A Two-omni-camera Stereo Vision System with an Automatic Adaptation Capability to Any System Setup for 3D Vision Applications

Shen-En Shih and Wen-Hsiang Tsai, *Senior Member*, *IEEE*

***Abstract*—A stereo vision system using two omni-cameras for 3D vision applications is proposed, which has an automatic adaptation capability to any system setup before 3D data computation is conducted. The adaptation, which yields the two omni-cameras' orientations and distance, is accomplished by detecting and analyzing the horizontal lines appearing in the omni-images acquired with the cameras and a person standing in front of the cameras. Properties of line features in environments are utilized for detecting more precisely the horizontal lines which appear as conic sections in omni-images. The detection work is accomplished by the use of carefully chosen parameters and a refined Hough transform technique. The detected horizontal lines are utilized to compute the cameras' orientations and distance from which the 3D data of space points are derived analytically. Compared with a traditional system using a pair of projective cameras with nonadjustable camera orientations and distance, the proposed system has the advantages of offering more flexibility in camera setups, better usability in wide areas, higher precision in computed 3D data, and more convenience for non-technical users. Good experimental results show the feasibility of the proposed system.**

***Index Terms*—system setup, automatic adaptation, 3D vision applications, omni-camera, omni-image, stereo vision.**

I. Introduction

With the advance of technologies, various types of vision systems have been designed for many applications, like virtual and augmented reality, video surveillance, environment modeling, TV games, etc. Among these applications, human-machine interaction is a critical area [1]-[4]. For example, Microsoft Kinect [5] is a controller-free gaming system in the home entertainment field, which uses several sensors to interact with players. Most of these human-machine interaction applications require acquisitions of the 3D data of human bodies, meaning in turn the need of precise system calibration and setup works to yield accurate 3D data computation results in the application environment. However, from a consumer's viewpoint, it is unreasonable to ask a user to set up a vision system very accurately, requiring, e.g., the system cameras to be affixed at accurate locations in precise orientations. Contrarily, it is usually desired to allow a user to choose freely where to set up the system components.

Additionally, many interactive systems used for the previously-mentioned applications are composed of traditional projective cameras which collect less visual information than systems using omni-directional cameras (*omni-cameras*). To overcome these difficulties, a 3D vision system which consists of two omni-cameras with a capability of *automatic adaptation* to any camera setup is proposed. While establishing the system, the user is allowed to place the two cameras freely in any orientations with any displacement.

Human-machine interaction has been intensively studied for many years. Laakso and Laakso [6] proposed a multiplayer game system using a top-view camera, which maps player avatar movements to physical ones, and uses hand gestures to trigger actions. In [7], a special human-machine interface is proposed by Magee *et al.*, which uses the symmetry between left and right human eyes to control computer applications. Zabulis *et al.* [8] proposed a vision system composed of eight cameras mounted at room corners and two cameras mounted on the ceiling to localize multiple persons for wide-area exercise and entertainment applications. Starck *et al.* [9] proposed an advanced 3-D production studio with multiple cameras. The design considerations are first identified in that study, and some evaluation methods are proposed to provide an insight into different design decisions.

Geometric features, like points, lines, spheres, etc., in environments encode important information for on-line calibrations and adaptations [10][11]. Several methods have been proposed to detect such features in environments. Ying [12][13] proposed several methods to detect geometric features when calibrating catadioptric cameras, which use the Hough transform to find the camera parameters by fitting detected line features into conic sections. Duan *et al.* [14] proposed a method to calibrate the effective focal length of the central catadioptric camera using a single space line under the condition that other parameters have been calibrated previously. Von Gioi *et al.* [15] proposed a method to detect line segments in perspective images, which gives accurate results with a controlled number

This work was supported in part by a grant from the Ministry of Economic Affairs, Republic of China under Project No. MOEA 98-EC-17-A-02-S1-032 in the Technology Development Program for Academia.

S. E. Shih is with the Institute of Computer Science and Engineering, National Chiao Tung University, Hsinchu, Taiwan 30010. E-mail: peter159.cs98g@nctu.edu.tw.

W. H. Tsai is with the Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan 30010. He is also with the Department of Information Communication, Asia University, Taichung, Taiwan 41354. E-mail: whtsai@cis.nctu.edu.tw.

of false detections and requires no parameter tuning. Wu and Tsai [16] proposed a method to detect lines directly in an omni-image using a Hough transform process without unwarping the omni-image. Maybank *et al.* [17] proposed a method based on the Fisher-Rao metric to detect lines in paracatdioptric images, which has the advantage that it does not produce multiple detections of a single space line. Yamazawa *et al.* [18] proposed a method to reconstruct 3D line segments in images taken with three omni-cameras in known poses based on trinocular vision by the use of the Gaussian sphere and a cubic Hough space [19]. Li *et al.* [20] proposed a vanishing point detection method based on cascaded 1-D Hough transforms, which requires only a small amount of computation time without losing accuracy.

In this study, we propose a new 3D vision system using two omni-cameras, which has a capability of *automatic adaptation* to *any* system setup for convenient *in-field* uses. Specifically, the proposed vision system, as shown in Fig. 1, consists of two omni-cameras facing the user's activity area. Each camera is affixed firmly to the top of a rod, forming an *omni-camera stand*, with the camera's optical axis adjusted to be horizontal (i.e., parallel to the ground). The cameras are allowed to be placed *freely* in the environment at *any* location in *any* orientation, resulting in an *arbitrary system setup*. Then, by the use of space line features in environments, the proposed vision system can adapt *automatically* to the arbitrarily-established system configuration by just asking the user to stand still for a little moment in the middle region of the activity area in front of the two cameras. After this adaptation operation, 3D data can be computed *precisely* as will be shown by experimental results in this paper.

As an illustration of the proposed system, Fig. 1(c) shows the case of a user using a cot-covered fingertip as a 3D *cursor point*, which is useful for 3D space exploration in video games, virtual/augmented reality, 3D graphic designs, and so on. The fingertip is detected and marked as red in that figure, whose 3D location can be computed by triangulation.

In contrast with a conventional vision system with two cameras whose configuration is fixed, the proposed system has several advantages. First, the system can be established *freely*, making it suitable for wide-ranging applications. This is a highly desired property especially for consumer electronics applications such as home entertainment or in-house surveillance, since the user can place the system components flexibly without the need to adjust the positions of the existing furniture in the application environment. Second, since the proposed vision system uses omni-cameras, the viewing angle of the system is very wide. This can be seen as an improvement over commercial products like Microsoft Kinect since the player can now move more freely at a close distance to the sensors. This advantage is very useful for people who only have small spaces for entertainments. Also, the two cameras in the proposed system are totally separated from each other at a larger distance, resulting in the additional merit of yielding better triangulation precision and 3D computation results due to the resulting longer baseline between the two cameras.

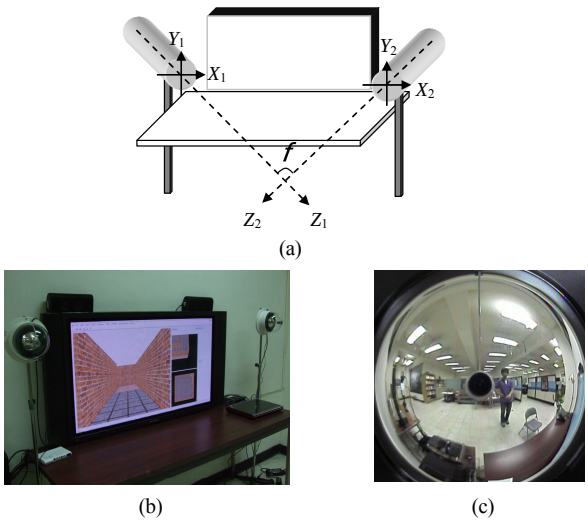


Fig. 1. Configuration and an illustration of the usage of proposed system. (a) An illustration. (b) Real system used in this study. (c) An omni-image of a user wearing a finger cot (marked as red).

In the remainder of this paper, an overview of the proposed system is described in Section II, and the details of the proposed techniques for use in the system are presented in Sections III through VI. Experimental results are included in Section VII, followed by conclusions in Section VIII.

II. Overview of Proposed System

The use of the proposed system for 3D vision applications includes three stages: (1) *in-factory* calibration; (2) *in-field* system adaptation; and (3) 3D data computation. The goal of the first stage is to calibrate the camera parameters efficiently in the factory environment. For this, a technique using landmarks and certain conveniently-measurable system features is proposed. In the second stage, an in-field adaptation process is performed, which uses line features in environments to automatically compute the orientations of the cameras and the distance between them (i.e., the baseline of the system). In this stage, a user with a known height is asked to stand in the middle region in front of the two cameras to complete the adaptation. Subsequently, the 3D data of any feature point (like the finger tip shown in Fig. 1(c)) can be computed in the third stage.

A sketch of the three operation stages of the proposed system is described in the following. To simplify the expressions, we will call the left and right cameras as Cameras 1 and 2, and their camera coordinate systems as CCSs 1 and 2, respectively.

Algorithm 1. *Sketch of the proposed system's operation.*
Stage 1. Calibration of omni-cameras.

Step 1. Set up a landmark and select at least two feature points P_i on it, called *landmark points*.
Step 2. Perform the following steps to calibrate Camera 1.
2.1. Measure manually the radius of the mirror base of the camera as well as the distance between the camera and the mirror, as stated in Section VII-A.
2.2. Take an omni-image I_1 of landmark points P_i with Camera 1 and extract the image coordinates of those pixels p_i which correspond to P_i.
2.3. Detect the circular boundary of the mirror base in I_1,

compute the center of the boundary as the *camera center*, and derive accordingly the focal length f_1 of the camera, as described in Section VII-A.

2.4. Calculate the eccentricity ε_1 of the hyperboloidal mirror shape using the coordinates of p_i and those of P_i, as stated in Section VII-A.

Step 3. Take an image I_2 of landmark points P_i with Camera 2 and perform operations similar to those of the last step to calibrate the camera to obtain its focal length f_2 and eccentricity ε_2.

Stage 2. Adaptation to the system setup.

Step 4. Place the two camera stands at proper locations with appropriate orientations to meet the requirement of the application activity.

Step 5. Perform the following steps to calculate the included angle ϕ between the two optical axes of the cameras as shown in Fig. 1(a).

5.1. Capture two omni-images I_1 and I_2 of the application activity environment with Cameras 1 and 2, respectively.

5.2. Detect space line features L_i in omni-image I_1 using the Hough transform technique as well as the parameters f_1 and ε_1, as described in Section IV.

5.3. Detect space line features R_i in omni-image I_2 similarly with the use of the parameters f_2 and ε_2.

5.4. Calculate angle ϕ using the detected line features L_i and R_i in a way as proposed in Section V.

Step 6. Perform the following steps to calculate the orientations of the two cameras and the baseline between them.

6.1. Ask a user of the system to stand in the middle region in front of the two omni-cameras and take two images of the user using the cameras.

6.2. Extract from the acquired images a pre-selected feature point on the user's body, and compute the respective orientations β_1 and β_2 of the two cameras using the angle ϕ, as described in Section VI-A.

6.3. Detect the user's head and foot in the images, compute the in-between distance up to a scale, and use the distance as well as the corresponding known height of the user to calculate the baseline D between the cameras, as described in Section VI-C.

Stage 3. Acquisition of 3D data of space points.

Step 7. Take two omni-images of a selected space feature point P (e.g., a fingertip, a handed light point, a body spot, etc.) with both cameras, and extract the corresponding pixels p_1 and p_2 in the taken images.

Step 8. Calculate as output the 3D position of P in terms of the coordinates of p_1 and p_2, the focal lengths f_1 and f_2, the eccentricities ε_1 and ε_2, the orientations β_1 and β_2, and the baseline D, using a triangulation based method described in Section VI-B.

Via the above algorithm, the meaning of *system adaptation*, which is the main theme of this study, can be made clearer now: only with the input of the knowledge of the user's height (see Step 6.3), the proposed system can infer the required values of the cameras' orientations β_1 and β_2 and baseline D for use in computing the 3D data of space points. This is *not* the case when using a conventional stereo vision system with two cameras in which the configuration of the cameras are *fixed* with their orientations and baseline *unchangeable*. This merit of the proposed system makes it easy to conduct system setup in any room space by any people for more types of applications, as mentioned previously.

III. Structure of Omni-Cameras

The structure of omni-cameras used in this study and the associated coordinate systems are defined as shown in Fig. 2. An omni-camera is composed of a perspective camera and a hyperboloidal-shaped mirror. The geometry of the mirror shape can be described in the camera coordinate system (CCS) as:

$$\frac{(Z-c)^2}{a^2}-\frac{X^2+Y^2}{b^2}=1\,,\ a^2+b^2=c^2,\ Z<c.$$

The relation between the camera coordinates (X, Y, Z) of a space point P and the image coordinates (u, v) of its corresponding projection pixel p may be described [22] as:

$$\tan\alpha=\frac{Z}{\sqrt{X^2+Y^2}}=\frac{(\varepsilon^2+1)\sin\beta-2\varepsilon}{(\varepsilon^2-1)\cos\beta}\ ; \tag{1}$$

$$\cos\beta=\frac{r}{\sqrt{r^2+f^2}}\ ;\ \sin\beta=\frac{f}{\sqrt{r^2+f^2}}\ ;\ r=\sqrt{u^2+v^2}\ , \tag{2}$$

where ε is the eccentricity of the mirror shape with its value equal to c/a, and α and θ are the *elevation* and *azimuth angles* of P, respectively. The azimuth angle θ can be expressed in terms of the image and camera coordinates as

$$\cos\theta=\frac{X}{\sqrt{X^2+Y^2}}=\frac{u}{\sqrt{u^2+v^2}};\ \sin\theta=\frac{Y}{\sqrt{X^2+Y^2}}=\frac{v}{\sqrt{u^2+v^2}}. \tag{3}$$

IV. Space Line Detection in Omni-Images

We now describe the proposed method to detect horizontal space lines in omni-images. Several ideas adopted to design the method are emphasized first. First, it is desired to eliminate initially as many non-horizontal space lines in each acquired image as possible since only *horizontal space lines* are used to find the included angle ϕ as described later in Section V. Second, it is hoped that the method can deal with large amounts of *noise* so that it can be used in an automatic process. Third, it is desired to utilize certain properties in man-made environments to improve the detection result, including the two properties that space lines are mostly horizontal or vertical, and that space line edges are usually not close to one another.

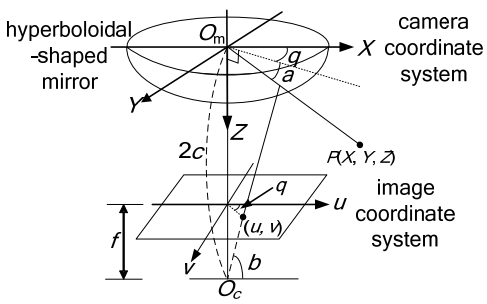


Fig. 2. Camera and hyperboloidal-shaped mirror structure.

This section is organized as follows. First, a quadratic formula describing the projection of a space line in an omni-image is derived in Section IV-A. Next, a refined Hough transform technique for detecting space lines is proposed in Section IV-B, which uses a novel adaptive thresholding scheme to produce robust detection results. Also, the projection of a vertical space line is derived and analyzed in Section IV-C. A peak cell extraction technique proposed for use in the refined Hough process is described at last in Section IV-D.

A. Projection of a Space Line in an Omni-Image

Given a space line L, we can construct a plane S which goes through L and the origin O_m of a CCS as shown in Fig. 3. Let $N_S = (l, m, n)$ denote the normal vector of S. Then, any point $P = (X, Y, Z)$ on L satisfies the following plane equation:

$$N_S \cdot P = lX + mY + nZ = 0. \tag{4}$$

where "·" denotes the inner-product operator. Combining (4) with (1) and (3), we get

$$lR\cos\theta + mR\sin\theta + nR\tan\alpha = 0, \tag{5}$$

where $R = \sqrt{X^2 + Y^2}$. Dividing (5) by $R/\sqrt{l^2 + m^2 + n^2}$ leads to

$$\frac{l\cos\theta}{\sqrt{l^2 + m^2 + n^2}} + \frac{m\sin\theta}{\sqrt{l^2 + m^2 + n^2}} + \frac{n\tan\alpha}{\sqrt{l^2 + m^2 + n^2}} = 0,$$

which can be transformed into the following form

$$A\cos\theta + \sqrt{1 - A^2 - B^2}\sin\theta + B\tan\alpha = 0 \tag{6}$$

with the two parameters A and B defined as

$$A = \frac{l}{\sqrt{l^2 + m^2 + n^2}},\ B = \frac{n}{\sqrt{l^2 + m^2 + n^2}}. \tag{7}$$

Accordingly, the normal vector N_S of plane S, originally being (l, m, n), can now be expressed alternatively as

$$N_S = (A,\ \sqrt{1 - A^2 - B^2},\ B). \tag{8}$$

It is assumed that $m \geq 0$ in (6) and (8) above. In case that $m < 0$, we may consider $N_S = (-l, -m, -n)$ instead, which also represents the same space plane S. Also, it can be seen from (7) that, parameters A and B satisfy the constraint $A^2 + B^2 \leq 1$, implying that the Hough space is of a circle shape.

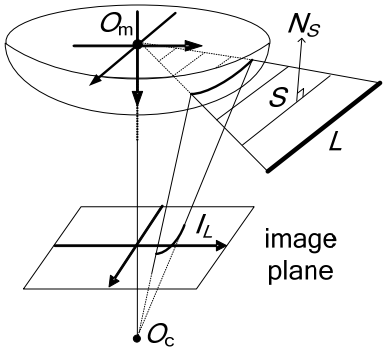


Fig. 3. Illustration of a space line L projected on an omni-image as I_L.

The parameters A and B are used in the Hough transform to detect space lines in omni-images. These two parameters are skillfully defined in (7), leading to several advantages. First, removals of vertical space lines can be easily achieved by ignoring periphery regions as described later in Section IV-C. Next, since the possible values of A and B range from −1 to 1, the size of the Hough space is *fixed* within this range. This is a necessary property in order to use the Hough transform technique, and is an improvement on a previous work [16]. Also, the parameters A and B are used directly to describe the directional vector of the space line L as will be shown later in (14). Hence, one may divide the Hough space into more cells to yield a better precision.

Combining (6) with (1) through (3), we can derive a *conic section* equation to describe the projection of a space line L onto an omni-image as follows:

$$F_{A,B}(u, v) = C_1u^2 + C_2uv + C_3v^2 + C_4u + C_5v + C_6 = 0, \tag{9}$$

where the coefficients C_1 through C_6 are:

$$C_1 = A^2 - B^2\left(C_7^2 - 1\right); \qquad C_2 = 2A\sqrt{1 - A^2 - B^2};$$

$$C_3 = 1 - A^2 - C_7^2B^2; \qquad C_4 = 2ABC_7f;$$

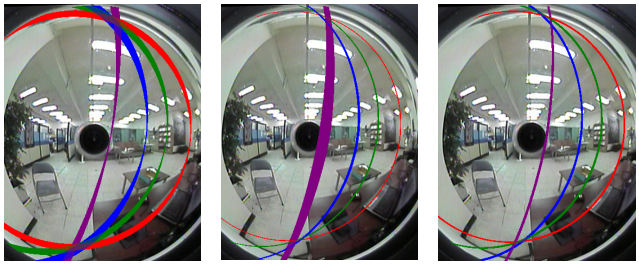
$$C_5 = 2BC_7\sqrt{1 - A^2 - B^2}\,f; \qquad C_6 = B^2f^2$$

$$C_7 = \frac{\varepsilon^2 + 1}{\varepsilon^2 - 1}.$$

The quadratic formula (9) will be called the *target equation* in the Hough transform subsequently, since the goal of the detection process is to find curves described by it in an omni-image.

B. Hough Space Generation with Adaptive Thresholding

We define the Hough space to be two-dimensional with the parameters A and B described previously. Furthermore, we define the *cell support* for a cell at (A, B) in the Hough space as the set of those pixels which contribute to the accumulation of the value of that cell. Let L denote a space line described by the two parameters (A, B). Two properties of cell supports are desirable: (1) the pixels of the projection I_L of L onto the omni-image are all included in the cell support for the cell (A, B); and (2) the pixels not on I_L are not included in this cell support. Furthermore, it is desired that the shape of the cell support is of a certain *fixed* width and not too "thin," so that (edge) pixels originally belonging to I_L but with small detection errors can still contribute to the cell value. In short, a cell support is desired to be a space line projection with a certain width everywhere along the line, which is called an *equal-width projection curve* hereafter. In this section, we first show that commonly-used curve detection methods do not generate desired equal-width projection curves as cell supports as shown in Figs. 4(a) and 4(b), so we propose in this study an adaptive method to solve this problem to yield better results like the one shown in Fig. 4(c).



(a) (b) (c)

Fig. 4. Shapes of cell supports of four chosen Hough cells yielded by three methods. (a) Using traditional accumulation method. (b) Using a threshold $\delta = 3000$. (c) Using the proposed technique.

A commonly-used method for curve detection to calculate the cell support is as follows [30][31][32]: for each pixel at coordinates (u, v), find all the Hough cells with their parameter values (A, B) satisfying the target equation (9), and increment the value of each cell so found by one. Some cell supports calculated by this method are shown in Fig. 4(a), showing that the cell supports for some cells are *not* with equal widths.

Another straightforward method to calculate the cell support is as follows [16][33]: define a threshold δ first, and for each (edge) pixel with coordinates (u, v), find all the Hough cells with their parameters (A, B) satisfying the equation

$$\left|F_{A,B}(u,v)\right| = \left|C_1u^2 + C_2uv + C_3v^2 + C_4u + C_5v + C_6\right| \le \delta \,, \quad (10)$$

and increment the value of each cell so found by one. However, as shown in Fig. 4(b), it is impossible to find a good threshold δ which makes all the projection curves to be of equal widths. To solve this problem, it is necessary to develop a new method to adaptively determine the threshold value δ for each different cell support and each different pixel.

Conceptually, to draw an equal-width curve of $F = 0$, we have to compute the function values of F on the projection curve boundary, and define the threshold δ accordingly. For this aim, the method we propose makes a novel use of *total derivatives* to estimate the function values of F on the boundary, and sets the threshold value δ in (10) accordingly. More specifically, δ is set in the proposed method to be

$$\delta(A,B,u,v) = \max_{(\Delta u,\Delta v)=(\pm 1,\pm 1)} \left(\frac{\partial F_{A,B}}{\partial u}\Delta u + \frac{\partial F_{A,B}}{\partial v}\Delta v \right) \quad (11)$$

for different Hough cells with parameters (A, B) and different pixels at coordinates (u, v). Accordingly, as shown in Fig. 4(c), the drawn curves are now with uniform widths.

As a summary, the Hough space can be generated using (10) with threshold δ calculated by (11). With this improvement, the cell supports become equal-width projection curves, making the Hough transform process more robust to yield a precise peak value which represents a detected space line.

C. Additional Constraint on Vertical Space Lines

In man-made environments, most lines are either parallel to the floor (which is called *horizontal space lines* hereafter) or perpendicular to the floor (which is called *horizontal space lines*). If we can eliminate vertical space lines from the detection results, the rest of them are much more likely to be horizontal ones which are desired as stated later in Section V. In this section, a constraint on the vertical space line is derived for the purpose of removing such lines.

As mentioned earlier, the omni-camera stands are vertically placed on the floor, with the Y-axis of the camera coordinate system being a vertical line as depicted in Fig. 1(a). As a result, the directional vector v_L of a vertical space line L is just $(0, 1, 0)$. Let S be the space plane going through L and the origin O_m which is at camera coordinates $(0, 0, 0)$. Also, let $N_S = (l, m, n)$ be the normal vector of plane S. By definition, normal vector N_S is perpendicular to v_L, leading to the constraint:

$$N_S \cdot v_L = (l, m, n)\cdot(0, 1, 0) = m = 0.$$

This constraint, when combined with (7), results in the equality $A^2 + B^2 = 1$, which shows subtly that the Hough cells of vertical space lines are located in the periphery region of the *circular* Hough space (as mentioned earlier in Section IV-A). As a result, vertical space lines can be easily removed by just ignoring the periphery region of the Hough space. In the proposed method, this is achieved automatically by applying a filter on the Hough space as described next in Section IV-D.

Note that, in general, vertical and horizontal space lines do not correspond to curve segments with vertical and horizontal chords in omni-images. In fact, the projections of horizontal space lines may be with any direction as shown in Fig. 11(f). Also, the removal of a vertical space line will sometimes also eliminate a few horizontal space lines lying on the plane which goes through the vertical space line and the origin of the camera coordinate system. However, as shown in Figs. 7(a), 7(b), 11(e), and 11(f), many horizontal space lines can still be extracted.

D. Peak Cell Extraction

After the Hough space is generated, the last thing to do is to extract cells with peak values, called *peak cells*, which represent the detected space lines. The simplest way to accomplish this is to find the cells with large values. However, if we do so to get peak cells like those shown in Fig. 5(a), we might get a bad detection result like that shown in Fig. 5(b) with many of the detected space lines being too close to one another, from which less useful space lines may be extracted.

To solve this problem, we notice that the line edges in an environment mostly are not so close mutually, meaning that two detected horizontal lines usually are separated for a certain distance. This in turn means that extracted peak cells should not be too close to one another. To find the peak cells which are not too close to each other, a filter is applied on the Hough space:

$$\frac{1}{25}\begin{bmatrix} -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & 24 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 \end{bmatrix}. \quad (12)$$

Then, we extract peak cells by choosing the cells with large values in the filtered Hough space to yield a better detection result, as shown by Figs. 5(c) and 5(d).

By the way, it is noted that when applying the filter to the Hough space, one of the side effects is the removal of the periphery region. This is a desired property mentioned in Section IV-C: the removal of the periphery region is equivalent to the removal of vertical space lines. Thus, expectedly we can get more horizontal lines as desired. To sum up, we have proposed a new method to detect horizontal space lines in omni-images, with several novel techniques also proposed in Sections IV-A through IV-D to improve the detection result. The proposed method for horizontal space line detection is summarized as an algorithm in the following.

Algorithm 2. *Detection of horizontal space lines in the form of conic sections in an omni-image.*

Input: an omni-image I.

Output: 2-tuple values (A_i, B_i) as defined in (7) which describe detected horizontal space lines in I.

Step 1. Extract the edge points in I by an edge detection algorithm [25].

Step 2. Set up a 2D Hough space H with two parameters A and B, and set all the initial cell values to be zeros.

Step 3. For each detected edge point at coordinates (u, v) and each cell C with parameters (A, B), if (u, v, A, B) satisfies (10) in which the threshold value δ is adaptively calculated by (11), then increment the value of C by one.

Step 4. Apply the filter described by (12) to Hough space H, choose those cells with maximum values, and take their corresponding parameters (A_i, B_i) as output.

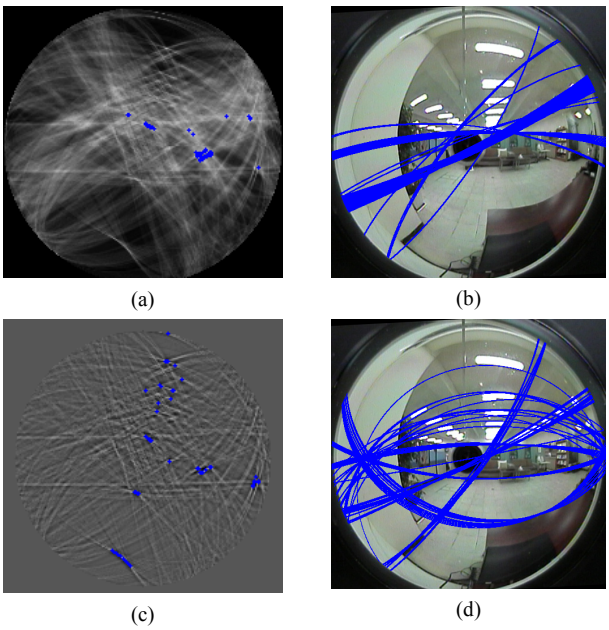


Fig. 5. Comparison of traditional peak cell extraction method and proposed one. (a) Hough space. (b) 50 detected space lines using traditional method. (c) Post-processed Hough space. (d) 50 detected space lines using proposed method.

V. Calculation of Included Angle ϕ between Two Cameras' Optical Axes Using Detected Lines

In the proposed vision system, the omni-cameras are mounted on two vertical stands with the optical axes being parallel to the floor plane as mentioned previously, but the cameras' optical axes are allowed to be non-parallel, making an included angle ϕ as depicted in Fig. 1(a). To accomplish the 3D data computation work under an arbitrary system setup, the included angle ϕ must be calculated first. A method to calculate the angle ϕ using a *single manually chosen* horizontal space line is proposed first in Section V-A. However, in order to conduct the adaptation process automatically, we have to calculate the angle ϕ using *multiple automatically extracted* horizontal space lines. To achieve this, a novel method is proposed next in Section V-B, which utilizes *all* the detected space lines from the two omni-images taken with the cameras.

The proposed method has several advantages. First, only the directional information of the space line, which is a robust feature against noise, is used. Next, no line correspondence between the two omni-images need be derived; that is, it is unnecessary to decide which line in the left omni-image corresponds to which one in the right omni-image. This makes the proposed method fast, reliable, and suitable for a wide-baseline stereo system like the one proposed in this study. Also, the proposed method makes use of a good property of the man-made environment — many line edges in such environments are parallel to one another, leading to an improvement on the robustness and correctness of the computation result.

A. Calculating Angle ϕ Using a Single Horizontal Space Line

In this section, a method to calculate the angle ϕ between the two cameras' optical axes is proposed, using a single horizontal space line L in the environment. Let (A_1, B_1) be the parameters corresponding to line L in an omni-image taken with Camera 1, $v_L = (v_x, v_y, v_z)$ be the directional vector of L in CCS 1, and S_1 be the space plane going through line L and the origin of CCS 1. The normal vector of S_1 can be derived, according to (8), to be

$$n_1 = (A_1,\ \sqrt{1 - A_1^2 - B_1^2},\ B_1)\,.$$

Since S_1 goes through line L, we get to know that v_L and n_1 are perpendicular, resulting in the following equality:

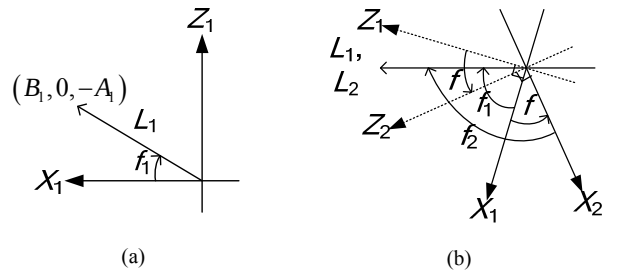
$$v_L \cdot n_1 = v_x A_1 + v_y \sqrt{1 - A_1^2 - B_1^2} + v_z B_1 = 0\,. \qquad (13)$$

Furthermore, since L, being horizontal, is parallel to the XZ-plane as shown in Fig. 1(a), we get another constraint $v_y = 0$. This constraint can be combined with (13) to get

$$v_L = (v_x, v_y, v_z) = (B_1, 0, -A_1). \qquad (14)$$

Next, by referring to Fig. 6(a), it can be seen that the angle ϕ_1 between the X-axis of CCS 1 and space line L is

$$\phi_1 = \tan^{-1}(-A_1/B_1).$$



(a) (b)

Fig. 6. Illustration of the angles ϕ_1, ϕ_2 and ϕ. (a) The definition of ϕ_1. (b) Relation between ϕ_1, ϕ_2 and ϕ.

Similarly, let (A_2, B_2) be the parameters corresponding to the horizontal space line L in Camera 2. By following the same derivations described above, the angle ϕ_2 between the X-axis of CCS 2 and line L can be derived to be

$$\phi_2 = \tan^{-1}(-A_2/B_2).$$

As depicted in Fig. 6(b) where L_1 and L_2 specify identically the single horizontal space line L, the angle ϕ between the two cameras' optical axes can now be computed easily to be

$$\phi = \phi_1 - \phi_2 = \tan^{-1}(-A_1/B_1) - \tan^{-1}(-A_2/B_2). \qquad (15)$$

B. *Calculating Angle ϕ Reliably Using Several Detected Lines*

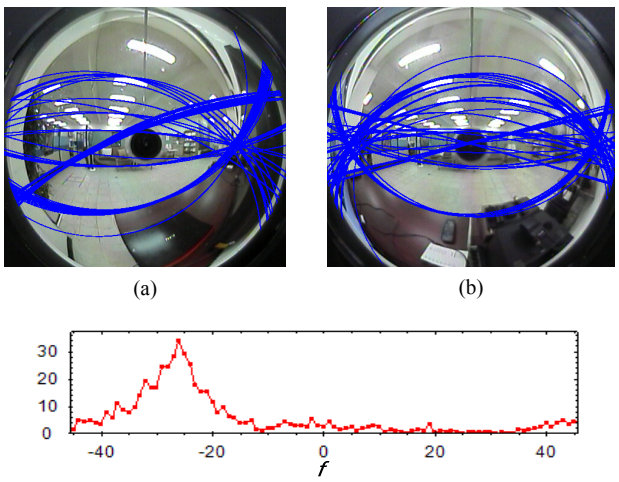
Horizontal space lines can be detected from an omni-image using Algorithm 2 as described in Section IV. Let L_1 be a space line so detected from the left omni-image with parameters (A_1, B_1), and let L_2 be another detected similarly from the right omni-image with parameters (A_2, B_2). As stated previously, the angle ϕ can be calculated using (15) if the space lines L_1 and L_2 are an identical horizontal space line L in the environment.

However, the *line correspondence* problem of deciding whether L_1 and L_2 are identical or not is difficult for several reasons, especially for a wide-baseline stereo system like the one proposed in this study. First, the respective viewpoints and viewing fields of the two cameras differ largely. Thus, environment features, like lighting and color, involved in the image-taking conditions at the two far-separated cameras might vary largely as well. Also, the extrinsic parameters of the two cameras are unknown; therefore, the involved geometric relationship is not available for use to determine the line correspondences. To get rid of these difficulties, we propose a novel statistics-based method to reliably find the angle ϕ *without* the need to find such line correspondences.

More specifically, the proposed method makes use of two important properties. First, it is noticed that the correct value of the angle ϕ can still be calculated using (15) even when the two space lines L_1 and L_2 are not an identical one, but are *parallel* to each other. This can be seen from the fact that the angles ϕ_1 and ϕ_2 remain the same if L_1 and L_2 are parallel so that the computed angle ϕ is still correct, as desired. Second, it can be seen that in man-made environments, many of the line edges are parallel to one another in order to make the environment neat and orderly. For example, tables, shelves, and lights are always placed to be parallel to walls and to one another. Combining these two properties, we can conclude that any two detected space lines L_1 and L_2 are very likely to be parallel to each other. Based on this observation, we assume every possible line pair L_1 and L_2 to be parallel, and compute accordingly a candidate value for angle ϕ, where L_1 is one of the space lines detected from the left omni-image, and L_2 is another detected from the right omni-image. Then, we infer a correct value for angle ϕ from the set of all the computed candidate values via a statistical approach based on the concept of "voting."

In more detail, the proposed method is designed to include three main steps. First, we extract space lines from the left omni-image as described in Algorithm 2, and denote the line parameters (A, B) of them as l_i. Similarly, we detect space lines from the right omni-image with their parameters denoted as r_j. In addition, we define two *weights* $w(l_i)$ and $w(r_j)$ for l_i and r_j, respectively, to be the cell values in the post-processed Hough space derived in Step 4 of Algorithm 2, which represent the *trust measures* of the detected space lines. Then, from each possible pair (l_i, r_j), we calculate a value ϕ_{ij} for angle ϕ using (15), as well as a third weight w_{ij} defined as $w(l_i)\times w(r_j)$. The value w_{ij} may be regarded as the *trust measure* of the calculated angle ϕ_{ij}. Finally, we set up a set of bins, each for a distinct value of ϕ, and for each computed value ϕ_{ij}, we increase the value of the corresponding bin by the weight w_{ij}. After such a weight accumulation work is completed, the bin with the largest value is found out and the corresponding angle ϕ_{ij} is taken as the desired value for angle ϕ.

An experimental result so obtained is shown in Fig. 7. In Figs. 7(a) and 7(b), fifty space lines with parameters l_i and r_j were detected using Algorithm 2 from the left and right omni-images, respectively. For each possible pair (l_i, r_j) where $1 \le i, j \le 50$, the corresponding angle ϕ_{ij} and weight w_{ij} were calculated and accumulated in bins as described previously. The accumulation result is shown in Fig. 7(c) with the maximum occurring at $\phi = -23°$, which is taken finally as the derived value of angle ϕ.



(a) (b)

(c)

Fig. 7. Experimental result of proposed adaptation method for detecting included angle ϕ. (a) and (b) Left/right omni-images, with the detected space lines superimposed on it. (c) Accumulation result for ϕ with maximum occurring at $\phi = -23°$.

VI. Proposed Technique for Baseline Derivation and Analytic Computation of 3D Data

The world coordinate system *X*-*Y*-*Z* is defined as depicted in Fig. 8. The *X*-axis goes through the two camera centers O_1 and O_2; the *Y*-axis is taken to be parallel to the *Y*-axes of both CCSs; the *Z*-axis is defined to be perpendicular to the *XY*-plane; and the origin is defined to be the origin O_1 of CCS 1. It is noted here that, since the two omni-cameras are affixed firmly on the omni-camera stands and adjusted to be of an identical height as described in Section I, the axes *X*, *Z*, X_1, Z_1, X_2, and Z_2 are all on the same plane as illustrated in Fig. 8.

Since the two omni-cameras are allowed to be placed arbitrarily at any location with any orientation, it is necessary to find the *baseline D* and the *orientation angles* β_1 and β_2 (as defined in Fig. 8) in advance to calculate the 3D data of space points. A novel method to calculate the orientation angles is proposed first in Section VI-A. After the orientations are derived, the 3D data can be determined *up to a scale* as discussed in Section VI-B. Then, a method using the known height of the user to determine the baseline *D* is proposed in Section VI-C. After the baseline *D* is derived, the *absolute* 3D data of space feature points can be derived by a similar method as proposed in Section VI-B. It is emphasized that all computations involved in these steps are done *analytically*, i.e., by the use of formulas without resorting to iterative algorithms.

A. Finding Two Cameras' Orientations

Let the camera coordinates of CCS 1 be denoted as (X_1, Y_1, Z_1), and those of CCS 2 as (X_2, Y_2, Z_2), as shown in Fig. 8. As mentioned previously, the two CCSs X_1-Y_1-Z_1 and X_2-Y_2-Z_2 are allowed to be oriented arbitrarily (with Y_1 and Y_2 parallel to each other), and the only knowledge acquired by the proposed system is the angle ϕ between the two optical axes Z_1 and Z_2, which is derived using the detected space lines, as described previously in Section V.

To derive the angles β_1 and β_2, the user is asked to stand in the middle region in front of the two omni-cameras so that a feature point P_{user} on the user's body may be utilized to draw a *mid-perpendicular plane* of the line segment O_1O_2 as shown in Fig. 8. Let (X_1, Y_1, Z_1) be the coordinates of P_{user} in CCS 1, and (u_1, v_1) be the corresponding pixel's image coordinates in the left omni-image. From (1) and (3), we have the equality:

$$\begin{bmatrix} X_1 & Y_1 & Z_1 \end{bmatrix}^{\mathrm{T}} = \sqrt{X_1^{\,2} + Y_1^{\,2}} \begin{bmatrix} \cos\theta_1 & \sin\theta_1 & \tan\alpha_1 \end{bmatrix}^{\mathrm{T}},$$

where $\cos\theta_1$, $\sin\theta_1$, and $\tan\alpha_1$ are computed from (u_1, v_1) according to (1) and (3). This equality shows that the directional vector between O_1 and P_{user} is ($\cos\theta_1$, $\sin\theta_1$, $\tan\alpha_1$) in CCS 1. An angle ψ_1 is defined on the *XZ*-plane as illustrated in Fig. 8, which can be expressed as $\psi_1 = \tan^{-1}(\tan\alpha_1/\cos\theta_1)$. Similarly, the angle ψ_2 defined on the *XZ*-plane can be derived to be $\tan^{-1}(\tan\alpha_2/\cos\theta_2)$. Accordingly, we can derive β_1 to be

$$\beta_1 = \psi_1 - \left(\frac{\pi}{2} - \frac{\psi_2 - \psi_1 + \phi}{2} \right) = \frac{\psi_1 + \psi_2 + \phi}{2} - \frac{\pi}{2},$$

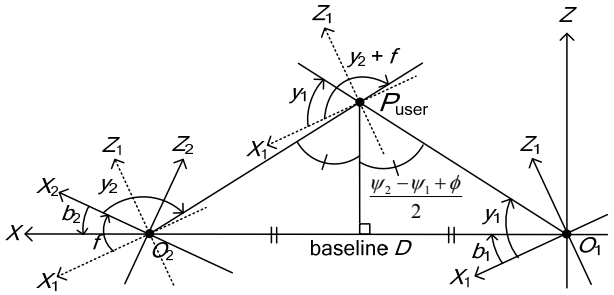


Fig. 8. A top-view of the coordinate systems. The baseline *D*, orientation angles β_1 and β_2, and a point P_{user} on the user's body are also drawn.

and β_2 is just $\beta_2 = \beta_1 - \phi$. This completes the derivations of the orientation angles β_1 and β_2 of the two cameras.

B. Calculating 3D Data of Space Feature Points

Let *P* be a space feature point with coordinates (*X*, *Y*, *Z*) in CCS 1, and let the projection of *P* onto the omni-image taken by Camera 1 be the pixel p_1 located at image coordinates (u_1, v_1). From (1) and (3) with $R_1 = \sqrt{X_1^{\,2} + Y_1^{\,2}}$, we have

$$\begin{bmatrix} X_1 & Y_1 & Z_1 \end{bmatrix}^{\mathrm{T}} = R_1 \begin{bmatrix} \cos\theta_1 & \sin\theta_1 & \tan\alpha_1 \end{bmatrix}^{\mathrm{T}} \tag{16}$$

where $\cos\theta_1$, $\sin\theta_1$, and $\tan\alpha_1$ are computed from (u_1, v_1) by (1) and (3). Equation (16) describes a light ray L_1 going through the origin O_1 with directional vector $d_1' = [\cos\theta_1 \ \sin\theta_1 \ \tan\alpha_1]^T$ in CCS 1. To transform the vector into the coordinate system *X*-*Y*-*Z*, we have to rotate d_1' along the *Y*-axis through the angle β_1 as illustrated in Fig. 8. As a result, the transformed light ray L_1 goes through (0, 0, 0) with its directional vector d_1 being

$$d_1 = \begin{bmatrix} \cos\beta_1 & 0 & -\sin\beta_1 \\ 0 & 1 & 0 \\ \sin\beta_1 & 0 & \cos\beta_1 \end{bmatrix} \begin{bmatrix} \cos\theta_1 \\ \sin\theta_1 \\ \tan\alpha_1 \end{bmatrix}. \tag{17}$$

Similarly, let the space feature point *P* be located at (*X*′, *Y*′, *Z*′) in CCS 2 and its projection onto the omni-image taken by Camera 2 be the pixel p_2 located at image coordinates (u_2, v_2). Then, similarly to the derivation of (16) we can obtain the following equation to describe L_2 in CCS 2:

$$\begin{bmatrix} X_2 & Y_2 & Z_2 \end{bmatrix}^{\mathrm{T}} = R_2 \begin{bmatrix} \cos\theta_2 & \sin\theta_2 & \tan\alpha_2 \end{bmatrix}^{\mathrm{T}}, \tag{18}$$

where $R_2 = \sqrt{X_2^{\,2} + Y_2^{\,2}}$. As illustrated in Fig. 8, we can transform the light ray L_2 from CCS 2 to the coordinate system *X*-*Y*-*Z* by rotating the ray through the angle β_2 and translating it by the vector $[D\ 0\ 0]^T$. As a result, the transformed light ray L_2 goes through (*D*, 0, 0) with its directional vector d_2 being

$$d_2 = \begin{bmatrix} \cos\beta_2 & 0 & -\sin\beta_2 \\ 0 & 1 & 0 \\ \sin\beta_2 & 0 & \cos\beta_2 \end{bmatrix} \begin{bmatrix} \cos\theta_2 \\ \sin\theta_2 \\ \tan\alpha_2 \end{bmatrix}. \tag{19}$$

We now have two light rays L_1 and L_2 both going through the space point *P*. If everything including the works of system setup, camera calibration, and feature detection is conducted

accurately without incurring errors, these two lines should intersect perfectly at one point which is just P. But unavoidably, various errors of imprecision always exist so that the intersection point does not exist. One solution to this problem is to estimate the coordinates of point P as those of the *midpoint* P_m on the *shortest line segment* between the two light rays L_1 and L_2, as illustrated in Fig. 9.

To obtain this solution, let d be the vector perpendicular to d_1 and d_2 as shown in Fig. 9, which can be expressed as $d_1 \times d_2$, where $\times$ denotes the cross-product operator. Since Q_1 is on light ray L_1, its coordinates (X_1, Y_1, Z_1) can be expressed as

$$\begin{bmatrix} X_1 & Y_1 & Z_1 \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^{\mathrm{T}} + \lambda_1 d_1 \tag{20}$$

where λ_1 is an unknown scaling factor. Let S_1 be the plane containing P_2, Q_1 and Q_2. As illustrated in Fig. 9, the normal vector n_1 of plane S_1 is $d_2 \times d$, or equivalently, $d_2 \times (d_1 \times d_2)$. Since P_2 and Q_1 are both on this plane, we get to know that the vector P_2Q_1 is perpendicular to n_1. This fact can be expressed by

$$\overrightarrow{P_2Q_1} \cdot n_1 = \left(\begin{bmatrix} X_1 & Y_1 & Z_1 \end{bmatrix}^{\mathrm{T}} - \begin{bmatrix} D & 0 & 0 \end{bmatrix}^{\mathrm{T}} \right) \cdot \left(d_2 \times \left(d_1 \times d_2 \right) \right) = 0 .$$

Combining the above equality with (20), we get

$$\left(\lambda_1 d_1 - \begin{bmatrix} D & 0 & 0 \end{bmatrix}^{\mathrm{T}} \right) \cdot \left(d_2 \times \left(d_1 \times d_2 \right) \right) = 0$$

from which the unknown scalar λ_1 can be solved to be

$$\lambda_1 = D \frac{\left(d_2 \times \left(d_1 \times d_2 \right) \right) \cdot \mathbf{e_1}}{\left(d_2 \times \left(d_1 \times d_2 \right) \right) \cdot d_1}, \tag{21}$$

where $\mathbf{e}_1 = [1\ 0\ 0]^{\mathrm{T}}$. Similarly, since Q_2 is on light ray L_2, the coordinates (X_2, Y_2, Z_2) of Q_2 can be expressed as

$$\begin{bmatrix} X_2 & Y_2 & Z_2 \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} D & 0 & 0 \end{bmatrix}^{\mathrm{T}} + \lambda_2 d_2 \tag{22}$$

where λ_2 is another unknown scaling factor. Let S_2 be the plane containing P_1, Q_1 and Q_2. The normal vector n_2 of this plane is $d_1 \times d = d_1 \times (d_1 \times d_2)$. Since P_1 and Q_2 are both on this plane, the vector P_1Q_2 is known to be perpendicular to n_2, leading to the following equality:

$$\overrightarrow{P_1Q_2} \cdot n_1 = \left(\begin{bmatrix} X_2 & Y_2 & Z_2 \end{bmatrix}^{\mathrm{T}} - \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^{\mathrm{T}} \right) \cdot \left(d_1 \times \left(d_1 \times d_2 \right) \right) = 0 .$$

Combining the above equality with (22), we get

$$\left(\begin{bmatrix} D & 0 & 0 \end{bmatrix}^{\mathrm{T}} + \lambda_2 d_2 \right) \cdot \left(d_1 \times \left(d_1 \times d_2 \right) \right) = 0 ,$$

which can be solved to get the unknown scalar λ_2 as

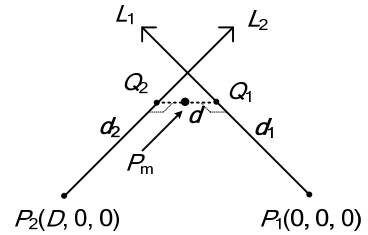


Fig. 9. Illustration of deriving the middle point P_m of light rays L_1 and L_2.

$$\lambda_2 = -D \frac{\left(d_1 \times \left(d_1 \times d_2 \right) \right) \cdot \mathbf{e_1}}{\left(d_1 \times \left(d_1 \times d_2 \right) \right) \cdot d_2} . \tag{23}$$

Since P_m is the midpoint between Q_1 and Q_2, the coordinates (X_m, Y_m, Z_m) of P_m can be expressed as

$$\begin{bmatrix} X_m \\ Y_m \\ Z_m \end{bmatrix} = \frac{1}{2} \left(\begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \end{bmatrix} + \begin{bmatrix} X_2 \\ Y_2 \\ Z_2 \end{bmatrix} \right),$$

which, when combined with (20), (21), (22), and (23), leads to the following estimation result for use as the desired 3D data of space point P:

$$\begin{bmatrix} X_m \\ Y_m \\ Z_m \end{bmatrix} = \frac{1}{2} D \left(\mathbf{e}_1 - \frac{\left(d_2 \times \left(d_1 \times d_2 \right) \right) \cdot \mathbf{e}_1}{\left(d_1 \times \left(d_1 \times d_2 \right) \right) \cdot d_2} d_1 - \frac{\left(d_1 \times \left(d_1 \times d_2 \right) \right) \cdot \mathbf{e}_1}{\left(d_1 \times \left(d_1 \times d_2 \right) \right) \cdot d_2} d_2 \right), \tag{24}$$

where $\mathbf{e}_1 = [1\ 0\ 0]^{\mathrm{T}}$ and D is the baseline to be determined.

C. Finding Baseline D

To compute the baseline D, we make use of a fact about triangulation in binocular computer vision: the 3D data can be determined *up to a scale* without knowing the value of the baseline D [26]. This fact can also be seen from (24), where the baseline D is a scaling factor of the computed 3D data.

Specifically, within the omni-images taken of the user standing in front of the two cameras as mentioned previously, we extract two points on the head and the feet of the user, respectively. Let P_{head} and P_{foot} denote their real 3D data, respectively. On the other hand, as stated previously, we can compute the 3D data up to a scale of the two points, which we denote as P'_{head} and P'_{foot}, respectively, using (24) with the term D in it ignored. Then, the relations between the data P_{head}, P_{foot}, P'_{head}, and P'_{foot} can be expressed as

$$P_{\mathrm{head}} = D \cdot P'_{\mathrm{head}}, \text{ and } P_{\mathrm{foot}} = D \cdot P'_{\mathrm{foot}},$$

where D is the *actual* baseline value. Let H' be the Euclidean distance between P'_{head} and P'_{foot}; and let H be the real distance between P_{head} and P_{foot}, which is just the known height of the user. Then, the baseline D can finally be computed as $D = H/H'$.

After finding the baseline D, the system parameters are now all adapted. To sum up, the three steps of the proposed adaptation method are briefly described as follows. First, the included angle ϕ between the two optical axes are determined using space line features as discussed in Section V. Then, by asking the user to stand at the middle point in front of the two omni-cameras, the orientation angles β_1 and β_2 of the two cameras are calculated as described in Section VI-A. Finally, the baseline D is calculated using the height H of the user as described in this section. An overview of the proposed adaptation method is also described in Algorithm 1.

VII. Experimental Results and Discussions

In this section, we describe first how we calibrate the omni-cameras to obtain their intrinsic parameters in Section

VII-A. Then, we present several experimental results to show the feasibility, reliability, and accuracy of the proposed line detection method, the system setup adaptation method, and the 3D computation process in Section VII-B through VII-D.

A. Omni-Camera Calibration

In the first step, the lens center and the focal length of the perspective camera should be calibrated. As illustrated in Figs. 10(a) and 10(b), the mirror boundary, appearing as a circle in each captured omni-image, was extracted to robustly estimate the camera center and the focal length according to [23]. Specifically, we found a circle to fit the circular mirror boundary like that appearing in Fig. 10(b), and defined the camera center as the center of the fitting circle. Also, as shown in Fig. 10(a), we derived the camera's focal length f, according to the properties of similar triangles and the rotational invariance of the omni-camera [27][28], as

$$f = M\frac{r}{R} \tag{25}$$

where M is the distance from the lens center to the camera center O_m, R is the radius of the mirror base in the real-world space, and r is the radius of the mirror base in the taken image. The measured values in our experiments are R = 4.0 cm, M = 8.6 cm, and r = 243 pixels for both cameras, from which the focal lengths f were derived to be 522.45 according to (25).

Next, we solve ε from (1) to get

$$\varepsilon = \frac{\sec\beta + \sec\alpha}{\tan\beta - \tan\alpha}.$$

Combining the above equality with (1) and (2), we can get

$$\varepsilon = \frac{\sqrt{1+\frac{f^2}{u^2+v^2}} + \sqrt{1+\frac{Z^2}{X^2+Y^2}}}{\frac{f}{\sqrt{u^2+v^2}} - \frac{Z}{\sqrt{X^2+Y^2}}}. \tag{26}$$

The above equation shows that, if we have a landmark point with known image coordinates (u, v) and known camera coordinates (X, Y, Z), then the eccentricity ε can be calculated.

Although the eccentricity ε is theoretically a constant value, we found in this study that we can achieve better accuracy in 3D data computation if a linear polynomial can be used to describe ε. The reason is that such a polynomial can be used to cope with some types of errors, including the radial distortion of the perspective camera's lens, the imprecise measurements coming from the calibration process, and the manufacturing imprecision of the hyperboloidal mirror shape. Accordingly, we propose the following first-order equation to describe the eccentricity ε, which comes from a functional expansion of ε with respect to the mirror's radius r according to the rotational invariance property as used in several studies [27][28]:

$$\varepsilon = g \cdot r + h, \tag{27}$$

where g and h are two coefficients, and r is as defined in (2).

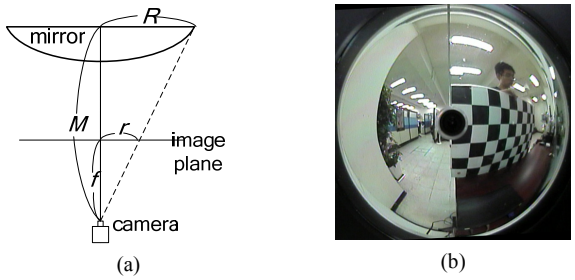


Fig. 10. Illustration of omni-camera calibration. (a) Relationship between mirror and image plane. (b) An omni-image of a calibration board.

In our experiments, a calibration board as shown in Fig. 10(b) was designed and put in front of the omni-camera. Each cross point P_i on the board was taken as a landmark point as stated in Algorithm 1, and used to calculate the eccentricity ε_i by (26). After the values ε_i corresponding to all the landmark points were derived according to (26), the coefficients g and h in (27) were computed finally using a Levenberg–Marquardt algorithm [29] to be –0.0022 and 1.9211, respectively.

To demonstrate the effectiveness of the first-order approximation method, we conduct two experiments as follows. In these experiments, we measure the 3D data of the 60 landmarks on a calibration board, and compute the 3D measurement errors. The average 3D measurement error is 6.3% with a standard deviation of 1.4% when using a constant eccentricity, which is reduced to an average error of 1.9% with a standard deviation of 0.71% when using the first-order approximation. This shows the effectiveness of the first-order approximation method for computing the eccentricity ε. It is noted here that the first-order coefficient g is supposed to be small since it should be a constant in theory. Otherwise, it means any of the three possibilities: (1) the measurements in the calibration are not accurate enough; (2) the lens of the perspective camera is heavily distorted; or (3) the mirror is not of a good hyperboloidal shape.

B. Space Line Detection Ability

In Sections IV-A, IV-D, and IV-B, three techniques of improvements on increasing the detection ability and reliability of the proposed Hough-based space line detection method have been proposed, which are called *parameterization*, *peak cell extraction*, and *accumulation*, respectively, henceforth. Some comparisons are provided here to show the effectiveness of the proposed improvement techniques. About parameterization, we compare the effect of our technique with that proposed in [16]. About peak cell extraction, we compare our technique using the proposed filter with a traditional method. And about accumulation, we compare the adaptive thresholding technique we propose with a traditional accumulation method [31][32]. Accordingly, four different space line detection experiments have been designed, which are listed in Table I.

The input omni-image of the four experiments is shown in Fig. 11(a). In each experiment, at first we found the edges in the omni-image to get those shown in Fig. 11(b). We then applied the Hough-based space line detection method to find 50 space

lines. And finally we drew the detected lines on the omni-image. The results of the four experiments are shown in Figs. 11(c) through 11(f), respectively.

As shown in Fig. 11(c), since the parameterization proposed in [16] has a singularity when $n = 0$, only space lines near the periphery region can be detected reliably. In contrast, when using the proposed parameterization technique, space lines in the center region can be detected as shown in Fig. 11(d), but are quite crowded. After using the proposed peak cell extraction technique, the detected lines are more separated as shown in Fig. 11(e). Finally, after the proposed adaptive thresholding technique was applied in the last experiment, the detection result was improved further, yielding lines with more diversified directions, as shown in Fig. 11(f).

To summarize, the proposed techniques have at least three advantages over the traditional ones. First, the proposed parameterization technique has no singularity problem, and the range of the Hough space is fixed within [−1, 1]. In contrast, the method proposed in [16] has a singularity when $n = 0$, and the range of the parameters goes from negative infinity to positive one. Second, space lines can be extracted more effectively by the proposed peak cell extraction technique. Third, the projection curve corresponding to the Hough cells in a cell support is of equal widths everywhere, which further improves the detection result.

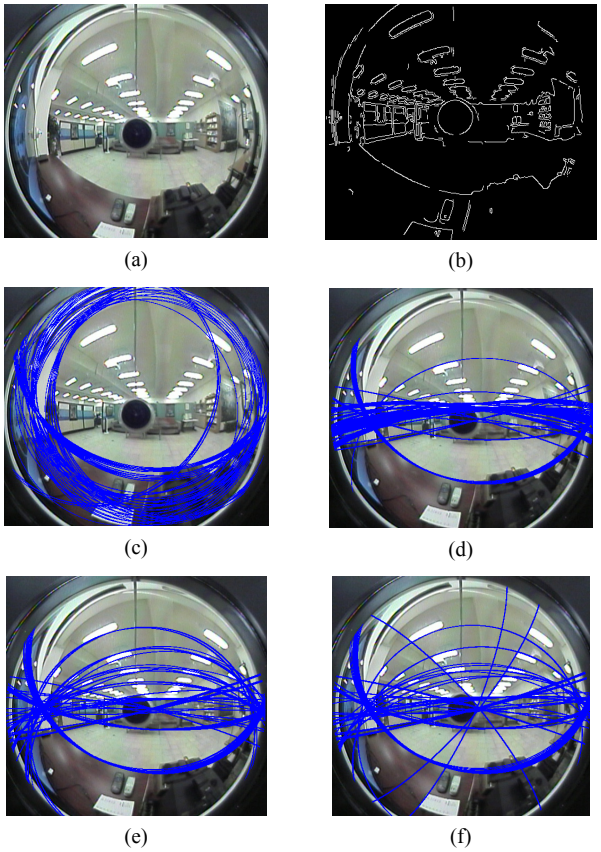


Fig. 11. The space line detection results of the four different experiments. (a) Input omni-image. (b) Edge detection results. (c)-(f) 50 space lines detected in Experiments 1 through 4, respectively, Experiement 4 based on use of all proposed improvement methods shows the best result.

TABLE I
FOUR DIFFERENT SPACE LINE DETECTION EXPERIMENTS

	parameterization	peak cell extraction	accumulation
Exp. 1	proposed in [16]	traditional	traditional
Exp. 2	proposed	traditional	traditional
Exp. 3	proposed	proposed	traditional
Exp. 4	proposed	proposed	proposed

C. *Adaptation Ability*

Some experimental results are given here to show the adaptation ability under different cameras and environments. Two types of cameras were used, which are perspective cameras and catadioptric omni-cameras, and three different environments were considered, which are a corridor, a hall, and a room, as shown in Figs. 12(a) through 12(c).

Four different experiments were conducted: Experiment 1 is conducted in the corridor with omni-cameras; Experiment 2 in the hall with omni-cameras; Experiment 3 in the room with omni-cameras; and Experiment 4 also in the room but with perspective cameras. In each experiment, the two cameras were oriented in different angles (i.e., −30°, −15°, 0°, 15°, and 30°). Fifty space line features were first extracted as proposed in Section IV. Then, the angle ϕ was automatically calculated using these lines as proposed in Section V. The results are shown in Fig. 12(d). The X-axis specifies the ground truth of the angle ϕ, and the Y-axis specifies the absolute error of the calculated angle ϕ.

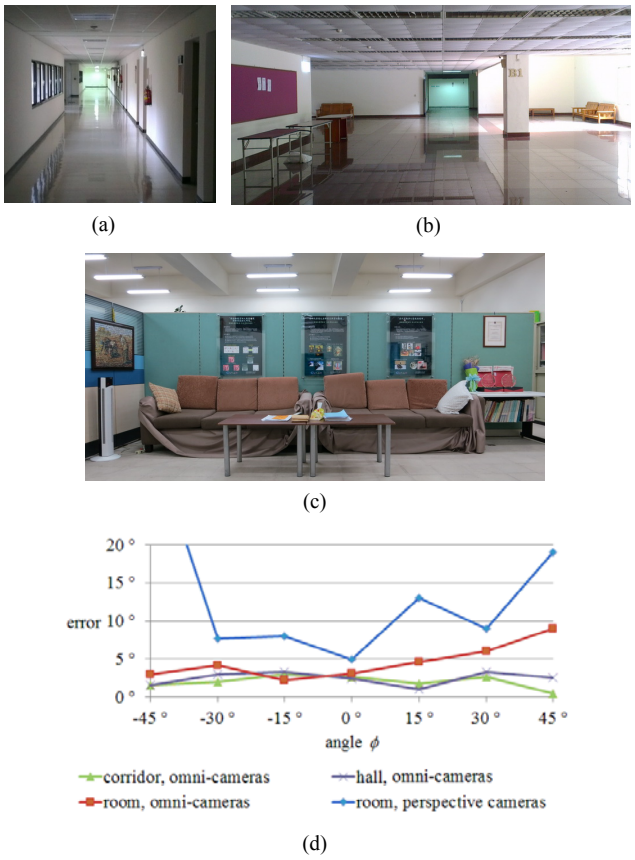


Fig. 12. Experimental results under different cameras and environments. (a) A corridor. (b) A hall. (c) A room. (d) Adaptation results of angle ϕ.

In Experiments 1 and 2, since the lines in the corridor and hall are relatively simple and obvious, the adaptation result is accurate with errors of about 2° as shown by the green and purple curves in Fig. 12(c). Also, since we use omni-cameras in these experiments, the lines can still be captured even when the two cameras were oriented with a large angle. Thus, the adaptation result remains accurate when the angle ϕ is large. In Experiment 3, since the space lines in the room are more complicated, the adaptation becomes more difficult. However, since the omni-cameras can capture a large field of view of the environment, a plenty number of space lines can be captured. Therefore, the adaptation result is accurate as well, with errors of about 4° as shown by the red curve in Fig. 12(c). In contrast, the adaptation errors are about 10° when perspective cameras were used, as shown by the blue curve in Fig. 12(c), and they become unacceptable (larger than 20°) when the included angle ϕ is large. These experimental results show the feasibility of the proposed adaptation methods, as well as the power of the omni-cameras in the automatic adaptation process.

D. Adaptation and 3D Acquisition Ability

A series of experiments are conducted to test the *adaptation ability* and the *3D acquisition precision* in the *room* environment shown in Fig. 12(b). In each of the experiments, the two cameras were placed at a distance about 180cm to each other, and both were oriented *randomly* within the range of ±40°. After the cameras were set up, two omni-images of the environment were captured as shown, for example, in Figs. 13(a) and 13(d), respectively, and used to calculate the included angle ϕ according to Step 5 of Algorithm 1. Next, a user was asked to stand in the middle region in front of the two cameras, as shown in Figs. 13(b) and 13(e), to calculate the orientation angles β_1 and β_2 and the baseline D according to Step 6 of Algorithm 1. After these adaptation tasks were done, a board with 60 landmarks was held by the user, as shown in Figs. 13(c) and 13(f), to test the precision of the resulting 3D computation.

In these experiments, three different *degrees of adaptation* were implemented and the corresponding results compared: (1) no adaptation was conducted with the camera orientations and baseline set to be $\beta_1 = \beta_2 = 0°$ and $D = 180$ cm (D is the ground-truth value); (2) the left omni-camera was set up to face forward with the values $\beta_1 = 0°$, $D = 180$cm, and β_2 adapted to be $-\phi$; and (3) all the parameters β_1, β_2, and D were adapted according to the proposed method. Denoting (X_i, Y_i, Z_i) as the ground-truth location of a landmark point, and (X_i', Y_i', Z_i') as the calculated location, we define the 3D error E of each landmark point as

$$E = \sqrt{(X_i - X_i')^2 + (Y_i - Y_i')^2 + (Z_i - Z_i')^2} \Big/ \sqrt{X_i^2 + Y_i^2 + Z_i^2} \quad . \quad (28)$$

The comparison results are shown in Fig. 14 in which the vertical axis specifies the average of the 3D errors, and the horizontal axis specifies the *system orientation angle* which is defined as the *maximum* of the two orientation angles β_1 and β_2.

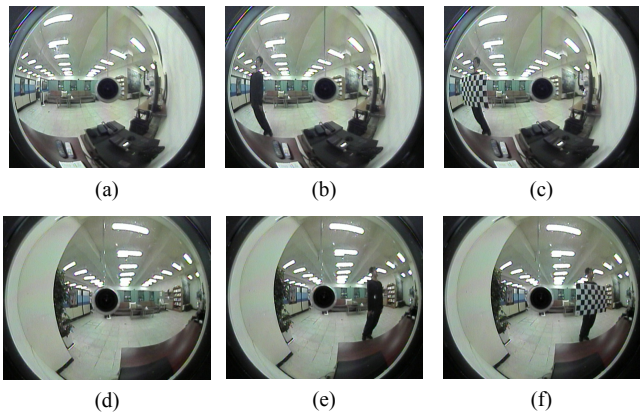


Fig. 13. Sample omni-images of an experiment. (a)(d) Taking a shot of the environment to calculate ϕ. (b)(e) A user standing in the middle region in front of the cameras to calculate baseline D and orientation angles β_1 and β_2. (c)(f) A board held by the user to test the 3D computation precision.

As can be seen from Figs. 14(a) and 14(b), when no parameter is adapted with the results shown by the blue curve, the 3D errors are seen to become larger as the orientation angle becomes larger, showing the necessity of an automatic system adaptation process. When only the orientation β_2 of the right omni-camera is adapted with the result shown by the red curve, it is observed that the 3D errors are sometimes lower but vary largely. This results from the fact that the left omni-camera is *assumed* to face forward in this case. Thus, if the left omni-camera is *actually* placed to face forward in the experiment, the error measure is lowered; otherwise, the error is large as expected. Finally, when all the parameters β_1, β_2 and D are adapted with the results shown by the purple curve, the 3D errors are lower than 8% even when the system orientation angle is large. This shows the feasibility, reliability, and validity of the proposed system adaptation method.

It is noted that these 3D measurements are calculated under a certain *unintended* inaccurate system setup. For example, it is required that the two omni-camera stands be adjusted to be at an identical height, but there might still exist a small distance, say 1*cm*, between the heights of the two stands. Similarly, although the optical axes are assumed to be parallel to the *XZ*-plane, a small angle, say 1°, might be included between the optical axes and the *XZ*-plane. To see the effect of such unintended system setup inaccuracy, a plot of the average 3D errors resulting from a series of planned inaccurate setups is drawn in Fig. 14(c). As can be seen, at the reasonable setup errors of 1 cm in height and 2° in included angle, the average 3D error is 2.805%, which is tolerable in real-time game playing according to our experimental experience.

Using the proposed vision system, we have also created a game application in our experiments, which allows a user to play a 3D maze game, as illustrated by Fig. 15. The game is played mainly by the use of a finger with a yellow cot as a cursor, controlling the avatar going around and up and down in the maze to reach the destination. The 3D position of the simulated cursor is computed by analyzing the omni-image pair to detect the feature point of the finger cot and calculating its

3D position by the proposed method. Fig. 15 shows the game playing environment and three views of the 3D maze from different directions at a certain moment. When playing the game, the avatar moves towards the correct direction and responds as quickly as the player's finger moves. This realtime effect comes mainly from the 3D computations all by the uses of the *analytic* formulas derived previously. It is noted in game playing that, if the player stands too far from the cameras, it will be too hard to detect the feature point on his/her finger, which influence the 3D calculations. Also, since there is a blind circle in omni-images, the 3D tracking process will fail momentarily. Otherwise, in normal cases, the avatar can be controlled by the player easily, which shows the feasibility of the proposed system for game playing and other similar applications.

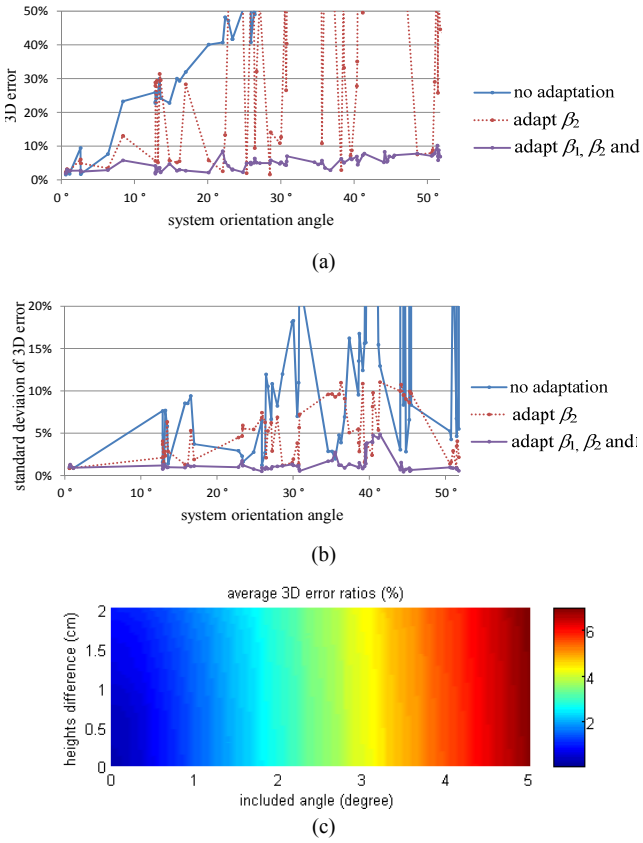


Fig. 14. Experimental results of three different degrees of adaptations. (a) The 3D errors. (b) The standard deviations of the 3D errors. The proposed adaptation methods yield the best results as shown by the purple curves. (c) The 3D errors resulting from the uninteded inaccurate setups.

VIII. Conclusions

A new two-omni-camera stereo vision system for general 3D vision applications with a capability of automatic adaptation to any camera setup has been proposed. The adaptation process yields the values of the two omni-cameras' orientations and distance (baseline), from which the 3D data of space feature points can be computed precisely. Experimental results show the feasibility of the proposed system. In contrast, the cameras' orientations and distance of the conventional binocular vision system are all fixed because of its nonadjustable configuration.

The proposed vision system has several advantages over conventional systems. First, the user can interact with the system within a *wide* area because the proposed system uses two omni-cameras, instead of traditional projective cameras, to capture omni-images which cover large fields of view. This is a desired property for many applications. For example, it can be used in exhibitions to interact with humans in a large area, in 3D indoor surveillance of large public spaces, or in future virtual sporting environments where people are walking or running in a wide area. Today, commercial products also try to solve the small field-of-view problems of conventional cameras. For example, the Microsoft Kinect uses a motorized tilt mechanism to track the user's activities to overcome this problem [5]. In contrast, the proposed system does not suffer from this problem. Second, the proposed system can be set up flexibly, and so is appropriate for more real applications and more convenient for non-technical users. Third, the proposed system yields better precision in computed 3D data than traditional short-baseline stereo systems. This comes from the merit of the structure of the proposed system — the two omni-cameras are affixed to two independent camera stands which may be placed farther away from each other. It is noted that the proposed system is less applicable in environments with natural scenes as backgrounds where horizontal parallel lines are fewer for use by the system.

Future studies may be directed to applying the proposed system to more human-machine interaction activities.

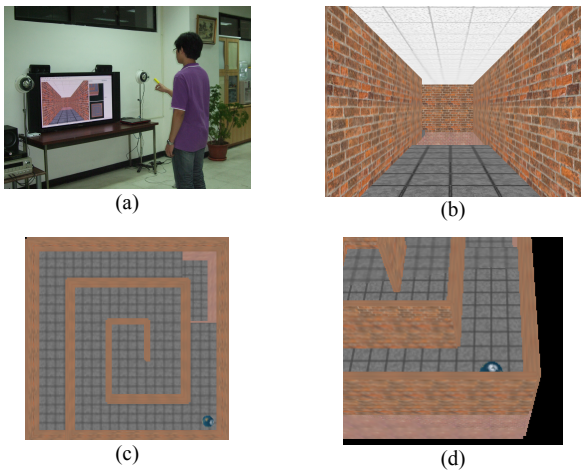


Fig. 15. Use of proposed vision system for a 3D maze game. (a) A player of the game wearing a yellow finger cot with the two omni-cameras. (b)(c)(d) Three different views of the 3D maze game at a certain moment.

Acknowledgment

A preliminary version of this paper appeared in *Advances in Multimedia Modeling*, *LNCS*, Vol. 6523, K. T. Lee, W. H. Tsai, H. Y. M. Liao, T. Chen, J. W. Hsieh and T. T. Tseng. (eds.), Springer, Berlin/Heidelberg, Germany, pp. 193-205, 2011.

References

[1] Z. Y. Zhou, A. D. Cheok, Y. Qiu, and X. Yang, "The Role of 3-D sound in human reaction and performance in augmented reality environments,"

IEEE Trans. on Systems, Man, and Cybernetics–Part A: Systems and Humans, vol. 37, no. 2, pp. 262–272, March 2007.

[2] B. J. Tippetts, D. J. Lee, J. K. Archibald, and K. D. Lillywhite, "Dense disparity real-time stereo vision algorithm for resource-limited systems," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 21, no. 10, pp. 1547–1555, Oct. 2011.

[3] Y. Sun, X. Chen, M. Rosato, and L. Yin, "Tracking vertex flow and model adaptation for three-dimensional spatiotemporal face analysis," *IEEE Trans.on Systems, Man and Cubernetics–Part A: Systems and Humans*, vol. 40, no. 3, pp. 461–474, May 2010.

[4] K. Li, Q. Dai, W. Xu, J. Yang, and J. Jiang, "Three-dimensional motion esimtation via matrix completion," *IEEE Trans. on Systems, Man, and Cybernetics–Part B: Cybernetics*, vol. 42, no. 2, pp. 539–551, April 2012.

[5] Wikipedia contributors, "Kinect," *Wikipedia, The Free Encyclopedi*a, http://en.wikipedia.org/wiki/Kinect (accessed Dec. 15, 2010).

[6] S. Laakso and M. Laakso, "Design of a body-driven multiplayer game system," *Computers in Entertainment (CIE)*, vol. 4, no. 4, Article 4C, Oct.-Dec. 2006.

[7] J. J. Magee, M. Betke, J. Gips, M. R. Scott, and B. N. Waber, "A human–computer interface using symmetry between eyes to detect gaze direction," *IEEE Trans. on Systems, Man, and Cybernetics–Part A: Systems and Humans*, vol. 38, no. 6, pp. 1248–1261, Nov. 2008.

[8] X. Zabulis, T. Sarmis, D. Grammenos and A. A. Argyros, "A multicamera vision system supporting the development of wide-area exertainment applications," *IAPR Conf.on Machine Vision Applications (MVA 2009)*, Yokohama, Japan, May 20-22, 2009, pp. 269-272.

[9] J. Starck, A. Maki, S. Nobuhara, A. Hilton, and T. Matsuyama, "The multiple-camera 3-D production studio," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 19, no. 6, June 2009.

[10] S. Sefvic and S. Ribaric, "Determining the absolute orientation in a corridor using projective geometry and active vision," *IEEE Trans. on Industrial Electronics*, vol. 48, no. 3, pp. 696–710, June 2001.

[11] R. Carelli, R. Kelly, O. H. Nasisi, C. Soria, and V. Mut, "Control based on perspective lines of a non-holonomic mobile robot with camera-on-board," *Int'l Journal of Control*, vol. 79, no. 4, pp. 362–371, 2006.

[12] X. Ying and H. Zha, "Simultaneously calibrating catadioptric camera and detecting line features using Hough transform," *Proc. IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems*, pp. 412–417, Aug. 2005.

[13] X. Ying, "Catadioptric Camera Calibration Using Geometric Invariants," *Proc. IEEE Int'l Conf. on Computer Vision*, vol. 2, pp. 1351–1358, Oct. 2003.

[14] F. Duan, F. Wu, M. Zhou, X. Deng, and Y. Tian, "Calibrating effective focal length for central catadioptric cameras using one space line," *Pattern Recognition Letters*, vol. 33, pp. 646-653, 2012.

[15] R. G. von Gioi, J. Jakubowicz, J.-M. Morel and G. Randall, "LSD: A fast line segment detector with a false detection control," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 32, no. 4, pp. 722–732, April 2010.

[16] C. J. Wu and W. H. Tsai, "An omni-vision based localization method for automatic helicopter landing assistance on standard helipads," *Proc. Int'l Conf. on Computer and Automation Engineering*, Singapore, pp. 327–332, 2010.

[17] S. J. Maybank, S. Ieng, and R. Benosman, "A Fisher-Rao metric for paracatadioptric images of lines," *Int'l Journal of Computer Vision*, vol. 99, no. 2, pp. 147–165, 2012.

[18] K. Yamazawa, Y. Yagi and M. Yachida, "3D line segment reconstruction by using hyperomni vision and omnidirectional Hough transforming," *Proc. Int'l Conf. on Pattern Recognition*, vol. 3, IEEE Computer Society, Washington, DC, USA, pp.3487–3490, 2000.

[19] S. T. Barnard, "Interpreting perspective images," *Artifical Intelligence*, vol. 21, pp. 435–462, 1983.

[20] B. Li, K. Peng, X. Ying, and H. Zha, "Vanishing point detection using cascaded 1D Hough Transform from single images," *Pattern Recognition Letters*, vol. 33, pp. 1-8, 2012.

[21] C. Geyer and K. Danilidis, "Catadioptric projective geometry," *Int'l Journal of Computer Vision*, vol. 45, no. 3, pp. 223–243, 2001.

[22] H. Ukida, N. Yamato, Y. Tanimoto, T. Sano and H. Yamamoto, "Omni-directional 3D measurement by hyperbolic mirror cameras and pattern projection," *Proc. 2008 IEEE Conf. on Instrumentation and Measurement Technology*, Victoria, BC, Canada, pp. 365–370, May 2008.

[23] J. Fabrizio, J. P. Tarel, and R. Benosman, "Calibration of panoramic catadioptric sensors made easier," *Proc. IEEE Workshop on Omnidirectional Vision*, pp. 45–52, 2002.

[24] R. I. Hartley and P. Sturm, "Triangulation," *Proc. ARPA Image Understanding Workshop*, pp. 957–966, 1994.

[25] R. C. Gonzalez and R.E. Woods, *Digital Image Processing*, New Jersey: Prentice Hall, 2002.

[26] B. Cyganek, J. P. Sievert, *Introduction to 3D Computer Vision Techniques and Algorithms*, Wiley, John & Sons, Incorporated, 2009.

[27] D. Scaramuzza, "Omnidirectional vision: from calibration to robot motion estimation", ETH Zurich, PhD Thesis no. 17635. Zurich, February 22, 2008.

[28] D. Scaramuzza, A. Martinelli, and R. Siegwart, "A flexible technique for accurate omnidirectional camera calibration and structure from motion," *Proc. IEEE Int'l Conf. on Computer Vision Systems*, pp. 45–52, Jan 2006.

[29] K. Levenberg, "A method for the solution of certain non-linear problems in least squares," *Quarterly of Applied Mathematics*, pp. 164–168, 1944.

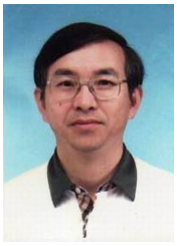
[30] J. Illingworth and J. Kittler, "A survey of the Hough transform," *Comput. Vision, Graph. Image Processing*, vol. 44, pp. 87–116, 1988.

[31] R. D. Duda and P. E. Hart, "Use of the Hough transform to detect lines and curves in pictures," *Communications of the ACM*, vol. 15, pp. 11–15, 1972.

[32] N. Bennett, R. Burridge, N. Saito, "A method to detect and characterize ellipses using the Hough transform," *IEEE Trans.on Pattern Analysis and Machine Intelligence*, vol. 21 no.7, pp. 652–657, 1999.

[33] S. E. Shih, and W. H. Tsai, "A new two-omni-camera system with a console table for versatile 3D vision applications and its automatic adaptation to imprecise camera setups," *Advances in Multimedia Modeling (MMM 2011) - Lecture Notes in Computer Science (LNCS)*, Vol. 6523, K. T. Lee, W. H. Tsai, H. Y. M. Liao, T. Chen, J. W. Hsieh and T. T. Tseng. (eds.), Springer, Berlin/Heidelberg, Germany, pp. 193-205, 2011.

Shen-En Shih received the B.S. degree in computer science from National Chiao Tung University, Taiwan, in 2009, and is currently pursuing the Ph.D. degree at the College of Computer Science, National Chiao Tung University. He has been a research assistant at the Computer Vision Laboratory in the Department of Computer Science at National Chiao Tung University from August 2009. His current research interests include computer vision, image processing, human-machine interfacing, and stereo vision.



Wen-Hsiang Tsai received the B.S. degree in EE from National Taiwan University, Taiwan, in 1973, the M.S. degree in EE from Brown University, USA in 1977, and the Ph.D. degree in EE from Purdue University, USA in 1979. Since 1979, he has been with National Chiao Tung University (NCTU), Taiwan, where he is now a Chair Professor of Computer Science. At NCTU, he has served as the Head of the Dept. of Computer Science, the Dean of General Affairs, the Dean of Academic Affairs, and a Vice President. From 1999 to 2000, he was the Chair of the Chinese Image Processing and Pattern Recognition Society of Taiwan, and from 2004 to 2008, the Chair of the Computer Society of the IEEE Taipei Section in Taiwan. From 2004 to 2007, he was the President of Asia University, Taiwan. Dr. Tsai has been an Editor or the Editor-in-Chief of several international journals, including *Pattern Recognition*, the *International Journal of Pattern Recognition and Artificial Intelligence*, and the *Journal of Information Science and Engineering*. He has published 144 journal papers and 227 conference papers received many awards, including the Annual Paper Award from the Pattern Recognition Society of the USA; the Academic Award of the Ministry of Education, Taiwan; the Outstanding Research Award of the National Science Council, Taiwan; the ISI Citation Classic Award from Thomson Scientific, and more than 40 other academic paper awards from various academic societies. His current research interests include computer vision, information security, video surveillance, and autonomous vehicle applications. He is a Life Member of the Chinese Pattern Recognition and Image Processing Society, Taiwan and a Senior Member of the IEEE.