Multi-modal System Architecture for Serious Gaming

Otilia Kocsis, Todor Ganchev, Iosif Mporas, George Papadopoulos, Nikos Fakotakis

Artificial Intelligence Group, Wire Communications Laboratory,

Dept. of Electrical and Computer Engineering, University of Patras, Rion 26500, Greece

{okocsis, tganchev, imporas, gpap, fakotaki}@upatras.gr

Abstract Human-computer interaction (HCI), especially in the games domain, targets to mimic as much as possible the natural human-to-human interaction, which is multimodal, involving speech, vision, haptic, etc. Furthermore, the domain of serious games, aiming to value-added games, makes use of additional inputs, such as biosensors, motion tracking equipment, etc. In this context, game development has become complex, expensive and burdened with a long development cycle. This creates barriers to independent game developers and inhibits the introduction of innovative games, or new game genres. In this paper the PlayMancer platform is introduced, a work in progress aiming to overcome such barriers by augmenting existing 3D game engines with innovative modes of interaction. Playmancer integrates open source existing systems, such as a game engine and a spoken dialog management system, extended by newly implemented components, supporting innovative interaction modalities, such as emotion recognition from audio data, motion tracking, etc, and advanced configuration tools.

1 Introduction

HCI has a long history, during which various interfaces were developed and currently aiming to a more natural interaction, involving 3D gesture recognition and speech-based interfaces. Achievement of naturalness involves progress from command or menu-based (system driven) to user-driven dialog management. System intelligence to allow adaptation to environment/context changes and user preferences is considered a must. The games domain has a special position in the area of HCI, holding a leading position in the research of attractive interfaces and interaction modes. Game development has become complex, expensive and burdened with a long development cycle, this creating barriers to independent games

developer and inhibiting the introduction of innovative games or new game genres, i.e. serious games or games accessible to communities with special needs.

Serious games (SGs) or persuasive games are computer and video games used as educational technology or as a vehicle for presenting or promoting a point of view. They can be similar to education games, but are often intended for an audience outside of primary or secondary education. SGs can be of any genre and many of them can be considered a kind of edutainment, intended to provide and engage self-reinforcing context in which to motivate and educate players towards non-game events or processes.

PlayMancer, a European Commission co-funded project, aims to implement a platform for serious games, which allows: (i) augmenting the gaming experience with innovative modes of interaction between the player and the game world, (ii) shorter and most cost-effective game production chain, (iii) evolvement of Universally Accessible Games principles for application into action based 3D games.

In this in-progress work, the PlayMancer concept and the architectural model of the multimodal platform are presented. The proposed platform architecture integrates a series of existing open source systems, such as a game engine, a spoken dialog management system, spoken interface components (speech recognition, understanding and synthesis). The existing components are augmented to support multimodality, to be adaptable to context changes, to user preferences/needs, and to game tasks. New interaction modes are provided by newly developed components, such as emotion recognition from speech audio data or motion tracking. One of the most important features of the proposed architecture is mixed-initiative dialogue strategy, enabled through dynamic generation of task-related interaction data, by coupling dialog and interactive 3D graphics objects at the design phase. Fast development of new games and adaptation to specific game scenario or user needs is facilitated by a configuration toolbox.

2 The PlayMancer Platform

The general architecture of the PlayMancer platform, illustrated in Fig. 1, has been designed taking into account: (i) functional and technical specifications derived from generic and specific domain user requirements, and (ii) the main technological challenge of the project – the rendering of an open source game engine multimodal. In particular, multi-modality is achieved by developing an enhanced multimodal dialogue interaction platform, based on the RavenClaw/Olympus architecture [1], which is further extended to render other modalities than speech.

The platform relies on a modular architecture, where components processing the data streams from the individual modalities and input sources interact through the central hub of Olympus, which allows synchronous or alternative use of input/output modalities. This way, in addition to traditional inputs/outputs used in games (joystick, keyboard, mouse, display), the PlayMancer architecture integrates also speech, touch, biosensors, and motion-tracking. These additional mo-

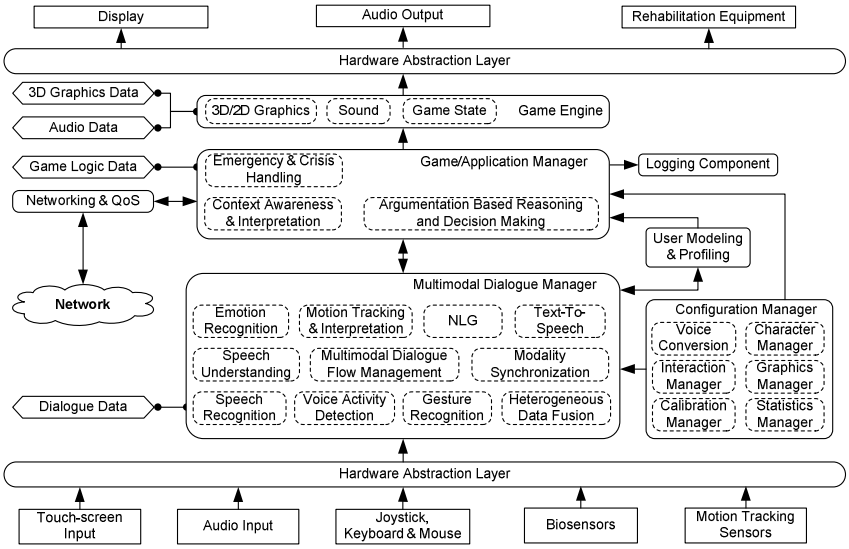

Fig.1 Architectural model of the PlayMancer platform

dalities allow the introduction of new game paradigms for fighting various eating and addiction disorders or for physical rehabilitation purposes, and, at the same time, offering enhanced game experience.

As illustrated in Fig. 1, the PlayMancer platform is composed of three major parts: (i) the Game Engine, (ii) the Game/Application Manager, and (iii) the Multimodal Dialogue Manager, as well as the Hardware Abstraction Layer, providing the interface to the input/output devices, and some auxiliary components. Of the auxiliary components, the Logging Component handles the long-term logging of data, which are processed statistically, and stored in a format convenient for interpretation by the healthcare supervisors (i.e. medical doctors, psychologists, rehabilitation experts, etc). The networking and Quality of Service (QoS) monitoring component provides the means for multiplayer experience over existing global or local area networks. The User Modeling and Profiling component is responsible for personalization and adaptation of interaction to user preferences [2].

The *Configuration Manger* (CM) is an off-line toolbox of utilities and resources (such as Voice Conversion tools and resources, game Character Manager, Graphics Manager, etc), which are not part of the run-time application. Instead, they provide to the developers, the supervisor or the player of the game the tool to create new games, to customize and fine-tune the entire game or specific game level, and to contribute own content. The CM also provides the means for configuration setup of the interaction modes, calibration of the sensors, introducing user-specific game settings, management of the overall game resources, developing new game characters/actors, game levels, etc. For the newly generated playing characters, a Voice Conversion component allows the user to create new voices that will fit to the profile of the new characters. The Statistics Manager allows the healthcare supervisor to configure the necessary health indicators to be monitored,

the settings and the output format for the statistical summarization of data.

The *Game Engine* provides the necessary components to display the game's graphics, play the sound and music, and manage the game state. The graphics component is a fully featured 3D graphics rendering engine, with additional support for 2D graphics, used for the graphical user interface. Evaluation of various open source game engines, with respect to the PlayMancer requirements, led to the selection of the Object-Oriented Graphics Rendering Engine (OGRE) [3] as the most suitable for integration in PlayMancer environment. The sound component creates realistic 3D sound from static and moving sound sources in the 3D environment. Its capabilities include playback of prerecorded sounds and music, as well as from in-memory buffers, thus facilitating the real-time procedural creation of sounds. The game state component manages the runtime state of all game entities, ensuring correlation between state changes and game logic.

The *Game/Application Manager* (GAM) hosts the principal management unit of the application, the PlayMancer-derived game. GAM, managing both game and real-world aspects, is responsible for the synchronization and smooth information exchange among the components and for the overall operation of the application.

Through the Context Awareness and Interpretation component, the GAM administers the smooth interaction between the physical and the synthetic world of the game, offering a high-level supervision of the user interaction, experiences and reactions. Emotion arousal, bio-indicators, and game-specific information from the synthetic world, provide context clues that can variously be exploited depending on the particular game. GAM also hosts the Emergency and Crisis Handling component, which enables alteration of the interaction style and the game flow, to prevent danger and harm to the player in case of atypical behaviors or development of emergency situation. Monitoring of context parameters and detection of atypical behaviors is supported by personalized game-specific user profile data, set-up by the supervisor for each player. The Argumentation-Based Reasoning and Decision Making component implements the top-level decision making logic of the system that is responsible for achieving the goals of the application.

The major task of the *Multimodal Dialogue Manager* is to handle user-machine interaction: handle events from different input modalities, interpret inputs, fuse and disambiguate these inputs when necessary, personalize and adapt interaction, etc. The PlayMancer-based interaction has the following features:

- The description of the 3D game interactive objects has been enhanced with information regarding interaction modalities allowed and dialogue data (interaction resources, such as understanding grammars for speech interface) to be used for each modality. This data will be used by the Multimodal Dialogue Manager to select the most suitable modality for interaction and the resources to be used by each input/output modality when interaction is requested.
- The use of modality independent interaction task plan, allowing to synchronize and fuse data from different modalities.
- Interaction ambiguities solving is employed by the Multimodal Dialogue Manager when commands, which apply to more than one interactive object, are issued by the player.

Integration of spoken dialogue interfaces in the game domain is a challenging research issue, especially when the gaming environment is multimodal, where the domain boundary is not always clear. In PlayMancer, we consider a speech interface which incorporates the following components: Voice Activity Detection (VAD), Speech Recognition, Speech Understanding, Text-to-Speech Synthesis (TTS) and Natural Language Generation (NLG) [4-6]. Data stream of the infrared Motion Tracking Sensor array is processed by the OpenTracker framework [7, 8], which in Fig. 1 is designated as Motion Tracking and Interpretation component.

The biosensor input is used to monitor various bio-indicators of the player. The biomedical data are recorded for diagnostic purposes. In addition, the speech and biosensor data streams are fed to the Emotion Recognition (ER) component [9]. The ER component detects the emotional state of the player among a group of emotional categories, such as neutral, angry, happy, panic, fear, boredom, etc. The information of the emotional condition of the player is used as a supplementary parameter for estimating the cognitive load of the player. Depending on the purpose of the game, the emotional state and the cognitive load of the player, the GAM and Game Engine can implement different strategies; change the interaction style; or the level of difficulty at the current scenario.

The data streams from all modalities are time-aligned in the Modality Synchronization component, and then disambiguated and fussed in the Heterogeneous Data Fusion component. The success of multimodal data integration depends on the abstraction level to which data are fused and the method applied to carry out the multi-sensory data fusion. In PlayMancer, various abstraction levels will be considered, depending on the sensor type and task goal.

The complexity of the PlayMancer platform implies modeling of knowledge in relation to the components involved and tasks to be completed. Data that are already handled by the existing components integrated into the platform, such as the 3D Graphics Data (3DGD) and Audio Data handled by the game engine, are related to higher abstraction level data, needed for the multimodal interaction. The 3DGD represents all graphical objects needed to build game environment in the virtual world, including shape, color and textures descriptions. Some of these objects are interactive, allowing different states. Thus a description of their alternative states and behavior when passing from one state to another, game logic data, is also attached to the 3DGD. In PlayMancer, the 3D Graphics Data is coupled to Dialog Data. Dialog Data consists of description of interaction modes that can be used to interact with the virtual object, and modality specific data used for interpretation of input row data to higher abstraction level concepts. This coupling allows dynamic generation of task-related interaction resources and high level of reusability of dialog data for different games or scenes design.

3 Conclusion

Playmancer is built on top of existing open-source software, such as OGRE 3D rendering engine [3] and RavenClaw/Olympus dialog management platform [1], aiming to provide a novel development framework for serious game development. The existing components are augmented to support multimodality, to be adaptable to context changes, to user preferences, and to game task. Game developers are provided with a series of configuration and management components, to shorten the development cycle of games and to enable high level of code re-usability. Architecture design of multimodal systems, especially for virtual reality environments in the games domain, requires addressing several issues related to redundancy of input/output modalities, complementarity, disambiguation, etc [10]. Common approaches for modality integration are frame-, grammar- or agent-based, while the concept of Interactive Cooperative Objects (ICO) is emerging [11]. PlayMancer considers a mixed approach, integrating features of the grammar-based method and ICO formalism for the dynamic modeling of multimodal interaction. Dynamic generation of task-related interaction data is enabled by coupling dialog and interactive 3D graphics objects at the design phase.

Acknowledgment This work was supported by the PlayMancer project (FP7 215839), which is partially funded by the European Commission.

References

1. Bohus, D., Rudnicky, A. (2003). RavenClaw: Dialog management using hierarchical task decomposition and expectation agenda. In: Proceedings Eurospeech 2003, pp.597-600, Geneva.
2. Vildjiounaite, E., Kocsis, O., Kyllonen, V., Kladis, B. (2007) Context-dependent user modeling for smart homes. In: Proceedings 11th International Conference on User Modeling, LNCS, vol. 4511/2007, pp. 345-349, Springer, Heidelberg.
3. Object-Oriented Graphics Rendering Engine, http://ogre3d.org
4. Nuance Recognizer, http://www.nuance.com.
5. The CMU Sphinx Group Open Source Speech Recognition Engines, http://cmusphinx.sourceforge.net/html/cmusphinx.php.
6. FestVox speech synthesizer, http://festvox.org.
7. Reitmayr, G., Schmalstieg, D. (2001). OpenTracker – an open software architecture for reconfigurable tracking based on XML. In: Proceedings IEEE Virtual Reality 2001, pp. 285-286.
8. OpenTracker – An open architecture for reconfigurable tracking based on XML, http://studierstube.icg.tu-graz.ac.at:80/opentracker/

9. Kostoulas, T., Ganchev, T., Mporas, I., Fakotakis, N. (2008). A real world emotional speech corpus for modern Greek. In: Proceedings LREC'2008, Morocco, May 28-30.
10. Bourguet, M.L. (2004). Software design and development of multimodal interaction. In: Proceedings IFIP 2004, pp. 409-414.
11. Navarre, D., Palanque, P., Bastide, R., Schyn, A., Winckler, M., Nedel, L.P., Freitas, C. (2005) A formal description of multimodal interaction techniques for immersive virtual reality applications. In: Proceedings INTERACT'05, LNCS, vol.3585, Springer, pp.170-183.