APPLICATIONS OF THE SPARSE HOUGH TRANSFORM FOR LASER DATA LINE FITTING AND SEGMENTATION [*]

Z. Song, Y. Q. Chen, K. L. Moore and L. L. Ma

Abstract

This paper proposes a novel algorithm called the sparse Hough transform, which is shown to have better performances than the standard Hough transform for sparse input data collected from a laser on a mobile robot. In the context of laser sensing and perception for autonomous ground robots, this paper studies performances of the sparse Hough transform and compares it with other segmentation and fitting algorithms. Pseudo-code for the algorithm, theoretical analysis, computer simulations, hardware experiments, and experimental analysis of the sparse Hough transform are presented.

Key Words: Hough transform, sparse Hough transform, Log-Hough transform, mobile robot.

1 Introduction

For an intelligent mobile robot, the perception of its environment via suitable sensing capacities plays a key role [2, 15]. Many perception algorithms for

[*] Accepted by ACTA Press *International Journal of Robotics and Automation.* The authors are with the Center for Self-Organizing and Intelligent Systems (CSOIS), Department of Electrical and Computer Engineering, 4160 Old Main Hill, Utah State University, Logan, UT 84322-4160, USA. This work is supported in part by U.S. Army Automotive and Armaments Command (TACOM) Intelligent Mobility Program (agreement no. DAAE07-95-3-0023). Corresponding author: Dr. YangQuan Chen. E-mail: yqchen@ieee.org.

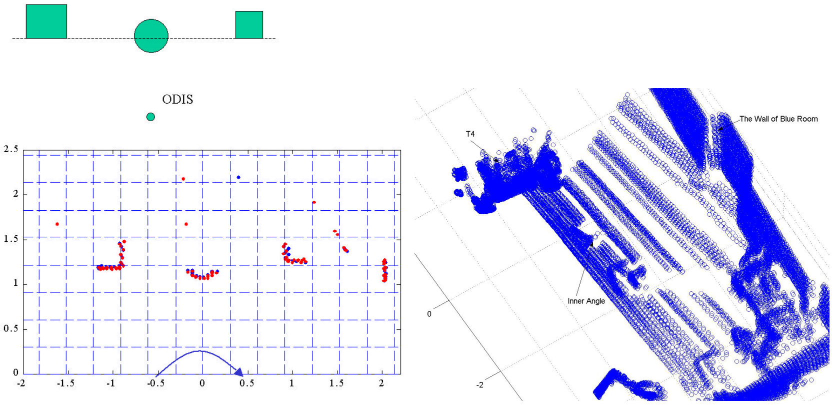
robots are similar to computer vision methods and can benefit from techniques used for image analysis [17]. However, the data collected by the laser sensors commonly used in robotics have different characteristics than generic image data. Hence, algorithms designed for image data may not be efficient for laser-based perception in robotic applications. For example, the proposed laser-data segmentation and fitting method in [1] for localization takes advantage of the fact that laser data are sequential with respect to the scanning angle. This characteristic is not available for generic image data. Thus algorithms originally developed for generic image data may not be suitable for laser data.

In this paper, we take advantage of a different characteristic of laser data than its sequential nature, namely, that for some applications laser data can be considered to be sparse. To see this, consider Fig. 1, which shows the empirical laser data collected from two different robots developed at Utah State University (USU). Figure 1(a) shows data collected by ODIS, a small, man-portable mobile robotic system used for autonomous or semi-autonomous inspection under vehicles in a parking area [14, 13, 4]. By rotating while firing the 1D laser, ODIS can scan the laser beam and get 2D data with its 1D laser scanner. The top of Fig. 1(a) shows the setting of objects in the scan region. The bottom of Fig. 1(a) is the corresponding data that collected by the 1D laser scanner. Figure 1(b) shows laser data collected by the T4/T2E robot using a 2D SICK laser scanner mounted on a gimbal system. The gimbal system can change the pitch angle of the laser scanner smoothly while the laser is firing, thus making it possible to collect a 3D point cloud. The data in Figure 1(b) were collected in our lab using the T4 robot with various objects in the laser's field of view.

The key observation we note about Fig. 1 is that the laser data are sparse: the 1D laser data are two-dimensional in rectangular coordinates, but are one-dimensional in polar coordinates. This is due to the fact that laser beam cannot see through objects. Similarly, the 2D laser data are two-dimensional in polar coordinates.

Because laser data such as those shown in Figure 1 are often processed through segmentation and line-fitting algorithms, as part of higher-level perception processes, it is common to use the the Standard Hough Transform (SHT), the line detection/fitting algorithm patented by P. Hough [8] in 1962. However, given the sparse nature of laser data, we are motivated to develop the Sparse Hough Transform (SPHT) proposed in this paper. Our analysis and experimental results indicate that the SPHT has better performances than SHT, in terms of speed and required memory, when input data are sparse enough. Compared to other related algorithms, such as the Log-Hough transform (LHT), least squares (LS), and Kalman filtering, the SPHT is unique and can be a good option for certain robotic applications, such as the laser-based perception example presented above.

The remainder of the paper is organized as follows. Sec. 2 introduces the fundamentals of the SHT and its uses for line segmentation and fitting. Sec. 3 presents the SPHT. Pseudo-code, analysis, and comparisons with other variants of the HT are given. Sec. presents some experimental results. Finally, section 5 concludes the paper.



(a) Data collected by 1D laser. (b) Data collected by 2D laser.

Figure 1: Laser data.

2 A Brief Review on Line Fitting and Segmentation for Robot Laser Data

2.1 The Hough Transforms

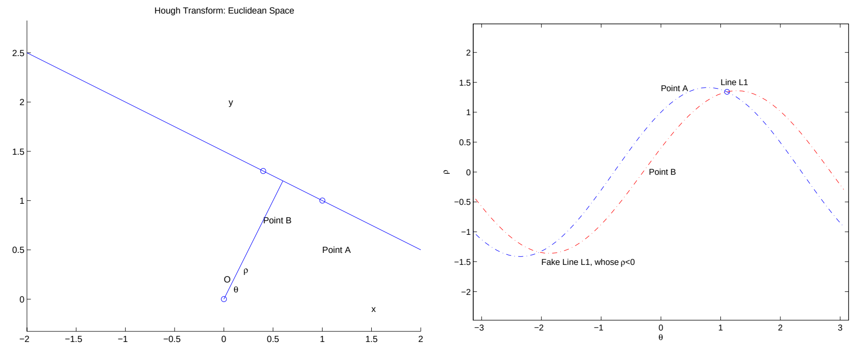
As noted, the standard Hough transform refers to the line detection/fitting algorithm patented by P. Hough [8] in 1962. Today, the term “Hough transform” (HT) refers to a large class of algorithms in the area of computer vision and pattern recognition [9] that are not limited to line detection, but can also be used to detect shapes such as circles [3], ellipses [7], and more complex binary patterns [6]. Even for line detection, many improvements have been proposed to improve the performances of the SHT, such as the probabilistic Hough transform [11], the progressive probabilistic Hough transform [12], the Log-Hough transform (LHT) [5], the randomized Hough transform [20], the connective randomized Hough transform [10], and the parallel Hough transform [19]. In this paper we propose one more HT variant to be added to this list: the sparse Hough transform.

The SHT algorithm can be described as follows. The SHT transfers each point of a data set in Euclidean space into one curve in the Hough accumulation space (also called the accumulation space) by the equation

$$\rho = x\cos(\theta) + y\sin(\theta), \tag{1}$$

where (x, y) is a point in the Euclidean space, (ρ, θ) is the line that passes through (x, y), as shown in Fig. 2(a). The distance between the `Line L` and the origin of the coordinate system is ρ, which has a θ angle between itself and the x axis. Thus, each (ρ,θ) represents one, and only one line in the Euclidean space passing through the point (x, y). However, if ρ can be negative, then there are two ρ and θ tuples to represent a single line, i.e., (ρ,θ) and $(-\rho,2\pi-\theta)$. From Eq. 1, it is easy to see that, for each tuple (x, y), there are many (ρ, θ) tuples associated with it, as depicted in Fig. 2(b). Notice that for a point in Cartesian space, such as `Point A` in Fig. 2(a), there are an infinite number of lines that can pass through the point. However, given another point in this space, e.g., `Point B` in Fig. 2(a), there exists one and only one line that connects both of the points. In Euclidean space, this line, (`Line L`), is the combination of infinite number of points. But, in the accumulation space, shown in Fig. 2(b), this line is just a point, while `Point A` and `Point B` become curves with infinite number of points. Both of these two curves pass the point that represents `Line L`, thus `Line L` is the line that connects `Point A` and `Point B`.

Note that in practice, both the Euclidean space and the accumulation space are discrete. However, while the Euclidean space is binary, the accumulation space is a grey image (based on the number of curves that pass through a point). If there are many data points in the Euclidean space, then there might be many intersection points in the accumulation space. To select the best fit line, the value of each pixel in the accumulation space is used as the index of the “fitness,” i.e., the number of curves that passed this point. The pixel with the highest value in the accumulation space is the best fit

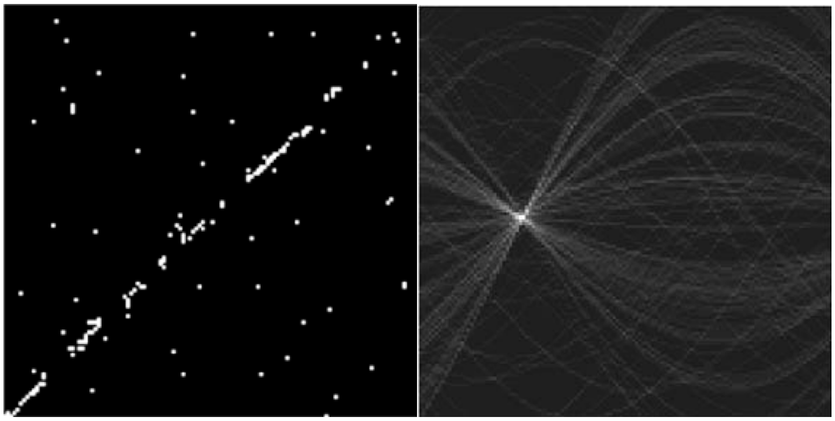


(a) A point in the Euclidean space. (b) A point in the accumulation space.

Figure 2: Standard Hough transform.

line. The process of counting the number of curves passing through a point in accumulation space is called "voting." The point with the highest number of votes (largest grey value) "wins" and is taken as the best-fit line. An example is shown in Fig. 3. Figure 3(a) shows a line in Euclidean space with "salt-and-pepper" noise. Figure 3(b) shows the corresponding accumulation space. The approach of the SHT is to take the bright point seen in the figure as the "best-fit" line for the data shown in Fig. 3(a).

Generally speaking, the attraction of the HT is its robustness to sensor noise, as illustrated in Figure 3. However, this robustness comes at the expense of slow computation. The computational cost of the SHT is $\mathcal{O}(n^3)$, where n is the number of data points. Many of the variants of the SHT have been motivated by trying to speed up the SHT computation. Among the none-standard Hough transforms, the Log-Hough transform (LHT) [5] uses an algorithmic approach to reduce complexity with respect to the SHT. Probability-based Hough transform methods also reduce computation, but by working with only a "proper" part of the data [11, 12, 20, 10]. The basic idea of the probability-based Hough transform is to randomly choose part of the data in the Euclidean space and process them by the SHT algorithm. Different methods have been proposed to increase the probability of choosing the aligned pixels to process. The details of different data selection strategies are beyond the scope of this paper. However, the concept of selecting data is important. The SPHT proposed below also attempts to reduce computation time by "properly" selecting the data to use. However, unlike other



(a) A line in a binary image with strong noises.

(b) The accumulation space.

Figure 3: The Hough transform for a line with noises.

fast strategies, which were developed generally for image data processing applications only, we attempt to exploit the sparseness of the data typically obtained from lasers.

2.2 Methods for Laser Data Line Segmentation and Fitting

HT methods can be used for both line segmentation and fitting. Of course, there are other, related methods that have the capability of either line segmentation or fitting. Segmentation can be achieved using geometric approaches [16]. This type of approach segments data mainly based on the tensor or the norm. No special characteristics of the laser data are considered. Sensor noise is also not considered explicitly.

An extended Kalman filter (EKF) can also be used for laser data segmentation. In [2], the laser beam angle θ is used as the time n to construct a dynamic system given by

$$x[(n+1)\Delta_\theta] = Ax[n\Delta_\theta]$$

$$y[n\Delta_\theta] = Cx[n\Delta_\theta]$$

where A and C are determined by the expected segmentation shapes, such as a line or a circle. $y[n\Delta_\theta]$ is a prediction on the distance measured laser beam. If the predicted distance is far from the measured distance, a segmentation point is generated. This type of EKF-based method takes into account the sequential nature of the data and can also be constructed based on the characteristics of the sensor noise.

Least square (LS) methods have also been applied to line fitting. This approach requires segmented data as the input. Using this method, the line fitting problem is equivalent to the solution of this equation

$$\begin{pmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \vdots & \vdots \\ x_n & y_n \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix},$$

where (x_i, y_i) are points on the prospective fitting line and (a, b) are the parameters of the line. A single noise point may significantly change the estimation on (a, b). On the other hand, the HT is robust to a small number of noise points, no matter how far away those points are from the fitting line. Thus both segmentation and line fitting can be achieved together by HT methods.

3 The Sparse Hough Transform

In this section we introduce and describe the SHT and then compare it to the SHT and other HT variants.

3.1 The Algorithm of Sparse Hough Transform

The SPHT is proposed to improve the performance of the SHT using the sparsity of the laser data. The resolution and fitting performance of the SPHT algorithm are exactly the same as that of the SHT. However, the SPHT saves time and memory over that of the SHT in the case of sparse data. The fundamental difference between the SPHT and the SHT is that the SPHT does not use a 2D array to record the votes in the accumulation space. Instead, it uses a 1D array for the vote counting. Since in case of sparse data most lines passing through only one point are not the best-fit line, we only need to compute those lines that pass through at least two points in the image, thus reducing computer memory and speed over the

```
NumLen=0
for i=1 to n−1
  for j=i+1 to n
    $\alpha$ = atan($\frac{Dat(i).y - Dat(j).y}{Dat(i).x - Dat(j).x}$)
    $\rho_0$ =Dat(i).x cos($\alpha$ −$\pi$/2) + Dat(j).y sin($\alpha - \pi/2$)
    $\theta$= $\alpha$−sign($\rho_0$) $\pi/2$
    $\rho = |\rho_0|$
    Cell$\theta$=floor($\theta/\Delta\theta$)∗$\Delta\theta$
    Cell$\rho$=floor($\rho/\Delta\rho$)∗$\Delta\rho$
    If the line $n$, where $n \in [1, NumLen]$, is close enough
        to (Cell$\rho$,Cell$\theta$), increase the vote of Lines($n$)
        by one, otherwise NumLen=NumLen+1, and add a
       line (Cell$\rho$, Cell$\theta$) to Lines with the vote
       equals to one.
  end for
end for
```

Figure 4: The pseudo-code for basic SPHT implementation.

SHT. Based on this argument, a pseudo-code implementation of the SPHT is shown in Fig. 4.

A graphical interpretation of the SPHT is plotted in Fig. 5. In Fig. 5(a) there are three points (point A, B, and C) and three lines that connect them (line AB, BC, and AC). The main difference between the SPHT and the SHT is that the accumulation space of the SPHT is represented by a 1D array, with each element representing a line in Euclidean space. The SPHT algorithm computes the (ρ, θ) tuples for each line in Fig. 5(a) and then constructs an array with three elements to store the tuples of these three lines. This array also stores the "vote" of each line, which represents the "fitness," just like the value of each cell in the accumulation space of the SHT. It is easy to see that if the input data are really sparse, the number of intersections in the accumulation space is limited, and thus the SPHT saves much more memory compared to SHT.

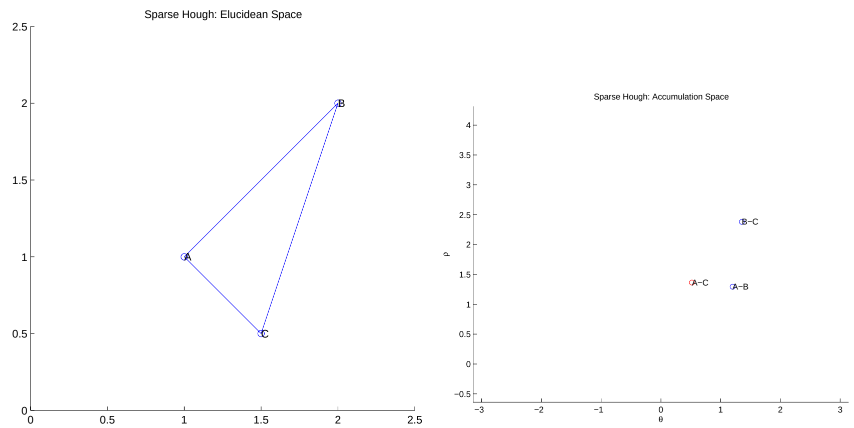


Figure 5: Interpretation on SPHT.

3.2 Analysis of Speed and Memory

Suppose that the angle resolution of scanned laser data is denoted by $\Delta\theta$ and the normal distance resolution is denoted by $\Delta\rho$. For the SHT, $M \times N$ cells are required for the accumulator space, where $M = \pi/\Delta\theta$ and $N = \rho_{max}/\Delta\rho$, where ρ_{max} is the longest normal distance. In other words, the memory size is $n \times M \times N$, where n is the word length of each cell. If there are K valid pixels in the input data, the computation complexity is $\mathcal{O}(K \times M)$. For the SPHT, if there are only K valid points in the input, $K \ll M$, and $K \ll N$, the maximum number of fitted lines will be $K \times (K-1)/2$. The resulting computational complexity is $\mathcal{O}(K \times (K-1)/2)$. The actual memory requirement depends on the number of lines passing through the data points in the input data. Normally, we expect that the number of lines, L, satisfies $L \leq K(K-1)/2$. In one experimental example described in subsection 4.2, $L = 149$ and $K = 46$. Thus, the ratio of $K(K-1)/2$ to L is 6.9 for this example. This is a typical example for robotic laser data processing.

Consider the Log-Hough transformation (LHT) proposed in [5]. The key feature of this algorithm is its fast speed, with the complexity of $\mathcal{O}(n)$, where n is the number of points on the image plane. The LHT uses the concept of a logarithmic cosine curve. Due to the logarithm computation, multiplication operations can be changed into shift and addition. Thus, once one standard logarithmic cosine curve is computed and stored in a look-up table, the mapping from the image plane to the accumulation plane is achieved simply by shifting that standard curve and adding it to the accumulation plane, while the shifting value is computed with respect to the coordinate of a point on the image plane. However, the logarithm computation also introduces an undesired feature: the resolution of LHT is nonuniform. The near-field resolution of the LHT is better than that of the far field. The following points compare the LHT with the SPHT:

- **Complexity:** The complexity of the LHT is $\mathcal{O}(K)$ and that of the SPHT is $\mathcal{O}(K(K-1)/2)$, where K is the number of valid points.
- **Memory:** The exact memory requirement of the SPHT depends on L, the number of fitted lines, which has an upper bound of $K(K-1)/2$. For the LHT, we need K segments on the range of the accumulation space with

  $$K = \lceil \log(\frac{r_1 - r_0}{r_0})/\delta \rceil$$

  where the notation $\lceil x \rceil$ indicates the maximum integer that is not more than x. The distance of interest stretches from r_0 to r_1. δ is the angle resolution of the laser system. This K is comparable to the N of the SHT. The N for the SHT can be expressed by the same notations:

  $$N = \lceil \frac{r_1 - r_0}{r_0} / \delta \rceil.$$

- **Resolution:** The distance resolution of the LHT is nonuniform. The resolution of the near field is better than that of the far field. However, the distance resolution of the SPHT and the SHT are uniform with respect to the distance, and the resolution of fitting output is as high as the range resolution of the laser scanner. This means that if the range resolution of a laser scanner is 1 cm, and its angle resolution is 0.5°, the LHT will get a range resolution of 0.88 cm for the objects at 50 cm away, and 16.6 cm for the distance of 1000 cm. Under the same situation, the distance resolution of the SPHT and the SHT will always be 1 cm, which is independent of the distance of detected objects.
- **Compatibility:** Due to the discussion in the “resolution” entry, the compatibility of the SPHT to the SHT is better than that of the LHT to the SHT. This undesirable feature of the LHT limits its applications. Actually, there exist cases where the SHT cannot be replaced by

the LHT, or at least cannot be replaced by the LHT without sacrificing performance. One example where this is true is the bumper fitter problem discussed below in subsection 4.2.

Figure 6 shows a comparison of the memory requirements, complexity (number of operations), and distance resolution of the SPHT, the LHT and the SHT, assuming that the area of interest stretches from $r_0 = 50$ cm to $r_1 = 1000$ cm, the angular resolution δ is 0.5°, and the distance resolution of the laser scanner is 1 cm. Note that the x axis of the upper and middle subplots are scanning angles. In our application, instead of processing the data from 360°, we generally process only a narrow area where objects are expected. As we can see from the figure, the smaller the scanning angle, the better the performance of the SPHT. The upper figure shows that when the data are sparse, i.e., the angle of interest is narrow, the SPHT demands the least amount of memory. In the middle subplot of Fig. 6, we can roughly estimate the speed of the algorithms based on their complexity. The SHT is slow, while the SPHT is faster at the beginning but later becomes slower than the SHT, and the LHT is the fastest when more data points come in. In the bottom subplot of Fig. 6, we can see that the range resolution of the SHT and the SPHT are exactly the same. The memory and speed benefits from the LHT are at the expense of a lower and distance-dependent resolution.

An important comment is that the complexity of the algorithms is related to their speed, but no simple relation exists between the two. The complexity is an index of the number of operations to be done, without any indication of the computation time of each operation. The actual computation time of an algorithm is proportional to its complexity, with the ratio dependent on the computer being used. Thus, algorithms with same number of operations do not necessarily have the same speed. For example, an algorithm of 100 addition operations generally needs less time than another that needs 100 multiplications, since addition is faster on most computers. The goal of complexity analysis is to understand the performance in a large scale. An algorithm with a complexity of $\mathcal{O}(n^3)$ will definitely take more time than an algorithm with $\mathcal{O}(n)$ complexity when n is large enough, no matter how fast each operation of the first algorithm is. Thus, the middle subplot of Fig. 6 should not be interpreted as that the SPHT is faster than the SHT when the scanning angle is 360° or less. The correct interpretation is that the number of operations for the SPHT is less than that of the SHT under the same situation. The speed comparison of the SPHT and the SHT is presented below in subsection 4.2, where experimental data are analyzed.

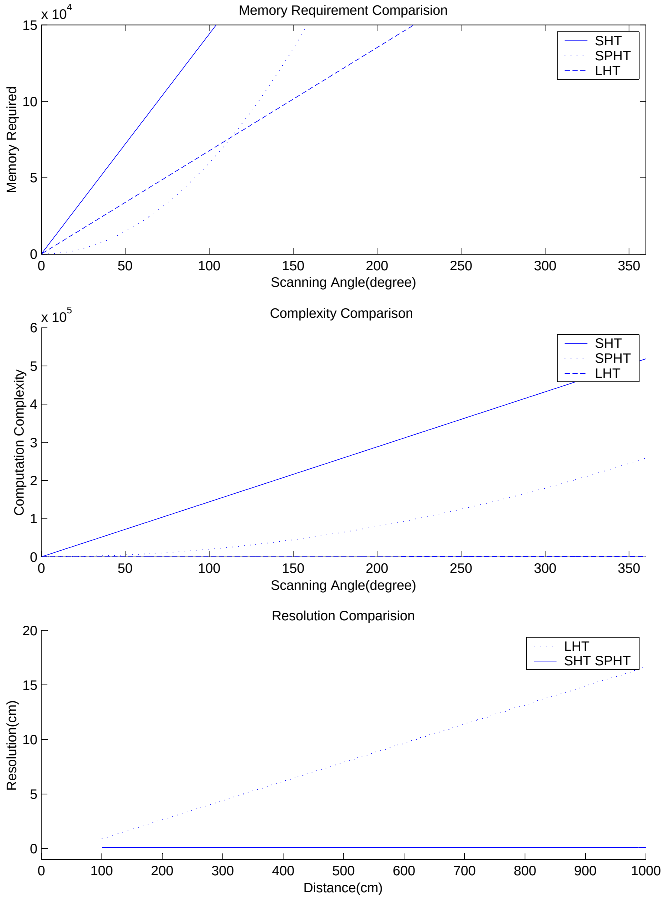


Figure 6: The comparison of sparse, log and standard Hough transform.

4 Some Experimental Results

In this section we present some experimental results, beginning first with the example of fitting a cylinder with 1D laser and discussing the fitting of an automobile bumper using 2D laser data. Here we present only the essential features. For more details see [?].

4.1 The Application of the SPHT on Cylinder Fitting of 1D Laser Data

One important advantage of the SHT or SPHT algorithms is their robustness. This feature can be shown in the following example, where we use a cylindrical dustbin as a landmark for the ODIS robot. Behind the dustbin there is a wall. The raw laser data are shown in Fig. 7. After processing all the data by the SPHT, the best-fit line is available, which is indicated by the dotted line in the figure. Once the line is fitted, it is easy to verify whether a point is on this line. After eliminating those on-line points, there are two types of points left: wild points and points corresponding to the cylinder to be fit. Wild points are the points that are separated significantly from other data points. The removal of those wild points is trivial: just delete the points that are too far away from their neighbor points. In Fig. 8, the wild points are indicated by stars, $(*)$, the points on the best-fit line are denoted by right triangles, $(\triangleright)$, and the points left for circle fitting are indicated by circles, $(\circ)$. After processing the "circle-fitting points" by an algebraic circle fitting method (see [?]), the fitted circle is shown in Fig. 8.

Note that in this example, the SPHT fits a line as well as segments the data. Other methods, such as Least square (LS) methods or Kalman filter-based methods, will typically require segmentation before doing line fitting, thus the data processing may be more complex.

4.2 The Experiment on Bumper Fitter for Laser Servoing

In the following example, a module called "bumper fitter" is implemented by the SHT and the SPHT to provide the T2E/T4 robots with a laser servoing feature for a better license plate recognition (LPR) rate. Part of this work is described in [18]. A key feature of this application is that we need to continually update the fit as new data arrives. This highlights another advantage of the SHT and the SPHT: they support progressive fitting. Data points can be added to the accumulation space (for the SHT) or accumulation array (for the SPHT) recursively, without wasting computations that have been done before. Least square algorithms, on the other hand, typically start over again whenever new data arrive, thus wasting previous computational effort.

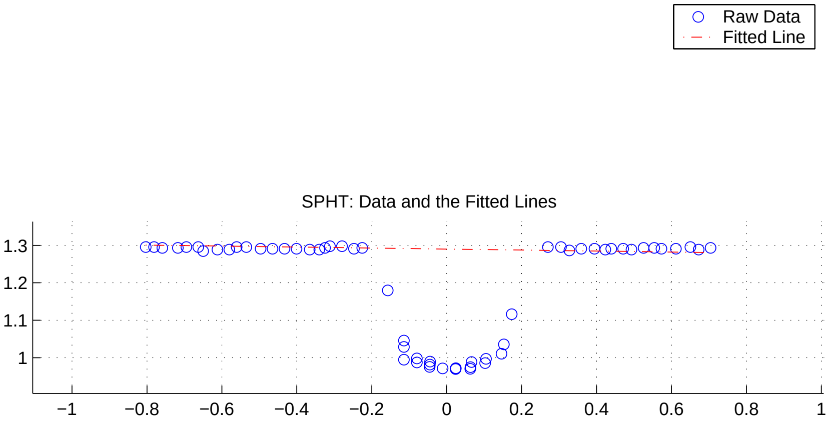


Figure 7: The line segmentation for a point cloud by SPHT.

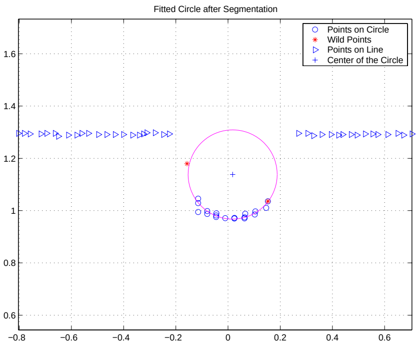


Figure 8: Fit a circle from a point cloud.

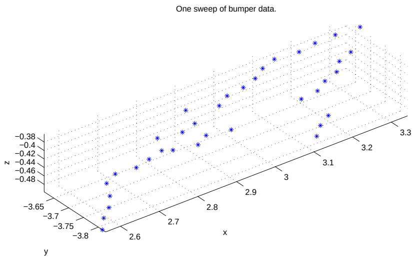


Figure 9: 3D data points of a bumper.

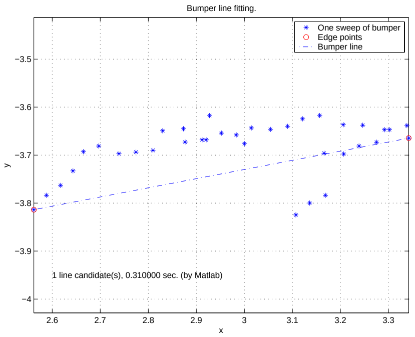


Figure 10: A bumper and its fitted bumper line.

The LPR system includes an IR camera/illumination system and a program that can identify license plate numbers of automobiles by analyzing the images obtained from the camera. To guarantee the recognition rate, the robot (e.g., the camera) must be positioned accurately with respect to the bumper of the automobile. The T2E/T4 robots achieve this via a laser servoing system that works as follows: at each sample time, a new sweep of laser data is collected, projected to 2D, segmented into different parking stalls, added to the accumulation space, and the latest best-fit line is computed. Those fitted lines are the bumpers of the vehicles. The robot thus can adjust its track according to the location of the bumper lines and then fire the LPR system at the proper positions.

Figure 9 shows the experimental laser data from a scan of a bumper and Fig. 10 shows the fitted bumper line obtained using the SPHT. The two points on the edge in Fig. 10 are the two extreme points on the fitted bumper line. Note that the data points in Fig. 10 were added progressively.

In our experiments the distance resolution is set to be 6 cm and the angular resolution is 9°, according to the requirements of the LPR system. In one of the fitting processes, the robot collects 50 data points in the target parking stall.

Due to system limitations, we cannot run the SPHT or the SHT alone on our robots to compare their performances. Instead, we ran the same C++ implementations on a 1.2Ghz AMD Athlon computer with 512Mb memory and Windows 2000. For 50 data points, the SPHT takes 1.563ms, and the SHT takes 0.938ms. Meanwhile, the memory requirement for the SPHT is 149×3 integer variables, and that of the SHT is 46×46. That is, the SHT is 1.6 times faster than the SPHT, but takes 4.7 times memory more than the SPHT in this example.

Next, inspired by the probabilistic-based Hough transforms, we improved the SPHT by sampling: if the indexes of two data points are closer than certain threshold, they are skipped. Remember that the index of 2D laser data is proportional to the scanning angle β, so the closeness of the index indicates that associated points are physically close. This modification reduces the memory requirement and improves speed. With the same system configurations, the sampled SPHT takes 0.297ms and the SHT takes 0.875ms this time. The memory used for the SPHT is 78×3 and that of the SHT is still 46×46. So the sampled SPHT is 3 times faster than the SHT and the SHT uses 9 times memory more than the sampled SPHT in this case. Thus, the sampled SPHT is seen to be a good replacement of the SHT when the input is sparse.

5 Conclusions

The major contribution of this paper is the proposal and study of the sparse Hough transform. The performance (speed and memory requirements) of the sparse Hough transform is better than that of the standard Hough transform when the input data are sparse. Meanwhile, the transforms of the two are identical from an input-output perspective in the sense that the two transforms are interchangeable. Pseudo-code, theoretical analysis, computer simulations, hardware experiments, and experimental analysis of the sparse Hough transform were presented. We can conclude that under the context of robot environment perception with laser sensors, the sparse Hough transform is often a better option for segmentation and line fitting than the SHT.

Acknowledgment

The authors would like to acknowledge the fruitful discussions with CSOIS members.

References

[1] M. Adams, S. Zhang, and L. Xie. Particle filter based outdoor robot localization using natural features extracted from laser scanners. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 2, pages 1493 – 1498, April 2004.

[2] M. D. Adams. *Sensor modeling, design and data processing for autonomous navigation*, volume 13. World Scientific Co. Ltd., Farrer Road, Singapore, 1999.

[3] T. J. Atherton and D. J. Kerbyson. Size invariant circle detection. *Image and Vision Compution*, 17:795–803, September 1999.

[4] M. Davidson and V. Bahl. The scalar ϵ-controller: A spatial path tracking approach for ODV, Ackerman, and differentially-steered autonomous wheeled mobile robots. In *Proc. of IEEE Int. Conf. on Robotics and Automation*, Seoul, Korea, 2001. IEEE.

[5] B. Giesler, R. Graf, R. Dillmann, and C. F. R. Weiman. Fast mapping using the Log-Hough transformation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 3, pages 1702–1707, 1998.

[6] N. Guil and E. L. Zapata. A new invariant scheme for the generalized Hough transform. In *IASTED International Conference on Signal and Image Processing*, pages 88–91, Orlando, FL, November 1996.

[7] N. Guil and E. L. Zapata. Lower order circle and ellipse Hough Transform. *Journal Pattern Recognition*, 30(10):1729–1744, October 1997.

[8] P.V.C. Hough. Methods and means for recognizing complex patterns. U.S. Patent No. 3069654, 1962.

[9] J. Illingworth and J. Kittler. Survey: A survey of the Hough Transform. *Journal on Computer Vision, Graphics, and Image Processing*, 44:87–116, October 1998.

[10] H. Kalviainen and P. Hirvonen. Connective randomized hough transform (crht). 1995.

[11] N. Kiryati, Y. Eldar, and A. M. Bruckstein. A probabilistic hough transform. 24(4), 1991.

[12] J. Matas, C. Galambos, and J. Kittler. Progressive probabilistic Hough Transform. In *Proceedings British Machine Vision Conference*, September 1998.

[13] K. Moore, N. Flann, S. Rich, M. Frandsen, Y. Chung, J. Martin, M. Davidson, R. Maxfield, and C. Wood. Implementation of an omnidirectionsl robotic inspection system (ODIS). In *Proc. of SPIE Conf. on Robotic and Semi-Robotic Ground Vehicle Tech.*, Orlando, FL., May 2001. SPIE.

[14] K. L. Moore and N. S. Flann. A six-wheeled omnidirectional autonomous mobile robot. *IEEE Control Systems Magazine*, 20(6):53–66, December 2000.

[15] I. R. Nourbakhsh. *Interleaving Planning and Execution for Autonomous Robots.* Kluwer Academic Publishers, Boston, 1997.

[16] D. L. Page. *Part Decomposition of 3D Surfaces.* PhD thesis, The University of Tennessee, Knoxville, May 2003.

[17] W. E. Snyder and H. R. Qi. *Machine Vision.* Cambridge University Press, 2004.

[18] Z. Song, K. Moore, Y. Q. Chen, and V. Bahl. Two-dimensional laser servoing for precision motion control of an ODV robotic license plate recognition system. In *Proceedings of International Society of Optical Engineering*, volume 5083, San Diego, CA, April 2003.

[19] R. S. Stephens. Parallel hough transform algorithm performance. 1991.

[20] L. Xu and E. Oja. Randomized hough transform (rht): Basic mechanisms, algorithms,and computational complexities. *Journal on Computer Vision, Graphics, and Image Processing*, (2), March 1993.