Journal of Theoretical and Applied Information Technology
31st October 2020. Vol.98. No 20

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

# SKELETONIZATION-BASED PATH PLANNING ALGORITHM FOR AUTONOMOUS ROBOTS USING A COMBINATION OF GEOMETRIC ALGORITHMS AND RECURSION

**[1]EDWAR JACINTO GÓMEZ, [2] HOLMAN MONTIEL ARIZA, [3]FERNANDO MARTÍNEZ SANTA**

[1,3]Assistant professor, Universidad Distrital Francisco José de Caldas, Technology Faculty, Bogotá, Colombia
[2]Associate professor, Universidad Distrital Francisco José de Caldas, Technology Faculty, Bogotá, Colombia

E-mail: [1]ejacintog@udistrital.edu.co, [2] hmontiela@udistrital.edu.co, [3]fmartinezs@udistrital.edu.co

**ABSTRACT**

This article presents a strategy for the calculation of paths in indoor and static labyrinth-type environments, using a camera on the stage as a sensor. An algorithm is applied that uses the skeletonization of a binary image as the basis for the calculation of the path, which, despite being a technique that comes from image processing, is used in a satisfactory way to this problem in a variety of scenarios with different degrees of difficulty, so that it can solve any type of labyrinth with different geometry and number of branches in its path. In this implementation a refined algorithm is used that applies two techniques derived from the skeletonization of the image and that take different paths to reach a possible solution; the algorithm determines the complexity of the problem by evaluating the amount of branches that the binary skeleton has, if the number of possible branches of the binary tree is less than a threshold, then, a technique that measures the distance of each path is applied as a strategy to calculate the shortest path, if the number of branches is greater than the determined threshold, a technique that removes points at the end of paths not connected to a possible path is applied, this is done by calling a routine in a recursive way until the only two end points in the path are the start and end points of the path.

**Keywords**: *Autonomous robot, Path planning, Skeleton Technique, Maze.*

## 1. INTRODUCTION

With the current computing capabilities, have allowed a number of solutions using image processing, either in local application or from the cloud, this work will show several applications of techniques used to calculate the trajectory of a mobile robot differential type in static environments, which in the end can be applied in dynamic environments such as ad-hoc vehicle networks applied to services in civil roads [1], seeking control of vehicles, recognition and tracking of people, to obtain a response to detect suspicious people and vehicles that do not comply with the rules established in the road [2] [3].

In route planning systems for mobile robots, systems are required to verify the position of the mobile robot and obstacles, so in most applications it is required, in addition to the use of machine vision system, to use sensor systems as backup to avoid collisions and / or local minimum, in this case the most used systems for its low cost and ease of implementation are ultrasonic sensors [4] and optical type sensors. These two systems have evolved and allow two-dimensional 360-degree maps to be made using sensor rings in the case of ultrasound or rotating systems when an optical sensor is used, making navigation systems more accurate and easier to use in support of machine vision tools.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

The autonomous operation of each sensor regardless of the nature [5][6] and their low cost, make these systems applicable to the development of more complex algorithms, which have given a new direction to autonomous on-board navigation systems [7][8]. These algorithms, supported by the advance of the sensor, have been applied to land vehicles and autonomous flyers [9] connected to the ground wirelessly. They process the information by means of SLAM (Simultaneous localization and mapping), which corrects the problems of loss of information, and to minimize these errors, an extended Kalman filter method (EKF) [10] has been incorporated to avoid errors in navigation, In other words, the combination of signal processing algorithms and the use of measurement systems results in very precise positions and 3D environment modeling [11] [12]. In addition to the position of the mobile phone, sensor systems can be used to measure other variables in highly dynamic environments such as forest fires, obtaining strategies from measurement of color signals and fire movement [13].

Satellite communication systems with operating sensors installed on the ground give two ways of obtaining relative information, with such a high amount of information that it is possible to obtain real positioning by removing external disturbances that influence the sensor located on the ground [14] [15], also systems for surgical instrumentation based on 3D images increase the performance of the processes [16], implementing in parallel Graph Cuts algorithms under CUDA() processes in the order of organ segmentation [17] [18].

Thanks to the high performance of today's computer equipment, it is possible to generate solutions that apply different strategies to solve a problem, each one of them applying different algorithms, making the system able to make decisions autonomously, as to which strategy to use in each case, improving response times and possible errors due to external disturbances [19] [20].

In this specific work it is decided to use an algorithm, which takes as a parameter the number of branching points to measure the complexity of the problem and use a faster geometric technique for simple labyrinths, but when the labyrinth has many branches it uses a recursive algorithm, which guarantees a successful resolution of the problem regardless of the degree of complexity.

## 2. METHODOLOGY

When performing route planning and navigation algorithms for an unmanned robot on a plane with immovable objects, it is important to perform the solution looking for the best use of resources, since the hardware on board is the key point for the selection of a processing algorithm that fits what we are looking for in this specific task.

After extensive research and new ideas have generated new ways to analyze a plane and cross it properly regardless of the size and geometry of the unmanned robot, many of these collections are based on analysis with an image acquisition system with zenithal positioning, this can be seen in Figure 1 and communication of the robot wirelessly, performing the processing of the entire plane in just fractions of seconds.

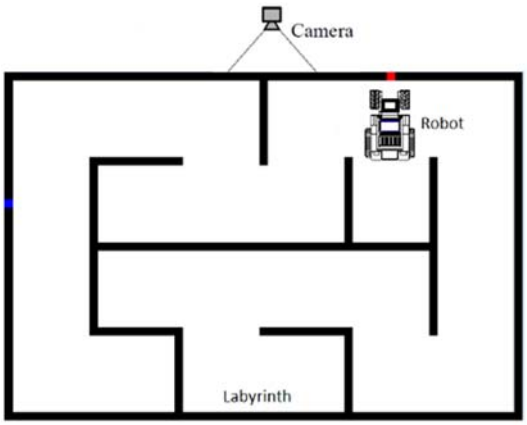



*Figure:1 Route Planning Using Zenithal Plane.*

The algorithms based on physical homologues are the most familiar so that a person who wants to make an implementation can understand the fundamentals of operation of each step for the trajectory backward.

### 2.1 Navigation Algorithms

A robot which has a notation point in the area of $R^2$ is positioned at the point $p_i$ within a $W$ navigation environment $\square$ $R^2$ connected and compact with no bay-type obstacles and clearly

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

defined boundaries. The limit or boundary of *W is* indicated at *∂W* shown in figure 2. The free space in which the robot can navigate is denoted by the letter *E*, according to the configuration of *W,* if there is at least one path in *E,* this allows the robot to move from the starting point $p_i$ to the point of destination $p_p$If the robot is not in the path, then the strategy must define an optimal (some shorter) and safe path for the robot's movement.

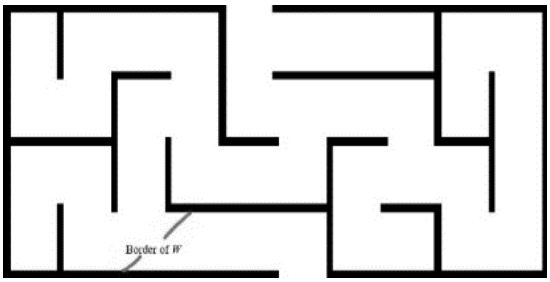



*Figure 2: Navigation environment with details of the problem formulation [21]*

As in known schemes, the navigation medium is divided into a finite set of regions, over which a finite transition system is defined. These achieved regions are delimited by the limits of the environment *∂W* and the edges resulting from the skeletonization process applied to the free space *E.* These edges establish the possible navigation paths, and are defined as points of equidistance from *∂W.*

Each edge of the skeleton is denoted by *Y,* and its ends are either common with *∂W* or are part of an edge or node of the skeleton. The set of all skeletal edges is denoted by □. Each $y_i \in \lceil$ is the image of an injectable and rectifiable curve defined between the boundaries of the *∂W* environment and the edges of the *E-skeleton*. rand all regions by *R*.

The robot is considered small in relation to *W* and the *r* regions ∈ *R. In* addition, the size of the robot is considered limited by a diameter circle *d* in $R^2$. The robot must move from $p_i$ to a destination $p_t$ navigating through *E*. In addition to this, we want the selected path to be as short as possible. The navigation algorithm must plan the navigation route so that the required borders are selected $y_i$to make a safe and efficient route between the $p_i$ y $p_t$provided that the geometry allows the robot to pass through the navigation space *E*.

## 2.2 Skeletonization

Morphological processes in image processing are applied with elements with a spatial component, more called a structuring object. Mathematical morphology is a non-linear image processing technique, based on set operations.

The geometrical vision of operation is to compare objects with different shapes, but one of them is with an easily recognizable pattern, this is called a structuring object. This is seen in figure 3.

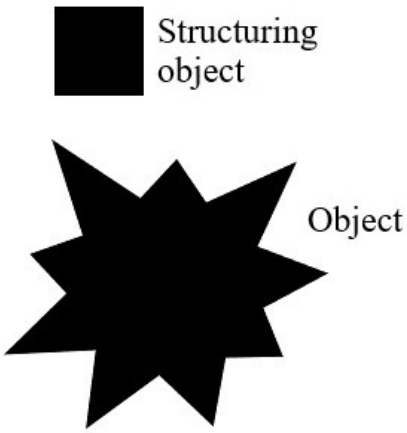



*Figure 3: Object to be processed and its structuring element [22]*

The behavior is seen from set theory, this is described in a full image, which will exemplify the types of operations that are used in general image processing based on set theory Figure 4 will describe the operations.

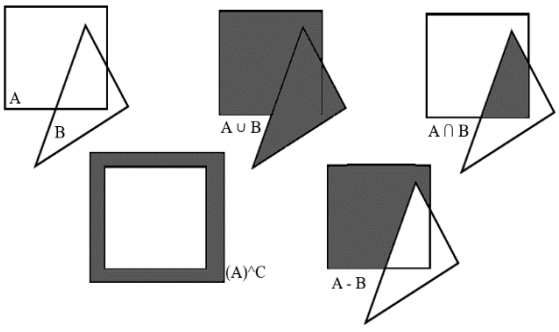



*Figure 4: Different geometric figures A and B show the possible assembly operations. [22]*

Journal of Theoretical and Applied Information Technology
31st October 2020. Vol.98. No 20

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

In the basic morphological operations used in image processing most often are dilation ⊕ and erosion ⊖. Its applications are innumerable, depending on the choice of the structuring object, can be used to find edges, correct errors, search for certain geometry in the original image, in figure 5 are shown a couple of examples of expansion in a simple image with two different structuring objects.

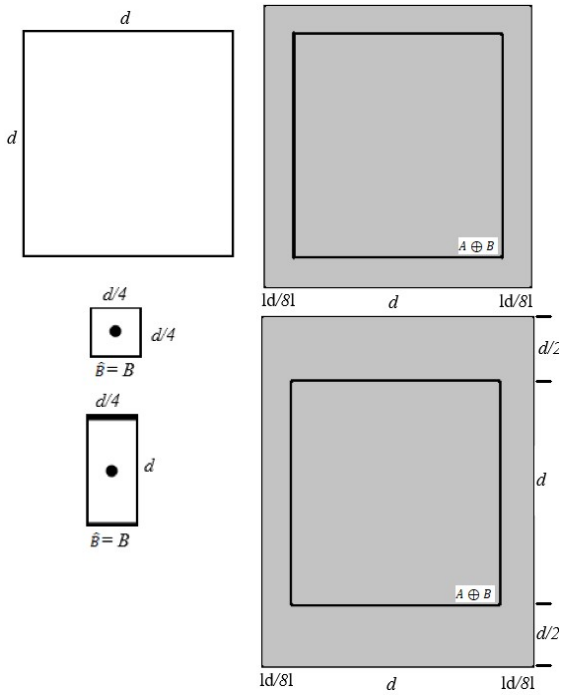



*Figure 5: Expansion processes with structuring elements of different geometries. [22]*

For A and B sets in $z^2$The dilation of A by B, denoted as A⊕B is defined by equation 1.

$$A \oplus B = \{Zl(\widehat{B})_Z \cap A \neq \square\} \quad (1)$$

The expansion properties are given by equations 2,3,4,5.

$$Commutative: A \oplus B = B \oplus A \quad (2)$$

$$Associative: A \oplus (B \oplus C) = (A \oplus B) \oplus C \quad (3)$$

$$Extensiveness: si\ 0 \in B, A \subseteq A \oplus B \quad (4)$$

$$Increasing\ expansion: A \subseteq B\ Itinvolves\ A \oplus D \subseteq B \oplus D \quad (5)$$

The morphological operation erosion is one of the most used operations, since it allows to correct objects, remove noise and search for edges, this using it in its basic form, as it is seen in figure 6 only changing the structuring object can achieve different effects, but the interesting thing of the morphological operations is that they can be used mixed, combined and even in an iterative way.

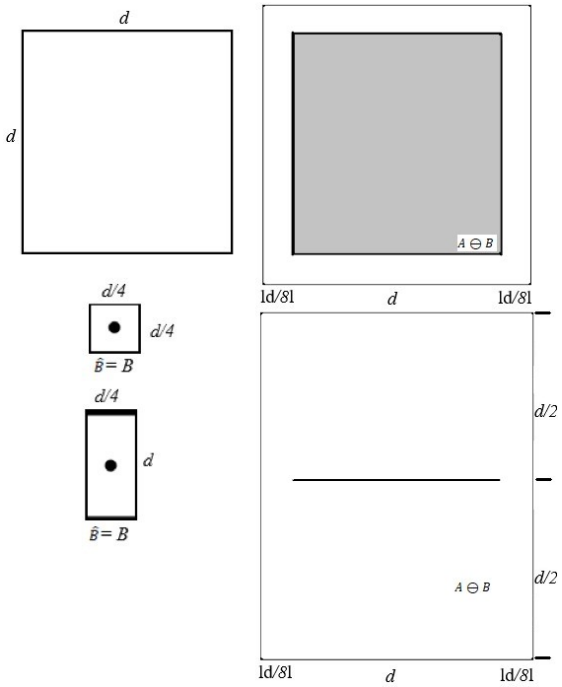



*Figure 6: Erosion processes with structuring elements of different geometries.*

For A and B sets in $z^2$The dilation of A by B, denoted as A⊕B is defined by equation 6.

$$A \ominus B = \{Zl(\widehat{B})_Z \subseteq A\} \quad (6)$$

Erosion properties are given by equations 7,8,9,10.

$$Non - commutative: A \ominus B \neq B \ominus A \quad (7)$$

$$Translational\ Invariant:$$

$$A_X \ominus B = (A \ominus B)_X$$

$$A \ominus B_X = (A \ominus B)_{-X} \quad (8)$$

$$Extensiveness: si\ 0 \in B, A \ominus B \subseteq A \quad (9)$$

$$Chain\ rule: A \ominus (B_1 \oplus \dots \oplus B_K)$$

$$= (\dots (A \ominus B_1) \ominus \dots \ominus B_K) \quad (10)$$

Journal of Theoretical and Applied Information Technology
31st October 2020. Vol.98. No 20

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

As said before, the operations erosion and dilation can be used in an iterative way, it is a skeletonization algorithm that is nothing more than an infinite iteration of an erosion to a binary image with a small structuring object of symmetric nature, this iteration only stops when the skeleton is only one pixel wide, this process can be seen graphically in Figure 7 in each of its stages going from the numeral a to d.

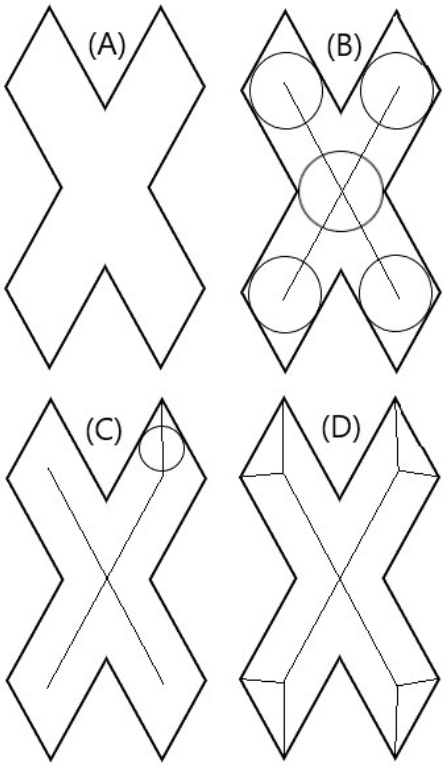



*Figure 7: (A) initial figure, (B)maximum disc size positions, (C)other maximum disc centered in the corners of the figure, (D)complete skeleton of the figure. [22]*

A mathematical definition of the skeletal function is made in terms of the basic functions, erosion, dilatation and their combinations, which is shown in equation 11.

$$S(A) = \bigcup_{K=0}^{K} S_K(A)$$

$$S_K(A) = (A \ominus KB) - (A \ominus KB) \odot B \quad (11)$$

With B being a structuring element and (A□KB) indicating the number of erosions of A, this change is seen in equation 12.

$$S_K(A) \oplus KB) = ((\dots (S_K(A) \oplus B) \oplus B) \oplus \dots) \oplus B \quad (12)$$

The skeletonization algorithm can be seen as a controlled and iterative erosion morphological operation. A repetitive erosion is performed on the image, until the object is thinned and regions separated by lines (edges) are formed, $y_i \in$ □). This erosion is done, using a structuring element or core matrix, which is square and smaller than *W*. The process of skeletonization produces navigable and safe edges.

The subsets of $y_i \in$ ⌈ may be part of the navigable and safe roads along *W*. Although, some edges of $y_i \in$ ⌈ lead the robot to collide with ∂W. In addition, two edges are missing, those that connect the graphic with the $p_i$ y $p_t$ This is shown in figure 8.

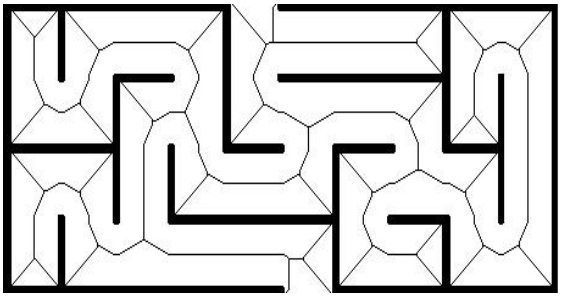
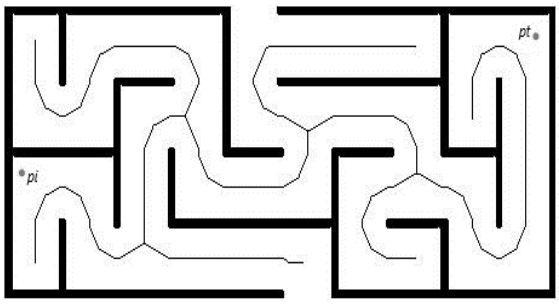


*Figure 8: Image skeleton generated for the E-space. [21]*

Before starting the path selection process, it is necessary to remove the colliding edges from the graph, which have points in common with ∂W. The filter applied for this, looks for the points of the graph that are intercepted with *∂W* (extreme points of the graph) and advances erasing the border until arriving at a vertex this is shown in figure 9.



ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

*Figure 9: Skeleton navigation environment plus starting point □□ and that's it □□. Edges have been removed and they had points in common with ∂W.* [21]

To detect both the pixels on the graph that intercept *∂W and the pixels on* a vertex, the eight neighboring pixels on each graph pixel are checked. The set of eight neighbors is defined for a pixel p (x, y), as shown in Equation 13. The number of active neighboring pixels is defined as shown in Equation 14.

$$N_\delta(p(x,y)) = \{p(x-1,y-1), p(x,y+1), p(x+1,y+1), p(x-1,y), p(x+1,y), p(x-1,y-1), p(x,y-1)p(x+1,y-1)\} \quad (13)$$

$$\nabla p(x,y) \in I: p(x,y) = 1 \rightarrow k = \sum_{j=1}^{\delta} N_\delta(p(x,y))_j \quad (14)$$

Pixels with k = 2 are border points, those with k = 1 are points that intersect ∂W, and those with k = 3 or more are vertices. To join the starting point $p_i$ and the end point $p_t$ with borders to the graph, the Euclidean distance between these points to each point on the graph borders is calculated. This allows the shortest distance to be found, a line that becomes a new border of the graph Figure 10.

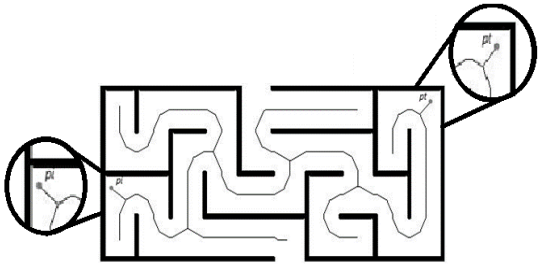


*Figure 10: Image of the skeleton composed of two edges connecting the points □□ y □□.* [21]

The image corresponds to the detection of *W,* information that is captured with a digital camera. Through the application of specific filters defined previously, a segmentation of the navigation environment *W* in regions *r ∈ R.* This segmentation uses the skeletonization of the free space *E, a* process that seeks to represent *E with a* graph where its edges are composed of points, which maintain the maximum distance to ∂W.

The image undergoes a basic process of identifying free spaces E. Each image is converted to grayscale and then to binary, once the obstacles are clearly identified from the background, an obstacle dilation or growth operation is applied to identify the safe area for the robot's navigation. This dilation process increases the size of the obstacles for processing by a factor of $\frac{d}{2}$This guarantees that the robot will not crash even though the path has been calculated for a single point.

The next step in the process is the selection of the optimal/sub-optimal path. The ideal criterion is to find the shortest path with the greatest turning angles. Due to the finite and small number of possible edge combinations and capable of being part of the path between $p_i$ y $p_t$first, a deep search algorithm is implemented. This algorithm is not applied to find the solution path, but to find all existing paths.

The algorithm begins by determining the starting point $p_i$ and end $p_t$ making a closed path in the image to be analyzed, after this it makes a tour through the different possibilities, making decisions of why branching adds the next path to the list of nodes, all this calculating the distance that the mobile robot has traveled, this is done until the point $p_t$In the end, a route is selected with the shortest distance possible, as shown in figure 11.

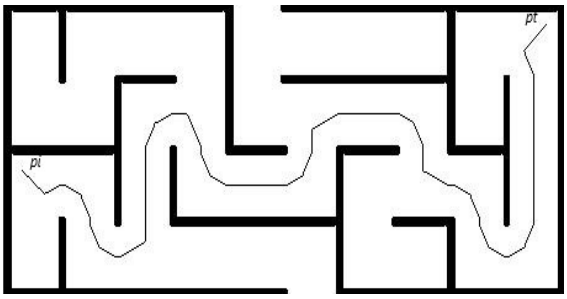


*Figure 11: Identification of the shortest path. Among all the possible edge combinations, the one with the lightest weight is selected.* [21]

This determines a possible path, which has a smaller distance metric between those analyzed, applying the image skeletonization technique, but it does not guarantee that it is the smallest of all, nor that the mobile can comply with it in terms of navigation, since it can present trajectories with closed angles or other type of anomalous trajectories or "artifacts" as a term derived from image processing techniques, to lower the possibility of

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

having this type of trajectories that have a characteristic.

one chooses to make the robot navigate through points and not be a line follower, reducing the set of points $p_t \subset \square$, using a technique called decimation, so that these points are only 5% of the original points, as shown in figure 8. To calculate this new family of points using arithmetic measurement, new points are calculated that lower the inertia of the displacement, smooth the trajectory and avoid acute angles and "artifacts" generated from the image processing technique and ensure that the robot travels along an average and possible trajectory, as shown in figure 12.

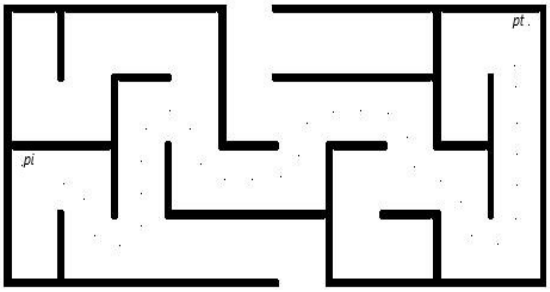



*Figure 12: Resulting points after doing a decimation process on the obtained path.* [21]

## 3. IMPLEMENTATION

An algorithm is made that takes as a basis for its operation image processing techniques applied to route planning, for this reason, a stage of pre-processing of the image is required, after that, low algorithmic complexity using a binary image, which accelerates the amount of calculations needed, after running the skeletonization algorithm that is the basis of the work done, followed by a process of capturing and joining the start and end points of the route to the original skeleton, at this point the algorithm, makes a count of the number of branches and makes the decision to perform an algorithm based on the measurement of Euclidean distance of each possible route segment or on the contrary if it uses a recursive algorithm that eliminates in an iterative way the end points until a possible solution is reached, this process is described in a graphic way in the flow chart in figure 13.

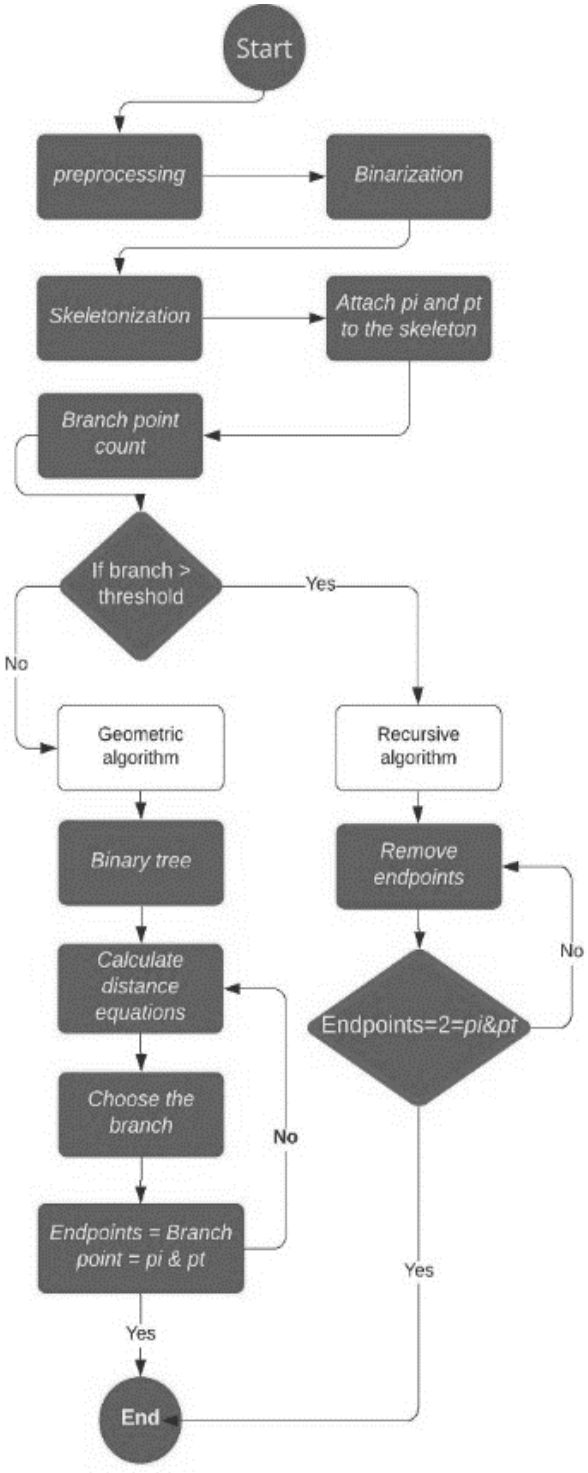



*Figure 13: Flow chart of the operation of the geometric and recursive algorithms for the solution of a maze.*

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

In the comparison of this type of navigation systems applied to robotics two approaches were raised, the application of this algorithm is based on how effective it is comparing the response time against the number of branch points. The approaches are based on the quantification of the length of the paths created with the points *pi* and *pt.* One taking the minimum distance possible and the other based on eliminating end points to find a possible route.

The description of the main points for the quantification algorithm will be described below:

```
%% Opens the stage image and
captures the start and end points
imageStage = imread('maze4.jpg');


% start and end points (x,y), in
values of the original resolution
figure(1)
imshow(imageScenario);
[px,py] = ginput(2);
initial = [px(1), py(1)];
end    = [px(2), py(2)];
```

The first part is to read the captured image and capture the start and end point, i.e. the start and end coordinates of the route for the calculation of the path.

```
%% image preprocessing
imageScenario =
preprocessing(imageScenario);
```

Image pre-processing is a function that scales and converts the maze into a binary system for processing

```
%% Skeletonization
imagePath =
bwmorph(imageScenario,'skel',inf);

%% Clears the dead ends
path2image =
clears_paths_without_out_output(pathim
age);
%pictureRoute2 = pictureRoute;

%% Joins the starting point and
the end point of the skeleton
path2 image = join_point(initial,path2
image);
path2image =
join_point(end,path2image);
%pictureRoute2 =
bwmorph(pictureRoute2,'skel',inf);
```

With the two skeletonized planes, one corresponds to the one generated by the initial point and the other by the end of the path, they are joined to complete the route joining the two skeletons, and it is complemented with the elimination of dead ends so that later the algorithm finds the possible route without losing processing time in this type of paths.

```
Unique recursive route selection
algorithm
[imageOutput, vectorPath] =
single_path(start,end,imagePath2,100);

%% Optimize your route
imRuteOptim =
optimize_path(pathvector,imageScenario
);
```

The fundamental part of this comparison is in the geometric algorithm, the route selection bases its calculation on a more robust mathematical system that seeks in a more sophisticated way the solution to an arbitrarily chosen labyrinth, this takes into account all the possible branching points and defines a route under all the possibilities of solution. An example of this and its execution time is seen in figure 14.

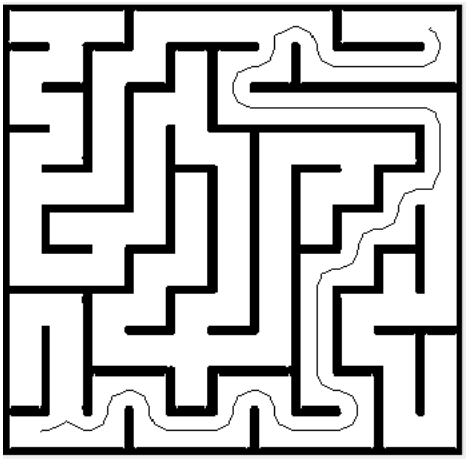


*Figure 14: Solution generated by the geometric algorithm, with a total of 82 branch points and 0.708 seconds of execution*

Now the behavior of the recursive algorithm will be shown, which, takes off the $p_t \subset \lceil$ to the points identified as end points of the original skeleton in a closed scenario, this algorithm does not

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

analyze the skeleton $p_t$ as a binary tree, but rather focuses on making use of the pure algorithm, through recursion as a unique and powerful tool, since by not performing mathematical calculations, but only focusing on a cloud of points, can attack more complex problems, but with a longer duration in the generation of satisfactory results, this results in a 100% effectiveness regardless of the number of branches and possible paths, routes or intersections that have the maze to solve.

The recursive algorithm performs the reading of the captured image, a pre-processing stage of the image until it reaches the skeleton, immediately after it performs a count and decomposition of the paths, and with the parameter of branching points, as a final part of the algorithm, the iterations of the paths generated in the path between pi and pt are eliminated recursively to find a solution to the path and that this represents the least distance this is shown in Figure 15.

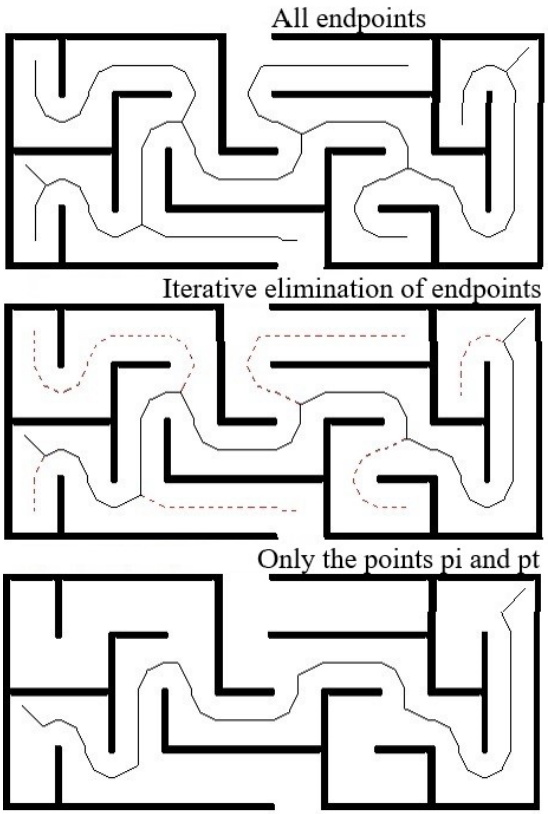



*Figure 15: Iterative endpoint removal.*

In this way, complex mathematical calculations are not performed, and the processing time decreases, guaranteeing a result regardless of the number of branching points (complex maze solutions are mostly appreciated). Thus, it is only a matter of time to find a way to more complex mazes without depending on the requirements of the system on which the algorithm is executed. An example of the solution given by this algorithm is shown in figure 16.

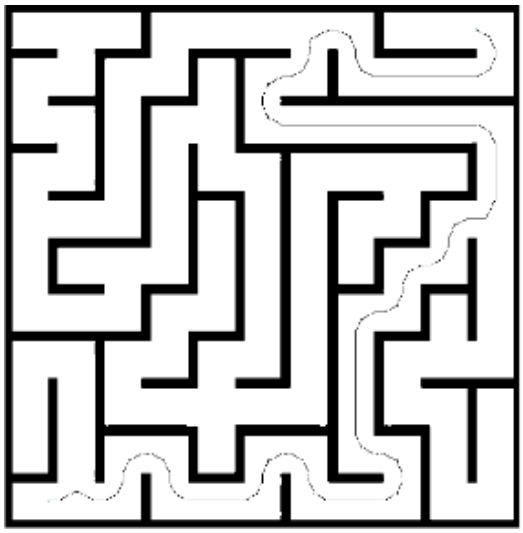


*Figure 16: Solution generated by the recursive algorithm, with a total of 82 branch points and 12,753 seconds of execution.*

## 4. RESULTS

The navigation strategies were evaluated with multiple mazes of which 11 were documented and chosen for their level of gradual difficulty, they were evaluated on the same device to ensure that both algorithms were evaluated impartially. In Table 1, the execution times of both algorithms can be seen with the number of branch points found and the easily observable limit of the number of branch points for each one.

*Table 1: Run times and branch points for the evaluated algorithms*

| Number of the labyrinth | Recursive algorithm time (S) | Geometrical algorithm time (S) | Number of branches |
|---|---|---|---|
| 1 | 0,6 | 0,236 | 20 |
| 2 | 3.102 | 0,64 | 18 |
| 3 | 5.462 | 0,361 | 32 |
| 4 | 5.588 | 0,44 | 68 |
| 5 | 12.463 | 0,932 | 82 |

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

| | | | |
|---|---|---|---|
| 6 | 3.720 | 1.018 | 88 |
| 7 | 4.387 | 0,509 | 162 |
| 8 | 15.864 | / | 189 |
| 9 | 4.192 | / | 283 |
| 10 | 26.707 | / | 636 |
| 11 | 56.471 | / | 956 |

An example of the execution of these algorithms is shown in Figure 17 comparing the execution times with the solution to the maze proposed for each of the algorithms.

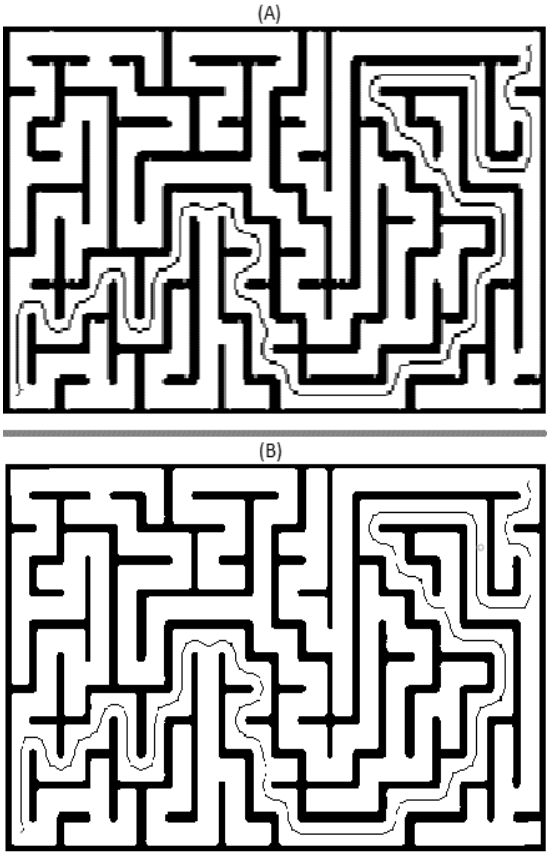



*Figure 17: Solution generated by geometric algorithm, image (A) and time 0.5094 seconds and counter position recursive algorithm, image (B) and run time 4.387 seconds.*

One of the most stringent tests was a maze of 636 branching points which had a solution time of 26,707 seconds for the recursive algorithm, whereas the geometric algorithm never had a solution for the proposal. Figure 18 shows the proposed solution, execution time and number of branches found.

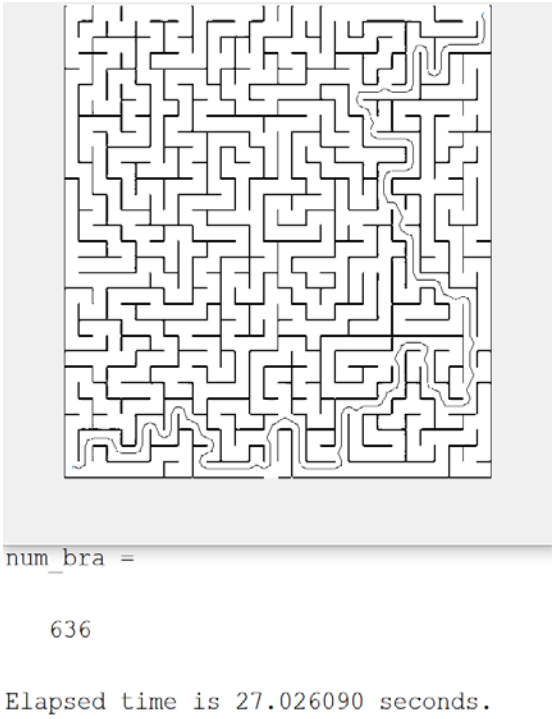



*Figure 18: Solution generated by geometric algorithm to a highly complex maze*

By testing in mazes with different levels of complexity, the recursive algorithm was able to find 100% of the solutions. In particular, the recursive algorithm showed excellent performance in really complex mazes, demonstrating a maze solution of up to 956 branch points.

In the case of the geometric algorithm, its speed of solution to labyrinths with less than 189 branch points is up to 3.5 times faster than the recursive algorithm, but in more complex labyrinths its use becomes unfeasible.

The tests were performed on an Intel i7-7700HQ with 12 Gigabytes of RAM running Windows 10 64 bits and Matlab R2019a and the exponential progression of the processing time with respect to the number of branching points can be observed.

## 5. CONCLUSION

Considering a series of previous works in route planning for mobile robots in static labyrinth-like environments, carried out by the ARMOS

research group, which are solved from the computational point of view, using image processing techniques combined with geometric and algorithmic strategies, a solution is reached that is applied depending on the complexity of the problem, measured in the amount of branches that the scenario analyzed has,

By using a mixed strategy ensures that always get a usable solution, so we made the integration of two algorithmic solutions, starting from a skeletonization technique, which, in the first place applies a geometric technique for small mazes and an algorithmic implementation for mazes with more branches, a threshold of 189 was determined to make the decision of which algorithm to use.

The recursive algorithm is erratic in its behavior, in that it is not linear taking as a metric the number of branches, which makes it applicable to off-line solutions, which do not have critical response times. This behavior can be seen in Table 1 and has to do with aspects of the maze geometry, the image resolution and the format of the input image file.

It is proposed as future work of the group and object of interest in the coming works to make algorithms to be implemented from video cards or FPGA's for parallel processing, and thus lower the processing times, so that they are usable in applications where the scenario is no longer static and that in the end lead to applications fully on board and totally autonomous.

## ACKNOWLEDGEMENT

This work was supported by Universidad Distrital Francisco José de Caldas and the Centre for Scientific Research and Development (CIDC). The views expressed in this paper are not necessarily endorsed by Universidad Distrital Francisco José de Caldas or the CIDC. The authors thank the research groups ARMOS and SIE and its research seedbeds for the evaluation carried out on prototypes of ideas and strategies proposed in this paper. The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

## REFERENCES

[1] C. Patruno, R. Marani, M. Nitti, T. D. Orazio, and E. Stella, "An Embedded Vision System for Real-Time Autonomous Localization Using Laser Profilometry," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 6, pp. 3482–3495, 2015, doi: 10.1109/TITS.2015.2459721.

[2] K. Hammoudi *et al.*, "Design , Implementation and Simulation of an Experimental Processing Architecture for Enhancing Real-time Video Services by Combining VANET , Cloud Computing System and Onboard Navigation System SYSTEM FOR REAL-TIME," *2015 Int. Conf. Pervasive Embed. Comput. Commun. Syst.*, pp. 174–179.

[3] V. John, K. Yoneda, Z. Liu, S. Member, and S. Mita, "Saliency Map Generation by the Convolutional Neural Network for Real-Time Traffic Light Detection Using Template Matching," *IEEE Trans. Comput. Imaging*, vol. 1, no. 3, pp. 159–173, 2015, doi: 10.1109/TCI.2015.2480006.

[4] A. Saleem, A. Al Maashri, L. Khriji, and M. Hussein, "An Integration Framework for UGV Outdoor Navigation System Based on LiDAR and Vision Data," *2015 16th Int. Conf. Res. Educ. Mechatronics*, pp. 16–21, doi: 10.1109/REM.2015.7380369.

[5] A. Novitsky and D. Yukhimets, "The navigation method of wheeled mobile robot based on data fusion obtained from onboard sensors and camera," in *ICCAS 2015 - 2015 15th International Conference on Control, Automation and Systems, Proceedings*, 2015, no. Iccas, pp. 574–579, doi: 10.1109/ICCAS.2015.7364984.

[6] B. Kazemipur, Z. Syed, J. Georgy, and N. El-sheimy, "Vision-based Context and Height Estimation for 3D Indoor Location," *2014 IEEE/ION Position, Locat. Navig. Symp. - PLANS 2014*, pp. 1336–1342, doi: 10.1109/PLANS.2014.6851508.

[7] S. S. Kamarudin, "Assessment on UAV Onboard Positioning in Ground Control Point Establishment," *2016 IEEE 12th Int. Colloq. Signal Process. Its Appl.*, no. March, pp. 210–215, 2016, doi: 10.1109/CSPA.2016.7515833.

[8] A. Spears, A. Howard, M. West, and T. Collins, "Sonar and Video Fusion for Vehicle Trajectory Estimation in Under-ice Environments," *Ocean. 2015 - MTS/IEEE Washingt.*, pp. 1–8, 2015, doi: 10.23919/OCEANS.2015.7401879.

[9] K. Mcguire, G. De Croon, C. De Wagter, K. Tuyls, and H. Kappen, "Efficient Optical Flow

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

and Stereo Vision for Velocity Estimation and Obstacle Avoidance on an Autonomous Pocket Drone," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 1070–1076, 2017, doi: 10.1109/LRA.2017.2658940.

[10] C. Cigla and L. Matthies, "Onboard Stereo Vision for Drone Pursuit or Sense and Avoid," *2018 IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Work.*, pp. 738–7388, 2018, doi: 10.1109/CVPRW.2018.00105.

[11] R. Huang, P. Tan, and B. M. Chen, "Monocular Vision-Based Autonomous Navigation System on a Toy Quadcopter in Unknown Environments," *2015 Int. Conf. Unmanned Aircr. Syst.*, pp. 1260–1269, 2015, doi: 10.1109/ICUAS.2015.7152419.

[12] S. Tijmons, G. C. H. E. De Croon, B. D. W. Remes, C. De Wagter, and M. Mulder, "Obstacle Avoidance Strategy using Onboard Stereo Vision on a Flapping Wing MAV," vol. 33, no. 4, pp. 858–874, 2017.

[13] C. Yuan, K. A. Ghamry, Z. Liu, Y. Zhang, and S. Member, "Unmanned Aerial Vehicle Based Forest Fire Monitoring and Detection Using Image Processing Technique," *2016 IEEE Chinese Guid. Navig. Control Conf.*, pp. 1870–1875, 2016, doi: 10.1109/CGNCC.2016.7829074.

[14] A. M. Boronahin, Y. V. Filatov, A. V. Gorelaya, B. D. Kodatskiy, V. A. Makarov, and V. Y. Venediktov, "Investigation of incoming airflow influence on the image stability in an optical system of relative objects position determination," *Proc. 2016 IEEE North West Russ. Sect. Young Res. Electr. Electron. Eng. Conf. EIConRusNW 2016*, pp. 156–159, 2016, doi: 10.1109/EIConRusNW.2016.7448143.

[15] M. B. Ekinci, "A Guidance Algorithm for Imaging with a LEO Satellite," *2019 9th Int. Conf. Recent Adv. Sp. Technol.*, pp. 729–733.

[16] M. Beul, N. Krombach, Y. Zhong, D. Droeschel, M. Nieuwenhuisen, and S. Behnke, "A High-performance MAV for Autonomous Navigation in Complex 3D Environments," *2015 Int. Conf. Unmanned Aircr. Syst.*, pp. 1241–1250, 2015, doi: 10.1109/ICUAS.2015.7152417.

[17] Y. Xu and H. Wang, "Image Processing Based Surgical Navigation System Building R ( xp R ( xp -lnPr ( I ( p ) I hist ' bkg ') -lnPr ( I ( p ) I hist ' ob }') Pr ( I ( p ) I hist ' ob }') Pr ( I ( p ) I hist ' bkg ') AR ( X ) + B ( X )," *2014 IEEE Work. Electron. Comput. Appl.*, pp. 991–994, 2014, doi: 10.1109/IWECA.2014.6845789.

[18] A. Dawson-elli, M. Potter, A. Bensch, and C. A. Linte, "An Integrated ' Plug & Play ' 3D Slicer Module for Image-guided Navigation for Training , Simulation and Guidance," *2014 IEEE West. New York Image Signal Process. Work.*, pp. 23–26, 2014, doi: 10.1109/WNYIPW.2014.6999479.

[19] X. Ma, X. Xia, Z. Zhang, G. Wang, and H. Qian, "Star image processing of SINS/CNS integrated navigation system based on 1DWF under high dynamic conditions," *Proc. IEEE/ION Position, Locat. Navig. Symp. PLANS 2016*, pp. 514–518, 2016, doi: 10.1109/PLANS.2016.7479740.

[20] T. Zeng *et al.*, "Multiangle BSAR Imaging Based on BeiDou-2 Navigation Satellite System : Experiments and Preliminary Results," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 10, pp. 5760–5773, 2015, doi: 10.1109/TGRS.2015.2430312.

[21] F. M. Santa, E. J. Gomez, and H. M. Ariza, "Global Path Planning for Mobile Robots using Image Skeletonization," *Indian J. Sci. Technol.*, vol. 10, no. 14, pp. 1–6, 2017, doi: 10.17485/ijst/2017/v10i14/112175.

[22] F. Prieto, "Morphological Process in Images Introduction," vol. 1, p. 84, 2014.