Perceptual User Interface by Human Body Action

Taniguchi, Rin-ichiro
Department of Intelligent Systems, Kyushu University

Date, Naoto
Department of Intelligent Systems, Kyushu University

Takaki, Kazuya
Department of Intelligent Systems, Kyushu University

Arita, Daisaku
Department of Intelligent Systems, Kyushu University

他

出版情報：Proceedings of 2nd International Workshop on Man-Machine Symbiotic Systems, pp.409-420, 2004-11
バージョン：
権利関係：

Perceptual User Interface by Human Body Action

Rin-ichiro Taniguchi, Naoto Date, Kazuya Takaki, Daisaku Arita, Satoshi Yonemoto†

Department of Intelligent Systems, Kyushu University
†Department of Intelligent Systems, Kyushu Sangyo University
e-mail: {rin,naoto,kazuya_t,arita,yonemoto}@limu.is.kyushu-u.ac.jp
http://limu.is.kyushu-u.ac.jp/

Keywords: perceptual user interface, real-time interaction, direct manipulation in 3D space, manipulation by human body posture, man-machine symbiotic systems

Abstract

This paper describes a vision-based direct manipulation interface for virtual reality systems which realizes both an avatar motion control and a virtual object manipulation. Our goal is to do seamless mapping of human motion in the real world into virtual environments. We think that the idea of direct human motion sensing will be used on future intelligent user interfaces. Our method can generate realistic avatar motion from the sensing data. We use virtual scene context as a priori knowledge. We assume that virtual environments provide action information for the avatar. In the virtual object manipulation, the user actions through physical-virtual interaction are interpreted as the object manipulation tasks. In other words, the actions which the user performs in the real world are converted into the scene events to manipulate the interesting virtual object.

1 Introduction

The goal of our research is to develop a vision-based 3D direct manipulation interface to realize seamless interaction in virtual environments, which are seamlessly coupled with the real world. Though many human interface devices force users to get used, essentially, systems must be adapted for the users[1]. From this point of view, 3D human motion sensing without physical restrictions, i.e., unwired sensing of human motion, is the most promising approach to realize seamless coupling between virtual environments in the system and the real world, where the user exists. This is because the human postures, i.e., 3D positions and directions of face, arm, feet, etc, indicate the user's intention.

In a virtual scene, a physical representation of the user corresponds to a 3D human figure model, i.e., an avatar, and, we think, realistic presentation of the substitute body is an important subject since the representation helps understand self body in the virtual scene or communicate with other existence. Toward realization of smart interaction, we have already developed a a real-time human motion capture by skin-color blob tracking[2]. The blobs mean a set of coherent image regions, which are mainly used as effective visual features[3, 4]. Though the reconstructed human figure motion should obey original motion of the user in detail, stable acquisition of details of human motion is a very difficult problem in the computer vision researches[5, 6, 7, 8, 9]. Therefore, we have developed motion synthesis from a limited number of perceptual cues, which can be stably estimated, to represent basic postures of the avatar in the virtual environments[10].

In addition, we address our framework to utilize virtual scene contexts as a priori knowledge. In order to make the virtual scene more realistic beyond the limitation of the real world sensing, we augment the reality in the virtual scene by simulating various events of the real world. A typical example of such augmentation is physical simulation. The physics law is defined, and in obedience to it, the virtual objects are moved realistically. In real-virtual interaction, every task is strongly related to objects in the virtual environments. In case of virtual object manipulation, the virtual environments can provide action information for the avatar. In other words, we assume that virtual objects can afford the avatar's action.

The idea corresponds to simulating affordance in virtual environments. In this paper, we outline our real-time human motion sensing and its application to 3D real-virtual interaction systems. As prototypical systems, we show two types of real-virtual interaction: one is interaction using full body motion, and the other is desktop interaction based on upper body motion.

2 Unwired Sensing of Human Action

Vision-based human action sensing is the key technique of our perceptual user interface, where real-time processing is quite important. Therefore, we have focused on realization of its real-time processing.

2.1 Sensing of human body postures

2.1.1 Basic idea

The basic algorithm flow of our real-time vision-based motion capturing is as follows:

1. Detection of visual cues
   - Color blob detection, silhouette detection, face direction detection.
   - Calculation of 3-D positions of features using multi-view fusion.
2. Human Motion Synthesis
   - Generation of human figure human body motion and rendering in the virtual space including calculation of the interaction.

The key point of our system is *human motion synthesis* referring to a limited number of visual cues, which are acquired by the perception process. To achieve this, we have established a human figure model and developed a vision-based algorithm for human motion sensing based on this model (details are discussed in 2.1.3).

When a real-time and on-line system is designed, an error recovery mechanism is quite important. If the system is based on feature tracking, once features fail to be tracked, the posture estimation process may reproduce unrealistic human postures, and it can not escape from the erroneous situation. Therefore, in order that we need not reset the system in such a case, we have introduced a simple error recovery process, which executes concurrently with the main human posture estimation process, and which always checks whether the human silhouette makes predefined shapes which are easy to recognize precisely. When the process finds the silhouette makes such shapes, it notifies the human posture estimation process, and the estimation process adopts the recognition result regardless of the estimation result. According to these considerations, we have designed a vision-based real-time human motion sensing as shown in Fig.1.

2.1.2 Acquisition of Perceptual Cues

Color blob detection We detect color blobs for a head, hands and feet, whose colors[1] are acquired in advance, and calculate 2D positions of the color blobs as the centroids of the blobs. Color blob detection is based on color similarity evaluation, which is done in HUV color space to exclude the influence of lighting condition as much as possible. We also extract a human silhouette region by background subtraction, which is used by an error recovery process and a human posture estimation process.

In principle, the 3D positions of the blobs can be calculated from two views. However, due to self-occlusion, we can not always calculate the 3D positions with only two views, and we have used a certain number of cameras. Therefore, we have to solve a stereo-pair correspondence problem in a multi-view

[1]Skin color blob for hands and a face and sock or shoe color for feet.

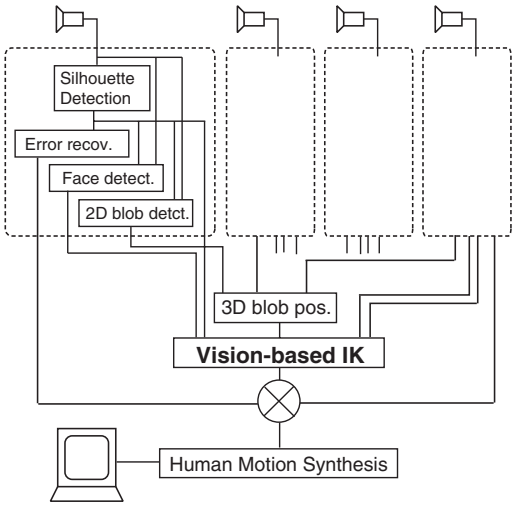
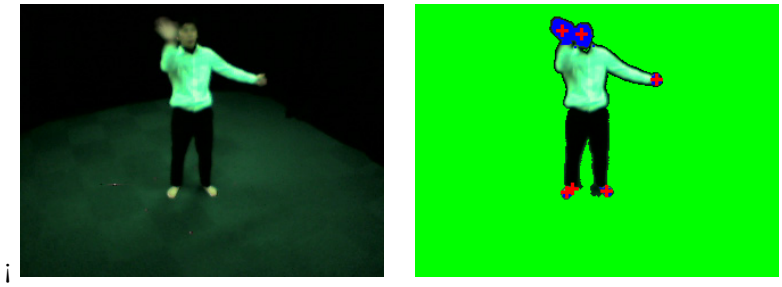


Figure 1: Software structure of real-time vision-based human motion sensing.



Input image Detected blob

Figure 2: Color blob direction

situation. We, first, extract possible stereo-pair candidates of the 2D blobs. When the distance of the two lines of sights passing through the centroids of two 2D blobs is small, the two blobs are a stereo-pair candidate. In this case, a point, which is the nearest to the both lines in a sense of the least square error, is judged to be their 3D point. Then, we classify their 3D positions into 5 clusters of feature points: head, right hand, left hand, right foot, left foot. Classification is done based on the distances from the feature points detected in the previous frame. In each cluster, we estimate the feature point position as the average position of the 3D position candidates after a dense part of the cluster is selected.

Face direction Face direction is an important feature in human posture and is indispensable for interactive application. The problems here is the low resolution of face region, because cameras are arranged to capture a full-body and a face becomes small in the field of view. Therefore, face features such as eyes and mouths can not be clearly detected, and, then, feature-based or structure-based techniques of face direction estimation such as [11, 12] can not be applied. We, here, have employed a template matching method preparing face templates

Figure 3: Face direction estimation

with multiple aspects.

Currently, we prepare about 100 templates for each person in advance. Making templates is quite easy: 3D rotation sensor is attached to top of the user's head and the user makes a variety of face directions in front of a camera before using the motion sensing system. Recording the output of the rotation sensor and the face image frames synchronously, we can get face templates with a variety of face directions. It takes only a couple of minutes. This approach is not sophisticated but quite practical, i.e., it is very simple but quite easy to use and robust.

To reduce the computation time, we have employed an eigen-space method[13]. The size of templates is 45×50, and the dimension of the eigen-space is 70. After detecting a face region, or skin-color region which corresponds to a face, the face region is normalized to have the same size as that of the template. Fig.3 shows an example of the face direction estimation. The estimation accuracy is not very high because of the low resolution images, but in most of interactive applications, we can get certain feedbacks and can modify the face direction based on them. We think the accuracy acquired by this approach is high enough to use.

2.1.3 Vision-based Inverse Kinematics

Our problem is to estimate human postures from a limited number of perceptual cues, which are blobs corresponding to hands, feet and head. This problem can be explained in a framework of Inverse Kinematics (IK) in the field of robotics. IK is to determine joint angles $\theta_{\mathbf{i}}$ of a manipulator so that the position of an end effector, or a final point, $\mathbf{P}_n$, coincides with a given goal point $\mathbf{G}$: $\mathbf{P}_n(\theta_1, ..., \theta_n) = \mathbf{G}$: where the manipulator has n segments. The difficulty here is that even if the goal is attainable[2], there may be multiple solutions and, thus, the inverse problem is generally ill-posed.

In our problem, end effectors are hands, feet and a head, and the goals are the blob positions acquired by the perceptual process. The posture estimation, which is to decide the positions of joints of the human model, is achieved by calculating the joint angles in the frame work of IK. In human posture sensing, each joint position acquired by IK should be coincide with a joint position of a given human posture, and, therefore, we have to find the unique and correct solution.

Our method to solve this problem is divided into two phases: *acquisition of initial solution* and *refinement of initial solution* For simplicity, here, we explain human posture estimation of a upper body.

Acquisition of initial solution Inverse Kinematics is solved by an ordinary numerical method[14] and initial candidates of 3D positions of shoulders and elbows are calculated. Here, we assume that the lengths of the bones in Fig.4 are given in advance. At time t, a hand position $(x(t), y(t), z(t))$ is represented as

$$(x, y, z) = \mathbf{P}(T_x(t), T_y(t), T_z(t), \theta_1(t), \theta_2(t), \cdots, \theta_N(t)) \tag{1}$$

where

- $T_x(t), T_y(t), T_z(t)$ indicate the head position in the world coordinate,
- $\theta_1(t), \theta_2(t), \theta_3(t)$ indicate rotation angles between the world coordinate and the local coordinate of the head, which are calculated from the face direction,
- $\theta_j(t)(4 \leq j \leq N(= 8))$ indicate rotation angles among connected parts .

We suppose that, at time $t+1$, the hand position moves to $(x(t+1), y(t+1), z(t+1))$, the head position moves to $(T_x(t+1), T_y(t+1), T_z(t+1))$, and the head direction changes to $(\theta_1(t+1), \theta_2(t+1), \theta_3(t+1))$. Here, we slightly modify $\theta_j(t+1)$, $(4 \leq j \leq N)$ so as that the hand position, i.e., the position of

[2]If the distance of the goal to the initial point of the manipulator is larger than the sum of the lengths of the segments, the goal is not attainable.

the end effector, $\mathbf{P}(T_x(t+1), T_y(t+1), T_z(t+1), \theta_1(t+1), \cdots, \theta_N(t+1))$ approaches the goal position $(x(t+1), y(t+1), z(t+1))$. Repeating this process until the end effector position coincides with the goal position, we acquire the positions of a shoulder and an elbow. In order to exclude impossible postures, we have imposed a possible range on each angle θ_j.

Refinement of initial solution The posture estimated in the previous step is just a solution of inverse kinematics, and it is not guaranteed that it coincides with the actual posture. This is due to ill-poseness of the inverse kinematics. To estimate the posture more accurately, we refine the acquired solution by referring to input image data. The basic idea is simple: if the shoulder and elbow positions acquired by the previous phase are correct, they should be inside of the human region in 3D space. Otherwise, the acquired solutions are not correct and they should be modified so as to be included in the human region. Here, we empirically assume that the shoulder position is acquired by solving the basic inverse kinematics, and we mainly refine the elbow position. Its basic algorithm is as follows:

- We have the shoulder position by solving the inverse kinematics and the hand position by color-blob analysis.
- When the lengths of its upper arm and forearm are given, the position of its elbow is restricted on a circle in 3D space. The circle is indicated by C.
- When the elbow is searched on the circle, we exclude impossible values, with which the arm get stuck in the torso.
- As shown in Fig.5, an elbow detection rectangle is established in a plane which is constructed by the shoulder, an hypothesized elbow and the hand.
- Then, in each view, the rectangle is reversely projected on the image plane and correlation between the projected rectangle and the human silhouette region is calculated.
- Then, by varying the position of the hypothesized elbow, the correlation R can be parameterized by ϕ, which is the angle around the center of the circle C. We search for ϕ giving the maximum R, which indicates the elbow position.

If the refinement process fails to find the correct solutions, the system restarts the modification process by changing the initial values. Since this system is required to work in real-time, this iteration is stopped when the deadline comes, and intermediate result is returned. Fig.6 shows a typical effect of vision-based IK, where the positions of elbows are collected. In addition, based on our rather complex, i.e., high DOF, human figure model, face direction is estimated, and the shoulder positions are slightly shifted according to the hand positions.

2.1.4 Computation cost

For full-body human motion analysis, we have used 8 sets of IEEE1394-based color cameras (Sony DFW-V500) with f:4mm lenses, which are geometrically calibrated in advance. The images are captured with the size of 640×480 pixels, and the frame rate is 15 fps. To realize real-time processing, we have implemented our vision-based human motion analysis on PC-cluster with 3.2 GHz PentiumIVs, where detection of perceptual cues in each view is executed in parallel, and where solving IK and human figure generation are executed in a succeeding PC. Computation times required in

Vision algorithm	Time
2D blob detection:	4ms
3D blob calculation:	5ms
face direction estimation:	5ms
IK initial solution:	3ms
IK solution refinement:	27ms
template matching for error recovery:	11ms

Table 1: Computation time

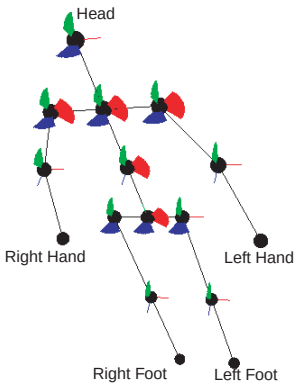


Figure 4: Human model

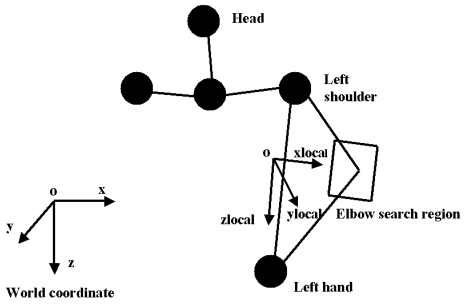
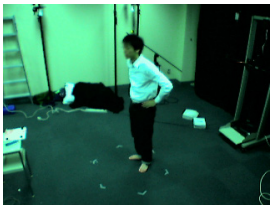
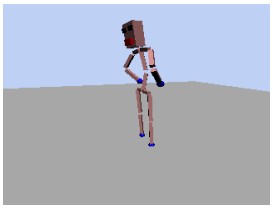


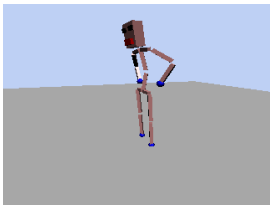
Figure 5: Elbow position estimation



(a) Actual posture



(b) Estimation by basic IK



(c) Estimation by vision-based IK

Figure 6: Result of human posture estimation by vision-based IK

major vision algorithms of our human motion analysis are shown in Table 1, which indicates that real-time processing of our vision-based human motion sensing can be achieved on the PC-cluster.

When the system is applied to a desktop interface, or when the system only analyzes upper-body motion of the user sitting in front of the display, the number of cameras, and the number of PCs as well, can be reduced.

2.2 Other features for human action sensing

Possible interaction modes of the perceptual user interface other than body postures are gaze and hand shape. The problem here is that the sizes of a face and a hand are relatively small compared with a human body, and thus, for example, we can not acquire their high resolution image when we capture a full body shape in a single camera view. Though a possible approach is using cameras specific to face and hand observation, we have to incorporate additional face and hand tracking mechanisms using active cameras into the system. Since user interface itself is not the main task of the system, we want to make the structure of the user interface as simple as possible, and, therefore, we have compromised as follows:

- For interaction with full body postures, we substitute the gaze with the face direction. This is because in situations where we freely move our bodies in 3D space the gaze usually coincides with the face direction.

- For desktop interaction with upper body postures, we estimate the gaze. This is because in the desktop interaction, the gaze conveys quite important information about the user's state. In addi-

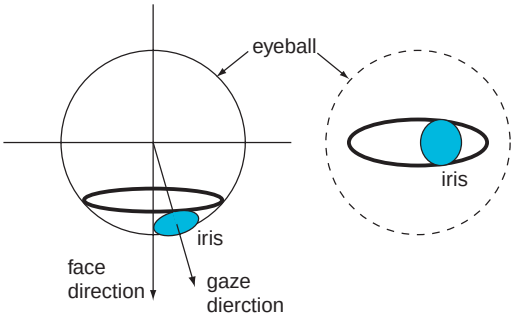


Figure 7: Geometrical model of eye

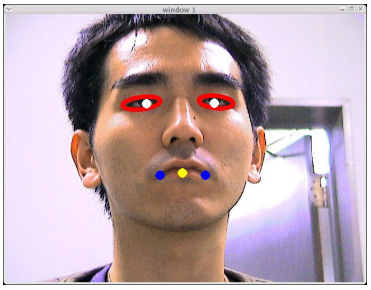


Figure 8: Detected eyes and irises

tion, we do not move our faces largely, and it is not very difficult get relatively high resolution face images, where we extract image features for gaze estimation.

- For the hand shape, we do not recognize hand shapes precisely, but we just classify them into a couple of categories, such as open and close, depending on simple 2D shape parameters. We currently do not expect to have detailed finger shapes for fine manipulation of the interface.

The vision-based gaze estimation we build for the perceptual user interface is summarized as follows. The algorithm is based on a geometrical model of eye structure shown in Fig.7. To calculate 3D positional information, the stereo with two cameras are used.

1. Face direction estimation based on image features[12], which is used to calculate the center of an eyeball. We assume that the eyeball center in 3D space is located on a face direction vector passing through the center of eye region. We also assume that the size of the eyeball is known in advance[3].

2. Detection of an eye region and an iris. They are detected by extracting elliptic features around possible eye regions (See Fig.8).

3. Matching the detected eye region and iris with our eyeball model, the rotational parameters of the eyeball are calculated and, then, the gaze is estimated.

3 3D Direct Manipulation User Interface

3.1 Basic concept

We assume that each object in the virtual environments affords essential information about the user's action. In other words, we assume that interaction among the virtual objects and the user (i.e. avatar) should be properly performed by making each virtual object give rise to afforded action. The states of the virtual objects are decided whether the virtual objects are handled or not, according to the distance between the virtual objects and the blob positions of the user. In addition, our system considers the scene constraints in the virtual environments to generate the realistic scene. Since the virtual scene is completely recognized by the system, the context reasoning in the interaction through the virtual environments is not more serious than in the real world sensing.

[3]We assume that its diameter is about 24mm based on anatomical knowledge

Manipulation of virtual objects In our system, the user handles static objects in the virtual environments such as cup placed on the shelf. According to the afforded action, the objects change their states, i.e. they are moved following user's motion, and then they are released. The user also communicates dynamic objects such as anthropomorphic human figure agents.

Interaction using scene constraints Based on afforded action information, the interaction system can have the scene constraints which lie between the virtual object and the user's motion. We assume that motion trajectory which the user can control in the virtual environments should be limited by the scene constraints. For example, motions of doors, levers and handles in the virtual environment should be circular, and those of drawers should be linear. When possible state of an virtual object can be defined by parameters $\mathbf{q}$ such as pose and shape deformation, and $\mathbf{p_o}(\mathbf{q})$ indicates position of the object limited by the scene constraints, and $\mathbf{p_v}$ is the blob position estimated by vision process, then $\mathbf{q}$ which minimizes $||\mathbf{p_o}(\mathbf{q}) - \mathbf{p_v}||$ is proper parameters to generate realistic motion, under the condition that $\mathbf{p_v}$ should be close to $\mathbf{p_o}(\mathbf{q})$ and that $\mathbf{q}$ should be within possible range of the parameters. The generated posture of the avatar is exactly followed the position $\mathbf{p_o}(\mathbf{q})$.

Interaction through real objects Observing real objects as well as the body postures, the system can obtain action information which should be difficult to be represented by only the body postures, such as rotation of virtual objects. For example, when the user handles a rectangle panel with a few beacons in the real world and the system can estimate the pose parameters of panels, they can be easily exchanged for the rotation of a virtual object in the virtual environments. In addition, the system can easily capture appearances, such as colors, aspects, and motions, of unknown objects attached to recognizable real objects employing measure space limited by positions of the recognizable objects.

Generation of secondary motion Vision processes can not acquire all the detailed human motion information. A typical example is the user's hand shapes, which can not be precisely acquired by the system. In our perceptual user interface, human actions are interpreted based on observable human motions and afforded information provided by virtual objects. Therefore, precise human motion are visualized based on interpreted action. For example, when the system recognizes that the user grasps an object, hand/finger of the avatar is visualized as the avatar grasps the object, though the user's hand/finger shape can not be observed in such detail. This can augment the reality, or the naturalness, of the interaction.

3.2 Role of gaze

To make the interface more efficient and more natural, we are introducing gaze-based control mechanisms. Currently we consider the following three kinds of gaze utilization:

Disambiguation Possible target objects and types of manipulation can be restricted referring to the gaze. For example, in the interface by human posture, the target object is basically decided by the positions and the shapes of the human hands. However, since hand shapes are not easy to recognize, it is not easy for the system to distinguish whether the hand is intentionally approaching the object for manipulation or the hand is accidentally approaching. Usually, the target object is located along the line of sight, and the gaze can give information whether the user is going to manipulate the object. Combining the human body motion and the gaze, we can disambiguate possible interpretation of human actions, and as a result, we expect that the number of objects and manipulations which can be handled efficiently is increased.

Assistance of manipulation When we can detect the target object which the user look at, we can attach annotations to the object, which helps the user's manipulation. The possible annotation is as follows:

- Enhance the visibility of the target object, such as high lighting, enlargement, or translation (pop-up) of the object.
- Display supplemental information of the target object, such as properties, possible manipulations, etc.

In addition, the system can recognize the user is confusing, referring to his/her gaze movement: when the user look around restlessly, the user can not find or decide what he/she does next. In such cases, the system also displays some assistance information, possible manipulation, manipulable objects, etc.

Concealment of system delay One of the problems of the vision-based interface is its delay due to vision processing, and the feedback of the human action displayed on a monitor is a bit delayed. This sometimes causes unnatural feeling in using the interface. If we can recognize the user's intention referring to the gaze and if we can predict the user's action, we can compensate the feedback, which can virtually hide the system delay. For example, suppose that the system delay is Δt, which is a period required to visualize an action on the display[4]. When the system recognizes that the user's hand is approaching an object referring to the gaze information, the system can predict a trajectory of the hand and the hand position at Δt ahead. By visualizing the hand not at its observed position but at the predicted position, the system delay can be virtually eliminated. When states of objects are changed by the user's action, object states should be also predicted and visualized.

In addition to the above functionalities, we can attach supplemental functionalities to the gaze, depending on the application. For example, change of the view can be controlled by the gaze. When we handle a large 3D virtual space, we can only see a part of the space through the display, and we have to change the view. In this situation, we can decide a new view depending on the gaze. Of course, some intentional commands for changing the view referring to the gaze should be interpreted.

4 Prototypical Applications

4.1 VR walk-through

As an interaction system using full body action, we have implemented a VR walk-through system (Fig.9). In this application, posture parameters of the user in the real space are applied to her/his avatar (the user's other self) in a virtual town in real-time and the user can control the avatar in a virtual town freely. To make correspondence between a limited moving space in the real world and a large-scaled virtual town, the walk-through system recognizes the user's pseudo walking action and makes the avatar walk.

When the user makes stamping gestures, her/his stamping motion is captured and reconstructed in a virtual space. The system recognizes the speed of up and down of the feet, and translates it into the walking speed of the avatar. Therefore, we can simulate not only walking but running, i.e., we can control the speed of moving. Usually, stamping gestures are interpreted as commands for moving forward, but they can be interpreted as commands for moving backward as well. When the user makes a stamping gesture, and when he/she at first moves a bit backward, the succeeding stamping gestures are interpreted as walking backward.

[4] an action at time t is normally visualized on a display at time $t + \Delta t$

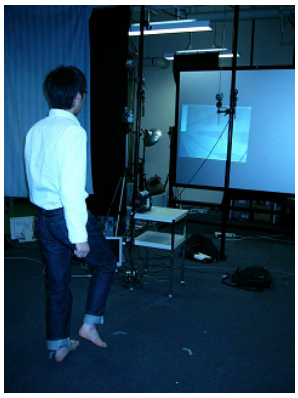


Figure 9: VR walk-through environment

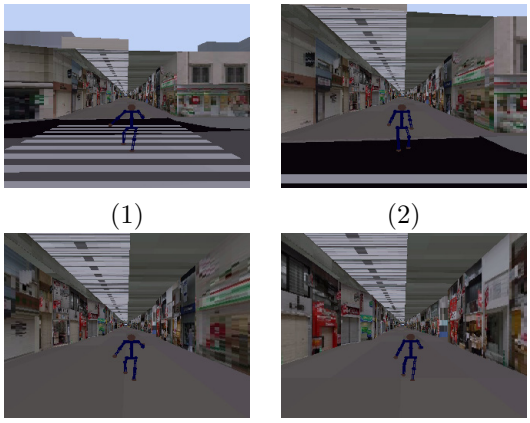


Figure 10: Generated images of VR walk-through

To rotate a field of view in a virtual space, the system recognizes the user's face direction. When the user rotates his/her face toward the right/left for a while, the view is changed to the right/left. The period when he/she look toward the right/left is translated into the angle of view change.

Currently human actions are interpreted as commands for moving in a virtual town, and no interaction with virtual objects are introduced. However, it can be easily implemented based on the framework of our perceptual user interface. We expect the full-body interaction can be applied to variety of large-scaled interactive system such as VR-based communication, artistic performance, amusement etc.

4.2 Virtual object manipulation on the desk

We have implemented several virtual object manipulation systems under the framework of desktop interface. For the desktop interface, we have used two cameras[5], because the moving space of the user is rather restricted and because we can expect that the user does not make very complex postures. The system is implemented on a standard PC and can perform in real-time and online from vision process to scene rendering. The user can only monitor the projected virtual scene with the 2D display. Fig.11 shows a typical system setup.

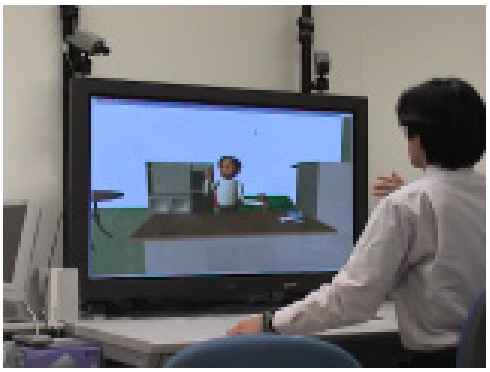


Figure 11: System overview

In order to evaluate our avatar-based direct manipulation, we acquired a scene data for a simple manipulation task in a virtual environment, which contains three virtual objects, A: a cup, B: a tea pot and C: a fruit, on a table, which are handled in the task. The scene data includes action events, which is afforded or picked up by the system. The data was captured in real-time for 1000 frames. Fig.12 shows several shots of the manipulation task (frame: 324, 332, 394, 431). In the task, the user handles object A, B, C and A by turns. Fig.13 shows the afforded

[5]Two firewire cameras (SONY DVW-V500) and a wide 2D display are set in front of the user.

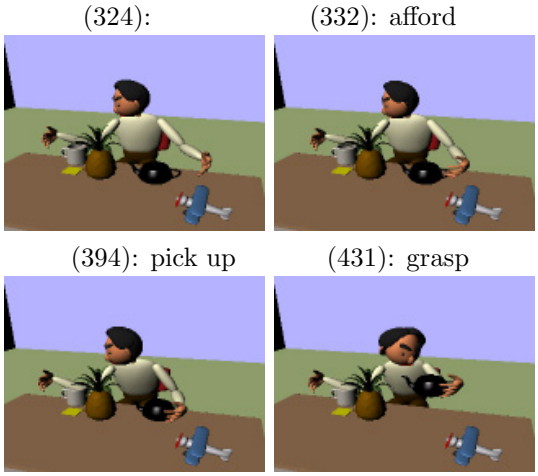


Figure 12: Object manipulation by an avatar

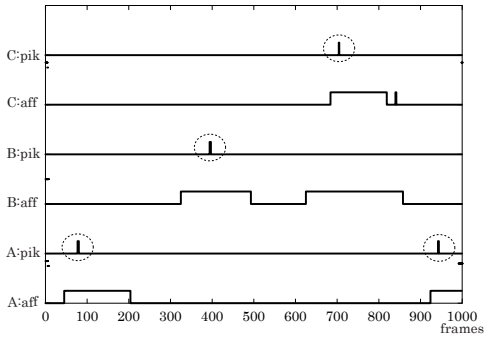


Figure 13: Secondary motion afforded by object manipulation

actions for the task. For example, "A: aff" column indicates whether the afforded action of the object A is driven or not. In this example, the afforded action corresponds to grasping action of the avatar (his fingers are bent). In this example, the state transition is *static*, *move* or *release*. "A: pik" column indicates whether the afforded action of the object A is picked up or not by the system. If the action is picked up, both the avatar motion, *grasping*, and the object motion, *move*, are driven, and the object A is moved along with the motion of the grasping hand. We can also see the afforded finger motion is driven at frame 332 in comparison with the appearance of left hand at frame 324, which augments the reality of the avatar's action.

5 Conclusion

With the aim of making computing systems suited for users, we have developed a vision-based 3D direct manipulation interface, and have applied it to prototypical interaction systems. We have proposed physically constrained human figure motion synthesis to generate realistic motion from a limit number of perceptual cues, and have realized smooth interaction by employing scene constraints in the virtual environments. The prototypical systems have shown the effectiveness of 3D direct manipulation by human body motion for smooth interaction between virtual environments and the real world.

As future work, we plan to extend our system to make more complicated interactions possible, recognizing high-level action derived by user's intention. Also we will introduce precise hand shape analysis to our perceptual user interface, especially to desktop interface, because hand shapes convey quite important information about object manipulation. However, a hand has a large number of DOFs, and real-time processing of its shape analysis is not an easy task.

References

[1] M. Weiser: "Some Computer Science Issues in Ubiquitous Computing," *Communications of the ACM*, Vol.36, No.7, pp.75-84, 1993.

[2] S. Yonemoto, et al: "Real-Time Human Motion Analysis and IK-based Human Figure Control," *Proc. Workshop on Human Motion (HUMO2000)*, pp.149-154, 2000.

[3] M. Etoh, et al: "Segmentation and 2D Motion Estimation by Region Fragments," *Proc. Int. Conf. on Computer Vision (ICCV93)*, pp.192-199, 1993.

[4] C. Wren, et al: "Pfinder: Real-Time Tracking of the Human Body," IEEE *Trans. on Pattern Analysis and Machine Intelligence*, Vol.19, No.7, pp.780-785, 1997.

[5] J.M. Rehg, et al: "Model-based Tracking of Self-Occluding Articulated Objects," *Proc. Int. Conf. on Computer Vision (ICCV95)*, pp.612-617, 1995.

[6] C. Bregler: "Learning and Recognizing Human Dynamics in Video Sequences," *Proc. Computer Vision and Pattern Recognition (CVPR97)*, pp.568-574, 1997.

[7] D. Metaxas: *Physics-based Deformable Models Applications to Computer Vision, Graphics and Medical Imaging*, Kluwer Academic Publishers, 1997.

[8] T. Nunomaki, et al: "Multi-part Non-rigid Object Tracking Based on Time Model-Space Gradients," *Proc. 1st Int. Workshop on Articulated Motion and Deformable Objects (AMDO2000)*, pp.72-82, 2000.

[9] J.Deutshcher, et al, "Automatic Partitioning of High Dimensional Search Spaces Associated with Articulated Body Motion Caputure," *Proc. Computer Vision and Pattern Recognition (CVPR2001)*, Vol.2, pp.669-676, 2001.

[10] N. Date, et al: "Real-time Human Motion Sensing based on Vision-based Inverse Kinematics for Interactive Applications," *Proc. Int. Conf. Pattern Recognition (ICPR04)*, Vol.III, pp.318-321, 2004.

[11] P.Yao, et al: "Face Tracking and Pose Estimation using Affine Motion Parameters", *Proc. Scandinavian Conf. Image Analysis*, pp.531-536, 2001.

[12] S. Ishii, et al: "Real-time Head Pose Estimation with Stereo Vision," *Proc. the 9th Korea-Japan Joint Workshop on Frontiers of Computer Vision (FCV2003)*, pp.79-83, 2003.

[13] H.Murase, et al: "Visual Learning and Recognition of 3-D Objects from Appearance," *International Journal of Computer Vision*, Vol.14, No.1, pp.5-24, 1995.

[14] L.Wang, et al: "A Combined Optimization Method for Solving the Inverse Kinematics problem of Mechanical Manipulators", *IEEE Trans. Robotics and Automation*, Vol.7, No.4, pp.489-499, 1991.