Efficient Omni-image Unwarping using Geometric Symmetry

Zhihui Xiong[1,2], Irene Cheng[2], Anup Basu[2], Wei Wang[1], Wei Xu[1] and Maojun Zhang[1]

[1]*College of Information System and Management, National University of Defense Technology*

Changsha, 410073 People's Republic of China

[2] *University of Alberta, Edmonton, AB T6G 2E8, Canada*

Correspondence author: Anup Basu

Phone: 001-780-492-3330 Fax: 001-780-492-1071 E-mail: basu@ualberta.ca

Abstract In order to save storage space of a pano-mapping table used in omni-image unwarping, a geometric symmetry method is proposed. First of all, this method partitions a 360 degree omni-image into eight 45 degree omni-image sectors. Then, we partition the pano-mapping table into eight regions accordingly, with each pano-mapping table region corresponding to exactly one omni-image sector. We analyze the geometric symmetry relationship among these omni-image sectors and pano-mapping table regions. We find that if we know the mapping data in any one pano-mapping table region, it is easy to calculate the mapping data of the other seven pano-mapping table regions. Thus, in the final step, we perform omni-image unwarping based on only one pano-mapping table region, which reduces pano-mapping table size by seven-eighths. Reducing the pano-mapping size is very useful for implementing omni-image unwarping in embedded systems. Experiments on TI DSP based embedded systems indicate that the proposed method reduces seven-eighths of pano-mapping table size, and improves the unwarping speed by a factor of 2.74.

Keywords *Catadioptric imaging, omni-directional image, computer vision, image processing, panorama*

1. Introduction

Catadioptric omni-directional imaging systems (or simply omni-cameras) are becoming popular in applications that need wide field of view (FOV), such as robot vision[1-5], visual surveillance[6-8] and 3D scene reconstruction[9]. The images captured with omni-cameras are usually called omni-images, as shown in Fig. 8(a). Unfortunately, although omni-images provide as wide as 360 degree FOVs, they are warped. Many applications require transforming omni-images into unwarped images (or panoramic images, as shown in Fig. 8(b)) that are suitable for human perception, this procedure is called omni-image unwarping.

In the literature, there are roughly two kinds of omni-image unwarping approaches [1] ─ ray-trace coordinate calculation and concentric circle approximation. The former approach [10-13] calculates the pixel position relationship between omni-images and their corresponding panoramic images. This method cal-

culates panorama images with both high precision and less deformation. However, it needs information on omni-camera parameters, including focal length of the lens, coefficients of mirror surface shape equations, etc., which in some situations is difficult to know or estimate. The concentric circle approximation approach [14] treats omni-images as a series of concentric circles, and maps these concentric circles to the same-length horizontal rows in panoramic images via trigonometric transformations.

Jeng, Tsai and Wu [15-16] propose a pano-mapping table method for unwarping omni-images taken by almost any kind of omni-cameras, without requiring prior knowledge about the camera parameters in advance. In this work, a pano-mapping table is regarded as a summary of omni-camera parameters, and this pano-mapping table is created only once for a specific omni-camera. However, the pano-mapping table is very large in size, so it needs significant memory space to store the table. This problem is especially serious in embedded system implementation of omni-cameras, since there is limited memory space in embedded systems, such as in a DSP and or a FPGA.

We propose reducing seven-eighths of a pano-mapping table based on geometric symmetry of omni-images and pano-mapping table. The ideas and procedures behind this method include: a) Treating the omni-image as a series of concentric circles, and partitioning the omni-image into eight geometrically symmetric regions in radial directions. That is, we partition the 360 degree of an omni-image into eight omni-image sectors uniformly, with each omni-image sector being 45 degree. b) We partition the pano-mapping table into eight regions accordingly, with each pano-mapping table region corresponding to exactly one omni-image sector. c) By deducing and establishing the geometrical symmetry relationship among these omni-image sectors and pano-mapping table regions, we can perform omni-image unwarping based on only one pano-mapping table region. This is because, based on the geometric symmetry relationship, the mapping data of the remaining seven pano-mapping table regions can be calculated with a few simple CPU operations.

Reducing pano-mapping size has extraordinary value in implementing omni-image unwarping in embedded systems, because there is usually limited memory in these systems. Experiments on TI DSP based embedded systems indicate that the proposed method reduces seven-eighths of the pano-mapping table size, and this method improves the unwarping speed by a factor of 2.74. The speed improvement comes mainly from the reduced memory access of a pano-mapping table.

The remainder of this paper is organized as follows: Section 2 reviews techniques for omni-image unwarping. In Section 3 we discuss geometric symmetry. Unwarping based on a reduced pano-mapping table is outlined in Section 4. Experiments are summarized in Section 5. Finally, some concluding remarks are given in Section 6.

2. Review of omni-images unwarping method based on pano-mapping table

In order to avoid the calibration of omni-cameras and reduce the computation time used for omni-images unwarping, Jeng and Tsai [16] proposed an approach that is based on a pano-mapping table. This pano-mapping table is created only once by a landmark learning process. The inputs of this learning process are a set of land-

mark points in the world and the set of corresponding points in a given omni-image. Fig. 1 illustrates the coordinate systems of an omni-camera, an omni-image and the world space, and Table 1 shows an example of a $M \times N$ size pano-mapping table [15].

In Fig. 1, P_1 and P_2 are two landmark points in the real world, and p_1 and p_2 are the corresponding image points. Jeng and Tsai [16] denote the coordinates of a real-world point P as (θ, ρ), where θ and ρ are the azimuth angle and elevation angle of an incident light ray; and denote the coordinates of an image pixel p as (μ, υ).

As shown in Table 1, the pano-mapping table is a 2-dimensional table, with the horizontal and vertical axes specifying the possible ranges of θ and ρ in M and N increments respectively. In the pano-mapping table, each entry E_{ij} has a pair of indices (i, j). Each entry E_{ij} specifies a pair (θ_i, ρ_j), defines an infinite set S_{ij} of real-world points on the light ray with azimuth angle θ_i and elevation angle ρ_j. All the points in S_{ij} project onto an identical pixel p_{ij} in an omni-image. This process is pano-mapping f_{pm} from S_{ij} to p_{ij} as shown in Fig. 2. In Table 1, each entry E_{ij} is filled with the coordinates $(\mu_{ij}, \upsilon_{ij})$ of pixel p_{ij} in the omni-image, where the values $(\mu_{ij}, \upsilon_{ij})$ are computed as:

$$\begin{cases} \theta_i = i \times (2\pi / M) & \text{for } i = 0,1,2,...,M-1, \\ \rho_j = j \times [(\rho_e - \rho_s) / N] + \rho_s & \text{for } j = 0,1,2,...,N-1, \\ r_j = f_r(\rho_j) = \sum_{n=0}^{4} (a_n \times \rho_j^n), & \\ \mu_{ij} = r_j \times \cos\theta_i, & \\ \upsilon_{ij} = r_j \times \sin\theta_i & \end{cases} \quad (1)$$

where ρ_e and ρ_s specify the omni-camera's maximum and minimum value of the elevation angles; $f_r(\rho)$ is a nonlinear function, it specifies the relation between elevation angle ρ and r, where r is the radial distance from image pixel p at coordinates (μ, υ) in the omni-image to the omni-image center; and the coefficients a_0 through a_4 of $f_r(\rho)$ are estimated using the image and real-world coordinate data of the previously mentioned corresponding landmark point pairs.

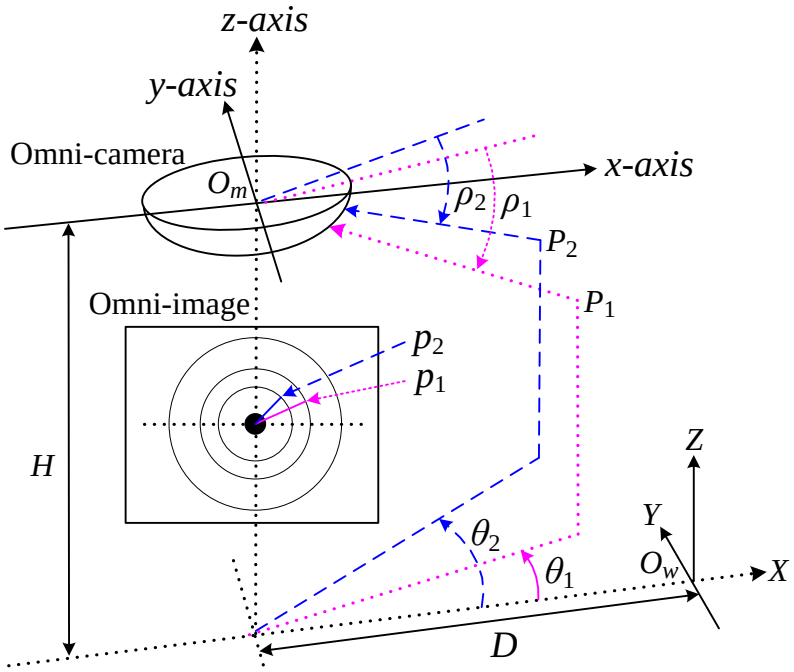


Fig. 1 Omni-camera, omni-image and world space coordinate systems, adapted from [15]

Table 1 Example of pano-mapping table of size $M \times N$

	θ_0	θ_1	…	θ_{M-1}
ρ_0	$(\mu_{00}, \upsilon_{00})$	$(\mu_{10}, \upsilon_{10})$	…	$(\mu_{M-1,\ 0}, \upsilon_{M-1,\ 0})$
ρ_1	$(\mu_{01}, \upsilon_{01})$	$(\mu_{11}, \upsilon_{11})$	…	$(\mu_{M-1,\ 1}, \upsilon_{M-1,\ 1})$
ρ_2	$(\mu_{02}, \upsilon_{02})$	$(\mu_{12}, \upsilon_{12})$	…	$(\mu_{M-1,\ 2}, \upsilon_{M-1,\ 2})$
ρ_3	$(\mu_{03}, \upsilon_{03})$	$(\mu_{13}, \upsilon_{13})$	…	$(\mu_{M-1,\ 3}, \upsilon_{M-1,\ 3})$
…	…	…	…	…
ρ_{N-1}	$(\mu_{0,\ N-1}, \upsilon_{0,\ N-1})$	$(\mu_{1,\ N-1}, \upsilon_{1,\ N-1})$	…	$(\mu_{M-1,\ N-1}, \upsilon_{M-1,\ N-1})$

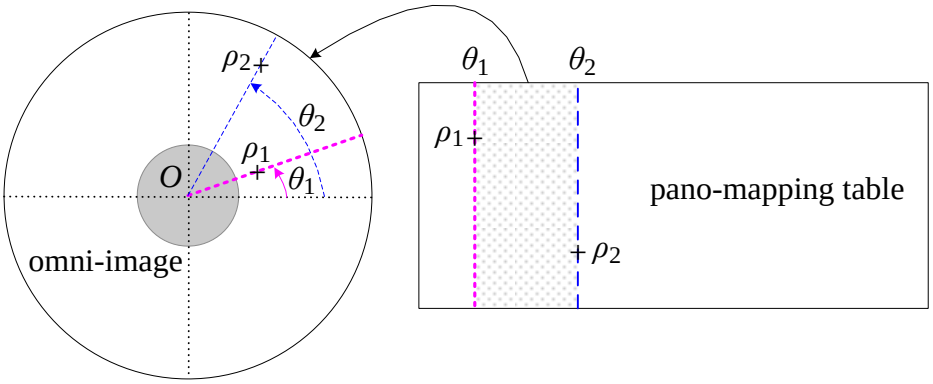


Fig. 2 Mapping between pano-mapping table and omni-image, adapted from[15]

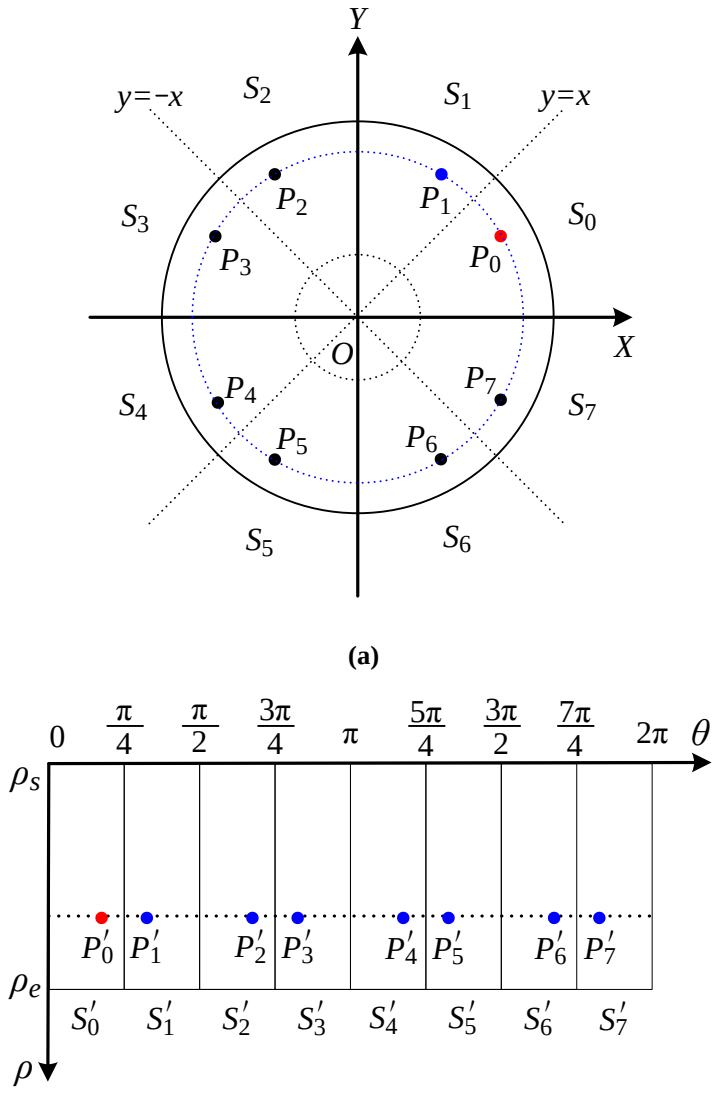


Fig. 3 Geometrical symmetry partitioning of omni-image and pano-mapping table. a Omni-image symmetry partitioning; b Pano-mapping table partitioning

3. Geometric symmetry analysis of omni-images unwarping

3.1 Geometric symmetry partitioning

We analyze the geometric symmetry properties in an omni-image and a pano-mapping table. According to Equation (1), we know that the azimuth angle θ ranges from 0 to 2π, and the values of $(\mu_{ij}, \upsilon_{ij})$ in a pano-mapping table are related with θ by trigonometric functions (i.e., sine and cosine functions).

In order to analyze the geometrical symmetry properties, we partition the omni-image and pano-mapping table into eight equal parts as shown in Fig. 3. In Fig. 3(b), the pano-mapping table is partitioned into eight parts, each part has an angle

interval of $\pi/4$. In Fig. 3(a), the omni-image is partitioned into eight equal sectors by four straight lines $X-axis$, $Y-axis$, $y=x$ and $y=-x$, these straight lines intersect at the circular center of the omni-image. The partitioned parts in an omni-image are omni-image sectors, denoted as S_k $(k=0,1,...,7)$ in Fig. 3(a); and the partitioned parts in pano-mapping table are pano-mapping table sectors, denoted as $S_k^{'}$ $(k=0,1,...,7)$ in Fig. 3(b). For the convenience of geometric symmetry deduction, we set the origin of an omni-image's coordinate system at the center of the omni-image, as is shown in Fig. 3(a), and we describe the proposed method based on this omni-image origin setting. When implementing the proposed method, we need to perform a coordinate system transformation for the omni-image.

Proposition 1: Entries that have the same elevation angle in a pano-mapping table correspond to a circle of pixels in an omni-image, and the circle center is the center of the omni-image.

Proof: Suppose there are $K(K\geq 2)$ entries that have the same elevation angle in pano-mapping table, their azimuth angle θ indices are $i\_k(k=0,1,...,K-1)$, and their elevation angle ρ indices are $j\_k(k=0,1,...,K-1)$.

Since $\sin^2\theta+\cos^2\theta=1$. So, for $k=0,1,...,K-1$ there exists:

$$\begin{aligned}&\mu^2_{i\_k,j\_k}+\nu^2_{i\_k,j\_k}\\&=r^2_{j\_k}\times\cos^2\theta_{i\_k}+r^2_{j\_k}\times\sin^2\theta_{i\_k}\\&=r^2_{j\_k}\times(\cos^2\theta_{i\_k}+\sin^2\theta_{i\_k})\\&=r^2_{j\_k}\end{aligned} \qquad (2)$$

Equation (1) tells us that:

$$r_{j\_k}=\sum_{n=0}^{4}(a_n\times\rho_{j\_k}^n) \qquad (3)$$

Substituting Equation (3) into Equation (2), for $k=0,1,...,K-1$ we have:

$$\mu^2_{i\_k,j\_k}+\nu^2_{i\_k,j\_k}=\left(\sum_{n=0}^{4}(a_n\times\rho_{j\_k}^n)\right)^2 \qquad (4)$$

Since the entries have the same elevation angle ρ, so there exists:

$$\mu^2_{i\_0,j\_0}+\nu^2_{i\_0,j\_0}=\mu^2_{i\_1,j\_1}+\nu^2_{i\_1,j\_1}=\mu^2_{i\_2,j\_2}+\nu^2_{i\_2,j\_2}=..... \qquad (5)$$

As we know from the definition of a pano-mapping table, entry values (μ_{ij},υ_{ij}) denote the coordinates of a pixel in an omni-image. According to Equation (5), pixels $P_k(k=0,1,...,K-1)$ lie on the same circle in an omni-image, and the circle center is the center of the omni-image. Therefore, the proposition is proven.

■

For example, entries $P_m^{'}(m=0,1,2,3,4,5,6,7)$ in Fig. 3(b) have the same elevation angle, and their corresponding pixels in an omni-image are $P_m(m=0,1,2,3,4,5,6,7)$ in Fig. 3(a), and pixels $P_m(m=0,1,2,3,4,5,6,7)$ lie on the same circle, centered at the center of the omni-image.

3.2 Geometric symmetry deduction on an omni-image

Based on the partitioned omni-image, we deduce the geometric symmetry properties. In Fig. 4, consider a pixel P_0 in an omni-image sector S_0, there are pixels $P_1, P_2, P_3, P_4, P_5, P_6, P_7$ in omni-image sectors $S_1, S_2, S_3, S_4, S_5, S_6, S_7$ separately, where: (a) P_1 is the point symmetric to P_0 with respect to (w.r.t.) the line $y = x$; (b) P_2 is the point symmetric to P_1 around the $Y-axis$; (c) P_3 is the point symmetric to P_2 w.r.t. the line $y = -x$; (d) P_4 is the point symmetric to P_3 around the $X-axis$; (e) P_5 is the point symmetric to P_4 w.r.t. the line $y = x$; (f) P_6 is the point symmetric to P_5 w.r.t. the $Y-axis$; (g) P_7 is the point symmetric to P_6 w.r.t. the line $y = -x$.

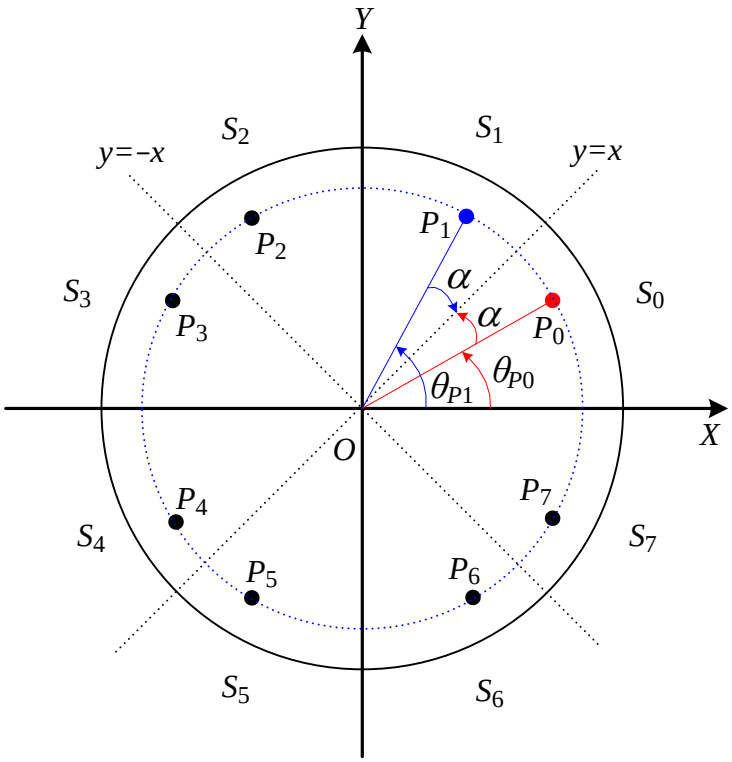


Fig. 4 Geometrical symmetry deduction on an omni-image

Proposition 2: In Fig. 4, suppose the coordinates of pixel P_0 are (x_0, y_0), then the coordinates of pixels P_m ($m = 0,1,2,3,4,5,6,7$) can be computed as:

$$\begin{cases} P_0 = (x_0, y_0) \\ P_1 = (y_0, x_0) \\ P_2 = (-y_0, x_0) \\ P_3 = (-x_0, y_0) \\ P_4 = (-x_0, -y_0) \\ P_5 = (-y_0, -x_0) \\ P_6 = (y_0, -x_0) \\ P_7 = (x_0, -y_0) \end{cases} \tag{6}$$

Proof: Suppose the angle between $X-axis$ and $\overrightarrow{OP_0}$ is θ_{P0}, the angle between $X-axis$ and $\overrightarrow{OP_1}$ is θ_{P1}. Since $\theta_{P0}+\alpha=\pi/4$, we have:

$$\alpha=\pi/4-\theta_{P0} \tag{7}$$

From Fig. 4 it follows that:

$$\theta_{P1}=\theta_{P0}+\alpha+\alpha \tag{8}$$

Substituting Equation (7) into Equation (8), we obtain:

$$\theta_{P1}=\pi/2-\theta_{P0} \tag{9}$$

If the coordinate values of pixel P_1 are (x_1,y_1), then:

$$\begin{cases} x_1=r_{P1}\times\cos\theta_{P1} \\ y_1=r_{P1}\times\sin\theta_{P1} \end{cases} \tag{10}$$

Since,

$$\begin{cases} x_0=r_{P0}\times\cos\theta_{P0} \\ y_0=r_{P0}\times\sin\theta_{P0} \end{cases} \tag{11}$$

Substituting Equation (9) into Equation (10) yields:

$$\begin{cases} x_1=r_{P1}\times\cos\theta_{P1}=r_{P1}\times\cos(\pi/2-\theta_{P0})=r_{P1}\times\sin\theta_{P0} \\ y_1=r_{P1}\times\sin\theta_{P1}=r_{P1}\times\sin(\pi/2-\theta_{P0})=r_{P1}\times\cos\theta_{P0} \end{cases} \tag{12}$$

Since pixels P_0 and P_1 are symmetric points about line $y=x$, which means $r_{P0}=r_{P1}$. Thus, based on Equations (11) and (12), we have:

$$\begin{cases} x_1=r_{P1}\times\sin\theta_{P0}=r_{P0}\times\sin\theta_{P0}=y_0 \\ y_1=r_{P1}\times\cos\theta_{P0}=r_{P0}\times\cos\theta_{P0}=x_0 \end{cases} \tag{13}$$

Thus, the coordinates of pixel P_1 can be denoted as (y_0,x_0).

Suppose the coordinate values of pixel P_2 are (x_2,y_2), since P_2 and P_1 are symmetric points w.r.t. the $Y-axis$, so $x_2=-x_1$ and $y_2=y_1$, after substituting into Equation (13) we obtain:

$$\begin{cases} x_2=-y_0 \\ y_2=x_0 \end{cases} \tag{14}$$

Thus, the coordinates of pixel P_2 can be denoted by $(-y_0,x_0)$.

A similar method can be used on pixels P_3,P_4,P_5,P_6,P_7. This proves the proposition.

■

3.3 Geometrical symmetry deduction on a pano-mapping table

For the eight symmetric pixels $P_m\,(m=0,1,2,3,4,5,6,7)$ shown in Fig. 4, we analyze their mapping entries $P_m^{'}\,(m=0,1,2,3,4,5,6,7)$ in the pano-mapping table separately. Fig. 5 illustrates the geometric symmetry properties in the pano-mapping table.

Proposition 3: For the eight symmetrical pixels $P_m\,(m=0,1,2,3,4,5,6,7)$ shown in Fig. 4, and their mapping entries $P_m^{'}\,(m=0,1,2,3,4,5,6,7)$ in pano-mapping table in Fig. 5, suppose that the coordinate of P_0 is (x_0,y_0), the angle between

the $X-axis$ and $\overrightarrow{OP_0}$ is θ_{P0} in Fig. 4, then the azimuth angle and elevation angle of entries $P_m^{'}(m=0,1,2,3,4,5,6,7)$ in Fig. 5 are $\theta_{Pm}^{'}$ and $\rho_{Pm}^{'}$. Where:

$$\sum_{n=0}^{4}(a_n \times {\rho_{Pm}^{'}}^n) = \sqrt{{x_0}^2 + {y_0}^2} \qquad (m=0,1,2,3,4,5,6,7) \quad (15)$$

and,

$$\theta_{Pm}^{'} = \begin{cases} m \times \pi/4 + \theta_{P0} & \text{if } m = 0,2,4,6 \\ (m+1) \times \pi/4 - \theta_{P0} & \text{if } m = 1,3,5,7 \end{cases} \qquad (16)$$

Proof: Following Proposition 2, we substitute Equation (6) into Equation (4) giving us:

$${x_0}^2 + {y_0}^2 = \left(\sum_{n=0}^{4}(a_n \times {\rho_{Pm}^{'}}^n) \right)^2 \qquad (m=0,1,2,3,4,5,6,7) \quad (17)$$

Based on Equation (17), Equation (15) is proven. Now, we prove that Equation (16) is true. Let us consider the case when $m=0$, which corresponds to pixel P_0 in Fig. 4 and entry $P_0^{'}$ in Fig. 5. For pixel P_0, there exists:

$$\theta_{P0} = arctg\frac{y_0}{x_0} \qquad (18)$$

Suppose the i, j indices of entry $P_0^{'}$ in the pano-mapping table (Fig. 5) are i_P0 and j_P0. Following Equation (1) we have:

$$\begin{cases} \mu_{i\_P0, j\_P0} = r_{j\_P0} \times \cos\theta_{i\_P0} \\ \upsilon_{i\_P0, j\_P0} = r_{j\_P0} \times \sin\theta_{i\_P0} \end{cases} \qquad (19)$$

Where,

$$\theta_{i\_P0} = \theta_{P0}^{'} \qquad (20)$$

Since entry $P_0^{'}$ in the pano-mapping table (Fig. 5) is mapped to pixel P_0 in the omni-image (Fig. 4), entry value of $P_0^{'}$ is the same as the coordinates of pixel P_0, which means:

$$\begin{cases} \mu_{i\_P0, j\_P0} = y_0 \\ \upsilon_{i\_P0, j\_P0} = x_0 \end{cases} \qquad (21)$$

Resolving Equations (18), (19), (20) and (21) yields:

$$\theta_{P0}^{'} = \theta_{P0} = 0 \times \pi/4 + \theta_{P0} \qquad (22)$$

Thus, Equation (16) is true when $m=0$.

Let us consider the case when $m=1$, which corresponds to pixel P_1 in Fig. 4 and entry $P_1^{'}$ in Fig. 5. For pixel P_1, according to Equation (6), the coordinate of P_1 is (y_0, x_0). Thus,

$$\theta_{P1} = arctg\frac{x_0}{y_0} \qquad (23)$$

Suppose the i, j indices of entry $P_1^{'}$ in the pano-mapping table (Fig. 5) are i_P1 and j_P1. Following Equation (1) we have:

$$\begin{cases} \mu_{i\_P1, j\_P1} = r_{j\_P1} \times \cos\theta_{i\_P1} \\ \upsilon_{i\_P1, j\_P1} = r_{j\_P1} \times \sin\theta_{i\_P1} \end{cases} \qquad (24)$$

Where,

$$\theta_{i\_P1} = \theta'_{P1} \qquad (25)$$

Since the entry P_1' in the pano-mapping table (Fig. 5) is mapped to pixel P_1 in the omni-image (Fig. 4), coordinates of P_1' is the same as P_1, implying:

$$\begin{cases} \mu_{i\_P1, j\_P1} = x_0 \\ \upsilon_{i\_P1, j\_P1} = y_0 \end{cases} \qquad (26)$$

Resolving Equations (24), (25) and (26), yields:

$$ctg(\theta'_{P1}) = \frac{x_0}{y_0} \qquad (27)$$

According to Fig. 4 and Fig. 5, we know that $0 \le \theta_{P0}, \theta_{P1}, \theta'_{P1} \le \pi/2$. From Equations (18) and (27) it follows that:

$$\theta'_{P1} = \pi/2 - \theta_{P0} = (1+1)\times\pi/4 - \theta_{P0} \qquad (28)$$

Thus, Equation (16) is true when $m = 1$.

A similar method can be used for $m = 2,3,4,5,6,7$. This proves the proposition.

■

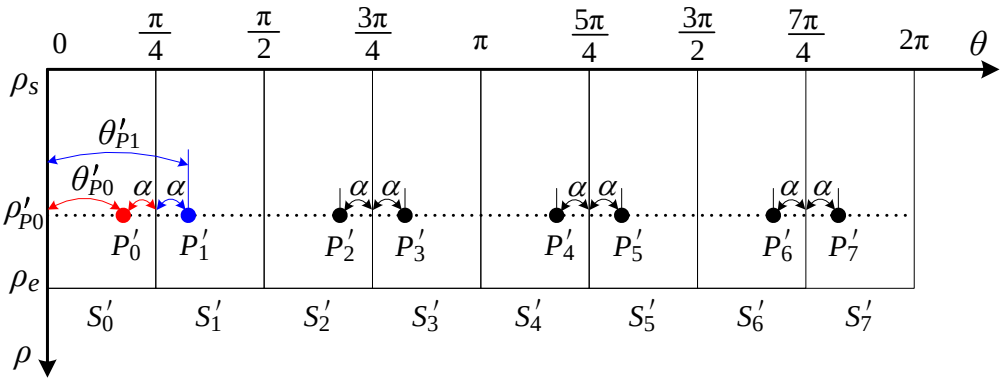


Fig. 5 Geometric symmetry deduction on pano-mapping table

4. Omni-images unwarping based on a one-eighth pano-mapping table

4.1 Creating a one-eighth pano-mapping table

In order to reduce the size of the pano-mapping table, we discuss how to utilize the geometric symmetry properties that exist in an omni-image and a pano-mapping table. Fig. 6 illustrates the one-eighth pano-mapping table.

We divide the range $\pi/4$ *rad* (45°) (i.e., one-eighth of 2π *rad* (360°)) of azimuth angles equally into L units, where $L = \left\lceil \frac{M}{8} \right\rceil$ and M is the number of azimuth angle units for dividing range 2π *rad* (360°) in Equation (1). Thus:

$$\theta_{T\_i} = i \times [(\pi/4)/L] \qquad for\ i = 0,1,2,...,L-1 \qquad (29)$$

Then, the range of elevation angles (from ρ_s to ρ_e) is divided into N units, which can be described as:

$$\rho_{T\_j} = j \times [(\rho_e - \rho_s)/N] + \rho_s \qquad for \;\; j = 0,1,2,...,N-1 \quad (30)$$

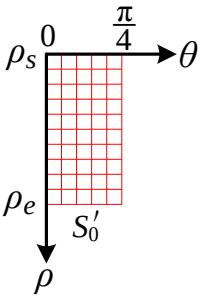
Entries in the one-eighth pano-mapping table (Fig. 6(c)) correspond directly to pixel coordinates of the omni-image sector S_0 (Fig. 6(b)), the coordinate computation formula is:

$$\begin{cases} g_{ij} = r_j \times \cos\theta_{T\_i} \\ h_{ij} = r_j \times \sin\theta_{T\_i} \end{cases} \quad (31)$$

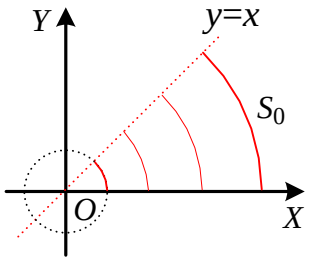
where,

$$r_j = \sum_{n=0}^{4} (a_n \times \rho_{T\_j}{}^n) \quad (32)$$

With $(a_0, a_1, a_2, a_3, a_4)$ sharing the same value and interpretation as those in Equation (1).



(a)



(b)

θ: 0 → $\frac{\pi}{4}$; ρ: ρ_s → ρ_e

	θ_{T_0}	θ_{T_1}	...	θ_{T_L-1}
ρ_{T_0}	(g_{00}, h_{00})	(g_{10}, h_{10})	...	$(g_{L-1,0}, h_{L-1,0})$
ρ_{T_1}	(g_{01}, h_{01})	(g_{11}, h_{11})	...	$(g_{L-1,1}, h_{L-1,1})$
...	...	...	...	...
ρ_{T_L-1}	$(g_{0,N-1}, h_{0,N-1})$	$(g_{1,N-1}, h_{1,N-1})$	...	$(g_{L-1,N-1}, h_{L-1,N-1})$

(c)

Fig. 6 One-eighth omni-image and one-eighth pano-mapping table creation. a Azimuth angle of the one-eighth pano-mapping table ranges from 0 to π/4; b One-eighth of an omni-image whose azimuth angle ranges from 0 to π/4; c One-eighth pano-mapping table created

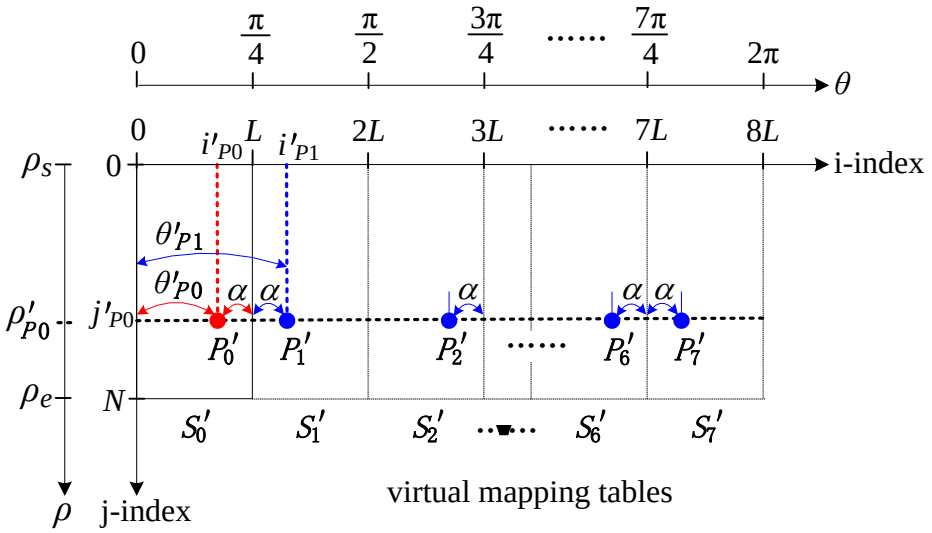


Fig. 7 Reusing the one-eighth pano-mapping table to create "virtual mapping tables"

4.2 Reusing the one-eighth pano-mapping table based on geometric symmetry

After creating the one-eighth pano-mapping table, we can find that it is easy to unwarp the first omni-image sector S_0. However, how can we unwarp the remaining seven omni-image sectors (i.e. sectors $S_1, S_2, S_3, S_4, S_5, S_6, S_7$)? The answer is reusing the created one-eighth pano-mapping table based on geometric symmetry.

Fig. 7 illustrates the idea behind this reusing approach. We create a "virtual mapping table" for each of the remaining seven omni-image sectors $S_1, S_2, S_3, S_4, S_5, S_6, S_7$, and denote these "virtual mapping tables" as VMP_1, VMP_2, VMP_3, VMP_4, VMP_5, VMP_6, VMP_7 separately. Accordingly, the one-eighth pano-mapping table created for omni-image sector S_0 is a "real mapping table (*RMP*)" that is stored in computer memory. We discuss the method for creating "virtual mapping tables" VMP_k $(k = 1, 2, 3, ..., 7)$ in Algorithm 1.

Algorithm 1. Creating "virtual mapping tables" VMP_k $(k = 1, 2, 3, ..., 7)$ based on one-eighth pano-mapping table RMP.

Step 1. Define the size of each "virtual mapping table" to be the same as RMP, i.e., the two dimensions have $L \times N$ entries for each "virtual mapping table" VMP_k $(k = 1, 2, 3, ..., 7)$.

Step 2. Define the corresponding azimuth angle range of VMP_k from $\pi/4 \times k$ to $\pi/4 \times (k+1)$, and elevation angle range from ρ_s to ρ_e. In order to ensure sequential continuity of i-indices between the created one-eighth pano-mapping table RMP and the seven "virtual mapping tables" VMP_k $(k = 1, 2, 3, ..., 7)$, we define the i-index of VMP_k $(k = 1, 2, 3, ..., 7)$ starting from $L \times k$ and ending at $L \times k + L - 1 = L \times (k+1) - 1$ (since the i-index length of each "virtual mapping table" is L). For example, the i-index of VMP_1 starts from L and ends at $L \times 2 - 1$, the i-index of VMP_2 starts from $L \times 2$ and ends at $L \times 3 - 1$.

Step 3. Fill in the entries in "virtual mapping tables" VMP_k ($k=1,2,3,...,7$). For each entry (say $P_0^{'}$) in RMP (see Fig. 7), do the following operations:

Substep 3.1. Suppose the i-index and j-index of $P_0^{'}$ are $i_{P0}^{'}$ and $j_{P0}^{'}$, the azimuth angle and elevation angle of $P_0^{'}$ are $\theta_{P0}^{'}$ and $\rho_{P0}^{'}$. Substituting into Equations (29) and (30), there exists:

$$\theta_{P0}^{'} = i_{P0}^{'} \times [(\pi / 4) / L] \qquad (33)$$

and

$$\rho_{P0}^{'} = j_{P0}^{'} \times [(\rho_e - \rho_s) / N] + \rho_s \qquad (34)$$

Substep 3.2. Suppose $P_k^{'}$ ($k=1,2,3,...,7$) in "virtual mapping tables" VMP_k ($k=1,2,3,...,7$) are geometrical symmetry entries of $P_0^{'}$, and the i-index and j-index of $P_k^{'}$ ($k=1,2,3,...,7$) are $i_{Pk}^{'}$ and $j_{Pk}^{'}$, the azimuth angle and elevation angle of $P_k^{'}$ are $\theta_{Pk}^{'}$ and $\rho_{Pk}^{'}$. Substituting into Equations (29) & (30), we have:

$$\theta_{Pk}^{'} = i_{Pk}^{'} \times [(\pi / 4) / L] \qquad for\ \ k=1,2,...,7 \quad (35)$$

and

$$\rho_{Pk}^{'} = j_{Pk}^{'} \times [(\rho_e - \rho_s) / N] + \rho_s \qquad for\ \ k=1,2,...,7 \quad (36)$$

Following Equations (15) and (16) in Proposition 3, and resolving Equations (33), (34), (35) and (36) yields:

$$i_{Pk}^{'} = \begin{cases} k \times L + i_{P0}^{'} & \text{if } k = 2,4,6 \\ (k+1) \times L - i_{P0}^{'} & \text{if } k = 1,3,5,7 \end{cases} \qquad (37)$$

and

$$j_{Pk}^{'} = j_{P0}^{'} \qquad for\ \ k=1,2,...,7 \qquad (38)$$

Substep 3.3. Fill the seven entries $P_k^{'}$ ($k=1,2,3,...,7$). Suppose the entry values of $P_0^{'}$ are $(g_{i'P0,j'P0}, h_{i'P0,j'P0})$ (note that $g_{i'P0,j'P0}$ and $h_{i'P0,j'P0}$ are already known by Equation (31)). According to Equations (6) in Proposition 2, and Equations (37) and (38), the values of entries $P_k^{'}$ ($k=1,2,3,...,7$) should be calculated using the following equations:

$$\begin{cases} P_1^{'} = (h_{i'P0,j'P0}, g_{i'P0,j'P0}) \\ P_2^{'} = (-h_{i'P0,j'P0}, g_{i'P0,j'P0}) \\ P_3^{'} = (-g_{i'P0,j'P0}, h_{i'P0,j'P0}) \\ P_4^{'} = (-g_{i'P0,j'P0}, -h_{i'P0,j'P0}) \\ P_5^{'} = (-h_{i'P0,j'P0}, -g_{i'P0,j'P0}) \\ P_6^{'} = (h_{i'P0,j'P0}, -g_{i'P0,j'P0}) \\ P_7^{'} = (g_{i'P0,j'P0}, -h_{i'P0,j'P0}) \end{cases} \qquad (39)$$

4.3 Unwarping omni-images based on one-eighth pano-mapping table

Given an input omni-image G and a one-eighth pano-mapping table RMP with $L \times N$ entries, we may generate a corresponding generic panoramic image

Q which is $8L$ pixels in width and N pixels in height. The algorithm for this unwarping approach is as follows:

Algorithm 2. Unwarping omni-image G into generic panoramic image Q with the aid of an one-eighth pano-mapping table RMP.

Step 1. Divide the panoramic image Q into eight equal sub-pano-images in the width direction, with each sub-pano-image sized $L \times N$, and denote these sub-pano-images as Q_k ($k = 0,1,2,3,...,7$).

Step 2. For each entry $P_0^{'}$ in a one-eighth pano-mapping table RMP, suppose the the i-index and j-index of $P_0^{'}$ are $i_{P0}^{'}$ and $j_{P0}^{'}$, and the entry values of $P_0^{'}$ are $(g_{i'P0,j'P0}, h_{i'P0,j'P0})$. Now, we fill in a pixel color for each of the eight sub-pano-images Q_k ($k = 0,1,2,3,...,7$) as follows:

Substep 2.1. The first sub-pano-image Q_0 corresponds to the first omni-image sector S_0 of omni-image G, and we can unwarp omni-image sector S_0 into sub-pano-images Q_0 based directly on the created one-eighth pano-mapping table RMP. The method assigns the color values of the pixel P_0 of omni-image G at coordinates $(g_{i'P0,j'P0}, h_{i'P0,j'P0})$, to the pixel q_0 of panoramic image Q at coordinates $(i_{P0}^{'}, j_{P0}^{'})$.

Substep 2.2. The remaining seven sub-pano-images Q_k ($k = 1,2,3,...,7$) correspond to omni-image sectors S_k ($k = 1,2,3,...,7$) of omni-image G separately, and we can unwarp these omni-image sectors S_k ($k = 1,2,3,...,7$) into sub-pano-images Q_k ($k = 1,2,3,...,7$) based on the "virtual mapping tables" VMP_k ($k = 1,2,3,...,7$). The method assigns the color values of pixel P_k of omni-image G at coordinates $P_k^{'}$, to the pixel q_k of panoramic image Q at coordinates $(i_{Pk}^{'}, j_{Pk}^{'})$, where $P_k^{'}$ is calculated using Equation (39), $i_{Pk}^{'}$ comes from Equation (37) and $j_{Pk}^{'}$ from Equation (38).

Substep 2.3. After all entries in the one-eighth pano-mapping table RMP are processed, the final Q is just the panoramic image that is unwarped from omni-image G.

As we can see in Algorithm 2, the one-eighth pano-mapping table is used to unwarp one omni-image sector (which is $\pi/4$ in radian, one-eighth of 2π). On the other hand, when unwarping the other seven omni-image sectors (which are totally $7\pi/4$ in radian, seven-eighth of 2π), we use Equations (37), (38) and (39) to find the corresponding coordinates for unwarping, and these Equations need about 4 arithmetic operations in CPU. Thus, the proposed method saves seven-eighth pano-mapping table with very few CPU operations.

5. Experiments

5.1 Omni-image samples

In this experiment, we unwarp omni-images that are acquired using the parabolic catadioptric omni-camera designed by us in [1]. The omni-camera captures omni-

images that are in 24 bit RGB format, with an effective resolution of 1888×1888 for each of the captured omni-image. In order to analyze the performance and the required one-eighth pano-mapping table size at different omni-image resolutions, we scale the captured omni-images to produce 944×944, 472×472 sized omni-images using down-sampling techniques.

The resolutions of the unwarped panoramic images can be defined according to the application requirements. However, in order to maintain the aspect ratio of the image, we define three typical panoramic image resolutions in this experiment, which are 2560×576, 1280×288, 640×144.

During the experiment, we capture twenty 1888×1888 sized omni-images with the parabolic catadioptric omni-camera; then scale them to get twenty 944×944 sized omni-images and twenty 472×472 sized omni-images. The images are divided into three groups of omni-image samples, with 20 omni-images in each group having the same resolution.

In the experiment, each omni-image in the three groups is unwarped 1000 times into panoramic images of different image size, including 2560×576, 1280×288 and 640×144, to get an accurate estimate of the unwarping time.

5.2 Experimental environment

The experimental scenario is an embedded system for omni-directional robot vision. In this embedded system, a C6000 TMS320DM642 DSP (Digital Signal Processor) from TI (Texas Instruments) corporation is used as the embedded CPU, which is specially designed to optimize image processing performance with a stunning capability of 5760MIPS at a clock rate of 720MHz. There is a 32MB RAM (Random Access Memory) that can be used for image and pano-mapping table storage. The compiler is Code Composer Studio (CCStudio) Version 2.20.18.

5.3 Experimental results

5.3.1 Mapping table storage reduction

Table 2 illustrates comparison of the proposed one-eighth pano-mapping table with that presented in [16] in terms of mapping table storage size. We define three different sized one-eighth pano-mapping tables (i.e., 320×576, 160×288 and 80×144), which correspond to the three different panoramic image sizes (i.e., 2560×576, 1280×288 and 640×144). For every entry in the one-eighth pano-maping table, in total 4 bytes are needed to store the x and y coordinates (2 bytes for each of the x and y coordinates).

As we can see in Table 2, the proposed one-eighth pano-mapping table needs just one-eighth storage space compared with that of Jeng [16]. This is because we can use the geometric symmetry properties of omni-images and pano-mapping tables to infer the correspondence between omni-images and panoramic images for seven-eighths of the unwarping process, with the help of Equations (37), (38) & (39) and Algorithm 2 above.

Table 2 Mapping table storage of the proposed approach compared to Jeng[16]

Omni-image resolution	Panoramic image resolution	Jeng[16] (Bytes)	Proposed (Bytes)	Reduced Storage Ratio
1888×1888	2560×576	5,898,240	737,280	0.875
944×944	2560×576			
472×472	2560×576			
1888×1888	1280×288	1,474,560	184,320	0.875
944×944	1280×288			
472×472	1280×288			
1888×1888	640×144	368,640	46,080	0.875
944×944	640×144			
472×472	640×144			
Reduced Storage Ratio: 0.875				

Table 3 Speed of the proposed approach compared to Jeng[16]

Omni-image resolution	Panoramic image resolution	Jeng[16] (ms)	Proposed (ms)	Improvement Ratio
1888×1888	2560×576	1243.9	544.7	2.28
944×944	2560×576	1105.7	365.9	3.02
472×472	2560×576	1008.1	276.4	3.65
1888×1888	1280×288	325.2	162.6	2.00
944×944	1280×288	288.6	107.7	2.68
472×472	1280×288	254.1	71.1	3.57
1888×1888	640×144	87.4	49.8	1.76
944×944	640×144	74.2	31.5	2.36
472×472	640×144	64.5	19.3	3.34
Average Improvement Ratio: 2.74				

5.3.2 Omni-image unwarping speed comparison

In the proposed one-eighth pano-mapping table method, only one omni-image sector can be unwarped directly using the mapping table created, while the other seven omni-image sectors can only be unwarped based on Equations (37), (38), (39) and Algorithm 2 indirectly, which need about 4 extra CPU arithmetic operations for each pixel.

Thus, it may appear that the proposed method will slow down the unwarping to some extent. However, experimental results shown in Table 3 give the opposite answer. According to the experimental results, we find that the proposed one-eighth pano-mapping table makes the unwarping speed 2.74 times faster on the average than the method in Jeng [16].

The speedup of proposed method comes mainly from the reduced pano-mapping table access. As we know, in embedded system implementation, memory access is a time-consuming operation, it costs much more CPU cycles to fetch data from external memory than performing CPU arithmetic operations [17,18]. Although we need about 4 extra CPU arithmetic operations to unwarp each pixel

in the seven omni-image sectors, we do not need to read data from the pano-mapping table which is stored in external or DSP cache memory.

5.3.3 Omni-image, one-eighth pano-mapping table and panoramic image

Fig. 8 illustrates a captured omni-image, the one-eighth pano-mapping table and panoramic image. As is shown in Fig. 8(a), the omni-image is divided into eight sectors by four straight lines $X-axis$, $Y-axis$, $y=x$ and $y=-x$, these sectors are denoted by $S_0, S_1, S_2, S_3, S_4, S_5, S_6, S_7$. In Fig. 8(b), a one-eighth pano-mapping table is created, which directly maps the omni-image sector S_0 to panoramic image sector Q_0 shown in Fig. 8(c). The other seven omni-image sectors (i.e., $S_1, S_2, S_3, S_4, S_5, S_6, S_7$) are unwarped to panoramic image sectors (i.e., $Q_1, Q_2, Q_3, Q_4, Q_5, Q_6, Q_7$) based on Equations (37), (38), (39) and Algorithm 2, which is derived from the geometric symmetry properties of the omni-image and the pano-mapping table.

In order to compare the quality of achieved panoramic images, we unwarp omni-images using both the proposed method and Jeng [16] method, the results are shown in Fig. 9. In this figure, the first column (Fig. 9(a)) are omni-images to be unwarped, their resolution is 1888×1888; the second and third columns (Fig. 9(b), (c)) are panoramic images achieved with the proposed method and Jeng [16] method respectively, which have same resolution of 1280×288. Based on geometric symmetry, the proposed method recovers identical mapping data as those in Jeng's pano-mapping table [16], which results in the same panoramic images in Fig. 9(b) and Fig. 9(c), showing that the proposed method does not degrade image quality.

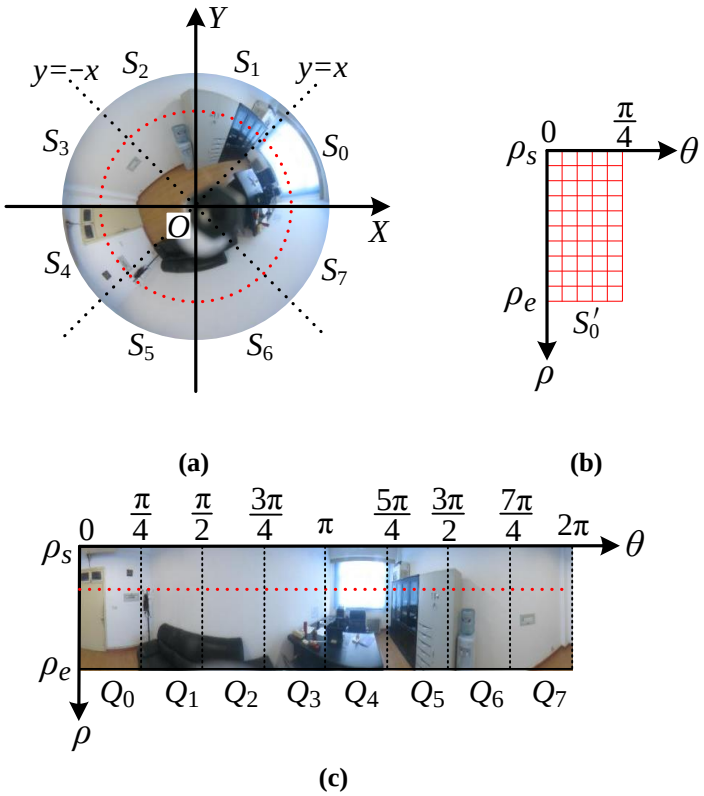


Fig. 8 Omni-image, one-eighth pano-mapping table and panoramic image. a Omni-image to be unwarped; b One-eighth pano-mapping table; c Panoramic image achieved by unwarping the omni-image with proposed one-eighth pano-mapping table

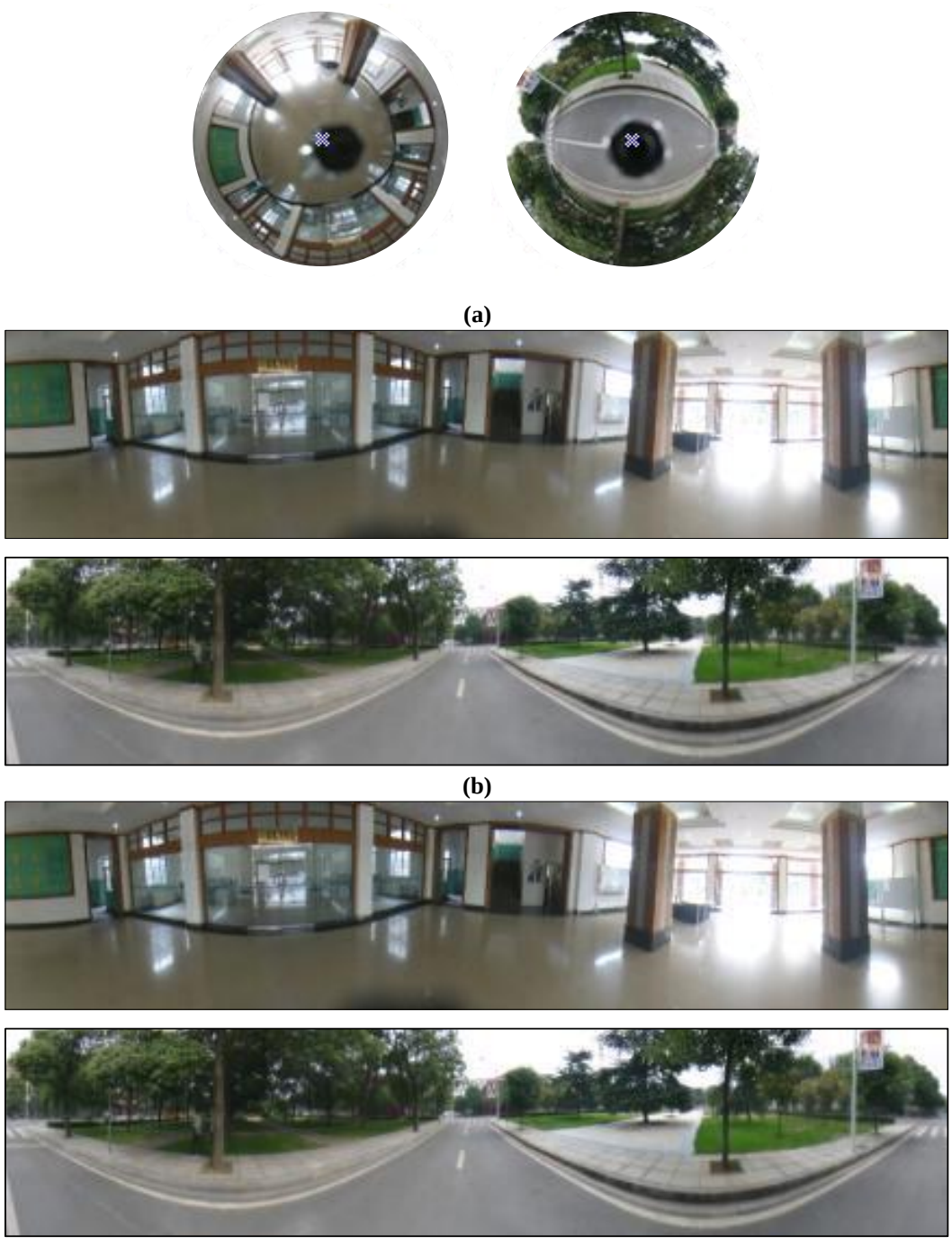


Fig. 9 Omni-image unwarping results of proposed method and Jeng [16] method. a Omni-images to be unwarped; b Panoramic images achieved with proposed method; c Panoramic images achieved with Jeng [16] method

6. Conclusions

A pano-mapping table size reduction method is proposed for omni-image unwarping, which is based on observing geometric symmetry properties of omni-images and the pano-mapping table. Experimental results demonstrate that the proposed method reduces seven-eighths of the pano-mapping table size and improves omni-image unwarping speed by 2.74 times. This work assumes that the omni-camera is

well aligned, which makes the geometric symmetry relationship possible. In the future work, we will study methods for reducing pano-mapping table size when an omni-camera is misaligned.

Acknowledgments This research was partially supported by National Natural Science Foundation (NSFC) of China with project No.60705013, No.60872150, No. 60803101 and No.60773023; China Postdoctoral Science Foundation special funding with project No.200902665, China Postdoctoral Science Foundation with project No.20070410977, Natural Science Foundation of Hunan Province in China with project No.08JJ4018, iCORE (Alberta Innovates) Alberta, and NSERC Canada.

References

1. Xiong, Z.H., Zhang, M.J., Wang, Y.L., Li, T., Li, S.K.: Fast panorama unrolling of catadioptric omni-directional images for cooperative robot vision system. Proc. of the 11th Int. Conf. on Computer Supported Cooperative Work in Design, Melbourne, Australia. 1100 – 1104, (2007)
2. Andreasson, H., Treptow, A., Duckett, T.: Self-localization in non-stationary environments using omni-directional vision. Robotics and Autonomous Systems. 55, 541 – 551, (2007)
3. Wu, C.J., Tsai, W.H.: Location estimation for indoor autonomous vehicle navigation by omni-directional vision using circular landmarks on ceilings. Robotics and Autonomous Systems. 57, 546 – 555, (2009)
4. Feng, H.M., Chen, C.Y., Horng. J.H.: Intelligent omni-directional vision-based mobile robot fuzzy systems design and implementation. Expert Systems with Applications. 37, 4009 – 4019, (2010)
5. Liu, Y.C., Lin, K.Y., Chen, Y.S.: Bird's-eye view vision system for vehicle surrounding monitoring. Proc. Conf. Robot Vision, Berlin, Germany, 207 – 218, (2008)
6. Boult, T.E., Gao, X., Micheals, R.J., Eckmann, M.: Omni-directional visual surveillance. Image and Vision Computing, 22, 515 – 534, (2004)
7. Gandhi, T., Trivedi, M.: Person tracking and reidentification: Introducing Panoramic Appearance Map (PAM) for feature representation. Machine Vision and Applications. 18, 207 – 220, (2007)
8. Ng, K.C., Ishiguro, H., Trivedi, M.M., Sogo, T.: An integrated surveillance system: human tracking and view synthesis using multiple omni-directional vision sensors. Image and Vision Computing. 22, 551 – 561, (2004)
9. Fiala, M., Basu, A.: Panoramic stereo reconstruction using non-SVP optics. Computer Vision and Image Understanding. 8, 363 – 397, (2005)
10. Koyasu, H., Miura, J., Shirai, Y.: Recognizing moving obstacles for robot navigation using real-time omni-directional stereo vision. Journal of Robotics and Mechatronics. 14, 147 – 156, (2002)

11. Peleg, S., Pritch, Y., Ben-Ezra, M.: Cameras for stereo panoramic imaging. Proc. of IEEE Conf. on Computer Vision and Pattern Recognition, Hyatt Regency, Hilton Head Island, South Carolina, 208 – 214, (2000)
12. Svoboda, T., Pajdla, T.: Epipolar geometry for central catadioptric cameras. Journal of Computer Vision. 49, 23 – 37, (2002)
13. Winters, N., Gaspar, J., Lacey, G., Victor, J.S.: Omni-directional vision for robot navigation. Proc. of IEEE Workshop on Omnidirectional Vision, Hyatt Regency, Hilton Head Island, South Carolina, 21 – 28, (2000)
14. Gaspar, J., Decco, C., Okamoto, J.: Constant resolution omni-directional cameras. Proc. of IEEE Workshop on Omnidirectional Vision, Copenhagen, Denmark, 27 – 34, (2002)
15. Wu, C.J., Tsai, W.H.: Unwarping of images taken by misaligned omnicameras without camera calibration by curved quadrilateral morphing using quadratic pattern classifiers. Optical Engineering.48, 087003(1) – 087003(11), (2009)
16. Jeng, S.W., Tsai, W.H. Using pano-mapping tables for unwarping of omni-images into panoramic and perspective-view images. IET Image processing. 1, 149 – 155, (2007)
17. Chen, C.W., Ku, C.J., Chang, C.H.: Designing a high performance and low energy-consuming embedded system with considering code compressed environments. Proc. of 11th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications, Hong Kong, 317 – 322, (2005)
18. Noergaard T.: Embedded Systems Architecture: A Comprehensive Guide for Engineers and Programmers. Elsevier, (2005)