INFORMATION
Volume 16, Number 2(B), pp.1567-1572

ISSN 1343-4500

Radial Processing for Feature Detection of Omnidirectional Images

Jin Shin* and SooYeong Yi**
*,** Department of Electrical Engineering, Seoul National University of Science and Technology, Korea
* gomlands@naver.com, ** suylee@seoultech.ac.kr

Abstract

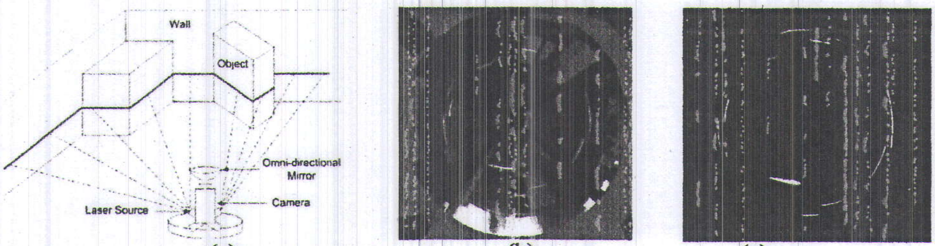
Since omnidirectional images contain many circular arc segments, radial directional processing for detection of certain features in the image is more appropriate than horizontal or vertical directional processing, as with usual rectangular images. Computation of the proposed radial processing algorithm is based on the well-known Bresenham raster algorithm for line drawing in discrete integer space, and thus it is very fast and efficient. Experimental comparisons between the proposed algorithm and conventional horizontal processing in terms of feature detection and computational efficiency are presented.

Keywords: Omnidirectional structured light image, Bresenham raster algorithm, Image feature point

1. Introduction

Omnidirectional image has a $360°$ all-directional view that contains much more information than conventional image. Omnidirectional images can be acquired through a catadioptric approach that uses an ordinary camera and a curved mirror[1]. The omnidirectional imaging system combined with a structured light technique can be used as an omnidirectional ranging system [3,4]. Since they are simple and cost effective, such omnidirectional ranging systems can be employed as an alternative to ultrasonic ring sensors or motorized laser scanners [2]. An exemplar $360°$ all-directional distance measurement system was proposed based on the catadioptric omnidirectional imaging approach with structured light, as illustrated in Fig. 1 [3,4]. To calculate the distance to an object from the extracted structured light image, image processing such as differentiation between neighboring pixels should be performed in order to detect feature points of the structured light pixels. In general, horizontal, vertical, or diagonal directional processing is appropriate for typical rectangular images from a conventional camera. However, for the omnidirectional image shown in Fig. 1 (b) or (c), radial directional processing from the optical center is more suitable for detecting edge feature points of the structured light pixels.

In this paper, an efficient radial directional processing algorithm for omnidirectional structured light images is proposed based on the computational structure of the well-known Bresenham raster algorithm. Commonly used in the field of computer graphics, the Bresenham raster algorithm draws a line between two given points using only simple operations such as integer addition or subtraction.



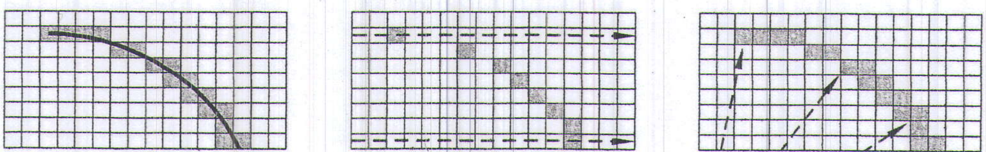
(a) (b) (c)

Fig. 1. Omnidirectional structured light imaging system and structured light pixel image: (a) Omnidirectional ranging system with structured light, (b) Omnidirectional structured light image, (c) Extracted structured light pixel image

2. Radial directional processing for feature detection

As shown in Fig. 1 (b) and (c), the omnidirectional image contains many circular arc edges. Thus, the edge feature points in the image might be missed by vertical or horizontal directional processing in the quantized image space of discrete pixel units, as demonstrated in Fig. 2. Fig. 2 (a) is an example of a circular arc image. Horizontal processing yields the result shown in Fig. 2 (b). For simplicity, horizontal directional differentiation of the image is assumed for the feature detection. In contrast, with radial processing, the direction angle resolution can be set arbitrarily; as a consequence, it is possible to detect every feature pixel on an arc edge, as shown in Fig. 2 (c).

To detect the feature points in an omnidirectional image through the radial directional algorithm, processing for the pixels in one directional angle from the optical center and repetition of the processing for all directional angles up to $360°$ should be performed. A simple brute-force radial processing approach for a certain directional angle θ is shown in Fig. 3 (a).



(a) (b) (c)

Fig. 2. Results of feature point detection for the omnidirectional structured light image: (a) Circular arc segment of an omnidirectional structured light image, (b) Result of horizontal processing, (c) Result of radial processing

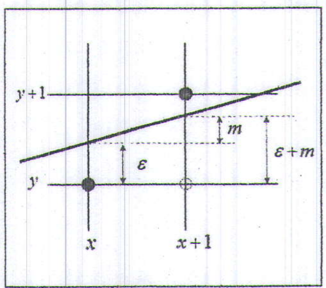
In Fig. 3 (a), (x_S, y_S) is the starting point of the radial processing, i.e., the optical center, (x_C, y_C). Assuming that the optical center coincides with the center of the 640×480 sized image, the length of radius r would be $\sqrt{320^2 + 240^2} = 400$. The function $round(\cdot)$ casts the argument as an integer, and processing(m, n) is an image processing function such as differentiation between neighboring pixels at location, (m, n). This simple approach includes floating point operations for $r \sin\theta$ and $r \cos\theta$; thus, numerous computations are required. Moreover, depending on the size of Δr, a certain pixel might be omitted or unnecessarily processed twice.

```
for (r = 0; r < 400; r += Δr)
{
    x = x_s + r cos θ;
    y = y_s + r sin θ;
    processing(round(x), round(y));
}
```

(a)

```
ε = 0,   y = y_s;
for (x = x_s; x < x_e; x++)
{
    putpixel(x, y);
    if(ε + m < 0.5)
        ε = ε + m;
    else
        y++,   ε = ε + m − 1;
}
```

(b)



(c)

Fig. 3. Algorithm comparison: (a) Simple radial processing algorithm for a directional angle θ, (b) Pseudocode for the Bresenham line drawing algorithm, (c) Illustration of the Bresenham algorithm

To solve these problems, the computation of the well-known Bresenham raster algorithm is adopted for the radial processing in this paper. The Bresenham line drawing algorithm determines the points in discrete integer space in order to plot an approximate line between two given points (x_s, y_s) and (x_e, y_e). The pseudocode representation of the Bresenham raster algorithm is given in Fig. 3 (b) assuming that the slope $m = (y_e - y_s)/(x_e - x_s)$ between two points is $0 \le m \le 1$. In this case, the reference coordinate is x. Starting from x_s, the value of x is incremented by 1 and the corresponding error ε in y is repeatedly increased by m. According to the fraction of the error, a pixel (x, y) is marked as an approximation point on the line at each step(Fig. 3 (c)). In the pseudocode, $putpixel(x, y)$ is the marking function. The line of $m > 1$ is symmetrical to that of $0 \le m \le 1$ with respect to the diagonal line $y = x$. Thus, the algorithm described in Fig. 3 (c) can be applied to the line of $m > 1$ simply by switching the variables x and y [5].

The radial processing proposed in this paper adopts the computational method of the Bresenham raster algorithm, as explained in Fig. 4. Without loss of generality, we assume 640×480 for the size of the image. It should be noted that the optical center, (x_c, y_c), does not coincide with the image center at $(320, 240)$. For a certain directional angle θ, a pixel position (x, y) on the line through the image center should satisfy the following: $\tan\theta = (y - y_c)/(x - x_c)$. When $0^o \le \theta \le 45^o$ or $0 \le m \le 1$, two points (x_s, y_s) and (x_e, y_e) for the radial processing based on the Bresenham algorithm are obtained as follows: Let iy_s be the round-off integer of y when $x = 0$, i.e., $iy_s = round\,(y_c - x_c \tan\theta)$ (Fig. 4). Then, if $iy_s \ge 0$, the pair of points (x_s, y_s) and (x_e, y_e) are

$$(x_s, y_s) = (0,\ iy_s),\quad (x_e, y_e) = (639,\ iy_e) \tag{1}$$

where iy_e is the round-off integer of y when $x = 639$, i.e., $iy_e = round\left\{y_c - (639 - x_c)\cdot\tan\theta\right\}$. And if $iy_s < 0$, the two points should be

$$(x_s, y_s) = (ix'_s,\ 0),\quad (x_e, y_e) = (ix'_e,\ 479) \tag{2}$$

where $ix'_s = round\left(x_c + y_c/\tan\theta\right)$ and $ix'_e = round\left\{x_c + \left(479 - y_c/\tan\theta\right)\right\}$. It should be noted that the processing region of x is $0 \le x \le 639$ when $iy_s \ge 0$ and $ix'_s \le x \le ix'_e$ when $iy_s < 0$. When $45^o < \theta \le 90^o$ or $1 < m < \infty$, it is possible to get (x_s, y_s) and (x_e, y_e) in a similar manner. It should be noted that the reference axis is y instead of x in this case. It is also possible to obtain the two points according to the method used in the case of $m < 0$. To summarize, (x_s, y_s) and (x_e, y_e) for the Bresenham-based radial processing are obtained using (1) and (2) for a certain directional angle θ, and the radial processing should be performed for all directional angles within the range $-90^o < \theta \le 90^o$.

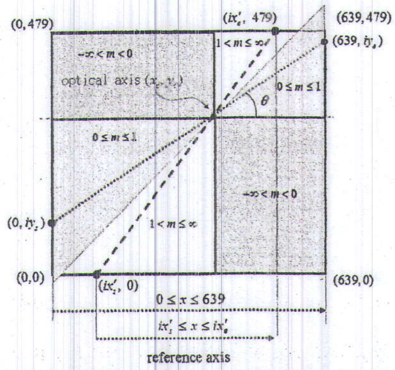
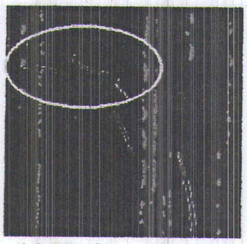


Fig. 4. Radial processing based on the Bresenham raster algorithm

3. Experiment Results

This section presents the performance of the proposed radial processing in terms of feature detection and computational efficiency and compares it with that of conventional horizontal processing. As the image processing for detecting features, simple differentiation between neighboring pixels and thresholding are employed in this paper. Fig. 5 shows the results of the feature detection for the omnidirectional structured light image in Fig. 1 (c). The resultant feature image from the proposed radial processing in Fig. 5 (b) is much brighter than the image in Fig. 5 (a) because of the density of the structured light pixels detected in the image processing. Many image features of the structured light pixels are missed, especially in the highlighted region of Fig. 5 (a). In order to compare the computational efficiencies, the repeat counts of the for-loops in the image processing algorithm are analyzed. In the brute-force image processing presented in Fig. 3 (a), processing of one dimensional line requires 400 for-loop counts, and the overall processing needs $400 \times 360^o \times 10 = 1,440,000$ counts when $\Delta\theta = 0.1^o$. In comparison, the repeat counts needed in the proposed radial processing with $\Delta\theta = 0.1^o$ are as follows: When $|m| \leq 1$, the reference axis is x and the counts are $640 \times 90^o \times 10 = 576,000$; when $|m| > 1$, the reference axis is y and the counts are $480 \times 90^o \times 10 = 432,000$. Thus, the total number of counts is $1,008,000$, which is approximately $2/3$ of the counts for the brute-force processing algorithm. It should be noted that the number of for-loop repetitions in the proposed radial processing is lower than in the above analysis since the explored interval in the reference axis is reduced according to the position of the optical center (x_c, y_c). Therefore, compared to the basic algorithm in Fig. 3 (a), the proposed radial processing algorithm is advantageous not merely because it uses only integer operations but also because it has fewer for-loop repetitions.


(a) Horizontal processing

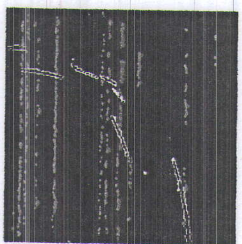

(b) Radial processing

Fig. 5. Results of image processing

4. Conclusion

This paper proposes radial directional image processing for feature detection in omnidirectional images. To detect edge feature points in a circular omnidirectional image, radial processing from an optical center is more suitable than conventional horizontal or vertical directional processing. The computational structure of the radial processing algorithm suggested in this paper is based on the well-known Bresenham raster algorithm, which uses only simple integer operations with fewer repeat counts and thus makes the overall computation highly efficient. Through experiments on an omnidirectional structured light image, it is verified that the proposed radial processing algorithm exhibits better performance in regards to edge feature detection of structured light pixels.

5. Acknowledgments

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2011-0009113).

References

[1] Nayar, S., Catadioptric omnidirectional camera, *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition* (1997), 482-488

[2] Jain, R., Kasturi, R. and Schunck, B., *Machine Vision*, McGraw-Hill (1995)

[3] Joung, I. and Cho, H., An active omnidirectional range sensor for mobile robot navigation, *Control Engineering Practice*, 6(3) (1988), 385-393

[4] Yi, S., Choi, B. and Ahuja, N., Real-time omnidirectional distance measurement with active panoramic vision, *Int'l Jour. of Control, Automation, and Systems*, 5(2) (2007), 184-191

[5] Bresenham, J., Algorithm for computer control of a digital plotter, *IBM Systems Journal*, 4(1) (1965), 25-30

[6] Shin, J. and Yi, S., An Image Processing for Omnidirectional Image Based on the Bresenham Raster Algorithm, Communications in Computer and Information Science 261 (2011), 8-16

**Corresponding author: Sooyeong Yi, Professor
Department of Electrical Engineering,
Seoul National University of Science and Technology,
232 Gongreung-Ro, Nowon-Gu, Seoul 139-743, Korea(South)
E-mail: suylee@seoultech.ac.kr