タブレット端末のための 複合状態を用いた音声対話コンテンツ編集手法

若林 敬太郎[1]　山本 大介[1]　高橋 直久[1]

概要：近年，携帯端末向けの音声対話システムが普及してきている．そこで，我々は Android 版 MMDAgent を開発し，携帯向け音声対話システムに関する研究を行ってきた．MMDAgent は，FST スクリプトを編集することで，ユーザ自身が任意に対話シナリオをカスタマイズすることができる．FST スクリプトは，状態遷移を再現する構文で対話シナリオを表現する．しかし，自然な対話を表現するためには対話シナリオの状態数が多くなってしまい，手作業で FST スクリプトを編集することは初心者にとって大変である．本論文では，これらの問題を解決するため，状態遷移図に複合状態を導入し， MMDAgent 用の対話シナリオを作成する手法を提案し，その実現方法を述べる．視覚的に状態遷移図を表示することで，直感的に対話シナリオの遷移の様子を理解することができる．複合状態を導入することで状態遷移を意味のあるまとまりにまとめることができ，一度に表示する状態数を減らすことができる．さらに，実現法をもとに開発したプロトタイプシステムを用いて評価実験を行い，ユーザビリティと状態数の変化を調べた．

1. はじめに

近年，Apple の Siri[1] や，NTT ドコモ のしゃべってコンシェル [2] などの携帯端末向け音声対話システムが普及しつつある．そこで，我々の研究室では音声インタラクション構築ツールキットの MMDAgent[3], [4] を Android 向けに移植した Android 版 MMDAgent[5] を開発し，Android 版 MMDAgent を用いて携帯端末向け音声対話システムの研究を行っている．

MMDAgent は 3D キャラクタが画面に表示され，そのキャラクタと会話ができる音声対話システムである．MMDAgent を利用した音声対話システムとして，名古屋工業大学正門前に設置してある双方向音声案内デジタルサイネージ [6] などにも応用されている．

MMDAgent の特徴として，ユーザによって任意に対話シナリオの編集が行えるという点が挙げられる．対話シナリオとは「ユーザがこう言ったなら，キャラクタはこう言い返してこういう動きをする.」といった対話の流れやその台本のようなものを意味する．対話シナリオは FST スクリプトと呼ばれる外部ファイルに記述する．MMDAgent の起動時に FST スクリプトが読み込まれ，記述した対話を行うことができる．

FST スクリプトは指定された形式でテキストファイルとして編集する必要があるが，手作業で FST スクリプトを編集することは大変である．

また，自然な対話を記述しようとすると様々な状況に対応させなければならず，どうしても状態数が多くなってしまう．状態数が多くなると FST スクリプトの行数が増え，手作業での管理が難しくなる．

そこで，我々は状態遷移図を用いて音声対話コンテンツを作成するシステム [7] を提案した．しかし，状態遷移図を用いて音声対話コンテンツを作成する際，状態数が多くなると GUI で編集しづらくなってしまう．そこで，本論文では複合状態を用いた音声対話コンテンツ編集手法を提案する．

本研究の目的は，Android 版 MMDAgent の対話シナリオをタブレットを用いてタッチ操作で編集でき，手軽に高品質な対話シナリオが作成でき，状態数が多い対話シナリオでも作成しやすいシステムを提案することである．

本論文の構成は次の通りである．第 2 章では MMDAgent についてさらに詳しく説明し，その問題点を挙げる．第 3 章では提案システムについて述べる．第 4 章では開発したプロトタイプシステムについてインターフェースや使用の流れなどを説明する．第 5 章では開発したプロトタイプシステムで行った評価実験とその考察について述べる．第 6 章では本論文と関連性のある研究を挙げる．最後に第 7 章でまとめを述べる．

[1] 名古屋工業大学 大学院工学研究科
Graduate School of Engineering，Nagoya Institute of Technology

2. 音声インタラクションシステム構築ツールキット MMDAgent とその問題点

2.1 MMDAgent の概要

MMDAgent は音声認識，音声合成，対話管理，3D モデルの描画，物理演算などの機能を統合したシステムであり，図 1 のように画面上に 3D キャラクタが登場し，そのキャラクタと音声による会話をすることが可能である．音声認識エンジンには Julius[8] を，音声合成エンジンには，OpenJTalk[9]，3D モデル・モーションファイルの形式には MikuMikuDance[10] の形式が採用されている．

図 1 Android 版 MMDAgent

我々は，音声インタラクションシステム構築ツールキット MMDAgent を Android 端末向けに移植したものである Android 版 MMDAgent を用いて，携帯端末向け音声対話システムの研究を行っている．

MMDAgent の特徴としてユーザによって任意に対話シナリオのカスタマイズが可能であるという点が挙げられる．MMDAgent は対話シナリオの記述形式に，有限状態オートマトンの一種である有限状態トランスデューサ (FST) を採用している．対話シナリオを記述された FST 形式のファイルは，「FST スクリプト」もしくは略して「FST」と呼ばれる．この FST スクリプトを編集することによって対話シナリオのカスタマイズが可能となっている．

2.2 FST スクリプトについて

FST スクリプトの一例を図 2 に示す．

図 2 のように，遷移前状態番号，遷移先状態番号，遷移条件となるイベントメッセージ，出力するコマンドメッセージの 4 つ組を 1 行でこの順で空白もしくはタブ文字で区切って記述する．この 4 つ組が状態遷移図での 1 つの辺に該当し，この 4 つ組の集合で対話シナリオの状態遷移を表す．

図 2 の前半は，ユーザの発話に「こんにちは」または「こんにちわ」という単語が認識できた場合，3D キャラクタ mei が「こんにちは」と返事をしながらおじぎをするという対話シナリオである．

FST スクリプトは，複数のファイルを用意して，それぞれ独立した状態遷移を並列に実行することができる．MMDAgent の実行時に FST スクリプトが読み込まれ，0 番の状態がそれぞれの FST スクリプトの開始状態として扱われる．

2.3 FST スクリプトの問題点

FST スクリプトの編集は初心者にとって大変である．

自然な対話を作成しようとすると，さまざまな状況に対応する対話シナリオを作成しなければならず，どうしても状態数が多くなってしまう．

FST スクリプトは状態遷移図をテキストで表現したものであり直感的ではなく，FST スクリプトを読んでどのような状態遷移をするかを理解するのは大変である．また FST スクリプトでは，同じ状態番号の遷移でも行を大きくまたいで遷移を記述することが可能で，そのような記述を避けていても複雑な状態遷移になればそうせざるを得ない場合がでてくる．そのような場合，状態遷移を理解しづらくなる．

そして，FST スクリプト編集時によくあるバグとして状態番号の意図しない重複がある．FST スクリプト編集で状態を追加する場合，使われていない番号はどれか，使おうとしている番号は本当に使われていないかを確認しなければならない．FST スクリプトの状態数が多い場合，番号の重複を見逃しやすくなる．

図 2 のサンプルパターン 1 の対話を状態遷移図で表すと図 3 のようになる．このように状態遷移図は視覚的に表した方が直観的にわかりやすい．

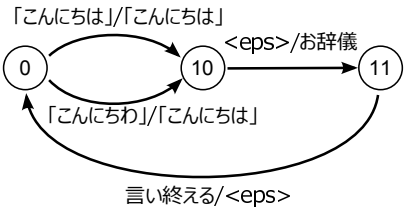

図 3 図 2 のサンプルパターン 1 を表した状態遷移図

ただ，状態遷移図を視覚的に表現しても状態数が多くなると状態遷移図中の要素が多くなり，GUI で画面が見づらかったり操作しづらくなったりという問題もある．状態数が多くても操作しやすくする方法を考えなければならない．

3. 提案手法

3.1 複合状態を用いた音声対話シナリオの作成手法

本論文では複合状態を導入し視覚的に表現した状態遷移図を用いて MMDAgent 用音声対話シナリオを作成する手法を提案する．特に Android 版 MMDAgent を使用する Android 端末上で動作するシステムを提案する．

```
# Sample Pattern 1 : Hello↓
↓
  0    10   RECOG_EVENT_STOP|こんにちは          SYNTH_START|mei|mei_voice_normal|こんにちは。↓
  0    10   RECOG_EVENT_STOP|こんにちわ          SYNTH_START|mei|mei_voice_normal|こんにちは。↓
 10    11   <eps>                               MOTION_ADD|mei|action|Motion¥mei_greeting¥mei_greeting.vmd|PART|ONCE↓
 11     0   SYNTH_EVENT_STOP|mei                 <eps>↓
↓
# Sample Pattern 2 遷移先状態番号 on↓                                    出力
↓
  0    20   RECOG_EVENT_STOP|自己紹介          SYNTH_START|mei|mei_voice_normal|メイと言います。↓
 20    21   SYNTH_EVENT_STOP|mei                SYNTH_START|mei|mei_voice_normal|よろしくお願いします。↓
 21         [illegible]                         MOTION_ADD|mei|action|Motion¥mei_self_introduction¥mei_self_introduction.vmd|PART|ONCE↓
 22    遷移前状態番号 STOP|mei    遷移条件      eps>↓
```

図 2　FST スクリプトの例

状態遷移図を視覚的に表現して対話シナリオを作成することによって，状態遷移を直感的に理解しやすくなるというメリットがある．また，複合状態を導入することによって状態遷移図を意味のある単位にまとめ，マクロ的な視点で状態遷移図が作成することができるようになる．そして，意味のある単位ごとにまとめてひとつの状態として扱うことができるため，一度に表示する状態数が減少し，GUI での操作性が向上する．

提案の特徴は以下の通りである．

特徴 1　複合状態を導入した状態遷移図を用いて対話シナリオを作成する．これにより，状態遷移図の見かけの状態数を減らすことができる．

特徴 2　複雑な対話であっても簡単に作成できるように，よく使う状態遷移パターンをテンプレート化する．

特徴 3　作成した状態遷移図からとテンプレートから FST スクリプトを生成する．

3.2　複合状態を導入した状態遷移モデル

FST スクリプトの状態遷移モデルはミーリ・モデルを採用している．ミーリ・モデルのまま状態遷移図にすると状態数が多くなり，状態遷移図が見づらくなってしまう．そこで，一度に表示すべき状態数を減らすため，提案手法ではミーリ・モデルに複合状態を導入した状態遷移モデルを採用する．

複合状態を導入することによって状態遷移を意味のあるまとまりごとにひとつの状態にまとめることができ，状態数が減少しマクロ的な視点で状態遷移図を作成することができる．

図 4 に提案する状態遷移モデルの例を示す．このように提案する状態遷移モデルの構成要素は，原子状態・複合状態・遷移の 3 つである．複合状態は内部遷移を含む状態を表し，原子状態は複合状態ではない状態を表す．また，遷移は状態同士を結ぶ有効辺である．

状態遷移モデルのデータ構造の例を図 5 に示す．図 5 の通り，状態遷移図は木構造をしている．四角が複合状態を示し，丸が原子状態を示す．矢印の終点の状態が矢印の始点の状態の内部状態であることを示す．

状態 root, S, M はシステムが管理していて，ユーザが自由に変更できるのは状態 M 以下である．

複合状態を導入することによって，状態遷移を意味のあるまとまりごとにまとめて管理することができる．

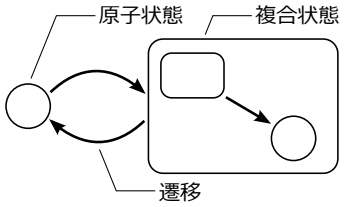

図 4　状態遷移モデルの例

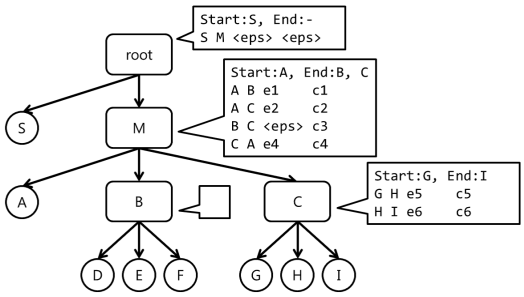

図 5　状態遷移図のデータ構造の例

3.3　FST テンプレート

提案システムでは，よく使う状態遷移パターンをテンプレート化し，より少ない操作回数で状態遷移図を作成できる機能を持つ．テンプレートを用いることで，複雑な状態遷移パターンやよく使う状態遷移パターンを簡単に作成することができる．

大まかな状態遷移をあらかじめ用意し，それを FST テンプレートと呼ぶこととする．外部ファイルで FST テンプレートと対応する入力フォームを定義しておき，FST テンプレートと入力フォームの内容を基に状態遷移の実態を生成する．生成された状態遷移をひとつの複合状態として状態遷移図に追加する．

FST テンプレートとそのテンプレートに入力するべき値を入力させるためのフォームは外部ファイルで定義する．入力フォームを構成する要素としてはテキストボックスとプルダウンリストを使うことができる．

FST テンプレートを用いた状態遷移図作成例を図 6 に示す．このようにフォームに入力された値と FST テンプレートを組み合わせることによって具体的な状態遷移の実態を生成する．生成された状態遷移はひとつの状態の内部遷移として書き出す．

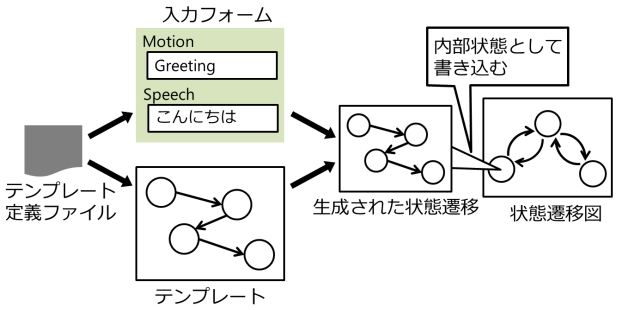

図 6　FST テンプレートの利用例

3.4　提案モデルから従来モデルへの変換

作成した対話シナリオを MMDAgent に読み込ませるためには FST スクリプトを生成しなければならない．しかし，現状の FST スクリプトの状態遷移モデルは複合状態を含まない．そこで，作成した対話シナリオを FST スクリプトに書き出す際に，状態遷移図を変換しなければならない．

変換の手順は次の通りである．

(1) 原子状態へ状態番号の割り当て
(2) 複合状態の展開
(3) FST スクリプトとして書き出す

以降では図 5 で表した状態遷移図データを例として変換について説明していく．

3.4.1　状態番号の割り当て

FST スクリプトでは状態を状態番号で識別している．そこで，FST スクリプトへ書き出す際に原子状態へ状態番号を重複しないように割り当てる．

状態番号の割り当て方法は次の通り．

(1) 状態 S に状態番号 0 を割り当てる
(2) 状態 M から再帰的にノードを探索していく
(3) ノードが原子状態ならまだ割り当てられてない最小の番号を割り当てる

状態 S にだけは確実に状態番号 0 を割り当てなければならない．なぜなら，FST スクリプトは 0 番の状態を対話シナリオの初期状態として扱うためである．提案システムで作成する対話シナリオは状態 S が対話シナリオの初期状態であるため，まず状態 S に 0 を割り当てる．

図 5 のデータに状態番号が割り当てられた後の様子は図 7 のとおりである．

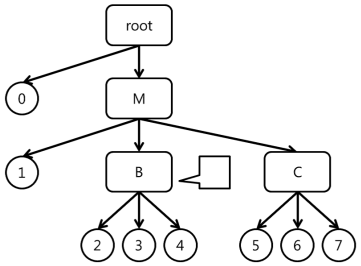

図 7　図 5 に状態番号を割り当てた後

3.4.2　複合状態の展開

現状の FST スクリプトの状態遷移モデルには複合状態がない．そのため，FST スクリプトへ書き出す際に複合状態を展開しなければならない．展開の方法は UML のステートマシン図と基本的には同様である．例えば図 8 のような状態遷移図を展開するとき，図 9 のように複合状態から出て行く遷移をすべての内部状態につなぐ．ただし，図 10 のように複合状態から出て行く遷移の遷移条件が<eps>であったときは，図 11 のように複合状態の終了状態にのみつなぐ．

FST スクリプトでは，遷移条件として <eps>が指定された場合，入力を待たずに直ぐに遷移する．そのため，展開時に全ての内部状態につないでしまうと開始状態からすぐに複合状態の外に出て行ってしまう．そのため，提案システムでは<eps>を「処理が終わると遷移する」という条件であるとみなし，複合状態の終了状態にのみ <eps>遷移をつなぐ

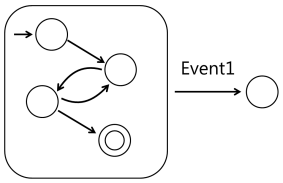

図 8　複合状態の展開前

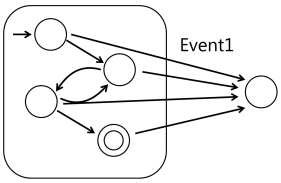

図 9　複合状態の展開後

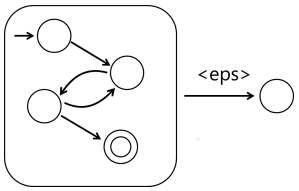

図 10　複合状態の展開前（<eps>の場合）

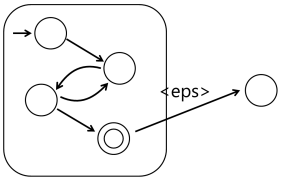

図 11　複合状態の展開後（<eps>の場合）

3.4.3　FST スクリプトとして書き出す

最後に，作成した対話シナリオを MMDAgent が読み込めるように FST スクリプトへ書き出す．

図 12 のように状態番号が割り当てられた状態遷移データは，図 13 のように書き出される．

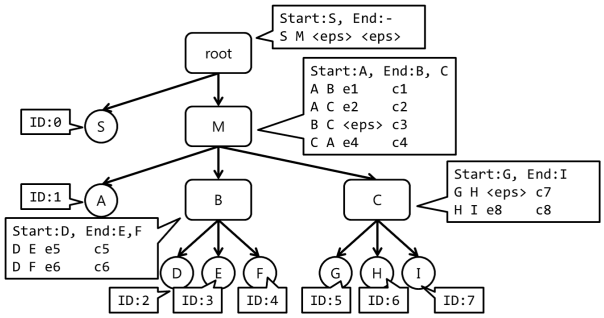

図 12　書き出し前の状態遷移データ

```
1   0  1  <eps> <eps>
2
3   1  2  e1    c1
4   1  5  e2    c2
5   3  5  <eps> c3
6   4  5  <eps> c3
7   5  1  e4    c4
8   6  1  e4    c4
9   7  1  e4    c4
10
11  2  3  e5    c5
12  2  4  e6    c6
13
14  5  6  <eps> c7
15  6  7  e8    c8
```

図 13　図 12 を FST スクリプトとして書き出したもの

4.　プロトタイプシステム

提案手法によって音声対話シナリオの編集を行うことができるプロトタイプシステムを実装した．プロトタイプシステムは Android SDK を用いて実装し，Android 端末で動作する．

4.1　使用時の流れ

このプロトタイプシステムを使用して FST の編集をする流れは以下のとおりである．

(1) プロトタイプシステムを起動する．
(2) 状態や遷移を状態遷移図に追加していく．
(3) 状態や遷移の詳細な内容を記述する．
(4) 2, 3 を繰り返す．
(5) 実行ボタンを押し FST を書き出す．

4.2　インターフェース

4.2.1　メイン画面

メイン画面を図 14 に示す．画面の左側に状態遷移図を描画するエリアがある．このエリアに状態や遷移を追加していって状態遷移図を描画する．画面の右側に MMDAgent を表示するエリアがある．作成した会話を試すことと操作のフィードバック，音声操作に用いる．

右上のメニューから “Execute” を選択すると，作成した状態遷移図を FST スクリプトに書き出し，MMDAgent に読み込まれる．

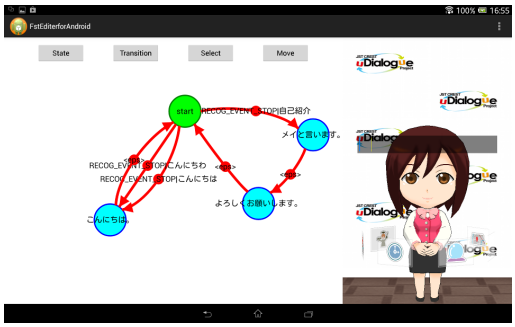

図 14　メイン画面

4.2.2　状態設定ダイアグラム

メイン画面で状態をロングタップすると図 15 の様に状態の設定ダイアグラムが出現する．ダイアグラム上部のボタンでテンプレートを選択することで，下部の入力フォームが変化する．

ダイアグラム下部の各ボタンの動作は以下の通りである．

- OK ボタンを押すとダイアグラムが消え，選択した状態の動作が入力フォーム通りに設定される．
- Try ボタン] と押すとダイアグラムは消えず，どのような動作が行われるのかを MMDAgent 画面で確認することができる．
- Cancel ボタンを押すと，設定を変更せずにダイアグラムが消える．
- Delete ボタンを押すと，選択した状態が状態遷移図から削除される．

図 15　状態設定ダイアグラム

4.2.3　遷移設定ダイアグラム

メイン画面で遷移をロングタップすると図 16 の様に遷移の設定ダイアグラムが出現する．ダイアグラム上部のボタンでどのイベントが発生したら遷移するかを選択する．選択したイベントによって入力フォームが変化する．

ダイアグラム下部の各ボタンの動作は以下の通りである．

- OK ボタンを押すとダイアグラムが消え，選択した遷移に設定が反映される．

- Cancel ボタンを押すと，設定を変更せずにダイアグラムが消える．
- Delete ボタンを押すと選択した遷移が状態遷移図から削除される．

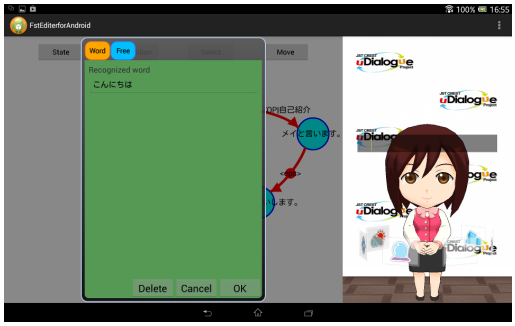

図 16　遷移設定ダイアグラム

4.3　プロトタイプシステムの機能

プロトタイプシステムは以下の機能を持つ．

- FST テンプレート
- キーワードの音声入力
- 矢印の位置調整

4.3.1　FST テンプレート

作成時によく使う一連の処理をテンプレートとして用意し，ユーザが作成したい処理に近いテンプレートを選択してデータを入力することでシステムが状態遷移を生成し，操作回数を減らし手軽に編集を行えるようにする．

提案システムは状態の設定をテンプレートを用いて行う機能を有する．例えば，「キャラクタにお辞儀をするなどの動作をさせながら何かセリフを言わせる」といったよく使うであろう一連の処理がある．その場合に，ユーザがテンプレートの中から動作とセリフを設定するテンプレートを選択し，入力フォームで動作を選択しセリフを入力することで処理の設定を行うといったよく使う一連の処理を簡潔に設定する機能である．

4.3.2　キーワードの音声入力

音声入力を行うことでキーボード操作を減らすことができる．また，遷移のトリガとなる言葉を設定する場合，音声認識モジュールが実際に認識する言葉を設定しなければならない．しかし，辞書登録されていない等の理由により音声の認識誤りが起こる可能性があり，意図した動作をしないということが起こり得る．

そこで，キーワードの設定の際も音声入力で行い，実際に使われている音声認識モジュールの認識結果をキーワードと設定することによって，認識誤りによって動作しないという問題を解決する．

4.3.3　矢印の位置調整

矢印を追加する際に，矢印を重ならないように配置することで見やすい状態遷移図を作成できる．

5.　実験と考察

5.1　ユーザビリティに関する実験

この実験の目的は，提案システムが FST スクリプトの問題点を解決し音声対話シナリオの編集が容易になったかを実証することである．

次に実験方法について説明する．指定した会話ができるように Android 版 MMDAgent の対話シナリオを 2 つの手法を用いて被験者に編集してもらい，それぞれの時間を測定する．その後アンケートをとって 5 段階評価してもらい 2 つの手法を比較する．またアンケートには自由コメント欄も用意した．被験者は大学生 8 人である．

編集手法は以下の 2 つである．被験者の半分は，先に手法 1 を後に手法 2 を行い，もう半分は逆の順番で行う．

手法 1　FstFileEditor[11] を用いて FST をテキストで編集する．

手法 2　提案システムを用いて FST を状態遷移図で編集する．

作成してもらう会話は，反応する言葉とその応答（台詞とジェスチャー）を細かく指定し作成してもらった．

アンケートは 5 段階評価と自由コメントを用意した．アンケートの項目は表 1 の通りである．手法 1，手法 2 の両方に関して，それぞれの項目の評価をしてもらった．評価は 5 段階でしてもらい，評価基準は「1：あてはまらない，2：ややあてはまらない，3：どちらともいえない，4：ややあてはまる，5：あてはまる」とした．

表 1　アンケート項目

番号	質問内容
1	使っていて楽しい
2	手軽に編集できる
3	一般ユーザ向けだ
4	また使いたい
5	直観的に編集できる
6	どの状態からどの状態へ遷移をつなげればよいかわかりやすい
7	状態遷移の様子が理解しやすい
8	編集方法を覚えやすい
9	思い通りの会話が作れた
10	品質の高い会話が作れる
11	複雑な会話が作れる
12	バグが発生しにくい
13	間違えやすい
14	操作がつかれる
15	作業が面倒だ

アンケートの各項目について評価値の平均をとったグラフを図 17 に示す．従来手法が FstFileEditor を用いた手法で，提案手法が提案システムを用いた手法である．項目 1 から項目 12 は数値が高いほど良い評価で，項目 13 から項目 15 は数値が高いほど悪い評価である．

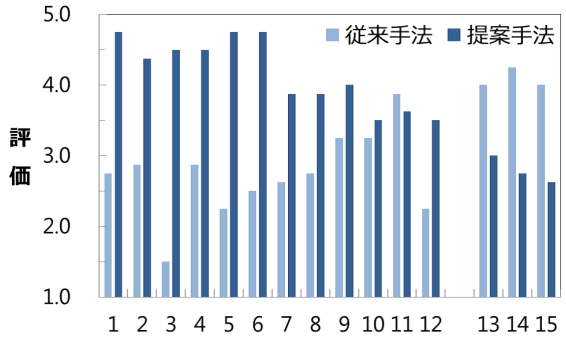

図 17 アンケート結果：1 から 12 までは値が大きいほど良い評価であり，13 から 15 までは値が大きいほど悪い評価である．

ほとんどの項目について，提案システムを用いた手法 2 の方が良い評価を得ていることがわかる．項目 1 から項目 6 については手法 2 の評価がどれもおよそ 4.5 と評価が特に高い．特に，項目 3 については手法 1 の評価が 1.5 となっており，手法 1 は一般ユーザ向けではないという結果が出た．この結果から，提案システムは一般ユーザにとって有効であると考えられる．

また，項目 12 については，手法 1 の評価が 2.3 に対して手法 2 の評価が 3.5 である．手法 1 のバグの発生しやすい問題を解決していると考えられる．

また，編集にかかった時間の平均を表 2 に示す．表 2 の通り被験者全員の平均で，FstFileEditor を用いた手法にかかった時間は 1288 秒，提案システムを用いた手法にかかった時間は 739 秒であり，449 秒早くなった．この結果より，従来手法で編集するより提案手法で編集する方がより速く編集することができることが分かった．

表 2 編集にかかった時間の平均 (秒)

	従来手法	提案手法
グループ A	1313	707
グループ B	1263	770
両グループ	1288	739

また，状態数の比較は表 3 にまとめた．提案手法の項目で変換前の状態数は，ユーザが作成した状態遷移図上での状態数であり，変換後の状態数は，作成した状態遷移図を FST スクリプトに変換した後の状態数である．

従来手法では状態数が平均 19.50 であったのに対し，提案手法の変換前の状態数が 15.13 と 4 分の 3 に減少している．ただ，提案手法の変換後の状態数は 47.90 と増加している．状態数の増加の原因は FST テンプレートの利用にある．

ユーザから「こんにちは」と話しかけられたときに，エージェントが「こんにちは」と返してお辞儀をするという対話シナリオを従来手法と FST テンプレートを用いた場合を比較する．従来手法で作成した場合の状態遷移図を図 18 に，FST テンプレートを用いて作成した状態遷移図を図 19 に示す．従来手法で作成した場合は図 18 のように，状態数 4 個で再現できる．台詞の発話と動作を行う FST テンプレートを用いて作成した場合は図 19 のように状態数が増える．テンプレートはひとつの複合状態にまとめられるためユーザから見ると状態数が 4 個から 3 個に減るが，全体では 4 個から 6 個に増える．テンプレートを適応するごとにこのような状態数の増加があるため，全体で状態数が大きく増加した．

表 3 状態数の比較

	従来手法	提案手法	
		変換前（見かけ）	変換後
状態数（平均）	19.50	15.13	47.90

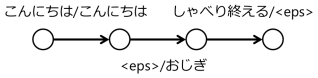

図 18 従来手法の状態遷移

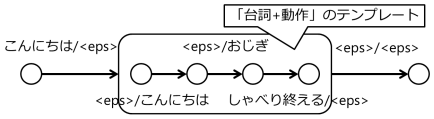

図 19 FST テンプレートを用いた状態遷移

5.2 初期対話シナリオの状態数比較

Android 版 MMDAgent の初期 FST スクリプトの状態数と，その対話シナリオを提案手法で再現した場合の状態数の試算の比較を表 4 にまとめた．展開後の状態数は，テンプレートを利用した状態数 1 つにつき，腹腔状態内の開始状態と終了状態が増えると考え，従来手法の状態数にテンプレートを利用した状態数の 2 倍の数を足し試算をした．テンプレートを利用した状態数は約 200 であった．

表 4 のように，従来手法に比べて提案手法の複合状態展開前の見かけの状態数が 10 分の 1 となった．ただし，提案手法の状態数は用意されたテンプレートによって変動する．

表 3 の比較より削減効果が高い理由は，初期対話シナリオの状態遷移はパターン化されている部分が多く，状態数の大きなテンプレートを作成することができ，多くの状態をひとつの複合状態にまとめることができたためである．

表 4 初期 FST での状態数の比較

	従来手法	提案手法	
		変換前（見かけ）	変換後
状態数	2729	257	3200

6. 関連研究

本論文と関連研究として Islay, MMDAE が挙げられる．

Islay[12] とは状態遷移図に基づく対話型アニメーション作成ツールである．ムーア型の状態遷移図を作成し，イベントドリブンのアニメーションが作成できる．小学生の利用も視野に入れて開発が行われた．Web 上で動作する．アニメーションだけでなく，ゲームの制作，ロボット制御，Web ブラウザのカスタマイズ用にも応用されている．

状態遷移図を用いたヴィジュアルプログラミングという点で本論文と関連がある．しかし，Islay は対話型アニメーションの作成を対象としているが，本論文では音声対話コンテンツを作成を対象としているという点で違いがある．

MMDAE[13] とは，web ブラウザ上で動作する FST スクリプトエディタである．FST スクリプトの入力補助機能がある．遷移条件や出力コマンドを入力途中で候補を補完することや，コマンドに対応する引数を補完するという特徴がある．また，web ブラウザで動作するため，さまざまなプラットフォームで動作が可能であるという特徴をもっている．そして，利用者の習熟度のレベルに合わせたインタフェースの変更が行えるという特徴をもつ．

MMDAgent の対話シナリオを作成するためのシステムであるという点で本論文と関連がある．しかし MMDAE は編集をテキストベースで行うが，本論文では GUI ベースで対話シナリオを編集するという点で違いがある．

7. おわりに

本論文では，携帯端末向け音声対話システム Android 版 MMDAgent の対話シナリオを編集するための複合状態を用いた編集手法を提案し，その実現方法について述べた．

提案手法は以下の特徴がある．

(1) 状態遷移図を用いて対話シナリオを編集でき，状態遷移図中で一度に表示すべき状態数を減らすために，階層的な状態遷移をもつ複合状態を採用
(2) よく使う状態遷移パターンのテンプレート化
(3) 作成した状態遷移図から FST スクリプトを生成する

評価実験により，使っていて楽しいというアンケート項目において，従来手法の評価平均 2.8 から提案手法の評価平均 4.8 となり提案手法の方が使っていて楽しいということが分かった．また，手軽に編集できるというアンケート項目において，従来手法の評価平均 2.9 から提案手法の評価平均 4.4 となり提案手法の方が手軽に編集できるということが分かった．そして，直観的に編集できるというアンケート項目において，従来手法の評価平均 2.3 から提案手法の評価平均 4.8 となり提案手法の方が直観的に編集できるということが分かった．特に，一般ユーザにとって有効であることが分かった．

さらに，複合状態を用いることで状態遷移図に表示する見かけの状態数が 10%に減少し， FST テンプレートが大きいほどその削減効果が高いことが分かった．

今後の課題として，提案システムで作った対話シナリオを配信・共有できるようにすることが挙げられる．また，FST テンプレートの，状態遷移のひな型とデータを組み合わせて具体的な状態遷移を生成する機能は，今回の提案システムだけではなく FST の生成を行うシステムで有効であると考えられる．FST テンプレートの機能をより汎用的にし，ライブラリ化することも今後の課題である．

謝辞

本研究は，JSPS 科研費 25700009，及び，科学技術振興機構 CREST の助成を受けたものです．ここの場を借りて感謝の意を表します．

参考文献

[1] Siri，入手先 ⟨http://www.apple.com/ios/siri/⟩ (2014.1.28 参照)．
[2] しゃべってコンシェル，入手先 ⟨https://www.nttdocomo.co.jp/service/information/shabette_concier/index.html⟩ (2014.1.28 参照)．
[3] Akinobu Lee, Keiichiro Oura, Keiichi Tokuda, MMDAgent - A fully open-source toolkit for voice interaction systems, Proceedings of the ICASSP 2013, pp. 8382-8385, 2013.5.
[4] MMDAgent，入手先 ⟨http://www.mmdagent.jp/⟩ (2014.1.28 参照)．
[5] 山本 大介，大浦 圭一郎，西村 良太，打矢 隆弘，内匠 逸，李 晃伸，徳田 恵一，スマートフォン単体で動作する音声対話 3D エージェント「スマートメイちゃん」の開発，情報処理学会シンポジウムシリーズ IPSJ Symposium Series，Vol.2013，No.1，pp.675-680，2013.
[6] 大浦 圭一郎，山本 大介，内匠 逸，李 晃伸，徳田 恵一，キャンパスの公共空間におけるユーザ参加型双方向音声案内デジタルサイネージシステム，人工知能学会誌，Vol.28，No.1，pp.60-67，2013.
[7] 若林 敬太郎，タブレット端末のための状態遷移図を用いた音声対話コンテンツ編集システムに関する研究，名工大卒業論文，2014.
[8] Lee Akinobu, and Tatsuya Kawahara, Recent Development of Open-Source Speech Recognition Engine Julius, Proceedings of the APSIPA ASC, pp.131-137, 2009.
[9] 大浦 圭一郎，酒向 慎司，徳田 恵一：日本語テキスト音声合成システム Open JTalk，日本音響学会春季講論集，Vol.1， No.2-7-6，pp.343-344，2010.
[10] MikuMikuDance， 入手先 ⟨http://www.geocities.jp/higuchuu4/⟩ (2014.1.28 参照)．
[11] FstFileEditorVer2.01 うｐしました。，入手先 ⟨http://d.hatena.ne.jp/CST_negi/20121215⟩ (2014.2.3 参照)．
[12] 岡本 秀輔，鎌田 賢，中尾 隆司，状態遷移図に基づく対話型アニメーション作成ツールの提案，情報処理学会論文誌．プログラミング，Vol.46，No.SIG 1(PRO 24)，pp.19-27，2005.
[13] 西村 良太，山本 大介，打矢 隆弘，内匠 逸，音声対話エージェントのための Web ブラウザを用いたシナリオエディタの開発，DICOMO2013 シンポジウム論文集，pp.1976-1799，2013.