Some Sensing and Perception Techniques for an Omnidirectional Ground Vehicle with a Laser Scanner

Zhen Song, YangQuan Chen, Lili Ma and You Chung Chung

Abstract—**This paper presents some techniques for sensing and perception for an omnidirectional ground autonomous vehicle equipped a laser scanner. In an assumed structured environment, the sensor data processing methods for both 1D and 2D laser scanner, are discussed. Raw data are segmented to lines, circles, ellipse, planes and corners by task depended segmentation algorithms. Each subset of data is then fit by known template shapes as listed above. An arbitrator is used to discriminate the shape for final object identification.**

Key Words: Hough transform; line fitting; corner fitting; arc/circle fitting; ellipse fitting; algebraic fitting; Omnidirectional vehicle (ODV).

I. Introduction

A. Motivation

For any intelligent mobile robot, the perception of its environment via suitable sensing capacities plays a key role [1], [2]. The environment perception depends largely on the properties of environment itself. Basically, the environment can be assumed to be either static (deterministic) such as office corridor or dynamic (changing) such as a parking lot with vehicles in and out dynamically. In terms of deterministicism of the environment, there are two types: deterministic environment and uncertain environment. With combination, we have four types of environments: static deterministic, dynamic deterministic, static uncertain and dynamic uncertain. For static and dynamic deterministic environments, the sensing and perception task is easier with a help of a map and the known motion patterns of the objects in the environment. For uncertain environment, it seems that no *off-the-shelf* solution exists for general sensing and perception tasks for mobile robots [1]. This is because, sensing and perception solutions are usually *platform-specific* for mobile robot with given or existing sensing capability [1], [2]. However, there are still some common and possibly low-level solutions applicable for all platforms. This paper attempts to present some commonly used techniques for sensing and perception for mobile robot equipped with a 1D or 2D laser scanner in static uncertain environment. Our test platform is an omni-directional ground autonomous vehicle developed in the CSOIS at Utah State University (USU).

Our approach for static uncertain environment perception is based on the following steps (1) laser scanner raw data collection, (2) segmentation, (3) fitting from the object library (or object templates), (4) object arbitration or decision. For step-1, pre-processing is performed to reject some obviously abnormal points or outliers. In step-2, the range data array from a laser scan is to be segmented to form some subsets of point clouds containing essential information for the object fitting. We assume that a library of objects such as line, corner/rectangle, arc/circle/ellipse etc. are available from the prior knowledge about the static uncertain environment. Then, in step-3, we fit the through the library since we do not know in advance which object the given point cloud represents. The arbitration or decision on which object in step-4 is done by a rule based comparison of the fitting scores. We believe that this procedure is natural and can be regarded as a common piece of utility in sensing perception using a laser scanner. In each step of the above proposed procedure, there are many technical as well as theoretical challenges. In the present paper, we shall be more focused on step-2 and step-3.

The major contribution of this paper is the collection of some commonly used object fitting algorithms well tested in our experiments with our C++ and Matlab codes publically available [1].

B. The USU ODIS/T4 Platform

The USU ODIS robot is a small, man-portable mobile robotic system that can be used for autonomous or semi-autonomous inspection under vehicles in a parking area [3]. Customers for such a system include military police (MP) and other law enforcement and security entities. The robot features (a) three "smart wheels" in which both the speed and direction of the wheel can be independently controlled, (b) a vehicle electronic capability that includes multiple processors, and (c) a sensor array with a laser, sonar and IR sensors, and a video camera. ODIS employs a novel parameterized command language for intelligent behavior generation. A key feature of the ODIS control system is the use of an object recognition system that fits models to sensor data. These models are then used as input parameters to the motion and behavior control commands. Figure 1 shows the mechanical layout of the ODIS robot. The robot is 9.8 cm tall and weighs approximately 20 kgs. Key ODIS subsystems include its mechanical, vehicle electronics (vetronics) and sensor systems. For more detailed description, see [4], [5].

Figure 2 shows the 1D laser scan data for a typical set of objects in front of the ODIS equipped with a 1D laser. Though 1D laser scanner can not change its yaw angle itself, ODIS can. To get 2D information, ODIS rotates with respect to the center of its mass while firing its 1D laser. By this way, the 1D laser system can get 2D point cloud. In our project, objects in parking lots can be decomposed into combinations of arcs/circles and lines/rectangles.

Currently, the cost of a 2D laser scanner is greatly reduced, which enables increasingly many robots be equipped with a 2D laser scanner. The 2D laser scanner is mounted on a servo system that can change the pitch angle. While

The authors are with the Center for Self-Organizing and Intelligent Systems (CSOIS), Dept. 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; Tel. 01-435-7970148; Fax: 01-435-7972003. URL: http://www.crosswinds.net/~yqchen.

[1]http://www.csois.usu.edu/code2share/isic2dlaser.zip

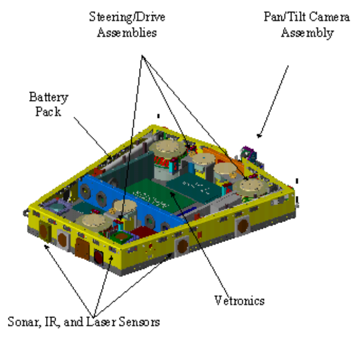

Fig. 1
THE MECHANICAL AND VETRONICS LAYOUT OF ODIS

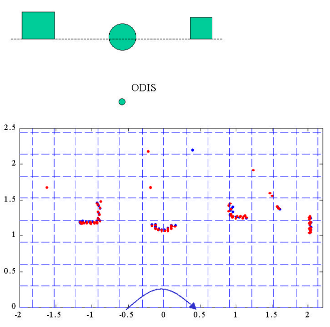

Fig. 2
1D LASER DATA (TOP: OBJECT SETTING. BOTTOM: LASER DATA)

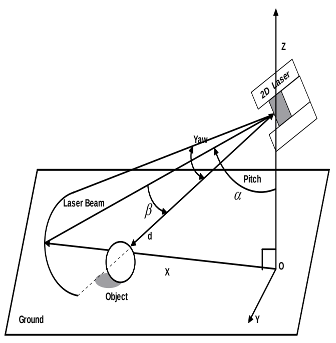

Fig. 3
THE 2D LASER SCANNER COORDINATE SYSTEM

the 2D laser scanner is shining the laser, the pitch angle can be changed smoothly. Then we can get 3D point cloud. Again, "*off-the-shelf*" algorithms for object identification or for environment features extraction have not been available although some very platform-specific schemes are proposed or tried, e.g., the research work reported in [6], [7], [8], [9], [10], [11], [12]. Figure 3 shows the coordinate system of the 2D SICK laser [2], which, for computational convenience, is different from the body frame coordinate system (BFCS) and inertial coordinate system (ICS). The height of 2D laser above ground, denoted by h, is constant. From the data of 2D laser scanner, we can get a set of raw data points (x', y') together with the corresponding pitch angle α from the pitching servo system. To convert the raw data into 3D coordinate system (x, y, z), we have $\beta = \text{atan}(y'/x')$ and $d = \sqrt{x'^2 + y'^2}$. So, $(x, y, z) = (d\cos(\beta)\sin(\alpha), d\sin(\beta), h - d\cos(\alpha)\cos(\beta))$. In Sec. III, we will introduce some object fitting algorithms such as plane fitting (Sec. III-D), 3D corner detection and fitting (Sec. III-E), and etc. for 2D laser scanner.

II. SEGMENTATION

A. Line Detection for Segmentation

Line detection is the first step to segment 1D laser data. After the line detection, we can judge if a point is on the line by checking the distance from the point to the line. In our project, it is reasonable to segment objects in parking lots into circles and lines. So, segmentation is simplified as line detection and circle fit.

Among many published methods on line detection, some took the frequency-domain approach [13] and others took time-domain approach, e.g., Hough transform [14], or color pattern analysis [15]. In our project, the input is the sequential laser range data, instead of an image. In the following discussion, we will focus on Hough transform.

Hough transform (HT) is a popular method for the extraction of geometric primitives [14], [16]. Initially, it is only an approach for line detection. Later, many variants are developed to detect circle [16], ellipse [17], or more complex binary patterns [18]. A survey on Hough transform can be found in [14].

Generally speaking, HT is robust to sensor noise at the expense of slow computation. The computational cost of the traditional HT is $\mathcal{O}(n^3)$ where n is the number of data points. Though some efforts were made to speed up the HT algorithm [19], those fast strategies were developed generally for image data processing applications only.

However, in our case, a 1D laser data set contains only *sparse* binary information in the region of interest. Furthermore, only a 1D array is required for laser data instead of a 2D matrix for the image data. More importantly, the laser data set is sequential or ordered. Taking these properties of laser data set into consideration, HT can be modified to reduce the computational time.

B. Sparse Hough Transform for 1D Line Detection

In order to make the best use of the above mentioned features of laser scanner data set, here we propose a sparse Hough transform (SPHT) that fits better to the laser data processing problem. The standard Hough transform (SHT) represents lines crossing a point by

$$\rho = x\cos\theta + y\sin\theta,$$

where ρ and θ are the length of the normal and the angle of the normal with respect to the positive x-axis. x and y are the coordinates of a point of interest in the image. ρ and θ are quantified as an accumulator matrix. The value of each cell of the matrix represents how many times there is a line, denoted by (ρ, θ), passing a point in the image. The idea of SHT is to provide an equivalent but faster and less memory-demanding approach. The inputs and outputs of the SPHT algorithm are *exactly* the same as SHT, except that in sparse cases SPHT saves time and memory. Here sparse means both the number of data points and the number of the lines passing these points are small, which is true in our case. Basically, SPHT does not use a 2D array to record the votes. Instead, it uses a 1D array for the vote-counting. Since in sparse case, most lines passing one point are not the fitted line, we can compute only those lines that pass at least two points in the image. Transferring all the lines corresponding to one point in the image plane, as SHT does, will take extra computer memory and reduce the speed.

Based on the above considerations, we present an implementation of SPHT by the following pseudo-code list: (`Dat` is the input array; `Dat(n).x` is the x value of the n-th point. `Lines` is the output array that stores (ρ, θ) of the fitted lines. `NumLen` is the number of the fitted lines.)

```
NumLen=0
for i=1 to n-1
```

[2]http://www.sickoptic.com/kommerce_server/

```
    for j=i+1 to n
        $\alpha = \text{atan}(\frac{Dat(i).y - Dat(j).y}{Dat(i).x - Dat(j).x})$
        $\rho_0$ =(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 $n \in [1, \texttt{NumLen}]$, 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 5 shows an example of the proposed SPHT. The SPHT transforms or fits the raw data set to 2 lines. Actually, the raw data could be interpreted as 2 lines, or 2 lines plus a small transition arc, or even just one line. The exact segmentation depends on the resolution, or the threshold. For example, with a precise laser sensor, we know that the data in Fig. 5 cannot be a single line. However, with a low-precision sonar sensor, the target object might be just a single line. With a given sensor in real project, we should choose some reasonable thresholds by referring to the specifications of the sensors. Further discussions on the trade-off between sensor accuracy and the object fit accuracy can be found in [20] and [21].

C. Analysis on Speed and Memory Size

In general, suppose that the angle resolution is denoted by $\Delta\theta$ and the normal resolution by $\Delta\rho$. For SHT, $M \times N$ cells are required for the accumulator space, where $M = \pi/\Delta\theta$ and $N = \rho_{max}/\Delta\rho$. In other words, the memory size is $n \times M \times N$, where n is the size of each cell. If there are K valid points in the input data, the computation requirement is $\mathcal{O}(K \times M)$.

For SPHT, if there are only K valid points in the input, clearly, $K \ll M$ and $K \ll N$. The maximum number of fitted lines will be $K \times (K-1)/2$. The computation requirement is $\mathcal{O}(K \times (K-1)/2)$. The actual memory requirement depends on how many lines passing the data points in the input data. Normally, we expect that the number of lines, L, for our data is 2 to 4. It is easy to show that $L \leq K(K-1)/2$.

In [22], another scheme called Log-Hough transformation (LHT) was described. This algorithm is fast, with the complexity of $\mathcal{O}(n)$, and easy to implement. The authors introduced the concept of logarithmized cosine curve. For an arbitrary set of points, we can store the logarithmized cosine curve of a point of the set in a look-up table, then shift it by other parameters. The following is a comparison between the LHT and the SPHT:

- Memory: The exact memory requirement of SPHT depends on the L, the line number. However, there is an upper bound of $K(K-1)/2$ units. For LHT, we need k segments on the range of the accumulation space, where $k = \lceil \log(\frac{r_1 - r_0}{r_0})/\delta \rceil$. The area of interest stretches from r_0 to r_1. δ is the angle resolution of the laser system.
- Resolution: The distance resolution of LHT is not uniform. The resolution of the near field is better than that of the far field. But the distance resolution of SPHT is uniform with respect to distance, with the resolution level the same as the range resolution of the laser scanner, e.g., the example in [22] showed that if the range resolution of a laser scanner is 1cm, and the angle resolution is 0.5°, LHT will get a range resolution of 0.88cm for the objects at 50cm distance, and 16.6cm for the distance of 1000cm. For the same case, the distance resolution of SPHT will be always 1cm, no matter what the distance is.
- Complexity: The complexity of LHT is $\mathcal{O}(K)$ and that of SPHT is $\mathcal{O}(K(K-1)/2)$, where K is the number of valid points.

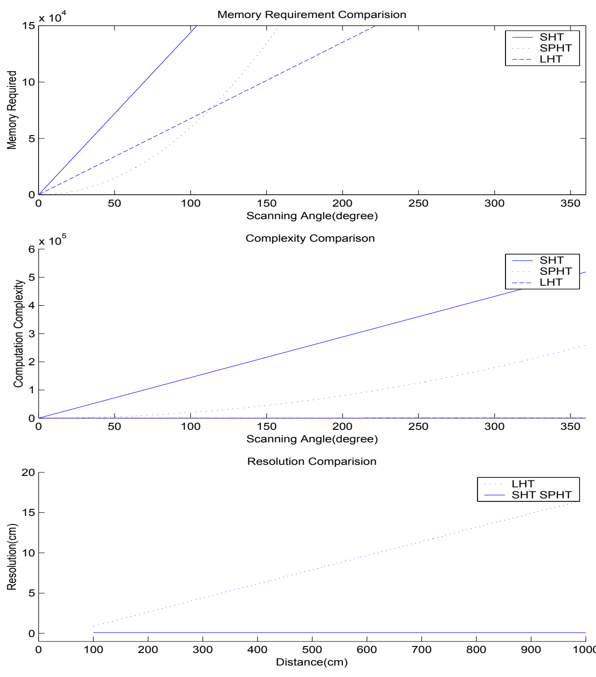

Fig. 4
The Comparison of Sparse, Log and Standard Hough Transform

Figure 4 shows the memory requirements, complexity and distance resolution of SPHT, LHT and SHT. Here we assume that the area of interest stretches from r_0 = 50cm to r_1 = 1000cm. The angular resolution δ is 0.5°. The distance resolution of the laser scanner is 1cm. Notice 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 SPHT has a good performance, as we can see from the figure. The upper figure shows that when the data set is very sparse, i.e., the angle of interest is narrow, SPHT demands the least memory. In the middle subplot of Fig. 4, we can estimate the rough speed of these algorithms based on their complexities. SHT is very slow, while SPHT is faster, and LHT is the fastest. In the bottom subplot of Fig. 4, we can see that the range resolutions of SHT and SPHT are exactly the same. The memory and speed benefits from LHT is at the expense of a lower and distance-dependent resolution.

D. Task-Dependent Segmentation

To process 2D laser data, segmentation is more complicated than in the case of 1D laser data. The idea of task-dependent segmentation is to simplify the situation by making a better use of the known information. Currently, we use lamp pole and side walk as landmarks to

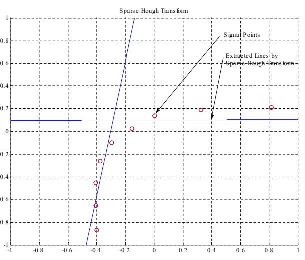

Fig. 5
An Example of Sparse Hough Transform

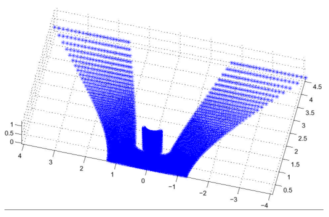

Fig. 6
Lamp Pole Simulation Data

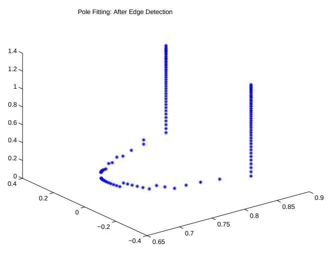

Fig. 9
Reverse The Edge of Lamp Pole Image

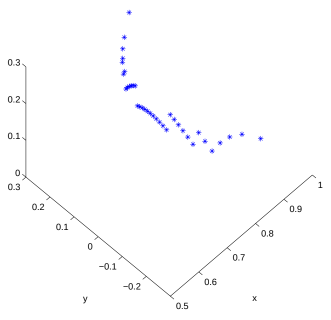

Fig. 10
Lamp Pole Data After Segmentation

calibrate the odometry system of the mobile robot.

Lamp pole is an important landmark in a parking lot. Figure 6 is the simulated data of a lamp pole from a 2D laser scanner mounted on T4, the mother robot vehicle for ODIS. Here we assume that the lamp pole is perpendicular to the ground and the ground is flat. The task is to segment the raw data so that the position of the center of the lamp pole (cylinder) can be computed by some fitting algorithms. Figure 7 is the image of the lamp pole that is projected to the ground. Here, our actual task is to separate the ground from the lamp pole. Figure 8 is the edge detection result of Fig. 7. Figure 10 is the 3D object when we project the edge in Fig. 8 back to the 3D space. At last, we set a threshold to eliminate those points too close to each other. The result is shown in Fig. 10.

After projecting these points to the ground, we can get a circle with the same center of the 3D lamp pole. In other words, when we feed the result of the segmentation to our fitting algorithms, we can get the position of the feature points. Then the relative position of the landmark will be available for mobile robot odometry calibration, or localization.

Another important landmark is the convex corner of the side walk. Since the height of side walk in a specific parking lot is constant, we can set a threshold to segment the ground and the upper layer. The segmentation is straightforward and we will not discuss it here due to space limitation.

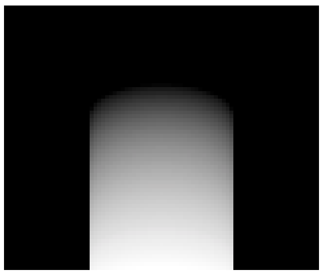

Fig. 7
Projected Lamp Pole Image

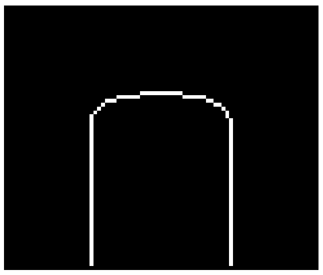

Fig. 8
The Edge Detection on Lamp Pole Image

III. Template Fitting Algorithms

With properly segmented data sets from a laser scan, it is not certain which template in the template library is to be used for the object fitting, without human involvement. A viable way is to try all templates in the library and then pick the "best" one via an arbitrator based on some rules from the possible prior knowledge about the objects in the static and uncertain environment.

For computational efficiency, we focus ourself on the algebraic fitting which does not require any iteration. For geometrical fitting, is may make more sense but the iterations may take a long time and some times, convergence cannot be guaranteed. It is of course, when possible, beneficial to use the geometrical fitting result to cross-validate that from the algebraic fitting.

Most of our codes for fitting are tested in Matlab and then converted to C++ [3] which are ready for running in our Linux box.

A. Algebraic Circle Fit

When the radius of a circle is unknown, a simple algebraic circle fitting method can be applied to find the best fit circle in the least squares (LS) sense.

Given a set of points with coordination $\{(x_i, y_i), i = 1, 2, \cdots, n\}$, find the best parameters $a_1, \cdots, a_4$ in the circle equation

$$a_1(x^2 + y^2) + a_2 x + a_3 y + a_4 = 0,$$

such that the algebraic error is to be minimized. The center of the circle is (x_c, y_c) with $x_c = -\frac{a_2}{2a_1}$, $y_c = -\frac{a_3}{2a_1}$.

Let $\mathbf{a} = [a_1, a_2, a_3, a_4]^T$. Construct the matrix $\mathbf{D}$ as

$$\mathbf{D} = \begin{bmatrix} x_1^2 + y_1^2 & x_1 & y_1 & 1 \\ x_2^2 + y_2^2 & x_2 & y_2 & 1 \\ x_3^2 + y_3^2 & x_3 & y_3 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ x_n^2 + y_n^2 & x_n & y_n & 1 \end{bmatrix}.$$

Then, the LS solution of equation

$$\mathbf{Da} = \mathbf{0}$$

is simply given by the SVD (singular value decomposition). via `[U,V,D]=SVD()`. Denote $\mathbf{V} = [\mathbf{v_1}, \mathbf{v_2}, \cdots, \mathbf{v_n}]$. Then, the solution is simply $\mathbf{a} = \mathbf{v_4}$. Figure 11 shows a typical circle fit to an experimental 1D laser scan data set.

[3]We used Dr. Robert Davies' free C++ library "`NewMat`" downloadable from `http://webnz.com/robert/cpp_lib.htm`.

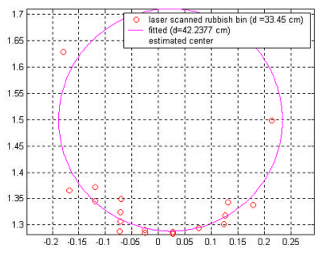

Fig. 11
CIRCLE FIT

B. Circle Fit with a Known Radius

Some of the round objects in the static uncertain environment may have known radii. So, although simpler, it is practically desirable to perform the circle fit with a known radius. Here we use the geometrical error for the fitness performance index.

Given a set of points with coordinates $\{(x_i, y_i), i = 1, 2, \cdots, n\}$, find the best parameters x_c and y_c, the center coordinate of the circle

$$(x - x_c)^2 + (y - y_c)^2 - R^2 = 0$$

where R is the known radius, such that the geometric error $\varepsilon = \sum_{i=1}^{n}(\sqrt{(x_i - x_c)^2 + (y_i - y_c)^2} - R)^2$ is minimized.

Via the gradient of ε with respect to x_c and y_c, an iterative scheme for x_c and y_c can be formulated and implemented. Since R is known, the convergence process is quite reliable and fast.

C. Ellipse Fit

We skip this part due to the space limit. The ellipse fitting codes are included in the `zip` file downloadable from the URL given in the first page of this paper. More technique details are available upon request.

D. Plane Fit

Different from the above fitting method mainly for 1D laser scan data set, the methods of this subsection and the next subsection are designed for 2D laser scanner data processing. Actually, the acquired data from a 2D laser scanner is a 3D array, as mentioned in Sec. II. The goal of the plane fitting problem is to find the optimal coefficients a_1, a_2, a_3, a_4, such that

$$\begin{bmatrix} x_1 & y_1 & z_1 & 1 \\ x_2 & y_2 & z_2 & 1 \\ x_3 & y_3 & z_3 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & z_n & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} = 0$$

has a solution in LS sense. Again, we can use the SVD technique for the plane fitting.

E. 3D Corner Fit/Detection

Corner in the context of 2D image processing is one of the basic image features and has been extensively studied in [23], [24]. For 3D corner detection, we can project the 3D corner to a 2D surface and detect the corner in 2D. This method, although simple, may not be able to detect some corners in certain 3D configurations. Here, we propose two effective 3D corner fit algorithms.

The first method fits an intersection point of 3 planes as the corner. So there is no limitation on whether the corner is convex or concave. The disadvantage of this algorithm is that it can not fit corner without a prior step for plane segmentation.

This corner fit problem is to find the best corner $\{x^*, y^*, z^*\}$, such that

$$\begin{bmatrix} A_1 & B_1 & C_1 \\ A_2 & B_2 & C_2 \\ A_3 & B_3 & C_3 \\ \vdots & \vdots & \vdots \\ A_n & B_n & C_n \end{bmatrix} \begin{bmatrix} x^* \\ y^* \\ z^* \end{bmatrix} + \begin{bmatrix} D_1 \\ D_2 \\ D_3 \\ \vdots \\ D_n \end{bmatrix} = \begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ \vdots \\ e_n \end{bmatrix}$$

or $MP = -D$, where $[x^* \quad y^* \quad z^*] = P^T$, $\sqrt{\sum_i e_i^2}$ is minimized. Based on the projection theorem, the solution is simply $P = (M^T M)^{-1} M^T(-D)$

The second algorithm can fit corners directly from the 3D point set. This approach eliminates the requirement on segmentation. We can separate points to small subsets, and then fit them by the following corner fit function. The limitation of the algorithm is that the corner function is only suitable for convex and symmetric corners. The proposed corner function is described by

$$z = e^{k_1(x-x_0)^2 + k_2(y-y_0)^2 + k_3}$$

where k_1, k_2, k_3, x_0, y_0 are parameters to be fit. In LS sense, we can re-formulate the above fitting problem into

$$\begin{bmatrix} x_1^2 & x_1 & y_1^2 & y_1 & 1 \\ x_2^2 & x_2 & y_2^2 & y_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n^2 & x_n & y_n^2 & y_n & 1 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \end{bmatrix} = \begin{bmatrix} \log(z_1) \\ \log(z_2) \\ \vdots \\ \log(z_n) \end{bmatrix}.$$

The rest of the algorithm is the same as in the previous 3D corner fitting method.

IV. Arbitrator

If the identity of the object is not clearly known, arbitration is necessary to find the suitable template for the fitting algorithm. The basic idea is that we can try to fit the unknown shape to all suspected models, and then find the best fit with respect to the geometry error.

Since we used algebraic fitting, the geometrical error have to be re-calculated. Here, we use the circle fit as an example.

For a circle $a_1(x^2+y^2)+a_2x+a_3y+a_4 = 0$, the geometry error E is

$$E = \sum_{i=1}^{n} \left| \sqrt{(x_i + \frac{a_2}{2a_1})^2 + (y_i + \frac{a_3}{2a_1})^2} - R \right|,$$

where $R = \sqrt{\frac{a_2^2}{4a_1^2} + \frac{a_3^2}{4a_1^2} - \frac{a_4}{a_1}}$.

The real objects, e.g., lamp pole and curb, are too complex for computers to identify directly. We follow the idea of "*segmentation–fitting*" to represent complex objects in basic geometry shapes, i.e., circle, ellipse, line, plane, corner. The geometry errors are the index of the "likelihood"

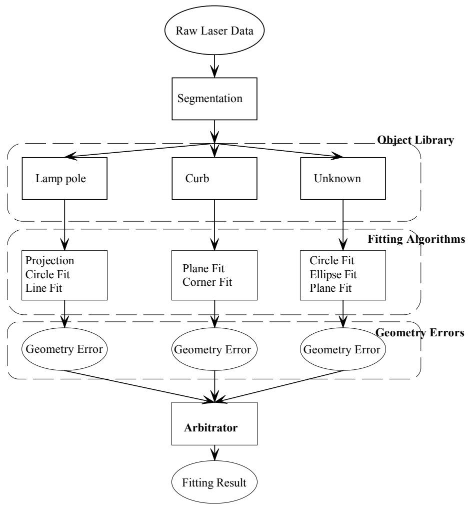

Fig. 12
The Flow Chart of Arbitration and Object Decision

for those fittings. The arbitrator can be regarded as a processor to reduce the raw data to simple, abstract and higher level information for the decision maker in the robot's higher-level control. Figure 12 is the flow chart of our arbitrator. Since our robot works in the structured unknown environment, we can put all possible objects in the objects library, and add more if the environment changed. This extendable feature is important in real intelligent robot systems.

V. Concluding Remarks

This paper presents some techniques for sensing and perception for an omnidirectional ground autonomous vehicle equipped with a laser scanner. In an assumed structured environment (static but uncertain), the sensing data processing methods for both 1D and 2D laser scanner are discussed. Raw data are segmented to lines, circles, ellipse, planes and corners by task dependent segmentation algorithms. Each subset of data is then fit by a known template shape as listed above.

Our immediate effort is to make use of these medium level information in our vehicle to infer its relative position with respect to the known landmarks and in turn help to determine its absolute position on the map, a procedure known as mobile robot localization. Other future efforts include (1) the motion estimation of dynamic obstacle(s) by assuming that the environment is dynamic uncertain; (2) the fusion with sonar, laser scanner and image sensing information for local map building and (3) relative navigation without absolution position information (or inertial/world coordinates) via sensor fusion and spatial filtering.

Acknowledgment

The authors would like to acknowledge the fruitful discussions with CSOIS members in Vetronics Group and Intelligent Behavior Group. In particular, the authors are grateful to Professor Kevin L. Moore, Director of CSOIS, for his support of this work.

References

[1] M. D. Adams, *Sensor modelling, design and data processing for Autonomous navigation*, vol. 13 of *World Scientific in Robotics and Intelligent Systems*, World Scientific, Singapore, 1999.

[2] I. R. Nourbakhsh, *Interleaving planning and execution for autonomous robots*, The Kluwer International Series in Engineering and Computer Science: Robotics: Vision, Manipulation and Sensors. Kluwer Academic Publishers, Boston, 1997.

[3] K. L. Moore and N. S. Flann, "A six-wheeled omnidirectional autonomous mobile robot," *IEEE Control Systems*, vol. 20, no. 6, pp. 53–66, Dec. 2000.

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

[5] M. Davidson and Bahl V., "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.

[6] D. Dedieu, V. Cadenet, and P. Souères, "Mixed camera-laser based control for mobile robot navigation," in *Proc. of the 2000 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2000.

[7] H. Surmann, K. Lingemann, A. Nuchter, and J. Hertzberg, "A 3D laser range finder for autonomous mobile robots," in *Proc. of the 32nd ISR2001 (Int. Symp. on Robotics), Seoul, Korea*, 2001, pp. 153–158.

[8] R. M. Taylor and P. J. Probert, "Range finding and feature extraction by segmentation of images for mobile robot navigation," in *Proc. for the 1996 IEEE Int. Conf. on Robotices and Automation Minneapolis, Minnesota*, Apr. 1996.

[9] J. Vandorpe, H. V. Brussel, and H. Xu, "Exact dynamic map building for a mobile robot using geometrical primitives produced by a 2D range finder," in *Proc. for the 1996 IEEE Int. Conf. on Robotices and Automation Minneapolis, Minnesota*, Apr. 1996.

[10] V. Cadenat, P. Souéres, and M. Courdesses, "Two milti-sensor-based control strategies for driving a robot amidst obstacles," in *Proc. for the 39th IEEE Conf. on Decision and Control Sydney, Australia*, Dec. 2000.

[11] J. Laurent, M. Talbot, and M. Coucet, "Road surface inspection using laser scanners adapted for the high precision 3D measurements of large flat surfaces," in *Proc. of the Int. Conf. on Recent Advances in 3-D Digital Imaging and Modeling IEEE*, 1997.

[12] J. Borenstein, H. R. Everett, and L. Feng, "Where am I, sensors and methods for mobile robot positioning," `http://www-personal.engin.umich.edu/~johannb/position.htm`, 1996.

[13] K. Abed-Meraim and Y. Hua, "Multi-line fitting and straight edge detection using polynomial phase signals," IEEE-IP, Dec. 1997.

[14] J. Illingworth and J. Kittler, "A survey of the Hough transform," *CVGIP*, vol. 44, pp. 87–116, 1998.

[15] J. Puzicha and S. Belongie, "Model-based halftoning for color image segmentation," in *Proc. of the Int. Conf. on Pattern Recognition*, Barcelona, Spain, 2000, pp. 629–32.

[16] T. J. Atherton and D. J. Kerbyson, "Size invariant circle detection," *Image and Vision Compution*, vol. 17, pp. 795–803, 1999.

[17] N. Guil and E. L.Zapata, "Lower order circle and ellipse Hough transform," *J. Pattern Recognition*, vol. 30, no. 10, pp. 1729–1744, Oct. 1997.

[18] N. Guil and E. L. Zapata, "A new invariant scheme for the generalized Hough transform," in *IASTED Int. Conf. on Signal and Image Processing*, Orlando, Fl,, Nov. 1996, pp. 88–91.

[19] J. Matas, C. Galambos, and J. Kittler, "Progressive probabilistic Hough Transform," in *Proc. British Machine Vision Conf. BMVC98*, Sep. 1998.

[20] H. Goto, "The efficient sampling interval of the scanning parameter in the Hough transform," *Systems and Computers in Japan*, vol. 29, no. 11, pp. 697–705, Apr. 1998.

[21] H. Goto and H. Aso, "Designing efficient Hough transform by noise-level shaping," *IEICE Trans. Inf. and Syst.*, vol. E83-D, no. 2, Feb. 2000.

[22] B. Giesler, R. Graf, R. Dillmann, and C.F.R. Weiman, "Fast mapping using the Log-Hough transformation," *IEEE/RSJ,Int. Conf. on Intelligent Robots and Systems*, vol. 3, pp. 1702–1707, 1998.

[23] M. A. Ruzon and C. Tomasi, "Corner detection in textured color images," in *Proc. of the IEEE Seventh Int. Conf. on Computer Vision*, Sep. 1999, pp. 1039–1045.

[24] S. M. Smith and J. M. Brady, "SUSAN - a new approach to low-level image processing,," in *Int. Jo. of Computer Vision*, 23(1): 1997, pp. 45–78.