Explicit Task Representation based on Gesture Interaction

Christian Müller-Tomfelde and Cécile Paris
CSIRO - Information and Communication Technologies Centre
Locked Bag 17, North Ryde, NSW 1670, Australia
Christian.Mueller-Tomfelde@csiro.au and Cecile.Paris@csiro.au

Abstract

This paper describes the role and the use of an explicit task representation in applications where humans interact in non-traditional computer environments using gestures. The focus lies on training and assistance applications, where the objective of the training includes implicit knowledge, e.g., motor-skills. On the one hand, these applications require a clear and transparent description of what has to be done during the interaction, while, on the other hand, they are highly interactive and multimodal. Therefore, the human computer interaction becomes modelled from the top down as a collaboration in which each participant pursues their individual goal that is stipulated by a task. In a bottom up processing, gesture recognition determines the actions of the user by applying processing on the continuous data streams from the environment. The resulting gesture or action is interpreted as the user's intention and becomes evaluated during the collaboration, allowing the system to reason about how to best provide guidance at this point. A vertical prototype based on the combination of a haptic virtual environment and a knowledge-based reasoning system is discussed and the evolvement of the task-based collaboration becomes demonstrated.

Keywords: task model, collaboration, gesture interaction, gesture recognition, virtual environment.

1 Introduction

The user interaction in virtual environments follows a spatial paradigm, while in traditional computer desktop environments the notion of Windows, Icon, Menus and Pointing interaction (WIMP) is prevailing. The interaction in virtual environments provides an intuitive access to simulated objects. Simulations have achieved nowadays considerable realism or plausibility, and the focus can be shifted towards more quality in the interaction (Smith *et al*., 1999). On the one hand, research has been done to specify new software models and specification languages to cope with non- and post-WIMP user interfaces (Jacob, 1996 and Beaudouin-Lafon, 2000). The challenges are identified in the parallel and continuous user interaction with the environment, with which traditional User Interface Management Systems (UIMS) are not designed to deal. On the other hand, effort has been made to populate a virtual environment with additional information to enrich the environment. In an early work of Feiner *et al.* (1993) on augmented reality systems, additional task-relevant graphical information is displayed as an overlay to the real environment. In contrast, in the approach of Bowman *et al*. (1999), additional information becomes integrated at appropriate spatial locations for the user to recognise, memorise and with which to interact. In general, explicit knowledge representation can be used to improve the interaction, e.g., to direct and assist the user in virtual environments (Aylett and Luck, 2000; Jung *et al.*, 1998). In the approach of Bowman *et al*. (1999), additional information appears when the user interacts with artefacts. We propose now, to not merely provide information about objects, but also make use of the explicit knowledge about what the user wants or has to do with the object. This knowledge about the task of the user allows the system to deliver to the user meaningful and relevant information at the right time. This can be understood as supporting the user's interaction at a conceptual level of interaction (Massink and Faconti, 2002).

We developed a vertical prototype for training motor-skills using a hand immersive haptic virtual environment. The interaction environment is enriched with information and makes uses of a comprehensive context model. The prototype shows how a formal organisation of the interaction in layers can be used to integrate different high and low level components. The interaction of the user can be characterised as parallel, continuous and multimodal at the physical and perceptual level (Nigay and Coutaz, 1993), while, at a higher level, the actions and goals are represented and processed in a serial, discrete and symbolic style. The user interaction with artefacts in the environment follows the metaphor of the computer as a tool or media, while on the task level, the computer can be understood as a dialogue partner (Schomaker *et al.,* 1995, Maybury and Wahlster, 1998). Finally, since the resulting system addresses issues of training and assistance, the described approach can be compared to those of Intelligent Tutoring Systems (ITS), where a computer is acting as a tutor, trainer or assistant and can collaborate with the user on tasks in simulated environments (Rickel *et al*. 2000). The majority of existing ITS are addressing the issue of teaching explicit knowledge as, e.g., described in Core *et al*. (2000). In contrast to that, the approach described here concentrates on training implicit or tactic knowledge, e.g., motor-skills.

In the following section, the paper describes a layered interaction model for information enriched interaction.

Then, in section 3, a detailed description of the used task representation is given. In section 4, a sensor network is described that provides the system with symbolic information about the user interaction. Finally, the vertical prototype of the integration of a haptic virtual environment and a knowledge-based system is described and the paper ends with a conclusion.

2 Information enriched interaction

The foundation for the support of user interaction in virtual environments has been already established by introducing interactive landmarks (Müller-Tomfelde *et al*., 2004). Interactive landmarks function as a meta model in order to deal with the different models of the representation of an object in a three dimensional interactive virtual environment and that in a knowledge-base reasoning system. This meta model avoids creating a new model with both representations. The landmark in the virtual environment has the function of an annotation of the scene-graph, for example, the knob of a door becomes annotated with an interactive landmark to specify a point of user interaction and to reason whether the user grabs the knob or not. The landmark enables the mapping of objects in the scene with those in the domain model of the reasoning system. This approach overcomes the structural barrier of virtual environments and knowledge-based reasoning systems and avoids reinventing the wheel, by making use of existing implementations.

2.1 Layered interaction model

It is supposed that the virtual environment is a generic environment, in which the user can interact with any sorts of artefacts. Each of these artefacts could host interactive landmarks. How the interaction in the environment is technically realised and implemented is not is the scope of this paper. Instead, only the user's actions and the spatial relation of the landmarks are relevant and considered for the interaction (Müller-Tomfelde *et al*. 2004). A layered model for human computer interaction has been chosen. It makes use of the reference framework for continuous interaction, as proposed by Massink and Faconti (2002).

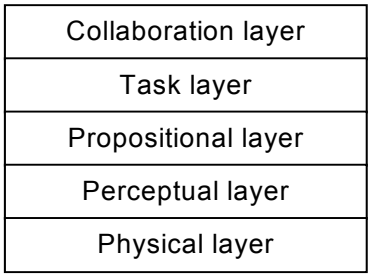

Figure 1 The layers of the interaction model based on the reference framework for continuous interaction (Massink and Faconti, 2002).

This framework for interaction addresses issues of continuous and parallel interaction while, at the same time, attempting to reduce the design complexity of continuous interaction systems. Each layer has clear and distinct functions and passes information up and down in the model or framework, respectively. In the system described in this paper, issues of continuous and parallel interaction are confined to the lower levels of the reference framework, where physical data streams are processed. Once these data streams have been transformed and interpreted into a symbolic representation, the interaction is characterised by serial and discrete processing of events, actions or states. As it will be discussed later, this boundary becomes less ridged than it may appear at this point.

In the physical layer, the interaction between the environment and, e.g., human sensors happens. Physical signals are perceived by the sensors and transformed to provide means at the perceptual layer. In the following the physical layer is not discussed since this would address the issues of psychophysics (like, e.g., described and discussed in Schomaker *et al.* 1995) and would go beyond the scope of this paper.

2.2 Collaboration and task layer

The top level of the interaction model is the collaboration layer. The collaboration management organises the activity of the participants in the collaboration. In a first and simple realisation of this organisation, a token is passed periodically to each participant and enables them to 'speak', while all others are 'listening'. This time period of 'speaking' in the collaboration is defined to be the *turn* of the participant. It has to be noticed here that the organisation of the collaboration at the collaboration layer is independent from the content of the task of each participant.

The task is to be understood as a structured sequence of primitive tasks or actions, that have to be performed in order to achieve the overall goal of the task. For example, the task to open a door becomes decomposed by the sequence: approach the door, grab the door knob, then turn the knob and push the door to open. Once a participant has the token, she pursues the task. The next step to do or the next goals to achieve in the situation is determined. It is conceivable that multiple executions of the same task result in different sequences of task primitives, but the overall goal of the task remains the same. This is due to the fact that a primitive task can be followed by a choice of other primitive tasks. Which next primitive task becomes selected for further processing depends on conditions and the actual state of the context of the task. In other words, multiple ways can lead to the same task goal, depending on the level of detail of the task description.

The task representation is declarative and does not contain any element that supports processing in a collaboration. Therefore a structure called *task environment* enables the management of the task processing and the storage of relevant information about the task during its execution. It contains, for example, the task name, the actual task primitive, the task processing mode, etc. Each participant of the collaboration owns a task environment and hence has a task to operate on

during the turns in the collaboration. The task environment also provides means for the collaboration management, e.g., to signal a 'claim' of a participant to 'speak' in the collaboration.

2.3 Propositional and perceptual layers

Once the next task primitive has been selected, the subgoal of this primitive becomes further planned top down in a content and presentation layer (equivalent to the propositional and perceptual layer, see in Figure 1). The execution of this plan then delivers information on the physical layer, e.g., to a visual display device or a text-to-speech generator. If the goal is a communicative goal, then the planning and the execution of the plan leads to an 'utterance' and can be used, e.g., to create text, as described in Moore and Paris (1993).

In case the computer functions as a trainer or assistant, the task of the trainer becomes executed by the computer, comparable to the execution of a computer program. But in contrast to a programming language, the task description is a specific language that supports the efficient description and authoring of tasks (more about this in section 3.1). The computer, in the role of a trainer, has the task to provide appropriate instructions and feedback to the user performing the task. This can be understood as a communicative goal which becomes processed by the content and presentation layer. This top down processing from the task layer to the perceptual layer enriches the environment at the physical layer by delivering tailored information to the user, based on the current state of the task. In a bottom up processing, from the perceptual layers to the propositional, the user's action becomes detected by refining the data streams from the environment and obtaining a reasonable representation of the current goal of the user (see section 4.1). This user goal gets evaluated to determine whether the goal matches with the stipulated task primitive or not.

3 Hierarchical task description

It is assumed that the user's mental model of A) the environment, the virtual environment in which the user is physically interacting and B) the training situation, in which the user is in collaboration with a trainer, are well established (Norman 1986). In other words, the user knows in what and with whom she is interacting. Now, we focus on the model of the topic of the training or assistance: the task of the user or the *domain task*. The domain task is a task that refers to a certain domain and is not further specified for the following considerations. Both the author, who creates the domain task and the user/trainee who performs or wants to learn this task, must have an unequivocal understanding about the represented task. This is considered to be a fundamental requirement of successful applications for training and assistance. Therefore, a hierarchical task representation based on a formalism that supports task annotations and procedural relationships (Tarby and Barthet, 1996) was selected for the task description. Advantages are efficient authoring and learning, as well as easy conceptualisation (Lu *et al*., 2000). The use of a task model has additional advantages at various stages in the Software Development Life Cycle, analysing the task of users, but also describing user interaction tasks in virtual environments (as for instance in Murray and Fernando, 1999). The domain task is graph-based represented and can become decomposed in a tree-like structure of primitive and composite tasks. The latter become further decomposed in their parts (see an example task representation in Figure 2). Once the model has been created, annotations and mapping functions enable further processing of the task model.

3.1 Domain task and tutorial task

A training situation is considered to be a collaboration in which a trainee is directed or taught by a trainer to do a domain task. Since participants of the training session are represented at the collaboration layer in the same manner, not only the trainee has a domain task, but also the trainer or assistant has a task, the *tutorial task*. This tutorial task is represented in the same way as the domain task and is explicitly available for processing by the computer. The goal of the tutorial task is to train the user performing this domain task.

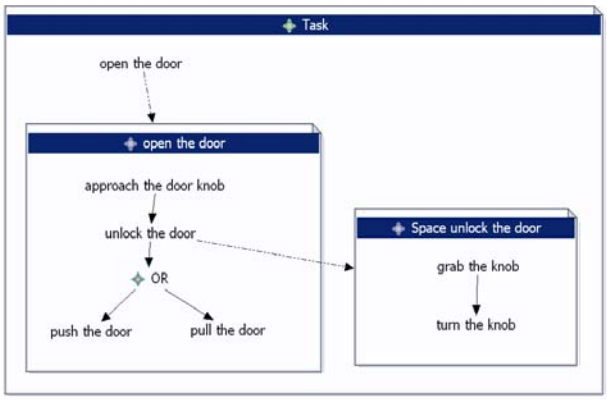

Figure 2 An example for a simple domain task

This reveals that both tasks need to be processed differently: On the one hand, the domain task is a description of what the user wants or has to do and becomes referred by the tutorial task. While, on the other hand, the tutorial task becomes executed on the computer. The separation between the tasks is a consequence, when considering the interaction in the environment as a collaboration of multiple participants. Both tasks have to be created by authors with expertise in the specific domain and in tutoring. Furthermore, to increase the impact of the explicit tutorial task it must be independent form the domain task to achieve, e.g., a high reusability. The explicitness of both tasks allow advanced processing, e.g., for documentation and transformation into other representations.

The tutorial task is implicitly defining the structure of the collaboration as a training session. A possible training session has a three-tier structure and consists of :

- **Introduction**: the trainer introduces the subject of training to the trainee. This could be, e.g., a list of subtasks that have to be fulfilled to complete the task.

- **Practice**: the trainer and trainee are working cooperatively on their individual tasks. The collaboration guarantees that each participant in the collaboration is able to 'speak' and operate on their individual task.
- **Summary**: the trainer informs the trainee about his or her overall performance and about the assessment for the session.

In order to enable the assessment of task performance the task description not only has to describe explicitly what has to be done in a task, but also provide means to decide whether the goal of a primitive task has been accomplished or not. An annotation on a task element called 'post condition' is used to reason about the goal achievement of the primitive task. This allows the trainer to evaluate the trainee's performance by examining the associate post condition of the task element the user is performing in the domain task. This link or connection between the domain task and tutorial task allows a sort of synchronisation of the domain task execution in the environment and its representation in the computer on the task layer of the interaction model (see Figure 1). This interaction can be understood as a conceptual interaction in contrast to the apparent interaction on the physical layer.

3.2 Tutorial strategy

During the tutorial practice, the strategy of the tutor how to teach can change. If, for instance, the candidate fails performing a sub task, the tutor can give further details about the actual task or otherwise reduce the amount of feedback if the user is performing well. The latter case helps avoiding overload of a good performing candidate with redundant and irrelevant information. Currently the tutorial task handles three levels of strategies, which can be changed by the tutorial 'reflection' during the interaction in the session practice:

- **Step-by-step**: During the collaboration, each step in the domain task becomes explicitly announced to the trainee and comprehensive feedback is provided.
- **Guide**: The trainee receives guidance for the task that has to be performed. Instructions and feedback are given whenever the trainee seems to require it. Situations where the trainee is not continuing in doing at least something, can be, for example, interpreted as an uncertainty on the trainee. Appropriate information at that point might help.
- **Rehearse**: This tutorial strategy mimics the situation where the trainee has to perform the domain task on his or her own. Nevertheless, the trainer monitors and tracks each step of the trainee in the domain task to give a comprehensive summary and assessment at the end of the training session.

In the course of the training session, the tutorial strategy influences the way the current instruction or feedback of the trainer becomes realised. The 'trainer' exhibits basic characteristics of individualised training by adapting of the tutorial strategy. This could lead to an optimised development of the learning rate over time, due to the fact that, on the one hand, minor errors of the trainee do not block the training progress, while on the other hand, a performing trainee is not forced to read instructions she not requiring. In a first approach, the adaptation process is designed asymmetrically. The step back to more verbose strategies, like, e.g., from guidance to step-by-step strategy requires less errors than it needs right actions stepping up again.

4 Action and gesture detection

The action and gesture detection is based on sensors, which transform incoming data streams into more meaning full information. For example, a stylus functions as the user representation at the physical level and its orientation, orientation and the applied force are fed into the sensors network to detect specific features. The state of one sensor becomes activated when its detection condition is fulfilled and all sensors become processed and their inherent states updated in real-time at a frequency of 60 Hz.

In this paper the term gesture refers to the user action that becomes recognised by processing and transforming the continuous data streams into a discrete and symbolic representation of the action. The sensor processing is located in the physical and perceptual layer and leads to a representation of the user actions, as an intention or goal at the propositional layer (see Figure 1). In other words, during the processing of the gesture recognition, a transition is made from a parallel continuous interaction, to a serial and discrete symbolic representation in the upper layer of the interaction model.

4.1 Sensor network

To determine a gesture, it is not useful just looking at the states of the sensors and pass them on for further reasoning. Ambiguous situations could occur, if the relation between the sensor results are not considered. Therefore, we organise the sensors in a hierarchical network to establish relationships between them (see Figure 3).

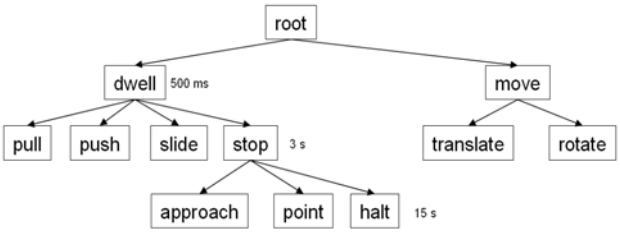

Figure 3 The sensor network of the action and gesture recognition. Each node represents a sensor for a specific action.

In the gesture recognition described in Latoschik (2001), nodes of a network are connected by a data routing mechanism. Instead, the connections in the network of sensors in the approach described in this paper refer to the conceptual relation between the sensors. On the basis of the current states of the sensors, a simple tree algorithm determines a resulting gesture or action that is passed on.

The algorithm of the search for a currently intended action by the user starts at the root of the tree and can be described as follows: Traverse the tree and return the activated sensor with no activated subsensors. The priority amongst multiple activated subsensors at one level is increasing from the right to the left. Repetitive detections are ignored. The resulting behaviour guarantees that only one gesture or action becomes passed on per detection cycle. This tree search algorithm also operates at real-time at 60 Hz, after the processing of all sensor states have been finished.

4.2 Temporal organisation

The relationships in the sensor network rely on spatio-temporal aspects, as well as on haptic aspects of the user interaction. The latter aspect addresses issues of the force applied by the user during the interaction, like push, pull, etc. In the simple example case, the user holds the stylus steady for more then 500 ms and applies a force in the direction of the orientation of the stylus, then the *dwell* and *push* sensors become activated, but only the *push* gesture is passed on. This behaviour is realised by the above proposed search algorithm and can be understood as early fusion of the hand position and orientation with the tactile modality (Oviatt *et al*., 2000). In another example the user released the stylus and no movement occurs anymore, the sequence of detected 'actions' is the following: after 500 ms the *dwell* action is detected, since in the next 3 s nothing changes, the *stop* sensor becomes activated. The tree algorithm determines *stop* as the next 'action' assuming that no other sensor at that level in the tree is activated. Finally, after another 15s the *halt* sensor gets activated. The used terminology is unfortunately inexact, since dwelling, stopping, etc. can not be considered as a user action; it is rather an absence of action. A more appropriate terminology is under study, following, e.g., the taxonomy of Bobick (1997).

4.3 Complex actions and gestures

So far, only basic actions or gesture like approach, point, pull, push and slide have been implemented. As depicted in Figure 3, an alternative to the *dwell* sensor is a *move* sensor. This branch in the sensor tree could serve to detect more motion oriented aspects of the user interaction like, *translation*, a linear trajectory in space, or *rotation*, a circular or curved trajectory in space. Since these sensors require more development, their realisation is postponed until application scenarios demand them. To illustrate further possibilities based on these sensors, we now describe the detection of a more complex gesture: assumed that the user's tool is a knife, and the user moves the tool forward in a linear manner, while at the same time applying force perpendicular to the blade of the knife. The user's action could then be detected in the sensor tree as a *slice* action. Furthermore, multiple consecutive *slice* actions with alternating translation direction could be grouped and constitute a *cutting* action. Such a complex gesture recognition would be the result of a late fusion, where a sequence of actions over a couple of seconds become grouped and classified as one action or gesture (Oviatt *et al*., 2000). The different complementary information of the different sensors over time becomes integrated at a sematic level into a single action or intention.

4.4 Queries to the sensors

The processing of the sensors network can be understood as a bottom-up oriented processing, which pushes new detected actions and gestures into higher layers in the interaction model. These actions become interpreted as the user's goal and drive the collaboration. During the planning of the information that is needed to be delivered to the user, it can be the case that information about the state of a sensor is required in order to optimally plan the delivery. In these situations, function calls enable to directly access the required information from the sensor in a pull-like or query-style manner. During planning, a top-down request is issued towards the sensors in the perceptual layer, and a response is fed back into the planning process. Therefore, managing the progress of the domain task and planning the information delivery can be influenced by or dependant on the states of the sensors. This pull-like access to sensor data blurs the boundary between the continuous and discrete layers in the interaction model, when, e.g., the success of a primitive task depends on a significant value of a physical sensor parameter.

5 Architecture of the vertical prototype

We developed vertical prototype for an information-enriched virtual environment to demonstrate the interplay of explicit task representations and gesture interaction. The system architecture of the prototype is the integration of a haptic virtual environment or Haptic Workbench (HWB, Stevenson *et al*., 1999) and the Myriad platform for information delivery (Paris *et al*., 2004) (Figure 4).

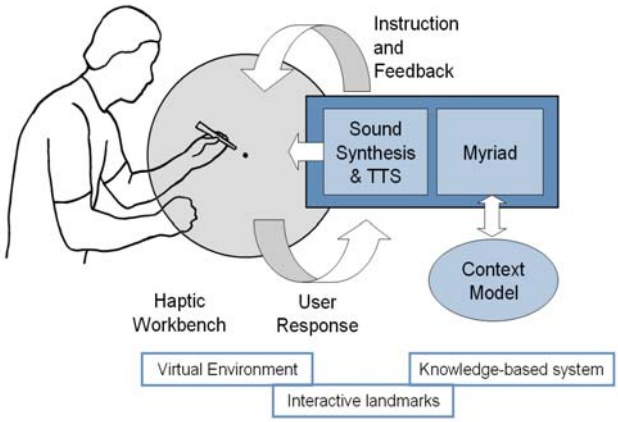

Figure 4. The basic components of the vertical prototype: a hand-immersive haptic virtual environment and the Myriad platform for information delivery.

These systems are heterogeneous in the structure of their representation of objects and information. We connected them through the concept of interactive landmarks. The user interacts in a hand-immersive virtual environment and receives force feedback corresponding to the dynamic data model of the virtual environment. Instructions and feedback generated by the Myriad

platform become visually displayed in the virtual environment, but can also be delivered via text-to-speech (TTS) or non-speech audio (as depicted in Figure 4) (Müller-Tomfelde, 2004). The communication between the HWB, the Myriad Platform and other applications make uses of standard computer network communication facilities.

The context model depicted in Figure 4 is accessed directly by the Myriad platform and provides the planning with all relevant information that is required for managing the collaboration and for executing the tutorial task. The context model specifies a domain knowledge model, a user model and a discourse history and the task models, amongst others.

5.1 The functional parts of the interaction model

The interaction model we described is realised partially in the haptic virtual environment and partially using the Myriad platform. While the continuous processing of the sensors detection is part of the rendering loop of the virtual environment, the Myriad platform basically handles with the symbolic output of this processing and sends queries directly to the sensors. The architecture of the of the prototype follows the schema of Figure 1, except that this representation (Figure 5) is rotated clockwise by 90 degrees.

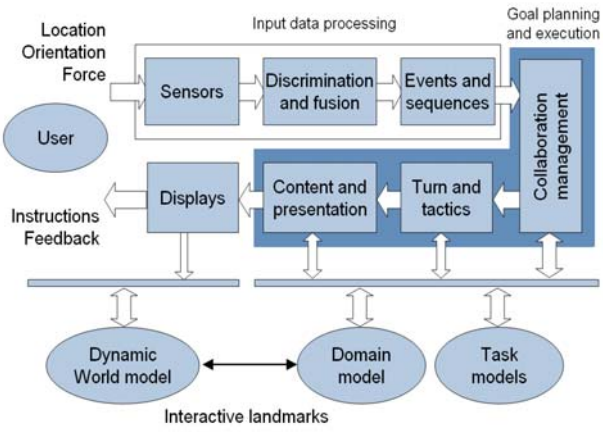

Figure 5. Details of the realisation of the interaction model in our prototype.

On the left hand side, the user interacts with the system on the physical level, i.e., the user moves a stylus in the haptic virtual environment. This user activity is processed by sensors, and the review of the states of all sensors provide actions and gestures after the fusion and discrimination (see section 4). Finally, the action or gesture becomes interpreted as the user's intention and represents the user's turn in the collaboration. At that stage, further development happens in the Myriad platform. The collaboration gets managed, and the turn is up to the trainer to act corresponding the tutorial task. The appropriate tactics are selected based on the prevailing tutorial strategy (see section 3.2). In the next layer the instruction and feedback becomes planned and finally delivered to the user in the virtual environment. In the Myriad platform, the embedded engine builds plans for all layers of the interaction model based on declarative plan operators, which become decomposed into subgoals (Paris *et al*., 2004). This chain of processing happens every time the sensor network emits an action based on the user's activity. Since the model of interaction is not hard coded, all the described processing include massive planning operations in real time. The dynamics of the collaboration and the task execution emerge through this planning with respect to an explicit domain model and the explicit tutorial and domain tasks.

6 First Experiences and future work

The vertical prototype as described in the prior section provides all the required feature and means to build an information-enriched virtual environment. Furthermore, the collaboration and task layer provides information about the task that is performed by the user. First experiences with simple training sessions reveal that the response time of the vertical prototype is reasonable using standard computer hardware. The sensor processing as well as the goal planner operate and communicate fast enough and provides the information to the user nearly instantaneously. More tasks have to be tested with the prototype to investigate the influence of, e.g., the complexity of the task on the system performance. Currently the turns of the participants in the collaboration is strictly alternating so when the 'speaker' remains silent the collaboration could be blocked. Although not a problem now, future demonstrators and applications have to show whether this approach has to be revisited or not. Finally, evaluations and user studies have to be planned and conducted to undermine the advantages of the proposed system and bring out the benefits for applications, like training or assistance.

7 Conclusion

This paper describes an approach for linking explicit task representations and gesture interaction to create an information–enriched virtual environment. The approach focuses on applications where the computer acts as a trainer or assistant to train implicit knowledge within a virtual environment. Another aspect of this approach is that it suits well for applications that require an explicit task representation, e.g., to document or assess the performance of user. The goal of this prototype is not only to enrich objects in the virtual environment, but also to support users with useful information to proceed in their tasks. Our approach makes use of a reference framework for continuous interaction to integrate the organisation of the collaboration and the tasks of the participants with the more continuous data processing for the gesture recognition. The explicit domain task representation allows the computer to mimic a trainer or assistant and to deliver appropriate information to the user in context. The tutorial task becomes executed on the computer to deliver the right instruction at the right time to the trainee during the course of the training session. First experiences with the vertical prototype using simple domain tasks are promising, and possible application scenarios considering other roles of the computer as a participant in a collaboration are investigated.

8 References

Aylett, R. and Luck, M. (2000): Applying artificial intelligence to virtual reality: Intelligent virtual environments. *Applied Artificial Intelligence*, 14(1):3-32.

Beaudouin-Lafon, M. (2000): Instrumental Interaction: An Interaction Model for Designing Post-WIMP User Interfaces. In *Proceedings of the CHI2000 Conference*, ACM, CHI Letters 2(1): 446-453.

Bobick, A. (1997): Movement, Activity, and Action: The Role of Knowledge in the Perception of Motion. *Phil. Trans. Royal Society London* B, 352, pp.1257-1265.

Bowman, D., Wineman, J., Hodges, L. and Allison, D. (1999): The Educational Value of an Information-Rich Virtual Environment. *Presence* 8(3): 317-331.

Core, M., Moore, J. and Zinn, C. (2000): Supporting constructive learning with a feedback planner. In *Papers from the 2000 AAAI Fall Symposium,* Menlo Park, CA: AAAI Press, pp, 1-9.

Feiner, S., MacIntyre, B., and Seligmann, D. (1993): Knowledge-Based Augmented Reality, *Communications of the ACM* 36(7): 53–62.

Jacob., R. (1996): A Visual Language for Non-WIMP User Interfaces. *IEEE Symposium on Visual Languages*, pp. 231-238.

Jung, B., Latoschik, M. and Wachsmuth, I. (1998): Knowledge-Based Assembly Simulation for Virtual Prototype Modeling. In *Proceedings of the 24th Annual Conference of the IEEE Industrial Electronics Society, Vol. 4*, IEEE, pp. 2152-2157.

Latoschik, M. (2001): A gesture processing framework for multimodal interaction in virtual reality. In *Proceedings of the 1st international Conference on Computer Graphics, Virtual Reality and Visualisation, AFRIGRAPH '01*. ACM Press, New York, NY, pp. 95-100.

Lu, S., Paris, C. and Vander Linden, K. (2002): Tamot: Towards a Flexible Task Modeling Tool. In *Proceedings of Human Factors*, Melbourne, Australia, pp. 25-27.

Massink, M. and Faconti, G. (2002): A reference framework for continuous interaction. In *International Journal Universal Access in the Information Society*, Electronic journal Vol. 1 n. 4, pp. 237-251.

Maybury, M. and Wahlster, W. (eds.) (1998): *Readings in Intelligent User Interfaces*. San Francisco: Morgan Kaufmann.

Müller-Tomfelde, C., Paris, C. and Stevenson, D. (2004): Interactive Landmarks: Linking Virtual Environments with Knowledge-Based Systems. In *Proceedings of the OZCHI*, Wollongong, Australia.

Müller-Tomfelde, C. (2004): Interaction sound feedback in a haptic virtual environment to improve motor skill acquisition. In *Proceedings of the International Conference on Auditory Display* (ICAD 2004).

Moore, J. and Paris, C (1993): Planning text for advisory dialogues: Capturing intentional and rhetorical information. In *Computational Linguistics,* 19(4), 651-694.

Murray, N. and Fernando, T. (1999): A Task Manager for Virtual Environments. In *Proceedings of the Workshop on User Centered Design and Implementation of Virtual Environments*, University of York.

Nigay, L. and Coutaz, J. (1993): A Design Space for Multimodal Systems Concurrent Processing and Data Fusion. In *Proceedings of the INTERCHI'93 Conference*, ACM, Amsterdam, pp. 172- 178.

Norman, D. A. (1986): Cognitive Engineering, Ch. 3 in Draper S.W. and Norman, D.A. (Editors) *UCSD, User-Centered System Design*, Lawrence Erlbaum Ass.

Oviatt, S.L., Cohen, P., Wu, L., Vergo, J., Duncan, L., Suhm, B., Bers, J., Holzman, T., Winograd, T., Landay, J., Larson, J. and Ferro, D. (2000): Designing the User Interface for Multimodal Speech and Pen-Based Gesture Applications: State-of-the-Art Systems and Future Research Directions. *Human-Computer Interaction, 15*, pp. 263-322.

Paris, C., Wu, M., Vander Linden, K., Post, M. and Lu, S. (2004): Myriad: An Architecture for Contextualized Information Retrieval and Delivery. In *AH2004: International Conference on Adaptive Hypermedia and Adaptive Web-based Systems*, The Netherlands, pp. 205-214.

Rickel, J., Ganeshan, R., Rich, C., Sidner, C. and Lesh, N. (2000): Task-Oriented Tutorial Dialogue: Issues and Agents. In *AAAI Fall Symposium on Building Dialogue Systems for Tutorial Applications*, North Falmouth, MA.

Schomaker, L., Nijtmans, J., Camurri, A., Lavagetto, F., Morasso, P., Benoît, C., Guiard-Marigny, T., Le Goff, B., Robert-Ribes, J., Adjoudani, A., Defée, I., Münch, S., Hartung, K. and Blauert, J. (1995): *A Taxonomy of Multimodal Interaction in the Human Information Processing System*. A Report of the ESPRIT Project 8579 MIAMI.

Smith, S., Duke, D. and Massink, M. (1999): The Hybrid World of Virtual Environments. In P. Brunet and R. Scopigno (eds), *Computer Graphics Forum*, 18(3), pp. C297-C307.

Stevenson, D., Smith, K., Veldkamp, P., McLaughlin J., Gunn, C. and Dixon, M. (1999): Haptic Workbench: A Multisensory Virtual Environment. *The Engineering Reality of Virtual Reality, Electronic Imaging '99*, San Jose.

Tarby, J.-C. and Barthet, M.-F. (1996): The Diane+ method. *In Computer-Aided Design of UserInterfaces, Proceedings of the Second International Workshop on Computer -Aided Design of User Interfaces (CADUI'96)*. Namur, Belgium. J. Vanderdonckt (eds), Presses Universitaires de Namur, Namur, pp. 95-119.