thanks_for_you_reduce_the_heat_of_Teppan_to_low&20@Avator_000 , Utterance=I_reduce_the_heat_of_Tepp	When the state of Avator is "20", it is made remarks "Thanks for you reduce the heat of Teppan to low". The following processing is not done.

Name in state 状態の名前	Description of state transition processing 状態遷移処理の記述	Content of state transition processing 状態遷移処理の内容
	an_to_low, Motion=ges01r,Motion=ges00r, Attr=1@Teppan_000	It is made remarks "I reduce the heat of Teppan to low". Operation that feels after the switch of the iron plate is done. Thermal power of the iron plate is weakened. もし、アバタの状態が「1」の場合、「Thanks for you reduce the heat of Teppan to low」と発言します。以下の処理は行いません。 「I reduce the heat of Teppan to low」と発言します 鉄板のスイッチを触る動作をします 鉄板の火力を弱くします
21	Increase the heat of Teppan to high UttrAndReject=Thanks_for_you_increase_the_heat_of_Teppan_to_high&21@Avator_000 , Utterance=I_increase_the_heat_of_Teppan_to_high, Motion=ges01r,Motion=ges00r, Attr=2@Teppan_000	When the state of Avator is "21", it is made remarks "Thanks for you increase the heat of Teppan to high". The following processing is not done. It is made remarks "I increase the heat of Teppan to high". Operation that feels after the switch of the iron plate is done. Thermal power of the iron plate is made strong. もし、アバタの状態が「1」の場合、「Thanks for you increase the heat of Teppan to high」と発言します。以下の処理は行いません。 「I increase the heat of Teppan to high」と発言します 鉄板のスイッチを触る動作をします 鉄板の火力を強くします
22	Put out the fire of Teppan UttrAndReject=Thanks_for_you_put_out_the_fire_of_Teppan&22@Avator_000 , Utterance=I_put_out_the_fire_of_Teppan, Motion=ges01r,Motion=ges00r, State=0@Teppan_000	When the state of Avator is "22", it is made remarks "Thanks for you put out the fire of Teppan". The following processing is not done. It is made remarks "I put out the fire of Teppan". Operation that feels after the switch of the iron plate is done. Thermal power of the iron plate is lost (Fire is extinguished). もし、アバタの状態が「1」の場合、「Thanks for you put out the fire of Teppan」と発言します。以下の処理は行いません。 「I put out the fire of Teppan」と発言します 鉄板のスイッチを触る動作をします 鉄板の火力を失くします(消火します)

4.8.1.5. お好み焼きの状態遷移処理の設定

The processing of the state transition of Okonomiyaki is only a change in "Externals" according to the state. The list of the processing of the state transition of Okonomiyaki is written as follows.

お好み焼きの状態遷移処理は、状態に応じた「見た目」の変化のみです。以下にお好み焼きの状態遷移の処理の一覧を記します。

Name in state 状態の名前	Description of state transition processing 状態遷移処理の記述	Content of state transition processing 状態遷移処理の内容
00X1	State=00X1@Okonomi_000,Visual=1	Externals of Okonomiyaki are changed into "Ingredients". お好み焼きの見た目を「生地」に変えます。
00X2	State=00X2@Okonomi_000	—
00A	State=00A@Okonomi_000	—
01A	State=01A@Okonomi_000,Visual=4	Externals of Okonomiyaki are changed into "Burnt state". お好み焼きの見た目を「焼けた状態」に変えます。

Name in state 状態の名前	Description of state transition processing 状態遷移処理の記述	Content of state transition processing 状態遷移処理の内容
02A	State=02A@Okonomi_000,Visual=9	Externals of Okonomiyaki are changed into "Burned state". お好み焼きの見た目を「こげた状態」に変えます。
01B	State=01B@Okonomi_000	—
02B	State=02B@Okonomi_000,Visual=9	Externals of Okonomiyaki are changed into "Burned state". お好み焼きの見た目を「こげた状態」に変えます。
11B	State=11B@Okonomi_000,Visual=4	Externals of Okonomiyaki are changed into "Burnt state". お好み焼きの見た目を「焼けた状態」に変えます。
12B	State=12B@Okonomi_000	—
21B	State=21B@Okonomi_000,Visual=9	Externals of Okonomiyaki are changed into "Burned state". お好み焼きの見た目を「こげた状態」に変えます。
22B	State=22B@Okonomi_000,Visual=9	Externals of Okonomiyaki are changed into "Burned state". お好み焼きの見た目を「こげた状態」に変えます。
abX3	State=abX3@Okonomi_000	—
abX4	State=abX4@Okonomi_000	—
abX5	State=abX5@Okonomi_000,Visual=8	Externals of Okonomiyaki are changed into "State of completion". お好み焼きの見た目を「完了の状態」に変えます。
abX6	State=abX6@Okonomi_000,Visual=8	Externals of Okonomiyaki are changed into "State of completion". お好み焼きの見た目を「完了の状態」に変えます。

Externals of Okonomiyaki (Visual command) can be defined by the bodily movement. Therefore, "Externals on which seaweed is put" is not processed here. Please see "Bodily movement" about a detailed content.

お好み焼きの見た目 (Visual コマンド) は、身体動作で定義できます。その為、「海苔がかけられた見た目」は、ここで処理しません。詳しい内容は「身体動作」をみてください。

4.8.1.6. 鉄板・ソース・海苔・鰹節・油・ポウルの状態遷移処理の設定

These agents do not process the state transition (It is not necessary). However, the error might go out when processing is specified for the dead letter character. Therefore, "State=1" is specified for "State 1". The example of the iron plate is written as follows.

これらのエージェントは、状態遷移の処理はありません (必要ありません)。但し、処理を空文字に指定した場合、エラーが出る場合があります。その為、「状態 1」には「State=1」を指定します。以下に、鉄板の例を記します。

Name in state 状態の名前	Description of state transition processing 状態遷移処理の記述	Content of state transition processing 状態遷移処理の内容
0	State=0@Teppan_000	The state of the iron plate is specified for "Current state". 鉄板の状態を「今の状態」に指定します。
1	State=1@Teppan_000	The state of the iron plate is specified for "Current state". 鉄板の状態を「今の状態」に指定します。
2	State=2@Teppan_000	The state of the iron plate is specified for "Current state". 鉄板の状態を「今の状態」に指定します。
3	State=3@Teppan_000	The state of the iron plate is specified for "Current state". 鉄板の状態を「今の状態」に指定します。
4	State=4@Teppan_000	The state of the iron plate is specified for "Current state". 鉄板の状態を「今の状態」に指定します。

4.9.1. 身体動作の定義

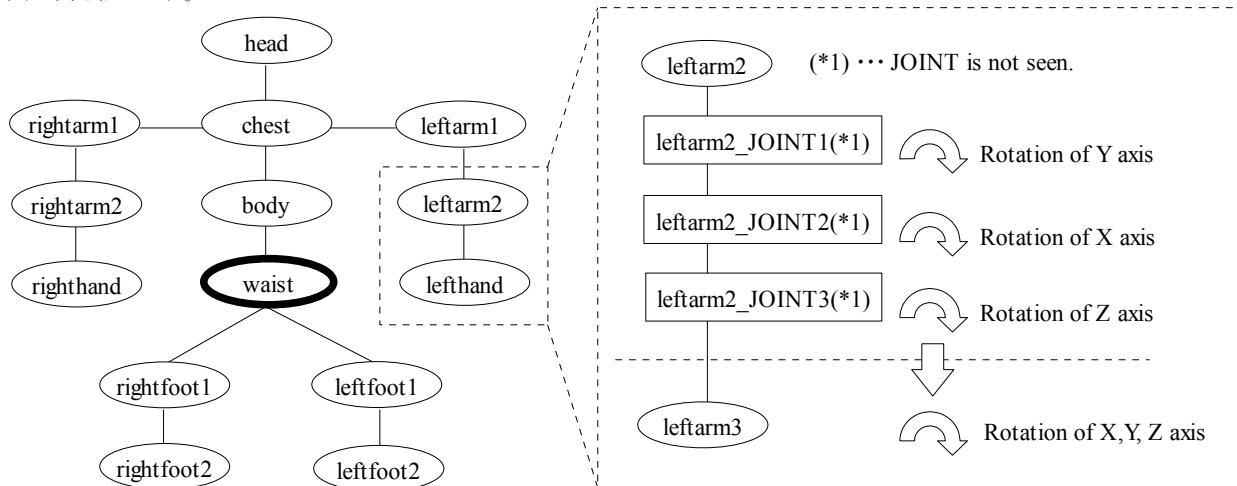
As for the Okonomiyaki cooperation dish, the human body with not a simple figure but indirectly complex can be operated. This is called a bodily movement. This operation is controlled with the file preserved in the "simserver/csv/motions/*" directory. In this paragraph, it explains the setting of the bodily movement.

お好み焼き協調理は、単純な図形ではなく、複雑な間接を持つ人体の動作が可能です。これは身体動作と呼びます。この動作は「csv/motions/*」ディレクトリに保存しているファイルで制御します。この項では、その身体動作の設定について説明します。

4.9.1.1. 身体動作の構造

The human body is composed of parts of 16 such as left chest, heads, and shoulders and right shoulders by centering on the waist. This is the same in the robot as Avator. And, the rotation axis not seen is defined between these parts. This is called "JOINT". JOINT can be rotated only in one direction of either X axis, Y axis or Z axis. These three JOINT is set between parts. These three JOINT is set between parts, and the rotation of three dimensions is enabled. Indirect is operated in this manner. The example of figure of the concept is written as follows.

人体は、腰を中心として、胸・頭・左肩・右肩などの16の部位から構成されます。これはアバタとロボットで一緒です。そして、この部位の間には、見えない回転軸を定義します。これを「JOINT」と呼びます。JOINTはX軸・Y軸・Z軸のいずれかの1方向のみに回転できます。部位の間にこのJOINTを3つ設定して、3次元の回転を可能にします。間接はこの様に動作させます。以下に概念の図の例を記します。



4.9.1.2. 身体動作の間接

Indirect rotates by JOINT and operates. In the CSV programming, the bodily movement is achieved specifying the angle for this JOINT. Therefore, the name is necessary for JOINT. JOINT of each joint and the list of the name are written as follows.

間接はJOINTにより回転して動作します。CSVプログラミングでは、このJOINTに角度を指定して身体動作を実現します。その為、JOINTには名前が必要です。以下に各関節のJOINTとその名前の一覧を記します。

Name of parts 部位名	Definition of JOINT JOINTの定義				Brief description 簡単な説明
	Parents' part names 親の部位名	Y axis Y軸	X axis X軸	X axis Z軸	
waist	—	—	—	—	—
body	waist	WAIST_JOINT2	CHEST	—	It rotates to Y axis and X axis by the waist. 腰でY軸とX軸に回転します
chest	body	—	—	—	—

Name of parts 部位名	Definition of JOINT JOINT の定義				Brief description 簡単な説明
	Parents' part names 親の部位名	Y axis Y 軸	X axis X 軸	X axis Z 軸	
head	chest	HEAD_JOINT1	HEAD_JOINT0	—	It rotates to Y axis and X axis by the neck. 首で Y 軸と X 軸に回転します
leftarm1	chest	LARM_JOINT0	LARM_JOINT1	LARM_JOINT2	It rotates to Y axis, X axis, and Z axis by a left shoulder. 左肩で Y 軸と X 軸と Z 軸に回転します
leftarm2	leftarm1	LARM_JOINT3	LARM_JOINT4	—	It rotates to Y axis and X axis by a left elbow. 左肘で Y 軸と X 軸に回転します
lefthand	leftarm2	LARM_JOINT5	LARM_JOINT6	LARM_JOINT7	It rotates to Y axis, X axis, and Z axis by a left wrist. 左手首で Y 軸と X 軸と Z 軸に回転します
rightarm1	chest	RARM_JOINT0	RARM_JOINT1	RARM_JOINT2	It rotates to Y axis, X axis, and Z axis by a right shoulder. 右肩で Y 軸と X 軸と Z 軸に回転します
rightarm2	rightarm1	RARM_JOINT3	RARM_JOINT4	—	It rotates to Y axis and X axis by a right elbow. 右肘で Y 軸と X 軸に回転します
righthand	rightarm2	RARM_JOINT5	RARM_JOINT6	RARM_JOINT7	It rotates to Y axis, X axis, and Z axis by a right wrist. 右手首で Y 軸と X 軸と Z 軸に回転します
leftfoot1	waist	—	LLEG_JOINT2	—	It rotates to X axis in the root of a right leg. 左脚の付け根で X 軸に回転します
leftfoot2	leftfoot1	—	LLEG_JOINT4	—	It rotates to X axis by a left knee. 左膝で X 軸に回転します
rightfoot1	waist	—	RLEG_JOINT2	—	It rotates to X axis in the root of a right leg. 右脚の付け根で X 軸に回転します
rightfoot2	rightfoot1	—	RLEG_JOINT4	—	It rotates to X axis by a right knee. 右膝で X 軸に回転します

4.9.1.3. 身体動作の命名規則

"Composition indirectly of human body" definition is necessary in the processing of the bodily movement. The purpose of the reason is for the relation the part and indirectly to have to learn the relation between them because of "1: many". This definition is set with the file said, "Joint definition file". The format is written the naming convention of the file as follows.

「Composition indirectly of human body」の定義は、身体動作の処理で必要です。理由は、部位と間接の関係が「1:多」の関係で、それらの関連を知る必要がある為です。この定義は「Joint 定義ファイル」と言うファイルで設定します。以下にそのファイルの命名規則と書式を記します。

Type name	Naming convention	Definition example
joint	"Agent name" + "_motions.csv"	Robot_000_motions.csv, Avator_000_motions.csv

Item name	Number of col	Content of item	Type of item	Input example
Parts name	1	The name of the part is specified. This name is used with the bodily movement definition file. 部位の名前を指定します。この名前が身体動作定義ファイルで利用されます。	Alphanumeric character in 48 bytes or less	robo_leftarm1

Item name	Number of col	Content of item	Type of item	Input example
X axis joint name	2	The name of JOINT that corresponds to the rotation of X axis of the part is specified. 部位の X 軸の回転に該当する JOINT の名前を指定します。	Alphanumeric character in 48 bytes or less	LARM_JOINT1
Y axis joint name	3	The name of JOINT that corresponds to the rotation of Y axis of the part is specified. 部位の Y 軸の回転に該当する JOINT の名前を指定します。	Alphanumeric character in 48 bytes or less	LARM_JOINT0
Z axis joint name	4	The name of JOINT that corresponds to the rotation of Z axis of the part is specified. 部位の Z 軸の回転に該当する JOINT の名前を指定します。	Alphanumeric character in 48 bytes or less	LARM_JOINT2
Direct movement	5	The bodily movement operates gradually. The stage is five stages in default. However, to operate directly up to a specified angle without gradually moving it, "1" is specified for here. 身体動作は、段階的に動作します。その段階はデフォルトで5段階です。ですが、段階的に動かさずに、直接指定の角度まで動作させたい場合、ここに「1」を指定します。	{0, 1}	1

Definition Ex.

```

robo_hip,WAIST_JOINT0,,,0
robo_body,CHEST,WAIST_JOINT2,,0
robo_chest,,,,0
robo_face,HEAD_JOINT0,HEAD_JOINT1,,0
robo_leftarm1,LARM_JOINT1,LARM_JOINT0,LARM_JOINT2,0 } 1 part difinition(1 row)
robo_leftarm2,LARM_JOINT4,LARM_JOINT3,,0
robo_lefthand1,LARM_JOINT6,LARM_JOINT5,LARM_JOINT7,0
robo_lefthand2,,,,0
robo_rightarm1,RARM_JOINT1,RARM_JOINT0,RARM_JOINT2,0
robo_rightarm2,RARM_JOINT4,RARM_JOINT3,,0
robo_righthand1,RARM_JOINT6,RARM_JOINT5,RARM_JOINT7,0
robo_righthand2,,,,0
robo_leftleg1,LLEG_JOINT2,,,1
robo_leftleg2,LLEG_JOINT4,,,1
robo_rightleg1,RLEG_JOINT2,,,1
robo_rightleg2,RLEG_JOINT4,,,1
robo_leftfood,,,,0
robo_rightfood,,,,0
    
```

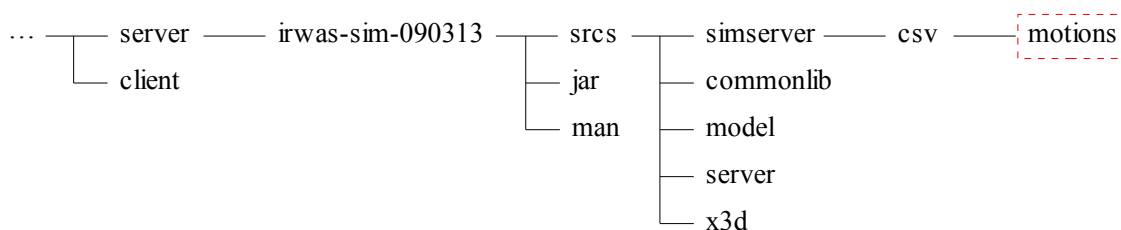
Parts name
Joint name

Indirect of Avator or the robot is moved by this definition. この定義でアバタまたはロボットの間接を動かします。

4.9.1.4. 身体動作の動作環境

The definition file of the bodily movement is preserved in directory "csv/motions" under "simserver" that Central Server starts. It is in the SIGVerse directory. The sample of Figure is written as follows.

身体動作の定義ファイルは、セントラルサーバが起動する「simserver」の下のディレクトリ「csv/motions」に保存されています。それはSIGVerseのディレクトリの中です。以下に図例を記します。



It moves to `simserver/csv/motions` and the list of the definition file is displayed. Please operate as follows, and confirm the definition file.

`simserver/csv/motions` に移動して定義ファイルの一覧を表示します。以下の操作を行い、定義ファイルを確認してください。

```

Operation Ex.   cd $HOME/sigverse-install-pack-090313/server/irwas-sim-090313/srcs/simserver/csv/motions
                  ls

Confirmation Ex.  ges00a ges08a ges15a ges22r ges42r ges56r ges74a ges86r Robot_000_motions.csv
                  ges00r ges08r ges15r ges23a ges43a ges62a ges74r ges92a Avator_000_motions.csv
                  ges01a ges09a ges16a ges23r ges43r ges62r ges75a ges92r
                  ges01r ges09r ges16r ges24a ges44r ges63a ges75r ges93a
                  Joint definition file
                  ⋮
                  bodily movement definition file
    
```

The definition files are files other than the Joint definition file. The name of the definition file is the same as the name of the bodily movement specified with Operation. Moreover, it is the same as the name specified by processing the state transition "Motion". The name of the definition file of this bodily movement is free. The bodily movement reads the content of these body definition files, and does various bodily movements.

定義ファイルは、Joint 定義ファイル以外です。定義ファイルの名前は、Operation で指定した身体動作の名前と同じです。また、状態遷移の処理「Motion」で指定する名前と同じです。この身体動作の定義ファイルの名前は、自由です。身体動作は、これら身体定義ファイルの内容を読み取り、様々な身体動作を行います。

4.9.1.5. 身体動作の書式

Difinition of the bodily movement is defined in the file of Comma Separated Value. Format is written as follows.

身体動作の定義は、CSV 形式のファイルに定義します。以下にその書式を記します。

Item name	Number of col	Content of item	Type of item	Input example
Parts name	1	The name of the part is specified. This name is a part name defined in the Joint definition file. 部位の名前を指定します。この名前は、Joint 定義ファイルで定義した部位名です。	Alphanumeric character in 48 bytes or less	robo_leftarm1
Agent name	1	When coordinates or externals are specified for "Excluding dead letter character" by this definition, the first row becomes not the part name but the name of the agent. In that case, please specify a correct name of the agent. この定義で座標か見た目を「空文字以外」に指定した場合、1 列目は、部位名ではなくエージェント名となります。その場合、正しいエージェント名を指定します。	Alphanumeric character in 48 bytes or less	Sauce_000
Coordinates of X	2	The agent who specified it by the first row is moved to specified X coordinates point. The X coordinates point is specified. 1 列目で指定したエージェントを指定 X 座標点に移動します。その X 座標点を指定します。	Numerical value	100
Coordinates of Y	3	The agent who specified it by the first row is moved to specified Y coordinates point. The Y coordinates point is specified. 1 列目で指定したエージェントを指定 Y 座標点に移動します。その Y 座標点を指定します。	Numerical value	70
Coordinates of Z	4	The agent who specified it by the first row is moved to specified Y coordinates point. The Y coordinates point is specified. 1 列目で指定したエージェントを指定 Y 座標点に移動します。その Y 座標点を指定します。	Numerical value	100
Turning angle	5	The part specified by the first row is rotated and the amount of a	Numerical value	-90

Item name	Number of col	Content of item	Type of item	Input example
degree of X axis		specified angle is rotated to X axis. The angle that rotates is specified. 1列目で指定した部位をX軸に指定角度の分回転させます。回転させたい角度を指定します。		
Turning angle degree of X axis	6	The part specified by the first row is rotated and the amount of a specified angle is rotated to X axis. The angle that rotates is specified. 1列目で指定した部位をX軸に指定角度の分回転させます。回転させたい角度を指定します。	Numerical value	90
Turning angle degree of X axis	7	The part specified by the first row is rotated and the amount of a specified angle is rotated to X axis. The angle that rotates is specified. 1列目で指定した部位をX軸に指定角度の分回転させます。回転させたい角度を指定します。	Numerical value	180
Name of externals	8	Externals of the agent who specified it by the first row are changed to specified externals. The name of externals that change is specified. This value is the same as the value specified by the Visual command. 1列目で指定したエージェントの見た目を、指定した見た目に変更します。変更したい見た目の名前を指定します。この値はVisualコマンドで指定した値と同じです。	Alphanumeric character in 48 bytes or less	8

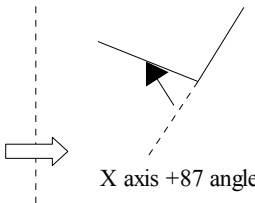
Definition Ex.

```

robo_hip,,,,10.000,0.000,0.000,0
robo_body,,,,12.000,20.000,0.000,0
robo_chest,,,,0.000,0.000,0.000,0
robo_face,,,,0.000,5.000,0.000,0
robo_leftarm1,,,,-87.000,0.000,0.000,0
    
```

Part name

 Angle information



When this definition specifies 2, 3, 4, and 8 of the rows for "Excluding dead letter character", one of the rows becomes no "Part name" it, and "Name of the agent". The definition example is written as follows.

この定義は、列の2,3,4,8を「空文字以外」に指定した場合、列の1は「部位名」ではなく、「エージェント名」になります。以下にその定義例を記します。

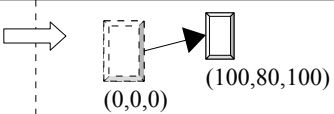
Definition Ex.

```

Sauce_000,100,80,100,,,, } 1 agent difinition(1 row)
    
```

Agent name

 pos information



4.9.1.6. ヒューマノイド型の身体動作

In the Okonomiyaki cooperation dish, it is assumed that only Avator and the robot are agents of a humanoid type, and the bodily movement can be done. The bodily movement is not done, and the definition is not necessary for other agents either. The definition of Avator and the robot is written as an example as follows. These bodily movements are the bodily movements done at the Okonomiyaki dish.

お好み焼き協料理では、アバタとロボットのみがヒューマノイド型のエージェントであり、身体動作を行えるとしています。その他のエージェントは、身体動作は行わず、その定義も必要ありません。以下に、アバタとロボットの定義を例として書きます。これらの身体動作は、お好み焼き料理の時に行う身体動作です。

(*1)・・・ Avator and the robot assume that it keeps assuming sitting always while cooking, setting the angle for the leg in the initial state, and maintaining it.

4.9.1.7. ロボットの身体動作の定義

Name in state	Action when state changes	Bodily movement				Movement and Rotation			Visual
		Parts name	X axis	Y axis	Z axis	X point	Y point	Z point	External
0	Initial state(*1) 初期状態	leftfoot1	-80	—	—	—	—	—	—
		leftfoot2	80	—	—	—	—	—	—
		rightfoot1	-80	—	—	—	—	—	—
		rightfoot2	80	—	—	—	—	—	—
1	Mix dough (It reaches oil.) 生地を混ぜる (生地に手を伸ばす)	body	12	20	—	—	—	—	—
		head	—	10	—	—	—	—	—
		leftarm1	87	—	—	—	—	—	—
		leftarm2	5	—	—	—	—	—	—
2	Oil is drawn on the iron plate. (The left hand is extended to oil.) 油を手元に引き寄せる (油に左手を伸ばす)	body	12	30	—	—	—	—	—
		head	—	10	—	—	—	—	—
		leftarm1	-80	—	15	—	—	—	—
		leftarm2	5	—	—	—	—	—	—
	Oil is drawn on the iron plate. (Oil is moved to the center of the iron plate.) 油を手元に引き寄せる (油を鉄板の中央に移動する)	body	12	-15	—	—	—	—	—
		head	—	10	—	—	—	—	—
		leftarm1	-80	-10	—	—	—	—	—
		leftarm2	-20	—	—	—	—	—	—
3	Oil is painted to the iron plate. (The brush is moved on the iron plate.) 油を鉄板にひく (刷毛を鉄板上で動かす)	body	12	-35	—	—	—	—	—
		face	12	—	—	—	—	—	—
		leftarm1	-80	-15	15	—	—	—	—
		leftarm2	-20	—	—	—	—	—	—
		Abura_000	—	—	—	80	70	100	—
	Oil is painted to the iron plate. (The brush is returned to former position.) 油を鉄板にひく (刷毛を元の位置に戻す)	body	12	—	—	—	—	—	—
		face	12	—	—	—	—	—	—
		leftarm1	-80	-15	—	—	—	—	—
		leftarm2	-20	—	—	—	—	—	—
		Abura_000	—	—	—	110	70	100	—
4	The amount of the iron plate of the gas remainder is confirmed. 鉄板のコンロのガス残量を確認する	body	12	-20	—	—	—	—	—
5	The fire is set fire to the iron plate. 鉄板のガスコンロに火を付ける	body	12	-15	—	—	—	—	—
		rightarm1	-55	-10	—	—	—	—	—
		rightarm2	-25	70	—	—	—	—	—
		Teppan_000	—	—	—	—	—	—	1

Name in state	Action when state changes	Bodily movement				Movement and Rotation			Visual
		Parts name	X axis	Y axis	Z axis	X point	Y point	Z point	External
6	The ingredients is put on the iron plate. (It reaches the ingredients.) 生地を鉄板の上へのせる (生地に手を伸ばす)	body	12	20	—	—	—	—	—
		face	—	15	—	—	—	—	—
		leftarm1	-90	—	—	—	—	—	—
		leftarm2	-10	—	—	—	—	—	—
	The ingredients is put on the iron plate. (The bowl is moved on the iron plate.) 生地を鉄板の上へのせる (ボウルを鉄板の上に移動させる)	body	—	-15	—	—	—	—	—
		face	12	—	—	—	—	—	—
		leftarm1	-70	-10	—	—	—	—	—
		leftarm2	-20	—	—	—	—	—	—
		Bowl_000	—	—	—	100	80	100	—
	The ingredients is put on the iron plate. (The bowl is inclined.) 生地を鉄板の上へのせる (ボウルを傾ける)	body	—	-10	—	—	—	—	—
		face	12	—	—	—	—	—	—
		leftarm1	-80	-15	—	—	—	—	—
		leftarm2	20	—	—	—	—	—	—
		Bowl_000	—	—	90	—	—	—	—
		Okonomi_000	—	—	—	—	—	—	2
The ingredients is put on the iron plate. It returns it based on the inclination of the bowl.) 生地を鉄板の上へのせる (ボウルの傾きを元に戻す)	body	—	-10	—	—	—	—	—	
	face	12	—	—	—	—	—	—	
	leftarm1	-70	-15	—	—	—	—	—	
	leftarm2	-20	10	—	—	—	—	—	
	Bowl_000	—	—	0	—	—	—	—	
7	Pork is taken. (It reaches pork.) 豚肉を手元に引き寄せる (豚肉に手を伸ばす)	body	12	-20	—	—	—	—	—
		rightarm1	-85	-10	—	—	—	—	—
		rightarm2	-25	25	—	—	—	—	—
	Pork is taken. (Pork is moved near the iron plate.) 豚肉を手元に引き寄せる (豚肉を鉄板の近くに移動)	body	12	15	—	—	—	—	—
		rightarm1	-90	—	—	—	—	—	—
		rightarm2	-15	20	—	—	—	—	—
8	Pork is put on the ingredients. 豚肉を生地の上へのせる	body	12	20	—	—	—	—	—
		rightarm1	-90	—	—	—	—	—	—
		rightarm2	-15	25	—	—	—	—	—
		Okonomi_000	—	—	—	—	—	—	3
	Pork is put on the ingredients. (Pork is pressed against Okonomiyaki.) 豚肉を生地の上へのせる (豚肉を押し付ける)	body	12	15	—	—	—	—	—
		rightarm1	-90	—	—	—	—	—	—
9	The combustion addition and subtraction of Okonomiyaki is seen.	body	15	—	—	—	—	—	—
		leftarm1	-20	—	—	—	—	—	—

Name in state	Action when state changes	Bodily movement				Movement and Rotation			Visual
		Parts name	X axis	Y axis	Z axis	X point	Y point	Z point	External
	お好み焼きの焼け加減を見る	leftarm2	-10	—	—	—	—	—	—
		rightarm1	-20	—	—	—	—	—	—
		rightarm2	-10	—	—	—	—	—	—
10	Okonomiyaki is turned inside out. (The hand is extended to the Hera.) お好み焼きを裏返す (手をへらに伸ばす)	body	20	10	—	—	—	—	—
		face	10	—	—	—	—	—	—
		leftarm1	-80	10	—	—	—	—	—
		leftarm2	-20	—	—	—	—	—	—
		rightarm1	-80	-10	—	—	—	—	—
		rightarm2	-20	—	—	—	—	—	—
	Okonomiyaki is turned inside out. (The direction of the Hera is changed.) お好み焼きを裏返す (へらの方向を変える)	body	20	—	—	—	—	—	—
		face	10	—	—	—	—	—	—
		leftarm1	-50	10	—	—	—	—	—
		leftarm2	-50	—	—	—	—	—	—
		rightarm1	-50	10	—	—	—	—	—
		rightarm2	-50	—	—	—	—	—	—
		Hera_000	90	—	—	—	—	—	—
	Hera_001	-90	—	—	—	—	—	—	
	Okonomiyaki is turned inside out. (Okonomiyaki is turned inside out.) (The Hera is lifted.) お好み焼きを裏返す (お好み焼きを裏返す) (へらを持ち上げる)	body	0	—	—	—	—	—	—
		face	5	—	—	—	—	—	—
		leftarm1	-80	15	—	—	—	—	—
		leftarm2	-40	10	—	—	—	—	—
		rightarm1	-80	-15	—	—	—	—	—
		rightarm2	-40	-10	—	—	—	—	—
		Hera_000	—	—	—	—	70	—	—
		Hera_001	—	—	—	—	70	—	—
	Okonomiyaki is turned inside out. (The spatula is lowered.) お好み焼きを裏返す (へらを持ち下げる)	body	10	0	—	—	—	—	—
		face	5	—	—	—	—	—	—
leftarm1		-50	15	—	—	—	—	—	
leftarm2		-50	10	—	—	—	—	—	
rightarm1		-50	-15	—	—	—	—	—	
rightarm2		-50	-10	—	—	—	—	—	
Hera_000		—	—	—	—	80	—	—	
Hera_001		—	—	—	—	80	—	—	
Okonomiyaki is turned inside out. (It returns it based on Hera.)	body	10	0	—	—	—	—	—	
	face	5	—	—	—	—	—	—	

Name in state	Action when state changes	Bodily movement			Movement and Rotation			Visual	
		Parts name	X axis	Y axis	Z axis	X point	Y point	Z point	External
	お好み焼きを裏返す (へらを元に戻す)	leftarm1	-50	15	—	—	—	—	—
		leftarm2	-50	10	—	—	—	—	—
		rightarm1	-50	-15	—	—	—	—	—
		rightarm2	-50	-10	—	—	—	—	—
		Hera_000	-90	—	—	—	—	—	—
		Hera_001	90	—	—	—	—	—	—
12	The source is taken. (It reaches the source.) ソースを手元に引き寄せる (ソースに手を伸ばす)	body	12	-20	—	—	—	—	—
		rightarm1	-85	-10	—	—	—	—	—
		rightarm2	-25	25	—	—	—	—	—
	The source is taken. (It is moved to the vicinity of the iron plate the source.) ソースを手元に引き寄せる (ソースを鉄板近くに移動させる)	body	15	15	—	—	—	—	—
		rightarm1	-90	—	—	—	—	—	—
		rightarm2	-15	25	—	—	—	—	—
Sauce_000	—	—	—	100	80	100	—		
13	The source is put on Okonomiyaki. (The source is inclined.) (Externals of Okonomiyaki are changed.) ソースをお好み焼きに塗る (ソースを傾ける) (お好み焼きの見た目を変える)	body	15	15	—	—	—	—	—
		rightarm1	-90	—	70	—	—	—	—
		rightarm2	-25	—	—	—	—	—	—
		Sauce_000	—	—	-90	—	—	—	—
		Okonomi_000	—	—	—	—	—	—	5
	The source is put on Okonomiyaki (Return it based on the source). ソースをお好み焼きに塗る (ソースを元に戻す)	body	15	—	—	—	—	—	—
		rightarm1	-80	-10	—	—	—	—	—
		rightarm2	-25	25	—	—	—	—	—
	Sauce_000	—	—	0	—	—	—	—	
	Okonomiyaki is put on the source (The source is returned to former position). ソースにお好み焼きを塗る (ソースを元の位置に戻す)	body	15	-20	—	—	—	—	—
rightarm1		-85	-10	—	—	—	—	—	
rightarm2		-20	20	—	—	—	—	—	
Sauce_000	—	—	—	42	69	95	—		
14	Seaweed is taken. (It reaches seaweed.) 海苔を手元に引き寄せる (海苔に手を伸ばす)	body	12	-20	—	—	—	—	—
		rightarm1	-85	-10	—	—	—	—	—
		rightarm2	-25	25	—	—	—	—	—
	Seaweed is taken. (Seaweed is moved near the iron plate.) 海苔を手元に引き寄せる (海苔を鉄板の近くに移動する)	body	15	15	—	—	—	—	—
		rightarm1	-90	—	—	—	—	—	—
		rightarm2	-15	25	—	—	—	—	—
Nori_000	—	—	—	100	80	100	—		
15	Seaweed is put on Okonomiyaki. (Seaweed is inclined.)	body	15	15	—	—	—	—	—
		rightarm1	-90	—	70	—	—	—	—

Name in state	Action when state changes	Bodily movement			Movement and Rotation			Visual	
		Parts name	X axis	Y axis	Z axis	X point	Y point	Z point	External
	(Externals of Okonomiyaki are changed.) 海苔をお好み焼きに振りかける (海苔を傾ける) (お好み焼きの見た目を変える)	rightarm2	-25	—	—	—	—	—	—
		Nori_000	—	—	-90	—	—	—	—
		Okonomi_000	—	—	—	—	—	—	6
	Seaweed is put on Okonomiyaki (Return it based on seaweed). 海苔をお好み焼きに振りかける (海苔を元に戻す)	body	15	—	—	—	—	—	—
		rightarm1	-80	-10	—	—	—	—	—
		rightarm2	-25	25	—	—	—	—	—
		Nori_000	—	—	0	—	—	—	—
	Okonomiyaki is put on seaweed (Seaweed is returned to former position). 海苔にお好み焼きを振りかける (海苔を元の位置に戻す)	body	15	-20	—	—	—	—	—
		rightarm1	-85	-10	—	—	—	—	—
		rightarm2	-20	20	—	—	—	—	—
		Nori_000	—	—	—	42	69	96	—
	16	The dried bonito is taken. (It reaches the dried bonito.) 鰹節を手元に引き寄せる (鰹節に手を伸ばす)	body	12	-20	—	—	—	—
rightarm1			-85	-10	—	—	—	—	—
rightarm2			-25	25	—	—	—	—	—
The dried bonito is taken. (The dried bonito is moved near the iron plate.) 鰹節を手元に引き寄せる (鰹節を鉄板の近くに移動する)		body	15	15	—	—	—	—	—
		rightarm1	-90	—	—	—	—	—	—
		rightarm2	-15	25	—	—	—	—	—
Katsuobushi_000	—	—	—	100	80	100	—		
17	The dried bonito is put on Okonomiyaki. (The dried bonito is inclined.) (Externals of Okonomiyaki are changed.) 鰹節をお好み焼きに振りかける (鰹節を傾ける) (お好み焼きの見た目を変える)	body	15	15	—	—	—	—	—
		rightarm1	-90	—	70	—	—	—	—
		rightarm2	-25	—	—	—	—	—	—
		Katsuobushi_000	—	—	-90	—	—	—	—
		Okonomi_000	—	—	—	—	—	—	7
	The dried bonito is put on Okonomiyaki (Return it based on the dried bonito). 鰹節をお好み焼きに振りかける (鰹節を元に戻す)	body	15	—	—	—	—	—	—
		rightarm1	-80	-10	—	—	—	—	—
		rightarm2	-25	25	—	—	—	—	—
		Katsuobushi_000	—	—	0	—	—	—	—
	The dried bonito is put on Okonomiyaki (The dried bonito is returned to former position). 鰹節をお好み焼きに振りかける (鰹節を元の位置に戻す)	body	15	-20	—	—	—	—	—
		rightarm1	-85	-10	—	—	—	—	—
		rightarm2	-20	20	—	—	—	—	—
Katsuobushi_000		—	—	—	54	69	112	—	
18	Okonomiyaki is divided. お好み焼きを切り分ける								
19	Okonomiyaki is piled up in the plate. お好み焼きを皿に盛る								

Name in state	Action when state changes	Bodily movement				Movement and Rotation			Visual
		Parts name	X axis	Y axis	Z axis	X point	Y point	Z point	External
20	The fire of the iron plate is weakened. 鉄板のガスコンロの火を弱める	body	12	-15	—	—	—	—	—
		rightarm1	-55	-10	—	—	—	—	—
		rightarm2	-25	70	—	—	—	—	—
		Teppan_000	0	0	—	—	—	—	—
21	The fire of the iron plate is strengthened. 鉄板のガスコンロの火を強める	body	12	-15	—	—	—	—	—
		rightarm1	-55	-10	—	—	—	—	—
		rightarm2	-25	70	—	—	—	—	—
		Teppan_000	0	0	—	—	—	—	—
22	The fire of the iron plate is put out. 鉄板のガスコンロの火を消す	body	12	-15	—	—	—	—	—
		rightarm1	-55	-10	—	—	—	—	—
		rightarm2	-25	70	—	—	—	—	—
		Teppan_000	0	0	—	—	—	—	—

4.9.1.8. アバタの身体動作の定義

Name in state	Action when state changes	Bodily movement				Movement and Rotation			Visual
		Parts name	X axis	Y axis	Z axis	X point	Y point	Z point	External
0	Initial state(*1) 初期状態	leftfoot1	-80	—	—	—	—	—	—
		leftfoot2	80	—	—	—	—	—	—
		rightfoot1	-80	—	—	—	—	—	—
		rightfoot2	80	—	—	—	—	—	—
1	Mix dough (It reaches oil.) 生地を混ぜる (油に手を伸ばす)	body	12	20	—	—	—	—	—
		head	—	10	—	—	—	—	—
		leftarm1	87	—	—	—	—	—	—
		leftarm2	5	—	—	—	—	—	—
2	Oil is drawn on the iron plate. (The left hand is extended to oil.) 油を手元に引き寄せる (油に左手を伸ばす)	body	12	30	—	—	—	—	—
		head	—	10	—	—	—	—	—
		leftarm1	-80	—	15	—	—	—	—
		leftarm2	5	—	—	—	—	—	—
	Oil is drawn on the iron plate. (Oil is moved to the center of the iron plate.) 油を手元に引き寄せる (油を鉄板の中央に移動する)	body	12	-15	—	—	—	—	—
		head	—	10	—	—	—	—	—
		leftarm1	-80	-10	—	—	—	—	—
		leftarm2	-20	—	—	—	—	—	—
3	Oil is painted to the iron plate. (The brush is moved on the iron	body	12	-35	—	—	—	—	—
		face	12	—	—	—	—	—	—

Name in state	Action when state changes	Bodily movement				Movement and Rotation			Visual	
		Parts name	X axis	Y axis	Z axis	X point	Y point	Z point	External	
	plate.) 油を鉄板にひく (刷毛を鉄板上で動かす)	leftarm1	-80	-15	15	—	—	—	—	
		leftarm2	-20	—	—	—	—	—	—	
		Abura_000	—	—	—	80	70	100	—	
	Oil is painted to the iron plate. (The brush is returned to former position.) 油を鉄板にひく (刷毛を元の位置に戻す)	body	12	—	—	—	—	—	—	
		face	12	—	—	—	—	—	—	
		leftarm1	-80	-15	—	—	—	—	—	
		leftarm2	-20	—	—	—	—	—	—	
	Abura_000	—	—	—	110	70	100	—		
4	The amount of the iron plate of the gas remainder is confirmed. 鉄板のコンロのガス残量を確認する	body	12	-20	—	—	—	—	—	
5	The fire is set fire to the iron plate. 鉄板のガスコンロに火を付ける	body	12	-15	—	—	—	—	—	
		rightarm1	-55	-10	—	—	—	—	—	
		rightarm2	-25	70	—	—	—	—	—	
		Teppan_000	—	—	—	—	—	—	1	
6	The ingredients is put on the iron plate. (It reaches the ingredients.) 生地を鉄板の上へのせる (生地に手を伸ばす)	body	12	20	—	—	—	—	—	
		face	—	15	—	—	—	—	—	
		leftarm1	-90	—	—	—	—	—	—	
		leftarm2	-10	—	—	—	—	—	—	
	The ingredients is put on the iron plate. (The bowl is moved on the iron plate.) 生地を鉄板の上へのせる (ボウルを鉄板の上に移動させる)	body	—	-15	—	—	—	—	—	
		face	12	—	—	—	—	—	—	
		leftarm1	-70	-10	—	—	—	—	—	
		leftarm2	-20	—	—	—	—	—	—	
		Bowl_000	—	—	—	100	80	100	—	
			The ingredients is put on the iron plate. (The bowl is inclined.) 生地を鉄板の上へのせる (ボウルを傾ける)	body	—	-10	—	—	—	—
	face	12		—	—	—	—	—	—	
	leftarm1	-80		-15	—	—	—	—	—	
	leftarm2	20		—	—	—	—	—	—	
	Bowl_000	—		—	90	—	—	—	—	
	Okonomi_000	—		—	—	—	—	—	2	
		The ingredients is put on the iron plate. It returns it based on the inclination of the bowl.) 生地を鉄板の上へのせる (ボウルの傾きを元に戻す)	body	—	-10	—	—	—	—	—
			face	12	—	—	—	—	—	—
			leftarm1	-70	-15	—	—	—	—	—
leftarm2			-20	10	—	—	—	—	—	
Bowl_000			—	—	0	—	—	—	—	
7	Pork is taken. (It reaches pork.)	body	12	-20	—	—	—	—	—	
		rightarm1	-85	-10	—	—	—	—	—	

Name in state	Action when state changes	Bodily movement				Movement and Rotation			Visual
		Parts name	X axis	Y axis	Z axis	X point	Y point	Z point	External
	豚肉を手元に引き寄せる (豚肉に手を伸ばす)	rightarm2	-25	25	—	—	—	—	—
	Pork is taken. (Pork is moved near the iron plate.)	body	12	15	—	—	—	—	—
	豚肉を手元に引き寄せる (豚肉を鉄板の近くに移動)	rightarm1	-90	—	—	—	—	—	—
		rightarm2	-15	20	—	—	—	—	—
8	Pork is put on the ingredients. 豚肉を生地の上へのせる	body	12	20	—	—	—	—	—
		rightarm1	-90	—	—	—	—	—	—
		rightarm2	-15	25	—	—	—	—	—
		Okonomi_000	—	—	—	—	—	—	3
	Pork is put on the ingredients. (Pork is pressed against Okonomiyaki.) 豚肉を生地の上へのせる (豚肉を押し付ける)	body	12	15	—	—	—	—	—
		rightarm1	-90	—	—	—	—	—	—
9	The combustion addition and subtraction of Okonomiyaki is seen. お好み焼きの焼け加減を見る	body	15	—	—	—	—	—	—
		leftarm1	-20	—	—	—	—	—	—
		leftarm2	-10	—	—	—	—	—	—
		rightarm1	-20	—	—	—	—	—	—
		rightarm2	-10	—	—	—	—	—	—
10	Okonomiyaki is turned inside out. (The hand is extended to the Hera.) お好み焼きを裏返す (手をへらに伸ばす)	body	20	10	—	—	—	—	—
		face	10	—	—	—	—	—	—
		leftarm1	-80	10	—	—	—	—	—
		leftarm2	-20	—	—	—	—	—	—
		rightarm1	-80	-10	—	—	—	—	—
		rightarm2	-20	—	—	—	—	—	—
	Okonomiyaki is turned inside out. (The direction of the Hera is changed.) お好み焼きを裏返す (へらの方向を変える)	body	20	—	—	—	—	—	—
		face	10	—	—	—	—	—	—
		leftarm1	-50	10	—	—	—	—	—
		leftarm2	-50	—	—	—	—	—	—
		rightarm1	-50	10	—	—	—	—	—
		rightarm2	-50	—	—	—	—	—	—
	Okonomiyaki is turned inside out. (Okonomiyaki is turned inside out.) (The Hera is lifted.) お好み焼きを裏返す (お好み焼きを裏返す) (へらを持ち上げる)	body	0	—	—	—	—	—	—
		face	5	—	—	—	—	—	—
		leftarm1	-80	15	—	—	—	—	—
		leftarm2	-40	10	—	—	—	—	—
		rightarm1	-80	-15	—	—	—	—	—

Name in state	Action when state changes	Bodily movement				Movement and Rotation			Visual
		Parts name	X axis	Y axis	Z axis	X point	Y point	Z point	External
		rightarm2	-40	-10	—	—	—	—	—
		Hera_000	—	—	—	—	70	—	—
		Hera_001	—	—	—	—	70	—	—
		Okonomi_000	—	180	—	—	—	—	—
	Okonomiyaki is turned inside out. (The spatula is lowered.) お好み焼きを裏返す (へらを持ち下げる)	body	10	0	—	—	—	—	—
		face	5	—	—	—	—	—	—
		leftarm1	-50	15	—	—	—	—	—
		leftarm2	-50	10	—	—	—	—	—
		rightarm1	-50	-15	—	—	—	—	—
		rightarm2	-50	-10	—	—	—	—	—
		Hera_000	—	—	—	—	80	—	—
		Hera_001	—	—	—	—	80	—	—
	Okonomiyaki is turned inside out. (It returns it based on Hera.) お好み焼きを裏返す (へらを元に戻す)	body	10	0	—	—	—	—	—
		face	5	—	—	—	—	—	—
		leftarm1	-50	15	—	—	—	—	—
		leftarm2	-50	10	—	—	—	—	—
rightarm1		-50	-15	—	—	—	—	—	
rightarm2		-50	-10	—	—	—	—	—	
Hera_000		-90	—	—	—	—	—	—	
Hera_001		90	—	—	—	—	—	—	
12	The source is taken. (It reaches the source.) ソースを手元に引き寄せる (ソースに手を伸ばす)	body	12	-20	—	—	—	—	—
		rightarm1	-85	-10	—	—	—	—	—
		rightarm2	-25	25	—	—	—	—	—
	The source is taken. (It is moved to the vicinity of the iron plate the source.) ソースを手元に引き寄せる (ソースを鉄板近くに移動させる)	body	15	15	—	—	—	—	—
		rightarm1	-90	—	—	—	—	—	—
		rightarm2	-15	25	—	—	—	—	—
Sauce_000	—	—	—	100	80	100	—		
13	The source is put on Okonomiyaki. (The source is inclined.) (Externals of Okonomiyaki are changed.) ソースをお好み焼きに塗る (ソースを傾ける) (お好み焼きの見た目を変える)	body	15	15	—	—	—	—	—
		rightarm1	-90	—	70	—	—	—	—
		rightarm2	-25	—	—	—	—	—	—
		Sauce_000	—	—	-90	—	—	—	—
		Okonomi_000	—	—	—	—	—	—	5
	The source is put on Okonomiyaki (Return it based on the source). ソースをお好み焼きにかける (ソースを元に戻す)	body	15	—	—	—	—	—	—
		rightarm1	-80	-10	—	—	—	—	—
		rightarm2	-25	25	—	—	—	—	—
		Sauce_000	—	—	0	—	—	—	—

Name in state	Action when state changes	Bodily movement				Movement and Rotation			Visual
		Parts name	X axis	Y axis	Z axis	X point	Y point	Z point	External
	Okonomiyaki is put on the source (The source is returned to former position). ソースにお好み焼きを塗る (ソースを元の位置に戻す)	body	15	-20	—	—	—	—	—
		rightarm1	-85	-10	—	—	—	—	—
		rightarm2	-20	20	—	—	—	—	—
		Sauce_000	—	—	—	42	69	95	—
14	Seaweed is taken. (It reaches seaweed.) 海苔を手元に引き寄せる (海苔に手を伸ばす)	body	12	-20	—	—	—	—	—
		rightarm1	-85	-10	—	—	—	—	—
		rightarm2	-25	25	—	—	—	—	—
	Seaweed is taken. (Seaweed is moved near the iron plate.) 海苔を手元に引き寄せる (海苔を鉄板の近くに移動する)	body	15	15	—	—	—	—	—
		rightarm1	-90	—	—	—	—	—	—
		rightarm2	-15	25	—	—	—	—	—
15	Seaweed is put on Okonomiyaki. (Seaweed is inclined.) (Externals of Okonomiyaki are changed.) 海苔をお好み焼きに振りかける (海苔を傾ける) (お好み焼きの見た目を変える)	body	15	15	—	—	—	—	—
		rightarm1	-90	—	70	—	—	—	—
		rightarm2	-25	—	—	—	—	—	—
		Nori_000	—	—	-90	—	—	—	—
		Okonomi_000	—	—	—	—	—	—	6
	Seaweed is put on Okonomiyaki (Return it based on seaweed). 海苔をお好み焼きに振りかける (海苔を元に戻す)	body	15	—	—	—	—	—	—
		rightarm1	-80	-10	—	—	—	—	—
		rightarm2	-25	25	—	—	—	—	—
		Nori_000	—	—	0	—	—	—	—
	Okonomiyaki is put on seaweed (Seaweed is returned to former position). 海苔にお好み焼きを振りかける (海苔を元の位置に戻す)	body	15	-20	—	—	—	—	—
		rightarm1	-85	-10	—	—	—	—	—
		rightarm2	-20	20	—	—	—	—	—
Nori_000		—	—	—	42	69	96	—	
16	The dried bonito is taken. (It reaches the dried bonito.) 鰹節を手元に引き寄せる (鰹節に手を伸ばす)	body	12	-20	—	—	—	—	—
		rightarm1	-85	-10	—	—	—	—	—
		rightarm2	-25	25	—	—	—	—	—
	The dried bonito is taken. (The dried bonito is moved near the iron plate.) 鰹節を手元に引き寄せる (鰹節を鉄板の近くに移動する)	body	15	15	—	—	—	—	—
		rightarm1	-90	—	—	—	—	—	—
		rightarm2	-15	25	—	—	—	—	—
Katsuobushi_000	—	—	—	100	80	100	—		
17	The dried bonito is put on Okonomiyaki. (The dried bonito is inclined.) (Externals of Okonomiyaki are changed.) 鰹節をお好み焼きに振りかける	body	15	15	—	—	—	—	—
		rightarm1	-90	—	70	—	—	—	—
		rightarm2	-25	—	—	—	—	—	—
		Katsuobushi_000	—	—	-90	—	—	—	—

Name in state	Action when state changes	Bodily movement			Movement and Rotation			Visual	
		Parts name	X axis	Y axis	Z axis	X point	Y point	Z point	External
	(鰹節を傾ける) (お好み焼きの見た目を変える)	Okonomi_000	—	—	—	—	—	—	7
	The dried bonito is put on Okonomiyaki (Return it based on the dried bonito). 鰹節をお好み焼きに振りかける (鰹節を元に戻す)	body	15	—	—	—	—	—	—
		rightarm1	-80	-10	—	—	—	—	—
		rightarm2	-25	25	—	—	—	—	—
		Katsuobushi_000	—	—	0	—	—	—	—
	The dried bonito is put on Okonomiyaki (The dried bonito is returned to former position). 鰹節をお好み焼きに振りかける (鰹節を元の位置に戻す)	body	15	-20	—	—	—	—	—
		rightarm1	-85	-10	—	—	—	—	—
		rightarm2	-20	20	—	—	—	—	—
		Katsuobushi_000	—	—	—	54	69	112	—
18	Okonomiyaki is divided. お好み焼きを切り分ける								
19	Okonomiyaki is piled up in the plate. お好み焼きを皿に盛る								
20	The fire of the iron plate is weakened. 鉄板のガスコンロの火を弱める	body	12	-15	—	—	—	—	—
		rightarm1	-55	-10	—	—	—	—	—
		rightarm2	-25	70	—	—	—	—	—
		Teppan_000	0	0	—	—	—	—	—
21	The fire of the iron plate is strengthened. 鉄板のガスコンロの火を強める	body	12	-15	—	—	—	—	—
		rightarm1	-55	-10	—	—	—	—	—
		rightarm2	-25	70	—	—	—	—	—
		Teppan_000	0	0	—	—	—	—	—
22	The fire of the iron plate is put out. 鉄板のガスコンロの火を消す	body	12	-15	—	—	—	—	—
		rightarm1	-55	-10	—	—	—	—	—
		rightarm2	-25	70	—	—	—	—	—
		Teppan_000	0	0	—	—	—	—	—

4.2. 知覚に関する設定

In this paragraph, it explains the setting of "Change in externals" that relates to perception. SIGVerse can offer information on specified agent's sight as an image. However, it doesn't use it directly to handle it in the Okonomiyaki cooperation dish by "Information on higher-order" said, "State". The setting of "Change in externals" is treated here.

この項では、知覚に関連する「見た目の変更」の設定を説明します。SIGVerseは、指定エージェントの視覚の情報を画像として提供できます。ですが、お好み焼き協調料理では、「状態」と言う「高次の情報」で取り扱う為、直接に利用しません。ここでは「見た目の変更」の設定を扱います。














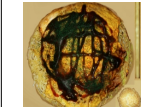


4.2.1. 設定のシナリオ

A present Okonomiyaki cooperation dish cooks the Okonomiyaki of "Kansai". "Kansai" is changed to the Okonomiyaki of "Hiroshima" by this setting. The procedure of the dish doesn't change. (* "Kansai" and "Hiroshima" are Japanese place-names.)

The relation between externals of the Okonomiyaki of externals of the Okonomiyaki of "Kansai" and "Hiroshima" is written as follows.

現在のお好み焼き協調料理は、「関西」のお好み焼きを料理します。この設定で「関西」を「広島」のお好み焼きに変更します。料理の手順は変わりません。(※「関西」と「広島」は日本の地名です)

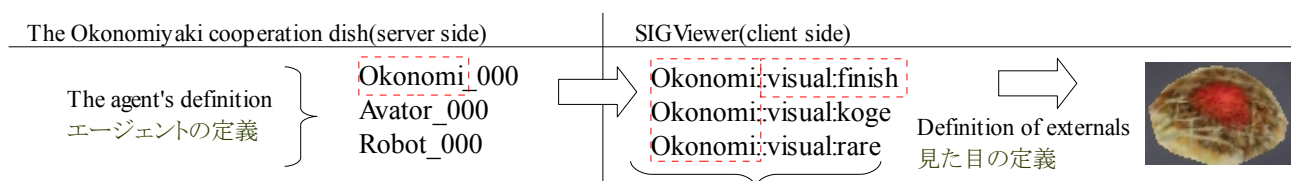
以下に「関西」のお好み焼きの見た目と「広島」のお好み焼きの見た目の関連を記します。

Dough	Ingredients	Okonomiyaki-A	Okonomiyaki-B				Consumption/ Setout	(Other) Burned
								
↓	↓	↓	↓	↓	↓	↓	↓	
								

4.2.2. 見た目の変更の設定

It is necessary to set it to the function of "Change in externals" concerning the CSV programming of the okonomiyaki cooperation dish (server side) and SIGViewer. Especially, the name of the agent set on the okonomiyaki cooperation dish is an important item to which externals are decided with SIGViewer. For instance, the agent who says "Okonomi_000" on the okonomiyaki cooperation dish is set. SIGViewer is displayed by externals in which "Okonomi" of this "Okonomi_000" is defined. These procedures and examples of figure are written as follows.

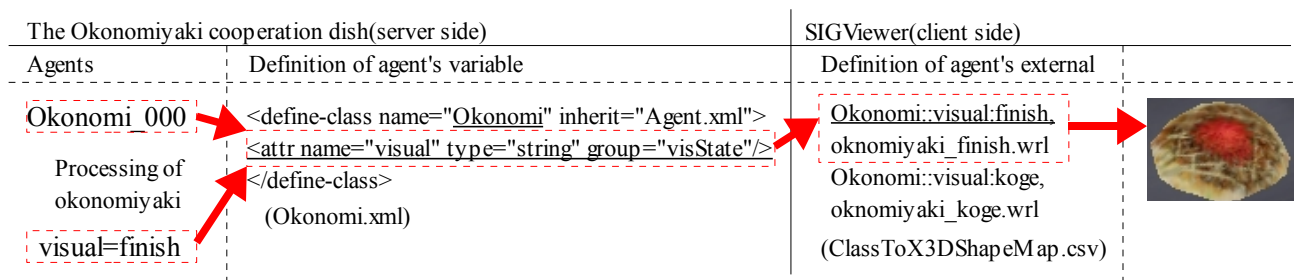
「見た目の変更」の機能は、お好み焼き協調料理(サーバサイド)のCSVプログラミングと、とSIGViewerに関する設定が必要です。特にお好み焼き協調料理上で、設定したエージェント名は、SIGViewerで見た目を決定する重要な項目です。例えば、お好み焼き協調料理上で「Okonomi_000」と言うエージェントを設定します。SIGViewerは、この「Okonomi_000」の「Okonomi」を定義されている見た目で表示します。以下にこれらの手順と図の例を記します。



4.2.2.1. 変数の設定(*.xml)

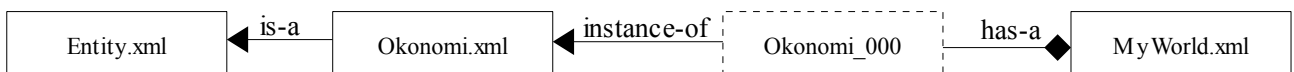
As for SIGVerse, each agent can set a favorite variable. And, the agent substitutes the value of the result of processing for the variable. For instance, the variable of "visual" is defined in the "Okonomi_000" agent. Agent's externals are substituted for this variable. The definition said that this display is done to SIGViewer when "visual" of the variable is "finish" is done at the same time. SIGViewer is "Okonomi*", and it displays it by externals of the specified variable. The function of "Change in externals" is achieved by these a series of processing. The procedure of those settings is written as follows.

SIGVerse は、各エージェントに好きな変数を設定できます。そしてエージェントはその変数に処理の結果の値を代入します。例えば、「Okonomi_000」エージェントに「visual」という変数を定義します。この変数にエージェントの見た目を代入します。同時に、SIGViewer に対して、変数の「visual」が「finish」の場合、この表示を行う、と言う定義をします。SIGViewer は「Okonomi*」であり、指定された変数の見た目で表示します。「見た目の変更」の機能は、これら一連の処理で実現します。以下にそれらの設定の手順を記します。



In SIGVerse, all attributes are defined with the same XML file as agent's name. For instance, it is "Okonomi.xml" for "Okonomi_000". The relation between "Okonomi.xml" and "Okonomi_000" is the relations of making to the instance. In a word, "Okonomi.xml" becomes a class and "Okonomi_000" becomes an object. The variable on SIGVerse is written in "Okonomi.xml" that is this class.

SIGVerse では、エージェントの名前と同じ XML ファイルで全ての属性を定義します。例えば、「Okonomi_000」の場合、「Okonomi.xml」です。「Okonomi.xml」と「Okonomi_000」の関係は、インスタンス化の関係です。つまり、「Okonomi.xml」はクラス、「Okonomi_000」はオブジェクトになります。SIGVerse 上の変数は、このクラスである「Okonomi.xml」に書きます。



The variable of okonomiyaki is defined. In addition, the same variable as "Entity.xml" and "Agent.xml" who is parents related to succession is defined. The example is written as follows.

お好み焼きの変数を定義します。更に継承関係の親である「Entity.xml」にも同じ変数を定義します。

```

Configure Ex.  <?xml version="1.0" encoding="utf8"?>
                <define-class name="Okonomi" inherit="Agent.xml">
                  <body filename="robot-body.xml">
                    <attr name="visual" type="string" group="visState"/>
                  </body>
                </define-class>
                (Okonomi.xml in "$SIGHOME/conf")
  
```

```

Configure Ex.  <!-- visible state control -->
                <attr name="visStateAttrName" type="string" group="visState"/>
                (Entity.xml in "$SIGHOME/conf")
  
```

"Okonomi_000" can use the "visual" variable by the above-mentioned definition. Please do not forget the definition of "Entity.xml".

以上の定義で、「Okonomi_000」は「visual」変数を利用できます。「Entity.xml」の定義を忘れないでください。

4.2.2.2. 見た目の設定(*_visual.csv)

"Okonomi_000" uses the definition of externals of "Okonomi_000_visual.csv". For instance, when externals are "1", "rare" is substituted for the variable of "visual". The definition is written in "Okonomi_000_visual.csv". This time, because externals do not increase, it doesn't change especially. The example is written as follows. Please confirm it.

「Okonomi_000」は「Okonomi_000_visual.csv」の見た目の定義を使用します。例えば、見た目が「1」の場合、「visual」の変数に「rare」を代入します。その定義は「Okonomi_000_visual.csv」に書かれています。今回は、見た目が増えない為、特に変更はしません。以下に例を記します。ご確認をお願いします。

Configure Ex.	visual	} The line writes the name of the variable. For instance, because "visual" was set to "Okonomi.xml" and "Entity.xml", "visual" is specified.
	1,nothing	
	2,kiji	} The content of the number and externals of externals is written since the second line. The content of externals is specified with SIGViewer "rare" "finish" "koge" etc.
	3,rare	
	4,ryoumen	} 二行目以降は、見た目の番号と見た目の内容を書きます。見た目の内容は、SIGViewer で指定する「rare」「finish」「koge」などです。
	5,sauce	
	6,aonori	
	7,okaka	
	8,finish	
	9,koge	

(Okonomi_000_visual.csv in "\$SIGHOME/conf/csv")

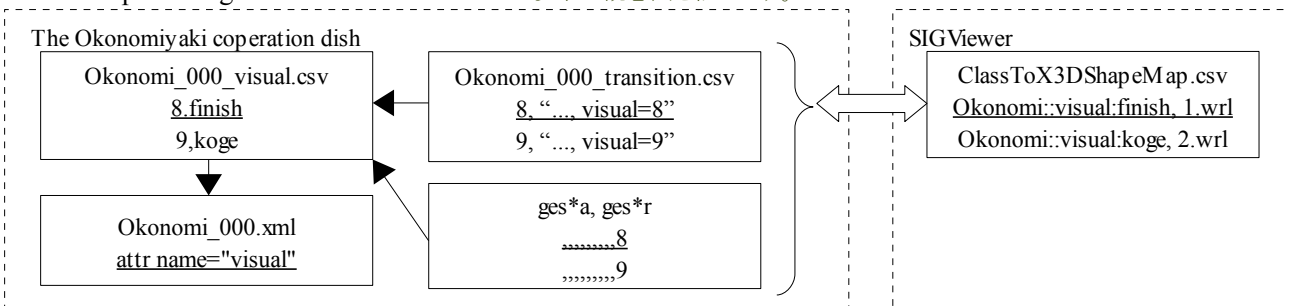
4.2.2.3. 状態遷移処理の設定(*_transition.csv)

"Externals number" set by "*_visual.csv" can be used by the state transition processing. This time, because externals have not increased, it doesn't change. The setting example is written as follows. Please confirm it.

「*_visual.csv」で設定した「見た目番号」は、状態遷移処理で利用できます。今回は、見た目が増えていない為、変更はしません。以下に設定の例を記します。ご確認をお願いします。

Configure Ex.	"00X1","State=00X1@Okonomi_000,Visual=1"	(Okonomi_000_transition.csv in "\$SIGHOME/conf/csv")
	"00X2","State=00X2@Okonomi_000"	
	"00A","State=00A@Okonomi_000"	
	"01A","State=01A@Okonomi_000,Visual=4"	
	"02A","State=02A@Okonomi_000,Visual=9"	
	"01B","State=01B@Okonomi_000"	
	"02B","State=02B@Okonomi_000,Visual=9"	
	"11B","State=11B@Okonomi_000,Visual=4"	
	"12B","State=12B@Okonomi_000"	
	"21B","State=21B@Okonomi_000,Visual=9"	
	"22B","State=22B@Okonomi_000,Visual=9"	
	"abX3","State=abX3@Okonomi_000"	
	"abX4","State=abX4@Okonomi_000"	
	"abX5","State=abX5@Okonomi_000,Visual=8"	
	"abX6","State=abX6@Okonomi_000,Visual=8"	

The conceptual diagram is written as follows. 以下に概念図を記します。



4.2.2.4. 身体動作の設定(ges*.csv)

"Externals number" set by "*_visual.csv" can be used by defining the bodily movement. This time, because externals have not increased, it doesn't change. The example is written as follows. Please confirm it.

「*_visual.csv」で設定した「見た目番号」は、身体動作の定義で利用できます。今回は、見た目は増えていない為、変更しません。以下に例を記します。ご確認をお願いします。

```
Configure Ex.  Sauce_000,,,,,-90,1
                Okonomi_000,,,,,1,5
                robo_hip,,,,15.000,0.000,0.000,0
                robo_body,,,,12.400,10.000,0.000,0
                robo_chest,,,,0.000,0.000,0.000,0
```

(ges74r in "\$SIGHOME/conf/csv/motions")

4.2.3. 見た目の為の SIGViewer の設定

SIGViewer displays the agent by using the texture of the wrl file and the image. Please look at the manual of SIGViewer about those detailed settings. In this paragraph, the "wrl" file that is information on the display is switched. As a result, externals of the okonomiyaki of "Kansai" become externals of the okonomiyaki of "Hiroshima".

SIGViewer は、wrl ファイルと画像のテクスチャを用いてエージェントを表示します。それらの詳しい設定は、SIGViewer の説明書をご覧ください。この項では、表示の情報である「wrl」ファイルを切り替えます。それにより、「関西」のお好み焼きの見た目が、「広島」のお好み焼きの見た目になります。

4.2.3.1. エージェントの見た目の設定(ClassToX3DShapeMap.csv)

When SIGViewer is installed in "C:\Program Files", the definition of externals of SIGViewer is filed in "C:\Program Files\SIGViewer\Release". The name is "ClassToX3DShapeMap.csv". The definition example and the setting method are written as follows. For instance, "Hirosima::visual:aonori, hirosimayaki_aonori.wrl" is mimicked and "Okonomi::visual:aonori, oknomiyaki_aonori.wrl" is changed with "Okonomi::visual:aonori, hirosimayaki_aonori.wrl".

SIGViewer の見た目の定義のファイルは、SIGViewer を「C:\Program Files」にインストールした場合、「C:\Program Files\SIGViewer\Release」にあります。その名前は、「ClassToX3DShapeMap.csv」です。以下にその定義の例と設定の方法を記します。例えば、「Okonomi::visual:aonori, oknomiyaki_aonori.wrl」を「Hirosima::visual:aonori, hirosimayaki_aonori.wrl」の真似をして「Okonomi::visual:aonori, hirosimayaki_aonori.wrl」と変更します。

<pre>Configure Ex. # ----- # お好み焼き (Okonomiyaki) # ----- Okonomi::, oknomiyaki_rare.wrl Okonomi::visual:finish, oknomiyaki_finish.wrl Okonomi::visual:koge, oknomiyaki_koge.wrl</pre>		<pre># ----- # お好み焼き (Okonomiyaki) # ----- Okonomi::, hirosimayaki_wrl Okonomi::visual:finish, hirosimayaki_finish.wrl Okonomi::visual:koge, hirosimayaki_koge.wrl</pre>
<pre>Hirosima::, hirosimayaki_wrl Hirosima::visual:finish, hirosimayaki_finish.wrl Hirosima::visual:koge, hirosimayaki_koge.wrl</pre>		<pre>Hirosima::, hirosimayaki_wrl Hirosima::visual:finish, hirosimayaki_finish.wrl Hirosima::visual:koge, hirosimayaki_koge.wrl</pre>

ClassToX3DShapeMap.csv in "C:\Program Files\SIGViewer\Release")

Moreover, to want to change other agents' externals or to change externals of okonomiyaki, the WRL file is made, and it sets it to "ClassToX3DShapeMap.csv". Please look at the explanatory material of SIGViewer about a detailed setting method.

また、他のエージェントの見た目を変えたい場合、お好み焼きの見た目を変えたい場合は、WRL ファイルを作成し、「ClassToX3DShapeMap.csv」に設定します。詳しい設定方法は、SIGViewer の説明資料を見てください。

4.2.3.2. マテリアルの設定(SIGVerseViewer.material)

The material (setting of the feeling of quality etc. of externals) is when externals do not change because of the above-mentioned setting and there is a possibility of wrong. The configuration file is in "SIGVerseViewer.material" of "C:\Program Files\SIGViewer\media\materials\scripts" when SIGViewer is installed in "C:\Program Files". Please confirm whether this setting is corresponding to the setting of the WRL file. The confirmation method is written as follows.

もし、上記の設定で見た目が変わらない場合、マテリアル(見た目の質感などの設定)が間違えている可能性があります。その設定ファイルは、SIGViewerを「C:\Program Files」にインストールした場合、「C:\Program Files\SIGViewer\media\materials\scripts」の「SIGVerseViewer.material」にあります。この設定とWRLファイルの設定が一致しているか確認してください。以下に確認の方法を記します。

<p>Configure Ex. # ----- # お好み焼き (Okonomiyaki) # ----- Okonomi::visual:finish_okonomiyaki_finish.wrl <u>Okonomi::visual:koge_okonomiyaki_koge.wrl</u></p>	<p>← The name of the WRL file used for the display is confirmed. 表示に利用している WRL ファイルの名前を確認します</p>
<p>ClassToX3DShapeMap.csv in “C:\Program Files\SIGViewer\Release”)</p>	


<p>Configure Ex. texture ImageTexture { <u>url "okonomiyaki_koge_c01.jpg"</u> repeatS TRUE repeatT TRUE }</p>	<p>← It confirms whether the file name of the texture image of the <u>WRL file</u> is correct. <u>WRL ファイル</u>のテクスチャ画像のファイル名が正しいか確認します</p>
<p>Okonomiyaki_koge.wrl in “C:\Program Files\SIGViewer\Release”)</p>	

<p>Configure Ex. material okonomiyaki_koge_c01 { technique { pass { texture_unit { <u>texture okonomiyaki_koge_c01.jpg</u> } } } }</p>	<p>← The file name of the texture image of the material file is confirmed. material ファイルのテクスチャ画像のファイル名を確認します</p>
<p>*The texture picture file is preserved in "C:\Program Files\SIG Viewer\Release". ※ テクスチャ画像ファイルは、「C:\Program Files\SIG Viewer\Release」に保存します</p> <p>SIGVerseViewer.material in “C:\Program Files\SIGViewer\media\materials\scripts”)</p>	

4.2.4. 設定のゴール

When externals of okonomiyaki change into the okonomiyaki of "Hiroshima" because of the above-mentioned setting, the setting is normally completed.

以上の設定で、お好み焼きの見た目が「広島」のお好み焼きに変わる場合、設定は正常に完了しています。

<p>Goal Ex.</p> 	<p>⇒ It changes it into externals of the okonomiyaki of "Hiroshima". 「広島」のお好み焼きの見た目に変えます</p>
---	--

4.3. 力学に関する設定

In this paragraph, it explains the setting of "Movement", "Rotation", and "Bodily movement" that relates to mechanics. The CSV file is chiefly edited and "Movement", "Rotation", and "Bodily movement" are set.

この項では、力学に関連する「移動」「回転」「身体動作」の設定を説明します。主に CSV ファイルを編集して「移動」「回転」「身体動作」を設定します。

4.3.1. 設定のシナリオ

The position of "Sauce", "Seaweed", "Dried bonito" and "Oil", "Bowl" is reversed. As a result, the right and left is reversed to the bodily movement of Avator and the robot. For instance, it changes as the thing's that the robot has by the left hand having by the right hand. The setting is done.

Moreover, "The bowl is made to fly up, and rotate" is set to the processing of the state transition of "Mix dough" as an example of easy "Movement" and "Rotation".

「ソース」「海苔」「鰹節」と「油」「ボウル」の位置を逆にします。これにより、アバタとロボットの身体動作は左右が逆転します。例えば、ロボットが左手で持つ物が、右手で持つ様に変化します。その設定を行います。

また、簡単な「移動」「回転」の例として、「Mix dough」の状態遷移の処理に「ボウルを飛び上がらせて、回転」を設定します。

4.3.2. SIGVerse の設定

The position of "Source", "Seaweed", and "Dried bonito" is written in MyWorld.xml. The place is "X" "Y" "Z". The example is written as follows.

「ソース」「海苔」「鰹節」の位置は MyWorld.xml に書きます。その場所は、「X」「Y」「Z」です。以下に例を記します。

```
Configure Ex.  <instanciate class="Sauce.xml">
                <set-attr-value name="name" value="Sauce_000"/>
                <set-attr-value name="dynamics" value="false"/>
                <set-attr-value name="x" value="42.838"/>
                <set-attr-value name="y" value="69.086"/>
                <set-attr-value name="z" value="95.991"/>
```

(MyWorld.xml in "\$SIGHOME/conf")

X coordinates of "Source", "Seaweed", and "Dried bonito" of these items are changed. Extent in which 100 X coordinates are done might be good. We will leave fine-tuning.

これらの項目の「ソース」「海苔」「鰹節」の X 座標を変えます。X 座標を 100 足した程度がよいでしょう。微調整は、お任せします。

Operation Ex.

```
<set-attr-value name="x" value="42.838"/>
<set-attr-value name="y" value="69.086"/>
<set-attr-value name="z" value="95.991"/>
→
<set-attr-value name="x" value="142.838"/>
<set-attr-value name="y" value="69.086"/>
<set-attr-value name="z" value="95.991"/>
```

Target agents Sauce_000
 Nori_000
 Katsuobushi_000

} These three agents' positions are changed.
これら三つのエージェントの位置を変えます

This time, the position of X of "Oil" and "Bowl" is changed. About 100 pulls the position of X. We will leave fine-tuning.

今度は、「油」「ボウル」の X の位置を変えます。X の位置を 100 程度引きます。微調整はお任せします。

```

Operation Ex.  <set-attr-value name="X" value="150.428"/>
                <set-attr-value name="y" value="69.231"/>
                <set-attr-value name="z" value="96.531"/>
                <set-attr-value name="X" value="50.428"/>
                <set-attr-value name="y" value="69.231"/>
                <set-attr-value name="z" value="96.531"/>
    
```

Target agents Abura_000 } These two agents' positions are changed.
 Bowl_000 } これら二つのエージェントの位置を変えます

The above-mentioned setting is a setting that changes agent's position immediately after the Central Server start. "The thing is moved" is set besides at "Bodily movement". Please look at the following procedures about it.

以上の設定は、セントラルサーバ起動直後のエージェントの位置を変える設定です。それ以外にも「身体動作」の時に「物を移動する」設定を行います。それは以下の手順を見てください。

4.3.3. 身体動作の設定

"Bodily movement" that moves "Oil" etc. is set by the coordinates point written in "ges**r,ges**a". The file of the following "Bodily movement" is seen with the text editor such as vi.

「油」などを動かす「身体動作」の設定は、「ges**r, ges**a」に書かれている座標点で行います。以下の「身体動作」のファイルを vi などのテキストエディタで見ます。

Procedure of dish	Content of bodily movement	Name of bodily movement	
		For Avator	For Robot
Take oil 油を手元に引き寄せる	Oil is moved on the iron plate. 油を鉄板の上に移動させます。	ges83a	ges83r
Put oil on Teppan 油を鉄板にひきます	Oil is returned to former position. 油を元の位置に戻します。	ges86a	ges86r
Put ingredients on the Teppan 生地を鉄板にのせます	The container that the ingredients enters is moved on the iron plate. 生地の入った入れ物を鉄板の上に移動します	ges93a	ges93r
	The container is returned to former position. 入れ物をものの位置に戻します	ges96a	ges96r
Take sauce ソースを手元に引き寄せる	The sauce is moved on the iron plate. ソースを鉄板の上に移動します	ges73a	ges73r
Put sauce on Okonomiyaki ソースをお好み焼きに塗る	It returns it based on the position of the sauce. ソースの位置を元に戻します	ges76a	ges76r
Take seaweed(Nori) 海苔を手元に引き寄せる	The seaweed is moved on the iron plate. 海苔を鉄板の上に移動します	ges63a	ges63r
Put seaweed on Okonomiyaki 海苔をお好み焼きに振りかける	It returns it based on the position of the seaweed. 海苔の位置を元に戻します	ges66a	ges66r
Take Katsuobushi 鰹節を手元に引き寄せる	The Katsuobushi is moved on the iron plate. 鰹節を鉄板の上に移動します	ges53a	ges53r
Put Katsuobushi on Okonomiyaki 鰹節をお好み焼きに振りかける	It returns it based on the position of the Katsuobushi. 鰹節の位置を元に戻します	ges56a	ges56r

For instance, "ges86r,ges86a" moves oil on the iron plate. The content is seen. Coordinates point "100,70,100" on the iron plate is defined. This is a demand that moves from the position defined by "MyWorld.xml" to "100,70,100".

例えば、「ges83r, ges83a」は、油を鉄板の上に移動します。その内容を見てみます。鉄板の上の座標点「100,70,100」が定義されています。これは「MyWorld.xml」で定義されている位置から「100,70,100」まで移動する要求です。

<p>Configure Ex.</p> <pre>Abura_000,100,70,100,,,,1 robo_hip,,,,10.000,0.000,0.000,0 robo_body,,,,12.000,-5.000,0.000,0 :</pre>	}	<p>It is a movement from the position of MyWorld.xml to 100,70,100. This might not have the alterations necessary. MyWorld.xml の位置から 100,70,100 までの移動です。これは、変更の必要はないでしょう。 (ges*** in "\$SIGHOME/csv/motions")</p>
---	---	---

This doesn't have the alterations necessary because it need not change coordinates of 100,70,100 in this change. (The position of the iron plate doesn't change.)A necessary setting is former position of the agent. This time, let's see "ges86r,ge86a". The content is written as follows.

100,70,100 の座標は、今回の変更では変える必要がない為、これは変更の必要はありません。(鉄板の上の位置は変わりません。)必要な設定は、エージェントの元の位置です。今度は、「ges86r, ge86a」を見てみましょう。以下にその内容を記します。

<p>Configure Ex.</p> <pre>Abura_000,167,69,95,,,,1 robo_hip,,,,10.350,0.350,0.350,0 robo_body,,,,12.350,30.350,0.300,0 :</pre>	}	<p>It is a movement from present location (100,70,100) to 167, 69,95. This changes "167,69,95". 現在位置から 169,69,95 までの移動です。これは「167,69,95」を変更します。</p>
--	---	--

This position is a place where the place was changed by this setting. Therefore, this coordinates are changed. This setting is Abura. 100 pulls X coordinates of Abura as set with MyWorld.xml. (We will leave fine-tuning.)

Let's change coordinates of the agent who should change other coordinates according to the same points as "MyWorld.xml". Those agents' lists are written as follows.

この位置(169,69,95)は、今回の設定で場所が変えられた場所です。その為、この座標を変えます。この設定は Abura です。MyWorld.xml で設定した様に、Abura の X 座標を 100 引きましょう。(微調整はお任せします)


それ以外の座標を変えらる必要のあるエージェントの座標を、「MyWorld.xml」と同じ要領で変更しましょう。以下にそれらのエージェントの一覧を記します。

<p>Target agents</p> <pre>Sauce_000 Nori_000 Katsuobushi_000 Abura_000 Bowl_000</pre>	}	<p>These agents' positions are changed. これらのエージェントの位置を変えます</p>
---	---	--

4.3.4. 設定の中間ゴール

When the setting is completed, "Central Server", "Okonomiyaki GUI", and "SIGViewer" are reactivated. Afterwards, the position of "Source", "Seaweed", "Dried bonito", and "Oil" and "Bowl" is reversely displayed. And, when it was possible to return to former position correctly in oil and the source at the okonomiyaki cooperation dish, the setting was normally done.

設定が完了した場合、「セントラルサーバ」と「お好み焼き GUI」と「SIGViewer」を再起動します。その後、「ソース」「海苔」「鰹節」と「油」「ボウル」の位置が反対に表示されます。そしてお好み焼き協調料理の時に、油やソースが正しく元の位置にもどれる場合、設定は正常に行われました。

<p>Goal Ex.</p> 	⇒	<p>The position is reversed. 位置を逆にします</p>
---	---	---

4.3.5. 状態遷移処理の設定




In this paragraph, "Movement" and "Rotation" are set by processing the state transition. "Movement" and "Setting" use the following commands. This is the same as the command input to SIGViewer. The format of the command is written as follows.

この項では、状態遷移の処理で「移動」と「回転」を設定します。「移動」と「設定」は以下のコマンドを利用します。これは SIGViewer に入力したコマンドと同じです。以下にコマンドの書式を記します。

Name of proccessing 処理の名前	Content of processing 処理の内容	Description example 記述の例
Move	It moves to coordinates that specify other agents. Coordinates are described by the X:Y:Z form. 他のエージェントを指定した座標に移動します。座標は X:Y:Z 形式で記述します。	Move=100:80:100@Sauce_000 (The source is moved on the iron plate.) (ソースを鉄板の上に移動する)
Angle	It rotates to the direction that specifies other agents. It describes it by "Direction (0,1) of X: direction (0,1) of Y: direction (0,1) of Z: the beginning angle: the end angle: increment angle" form. This is the same as the form operated with Operation. 他のエージェントを指定した向きに回転します。「X 方向 (0,1):Y 方向(0,1):Z 方向(0,1):開始角度:終了角度:増分角度」形式で記述します。これは Operation で操作した形式と同じです。	Angle=1:0:0:0:180:10@Okonomi_000 (Okonomiyaki is turned inside out.) (お好み焼きを裏返す)

This time, the bowl is made to fly up at "Mix Dough" of Avator, and it is rotated. The agent of the object is "Bowl_000". The position of the bowl is "150,69,96". Flying up is a movement to "150,80,96". It moves to Y axis by about ten coordinates. The rotation is assumed to be 360 degrees from 0 ,in a word, a round with X axis. In addition, it moves to former position. The coordinates are "150,69,96". These instructions are written as follows.

今回は、アバタの「Mix Dough」の時に、ボウルを飛び上がらせて、回転させます。対象のエージェントは「Bowl_000」です。ボウルの位置は「150,69,96」です。飛び上がりは「150,80,96」に移動です。Y 軸に 10 座標ほど移動します。回転は X 軸で 0 度から 360 度、つまり一周とします。更に元の位置に移動します。その座標は「150,69,96」です。これらの命令は以下の様子に書きます。

Command Ex. Move=150,80,96@Bowl_000		Bowl flies. ボウルを飛ばします
Command Ex. Angle=1:0:0:0:360:10@Bowl_000		Bowl is rotated. ボウルを回転します
Command Ex. Move=150,69,96@Bowl_000		Bowl is returned. ボウルを戻します

These instructions are added to "Mix dough" (=1) of state transition processing (Avator_000_transiton.csv) of Avator. The example is written as follows.

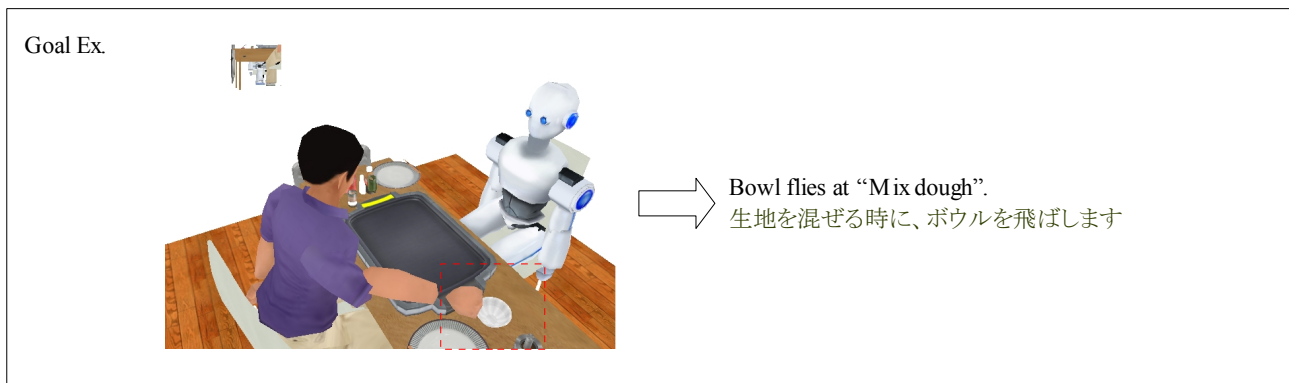
これらの命令をアバタの状態遷移処理(Avator_000_transiton.csv)の「Mix dough」(=1)に追加します。以下に例を記します。

Configure Ex.	"" ""	
	0,"Motion=ges00a,State=00X1@Okonomi_000,Move=150:80:96, Angle=1:0:0:0:360:10,Move=150,69,96"	} The instruction is added. 命令を追加します
	1,"Motion=ges92a,Motion=ges00a,State=00X2@Okonomi_000"	
	2,"Motion=ges82a,Motion=ges83a,State=1@Abura_000"	(Avator_000_transiton.csv in "\$SIGHOME/csv")

4.3.6. 設定のゴール

When the setting is completed, "Central Server", "Okonomiyaki GUI", and "SIGViewer" are reactivated. And, if the bowl flies by "Mix dough" of Avator at the okonomiyaki cooperation dish, setting is normal and completion.

設定が完了した場合、「セントラルサーバ」と「お好み焼き GUI」と「SIGViewer」を再起動します。そしてお好み焼き協調料理の時に、アバタの「Mix dough」でボウルが飛ばば、設定は正常に完了です。



4.1. 対話に関する設定

In this paragraph, it explains the setting that relates to the conversation. The conversation is achieved by chiefly using "State transition processing". As a result, "Remark" and "Reward", etc. are achieved.

この項では、対話に関連する設定を説明します。対話は主に「状態遷移処理」を用いて実現します。それにより、「発言」「お礼」などを実現します。

4.1.1. 設定のシナリオ

Robot only advises when a supplementary level is "low" now. When Avator acts as shown in advice, the setting that means "Reward" is done.

現在、補助度が「low」の場合、ロボットはアドバイスのみします。もし、アバタがアドバイスの通りに行動した場合、「お礼」を言う設定を行います。

4.1.2. 状態遷移処理の設定

The command that means the reward uses "Instruction on which it makes remarks with a certain agent exists" of the instruction of the state transition processing. The example is written as follows.

お礼を言うコマンドは状態遷移処理の命令の「あるエージェントがある状態の時に発言する命令」を使います。以下に例を記します。

Command Ex.	UtrrAndReject=Thank_you_very_much&1@Avator_000
-------------	--

The content of this command is written as follows.

以下にこのコマンドの内容を記します。

Name of proccessing 処理の名前	Content of processing 処理の内容	Description example 記述の例
Utterance	When it meets a specified requirement, it makes remarks. Transition afterwards is not done. 指定の条件を満たす場合、発言します。その後の Transition は行いません。(アバタが1(油をとった)の場合、お礼を言います。自分はその後の Transition(油をとる)を行いません。)	<u>UtrAndReject=Thanks&1@Avator</u> (When Avator's state is 1 "Oil was taken", the reward is said. I do not do Transition afterwards "Oil is taken".)

A supplementary level of the robot sees the state transition processing of "low" situation. It refers with the text editor such as vi. The content is written as follows.

ロボットの補助度が「低」場合の状態遷移処理を見ます。viなどのテキストエディタで参照します。以下に内容を記します。

Configure Ex.	<pre> "" "" 0,"" "" 1,"Utterance=Please_mix_dough&20"" "" 2,"Utterance=Please_take_oil&20"" "" 3,"Utterance=Please_put_oil_on_Teppan&20"" "" </pre>	<p>The instruction of advice is written. アドバイスの命令が書かれています</p> <p>(Avator_000_transition.csv in "\$SIGHOME/csv")</p>
---------------	---	---

The command of UtrAndReject is added to this state transition processing. The edit example is written as follows.

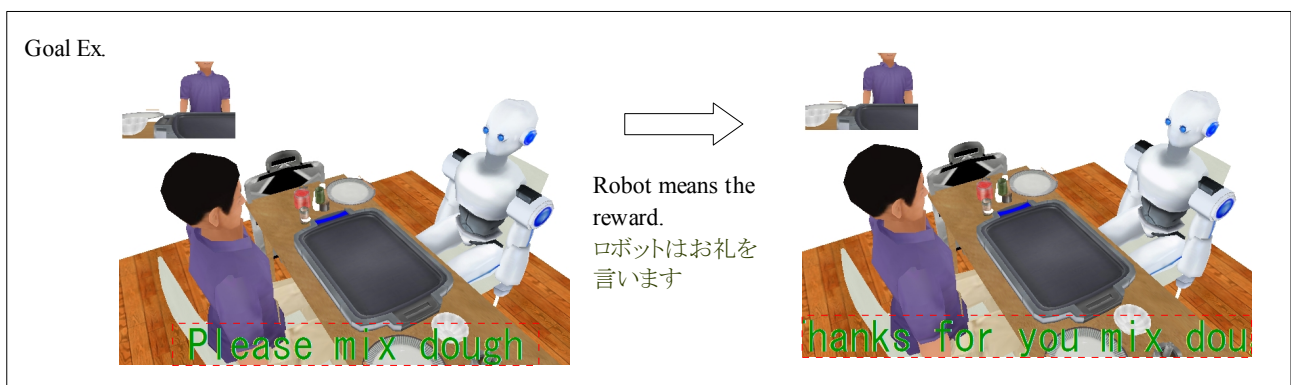
この状態遷移処理に UtrAndReject のコマンドを追加します。以下に編集した例を記します。

Operation Ex.	<pre> "" "" 0,"" "" 1,"Utterance=Please_mix_dough&20,UtrAndReject=Thanks_for_you_mix_dough&1@Avator_000"" "" 2,"Utterance=Please_take_oil&20"" "" 3,"Utterance=Please_put_oil_on_Teppan&20"" "" </pre>	<p>The instruction is added. 命令を追加します</p>
---------------	--	---

4.1.3. 設定のゴール

It initializes it pushing the "Initialize" button of "Okonomiyaki GUI" when the setting is completed. Afterwards, the "Start" button is pushed. When Avator does earliness "Mix dough" from the robot, the robot says the reward to Avator. If it is done, it is completion.

設定が完了した場合、「お好み焼き GUI」の「Initialize」ボタンを押して初期化します。その後「Start」ボタンを押します。アバタがロボットより先に「Mix dough」をした場合、ロボットはアバタにお礼を言います。それが行われれば、完了です。



5. お好み焼き協調料理の改造

Contents

- ・お好み焼き協調料理の改造
 - －お好み焼き協調料理の機能
 - －SIGVerse の提供する API
 - －お好み焼き協調料理の構造
 - －お好み焼き協調料理の知覚の改造
 - －お好み焼き協調料理の力学の改造
 - －お好み焼き協調料理の対話の改造

The okonomiyaki cooperation dish has achieved various functions by using API of general SIGVerse that SIGVerse offers. The computer program language used in that case is C++ language. The program module of the agent who changes agent's externals is made by using API that relates to the perception of SIGVerse in the C++ language. In this paragraph, to mount the application of new SIGVerse that especially corresponds to the okonomiyaki cooperation dish, it explains the program structure of the okonomiyaki cooperation dish and the use of SIGVerse-API.

お好み焼き協調料理は、SIGVerse が提供する汎用的な SIGVerse の API を用いて各種機能を実現しています。その際に用いられるコンピュータプログラム言語は、C++言語です。C++言語で SIGVerse の知覚に関連する API を使用しエージェントの見た目を変更するエージェントのプログラムモジュールを作成します。この項では、特にお好み焼き協調料理に相当する新しい SIGVerse のアプリケーションが実装出来る様に、お好み焼き協調料理のプログラムの構造と SIGVerse-API の使用方法を説明します。

5.1. お好み焼き協調料理の機能

The okonomiyaki cooperation dish has six functions of "Change in externals", "Rotation", "Movement", "Bodily movement", "Conversation between agent Avator", and "Mutual conversation between agents" based on three functions of SIGVerse. The list of the function is easily recorded as follows.

お好み焼き協調料理は、SIGVerse の三つの機能を元に「見た目の変更」/「回転」「移動」「身体動作」/「エージェント・アバタ間対話」「エージェント間相互対話」の六つの機能を持ちます。以下に簡単に機能の一覧を記します。

Function name 機能名	Ability of relating SIGVerse 関連する SIGVerse の能力	Outline of function 機能の概要
Change in externals 見た目の変更	Perception 知覚	Agent's externals are switched. エージェントの見た目を切り替えます
Rotation 回転	Mechanics 力学	The agent on a three-dimensional space is turned around (It is rotated). 三次元空間上のエージェントの向きを変える(回転させる)
Movement 移動		The position of the agent on a three-dimensional space is changed (It is moved). 三次元空間上のエージェントの位置を変える(移動させる)
Bodily movement 身体動作		Complex operation that makes indirect of the human body a radical is done. 人体の間接を元にした複雑な動作を行う
Conversation between Agent and Avator エージェント・アバタ間対話	Conversation 対話	Other agents are told the user's demand through Avator. 利用者の要求をアバタを介して他のエージェントに伝える
Mutual conversation between agents エージェント間相互対話		The agent including me is told the demand (remark) between agents. エージェント間の要求(発言)を自分を含むエージェントに伝える
State transition 状態遷移	Okonomiyaki cooperation dish and peculiarity お好み焼き協調料理・特有	"Information", "Condition", and "Processing" of the state transition are controlled. 状態遷移の「情報」「条件」「処理」を制御します

5.2. SIGVerse の提供する API

SIGVerse achieves various functions through API that can be used by the C++ language or C language. The name, the function, and the interface of the API are recorded as follows.

SIGVerse は C++ 言語、または C 言語で利用可能な API を通じて各種機能を実現します。以下にその API の名前と機能とインターフェイスを記します。

Name of API API の名前	Ability of SIGVerse SIGVerse の能力	Interface of API API のインターフェイス	Example of API API の例
Change in externals 見た目の変更	Perception 知覚	setAttrValue(const char, /* target name(対象名)*/ char) /* setting value(設定値)*/	setAttrValue("visual", "koge"); /* Externals are changed into scorching. */ /* 見た目を焦げに変える*/
Rotation 回転	Mechanics 力学	setAxisAndAngle(double, /* x-axis(x 軸)*/ double, /* y-axis(y 軸)*/ double, /* z-axis(z 軸)*/ double) /* degree(角度)*/	SetAxisAndAngle(1, 0, 0, 45 * (PI / 180)); /* It rotates to x axis by +45 degrees. */ /* x 軸に+45 度回転する*/
Movement 移動		setPosition(double, /* x-position(x 座標)*/ double, /* y-position(y 座標)*/ double) /* z-position(z 座標)*/	setPosition(100, 80, 100); /* It moves to coordinates (x=100, y=80, z=100)*/ /* 座標(x=100, y=80, z=100)に移動する*/
Bodily movement 身体動作		setJointAngle(const char, /* target name(対象名)*/ double, /* x-axis(x 軸)*/ double, /* y-axis(y 軸)*/ double, /* z-axis(z 軸)*/ double) /* degree(角度)*/	setJointAngle("LARM_JOINT1", 1, 0, 0, 45); /* A left shoulder is rotated by 45 degrees. */ /* 左肩を 45 度回転する*/
Conversation between Agent and Avator エージェント・アバタ 間対話	Conversation 対話	sendMessage(const char, /* Agent name(対象名)*/ int, /* line num(文言行数)*/ char**) /* setting text(送信文)*/	char *text = "Hello, SIGVerse"; sendMessage("Avator", 1, (char*)&text); /* It transmits to the robot with "Hello, SIGVerse". */ /* ロボットに「Hello, SIGVerse」と送信する*/
Mutual conversation between agents エージェント間相互 対話			char *command = "Move=0:0:0"; sendMessage("Abura", 1, (char*)&text); /* The demand that moves to the starting point is sent to oil. */ /* 油に原点に移動する要求を送る*/
State transition 状態遷移	—	—	—

As for the type that offers these API, the instance becomes Avator and a robot in the agent class. SIGVerse offers the agent's instance by the all-purpose function. The API and the example are recorded as follows.

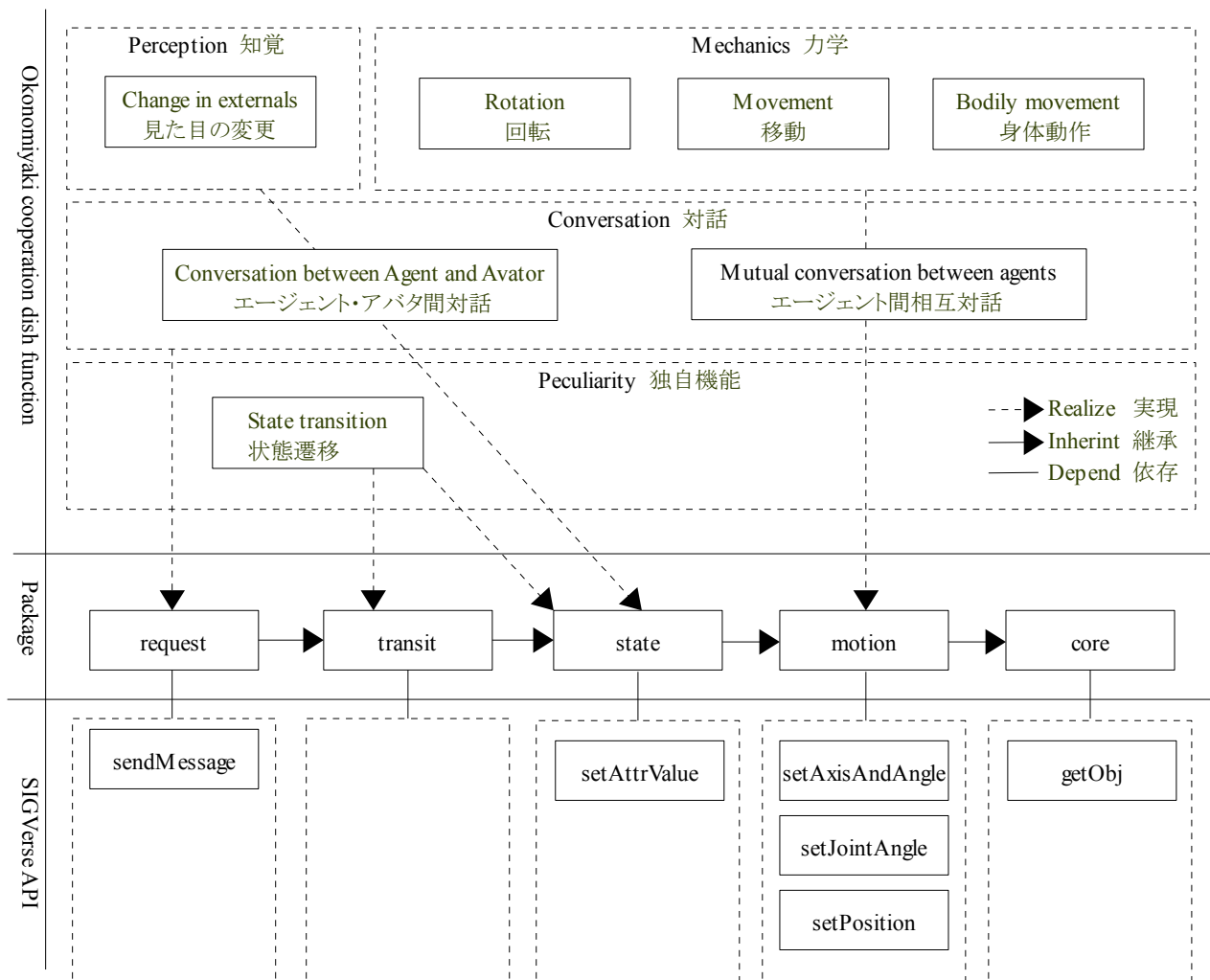
これらの API を提供する型は、エージェントクラスでそのインスタンスがアバタやロボットになります。SIGVerse は汎用関数でそのエージェントのインスタンスを提供します。以下にその API と例を記します。

Name of API API の名前	Ability of SIGVerse SIGVerse の能力	Interface of API API のインターフェイス	Example of API API の例
Agent's acquisition- エージェント取得	General purpose 汎用	getObj(const char) /* agent name(対象名)*/	SimObj* obj = getObj("Robot"); /* Robotic agent is acquired. */ /* ロボットエージェントを取得する*/

5.2. お好み焼き協調料理の構造

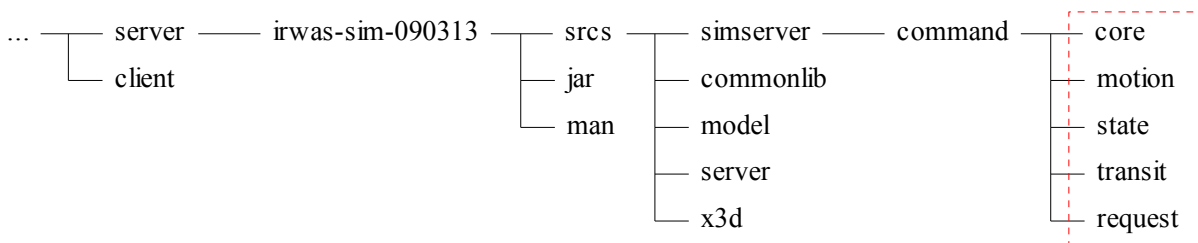
The okonomiyaki cooperation dish has achieved the above-mentioned six functions by using SIGVerse-API. The relation between the function of those okonomiyaki cooperation dishes and SIGVerse-API and program structures are recorded as follows.

お好み焼き協調料理は、SIGVerse-APIを用いて、上記の六つの機能を実現しています。それらお好み焼き協調料理の機能と、SIGVerse-APIとの関連と、プログラムの構造を以下に記します。



The package in this concept corresponds to the physics package of the C++ language as it is. The composition in the physics is recorded as follows.

この概念上のパッケージは、そのままC++言語の物理パッケージに相当します。以下にその物理上の構成を記します。



Moreover, the physical directory composition, the program file, and the list of the role of the program are recorded in the following.

また、その物理ディレクトリ構成とプログラムファイル、プログラムの役割の一覧を下記に記します。

Name of program プログラムの名前	Storage directory 格納ディレクトリ	Role of Program プログラムの役割
CommandBase	\$SIGHOME/command/core	The common processing done with the okonomiyaki cooperation dish is offered. For instance, getObj assistance is processed, and supplementary processing of the setAttrValue. お好み焼き協調理で行う共通処理を提供します。例えば、getObj の補助処理、setAttrValue の取得の補助処理です。
Supplier		It is offered to process it according to common information done with the okonomiyaki cooperation dish. For instance, it is reading of the CSV file. お好み焼き協調理で行う共通の情報に依存した処理を提供します。例えば、CSV ファイルの読み込みです。
Define		The global variable used with the okonomiyaki cooperation dish is defined. For instance, it is a phased frequency of the current directory and the indirect corner operation of the CSV file to operate, and the real times, etc. at one unit time. お好み焼き協調理で使われる大域変数を定義します。例えば、CSV ファイルのカレントディレクトリ、間接角動作の段階的な動作の回数、1 ユニット時間の実時間などです。
CommandMotion	\$SIGHOME/command/motion	Common processing concerning mechanics is offered. For instance, it is a history of the bodily movement that oneself should do as follows record etc. 力学に関する共通処理を提供します。例えば、自身が次に行うべき身体動作の履歴記録などです。
CommandMove		Processing concerning the movement is offered. For instance, the movement gradually moves gradually to target coordinates in the okonomiyaki cooperation dish. Such a control is done by using SIGVerse-API. 「移動」に関する処理を提供します。例えば、お好み焼き協調理では、移動は目標「座標」まで段階的に徐々に移動します。その様な制御を SIGVerse-API を用いて行います。
CommandRotation		Processing concerning the rotation is offered. For instance, the rotation moves gradually up to the angle of the target gradually in the okonomiyaki cooperation dish. Such a control is done by using SIGVerse-API. 「回転」に関する処理を提供します。例えば、お好み焼き協調理では、回転は目標「角度」まで段階的に徐々に移動します。その様な制御を SIGVerse-API を用いて行います。
CommandRolling		Supplementary processing of the processing that rotates while moving is done. The main processing becomes CommandWalk. 移動しながら回転する処理の補助処理を行います。主処理は CommandWalk になります。
CommandGesture		The bodily movement is offered. A specified angle is assumed to be an angle of the target to a specified part, and it moves it gradually gradually. 身体動作を提供します。指定部位に指定角度を目標角度とし、段階的に徐々に動かします。
CommadWalk		Turnabout at the median center when there is a median center between operation where it walks and an actual movement and target coordinates. This processing is not used with the okonomiyaki emphasis dish now. 歩く動作と実際の移動、及び目標座標までの間に中間地点がある場合、その中間地点で方向転換します。現在、この処理はお好み焼き強調料理で使われていません。

Name of program プログラムの名前	Storage directory 格納ディレクトリ	Role of Program プログラムの役割
SupplierGesture		Various configuration files used by the bodily movement are read. Information on the bodily movement is offered. 身体動作で使用する各種の設定ファイルを読み込みます。身体動作に関する情報を提供します。
CommandState	\$\$SIGHOME/command/state	The processing related to the state transition is offered. An actual state transition and the state transition that uses the transition condition are judged. 状態遷移に関わる処理を提供します。実際の状態遷移や遷移条件を用いた状態遷移の判定を行います。
CommandFilter		<u>The processing of "State transition condition" when the state changes is offered.</u> For instance, whether state transition condition "State" fills the limiting condition is judged. Moreover, the objection that can be specified for all the state transition conditions is defined here. 状態遷移時の「状態遷移条件」の処理を提供します。例えば、状態遷移条件「State」がその制約条件を満たしているかの判定を行います。また、全ての状態遷移条件に指定可能な文言はここで定義されています。
SupplierState		All CSV files concerning the state transition are acquired. Those files are "State transition information", "State transition condition", and "State transition processing." 状態遷移に関する全ての CSV ファイルの取得を行います。それらのファイルは「状態遷移情報」「状態遷移条件」「状態遷移処理」です。
CommandTransit	\$\$SIGHOME/command/transit	Processing that specializes in " <u>State transition processing</u> " in concerning the state transition is offered. 状態遷移に関する中で「状態遷移処理」に特化した処理を提供します。
TransitAngle		Processing concerning state transition processing "Angle" ,in a word, the rotation is offered. 状態遷移処理「Angle」、つまり回転に関する処理を提供します。
TransitMotion		Processing concerning state transition processing "Motion" ,in a word, the bodily movement is offered. 状態遷移処理「Motion」、つまり身体動作に関する処理を提供します。
TransitReject		Processing that doesn't execute the state transition processing after that is offered in case of state transition processing "Reject" , for instance, having already changed in the state. 状態遷移処理「Reject」、例えば既にその状態に遷移していた場合、それ以降の状態遷移処理を実行しない処理を提供します。
TransitState		Processing concerning state transition processing "State" ,in a word, the state is offered. 状態遷移処理「State」、つまり状態に関する処理を提供します。
TransitUtrrAndAction		State transition processing "UtrrAndAction" of a fixed time ,in a word, after it passes, the processing processed after that is offered. 状態遷移処理「UtrrAndAction」、つまり一定時間の経過後、それ以降の処理を行う処理を提供します。
TransitVisual		Processing that changes state transition processing "Visual" ,in a word, externals is offered. 状態遷移処理「Visual」、つまり見た目を変える処理を提供します。
TransitAttr		Processing that changes state transition processing "Attr" ,in a word, the attribute is offered. 状態遷移処理「Attr」、つまり属性を変える処理を提供します。
TransitMove		Processing concerning state transition processing "Move" ,in a word, the movement is offered. 状態遷移処理「Move」、つまり移動に関する処理を提供します。

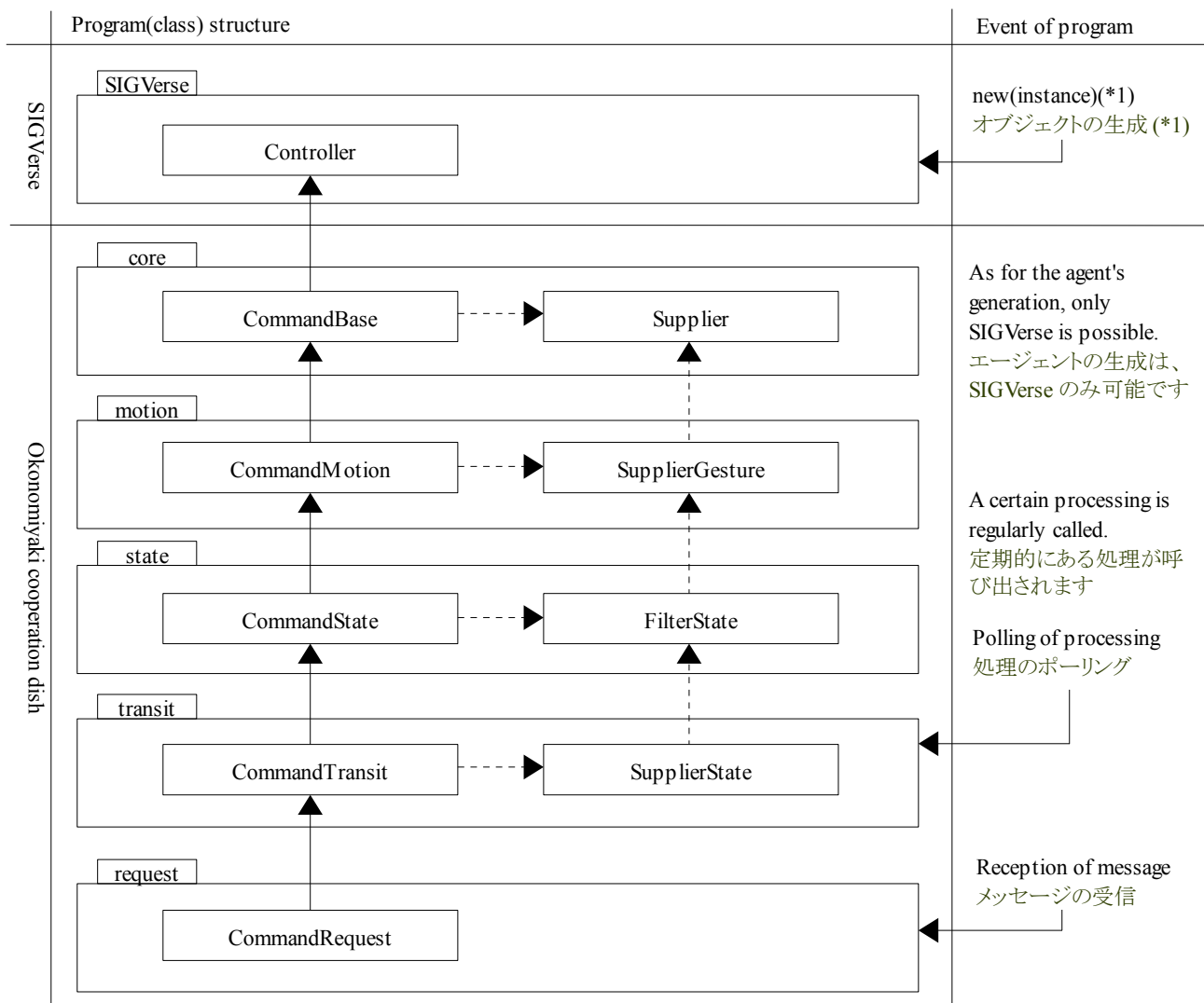
Name of program プログラムの名前	Storage directory 格納ディレクトリ	Role of Program プログラムの役割
TransitSend		Processing concerning the transmission is offered to state transition processing "Send", in a word, the agent besides the objection. (* This processing is not recommended now.) 状態遷移処理「Send」、つまり文言の他エージェントに対して送信に関する処理を提供します。(*現在、この処理は推奨されていません)
TransitUtterance		The processing of state transition processing "Utterance", in a word, the remark is offered. 状態遷移処理「Utterance」、つまり発言の処理を提供します。
TransitUtrAndReject		It makes remarks on specification when it meets state transition processing "UtrAndReject", in a word, the requirement, and the processing not processed after that is offered. 状態遷移処理「UtrAndReject」、つまり条件を満たす場合、指定の発言を行い、それ以降の処理を行わない処理を提供します。
CommandRequest	\$\$SIGHOME/command/request	Processing is offered concerning "Demand". For instance, processing that should receive the content transmitted with sendMessage, and execute it along with it is executed. This offers common processing concerning the demand. 「要求」に関する処理を提供します。例えば、sendMessageで送信した内容の受信、それに伴い実行すべき処理を実行します。これは要求に関する共通的な処理を提供します。
RequestAttr		"Attr" The attribute of the attribute specified in a word in which the processing of the demand is offered is set. 「Attr」要求の処理を提供します、つまり指定された属性に自身の属性を設定します。
RequestGesture		"Gesture" The specified bodily movement that offers the processing of the demand is executed. 「Gesture」要求の処理を提供します、指定された身体動作を実行します。
RequestJudgement		This processing is internal processing. For instance, the agent judges whether the received demand can be done because the bodily movement cannot be suddenly done while rotating now. The processing of the judgment is offered. この処理は、内部的な処理です。例えば、エージェントは現在、回転中に突然身体動作を行えない為、受信した要求が行えるか判定します。その判定の処理を提供します。
RequestParam		It processes it bringing the demand of "Aid" "Utr" "Visual" together. For instance, the specified supplementary level is set for Aid. Moreover, a part of demand not being used now is included. 「Aid」「Utr」「Visual」の要求をまとめて処理します。例えば、Aidの場合は指定された補助度に自身を設定します。また、一部現在利用していない要求も含まれています。
RequestReqState		The demand that it wants you to transmit the existing state of things is accepted, and the processing returned to the agent in the transmission origin is offered. This processing doesn't operate normally now. 現在の状態を送信して欲しい要求を受け付けて、送信元のエージェントに返す処理を提供します。現在、この処理は正常に動作していません。
RequestStat		A supplementary level and the remark level including agent's present state are displayed. (* This function doesn't relate directly to the okonomiyaki cooperation dish. It is a function for development.) エージェントの現在の状態を始めとする、補助度、発言度を表示します。(※この機能は直接お好み焼き協調料理に関係しません。開発用の機能です。)
RequestCatch		The agent (object) to whom the distance is the shortest in 100 coordinates or less of the agent forward is gripped. This function is not used in the

Name of program プログラムの名前	Storage directory 格納ディレクトリ	Role of Program プログラムの役割
		<p>okonomiyaki emphasis dish. When the thing is gripped, the bodily movement function is used. And, this function is non-recommendation now.</p> <p>エージェントの前方 100 座標以内で最も距離が近いエージェント(物体)を掴みます。この機能は、お好み焼き強調料理では利用していません。物を掴む場合は、身体動作機能を利用します。そしてこの機能は現在、非推奨です。</p>
RequestInitialize		<p>This is internal processing. It doesn't use it directly with the okonomiyaki cooperation dish.</p> <p>For instance, when the necessity for the state's changing at the time of moving the thing or moving and doing a new movement is generated, it is necessary to initialize information on the movement and the rotation. The judgment and initialization are executed.</p> <p>これは内部処理です。直接お好み焼き協調料理で利用しません。例えば、物を動かしているまたは移動している最中に、状態が変化して新たな移動を行う必要が発生する場合、移動や回転に関する情報を初期化しなければなりません。その判定と初期化を実行します。</p>
RequestMessage		<p>It is processing for the examination to acquire the objection Message = from now on. It doesn't use it directly in the okonomiyaki cooperation dish.</p> <p>「Message=」以降の文言を取得する試験用の処理です。お好み焼き協調料理では直接利用しません。</p>
RequestPosition		<p>It is a demand that moves directly to specified coordinates without doing a phased movement. For instance, it shows it like using and carrying externals and this demand that the gap is not caused in coordinates though it is had with the holder when the thing is carried. (* There is a possibility being mounted for a formal function to say that the one will be gripped in the future.)</p> <p>段階的な移動を行わずに、直接指定の座標に移動する要求です。例えば、物を持ち運ぶ場合、所持者と持たれるものの座標にずれが生じない様、この要求を利用して持ち運んでいる様に見せます。(※将来、ものを掴むと言う正式な機能が実装される可能性があります)</p>
RequestRequiptment		<p>The demand of present location and the direction is accepted. However, this function is not used now. And, it is non-recommendation.</p> <p>現在位置や方向の要求を受け付けます。ですが、現在この機能は使われていません。そして非推奨です。</p>
RequestState		<p>The state in the demanded state is set. When the requirement of changing specifying it is not met, this demand is disregarded.</p> <p>要求された状態に自身の状態を設定します。もし、指定の状態に遷移する条件が満たされていない場合、この要求は無視されます。</p>
RequestAngle		<p>It rotates angling of the demand. This is the same as the rotation done by the operation and the state transition processing.</p> <p>要求された角度に回転します。これは操作や状態遷移処理で行う回転と同じです。</p>
RequestClear		<p>The agent is returned initial. For instance, it is a state, and an attribute.</p> <p>エージェントを初期の状態に戻します。例えば、状態、属性です。</p>
RequestJoint		<p>Indirect of specify is angling specification set directly. This function is not used in the okonomiyaki cooperation dish. It is processing for development.</p> <p>指定の間接を指定の角度に直接設定します。この機能はお好み焼き協調料理では利用していません。開発用の処理です。</p>
RequestMove		<p>It intergrades at the position of specified xyz. This is the same as Move used by the operation and the state transition processing.</p>

Name of program プログラムの名前	Storage directory 格納ディレクトリ	Role of Program プログラムの役割
		指定の x,y,z 座標に段階的に移動します。これは、操作と状態遷移処理で利用した Moveと同じです。
RequestRelease		It is a demand to separate the gripped one. It is not used in a present okonomiyaki cooperation dish. 掴んでいるものを離す要求です。現在のお好み焼き協調料理では利用されていません。
RequestResState		The answer is returned to demanded "Offer of the existing state of things" etc.This function is not operating normally now. 要求された「現在の状態の提供」などに返答を返します。この機能は現在、正常に動作していません。

The structure and the relation of these programs (It is assumed the unit of the class here) are recorded as follows.

以下にこれらのプログラム(ここではクラス単位とします)の構造と関連を記します。

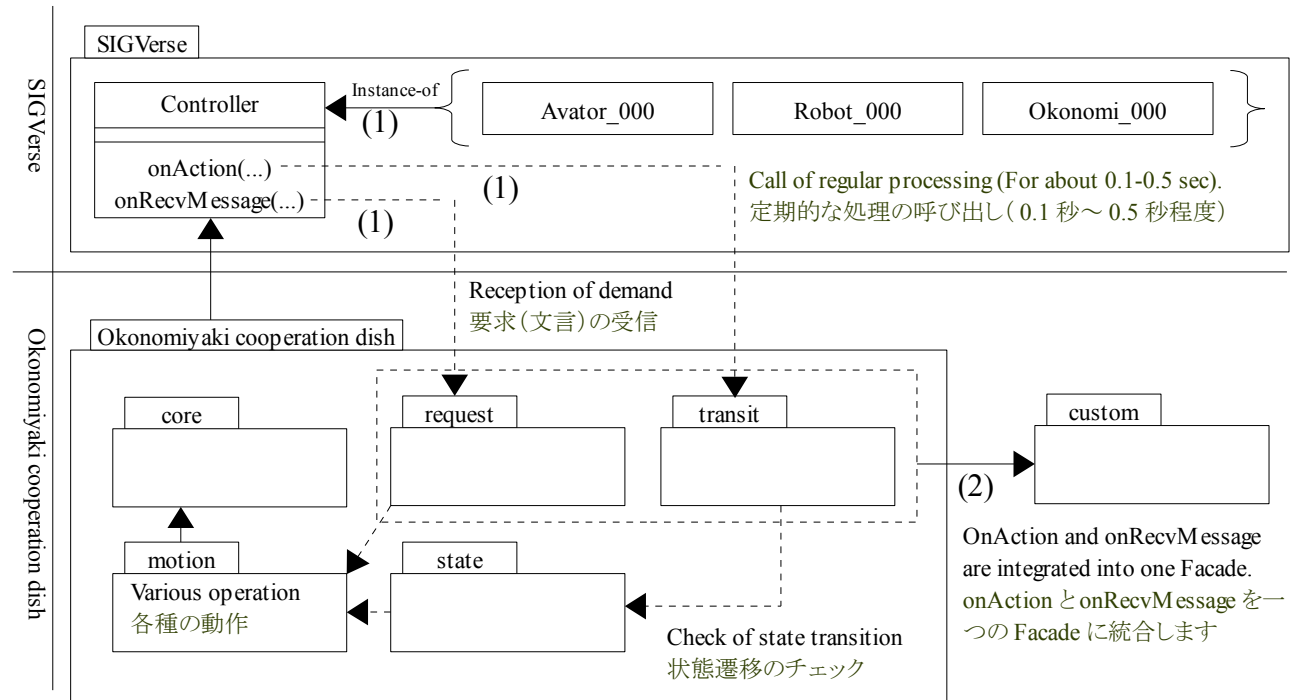


(*1)Strictly, the object is generated with the okonomiyaki cooperation dish. This looks like the Template pattern that leaves the generation of the object that the parent class generates to the child class.

(*1)厳密には、オブジェクトの生成は、お好み焼き協調料理で行います。これは、親クラスは子クラスに生成するオブジェクトの生成を任せる、Template パターンに似ています。

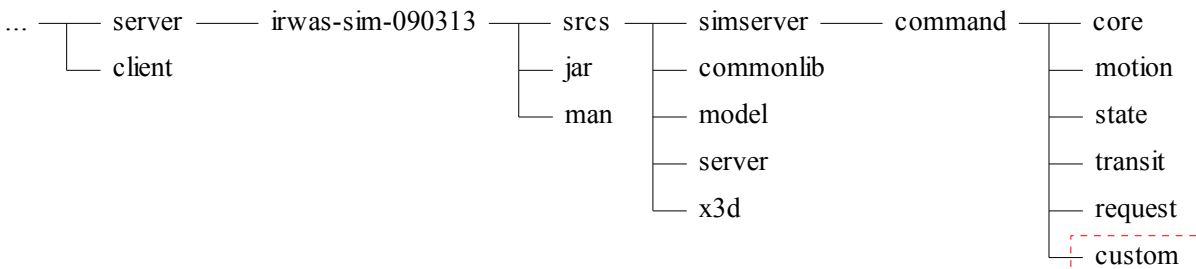
(1)SIGVerse manages agent's generation and life cycle. Moreover, the reception objection is offered with a regular call of processing. As for the okonomiyaki cooperation dish, the agent is succeeded to, and agent's (object) offer, a regular call, and the reception objection are received. The state transition is processed at each regular call, and the demand is processed similarly at each reception. The okonomiyaki cooperation dish has been achieved by these a series of processing.

(1)SIGVerse はエージェントの生成とライフサイクルを管理します。又、定期的な処理の呼び出しと受信文言の提供を行います。お好み焼き協調料理は、エージェントを継承し、エージェント(オブジェクト)の提供と、定期的な呼び出し、受信文言の受け取りを行います。定期的な呼び出し毎に、状態遷移を処理し、受信毎に同様に要求を処理します。これら一連の処理でお好み焼き協調料理は実現されています。



(2)However, the boundary of SIGVerse and the okonomiyaki cooperation dish is a scatter denunciation and, the way things are going, because it becomes difficult to understand, will arrange OkonomiyakiController in the custom package as Facade in each package.

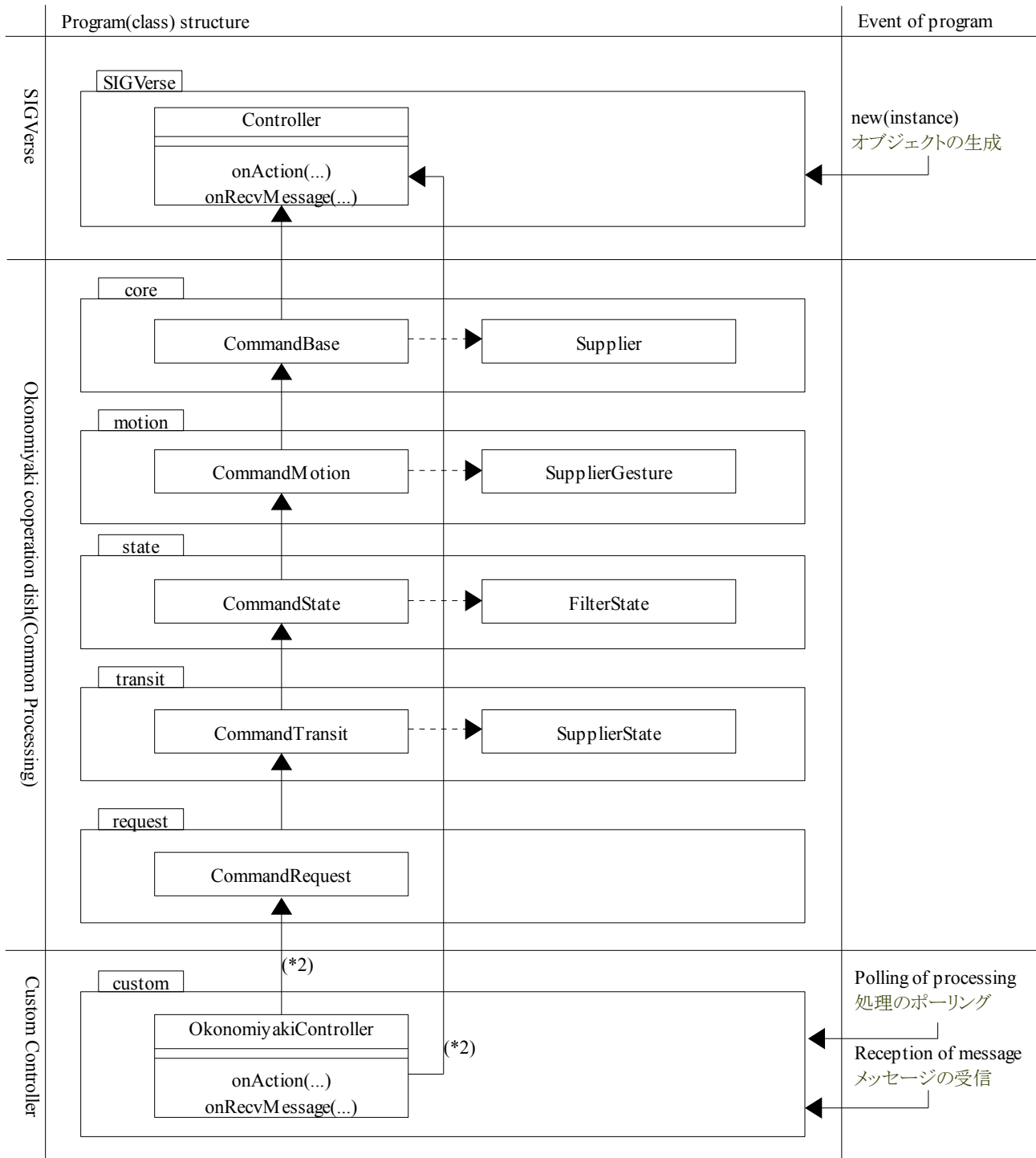
(2)但し、このままでは SIGVerse とお好み焼き協調料理の境界が各パッケージに散ばり、把握しづらくなる為、Facadeとして、OkonomiyakiController を custom パッケージに配置しています。以下に custom の関連情報を記します。



Name of program プログラムの名前	Storage directory 格納ディレクトリ	Role of Program プログラムの役割
OkonomiyakiController	\$\$SIGHOME/command/custom	SIGVerse and the okonomiyaki cooperation dish are mediated. The main processing is processing that accepts the event of SIGVerse, and forwards it to necessary processing. SIGVerseとお好み焼き協調料理の橋渡しを行います。主な処理は、SIGVerse のイベントを受け付け、必要な処理に転送する処理です。

The structure of a final okonomiyaki cooperation dish becomes as follows.

最終的なお好み焼き協調料理の構造は以下の通りになります。



(*2)When the okonomiyaki cooperation dish is succeeded to, OkonomiyakiController can use the processing of the bodily movement according to the state transition and it. The agent who doesn't use the okonomiyaki cooperation dish as ..custom.. controller in the opposite either can mount.

(*2)OkonomiyakiController は、お好み焼き協調料理を継承する場合、状態遷移やそれに伴う身体動作の処理を利用出来ます。逆にカスタムなコントローラとしてお好み焼き協調料理を利用しないエージェントも実装可能です。

5.3. お好み焼き協調料理の処理

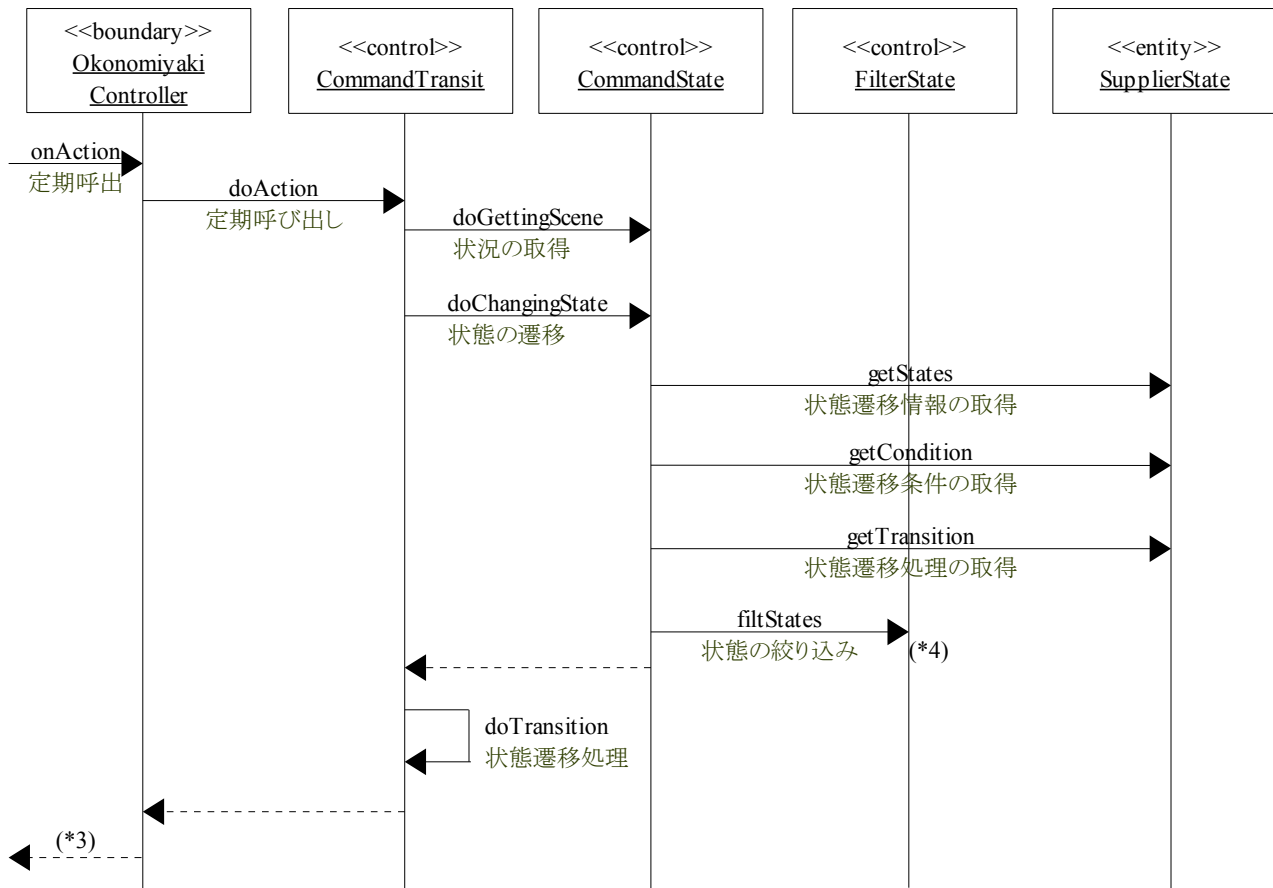
It explains a dynamic structure to cook the okonomiyaki cooperation recently. Processing from onAction and processing from onRecvMessage are chiefly recorded as follows. A dynamic structure looks like the relation among Boundary, Controller, and Entity. OkonomiyakiController corresponds to it in the meaning that accepts the demand from the outside though there is no Boundary (screen). Classes from which the head of the class starts by Command are almost equal to MacroCommand, and correspond to Controller. As for Entity, Supplier with the obligation of the access of the CSV definition file corresponds to it.

この節では、お好み焼き協調料理の動的な構造を説明します。主に onAction からの処理と onRecvMessage からの処理を以下に記します。動的な構造は、Boundary、Controller、Entity の関係に似ています。Boundary(画面)はありませんが、外部からの要求を受け付ける意味で OkonomiyakiController がそれに相当します。クラスの頭が Command で始まるクラス類は MacroCommand にほぼ等しく、Controller に相当します。Entity は CSV 定義ファイルのアクセスの責務を持つ Supplier がそれに相当します。

5.3.1. 状態遷移時の振る舞い(onAction)

The state transition is taken the state of the agent of the turn, acquires state transition information, the condition, and processing, is narrowed in the following state, changes if it is possible to change in the following state the state, executes processing, and works out with the flow.

状態遷移は、回りのエージェントの状態を取る、状態遷移情報・条件・処理を取得する、次の状態に絞り込む、もし次の状態に遷移が可能であれば状態遷移して処理を実行する、流れから成り立ちます。



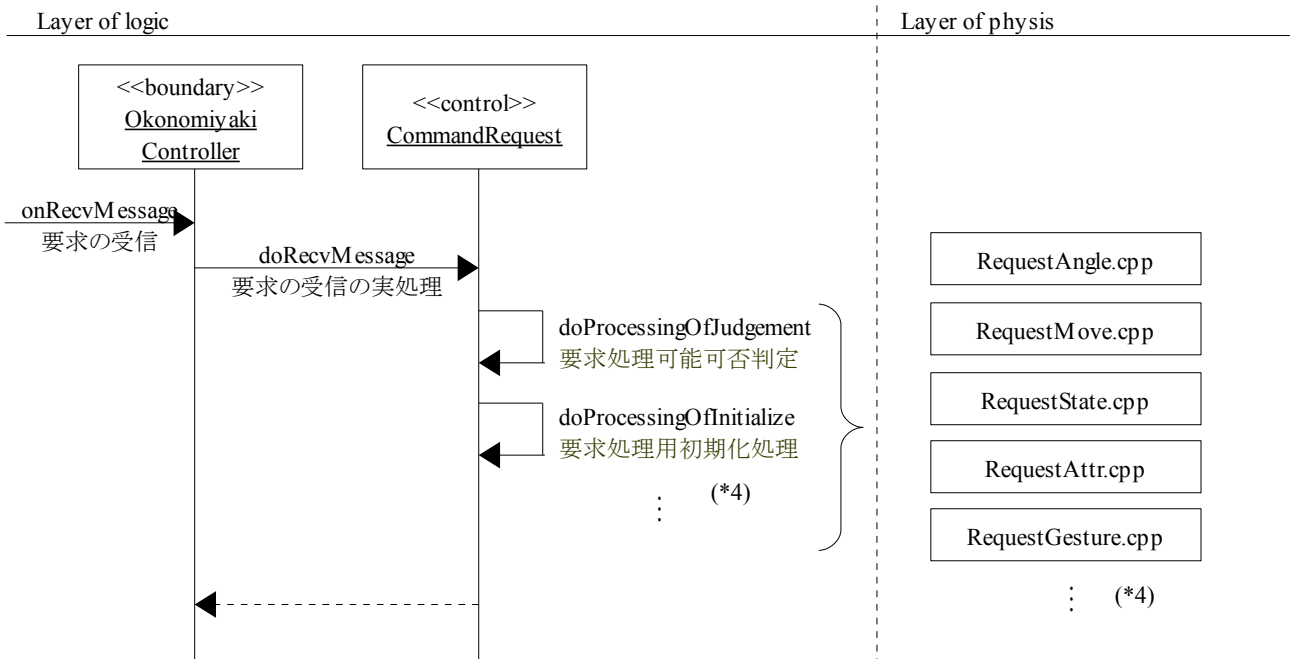
(*3)The interval time of this regular call is specified by the value returned with onAction. For instance, when 0.1 is returned, a regular call will be done in 0.1 seconds as follows.

(*3)この定期呼び出しの時間間隔は、onAction で返す値で指定します。例えば、0.1 を返した場合、次は 0.1 秒後に定期呼び出しが行われます。

5.3.2. 要求受信時の振る舞い(onRecvMessage)

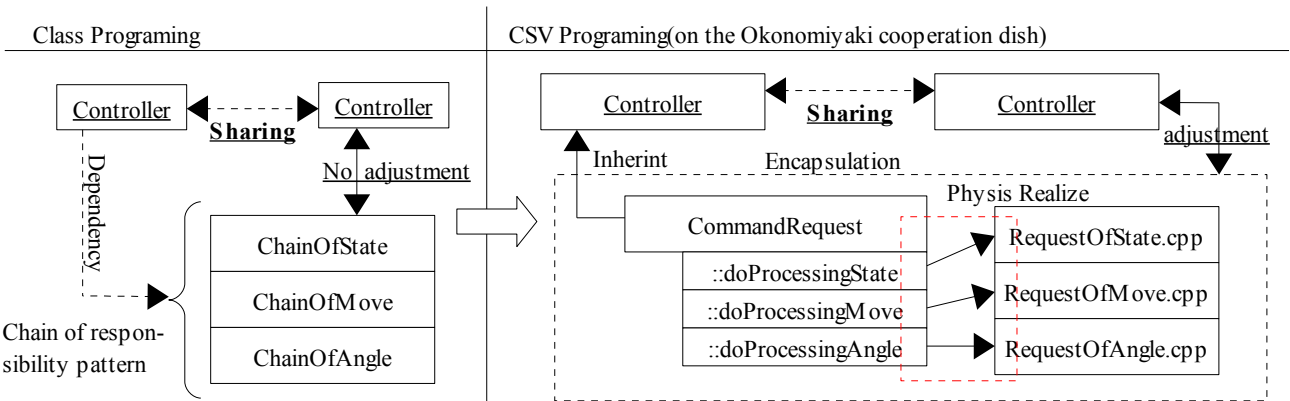
When the imperative sentence is transmitted from SIGViewer, the following operation is done. However, it is in onAction to execute an actual movement, the rotation, and the bodily movement, and the demand is chiefly converted into information for which onAction is necessary in this processing.

SIGViewerから命令文を送信した場合、以下の動作が行われます。但し、実際の移動や回転、身体動作を実行するのは onAction 内であり、この処理では主にその要求を onAction が必要な情報に変換します。



(*4)Because SIGVerse manages, and maintained information is shared at agent's (controller's instance) life cycle, the okonomiyaki cooperation dish doesn't generate the instance to maintain and to secure the correspondence with it at all. It means the thing without the processing of the okonomiyaki cooperation dish for which the agent depends. In the assumption, the processing of the demand adopts the method added to the agent as a method without making it to the class. For instance, processing to demand "State" adds not the class but method "doProcessingOfState". And, a physical composition of the program becomes the method of making one file a method and giving the readability.

(*4)エージェント(コントローラのインスタンス)のライフサイクルは、SIGVerse が管理し、保持する情報の共有も行われる為、お好み焼き協調料理はそれとの整合性を維持・保障する為に、一切のインスタンス生成を行いません。それは、エージェントは依存するお好み焼き協調料理の処理を持たない事を意味します。その前提において、要求の処理はクラス化せずに、エージェントにメソッドとして追加していく方式をとります。例えば、要求「State」に対する処理は、クラスではなくメソッド「doProcessingOfState」の追加を行います。そしてプログラムの物理構成は、1メソッドにつき、1ファイルを作成して可読性をあげる方法となります。



5.3.3. 移動・回転・身体動作の振る舞い(onAction)

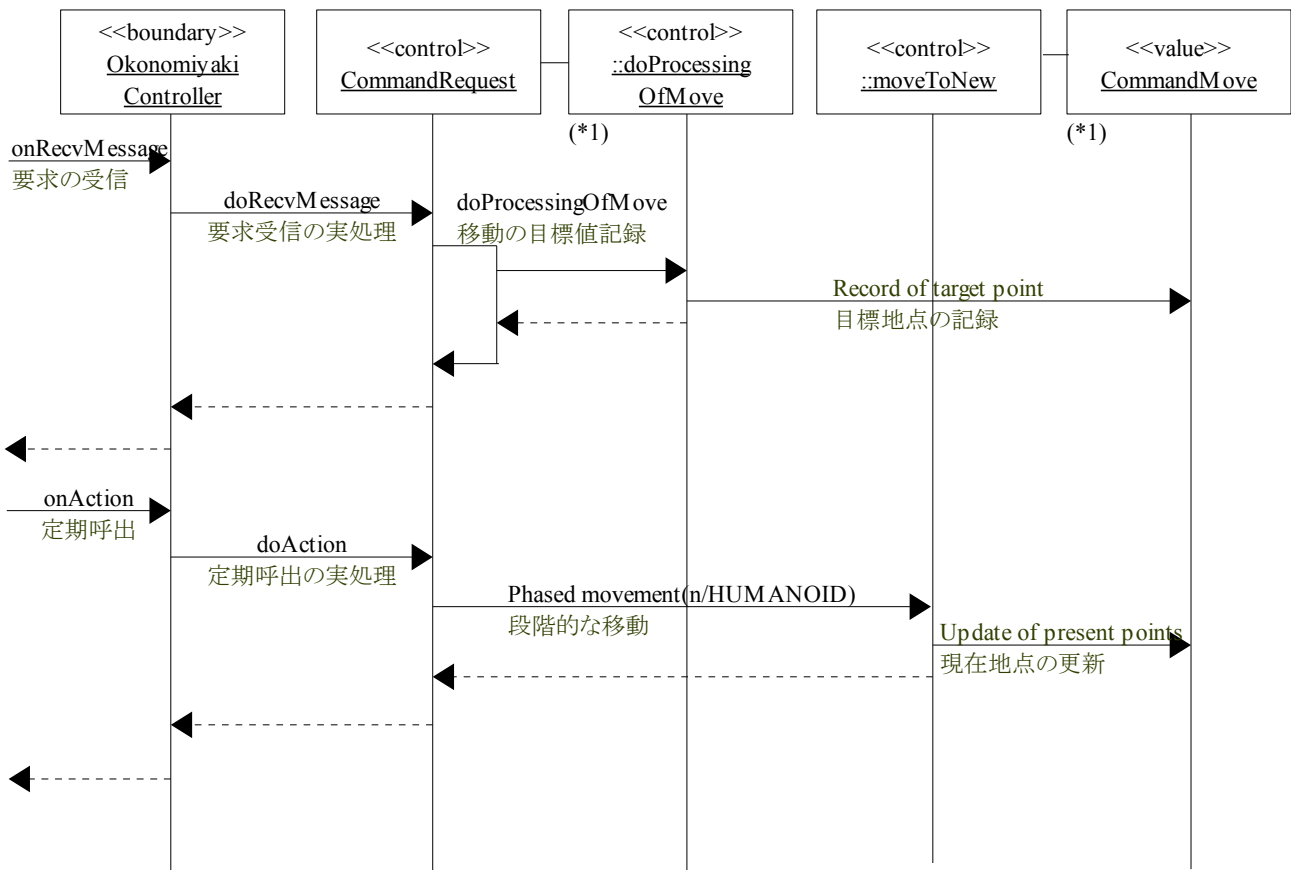
The okonomiyaki cooperation dish does the state transition processing according to the timing of onAction that is regularly called. Various movement, rotation, and bodily movements also do intergrade, the rotation, and the bodily movement according to the timing of onAction as well as it for the angle indirectly of the angle and the target of the point and the target of the target.

The frequency of this stage is being defined in five by the constant named HUMANOID by Define.h. It comes close to one's goal by ten stages if the frequency of this stage is changed from five to ten. And, the value of the target is information on the movement defined by CommandMove.h in case of the movement, Information on angle defined in CommandMotion.h in case of rotation, Information on angle indirectly of definition in CommandGesture.h in case of bodily movement, is maintained. It gradually approaches the value of the target at onAction. A dynamic sample of Figure of the processing is recorded as follows.

お好み焼き協料理は、定期的と呼ばれる onAction のタイミングで状態遷移処理を行います。それと同様に、各種の移動・回転・身体動作も目標の地点・目標の角度・目標の間接の角度を対象に、onAction のタイミングで段階的に移動・回転・身体動作を行います。

この段階の回数は Define.h に HUMANOID という定数で 5 に定義されています。この段階の回数を 5 から 10 に変更すれば、10 段階で目標に近づきます。そして目標の値は、移動の場合、CommandMove.h で定義されている移動に関する情報。回転の場合、CommandMotion.h に定義されている角度に関する情報。身体動作の場合、CommandGesture.h に定義されている間接の角度に関する情報。に保持されます。onAction の度に、段階的にその目標の値に近づきます。以下にその処理の動的な図例を記します。

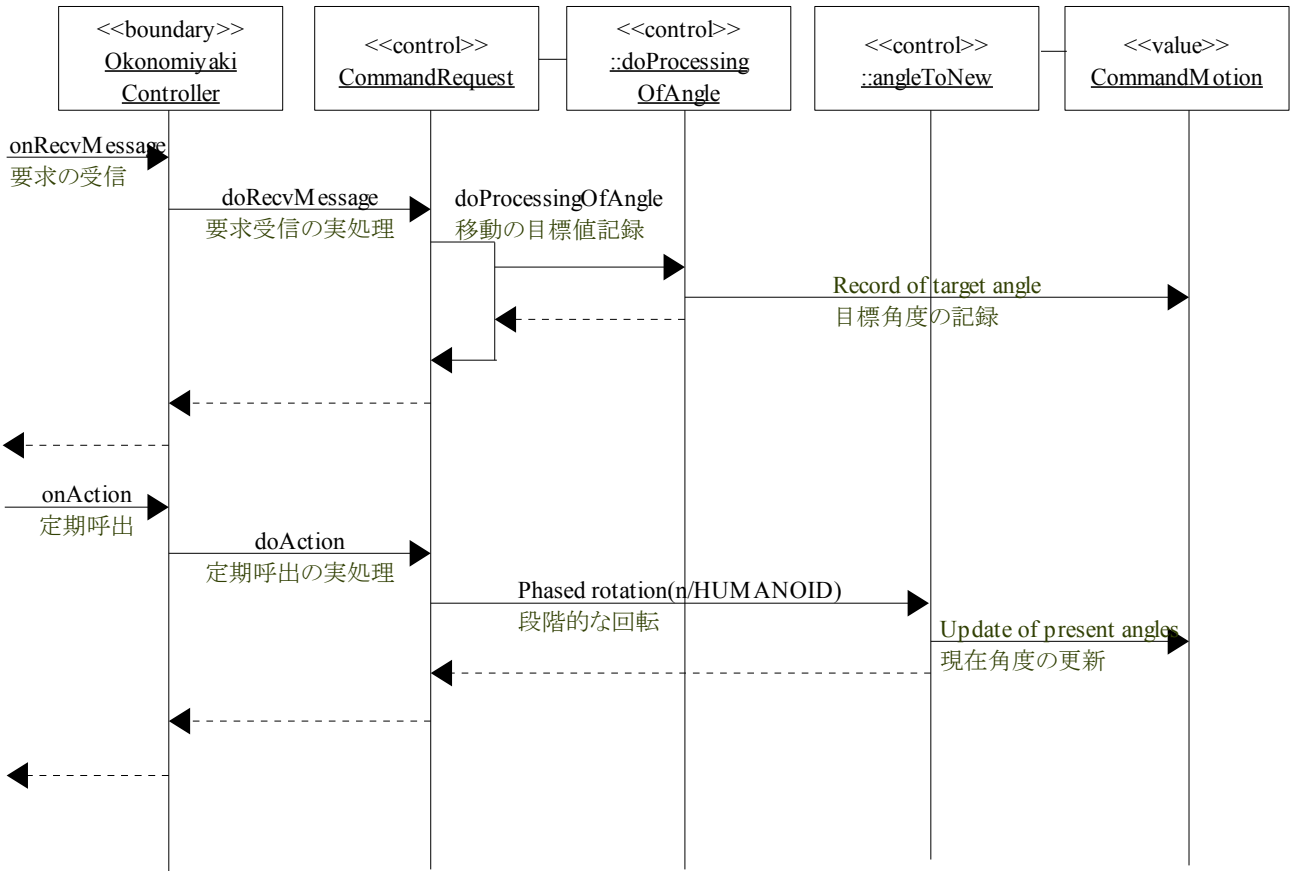
5.3.3.1. 移動の振る舞い(onAction)



(*1)In the relation between CommandRequest and ::doProcessingOfMove, method ::doProcessingOfMove is involving to class ComandRequest.

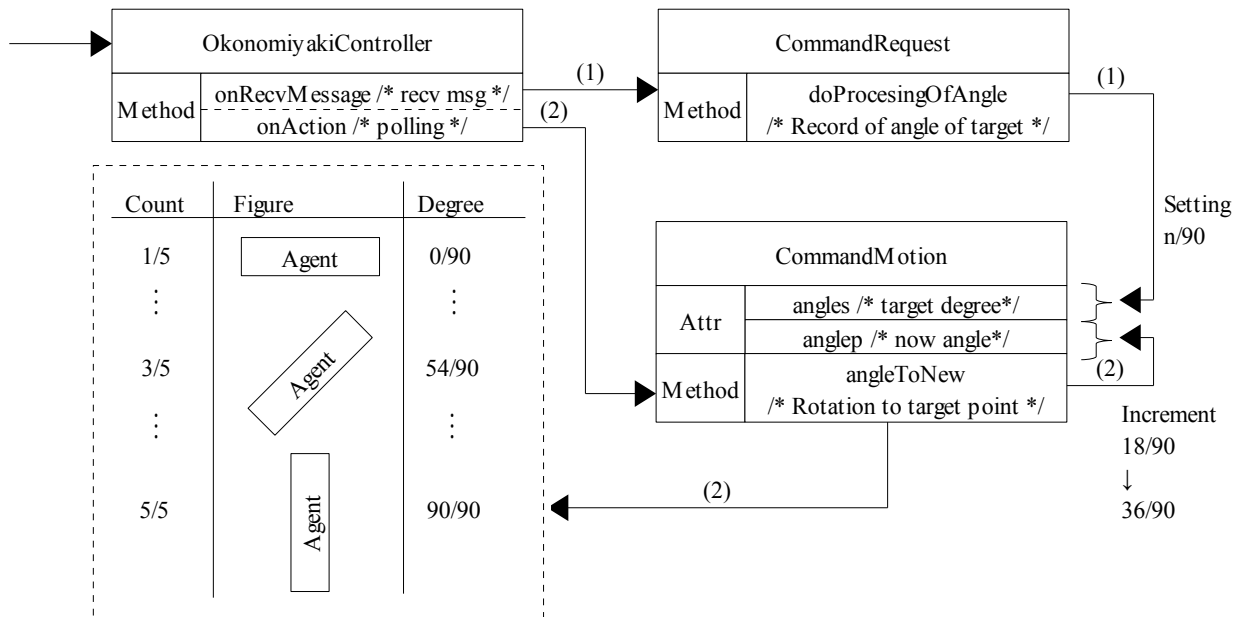
(*2)CommandRequest と ::doProcessingOfMove の関係は、メソッド ::doProcessingOfMove はクラス ComandRequest に内包です。

5.3.3.2. 回転の振る舞い(onAction)

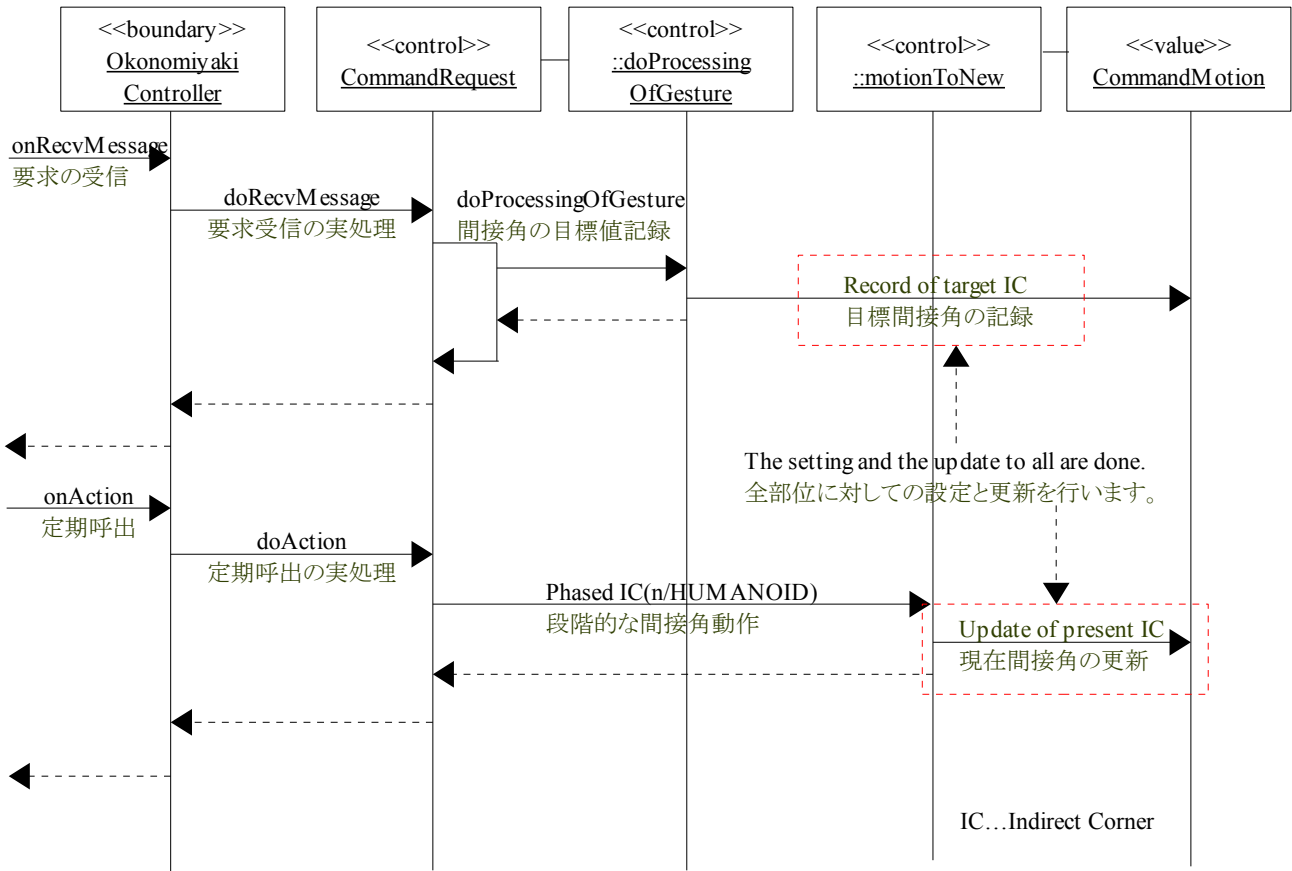


Processing concerning the rotation is almost the same as processing concerning the movement. Information that aims maintains in not CommandMove but CommandMotion, and the point that processing changes into not CommandMove but ComandRotation is different. Two of the processing of the movement and the rotation are matched, and the sample of Figure of processing is recorded as follows.

回転に関する処理は、移動に関する処理と殆ど同じです。目標とする情報が CommandMove ではなく、CommandMotion に保持、処理が CommandMove ではなく ComandRotation に変わる点が異なります。移動と回転の処理の二つを合わせて、以下に処理の図例を記します。

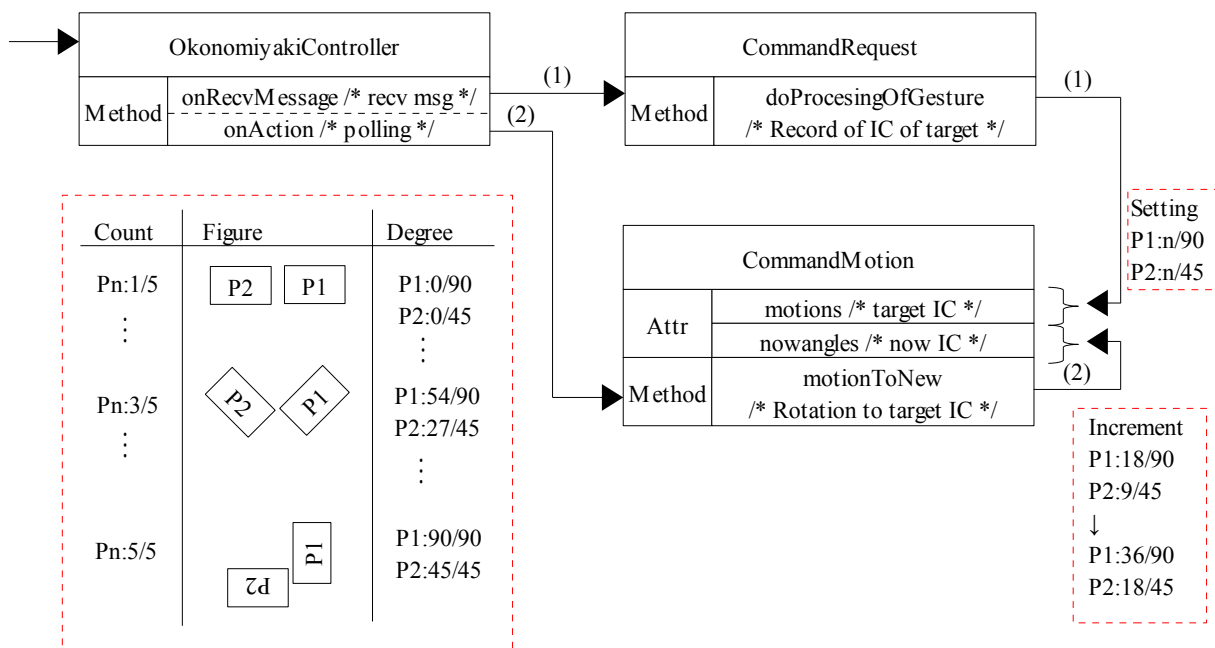


5.3.3.3. 身体動作の振る舞い(onAction)



The operation of an indirect corner is almost the same as the movement and the rotation. However, an indirect corner of each part is processed though the movement and the rotation are processed by the unit of the agent. It is greatly different there.

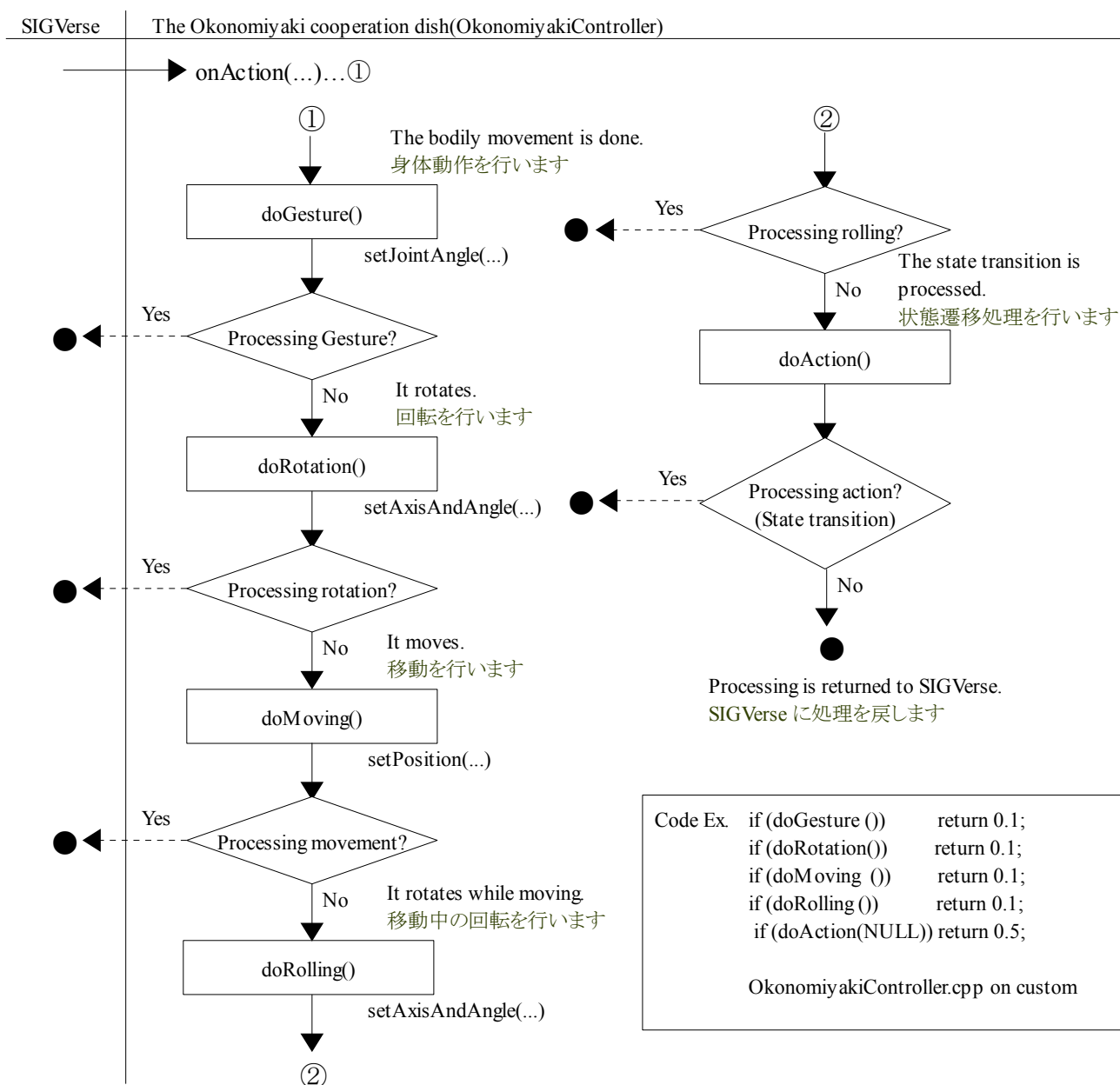
間接角の動作は、移動・回転とほぼ同じです。但し、移動・回転はエージェント単位で処理を行いますが、間接角は部位毎に処理を行います。そこが大きく違います。



5.3.4. 移動・回転・身体動作の順序 (onAction)

The okonomiyaki cooperation dish is processed based on the call of onAction. Moreover, the necessity executed at the same time doesn't have a present place for the movement, the rotation, and the bodily movement. Therefore, processing is executed in the okonomiyaki cooperation dish putting the priority level for the movement, the rotation, the bodily movement, and the state transition. The outline of the processing flow is recorded as follows. This priority level is the processing order written in onAction of OkonomiyakiController. To change this order, the content of OkonomiyakiController is changed at the right time.

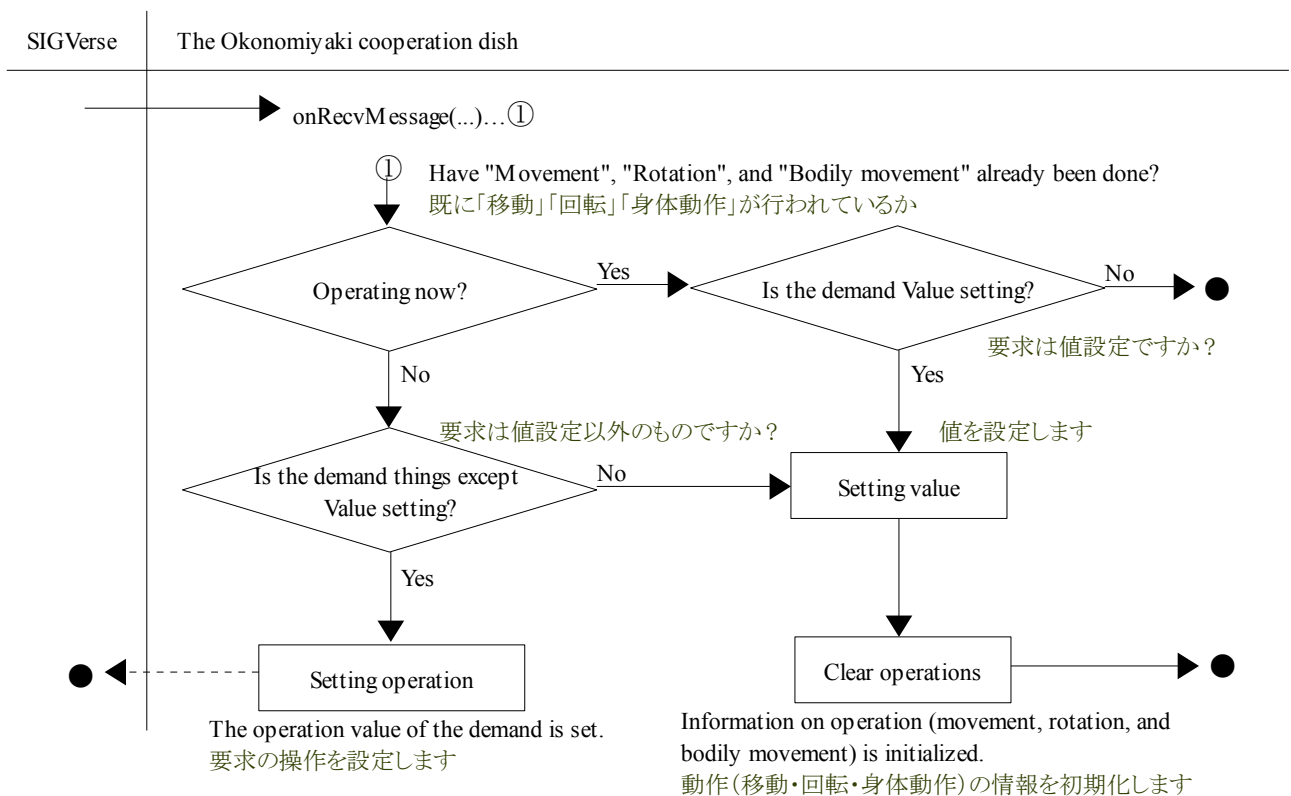
お好み焼き協調理は onAction の呼出を元に処理を行います。また、移動・回転・身体動作に関して、同時に実行する必要は現在の所ありません。その為、お好み焼き協調理では、移動・回転・身体動作・状態遷移に関して優先順位を付けて処理を実行します。以下に処理フローの概要を記します。この優先順位は、OkonomiyakiController の onAction 内に書かれている処理順序です。この順序を変えたい場合は、OkonomiyakiController の内容を適時変更します。



The okonomiyaki cooperation dish assumes operation already if the demand is operation (movement, rotation, and bodily movement) when a demand specified from another agent is received and disregards a demand. It gives priority more than operation if the demand is a state transition, and the attribute value settings other than operation, the information is recorded as the state transition, and information on operation (movement, rotation, and bodily movement) is cleared oppositely. The list of the behavior at the demand is recorded as follows.

お好み焼き協調料理は、他エージェントから明示的な要求を受信した場合、その要求が動作(移動・回転・身体動作)であれば、既に動作中であるとして要求を無視します。その要求が動作以外の状態遷移、属性値設定であれば動作より優先して状態遷移とその情報の記録を行い、逆に動作(移動・回転・身体動作)の情報をクリアします。以下にその要求時の振る舞いの一覧を記します。

Requirement 要求名	Category 要求分類	Requirement content 要求内容	Interrupt 割り込み	Example of requirement 具体例
State transition 状態遷移	Value setting 値設定	The state is specification set. 指定状態に状態を設定します	Permission 許可	State=1
Attribute definition 属性定義	Value setting 値設定	The value is set to a specified attribute. 指定属性に値を設定します	Permission 許可	Attr=3
Movement 移動	Operation setting 動作設定	It moves to the specified coordinates position. 指定座標位置に移動します	Rejection 却下	Move=100:100:300:0:90:10
Rotation 回転	Operation setting 動作要求	It rotates to a specified angle with a specified axis. 指定軸で指定角度に回転します	Rejection 却下	Angle=0:1:0:0:180:10
Bodily movement 身体動作	Operation setting 動作要求	Indirect is rotated to specification and indirect an angle. 指定間接角度に間接を回転します	Rejection 却下	Gesture=ges92r
Supplementary level 補助度	Value setting 値設定	It sets it to specification and assistance levels. 指定補助度に設定します	Permission 許可	Aid=3
Remark level 発言度	Value setting 値設定	It sets it to the specified remark level. 指定発言度に設定します	Permission 許可	Uttr=2

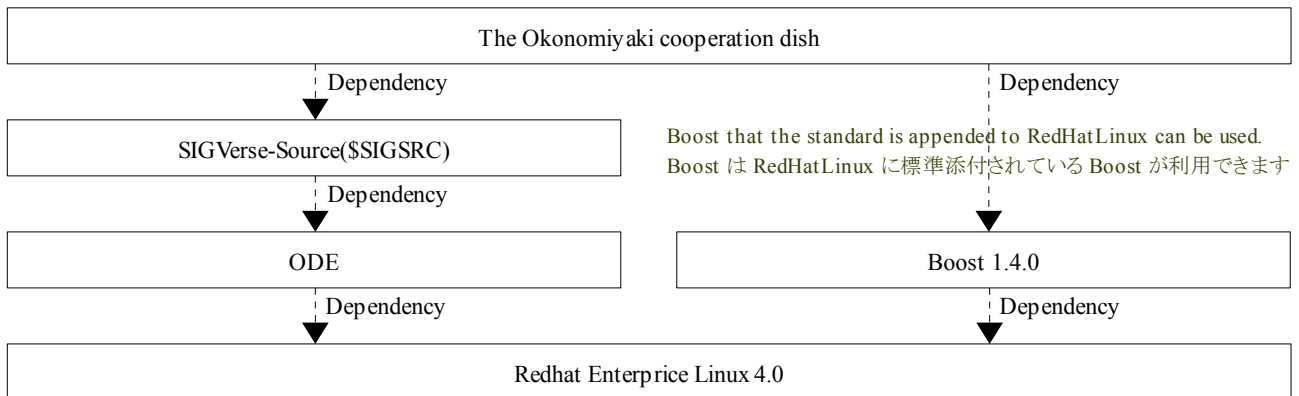


5.4. お好み焼き協調料理のコンパイル

The okonomiyaki cooperation dish can be compiled on Linux-OS, and a general G++ compiler be used. The depending software becomes SIGVerse and Boost1.4.0 or more. The sample of Figure of the dependence and the procedure of the compilation are recorded as follows.

お好み焼き協調料理は、Linux-OS 上でコンパイルし、一般的な G++コンパイラが利用出来ます。依存するソフトウェアは、SIGVerse と Boost1.4.0 になります。以下に依存関係の図例とコンパイルの手順を記します。

5.4.1. お好み焼き協調料理の依存ソフトウェア



5.4.2. お好み焼き協調料理のコンパイル

The compilation of the okonomiyaki cooperation dish only specifies SIGVerse(\$SIGSRC) and Boost for an include file besides the compilation of the usual C language program. The example of the argument of G++ when compiling with the unit is recorded as follows.

お好み焼き協調料理のコンパイルは通常の C 言語プログラムのコンパイル以外に、インクルードファイルとして、SIGVerse(\$SIGSRC)と Boostを指定するだけです。以下に単体でコンパイルする場合の G++に対する引数の例を記します。

```

Compile Ex.  g++ -DCONTROLLER -DNDEBUG (*1)
              -I"/home/nii/command/action"
              -I"/home/nii/command/core"
              -I"/home/nii/command/custom"
              -I"/home/nii/command/motion"
              -I"/home/nii/command/request"
              -I"/home/nii/command/shared"
              -I"/home/nii/command/state"
              -I"/home/nii/command/transit"
              -I/home/nii/sigverse-install-pack-090313/server/irwas-sim-090313/srcs
              -I/home/nii/sigverse-install-pack-090313/server/irwas-sim-090313/srcs/model
              -I/home/nii/sigverse-install-pack-090313/server/irwas-sim-090313/srcs/commonlib
              -I/home/nii/libs/boost_1_40_0
              -O3 -fPIC -Wall
              CommandRequest.cpp
    
```

It is a source program of the okonomiyaki cooperation dish.
お好み焼き協調料理のソースプログラムです

It is a source of SIGVerse.
SIGVerse ソースです

It is a source of Boost.
Boost ソースです

In this example, \$SIGSRC is assumed to be "/home/nii/sigverse-install-pack-090313/server/irwas-sim-090313/srcs". Moreover, the okonomiyaki cooperation dish is copied following "/home/nii/command".

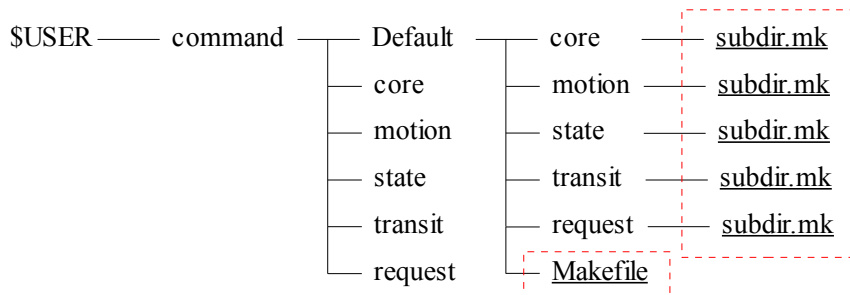
この例では、\$SIGSRC は「/home/nii/sigverse-install-pack-090313/server/irwas-sim-090313/srcs」としています。また、お好み焼き協調料理を「/home/nii/command」以下にコピーしています。

(*1)The compilation includes an optional compilation peculiar to SIGVerse. It is "-DCONTROLLER" and "-DNDEBUG". "-DNDEBUG" is a flag for the assertion used by usual C language, and the debug processing is assumed to be invalid specifying this. When "-DCONTROLLER" is added, bodily movement (setJointAngle) becomes effective. The bodily movement becomes invalid when not adding. (SetJointAngle is lost.)

(*1)コンパイルには、SIGVerse 特有のコンパイルオプションがあります。それは「-DCONTROLLER」と「-DNDEBUG」です。「-DNDEBUG」は通常の C 言語で用いられるアサーションの為のフラグであり、これを指定してデバック処理を無効とします。「-DCONTROLLER」は、付加した場合、身体動作 (setJointAngle) が有効になります。付加しない場合は、身体動作は無効になります。(setJointAngle 自体がなくなります)

The okonomiyaki cooperation dish makes SharedObject for the okonomiyaki cooperation dish by using an original makefile. This is a makefile that Eclipse-IDE made by the automatic operation. The list of the whereabouts of the makefile and the file according to it is recorded as follows.

お好み焼き協調理は、独自のメイクファイルを用いてお好み焼き協調理用の SharedObject を作成します。これは Eclipse-IDE が自動で作成したメイクファイルです。以下にメイクファイルの所在とそれに依存するファイルの一覧を記します。



Name of file ファイルの名前	Whereabouts of file ファイルの所在	Outline of file ファイルの概要
Makefile	command/Default/Makefile	The okonomiyaki cooperation dish is compiled, and SharedObject for the agent is generated. お好み焼き協調理をコンパイルし、エージェント用の SharedObject を生成します。
subdir.mk	command/Default/core/subdir.mk	The program file name, the objectfile name, and the dependence library of the program compiled by the directory are defined. そのディレクトリでコンパイルするプログラムのファイルの名前とオブジェクトファイル名、依存ライブラリを定義します。
subdir.mk	command/Default/motion/subdir.mk	
subdir.mk	command/Default/state/subdir.mk	
subdir.mk	command/Default/transit/subdir.mk	
subdir.mk	command/Default/request/subdir.mk	

The makefile executes the compilation by the "make" command as well as a general makefile.

メイクファイルは一般的なメイクファイルと同様に「make」コマンドでコンパイルを実行します。

```

Make Ex.  cd $USER/command/Default
          make -f Makefile

```

The makefile executes the compilation by the "make" command as well as a general makefile.

結果、「\$USER/command/Default」に libcommand.so が生成されていれば、正常にコンパイルできています。

```

Confirm Ex.  cd $USER/command/Default
             ls -l
             496853 1月 4 12:11 libcommand

```

5.5. お好み焼き協調料理の知覚の改造

5.5.1. 視界内のエージェントの一覧を取得する

In the okonomiyaki cooperation dish, the state of the agent of the turn is acquired as a situation. In that case, the name of the agent of the object that takes the state is maintained in "agents.csv". However, SIGVerse can acquire all agent's names that enter agent's view. The use of the API and the method of improving the okonomiyaki cooperation dish are recorded as follows.

お好み焼き協調料理では、状況として回りのエージェントの状態を取得します。その際に、状態を取る対象のエージェント名は「agents.csv」に保持されます。ですが、SIGVerse はエージェントの視界に入る全てのエージェントの名前を取得できます。以下にその API の使用方法と、お好み焼き協調料理の改善の方法を記します。

5.5.1.1. detectEntities

DetectEntities enumerates the name of the agent that enters agent's view. DetectEntities has the format of the following API.

detectEntities はエージェントの視界に入るエージェント名を列挙します。detectEntities は以下の API の書式を持ちます。

Argument	I/O	Type	Explanation of argument	Example
1 st argument	OUT	vector<string>	All agent's names that come into sight are substituted. 視界に入る全てのエージェントの名前を代入します。	Avator_000, Okonomi_000
Return value	OUT	bool	When true and the mistake are found when correctly processing it, false is substituted. 正しく処理した場合は true、間違いがある場合は false を代入します。	true, false

The example of describing detectEntities is recorded as follows. For instance, when this sample is mounted on the robot, all agents, Avator, and the seaweed, etc. that enter the view of the robot can be acquired.

以下に detectEntities の記述例を記します。例えば、このサンプルをロボットに実装した場合、ロボットの視野に入る全てのエージェント、アバタ、海苔などが取得できます。

Code Ex.

```
try {
    vector<string> agents;
    bool b = detectEntities(agents);
    if (b) {
        for (vector<string>::iterator i=agents.begin(); i!=agents.end(); i++) {
            std::string name = *i;
            LOG_MSG((" %s detected %s", myname(), name.c_str()));
        }
    }
} catch (SimObj::Exception &) {
    LOG_ERR(("detectEntities failure"));
}
```

} The list of the name of the agent in view is acquired with detectEntities.
detectEntities で視界内のエージェントの一覧を取得します

} The list of the name of the agent is displayed in the "MSG" tag of SIGViewer as a log message. .
ログメッセージとして、エージェント名の一覧を SIGViewer の「MSG」タグに表示します。

} The exception handling is done.
例外処理を行います

And, the list of the name of the agent acquired in these detectEntities is the same as the content of detectEntities displayed with SIGViewer.

そして、これらの detectEntities で取得されたエージェント名の一覧は SIGViewer で表示されている detectEntities の内容と同じです。



5.5.1.2. getAllEntities

In the okonomiyaki cooperation dish, the part where all names of the agent are acquired is described in getAllEntities in "core/CommandBase.cpp". This content is replaced with detectEntities.

お好み焼き協調理では、全てのエージェント名を取得する箇所は、「core/CommandBase.cpp」内の getAllEntities に記述されています。この内容を detectEntities に置き換えます。

```

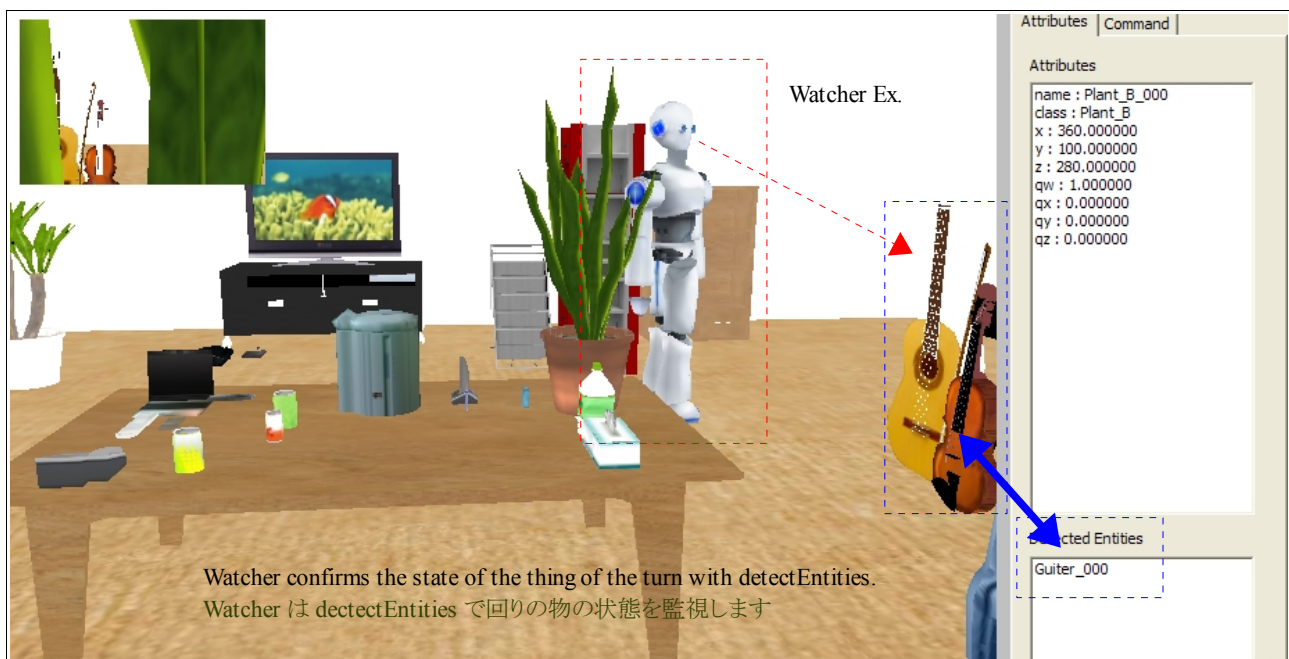
Revise Ex. int CommandBase::getAllEntities(Values *values)
{
    string ftarget = string(CSVDIR) + string("agents.csv");
    try {
        ifstream fin(ftarget.c_str(), ios::in);
        if (!fin) {
            LOG_MSG(("not found agents data %s", ftarget.c_str()));
            return false;
        }
        string line;
        while (fin >> line) {
            values->push_back(line);
        }
        fin.close();
    } catch (...) {
        LOG_MSG(("read attribute file failure(%s)", ftarget.c_str()));
    }
    return true;
}

```

CommandBase.cpp on core

The state of the agent in view is changed to a ..okonomiyaki cooperation dish.. not passive agent in the above-mentioned change as confirmed. The robot from which the dining etc. are watched with \$SIGSRC/model/samples/Watcher.so that observes it with detectEntities for instance can be achieved by using this detectEntities.

以上の変更で、お好み焼き協調理上の受動的ではないエージェントは、視界内のエージェントの状態を確認する様に変更されます。この detectEntities を用いて、例えば detectEntities で監視を行う \$SIGSRC/model/samples/Watcher.so でダイニング等を見張るロボットを実現できます。



5.6. お好み焼き協調料理の力学の改造

5.6.1. ある方向に力をかける

Because it doesn't limit to the okonomiyaki cooperation dish, and SIGVerse uses a part of ODE for the target, a physical operation concerning mechanics can be calculated. However, only the calculation excluding the man type agent (agent who has an indirect corner) is at present possible. In this paragraph, it explains the easy example, setting method, and API used.

お好み焼き協調料理に限らず、SIGVerseはODEを一部分的に使用している為、力学に関する物理演算が計算可能です。ですが、現在の所、人間型エージェント(間接角を持つエージェント)以外での計算のみ可能です。本項では、その簡単な例と設定方法、使用するAPIを説明します。

5.5.1.2. moveTo

MoveTo turns specified agent's power to the specified coordinates point. For instance, when power at 1000 levels is set to the agent whose initial coordinates are the world coordinates point starting points, and power is turned in x=300, y=0, z=300 directions with moveTo, SIGVerse faithfully executes a physical operation according to the setting. The result and the agent move with ground while repulsing it. The moveTo has the following formats.

moveToは、指定エージェントが持つ力を指定座標点に向けます。例えば、初期座標が世界座標点原点であるエージェントに千程度の力を設定し、moveToで300,0,300方向に力を向けた場合、SIGVerseはその設定に従い忠実に物理演算を実施します。結果、エージェントは地面と反発しながら移動します。そのmoveToは以下の書式を持ちます。

Argument	I/O	Type	Explanation of argument	Example
1 st argument	IN	double	X coordinates point is specified. X座標点を指定します。	0, 100, 500
2 nd argument	IN	double	Y coordinates point is specified. Y座標点を指定します。	0, 100, 500
3 rd argument	IN	double	Zcoordinates point is specified. Z座標点を指定します。	0, 100, 500

The example of describing moveTo is recorded as follows. MoveTo is API that indicates the direction, and no function to move the agent to specified coordinates accurately. It depends on the size of power to set where the agent moves. If it is few, it doesn't reach, and if it is too large, it passes. It is necessary to adjust strength of power at the right time.

以下にmoveToの記述例を記します。moveToは方向を示すAPIであり、指定座標まで正確にエージェントを移動する機能ではありません。どこまでエージェントが移動するか設定する力の大きさによります。少なければ届きませんし、大きすぎれば通り過ぎます。適時力の強さを調整する必要があります。

Code Ex.	<code>moveTo(0, 0, 0);</code>	It is very simple. The coordinates point of X, Y, and Z is specified. とてもシンプルです。X,Y,Zの座標を指定します。
----------	-------------------------------	--

the value of power to each axis is set. For instance, when 1000 is set to X axis, it describes it as follows.

更に各軸に対する力の値を設定します。例えば、X軸に1000を設定する場合は、以下の様に記述します。

Code Ex.	<code>SimObj *obj = getObj(myname()); obj->fx(1000);</code>
----------	--

This FX can be set even by "MyWorld.xml". FX in this case becomes a definition of initial power.

このFXは「MyWorld.xml」でも設定できます。この場合のFXは初期の力の定義になります。

Configure Ex.	<code><instanciate class="Toy_D.xml"> <set-attr-value name="fx" value="1000.0"/></code>	My World.xml on \$SIGHOME/conf
---------------	---	--------------------------------

This time, a virtual space is newly made on SIGVerse. Two kinds (the agent who doesn't simply indicate the direction with moveTo and the agent who indicated the direction in 300,0,300 coordinates with moveTo) are arranged. MyWorkd.xml is defined as follows. New MoveController is made for moveTo this time because there is no processing that corresponds to moveTo in the okonomiyaki cooperation dish. MoveController is described as follows. It is assumed "/home/nii/samples/MoveController.cpp" though any file name is not cared.

今回は、新規に SIGVerse 上に仮想空間を作成します。シンプルに moveTo で方向を示さないエージェントと、moveTo で 300,0,300 座標に方向を示したエージェントの2種類を配置します。お好み焼き協調料理では moveTo に該当する処理はない為、今回 moveTo 用に新しい MoveController を作成します。MoveController は以下の様に記述します。ファイル名は何でも構いませんが、仮に「/home/nii/samples/MoveController.cpp」とします。

```

Code Ex.  #include "Controller.h"
           #include "Logger.h"
           } Controller of SIGVerse and the use of the logger are declared.
           } SIGVerse の Controller とロガーの利用を宣言します。

class MoveController : public Controller {
public:
    double onAction(ActionEvent&);
};

double MoveController::onAction(ActionEvent &evt) {
    SimObj *obj = getObj(myname()); ... ①
    moveTo(0, 0, 300); ... ②
    obj->fz(1000); ... ③
    return 5.0; ... ④
}

extern "C" Controller * createController() {
    return new MoveController;
}
    } Own instance is returned to SIGVerse.
    } 自身のインスタンスを SIGVerse に返します。
    
```

MoveController.cpp on /home/nii/samples

This MoveController is compiled and SharedObject is generated. The name of SharedObject is assumed to be "Move.so". The source directory of SIGVerse is specified include ahead.

この MoveController をコンパイルして SharedObject を生成します。SharedObject の名前は、「Move.so」としています。インクルード先に SIGVerse のソースディレクトリを指定します。

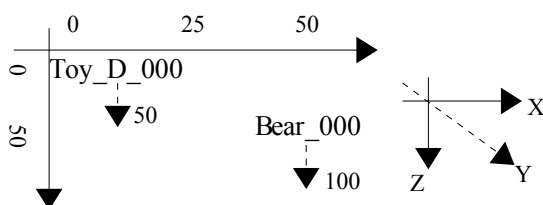
```

Compile Ex.  export SRCDIR=/home/nii/sigverse-install-pack-090313/server/irwas-sim-090313/srcs/

              g++ -shared -o Move.so -fPIC -DNDEBUG -DUSE_ODE -DCONTROLLER -I $SRCDIR/model
              -I $SRCDIR/model/commonlib ./MoveController.cpp
    
```

Two agents are arranged in a virtual space of SIGVerse. The name is made "Toy_D_000" and "Bear_000". "Toy_D_000" is arranged in the place where "Bear_000" parts a starting point a little. This time, it makes the assumption of moveTo effective only Toy_D_000. The conceptual diagram in a virtual section is recorded as follows.

SIGVerse の仮想空間には 2 エージェントを配置します。名前は「Toy_D_000」と「Bear_000」にしましょう。「Toy_D_000」は原点に、「Bear_000」はやや離れた場所に配置します。今回、moveTo を有効とするのは Toy_D_000 のみにします。以下に仮想区間内の概念を記します。



Name of Agent	Outline of operation
Toy_D_000	Power 1000 joins 300 and 0,300 directions. 300,0,300 方向に力 1000 が加わります
Bear_000	Nothing is controlled. 何も制御しません

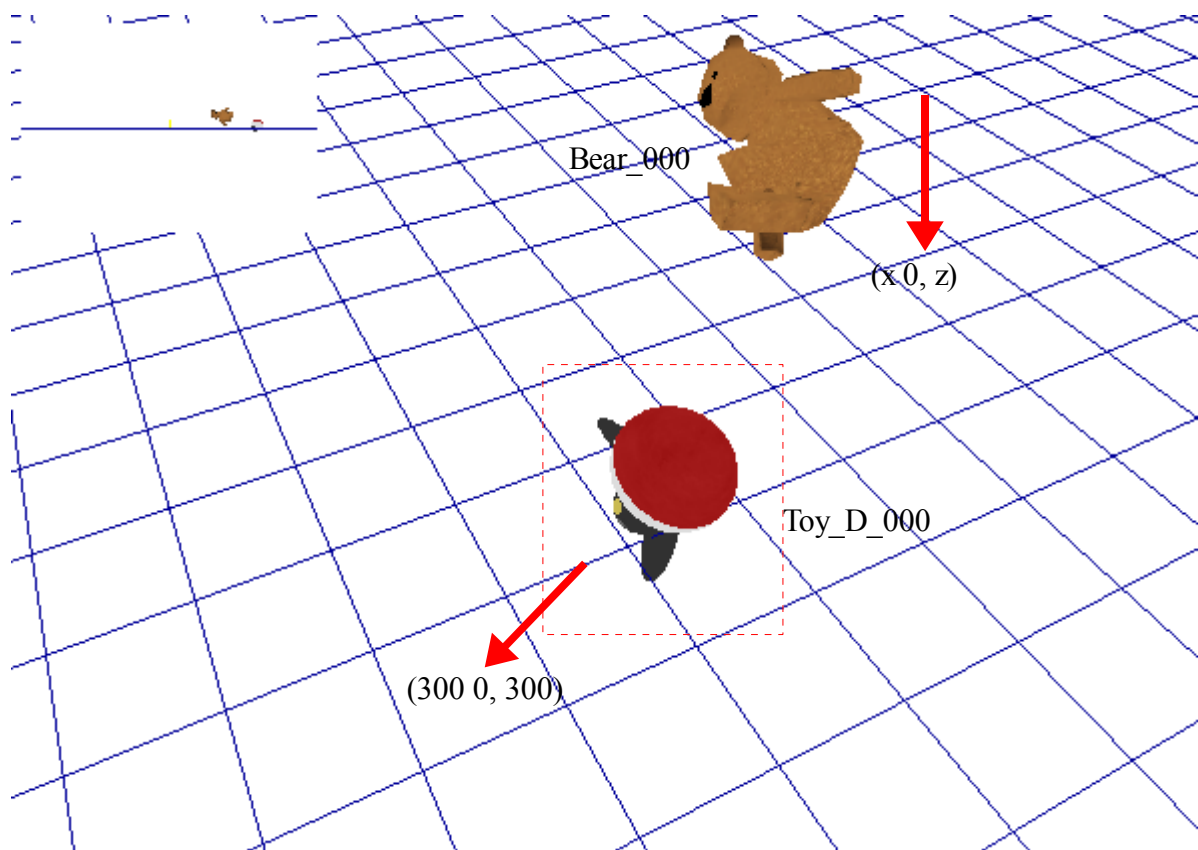
The definition of the virtual world of SIGVerse is written in "MyWorld.xml". The definition example is recorded as follows. Especially, it is confirmed that the setting of a physical operation is effective (dynamics, true).

SIGVerse の仮想世界の定義は、「MyWorld.xml」に書きます。以下に定義例を記します。特に物理演算の設定が有効 (dynamics, true) であると確認します。

<pre> Difinition Ex. <world name="myworld"> <gravity x="0.0" y="-9.8" z="0.0"/> <instanciate class="Toy_D.xml"> <set-attr-value name="name" value="Toy_D_000"/> <set-attr-value name="dynamics" value="true"/> <set-attr-value name="language" value="c++"/> <set-attr-value name="implementation" value="/home/nii/samples/Move.so"/> <set-attr-value name="x" value="0.0"/> <set-attr-value name="y" value="50.0"/> <set-attr-value name="z" value="0.0"/> <set-attr-value name="fx" value="1000.0"/> <set-attr-value name="fy" value="1000.0"/> <set-attr-value name="fz" value="1000.0"/> </instanciate> </pre>	<pre> <instanciate class="Bear.xml"> <set-attr-value name="name" value="Bear_000"/> <set-attr-value name="dynamics" value="true"/> <set-attr-value name="x" value="50.0"/> <set-attr-value name="y" value="100.0"/> <set-attr-value name="z" value="50.0"/> <set-attr-value name="qx" value="0.0"/> <set-attr-value name="qy" value="0.0"/> <set-attr-value name="qz" value="0.0"/> </instanciate> </world> </pre>
---	--

The Central server is started like starting the okonomiyaki cooperation dish now.

後は、お好み焼き協調料理を起動する要領でセントラルサーバを起動します。



Bear_000 falls to the right under according to gravity as shown in the above figure, and Toy_D_000 flies in 300 and 0,300 directions.

上図の通り、Bear_000 は重力に従い真下に落下し、Toy_D_000 は 300,0,300 方向に飛んで行きます。

5.7. お好み焼き協調料理の対話の改造

5.7.1. 全てのエージェントに話しかける

In the okonomiyaki cooperation dish, the remark of the robot takes the shape called a log output. As for this, any robot doesn't actually utter, and stay in the record of a systematic log. In this paragraph, the utterance of this robot is improved to the utterance to hear all agents.

お好み焼き協調料理では、ロボットの発言はログ出力と言う形をとります。これは、実際にロボットは何も発話しておらず、体系的なログの記録に留まります。本項では、このロボットの発話を、全エージェントが聞こえる発話に改善します。

5.7.1.1. broadcastMessage

BroadcastMessage does the utterance that all agents who are to be near hear. Each agent receives this utterance with onRecvMessage. The format of API is recorded as follows.

broadcastMessage は、近くにいるエージェント全てに聞こえる発話を行います。各エージェントは、この発話を onRecvMessage で受け取ります。以下に API の書式を記します。

Argument	I/O	Type	Explanation of argument	Example
1 st argument	IN	int	The number of sentences of sent messages is specified. = It is two for {"1", "2"}. 送るメッセージの文数を指定します。例えば、char *msg[] = {"1", "2"}の場合は、2です。	1, 2
2 nd argument	IN	char**	The sent message is specified. = It is {"Hello", "How are you"}. 送るメッセージを指定します。例えば、char *msg[] = {"Hello", "How are you"}です。	"a", "b"

The example of describing broadcastMessage is recorded as follows. For instance, when this message reaches, Avator outputs the content in the log. Agents (It is significant) other than Avator do not output the log oppositely. As a result, only the utterance to have heard Avator can be displayed to SIGViewer.

以下に broadcastMessage の記述例を記します。例えば、アバタはこのメッセージが届いた場合にその内容をログ出力します。逆にアバタ以外のエージェントは(意味のある)ログを出力しません。それにより、SIGViewer はアバタが聞こえた発話のみの表示が行えます。

Code Ex.	<pre>#define ARY_SIZE(ARY) ((int)(sizeof(ARY)/sizeof(ARY[0]))) char *msgs[] = {"Hello!!", "How are you"}; broadcastMessage(ARY_SIZE(msgs), msgs);</pre>	}	I utter a specified objection. 指定の文言を発話します
----------	---	---	---

5.7.1.2. TransitUtterance(TransitUttrAndAction/TransitUttrAndReject)

The remark of the robot is described in "transit/TransitUtterance.cpp" because it corresponds to "Utterance" by the state transition processing. "LOG_MSG" in the source is a log output. This LOG_MSG is changed to broadcastMessage. Mended points are the same though there are "TransitUttrAndAction.cpp" and "TransitUttrAndReject.cpp" besides "transit/TransitUtterance.cpp". The change example is recorded as follows.

ロボットの発言は、状態遷移処理で「Utterance」に該当する為、「transit/TransitUtterance.cpp」に記述されています。そのソース内の「LOG_MSG」がログ出力です。この LOG_MSG を broadcastMessage に変更します。「transit/TransitUtterance.cpp」以外にも、「TransitUttrAndAction.cpp」と「TransitUttrAndReject.cpp」がありますが、直す要領は同じです。以下に変更例を記します。

Revise Ex.	<pre>LOG_MSG((" %s said %s", myname(), comment));</pre>	}	It ..utterance.. mends. 発話する処理に直します
	<pre>broadcastMessage(1, (char*)&comment);</pre>		
			TransitUtterance.cpp on transit

To catch it, onRecvMessage is this time changed. OnRecvMessage is in "OkonomiyakiController.cpp". As for this onRecvMessage, doRecvMessage that first processes fixed form "Move=" etc. is done. Afterwards, the state transition to "MixDough" etc. sent from SIGViewer is executed. When it doesn't correspond to either the above-mentioned afterwards, this utterance outputs the log. In addition, to output only the utterance that Avator heard, the condition only for Avator is applied. The change example is recorded as follows.

今度は聞き取る為に、onRecvMessageを変更します。onRecvMessageは、「custom/OkonomiyakiController.cpp」にあります。このonRecvMessageはまず定型的な「Move=」等処理するdoRecvMessageが行われます。その後、SIGViewerから送信される「MixDough」等に対する状態遷移を実行しています。今回の発話は、その後に上記のいずれにも該当しない場合にログ出力します。更にアバタが聞いた発話のみを出力する為に、アバタの場合のみの条件もつけます。以下に変更例を記します。

```

Revise Ex.  if (value == "PutOutTheFireOfTeppan") {
              if (PutOutTheFireOfTeppan(value.c_str()) == false) return;
              proc = true;
            }
Append      if (atoi(avator) == 1) {
              LOG_MSG("%s said '%s'", evt.getSender(), evt.getString(0));
              proc = true;
            }
            if (proc == false) LOG_MSG("%s said 'Sorry, I do not know action for %s'", myname(), value.c_str());
    
```

Only the utterance that Avator heard is output to the log
アバタが聞いた発話をログに出力します

OkonomiyakiController.cpp on custom

The objection that only Avator heard is output to the log. When this log corresponds to "Robot_000 said '*****'", okonomiyaki CUI executes superimpose and the voice output on the screen.

アバタのみが聞いた文言をログに出力します。お好み焼きCUIは、このログが「Robot_000 said '*****」に該当する場合、画面上のスーパーインポーズと音声出力を実施します。

Operation Ex.

The image shows a 3D virtual environment with a person and a robot. A speech bubble from the robot says "I take oil". Below the scene is a System log window showing messages, with the line "Robot_000(127.0.0.1) Robot_000 said 'I take oil'" highlighted. A large blue text overlay "I take oil" is shown on the screen, and a small speech bubble "(Voice)" is also present.

A1. お好み焼き協調料理のインストール

In this paragraph, it explains the installation of the software that relates to the Okonomiyaki cooperation dish. All the software that relates to Okonomiyaki ends if the software set up is executed. The method of installing each software is written as follows.

この項では、お好み焼き協調料理に関連するソフトウェアのインストールを説明します。お好み焼きに関連する全てのソフトウェアは、セットアップするソフトウェアを実行すると終わります。以下に各ソフトウェアのインストール方法を記します。

A1.1. お好み焼き協調料理のインストール(サーバ側)

The Okonomiyaki cooperation dish is installed in the server into which Central Server (SIGVerse server) has been introduced. Therefore, The mount is done directly to the server or installation DVD is copied and the "/setup/okonomi_server" directory of installation DVD is copied onto an arbitrary directory on the server.

お好み焼き協調料理は、セントラルサーバが導入されているサーバにインストールします。その為、インストール DVD を直接サーバにマウントするか、インストール DVD の「/setup」ディレクトリをサーバ上の任意のディレクトリにコピーします。

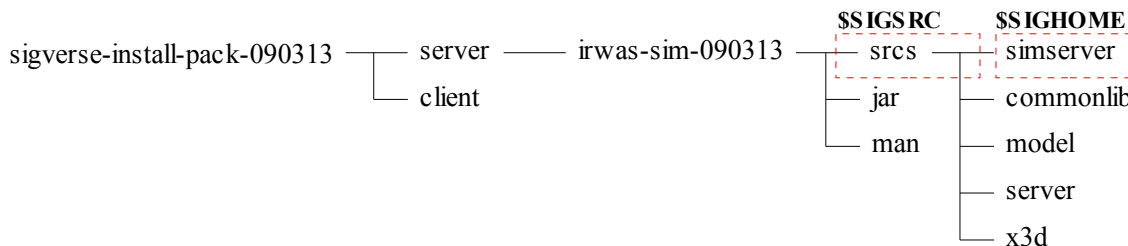
A1.1.1. 環境の前提条件

When SIGVerse has already been set up, it explains the installation destination as "\$SIGHOME". For instance, when SIGVerse "sigverse-install-pack-090313.tar.gz" is installed in "/home/nii", "\$SIGHOME" is "/home/nii/sigverse-install-pack-090313/server/irwas-sim-090313/srcs/simserver".

既に SIGVerse がセットアップされている場合、そのインストール先を「\$SIGHOME」として説明します。例えば、SIGVerse「sigverse-install-pack-090313.tar.gz」を「/home/nii」にインストールした場合、「\$SIGHOME」は「/home/nii/sigverse-install-pack-090313/server/irwas-sim-090313/srcs/simserver」です。

Moreover, it explains the directory of the source code of SIGVerse as "\$SIGSRC". For instance, when SIGVerse "sigverse-install-pack-090313.tar.gz" is installed in "/home/nii", "\$SIGHOME" is "/home/nii/sigverse-install-pack-090313/server/irwas-sim-090313/srcs". The example of figure is written as follows.

また、SIGVerse のソースコードのディレクトリを「\$SIGSRC」として説明します。例えば、「sigverse-install-pack-090313.tar.gz」を「/home/nii」にインストールした場合、「\$SIGHOME」は「/home/nii/sigverse-install-pack-090313/server/irwas-sim-090313/srcs」です。以下に、図の例を記します。



A1.1.2. お好み焼き協調料理のコンパイル

The Okonomiyaki cooperation dish is compiled according to the following procedures.

お好み焼き協調料理のコンパイルは、以下の手順で行います。

A1.1.2.1. お好み焼き協調料理の解凍

"release_20100324_command.tar.gz" of installation DVD is defrosted with \$SIGHOME.

インストール DVD の「release_20100324_command.tar.gz」を「\$SIGHOME」上で解凍します。

Operation Ex.	cd \$SIGHOME tar xzvf \$DVEDMNT/release_20100324_command.tar.gz
---------------	--

A1.1.2.2. お好み焼き協調料理のコンパイルの為の設定

"\$SIGHOME/command/make.sh" is matched to your environment and it sets it.

「\$SIGHOME/command/make.sh」を貴方の環境に合わせて設定します。

```
Operation Ex.          vi $SIGHOME/command/make.sh
```

Environment variable "SRCDIR" in "make.sh" is changed to your environment. This is the same as "\$SIGSRC".

「make.sh」の中の環境変数「SRCDIR」を貴方の環境に変更します。これは「\$SIGSRC」と同じです。

```
Setting Ex.           export SRCDIR=/home/nii/sigverse-install-pack-090313/server/irwas-sim-090313/srcs/
```

"make.sh" is executed.

「make.sh」を実行します。

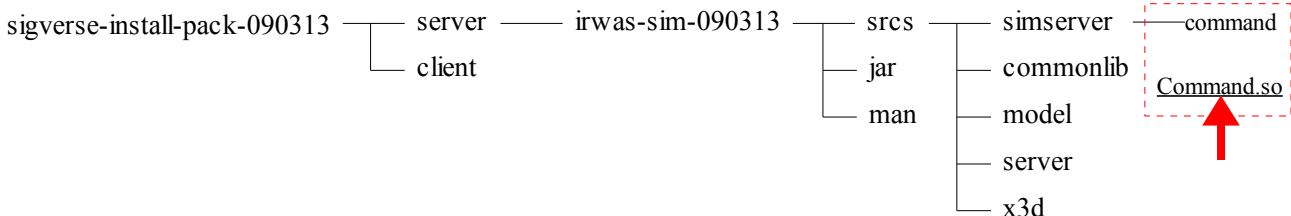
```
Operation Ex.          ./make.sh
```

"\$SIGHOME/command/Command.so" is confirmed.

「\$SIGHOME/command/Command.so」を確認します。

```
Operation Ex.          cd $SIGHOME/command
                      ls -l
```

```
Confirmation Ex.      12 22 16:35 Command.so ← confirm
```



The compilation is normally completed when there is "Command.so".

「Command.so」がある場合、コンパイルは正常に完了しています。

A1.1.3. お好み焼き協調料理の環境の設定

The configuration file of the Okonomiyaki cooperation dish has been introduced simultaneously with the decompression of "release_20100324_command.tar.gz". It is confirmed whether the following directories are in "\$SIGHOME/command".

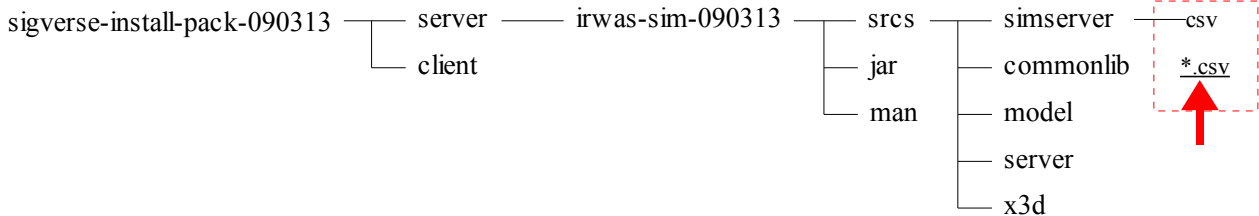
お好み焼き協調料理の設定ファイルは、「release_20100324_command.tar.gz」の解凍と同時に導入されています。以下のディレクトリが「\$SIGHOME/command/csv」にあるか確認します。

```
Operation Ex.          cd $SIGHOME
                      ls -l
```

```
Confirmation Ex.      11 27 14:03 csv } ← confirm *it is deictory
```

```
Operation Ex.          cd $SIGHOME/csv
                      ls -l
```

Confirmation Ex.	Abura_000_attribute.csv	Nori_000_condition.csv	} ← confirm *it is State transition files
	Abura_000_condition.csv	Nori_000_state.csv	
	Abura_000_state.csv	Nori_000_transition.csv	
	⋮		



The setting is normally completed when there is "*.csv".

「*.csv」がある場合、設定は正常に完了しています。

A1.1.3.1. セントラルサーバの設定の解凍

"release_20100324_conf.tar.gz" of installation DVD is defrosted with \$SIGHOME.

インストール DVD の「release_20100324_conf.tar.gz」を「\$SIGHOME」上で解凍します。

Operation Ex.	<pre>cd \$SIGHOME tar xzvf \$DVMNT/release_20100324_conf.tar.gz</pre>
---------------	---

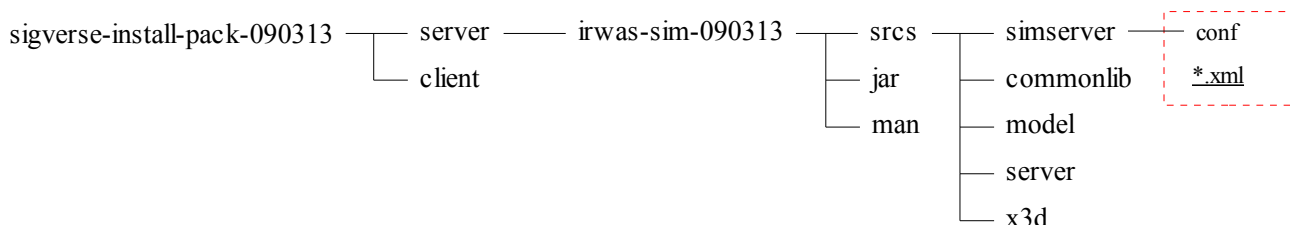
A1.1.3.2. セントラルサーバの環境の設定

"\$SIGHOME/conf" The setting has ended normally when directory under "XML file" is made.

「\$SIGHOME/conf」ディレクトリの下に「XML ファイル」が作成されていた場合、設定は正常に終了しています。

Operation Ex.	<pre>cd \$SIGHOME/conf ls -l</pre>
---------------	------------------------------------

Confirmation Ex.	Agent.xml	Entity.xml	Katsuobushi.xml	Nori.xml	} ← confirm
	TeppanFuta.xml	Avator-x3d.xml	Floor.xml	Katsuobushi.xml	
	Avator.xml	Ginger.xml	Okonomi.xml	nii_man.x3d	
	Bowl.xml	GingerLid.xml	nii_newman.x3d		
	⋮				



The content of "\$SIGHOME/conf/MyWorld.xml" of the definition of the virtual world is confirmed.

"MyWorld.xml" is edited with the text editor such as vi for the text form.

仮想の世界の定義の「\$SIGHOME/conf/MyWorld.xml」の内容を確認します。「MyWorld.xml」はテキスト形式の為、viなどのテキストエディタで編集します。

Operation Ex.	<pre>cd \$SIGHOME/conf vi MyWorld.xml</pre>
---------------	---

```
Confirmation Ex.      <set-attr-value name="language"    value="c++"/>
                      <set-attr-value name="implementation" value="command/Command.so"/>
```

} ← confirm

When "command/Command.so" is specified for all of agents' definitions, the setting is normal. (There is an agent who doesn't want "command/Command.so". It need not specify "command/Command.so".)

全てのエージェントの定義に「command/Command.so」が指定されている場合、設定は正常です。（「command/Command.so」が必要なエージェントがあります。それは「command/Command.so」を指定する必要はありません。）

When CentralServer starts after these settings, all the settings are normal.

これらの設定の後、CentralServer が起動した場合、全ての設定は正常です。

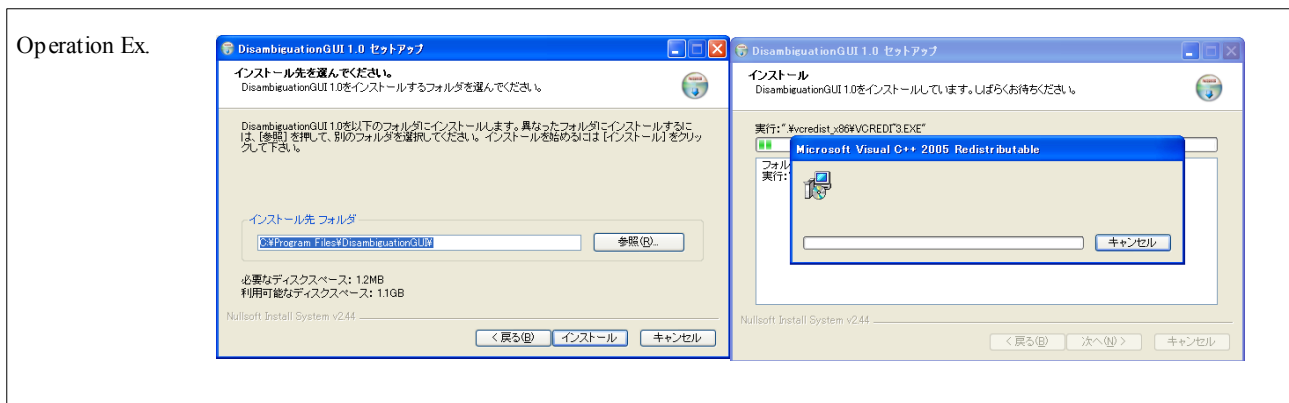
A1.2. お好み焼き GUI のインストール(クライアント側)

This chapter explains the method of installing "Okonomiyaki GUI". "Okonomiyaki GUI" executes and installs the installation program as much as general Window applications.

この章では、「お好み焼き GUI」のインストール方法を説明します。「お好み焼き GUI」は一般的な Window アプリケーションと同様に、インストールプログラムを実行してインストールします。

"OkonomiyakiGUI.exe" of installation DVD is executed. And all software is installed.

インストール DVD の「OkonomiyakiGUI.exe」を実行します。それで全てのソフトウェアがインストールされます。



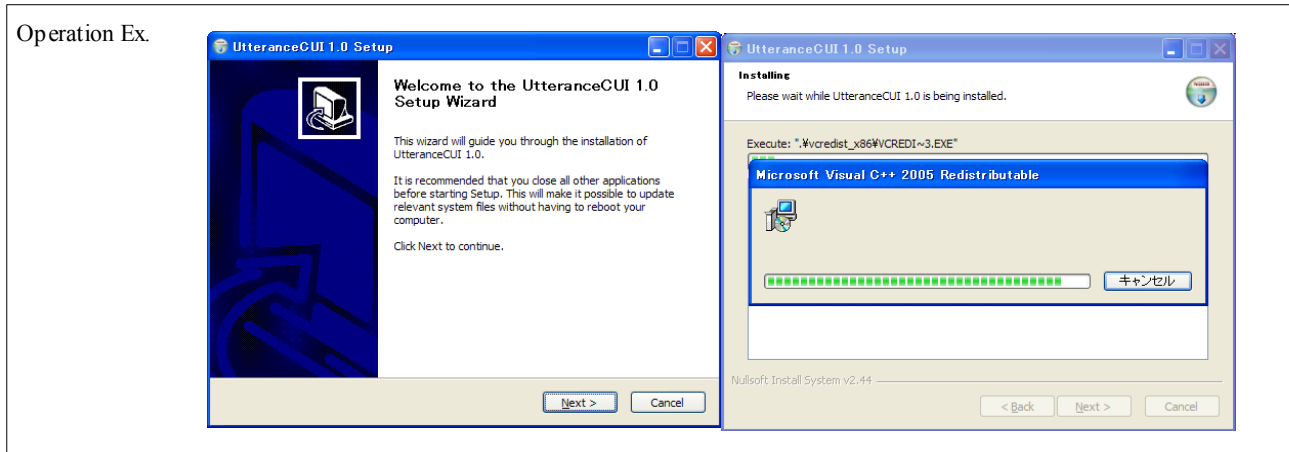
"OkonomiyakiGUI" is executed from the start menu of Windows. When "OkonomiyakiGUI" starts normally, the installation is normally completed.

Windows のスタートメニューから「OkonomiyakiGUI」を実行します。正常に「OkonomiyakiGUI」が起動した場合、インストールは正常に完了しています。

A1.3. お好み焼き CUI のインストール(クライアント側)

"OkonomiyakiCUI.exe" of installation DVD is executed. And all software is installed.

インストール DVD の「OkonomiyakiCUI.exe」を実行します。それで全てのソフトウェアがインストールされます。



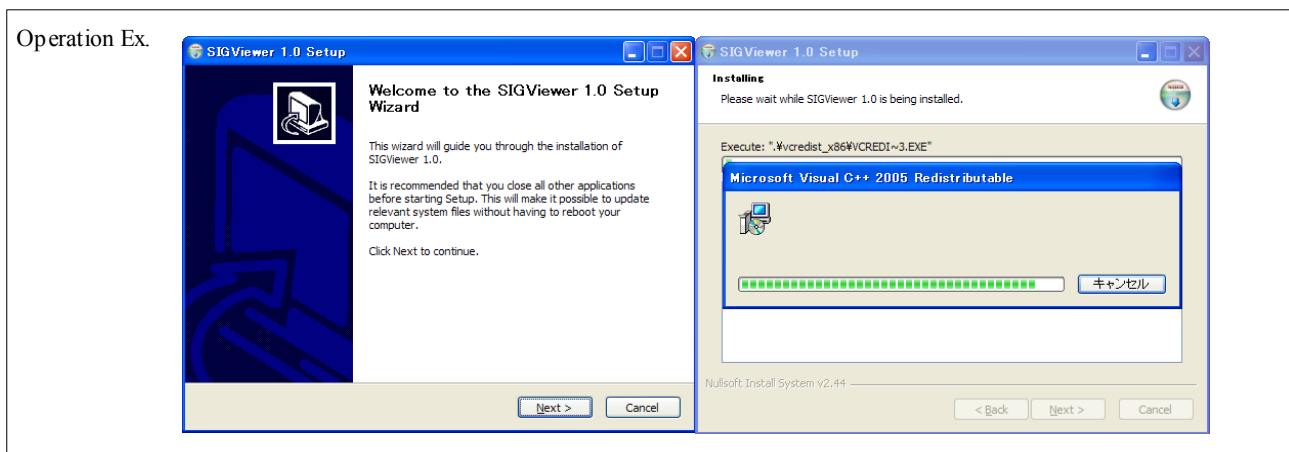
"OkonomiyakiCUI" is executed from the start menu of Windows. When "OkonomiyakiCUI" starts normally, the installation is normally completed.

Windows のスタートメニューから「OkonomiyakiCUI」を実行します。正常に「OkonomiyakiCUI」が起動した場合、インストールは正常に完了しています。

A1.4. SIGViewer のインストール(クライアント側)

"SIGViewer.exe" of installation DVD is executed. And all software is installed.

インストール DVD の「SIGViewer.exe」を実行します。それで全てのソフトウェアがインストールされます。



"SIGViewer" is executed from the start menu of Windows. When "SIGViewer" starts normally, the installation is normally completed.

Windows のスタートメニューから「SIGViewer」を実行します。正常に SIGViewer が起動した場合、インストールは正常に完了しています。

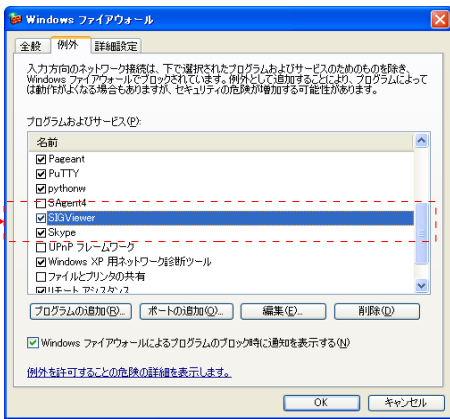
A2. トラブルシューティング

A2.1. 起動に関するトラブルシューティング

A2.1.1. セントラルサーバに接続ができない

Has the connection of the 9000th or the 8000th been permitted by the port defense by the anti virus or the firewall of Windows? Or, as for the port of the server that introduces the SIGVerse Central server, has the connection been permitted by 9000 fee counters?

アンチウィルスによるポート防御、または Windows のファイアウォールで 9000 番または 8000 番での接続は許可されていますか？ または、SIGVerse セントラルサーバを導入したサーバのポートは 9000 番台で接続が許可されていますか？

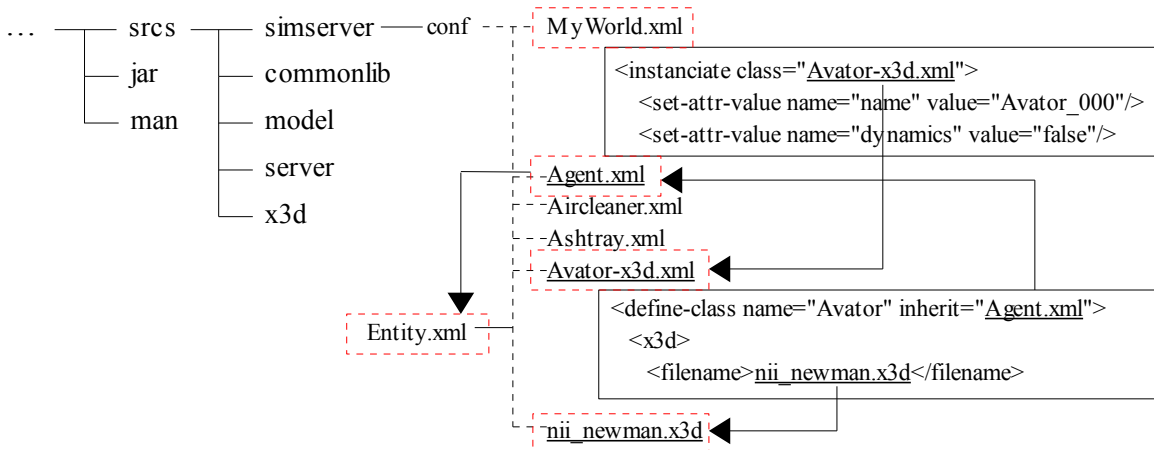
<p>Windows Ex. (Client firewall)</p> 	<p>Iptables Ex. (Server firewall)</p> <pre>[root@cobalt ~]# iptables --list Chain INPUT (policy ACCEPT) target prot opt source destination Chain FORWARD (policy ACCEPT) target prot opt source destination Chain OUTPUT (policy ACCEPT) target prot opt source destination</pre>
---	---

A2.1.2. セントラルサーバが起動できない

A2.1.2.1. No name または No attribute と表示される場合

Does "Robot-x3d.xml" etc. written in MyWorld.xml include in \$\$SIGHOME/conf? Moreover, is the file such as "nii_robot.x3d" specified in them "Robot-x3d.xml" in this directory? When Robot-x3d.xml etc. are edited and the variable is added, is this variable added to "Entity.xml"?

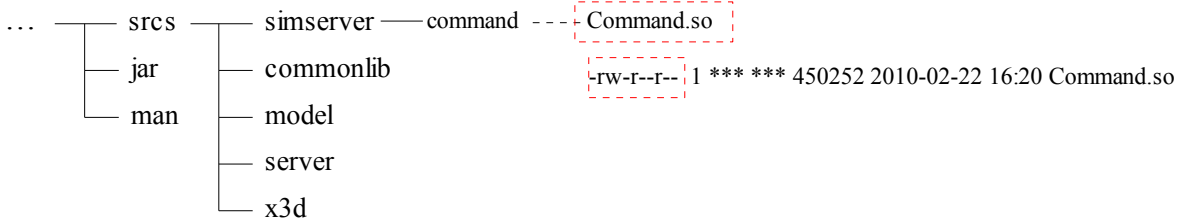
MyWorld.xml 内に書かれている「Robot-x3d.xml」等が \$\$SIGHOME/conf 内にありますか？ またそれら「Robot-x3d.xml」内で指定されている「nii_robot.x3d」等のファイルは同ディレクトリにありますか？ もし Robot-x3d.xml などを編集して変数を追加した場合、「Entity.xml」にも同変数を追加していますか？



A2.1.2.2. エージェントが起動しない

Is "Command.so" in MyWorld.xml in \$SIGHOME/command? Or, is the access authority given? The server used will be 32 bit operation system. In that case, we will recommend the recompile of the okonomiyaki cooperation dish.

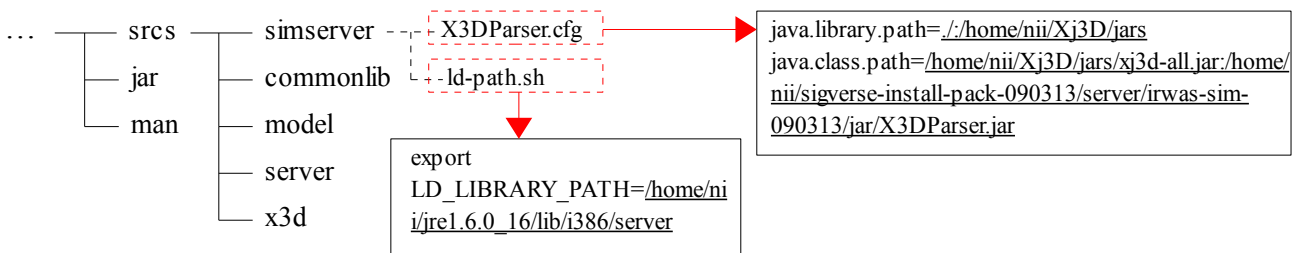
MyWorld.xml 内の「Command.so」は\$SIGHOME/command 内にありますか？ またはアクセス権限が与えられていますか？ お使いのサーバは 32 ビットオペレーションシステムではありませんか？ その場合は、お好み焼き協調理の再コンパイルをお勧めします。



A2.1.2.3. ヒューマノイドタイプ(身体動作が可能)のエージェントが起動しない

Is the specification of the execution environment of Java correct? Please confirm the content of "\$SIGHOME/X3DParser.cfg" and "\$SIGHOME/ld-path.sh".

Java の実行環境の指定は正しいですか？ 「\$SIGHOME/X3DParser.cfg」と「\$SIGHOME/ld-path.sh」の内容を確認して下さい。



A2.1.2.4. エージェントの位置が正しくないと警告される

Is not existing ODE that has been introduced into Linux-OS used? Only the binary compiled by the double precision specification can be used for ODE with SIGVerse. When existing ODE is used, ODE (source version) is downloaded from the following web pages, the compilation of which it is effective is double precision is done, and it installs it in the server.

Linux-OS に導入されている既存の ODE を使用していませんか？ ODE は倍精度指定でコンパイルしたバイナリだけが SIGVerse で利用可能です。もし、既存の ODE をお使いの場合は、以下のウェブページから ODE (ソース版) をダウンロードし、倍精度を有効とするコンパイルを行い、サーバにインストールします。

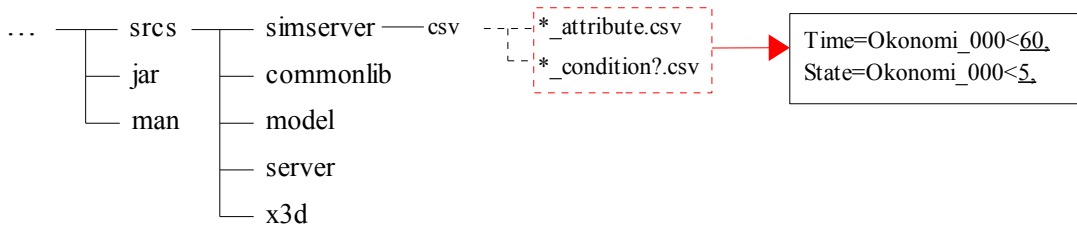
```
ODE Configure Ex. ./configure --enable-double-precision
```

```
Download http://sourceforge.net/projects/opende/
```

A2.1.2.5. エージェントが起動と同時にダウンする

Is not the character input to the numeric input item in various CSV configuration files? Especially, please note the condition related to the bigness and smallness of *_attribute.csv and *_condition.csv.

各種の CSV 設定ファイル内の数値入力項目に、文字を入力していませんか？特に*_attribute.csv と*_condition.csv の大小関係条件にご注意をお願いします。



A2.1.2.6. ポート番号が重複していると警告される

A multiple start by the same port number cannot be done. Or, the Central server that works by the port number that starts by the kill command etc. is stopped so that only the agent might keep starting. it starts again.

同じポート番号での多重起動は出来ません。またはエージェントのみが起動し続けている場合がある為、kill コマンドなどで起動したいポート番号で動作しているセントラルサーバを停止します。その後、再度起動を行います。

A2.1.3. お好み焼き GUI が起動しない

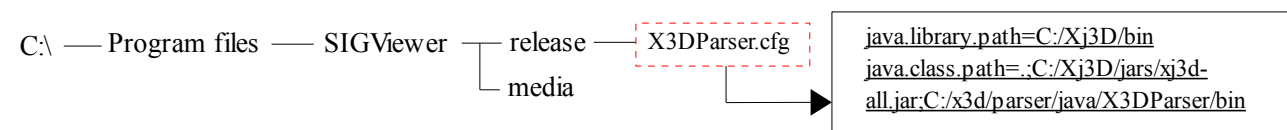
Please permit it if use is being prohibited by the anti virus software. Okonomiyaki GUI transmits any neither individual information nor environmental information to the outside. Okonomiyaki GUI is made based on Microsoft.NET ver3.5. We will recommend the introduction that uses introduction of this execution environment or setup DVD.

もしアンチウィルスソフトに利用を禁止されている場合、許可を与えて下さい。お好み焼き GUI はいかなる個人情報・環境情報も外部に送信しません。お好み焼き GUI は、Microsoft.NET ver3.5 を元に作成されています。同実行環境の導入かセットアップ DVD を用いての導入をお勧めします。

A2.1.4. SIGViewer が起動しない

Is graphics card (GPU) set up in PC used? It is likely not to operate well for onboard GPU. Or, is the specification of the execution environment and the use library correct in Java? Please confirm the content of "X3DParser.cfg".

お使いの PC にグラフィックスカード (GPU) は設置されていますか？オンボード GPU の場合、うまく作動しない場合があります。または、Java に実行環境及び利用ライブラリの指定は正しいですか？「X3DParser.cfg」の内容を確認して下さい。



A2.1.5. お好み焼き CUI が起動しない

Please permit it if use is being prohibited by the anti virus software. Okonomiyaki CUI transmits any neither individual information nor environmental information to the outside. Okonomiyaki CUI is made based on Microsoft.NET ver3.5. We will recommend the introduction that uses introduction of this execution environment or setup DVD. Okonomiyaki CUI might not operate correctly when the operation system that uses it is Windows 95, WindowsMe, and WindowsNT.

もしアンチウィルスソフトに利用を禁止されている場合、許可を与えて下さい。お好み焼き CUI はいかなる個人情報・環境情報も外部に送信しません。お好み焼き CUI は、Microsoft.NET ver3.5 を元に作成されています。同実行環境の導入かセットアップ DVD を用いての導入をお勧めします。もしお使いのオペレーションシステムが Windows95、WindowsMe、WindowsNT の場合、お好み焼き CUI は正しく動作しない場合があります。

A2.2. 操作に関するトラブルシューティング

A2.2.1. お好み焼き GUI で「Intialize」鈕を押下してもお好み焼きの見た目が変わらない

Externals might not change in the timing of the display. Please press the button in that case several times "Initialize".

表示のタイミングで見た目が変わらない場合があります。その場合は何度か「Initialize」鈕を押下して下さい。

A2.2.2. お好み焼き GUI で指定した動作が行われない

The operation of Avator is operated according to the rule of the state transition. Therefore, even if the operation etc. dished up to the plate are demanded immediately after okonomiyaki was burnt, it is likely not to be processed.

アバタの操作に関しても状態遷移のルールに従い動作しています。その為、お好み焼きを焼いた直後に、皿に盛り付ける操作などを要求しても処理されない場合があります。

A2.2.3. お好み焼き GUI で操作していると、海苔などが鉄板の上に残る場合がある

For instance, seaweed is put on the place when other operations are demanded before a series of operation that puts seaweed ends and it gives priority to the operation of the demand. Please wait for the operation for a little while until the operation of Avator ends.

例えば、海苔をかける一連の操作が終わる前に、他の操作を要求した場合、海苔をその場に置いて要求の動作を優先します。操作はアバタの動作が終わるまで少しの間お待ち下さい。

A2.2.4. お好み焼き GUI でお好み焼きの見た目が変わらない、回転が行えない、移動が行えない場合がある

"Start" The operation of the okonomiyaki cooperation dish is given to priority more than the operations through SIGViewer after the dish is started pressing the button. In that case, please initialize it pushing the "initialize" button.

「Start」鈕を押下して料理を始めてからは、SIGViewer を通しての操作よりお好み焼き協調料理の動作が優先されます。その場合は、「initialize」鈕を押して初期化して下さい。

A2.3. 設定に関するトラブルシューティング

A2.3.1. 状態遷移の設定で 0→1、1→0 と設定した場合、0 と 1 の遷移を繰り返してしまう

AvatorFlag of "*_attribute.csv" to which it gives priority to the existing state of things when the state transition condition is added to the transition of 0→1 or there are the state transition ahead plurals is specified for one. this *_ It gives priority excluding the existing state of things when RobotFlag of attribute.csv is adjusted to one or there are the state transition ahead plurals.

0→1 の遷移に状態遷移条件を付加するか、状態遷移先が複数ある場合に現在の状態を優先する「*_attribute.csv」の AvatorFlag を 1 に指定します。同*_attribute.csv の RobotFlag を 1 にした場合、状態遷移先が複数ある場合、現在の状態以外を優先します。

A2.3.2. 状態遷移を設定した場合、警告が SIGViewer またはサーバのコンソールに表示されて処理されない

The description of the state transition condition and the state transition processing might be warned of that the specification such as "@" is invalid in a certain processing because of the correspondence in the specification. Please continue your favors toward externals in which the description can be mended as shown in warning I am sorry in this case.


状態遷移条件、状態遷移処理の記述は仕様上の整合性の為に、ある処理では「@」等の指定は無効と警告する場合があります。この場合は、申し訳ありませんが、警告の通りに記述を直して頂けます様、よろしく申し上げます。

Robot_000-configure Ex	State=1@Robot_000		State=1	For instance, this is the same definition. "@" It removes from now on. 例えば、これは同じ定義です。「@」以降を取り除きます。
------------------------	-------------------	---	---------	---

A2.3.3. 状態遷移処理を空に設定した場合、何も動作しなくなる

The end condition cannot be specified when setting it to the sky and the state transition might not end. Please be not empty, and, for instance, specify ""State=1@Robot_000"" etc. and what loans for the state transition processing for state 1.

空に設定した場合、終了条件が特定出来ずに状態遷移が終わらない場合があります。状態遷移処理は空ではなく、例えば状態 1 の場合は「"State=1@Robot_000"」等と何かしらの指定をお願いします。

*_transition.csv Ex.	“0”, “” “1”, “”		“0”, “State=0” “1”, “State=1”	Please be not empty and specify something like the example for safety. 安全の為に、空ではなく例の様に何かを指定してください。
----------------------	--------------------	---	----------------------------------	---

A2.3.4. 状態遷移に関する情報自体が読み込まれていない1

The definition files concerning the state transition assume making with the Linux base. For instance, CR and ..changing line.. character-code require UTF-8 (BOM none). Please confirm file formats of the definition files if the definition concerning the state transition has not adjusted at all.

状態遷移に関する定義ファイル類は、Linux ベースでの作成を想定しています。例えば、改行は CR、文字コードは UTF-8 (BOM なし) を前提としています。もし状態遷移に関する定義が全く適応されていない場合、定義ファイル類のファイルフォーマットをご確認下さい。

A2.3.5. 状態遷移に関する情報自体が読み込まれていない2(発言が途中までしか出力されない)

"\" (backing slash)" and "Space" character cannot be used for the definition file concerning the state transition because of the character-code. Please replace with "_" and use "Space" without using "\".

状態遷移に関する定義ファイルは、文字コードの関係上、「¥ (バックスラッシュ)」と「スペース」文字が利用出来ません。「¥」は用いずに、「スペース」は「_」に置き換えてご利用下さい。

A2.3.6. 身体動作で自分自身の座標の移動を定義したが、移動しない

Oneself cannot adjust the movement of the prescribed agent who can specify it in the bodily movement. In this case, please describe the movement requirement in the state transition processing.

身体動作内で指定可能な所定エージェントの移動は、自分自身には適応出来ません。この場合は、状態遷移処理に移動要求を記述してください。

A2.3.7. 身体動作で回転と移動を同時に定義したが、何も動作しない

The movement and the rotation cannot be defined at the same time in the bodily movement. The purpose of each agent is not to process the movement and the rotation at the same time though the demand is transmitted. In this case, please clarify and define the order separately for two bodily movements.

身体動作内では移動と回転を同時に定義出来ません。要求は送信されますが、各エージェントは移動と回転を同時に処理しない為です。この場合、2つの身体動作に分けて順序を明確にして定義して下さい。

A2.3.8. 移動と回転を状態遷移処理と身体動作で複合的に定義したが、何も動作しない

The movement and the rotation cannot be defined at the same time in the bodily movement. The purpose of each agent is not to process the movement and the rotation at the same time though the demand is transmitted. In this case, please define it by either of the bodily movement or the state transition processing.

身体動作内では移動と回転を同時に定義出来ません。要求は送信されますが、各エージェントは移動と回転を同時に処理しない為です。この場合、身体動作か状態遷移処理のいずれかで定義して下さい。

A2.4. 改造に関するトラブルシューティング

A2.4.1. DetectEntities が動作しない場合がある

The log of the Central server is confirmed on the Central server. When connecting it with SIGViewer, it doesn't operate correctly when connected Internet Protocol address (*2) of the following connected Internet Protocol addresses (*1) and the service providers is different. We will recommend the client to have global IP at use in Intranet or the connection via the electronic environment.

セントラルサーバ上でセントラルサーバのログを確認します。SIGViewer との接続時に以下の接続 IP アドレス(*1)とサービスプロバイダの接続 IP アドレス(*2)が異なる場合、正しく動作しません。イントラネット内での利用、またはインターネット環境を経由しての接続の場合はクライアントがグローバル IP を持つ事をお勧めします。

```

Log Ex.  + ./simserver -p 9000
          [SYS] waiting for connection...
          [SYS] Controller attached to "Toy_000"
          [SYS] 127.0.0.1 connected
          [SYS] Toy_000 : dataport
          [SYS] 127.0.0.1 connected
          [SYS] attach view : polling
          [SYS] 192.168.3.4 connected(*1)
          [SYS] 192.168.3.4:8000 : service provider(*2)
          [SYS] 192.168.3.4 connected
    
```

The port number when these are connected should be corresponding
これらの接続時のポート番号は一致する必要があります。

A2.4.2. moveTo で指定した座標にエージェントが移動しない

There is a possibility that given force is few when the agent who executed moveTo stays in the place. Three times distance? Please specify about five times force.

moveTo を実行したエージェントがその場に留まる場合、与えている force が少ない可能性があります。距離の 3 倍～5 倍程度の force を指定して下さい。

A2.4.3. あるエージェントを複数起動したいが起動しない

When two or more agents are started with the display thoroughly (When you want to start the agent like the instance of the class), the definition to start up two or more kinds of same agents with MyWorld.xml is described. For instance, to arrange three robots in a virtual space, it defines it as follows.

同じ表示を持つエージェントを複数起動させる場合(クラスに対するインスタンスの様にエージェントを起動したい場合)、MyWorld.xml で同じ種類のエージェントを複数立ち上げる定義を記述します。例えば、ロボットを3台仮想空間内に配置したい場合は、以下の様に定義します。

```

Configure Ex. <instanciate class="Robot-x3d.xml">
                <set-attr-value name="name" value="Robot_001"/>
            </instanciate>
            <instanciate class="Robot-x3d.xml">
                <set-attr-value name="name" value="Robot_002"/>
            </instanciate>
            <instanciate class="Robot-x3d.xml">
                <set-attr-value name="name" value="Robot_003"/>
            </instanciate>
    
```

MyWorld.xml on \$SIGHOME/conf

A2.4.4. 状態遷移の契機タイミングをもう少し早くしたい

Definition "NORMAL_UNIT" "FAST_UNIT" "SMALL_UNIT" at the period related to the state transition is being done by core/Define.h. Timing of the opportunity of the state transition is $NORMAL_UNIT * SMALL_UNIT$ number of seconds. Please reduce $SMALL_UNIT$, improve the entire frequency or lower the value of $NORMAL_UNIT$ at the right time. It is appropriate every 0.1-0.3 second ..largeness.. somewhere else.

core/Define.h に状態遷移に関わる期間の定義「NORMAL_UNIT」「FAST_UNIT」「SMALL_UNIT」がされています。状態遷移の契機タイミングは $NORMAL_UNIT * SMALL_UNIT$ 秒数です。SMALL_UNIT を少なくして全体の頻度を高めるか、NORMAL_UNIT の値を適時下げて下さい。大よそ 0.1～0.3 秒間隔が適切です。

```
C++ Ex.  /*!
          * @brief Unit time interval of normal state
          * @brief 通常状態時のユニット時間間隔です
          */
          #define NORMAL_UNIT 5

          /*!
          * @brief It is a value of the interval of time to get the control from SIGVerse minimum.
          * @brief 最小単位の SIGVerse から制御をもらう時間間隔値です
          */
          #define SMALL_UNIT 0.1
```

Define.h on core/Define.h

A2.4.5. 身体動作・移動・回転の段階的な動作をもう少し細かく(粗く)したい

Frequency "HUMANOID" of phased operation is defined in core/define.h. It is five now. Please change at the right time and compile this value. However, it is necessary to consider the number of SIGViewer of drawing frames. The best value is found while confirming it by watching at the right time. 5-10 is appropriate ..largeness.. somewhere else.

core/define.h に段階的な動作の回数「HUMANOID」が定義されています。現在 5 です。この値を適時変更してコンパイルして下さい。但し、SIGViewer の描画フレーム数を考慮する必要があります。適時目視で確認しながら最適な値を見つけます。大よそ 5～10 が適切です。

```
C++ Ex.  /*!
          * @brief Angle ratio which can progress every call, of body motion of onAction
          * @brief 身体運動の onAction の 1 回コール毎に進める角度割合です
          */
          #define HUMANOID 5
```

Define.h on core/Define.h

A3. 状態遷移に関する記述一覧

A3.1. 登場人物一覧

Example of figure		All charactes(Agents)		
		Number in figure	Agent's Name	Name of agent who uses it on SIGViewer
		1	Avator	Avator_000
		2	Robot	Robot_000
		3	Oil	Abura_000
		4	Bowl	Bowl_000
		5	Iron plate	Teppan_000
		6	Okonomiyaki	Okonomi_000
		7	Sauce	Sauce_000
		8	Seaweed	Nori_000
		9	Katsuobushi	Katsuobushi_000
Character's name 登場人物名	Distinguished name 分類名	Predominant role 主な役割		
Avator アバタ	User 利用者	It is a leading part of the okonomiyaki cooperation dish, and it serves as user's taking the place on SIGVerse virtual space. お好み焼き協調料理の主役であり、SIGVerse 仮想空間上で利用者の代わりに務めます		
Robot ロボット	Robot ロボット	It operates autonomous, and the dish of Avator is assisted and it advises. 自律的に動作し、アバタの料理の補助やアドバイスなどを行います		
Okonomiyaki お好み焼き	Material 材料	It is an object of the dish, and a dish baked mixing a favorite tool material with the cloth. 料理の対象であり、好きな具材を生地に混ぜて焼き上げる料理です		
Pork 豚肉		It is one of the materials of okonomiyaki, and an indispensable material to the Kansai okonomiyaki. お好み焼きの材料の一つであり、関西お好み焼きに必須の材料です		
Sauce ソース	Seasoning 調味料	It is one of the seasonings put on okonomiyaki, and a source for peculiar okonomiyaki to Japan. お好み焼きにかける調味料の一つであり、日本特有のお好み焼き用のソースです		
Seaweed(Nori) 海苔		It is one of the seasonings put on okonomiyaki, and a seasoning of which the origin is peculiar seaweeds to Japan. お好み焼きにかける調味料の一つであり、日本特有の海草を元とした調味料です		
Katsuobushi 鰹節		It is a seasoning that one of the seasonings put on okonomiyaki it, ferments a peculiar fish(bonito) to Japan, and is dry. お好み焼きにかける調味料の一つであり、日本特有の魚を発酵させて乾燥させた調味料です		
Oil 油	Cooking utensil 調理器具	In the oil used when okonomiyaki is burnt on the iron plate, it paints it with the tool called a brush. お好み焼きを鉄板の上で焼く時に使う油で、刷毛と言う道具で塗ります		
Hera へら		It turns inside out, and the cooking utensil of the divided fork of okonomiyaki. お好み焼きを裏返ししたり、切り分けるフォークの様な調理器具です		
Teppan 鉄板		It is a cooking utensil of the board of the plastron to bake okonomiyaki. お好み焼きを焼き上げる鉄製の板の調理器具です		
Bowl ボウル		It is a container where the cloth before okonomiyaki is burnt is put. お好み焼きを焼く前の生地を入れておく容器です		

A3.2. 状態遷移条件

Name of condition 条件の名前	Content of condition 条件の内容	Description example 記述の例
State	When my state is specified, the state transition is permitted. 自分の状態が指定された状態の場合、状態遷移を許可します	State=2
	When other agents' states are specified, the state transition is permitted. 他のエージェントの状態が指定された状態の場合、状態遷移を許可します	State=2@Avator_000
	When either state of two agents is specified, the transition is permitted. 2つのエージェントのどちらかの状態が指定された状態である場合に遷移を許可します	State=2@Avator_000 Robot_000
	The transition is permitted for less than value that the state specified with State. <u>(*In this case, only "Numerical value" of the state name is effective.)</u> 状態が State で指定した値未満の場合に遷移を許可します ※この場合、状態名は「数値」のみが有効です	State=2>Avator_000
	When states are larger than the values specified with State, the transition is permitted. <u>(*In this case, only "Numerical value" of the state name is effective.)</u> 状態が State で指定した値より大きい場合に遷移を許可します ※この場合、状態名は「数値」のみが有効です	State=0<Avator_000
Time	After it changes in the state, the transition is permitted for less than specified time. <u>(*This time is set to 0 at the state transition.)</u> その状態に遷移してから指定時間未満の場合に遷移を許可します ※この時間は、状態遷移の度に0に設定されます	Time=100>Okonomi_000
	When it is larger than specified time after it changes in the state, the transition is permitted. <u>(*This time is set to 0 at the state transition.)</u> その状態に遷移してから指定時間より大きい場合に遷移を許可します ※この時間は、状態遷移の度に0に設定されます	Time=500<Okonomi_000
Attr	When own attribute value is a specified value, the transition is permitted. 自分自身の属性値が指定値の場合に遷移を許可します	Attr=1
	When the attribute value of a specified agent is a specified value, the transition is permitted. 指定エージェントの属性値が指定値の場合に遷移を許可します	Attr=2@Teppan_000
	The transition is permitted, except when own attribute is a specified value. 自分自身の属性が指定値以外の場合に遷移を許可します	NotAttr=2
Aid	When own supplementary level is a specified value, the transition is permitted. 自分自身の補助度が指定値の場合に遷移を許可します	Aid=2
	The transition is permitted, except when own supplementary level is a specified value. 自分自身の補助度が指定値以外の場合に遷移を許可します	NotAid=2
Uttr	When own remark level is a specified value, the transition is permitted. 自分自身の発言度が指定値の場合に遷移を許可します	Uttr=1
	The transition is permitted, except when own remark level is a specified value. 自分自身の発言度が指定値以外の場合に遷移を許可します	NotUttr=2

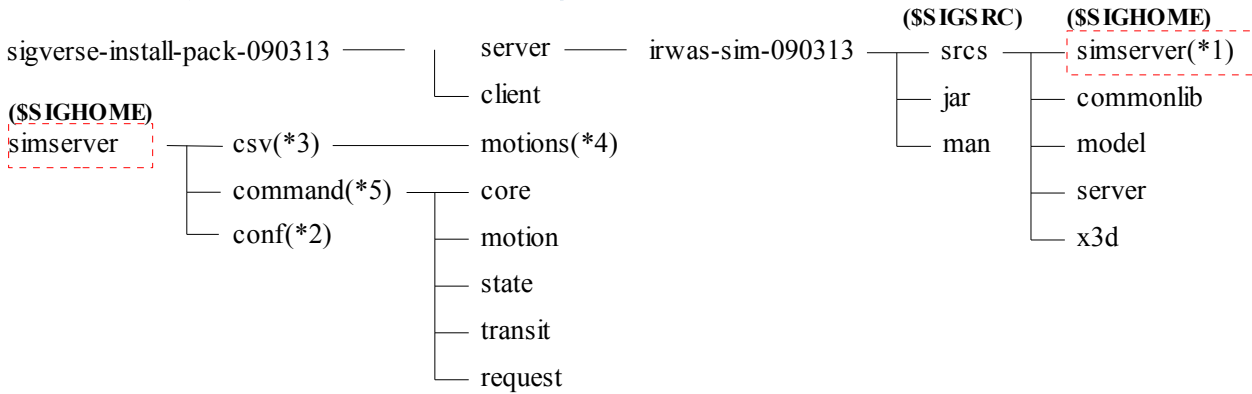
A3.3. 状態遷移処理

Name of proccessing 処理の名前	Content of processing 処理の内容	Description example 記述の例
State	It changes in the state to specify other agents' states. <u>However, when other agents do not meet the requirement of the transition of the state, the demand is disregarded.</u> 他のエージェントの状態を指定した状態に遷移します。但し、他のエージェントが状態の遷移の条件を満たさない場合、その要求は無視されます。	<u>State=1@Abura_000</u> (Oil is made used.) (油を使用済みにする)
Attr	It sets it to the attribute in which other agents' attributes are specified. For instance, when thermal power of the iron plate is set, it uses it. 他のエージェントの属性を指定した属性に設定します。例えば、鉄板の火力を設定する場合に使用します。	<u>Attr=2@Teppan_000</u> (Thermal power of the iron plate is made "It is strong".) (鉄板の火力を「強」にする)
Move	It moves to coordinates that specify other agents. Coordinates are described by the X:Y:Z form. 他のエージェントを指定した座標に移動します。座標は X:Y:Z 形式で記述します。	<u>Move=100:80:100@Sauce_000</u> (The source is moved on the iron plate.) (ソースを鉄板の上に移動する)
	It is moved while passing coordinates that specify other agents. Coordinates are described by "X:Y:Z X:Y:Z form". Many coordinates can be specified. 他のエージェントを指定した座標を経由しながら移動させます。座標は「X:Y:Z X:Y:Z 形式」で記述します。座標はいくつでも指定できます。	<u>Move=100:70:100 100:80:100@Sauce_000</u> (The source is moved on the iron plate and lifted.) (ソースを鉄板の上に移動して持ち上げる)
Angle	It rotates to the direction that specifies other agents. It describes it by "Direction (0,1) of X: direction (0,1) of Y: direction (0,1) of Z: the beginning angle: the end angle: increment angle" form. This is the same as the form operated with Operation. 他のエージェントを指定した向きに回転します。「X 方向(0,1):Y 方向(0,1):Z 方向(0,1):開始角度:終了角度:増分角度」形式で記述します。これは Operation で操作した形式と同じです。	<u>Angle=1:0:0:180:10@Okonomi_000</u> (Okonomiyaki is turned inside out.) (お好み焼きを裏返す)
Visual	Other agents' externals are changed. For instance, it puts some sauce on in Okonomiyaki. 他のエージェントの外見を変えます。例えばお好み焼きにソースをかけた場合などです。	<u>Visual=5@Okonomi_000</u> (Externals of Okonomiyaki are changed into "Externals on which the source is put".) (お好み焼きの外見を、「ソースがかかけられた見た目」に変える。)
Motion	Specification is operated. Please see "Definition of bodily movement(gesture)" about details. This is specification of the bodily movement. (指定の動作を行います。詳細は、「Definition of bodily movement(gesture)」をみてください。これは身体動作の指定です。)	<u>Motion=ges92r</u> (It reaches the bowl.) (ボウルに手を伸ばす)
Utterance	It makes remarks on a specified objection. <u>However please specify space for " _ ".</u> 指定の文言を発言します。但し、スペースは「_」に指定してください。	<u>Utterance=Hello</u> ("Hello" Make remarks.) (「Hello」と発言する)
	It makes remarks on the specified objection repeating after the passage of specified unit time. And, all Transition after this specification is disregarded. 指定の文言を発言します。指定した文言は、指定ユニット時間経過の後、繰り返して発言します。そして、この指定の後の Transition は全て無視されます。	<u>Utterance=Hello&20</u> (Make remarks being repeat in every "Hello" and 20 unit times.) (「Hello」と 20 ユニット時間間隔で繰り返して発言します)

Name of processing 処理の名前	Content of processing 処理の内容	Description example 記述の例
	<p>It makes remarks on a specified objection only first time to the state transition. 状態遷移した初回のみ、指定の文言を発言します。</p>	<p>Utterance=Hello&first (It is made remarks "Hello" only once in the state.) (その状態で一度のみ「Hello」と発言します。)</p>
	<p>It waits during the time of a specified unit. It makes remarks on a specified objection when the time of a specified unit passes, and Transition of continuation is executed. 指定ユニット時間の間、待ちます。指定ユニット時間が過ぎた場合、指定の文言を発言して、続きの Transition を実行します。</p>	<p>UtrrAndAction=GoodBye&10 (When 10 unit time passes, it is made remarks, "Goodbye".) (10 ユニット時間が経過した場合、「さようなら」と発言します。)</p>
	<p>When it meets a specified requirement, it makes remarks. Transition afterwards is not done. 指定の条件を満たす場合、発言します。その後の Transition は行いません。(アバタが1(油をとった)の場合、お礼を言います。自分はその後の Transition (油をとる)を行いません。)</p>	<p>UtrrAndReject=Thanks&1@Avator (When Avator's state is 1 "Oil was taken", the reward is said. I do not do Transition afterwards "Oil is taken".)</p>
Reject	<p>Processing is not executed for the state transition that has already been done. 既に行われた状態遷移の場合、処理を実行しません。</p>	Reject=AlwasState

A4. お好み焼き協調料理ソフトウェアの俯瞰

A4.1. お好み焼き協調料理のディレクトリ構造



Number 項番	Name of directory ディレクトリ名	Role of directory ディレクトリの役割	The main storage file 主な格納ファイル	Role of file ファイルの役割
(*1)	simserver	The Central server is stored. セントラルサーバを格納します	simserver.sh	The Central server (okonomiyaki cooperation dish) is started. セントラルサーバ(お好み焼き協調料理)を起動します
(*2)	conf	The definition of a virtual space and the agent's definition are stored. 仮想空間の定義、エージェントの定義を格納します	MyWorld.xml	Definition of virtual world 仮想世界の定義
			Robot-x3d.xml	Variable and model definition of robot ロボットの変数及びモデル定義
			nii_robot.x3d	Body model definition of robot ロボットの身体モデル定義
(*3)	csv	The CSV file related to the state transition of the okonomiyaki cooperation dish is stored. お好み焼き協調料理の状態遷移に関わる CSV ファイルを格納します	Robot_000_state.csv	Definition of state transition information on robot ロボットの状態遷移情報の定義
			Robot_000_condition.csv	Definition of state transition condition of robot ロボットの状態遷移条件の定義
			Robot_000_transition.csv	Definition of state transition processing of robot ロボットの状態遷移処理の定義
(*4)	motions	The data file related to the bodily movement of the okonomiyaki cooperation dish is stored. お好み焼き協調料理の身体動作に関わるデータファイルを格納します	ges92r	Definition of bodily movement that reaches oil of robot ロボットの油に手を伸ばす身体動作の定義
			ges92a	Definition of bodily movement that reaches oil of Avator ロボットの油に手を伸ばす身体動作の定義
(*5)	command	The C++ source of the agent of the okonomiyaki cooperation dish is stored. お好み焼き協調料理のエージェントの C++ソースを格納します	libcommand	Agent program for okonomiyaki cooperation dish (SharedObject) お好み焼き協調料理用のエージェントプログラム (SharedObject)

A4.2. お好み焼き協調料理ソフトウェア一覧

Name of software	Operating layer	Role of software
Central Server セントラルサーバ	Server side サーバサイド	Central Server controls the conversation with the calculation of mechanics with the main of the SIGVerse server. The user application moves on this software. セントラルサーバは力学の計算と対話を制御する SIGVerse の本体サーバです。ユーザアプリケーションは、このソフトウェア上で動作します。
The Okonomiyaki cooperation dish お好み焼き協調料理	Server side サーバサイド	This is a user application that controls the Okonomiyaki cooperation dish. Avator and Robot are managed by the unit called "Agent". これはお好み焼き協調料理を制御するユーザアプリケーションです。アバターやロボットは「エージェント」と呼ばれる単位で管理されます。
Service Provider(SIGViewer) サービスプロバイダ (SIGViewer)	Client side クライアントサイド	This controls perception. For instance, the list is made though Robot is seen. Moreover, Viewer of the simulation operates, too. This function is offered with software named SIGViewer. エージェントの視覚情報や音声情報を提供します。例えば、ロボットが見たもののリストを作成します。また、シミュレーションの動作の Viewer としての役割も持ちます。
The Okonomiyaki GUI お好み焼き GUI	Client side クライアントサイド	The Okonomiyaki GUI offers the operation of Avator. Initialization and beginning the scenario of the Okonomiyaki cooperation dish are done besides. お好み焼き GUI は、アバターの操作を提供します。それ以外にお好み焼き協調料理のシナリオの初期化と開始を行います。
The Okonomiyaki CUI お好み焼き CUI	Client side クライアントサイド	OkonomiyakiCUI is superimpose and outputs the voice of the remark of Robot. This corresponds to the assistance of the display. お好み焼き CUI は、スーパーインポーズやロボットの発言の音声出力を行います。これは表示の補助に相当します。

A4.3. お好み焼き協調料理の主たる操作

Operating type	Name of software	Method of operation
Beginning 開始	Central Server セントラルサーバ	\$SIGHOME/simserver.sh -p 9000
	The Okonomiyaki cooperation dish お好み焼き協調料理	
	Service Provider(SIGViewer) サービスプロバイダ (SIGViewer)	It begins from the start menu of Windows. Windows のスタートメニューから開始します
	The Okonomiyaki GUI お好み焼き GUI	
	The Okonomiyaki CUI お好み焼き CUI	
End 終了	Central Server セントラルサーバ	kill (Linux command)
	The Okonomiyaki cooperation dish お好み焼き協調料理	
	Service Provider(SIGViewer) サービスプロバイダ (SIGViewer)	It ends pushing the "x" button as well as a general Windows application. 一般的な Windows アプリケーションと同じく「x」鈕を押して終了します
	The Okonomiyaki GUI お好み焼き GUI	
	The Okonomiyaki CUI お好み焼き CUI	