Vol.2017-SLP-115 No.9
2017/2/18

実ロボット用音声対話システム開発環境の構築

鈴木 基之[1,a)]　岸田 拓也[1]

概要：実ロボットにおける音声対話システムは様々な場面で実用化されつつあるが，その発話や動作等を定義し，対話システムを構築するためには，各ロボットごとに個別に開発する必要がある。こうした作業は，ロボットの制御や音声情報処理等に関する深い知識が必要とされるため，容易ではない。
そこで本論文では，そうした知識のない人でも手軽に音声対話システムを開発できるような開発環境を構築する。手軽に利用可能な音声対話システムのひとつである MMDAgent をベースとし，その対話シナリオファイルを自動で変換することで，実ロボットで動作できるようにする。また対話文から自動でエージェントの動作を推定するシステムを組み合わせることで，自然な動作を行いながら対話するシステムを手軽に構築できるようにする。

キーワード：音声対話システム，シナリオエディタ，MMDAgent

Construnction of a new development environment for a spoken dialog sysytem in real robot

Abstract: A spoken dialog system in a real robot is being put to practical use in various situations. However,it is necessary to develop a spoken dialog system for each robot, and it requires a professional technology. In order to develop a spoken dialog system easily, the development environment for non-expert users is proposed in this paper. It is based on the MMDAgent spoken dialog system, and a dialog scenario file can be commonly used for the spoken dialog system in a real robot.
Moreover, the dialog scenario editor which can select an appropriate motion for the robot by using spoken text is combined with the development environment. It realizes to make a natural communication between human and robot.

Keywords: Spoken dialog system, Scenario editor, MMDAgent.

1. はじめに

近年，SoftBank の Pepper[1] や SHARP の RoBoHoN[2] 等，家庭用ロボットの実用化が現実味を帯びてきてきる。こうしたロボットにおいては，いかに自然にユーザとコミュニケーションをとるかが重要であり，音声対話システムを搭載することは必須となっている。

しかし，自然な音声対話システムを構築することは簡単ではない。各種最新技術を用いたとしても，ユーザの発話を 100%正確に認識し，その意味や意図を正しく理解した上で適切な応答を生成する，ということは実現できていない。そのため，どのような発話があった時にどのようなリアクションをとるのか，使用される場面やユーザの属性，ロボットの性格といった様々な要因を勘案した上で，最終的には技術者が細かく調整をしていく必要がある。

更に実ロボットを用いた音声対話においては，ロボット自身のジェスチャや表情，目線といった動作も非常に重要になる。これについても「こうすればよい」といった定番や理論等はなく，技術者が場面ごとに職人芸のようにして調整を行う必要がある。特に動作に関しては，ロボットによってモータの数や稼動域がそれぞれ異なるため，ハードウェアが変更になれば最初から作り直し，といった事態にもなりやすい。

こうした現状から，実ロボットを用いた音声対話システムの開発は一般の人々には敷居が高く，ロボットや音声情

[1] 大阪工業大学 情報科学部
Osaka Institute of Technology, 1-71-1, Kitayama, Hirakata, Osaka, 573–0196, Japan
[a)] moto@m.ieice.org

報処理技術者といった専門家がいなければ，参入障壁が高いといわざるを得ない。いくらハードウェアとしてのロボットが身近になったとしても，簡単に活用できるわけではないのである。

そこで本論文では，専門の知識を持たない人であっても，比較的手軽に音声対話システムを構築できる開発環境を提供することを目指す。簡単に対話シナリオを記述できる事に加え，ソフトウェアエージェントによる対話システムとシナリオフォーマットを共通化することで，開発段階ではソフトウェアエージェントで動作を確認し，ある程度シナリオが完成した段階で実ロボットによる動作確認をする，といった開発方法がとれるようにする。

2. 音声対話システムの開発環境

2.1 設計方針

本システムは，ロボットや音声情報処理の専門家でなくても手軽に実ロボットを用いた音声対話システムが構築できる事を目指す。一般に実ロボットを用いた音声対話システムでは，ロボットの動作制御と対話の制御を両方同時にこなす必要があるため，それらが一体になったプログラム開発となることが多い。しかし，この方法では，それぞれの制御に関する専門的な知識が必須となってしまう。そこで，ロボットの動作制御に関する事と，音声対話制御（対話シナリオ）を分離し，どのようなロボットであっても（動作定義等を別途行う事で）共通のシナリオを利用可能とする。

全体の動作記述としては，ソフトウェアエージェントによる音声対話システムである MMDAgent[3] のシナリオ記述フォーマットを採用する。このフォーマットはオートマトン制御による音声対話シナリオが記述可能であり，指定されたキーワードが認識されたら，何を発話し，どう動作するか，といった事を状態毎に記述することができる。この方式は高度な意図解析等は行わないため，複雑な対話の実現は難しいと思われるが，一方で助詞・助動詞等の誤認識の影響を受けないため，音声認識精度が高くない状況でも頑健な音声対話を実現することが可能となる。

この方式では，「この文を発話せよ」「（事前に定義した）この動作を再生せよ」といったレベルで音声対話やロボットの動作が記述される。そのため，音声情報処理の詳細（どうやって合成音声を生成するか等）やロボットの動作制御の詳細を知らなくても，音声対話シナリオを記述することが可能となる。

ロボットの動作に関しては，すべての動作について別途作成しておく必要があるが，それについては，ロボット製作者等が提供する動作の作成ソフトウェア等を利用し，事前に作成しておくものとする。

また，MMDAgent のシナリオ記述を用いることで，何の変更もせずに MMDAgent を用いた音声対話が可能となる。そのため，シナリオの作成途中で MMDAgent を用いて動作確認してみる，といったことが可能となる。一般に実ロボットを用いた音声対話システムの開発においては，ロボットを動作させる場所の確保やシステム立ち上げの手間，またロボットの台数による同時開発者数の制限等があり，手軽に開発する，とはいかない。一方ソフトウェアエージェントであれば，開発 PC さえあれば誰でもどこでも何人でも実行することができるため，特にシナリオの開発段階においては非常に重宝する。そこで本システムでは，音声対話シナリオを MMDAgent のシナリオ記述と共通フォーマットにすることで，開発段階は MMDAgent で動作確認をし，ある程度開発できた段階で実ロボットによる検証を行う，といった開発作業を可能とした。

本開発環境の特長は以下のとおりである。

- 対話シナリオはオートマトン制御
- 意図解析等はせず，キーワードに注目した対話管理
- MMDAgent を用いた手軽なシナリオの検証
- 音声認識や合成の詳細な知識は不要
- ロボットの動作は別途（GUI ツール等で）事前に作成
- 動作定義を別途行えば，他のロボットにも対話シナリオを転用可能

2.2 今回使用した実ロボット

今回システムを開発する上で，実ロボットとして Vstone 社の Sota[4] を用いた。Sota の外観を図 1 に示す。このロボットは腰から上だけのロボットであり，胴体の回転，肩と肘がそれぞれ 2 軸，首に 3 軸の，合計 8 自由度を持つ。また目のまわりと口をフルカラー LED で光らせることが可能であるが，表情等を変化させたり，唇を動かしたり，といったことはできない。

本体には計算機として Raspberry Pi が搭載されており，またモノラルマイクとスピーカも内蔵されている。また WiFi も装備され，外部 PC 等と通信することも可能である。

図 1 Sota の外観

2.3 音声対話システム開発環境の全体構成

本論文で提案する音声対話システム開発環境の全体構成を図 2 に示す。システムは大きくわけてふたつのソフトウェアからなる。ひとつは対話全体を制御するための “Dialog Manager” であり，もうひとつはロボットを制御するための “Sota Controller” である。 “Sota Controller” は Sota に搭載された計算機（Raspberry Pi）上で動作させるが，この PC は計算性能等が十分ではないため， “Dialog Manager” は対話制御用 PC 上で動作させ，両者は TCP/IP で通信を行う。

“Dialog Manager” は別途モジュールモードで起動された julius [5] と TCP/IP 通信し，音声認識結果を得る。julius はロボットに搭載されたマイクを利用するため，本来であれば Sota 上の計算機で動作させるべきであるが，計算性能の問題から，そこでは録音モジュール（ “adintool” ）のみ動作させ，音声データを TCP/IP で対話制御用 PC 上で動作している julius に通信している。

一方，音声合成器として利用している OpenJTalk[6] は，Sota 上の計算機で動作させ，直接 Sota のスピーカから合成音声を再生している。こちらも計算性能の問題から，音声を合成するまでに時間がかかり，そのままでは対話のテンポが悪くなってしまう。そのため，一度生成した合成音は wav ファイルでキャッシュに保存しておき，同じ発話内容の合成指令が来た時には wav ファイルの再生のみを行う。始めての発話内容であれば OpenJTalk を用いて合成音声を生成するが，その時にかかる時間を埋めるため，「え〜と」と発話させて間を持たせている。

“Dialog Manager” は与えられた対話シナリオに従って，対話全体を制御する。julius から送られてくる認識結果を分析し，キーワードが含まれていれば，シナリオに従って “Sota Controller” に発話や動作の要求を出す。“Sota Controller” は， “Dialog Manager” から送られてくる要求に従って動作や音声合成を行う。動作は，事前に Vstone 社から提供されているロボットの動作を設計するソフトウェアである “Vstone Magic” を利用して作成しておく。“Vstone Magic” は，ロボットの各モータを GUI で動かし，動作を定義することができるソフトウェアである。それぞれのモータを個別に設定する必要があるため，ひとつの動作を定義するのは簡単ではないが，実際にロボットを動かしながらモータの設定を決めていくことができるため，ロボットの制御プログラム等の知識がない人でも動作定義を行うことが可能である。

対話シナリオには，事前に作成された動作の名称（ファイル名）を記述しておき， “Sota Controller” は，送られてきた動作の名称に対応する動作ファイルを “Motion Controller” に送信し，そこで動作ファイルが再生されることで，ロボットを動かす。

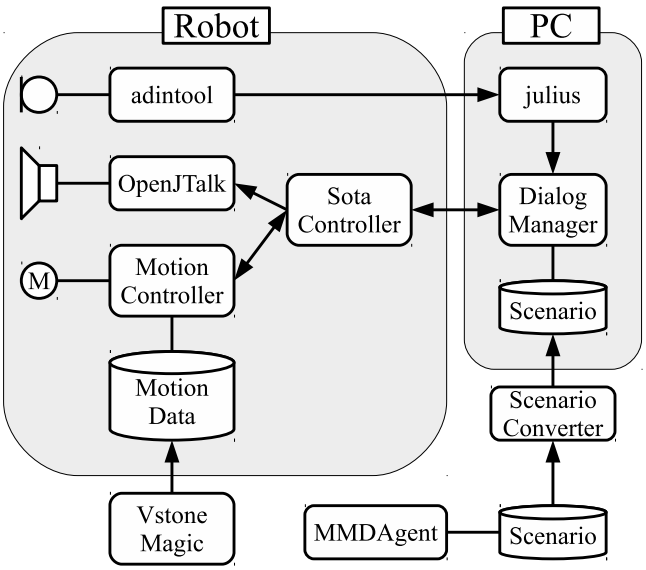

図 2 音声対話システム開発環境の全体構成図

2.4 MMDAgent との共通利用の仕組み

対話シナリオは，MMDAgent が利用しているフォーマットをそのまま利用する。そのため，対話シナリオの開発途中においては，MMDAgent を利用して実行することが可能であり，手軽にテストすることができる。この際，MMDAgent 内のソフトウェアエージェントのモーションデータと，Sota の動作ファイルに対応がとれていないと，シナリオデータをそのまま転用することができない。そのため，事前に MMDAgent 側に用意されているエージェントのモーションデータと同じ動作を行う動作ファイルを “Vstone Magic” を用いて定義しておく必要がある。

MMDAgent 用のシナリオデータには，「ソフトウェアエージェントを表示せよ」といった，実ロボットの対話システムには不要なコマンド（MMDAgent では，「メッセージ」と呼ばれる）等も記述されている。また，エージェント用のモーションデータと Sota 用の動作ファイルの名称が異なることも考えられるため，MMDAgent 用の対話シナリオファイルをそのまま転用することはできない。そこで， “Scenario Coverter” を用いて，MMDAgent 用のシナリオデータを “Dialog Manager” 用のシナリオデータへと変換する。基本的なファイル構造はどちらも同じであるが，Sota で実行する必要がない（もしくは実行できない）メッセージは無視するように書き換えたり，動作ファイルの名称を（与えられる対応表に従って）変換したり，といった事を行う。

「MMDAgent クリエイター向けリファレンス」[7] によると，MMDAgent で利用できるメッセージは（実行メッセージ，実行条件メッセージをあわせて）全部で 64 種類定義されている。これらのメッセージについて，「実装可能」や「実装不要」といったカテゴリに分類し，「実装可能」なものから順次実装を行った。これらのメッセージに

表 1　メッセージの対応状況

対応状況	メッセージ数	メッセージ例
実装済	12 (19%)	RECOG_EVENT_STOP, SYNTH_START, TIMER_START, VALUE_SET 等
実装不要	12 (19%)	MODEL_ADD, LIGHT_COLOR, CAMERA, KEY 等
実装不能	22 (34%)	MOTION_CHANGE, MOVE_START, LIPSYNC_START 等
未実装	18 (28%)	SOUND_START, PLUGIN_ENABLE, EXECUTE, KEY_POST 等

対する対応の内訳を表 1 に示す。

全メッセージのうち，12 種類は音声対話システムに必要であるため，そのまま利用可能として “Dialog Manager” に実装した。一方，「エージェントモデルをロードする」や「視点の変更」といったメッセージ（12 種類）は実ロボットの音声対話システムには不要なメッセージであるため，無視する仕様とした。

また，「エージェントを移動させる」や「唇を動かす」といったメッセージ（22 種類）も Sota では実行不可能であるため，同様に無視する仕様とした。しかしこのカテゴリのメッセージは，使用するロボットによっては実行可能なものもあるため，本来はシナリオデータには残しておき，“Sota Controller” 側で無視するのが正しい設計であると思われる。しかし今回は簡略化のため，シナリオコンバータで無視する仕様としている。

また，「音ファイルを再生する」「プラグインを有効にする」といったメッセージ（18 種類）は，現在では未実装となっているが，これらは順次実装していく予定である。

3. 開発環境の使用実験

本開発環境が正しく動作する事を確認し，またどの程度実用に耐え得るかを実証するため，いくつかシナリオの開発を行った。

3.1 MMDAgent “Sample Script” の実装

典型的なシナリオの例として，MMDAgent の公式 Web ページ [3] で公開されている “Sample Script” を利用した。このシナリオでは，挨拶等の簡単な対話に加え，名古屋工業大学の案内等を行う。そこで利用されているモーションデータは 15 個あり，それらのうち 5 個の動作に対応する Sota 用の動作ファイルを作成した。

残りの 10 個のうち，3 個は待機時の動作を定義したものであった。MMDAgent は待機時にも（待っている）動作を実行し続け，他の動作命令がきたら，その動作を実行した後で，待機動作に戻る，ということができる。一方 “Sota Controller” では，動作ファイルを実行すると，それが終わるまで他の動作をせずにブロックする，という仕様となっているため，待機動作のような事ができない。そこで，これら 3 個の動作は無視されることになった。

また，4 個のモーションは「頬に手をあてる」「胸のリボンに手をやる」といった動作であり，Sota ではモータの稼動域の関係から実行不可能な動作であった。更に 3 個のモーションは表情に関するものであり，これらも（顔の表情変化をすることができない）Sota では実行不可能な動作であった。

一方，メッセージについては，「外部プログラムの起動」等，いくつか未実装・実装不能なものがあったが，シナリオ全体としては，（「無条件に/何もしない」を表す “<eps>” もあわせて）全メッセージ数の 83%程度に対応しているという結果になった。

このシナリオを動作させてみたところ，モーションが減ってしまったためにロボットが動かずに対話が進む，といった場面も出てしまったが，それ以外は MMDAgent と同様に対話が行えることを確認した。

3.2 動作の自動推定機能を持つシナリオエディタへの対応

一般に実ロボットを用いた音声対話システムのシナリオを作成する際，ロボットに発話にあった動作をさせるのは難しく，試行錯誤しながら設定をする必要がある。そこで以前我々は，エージェントの発話テキストにあわせた動作を事前に準備したモーションデータの中から自動で選択するシステムを提案し，この機能を実装した，MMDAgent 用のシナリオエディタを開発した [8]。このシナリオエディタを利用して対話シナリオを作成することを想定し，どれだけ今回の環境でも実行可能となるか検証を行った。

3.2.1 動作ファイルの作成

このシナリオエディタには，31 種類のモーションデータが準備されている。それぞれのモーションデータについて，同じ動きをするような Sota の動作ファイルを人手で作成した。その結果，20 種類は作成することができたが，「両手をあわせる」「人差し指を出す」等，モータの稼動域やハードウェアの問題（Sota に指はない）で，11 種類は実装することができなかった。

また動作ファイルを作成する際，ソフトウェアエージェントと全く同じ動きを再現することは不可能であるため，1 種類のモーションデータにつき，同じ意味を表すように類似した 2 種類の動作（1 動作だけは 1 種類）を作成した。

3.2.2 動作の妥当性の検証

このシナリオエディタでは，ユーザが入力したエージェントの発話テキストを形態素解析し，動作の意図や意味に関連するキーワードが含まれていた場合に，そのキーワードに関連しているモーションデータを採用する，という方

表 2　動作の自然性に関する評価結果（抜粋）

動作	発話	評価結果				理由
		自然	やや自然	やや不自然	不自然	
額に手をやる	困りました	9	3	1	0	
	弱った	11	1	1	0	
額をたたく	失敗しました	7	5	1	0	
胸に手をあてる	安心しました	11	2	0	0	
泣く	つらい	8	1	3	1	首を振りすぎ
頭を抱え込む	少し考えたい	2	4	3	4	落ち込んだ感じがしない
	うそだ!	7	4	1	1	声が強くない
頭をかく	すみません	2	5	4	2	音声のトーンが高い

法でエージェントの動作を自動推定している。どのようなキーワードを採用すべきか，また否定文や疑問文，敬語等が用いられた場合はどうなるか，といった点については，実際に様々な発話文とエージェントの動作をあわせたビデオを作成し，被験者に自然性を評価してもらうことで決定していた。

しかし，ソフトウェアエージェントでは「自然である」と評価された動作と発話の組み合わせについても，実ロボットで行った場合は印象が異なる可能性がある。そこでSota でも動作可能と判断された 20 種類の動作（39 個の動作ファイル）について，それぞれの動作に対応する発話を 1 個の動作ファイルにつき 2 発話割り当て，全部で 78 個のビデオを作成した。これを 13 名の評価者にそれぞれ見てもらい，「自然である」から「不自然である」の 4 段階で評価してもらった。

実験結果（一部抜粋）を表 2 に示す。最終的に，否定的な評価（「不自然である」か「やや不自然である」）が 13 名中 1 名以内であるビデオを「自然である」と判定し，同じ動作データに付与した発話が 2 つとも「自然である」と判定された動作データを「採用」することとした。

その結果，10 個の動作だけが「採用」され，残りの 29 個は「不採用」となった。「不採用」となった動作について，その理由を評価者から聞きとり分類してみると，「モーションが早すぎる」「首をかしげすぎる」といったように，動作の微調整が足りなかったものが多かった。今回のコメントを元に動作ファイルを改良していくことで，より多様な動作に対応させることが可能であると思われる。

一方で，「音声がよく聞きとれなかった」「音声から（モーションが表しているような）感情が感じられない」といった，合成音声の品質が原因で「不採用」になったと思われる動作が 10 個あった。こちらについては，合成音声の品質向上を行う必要があるかと思われる。

最終的に，2 個の動作ファイルがいずれも「採用」となったのは「額に手をやる」（困惑を表す）と「首をふる」（否定を表す）の 2 種類だけであり，残りの 6 個の採用された動作ファイルは，同じ意味を表す類似した動作ファイルは「不採用」と判定されていた。

4. まとめ

本論文では，誰でも手軽に実ロボットを用いた音声対話システムを構築できるようにするため，実ロボット用音声対話システムの開発環境を提案した。本システムは大きくふたつのソフトウェアからなり，音声対話の制御とロボットの制御を分離することで，音声対話シナリオを様々なロボットに共通に利用可能としている。また，音声対話シナリオをソフトウェアエージェントによる音声対話システムである MMDAgent と共通フォーマットにすることで，開発途中は MMDAgent を用いて動作確認することができ，開発効率を向上させている。

Vstone 社の Sota を用いて実ロボットを用いた音声対話システムを構築し，本開発環境による音声対話システム開発の有効性を確認した。MMDAgent で定義されているメッセージのうち，72%に対応（実装済もしくは対応不要）しており，ほぼ同じ動きをすることが確認された。また，エージェントのセリフにあわせた動作を自動推定するシナリオエディタとの組み合わせでは，31 種類の動作のうち 8 種類が Sota でも「自然である」と判断され，それなりに有効であることがわかった。

今後，MMDAgent のメッセージのうち未実装となっているものの実装をすすめ，また動作ファイルの内容の微調整を行うことで，より「自然である」と感じられる動作を増やし，様々な対話内容に対応した開発環境を整備していく予定である。

参考文献

[1] Softbank, “pepper,” 2015. [Online]. Available: http://www.softbank.jp/robot/special/pepper/
[2] SHARP, “RoBoHoN,” 2016. [Online]. Available: https://robohon.com/
[3] Nagoya Institute of Technology, “MMDAgent,” 2009. [Online]. Available: http://www.mmdagent.jp/
[4] ヴィストン株式会社, “普及型社会的対話ロボット「sota」,” 2016. [Online]. Available: https://www.vstone.co.jp/products/sota/index.html
[5] A. Lee, T. Kawahara, and K. Shikano, “Julius — an open source real-time large vocabulary recognition engine,” in

Proc. EUROSPEECH, 2001, pp. 1691–1694.

[6] Nagoya Institute of Technology, "The Japanese TTS system OpenJTalk," 2009. [Online]. Available: http://open-jtalk.sourceforge.net/

[7] 名古屋工業大学, *MMDAgent クリエイター向けリファレンス v1.03*, 2016. [Online]. Available: http://www.udialogue.org/ja/encyclopedia-ja/creator-manual-jp.html

[8] M. Suzuki and K. Kawashima, "Automatic motion selection method for spoken dialog scenario editor," in *Proc. 20th Annual Conference on Knowledge-Based and Intelligent Information & Engineering Systems*, 2016, pp. 410–417.