A Software Framework for Developing Mathematical Model Driven Virtual Human

Chao Mei*
Department of Computer Science
The University of Texas at San Antonio

John Quarles*
Department of Computer Science
The University of Texas at San Antonio

ABSTRACT

Virtual Humans (VH) have many benefits, such as simulating humans when using real humans is difficult, impossible, or dangerous. VHs realism in behavior and response may be improved by mathematical models, which can provide dynamic responses and interactions and have the potential to be widely applied in training and education, such as medical training (e.g. physiological models of blood flow). However, the difficulties of developing mathematical model driven VHs may restrict the application of it, especially in the areas where the professionals do not know how to program (e.g. doctors). We present Mathematical Virtual People (MVP), which is a software framework that may simplify the developing job and encourage the applications of VHs driven by mathematical models, such as the drug reaction models and the cardiovascular system models.

Keywords: Virtual Human, Mathematical Model, Software Framework.

Index Terms: H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems-Artificial, augmented, and virtual realities; D.2.2 [Software Engineering]: Design Tools and Techniques-Modules and interfaces

1 INTRODUCTION

Virtual Humans (VH) may serve as substitutes [1] for real humans in many cases when using real humans is difficult, impossible, or dangerous (e.g. military training [2], medical education training [4] [15]). Often in these applications, VHs are pre-scripted. For example, in the NERVE [14] system developed by the Virtual Experiences Research Group (VERG) group in the UF, the VHs have pre-scripted doctor-patient conversation reactions. When the sentence such as "How are you feeling?" is input, the VH would react with "I am doing well, thanks for asking!" Other approaches can provide dynamic responses, such as VHs controlled by human operators. For example, considering a scenario in which a VH is undergoing an anesthesia procedure, the simulated patient could be any age, weight, height, race, gender etc. Users may give arbitrary amount of sedation drug to the VH, and small change in any variable (age, weight, etc.) may make the drug reaction different. In this case, experienced professionals (anesthetists) can operate the VHs. However, the operators may not be available all the time and pre-scripting all possible reactions will be complex - it cannot be a substitute for the real human professionals.

VHs driven by mathematical models, such as the VH described by Jendrusch et al [4], can potentially provide dynamic responses and be more accessible than human operators.

Mathematical models have already been very helpful tools in medical training and practice. They can precisely describe and predict human body systems and activities, such as human anesthetic drug reactions [4] and cardiovascular system processes [11], using mathematics. In those models, the number of the variables that could be manipulated is huge, and they all interact with each other. For example, CVSim-21[11] is a mathematical model of the human heart. It simulates blood flow through the different parts of the heart (e.g., left/right ventricle, venous and arterial etc.). It is all based upon known equations of venous pressure-volume relationships. The user can control many variables, such as blood volume. The outputs from the CVSim-21 include many other variables such as blood pressure. A simple use case of the simulator could be: a student increases blood volume variable, which causes higher blood pressure when other variables remain the same.

There are potential benefits of VHs driven by mathematical models. With this approach, efforts could be saved in scripting the responses or finding an expert who is always available to operate the VH. In comparison, they are potentially more available than real expert operators of the VH and inherit the advantages from both pre-scripted VH and mathematical models (e.g. VH moves and speaks like a patient suffering from heart attack, only when the outputs from the cardiovascular model-Cvsim-21 describe so). Mathematical model driven VHs could potentially improve the quality of teaching and training. Although there are a large amount of successful mathematical models that have been developed and applied in medical training, there are few VHs driven by mathematical models besides Jendrusch's work [4]. The reasons could be: 1) mathematical model driven VHs are a new concept still not well known by people, and 2) based on our communication with medical trainers, they cannot do the development work by themselves, and in addition, the mathematical models evolve frequently which will make the VH systems change frequently. Thus, the complexities of developing mathematical model driven VHs may impede the benefits they could bring. To address these limitations and to expand the known range of effective uses for VHs, a novel, more general approach to develop mathematical model driven VHs is presented. In this paper, we present Mathematical Virtual People (MVP), which is a software framework to simplify the development of mathematical model driven VHs. We present the design of the framework and case study of an application developed with MVP.

2 RELATED WORK

In this section, we overview related work that motivates the need for a software framework for mathematical model driven VHs.

* meichaomc@gmail.com
* John.Quarles@utsa.edu

2016 IEEE Virtual Humans and Crowds for Immersive Environments (VHCIE)
20 March, Greenville, SC, USA
978-1-5090-0829-2/16/$31.00 ©2016 IEEE

2.1 Virtual Human

Virtual Humans (VH) have been widely studied and used especially in Virtual Environment (VE) for education [1]. L.D. Wandner et al [3] used VH technology to investigate the effects of VH patients' demographic cues on dentists' pain management decisions. Using VHs makes it possible to control the conditions of the environment in this study. The Mission Rehearsal Exercise trainer [2], developed by USC's ICT group, is designed to teach critical decision-making skills to small-unit leaders in the U.S. Army. The extensive application of VHs in this system could effectively reduce the cost and ensure the security of such training. Jendrusch et al [4] integrated VHs with the mathematical model of propofol, which simulates the drug effects of propofol to human bodies, so that the outputs of that mathematical model are observable directly through the VH's animations and audio. This is an example of when it is very dangerous to train drug dosing skills on a real person. In addition, there are some rare cases such as patients who are very sensitive or insensitive to the drug of propofol. It would save much more efforts to build a VH than finding a real person with such properties.

2.2 Frameworks of Virtual Environment Development

Although VHs have brought many benefits, it is difficult and time consuming for the researchers to set up, script, and test VHs. e.g. Developers may need the VH to groan when the propofol dose is not enough- when the VH feels the pain. The VH should adjust breathing to a low rate when it is fully sedated. Development work of such VHs with simulated intelligence have to deal with the 3D rendering processes, the way to set up the rules of animation state switching, the interactions with the 3D environments, the tools that are used to help with the interactions and the integration and management of these modules.

Using software frameworks to help develop the VE could effectively reduce the time of development, since the frameworks can help maximize the code reuse. In addition, from an engineering perspective, the careful analysis, design, implementation, testing and maintenance of the VH frameworks could result in more VR applications with high quality. There are several existing software frameworks for simulations [5][6][8][10][12]. For example, OpenViBE [8] is a software framework with which researchers could design, test, and use brain–computer interfaces (BCI) in VE. Avango, developed by Tramberend [5], is a distributed framework for Virtual Reality applications. By encapsulating and extending technologies of dynamic 3D scenes, interaction devices, display setups, distributed VR and other VE frameworks, the Avango enabled large range of application development, from desktop PCs to large-scale immersive VR installations. Shared Simple Virtual Environment (SSVE) [6] is another example of a Virtual Reality framework. Its features are mostly concentrated on the aspect of group work in VEs. Applications that use this framework could be used for the research of group collaboration mechanisms in highly-interactive shared VE. These frameworks may help to reduce the development time and improve the quality of applications.

On the specific frameworks for VH development, the choices are very limited. Ponder introduced a VR/AR framework with components that could help developing rendering effects and animations of VH [10]. Faller et al presented a framework that could help to use the Steady-state visual evoked potentials (SSVEPs) signal, which are brain signals used in brain–computer interfaces(BCI), to control Avatar. Shoulson et al presented ADAPT which is a framework to incorporate character animation, navigation, and behavior with modular interchangeable components to produce narrative scenes [16]. Frameworks as such are rarely seen despite its wide range of applications. The existing ones are usually too old, so that they focused on different problem domain with today's, designed for professional VH developers (e.g. ADAPT [16]) but not medical professionals, or restricted to cases (e.g. integrating BCI) other than supporting mathematical models to drive the VH.

2.3 Mathematical Models

Using a mathematical model to simulate human body activities is a common approach for teaching students about the human body. For example, CVSim is a lumped-parameter model (i.e., a model that simplifies the distributed system into a discrete entities) of the human cardiovascular system. CVSim has been used for research and for teaching quantitative physiology courses at MIT and Harvard Medical School since 1984[11]. PK/PD models are mathematical models that simulate "what the body does to a drug" (pharmacokinetics), and "what a drug does to the body" (pharmacodynamics) with mathematical expressions and outputs the real time drug effects. There are many other mathematical models of the processes in the human body [13].

Although these models are helpful to use in teaching, more methods, such as integrating VH, of visualizing the data are needed to observe and understand them. However, frameworks and applications connecting the models to VH simulation are seldom found [4]. We hypothesize that this is because of the lack of a general framework for developing them. A software framework could serve as such an approach that would reduce the difficulties of the development of mathematical model driven VHs, and promote the usage of such applications in education.

3 System Design

3.1 System Overview

The main purposes of the MVP framework are: 1) connecting the mathematical models with VHs, 2) enabling the users who do not have programming skills, such as medical professionals, to easily set up the rules of how should the VHs react to the mathematical model outputs and 3) facilitating the communications between the users who set up rules and the users who are the developers of the VH.

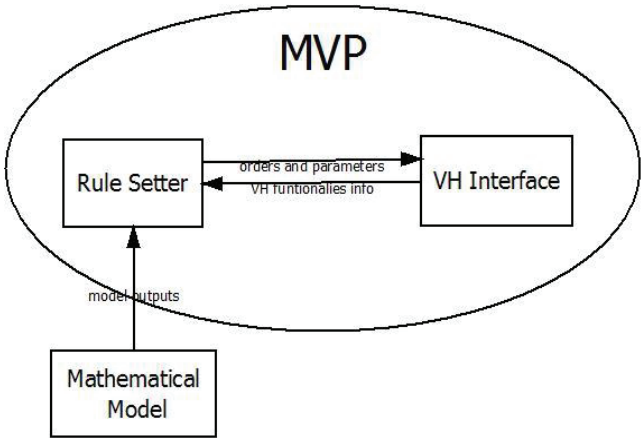

Figure 1. The top level structure of MVP

MVP is developed based on Java and Unity 3D platforms. Figure. 1 shows the main structure of MVP. It consists of two parts – VH Interface and Rules Setter. VH Interface is a Unity

package, with which the VH developers can define the functionalities that a VH can provide. The Rules Setter part provides a GUI for application domain professionals (such as medical trainers) to set up the mathematically driven rules to control the VH. These two parts connect to each other through TCP network sockets. The mathematical models that are compatible with the MVP framework should also provide its outputs through TCP network sockets. All the technologies and the platforms (sockets, Java, Unity) used to implement the MVP are not restricted to a specific operating system, therefore the MVP system is a cross-platform framework.

When the MVP is running, the Rule setter first accepts the inputs from the mathematical models and pulls the information of the VH's functionalities from the VH Interface, then it calls the functions of the VH according the mathematical model inputs and the rules have been set.

3.2 VH Interface

The VH Interface is a Unity package which defines a protocol for the VH side to provide the VH's functionality information to the Rule Setter, as well as receive orders from the Rule Setter. It automatically handles the network communication, so that the VH developers only need to focus on setting up the interactive functions of the VH.

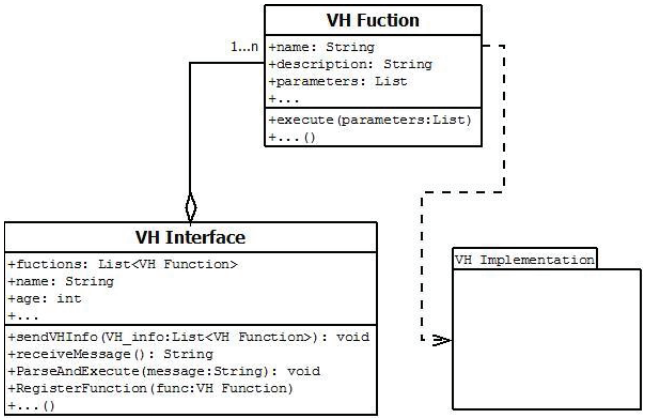

Figure 2. Class diagram of VH Interface

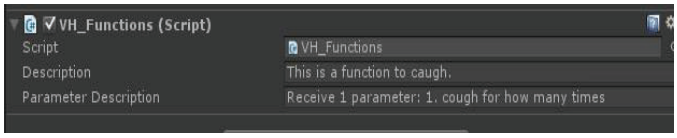

Figure 3. Set up the description of a VH Function in Unity Inspector

Figure 2 shows the class diagram of the VH Interface. The VH Interface class contains a list of VH Functions that the VH can provide to the Rule Setter side. These functions are decided through the discussion between the VH developer and the application domain developer. Each function should be abstracted to the application domain so that the application domain developer can understand and utilize the function easily. For example, for the application domain developer who is a doctor, there may be a VH function with the name of "cough" and take an integer indicating frequency of cough. This function should encapsulate many lower level functions such as find cough animation, calculate and set up animation speed, play animation, play sound, set up reply etc. To set up a VH Function, the VH developer creates a subclass of the VH Function and attaches it as a script in Unity. In the Unity inspector as shown in Figure 3, the VH developer sets up the description of the function, such as name, usage and parameters taken. This information will be send to the rules developer (application domain developer) as documentation that can assist with their development. In addition, the VH developer should override the "execute" function inherited from the VH Function class to handle the parameters sent from the Rule Setter and call the corresponding lower level functions from the VH's original implemented package.

3.3 Rule Setter

The Rule Setter receives the information sent from the mathematical model and the VH Interface side and displays the

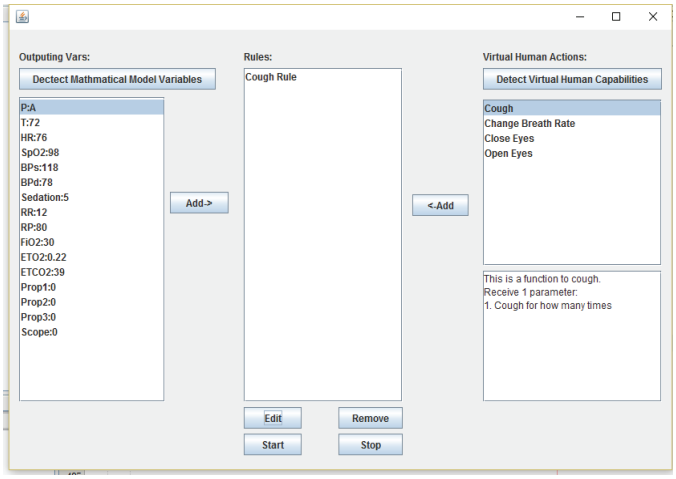

Figure 4. Main GUI of the Rule Setter

information on its main GUI as shown in Figure 4. The left list shows the variables input from the mathematical model. The right side shows the VH functions that are available to use and the description of the selected function. The rule developer is free to select any input variable or VH function and uses the "Add" button to associate a rule with the selected variable/function.

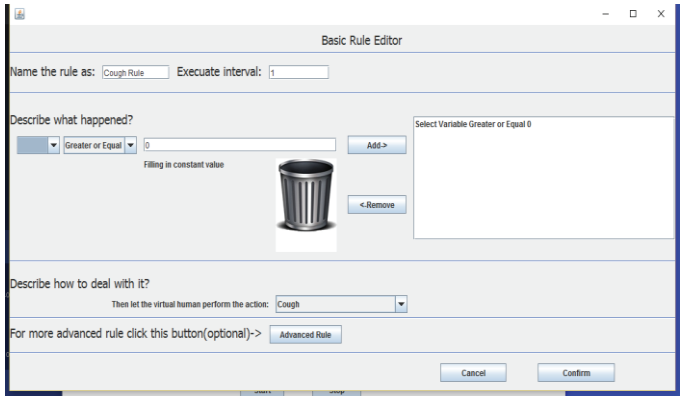

Figure 5. The GUI of the Rule Editor

The rule can be added, removed and edited. To edit a rule, the developer uses the rule editor as shown in Figure 5. There are two types of rules that can be edited with this GUI. First, when one or more of the input variables reaches its threshold, then it will send

the command to execute the associated VH function. For example, when the Rule Setter receives the sedation level greater or equal to 3 and less than 5 from the mathematical model, it will send a command to open the eyes with parameter of 0.5 (half way open). Second, the rule setter application directly sends the command to execute a VH function whenever it receives the associated variable(s). For example, change the heart rate of the VH at every frame the Rule Setter receives the heart rate variable. The parameter of the VH function can be any input variables or values defined by the rule developer. In addition, for rule developers who knows how to code, we provide another rule editor which allows the developers to pre-process the input variables with Java code.

4 Case Study

4.1 CVSim-21

CVSim-21[11] is a mathematical model of the human heart. We have created a mathematically driven VH using CVSim and MVP. Here we present this application as a case study of MVP.

Figure 6. Normal State

According to [11], certain traumatic injuries (e.g. gunshot, knife wound, etc.) to regions where arteries and veins run together (such as the femoral area) may lead to the development of significant arterio-venous fistulas (i.e. "short circuits" between artery and vein). A fistula will dramatically lower total peripheral resistance. We simulated the AV fistulas on a VH. The VH has the abilities to show 3 kinds of fistulas – the fistula on lower, middle and upper side of his arm (the closer the position to the heart of the fistula, the lower peripheral resistance the human would have). The VH is designed with the VH functions showing the fistulas at different positions. Each function should be executed at a correct moment, so we used the MVP framework to set up its rules. The rules are set up as: 1) when peripheral resistance > 0.9, the VH shows normal state (Figure. 6), 2) when 0.9 => peripheral resistance > 0.6, the VH shows the fistula at lower side of the arm, 3) when 0.6 => peripheral resistance > 0.3 (Figure. 7), the VH shows the fistula at middle side of the arm, and 4) when 0.3 => peripheral resistance, the VH shows the fistula at upper side of the arm.

Setting up the rules with MVP takes only a few minutes. In addition, users are free to modify the rules to adjust the value at any time, so that they could demonstrate different effects of the fistulas to different people (e.g. the people with higher blood fat/longer blood vessels may still have higher peripheral resistance than normal people when some fistula occurs). Moreover, the users do not have to know any programming languages to implement the rules.

Figure 7. Peripheral resistance=0.6

5 Discuss and Conclusion

In this paper, we present the MVP, which is a software framework to simplify the development of mathematical model driven VHs and encourage new applications for mathematical model driven VHs. Through the case study of an application developed with MVP, we found that MVP can reduce the time it takes to connect mathematical models with VHs, and potentially enable the users who do not have programming skills, such as medical professionals, to easily set up the mathematically driven rules of how VHs should react to the mathematical model outputs. In addition, we expect that the MVP will facilitate communication between the rules developer and the VH developer, since it defines the interface for their collaboration. For example, if the CVSim-21 model changes its heart rate output from per minute to per hour, the rules developer can easily point out which VH function(s) is affected and should be modified by the VH developer.

In future work, we plan to expand the benefits of VH by creating more mathematical model driven VHs with the MVP framework. In addition, we plan to extend the Rule Setter to handle more complex rules.

References

[1] Johnsen, Kyle, Andrew Raij, Amy Stevens, D. Scott Lind, and Benjamin Lok. "The validity of a virtual human experience for interpersonal skills education." In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 1049-1058. ACM, 2007.

[2] Hill Jr, Randall W., Jonathan Gratch, Stacy Marsella, Jeff Rickel, William R. Swartout, and David R. Traum. Virtual Humans in the Mission Rehearsal Exercise System. *KI,* 17(4) . 2003

[3] Wandner, L. D., A. T. Hirsh, C. A. Torres, B. C. Lok, C. D. Scipio, M. W. Heft, and M. E. Robinson. Using virtual human technology to

capture dentists' decision policies about pain. *Journal of dental research* , 92(4), pages 301-305. 2013.

[4] Jendrusch, Jason, Samsun Lampotang, David Lizdas, Nikolaus Gravenstein, Dwayne Ham, Benjamin Lok, and John Quarles. Virtual humans for inter-ethnic variability training in sedation and analgesia. *Medicine Meets Virtual Reality 21: NextMed/MMVR*, 21(196), page 175. 2014.

[5] Tramberend, Henrik. Avango: A Distribuited Virtual Reality Framework. 2000

[6] Linebarger, John M., Christopher D. Janneck, and G. Drew Kessler. Shared simple virtual environment: An object-oriented framework for highly interactive group collaboration. In *Proceedings of the Seventh IEEE International Symposium on Distributed Simulation and Real-Time Applications.* pages. 170-180. IEEE, 2003.

[7] Faller, Josef, Gernot Müller-Putz, Dieter Schmalstieg, and Gert Pfurtscheller. An application framework for controlling an avatar in a desktop-based virtual environment via a software SSVEP brain-computer interface. *Presence: teleoperators and virtual environments,* 19(1), pages 25-34. 2010.

[8] Renard, Yann, Fabien Lotte, Guillaume Gibert, Marco Congedo, Emmanuel Maby, Vincent Delannoy, Olivier Bertrand, and Anatole Lécuyer. OpenViBE: an open-source software platform to design, test, and use brain-computer interfaces in real and virtual environments. *Presence: teleoperators and virtual environments* 19(1), pages 35-53. 2010.

[9] Machado, Liliane S., Ronei M. Moraes, Daniel FL Souza, Leandro C. Souza, and I. L. L. Cunha. A framework for development of virtual reality-based training simulators. *Studies in Health Technology and Informatics*, pages 174-176. 2009.

[10] Ponder, Michal, George Papagiannakis, Tom Molet, Nadia Magnenat-Thalmann, and Daniel Thalmann. VHD++ development framework: Towards extendible, component based VR/AR simulation engine featuring advanced virtual character technologies. In *Computer Graphics International, 2003. Proceedings*, pages 96-104. IEEE, 2003.

[11] Heldt, Thomas, Ramakrishna Mukkamala, George B. Moody, and Roger G. Mark. CVSim: an open-source cardiovascular simulator for teaching and research. *The open pacing, electrophysiology & therapy journal,* 3, page 45. 2010.

[12] Mei, Chao, Xiaoyang Zhang, Wenfei Zhao, Kasi Periyasamy, and Mark Headington. A tool for teaching Petri nets. *Journal of Computing Sciences in Colleges,* 26(5), pages 181-188. 2011.

[13] Physiologic models and simulations, 2013 http://physionet.org/physiotools/software-index.shtml#pms

[14] Aaron Kotranza, Juan C Cendan, Kyle Johnsen, Benjamin Lok. Virtual Patient with Cranial Nerve Injury Augments Physician.-Learner Concern for Patient Safety. *Bio-Algorithms and Med-Systems,* 6(11), pages 25-34. 2010

[15] Mei, Chao, Lee Mason, and John Quarles. How 3D Virtual Humans Built by Adolescents with ASD Affect Their 3D Interactions. In *Proceedings of the 17th International ACM SIGACCESS Conference on Computers & Accessibility*, pages. 155-162. ACM, 2015.

[16] Shoulson, Alexander, Nathan Marshak, Mubbasir Kapadia, and Norman I. Badler. Adapt: the agent developmentand prototyping testbed. *IEEE Transactions on Visualization and Computer Graphics*, 20(7), pages 1035-1047. 2014