2D Tracking Control Algorithms

POPESCU MARIUS-CONSTANTIN, PETRIŞOR ANCA
Faculty of Electromechanical Engineering
University of Craiova
Bd. Decebal, Nr.107, 200440, Craiova
ROMANIA

Abstract: - The paper deals with the problem of control algorithms design and implementation, necessary to position a working tool in a fix point or to track a determined pattern. In this paper is described the way to accomplished the electrical and mechanical part of a working tool and also the way to design and implement a control algorithm necessary to optimize 2D trajectory. Finally the proposed control algorithm is validated through computer simulations.

Key-Words: - Tracking algorithm, optimized algorithm, transformation, translation, rotation, simulations.

1 Introduction

There are many approaches of smart algorithms for 2D trajectories in the specialty literature [1], [2], [3], [4].

This paper considers the problem of trajectory tracking control design for a working tool. A solution to the problem of controlling this working tool is proposed, based on the development of a control algorithm.

In contrast with existing algorithms, shortly presented in the paper, this algorithm is optimized for tracking systems control.

2 Physical structure of the system

The experimental stall structure for positional adjustment on X-Y axes is presented in Fig.1.

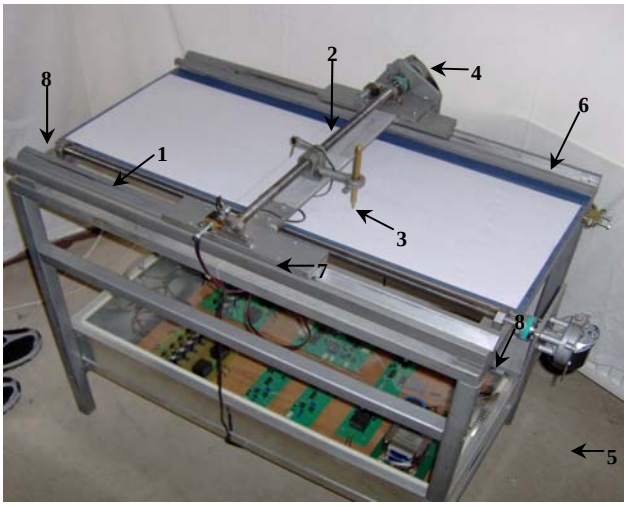

Fig.1 Experimental stall general view:
1-screw-nut assembly; 2 – filleted rod; 3 – working tool; 4, 5 – D.C. motors; 6 – fixed table, 7 - nut, 8 – piece.

3 Two-dimensional graphical transformations

Two-dimensional graphical transformations allow drawings representation at desired scale, performing the operations of images decrease and particularization, etc.

Translation.[5] Translation is the transformation through which an object is moved from its position, with an given distance. From mathematical point of view, the translation is specified by a vector, $v = tx \cdot I + ty \cdot J$. If (x, y) are the coordinates of a point *P* of an object *O*, then by the object translation with a distance equal with the value of the vector *v*, on the vector direction, the point *P* is transformed in $P'(x', y')$ (Fig. 2), where x' and y' are thus defined:

$$\begin{cases} x' = x + tx \\ y' = y + ty \end{cases} \quad (1)$$

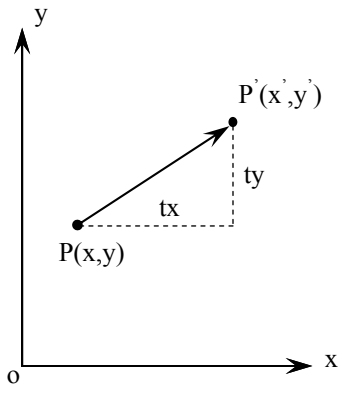

Fig.2 Explicative figure regarding translation

Scaling relative to the origin. [5] Scaling the point $P(x,y)$ relative to the origin, with the factors *sx*, *sy*, means scaling the position vector $OP(x,y)$ which join the origin with the point P. The resulted scaling vector OP' has the components x', y', where

$$\begin{cases} x' = x * sx \\ y' = y * sy \end{cases} \quad (2)$$

Rotation relative to the origin. [5] Let be $P(x,y)$ a point and *u* the rotation angle. The calculation of the point $P'(x',y')$, obtained by the rotation of the point *P* arround the origin with the angle *u*, is easier if is considered the relation between cartesian and polare coordinates of the point *P*, respective those of the point *P'*

$$\begin{aligned} x &= r * \cos(t) \\ y &= r * \sin(t) \end{aligned} \quad (3)$$

$$\begin{aligned} x' &= r * \cos(t+u) \\ y' &= r * \sin(t+u) \end{aligned} \quad (4)$$

where *r* is the length of the position vector OP, and *t* is its angle with the horizontal axis.

Homogeneous coordinates. The upper presented elementary transformations are expressed in homogeneous coordinates.

Translation:

$$\begin{bmatrix} x' & y' & 1 \end{bmatrix} = \begin{bmatrix} x & y & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ tx & ty & 1 \end{bmatrix} \quad (5)$$

Scaling relative to the origin:

$$\begin{bmatrix} x' & y' & 1 \end{bmatrix} = \begin{bmatrix} x & y & 1 \end{bmatrix} \begin{bmatrix} sx & 0 & 0 \\ 0 & sy & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (6)$$

Rotation relative to the origin:

$$\begin{bmatrix} x' & y' & 1 \end{bmatrix} = \begin{bmatrix} x & y & 1 \end{bmatrix} \begin{bmatrix} \cos(u) & \sin(u) & 0 \\ -\sin(u) & \cos(u) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (7)$$

Let be $F(xf,yf)$ the fix point of the transformation, *u* the rotation angle and *sx, sy* the scaling factors. The composition of these transformations is thus expressed:

Scaling relative to the point F:

$$\begin{bmatrix} x' & y' & 1 \end{bmatrix} = \begin{bmatrix} x & y & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -xf & -yf & 1 \end{bmatrix} \begin{bmatrix} sx & 0 & 0 \\ 0 & sy & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ xf & yf & 1 \end{bmatrix} \quad (8)$$

Rotation relative to the point F:

$$\begin{bmatrix} x' & y' & 1 \end{bmatrix} = \begin{bmatrix} x & y & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -xf & -yf & 1 \end{bmatrix} \begin{bmatrix} \cos(u) & \sin(u) & 0 \\ -\sin(u) & \cos(u) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ xf & yf & 1 \end{bmatrix} \quad (9)$$

Other transformations.

Reflexion relative to x axis (Fig. 3.a):

$$\begin{aligned} x' &= x \\ y' &= -y \end{aligned} \quad \text{or} \quad \begin{bmatrix} x' & y' & 1 \end{bmatrix} = \begin{bmatrix} x & y & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (10)$$

Reflexion *relative* to *y* axis (Fig.3.b):

$$\begin{aligned} x' &= -x \\ y' &= y \end{aligned} \quad \text{or} \quad \begin{bmatrix} x' & y' & 1 \end{bmatrix} = \begin{bmatrix} x & y & 1 \end{bmatrix} \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (11)$$

Reflexion *relative* to the *origin* (Fig.3.c):

$$\begin{aligned} x' &= -x \\ y' &= -y \end{aligned} \quad \text{or} \quad \begin{bmatrix} x' & y' & 1 \end{bmatrix} = \begin{bmatrix} x & y & 1 \end{bmatrix} \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (12)$$

Reflexion *relative* to the line *x=y* (Fig.3.d):

$$\begin{aligned} x' &= y \\ y' &= x \end{aligned} \quad \text{or} \quad \begin{bmatrix} x' & y' & 1 \end{bmatrix} = \begin{bmatrix} x & y & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (13)$$

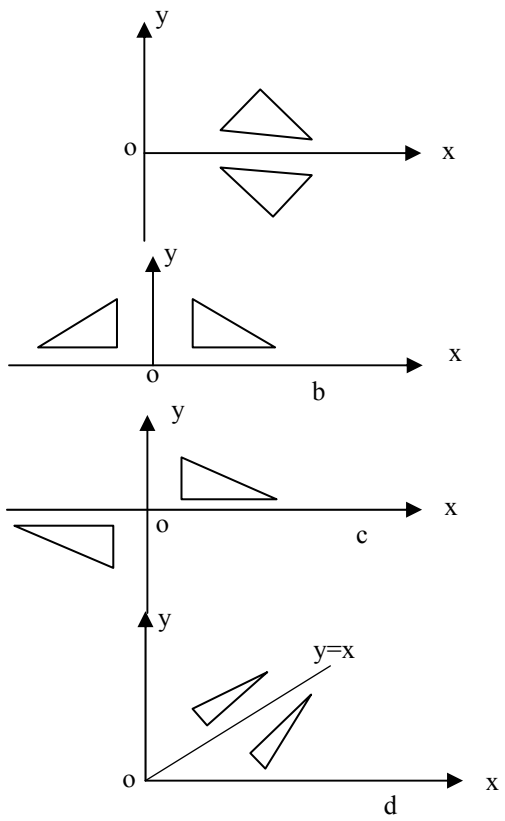

Fig.3 Explicative figure regarding other transformations

4 Fundamental algorithms for 2D trajectory synthesis

In the following are presented smart algorithms used for implement 2D trajectory systems.

The DDA algorithm (Digital Differential Analyzer). Let be the given segment $(x_1,y_1)-(x_2,y_2)$ and (x',y'), (x'',y'') two consecutive points on the segment. Then,

$$\frac{y''-y'}{x''-x'}=\frac{(y_2-y_1)}{(x_2-x_1)}=m \qquad (14)$$

For $abs(m)\le 1$ and $x_1<x_2$ the segment is generated by incrementing *x* with 1, so:

$$x''=x'+1\rightarrow x''-x'=1\rightarrow y''=y'+m \qquad (15)$$

For $abs(m)>1$ and $y_1<y_2$ the segment is generated by incrementing *y*, so:

$$y''=y'+1\rightarrow y''-y'=1\rightarrow x''=x'+1/m \qquad (16)$$

Bresenham algorithm. [6] This algorithm is also based on the incremental method, but it contains only operations with integer number. The algorithm is defined for vectors having the gradient between 0 and 1. For every value of x it is chosen that point of the discreet space which is closer to the point on the theoretic vector. The choice is based on the comparison of the distances from the two candidate points (the point above the vector and that under the vector) at the correspondent point on vector. [7]

Let be $m=\frac{(y_2-y_1)}{(x_2-x_1)}$ the vector gradient and (x_i,y_i) the last point of the discrete space selected in the vector generation process (Fig. 4).

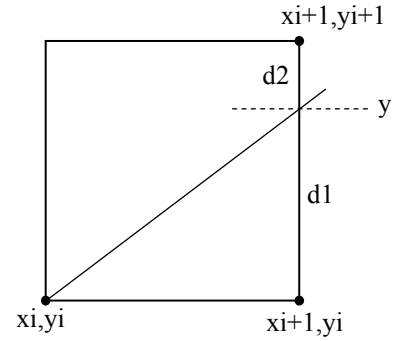

Fig.4 Explicative figure regarding Bresenham algorithm design

By noting with d_1 the distance from the theoretical vector at the point $O(x_i+1,y_i)$ and with d_2 the distance from the theoretical vector at the point $D(x_i+1,y_i+1)$, the following selected point will be *O* if $d_1<d_2$ and the point *D* in the other case.

The Pitteway-Castle algorithm. In Pitteway-Castle algorithm a vector generation is described by a shot of axial and diagonal movements. In the case of segments for which *u* and *v* are prime between them, the corresponding sequences are palindromes (they are the same when the reading is made from the left to right or from the right to left). The sequences of movements are generated by applying some productions as:

$R=T\#R^{-1}$, where # means the concatenation operator, and R^{-1} is the sequence resulted by inversion the sequence *R*.

Vector generation algorithm starting from micro vectors. This algorithm is especially used to create drawings at the plotters with pen.

5 The implementation of the algorithms library for 2D trajectory

The necessary software for this structure is divided in two distinct parts, but interdependent for functioning together. These parts are: the micro controller software and the application software.

The microcontroller software is a program written in an assembler language and applied in ATMEL microcontroller EEPROM memory; the registration is made with a special programmer.
The development environment LabWindows CVI 7.0 was used to implement the application software

Graphic user interface is presented in Fig. 5.

This interface contains two modules and a graphic on which the line or the circle having the established coordinates is plotted. The graphical interface used for drawing the lines with Bresenham algorithm is presented in Fig. 6.

It has two control buttons: OK and QUIT, as well as four references through which can be established the coordinates of the initial point (x_initial, y_initial), respectively the coordinates of the final points of the desired line to be drawn.
In fig. 7 is implemented an algorithm for plotting a circle having the coordinates X_CENTRE, Y_CENTRE and RADIUS established by the programmer.

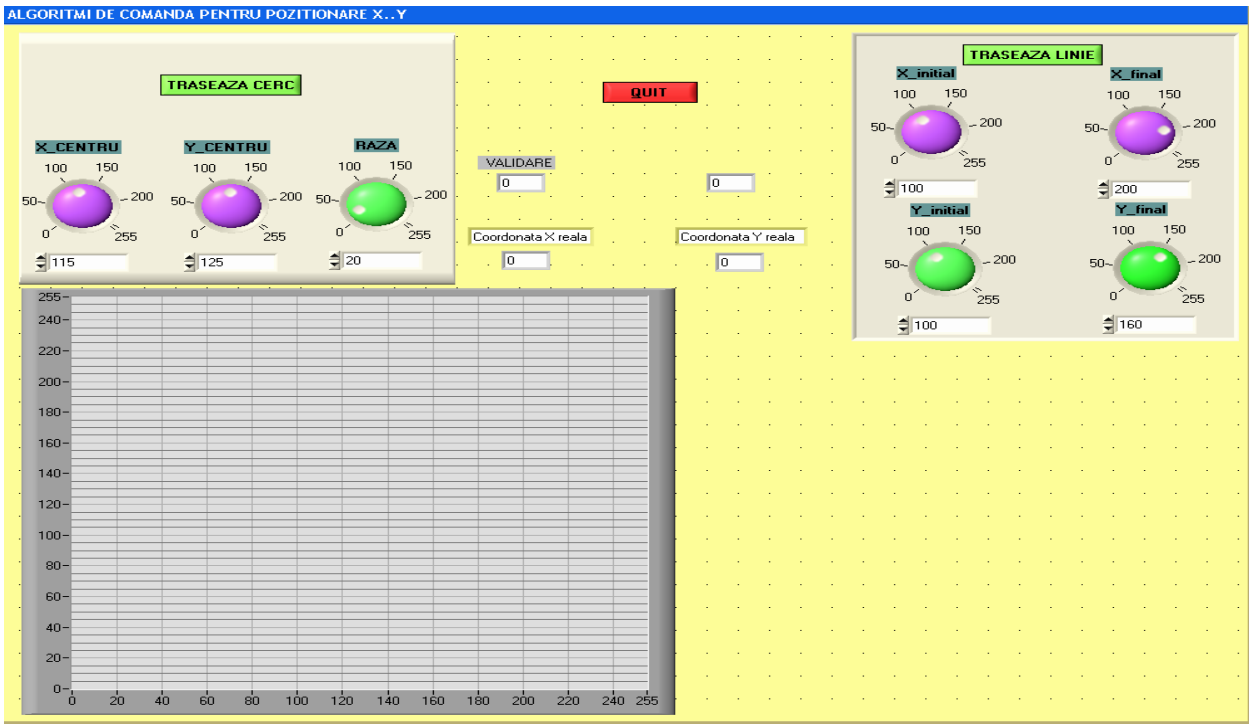

Fig.5 Graphic user interface

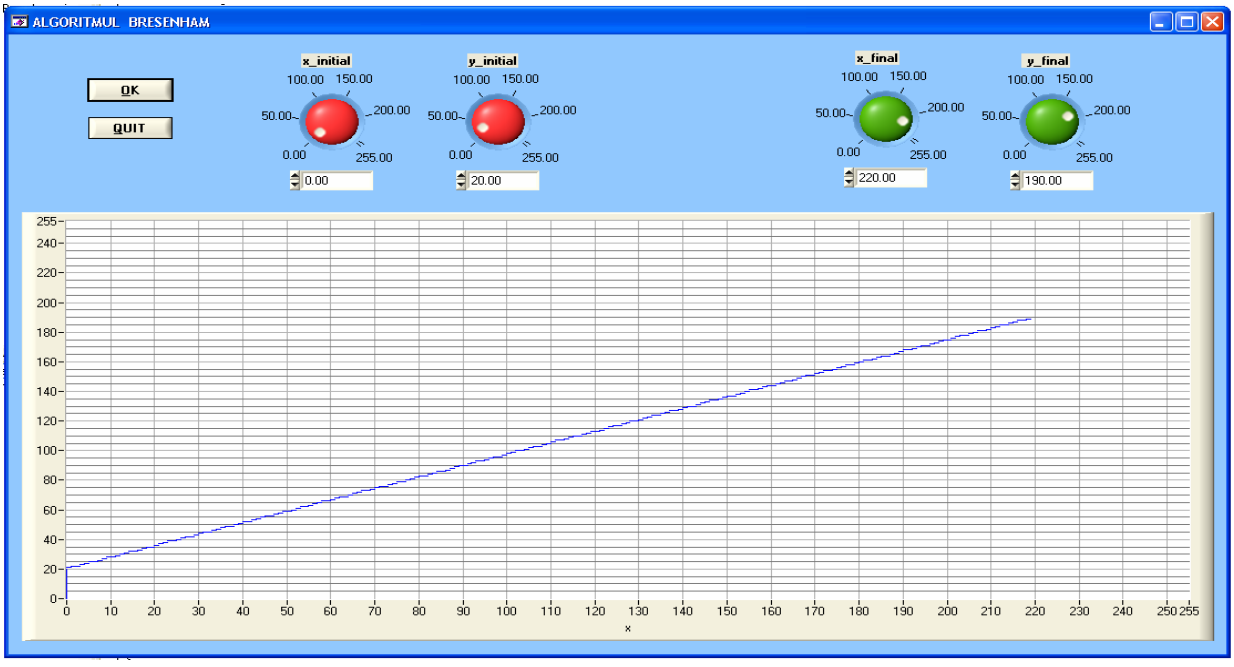

Fig.6 Graphic user interface for drawing lines with Bresenham algorithm

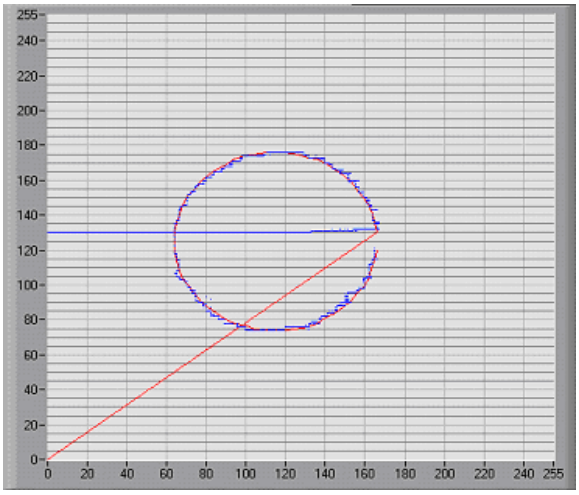

Fig.7 Drawing a circle

In Fig.8 is visualized the graphical interface for drawing other geometrical figures, with the difference that, as can be observed, there is moreover control respectively **Points number.**

Using this control can be establish the number of points (0..100). For instance, if a triangle is desired to be drawn then **points number** = 3; for a rhombus – **points number** = 4; for a hexagon – **points number** = 6 and so on.

Considering **points number** = 3, 4, 6, 8, in Fig. 6, on the interface graphic from Fig. 8 are drawn a triangle, a rhombus, a hexagon and respectively an octagon. and on the graphic from Fig.9.

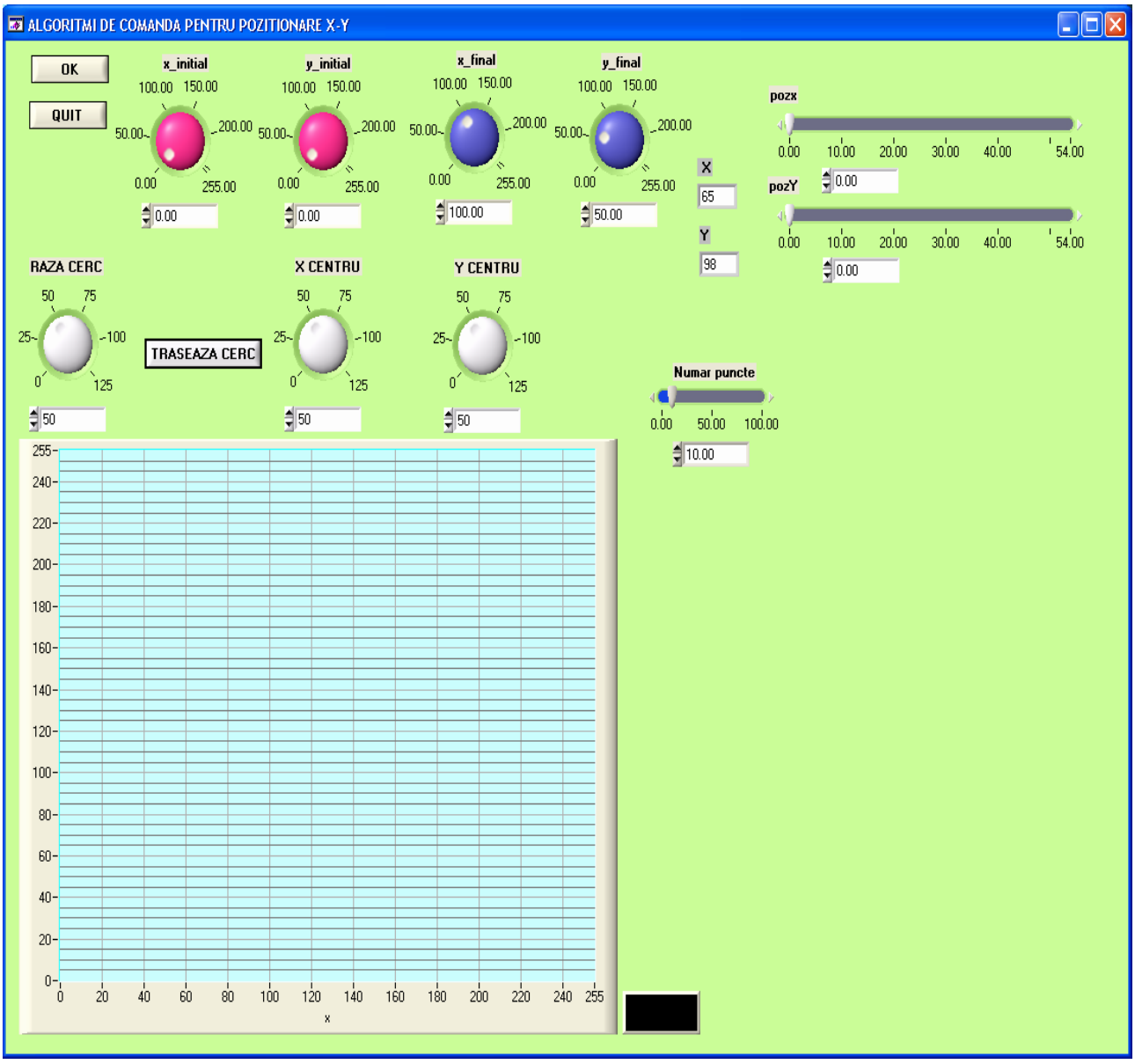

Fig.8 Graphical interface for drawing other geometrical figures

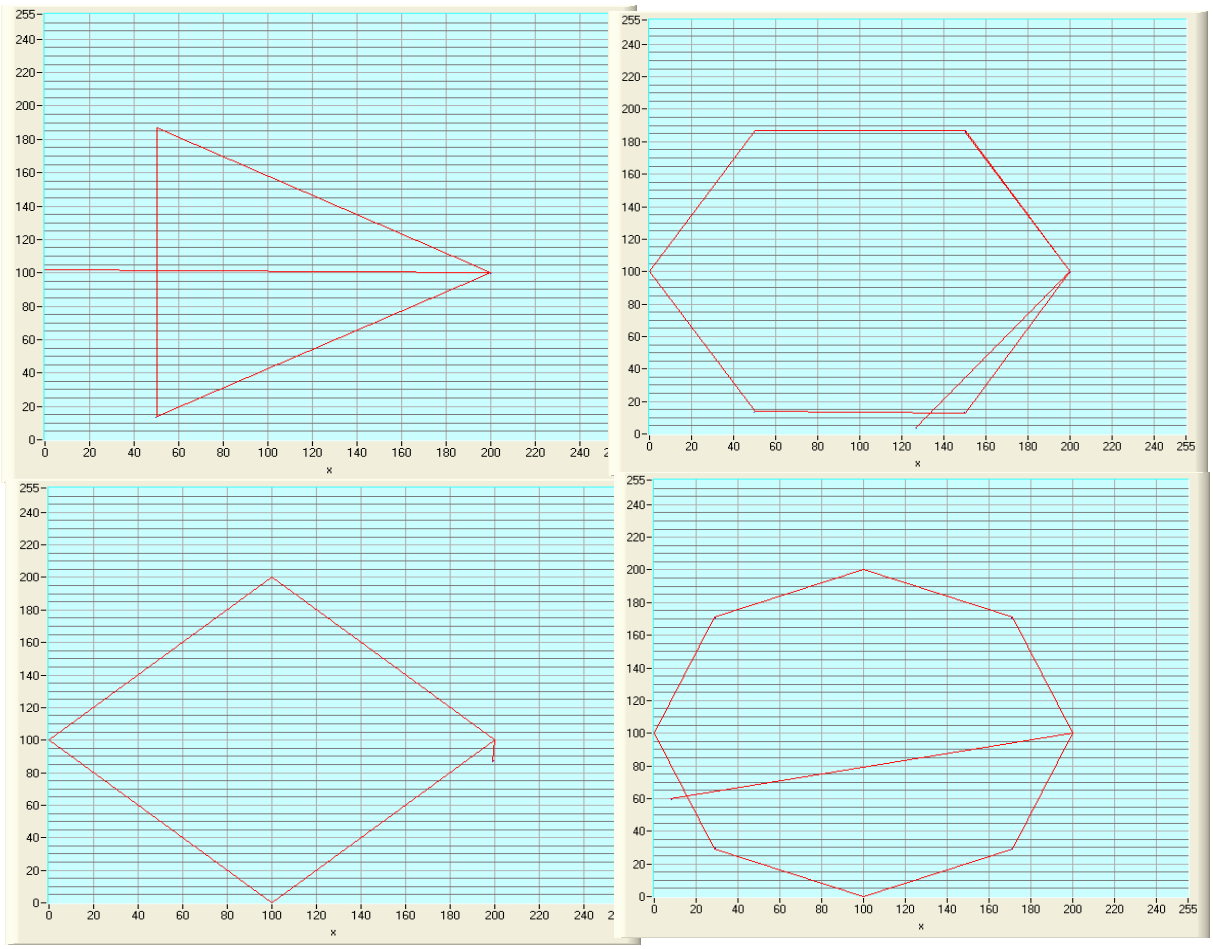

Fig.9 Example of geometrical figures (triangle, rhombus, hexagon and octagon)

6 Conclusion

The movement on the two axes of the support where is the working tool which must arrive in a selected point or to describe an establish way, is accomplished with two screw-nut systems which transform the rotation movement of two D.C. motors in a fixed support translation movement.

The acquisition system is of multichannel type. Data acquisition board assures the conversion on the two analogical channels and receives the following controls from the computer: data transmission from conversion; motors control to arrive in a reference position. For this, the microcontroller receives from the computer the new position reference and start motors control with the speed prescribed by the computer. When this reference position is touched, the microcontroller delivers this to the computer. The displacement sense of the motors is calculated based on the already existed position and based on the new position in which it must arrive.

Bresenham algorithms for drawing lines and an own algorithm for plotting a line are implemented. The difference between the other algorithms existed in the specialty literature and our algorithm consists in the fact that it is realized an optimal 2D control.

References:

[1] Bresenham J.E., *Algorithm for Computer Control of a Digital Plotter*, IBM Systems Journal, 4(1) pp.25-30, 1965.

[2] Foley J.D., Van Dam A.., Feiner S., and Hughes J.F., *Computer Graphics Principles and Practice*. Addison-Welsey, 1997.

[3] http://www.ecse.rpi.edu/Homepages/wrf/Research/Short_Notes/bresenham.html

[4] Freeman H., On the encoding of arbitrary geometric configurations, *IRE Trans.EC-102*, pp. 260-268, 1961.

[5] Coiffet P., *Modelling and Control*, Hermes Publishing, London, 1983, ISBN 1 85121 0008.

[6] Bresenham J.E, *Fundamental algorithms for computer graphics*, Springer-Verlag, 1985.

[7] Bresenham J.E., Earnshaw R.A., Forrest A.R., *Theoretical foundations of computer graphics and CAD*, Springer-Verlag, 1988.