Space Walker: a Hyper Interaction Platform for Cosmonaut Training

Chao-Kang Feng, Timothy K. Shih, Hui-Huang Hsu,
Chun-Hong Huang, and Nick C. Tang
Tamkang University
Taiwan
{tshih, hhsu}@cs.tku.edu.tw

Stanislav Klimenko, Valery Afanasiev, and
Eugene Slobodyuk
Institute of Computing for Physics and Technology
Russia
klimenko@sim.ol.ru

Abstract

We present a hyper-interaction platform due to a joint project between the Institute of Computing for Physics and Technology, Russia and Tamkang University, Taiwan. The platform is an integration of several newly developed techniques, including a motion reaction control mechanism and a high precision 3-D model of the Russia/International Space Station. The platform can be used for mission training of cosmonauts. The integrated technologies can also be used in other virtual reality environments.

Key words: virtual reality, motion tracking, behavior understanding, back-propagation neural networks

1. INTRODUCTION

Human motion tracking was developed in the past few years. The MARG sensors were used to develop a system which can embed skeleton into a virtual environment [1]. Instead of using sensors, video-based tracking strategy was proposed [2, 3]. The difficulty of video-based approach is on how to deal with incomplete motion such as occlusion or bad viewing angle. Another difficulty is due to variations of light sources. Since sensors are expensive in general, video-based approach is usually used in commercial applications. In our earlier work, we have developed a tracking system that can reconstruct 3-D coordinates of markers on a human skeleton. In this paper, we propose a motion classification system using Back-propagation Neural Networks. As long as the type of motion is classified, a set of pre-defined reactions is applied to the motion on a 3-D environment. The particular platform that we use is a high precision model of Russia Space Station. We hope that, from motion detection, tracking to understanding, motion reaction can be considered as an interesting research topic.

We built a studio (see Figure 4) that has a surrounding background (270 degrees) with black color. Light sources are placed on the floor and on the ceiling to stabilize lighting parameters for video tracking. Two cameras are used for calibration of tracking points on a special designed suit. A back-projected screen and a projector are equipped such that the 3-D mode can be rendered on the screen without interfering (i.e., shadowing) by the actors in the studio.

2. MOTION CLASSIFICATION

In our earlier result [4], 3-D coordinates of tracking points were obtained from 2 or more cameras. In order to identify what motion the actor(s) perform, we use a neural network-based approximation strategy. A human motion specification can be represented as 16 motion vectors w. r. t. the 16 tracking points on a skeleton. For each tracking point, multiple vectors are used. Each motion vector is represented by a pair of angles (i.e., angles α and β defined in Figure 1). Angle α is in between the projection of motion vector V_{01} on the XZ-plane and the X-axis. Angle β is between the motion vector and its projection. An angle has a value between 0° and 360° (inclusive).

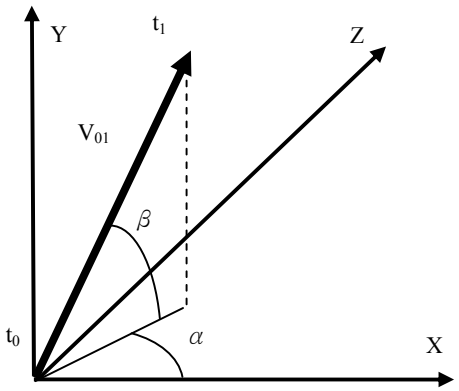


Figure 1: Definition of Motion Vectors

Instead of using precise coordinates in the motion identification computation, we use an approximation approach which only computes the rough location of each motion vector. For each angle (i.e., α or β), we define a Angle Code as

$$AC = floor(Angle / n)$$

where *Angle* can be α or β and *floor* is the floor function. The parameter *n* can be set to *45*. As such, Angle Codes are between 0 and 7. It is possible to set *n = 22.5*. Thus, Angle Codes will be in between 0 and 15. In our tracking procedure, since the length of a motion vector is short in general, coordinates of tracking points may not reflect the actions precisely. Using an approximation approach, on the other hand, can reduce the computational cost in our neural network model. For the j^{th} motion vector, angles α and β are represented by a pair of Angle Codes $C_j = (AC_{\alpha j}, AC_{\beta j})$. Thus, each motion vector, $M_{k,\ k+1}$, in between two consecutive time slots (time *k* and time *k+1*) with 16 tracking points is defined as

$$M_{k,\,k+1} = [C_{01}^{k+1} - C_{01}^{k},\ C_{02}^{k+1} - C_{02}^{k},\ldots,\ C_{15}^{k+!} - C_{15}^{k}]$$

Therefore, a *motion representation* can be defined as

$Motion = \{M_{0,1}, M_{1,2}, M_{2,3}, \ldots, M_{K-2,K-1}\}$

where K is the maximum number of motion vectors used in the motion specification. For instance, a motion representation can be *Motion* = {[(2, 3), (14, -3), …, (15, -9)], [(7, -2), (4, -13), …, (5,9)]} if the maximum number of motion vectors is set to 3. The representation of motions can be used in a neural network model. A trained neural network can be used as the mapping function from a motion representation to a *motion classification code* (MCC)

$$MCC_i = f_1 (Motion_x)$$

where i is between 1 and the maximum number of motion code to be classified, and x is assumed to be the maximum number of possible motions. The mapping, f_1, decides the MCC via the first level classifier. We propose a two-level classification mechanism using neural network.

As soon as a motion classification code, MCC_i, is classified by the first level classification mechanism, a series of MCCs may represent a continuous motion. However, similar continuous motions may have different durations. Therefore, a *motion window* is defined as an ordered list of motion control codes, with a variable number of MCCs. Each motion window starts at a motion control code. Thus, the number of motions windows (i.e., n) is equal to the number of motions in an action. A second level classifier, f_2, takes as input a motion window, and returns a motion signature (i.e., MS_j).

$$MS_j = f_2 (MW_j), \text{ where } 1 \leq j \leq n.$$

Motion signatures will be used to trigger reactions, which we will discuss in the next section.

3. MOTION REACTION AND CONTROL

Motion reaction can be defined in two types. The first type of reaction is changing the camera view of a virtual reality browser, by rendering the 3-D model (represented in VRML or the like) form different viewing angles. This type of reaction depends on the movement of a trainer. Thus, the definition of motions should deal with the movement of a trainer in the studio. Yet, the movement specification should consider the limited space in the studio. Special posture of the trainer is recognized as a particular movement. The second type of reaction includes changing objects in the 3-D model, such as moving an object or assembling an object in the space station. In addition, the second type of reaction may include the reaction of other avatars in the 3-D scene, if the system is implemented on a shared Web-based VR environment. From motion detection, motion tracking, motion understanding, to motion reaction, the pioneer perspective of this research project is to investigate how a virtual environment (including its avatars) should react with respect to a particular motion identified by the system.

For the two types of reactions controls (i.e., changing camera view and changing avatars), we implement them in the Virtools environment as the following:

Control Camera View (MRA_{Camera})

1. Create a target camera
2. Set camera position and viewing angle
3. Attach camera position to an avatar

Control Avatars (MRA_{Avatar})

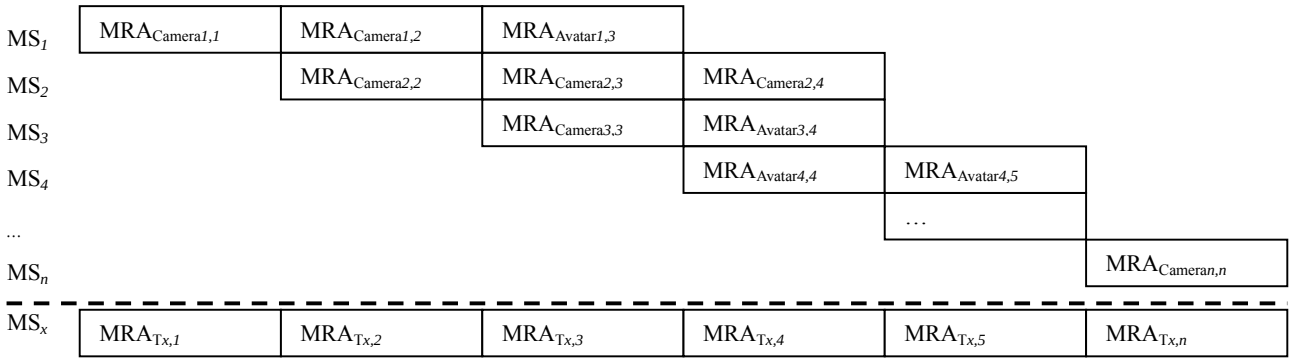
1. Create a database record by importing the avatar and 3-D model resources
2. Load the avatar and 3-D model into the Virtools environment
3. Add animations to the avatar
4. Initiate keyboard controller (using motion control) and connect character control to animations

After motion signatures are identified by the back-propagation neural network, each motion signature is associated with a list of motion reaction animations (i.e., MRA_{Camera} and MRA_{Avatar}). The animations rely on the above two types of controls in the Virtools. In addition, it may be necessary to have additional object transformations (or to add additional objects) in the 3-D scene, if the mission specifications have such a requirement.

3.1. Reaction Resolution

In the process of identifying what reactions should be applied to each motion signature, a resolution mechanism is used. Since each motion signature contains a variable number of motion control codes, it is possible to have two or more signatures that share a list of same control codes. In addition, since each motion signature is associated with a list of motion reaction animations, different signatures should share reaction animations in overlapped time slots. Thus, a mechanism to resolve the difference of reaction animations with respect to different motion signatures is necessary. Figure 2 illustrates a list of motion reaction animations for each motion signature. Each MS_x at time slot x has a list of motion reaction animations $MRA_{T\,x,\,t}$. Assuming that a motion reaction animation, $MRA_{T\,x,\,t}$, has a control type T, where T represents Avatar or Camera. The two subscripts, x and t, represents the indices of motion signatures and the index of time slots. For the two types of controls, different strategies are used to deduce the reactions.

The first equation in Figure 2 indicates that for the MRA with type Camera, the deduced motion reaction animations (DMRA) takes the average camera movements. That is, a DMRA will refer to all camera motions up to a current time slot t (i.e., $x' \leq x \leq t$), by taking the average camera coordinates in a 3-D space. This strategy allows a



$$DMRA_{camera\,t} = \left(\sum_{x' \le x \le t} MRA_{Camera\,x,t}\right) / (t - x' + 1) \quad \text{-----(1)}$$

$$DMRA_{Avatar\,t} = \begin{cases} MRA_{Avatar\,x'}, \text{ if Avatar } x' \text{ is not complete} \\ MRA_{Avatar\,t}, \text{ if Avatar } t \text{ is new} \end{cases} \quad \text{-----(2)}$$

Figure 2: Resolution of Motion Reaction Animations

continuous motion to consider snapshots taken in current time slot, as well as a few slots before. The deduced motion reaction animations results in a smooth camera movement. The second equation is for the Avatar type of animations. It is necessary to consider multiple avatars which response to a motion signature. For new avatar, a new animation associated with the avatar will start. However, animation of avatars started in previous time slots will continue.

The computation of DMRA is dynamic. That is, at each time slot *t*, due to each motion signature, a list of reaction behavior is sent to the reaction controller (to be discussed). Thus, the dynamic reaction results in a smooth motion.

3.2. Rendezvous Communication Control

A rendezvous control is a mechanism which allows two processors to meet at a certain time point, where the two processors complete certain tasks by their own before the rendezvous happen. We use two servers. The tracking server and the VR rendering server are synchronized in real-time. As soon as a motion signature (MS) is identified, the MS is kept in a *motion signature queue*, which is controlled by the tracking server. An action controller in the tracking server synchronizes with a reaction controller in the VR rendering server, by the rendezvous communication control mechanism. The mechanism decides where the two controllers will meet. As soon as the controllers meet, the action controller will send a motion message to the reaction controller, which interacts with a *motion reaction classifier* to identify how to trigger avatars in the 3-D VR environment via control messages (CMs). The activation of avatars and rendering of 3-D scenes consume heavy computation. Thus, the reaction controller needs to wait till the computation is complete in order to move to the next step. Similarly, the action controller needs to wait till the next motion signature is identified. The two controllers rendezvous. Thus, reactions in the 3-D environment will be synchronized with the motions by the actor. Figure 3 illustrates different entities of the rendezvous communication control.

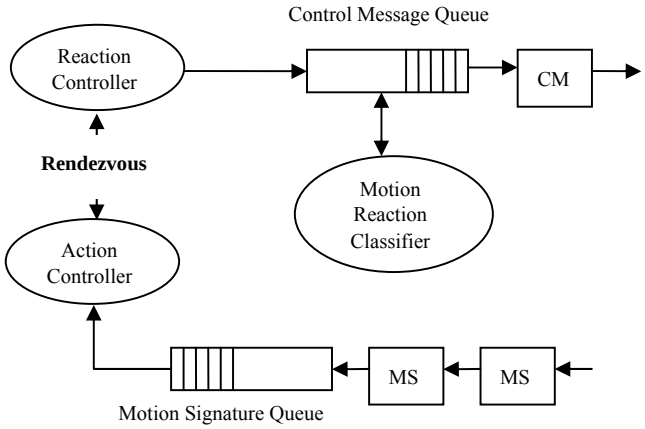


Figure 3: Rendezvous Communication Control

4. MISSION SPECIFICATION AND CONTROL

A mission specification is defined as a series of actions that a trainer needs to accomplish by using the system. The specification may include some check points, which is used as the base of an assessment model. Mission assessment can be maintained in a database for the analysis of individual training performance. The specification and simulation model are designed by domain experts. A mission specification can be defined as a list of events.

Each event is associated with a location in the 3-D space and a set of actions:

$$Mission_m = (Event_1, Event_2, Event_3, \ldots, Event_x)$$

$$= ((Location_1, \{A_{11}, A_{12}, \ldots, A_{1i}\}), (Location_2, \{A_{21}, A_{22}, \ldots, A_{2j}\}), \ldots, (Location_x, \{A_{x1}, A_{x2}, \ldots, A_{xk}\})$$

The order of actions within a mission can be defined in any topology. Thus, each action is associated with a predecessor and a successor, except the last and the first action. Actions can be performed in sequential or parallel, as suggested by the topology, which is designed by using a graphical user interface. The assessment of a mission includes an evaluation function which takes two parameters as input: an event in the mission and an event performed by the trainer. Comparison of coordinates can be asserted into the evaluation function as a check point (critical mission-based) or quantitative outcome (overall performance). Comparison of actions in each location can be accomplished by two factors: a detailed movement of the trainer, and the reaction from the avatar. Mission specifications as well as the assessments for each trainer will be maintained in a mission database. An analytical model based on the outcome of assessment can be used by a high ranking officer to evaluate individual trainers.

We implement a video studio and its software system, which is running on two LAN connected computers. Figure 4 shows the studio, the screen, and several examples of the 3-D model of Russia/International Space Station.

5. CONCLUSION

This paper proposes a platform and its underlying technology for cosmonaut mission training in the Russian/International Space Station. The system shows research collaboration results that we have accomplished in the past few years. The system use existing tracking techniques, with a new interesting issue in motion reaction control. The software is implemented on two computers with a rendezvous control mechanism to adjust speed of reactions. Our experience encourages us to further enhance the collaboration to include the interior model of the space station. We hope that, motion reaction will be an important issue in addition to detection, tracking, and understanding in video technology.

6. REFERENCES

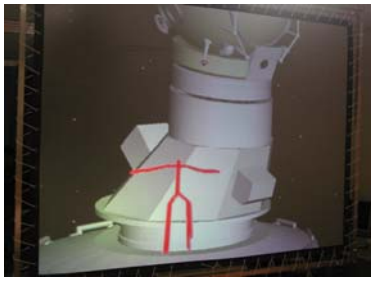
[1] Eric R. Bachmann, Robert B. McGhee, Xiaoping Yun, and Michael J. Zyda "Inertial and magnetic posture tracking for inserting humans into networked virtual environments," Proceedings of the ACM symposium on Virtual reality software and technology, November 2001, pp. 9-16.

[2] Z. Luo, Y. Zhuang, F. Liu, and Y. Pan, "Incomplete Motion Feature Tracking Algorithm in Video Sequences," in Proceedings of the IEEE 2002 International Conference on Image Processing, New York, USA, 2002, pp. III/617-III/620.

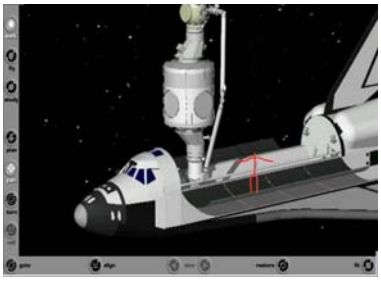
[3] L. Sigal, S. Bhatia, S. Roth, M.J. Black, and M. Isard. "Tracking loose-limbed people," in CVPR, 2004. pp. I421-I428.

[4] Timothy K. Shih, Hui-Huang Hsu, Chia-Ton Tan and Louis H. Lin. "Play With Video: An Integration of Human Motion Tracking and Interactive Video," the 11th International Conference on Distributed Multimedia Systems (DMS'2005), Banff, Canada, September 5 - 7, 2005.

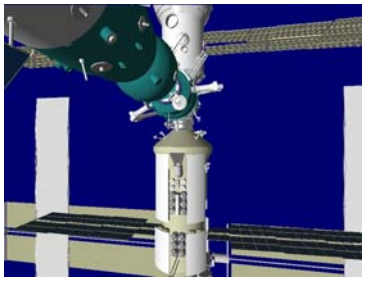
(a) The Video Studio with an Actor



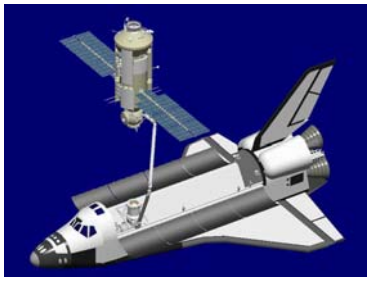
(b) The Back-Projected Screen



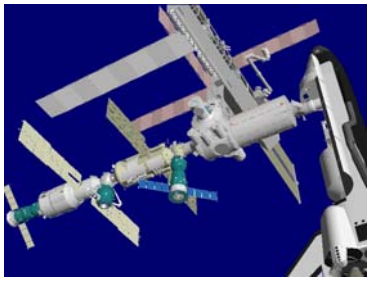
(c) Example of 3-D model



(d) Snapshot of Space Station



(e) Snapshot of Space Shuttle



(f) Snapshot of 3-D Model

Figure4: Implementation of the System and Studio

About the author

Chao-Kang Feng is a Professor in the Department of Aerospace Engineering, and the Vice President for Academic Affairs at Tamkang University, Taiwan. His current research interests include aerodynamics, similarity method and perturbation method, engineering mathematics. His contact email is ckfeng@mail.tku.edu.tw.

Timothy K. Shih is a Professor in the Department of Computer Science and Information Engineering at Tamkang University, Taiwan. He is a member of ACM and a senior member of IEEE. Dr. Shih also joins the Educational Activities Board of the IEEE Computer Society. His current research interests include multimedia computing and distance learning. His contact email is tshih@cs.tku.edu.tw.

Hui-Huang Hsu is an Associate Professor in the Department of Computer Science and Information Engineering at Tamkang University, Taiwan. He is a senior member of IEEE. His current research interests are in the areas of multimedia processing, bioinformatics, data mining, and machine learning. His contact email is hhsu@cs.tku.edu.tw.

Chun-Hong Huang is a Ph.D. student in the Department of Computer Science and Information Engineering at Tamkang University. His contact email is 892190017@s92.tku.edu.tw.

Chia-Ton Tan is a Ph.D. student in the Department of Computer Science and Information Engineering at Tamkang University. His contact email is 892190017@s92.tku.edu.tw.

Stanislav Klimenko is a Professor and Chairman in the Department of General and Applied Physics of the Moscow Institute of Physics and Technology (State University) and Director of the Institute of Computing for Physics and Technology. He is a member of ACM and a senior member of IEEE. His current research interests include virtual environments and distance learning. His contact email is klimenko@sim.ol.ru.

Valery Afanasiev is a senior scientist in the Institute of Computing for Physics and Technology. His research interests include virtual environment applications for space research. His contact email is v-afanasiev@korolev-net.ru.

Eugene Slobodyuk is an Associate Professor in the Department of General and Applied Physics of the Moscow Institute of Physics and Technology (State University). His research interests include data mining and virtual environment applications. His contact email is slobodyuk@rdtex.ru.