# Vision-based Latency-free Real-time Human Motion Sensing

Yoshimoto, Hiromasa
Department of Intelligent Systems, Kyushu University

Date, Naoto
Department of Intelligent Systems, Kyushu University

Arita, Daisaku
Department of Intelligent Systems, Kyushu University

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



出版情報：Proceedings of the 10th Korea-Japan Joint Workshop on Frontiers of Computer Vision, pp.244-249, 2004-02
バージョン：
権利関係：

# Vision-based Latency-free Real-time Human Motion Sensing

Hiromasa YOSHIMOTO, Naoto DATE, Daisaku ARITA and
Rin-ichiro TANIGUCHI

Laboratory for Image and Media Understanding
Department of Intelligent Systems, Kyushu University
http://limu.is.kyushu-u.ac.jp

**Abstract** In this paper, we discuss a latency-free vision-based real-time human motion sensing system. Vision-based human motion sensing has a merit that it does not impose any physical restrictions on humans, which provides a natural way of measuring human motion. However, there are several issues to be solved. A topic discussed here is a computational aspect of vision-based human motion sensing. Since most of vision algorithms applied for human motion sensing are time consuming, latency, or delay, of the system, becomes large. This delay often becomes crucial especially to real time interactive applications.

To solve this problem we introduce confidence-driven architecture, which has a mechanism of prediction, and which can make trade-off between latency and accuracy. In some cases, it is better to make the latency small even if the accuracy of the analysis becomes low, and, in other cases, it is required to get accurate results even if it is time consuming. There is a difficult trade-off between the latency and the accuracy, and it is required to control the trade-off smoothly without rebuilding the system. In the confidence-driven architecture, the trade-off can be controlled by specifying a generalized parameter called *confidence*, which indicates how accurate the analysis should be. Applying this architecture to vision-based human motion sensing, we can develop a latency-free human motion sensing system.

## 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 methods for human motion sensing, many motion capture devices with special markers or sensor attachments have been employed, which often impose physical restrictions on humans, and which are not comfortable for their users. On the other hand, recently, fully image-feature-based, or vision-based, motion capturing systems which do not impose such restrictions have been developed as a computer vision application[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. Problems of the vision-based approach are summarized as follows.

- Image features which can be robustly detected are limited, and, therefore, 3D human postures should be estimated from the limited cues.
- A human body makes various poses, which often cause serious self-occlusion.
- Vision algorithms are often time consuming, and, therefore, to realize real time processing an efficient computing mechanism is required.

To solve the problems, an approach of multi-view image analysis, which is implemented on a parallel computer such as PC-cluster to realize real time processing has been adopted. It is a promising approach but still has problems to be solved. One of the biggest problems is a problem of latency. Vision-based systems usually analyze images frame by frame and, in addition, usually a pipeline structure is employed to gain high throughput, which make the latency three times or more of the frame rate, i.e., 100 $\sim$ 200 msec. Such latency is often crucial especially for online interactive systems and it is required to reduce the latency as much as possible. In this paper, we introduce a latency reduction mechanism, confidence-driven memory (CDM), to vision-based real-time human motion sensing, and show its effectiveness through several experiments.

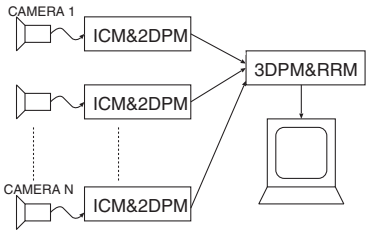


Fig. 1: Overview of vision-based human motion sensing

# 2 Vision-based Real-time Human Motion Sensing

## 2.1 System Overview

To solve the self-occlusion problem we have adopted a multi-view approach and the basic algorithm flow of our real-time motion capturing is as follows:

1. Perception (Detection of cues)
   - Silhouette detection, skin color blob detection, and 2-D feature extraction (2DPM).
   - Calculation of 3-D positions of features using multi-view fusion (3DPM).
2. Human Motion Synthesis
   - Generation of human figure full-body motion and rendering in the virtual space including calculation of the interaction (RRM).

Details of the algorithm is shown in a later section. To realize real-time processing, we have implemented the algorithm on a PC cluster, in which a parallel-pipeline structure is employed to describe a parallel version of the algorithm (see Fig. 1). The latency introduced into a current implementation is about 400 msec because the frame rate of our IEEE-1394 cameras is restricted to 15 fps.

## 2.2 Algorithms for Human Motion Sensing

### 2.2.1 2D Image Processing

To analyze human motion, image features such as blobs (coherent region)[1][3] or silhouette contours[4] are usually employed. The problem is that a limited number of perceptual cues of a human body can be detected robustly from images. Here, we have employed skin-color blobs and sock-color blobs are used as major perceptual cues, which correspond to a head[1], hands and feet. The blobs are detected based on color similarity analysis[5] after a silhouette of a human body is extracted by background subtraction. The centroids of the blobs are used to calculate the 3D positions of those cues. The calculation is based on binocular stereo, which will be presented in the next section.

[1]Strictly speaking, it corresponds to a face.

### 2.2.2 Estimation 3D Positions of Feature Points

In general, when projected 2D image points of a 3D point are found in two image planes, the 3D position of the point is calculated from the 2D positions by calculating a crossing point of the two lines of sight, each of which passes through a camera center and a projected 2D point. Here, we have more than two cameras to solve self-occlusion of a human body. Therefore, essentially, there exist multiple pairs of the lines of sight and multiple 3D position candidates for one skin-color blob. Therefore, after obtaining all 3D position candidates of skin-color blobs (and sock-color blobs), we classify them into 5 clusters of feature points: a head, a right hand, a left hand, a right foot, a left foot. The clustering is done based on the distance to the feature points detected in the previous frame[2].

In each cluster, we estimate the feature point position. Actually, as the feature point position, we take the average position of the 3D position candidates after a dense part of the cluster is selected.

To estimate the human posture from the above five feature points, we need the positions of shoulders and hips. Although it is not easy to detect the shoulders and the hips in the captured images directly, we estimate those positions from the torso position, which is estimated as the centroid of the body region, referring to our human figure model having predefined positions of shoulders and hips.

### 2.2.3 3D Human Pose Estimation

Since information acquired in the perception process is just 3-D positions of a torso, a head, hands and feet of a human body, we have to estimate the body posture from these cues, the number of which is less than the degree of freedom of the body. Actually, we have to estimate 3D positions of elbows and knees. Based on a very simple human body model, which has 14 parts and 23 DOFs (see Fig. 2), we have implemented a simple algorithm called *search by reverse projection* (SRP).

Basic idea of estimation of elbow and knee positions based on *search by reverse projection* is as follows. After we estimate the positions of a hand

[2]We assume an initial pose is a pose with arms and legs wide open.

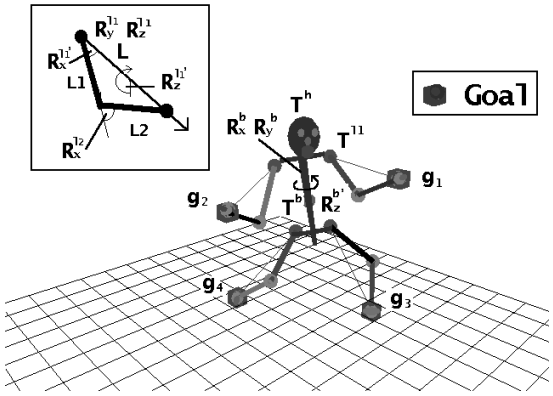


Fig. 2: Human body model

and a shoulder for one arm, if we know the lengths of its upper arm and forearm, the position of its elbow is restricted on a circle in 3D space shown in Fig. 3, and we only have to search for the elbow position on the circle. When we select a point on the circle and map it onto an image plane of each view, the point should be included in every silhouette region if it is an elbow point (see Fig. 4). In other words, points which satisfy the above condition might coincide with the elbow position with high probability. Fig. 5 shows an online demo shot of our real time human motion sensing.

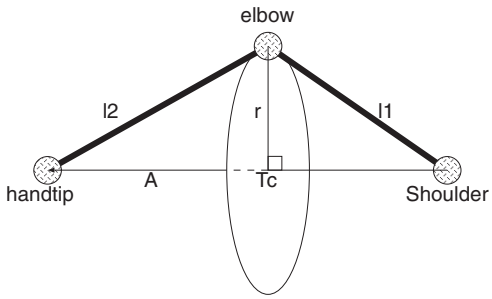


Fig. 3: Positional relation among shoulder, elbow and hand

## 3 Latency Reduction by Confidence Driven Memory

### 3.1 Latency vs Accuracy

It is important for vision systems to achieve data stream interpretation accurately without latency. However, there is a trade-off: a long computation time is required to get accurate information from the input-images; on the other hand, fast computation is desired to get information as soon as possible. The trade-off is influenced by many factors: whether a user assigns high priority to higher accuracy or to lower latency; how many computational resources such as computation time and bandwidth of the network are available. We have to make difficult trade-off between the accuracy and the latency in dynamically so that the quality of the system's output keeps the best one.

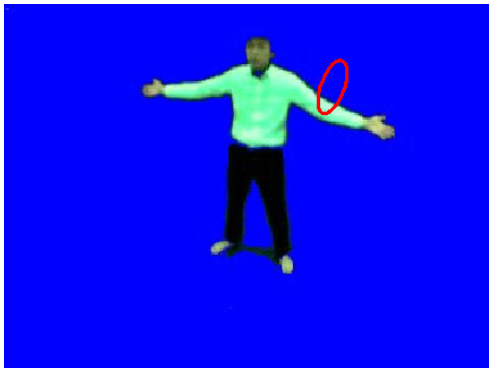

Fig. 4: Reversely projected trajectory of the elbow position

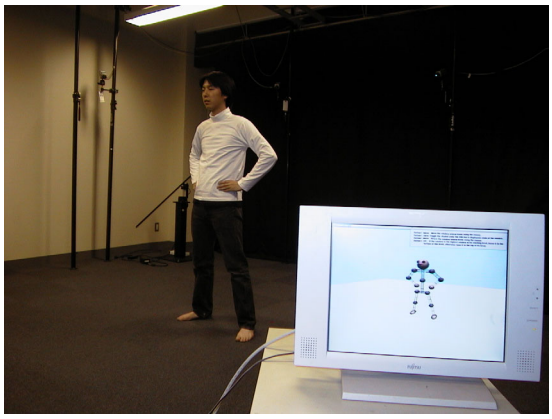

Fig. 5: Online demo shot

In many vision-based applications, a lot of implicit trade-offs are considered. For example, suppose a case that we use a template matching method to find a target. When the template matching is used, as the resolution of the template images increases, the accuracy of the search results becomes higher but the computation time becomes longer. Therefore, we have to make trade-off to determine the resolution of the templates. In another aspect, when we deal with real time image sequences, we can often assume that the target moves continuously, or that the target position is a continuous variable. Then, reduction of the redundancy which the variable implicitly has can lead to decrease the

computation time. An estimator such as Kalman filter is often efficient to increases the throughput. The estimator provides results faster than the template matching but the accuracy becomes lower. Trade-off here is to determine the ratio of the use of the estimator to that of the template matching.

### 3.2 Confidence-driven Architecture

In the above example, it is important that the template matching and the estimator have the same characteristic, that is, as the time increases the accuracy of the output becomes higher monotonically. This monotonic characteristic is a basis of imprecise computation model[2]. If the relation between allowable computation time and the accuracy are known, we can control the trade-off according to the relation. Since, usually, the relation is affected by several factors and is not known precisely in advance, instead we hypothesize a simple relation such as linearity between the time and the accuracy. It is important that there is no generic way to calculate absolute accuracies of vision algorithms. However according to characteristics of the algorithms, we can estimate relative accuracies heuristically. Therefore we have introduced a term "confidence", ranging from 0 to 1. The confidence is an abstraction of the accuracy. The higher the confidence is, the more accurate the estimation is.

We formalize various trade-off problems as a simple producer-consumer problem. To clarify the description, we have introduced several functions. Target information is denoted as a time function $x(t)$. $c(t)$ is the confidence of $x(t)$. In the system, the producer computes a value of $x(t_n)$ from an input stream. The heavier computation method the producer uses, the higher confidence $c(t_n)$ of $x(t_n)$ becomes. The producer provides a history $h$ of the stream, a set of $\{t_n, x(t_n), c(t_n)\}$. $F_x(h,t)$ is an estimator of $x(t)$, which refers to the history $h$. When the history has not enough information to estimate $x(t)$, the estimation of $F_x(h,t)$ has less confidence. We have also introduced $F_c(h,t)$, which is an estimator of $c(t)$. In general, the confidence is getting lower as the estimator predicts data in more distant future (see Fig. 6). Fig. 7 shows the basic idea to reduce the latency.

The key idea of our architecture is that it is not the data but the confidence which the consumer waits for, although the architecture is similar to Dynamic Memory[6] and dead reckoning[7] in the sense that an estimator is adopted. The data $x(t)$ is estimated by $F_x(h,t)$ in any time, and the consumer waits until the $F_c(h,t)$ becomes high enough. We call this scheme as "confidence-driven." Using the confidence-driven scheme, the trade-off between the accuracy and the latency is converted to the trade-off between the confidence and the latency.

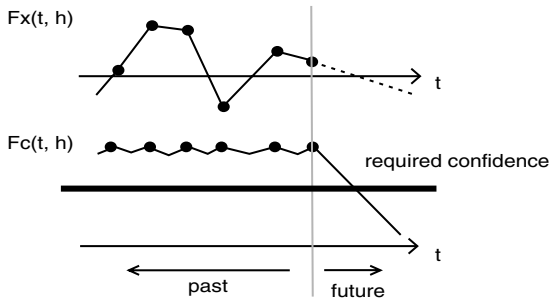


Fig. 6: Characteristics of estimator and confidence

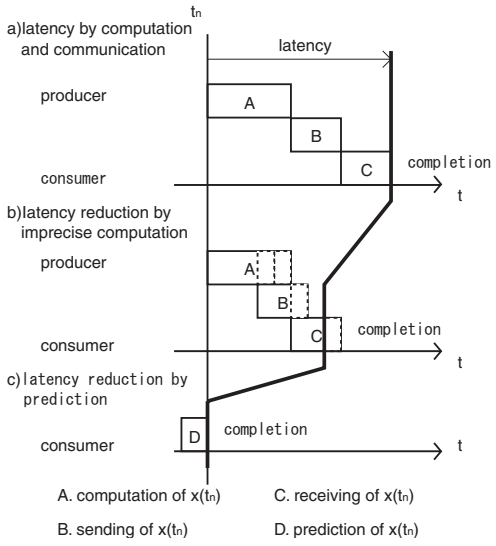


Fig. 7: Basic idea of latency reduction

### 3.3 Implementation of Confidence-driven Architecture

Confidence-driven memory is an implementation of confidence-driven architecture, and is a shared memory based on the confidence-driven scheme. Producer and consumer are implemented as threads and can communicate over the the confidence-driven memory (CDM) with three operations, read, write, poll referring to two user-defined functions $F_x(h,t)$ and $F_c(h,t)$.

- `read(m, t, C`$_r$`, deadline)`
  it is an operation that reads from a confidence-driven memory $m$ specifying a required time $t$ and confidence $C_r$. If acquired confidence $c(t)$ is less than the required confidence $C_r$, wait until one of the following conditions is satisfied:

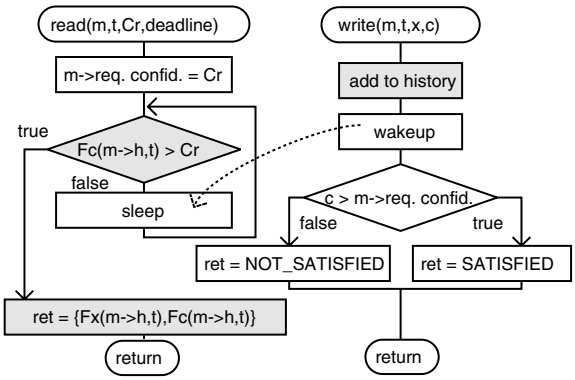


Fig. 8: Relation between read and write operations

$c(t)$ becomes greater than $C_r$; or *deadline* is passed. If the deadline is passed, an estimation result $x(t)$ and acquired confidence $c(t)$ at this time are returned.

- `write(m, t, x, c)`
  it appends a tuple of $\{t, x(t), c(t)\}$ to a history $h$ held in the confidence-driven memory $m$.
- `pole(C`$_r$`)`
  it is an operation that waits until multiple confidence-driven memories have confidence higher than $C_r$.

Fig. 8 shows the relationship between a read operation and a write operation. The consumer task executes a read operation, specifying required confidence, and suspends until the confidence to be high enough. When the consumer is suspended, the producer thread computes data to make the confidence bigger. Thus each threads synchronized according to the confidence. In general, there is a certain overhead for the threads to share the data completely. However, in the confidence-driven scheme, each thread shares estimated data without the overhead while the confidence $F_c(h,t)$ is higher than the confidence the consumer requires.

### 3.4 Application to Real-time Motion Sensing

Our real-time motion sensing described in section 2 is based on the observation of blob positions. Since we can assume continuity of those values, we can apply the confidence driven memory architecture to communication of the blob positions in the system and we expect the reduction of delay due to the confidence-driven architecture. In addition, the confidence-driven memory makes asynchronous execution of read and write operations, with which read operations can be issued more frequently than write operations are issued. Thus, rendering of a CG avatar in the system (see Fig. fig:demo) can be done frequently, which makes the avatar's motion smooth and natural. The new software structure of motion sensing is illustrated in Fig. 9. It is easily achieved by inserting CDMs in communication paths of the previous system.

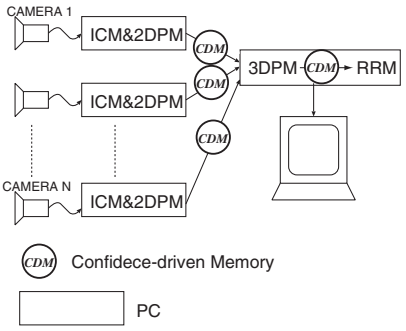


Fig. 9: Real-time human motion sensing implemented with CDM

## 4 Experiment and Evaluation

Here, we evaluate the effectiveness of the confidence-driven architecture when it is applied to vision-based human motion sensing. For estimating time-varying variables in the system such as the positions of 3D blobs, a linear prediction is used. On the other hand, the characteristic of "confidence", $F_c(h,t)$, is defined as follows in the system.

$$F_c(h,t) = \begin{cases} 1 & t < T \\ (1 - \frac{|t-T|}{400})c(T) & \\ & T \le t < T + 400 \\ 0 & T + 400 \le t \end{cases}$$

Table 1 shows the relation between the required confidence and the error of 3D blob position estimation. The error is the difference between the 3D blob position estimated by the vision process with CDM and the one measured by a 3D position sensor. The result shows that the higher the required confidence becomes, the higher the accuracy of estimation becomes.

Table 2 shows the relation among required confidence, the latency and the throughput, or the frequency of CG avatar rendering. The lower the required confidence becomes, the smaller the latency becomes. In other words, low latency can be achieved by sacrificing the accuracy. However, it achieves relatively high accuracy due to an estimation mechanism built in CDM. Since the confidence-driven memory makes asynchronous execution of

| Required Confidence | Error (mm) Min. | Max. | Avg. |
|---|---|---|---|
| 0.3 | 10.2 | 797.6 | 150.1 |
| 0.5 | 21.2 | 777.5 | 144.6 |
| 0.7 | 16.3 | 718.1 | 149.5 |
| 0.8 | 25.5 | 609.7 | 123.1 |
| N/A | 20.2 | 195.7 | 89.2 |

Table 1: Required confidence and error

| Required Confidence | Latency (msec) | Throughput (fps) |
|---|---|---|
| 0.3 | 183 | 55 |
| 0.5 | 200 | 45 |
| 0.7 | 250 | 31 |
| 0.8 | 302 | 26 |
| N/A | 428 | 15 |

Table 2: Required confidence vs latency and throughput

read and write operations, rendering of a CG avatar in the system is done frequently, which makes the avatar's motion smooth and natural, and the lower the required confidence becomes, the more frequently the avatar is rendered. It is important to notice that by changing the required confidence, we can make the trade-off between the latency and the precision dynamically and smoothly.

## 5 Conclusion

In this paper, we have discussed a latency-free vision-based real-time human motion sensing system. Vision-based human motion sensing has a merit that it does not impose any physical restrictions on humans, which provides a natural way of measuring human motion. However, its long latency due to time consuming vision algorithms becomes a problem especially for real time interactive applications. To solve this problem, we have introduced the confidence-driven architecture, which has a mechanism of prediction, and which can smoothly make the trade-off between the latency and the accuracy without rebuilding the system.

Applying the confidence-driven architecture to vision-based real time human motion sensing, we can acquire a latency-free real time motion capture system without a large decrease of the accuracy, due to a prediction mechanism built in the architecture. This is shown in experimental results stated here. For future works, prediction mechanism should be investigated to acquire accurate estimation, which makes the confidence-driven architecture more efficient. Since good prediction mechanism depends on applications, we are going to apply our idea to many vision-based applications. Also, we are going to improve vision algorithms for human motion sensing to make the system more accurate and more robust.

## 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] J. Liu, et al, "Imprecise Computation," *Proc. IEEE*, Vol.82, No.1, pp.83-94, 1994.

[3] M.Etoh, Y.Shirai, "Segmentation and 2D Motion Estimation by Region Fragments", *International Conference on Computer Vision*, pp.192-199, 1993.

[4] K.Takahashi, T.Sakaguchi, J.Ohya, "Remarks on a Real-Time 3D Human Body Posture Estimation Method using Trinocular Images," *International Conference on Pattern Recognition*, Vol.4, pp.693-697, 2000.

[5] S.Yonemoto, D.Arita and R.Taniguchi "Real-Time Visually Guided Human Figure Control Using IK-based Motion Synthesis," Proc. 5th Workshop on the Application of Computer Vision, pp.194-200, 2000.

[6] T. Matsuyama, et al, "Dynamic Memory: Architecture for Real Time Integration of Visual Perception, Camera Action, and Network Communication," *Proc. of CVPR*, pp.728-735, 2000.

[7] S. Singhal and M. Zyda. *Networked Virtual Environments*, Addison Wesley, 1999.