Multisensor Integrated Prediction for Virtual Reality

Abstract

Unconstrained measurement of human head motion is essential for HMDs (head-mounted displays) to be really interactive. Pohrornus sensors developed for that purpose have deficiencies of critical latency and low sampling rates. Adding to this, a delay for rendering virtual scenes is inevitable. This paper proposes methods that compensate the latency and raise the effective sampling rate by integrating Pohrornus and gyro sensors. The adoption of quaternion representation enables us to avoid singularities and the complicated boundary process of rotational motion. The ability of proposed methods under various rendering delays was evaluated in the respect of RMS error and a new correction technique, which enables us to check the latency and fidelity of a magnetic tracker, and to assess the environment where the magnetic tracker is used. The real-time implementation of our simpler method on personal computers is also reported in detail.

1 Introduction

The synchronization of a user’s motions in actual space and virtual space is significant in virtual reality (VR), because under the situation, where the time lag between these spaces is greater than 100 ms, users tend to feel motion sickness and so perceive such a virtual environment as less interactive.

Motion sensors used in VR systems are classified into two groups. One is the mechanical link-type sensors like BOOM, and the other is the unconstrained sensors like Pohrornus Tracker. The latter has the advantage of unconstrainedness and wide motion range, but has deficiencies of low sampling rates (−60 Hz) and critical latency (−100 ms). Although Pohrornus “Fastrack” is improved in the respect of sampling rate (−120 Hz), communication delay (−40 ms with RS232C), and an ad-hoc linear prediction filter, it is still insufficient for interactivity in the virtual environment. Most VR systems use raw output of their motion sensor in spite of the decrease in interactivity by this latency, or try ad-hoc filtering in order to compromise for the latency such as linear extrapolation from back smoothed measurement and crude estimates of instantaneous differential of the user’s motion.

Friedmann, Steurer, and Pentland (1992) pointed out that by such an ad-hoc approach, the user’s quick motion causes poor prediction that overshoots the actual motion and forces users to make slow deliberate motions. They solved this problem by translational motion by means of the prediction based on the optimal linear estimation theory. However, they ignored rotational motion, which is far more critical for HMD. Liang, Shaw, and Green (1990) proposed...
to adopt the equivalent angle-axis representation for rotational motion, but they deal a single rotation by decomposing into four independent system models, and applied Kalman filter theory. The derivation that appeared in their state vector was something other than angular velocity, and their prediction had no kinematic ground.

Our main idea is to combine the sensor of angular velocity to compensate for the latency of measurement of the absolute posture of a human head. In order to integrate two different types of sensors, we derived a kinematically consistent system model. The methods we proposed can not only compensate for the latency, but also raise the sampling rate through integration of multisensor information. The integrator algorithm used maximum-likelihood estimation, which is also the basis of Kalman filter theory.

We proposed multisensor integration approach in Section 2. We used quaternion representation to avoid singularity and complicated boundary processing. The parameters for the proposed methods were decided based on careful inspections in Section 4. In order to check the performance of these methods in detail, we proposed the new powerful evaluation method of motion-tracking sensors based on correlation functions in Section 5. The outputs of these methods with various rendering delay were compared offline and checked by RMS error and the correlational method in Section 6. In Section 7 we also reported real-time implementation of the simpler method into ordinary personal computers. The validity of the sensing system combined with a see-through HMD was checked by video recording.

2 Quaternion

2.1 Quaternion representation

We adopt quaternion (Goldstein, 1980) instead of Euler angles as the representation of orientation. The reasons are

1) when a human body rotates over 2π radians, the

Euler angle exceeds the measure range of the mag-

netic tracker, \( -\pi \leq \alpha \leq +\pi \) and the management

around the boundary is complex.

2) the system model of human rotational motion de-

scribed by Euler angles has singular points.

Let \( p \) be a vector in 3-D space. If we rotate the coordi-

cate by ZXY Euler angle \( \gamma, \beta, \alpha \), the new description of \( u \) satisfies

\[
\mathbf{e} = R_z(\gamma)R_y(\beta)R_x(\alpha)\mathbf{e}.
\]

(1)

It is known that rotation is also represented by com-

plex \( 2 \times 2 \) matrix in general as \( 3 \times 3 \) rotational matrix

(Goldstein, 1980). This matrix consists of four real

numbers \( g, g, g, g \).

\[
Q = \begin{bmatrix} g + ig \alpha & g - ig \beta & -ig \gamma \\
-g + ig \beta & g + ig \alpha & ig \gamma \\
ig + ig \gamma & -ig \gamma & g - ig \alpha
\end{bmatrix}
\]

(2)

\[
g^2 + g^2 + g^2 + g^2 = 1.
\]

(3)

We call \( g \) the quaternion. If we rotate the coordinate

by ZXY Euler angle \( \gamma, \beta, \alpha \), each rotation is represented by these \( 2 \times 2 \) matrices.

\[
Q_x(\gamma) = \begin{bmatrix} e^{i\gamma/2} & 0 \\
0 & e^{-i\gamma/2}
\end{bmatrix}
\]

(4)

\[
Q_y(\beta) = \begin{bmatrix} \cos \beta/2 & \sin \beta/2 \\
-\sin \beta/2 & \cos \beta/2
\end{bmatrix}
\]

(5)

\[
Q_z(\alpha) = \begin{bmatrix} \cos \alpha/2 & i \sin \alpha/2 \\
-i \sin \alpha/2 & \cos \alpha/2
\end{bmatrix}
\]

(6)

Consider \( \mathbf{v} \) below instead of the vector \( \mathbf{p} \). Then the rota-

tion by Euler angle can be described in another form.

\[
\mathbf{v}' = \mathbf{Q} \mathbf{v}.
\]

(7)

\[
Q = Q_z(\alpha)Q_y(\beta)Q_x(\gamma)
\]

(8)

\[
\mathbf{v}' = Q^\dagger \mathbf{v} Q
\]

(9)

The quaternion of \( Q \) is given by

\[
g = \cos \frac{\gamma}{2} \cos \frac{\beta}{2} \cos \frac{\alpha}{2} + i \sin \frac{\gamma}{2} \sin \frac{\beta}{2} \sin \frac{\alpha}{2},
\]

(10)
\[ g_1 = \cos \frac{\gamma}{2} \cos \frac{\beta}{2} \cos \frac{\alpha}{2} - \sin \frac{\gamma}{2} \sin \frac{\beta}{2} \cos \frac{\alpha}{2} \] (11)

\[ g_2 = \cos \frac{\gamma}{2} \sin \frac{\beta}{2} \cos \frac{\alpha}{2} + \sin \frac{\gamma}{2} \cos \frac{\beta}{2} \sin \frac{\alpha}{2} \] (12)

\[ g_3 = \sin \frac{\gamma}{2} \cos \frac{\beta}{2} \sin \frac{\alpha}{2} - \cos \frac{\gamma}{2} \sin \frac{\beta}{2} \sin \frac{\alpha}{2} \] (13)

Let \( \omega = (\omega_x, \omega_y, \omega_z) \) rad/sec be angular velocity in body coordinate. Let \( q_0 \) be the quaternion at time \( t \), \( q_1 \) be the quaternion at time \( t + dt \), and \( dQ \) be the 2 \( \times \) 2 matrix corresponding to the rotation between \( t \) and \( t + dt \). \( dQ \) is given in terms of \( \omega \) by

\[
dQ = \begin{pmatrix}
1 + i \frac{\omega_x}{2} dt & i \frac{\omega_y}{2} dt + i \frac{\omega_z}{2} dt \\
-i \frac{\omega_y}{2} dt + i \frac{\omega_z}{2} dt & 1 - i \frac{\omega_x}{2} dt
\end{pmatrix}
\] (14)

and \( dQ \) is given in terms of \( p_0 \) by

\[
dQ = \begin{pmatrix}
p_0 + i p_x & p_y + i p_z \\
-p_y + i p_x & p_0 - i p_z
\end{pmatrix}
\] (15)

Define \( F \) as below.

\[
F = \begin{pmatrix}
-g_1 & -g_2 & -g_3 \\
g_2 & -g_1 & -g_3 \\
g_3 & g_2 & -g_1
\end{pmatrix}
\] (16)

Combining (14) and (15), and using (16), we obtain the process dynamics expressed by quaternion and angular velocity in body coordinate.

\[
(p_0, p_1, p_2)^T = (g_0, g_1, g_2)^T + F \omega dt / 2
\] (17)

If we adopt Euler angle representation, the process dynamics is expressed as

\[
r = (\gamma \, \beta \, \alpha)^T
\] (18)

\[
r' = r + \begin{pmatrix}
\cos \alpha / \cos \beta & \sin \alpha / \cos \beta & 0 \\
-\sin \alpha & \cos \alpha & 0
\end{pmatrix} d\theta dt
\]

\[
\begin{pmatrix}
\cos \alpha \tan \beta & \tan \alpha \tan \beta & 0
\end{pmatrix}
\] (19)

Apparently, this equation has singular points at

\[ \beta = (\frac{\pi}{2} + \pi)n. \] (20)

The displacement of Euler angle becomes infinite though the angular velocity is finite. We can easily avoid this singularity problem by the quaternion representation.

3 Multisensor Integration

Prediction is often used for compensation of delay; Kalman filter theory is well known as the method of designing an optimal prediction filter taking into account the complex motions of target system and measurement error. This has been applied to the delay of the magnetic tracker (Friedmann, Starner, & Pantland, 1992; Liang, Shaw, & Green, 1990). Prediction of physical quantity needs some kind of its derivative in state vector. What was measured in this research was only position or quaternion; its derivative was not measured. We expected the improvement of estimates by adding the measurement of these derivatives.

This demands the kinematically consistent system model. As for rotational motion, the relation between Euler angle or quaternion and angular velocity is needed. For that purpose, we derived the approximated linear system model, which describes the relation between quaternion and angular velocity explicitly.

Kalman filters have only one measurement model, which means that the filter enables the integration of simultaneous measurements of various sensors, but that the estimate rate is bound to the slowest sensor at most if we integrate sensors of various sampling rates by Kalman filter. In order to improve the estimate rate, the information of the quick sensor should be reflected to the estimates rationally in some way.

In our problem, the slowest sensor is the magnetic tracker, and the quick sensor is the gyro sensor. When the measurements of both quaternion and angular velocity are available, we propose the optimal prediction by Kalman filter. As for how to reflect measurements of gyro sensor to the estimates, we first considered the
maximum likelihood estimation of the state vector based on the measurement of angular velocity. We assumed that the system model and process noise were well known. This is the first method we propose. This method gives us the optimal prediction; however, it was expected that the computation load of maximum likelihood estimation is a bit heavy for the target computer of real-time implementation. So, we also considered the extrapolation by integrating the displacement of quaternion calculated from the measured angular velocity. This is the second method we propose.

Let $H$ be the measurement matrix corresponding to simultaneous measurements of both the Polhemus and the gyro sensor, and let $H'$ be the measurement matrix corresponding to the measurement of the gyro sensor only.

$w$ = process noise
$Q$ = covariance matrix of $w$
$H$ = measurement matrix of both Polhemus and gyro
$\sigma$ = measurement noise by $H$
$R$ = covariance matrix of $\sigma$
$H'$ = measurement matrix of gyro
$R'$ = covariance matrix of $\sigma'$

\[
F = \begin{bmatrix} 1 & dt \hline 0 & 1 \end{bmatrix}
\]
(21)

\[
x_{k+1} = F x_k + w_k
\]
(22)

\[
w_k \approx Q \Delta t
\]
(23)

\[
H = \begin{bmatrix} I & -T \hline 0 & I \end{bmatrix}
\]
(24)

\[
P_k = \begin{bmatrix} 0 & I \end{bmatrix}
\]
(25)

\[
\gamma_k^l = F \gamma_k + \nu_k^l
\]
(26)

\[
\gamma_k = H \gamma_k + \sigma_k
\]
(27)

\[
\gamma_k \approx R \gamma_k
\]
(28)

\[
\nu_k \approx R \nu_k
\]
(29)

### 3.1 Algorithm of Method I

When both the Polhemus and gyro sensor are available, the measurement model is given by equation (26). When only the gyro sensor is available, the measurement model is given by equation (27). The algorithm (Kimura & Sachi, 1993) consists of two procedures. Procedure 1 is executed when both the Polhemus and gyro measurements are available. Procedure 2 is executed when only the gyro measurement is available.

#### Procedure 1

\[
x_{k+1,1} = F x_k + w_k
\]
(30)

\[
P_{k+1,1} = F P_k F^T + Q Q^T
\]
(31)

\[
x_{k+1} = x_{k+1,1} + K_k (\gamma_k - H P_j x_{k+1,1})
\]
(32)

\[
P_k = P_{k+1,1} - K_k H P_k x_{k+1,1}
\]
(33)

\[
K_k = P_{k+1,1} H^T (H P_k x_{k+1,1} H^T + R')^{-1}
\]
(34)

#### Procedure 2

\[
x_{k+1,2} = F x_k + w_k
\]
(35)

\[
P_{k+1,2} = F P_k F^T + Q Q^T
\]
(36)

\[
x_{k+1} = x_{k+1,2} + K_k (\gamma_k - H P_k x_{k+1,2})
\]
(37)

\[
P_k = P_{k+1,2} - K_k H P_k x_{k+1,2}
\]
(38)

\[
K_k = P_{k+1,2} H^T (H P_k x_{k+1,2} H^T + R')^{-1}
\]
(39)

The algorithm used is procedures 1 and 3's the same is that of a discrete Kalman filter (Gelb, 1974).

### 3.2 Algorithm of Method II

The algorithm is divided into two procedures. Procedure 1 is for the extrapolation of the state vector based on the measurement of angular velocity only. Procedure 2 is for the optimal prediction based on the simultaneous measurement of both quaternion and angular velocity. Note that the sampling rate of quaterion of Method II is half that of Method I.
Procedure 1

Substitute angular velocity in state vector \( \omega \), of mean angular velocity within previous 30 ms

\[
\bar{\omega} = \left( \omega_x, \omega_y, \omega_z \right)
\]  

and update as below.

\[
x_{a+1/a} = \begin{bmatrix} x_{a+1/a} \end{bmatrix} = \begin{bmatrix} \omega_{a+1} + dt \cdot F \omega_{a} \end{bmatrix}
\]

\[
P_{a+1/a} = FP_{a+1/a} + GG^T
\]  

Procedure 2

\[
x_{a+1/a} = Fx_{a+1/a}
\]

\[
P_{a+1/a} = FP_{a+1/a}F^T + GG^T
\]

\[
x_{a+1/a} = x_{a+1/a} + K_e \left[ y_e - HP_{a+1/a} \right]
\]

\[
P_{e+1/a} = P_{e+1/a} - K_eHP_{a+1/a}
\]

\[
K_e = \begin{bmatrix} K_{e1} & \ldots & K_{en} \end{bmatrix}
\]

Procedure 2 is the same as the algorithm of a discrete Kalman filter (Gelb, 1974).

Figure 1 is the timing chart of available sensors and procedures. We measured the orientation at 25 Hz and the angular velocity at 100 Hz. The output was obtained at 100 Hz.

4 Approximation of Parameter

4.1 Rendering Delay

Not only the delay of sensors but the delay by rendering must be considered in our HMD problem. Assume that we got the measurement of gyro at \( t = 0 \). Using the definitions below, we see the relation of observed value and estimated value. Note that the measured value of \( X \) is expressed as \( X^{0} \).

\[
f_{\text{gyro}} = \text{delay of magnetic tracker}
\]

\[
f_{\text{render}} = \text{delay by rendering}
\]

\[
(q_0, q_0) = \text{state vector at } t = t_{\text{gyro}}
\]

\[
(q_0, w) = \text{state vector at } t = 0
\]

\[
(q_0, w) = \text{state vector at } t_{\text{render}}
\]

\[
t = t_{\text{gyro}}
\]

\[
t = 0
\]

\[
t = t_{\text{render}}
\]

Before rendering the virtual scene, \( q^{0} \) and \( w^{0} \) must be known and are derived as below by using \( dq \) and \( dw \), which are statistical variables.

\[
T = f_{\text{gyro}} + f_{\text{render}}
\]

\[
q_0 = q_0 + 7T \omega_0 + dq
\]

\[
q^{0} = q^{0} + 7T \omega^{0} + dq
\]

\[
\omega_0 = \omega_0 + dw
\]

\[
\omega^{0} = \omega^{0} + dw
\]

In the state vector \( (q, w)^T \), there is no derivative term of angular velocity \( w \), so the prediction like equation (50) is impossible. Instead, we decided to approximate \( \omega^{0} \) by \( \omega^{1} \) and to consider that the measurement error becomes a bit larger, because the output of the gyro sensor became noisy (Figure 2), and the matching between real motion and the measurement of the gyro is not so affected by the slight shift of approximation \( \omega^{0} \) by \( \omega^{1} \).

4.2 Process Noise

Covariance matrices of process noise and measurement noise were given by \( Q \) and \( R \). We assume process noise and measurement noise to be independent.
Let $I_3$, $I_1$ be $3 \times 3$ and $4 \times 4$ unit matrices.

\[ Q = \begin{pmatrix} r^2 I_3 & 0 \\ 0 & r^4 I_4 \end{pmatrix} \] (52)

\[ R = \begin{pmatrix} r^2 I_3 & 0 \\ 0 & r^4 I_4 \end{pmatrix} \] (53)

\[ R_c = (r^2 I_3) \] (54)

$u_w$, $v_w$ are determined as the standard deviation of random error when the output of the high-precision link-type sensor (Tschi, Ari, & Macda, 1990) sampled at 9 Hz was fitted to system model equation (30).

\[ x_1 = 0.0212 \quad u_2 = 0.10[\text{rad/s}] \]

4.3 Measurement Noise

The measurement error cannot be determined as easily as process noise, because we must take into account the Polliceps and rendering delay. The covariance matrices we should consider are $(q_3 - q_1)(q_3 - q_1)^T$ and $(w_3 - w_1)(w_3 - w_1)^T$. Assume $q_0$, $dq$, $w_1 - w_0$ is statistically independent.

\[ \begin{align*}
(q_3 - q_0)(q_3 - q_0)^T \\
= (q_3 - q_0)(q_3 - q_0)^T + w_3 dq_3 \\
+ T^2 (w_3 - w_1)(w_3 - w_1)^T/F^2
\end{align*} \] (55)

From equation (3), we approximate

\[ F^2 P = I. \] (56)

\[ (q_3 - q_0)(q_3 - q_0)^T \] was given as the measurement error of raw output of gyro tracker. $dq$, $dq^2$ and

\[ (w_3 - w_1)(w_3 - w_1)^T \] was approximated by random walk.

\[ \frac{dq}{dt} = \frac{f_{out} + f_{correlation}}{I_4 w_1^2} \] (57)

\[ F(w_3 - w_1)(w_3 - w_1)^T/F^2 \approx \frac{T}{dt} w_1^2 + r_2^2 I_4 \] (58)

Finally, we obtain $n_1 = 0.0058$ if $f_{correlation} = 0$.

As for $n_2$, we applied the same technique. $du/dw_0$ can be approximated by random walk, and

\[ (w_3 - w_1)(w_3 - w_1)^T \] was given as the measurement error of raw output of gyro sensor.

\[ \begin{align*}
x_1^2 &= (w_3 - w_2)(w_3 - w_2)^T \\
&= (w_3 - w_1)(w_3 - w_1)^T + du/dw_0 \\
&= (w_3 - w_1)(w_3 - w_1)^T + du/dw_0 \\
&= (w_3 - w_1)(w_3 - w_1)^T + du/dw_0 \\
\end{align*} \] (60)

\[ \frac{du}{dw_0} \approx f_{correlation}/I_4 w_1^2 \] (61)

\[ (w_3 - w_1)(w_3 - w_1)^T = 0.12^2 I_4 \] (62)

5 Off-line Experiment

5.1 Apparatus

We used Polhemus Tracker as the sensor of orientation, and three compact gyro sensors from Murata Co. (GYROSTAR ENC-05S) as the sensor of angular velocity in the body coordinate fixed to the human head. In order to evaluate performance, we compared the output of the proposed method with measurement of the link-type motion tracker (Tschi, Ari, & Atache, 1990; Ozawa, Tschi, & Inoue, 1993). Most tracking in this motion tracker are made out of metallic bars but FRP (fiber-reinforced plastics). The precision of the link-type sensor (accuracy 0.0125°, resolution 0.025°) was
much better than that of Tracker (accuracy 0.1°, resolution 0.5°) because the rotational angle of each joint was measured by a high-precision optical encoder and up-down counter.

The output of the up-down counter was directly connected to the data bus of the computer, and the lead-time of this up-down counter was only 2 μs. CPU performance determined the delay of this system. This system with 80286 (10 MHz) measured the operator’s motion and controlled the robot at least 300 Hz. Therefore, the response delay and communication delay of this link-type sensor was negligible compared with the Polhemus Tracker (~80 ms).

We checked three points:

1. We set the source and the receiver of magnetic motion trackers as closely as possible in several runs for gathering data. The distance between the source and the receiver was always within 0.5 m.

2. We made a testbed with a single axis for evaluating magnetic trackers. Most of the testbed is plastic, and the least-metallic parts like a potentiometer and bearings are used. We took normalized cross-correlation on this device, and compared it with the normalized cross-correlation taken on the six-DOF mechanical link system. The result is that the peak value with the six-DOF motion tracker is almost the same as the peak value without testbed.

3. The noise by the backlight of the HMD's LCD can cause bigger skew when the distance between the receiver and the LCD is within 1.5 cm than the link-type motion tracker.

We conclude that the skew by the six-DOF link tracker is small enough from the above observations, and used the output as a standard signal.

§ 2 Result (t_{connect} = 0 ms)

In order to make performance comparison easier to understand, we converted the computed quaternion into ZYX Euler angle ϕ, β, α and compared various outputs with the standard signal (output of the link-type sensor). Figures 3 and 4 show the standard signal, the raw output of Polhemus and the output of the proposed method. The average delay of Tracker was nearly 80 ms. It is apparent that the proposed method compensated this delay well.

Table 1 shows the RMS (Root Mean Square) error of various outputs from the standard signal. The RMS error of the Kalman-filtered raw Polhemus Tracker was greater than that of the raw Tracker because of over
Table 1. Comparison of RMS Error: "Polhemus" Denotes Raw Data of Polhemus Tracker, "Kolman" Denotes Kolman-filtered Raw Polhemus Data, and "Proposed" Denotes the Output of the Proposed Method

<table>
<thead>
<tr>
<th></th>
<th>RMS (×10^{-2} [rad])</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>γ</td>
</tr>
<tr>
<td>Polhemus</td>
<td>7.12</td>
</tr>
<tr>
<td>Method I</td>
<td>6.95</td>
</tr>
<tr>
<td>Method II</td>
<td>7.10</td>
</tr>
</tbody>
</table>

Figure 5. Comparison of link data: Polhemus, and Method I (proposed).

The output of Method I predicts 170 ms ahead of the original signal. There are slight overshoots around the peak. The output of Method II is not always continuous as Method I, and the overshoots around the peak are a bit greater than Method I. We see that the RMS error decreased to one-half to two-thirds of that of the original signal, and that Method I gave better estimation than Method II.

6 Correlational Evaluation

6.1 Principle

Let s(t) be the original signal, and let y(t) be the measurement of s(t) by a target sensor. Cross-correlation function \( \Phi_{ys}(\tau) \) (Beauchamp, 1973) is defined as

\[
\Phi_{ys}(\tau) = \frac{1}{T} \int_{-T}^{T} s(t) y(t + \tau) dt
\]

The normalized cross-correlation function \( r_y(t) \) (Beauchamp, 1973) is defined by using \( \Phi_{ys}(\tau) \) as

\[
r_y(\tau) = \frac{\Phi_{ys}(\tau)}{\sqrt{\Phi_{ss}(0) \Phi_{yy}(0)}}
\]
Table 2. Performance Comparison by RMS Error. Rendering
Delay (90 ms) Is Considered

<table>
<thead>
<tr>
<th></th>
<th>RMS ($\times 10^{-2}$ [rad])</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\gamma$</td>
</tr>
<tr>
<td>Polhemus</td>
<td>7.28</td>
</tr>
<tr>
<td>Method I</td>
<td>8.21</td>
</tr>
<tr>
<td>Method II</td>
<td>8.80</td>
</tr>
</tbody>
</table>

This normalized cross-correlation function has properties. First, the range of $\rho_d(\tau)$ is between $-1$ and $1$.

\[ -1 \leq \rho_d(\tau) \leq 1 \]

Second, if signal $x(t)$ follows $x(t)$ with lag of $T_d$,

\[ x(t) = y(t + T_d) \]  \hspace{1cm} (66)

$\rho_d(\tau)$ has a peak at $\tau = T_d$. Third, suppose independent random noise is added on signal $x(t)$. Let $S$ be the signal power, and $N$ be the noise power. Then

\[ \Phi_x(0) = S \]  \hspace{1cm} (67)
\[ \Phi_x(\tau) = S + N \]  \hspace{1cm} (68)
\[ \Phi_\alpha(\tau) = S \]  \hspace{1cm} (69)

The peak of $\rho_d(\tau)$ can be described

\[ \rho_d(T_d) = \frac{S}{\sqrt{\Phi_x(0) \Phi_x(\tau)}} \]  \hspace{1cm} (70)

\[ = \frac{S}{\sqrt{S + N}} \]  \hspace{1cm} (71)

\[ N \]  \hspace{1cm} (72)

$\rho_d(T_d)$ is the function of $S/N$ ratio and can be the index of the fidelity of $x(t)$ to $s(t)$.

Therefore, we can evaluate the delay and fidelity of the target sensor quantitatively by checking the peak of its normalized cross-correlation function (Einsatz & Tachi, 1994).

6.2 Result ($t_{render} = 0$ ms)

Figure 7 shows the $\rho(x)$ of Polhemus Tracker, Fastrak, and the proposed methods. We used the output of mechanical link-type sensor as the standard signal. The delay of Fastrak was half that of the Polhemus Tracker, but there remained still 30 ms of delay. There remained no delay for the proposed method, and its fidelity index was slightly improved.

6.3 Result (various $t_{render}$)

By the correlation technique above, we can plot the total compensated delay and the $S/N$ ratio of the output as a function of assumed rendering delay. Note that the standard signal is not the output of the link-type sensor, but the raw output of the magnetic tracker. Figure 8 shows that Method I can compensate for a longer delay than Method II. The upper limit of compensation is 200 ms for Method I and 120 ms for Method II. If we apply Method II with $t_{render} = 90$ ms, there still remained 50 ms delay, but it is half of the critical delay 100 ms. The $S/N$ ratio is nearly 1% (Figure 9), and small enough for practical use.
7 Real-Time Implementation

7.1 Hardware

The multi-sensor system consisted of a receiver of Polhemus Tracker and three compact gyro sensors (Piezo-electric vibrating gyroscope GYROSTAR ENC-035 from Murata MFG Co.). This sensing system was attached to a see-through HMD. A PC (Intel 80486 Overdrive 40 MHz) read the measurement of Polhemus through RS232C (19.2 Kbps), and that of the gyro sensors via an AD board. See Figures 10 and 11.
result to task B. Task B copied the result of task C if it was available; otherwise it kept extrapolation based on the measurements of gyro.

Three-dimension modeling of virtual space was done on an IBM PC by World Tool Kit (Sense8 Co.). The IBM PC sent a request to PC every image frame (10 ~ 12 frames/sec.). If requested, task B sent the latest estimate of human head motion to the IBM PC via parallel port for printer.

7.1 Verification by See-through HMD

We made a virtual copy of an actual room (consisted of 49 polygons) and 3-D images of this room were updated 11 frames/sec. on average and communication between the PC9801 and IBM PC via parallel port took only 1.4 ms.

Synchronization between the virtual and actual room was checked by see-through HMD and video camera. We set a 8 mm video camera in the see-through HMD and recorded the real scene overlaid by the virtual scene from the position of a human eye under various rotational motions at 30 frame/sec. Then we played back the video tape frame by frame, and checked how many frames passed since the virtual scene started to follow the real motion.

In the case of the raw output of the Polhemus Tracker, the delay was 166 ~ 200 ms (5 ~ 6 frames). Our real-time method decreased the delay to 33 ~ 66 ms (1 ~ 2 frames). This fits well with the result of the off-line computation of Method II.

8 Discussion

There are a lot of factors in VR systems that would cause time lag such as dynamic response of the motion sensor and communication between the sensor and computer, computation for the generation of virtual space, and communication between computers.

Even if we could develop a sensor without any measurement delay, this would not solve all these problems. We discuss the relation between our method and these problems in this section. We also discuss the application of our method to teleoperation.
6.1 Multiple Tracking

Though the maximum sampling rate of Fastrak is 120 Hz, which is higher than that of our real-time implementation now, the sampling rate of the proposed methods has no upper limit in principle, and is determined only by the computation speed of computer used.

When Fastrak is tracking motions of k (1 ≤ k ≤ 4) points, each sampling rate decreases to 1/k uniformly. However, these k points are each on a part of the human body, and the sampling rates should be varied according to the speed of each part. Our method enabled us to raise the sampling rate decreased to 1/k.

8.2 Generation of Virtual Scene

The real-time generation of 3-D images is one of the biggest problems in VR. It is obvious that 3-D images, built up by complicated objects or processed by computationally heavy techniques like texture mapping, give us better reality. However, the more complicated the virtual space is, the more it takes to render 3-D images and the slower the frame rate is. For us to feel interactivity, the frame rate must be at least 10 frame/sec, where the maximum rendering delay can be estimated easily at 100 ms.

Even if we could measure human motions with 0 μs delay, there would remain this rendering delay. Some kind of prediction is inevitable. Therefore our approach, better prediction by multisensor-integration method, will remain important if the performance of the magnetic tracker improves drastically and the measurement delay is almost 0 μs.

8.3 Networked Virtual Environment

In order to join people over distance through virtual environments, the sites should be networked computers. It takes some time for changes in virtual space to propagate and to synchronize on the network. In each computer, synchronization is realized by interruption processes such as network arbitration, network control, and low-level system activity, which causes latency. This should be compensated for by any kind of prediction. Though it is expected that the effects of such eons will appear as time lags in nonstationary manner, the fixed time-lag compensation proposed in this paper could be the first-order approximation of the final solution to this problem.

8.4 Application for Teleoperation

In the situation where we connect an operator not to virtual space in a computer but to a robot in actual space (“advanced teleoperation” or “tele-presence”), the cycle time of human motion sensing needs to be less than a few milliseconds to control the slave robot smoothly. This rate is much higher than that of the Polhemus. In order to apply low-sampling-rate, remotely sensing device like the magnetic tracker, our multisensor integration approach provides a significant advantage.

9 Conclusions

We propose the concept of multisensor integration to solve the time-lag problem of HMDs in single virtual space. Its validity has been shown through off-line computation. The simpler algorithm was able to be implemented successfully in real-time VR system, and the performance was checked. We also proposed a correlational method for evaluating the delay and fidelity of motion-tracking sensors, and showed the relation between the fidelity and the N/S ratio of target sensors, which is very useful to assess the experimental environment where a magnetic tracker is used.

References


Emura and Tachi 421
International Workshop on Robot and Human Communication, 124–129.


