Fast Line, Arc/Circle and Leg Detection from Laser Scan Data in a Player Driver

João Xavier*, Marco Pacheco†, Daniel Castro†, António Ruano† and Urbano Nunes*
* *Institute of Systems and Robotics - ISR, University of Coimbra, Portugal*
† *Centre for Intelligent Systems - CSI, University of Algarve, Portugal*
smogzer@gmail.com, marcopacheco@zmail.pt, {dcastro,aruano}@ualg.pt, urbano@isr.uc.pt

***Abstract*— A feature detection system has been developed for real-time identification of lines, circles and people legs from laser range data. A new method suitable for arc/circle detection is proposed: the Inscribed Angle Variance (IAV). Lines are detected using a recursive line fitting method. The people leg detection is based on geometrical relations. The system was implemented as a plugin 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 the *Robotic Freedom* [1] manifesto to become possible, 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 these 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, cars, etc.) structures.This geometric perception is important to make spatial inferences from which Scene Interpretation is achieved.

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 improving Simultaneous Localization and Mapping (SLAM) accuracy by removing probable legs from scan matching.

This work proposes a new technique for circle detection that can be also used in line detection, the Inscribed Angle Variance (IAV). This technique presents a linear complexity $O(n)$ where the average and the standard deviation (`std`) are the heaviest calculations. Compared to other methods like the Hough transform for circle detection [2], [3] that have parameters to tune like the accumulator quantification, ours is simpler to implement, with lower computational cost, proving to be an excellent method for real time applications. A recursive line fitting method with complexity $O(n \cdot \log(n))$, similar to [5] and [6] was also implemented.

The robotics research community can greatly benefit from the availability of algorithms ready to deploy and compare, in a modular framework. With this idea in mind we developed our algorithms in a standard open robotics platform, Player [7], a network server for robot control. In 2003 the Robotics Engineering Task Force identified Player as the *de facto* standard open robotics protocol.

A. Background and Related Work

The detection of objects can be done by feature based approaches (using laser scanners) [8]–[10]. As discussed in [11] feature to feature matching is known to have the shortest run-times of all scan matching techniques.

Next we introduce some relevant research addressing the detection of the features we work with.

1) Circle Detection: In [12] circle detection was performed using least squares algebraic circle fitting. In [13] were tested two methods for circle detection: an on-line method using the unscented Kalman filter; and an off-line method using Gaussian-Newton optimization of the circle parameters. Extracting tree trunks was also attempted. None of these approaches solves the problem using the geometric properties of arcs, as we suggest in this paper.

2) Leg detection: In [11] is shown that SLAM accuracy is affected by moving segments, therefore leg-like segments identification and its removal from scan matching would benefit SLAM tasks.

The most common approach for leg detection is scan matching [8], which can only recognize moving persons with the restriction that the only moving segments on the environment correspond to persons. Another common approach takes advantage of geometric shape of legs [14], similar to arcs as seen by the laser. This approach doesn't track background changes but requires the legs to be well defined. Our proposed leg detection is based on the latter approach.

3) Line detection: Different approaches for extracting line models from range data have been presented. The most popular are clustering algorithms [3], [15] and split-and-merge algorithm [4]–[6]. The Hough transform is usually used to cluster collinear points. Split-and-merge algorithms, on the other hand, recursively subdivide the scan into sets of neighbor points that can be approximated by lines.

This work is partially supported by FCT Grant BD/1104/2000 to Daniel Castro and FCT Grant POSI/SRI/41618/2001 to João Xavier and Urbano Nunes

B. System Overview

Our hardware is composed by a Sick Laser Measurement System (LMS), mounted on a Activemedia Pioneer 2-DX mobile robot that also supports a laptop. The connection from the laser to the computer is made with a rs422-to-usb adapter. The experiments are done with the LMS doing planar range scans with an angular resolution of $0.5°$ and a field of view of $180°$. The laser scan height is 0.028 meters, which corresponds to about $\frac{2}{3}$ of the knee height of a normal adult.

The algorithms were implemented in C++ as a plugin driver of Player, and therefore run on the server side. The feature visualization tool runs on the client side and was developed in OpenGL [20].

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 Section V.

II. Feature Detection

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

A. Range Segmentation

Range segmentation produces clusters 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. Since our laser readings have a maximum statistical error of 0.017 m [18] no error compensation scheme was implemented.

B. Circle Detection

Circle detection uses a new technique we called the Inscribed Angle Variance (IAV). This technique makes use of trigonometric properties of arcs: every point in an arc has congruent angles (angles with equal values) in respect to the extremes [16].

As an example of this property (see Fig.1) let P_1 and P_4 be distinct points on a circle centered in O, and let P_2 and P_3 be points on the circle lying on the same arc between P_1 and P_4, then

$$\angle P_1P_2P_4 = \angle P_1P_3P_4 = \frac{\angle P_1OP_4}{2} \qquad (1)$$

where the angles are measured counter-clockwise. The detection of circles is achieved calculating the average and `std` of the inscribed angles. Positive detection of circles occurs with standard deviation values below $8.6°$

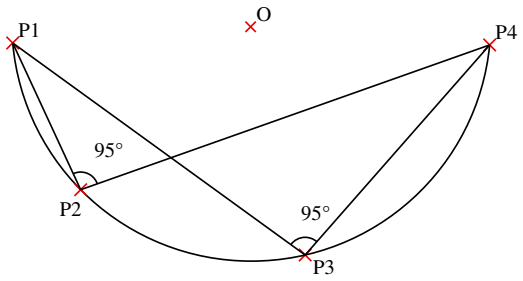

Fig. 1. The inscribed angles of an arc are congruent

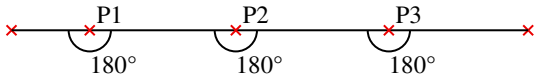

Fig. 2. The inscribed angles of a line are congruent and measure $180°$

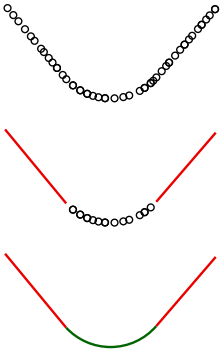

Fig. 3. The method for detection of arcs embedded in lines

and average values between $90°$ and $135°$. These values were tuned empirically to detect the maximum number of circles, while avoiding false positives. The confidence of the analysis increases with three factors: (1) the number of in-between points; (2) tightness of `std`; (3) the average inscribed angle value near $90°$. For an inscribed angle average of $90°$ half 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 procedure of circles but the average 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, see Fig. 3. For this technique to work we cannot allow small polygons to be formed, this imposes line detection configuration parameters with a very low peak error and high distance between line endpoints.

3) Detecting "arc-like" shapes: Suppose that we want to detect a tree trunk, or an object that resembles a cylinder; this is possible by increasing the detection threshold of the `std`. Experimental tests demonstrated that not so regular arcs can be found with `std` in the range between $8.6°$ and

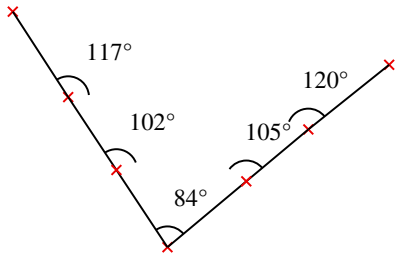

Fig. 4. The inscribed angles of a "V" shaped corner grow around a local minimum

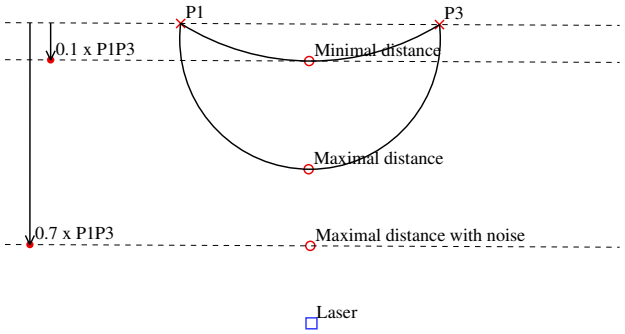

Fig. 5. Condition used in the selection of segments for circle analysis

$23.0°$.

When this acceptance threshold is increased false positives can occur. A further analysis is then required to isolate the false positives; an example of a false positive is the "V" shaped corner depicted in Fig.4. This case can be isolated by finding the point associated to the smallest inscribed angle and verifying if four neighbor points have progressively bigger inscribed angles, if this verifies we have a false positive detection.

C. Identification of the Circle Parameters

With the certain we have detected a circle, we need to find it's center and radius. From Euclid [16] we know that $P_1OP_4 = 360 - IAV$. It's also known that the sum of the angles of the triangle equals 180, so $P_1OP4 + 2\theta = 180°$, where θ is the angle OP_1P_4. Solving these equations we have $\theta = IAV - 90°$. To make the next part simpler lets translate the points P_1 and P_4 to the referential origin so that P_1 becomes 0, and rotate P_4 to make it lay in the XX axis. Next we calculate the mid-point between these two points that is $middle = \frac{P_4}{2}$, which will be the x coordinate of our temporary circle center. We calculate the height of the temporary circle center, $height = P_4 \tan\theta$. The center of the temporary circle will be $center = (middle, height)$. The circle radius is the distance from the origin of the referential to the center of the temporary circle. To get the final circle center just rotate and translate back to the original place.

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, the average and the `std` of the inscribed angles.

D. Circle Prerequisites

To avoid analyzing all segments for circles, each segment must validate the following geometric condition: the middle point of the segment must be inside an area delimited by two lines parallel to the extremes of the same segment, as can be seen in Fig. 5. To make this validation simpler, it is preceded by a rotation around the $x - axis$. Assuming that $P1$ is in the origin to simplify translations, the full operation is:

$$\theta = \arctan \frac{x_3 - x_1}{y_3 - y_1} \tag{2}$$

$$x_2 = -(x\cos(\theta) - y\sin(\theta)) \tag{3}$$

$$0.1 \cdot d(P_1, P_3) < x_2 < 0.7 \cdot d(P_1, P_3) \tag{4}$$

where θ is the angle to rotate, $P1$ and $P3$ are the Cartesian coordinates of the leftmost and rightmost points of the segment, x_2 is the x coordinate of the middle point, and $d(\cdot)$ is a Cartesian distance function.

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 the segments to analyze.

E. Leg Detection

The procedure for detecting legs is an extension of the circle prerequisites in Section II-D, with the extra constraint of the distance between end-points falling within the range of expected leg diameters ($0.1m$ to $0.25m$) and the segment not being partially occluded. Sometimes legs get classified as circles also.

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.

F. Line Detection

The line detection procedure uses the Recursive Line Fitting algorithm, that is summarized in three steps: (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 [19]. 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 recursive process break conditions are: (1) number of points to analyze below `line_min_pts` (see Table II); (2) distance(m) between extremes is below

`line_min_len` (see Table II); (3) fitting error under threshold (line fits).

To avoid creation of small polygons we established three requisites that all segments must obey : (1) at least 5 points or more; (2) distance between extremes greater than 0.1 m; (3) maximum fitting error 0.02 m;

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.

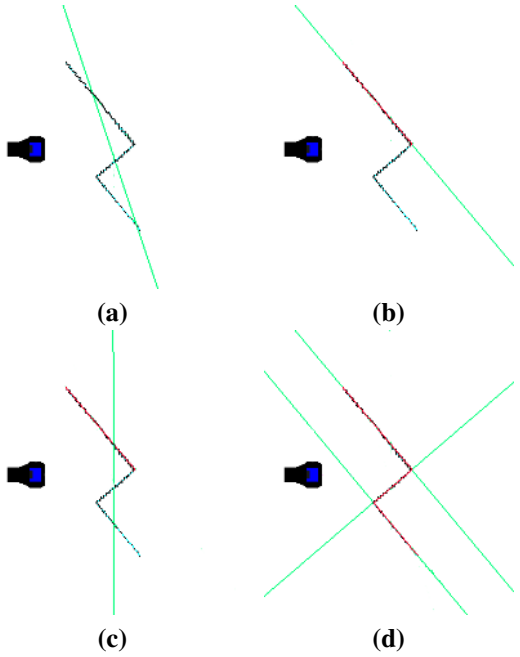

Fig. 6. The process of recursive line fitting: a) principle directions, b) first line segment identified, c) principle direction of second segment, d) both lines detected

III. Encapsulation in Player

The proposed algorithms were implemented as a Player [7] plugin driver. Player offers the following possibilities: (1) Hardware support; (2) Saving log files; (3) Client / Server architecture; (4) Open Source; (5) Modularity; (6) Stage and Gazebo, two simulators distributed in the same site [7]; 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 structures for data communications between clients and a server called interfaces. We proposed a new interface called the `laser feature 2D` that is composed of 11 fields. The first field is called `type` and as the name suggests indicates the detected feature; the following 10 fields further characterize the feature, as detailed in Table I.

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 is also possible to change/read configuration on the fly by sending a `configuration` request. Three configuration requests are possible: parameters modification, parameters query and laser pose.

TABLE I

Format of the interface data fields

Field #	Feature Type		
	Line	Circle	Leg
1	laser index of first point		
2	laser index of last point		
3	$left_x$	$center_x$	$middle_x$
4	$left_y$	$center_y$	$middle_y$
5	$right_x$	radius	—
6	$right_y$	average	—
7	slope	std	—
8	bias	—	—
9	# points	—	—
10	error	—	—

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
`line_err`	0.1	Line maximum error

The plugins are avaiable at http://miarn.cjb.net.

A. Visualization Tool

A visualization tool named PlayerGL was developed in OpenGL [20]. PlayerGL is a invaluable tool for testing, comparing and tuning the methods presented. It has the following capabilities: (1) Two type of camera views (plant and projection) with full freedom of movement; (2) On the fly configuration of the `laser` devices; (3) Two grid types, Polar and Cartesian; (4) Screen logging of detected features.

IV. Experimental results and discussion

To illustrate the accuracy of our methods two tests are shown, one with the robot moving and another with it stopped. Our laser was configured with a scanning angle of 180° with 0.5° of angular resolution for a total of 360 points, operating at a frequency of 5 Hz. Our robot is represented by the black object with a blue cube (representing the laser) on top. The Euclidean grid unit is $meters$.

A. Dynamic test

With the objective of showing the possibilities of SLAM using the proposed line detection method we run this test in a corridor of University of Algarve. The robot starts in an

interception of two corridors and then turns around 180°, in the process the number of observed corners increases, then it follows the corridor for some time. The corridor has paralell entrances that lead to classroom doors.

B. Static Test

This test occurred in a large hallway entrance, with some architectonic entities: two benches, two columns, walls and corners. Benches are formed by two short cylinders with radius equal to the columns radius. The robot stands static near one wall.

The test begins with a student pulling a perfectly round cylinder of radius equal to the one of columns, around two architectonic columns. Without being planned a second student passed by the low right corner, as shown in the sequence of photographs in Fig. 10.

One challenge is to detect correctly the mobile cylinder in all the course; right now false positive of benches happen when the mobile cylinder is at a distance from the other cylinders equal to the bench distance.

C. Results

The temporal sequence of snapshots in Fig.7 shows lines perfectly detected, this result is observed during the whole test.

The results in Fig. 11 show that the circle detection (green circumferences) was accurate, and no objects in the background were misdetected as circles. Scene interpretation is executed so that the architectonic entities (walls, corners, columns, benches) present are visualized.

Detected legs are represented as the tiny purple circles. It is possible to see the tracks of both students. 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.

The consistency of leg detection decreases as the range increases. This happens because the algorithm operates with at least 3 points. One way to compensate this loss of detail is increasing the laser angular resolution to $0.25°$.

D. Computational Time

The plot in Fig. 8 illustrates the computational time required to identify all lines and circles in every iteration (360 range points). Computational time concerning leg detection is not shown since the algorithm involves few calculations. These tests were executed in the Celeron 500 Mhz laptop equipped with 128Mb of RAM that stands on the robot. Analysing the plots we conclude: (1) line detection times oscilates with the number of subscan divisions, i.e. corners; (2) circle pre-requisites saves precious processing time; (3) both algorithms are fast, specially when following corridors without corners; (4) the worst time for lines was $1.940ms$ and for circles was $1.082ms$.

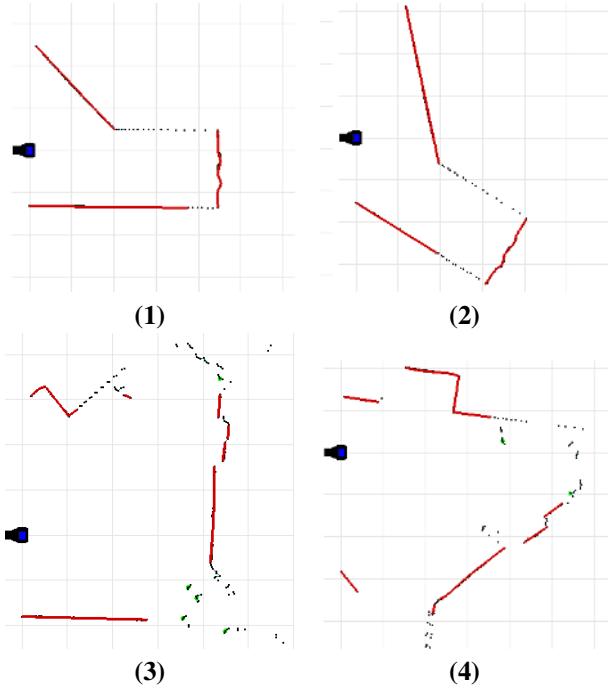

Fig. 7. Lines detected (in red) in a real test scenario

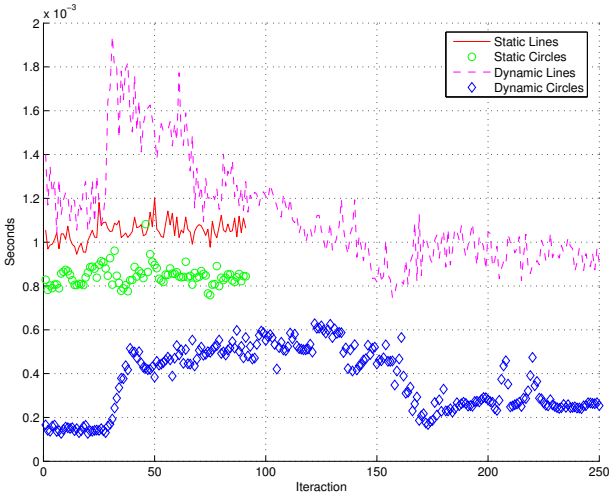

Fig. 8. Computational times for detection of lines and circles for the static and the dynamic tests

V. Conclusion and Future Work

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

The Player feature detection driver provides an abstraction layer for top layers. Distribution in Player ensures that these algorithms will be further improved and others methods will be added. It's also previewed a output plugin for grouping of partially occluded features by fitting points to the known features parameters. Our visualization tool PlayerGL will interpret sets of primitives for real-time 3D interpretation of trashcans, doors, chairs, etc. The final goal is the reconstruction from laser scan data of a dynamic

scene, with moving persons, and differentiable structures.

REFERENCES

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

[2] R.O. Duda, P.E. Hart, "Use of the Hough Transform to Detect Lines and Curves in Pictures", *Com. of ACM*, Vol.18, No.9, pp.509-517, 1975.

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

[4] G. Borges,and M. Aldon, "Line Extraction in 2D Range Images for Mobile Robotics", *Journal of Intelligent and Robotic Systems*, Vol. 40, Issue 3, pp. 267 - 297

[5] A. Mendes, and U. Nunes, "Situation-based multi-target detection and tracking with laser scanner in outdoor semi-structured environment", *IEEE/RSJ Int. Conf. on Systems and Robotics*, pp. 88-93, 2004

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

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

Fig. 9. Panoramic photograph of static test scenario.

Fig. 10. Temporal sequence of photographs of our static test scenario.

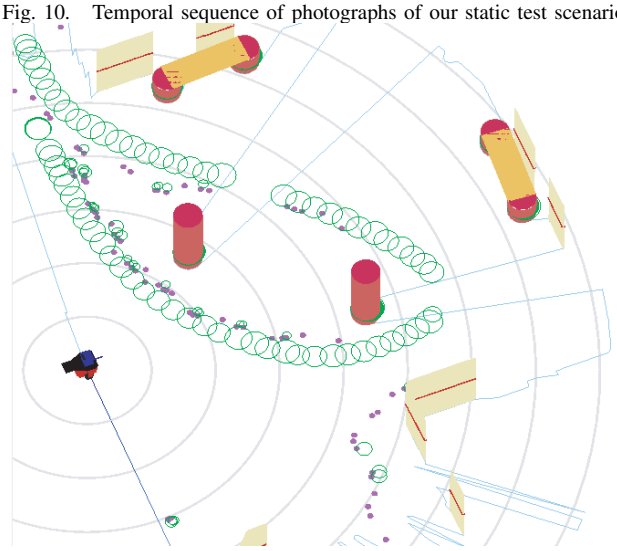

Fig. 11. Screen capture of PlayerGL at the end of the static test. The detected mobile cylinder is represented by the green circumferences, and legs are represented by tiny purple circles. Interpretation of walls, columns and benches is also achieved.

[8] 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 *IEEE Int. Conf. on Robotics and Automation*, pp.1665-1670, Seul 2001.

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

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

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

[12] Zhen Song, Yang Quan Chen, Lili Ma, and You Chung Chung, "Some sensing and perception techniques for an omni-directional ground vehicles with a laser scanner," *IEEE Int. Symp. on Intelligent Control*, Vancouver, British Columbia, 2002, pp. 690-695.

[13] Sen Zhang, Lihua Xie, Martin Adams, and Tang Fan, "Geometrical feature extraction using 2D range scanners" *Int. Conference on Control & Automation*, Montreal, Canada, June 2003.

[14] Kay Ch. Fuerstenberg, Dirk T. Linzmeier, and Klaus C.J Dietmayer, "Pedestrian recognition and tracking of vehicles using a vehicle based multilayer laserscanner" *ITS 2003, 10th World Congress on Intelligent Transport Systems*, November 2003, Madrid, Spain.

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

[16] Euclid, "The Elements", Book III, proposition 20,21.

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

[18] C. Ye and J. Borenstein, "Characterization of a 2-D laser scanner for mobile robot obstacle negotiation," *IEEE Int. Conference on Robotics and Automation*, Washington DC, USA, 2002, pp. 2512-2518

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

[20] OpenGL, [Online], http://www.opengl.org.