Study and Improvements in Landmarks Extraction in 2D Range Images Based on an Adaptive Curvature Estimation

Novel Certad, Raul Acuna, Ángel Terrones, Dimitar Ralev, Jose Cappelletto, Juan Carlos Grieco
Mechatronics Research Group
Universidad Simón Bolívar, Baruta, Venezuela
Email: ncertad@usb.ve

Abstract—**In this work we present an analysis of the natural reference extraction method (corners, intersections, hallways) using the estimated curvature function for 2D images provided by a laser range finder. Some of the problems from the original method introduced in 2007 are considered and possible solutions to this problems are proposed. Finally, the original and modified methods are submitted to various simulated benchmarks in order to compare their performance as a function of uncertainty, correctness, execution time and other parameters.**

***Keywords*-Feature extraction; Adaptive estimation; 2D range images; Robot sensing systems**

I. Introduction

In the area of mobile robot navigation in indoors structured environments, no matter what the goal or task assigned to the robot, always the main problem is to obtain an estimate as accurate as possible of the position as it moves through the environment. In the last 15 years, it has been common the use of position estimators based on the observation of references belonging to the environment (corners, hallways, columns, walls) where the position of this references together with the position of the robot, conforms the estimated state vector [1]. One of the sensors more used in this type of work is the laser range finder, and the extraction and processing of spatial references (landmarks) from the data supplied by these sensors has become a subtopic of vital importance for the navigation of mobile robots. The data provided by a single scan of the laser range finder, commonly known as 2D range image, must be processed to extract a more compact group of references which may be compared with others previously found and saved in a stored map. Different types of references are considered depending on the type of environment where navigation is performed. In this paper we consider only some references of those proposed by [2]. These are:

- *Breakpoints:* are scan discontinuities due to the change of surface being scanned or discontinuities between a surface and out of range scan (ruptures). They are only considered for extraction of the remaining references.
- *Corners:* are due to the change in the orientation of the scanned surface or due to the change of surface being scanned in a continuous scan.
- *Edges:* are breakpoints associated with the end of plane surfaces. A convex corner could be detected as an edge.

II. Reference Extraction

First is necessary to consider the notation. Let it be the n-th measurement obtained by the laser range finder in a single scan as $p_n = (r_n, \phi_n)$, where $n = \{1, 2, 3, \cdots, N\}$ for a single scan of N points. For each point, p_n, r_n is the range measurement from the laser to the obstacle and ϕ_n is the orientation in respect to the laser axis of reference. In order to extract spatial references of the information contained in the measurements, several steps must be executed which will be explained in this section. First, the breakpoints detector in charge of separating the data into subsets of points corresponding to continuous scanning will be introduced. Later the function of curvature of each subset is estimated to subsequent perform the segmentation based on it.

A. Breakpoint detector

For each measurement, p_n,exist a flag k_n^b, which denotes the presence or absence of a breakpoint. So, if between two consecutive measurements p_n y p_{n-1} exists a discontinuity large enough to be considered a breakpoint, the flags k_n^b y k_{n-1}^b will be set to the logical value TRUE.

$$\text{if } ||p_n - p_{n-1}|| \geq D_{max}, \text{ then } \; k_n^b = k_{n-1}^b = \text{TRUE.}$$

Where D_{max} is the maximum euclidean distance that can exist between the points p_n and p_{n-1} so they will not be considered as a breakpoint. Unlike the trivial approach of using a fixed distance, in [3] the utilization of an adaptative threshold D_{max} is proposed as shown in (1).

$$D_{max} = r_{n-1} \cdot \frac{\sin(\Delta\phi)}{\sin(\lambda - \Delta\phi)} + 3\sigma_r \tag{1}$$

where λ is set experimentally. In tests, the detector showed excellent results, however, the fact that the threshold D_{max} is proportional to r_{n-1} and r_n is not considered determine the outcome of detection. If measurements p_n are iterated in the $n = 1 \rightarrow N$ direction, different results will be obtained than iterating in the $n = 1 \leftarrow N$ direction. For example, measurements in corridors fairly symmetrical (a frequent case) the thresholds used in measurements corresponding to the right wall are not equal to those used in the left wall. To solve this problem we introduce a modification in (1).

$$D_{max} = min(r_{n-1}, r_n) \cdot \frac{\sin(\Delta\phi)}{\sin(\lambda - \Delta\phi)} + 3\sigma_r \quad (2)$$

In the Fig. 1, original detector and the proposed one are being compared.

B. Adaptive Curvature Estimation

The function of curvature is expressed in degrees and represents how far a set of points are from being straight. The peaks of the curvature function correspond to corners and points with zero curvature on average corresponds to line segments.

As indicated above, the breakpoints divide the data into continuous measurements segments related to the detection of a continuous surface. However, this surface may be curved or be the union of several line segments with different orientations. In 2007 [2], [4] a method is introduced to solve this issue in conjunction with an algorithm to estimate the curvature function of a set of discrete and noisy points. The algorithm is based on determining two local vectors, $\vec{f_n}$ and $\vec{b_n}$, associated with each measurement of distance. Where $\vec{f_n}$ estimates the direction between p_n and the consecutive points and $\vec{b_n}$ estimates the direction between p_n and the preceding points. The angle associated with each measurement is calculated by (3) .

$$k_n = \arctan(\frac{\vec{f_n}.\vec{b_n}}{|\vec{f_n}|.|\vec{b_n}|}) \quad (3)$$

The determination of the vectors $\vec{f_n}$ and $\vec{b_n}$ depends on the adjustment of the threshold u_k, which determines how smooth or how noisy is the estimated curvature function. While smaller are the values of u_k the estimated function is noisier, however, the peaks belonging to the detection of corners are better defined. As the U_k values are larger, the estimated function is smoother but the peaks are less defined and even may loose height.

C. Segmentation

From the estimated curvature function the remaining geometric references described in section I can be extracted. A measurement, p_n, belongs to a corner if its corresponding K_n is a local maximum and is above the detection threshold *CornerCurvThres*. A set of consecutive N_s measurements, $p_n, \cdots, p_{n+N_s}$, are considered a segment if they are below the detection threshold *SegmentCurvThres* and $Ns \geq MinNumPoints$. An edge is a point, P_n, which is simultaneously a breakpoint of the endpoint of a line, also, when compared with the next nearest breakpoint, P_m, it mus comply that $r_n < r_m$ to ensure that it is not a false edge produced by the obtaculizacion of a surface with another one in front.

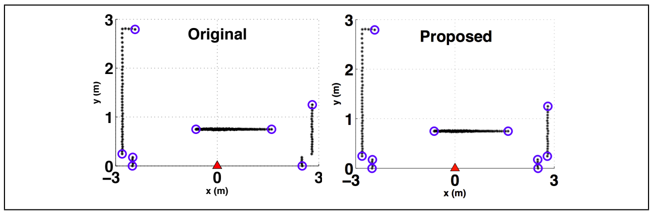

Figure 1. Breakponit detector comparison, dots are measures and blue circles are detected breakpoints. (left) original, (right) proposed.

D. Modifications

During the analysis of the results that were obtained with different values U_k, it was observed that while trying to adjust its value to encourage the detection of corners, the detection of edges was hindered and vice versa. Therefore it was sought a method that would combine two different estimates of the curvature function. The resulting proposal is based on estimating two different curvature functions (using two threshold values ($U_k = [U_{k1} \quad U_{k2}]$)) and then taking the minimum of both values $U_{k1} < CompThreshold$ and the maximum when $U_{k1} > CompThreshold$. Below is shown a more complete description of the algorithm:

Algorithm 1 Generation of a new curvature function

```
Require: two estimated curvature function $k_n^1$ provided by
    the original algorithm $k_n^2$
Ensure: A new estimated curvature function: $k_n^{new}$
 1: for $n = 1$ to $N$ do
 2:     $k_n^{min} \leftarrow min(k_n^1, k_n^2)$
 3:     $k_n^{max} \leftarrow max(k_n^1, k_n^2)$
 4:     if $k_n^{min} > CompThreshold$ then
 5:         $k_n^{new} \leftarrow k_n^{max}$
 6:     else
 7:         $k_n^{new} \leftarrow k_n^{min}$
 8:     end if
 9: end for
10: return
```

In section IV it will be shown that the estimated curvature function obtained with this algorithm has the advantage of being less noisy and better with more defined peaks than the originals. On the other hand, when it comes to determining the estimated curvature in the vicinity of the breakpoints, proturding peaks originate although the data corresponds to a line segment, this is because the points beyond a breakpoint for the calculation of the corresponding vector ($\vec{f_n}$ or $\vec{b_n}$, depending on the case) can not be considered. In some cases these points rise above *SegmentCurvThres* therefore they are no longer considered as part of the segment and the edge can no longer be detected. The last proposed modification solves this problem greatly. Once a continuos set of points $p_n, \cdots, p_{n+N_s}$ are considered part of a segment, the endpoints p_n and p_{n+N_s} are checked. If there is a single point between one endpoint and the closest breakpoint, the curvature of that point is compared with 2**SegmentCurvThres* (two times the used threshold) and if it is less, it is considered part of the segment. This approach allows the detection of a large amount of additional edges to the ones detected by the original algorithm.

III. Implementation

A. Software

The algorithms described in section II were implemented using MATLAB 2009b together with Webots 6, running on a laptop with Core i5@2.4 GHz and 4 GB of memory. Webots 6 allows the use of an ODE physics engine that simulates the dynamic behavior real time, it can also simulate a range finder laser with the statistical error associated in the measurements while considering the conditions of luminance and color of the object surfaces. A robotic platform of size, shape, weight and other properties similar to MobileRobotics Inc.'s Amigobot was implemented, as well as a rage finder similiar to the SICK-LMS200 model, with the following features:

- $\sigma_r = 0.005$: standard deviation of range measures provided by the Laser range finder.
- $\Delta\phi = 1°$: angular resolution of the laser range finder.
- $ROV = 10m$: Range of view.
- $AOV = 180°$: Angle of view.

B. Parameters

In order to maximize the detection reference, the algorithm parameters for the extraction and estimation of the curvature function were established after several tests.

- U_k*:* 0.001, **0.005**, 0.01, **0.05**, Thresholds used in the estimation of the curvature function(thresholds in bold were used for proposed algorithm).
- *MinNumPoints = 8:* Minimum number of points per line segment.
- *CornerCurvThres = 75°:* Threshold for detect a Corner.
- *SegmentCurvThres = 10°:* Threshold for detect a Segment.
- *CompThreshold = 55°:* Threshold for the proposed algorithm.

IV. Results

In order to perform a graphical analysis several 2D range images were obtained with the simulation software. Below is an analysis of a couple of images. In Fig. 2 are shown the estimated curvature functions at 4 different treshold values, U_k, and with the proposed modifications. Fig. 3 shows the references extracted by considering each of the curves shown in Fig. 2, the points represent the measurements obtained by the laser, the diamonds are the corners detected and the squares are the edges. Fig. 4 and Fig. 5 show the same information for another scan of the range finder laser. For Fig. 2 as well as Fig. 4 is found that for smaller values

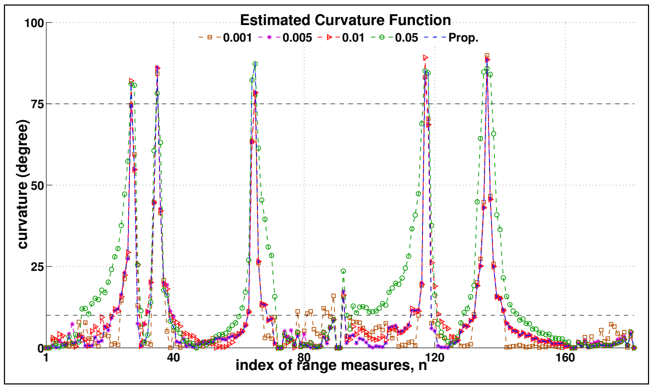

Figure 2. Estimated Curvature Functions for the scans shown in Fig. 3

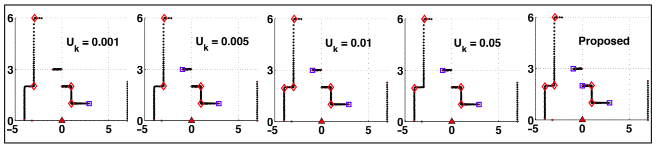

Figure 3. Extracted references with different threshold settings: $U_k = 0.001, 0.005, 0.01, 0.05$ and proposed (from left to right)

of U_k the function is more noisy, with peaks that can be interpreted as false corners such as in the 6th, where there is a corner in the middle of a line segment because of a noise spike in the corresponding curvature function. Studying in detail the estimated curvature functions in Fig. 4 we see that in the $n = (100, 180)$ range, the first three functions are very noisy and make impossible the detection of the line segment formed through the corresponding points in the measurements. In the $n = (30, 50)$ range takes place a high peak in all the estimated curvature functions, except for the modified, because there are a pair of breakpoints in $n = (100, 180)$ that are detected by the latter only, being the only function that remains close to zero in that range, which allows the detection of a pair of edges (Fig. 5) that are not detected by any of the other implementations. Fig. 3 shows how the algorithm with the proposed modifications is able to detect the 5 corners and 3 edges in the image. In the following case, Fig. 5 does not detect an edge, yet it detects the most references of the 5 implementations studied.

To compare numerically the results were used parameters similar to those proposed in [5] adapted to the extraction of point references. The algorithms were compared for the amount of extracted references, the correctness of the extracted references and the execution time. The determination of the parameters of comparison was made using a test bench of more than 230 range images obtained by 2D simulation. Table I shows the results obtained for each implementation.

The time shown is obtained by dividing the total execution time among the number of 2D range images stored in the test bench. The correctness measures are defined as follows

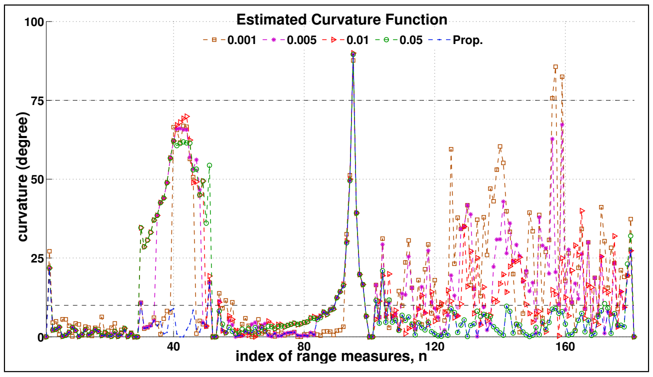

Figure 4. Estimated Curvature Functions for the scans shown in Fig. 5

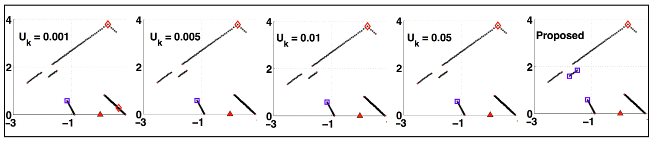

Figure 5. Extracted references with different threshold settings: $U_k = 0.001, 0.005, 0.01, 0.05$ and proposed (from left to right)

[5] :

$$TrueP = \frac{NumMatches}{NumTrueRef}\% \quad (4)$$

$$FalseP = \frac{NumRefExByAlgo - NumMatches}{NumRefExByAlgo}\% \quad (5)$$

where:

- $NumTrueRef$: is the number of real reference points.
- $NumRefExByAlgo$: is the number of reference points extracted by an algorithm.
- $NumMatches$: is the number of matches to real reference points.

Looking at the last two columns of Table I, we see that the proposed method has the highest rate of true detection in conjunction with one of the lowest rates of false detections. The four implementations of the original algorithm have similar true detection rates but as the threshold,U_k, decreases, the false detection rate increases. In terms of execution time, the original algorithm is more faster than the proposed algorithm, this is because the proposed algorithm is based on the calculation of two estimates of the function of curvature. The value of U_k is proportional to the time of execution because the number of points taken into account to estimate each value of the function of curvature is greater when U_k is greater. The execution time can be crucial if required real-time processing and not have a computer with great features. The modification made in the breakpoint detector is evident when analyzing the number of breakpoints found, however, its effect is also implicit in the number of references found. Comparing the four original algorithm implementations to each other, we find that the studied parameters are quite similar, except for the false positive rate of the first algorithm which is the largest of the list. This false positive rate so high, is easily explained by looking at the Fig. 2 and Fig. 4 which shows that the estimated curvature function corresponding to $U_k = 0.001$ is the most noisy, with many spurious peaks.

Table I
COMPARISON OF RESULTS

	Threshold U_k	Time [ms]	Num. Ref.	Num. Break.	Correctness	
					TrueP [%]	FalseP [%]
Orig.	0.001	12,2	1152	3670	62	18
	0.005	13,3	1044	3670	63	8
	0.01	14,2	1004	3670	64	3
	0.05	16,5	940	3670	61	1
Prop.	0.01 & 0.05	26,2	1215	3759	77	4
Truth	-		1531	-		

V. Conclusion And Future Work

In this paper we proposed three simple modifications to the reference extraction method for estimated curvature function. These were: A small modification in the breakpoints detector, an algorithm to fuse two estimates of the curvature function and make it a better one, and an amendment to the segmentation algorithm that affects the extraction of edges. The results show that the proposed algorithm is more reliable and detects a greater number of references to the original, at the cost of a considerable increase in processing time. Future work includes the physical implementation of the proposed method in a real robotic platform a laser range finder that will be acquired soon by the research group.

References

[1] J. H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping (SLAM): Part I the essential algorithms", *IEEE Robotics & Automation Magazine*, vol. 13, pp. 99-110, Sep. 2006.

[2] P. Núñez, R. Vázquez-Martín, J. C. Toro, A. Bandera, and F. Sandoval. (2008, Mar.). Natural landmark extraction for mobile robot navigation based on an adaptive curvature estimation. Robotics and Autonomous Systems [Online]. 56(3), pp. 247-264. Available: http://dx.doi.org/10.1016/j.robot.2007.07.005

[3] G.A. Borges, M. Aldon. (2004, Jul.). Line extraction in 2D range images for mobile robotics, J. of Intelligent and Robotic Systems [Online]. 40(3), July 2004, pp. 267-297. Available: http://dx.doi.org/10.1023/B:JINT.0000038945.55712.65

[4] P. Núñez, R. Vázquez-Martín, J. C. Toro, A. Bandera, and F. Sandoval, "A Curvature based Method to Extract Natural Landmarks for Mobile Robot Navigation," in IEEE Int. Symp. Intelligent Signal Processing, Alcalá, Spain, 2007, pp. 1-6.

[5] V. Nguyen, A. Martinelli, N. Tomatis, R. Siegwart, "A comparison of line extraction algorithms using 2D laser rangefinder for indoor mobile robotics," in IEEE/RSJ Int. Conf. Intelligent Robots and Systems, Edmonton, Canada, 2005, pp 1929-1934.