A Hybrid Registration Method for Outdoor Augmented Reality

Kiyohide Satoh, Mahoro Anabuki, Hiroyuki Yamamoto, and Hideyuki Tamura
Canon Inc. †
2-2-1 Nakane, Meguro-ku, Tokyo 152-0031, Japan
{ksato, mahoro, ymmt, tamura}@mr-system.co.jp

Abstract

In this paper, a registration method for outdoor wearable AR systems is described. Our approach is based on using a high precision gyroscope, which can measure 3DOF angle of head direction accurately, but with some drift error. We solved the drift problem with a vision-based drift compensation algorithm, which tracks natural features in the outdoor environment as landmarks from images captured by a camera on an HMD. This paper first describes the detail of the vision-based drift compensation method. Then, a calibration method for the orientation sensor is proposed. Finally, using results from an actual wearable AR system, a comparison of registration error with and without vision-based drift compensation demonstrates the feasibility of the proposed method.

1. Introduction

Registration of a physical scene and a virtual space is one of the most important technical aspects of AR systems [1]. Various applications such as outdoor visual simulation could be realized if it were possible to build a wearable AR system with highly precise registration. Consequently, such a wearable AR system has become one of our research themes.

To align a virtual space to a physical scene, the position and orientation of the user's viewpoint must be known. When the user's position is limited to a certain area, a fixed sensor covering this limited area can be used with vision-based marker tracking to achieve highly precise registration [2][3]. But in the case that the user is walking around outdoors, a mobile sensor with no operation area limit becomes necessary.

It is quite difficult to measure accurately the position of a user's viewpoint in real time in a wide-open area. The viewpoint measurement accuracy, however, required for registration is generally determined in relation to the distance to the observation target. In other words, if the observation target in an outdoor environment is relatively far away, precise position measurement of viewpoint is not always required. Once approximate viewpoint position is known or obtained in some way, the issue of aligning virtual space to a physical scene becomes focused on precise measuring the orientation of the user's viewpoint.

We have developed a wearable AR system called "TOWNWEAR" [4][5], which stands for "Towards Outdoor Wearable Navigator With Enhanced & Augmented Reality." This system utilizes a high precision gyroscope [1] to measure the orientation of the user's viewpoint. In the first version of TOWNWEAR [4], we used only the gyroscope to measure the orientation. This allowed sufficient accuracy in registration for a short period, but drift error accumulated over time and led to registration distortion.

In this paper, we describe a hybrid registration method implemented in the improved version of TOWNWEAR [5],

† Part of this work was done while the authors were at Mixed Reality Systems Laboratory Inc.

[1] Generally, the gyroscope itself can measure only an angular velocity, but it is possible to produce a 3DOF orientation sensor by combining several gyroscopes and accelerometers. In this paper, we refer to this gyroscope-based 3DOF orientation sensor as a "gyroscope."

which uses the gyroscope and vision-based natural feature tracking for precise registration. In Section 2, related researches on registration for outdoor AR systems are introduced. In the following sections, the details of our approach are described. Then, a calibration method for an orientation sensor is proposed. Finally, using the results with an actual wearable AR system, a comparison of registration error with and without vision-based drift compensation demonstrates the feasibility of the proposed method.

2. Related works

Several researches have been done on wearable AR systems conducted for campus guides [6], situational awareness of soldiers [7], etc. For each system, a registration method for outdoor environment is being studied, but a wearable system with highly precise registration has yet to be realized.

In many outdoor AR systems, orientation of the user's viewpoint is simply measured by an orientation sensor, which is commercially available and used in the field of traditional VR (Virtual Reality). Höllerer et al [6], for example, used a hybrid sensor combining magnetic compasses and inertial sensors (IS-300PRO). Hirose and Hiroto [8] used magnetic compasses and inclinometers, and Behringer [9] used a hybrid sensor combining these (CyberTrack II)[2]. Accuracy of these sensors is sufficient for VR, and could be enough for AR applications where the registration of a physical scene and a virtual space is not highly important. In order to allow the user to see virtual objects as if they actually exist in a physical scene, however, the level of registration achieved by existing sensors is insufficient.

Some researches have been conducted to solve the lack of sensor precision by using hybrid sensors. Hoff and Azuma [10], for example, used an accelerometer to correct the error on an electronic compass caused by distortions in geomagnetism.

Another approach is the use of a computer vision technique for registration. However, due to constraints including difficulties in adjusting outdoor environment, such as pasting fiducials and tuning light conditions, and the lack of computation power offered in a wearable system, registration solely dependent on computer vision in an outdoor environment, for example, is much more difficult than for indoor use. Further, even with a hybrid method using a sensor with low accuracy and image processing, registration would still depend heavily on image processing, which still leaves us with the computation power problem.

You et al [7], for example, proposed correcting registration error caused by angular velocity sensors by tracking natural features. This method accomplished high precision registration, but involved much computing, and SGI Onyx2 was used to enable a frame rate of about 10fps.

Kourogi et al [11] proposed using a panoramic image database in pattern matching for registration. A PC cluster connected by wireless LAN is assigned for large scale computing, allowing a wearable PC to achieve 8fps. However, since a simple matching algorithm is used to enable real-time processing, stable registration cannot be achieved. They combined this system with orientation data from an inertial sensor to improve stability and throughput [12].

3. Goal and our approach

We set the following targets for our AR system:

- Precise registration
- Video-level frame rate
- Wearable system for outdoor use

As stated in the previous section, relying heavily on a computer vision technique for registration results in a trade-off between precision and frame rate. Processing capacity is especially limited for a wearable system, and it is difficult at this current stage to create a wearable system with enough processing capacity to compensate for this trade-off. On the other hand, if we try to rely on sensors, those widely used in VR are not precise enough for our purposes. However, in aircraft control and other areas, high-precision sensors such as fiber optic gyroscopes (FOG) or ring laser gyroscopes (RLG) are used. By adapting this kind of technology to an outdoor AR system, we sought to overcome the disadvantages in the current wearable AR systems in order to achieve our targets.

[2] He used terrain horizon silhouettes for initial registration.

To reduce the overload for image processing, we thought it necessary for the orientation sensor to produce precise output by itself to achieve accurate registration, as long as it is for a short period. If allowing a registration error within ±0.1 degree for each axis, the accuracy of head direction should be less than 6 degrees per hour in static drift error to satisfy this requirement for 1 minute. As for the weight, it should be under several hundred grams so that it is mounted on the head. These conditions can more or less be met with a "medium level" FOG, which we decided to use in our system.

In principle, a gyroscope has drift. Most gyroscopes —gyroscope-based 3D orientation sensors—, however, use accelerometers to prevent the drift of pitch and roll angle. Thus, yaw is the only axis that we have to make efforts to compensate for reducing drift error.

We use natural features in the outdoor environment as landmarks for the drift compensation. The features include corners of buildings, house roofs, and other physical objects whose image features can be used as indices for registration. The error correction algorithm is basically similar to that proposed by Bajura et al [13], and works by detecting landmarks in a captured image and comparing them with their predicted coordinates calculated from sensor output. The main issue here is how to detect landmarks in the captured image as accurately and consistently as possible.

4. Landmark detection

4.1. Outline

A template matching technique is used for landmark detection. Generally, two-dimensional rotation of the landmark on the image is one of the problems that makes it difficult for template matching. For example, suppose that we try to detect landmark *L* in the captured image *I* as shown in Fig.1(b) by using template image *T* shown in Fig.1(a), simple template matching cannot detect the landmark robustly because of the influence of the image rotation.

To deal with image rotation and enable robust detection of the landmark, it is effective to use several templates, each of which is made by adding rotation with different

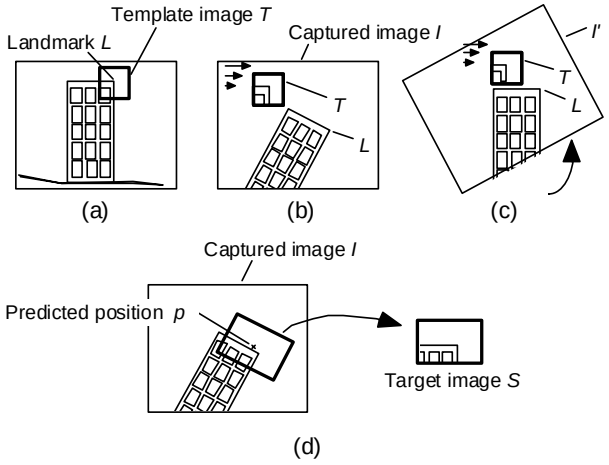


Figure 1 Template matching by rotating image

angles to the base template image *T*, or to use rotational invariants. However, these approaches increase computational cost and ambiguity, or the risk of detecting false feature points.

In our hybrid situation, we can use the rough orientation of the viewpoint that is supplied by the gyroscope. In the three measured values from the gyroscope, roll and pitch angles are relatively correct in comparison with the yaw angle in terms of drift error. By rotating captured image *I* by the roll angle of the viewpoint, we can generate image *I'* without the roll component as shown in Fig.1(c). Then, matching template *T* to image *I'* makes it possible to detect landmark *L* with no effect of image rotation.

The gyroscope also measures the rotation around the other two axes, and the drift correction value for the yaw angle has already been calculated from the previous frame. As shown in Fig.1(d), we can roughly predict *p*, the position of the landmark in captured image *I*, based on this information. Only the area around the predicted position is rotated in the manner above, and target image *S* is created for landmark search for limiting the search area. In this way, landmark detection by template matching is conducted rapidly and consistently.

To realize this method, the registration error for landmarks that results from the difference between the pre-registered position of the viewpoint and its actual position must be small enough to be ignored. Therefore, it is important to choose landmarks far enough from the user's viewpoint when choosing a feature point, as explained in the next section.

4.2. Defining landmarks and template images

Landmarks are defined by giving either a 3D position in the world coordinate system or a 2D position in a captured image. Following is the landmark setting and template image creation procedure for each case.

Here, we suppose that the gyroscope gives the orientation of the viewpoint in world coordinates by three rotation angles around each of *z*, *x*, and *y* axes[3]. ϕ, ψ, and θ denote the value of the rotation angles around each axis, i.e., roll, pitch, and yaw. $Rz(\phi)$, $Rx(\psi)$, and $Ry(\theta)$ denote the rotation matrices around each axis, respectively. Matrix R, which denotes the orientation of the viewpoint, is determined as $R = Rz(\phi)\ Rx(\psi)\ Ry(\theta)$.

Case 1: Landmark defined in world coordinates

Suppose that a landmark L_i (i is the identification number of the landmark) is defined as a feature point whose 3D coordinate is known in the world coordinate system. As the position of the viewpoint is known, we can define "initial camera coordinate system" with a parallel translation of the origin in world coordinates to the viewpoint. It is easy to calculate the 3D coordinate of the landmark L_i in this new coordinate system, $Pc_i^0 = (Xc_i^0, Yc_i^0, Zc_i^0, 1)$.

The 2D coordinate of landmark L_i in captured image I^t at time t, $p_i^t = (x_i^t, y_i^t)$, can be calculated based on the orientation of the viewpoint, R^t, as determined by the gyroscope, in a situation where its drift error is small enough to be ignored, such as immediately after sensor alignment. When landmark L_i is observed in captured image I^t, the template image T_i for the landmark is generated as a $N \times N$ rectangle as follows:

$$T_i(j,k) = I^t(x_i^t + j\cos\phi^t + k\sin\phi^t,\ y_i^t - j\sin\phi^t + k\cos\phi^t) \quad (1)$$

where $-N/2 \le j,k \le N/2$ and ϕ^t denotes roll angle of the viewpoint measured at time t.

Case 2: Landmark defined in an image

As mentioned before, the position of the user's viewpoint is assumed to be fixed. Thus, the only information necessary to determine the landmark position in the image is the orientation of landmark from the viewpoint. The distance from the viewpoint to the landmark is basically unnecessary. Thus, even if the world coordinate of a feature point in the actual scene is unknown, it can be used as a landmark by specifying its position in the captured image.

Suppose that a landmark L_i is defined by selecting a feature point on captured image I^t. In general, one point has been specified on an image, and this allows its pseudo position in camera coordinates to be defined with an arbitrary depth. Then, we can set landmark position Pc_i^t in the camera coordinate system. Further, based on the orientation of the viewpoint detected by the gyroscope, landmark position Pc_i^0 in the initial camera coordinate system mentioned above can be calculated as $R^{t^{-1}} Pc_i^t$. Once the position has been determined in the initial camera coordinate system, all subsequent calculations can be done in the same way as when the landmark is defined in the world coordinate system.

4.3. Detecting landmarks

To detect landmark L_i in the image I^t, landmark position p_i^t is predicted based on the sensor output. The space surrounding the predicted position is extracted as the area in which to search for the landmark, or target image S_i^t. Then, the template is used against the target image for matching. Details are as follows.

The position of landmark L_i in the image is predicted in a similar way as the creation of the template image. However, one difference is that sensor output cannot be used directly, and the drift correction value from previous calculations is used to calculate the position of the landmark. This means that when the measured direction of the viewpoint at time t is θ^t, and drift correction value from previous calculations is θ_d^{t-1}, we can obtain the viewpoint direction angle θ'^t as $\theta^t + \theta_d^{t-1}$. This angle is used to calculate predicted position p_i^t.

The target image S_i^t is created for each landmark L_i in a similar way as the creation of template images. In this case, the size of the target image needs to be bigger than the template image to include scope of searching. When

[3] The raw output of the sensor must be converted to the orientation of the viewpoint in the world coordinates. Sensor calibration for this conversion is explained in Section 6.

searching in an area from the predicted position of $\pm m$ pixels sideways, and $\pm n$ pixels up and down, the size of image $N' \times N''$ should be $N' = N + 2m$, $N'' = N + 2n$.

For each landmark, target image S_i^t is matched with template image T_i, and the position of landmark L_i in the target image is calculated.

5. Drift compensation

Theoretically, it is possible to compensate for drift error by one landmark. If landmark L_i is detected at (u_i^t, v_i^t) in the target image, the following steps update the drift correction value by using the value u_i^t, which is related to the yaw drift.

Step1: Deduce the detected position of the landmark, $\hat{p}_i^t = (\hat{x}_i^t, \hat{y}_i^t)$, in image I^t as

$$\begin{cases} \hat{x}_i^t = x_i^t + u_i^t \cos\phi^t \\ \hat{y}_i^t = y_i^t - u_i^t \sin\phi^t \end{cases}$$

Step2: Calculate the pseudo position of landmark, $\hat{P}c_i^t$ in the camera coordinate system by using $\hat{p}_i^t$, arbitrary depth, and an inversion of the projection matrix.

Step3: Transform $\hat{P}c_i^t$ to initial camera coordinates $\hat{P}c_i^0 = (\hat{X}c_i^0, \hat{Y}c_i^0, \hat{Z}c_i^0, 1)$ by using rotation matrix $R'^t = Rz(\phi^t)Rx(\psi^t)Ry(\theta'^t)$.

Step4: Deduce drift update value $\Delta\theta_d^t$ by comparing $\hat{P}c_i^0$ and Pc_i^0. The relation of these values is formulated as

$$a\hat{P}c_i^0 = Ry(\Delta\theta_d^t)Pc_i^0$$

where a is a constant. By solving this equation, $\Delta\theta_d^t$ is deduced as

$$\Delta\theta_d^t = \arctan\frac{\hat{Z}c_i^0 Xc_i^0 - \hat{X}c_i^0 Zc_i^0}{\hat{X}c_i^0 Xc_i^0 + \hat{Z}c_i^0 Zc_i^0} \qquad (2)$$

Step5: Update the drift correction value θ_d^t:

$$\theta_d^t = \theta_d^{t-1} + \Delta\theta_d^t \qquad (3)$$

When the above process of calculating the correction value is done for each rendering loop, the orientation of the viewpoint used for rendering computer graphics image can be $(\phi^t, \psi^t, \theta^t + \theta_d^t)$.

Using several landmarks is a effective way in making the algorithm work more robustly. When more than two landmarks are detected, drift update value $\Delta\theta_{di}^t$ is calculated for each landmark L_i. Then, representative value $\Delta\theta_d^t$ is calculated by obtaining the average of the drift update values. In this process, only the landmarks with good matching results are taken into account in order to eliminate output with low credibility. Also, when reliable landmarks are fewer than specified, the drift correction value is not updated for time t.

6. Sensor calibration

Up through the previous section, the sensor output was assumed to be data referring to the orientation of the viewpoint in the world coordinate system. However, in actuality, the sensor output is based on the orientation of the sensor itself in sensor coordinates, and not on the orientation of the video camera or user's viewpoint in world coordinates, which we are trying to measure. In this section, we discuss the calibration of the orientation sensor required to achieve this translation of coordinates.

6.1. Coordinate systems

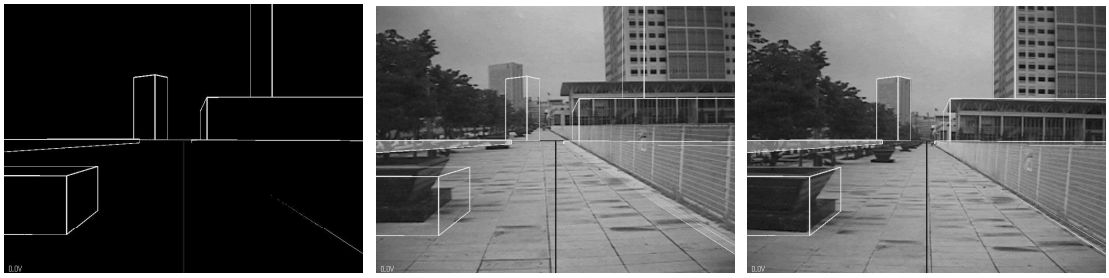
If R_{BA} represents the rotation matrix transforming the 3D position of a point from coordinate system A to B; R_{VW} represents the orientation of the viewpoint in the world coordinate system; and R_{ST} represents the orientation of the sensor in the sensor coordinate system, R_{ST} and R_{VW} are related as follows:

$$R_{VW} = R_{VS}\, R_{ST}\, R_{TW} \qquad (4)$$

Here, R_{TW} is the orientation of the sensor coordinates in the world coordinate system, and R_{VS} is the relative orientation of the viewpoint as seen by the sensor. R_{TW} and R_{VS} are required to transform the sensor output R_{ST} to R_{VW}.

Most inertial 3D orientation sensors can measure the orientation of gravity by using accelerometers, so that the y-axis in the sensor coordinate system is always perpendicular to the surface of the earth. Thus, by determining the x-z plane of the world coordinate system as parallel to the earth's surface, we can align the y-axial direction of the world and sensor coordinate systems. Here, for the orientation of the sensor coordinates in the world coordinate system R_{TW}, rotation components for x and z axes can be unit matrices. Only the rotation component for the y-axis, Ry_{TW}, is unknown. Equation (4) changes to

$$R_{VW} = R_{VS}\, R_{ST}\, Ry_{TW} \qquad (5)$$



(a) Wire-frame overlaid on live image (b) In case of wrong direction (c) In case of correct direction

Figure 2 Transferring viewpoint to initial orientation R^0_{VW}

As a result, we have four unknown parameters: the triaxial rotation angles R_{VS} (ϕ_{VS}, ψ_{VS}, θ_{VS}), and Ry_{TW} (θ_{TW}).

6.2. Sensor calibration using visual cues

It is difficult to fine-tune all unknown parameters manually and set them interactively. Azuma et al [14] conducted a bore-sight operation using visual cues to guide the viewpoint to a predetermined position, and calculated the eye-to-tracker transformation based on sensor output. Their case is different from ours in that the sensor used was a 6DOF, and the sensor output was already known based on world coordinates. However, by using a similar method in our project, we were able to reduce the number of parameters that must be set manually.

This means that, using visual cues, the viewpoint is moved to an initial orientation R^0_{VW}, which is pre-determined in the world coordinate system. At that point, sensor output R^0_{ST} is captured, and based on R^0_{VW} and R^0_{ST}, the unknown parameters in equation (5) are calculated. We now discuss the following two issues that need to be resolved for this process:

- How to transfer the viewpoint to initial orientation R^0_{VW},
- How to solve equation (5) based on R^0_{VW} and R^0_{ST}.

6.3. Transferring viewpoint to initial orientation

We explain this using the video see-through type of AR as an example. Architectural objects (e.g. office buildings) existing in the user environment are used as visual cues to transfer the viewpoint to initial orientation R^0_{VW}. Presuming that the viewpoint is at the initial orientation (i.e. camera parameter for rendering is set at the initial orientation), a wire-frame image of the cue buildings is rendered, overlaid onto the live captured image, and displayed on the HMD. Figure 2(a) shows the wire-frame image, and Fig.2(b) shows this image overlaid onto the live image.

If the viewpoint is at the initial position R^0_{VW}, the wire-frame and actual buildings should align. The user adjusts viewpoint position and orientation so that the image of the actual buildings and the wire-frame image overlap sufficiently. When this is done, as shown in Fig.2(c) (i.e. when viewpoint is at the initial orientation), the user presses a special key or otherwise makes some kind of input.

If data on the actual shape of buildings has already been compiled for the application to reproduce the mutual occlusion between the virtual objects and actual buildings, this information can be applied for the wire-frame models of the buildings.

In the case of the optical see-through type of AR, the same function is realized by rendering the wire-frame image of the buildings at initial orientation and displaying this on a see-through HMD.

6.4. Calculating unknown parameters

To facilitate calculations, we set initial orientation R^0_{VW} at $R^0_{VW} = Ry^0_{VW}(\theta^0_{VW})$, i.e., the viewpoint is parallel to the earth's surface. From equation (5) we have:

$$Ry^0_{VW} = R_{VS} R^0_{ST} Ry_{TW} \quad (6)$$

Using R_{SV}, which is the reverse matrix of R_{VS}, and expanding the matrices to the component of each axis, equation (6) is:

$$Rz_{SV} Rx_{SV} Ry_{SV} Ry^0_{VW} = Rz^0_{ST} Rx^0_{ST} Ry^0_{ST} Ry_{TW} \quad (7)$$

Since each side of the equation (7) is the product of the

components around the *z*, *x*, and *y* axes, the following equations can be obtained for each component:

$$Rz_{SV} = Rz^0_{ST} \quad (8)$$

$$Rx_{SV} = Rx^0_{ST} \quad (9)$$

$$Ry_{SV} Ry^0_{VW} = Ry^0_{ST} Ry_{TW} \quad (10)$$

So we can obtain Rz_{SV} and Rx_{SV} directly from equations (8) and (9). As for the *y*-axis, equation (10) cannot be solved directly because Ry_{TW} and Ry_{SV} are unknown.

In terms of rotation angle, equation (10) is:

$$\theta_{TW} = \theta^0_{VW} - \theta^0_{ST} + \theta_{SV} \quad (11)$$

If the user interactively adjusts the value of θ_{SV}, this value can be used in equation (11) to determine θ_{TW}. By applying these values constantly as sensor calibration results and choosing values where registration error caused by roll rotation can be resolved by visual observation, we can easily determine the unknown parameters.

6.5. Sensor alignment

The above calibration procedure deduces the matrix R_{VS} which should be calculated basically only once, when the sensor is attached to the HMD. R_{VS} does not change during use, so only $Ry_{TW}(\theta_{TW})$ needs to be calculated after the first calibration. We therefore attempt to calculate the unknown parameter using visual cues, as we have done above.

In this case, equation (5) is transformed:

$$R_{VW} = R_{VS} Rz_{ST} Rx_{ST} Ry_{ST} Ry_{TW} = R_{VS} Rz_{ST} Rx_{ST} Ry_{SW} \quad (12)$$

where Ry_{SW} is the rotation matrix determined by sensor's yaw angle, θ_{SW}, in the world coordinate system. By keeping θ_{SW} fixed at an initial yaw angle, θ^0_{SW}, and using sensor output for *roll* and *pitch* components as:

$$R'_{VW} = R_{VS} Rz_{ST} Rx_{ST} Ry^0_{SW} \quad (13)$$

R'_{VW} is used to render the target buildings' wire-frame image and it is overlaid on the live captured image. An overlay image is displayed where roll and pitch registration is in concurrence with view orientation movement, while only the direction of the viewpoint is fixed. Then, the only thing that the user has to do is adjust the direction of the viewpoint so that the live image and wire-frame are aligned. Note that, in this case, the attitude of the viewpoint can be arbitrary. By obtaining yaw angle θ^0_{ST} from the sensor at this time, we can calculate the unknown parameter θ_{TW} as:

$$\theta_{TW} = \theta^0_{VW} - \theta^0_{ST} \quad (14)$$

7. Experiment

This section shows the experimental results of incorporating the registration framework described above into the wearable AR system TOWNWEAR [5].

7.1. System components

Figure 3 shows the equipment of TOWNWEAR. As an orientation sensor, we have selected Tokimec's gyroscope TISS-5-70 and modified it to meet our requirements. Table 1 shows the principle specifications of the improved gyroscope named TISS-5-40 [15].

We used Toshiba's PORTEGE 3480, a B5-size notebook PC with Mobile Pentium III 600MHz and Savage IX graphics chip. A CardBus type video capture card, MSVCC03 by Hitachi ULSI, is also installed to capture full-size 16-bit color images at 30fps. The field of view of the CCD camera built in our HMD is 51 degrees horizontally and 39 degrees vertically.

Figure 3 System appearance

Table 1 Specifications of TISS-5-40

Parameter		Specification
Heading drift		1 deg / h
Attitude accuracy	pitch	±0.5 deg
	roll	±0.5 deg
Heading scale factor accuracy		±0.1 %
Maximum input rate		458 deg / sec
Update rate		250 Hz
Latency		2 msec
Weight		550 g

Figure 4 Experiment in outdoor environment

Figure 5 Part of landmarks used in experiment

Figure 4 shows how the equipment is actually used in the experiment.

7.2. Setting for experiment

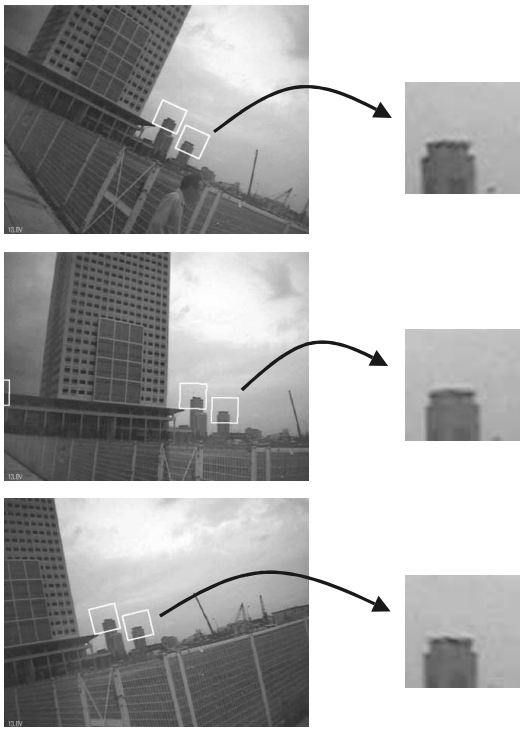
Figure 5 shows the view from the place where we conducted the experiment. The white squares show parts of the landmarks used in the experiment. We looked at the landscape from this viewpoint and chose landmarks that were far enough away, and which stood out in the captured image. These landmarks were selected on the live captured image using a mouse.

In this experiment, we set 10 landmarks around the user, creating a 50×50 pixel template image for each. The template search range was set at ±3 pixels high and ±5 pixels wide, setting the target image at 56×60. Figure 6(a) shows captured images at various times, and Fig.6(b) shows target images extracted for a landmark from these captured images. Figure 6 shows that, despite changes in viewpoint orientation, target images are retrieved consistently.

For template matching, we used SAD (Sum of Absolute Difference) as the metric of dissimilarity, which was calculated by summing up the absolute difference between pixels over the images as follows:

$$\mathrm{SAD}(u,v) = \sum_{\substack{-N/2 \le j \le N/2 \\ -N/2 \le k \le N/2}} \left| S_i^t(u+j, v+k) - T_i(j,k) \right|$$

Then, a pair of (u,v) that minimizes SAD(u,v) was selected as the landmark position on the target image. Also, in order to improve processing stability, we decided to update the drift correction value only when more than two landmarks were detected simultaneously with sufficient reliability.



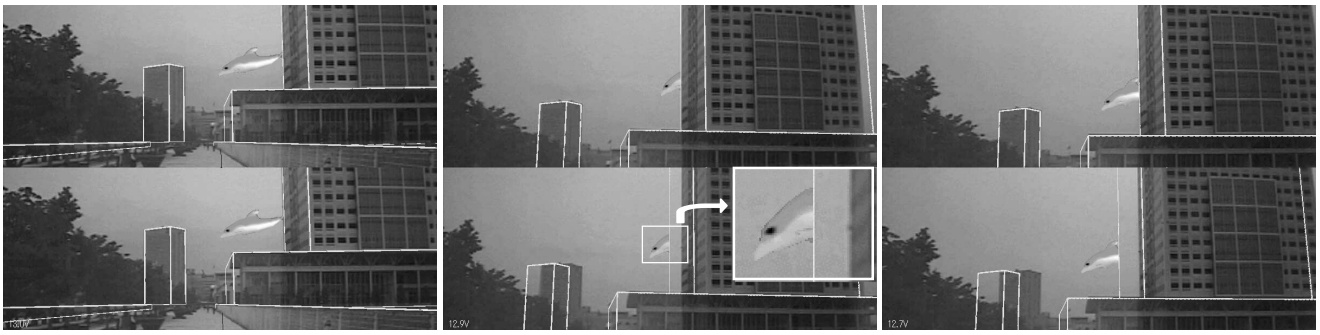
(a) (b)

Figure 6 Examples of target image extraction

7.3. Results

We generated augmented images with and without drift compensation simultaneously for the same captured image and same sensor output. The operator stood on a manhole cover (see Figure 4) as a viewing position. He moved his head naturally by looking around 360 degrees during the experiment and captured a screenshot every 2 minutes.

Figure 7 shows augmented images generated simultaneously with (top row) and without (bottom row) drift compensation. The virtual objects shown in the images are wire-frames of major buildings in view and one



(a) 2 min. after sensor alignment (b) 20 min. after sensor alignment (c) 40 min. after sensor alignment

Figure 7 Augmented images with/without drift compensation

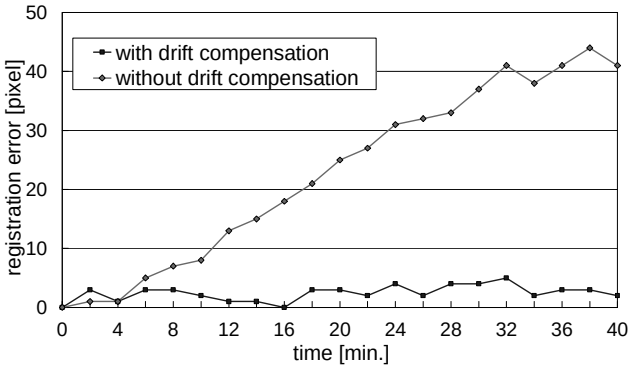


Figure 8 Registration errors with/without drift compensation

virtual animal (dolphin) swimming. Figure 7 (a) is the image 2 minutes after the sensor alignment described in Section 6.5. There is no major difference between the two. Figure 7 (b) is the image 20 minutes after the sensor alignment. When error correction is not conducted, there is a registration distortion caused by the sensor's drift (the dolphin that should be hidden by the actual building on the right side is not occluded correctly in the lower picture), but no obvious error occurs in the corrected case. Figure 7 (c) shows the image after 40 minutes, and the high-quality registration in the corrected image is maintained.

To measure registration error, we used several feature points apart from the landmarks being used for registration, and took the average distance between predicted observation coordinates for these points and the image coordinates appearing on the captured image in visual observation. Figure 8 shows the change of registration error over 40 minutes. As shown in the figure, registration error increased approximately in proportion to time, at 1 degree (about 12 pixels) every 12 minutes, if the drift compensation is not conducted. On the other hand, registration error was always less than 5 pixels (about 0.4 degree) if drift compensation was active.

Ideally, our method requires the user to fix his viewing position at the predetermined one, because movement will be sensed as a drift error. However, by selecting landmarks that were far enough away, 470m away on the average in this experiment, ±80 cm movement of the viewpoint causes only ±0.1 degree error in practical use.

8. Performance issues

In Section 5, we explained that the drift correction value is calculated for each rendering loop. However, if the matching algorithm is advanced to increase stability, or the size of the template image and number of the landmarks are increased, computation costs become a burden on image processing, and the rendering frame rate becomes too slow. On the other hand, if image processing is simplified to ensure rendering frame speed, error correction stability is sacrificed.

In our case, the correction value does not necessarily have to be updated in all frames. Rather, it is enough to update it only before significant registration error occurs due to drift of the gyroscope. Thus, we separated the rendering loop and drift compensation loop into two processes (or threads), and allocated independent updating cycles (for example, 30Hz for the rendering loop and 10 sec/frame for the drift compensation loop).

This means that image $I^{t'}$ and orientation $(\phi^{t'}, \psi^{t'}, \theta^{t'})$ at time t' are input for the drift compensation loop to calculate correction value $\theta_d^{t'}$ using the algorithm explained in the previous sections. This is sent to the

rendering loop as the latest correction value θ_d^{latest}. The rendering loop receives the sensor output $(\phi^t, \psi^t, \theta^t)$ independently, and uses $(\phi^t, \psi^t, \theta^t + \theta_d^{latest})$ as the camera parameter for rendering. By giving higher priority to the rendering loop, drift correction values are updated without slowing down the rendering loop.

The update rate of the rendered image depends largely on computer graphic (CG) image content, but in the experiment shown in Section 7.3, a rate of about 22 fps was maintained when drift compensation was not conducted. When using our drift compensation method, the update rate was about 18fps. When only simple CG images such as wire-frames were shown, this was kept to more than 30fps in both cases.

The update rate of the drift correction value depends on the number of landmarks in the image, but it was approximately every 2 seconds. This rate was more than enough to correct error in our gyroscope. This shows that there is much room for tuning the priority of the processes to improve the update rate of the rendering loop, or room for implementing a more time-consuming matching algorithm to improve robustness under various lighting or occlusion conditions.

9. Conclusion

By using the equipment and registration method described in this paper, we were able to achieve precise registration in our outdoor wearable AR system. Some additional developments, such as lightweight, accurate, and cost-effective tracking sensors or computer vision techniques that can cope with movement of the viewpoint position are required to further expand practical area of outdoor AR systems.

Acknowledgements

The authors would like to thank the people of Mixed Reality Systems Laboratory Inc. for their cooperative works and useful discussions.

References

[1] Y. Ohta and H. Tamura (eds.), *Mixed Reality - Merging Real and Virtual Worlds*, Ohmsha & Springer-Verlag, 1999.

[2] A. State, G. Hirota, D. T. Chen, B. Garrett, and M. Livingston, “Superior augmented reality registration by integrating landmark tracking and magnetic tracking,” *Proc. SIGGRAPH'96*, pp.429-438, 1996.

[3] K. Satoh, T. Ohshima, H. Yamamoto, and H. Tamura, “Case studies of see-through augmentation in mixed reality project,” *Proc. IWAR'98*, pp.3-18. 1998.

[4] K. Hara, M. Anabuki, K. Satoh, H. Yamamoto, and H. Tamura, “A wearable mixed reality system for outdoor use: design and implementation,” *Proc. VRSJ Fifth Annual Conf.*, pp.407-410, 2000 (in Japanese).

[5] K. Satoh, K. Hara, M. Anabuki, H. Yamamoto, and H. Tamura, “TOWNWEAR: An outdoor wearable MR system with high-precision registration,” *Proc. ISMR 2001*, pp.210-211, 2001.

[6] T. Höllerer, S. Feiner, and J. Pavlik, “Situated documentaries: Embedding multimedia presentations in the real world,” *Proc. ISWC'99*, pp.79-86, 1999.

[7] S. You, U. Neumann, and R. Azuma, “Orientation tracking for outdoor augmented reality registration,” *IEEE Computer Graphics & Applications*, Vol. 19, No. 6, pp.36-42, 1999.

[8] M. Hirose and K. Hiroto, “Overlaying image on real object in outdoor exhibition space,” *Proc. VRSJ Fifth Annual Conf.*, pp.385-388, 2000 (in Japanese).

[9] R. Behringer, “Registration for outdoor augmented reality applications using computer vision techniques and hybrid sensor,” *Proc. IEEE Virtual Reality '99*, pp.244-251, 1999.

[10] B. Hoff and R. Azuma, “Autocalibration of an electronic compass in an outdoor augmented reality system,” *Proc. ISAR 2000*, pp.159-164, 2000.

[11] M. Kourogi, T. Kurata, K. Sakaue, and Y. Muraoka, “A panorama-based technique for annotation overlay and its real-time implementation,” *Proc. ISME 2000*, pp.657-660, 2000.

[12] M. Kourogi, T. Kurata, K. Sakaue, and Y. Muraoka, “Improvement of panorama-based annotation overlay using omnidirectional vision and inertial sensors,” *Proc. ISWC 2000*, pp.183-184, 2000.

[13] M. Bajura and U. Neumann, “Dynamic registration correction in video-based augmented reality systems,” *IEEE Computer Graphics & Applications*, Vol.5, No.15, pp.52-60, 1995.

[14] R. Azuma and G. Bishop, “Improving static and dynamic registration in a optical see-through HMD,” *Proc. SIGGRAPH'94*, pp.197-204, 1994.

[15] K. Sawada, M. Okihara, and S. Nakamura, “A wearable attitude measurement system using a fiber optic gyroscope,” *Proc. ISMR 2001*, pp.35-39, 2001.