CoPuppet: Collaborative Interaction in Virtual Puppetry

Paolo Bottoni[1], Stefano Faralli[1], Anna Labella[1], Alessio Malizia[2], Mario Pierro[1], and Semi Ryu[3]

[1] Department of Computer Science, University of Rome "La Sapienza"
{bottoni,faralli,labella,pierro}@di.uniroma1.it
[2] Universidad Carlos III de Madrid
alessio.malizia@gmail.com
[3] Virginia Commonwealth University, Richmond, VA, USA
sryu2@vcu.edu

Abstract. Puppetry is one of the most ancient forms of representation, diffused all over the world in different shapes, degrees of freedom in movements and forms of manipulation. Puppets make an ideal environment for creative collaboration, inspiring the development of supporting technologies (from carpentry to digital worlds). The CoPuppet project explores the possibilities offered by multimodal and cooperative interaction, in which performers, or even audience members, are called to affect different parts of a puppet through gestures and voice. In particular, we exploit an existing architecture for the creation of multimodal interfaces to develop the CoPuppet framework for designing, deploying and interacting during performances in which virtual puppets are steered by multiple multimodal controls. The paper illustrates the steps needed to define performances, also showing the integration of digital and real puppetry for the case of wayang shadowplay.

1 Introduction

Puppetry is one of the most ancient forms of performance, diffused all over the world in different shapes, degrees of freedom in movements, and forms of manipulation. While forms and techniques of puppetry are varied and related to specific local settings, its essential spirit is universally found in aspects such as traditional oral storytelling, live improvisation, mixed reality, and public engagement, coherently with its roots in ancient rituals.

By developing the CoPuppet project, we aim at creating a digital translation of puppetry, with emphasis on oral storytelling, free improvisation and public engagement. We propose new ideas for collaboration between users (puppeteers), to enhance the original spirit of puppetry, using interactive tools such as virtual sensors - here intended as maps of physical phenomena onto a virtual support - and voice activation.

We explore here the possibilities offered by multimodal and cooperative interaction with puppets in constructing a communicative experience among performers, or even audience members, called to affect different parts of a puppet

R. Adams, S. Gibson, and S. Müller Arisona (Eds.): DAW/IF 2006/2007, CCIS 7, pp. 326–341, 2008.

through gestures and voice. The proposal is based on the use of the Chambre architecture for the creation of multimodal interfaces [1,2], in which users can interact with multimedia sources through traditional or multimodal widgets. Chambre can also integrate virtual reality environments, and users may affect them through modifications of their parameters by gesture or voice commands. As a result, CoPuppet allows the creation of "live improvised" storytelling and actions between puppeteers.

In the rest of the paper, we set the scene of traditional puppetry in Section 2 and propose a brief review of related work in Section 3, while the Chambre architecture exploited in CoPuppet is presented in Section 4. Section 5 presents the CoPuppet application, while Sections 6 and 7 discuss its configuration and illustrate a performance setting and development. Section 8 explores interactions between digital and traditional puppets, with reference to wayang shadowplays. Finally, Section 9 draws conclusions.

2 Background

Puppets are diffused all over the world in different forms and used in different ways. Aa an example, Neapolitan puppets (known as Punch and Judy in the Anglo-Saxon world) are operated by single hands fitting inside the puppets, while Sicilian ones may be significantly sized, with different components steered via threads moved from above. Shadowplays, such as the Javanese wayang kulit, or the Turkish Karagöz, happen behind screens with the puppets' limbs controlled via horizontally held sticks. The Prague Black Theater incorporates puppets and real actors whose body parts are made invisible by wearing black clothes. Korean puppetry works with rods, and sometimes hands inside, with limited control, showing very primitive expressions. One of the puppeteers sits in front of the stage with the audience and talks constantly with the puppet during the play, breaking the boundary between the puppet and the real world. In the Japanese Bunraku puppet theater, 3 puppeteers in black costumes work together to establish the puppet's emotional expressions and gestures as a whole.

In general, puppets seem to evolve from the hinged and jointed masks worn by shamans, to become shamanic objects used in rituals, their role being to favor the shaman's trance state. This progression from masks, to puppet masks, to marionettes, appears to have occurred in a number of primitive societies [3].

Oral storytelling is an important aspect of ritual and puppetry. Puppeteers were usually illiterate, belonging to lower classes, and the narrative was orally transmitted over a long period of time [4], encouraging continuous improvisation and revisions in a flexible form of storytelling. For example, in Mediterranean area, stories could be improvised or developed under the influence of recent chronicles. Korean puppet drama is also preserved only through oral tradition.

Sometimes the puppet drama has a fixed narrative to be told exactly as trained, especially when it belongs to some noble form of art. For example, before a Bunraku performance in Osaka, the chanter holds up the text and bows before it, promising to follow it faithfully. However, on the near island of Awaji,

a simpler type of folk Bunraku is practiced by the whole community, to less refined results, but bringing excitement and enthusiasm on [3].

3 Related Work

The idea of digital puppetry, as a means to go back to the shamanic roots of the puppetry tradition, had a first instantiation in the virtual interactive puppet performance, "YONG-SHIN-GUD" [5], involving live music and storytelling, together with 3D graphics to represent a virtual puppet. The puppet movements and facial expressions are steered in real time by the sounds captured by a microphone, either produced by instruments or by the storyteller voice. Meanwhile, the virtual puppet constantly speaks and sings back to the puppeteer, as a real-time echo and mirror reflection. The goal of the "YONG-SHIN-GUD" performance was to let the puppeteer eventually enter some form of trance, by spiraling interactive dialogues with the virtual puppet. To the audience, the real-time lip synchronization process appears as a continuous transformation process involving the virtual puppet and the human puppeteer, in a Yin-Yang equilibrium. CoPuppet adopts this continuous process of transformation as a tool to break boundaries and stereotypes, bringing forth freely imagined oral storytelling from puppeteers.

Nowadays, avatars, forms of "digital puppets", are in wide use, in online game environments, or in community-based sites, such as Second Life[1]. The word "puppet" has frequently appeared in the digital realm; however, due to the Western cultural understanding of subject and object, it usually represents something to be manipulated and controlled. The definition of puppet has been narrowly understood in this Western cultural context, and inherited in the digital media culture. In this paper, connecting with the origin of puppetry, we propose virtual puppets as ritual objects, emphasizing puppet-puppeteer relationships, as well as interactions between puppeteers. A combination of storytelling and multimodal interaction is presented in [6], where humans interact with puppets in a virtual environment, in which recognition of human gestures and speech steers dialogues and interaction with the puppets.

The construction of virtual sensors can take advantage of the availability of tools for pattern recognition and motion detection. Currently, original and simple algorithms have been incorporated into Chambre, typically based on finger position identification. Differently from [7], we are not restricted to 2D positioning, but can also exploit (partial) 3D information. The open structure of Chambre and the simplicity of its component model, however, make it easy to embody more sophisticated algorithms.

4 The Chambre Architecture for Multimodal Interaction

Multimodal Interfaces [8] allow users to interact with a system through several input devices, such as keyboard, mouse, voice, face and gesture recognition. In

[1] http://www.secondlife.com/

the design of real-time systems with multimedia capabilities and/or featuring multimodal interfaces, recurrent issues are encountered, such as the design of efficient data acquisition and transmission protocols, of parallel or serial compositions of data processors, of data generation algorithms, and so on. *Software frameworks* [9] offer a reusable set of components and inter-component communication protocols: they are thus often used in the design of multimedia/multimodal systems, as this greatly reduces the system's development time and allows the developer to experiment with different designs quickly. Frameworks often come with a visual editor, allowing the programmer to express the system structure via a graphical language. A graph representation of the system is commonly used, where classes constitute the graph vertexes, while directed edges represent communication channels between them. Several frameworks which exploit this metaphor are currently available both from industrial efforts commercially and through the open-source community, two examples being Virtools [10] and Pure Data [11].

Chambre is an open framework able to accommodate different protocols, sensors, generation and interaction techniques [1]. The generative process can be steered, both explicitly and implicitly, by human users, through inputs acquired by external multisensor devices. For example, a webcam can capture user movements, while image analysis tools can interpret them to detect presence in specific zones or evaluate variations with respect to previous or fixed reference images. As a result, parameters are generated to steer the system response. Specific inputs can also trigger, in real-time, modifications of the interpretation process. This ability makes Chambre a flexible and open tool, easily adaptable to different situations and applications. The Chambre architecture, sketched in Figure 1, allows a component-based style of programming, where components are endowed with communication interfaces and a system results from connecting them. A designer can interactively define a Chambre network in the form of a graph, exploiting the convention described above.

The Chambre framework was started as a distributed component-based architecture for the production of multimedia objects. Hence, it incorporates several plug-ins managing different multimedia data formats [1]. Currently available plug-ins offer: 1) interpretation of stimuli from real sensors (physical measures) such as: webcams, mouse and keyboard signals, etc.; 2) signal generation; 3) signal mapping; 4) mapping-driven multimedia streaming generation (audio and video synthesis), thus favoring the rapid prototyping of Multimodal Interfaces.

Symmetrically to the real case, (groups of) virtual actuators can interact with and modify virtual environments through Virtual Multimodal Interfaces (VMI). In order to manage the state of virtual actuators, the *FeedBack Manager* module sends streams of control commands and data to the *Computational System*.

A VMI is characterised by specific Chambre nodes in charge of mapping real stimuli to virtual ones through an interpretation process. A virtual stimulus is a configuration of a data collection produced by some computational process,

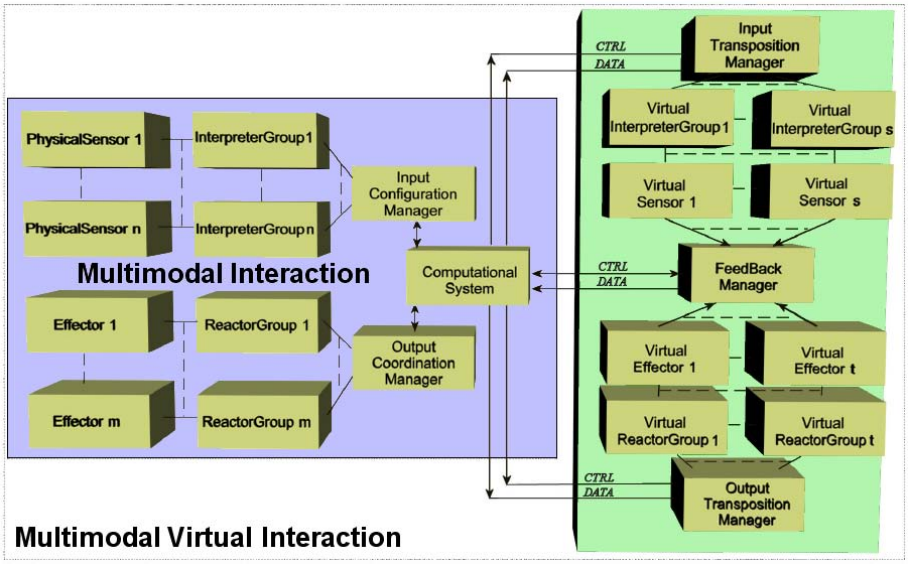

Fig. 1. The architectural model of Virtual Multimodal Interaction

which can be interpreted as the result of a measurement process. In particular, a *Virtual Component* defines an *Appearance*, a *Behavior* and a *Measurement* method. An innovative feature in the CHAMBRE implementation of VMIs is the possibility of positioning virtual sensors onto a virtual support. As an example, a video virtual support can transpose the frames taken from a video stream by sampling (clipping) and positioning them into the virtual space (see Figure 2). As another example, a virtual slider acting onto a video virtual support can perform measurements by detecting the point closest to an extreme P_2, in a segment P_1P_2, which intersects a moving shape, as shown in Figure 3.

Figure 4 shows instances of Virtual Button and Virtual Slider widgets performing their measurements on video supports.

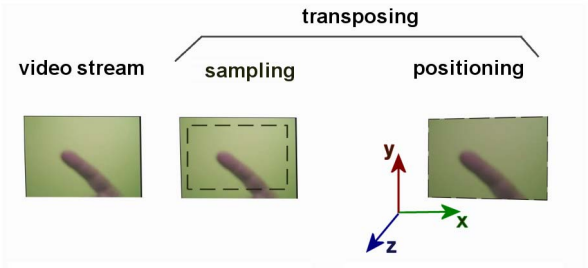

Fig. 2. Construction of a support for a virtual sensor

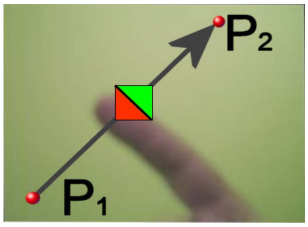

Fig. 3. An example of Virtual Slider

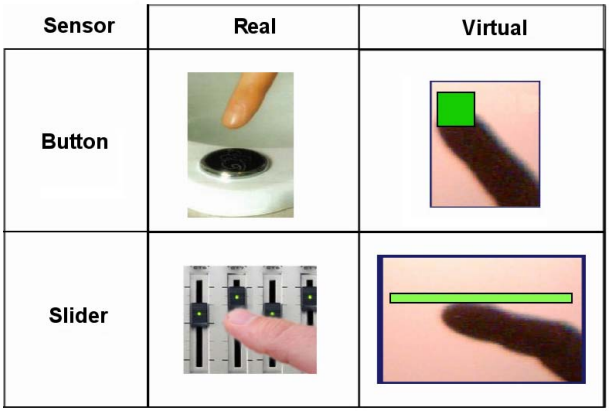

Fig. 4. Examples of Virtual Sensors

5 The CoPuppet Environment for Digital Puppetry

CoPuppet is a class of Chambre applications for Virtual Puppetry. A CoPuppet application is a Chambre network featuring instances of MVI_VirtualPuppet (MVI_VP for short), a Chambre software component which, differently from virtual sensors, does not define a measurement method but only an Appearance and a Behavior. It relies on other Chambre components, possibly running on remote machines, to perform the measurements needed to generate the data determining the puppet's posture. An MVI_VP appearance is given by a 3D digital puppet, produced as a tree of labeled Java 3D nodes (see Figure 5).

The Java 3D graph-based scene model [12] provides a simple and flexible mechanism for representing and rendering scenes. Chambre embedded graphical engine is based on Java 3D technology and is capable of rendering skinned bones. A scene graph contains a complete description of the entire scene. This includes the geometric data, the surface attributes and the visualization position information needed to render the scene from a particular point of view. Hence, the Appearance of an MVI_VP is a labeled extension of a Java 3D scene graph, with nodes of two kinds: `BranchGroup` and `TransformGroup`. A

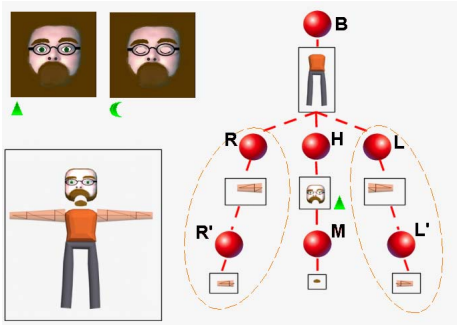

Fig. 5. The appearance of a Virtual Puppet

`BranchGroup` identifies body parts, and its elements have labels in the set $BG = \{B, H, M, R, R', L, L'\}$, for body, head, mouth, right and left arm and forearm, respectively. A `TransformGroup` specifies control elements and a Transformation Matrix for the different body parts. The labels in the set TG are obtained by concatenating a label from BG with one from the set $\{Tx, Ty, Tz, x, y, z, s\}$. Here, "$T$" labels indicate translation parameters along the three Cartesian axes, the next three indicate rotation angles around these axes, and s denotes the scale factor. As an example, the label "MTy" identifies the TransformGroup node used to translate the mouth BranchGroup along the y axis, producing a vertical shift of the puppet's mouth.

The *Behavior* of an MVI_VP is determined by a simple protocol: the object accepts messages of the form "*label value*", with $label \in TG$, and *value* a parameter used to define the new transformation matrix associated with the Appearance node identified by *label*. Values generated by Virtual Sensors are normalized to the $[0.0, 1.0]$ interval and mapped to labels in TG. A message can also modify attributes defining the material used to render the corresponding nodes of the puppet's 3D representation. For example the message "`MTy value`", besides defining the vertical position of the mouth, can also be used to change the texture of the puppet's head, thus modifying its facial expressions.

The definition of Virtual Puppets as CHAMBRE components favors the use (and reuse) of puppets in different contexts of execution. Actually, every software component (even non CHAMBRE ones) which respects the Behavior protocol can produce messages for the puppet controls. We focus now on two specific puppetry paradigms which can be realized in COPUPPET.

5.1 Single Puppet, Collaborative Controls

Single puppets can be operated on by one or more performers, through interactions in which separate channels control different aspects of the puppet, such as limb movement or facial expressions. As new controllable aspects are introduced, so can new multimodal input channels be added, to be managed by a

single performer, or distributed among many. In particular, we are interested in the evolution of patterns of real time collaboration among performers during free improvisation of the puppet performance.

In the proposed scenario, the puppet is set in a virtual environment of arbitrary complexity, and its movements, utterances and facial expressions are determined by the actions of independent users. The environment itself can present objects which can be animated by user interaction. Hence, users could log into a performance either as puppet body parts such as mouth, arms, lower body, or as objects in the scene, such as trees, umbrellas and hand mirrors.

In the current implementation, there are no constraints on relative motions of body parts - except that they be kept connected - so that physically impossible situations can arise. This is not in contrast with the aims of the project, and is possibly a desired effect of it, as it would be interesting to see how the performers draw themselves out of such situations, or engage in power struggles to make others conform to their choices. However, constraints could be added by exploiting the Java 3D tree structure of objects

5.2 Multiple Puppets, Collaborative Controls

A Multiple Puppet scenario can be produced by replicating a number of instances of the Single Scenario. Chambre capability of distribution helps the configurations of scalable stages, where teams of puppeteers can perform real-time shows and/or produce databases of recorded session for further post productions needs.

An asynchronous form of multiple puppets can be realized by recording actions of single puppets and merging them afterwards onto a new environment.

In a multiple puppet scenario based on replication of the single one, users enrol not only on body parts, but have to define which puppet they are managing. This may lead to the design of puppet chatting stages, in a way similar to how avatars may be used. Hence, users can engage in virtual dialogues, observing the reactions of the different participants through the modifications of their representative puppets. Sessions can also be recorded and replayed by the participants or by external observers.

6 Configuring CoPuppet

CoPuppet does not make particular demands on which software components can be used as a front-end, so that third party suites like Virtools [10] can be integrated into a CoPuppet application. The use of well-known tools can favor the artistic and technical development process of script creation and relieve the composer from advanced programming aspects such as the development of particular graphical routines.

Among the large array of possible configurations, we show how to develop both single and multiple scenarios. In CoPuppet a configuration usually consists of a network of computers running instances of the Chambre framework. Each Chambre instance contains a replication of all the Virtual Puppets used for the

performance; however different instances may differ in the graph used, which represents the connection scheme of the used CHAMBRE software components. Each puppeteer has a global view of the virtual puppets and of the current configuration of the software components needed to drive the body parts for which he/she is responsible, being thus focused on one of the CHAMBRE instances where the global behavior of the puppets is represented.

The behavioral aspects of a puppet, steered by its puppeteer, are transmitted to the other CHAMBRE instances in the network. Computation is thus distributed over the network and synchronization is achieved by message passing.

As mentioned before, external tools can be used for graphical and audio improvements. For example in Infinite Cemetery [13], spatial information about Virtools entities is transmitted via network to a program implementing audio spatialization and synthesis algorithm running on a separate machine.

7 CoPuppet Performance

In a CoPuppet collaborative performance, the puppeteers are placed at different locations in front of a screen on which the appearance of the virtual puppet is projected, as shown in Figure 6. Each user logs in as a puppet body part, or an object, for which controls have been defined, and produces collaborative movements of the virtual puppet by exploiting previously defined mappings between the available input channels and the puppet's behavior, as described in Section 7.2. A narrator, in charge of the puppet's face movements, tells a story into a microphone, to which the system responds in real time by producing mouth movements and facial expression of the virtual puppet on screen, according to some interpretation of the audio stimulus (see Section 7.2). Puppeteers in charge of body parts use their fingers in front of a webcam, thus activating virtual sensors connected with the corresponding body part on screen. The movements are reminiscent of those performed by real puppeteers steering movements through threads. While each puppeteer can create only simple movements, the whole gesture of the puppet results from their combination, producing powerful and interesting effects.

The performance can also allow for audience intervention. In particular, while retaining a single storyteller, CoPuppet can partially open several body controls to the public, to foster forms of collaboration between the performers and the audience. A completely open instantiation of the framework can also be envisaged, in which computers, microphone and webcams would be set up for users to participate in the performance space.

7.1 Collaboration Aspects

The main intended use of CoPuppet involves a storyteller, a crew of body parts performers, and possible intervention from the audience.

As seen in Section 2, the Bunraku theatre presents aspects of collaboration between body puppeteers and a storyteller. However, they rely on precise interpretations of the scripts, and their highly trained skills and practices. On

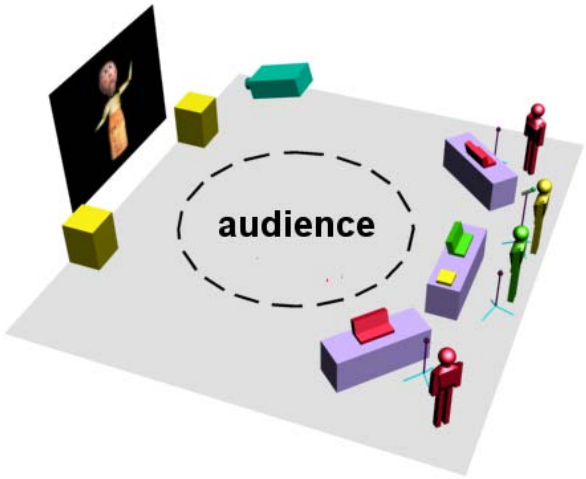

Fig. 6. Performance system layout

the contrary, CoPuppet supports forms of collaboration between users who do not have to follow a precise script, or event to have agreed in advance on some behavior.

Indeed, performers do not have to define in advance the kind of story they want to tell, but can come up at any time with ideas about the situation and the story, based on the materials in the scene, which can all become "performing objects". In a sense, performers are designing their own scene, depending on which objects are considered as active and which are the ones for which some user is registered. It is expected that the real time observation of the consequences of one's own actions, as well as those of the other puppeteers, will foster interaction between puppeteers, who will engage into collaborative or competitive behavior patterns. Typical patterns might be mimicking or counteracting the actions of one another, introducing delays in replicating some movement, achieving unison through the iteration of the same movements. Interestingly, these patterns may occur directly among performers as well. In this context, CoPuppet functions as an intangible social interface, creating new relationships and interaction between people, which could be resonated into the real community. We envisage that CoPuppet, through its simple and intuitive controls, can become a playground for children, while adults might find it interesting as a form of stage for free speech, as well as imaginative storytelling.

7.2 Technical Description

In this Section we describe one of the possible deployments of the CoPuppet framework, namely the one presented at the Digital Art Week in Zurich, 2006.

In this setting, each puppeteer uses one or more *VirtualSlider* to translate his/her hand movements into puppet actions. The slider position is determined

from the hand position, acquired via webcam. Puppeteers can see their hands projected into the virtual space, and interact with the virtual controllers (via their laptop computer screen) and the virtual puppet (via the projection screen). Figure 7 shows how the different transform nodes can be operated on through different multimodal channels. Labels from the set TG indicate both a puppet body part and the corresponding *VirtualSlider* setting the value of the control parameter. In particular, RX, Ry, $R'y$, LX, Ly, and $L'y$ are the labels for angle rotation around the x and y axes for the articulations of the puppet's arms, while Hx, Hy are labels for angle rotation around the x and y axis for the puppet's head. Finally, MTy is the parameter that determines the puppet's mouth opening and facial expression. MTy is determined by calculating the average amplitude value m of the signal coming from a microphone connected to the puppeteer's laptop. The value of m is calculated on a buffer of 1024 audio samples acquired every 0,023 seconds, so that the system response to audio storytelling is fast and accurate. The values of MTy are directly mapped to the puppet's "jaw" y-position, thereby making the puppet "talk" in response to incoming audio transients. An empirically determined constant k is used for the mapping, so that $MouthPosition = k * MTy$. Facial expression behavior is simply obtained by comparing the incoming MTy values against a threshold value: if the threshold is crossed, the puppet's eyes texture is switched, so that it looks like the puppet is shouting. Even if the mapping strategy is really simple, we have found it to be expressive and amusing to the audience.

In the Zurich exhibition, the global setup (Figure 7) consists of three laptops connected via a Local Area Network (LAN). Each laptop is connected with a USB webcam: the first laptop is used for Rx, Ry, $R'y$, controls, the second for the Lx, Ly, $L'y$ and the third for the Hx and Hy controls. For MTy, a microphone is connected to the *audio in* port of the second laptop. Finally, the second

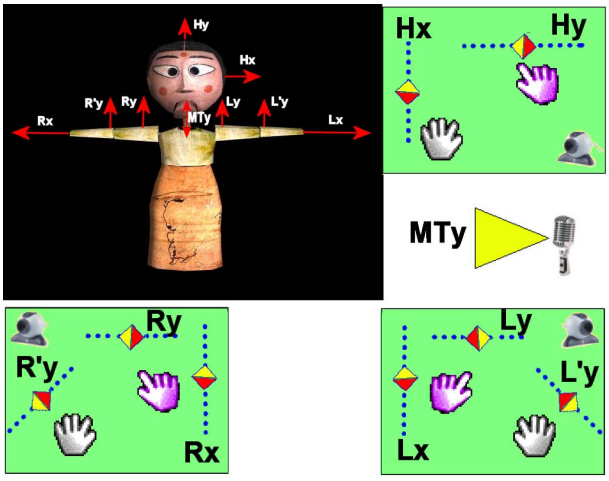

Fig. 7. Puppet controllers

laptop is connected to the video projector through a VGA cable, and, using an audio cable, is also connected to the mixer which drives the audio speakers for the audience. Effective interaction among puppeteers relies on the high speed of the 100mps LAN connection between the laptops, which transfers raw data generated by the virtual sensors. Collaborative performance by puppeteers in remote locations would require more sophisticated handling of messages exchanged between system components, including time-stamping of messages and reconstruction of missing data in case of network delays and/or failures.

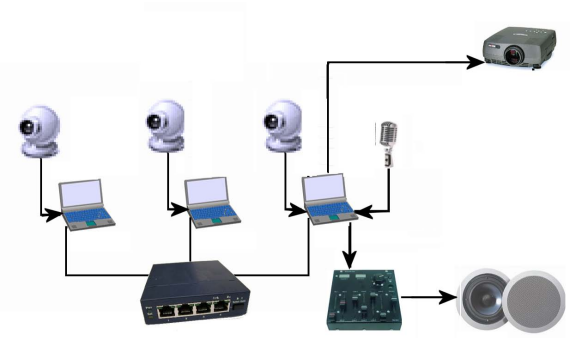

Fig. 8. Setup of the peripherals for CoPuppet

8 Combining Virtual and Real Puppets: A Case Study

The CoPuppet framework can also accommodate different types of virtual puppetry. As an example, "Experimental virtual wayang", by two of the authors, adapts the traditional Balinese shadow puppet performance ("wayang kulit"), injecting live improvisation into storytelling. This project also includes the collaboration of I Gusti Putu Sudarta, Andrew McGraw and the *eighth blackbird* sextet. A live performance was staged from October to November 2007.

Figure 9 shows some pictures taken during the whole process (from the first experiments to the public display) of the Experimental Virtual Wayang performance. Here the shadow puppet master changes the virtual puppet's look and control its movements with his voice. In order to maintain some harmony between the traditional and digital worlds, virtual puppets have been designed as 2D shapes, adding features such as breathing fire, colorful shadows, global animation and audio interaction. Experimental Virtual Wayang achieves even more dynamic improvisation when incorporated into CoPuppet, by distributing puppeteer's interactive controls to multiple users. Instead of using predefined, static animation data, articulated body parts of the used puppets are steered by virtual sensors, as is shown for the Virtual Dragon in Figure 10.

Figure 11 illustrates the collaborative performance setting, where one puppeteer works with traditional shadow puppets, and virtual puppets can be steered by multiple users.

Fig. 9. Pictures of Experimental Virtual Wayang performance, taken during Usability Test (Top left), Rehearsal (Top right), and Performance (Bottom left and right)

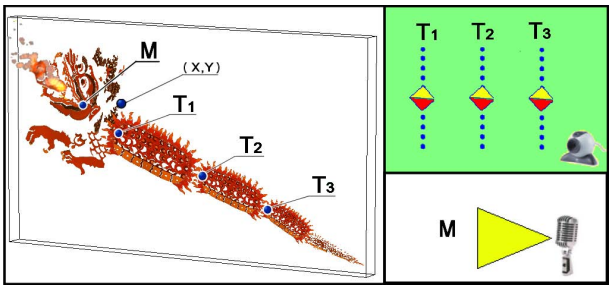

Fig. 10. A digital dragon for virtual Wayang

In the collaborative setting, the Nintendo Wii Remote controller (popularly called Wiimote) [14] is used instead of sticks, controlling global position, orientation and event generation (breathing fire, etc) on virtual puppets (1,2). The Wiimote is a commercially-available remote control, which is able to determine its position and orientation in space, sending the generated data via Bluetooth to the controlled device.

The Wiimote offers a wider range of interaction capabilities than traditional sticks. A main feature of the Wiimote is its motion sensing capability, which

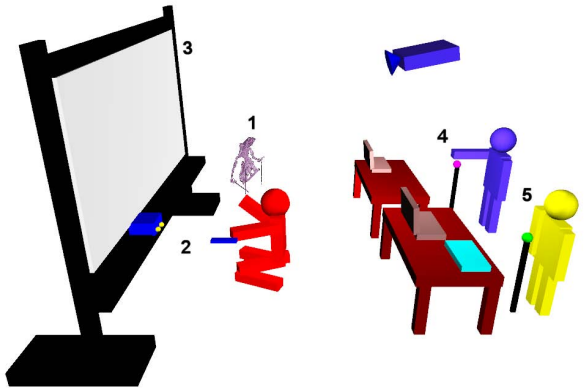

Fig. 11. Layout for collaboration between virtual and real Wayang

Fig. 12. The Wii Remote controller

allows the user to interact with and manipulate items on screen via movement and pointing through the use of accelerometer and optical sensor technology. Another interesting feature of Wiimote (see Figure 12) is the capability to track infrared source lights by a built-in infrared camera. Moreover, we have found that the Wiimote solution is easier to use than a prototype input device based on tracking of sticks movements we developed before. Microphones for sound input such as storytelling (5) and virtual sensors for articulated body movements (4) can also be distributed to multiple users for collaboration. Hence, traditional puppet manipulated by one puppet master, and virtual puppet collaboratively operated on by multiple users can interact with each other on the same screen.

Figure 13 illustrates the use of Wiimote to control the Virtual Dragon puppet orientation, puppet position (x, y, z), puppet action (breath fire) and character

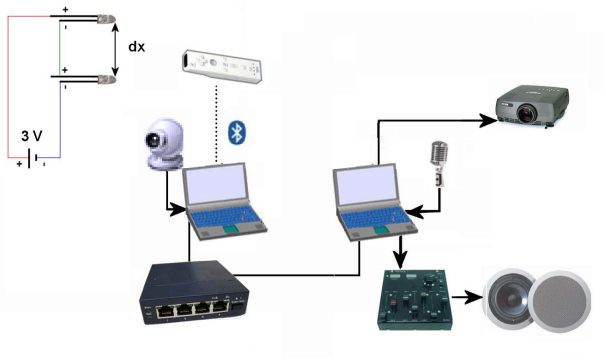

Fig. 13. Setup of peripherals for collaboration between virtual and real Wayang

selection. The coordinates of a puppet's position are estimated by tracking two infrared light sources using the Wiimote infrared camera feature. In particular the z component is obtained by measuring the distance between the two tracked infrared light sources.

For "Experimental virtual wayang", a CHAMBRE instance is used to manage the Wiimote and the virtual sensors controllers. From the CHAMBRE network, messages are delivered to Virtools which handles also audio interaction and graphical rendering. The distribution of controllers, as well as the number or roles of puppeteers, can vary depending on users' abilities and performance needs.

9 Discussion and Conclusions

We have presented COPUPPET, a distributed system for digital puppetry which exploits the collaborative performance of multiple puppeteers. Currently, COPUPPET allows interactions between puppeteers within a limited distance, but we envision remote collaborative puppet control over networks. We are also investigating the possibility of employing software agents in puppet controls, to enable more sophisticated mappings between data incoming from the virtual sensors and puppet movements. Future work on the multiple puppet scenario will aim at setting both puppets and humans in a mixed reality environment [15]. This last scenario requires the introduction of new software components able to solve problems concerning occlusion, contact, avoidance between human and virtual actors [16]. The open nature of the COPUPPET framework allows the incorporation of different sensors and the definition of different articulations and forms of rendering built on the Java 3D tree. On the other hand, its incorporation within the CHAMBRE environment, with its component-based structure, makes it possible to reuse definitions of behavior, appearance and measures in different contexts, such as collaboration scenarios, remote conferencing, or to associate puppets' controls to different sources of measures, whether physical or virtual.

From the artistic point of view, we envisage that the collaborative features offered by the CoPuppet framework will provide unique opportunities for virtual and real puppetry to explore critical issues related to improvisation, collaboration and interaction.

References

1. Bottoni, P., Faralli, S., Labella, A., Scozzafava, C.: CHAMBRE: A distributed environment for the production of multimedia events. In: Proc. DMS 2004, KSI, pp. 51–56 (2004)
2. Bottoni, P., Faralli, S., Labella, A., Malizia, A., Scozzafava, C.: CHAMBRE: integrating multimedia and virtual tools. In: Proc. AVI 2006, pp. 285–292. ACM Press, New York (2006)
3. Baird, B.: The Art of the Puppet. Macmillan, Basingstoke (1965)
4. Cho, O.K.: Korean puppet theatre: Kkoktu kaksi. Asian studies center East Asia series Michigan State University occasional paper 6(20) (1979)
5. Ryu, S.: Ritualizing interactive media: from motivation to activation. Technoetic Arts 3, 105–124 (2005)
6. Cavazza, M., Charles, F., Mead, S.J., Martin, O., Marichal, X., Nandi, A.: Multimodal acting in mixed reality interactive storytelling. IEEE MultiMedia 11, 30–39 (2004)
7. Myers, B., McDaniel, R., Miller, R., Ferrency, A., Faulring, A., Kyle, B., Mickish, A., Klimovitski, A., Doane, P.: The Amulet environment: New models for effective user interface software development. IEEE Transactions on Software Engineering 23, 347–365 (1997)
8. Bianchi-Berthouze, N., Bottoni, P.: Articulating actions in multimodal interaction. 3D Forum 16, 220–225 (2002)
9. Johnson, R., Foote, B.: Designing reusable classes. Journal of OO Programming 1, 22–35 (1988)
10. Virtools: Virtools (2004), `http://www.virtools.com`
11. Pure Data: Pure data (2008), `http://puredata.info`
12. Sun: Java 3D API Specification (2004), `http://java.sun.com/products/java-media/3D/forDevelopers/j3dguide/Intro.doc.html`
13. Ryu, S., Scozzafava, C., Faralli, S.: Infinite cemetery. In: Generative Art, Execution on, December 16 (2005)
14. Nintendo: Wiimote controller (2007), `http://wii.nintendo.com/controller.jsp`
15. Thalmann, D., Boulic, R., Huang, Z., Noser, H.: Virtual and real humans interacting in the virtual world. In: Proc. International Conference on Virtual Systems and Multimedia 1995, pp. 48–57 (1995)
16. Schuemie, M.J., van der Straaten, P., Krijn, M., van der Mast, C.: Research on presence in VR: a survey. Cyberpsychology and Behavior 4, 183–202 (2001)