Fast line, arc/circle and leg detection from laser scan data in a Player driver*

João Xavier, Marco Pacheco, Daniel Castro and António Ruano
Centre for Intelligent Systems - CSI, University of Algarve
Campus de Gambelas, 8000-117 Faro, Portugal
smogzer@yahoo.com, marcopacheco@zmail.pt, {dcastro,aruano}@ualg.pt

***Abstract*— A feature detection system has been developed for real-time identification of lines, circles and legs from laser data. A new method suitable for arc/circle detection is proposed: the Internal Angle Variance (IAV). Lines are detected using a recursive line fitting method. The people leg detection is based on geometrical constrains. The system was implemented as a fiducial driver in Player, a mobile robot server. Real results are presented to verify the effectiveness of the proposed algorithms in indoor environment with moving objects.**

***Index Terms*— Arc/circle detection, leg detection, laser feature detection, Player, mobile robot navigation**

I. Introduction

For *Robotic Freedom* [1] become a reality, robots must be able to recognize objects, structures and persons, so they can perform tasks in a safe and proficient way. Stock replacement, street cleaning and zone security are some examples of tasks not adequate for persons, they could be delegated to machines so that persons live in more hedonic societies. Although Laser Measurement Systems (LMS) do not provide sufficient perception to accomplish this tasks on their own, they can be a great help, specially for navigation tasks where we require fast processing times not achievable with camera based solutions.

Current navigation systems benefit from detecting indoor (columns, corners, trashcans, doors, persons, etc.) and outdoor (car parking poles, walls, cars, etc.) structures for intelligent and safe navigation while performing tasks involving those structures. Many structures are decomposable in shapes of geometric primitives, that can be used for scene interpretation. This geometric perception is important to make spatial inferences, this is a building block of the Artificial Intelligence required to accomplish complex tasks.

Our choice of primitive feature to detect are lines, arcs/circles and legs. Lines and arcs are mostly used for navigation, scene interpretation or map building. Leg detection applications range from following persons, to excluding leg segments from Simultaneous Localization And Mapping (SLAM) scan matching in order to reduce noise. With fast feature extraction techniques, there is processing time left for other tasks, like scan matching, navigation, scene interpretation, etc.

*This work is partially supported by FCT Grant BD/1104/2000 to D. Castro

This work proposes a new technique for circle detection that can be also used in line detection, the Internal Angle Variance (IAV). This technique presents a linear complexity $O(n)$ where the average and the standard deviation are the heaviest calculations, and are also the only parameters the user needs to specify. Compared to other methods like the Hough transform for circle detection [2] that have parameters to tune like the accumulator quantification, ours is simpler to implement, with lower computational costs, proving to be an excellent method for real time applications. A recursive line detection method similar to [3] and [4] was also implemented.

The robotics research community should have algorithms ready to deploy and compare, in a expansible, unconstrained framework. With this idea in mind we developed our algorithms in a standard open robotics platform, Player [5], a network server for robot control. In 2003 the Robotics Engineering Task Force identified Player as the *de facto* standard open robotics protocol. We also want to encourage the academic community to join this design/development philosophy.

A. Background and related work

The detection of objects can be done by feature based approaches (using laser scanners) [6]–[8]. As discussed in [9] feature to feature matching is known to have the shortest run-times of all scan matching techniques.

Next we introduce the more relevant research in the detection of the features we work with.

1) Circle detection: In [10] line detection was executed using the sparse hough transform and circle detection through algebraic circle fitting by least squares. In [11] were tested two methods for circle detection, one on-line and other offline. On-line circle detection using the unscented Kalman filter, and offline using Gaussian-Newton optimization of the circle parameters. Extracting tree trunks was also attempted. None of these approaches solves the problem in an elegant manner making use of geometric properties of arcs that we suggest in this paper.

2) Leg detection: In [9] is shown that SLAM accuracy is affected by moving segments, this system would benefit with identification of leg-like segments for removal from scan matching.

Two main approaches for leg detection are known. The most common approach is by scan matching [6], which can only recognize moving persons. The second approach takes advantage of geometric shape of legs [12], similar to arcs as seen by the laser. Our proposed leg detection is based on the geometric shape approach.

3) Line detection: Different approaches for extracting line models from range data have been presented. The most popular are clustering algorithms [2], [13] and split-and-merge algorithm [3], [4]. The Hough transform is usually used to cluster collinear points. Split-and-merge algorithms, on the other hand, recursively subdivides the scan into sets of neighbor points that can be approximated by lines.

B. System Overview

Our system is composed by a Laser Measurement System (LMS), mounted on a Activemedia Pioneer 2-DX mobile robot, that also supports a laptop equipped with Celeron 500MHz processor with 128 Mb of memory. The connection from the laser to the computer is made with a serial-to-usb adapter. The experiments are done with the LMS doing planar range scans with angular resolutions of 180° operating at frequencies of about 60 Hz with a empirical error of 0.017 meters. The total (laser and robot) scan height is 0.028 meters, this corresponds to about $\frac{2}{3}$ of the knee height of a normal adult.

The algorithms were implemented in C++ as a `Fiducial` driver of Player, a GPL [14] robot device server that supports our Pioneer 2-DX and SICK LMS200 [15] hardware. The feature visualization tool runs on the client side and was developed in OpenGL.

C. Paper structure

This paper is organized as follows. Section II presents the algorithms used to perform the extraction of features. Section III describes the encapsulation and software architecture used by player and the developed drivers. In section IV real experiments are performed in an indoor environment. Final remarks are given in the section V.

II. Feature Detection

This layer is composed by two procedures: the range segmentation and the feature extraction. First the scan data segmentation creates clusters of neighbor points. Later these segments are forwarded to the feature extraction procedure, where the following features are considered: circles, lines and people legs.

A. Range Segmentation

Is the clustering of consecutive scan points, which due to their proximity probably belong to the same object. The segmentation criterion is based on the distance between two consecutive points P_i and P_{i+1}. Points belong to the same segment as long as the distance between them is less than a given threshold. Isolated scan points are rejected. Because our laser readings have a maximum statistical error of 0.017 meters [16] we don't have any error compensation scheme.

B. Circle detection

Circle detection uses a new technique we called the Internal Angle Variance (IAV). This technique makes use of trigonometric properties of arcs: every point in an arc has congruent angles (angles that have the same number of degrees) in respect to the extremes.

An example (Fig.1) of this property : Let P_1,P_4 be distinct points on a circle. Let P_2,P_3 be points on the circle lying on the same arc between P_1 and P_4. Then

$$m(\angle P_1 P_2 P_4) = m(\angle P_1 P_3 P_4) \quad (1)$$

This is because both angles measure one-half of $m(\angle P_1 O P_4)$ The detection of circles involves calculating the mean and standard deviation. Positive detection occurs with standard deviation values below 0.15 radians and values of mean aperture between $90°$ and $135°$. These values that were tuned empirically to detect the maximum number of circles, while avoiding false positives. The confidence of the analysis increases with three factors:

- the number of in-between points (more than 2)
- the apertures of the in-between points must have a low standard deviation
- the average aperture angle near the value $\frac{\pi}{2}$ radians, this means that more of the circle is visible

There are some particular cases and expansions that we will detail next:

1) Detecting lines: The line detection procedure using IAV uses the same same procedure of circles but the mean is $180°$ as can be seen in Fig. 2

2) Detecting arcs embedded in lines: Identifying round corners of a wall is possible after standard line detection. This happens because line detection excludes scan points previously identified as belonging to lines from circle detection. For this technique to work we cannot allow small polygons to be formed, this implies line detection with a minimum peak error and distance between endpoints. The idea is summarized in Fig. 3.

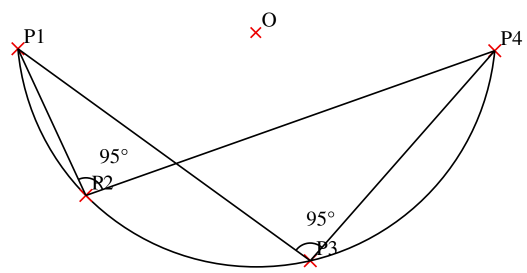

Fig. 1. The angles of points inside an arc are congruent in respect to the extremes

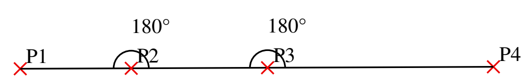

Fig. 2. The angles inside a line are congruent in respect to the extremes and measure 180 degrees

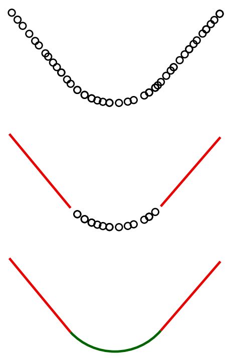

Fig. 3. Detection of embedded arcs

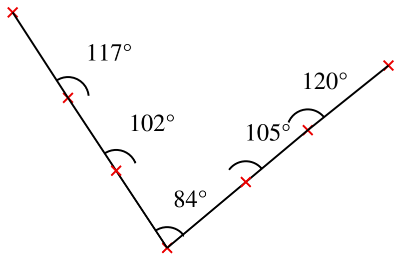

Fig. 4. The apertures of a "V" shaped corner grow around a local minimum

3) Detecting "arc-like" shapes: Suppose that we want to detect a tree trunk, or a object that resembles a cylinder, this is possible by increasing the threshold of the standard deviation. Experimental tests demonstrated that not so regular arcs can be found between 0.15 and 0.40 radians of standard deviation.

When this acceptance threshold is increased false positives can be found. A further analysis is then required to isolate the false positives, an example of this is the "V" shaped corner in Fig.4. This case can be isolated by finding the point with the smallest aperture (middle in this case) and verifying if four neighbor points have progressively bigger apertures, if this happens we have a negative detection. With the certain we have detected a circle all that we need now is to estimate it's center and radius. From analytic geometry, we know that there is a unique circle that passes through three points. For these points we will use the extremes of the segment P_1 and P_3 and the point with the aperture value most close to the average P_2. Two lines can be formed through 2 pairs of the three points, the first passes through the first two points P_1 and P_2, the second line through the next two points P_2 and P_3. The equation of these two lines is

$$y_a = m_a(x - x_1) + y_1 \tag{2}$$

$$y_b = m_b(x - x_2) + y_2 \tag{3}$$

Where m is the slope of the line given by

$$m_a = \frac{y_2 - y_1}{x_2 - x_1} \tag{4}$$

$$m_b = \frac{y_3 - y_2}{x_3 - x_2} \tag{5}$$

The center of the circle is the intersection of the two lines perpendicular to and passing through the midpoints of the lines P_1 P_2 and P_2 P_3. The perpendicular of a line with slope m has slope $\frac{-1}{m}$, thus equations of the lines perpendicular to lines a and b and passing through the midpoints of P_1 P_2 and P_2 P_3 are

$$y'_a = -\frac{1}{m_a}(x - \frac{x_1 + x_2}{2}) + \frac{y_1 + y_2}{2} \tag{6}$$

$$y'_b = -\frac{1}{m_b}(x - \frac{x_2 + x_3}{2}) + \frac{y_2 + y_3}{2} \tag{7}$$

These two lines intersect at the center, solving for x gives

$$x = \frac{m_a m_b(y_1 - y_3) + m_b(x_1 + x_2) - m_a(x_2 + x_3)}{2(m_b - m_a)} \tag{8}$$

The y value of the center is calculated by substituting the x value into one of the equations of the perpendiculars. The value of the radius is the distance from point P_2 to the center of the circle.

The data we keep from circles is the first and last indexes of the laser scan where the circle was detected, the Cartesian coordinates of the center, the radius, and the standard deviation.

C. Circle prerequisites

To avoid analyzing all segments for circles, each segment must validate one geometric condition, the middle point must be inside a area delimited by two lines parallel to the line segment composed by the extremes of the segment, as can be seen in Fig. 5. To make this validation simpler, it is preceded by a rotation around the x axis. The full operation is explained by the following equations.

$$angle = \arctan \frac{left_x - right_x}{left_y - right_y} \tag{9}$$

$$point = x\cos(angle) - y\sin(angle) \tag{10}$$

$$diameter = d(leftmost, rightmost) \tag{11}$$

$$-0.7 \times diameter < point < -0.1 \times diameter \tag{12}$$

Where $angle$ is the angle to rotate (robot facing x axis), $left$ and $right$ are the Cartesian coordinates of the left and right extremes of the segment and d is a Cartesian distance function and $diameter$ is the distance between the leftmost and rightmost points.

Note that the delimiting zone is adjusted to include circles of small sizes where the SNR of the laser is small. When detecting big circles it is possible to shorten this limit for better filtering of the segments to analyze.

D. Leg detection

The procedure for detecting legs is an extension of the circle prerequisites II-C, with the extra constrains of the distance between end-points falling within the range of expected leg diameters ($0.1m$ and $0.25m$) and the segment not being partially occluded. Sometimes legs get classified as circles also. The ambiguity should be solved by higher order layers.

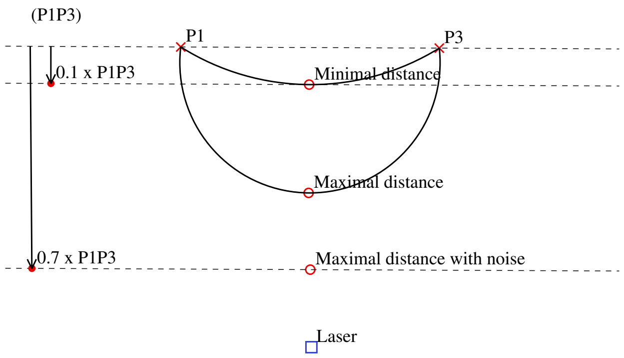

Fig. 5. Condition that selects segments for circle analysis

The data we keep from leg detection are the first and last indexes of the laser scan where the leg was detected, and also the Cartesian coordinates of the middle point of the segment.

E. Line detection

The line detection procedure uses the Recursive Line Fitting algorithm. This algorithm will try to fit lines to the segment being analyzed.

1) Obtain the line passing by the two extreme points
2) Obtain the point most distant to the line
3) If the fitting error is above a given error threshold, split (where the greatest error occurred) and repeat with the left and right sub-scan

The fitting function is the Orthogonal Regression of a Line [17]. This approach tries to find the "principle directions" for the set of points. An example of the process can be seen in Fig. 6.

The recursively process break conditions are:

1) few points to analyze
2) distance between extremes is outside the specified in configuration
3) fitting error under threshold

To avoid creation of small polygons three rules were stated:

1) more than 5 points
2) distance between extremes greater than 0.1 meters
3) maximum fitting error 0.02 meters

For each line we keep the first and last indexes of the laser scan where the line was detected, the slope, bias, number of points analyzed and the error returned.

III. Encapsulation in Player

The proposed algorithms were implemented as a Player [5] driver. This software offers the following possibilities:

- Hardware support
- Saving log files
- Client / Server architecture
- Open Source
- Modularity
- Stage and Gazebo, two simulators distributed in the same site [5]

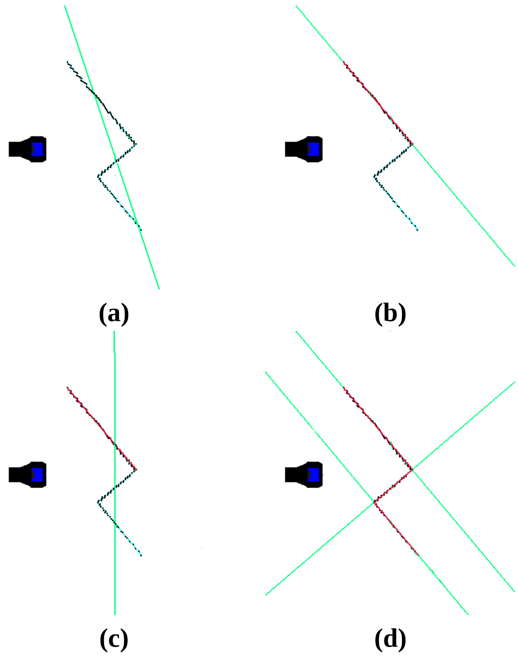

Fig. 6. The process of Recursive Line Fitting: a) principle directions, b) first line segment identified, c) principle direction of second segment, d) both lines detected

TABLE I
Format of fiducial data fields

Field #	Feature Type Line	Circle	Leg
1	laser index of first		
2	laser index of last		
3	slope	$center_x$	$middle_x$
4	bias	$center_y$	$middle_y$
5	# points	radius	—
6	error	std	—

The feature detection code runs on the server (where the hardware is located) but can also run on simulations either using Stage or Gazebo.

Player defines data structures and procedures for communicating between clients and a server. Of the available data structures we choose the `fiducial` interface for communicating with our client. This interface was originally conceived to provide access for devices that detect coded fiducials (markers) in the environment. It communicates to the client a list of detected fiducials with one `id`, one `type` and 6 arbitrary data fields. We altered the `fiducial` proxy on the client side, so that no default conversions were applied to the units on the data fields, allowing the fields to act like a list of 6 integers.

The `id` field is used to define a unique key for the detected feature, the `type` field as the name suggests is an integer representing the detected geometrical primitive, the other 6 fields further characterize the feature. See table I for detailed information.

When the driver class is instantiated it reads parameters from a file. If some parameters are not defined in the file, they are filled with default parameters specified in the program source code. It's also possible to change/read

TABLE II

AVAILABLE CONFIGURATION PARAMETERS

Name	Default	Meaning
`laser`	0	Index of the `laser` device to be used
`seg_threshold`	80	Segmentation threshold (mm)
`circles`	1	Algorithm for circles (0 disable)
`lines`	1	Algorithm for lines (0 disable)
`legs`	1	Algorithm for legs (0 disable)
`circle_std_max`	0.15	Maximum standard deviation allowed
`circle_max_len`	99999	Analyze segment for circles if length(mm) is inferior
`circle_min_len`	150	Analyze segment for circles if length(mm) is superior
`circle_min_pts`	4	Minimum number of points for circle analysis
`line_min_len`	150	Analyze segment for lines if length (mm) is superior
`line_min_pts`	5	Minimum number of points for line analysis

configuration on the fly by sending a `configuration` request.

Four configuration request are possible: parameters change, parameters query, position of the laser and field of view changes.

A. Visualization tool

To test the accuracy our system a visualization tool named PlayerGL was developed in OpenGL [18]. PlayerGL has the following capabilities:

- Two type of camera views (plant and projection) with full freedom of movement
- On the fly configuration of the `laser` and `fiducial` devices
- Two grid types, Polar and Cartesian
- Screen logging of detected features

PlayerGL is a invaluable tool for testing, comparing and tuning the methods presented.

IV. EXPERIMENTAL RESULTS AND DISCUSSION

A. Tests performed

To illustrate the accuracy of our methods two tests are shown, one for line detection and other for arc/circle and leg detection with some scene interpretation. Our robot is the black figure with a blue square (representing the laser). The Euclidean grid unit is $meters$.

1) Line detection tests: With the objective of showing the possibilities of SLAM using line detection, we run this test in a corridor of University of Algarve, with some students appearing in the laser scan.

The snapshop in Fig.7 shows lines perfectly detected in red. Leg detection is represented in green to demonstrate the possibilities of tracking of persons and exclusion of leg segments from SLAM trackers for noise removal.

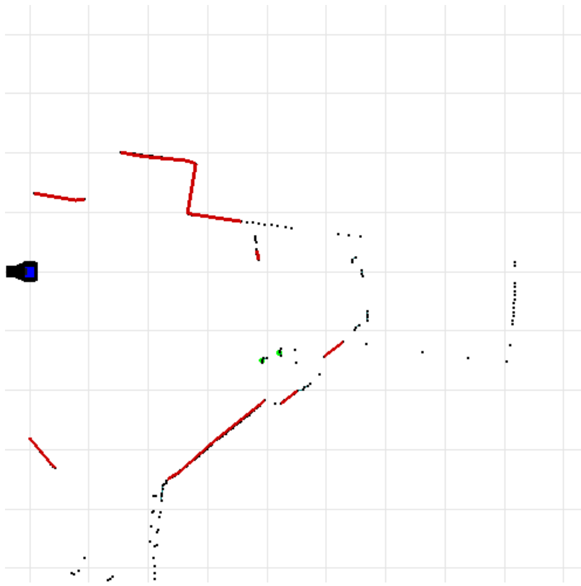

Fig. 7. Lines detected (color red) in a real test scenario

2) Circle and leg detection tests: This test occurred in a hallway entrance, with some architectonic entities: two benches, two columns, walls and corners. Benches are formed by two cylinders with radius equal to the columns radius.

In this test a student pulls a perfectly round cylinder of radius equal to the column, mounted on a wheeled platform around two columns. Without being planned a student lady (wearing jeans) passed by the low right corner.

In this scenario benches are a technical challenge to recognize; we defined Benches as two cylinders (with radius equal to the columns radius) separated by a distance of 1.3 meters from their centers.

One challenge is to detect correctly the mobile cylinder in all the course; right now lots of false positive of benches happen when the mobile cylinder is at a distance from the other cylinders equal to the bench distance. To overcome this misdetection we plan to introduce *id anchors* with trackers, in a higher processing layer, not described in this paper.

The results in Fig. 9 show that the circle detection (green circumferences) was accurate, and no objects in the background were misdetected as circles. Scene interpretation is done when possible so that the architectonic entities (walls, corners, columns, benches) present are also visualized.

Detected legs are represented as the tiny purple circles. It is possible to see the tracks of the student pulling the mobile cylinder as well as the girl passing in the corner. If the environment is not crowded those tracks should suffice for following a person. Note that there are no detection of legs in the background, mostly in the cylinders where some false positive legs could happen. That was possible due to the occlusion check that only recognizes legs if they are not partially occluded. Sometimes legs were classified as circles.

The number of leg detections decreases as the range increases. This happens because the algorithm operates with at least 3 points. To compensate this loss of detail increasing the resolution to $0.25°$ works well, detecting legs even at 8 meters.

Fig. 8. Panoramic photograph of test scenario for circle and leg detection.

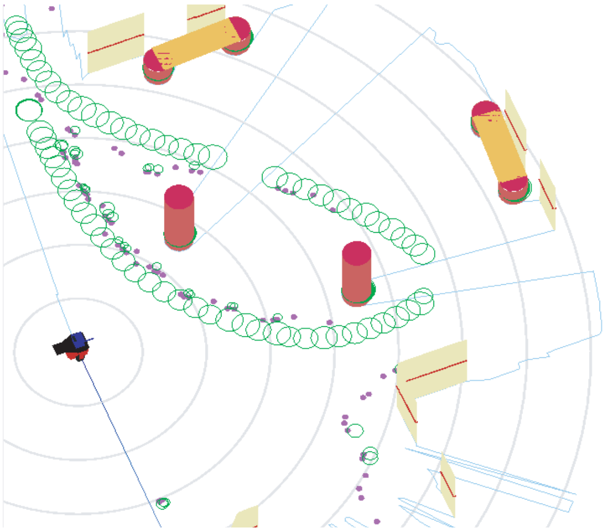

Fig. 9. Screen capture of PlayerGL, at the end of our feature detection test. Circle detection of the mobile cylinder are the green circumferences, legs are purple circles. Interpretation of walls, columns and benches is also achieved.

V. Conclusion and Future Work

We propose algorithms for feature detection that are fast and accurate. The Internal Angle Variance is a novel technique to extract features from laser data. It is accurate in circle detection and very helpful detecting other features.

The Player feature detection driver provides a abstraction layer for top layers like scene interpretation, navigation, cooperative robot geometric map building, etc. Distribution in the Player robot server ensures that this algorithms will be further tested and improved.

In the sequence of this work the driver will be expanded to support more fast laser feature detection algorithms, mostly lines and polygons and also benchmark statistics. Our visualization tool PlayerGL will interpret sets of primitives for real-time 3D interpretation of trashcans, doors, chairs, etc. When that is finished we will transpose that interpretation heuristics to another Player fiducial driver, the final goal being the reconstruction from laser scan data of a dynamic scene, with moving persons, and easily differentiable structures.

References

[1] Robotic Freedom by Marshall Brain. http://marshallbrain.com/robotic-freedom.htm

[2] D. Castro, U. Nunes, and A. Ruano, "Feature extraction for moving objects tracking system in indoor environments," in Proc. 5th IFAC/euron Symposium on Intelligent Autonomous Vehicles, Lisboa, July 5-7, 2004.

[3] A. Mendes, L. Conde and U. Nunes, "Multi-target detection and tracking with a laserscanner," in *Proc. IEEE Intelligent Vehicles Symposium*, Parma, June 2004.

[4] T. Einsele, "Localization in indoor environments using a panoramic laser range finder," Ph.D. dissertation, Technical University of Mĺunchen, September 2001.

[5] Gerkey B.P., R.T. Vaughan and A. Howard (2003). The Player/Stage Project. http://playerstage.sf.net .

[6] D. Schulz, W. Burgard, D.Fox and A.Cremers, "Tracking multiple moving targets with a mobile robot using particle filters and statistical data association," in Proc. IEEE Int. Conf. on Robotics and Automation, pp.1665-1670, Seul 2001.

[7] B. Kluge, C. Köhler and E. Prassler, "Fast and robust tracking of multiple moving objects with a laser range finder," in Proc. IEEE Int. Conf. on Robotics and Automation, pp.1683-1688, Seoul, 2001.

[8] A. Fod, A. Howard and M. J. Mataric, "A laser-based people tracker," in Proc. IEEE Int. Conf on Robotics and Automation, pp. 3024-3029, Washington D.C., May 2002.

[9] C-C Wang and C. Thorpe, "Simultaneous localization and mapping with detection and tracking of moving objects," in Proc. IEEE Int. Conf. on Robotics and Automation, pp.2918-2924, Washington D.C., 2002.

[10] Zhen Song, YangQuan Chen, Lili Ma, and You Chung Chung, "Some Sensing and Perception Techniques for an Omni-directional Ground Vehicles with a Laser Scanner," *IEEE International Symposium on Intelligent Control (IEEE ISIC)*, Vancouver, British Columbia, October 27-30, 2002, pp. 690-695.

[11] Sen Zhang, Lihua Xie, Martin Adams and Tang Fan, "Geometrical feature Extraction Using 2D Range Scanners" *International Conference on Control & Automation*, Montreal, Canada, June 2003.

[12] Fuerstenberg, Kay Ch.; Linzmeier, Dirk T.; Dietmayer, Klaus C.J.: Pedestrian Recognition and Tracking of Vehicles using a vehicle based Multilayer Laserscanner. *Proceedings of ITS 2003, 10th World Congress on Intelligent Transport Systems*, November 2003, Madrid, Spain.

[13] K. Arras and R. Siegwart, "Feature-extraction and scene interpretation for map-based navigation and map-building, " in Proc. of SPIE, Mobile Robotics XII, vol. 3210, 1997.

[14] Free Software Foundation. *GNU General Public License*, June 1991. Version 2.

[15] SICK, Proximity laser scanner, SICK AG, Industrial Safety Systems, Germany, Technical Description, 05 2002.

[16] Ye, C. and Borenstein, J., 2002, "Characterization of a 2-D Laser Scanner for Mobile Robot Obstacle Negotiation," *Proceedings of the 2002 IEEE International Conference on Robotics and Automation*, Washington DC, USA, 11 - 15 May 2002, pp. 2512-2518

[17] mathpages, Perpendicular regression of a line, http://mathpages.com/home/kmath110.htm.

[18] OpenGL http://www.opengl.org.