Vol.2012-CDS-5 No.4
Vol.2012-DCC-2 No.4
2012/10/17

Animated Chatting System with Emotional Expression Using TVML

IN JOON CHO[†1] MAMORU DOKE[†2]
HIROYUKI KANEKO[†2] SEIKI INOUE[†2]

A proposal for animated chatting system for facilitating the communication of emotion is presented. The system exploits event-driven text-to-video generation technology called TVML(TV program Making Language) and enables its chat-clients to exchange various emotions using their CG characters' facial expressions and gestures. The core mechanism and examples of possible applications of the system will be explained.

1. Introduction

For over several decades, the amount of communication between people has increased like no other things in our history. Moreover, the rapid growth of telecommunication industry has gone further and brought us to the smart-phone age letting us be online anytime and anywhere. During the same period, many communication methods have underwent various vicissitudes where online chat made its way and successfully penetrated deep into our daily lives. Now people can chat online while they are at business, in motion, at leisure using computers and smart-phones.

Unlike spoken language, written language falls terribly short when it comes to articulating the rich contents of human emotions. When we speak something face to face, there are lots of additional information revealed by speaker's tone of voice, facial expressions and gestures. But written language is devoid of the above additional information and often fails to articulate delicate mood of human mind. This view about the deficiency of written language is so intuitively appealing and deeply entrenched in our everyday experience that one seldom raises questions about it.

Necessarily, the natural reasoning is that if we use graphical vocabulary like clip-arts, emoticons, avatar's gestures and facial expressions for expressing subtle and profound human emotions, we can express our inner thoughts or feelings in a more direct and effective way, sometimes interestingly, too. This helps communication much more than our imagination.

With rapidly evolving computer technology and network connections, the shortcoming of written language mentioned above was addressed and online chat stared to take the advantages of graphical representation like clip-arts, emoticons, and avatar to help more effective communication by letting participants in a chat intuitively grasp the speaker's mood. People find these visual components appealing and agree that they hugely contribute to richer user experience by enabling the participants to express their emotions with extreme effectiveness.

The primal limitation of written language and attempts shown by some chat services to overcome it hint that adding visual components to chat service could improve the way people communicate. In this sense, it is also true that delivering emotional quality residing in different universe where mere words couldn't reach will be a big challenge to chat services.

The idea of enhancing ever-popular online chat by enabling it to convey emotions into 3D animation made us to propose a avatar chat model using TVML(TV Program Making Language)[1].

TVML is a text-based language for automatic generation of TV programs in the form of 3D animation and its basic purpose is to make TV program production easy for ordinary people without significant knowledge in 3D graphics. The most appreciable merit of TVML when applied to 3D avatar chat is that it has all the necessary elements for TV program production like characters, set, lighting, camera, props, graphic, sound effect, etc. if these elements meet right stagecraft, speaking presentation of the mood and emotions residing in participants' conversation could come to life, and this is major motivation of our attempt to extend TVML to the realm of avatar chat.

For the flexible construction of a chat session, TVML avatar chat provides choices for CG characters and studio sets written in TVML script. Users can pick arbitrary CG characters for their avatars, choose an arbitrary studio set for the background of their chat session, and put their avatars in the positions pre-designated in the studio set. For the positioning of a CG character, a TVML script for the studio set must know the names of CG characters which appear in the studio set because a TVML script uses CG characters' name to position them. But there could be no way for the TVML script of a studio set to know which CG characters' name to use due to the fact that no definition about characters is included in the TVML script of a studio set. For this reason, We use so-called dummy name when we position characters which don't exist in the studio set yet. These dummy names should be substituted by real names of CG characters after users pick their CG characters, and this raises an issue about devising a method for matching real names of CG characters to the dummy names used in the TVML script for a studio set.

A second problem is that unlike TVML's application to TV program production where a TVML script is written by one person who has every knowledge and information about his CG characters, TVML avatar chat must be run by multiple persons who are ignorant about their CG characters. They don't know what gestures and what facial expressions their avatars can make, and this hampers visual expression of emotions.

The last problem is omnipresent in our everyday experience. We can quite often see many people who are suffocated by thick

†1 KBS TRI (Korean Broadcasting System Technical Research Institute)
†2 NHK STRL (NHK Science and Technology Research Laboratories)

instruction manual when they bought cutting-edge electronic devices. It is a course of nature that you have more switches and buttons when you have high-tech devices with fantastic functions. Nevertheless, it is seldom easy to accept the reality that you should learn something new and practice harder to be able to enjoy the quintessence of the up-to-date technology. The same logic can be extended to our chat model. Users would feel pretty burdensome if they have to learn something new to enjoy avatar chat or make inputs every time they want to change their avatar's facial expressions and gestures to make their mood or emotion revealed in more vivid ways, not to speak of doing something with cameras or lightings. Therefore, not only direct control of avatar's facial expressions and gestures but also automated scene creation consisting of complex combination of avatar's action, camerawork, sound effect triggered by content of dialogue are really necessary, and this helps the participants concentrate only on their chat itself and enjoy dramatic scenes provided automatically.

These problems have been addressed in the course of our experimental chat system development, and we managed to lay the proper foundation of TVML avatar chat. Following sections will give details of the methodology that we used in our application of TVML, and techniques we invented for impressive audiovisual rendering of emotions.

2. Experiment Chat System Overview

Figure 1 depicts the overall composition of experiment system for verifying the feasibleness of enriching emotional expression with audiovisual sensation. The system consists of chat server, chat client, and rendering/streaming system. In this experimental implementation, each instance of chat server corresponds to each chat session.

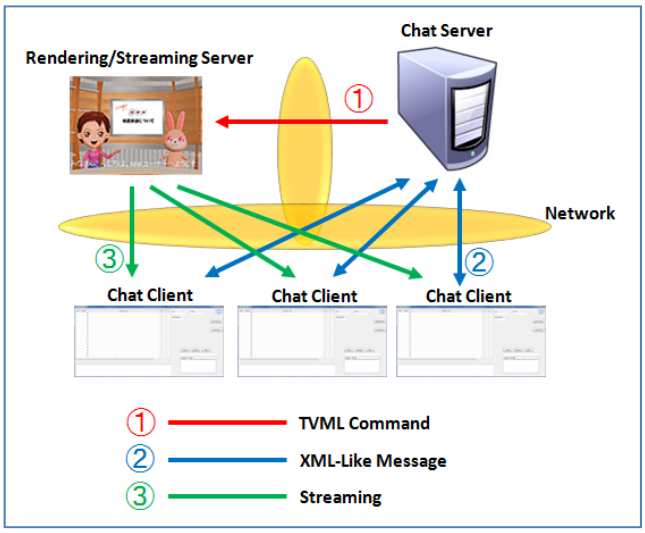

Figure 1. Experiment Chat System

2.1 Chat Server

Chat server's role is chat session managing, interpretation between rendering/streaming sever and chat clients, and making of automatic TVML commands based on participants' dialogues to facilitate rich audiovisual sensation.

2.2 Chat Client

Figure 2 is chat client's UI. The rectangle (1) is the grid showing chat history to users, the rectangle (2) is the edit box for conversation, the rectangle (3) is combo boxes for specifying avatar's poses, key-frame animations, facial expressions.

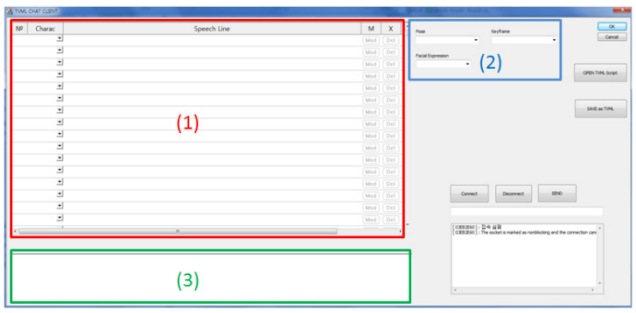

Figure 2. Chat client UI

2.3 Rendering/Streaming Server

The essential of Rendering/Streaming server in Figure 3 is providing text-to-video web service, i.e. users send a certain form of text script to the server, and the server automatically generates video content from the script, and streaming it to the user in real time. This is an entirely new way of video content creation and sharing.

The system is able to perform as many rendering and encoding processes as it can, whereas the number of simultaneous rendering and encoding processes of common S/W encoders using real audio and video signal is limited by the number of external audio and video inputs of the hardware on which it is run.

It is owed from the fact that the system is built on a new mechanism in which the video and audio outputs of the software called TVML Player are virtualized[2]. This virtualization makes encoding software recognize them as external inputs. This approach makes it possible to increase the number of rendering and encoding processes regardless of the number of external audio and video inputs of the hardware.

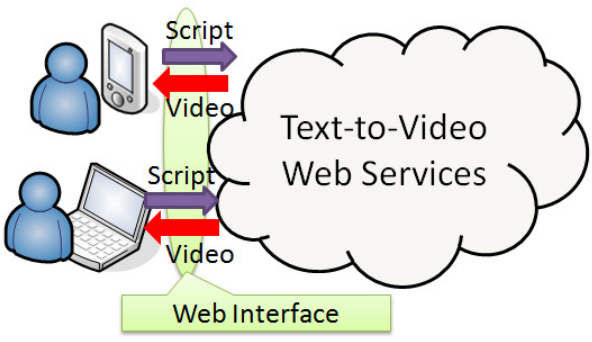

Figure 3. Conceptual diagram of Text-to-Video web service

2.4 Communication Between The Chat Server and The Chat Client

The communication between a chat server and chat clients is exchanged in XML-like text message. The use of this XML-like message instead of direct TVML commands is to leave room for future sophisticated intervention of server, i.e. adding background music or lighting effect automatically according to the semantics of conversation which helps participants be in empathy with one another while they are still left undisturbed by complicated stage effects and deal with only the nub of their dialogue. Some simple examples of XML-like text messages are as follows.

- An example of the message from the chat server

Figure 4 is a message from the chat server to notify a chat participant of the possible key-frame animations of his avatar. In the example, the participant's name is 'Nakaya' and his avatar has three key-frame animations, 'WavingHands', 'Shudder', 'Hooray'.

```
<TVML_Message>
  <KeyframeList>
    <ClientName>Nakaya</ClientName>
    <KeyframeName_0>WavingHands</KeyframeName_0>
    <KeyframeName_1>Shudder</KeyframeName_1>
    <KeyframeName_2>Hooray</KeyframeName_2>
  </KeyframeList>
</TVML_Message>
```

Figure 4. An example of server message

- An example of the message from the chat client

Figure 5 is a message from a chat participant asking the chat server to make his avatar speak. In the example, the participant's name is 'Nakaya' and his request is to make his avatar say "Good morning, everybody!"

```
<TVML_Message>
  <NewSpeechLine>
    <ClientName>Nakaya</ClientName>
    <Speech>Good morning, everybody! </Speech>
  </ NewSpeechLine >
</TVML_Message>
```

Figure 5. An example of chat client message

2.5 Communication Between The Chat Server and The Rendering/Streaming Server

The connection between the chat server and the rendering/streaming server is done by the TVML Player connection API[3], and TVML commands are fed to rendering/streaming server by the chat server. The TVML commands sent are the processed results of communication between chat server and chat client, i.e. when a participant named 'Nakaya' type 'Good morning, everybody!', his avatar's new speech line, in the edit box of the chat client, the message in Figure 5 is sent to the chat server, and it processes the client message requesting for a new speech and sends TVML command for the input to the rendering/streaming server. Supposing that the participant's avatar uses his name as its name, the TVML command would be like this:

```
character: talk(name=Nakaya, text = "Good moring, everybody !")
```

When the Rendering/Streaming server receives the TVML command above, It searches the avatar name 'Nakaya' in the scene and make it say "Good morning, everybody!"

3. TVML and Its Application to Avatar Chat

3.1 TVML

The video content production platform of this avatar chat system uses TVML technology, which NHK STRL has developed in various other R&D initiatives, as the video generation engine. TVML is a computer language for describing television program (video content) scenarios. As shown in Figure 6, a description of a video content scenario in TVML (a TVML script) is input to an application called the TVML Player. The TVML Player uses real-time 3D CG and voice synthesis to generate video contents immediately, according to the description in the TVML script. TVML language includes commands for controlling, for example, CG character, studio set, camera, lighting, prop, sound effect, etc. Therefore, it is excellent for making video content more portable, reproducible, editable, and distributable. To add one more word, TVML Player is its event-driven characteristic. The rendering/streaming server of our experiment system has built-in TVML Player.

Wide control over animation, especially over CG characters, and real-time/event-driven characteristics mentioned above gave us glimpse of applying TVML to avatar chat to make dialogue between chat participants emotionally rich through enhanced audiovisual sensation.

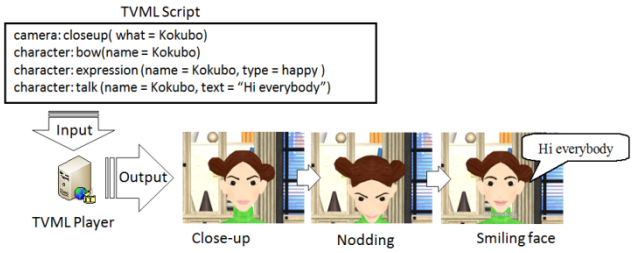

Figure 6. TVML script and video output example

3.2 Data Structure for Easy Handling of TVML Script

To set up a new chat session and fall into conversation, the chat server is needed to give each participant the list of possible options of facial expressions and gestures defined in the avatar's TVML character template (TVML analysis). Only after receiving these lists can participants proceed to share conversation, actively turning their emotions into avatar's actions. During the conversation, chat server processes each participant's inputs to make suitable TVML commands to drive TVML Player to render the chat session into 3D animation (TVML manipulation). For smooth running of a chat session,

easy and flexible TVML analysis and manipulation method is needed. Therefore, we designed an exclusive data structure for TVML analysis and manipulation.

TVML script can be regarded as a list of directions for each component of TV program production like characters, cameras, lightings, props, sets, sound, subtitle, etc. To best describe the characteristic of TVML, we choose to represent the data structure for TVML as a group of lists of C++ structure as shown in Figure 7.

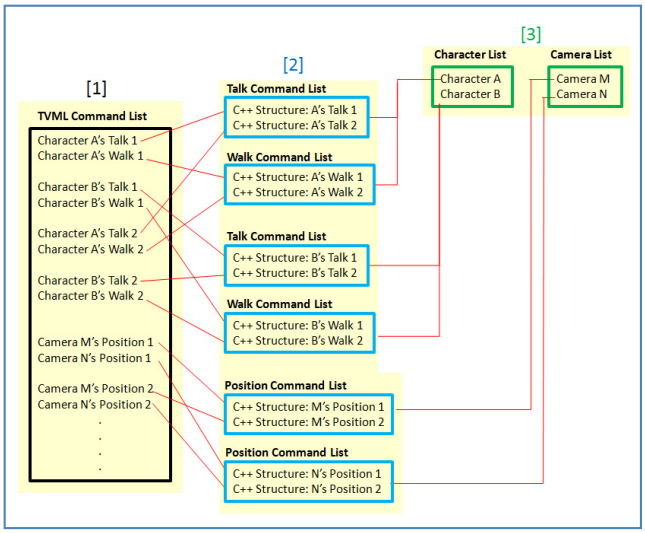

Figure 7. Data structure for TVML analysis and manipulation

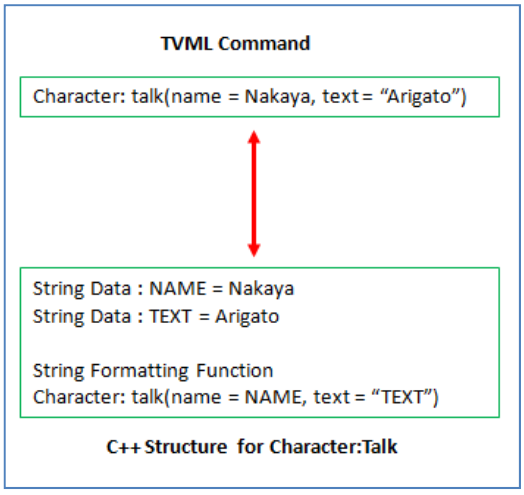

Figure 8. An example of TVML command and its C++ structure

As soon as a TVML script is fed to our data structure one by one, each line of the script is attached to TVML command list(region[1] in Figure 7), and next the TVML command is parsed to create its C++ structure image(elements in region[2] in Figure 7) like the one in Figure 8, and then they are mutually connected via pointers. In doing so, these C++ structures for each TVML command are also sorted and listed according to the criteria of their action targets(elements in region[3] in Figure 7) first and command types(something like walk command or talk command in Figure 7) next, and finally form a group of interconnected lists like the one in Figure 7. Using this data structure, we can easily see, for instance, what character is doing what while a specific camera is changing position from here to there, and much more.

- Analysis

Once a TVML script is fed to our data structure, looking into the lists marked as region[2] in Figure 7 is enough for analysis. We can identify every detail of TVML commands without effort.

- Manipulation

Figure 8 is the typical example of a TVML command and its C++ structure. Every TVML command's arguments have one-to-one correspondence with the elements of its C++ structures, and the C++ structure has a function to generate the TVML command from its elements.

To create a new TVML command or edit existing one, we first create or access the relevant C++ structure for TVML command to be created or edited, and make changes to the parameters in the C++ structure, and call its function for generating corresponding TVML command, and finally substitute the old TVML command with the newly generated one. Deletion process is so trivial that it is exempted from further explanation.

3.3 Construction of a Chat Session

Before starting a chat session, participants can pick a studio set and avatars(CG characters for participants) arbitrarily and select for each avatar a pre-designated position in the studio set. We leave automatic positioning of avatars to future study.

We will name a TVML script containing studio set and positions for avatars 'stage template', and a TVML script for CG character 'character template'.

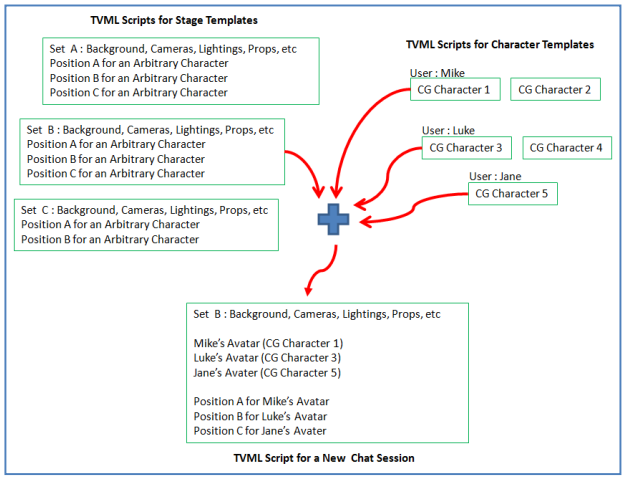

Figure 9. The procedure for the construction of a chat session

The construction of a chat session by an arbitrary combination of stage template and character template was made possible by welding the names of users with dummy names of CG characters in character templates and dummy names used in specifying the positions in stage template when combining them together as shown in Figure 9.

The procedure for the construction of a chat session is as follows.

1. The host of a chat session picks a stage template from the list of possible choices.
2. Every participant selects CG character for their avatars from the list of TVML character template.
3. Positions of avatars should not be overlapped. Therefore, the selection process of the positions is done in FIFO manner, i.e. the participant connected first chooses his avatar's position in the list and it is eliminated in the possible choices, and the participant connected next chooses his avatar's position among the remaining positions, and so on.

3.4 Character Control

In TVML, we can define different poses, key-frame animations, and facial expressions for different CG characters, and this can be a great advantage in individuating each CG character and as a result, endowing personality to participants' avatars.

Poses and key-frame animations for each CG character are defined explicitly in TVML command, and this leaves it self-contained to deliver information about them. But unlike poses and key-frame animations, a list of possible facial expressions for each character can't be retrieved directly from TVML script because of their implicitness in the CG character's model data. Therefore, we inevitably added additional information about possible facial expressions of each CG character on the top of its character template in the form of comments to avoid any mishap of causing a malfunction of TVML Player when they are fed.

Figure 10 is an example of a character template. Information of the possible facial expressions for the CG character, the TVML definition of key-frame animations, and poses are marked by region [1], [2], [3] in the Figure 10, respectively. 'FACIAL EXPRESSION LIST START' and 'FACIAL EXPRESSION LIST END' on the top of Figure 10 are to indicate the start and the end of the added information.

```
// FACIAL EXPRESSION LIST START
// neutral
// happy
// komaru
// surprise
// nigawarai
// angry
// think
// wink
// FACIAL EXPRESSION LIST END                                        [1]

character: casting( name=A )
character: openmodel( modelname=R, filename="R-summer¥R-summer.bm" )
character: bindmodel( name=A, modelname=R )
character: setvoice( name=A, voicetype="Mike" )
character: visible( name=A, switch=on )
character: position( name=A, x=2.8, y=0.0, z=2.5, d=-30.0, posture=standing )
character: expression( name=A, type=neutral )

character: openkeyframe(name=A, keyframename=yubisasiL, filename="chat-yubi-L.bvh")
character: openkeyframe(name=A, keyframename=yubisasi, filename="chat-yubi-R.bvh")
character: openkeyframe(name=A, keyframename=moto, filename="R-moto.bvh")          [2]
character: openkeyframe(name=A, keyframename=sasu-L, filename="chat-sasu-L.bvh")
character: openkeyframe(name=A, keyframename=sasu-R, filename="chat-sasu-R.bvh")
character: openkeyframe(name=A, keyframename=R-udekumi, filename="chat-udekumi-R.bvh")

character: definepose( name=A, pose=Getwhiskey_A, joint=LeftUpperArm, rotx=-105.00,roty=25.00 )
character: definepose( name=A, pose=Getwhiskey_B, joint=Chest, rotx=5.00, roty=-5.00, rotz=0.00 )  [3]
```

Figure 10. An example of a character template

After reading character templates for each avatar, chat server sends lists of possible facial expressions, poses, key-frame animations to participants to let their emotions guide their avatar's actions.

Assume that a participant named 'Nakaya' chooses the character template of Figure 10 for his avatar, then he will receive the message shown in Figure 11 from the chat server. The regions having the same number in Figure 10 and Figure 11 are corresponding regions.

```
<TVML_Message>                                                    [1]
  <FacialExpressionList>
    <ClientName>Nakaya</ClientName>
    <FacialExpressionName_0>neutral</FacialExpressionName_0>
    <FacialExpressionName_1>happy</FacialExpressionName_1>
    <FacialExpressionName_2>komaru</FacialExpressionName_2>
    <FacialExpressionName_3>surprise</FacialExpressionName_3>
    <FacialExpressionName_4>nigawarai</FacialExpressionName_4>
    <FacialExpressionName_5>angry</FacialExpressionName_5>
    <FacialExpressionName_6>think</FacialExpressionName_6>
    <FacialExpressionName_7>wink</FacialExpressionName_7>
  </ FacialExpressionList >
</TVML_Message>

<TVML_Message>                                                    [2]
  <KeyframeList>
    <ClientName>Nakaya</ClientName>
    <KeyframeName_0>yubisasiL</KeyframeName_0>
    <KeyframeName_1>yubisasi</KeyframeName_1>
    <KeyframeName_2>moto</KeyframeName_2>
    <KeyframeName_3>sasu-L</KeyframeName_3>
    <KeyframeName_4>sasu-R</KeyframeName_4>
    <KeyframeName_5>udekumi</KeyframeName_5>
  </KeyframeList>
</TVML_Message>

<TVML_Message>                                                    [3]
  <PoseList>
    <ClientName>Nakaya</ClientName>
    <PoseName_0>Getwhiskey_A</PoseName_0>
    <PoseName_1>Getwhiskey_B</PoseName_1>
  </ PoseList >
</TVML_Message>
```

Figure 11. An example of chat server messages for possible facial expressions, key-frame animations, and poses

3.5 Automatic Scene Creation

As stated earlier, TVML was originally designed for automatic TV program production. Due to this nature of initial design, commands to control CG characters, studio sets, cameras, lightings, sound effects, etc. exist aboundingly in TVML and they are good enough to handle most of conventional visual grammar well. Needless to say, good use of these existing commands alone can elevate the level of dramatic rendering of a scene to a new height.

However, it is almost impossible to type in dialogue and control avatar's actions, camerawork, and sound effects together at the same time even though the exquisite harmony of them is capable of creating breathtakingly impressive scenes. Also, most of the time, it is better not to take much time and effort from users to leave them concentrate on their conversation.

The only way to bail out of this antinomic situation is to automatically generate TVML commands based on the content

of dialogue. The best way would be identifying semantics of dialogue, but we are only at the very early stage of developing system and have had no relevant easy-to-apply technology. Therefore, as the first step, we used keyword approach which searches predefined words or signs in the dialogue for the activation of the automatic scene creation. If a predefined word or sign is found in a avatar's speech, the chat server automatically sends predefined TVML commands developing an impressive scene. The examples of this technique are shown in Figure 12. (a) of Figure 12 is the combination of facial expression, gesture, and graphic effect triggered by the question mark in the words of the man on the right. (b) of Figure 12 is the combination of camerawork, facial expression, and gesture triggered by the exclamation mark in the words of the man on the right. Definitions of automatic scene creation are inserted on the top of a character template and stage template.

(a) Automatic scene creation triggered by '?' in the dialogue

(b) Automatic scene creation triggered by '!' in the dialogue

Figure 12. Examples of automatic scene creation

The picky part of defining automatic scene creation is that since each character has its own gestures, some might not have certain gestures which others have, and every studio set has different cameras and lightings. For this reason, the definition of automatic scene creation must be splitted into two parts, one for character template, the other for stage template. Figure 13 and Figure 14 are examples of the automatic scene creation scripts for (a) of Figure 12. The scripts are marked as comments for the same reason as CG character's facial expression list, and words in bold type are indicators.

'?' in (1) of Figure 13 and (2) of Figure 14 is the keyword for activation of TVML commands in (3) of Figure 13 and (4) of Figure 14. (2) of Figure 13 and (3) of Figure 14 are information for users telling what happens if they include the keyword in their dialogue. 'CHARACTER'S SPEECH' indicator in (3) of Figure 13 is where the avatar's TVML talk-command which includes the keyword is inserted. 'CHARACTER'S ACTION' indicator in (4) of Figure 14 marks the position where the TVML scripts in (3) of Figure 13 is inserted when the relevant automatic scene creation is activated. Besides these two simple examples of '?' and '!', various emoticons like (-_-;), (-.-), (T_T), and (>.<) and their corresponding character actions can be defined and used.

In the case where multiple keywords are detected in an avatar's speech, to prevent confliction between different automatic scene creation procedures, the priority is given to the keyword appearing first in the stage template's automatic scene creation list.

```
// DEFINE AUTOMATIC SCENE START
// DEFINE KEYWORD START                                          (1)
// ?
// DEFINE KEYWORD END
// DEFINE DESCRIPTION START                                      (2)
// Udekumi & Shaking Head
// DEFINE DESCRIPTION END
// DEFINE CORRESPONDING TVML SCRIPT START                        (3)
// character: expression( name=A, type=shock)
// CHARACTER'S SPEECH
// character: keyframe(name=A, keyframename=R-udekumi, track=1, stopmode=remain, wait=no)
// character: action( name=A, action=no, speed=1.5, degree=5.0, wait=no )
// DEFINE CORRESPONDING TVML SCRIPT END
// DEFINE AUTOMATIC SCENE END
```

Figure 13. Automatic scene creation for character template

```
// DEFINE AUTOMATIC SCENE START
// DEFINE TARGET PERSON START                                    (1)
// PERSON ON THE RIGHT
// DEFINE TARGET PERSON END
// DEFINE KEYWORD START                                          (2)
// ?
// DEFINE KEYWORD END
// DEFINE DESCRIPTION START                                      (3)
// Question Mark Superimpose & Sound Effect
// DEFINE DESCRIPTION END
// DEFINE CORRESPONDING TVML SCRIPT START                        (4)
// camera: switch( name=CamD)
// CHARACTER'S ACTION
// sound: play(name=hatena, repeat=1)
// super: on(type=preopenimage, name=hatena, x=510, y=210, layer=2, action=fadein, frame=2)
// wait(time=1.5)
// super: off(name=hatena)
// DEFINE CORRESPONDING TVML SCRIPT END
// DEFINE AUTOMATIC SCENE END
```

Figure 14. Automatic scene creation for stage template

4. Discussions

The prominent merit of our chat system is automatic scene creation capability which can provide outstanding audiovisual sensation beyond avatars' facial expressions and gestures by means of the complex combination of dynamic camera work, sound effect, graphic, and other TVML effects. Also, this can be done without any of user's effort. The system automatically detects predefined features in user's dialogue and inserts the predefined TVML scripts for dynamic scene creation between users' dialogue. Furthermore, this predefined TVML scripts for automatic scene creation exists as an editable texts in stage template, and those texts can be easily revised to fit the up-to-date stagecraft. This dynamism and flexibility of automatic scene creation is more than enough to stir up users' emotions and make extraordinary user experience deeply impressed on their minds.

Another merit of our system is its data structure for analysis and manipulation of TVML script. This data structure makes it easy to create, edit, and delete TVML script. Besides, it can

exercises its own critical faculties if the final result of a chat session is to be published through network. When an animated chat session is to be released publicly, typographical errors in conversation should be corrected, excessive repetition of meaningless words or gestures must be modified, etc. In these finishing works, the data structure can serves as a useful tool for the smooth running of the editing process.

5. Conclusions

In an effort to articulate our emotions which written or spoken language sometimes fails to clarify, we developed a prototype TVML avatar chat system to make communication equipped with the most vivid and lively language, "visual language". The three key components of our system, "XML Style Message Format" between the chat server and the chat client, "TVML Data Structure" for analysis and manipulation of TVML script, "Predefined Description Format" for producing suitable direction automatically, is so versatile and competent that TVML avatar chat could be expanded as far as we can imagine, whatever we may dream of.

Our experiment system is no more than a tiny step towards the extension of TVML to the vast continent of audiovisual language for emotional expression. To mature and expand the horizon of TVML avatar chat, it must be aided with more powerful technology to identify semantics of the conversation. Undoubtedly, semantic identification will significantly boost the level of automatic scene creation, and consequently catalyze emotional communication among chat participants.

Secondly, the abstract nature of our experimental chat system can be considered as a collaborative 3D animation making tool. Hence, if we combine TVML avatar chat with other existing TVML based services like "Connected Studio"[4], TVML avatar chat could also serve as an authoring tool for other various purposes other than chat itself.

To sum up, our future work will be focused on identifying semantics of the conversation to make the TVML avatar chat fluent in visual language, and making it collaborative authoring tool for various SNS platforms to pave innovative ways of connecting people through CG contents.

Acknowledgement

This TVML avatar chat research was supported under NHK STRL's ABU(Asia-Pacific Broadcasting Union) program.

Reference

1) Douke, Ariyasu, Hamaguchi, Hayashi,"Production of TV Programs On A Single Desktop PC –Special Scripting Language TVML Generates Low-Cost TV Programs-", BroadcastAsia2002 International Conference Worldcasting Book 2 of 2 (2002)
2) Hamaguchi, Kaneko, Doke, Inoue, Kumazawa,"Live Text-to-Video System Using Real-time Server-side Rendering", International Workshop on Advanced Image Technology (IWAIT2011) Proceedings (CD-ROM) (2011)
3) 井上, 金子, 浜口, 道家,"TVML プレイヤー外部制御機能の高機能化", 映像情報メディア学会年次大会, 講演予稿集 17-4 (2008)
4) 道家, 金子, 浜口, 井上, "コネクティッドスタジオの試作 ~多人数仮想参加型番組生成システム~", 映像情報メディア学会技術報告 ME2012-53, pp.171-176 (2012)