A Testbed System for Multimodal Presentation Markup Language in Flash Medium with Avatar Control

Zhenglu Yang Mitsuru Ishizuka
Graduate School of Information Science and Technology
University of Tokyo

Abstract

Nowadays, employing realistic-looking virtual humans as presenters in various situations becomes more effective and important. To make it easier for the author to write such multimodal presentation, we have developed MPML (Multimodal Presentation Markup Language). By extending MPML, we have developed MPML-FLASH (Multimodal Presentation Markup Language for Flash), which facilitates multimodal presentation in Flash medium. In this paper, we present a framework as the testing platform and related techniques for MPML-FLASH.

1. Multimodal Presentation Markup Language for Flash (MPML-FLASH)

The Multimodal Presentation Markup Language [1] which is developed at the Ishizuka Lab, Tokyo University, is an XML styled markup language designed to allow authors to easily script animated agent presentations.

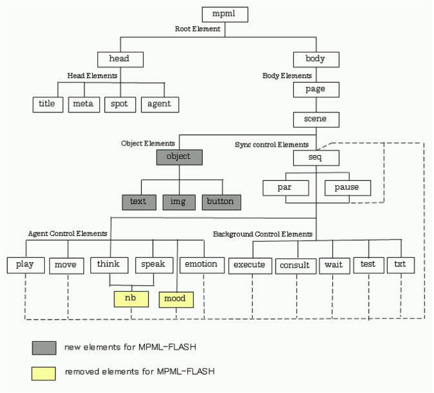


Fig.1: MPML-FLASH tag tree

By extending MPML, we have developed MPML-FLASH to control character in Flash medium. The XML tag tree of MPML-FLASH is shown as Fig.1. The new elements in MPML-FLASH compared with MPML are described in the gray box. And the yellow box shows the removed elements in MPML-FLASH. The detailed specification is available at the web site [1].

2. Lip Synchronization

In our system, we employ Festival Speech Synthesis System [2] to produce the audio stream that will be subsequently played back in synchronization with the facial animation and the temporized phonemes, which are used to create the corresponding visemes to fulfill the lip synchronization.

3. Finite State Machine

To control the character, we use a classic technique called FSM (Finite State Machine). Fig.2 illustrates the structure of a Body FSM. The body of character is divided into several parts. The parent image is rendered with a “hole” punched out of it, and then the child image is drawn into that hole. Different child parts, such as the mouth and eyes, can be drawn onto the same head parent.

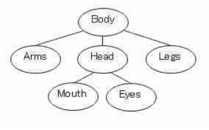


Fig.2: Body FSM structure

State/transition Diagrams

Fig.3 is a simple state machine that defines the turning action of the character. The circles represent states and a character can remain in a state for any period of time. The arrows denote transitions which contain several frames that will bring a character from one state to another.

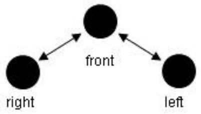


Fig.3: Turning FSM

The character’s animation library is written as XML files in which define all the states and transitions of the character.

4. Chatterbot

Chatterbots are conversational agents engaging in a natural language-based interaction with user. In our system, we use AIML [3] as our chatterbot language.

5. System Structure

5.1. Pre-script Based Model

As shown in Fig.4, the author first creates MPML at the client side and then transfers it to the server, where the corresponding Flash is created based on the techniques presented earlier. When different users request from their own client side some time later, the server will send the same pre-script Flash to them.

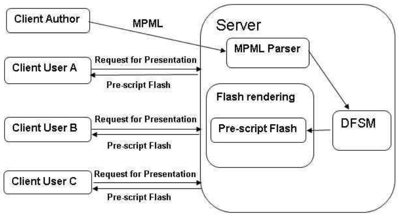


Fig.4: Script based system structure

The detail work flow of the Flash rendering is shown as Fig.5. The MPML parser assigns the work to different parts such as the background rendering, the character body rendering and the Text-To-Speech according to MPML script. Based on the phonemes, which are created from input texts, the system synchronizes the lip of the character and uses FSM to model the character. At the end, it combines the three rendering parts to get the final rendered Flash presentation.

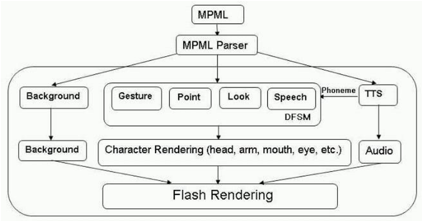


Fig.5: Script based system work flow

5.2. Chatterbot Based Model

In this modal, we hope the character acts like a real person based on his own judge in the virtual circumstance not on predefined plot. In other words, it means an autonomous character behaves properly to the users.

First we employ the natural language processing technique, say chatterbot, as the "brain" of the character. Second we use non-deterministic finite state machine method, random actions like blinking or flexing hand, to the character to achieve the autonomous goal.

As shown in Fig.6, different users should ask different questions to the server. Based on the domain knowledge provided by the author using AIML and general knowledge provided by system, corresponding answers will be created dynamically and sent back to the users. The detail work flow is similar to the script based model yet the dynamic rendering part is included.

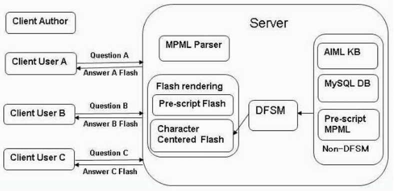
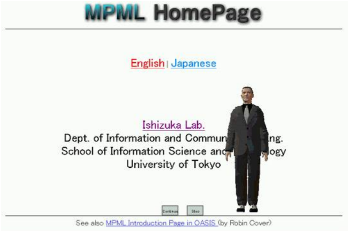


Fig.6: Character based system structure

6. Demo

We present a live demo (Fig.7) on the web [1] to show our system. In this demo, we present some scenarios to introduce our MPML project.


MPML HomePage

English | Japanese

Ishizuka Lab.
Dept. of Information and Commun ng.
School of Information Science and ogy
University of Tokyo

See also MPML Introduction Page in OASIS (by Robin Cover)

Fig.7: Demo snapshot

7. Evaluation and Future Work

MPML-FLASH employs Client-Server model that the Flash file is created at the server side. By this way we reduce the burden of the user's side while increasing the burden of the server's side, which can be solved by setting a powerful configuration such as more memory, higher CPU and even multiple CPUs.

At present, the Flash character has 22 animations. If users want to get some new actions, the character designers such as us should do this job at the server side so that all users can use it and do not need to download the character file.

We will create more actions for the character. Additionally, we plan to promote MPML-FLASH that uses it to control Flash which is created by Macromedia Flash Tool.

References

[1] MPML project, Retrieved March 8, 2003 from http://www.miv.t.u-tokyo.ac.jp/MPML

[2] A. Black and P. Taylor, Festival Speech Synthesis System: system documentation(1.1.1), Human Communication Research Centre Technical Report HCRC/TR-83, UK, 1997.

[3] R. S. Wallace, Don't read me – A.L.I.C.E. and AIML documentation, Retrieved March 8, 2003 from http://alicebot.org/articles/wallace/dont.html