High-level Human Figure Action Control for Vision-based Real-time Interaction

Satoshi Yonemoto
Kyushu Sangyo University
2-3-1, Matsukadai Fukuoka, 813-8503 Japan
yonemoto@ip.kyusan-u.ac.jp

Rin-ichiro Taniguchi
Kyushu University
6-1 Kasuga-koen Fukuoka, 816-8580 Japan
rin@limu.is.kyushu-u.ac.jp

Abstract

This paper presents an real-time interaction system which consists of human motion tracking using skin-color-based Gaussian blobs and human motion synthesis based on real-time inverse kinematics. Our purpose is to do seamless mapping of human action in the real world into virtual environments. In general, virtual environment applications such as smart interaction require real-time human motion tracking without special devices or markers. For the sake of realization of smart interaction, we assume that virtual objects existing in virtual environments can afford human figure action, that is, the virtual environments can provide action information for human figure model, or avatar. In this paper, we demonstrate a real-time, on-line and smart desktop interaction which realizes high-level action recognition through interaction with virtual objects.

1. Introduction

Man-machine seamless 3-D interaction is an important tool for various interactive systems such as virtual reality systems, video game consoles, etc. To realize such interaction, the systems have to estimate motion parameters of human bodies in real-time. Up to the present, as a method for human motion tracking, many motion capture devices with special markers or magnetic sensor attachments have been employed. Since they need special marker-sensors, they often impose physical restrictions on the objects, or the humans.

On the other hand, recently, various image-feature-based motion capturing systems which do not impose such restrictions have been developed as computer vision applications[1]. Although the vision-based approach still has problems to be solved, it is a very smart approach which can achieve seamless human-machine interaction. Therefore, we are undertaking to develop an image-feature-based motion capturing system, giving consideration to alleviating scene constraints and physical constraints imposed on the system as little as possible.

To analyze human motion, image features such as blobs (coherent region)[1][3][4] are usually employed. Many researchers have developed skin-color region tracking and stereo reconstruction methods using general region clustering. In particular, *Pfinder*[1] has shown that blob tracking is applicable to many simple applications. Recently, *purposeful human motion*[5] employing the above blob tracking has been proposed. It is based on an analysis and synthesis framework with dynamics engine[2] and an HMM based multiple behaviour model. The method is applied to upper body motion estimation, and temporary occlusion in the intersection between blobs of both hands or head-and-hand is handled, although the method does not solve the corresponding problem yet. In gesture recognition systems, as well as in human motion tracking, human motion primitives are also dealt with. In these systems, gesture representation is symbolically defined, and the system input acquired by usual 2D vision process can be used as only 2D-based action signals. Such 2D-based approaches are not appropriate for our purpose, which has to generate actual 3-D body postures, or 3-D positions of head, arms, feet, etc.

In our case, i.e., in case of object manupulation in virtual environments, the most important point is that every task in the virtual environments is strongly related to, or afforded by, objects in the virtual environments. In particular, if the work space is constrained only in a virtual environment, and if 3D human motions achieved in the real world are only used to manipulate objects in the virtual environment, vision process can be simlified. In other words, the virtual environment provides action information for human figure model, and all the detected events can be interpreted so as to be matched with the virtual environment, while mismatched interpretations can be abandoned. Use of *a priori* knowledge for the virtual objects can make vision process robust and can make it possible for the stystem to understand high-level action by *afforded* information, although

usual vision process focuses on real-world sensing under uncertain environments.

See the following example of high-level action in a virtual environment:

A user grasps a cup and a teapot,
he fills the cup with water,
and then he drinks the water.

Since the virtual objects are completely understood by the system, action interpretation can be realized through interaction among the objects and the user. High-level action is represented by a chain of symbolized action fragments through interaction with the virtual objects. As a result, motion trajectory constructed by the user's input can be identified as an practical example of *interaction scenario*. Moreover, considering the scene context, detail of motion which is difficult to acquire for vision process can be automatically generated as a secondary motion.

In this paper, we present blob-based human motion analysis (Section 2), and a real-time motion synthesis method which can estimate human postures with limited perceptual cues (Section 3). And then we show an overview of our real-time interaction system.

2. 3D Blob Tracking

In general, to extract a rough sketch of human body from the image, coherent image regions, or Gaussian blobs are mainly used as an effective visual features. However, in fact, observed regions of a face and hands in the image variously change their shapes, sizes and motions. In this case, to classify pixels with skin color as a blob, that is, to segment each blob, the detailed shape of the blob should be caught. In particular, since regions with skin color such as a face or hands are uniform and have very similar colors, it is almost impossible to make exact distinctions among them only from color information. Therefore, not only color information but also shapes of region boundaries must be considered. Therefore, we introduce a deformable shape property to Gaussian blob.

2.1. Skin Color Detection

As a low-level vision process, we use skin color detection to extract several blobs.

1. Background subtraction and detection of bounding box: Bounding box containing a human body is detected after background image subtraction and thresholding are applied.

2. Color identification: In this system, skin color regions observed in an input image are interpreted as hands and a face blobs. We assume their colors can be represented in a simple parametric form which is relatively robust for illumination changes. In other words, we assume their color features (r,g,b) of each pixel are represented in the following quadratic equations of intensity i:

$$R = R_2 i^2 + R_1 i,\ G = G_2 i^2 + G_1 i,\ B = B_2 i^2 + B_1 i \quad (1)$$

For skin color to be identified, six model parameters, or coefficients, $R_1, \cdots, B_2$ are estimated in advance from a training data set, or real skin color blob images. In color identification, system computes the following error between observed color features (r, g, b) of a pixel and the model color features (R, G, B) calculated, according to the above equation, from the intensity i of the pixel.

$$error = (\hat{r} - \hat{R})^2 + (\hat{g} - \hat{G})^2 \quad (2)$$

where, $\hat{r} = r/(r+g+b)$, $\hat{g} = g/(r+g+b)$, $\hat{R} = R/(R+G+B)$, $\hat{G} = G/(R+G+B)$.
The system identifies color of a pixel as a color giving the minimum error that is less than a certain threshold value.

2.2. 2D blob Tracking by Pixel Classification

Blob Formulation Gaussian blob (i.e., ellipse) $\mathbf{e} = (\mu, \mathbf{\Sigma})$ is described as a statistical Gaussian model, a mean μ and a covariance $\mathbf{\Sigma}$. Since our explicit goal of 2D blob tracking is to estimate μ stably, it is necessary to do exact pixel classification, or exact region boundary detection.

In addition, we extend a blob definition so that local deformation can be represented:

$$\mathbf{p}(s) = \mathbf{e}(s) + \mathbf{d}(s) \quad (3)$$

where, d is an additional deformation term and s is a perimeter.

Initial labelling: In each frame, initial labeling of blobs are executed as follows:

1. Skin color detection mentioned above is achieved, and pixels within the bounding box are classified into either skin color pixels, non-skin-color foreground ones, or background ones.

2. Pixels classified as skin color ones are clustered into initial blobs based on the spatial proximity to the blobs detected in the previous frame.

3. Temporary moment parameters of each initial blob, (μ', $\mathbf{\Sigma}$') are calculated.

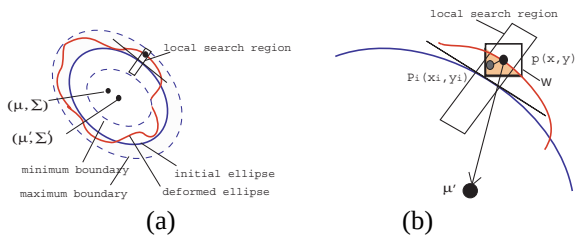

Figure 1. (a)Deformable ellipse. (b)Local search region.

Local boundary search by a deformed ellipse: The second step is to estimate exact local region boundary using (μ',$\mathbf{\Sigma}$'). For the sake of an estimation of the local region boundary, a search region is aligned with envelop points of the perimeter (see **Figure1**). At each search points, likelihood of region boundary is computed in the local support (i.e., window W), and then the point having maximum likelihood is acquired. Although a visual feature of edgeness is usually employed to detect region boundary, a lot of edges within regions of shirt and pants, which are adjacent to region of skin color, are also observed. Therefore, in this paper, as likelihood of region boundary, the following separability $J(\mathbf{p})$ is employed.

$$if \sum_{(x_i,y_i)\in W} n(x_i,y_i) > 0, \text{ then}$$

$$J(\mathbf{p}) = \left| \frac{N_W}{2} - \sum_{(x_i,y_i)\in W} p(x_i,y_i|skin) \right|^{-1} \quad (4)$$

otherwise J is set to minimum value.
$p(x_i, y_i|skin)$ is likelihood of skin color, N_W is the number of pixel within W, and $n(x_i, y_i) = \overline{\mathbf{p}_i\mathbf{p}} \cdot \overline{\mu'\mathbf{p}}$ (see **Figure1(b)**). Hence, $\mathbf{d}(\mathbf{s})$ is determined by $\mathbf{p}$ which maximizes $J(\mathbf{p})$. Because blobs often moves variously and change their shapes non-rigidly, the correspondence calculation in successive frames is not robust. Therefore an approach without explicit correspondence calculation, which consists of initial labeling and expansion/shrink of ellipse, works better. By gathering the center of mass μ within the deformed ellipse again, the 2D position of a blob is exactly modified.

2.3. Estimation of 3D blobs

When a 2-D blob is detected in two views, or in multiple views, the 3-D position of the blob can be estimated by stereo vision. In blob tracking, precise estimation is not required and, therefore, we have employed a simple but fast multi-view fusion strategy. The algorithm of 3-D blob position calculation adopted here is as follows: according to camera calibration information, for each of the views, a line of sight, or a vector from the origin of the camera coordinate system to the center of mass of the blob, is calculated. Referring to the lines of sight, the 3-D position of each blob is calculated. When a line of sight calculated for one view is parameterized as $\mathbf{T}_1 = \mathbf{o}_1 + t_1\mathbf{d}_1$ (t_1 is a parameter), and the rest of the lines of sight as $\mathbf{T}_j = \mathbf{o}_j + t_j\mathbf{d}_j$ (t_j is a parameter; $j = 2, \cdots, J$), the intersection point $\mathbf{T}$ is approximated as a point on the line of sight $\mathbf{T}_1$ whose average distance to the other lines of sight is smallest in the sense of the least squares error.

$$\mathbf{T} = \mathbf{o}_1 - \frac{\sum_{j=1}^{J}(\mathbf{d}_1 \times \mathbf{m}_j, \mathbf{o}_1 \times \mathbf{m}_j - \mathbf{n}_j)}{\sum_{j=1}^{J} ||\mathbf{d}_1 \times \mathbf{m}_j||^2}\mathbf{d}_1, \quad (5)$$

where

$$\mathbf{m}_j = \frac{\mathbf{d}_j}{\sqrt{1 + ||\mathbf{o}_j \times \mathbf{d}_j||^2}}, \mathbf{n}_j = \frac{\mathbf{o}_j \times \mathbf{d}_j}{\sqrt{1 + ||\mathbf{o}_j \times \mathbf{d}_j||^2}}.$$

3. Model Fitting for Perception Data

3.1. Real-time Inverse Kinematics

Information acquired in 3D vision process is just 3-D positions of blobs, which correspond to a torso, a head, hands and feet of human body. Therefore, to estimate the other body posture from these cues, the number of which is less than the degree of freedom of the body, we have to solve the inverse kinematics. In our case, a human full body is represented as a multi-part articulated object, or as 14 parts with 23 degrees of freedom, and the 3D blob positions are given as the goal positions, or the end effectors. Of course, there are approaches in which knees and elbows are detected based on contour analysis, or silhouette analysis, but they cannot stably detect those positions for many postures.

The goals of inverse kinematics which we have designed can be summarized as follows:

- Inverse kinematics of four connecting links, which are two arms and two legs, can be solved in real-time.
- Even when goal positions (3-D blob positions) given by 3D vision process are not precise, a solution can be derived to some extent.
- The solution gives us continuous and natural-looking motion of human body.

In our case, as mentioned above, the 3-D blob positions acquired by the vision process are sometimes imprecise. In

other words, the goal positions are sometimes established at positions where physically possible solutions cannot be derived. Therefore, we interpret each of the given goals as the combination of the direction of goal and the distance to goal. When the goal position is located where physically possible solution can not be derived, we find a solution in which the direction of the connecting link coincides with the goal direction.

$$\begin{aligned}\mathbf{g}_i^w = \quad & \mathbf{T}^b\mathbf{R}^b(0, {R_y}^b, {R_x}^b)\mathbf{R}^{b'}({R_z}^{b'}, 0, 0) \\ & \mathbf{T}^{l_1}\mathbf{R}^{l_1}({R_z}^{l_1}, {R_y}^{l_1}, 0)\mathbf{R}^{l'_1}({R_z}^{l'_1}, 0, {R_x}^{l'_1}) \\ & \mathbf{T}^{l_2}\mathbf{R}^{l_2}(0, 0, {R_x}^{l_2})\ \mathbf{t}^e\end{aligned} \tag{6}$$

where $\mathbf{g}_i^w = (g_x^w, g_y^w, g_z^w, 1)^T$ is a goal vector ($i = 1, \cdots, 4$); $\mathbf{T}^b$, $\mathbf{R}^b$ and $\mathbf{R}^{b'}$ are matrices representing the body pose; $\mathbf{T}^{l_1}$, $\mathbf{R}^{l_1}$,$\mathbf{R}^{l'_1}$, $\mathbf{T}^{l_2}$ and $\mathbf{R}^{l_2}$ are pose matrices related to link 1 (L1) and link 2 (L2) respectively; $\mathbf{t}^e$ is a translation vector related to the end-effector position of L2.

Here, some rotation elements are represented in two matrices—$\mathbf{R}^{l_1}$ and $\mathbf{R}^{l'_1}$ of L1, for example. We have divided the original rotation matrix into two matrices to simplify analytical solution of our inverse kinematics. In $\mathbf{R}$, R_z, R_y, R_x represent roll, pitch, and yaw angle respectively (see [8]).

3.2. Natural-looking Motion Adaptation from Motion Capture Data

Parameters which are not represented explicitly in the solution of the inverse kinematics, such as the pitch of elbow (${R_z}^{l'_1}$) (we call it *characteristic angle*), can be used to control more natural-looking human body posture if necessary. In this paper, we have used a pre-learned constant value based on measurements of limb postures, which is measured by using color-marker based motion capture device. The learning process is realized by estimating the typical characteristic angle for various real arm or leg motions.

4. Smart 3D Desktop Interaction

Based on the 3D vision process and the motion synthesis the above mentioned, we have developed a real-rime 3D desktop interaction system, which has vision-based 3D motion input, and which can control human figure model.

Features of the system are summarized as follows:

- Vision-based 3D motion input without any special markers (head and hands)
- Real-time motion synthesis by model fitting, based on analytical inverse kinematics from limited perceptual data, or physically-based motion synthesis [1]
- Error recovery for robust vision process
- Affordance-based high-level action recognition
- Support of various *interaction scenarios*

In the next, we show about affordance-based high-level action recognition, and error recovery for robust vision process.

4.1. Affordance-based Human Action Recognition

We assume that each object in the virtual environments affords essential information about user's actions. For example, when a user stretches his arm for a cup, on the shelf and grasp it, the virtual object, or the cup predicts that it may be grasped and moved. Therefore, supposed that each virtual object drives the reaction, an intentional action for the user can be recognized for the system, according to 3D action input estimated by the vision process.

Chaining action information *afforded* by virtual objects, or action fragments obtained through interaction among the user and the virtual objects, high-level action recognition can be realized. The concrete examples will be shown in the experiments.

4.2. Error Recovery for Robust 3D Vision Process

Action information *afforded* by virtual objects can also make 3D vision process robust. We propose landmark monitoring based on action information *afforded* by virtual objects. The landmark indicates a 2D position of a virtual object, which is calculated by projecting its 3D position for each camera view. Our landmark monitoring algorithm is achieved as follows: Using the virtual camera model defined by calibration data acquired in advance, 3D positions of virtual objects are projeted for each camera view. By monitoring the projected 2D positions, even if the tracking of the blob fails, the blob can be captured and tracked again when the blob encounters the monitored landmark. **Figure2** shows an example of projected landmark points overlayed in the real images (four white blocks) and the objects (cube, cup, cylinder and teapot).

For each visible object, whose action is defined, the following process is executed.

1. 2D monitoring Set search windows around each landmark point, and then check whether 2D blob is found by observing skin color fragment.

2. 3D checking Check whether a 3D blob is found by integrating the results of multiple views. If a 3D blob is found, then we determine it corresponds to a lost blob, and the blob trajectory is modified.

The algorithm is also employed in detecting default user position.

[1] In this paper, the detail of the latter method is omitted.

Figure 2. (left) Four landmark points of virtual objects. They are overlayed in two real images. (right) The virtual objects (cube, cup, cylinder and teapot).

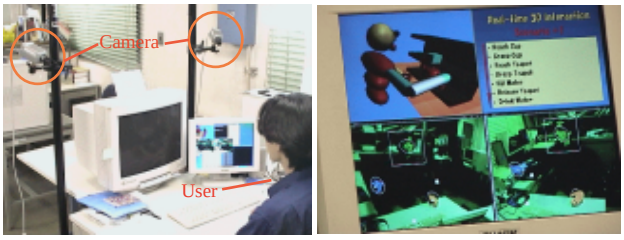

Figure 3. Interaction system overview: (left) the real scene. (right) the virtual scene.

5. Experiments

A prototypical system developed here is a desktop-type 3D interaction system, and perform in real-time and on-line from vision process to rendering. We have implemented the system on a PC (Pentium III 850MHz x2), which has two IEEE1394 based camera (**Figure3**). The system executes three main modules: The first module is image capturing and 2D image analysis, the second one is 3D estimation and model fitting, and the last one is rendering and engine of *interaction scenario*. A user controls CG avatar in the virtual environment by handling his action. Registration of real scene and virtual one is archived in advance.

In order to demonstrate high-level action recognition, we have constructed a chain of user's action events as an instance of *interaction scenario*. *The interaction scenario* is promoted through real-time interaction by user's 3D motion.

scenario#1: Object manipulation - avatar handles teapot and cup - **Table1** shows the practical example of *the interaction scenario*. The right column in the **Table 1** indicates user's input action, and the left column indicates *afforded* motion with each virtual object. Observing *afforded* action for each virtual object, high-level actions which the user intends can be understood. **Figure4** shows several shots of the real interaction scene. The number in the **Table1** corresponds to scene number(1-8). No.1 is the initial scene. No.2-8 correspond to the scenes after executing the user's (avatar's) action in the scenario#1 respectively. According to the scenario, *afforded* action is properly derived, and interaction among the objects and the avatar is realized. This means that recognition of high-level action is achieved[2].

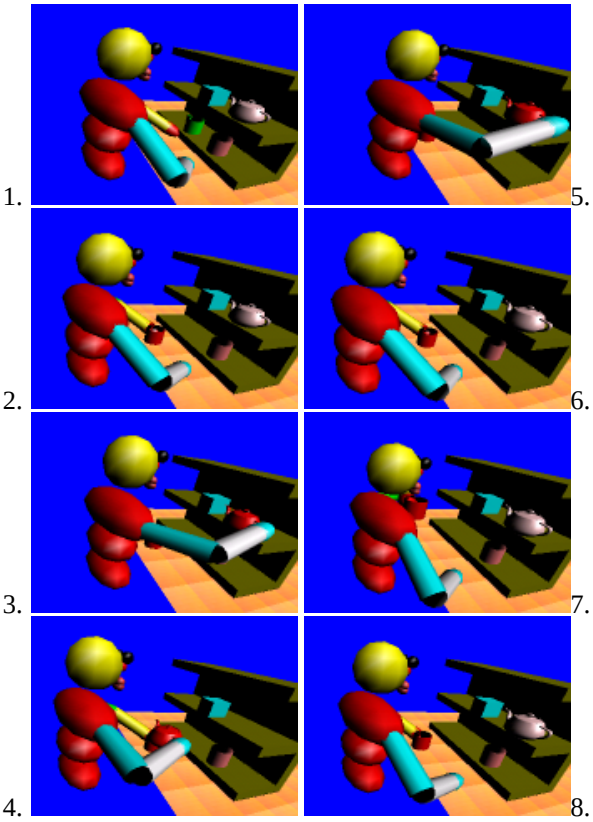

Figure 4. Scenario#1: - Avatar handles a teapot and a cup -

scenario#2: Interaction with virtual robot The next scenario consists of interaction among the avatar and a virtual robot. **Figure5** shows several shots of the real interaction scene. **Table2** shows the practical example of *the interaction scenario*. The number in the **Table2** corresponds to scene number(1-4). No.1 is the initial scene. No.2-4 correspond to the scenes after executing the user's (avatar's) action in the scenario#2 respectively. According to the sce-

[2] In these shots, secondary motion should be generated from *afforded action* is omitted.

nario, *afforded* action is properly derived, and then, robot's reactions are automatically generated from their *afforded* actions.

No.	object	*afforded* user's action	user's action
			left hand reaches CUP
	CUP	grasp and move	
2			left hand reaches on the DESK
3			right hand reaches TEAPOT
	TEAPOT	grasp and move	
4			right hand reaches CUP
	CUP	fill with water	
5			return TEAPOT
	TEAPOT	release on the SHELF	
7			left hand reaches mouth
	CUP	drink the water	
8			left hand reaches on the DESK

Table 1. The practical example of Scenario #1.

No.	object	*afforded* action	user's action
2			hand reaches robot
	ROBOT	recognize handshake and move robot's hand	
			hand reaches robot's hand
	ROBOT	keep handshake	
3			shake hand
4			return hand
		release handshake	

Table 2. The practical example of Scenario #2.

6. Conclusion

In this paper, we have shown a real-time 3D interaction system based on a human motion analysis without special marker-sensors, and based on human motion synthesis. We have adopted deformed blob model to catch exact skin blob region. We have also introduced inverse kinematics to realize human body motion synthesis from a limited number of perceptual cues. We have applied these basic analysis and synthesis techniques to smart interaction which works in real-time and online. Assuming that virtual objects existing in virtual environments can afford human figure actions, high-level action recognition can be realized.

As future works, we will plan to extend our interaction system to physically controlled virtual environments, and to adapt it to wider virtual environment by employing fulll body motion. Events of the other real objects must be also taken in virtual environments.

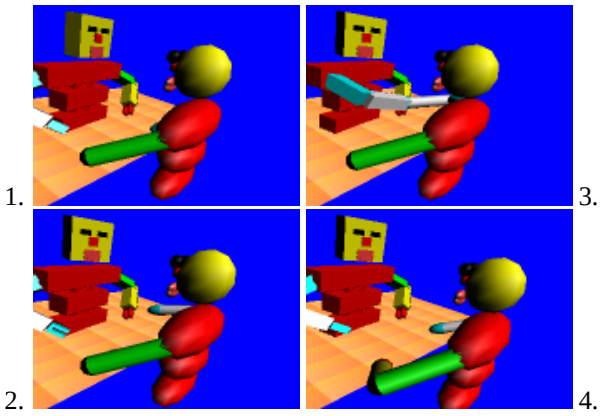

Figure 5. Scenario#2: Avatar shakes hand with virtual robot.

References

[1] C.Wren, A.Azarbayejani, T.Darrell, A.Pentland, "Pfinder: Real-Time Tracking of the Human Body", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.19, No.7, pp.780–785, 1997.

[2] A.Witkin, M.Gleicher, W.Welch, "Interactive Dynamics", *in ACM SIGGRAPH*, Vol.24, no.2, pp.11–21, 1990.

[3] C.Bregler, "Learning and Recognizing Human Dynamics in Video Sequences", *in Computer Vision and Pattern Recognition*, pp.568–574, 1997.

[4] M.Etoh, Y.Shirai, "Segmentation and 2D Motion Estimation by Regio¡n Fragments", *in International Conference on Computer Vision*, pp.192–199, 1993.

[5] C.Wren, A.Pentland, "Understanding Purposeful Human Motion", *in Fourth IEEE International conference on Automatic Face and Gesture Recognition*, 2000.

[6] J.Kuffer Jr., "Autonomous Agents for Real-time Animation", *PhD thesis Stanford University*, 1999.

[7] Y.Koga, "Planing Motions with Intentions", *Proc. of SIGGRAPH'94*, pp.24–29, 1994.

[8] Satoshi Yonemoto, Daisaku Arita and Rin-ichiro Taniguchi, Real-Time Human Motion Analysis and IK-based Human Figure Control, in Proceedings of Workshop on Human Motion (HUMO2000), pp.149–154, 2000.