# PROGRAM FOR DETERMINATION OF THE MOVEMENT TRAJECTORY OF A LOCOMOTOR ROBOT THROUGH THE VORONOI DIAGRAM

**Max Suell Dutra**
Universidade Federal do Rio de Janeiro, COPPE/PEM, C.P. 68503 – CEP. 21945-970, Rio de Janeiro, RJ, Brazil
E-mail: max@serv.com.ufrj.br

**Armando Carlos de Pina Filho**
Universidade Federal do Rio de Janeiro, COPPE/PEM, C.P. 68503 – CEP. 21945-970, Rio de Janeiro, RJ, Brazil
E-mail: pina-filho@bol.com.br

**Aloísio Carlos de Pina**
Universidade Federal do Rio de Janeiro, COPPE/PESC, C.P. 68511 – CEP. 21945-970, Rio de Janeiro, RJ, Brazil
E-mail: long17@bol.com.br

***Abstract:*** *The Voronoi diagram has as main application in robotics the determination of optimal trajectories for the movement of a robot in a determined environment. By means of a program developed using concepts of vectorial calculus and analytical geometry it becomes possible to analyze the distance between two points of the work space, considering the existing obstacles and verifying the best route to be followed by the robot. For particular environment, the program creates a database storing the information of the best route. The results found for the application of the program in determined layout had demonstrated the efficiency of the program. In relation to the difficulty to trace the Voronoi diagram, the most challenging issue is to determine its vertices and the limits of its edges, task that seems to become more difficult with the increase of the number and complexity of the obstacles. Based in the results, we come to the conclusion that the program based on the Voronoi diagram can be used as a tool to assist the control design of a locomotor, being this traditional control (for fixed obstacles) or adaptative control (for mobile obstacles).*

***Keywords:*** *Control, Voronoi Diagram, Locomotor Robot.*

## 1. Introduction

The objective of the work presented here is the development of an efficient and robust program to find the best path to be performed by a locomotor robot in determined environment. Through the Voronoi Diagram, the program determine the faster path between two points (start and goal). Such program could be used in the control of an autonomous robot in indoor environments, such as: offices and small plants. The main aspect to be considered in the work is the ability of the robot in walking across the environment turning aside itself of the existing obstacles. So that the robot can perform this task with efficiency is necessary to get a mapping of the place. According to Setalaphruk *et al.* (2002), this information can to either be obtained by active exploration of the environment, or be supplied by human.

Exist many works about active exploration and environment mapping, such as: Leonard and Durrant-Whyte (1991), Ersson and Hu (2001). These methods enable robot to gain knowledge about the environment, hopefully without need of human intervention. Such procedure is many times complex and with difficult implementation. Alternatively, the robot can use a map provided by human, but such map usually requires a lot of details, some of which are not easily accessible by human. A solution in this case would be the use of floor plans found in buildings. In our work the map of the environment is supplied to the robot previously. Such map is sufficiently simplified, contends few obstacles, since our greater motivation is the creation of a functional program to determine the best path to be made by the robot.

In more complex works, the obstacles are mobile (or changeable) or then unknown. In these cases we have an environment that isn't known previously, then becomes necessary to realize a update of information that consider the changes in the place, making a replanning of the trajectory that will be executed by the robot. An excellent work about this subject is presented by Koenig and Likhachev (2002). Other works related to the subject studied here are presented by Trovato (1990), Latombe (1991), and Hoff III *et al.* (1999).

## 2. Used geometric calculations

To construct the Voronoi diagram is necessary to perform a series of geometric calculations. In this section we present some of the mathematical concepts of vectorial calculus and analytical geometry (Steinbruch and Winterle, 1987) used in the development of the program. All the other used calculations are trivial and they do not need commentary.

### 2.1. Angle between two straight lines

Consider the straight lines $r_1$, that it passes for the point $A_1$ $(x_1, y_1)$ and has the direction of a vector $\vec{v}_1 = (a_1, b_1)$, and $r_2$, that it passes for the point $A_2$ $(x_2, y_2)$ and has the direction of a vector $\vec{v}_2 = (a_2, b_2)$ (see Fig. (1)). Is called angle between two straight lines $r_1$ and $r_2$ the minor angle between a director vector of $r_1$ and a director vector of $r_2$. Then, if θ is that angle, we have:

$$\cos\theta = \frac{|\vec{v}_1 . \vec{v}_2|}{|\vec{v}_1||\vec{v}_2|}, \text{ with } 0 \le \theta \le \pi/2 \qquad (1)$$

or, in coordinates:

$$\cos\theta = \frac{|a_1a_2 + b_1b_2|}{\sqrt{a_1^2 + b_1^2}\sqrt{a_2^2 + b_2^2}} \qquad (2)$$

In the Fig. (1), the angle α is the supplement of θ, therefore, cos α = –cos θ. The angle α is the angle formed by $-\vec{v}_1$ and $\vec{v}_2$ or $\vec{v}_1$ and $-\vec{v}_2$.

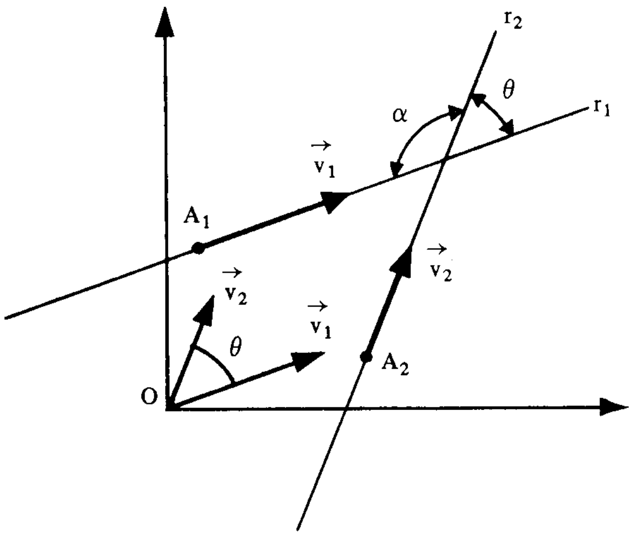

Figure 1. Angle between two straight lines (Steinbruch and Winterle, 1987).

## 2.2. Parallelism condition of two straight lines

The parallelism condition of two straight lines $r_1$ and $r_2$ is the same one of the vectors $\vec{v}_1 = (a_1, b_1)$ and $\vec{v}_2 = (a_2, b_2)$, that define the directions of these straight lines, that is:

$$\vec{v}_1 = m\vec{v}_2 \qquad (3)$$

or:

$$\frac{a_1}{a_2} = \frac{b_1}{b_2} \qquad (4)$$

Consider a straight line $r_1$, that it passes for a point $A_1 (x_1, y_1)$ and has the direction of a vector $\vec{v}_1 = (a_1, b_1)$, express for the equation:

$$\frac{x - x_1}{a_1} = \frac{y - y_1}{b_1} \qquad (5)$$

Any straight line $r_2$, parallel to the straight line $r_1$, has director parameters $a_2$, $b_2$, proportional to the director parameters $a_1$, $b_1$, of $r_1$. In particular, $a_1$, $b_1$, are director parameters of any parallel straight line to the straight line $r_1$. In these conditions, if $A_2 (x_2, y_2)$ is a any point of the space, the equation of the parallel to the straight line $r_1$, that it passes for $A_2$, is:

$$\frac{x - x_2}{a_1} = \frac{y - y_2}{b_1} \tag{6}$$

If the straight lines $r_1$ and $r_2$ will be express, respectively, for the reduced equations:

$$r_1:\ y = m_1 x + n_1 \tag{7}$$

and

$$r_2:\ y = m_2 x + n_2\ , \tag{8}$$

whose directions are given, respectively for the vectors $\vec{v}_1$ = (1, $m_1$) and $\vec{v}_2$ = (1, $m_2$), the parallelism condition allows to write:

$$\frac{1}{1} = \frac{m_1}{m_2} \tag{9}$$

or:

$$m_1 = m_2 \tag{10}$$

**2.3. Perpendicular condition of two straight lines**

Perpendicular condition of two straight lines $r_1$ and $r_2$ is the same one of the vectors $\vec{v}_1$ = ($a_1$, $b_1$) and $\vec{v}_2$ = ($a_2$, $b_2$), that define the directions of these straight lines, that is:

$$\vec{v}_1 \,.\, \vec{v}_2 = 0 \tag{11}$$

or:

$$a_1 a_2 + b_1 b_2 = 0 \tag{12}$$

**2.4. Distance between two points**

The distance d between the points $P_1$ ($x_1$, $y_1$) and $P_2$ ($x_2$, $y_2$) is the modulus of vector $\overrightarrow{P_1P_2}$ , that is:

$$d(P_1, P_2) = |\,\overrightarrow{P_1P_2}\,| \tag{13}$$

and, therefore:

$$d(P_1, P_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{14}$$

**2.5. Distance of a point to a straight line**

Consider the straight line r defined for a point $P_1$ ($x_1$, $y_1$) and by director vector $\vec{v}$ = (a, b) and consider $P_0$ ($x_0$, $y_0$) a any point of the space. The vectors $\vec{v}$ and $\overrightarrow{P_1P_0}$ determine a parallelogram whose height corresponds a distance d of the $P_0$ to r that we intend to calculate (see Fig. (2)).

The area of the parallelogram is given by the product of the base for the height:

$$A = |\vec{v}|\, d \tag{15}$$

or, in accordance with the geometric interpretation of the modulus of the vectorial product, for:

$$A = |\vec{v} \times \overrightarrow{P_1P_0}\,| \tag{16}$$

Equaling the Eq. (15) and (16), we have:

$$|\vec{v}|\, d = |\vec{v} \times \overrightarrow{P_1P_0}| \tag{17}$$

and

$$d = d(P_0, r) = \frac{|\vec{v} \times \overrightarrow{P_1P_0}|}{|\vec{v}|} \tag{18}$$

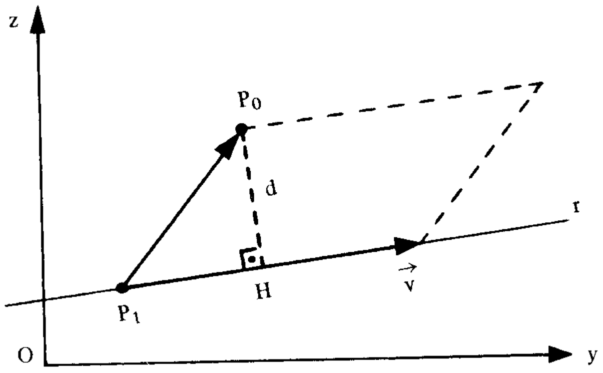

Figure 2. Distance of a point to a straight line (Steinbruch and Winterle, 1987).

## 3. Development of the program

The program was developed using the Delphi language (Borland Delphi 4.0, Copyright 1983, 1998, Inprise Corporation). In relation to the structure of data, all the data of the Voronoi diagram are stored in text format thus that they are calculated. These data can recorded and be consulted in any text editor.

By means of the program developed using concepts of vectorial calculus and analytical geometry it becomes possible to analyze the distance between two points of the work space, considering the existing obstacles and verifying the best route to be followed by the robot. For particular environment, the program creates a database storing the information of the best route.

To specify the called "edges of Voronoi", the program uses geometric calculations of mediatrix and bisectrix. In principle, the edges of Voronoi are calculated between the external limits of the environment. To follow, for each edge of each one of the obstacles located in the environment, the existing edges of Voronoi between the edges of the obstacle and the limits of the environment are calculated, as well as it between the edges of each obstacle and the visible edges of the other obstacles. In certain points, the program uses parabolas to conclude the construction of the Voronoi diagram. This is necessary so that in any point of the path, the distances between the edges of Voronoi and the limits of the environment and/or visible obstacles are equal. The vertices of Voronoi are characterized by the intersection of two or more edges of Voronoi. Table (1) presents the algorithm to compute an edge of Voronoi. Table (2) presents the algorithm to determine the Voronoi diagram.

Table 1. Algorithm to compute an edge of Voronoi.

```
Input: Two segments
Output: An edge of Voronoi

procedure ComputeEdgeVoronoi(segment1, segment2);
begin
  // Compute the boundaries of the bisectrix
  // that will be part of the edge
  Bisectrix(segment1, segment2);

  // Compute the boundaries of the parabolas
  // traced at each end of the bisectrix
  Parabola(segment1, segment2);

  // Compute the boundaries of the mediatrices
  // traced at the end of the parabolas
  Mediatrix(segment1, segment2);

  Report edge of Voronoi;
end;
```

Table 2. Algorithm to determine the Voronoi diagram.

```
Input: A set of obstacles and the definitions of the
       environment´s borders (upper, lower, left and right)
Output: A Voronoi diagram

procedure Voronoi(obstacles, borders);
begin
  // Compute the edges of Voronoi between the borders of the
  // environment
  for each pair of borders B1 and B2 do
    ComputeEdgeVoronoi(B1, B2);

  for each obstacle Obst do
  begin
    for each edge E of Obst do
    begin
      // Compute the edges of Voronoi between the obstacle´s
      // edge and the borders of the environment
      for each border B do
        ComputeEdgeVoronoi(E, B);

      // Compute the edges of Voronoi between the obstacle´s
      // edge and the edges of the other obstacles
      for each obstacle Obst2 do
        for each edge E2 of Obst2 do
          ComputeEdgeVoronoi(E, E2);
    end;
  end;

  Report Voronoi diagram;
end;
```

## 4. Application Example

The program for determination of the movement trajectory of the robot needs in principle of the two-dimensional mapping of the region where the robot will go to operate. Such mapping must include the existing obstacles in the place, which are represented by polygons of diverse forms. We define an environment as shown in the Fig. (3).

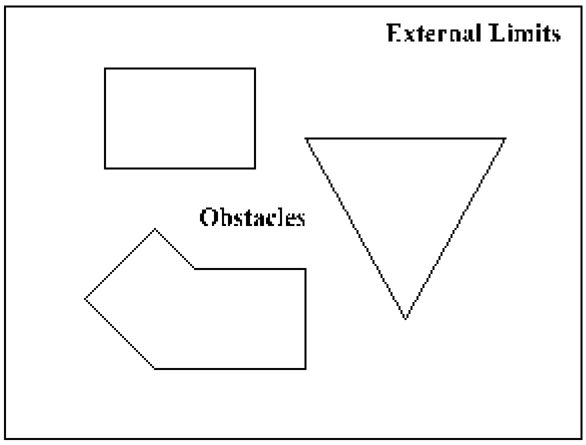

Figure 3. Environment where the robot will go to operate.

After to define the environment, the program supplies the Voronoi diagram, that is characterized by lines that denote the best paths to be traversed, avoiding the obstacles and keeping a determined distance between the limits and edges (see Fig. (4)).

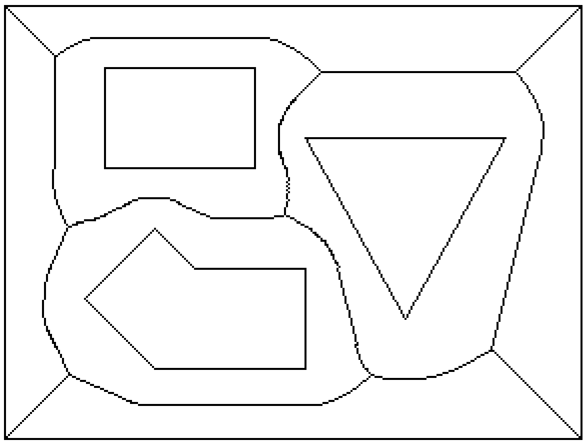

Figure 4. Voronoi diagram.

With the Voronoi diagram we can define two points in the environment, being one start and another goal (see Fig. (5)), evaluating the best path to be made by the robot (Fig. (6)).

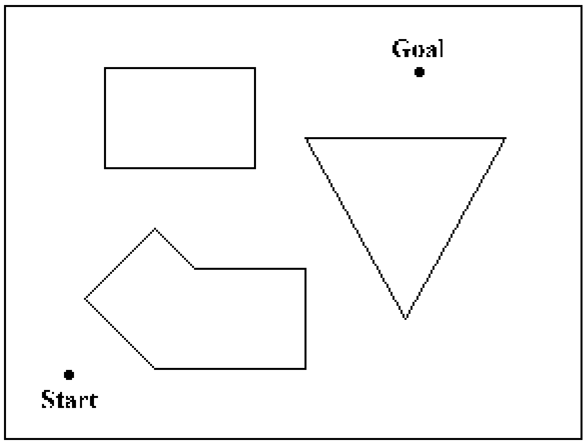

Figure 5. Environment with the start and goal points of the path.

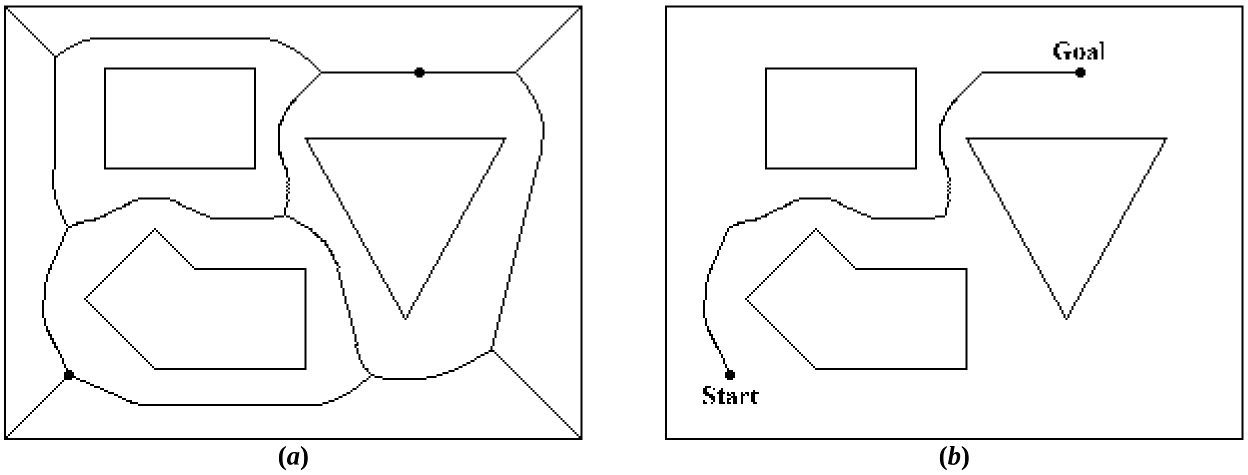

Figure 6. Voronoi diagram with the start and goal points (*a*) and better path to be made by the robot (*b*).

Figure (6-*b*) was constructed from the Fig. (6-*a*), supplied by the program, to emphasize the best path to be performed by the robot. The points without continuity are vertices of Voronoi, that are characterized by the intersection of two or more edges of Voronoi.

It is interesting to notice that a small change in the localization of one of the points can modify the choice of the path. We present in the Fig. (7) the environment with the same start point, but with the goal point a little displaced for right. In this case the best path to be made by the robot finishes if modifying (see Fig. (8)).

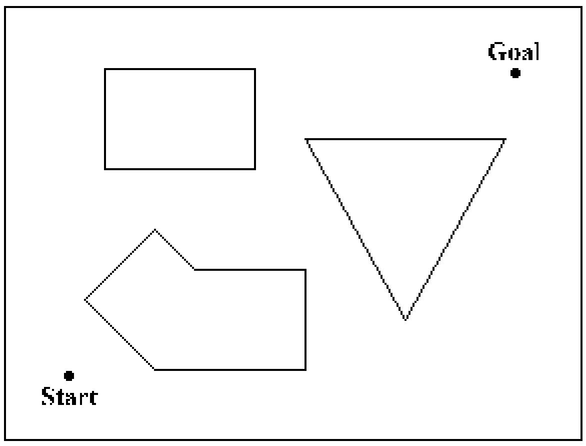

Figure 7. Environment with the goal point a little displaced for right.

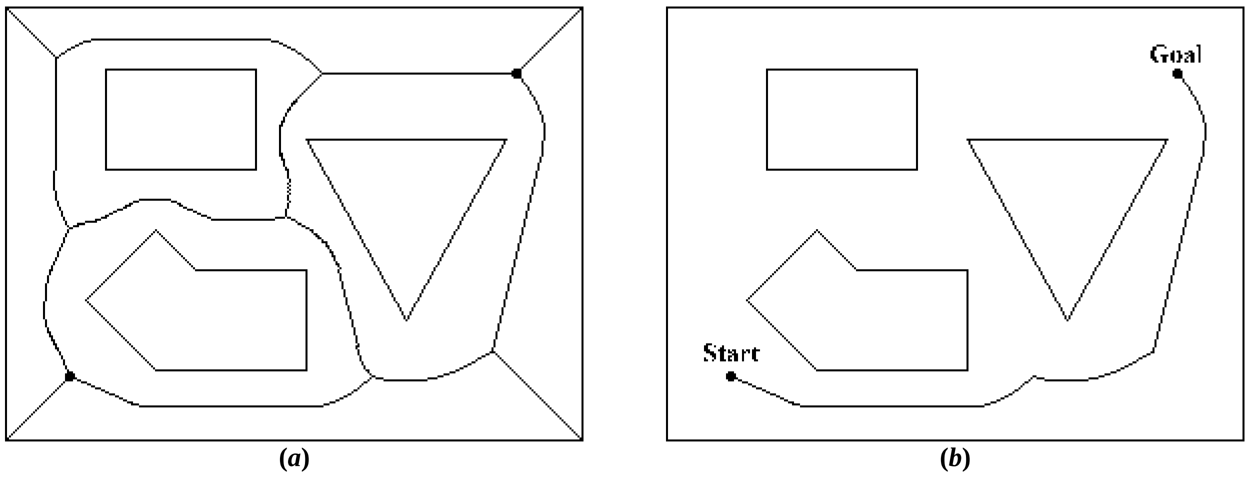

Figure 8. Voronoi diagram with the start and goal points (*a*) and better path to be made by the robot (*b*).

Similarly to Fig. (6), Fig. (8-*b*) was constructed from the Fig. (8-*a*), supplied by the program, to emphasize the best path to be performed by the robot. The movement trajectories can be executed by the most of the mobile robots.

## 5. Conclusion

In this work, we create a program that can be used in robot navigation, from a given simplified map of a determined environment. The simplified map of the environment, which can be easily created, is obviously useful as an easy way to give information to the robot, enable it to navigate efficiently across the environment without having to spend time wandering around learning the map. In the application example we use a simple environment, but it is possible to define a more complex environment with walls, corridors, rooms, doors and other obstacles, characterizing offices or small plants.

The results found for the application of the program in determined layout had demonstrated the efficiency of the program. In relation to the difficulty to trace the Voronoi diagram, the most challenging issue is to determine its vertices and the limits of its edges, task that seems to become more difficult with the increase of the number and complexity of the obstacles. Based in the results, we come to the conclusion that the program based on the Voronoi diagram can be used as a tool to assist the control design of a locomotor, being this traditional control (for fixed obstacles) or adaptative control (for mobile obstacles).

## 6. Acknowledgments

The authors would like to express their gratitude to CNPq and CAPES for the financial support provided during the course of this present research.

## 7. References

Ersson, T., Hu, X., 2001, “Path planning and navigation of mobile robots in unknown environments”, In: Proceedings of the International Conference on Intelligent Robots and Systems.

Hoff III, K. E., Keyser, J., Lin, M., Manocha, D., Culver, T., 1999, “Fast computation of generalized Voronoi diagrams using graphics hardware”, In: Proceedings of the 26th annual conference on Computer graphics and interactive techniques, pp. 277-286.

Koenig, S., Likhachev, M., 2002, “Improved Fast Replanning for Robot Navigation in Unknown Terrain”, College of Computing Georgia Institute of Technology, Atlanta, GA.

Latombe, J.-C., 1991, “Robot Motion Planning”, Kluwer Academic Publishers, Norwell, MA.

Leonard, J., Durrant-Whyte, H., 1991, “Simultaneous map building and localization for an autonomous mobile robot”, In: IEEE Int. Workshop on Intelligent Robots and Systems, pp. 1442-1447.

Setalaphruk, V., Uneno, T., Kono, Y., Kidode, M., 2002, “Topological Map Generation from Simplified Map for Mobile Robot Navigation”, In: 16th Annual Conference of Japanese Society for Artificial Intelligence.

Steinbruch, A., Winterle, P., 1987, “Geometria Analítica”, Second edition, McGraw-Hill, São Paulo, Brazil.

Trovato, K., 1990, “Differential A*: An adaptive search method illustrated with robot path planning for moving obstacles and goals, and an uncertain environment”, Journal of Pattern Recognition and Artificial Intelligence, 4(2).