Building a Sensory Infrastructure to Support Interaction and Monitoring in Ambient Intelligence Environments

Emmanouil Zidianakis[1], Nikolaos Partarakis[1],
Margherita Antona[1], and Constantine Stephanidis[1,2]

[1] Foundation for Research and Technology – Hellas (FORTH)
Institute of Computer Science Heraklion, GR-70013, Greece
{zidian,partarak,antona,cs}@ics.forth.gr
[2] University of Crete, Department of Computer Science

Abstract. In the context of Ambient Intelligence (AmI), the elaboration of new interaction techniques is becoming the most prominent key to a more natural and intuitive interaction with everyday things [2]. Natural interaction between people and technology can be defined in terms of experience: people naturally communicate through gestures, expressions, movements. To this end, people should be able to interact with technology as they are used to interact with the real world in everyday life [19]. Additionally, AmI systems must be sensitive, responsive, and adaptive to the presence of people [16]. This paper presents the design and implementation of an interaction framework for ambient intelligence targeting to the provision of novel interaction metaphors and techniques in the context of AmI scenarios. The aforementioned infrastructure has been deployed in vitro within the AmI classroom simulation space of the FORTH-ICS AmI research facility and used to extend existing applications offered by an augmented interactive table for young children (Beantable) to support also games that facilitate biometric information, rich interaction metaphors and speech input [20].

1 Introduction

Ambient Intelligence (AmI) presents a vision of a technological environment capable of reacting in an attentive, adaptive and active (sometimes proactive) way to the presence and activities of humans and objects in order to provide appropriate services to its inhabitants [17]. According to the Institute for the Future, “Emerging technologies are transforming everything that constitutes our notion of “reality” - our ability to sense our surroundings, our capacity to reason, and our perception of the world” [1]. In the context of Ambient Intelligence, several challenges emerge in the contributing domains of ubiquitous computing, mixed reality and HCI. This work presents the design and implementation of a technological framework to support the interaction requirements of AmI at large by offering a number of alternative natural interaction techniques such as gestures, face and skeleton tracking, head position estimation and speech recognition.

[1] Blended Reality report: `http://www.iftf.org/uploads/media/SR-122~2.PDF`, page 1.

N. Streitz and P. Markopoulos (Eds.): DAPI 2014, LNCS 8530, pp. 519–529, 2014.

2 Background and Related Work

2.1 Ambient Intelligence

The term “Ambient Intelligence” was coined from Philips Research’ vision of “people living easily in digital environments in which the electronics are sensitive to people's needs, personalized to their requirements, anticipatory of their behavior and responsive to their presence” [15]. This concept was adopted by the Information Society Technologies Advisory Group (ISTAG) as one of their research focus. In their report [5], ISTAG show the concrete vision that humans will, in an Ambient Intelligent Environment, be surrounded by intelligent interfaces embedded in everyday objects (as furniture, clothes and the environment). Aarts and Marzano review the five key technology features that portray an AmI system [1]: (a) Embedded, (b) Context aware (c) Personalized, (d) Adaptive and (e) Anticipatory.

2.2 Interaction Techniques for Ambient Intelligence

Gestures are employed in the context of AmI for providing alternative ways of user input in a human like fashion. Gestures when used in the context of human to human communication are a quick and intuitive way of communication. On the contrary, identifying human gestures in a computerized environment is not an easy task. Research conducted in this field involves the usage of gestures for providing input to augmented desk interface systems using multiple fingertips recognition (identify fingertips and their trajectories and infer gestures based on these trajectories) [14], [9]. Computer vision is used for identifying hand gestures, facial expressions and body postures [12]. Furthermore, the usage of thimble-shaped fingertip markers made of white printing paper with a ’black light’ source has been proposed for providing gesture recognition in the context of back projection walls [8].

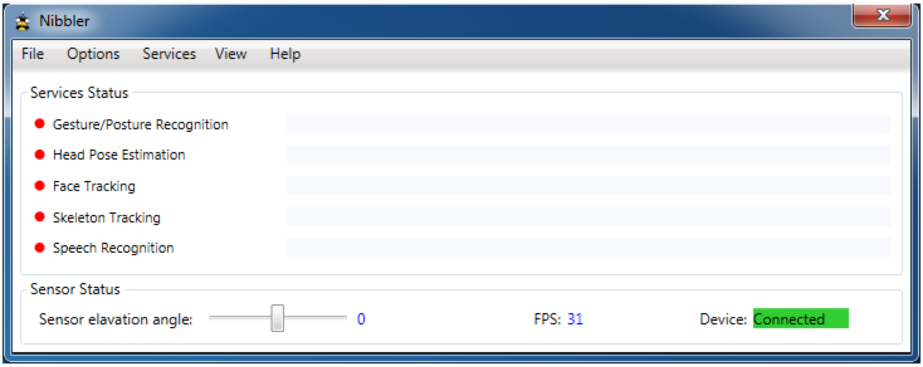
Gaze recognition has recently got the attention of scientific community and is employed for facilitating alternative gesture based input [9, 10, 11]. Although this form of interaction is not generically applicable, it can be used in conjunction with face tracking and head position estimation to support various forms of natural implicit interaction with the environment.

Using speech as an input channel in ICT is not new. Research has been conducted in this field for years, and today thousands of commercial and research product have been developed.

This paper presents the design and implementation of a unified framework which supports a number of alternative natural interaction techniques through the integration of facilities to support: (a) gestures recording, parameterisation and recognition, (b) face tracking, (c) head position estimation, (d) skeleton tracking and (e) speech recognition. This novel interaction framework has been deployed in vitro in an augmented interactive table for young children (Beantable) [21]. Beantable, although supporting native means of interaction through the provision of a multi-touch surface (main features: blob tracking and objects recognition), also integrates the framework presented here in order to provide supplementary forms of interaction with children, thua extending the educational and learning experience into the surrounding environment.

3 Implementation

The sensory modules created in the context of this research work are embedded in a software platform called ‘Nibbler’. This platform builds on the Microsoft Kinect sensor and was developed using the C# programming language, Microsoft Kinect software development kit (SDK) v1.8 and the .NET Framework. It is organized in various modules each of which is responsible for specific sensory requirements. The GUI of the proposed software platform is presented in Fig. 1. Each module is presented thoroughly in the following sections.



● Gesture/Posture Recognition
● Head Pose Estimation
● Face Tracking
● Skeleton Tracking
● Speech Recognition

Sensor elavation angle: 0 FPS: 31 Device: Connected

Fig. 1. Nibbler is the program implements the most of sensory services

3.1 Skeleton Tracking Module

The **skeleton tracking** module is responsible for reporting position information of each skeleton joint. This module performs geometric transformations on each skeleton joint position constituting every real time skeleton frame. This happens in order to get the same valid results regardless of the position of the user who may be located everywhere inside the sensor’s field of view.

The skeleton tracking module transforms the user’s skeleton to a local scope, i.e., expressed relatively to the 3-axis coordination system centered in the middle of the user’s shoulders. The transformation is applied with respect to three distinct steps, translation, scaling and rotation as shown in Fig. 7②. Firstly, each joint’s x-axis position is subtracted with the position dynamically calculated as the center of both shoulders: this way, skeleton tracking is performed regardless of the user’s relative position to the sensor, as presented in Fig. 2. Secondly, the joints’ positions are normalized in order to be scale-independent. Finally, the module rotates the skeleton so as to align the user’s skeleton to the sensor. This is accomplished by multiplying each joint’s position with a matrix calculated from the angle θ, where θ is equal to –yaw (yaw is the angle of line between the right and the left shoulder).

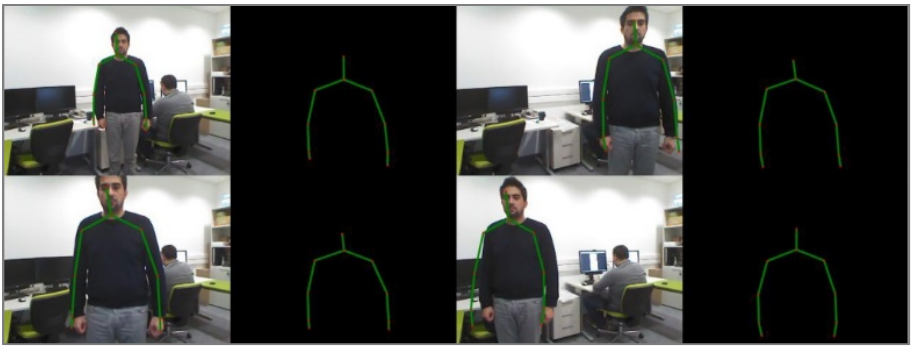


Fig. 2. An example of position independence between user and sensor

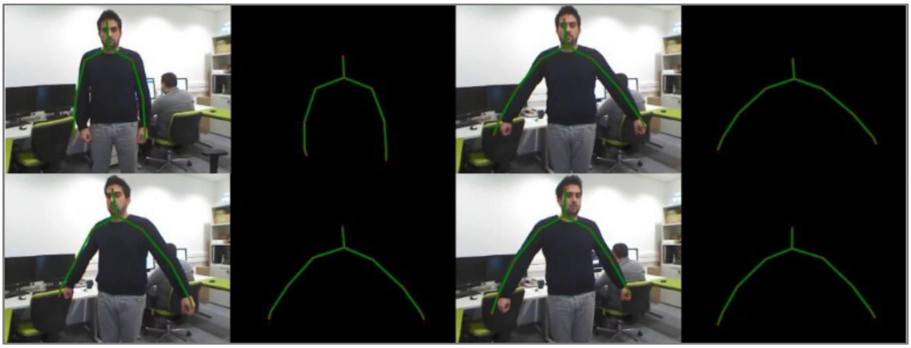


Fig. 3. An example of alignment independence between user and sensor

3.2 Gesture/Posture Recognition Module

Although much work has been done to date in the domain of Gesture/Posture recognition, no ready to use solutions are widely available to developers wishing to incorporate kinect based gesture recognition in their applications, with the exception of the Microsoft Kinect SDK samples (a sample is provided that implements two gestures: next, previous). The developed Gesture/Posture recognition module implements the dynamic time warping (DTW) algorithm [17] for measuring similarity between two skeleton sequences which may vary in time or speed. Additionally, it provides a training platform that allows developers to fine tune their gestures having access to a number of alternative biometric parameters. In general, the DTW algorithm can be applied to any data which can be turned into a linear sequence, i.e., a well-known application has been automatic speech recognition, to cope with different speaking speeds[2]. The **first skeleton sequence** is captured and fine-tuned only once during the training process while the **second** one is captured constantly in real time. During the training process, the author is able to record a skeleton sequence using his own body

[2] http://en.wikipedia.org/wiki/Dynamic_time_warping

as input data and store it in the database (Fig. 4). The number of the frames in a sequence may vary from 1 up to a maximum predefined variable, which is usually 30 considering that 30 is the maximum sensor's frame rate according to its specification details.

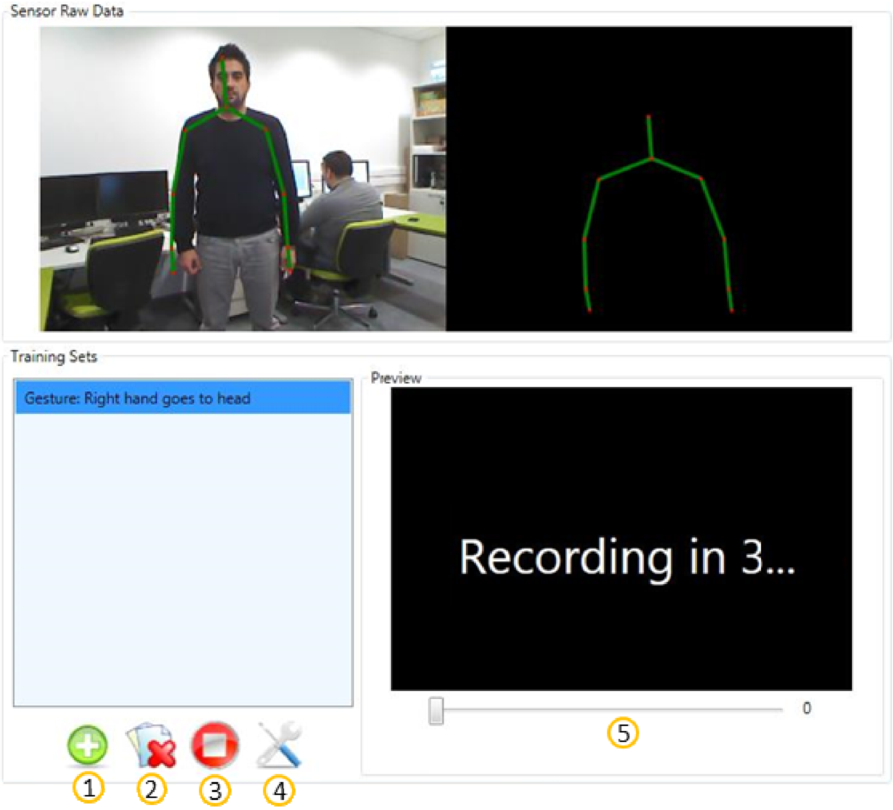


Fig. 4. The author starts recording of a skeleton sequence in seated mode

End users are provided with several functions to optimally adjust gesture/posture recognition. In particular, end users may (see Fig. 4): add a new gesture using button①, delete an existing one②, or start/stop skeleton sequence recording③. Once recording has been completed, the author is able to preview the recorded skeleton sequence, edit it and fine tune the captured skeleton sequence by pressing button ④ and using the pop up configuration window as shown in Fig. 5. This window offers functionality for: (a) renaming①, (b) adjusting the maximum distance with the real time skeleton sequence ② for successful recognition (see below for further details), (c) modifying the number of the minimum frames③ that have to be captured in real time before the recognition process is triggered, (d) trimming the corresponding skeleton sequence using⑤,⑥ and (e) adjusting some of the basic parameters used in the

DTW algorithm such as the slop constraint which determines the maximum slope in the optimal path⑦. Additionally, the author can select only the joints which mainly characterize a gesture⑧, i.e., when the goal is to recognize a gesture in which the user uses his right hand to select the next photo by slightly moving it from right to left, the remaining joints of the body do not need to be taken into account. Furthermore, if some axis doesn't play an important role for a gesture, such as the Z axis (the axis of depth) in the aforementioned example, the author is can disable it by unselecting the corresponding checkbox in④. Lastly, a gesture playback panel⑨ is available to allow the author to preview the recorded skeleton sequence in front and side realization.

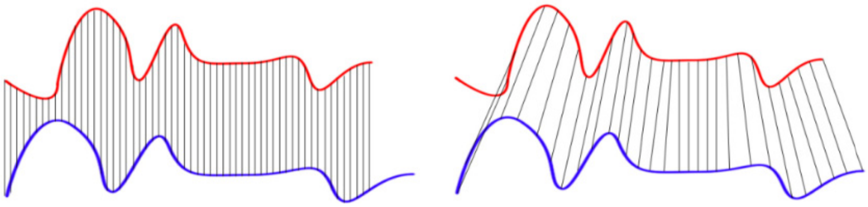


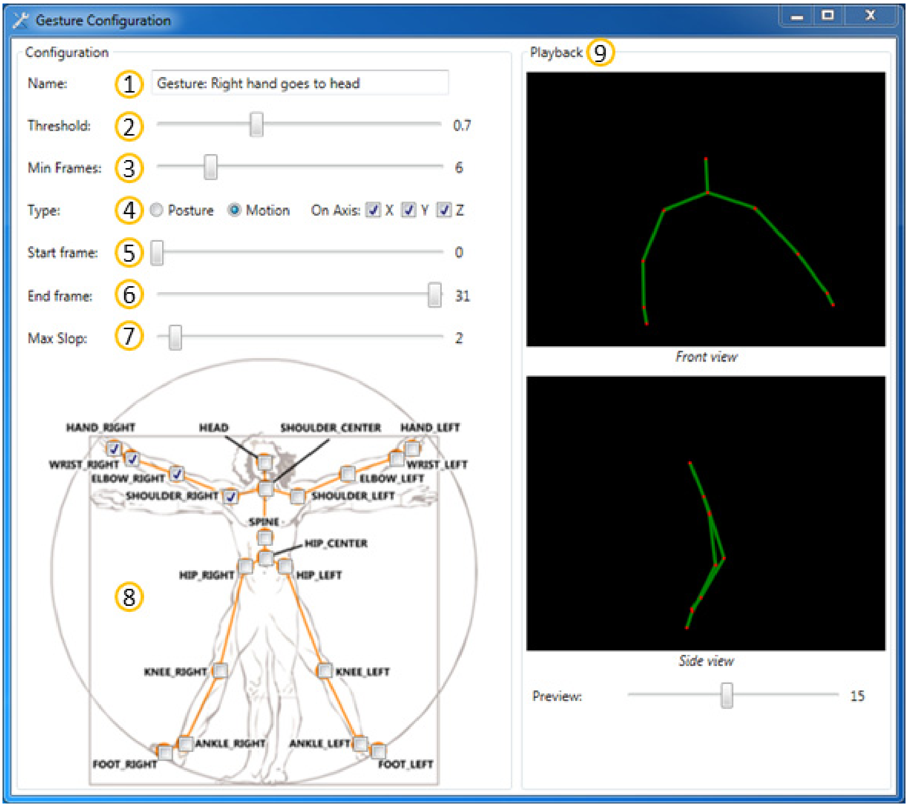
Fig. 5. Euclidean vs. Dynamic Time Warping Matching

When the recognition module is running, it captures constantly, in real time, skeleton frames, and when their total number reaches the number equivalent to one second then it starts the matching process. The latter calculates an optimal distance between the real time sequence and every sequence which is stored in the database. The sequences are "warped" non-linearly in the time dimension to determine a measure of their similarity independent of certain non-linear variations in the time dimension. It should be mentioned that Dynamic Time Warping (DTW) allows elastic shifting in the time domain and matches sequences that are similar but out of phase as shown in Fig. 6.

3.3 Head Pose Estimation/Face Tracking Module

The head pose estimation/face tracking module enables the creation of applications based on more natural and intuitive interaction styles. The implementation of this module builds on the Microsoft Kinect SDK and the provided facilities to track human faces in real time. The module reports measurements about the three axes of rotation of user's head pose (pitch, roll, and yaw) based on a right-handed coordinate system. Additionally, it reports the rectangle enclosing the head in the captured frame. For face tracking, six animation units (AUs) are tracked in real-time which are a subset of what is defined in the Candide3 model[3]. The results are expressed in terms of numeric weights which are deltas from the neutral shape varying between -1 and +1.

[3] http://www.icg.isy.liu.se/candide/


Gesture Configuration

Configuration

Name: 1 Gesture: Right hand goes to head

Threshold: 2 0.7

Min Frames: 3 6

Type: 4 ○ Posture ● Motion On Axis: ☑ X ☑ Y ☑ Z

Start frame: 5 0

End frame: 6 31

Max Slop: 7 2

Playback 9

Preview: 15

Fig. 6. Configure and fine tuning of a gesture

3.4 Speech Recognition Module

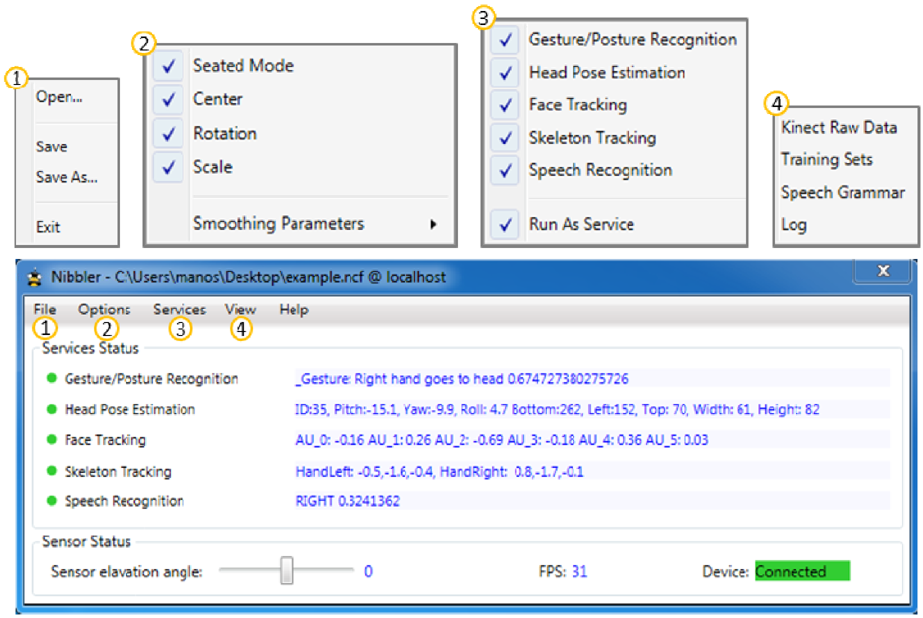
The speech recognition module is based on the Microsoft.Speech library found in the Microsoft Kinect SDK and the sensor's built-in microphone array as voice input device. This module takes a grammar as input and loads it to the recognition engine. At this time, speech recognition is supported only in a few languages. The module reports information about the spoken word or phrase which contains a unique tag, confidence as well as the speaking duration.

4 Using the Framework from a Developer's Perspective

The component modules of the Nibbler framework run simultaneously without any performance issues at almost 30fps on an ordinary pc (see Fig. 7). Fig. 7 illustrates the settings that are available for configuring Nibbler for the desired context of use. Nibbler communicates with clients and reports each module's measurements via a middleware network layer. The latter is designed to facilitate the communication of systems that are deployed on diverse platforms, as presented in [4]. Additionally, Nibbler can accept requests from clients in real time to change either the gesture training set or the grammar used for speech recognition. In this context, Nibbler's functionality is described in the interface definition language (IDL[4]) as follows:

Gesture recognition	ami::StringSeq GetGestureNames ();
	boolean LoadGestures(in ami::OctetSeq gesturesConfigStream);
	void Event_GestureRecognized (in string gesture, in double distance);
Head pose estimation	void Event_HeadPoseChanged (in double pitch, in double yaw, in double roll);
	void Event_HeadRectChanged (in long left, in long bottom, in long top, in long width, in long height);
Face tracking	void Event_FaceAnimationUnitsChanged (in ami::FloatSeq faceAnimationUnits);
Skeleton tracking	void Event_HandRightPositionChanged (in Point3D position);
	void Event_HandLeftPositionChanged (in Point3D position);
	void Event_SkeletonChanged (in JointSeq joints);
Speech Recognition	SemanticResultValueSeq GetLoadedGrammar (out string culture);
	boolean LoadGrammar(in SemanticResultValueSeq grammar, in string culture);
	void Event_SpeechRecognized (in string value, in float confidence, in long long duration);
	void Event_SpeechRejected ();

[4] http://en.wikipedia.org/wiki/Interface_description_language



Services Status

● Gesture/Posture Recognition	_Gesture: Right hand goes to head 0.674727380275726
● Head Pose Estimation	ID:35, Pitch:-15.1, Yaw:-9.9, Roll: 4.7 Bottom:262, Left:152, Top: 70, Width: 61, Heigh:: 82
● Face Tracking	AU_0: -0.16 AU_1: 0.26 AU_2: -0.69 AU_3: -0.18 AU_4: 0.36 AU_5: 0.03
● Skeleton Tracking	HandLeft: -0.5,-1.6,-0.4, HandRight: 0.8,-1.7,-0.1
● Speech Recognition	RIGHT 0.3241362

Sensor Status

Sensor elavation angle: 0 FPS: 31 Device: Connected

Fig. 7. Nibbler in a fully working mode

5 In Vitro Instantiation of the Framework

The interaction framework presented in this paper has been deployed and tested in vitro to extend the interaction capabilities of an augmented interactive table for young children (for ages 3 to 7) named Beantable [20]. Beantable has been developed in the context of the FORTH-ICS's AmI research program and is currently hosted within the AmI classroom simulation space of the FORTH-ICS AmI Research Facility. The purpose of Beantable is to support children's development through the monitored use of appropriate smart games in an unobtrusive manner. Beatable monitors the children's interactions and extracts indications of the achieved maturity level and skills by taking into account: (a) the way the child plays, (b) the selection of materials and game themes and (c) the way the child takes part into the activities. The addition of the interaction framework presented in this paper has expanded impressively the potential instantiations of Beantable by supporting also games facilitating biometric information and speech such the "Mimesis" game. The latter is an imitation game in which a virtual partner assumes various body postures and simultaneously invites the young child to imitate him as fast as he cans. The final setup of Beantable also includes a large wall mounted screen with an embedded Microsoft Kinect sensor collecting the data to be employed by the framework, as shown in Fig. 8.

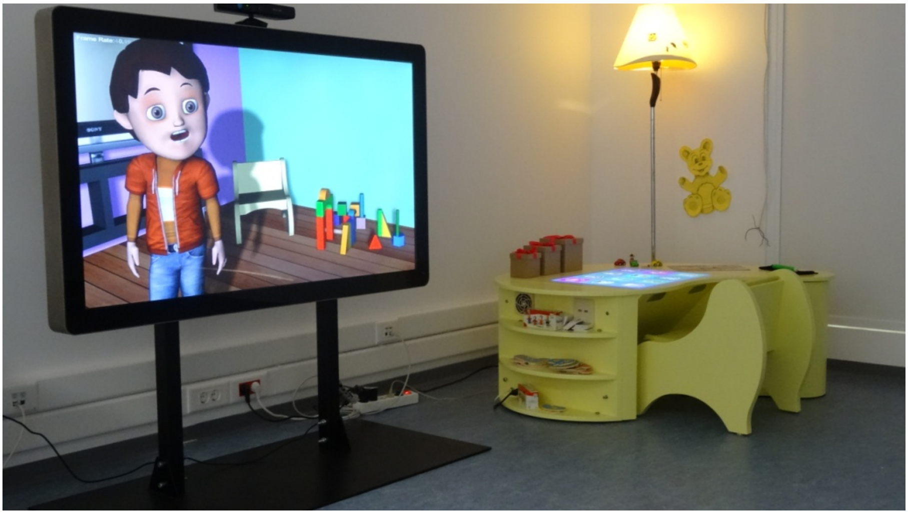


Fig. 8. The extended Beantable setup

6 Discussion and Future Work

This paper has presented the implementation of a unified framework to support interaction requirements in AmI environments at large, focusing on the architecture, driving technology and software infrastructure of the system. Regarding future improvements two are the main directions. First, a large scale user based evaluation is going to be conducted so as to evaluate the framework in different contexts and with alternative interaction scenarios, taking also into account biometric information of the target user population. Second, the framework itself is considered a living organism and is going to be enriched to take advantage of technology advancements both by creating new artefacts and enriching the supported interaction techniques. The first planned improvement is the integration of gaze tracking functionality so as to complete the suite of head tracking facilities.

Acknowledgments. This work is supported by the FORTH-ICS internal RTD Programme 'Ambient Intelligence and Smart Environments'.

References

1. Aarts, E.H., Marzano, S. (eds.): The new everyday: Views on ambient intelligence. 010 Publishers (2003)
2. Aarts, E., Encarnacao, J.L.: True Visions. The Emergence of Ambient Intelligence. Springer (2008) ISBN 978-3-540-28972-2
3. Alcañiz, M., Rey, B.: New technologies for ambient intelligence. The Evolution of Technology, Communication and Cognition Towards the Future of Human-Computer Interaction, 3–15 (2005)

4. Georgalis, Y., Grammenos, D., Stephanidis, C.: Middleware for Ambient Intelligence Environments: Reviewing Requirements and Communication Technologies. In: Stephanidis, C. (ed.) UAHCI 2009, Part II. LNCS, vol. 5615, pp. 168–177. Springer, Heidelberg (2009)
5. ISTAG, Ambient Intelligence: From Vision to Reality. In: Lakatta Riva, G., Vatalaro, F., Davide, F., Alcañiz, M. (eds.) Ambient Intelligence. IOS Press (2005)
6. Kaltenbrunner, M., Bencina, R.: reacTIVision: A computer-vision framework for table-based tangible interaction. In: Proceedings of the 1st International Conference on Tangible and Embedded Interaction. ACM (2007)
7. Kameas, A., Mavrommati, I., Markopoulos, P.: Computing in Tangible: Using Artifacts as Components of Ambient Intelligence Environments. IOS Press (2005), http://www.ambientintelligence.org
8. Kim, H., Fellner, D.W.: Interaction with hand gesture for a back-projection wall. In: Proceedings of the IEEE International Computer Graphics, pp. 395–402 (June 2004)
9. Nakanishi, Y., Oka, K., Kuramochi, M., Matsukawa, S., Sato, Y., Koike, H.: Narrative Hand: Applying a fast finger-tracking system for media art
10. Ohno, T., Mukawa, N., Kawato, S.: Just blink your eyes: A head-free gaze tracking system. In: CHI 2003 Extended Abstracts on Human Factors in Computing Systems, pp. 950–957. ACM (2003)
11. Ohno, T., Mukawa, N.: A Free-head, Simple Calibration, Gaze Tracking System That Enables Gaze-Based Interaction
12. Ohno, T., Mukawa, N., Yoshikawa, A.: FreeGaze: A gaze tracking system for everyday gaze interaction. In: Proceedings of the 2002 Symposium on Eye Tracking Research & Applications, pp. 125–132. ACM (March 2002)
13. Ohya, J.: Computer Vision Based Analysis of Non-verbal Information in HCI
14. Oka, K., Sato, Y., Koike, H.: Real-time tracking of multiple fingertips and gesture recognition for augmented desk interface systems. In: Proceedings of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition, pp. 429–434. IEEE (May 2002)
15. Philips Research (2005), http://www.research.philips.com/technologies/syst_softw/ami/index.html
16. Phillips Research, Ambient intelligence: Changing lives for the better (2007), http://www.research.phillips.com/
17. Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. IEEE Trans. on Acoust., Speech, and Signal Process., ASSP 26, 43–49 (1978)
18. Stephanidis, C.: Human Factors in Ambient Intelligence Environments. In: Salvendy, G. (ed.) Handbook of Human Factors and Ergonomics, 4th edn., ch. 49, pp. 1354–1373. John Wiley and Sons, USA (2012)
19. Szeliski, R.: Image alignment and stitching: A tutorial. Foundations and Trends in Computer Graphics and Vision. Now Publishers Inc. (2006)
20. Valli, A.: The design of natural interaction. Multimedia Tools and Applications 38(3), 295–305 (2008)
21. Zidianakis, E., Antona, M., Paparoulis, G., Stephanidis, C.: An augmented interactive table supporting preschool children development through playing