1. INTRODUCTION

In this master's thesis, an evaluation/emulation system was developed for robot control. This system includes the real motor control part to provide a gantry motion and the virtual part where the virtual Cartesian plotter emulation software (as a test instrument) was developed to show the trajectory of the gantry motion. Because of these two parts, virtual and real parts, this system is named "Semi-Virtual System".

The virtual part that was mentioned above, was the virtual plotter software that emulates a pen plotter. In this part, instead of making effort to design a mechanical plotter, its job is emulated on a PC so that we can focus on the electronics and control system. This software shows the trajectory of the motor motion and also provides the up and down movements of the virtual pen. This virtual motion is named as a "half axis" motion that is the 0.5 in the 2.5 found in thesis title.

Real motor control part includes the two DC motors, encoders for feedback and the controller card that drives the motors to desired positions. In this control system, position control is provided with PID technique. This process is controlled by the microcontroller.

Beside these algorithms, a unique machine vision (MV) algorithm was developed in MATLAB. The MV algorithm takes the black and white drawings and converts them to a compatible format for the virtual plotter software, because virtual plotter software uses HPGL format like the real plotters. In this regard, raster based drawings are converted to the vectors. This process is called vectorization. In this machine vision software, a unique vectorization algorithm was developed. According to this algorithm, a dynamic

threshold value that varies according to the figures was used to eliminate unnecessary pixels. This dynamic structure makes this algorithm flexible and differs this algorithm from other vectorization algorithms.

Our goal in this thesis was to create a “test-bed” for students/researchers to develop new control algorithms. This test-bed and thesis will allow students to learn position control using the PID technique without any effort on mechanical side of things. By modifying controller code, students can develop more complex control algorithms. If they like, they may also modify the hardware and also use different motors, but they do not have to.

The system is controlled by a PC via serial port and powered by a battery. Complete system is shown in Figure 1-1.

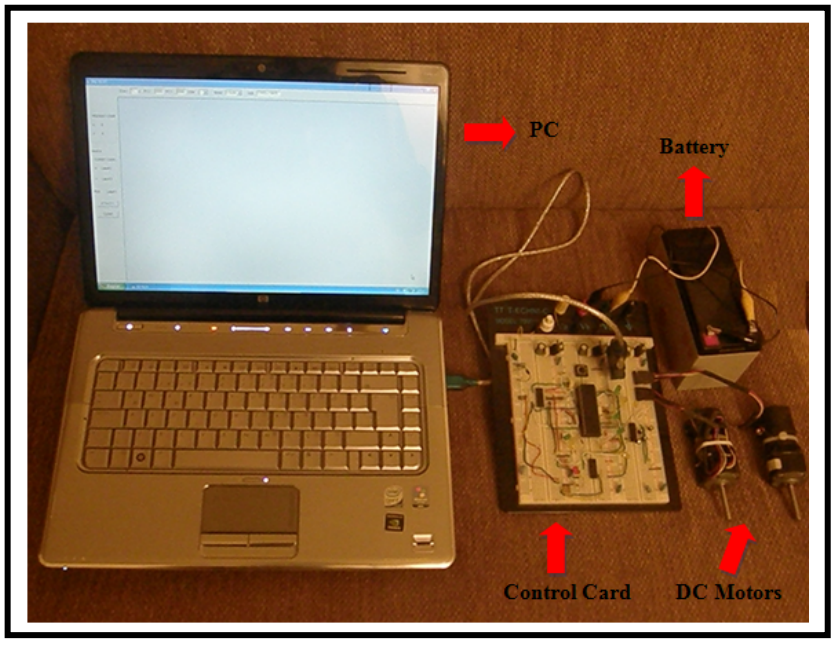


Figure 1-1: System View

The PC is used as a MV processor unit and also virtual plotter software screen. Control circuit is the unit that provides serial communication with computer, applies control

algorithm to the motors and determines the motor positions with the signals that come from motor encoders. These algorithms are shown in Figure 1-2.

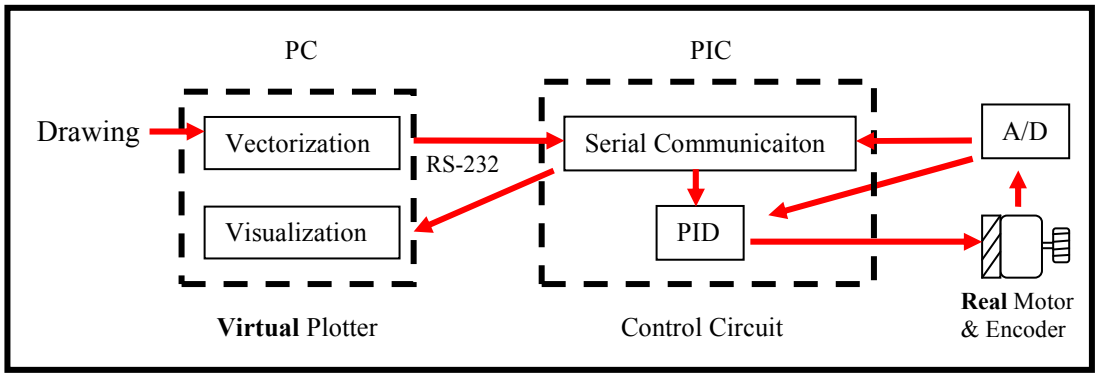


Figure 1-2: Top-Level Block Diagram and Where Algorithms Are Executed

First step is the tracking and recording all pixels of drawings. Then vectorization algorithm is applied to these pixels. Unnecessary pixels are eliminated according to algorithm and at the end of this algorithm, figures on drawings are expressed with the minimum number of pixels.

In second step, pixel coordinates that are obtained from first step, are sent to microcontroller via serial port to drive motors to these coordinates. During this process, PID algorithm is running and decides the motor speeds according to the position signals that come from encoders.

In third step, position pulses that come from encoder are sent to the PC via serial port. Virtual plotter software takes these pulses, determines the motor positions and visualizes the motor positions instantly on the Cartesian coordinate system. Second and the third steps run in order instantly.

2. PREVIOUS WORK

In this thesis, there are two main subjects that are found in the literature. One of them is the control system for a two-axis Cartesian gantry that is used for many purposes. Other one is the vectorization method that was studied in the past for engineering and architecture drawings.

Many vectorization methods have been developed and implemented since machine vision techniques were introduced more than 30 years ago. These methods use various algorithms and techniques.

2.1 VECTORIZATION ALGORITHMS

One of vectorization methods is the contour based. Contour based methods are the vectorization methods whose main idea is finding the shapes and edges of the lines and then calculating the middle points of the pair points on two parallel edges. In this method, edge detection algorithms are used heavily. (Jimenez and Navalon, 1982)

The most important problem of this method is processing the intersection points of the lines, especially at slanted lines. So it is not a effective algorithm to use in the images that contain a lot of intersections and slanted lines.

Thinning based vectorization method, which is also called “skeletonisation”, is used to convert thick lines to one pixel width lines. These algorithms make the image easier to operate on and analyze. However, it is possible that there are distortions during this process. To overcome these distortions and to make the algorithm efficient, a parallel

picture processing algorithm was developed. This method, which is based on deleting the pixels that do not belong to the skeleton by iterative algorithms, is more successful about protecting the intersection points of the lines. (Zhang and Suen, 1984).

Mesh pattern based methods have an idea that divides the whole image into meshes and looks for the black pixels (searching color) to determine the characteristics of the image. Using these characteristics, a control map of image is formed. During this process, some complex meshes are not operated and processed later. Finally, according to the control this map, the longest straight line is extracted. (Lin, Shimotsuji, Minoh, Sakai, 1985)

In this method, size of meshes are important for the process. It takes too much time for large meshes to process. On the other hand if a mesh is too small, it is difficult to detect it correctly.

In the following works, the mesh pattern algorithm is improved. Extended mesh pattern algorithm is designed for engineering problems. Dynamic mesh algorithm divides the meshes into smaller parts if it is necessary. Mesh pattern algorithms have more precision. (Vaxivere and Tombre, 1992)

Run-graph based methods scan the raster images among lines or columns. Then, these scan results are analyzed to produce a graphics structure. In line based images, middle points of the lines are determined. In region based images, nodes that connect the parallel edges are produced. Run graph based methods are successful at catching lines, getting data and it is easy to operate. But in some figures makes mistakes, because this method is sensitive for the noise and so it can cause distortions on the intersection points. (Boatto, 1992)

Orthogonal Zig Zag (OZZ) based methods are the vectorization methods whose basic idea is to track the virtual light beam that turns back orthogonally when it hits the edge of the area covered by the background colored pixels. The intersection point of the light beam and the area is recorded. If records are larger than the predefined threshold, the process stops and the last point is recorded. (Chai and Dori, 1992)

Hough Transform (HT) is a popular method, which is used in many vectorization algorithms. This algorithm is used to find the critical points of the line equation $y = m.x + n$. According to the intersection points of the lines and comparing the line slopes with the predefined threshold values, critical points of the lines are determined. Hough Transform based algorithms are successful for engineering documents and poor quality drawings etc., but the algorithm is too slow especially for circular arcs. It is an important problem for many applications. A lot of work was done to overcome this problem and to make this algorithm more efficient. (Yan, Takeshi, Tomoharu, 1993)

HT can be used to detect lines in noisy images. This is the good sight of this method. Since peak points of the figures are formed in the plane they may not be as precise as the original lines. Hence, the quality of lines detected is far less precise for slanted lines. which is produced by an implementation of the HT-based vectorization method.

Sparse Pixel Vectorization (SPV) is designed by the development of Orthogonal Zig Zag method. SPV has a special procedure that finds a reliable starting point for the tracking, a junction recovery procedure and a general tracking procedure that scans pixels horizontally, vertically and slant. By the effects of these algorithms SPV algorithm becomes faster than OZZ algorithm. (Liu, Dori, 1996)

Song, Su, Li, Cai (2002) designed a method that consists of two main algorithms. One of them is the seed-segment based line-network that is successful in converting the lines and line networks. Another algorithm is the knowledge supported vectorization algorithm, which eliminates the noise and degradation completely. It is especially appropriate for engineering and architecture drawings.

Koçer and Yildiz (2006) developed a vectorization method and named this method as "Full Automatic Vectorization by Using Derivative Base Edge Detection". Their study was conducted to vectorize close range, rectified building images. By using the algorithm that they developed and called "Derivative base edge detection" they did not need an additional parameter which is taken from user. This is the important feature that

changes the operation parameters (threshold values) according to the figure and gives the name “Full Automatic” to study. Disadvantage of this method is the weak edge tracking algorithm. Except this weakness, this method is an effective method for vectorization of the less detailed images and can be used in topographic photogrammetry.

Although these works in the literature offer contributions on the vectorization process and they are partially successful in many cases, a perfect vectorization algorithm that solves this problem completely has not been developed yet.

Our algorithm that is developed in this thesis, determines the critical points of the figure with the dynamic threshold value that is based on the equation of the line $y = m.x+n$. According to this equation, slope of the angles between neighboring line segments are calculated and compared each other. As a result of this comparison, critical points of figure are saved and the others are deleted.

The most important feature of our vectorization algorithm is the dynamic threshold value that does not exist in almost all algorithms except Full Automatic Vectorization by Using Derivative Base Edge Detection Method. This dynamic structure changes the threshold value for the different parts of the figure to make algorithm flexible. Thus this algorithm does not need a parameter that is taken from user as a threshold. The pixel tracking method of our algorithm is similar to the HT based methods and both of these algorithms use the equation of the line $y = m.x+n$ for vectorization. But the junction recovery feature differs two algorithms and makes ours more successful. But the weak part of our algorithm against other algorithms is the absence of thinning algorithm. With thinning algorithm, our vectorization algorithm will be more powerful and can be used as a corner detection algorithm.

The vectorization methods that are explained so far, and our algorithm are compared with some basic criteria that exists in literature about vectorization in Table 2-1.

Method	Sub-process	Quality of Line Geometry	Line Width Process	Junction Recovery	Automatic Threshold
Contour Based	Edge detection Line tracking	Poor	Yes	No	No
Thinning Based	Medial axis point sampling, Line tracking	High	No	No	No
Mesh Based	Medial axis point sampling, Line tracking	Poor	Yes	Yes	No
Run-Graph Based	Run graph construction Line tracking	Poor	Yes	No	No
OZZ	Medial axis point sampling, Line tracking	Poor	Yes	Yes	No
HT	HT, Line tracking	Poor	No	Yes	No
Sparse Based	Medial axis point sampling, Line tracking	Good	Yes	Yes	No
Full Automatic Vectorization	Edge detection Line Tracking	Good	Yes	No	Yes
Our Algorithm	Line tracking, Corner detection	Good	No	Yes	Yes

Table 2-1: Comparison of Vectorization Algorithms

2.2 GANTRY CONTROL ALGORITHMS

Another subject of this thesis is the two axis Cartesian gantry control that is available to apply to many systems like pen plotters or cutters. In the literature, gantry control

studies are generally about overhead cranes, x-y tables and robots for laboratory experiments.

Yang and Red (1996) developed a method to control a real-time Cartesian motion along spatially complex curves such as Non-Uniform Rational B-splines (NURBS). The method dynamically creates a map that consists of the critical trajectory parameters between parameter space, Cartesian space, and joint space. This trajectory determines speed and accelerations of Cartesian tool for the motion of the gantry robots or the other mechanisms.

Huang, Lin and Chen (1998) optimized a fuzzy sliding-mode controller through real-coded genetic algorithms and this algorithm is applied to an industrial X-Y table. These algorithms are combined as Real-coded Genetic Algorithm based Fuzzy Slide Mode Control that is designed to compensate the nonlinear behaviors through repeated learning process.

Rieber and Taylor (2004) designed a Cartesian two axis mechanism that makes point to point motion. This mechanism is used in the circuit board assembly industry. H_∞ control method that synthesizes the controllers achieving robust performance or stabilization is applied to achieve three goals: compensate for the varying mass distribution, suppress structural bending vibrations and friction disturbances, and at the same time achieve a small motion settling time.

Ratcliffe, Hatönen, Lewin, Rogers, Harte, Owens, (2005) designed an iterative learning control (ILC) technique to make the gantry robot learn the repeated reference trajectories. This technique improves the performance of tracking control system especially in industry with the assistance of PID control algorithm by reducing repeating disturbances.

Many industrial applications require robust control techniques in order to obtain fast response and to improve the dynamic performances. Therefore Giam, Tan, Huang

(2007) used Sliding Mode Control (SMC) that is a robust tracking control method and have a variety applications area in automation and robotic, and Linear Quadratic Regulator (LQR) based PID algorithm to provide high precision for the gantry control.

Chen, Hsieh (2007) developed a control strategy of repetitive controller that eliminates the periodic tracking errors for a gantry type machinery with linear motors. Purpose of this method is increasing the stabilizing range and improving the robustness. A feed forward control algorithm is added to repetit
ive algorithm that cannot compensate the first period error to improve the performance and to enhance the tracking ability.

3. THE MACHINE VISION ALGORITHM

Our machine vision algorithm includes four main algorithms. First algorithm is the pixel detection algorithm that detects the searching pixels and records pixel coordinates in order. Second algorithm is the vectorization algorithm that eliminates the unnecessary pixel that will not be used in drawing part. Third algorithm is the error analysis algorithm that determines the errors in vectorization of figures. Last algorithm is HPGL translation algorithm that converts vectorization outputs to HPGL standards.

3.1 PIXEL DETECTION ALGORITHM

Pixel detection algorithm is designed for the black and white 256 color raster images to record pixel coordinates in order. The algorithm firstly determines the color (white or black) of the searching pixel and then applies the horizontal pixel scanning method. This scanning algorithm scans the image from left to the right and from right to the left to follow the searching pixels correctly. While the scanning algorithm is going on, when the first searching pixel is found, this pixel is considered as a central pixel and 8 neighboring pixels around it are examined to find a new searching pixel as it is shown in Figure 3-1. When a new pixel is found, algorithm looks for a new searching pixel. During this procedure, coordinate of every searching pixel is recorded and its color value is converted to the background color not to deal with it again as shown in Figure 3-2.

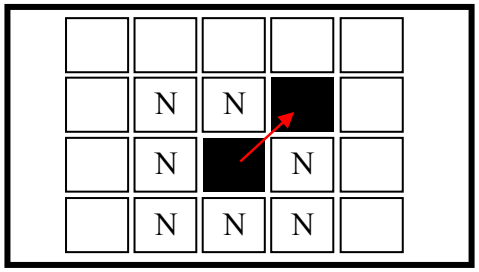


Figure 3-1: Neighboring Pixel Tracking

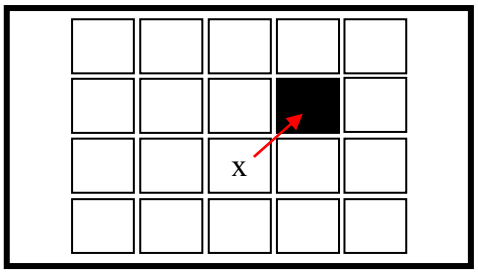


Figure 3-2: Deleting Color of Recorded Pixel

If the number of neighboring pixels is more than the threshold value (10 pixels), this procedure is going on until all pixels of that figure are located. When this procedure ends, horizontal scanning is activated again to find a new a pixel of a new figure on image. If neighboring pixels are less than threshold value, direction of horizontal scanning is changed, the process (detection of neighboring pixels and converting their color to background color) that is done for that figure is taken back and the these pixels are left for the scanning that that will be performed from opposite direction later.
If the scanning that performs from opposite direction cannot find more than 10 pixels for same figure too, it is decided that this figure is a little figure and its all pixels are recorded. After this exceptional situation, algorithm runs until all searching pixels are located and recorded. The block diagram of this algorithm is shown in the Figure 3-3. After this algorithm, all searching pixels are located and recorded in order as much as possible.

The next step is the distance calculation between the coordinates that were recorded before, to determine the number of different figures, their starting and ending points. At the end of these steps, all pixel coordinates are recorded in order and grouped according to positions of the figures on drawings.

The number of recorded pixel coordinates is too high to use for the plotter. So it is not a rational idea and it will be a time and energy wasting for a plotter to use all these pixels to draw figures. In order, it will take more time to complete drawings and this means more wearing for the motors and other mechanical parts. So that the number of pixel coordinates should be minimized while characteristics of figures are being kept. For this reason a unique vectorization algorithm is developed.

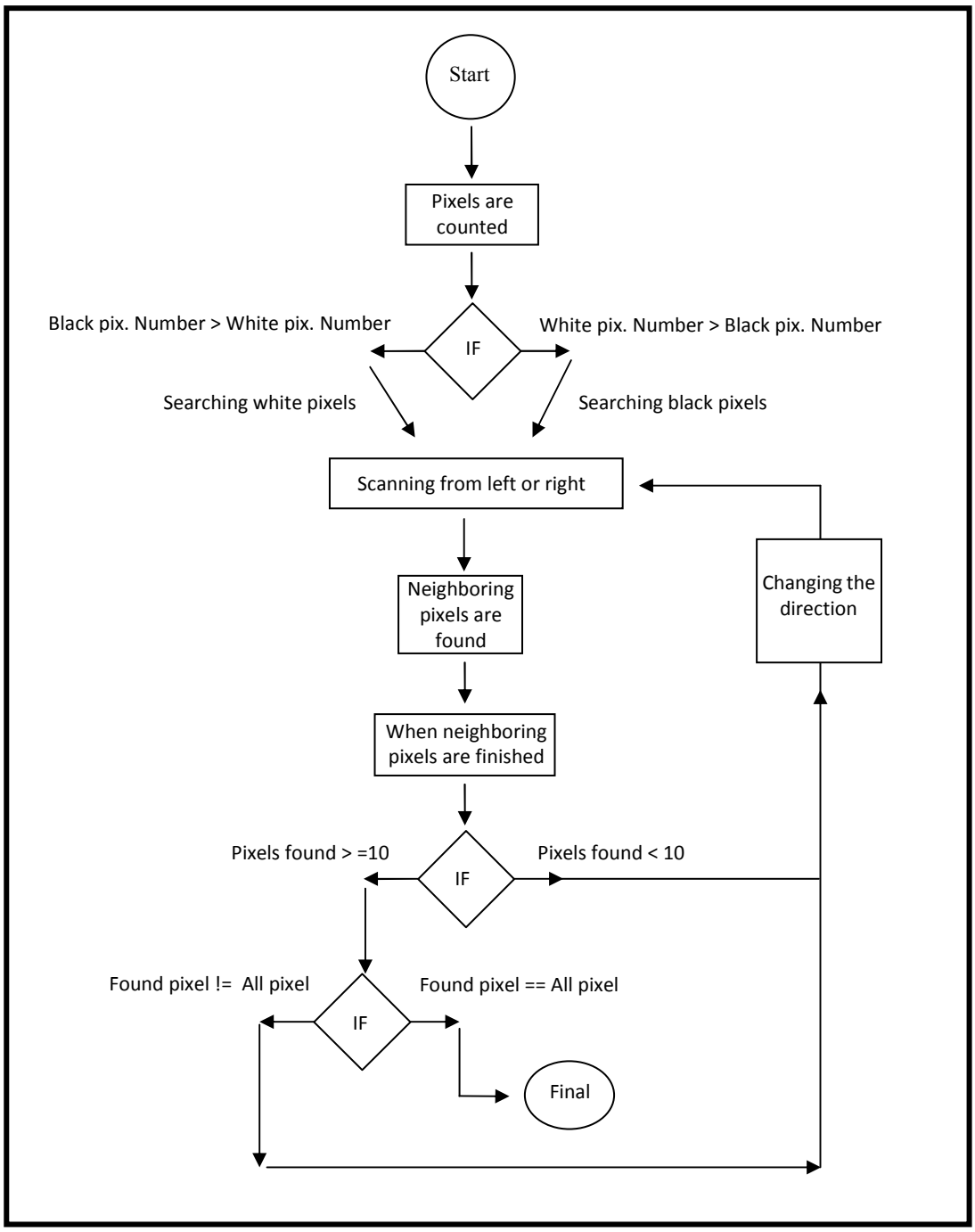


Figure 3-3: Pixel Detection Algorithm Blok Diagram

3.2 VECTORIZATION ALGORITHM

To draw a line, one needs to have at least two points or a point and the slope of the line. From this principal, previous 10^{th} pixel and following 10^{th} pixel of every pixel are thought to be tied with imaginary line segments. Angles between these neighboring line segments are calculated mathematically. This calculation is shown in Figure 3-4 where the point A and the point C are the 10^{th} previous and 10^{th} following pixels alternately and the point B is the checking pixel.

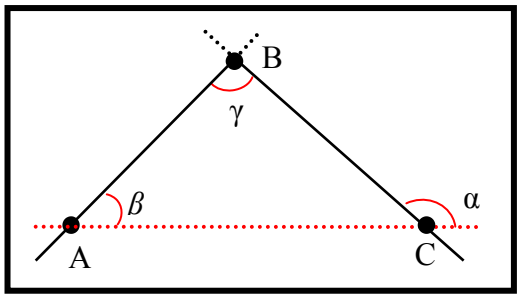


Figure 3-4: Pre-elimination Algorithm

$$\gamma = \alpha - \beta \tag{3.1}$$

$$tan\,\gamma = tan\,(\alpha\text{-}\beta) = \frac{tan\,\alpha - tan\,\beta}{1 + tan\,\alpha\,.\,tan\,\beta} \tag{3.2}$$

$$|m_{BC} - m_{AB}| = |tan(\alpha - \beta).(1 + m_{AB}.m_{BC})| \tag{3.3}$$

Then the checking pixel is controlled if it is necessary to record or not by comparing the slopes of line segments. If the absolute vale of difference between slopes of these line segments is smaller than the tolerance value, it is decided that, this pixel is on the same line with the pixels around as shown in Figure 3-5. In that case it is considered that, this pixel is not a necessary pixel and if this pixel is eliminated, figure will not be affected. According to the algorithm, this pixel is deleted from the database. On the other hand if the difference between the slopes of these line segments is greater than the tolerance, the algorithm records that pixel like the point B in the Figure 3-6. This algorithm is going on until the last pixel on drawing is processed.

Tolerance value that is mentioned above is the tangent of the angle γ. This tolerance value is the optimum value that is obtained after many experiments and it is 0.15 that is the tangent value of 8.5 °.

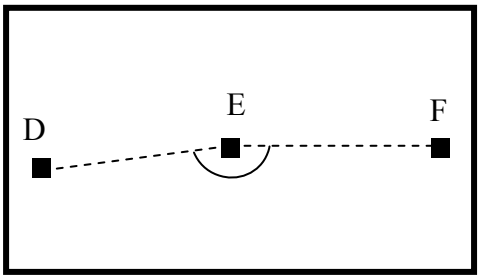


Figure 3-5: Linear Pixel Distribution

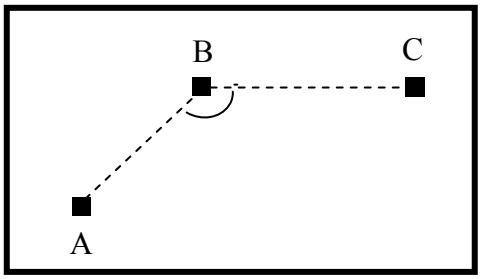


Figure 3-6: Polygonized Pixel Distribution

This algorithm that is explained above is planned as a pre-elimination algorithm. Although it eliminates so many unnecessary pixels, in some cases cannot be effective. Because it is clear that every pixels of a line segment may not be on the same line. When the image is zoomed in, it is seen that pixels are located like stairs as shown in Figure 3-7 and Figure 3-8. So comparing angles between neighboring line segments cannot always give the correct results. For example, when this algorithm is applied the drawings in Figure 3-7 and 3-8, there will be some unnecessary pixels in output drawings that are shown in the Figure 3-9 and 3-10.

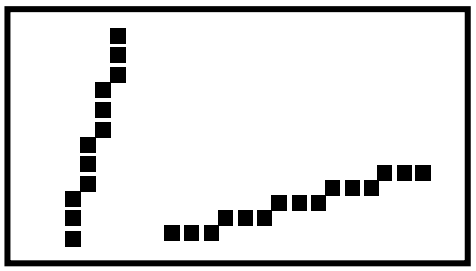


Figure 3-7: Drawing-1

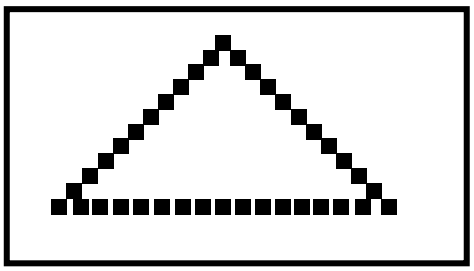


Figure3-8: Drawing-2

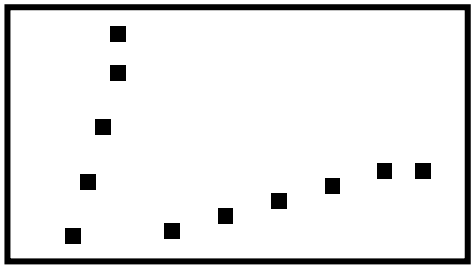


Figure 3-9: Drawing-1 Pre-elimination

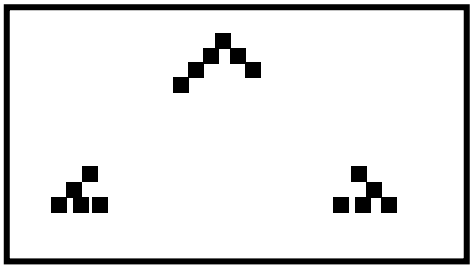


Figure 3-10: Drawing-2 Pre-elimination

Purpose of this vectorization algorithm is, to express the figures with the minimum pixels as possible as it can be. For example, a rectangle should be expressed with 4 corner points, triangle with 3 corner points, line segments with 2 points and curve figures should be expressed by the optimum number of pixels without losing originality. So a supplementary algorithm is designed.

In this algorithm, the first pixel of the figure is named base pixel. It is thought that there is an imaginary line segment between base pixel and checking pixel, and another line segment between the base pixel and the following pixel of the checking pixel as shown in the Figure 3-11. Algorithm calculates the angle geometrically between these two line segments and compares this value with the threshold value. If the angle is less than the tolerance value, checking pixel is eliminated. Otherwise if the angle is greater than threshold value, checking pixel is recorded and this pixel is named base pixel. This comparing method is applied for all the pixels that pass the first elimination.

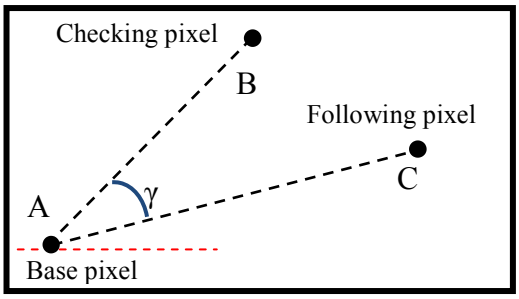


Figure 3-11: Vectorization Algorithm

In this algorithm threshold value is not fixed, it is a dynamic threshold and varies inversely proportional to the distance between base pixel and the checking pixel. This dynamic structure makes the algorithm more flexible. Because in some figures, the distance between the base pixel and checking pixel is too long. In this case if the threshold value was a fixed value, it would be impossible to detect the corner points.

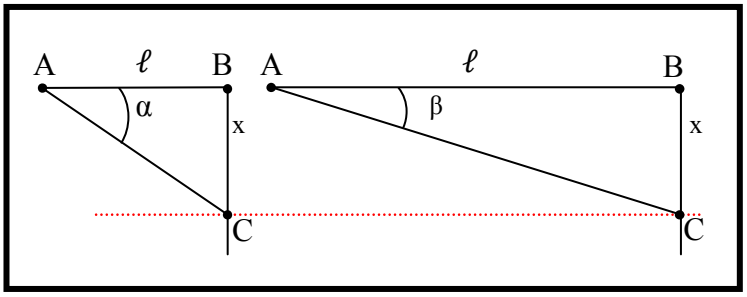


Figure 3-12: Dynamic Threshold Value

As shown in the Figure 3-12 if the length ℓ, between the points A and B, was longer, the difference between the slopes of the line segments AB and AC would decrease and this value would be smaller than the tolerance. It means the corner point B cannot be detected by algorithm although the length of the line segment BC is constant. So as it was mentioned previously a dynamic threshold is used. It is based on Equation 3.4.

$$m_{AB} - m_{AC} = \tan\gamma \,.\, \frac{(1+m_{AB}.m_{AC})}{\sqrt{(X_{base}-X_{following})^2+(Y_{base}-Y_{following})^2}} \,.\, C \qquad \textbf{(3.4)}$$

C is a constant value that was determined experimentally.

m_{AB} is the slope of the line segment AB.

m_{AC} is the slope of the line segment AC.

γ is the angle between the line segment AB and AC and it is 0.025 that is the tangent of the angle 1.43°.

$\sqrt{(X_{base} - X_{following})^2 + (Y_{base} - Y_{following})^2}$ is the length between the base pixel and the following pixel of the checking pixel.

After these steps, vectorization algorithm is completed and the sample results are shown in Figure 3-13 and 3-14.

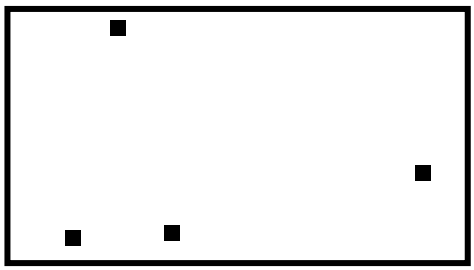


Figure 3-13: Drawing-1 Vectorization

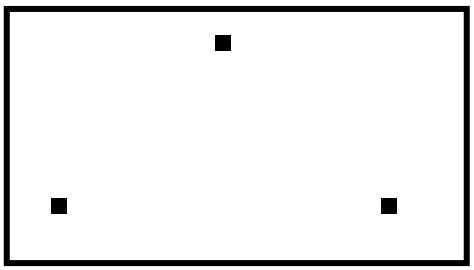


Figure 3-14: Drawing-2 Vectorization

3.3 HPGL TRANSLATION

In this part, HPGL and the HPGL Converter software that is developed in this thesis are introduced.

3.3.1 HPGL

HPGL is a common plotter language that was developed by Hewlett Packard. HPGL (Hewlett Packard Graphic Language) was the primary plotter control language for HP plotters and then it became a standard language for all plotters in time. This language includes groups of text data starts with two letter code and the varying parameters as shown in Table 3-1.

In time HPGL2 language was developed. This language has some new command and mew features like determining the line width and high resolution.

Command	Meaning
PU 230,300	Lift the Pen Up and move to the coordinate X: 230, Y:300
PD 100,400	Put the Pen Down and move to the coordinate X:100, Y:400
SP1	Select Pen 1
CI 10;	Draw a circle whose radius is 10 unit
SS;	Select the standard font

Table 3-1: HPGL Format

3.3.2 HPGL Converter Algorithm

After the vectorization algorithm, non-eliminated pixels are recorded in database. Then HPGL converter algorithm works and reads these coordinates and converts them into the HPGL compatible format. In this format there are two basic commands that lift the pen up and put the pen down on the paper are used. A short sample text file that is an output of this algorithm is shown below in Figure 3.15.

```
Command, X coordinate, Y coordinate
        PU,     02993, 13955
        PD,     02995, 15200
```

Figure 3-15: HPGL Converter Software

The purpose of this algorithm is to produce HPGL file that can be used in real plotters. So outputs of this algorithm were tested with F-PLOT software which displays HPGL files and prints HP-GL/2 plot files for plotters on. These results show that HPGL converter algorithm works successfully. It means that this vectorization algorithm and converter software is compatible for the real plotters.

3.4 ERROR ANALYSIS

Error analysis is the study of determining the errors between vectorization output and input raster data. This algorithm works after vectorization process and shows the success of vectorization. When the vectorization algorithm is applied to the arc that is in the Figure 3-16 and sample output is shown in the Figure 3-17. (Errors are magnified to be shown). According to the algorithm, these points are tied with imaginary line segments. These line segments and the original arc figure are both shown in the Figure 3.18.

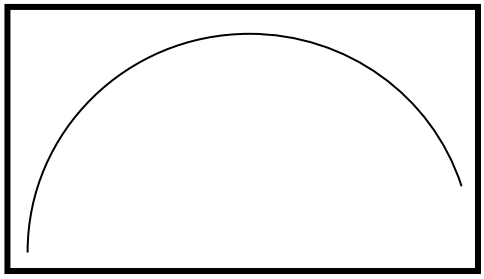


Figure 3-16: Vectorization Input

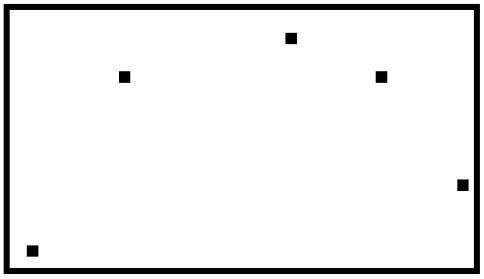


Figure 3-17: Vectorization Output

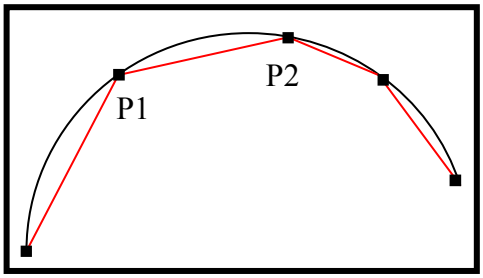


Figure 3-18: Linear Approach of Arc

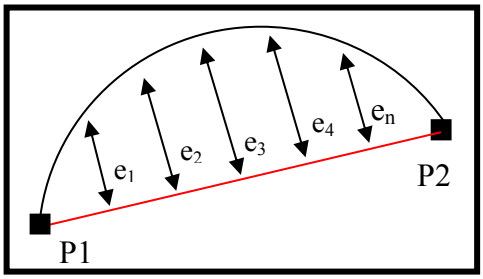


Figure 3-19: Errors between P1 and P2

The distances between the arc and the line segment are shown in the Figure 3-19. These distances are the errors of the algorithm and calculated geometrically with the Equation 3.5 Where A, B, C are the coefficients of the line equation and x, y are the pixel coordinates of the arc.

$$\text{error} = \frac{|Ax+By+C|}{\sqrt{A^{2+}+B^{2}}} \qquad \textbf{(3.5)}$$

After all errors are determined, average error, standard deviation of errors, and the maximum error value are calculated and the error distribution graph is plotted.

4. MOTOR CONTROL

Motor control is the second part of this thesis. In this part, DC motors, control systems, control algorithms and the other assistant algorithms are found.

4.1 DC MOTORS

Direct current motors are the motors that are used widely in industrial applications and at robotic applications. They have wide supply voltage and power range and different types for different purposes. The most important advantage of dc motor is that its torque can be increased by reducing the speed like gearboxes in automobiles. This advantage makes them more popular than the opponent motor type, alternative current (AC) motors. Torque-speed characteristics of DC and AC motors are shown in Figure 4-1 and 4-2. AC motors take place in industry in recent years by development of frequency converters. Low maintenance cost makes AC motors preferable.

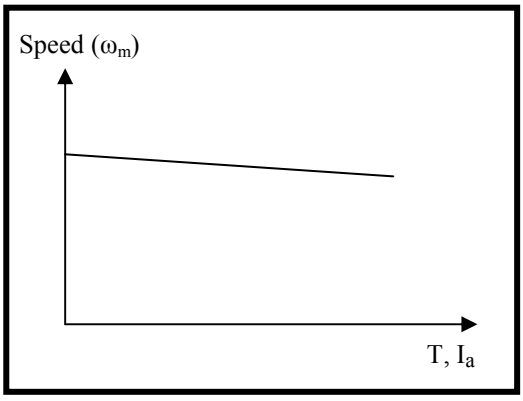


Figure 4-1: DC Motor Speed – Torque and Current Characteristics

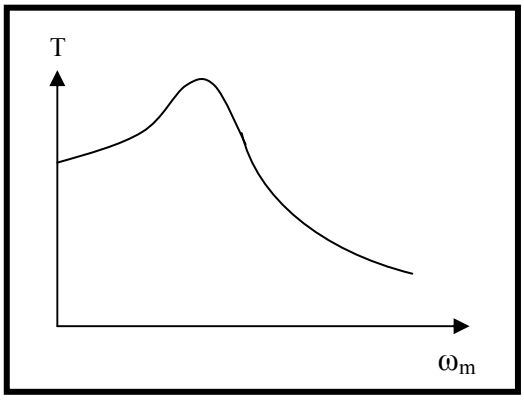


Figure 4-2: AC Motor Speed – Torque Characteristics

The equations for the DC motors are below:

$$T = K_t i_a \quad \textbf{(4.1)}$$

$$T = j\alpha + T_d + T_f \quad \textbf{(4.2)}$$

$$i_a = \frac{J\alpha}{K_t} + \frac{(T_d+T_f)}{K_t} \quad \textbf{(4.3)}$$

Also,

$$V = K_e\,\omega + L\,\frac{di_a}{dt} + R_a i_a \quad \textbf{(4.4)}$$

$$V = K_e\,\omega + L\,\frac{j}{K_t}\,\frac{d^2\omega}{dt^2} + \frac{R_a j}{K_t}\,\frac{d\omega}{dt} + R_a\,\frac{(T_d+T_f)}{K_t} \quad \textbf{(4.4a)}$$

The equation is arranged like this

$$K_e\,\omega + \frac{R_a j}{K_t}\,\frac{d\omega}{dt} + \frac{Lj}{K_t}\,\frac{d^2\omega}{dt^2} = V - R_a\,\frac{(T_d+T_f)}{K_t} \quad \textbf{(4.4b)}$$

is obtained.

T = Motor torque
j = Inertia moment
α = Angular Acceleration
T_d = Load moment
T_f = Friction torque
ω = Angular speed
K_e = Back EMF constant
K_t = Motor torque constant
R_a = Motor armature resistance
I_a = Armature current

This equation is a non homogenous linear differential equation that can be faced in electrical and mechanic events. If the right hand side of the Equation 4.4b was a constant term, second and third term of the left hand side would go to zero and voltage would be linear with the speed of the motor but it would not. Because in this control system, the voltage V is varying in time. T_d cannot be predicted and may be variable. Equation 4.5 can be written as a result of Equation 4.4b;

$$\omega(t) = \omega_p(t) + A.e^{-\lambda_1 t} + B.e^{-\lambda_2 t} \quad \textbf{(4.5)}$$

λ_1, λ_2 are assumed different, positive real numbers, $\omega_p(t)$ is a particular solution, A, B are the coefficients that are determined with boundary condition, $A.e^{-\lambda_1 t}$ *and* $B.e^{-\lambda_2 t}$ are the complementary solution that goes to zero in time t in practice (in theory t = ∞).

In Equation 4.5 first term from the right is a particular solution that depends on the right side of the Equation 4.4b. If it is assumed that all parameters are constant at right side of Equation 4.4b equation will be shown in Equation 4.5a.

$$\omega_p(t) = \frac{1}{Ke}\left(V - R_a \frac{(T_d + T_f)}{K_t}\right) \quad \textbf{(4.5a)}$$

At steady state, terms $A.e^{-\lambda_1 t} + B.e^{-\lambda_2 t}$ go to zero and there is a linear relationship as shown in Equation 4.5b.

$$\omega(t) = \omega_p(t) = \frac{1}{Ke}\left(V - R_a \frac{(T_d + T_f)}{K_t}\right) \quad \textbf{(4.5b)}$$

4.2 DC MOTOR SPEED CONTROL METHODS

In this system, DC motor position is controlled by changing the motor speed. It is clear that $\theta = \int_0^t \omega(t)$ and θ_x is the motion at x axis, θ_y is the motion at y axis. So that, the speed of motor should be controlled according to the position. DC motor speed is controlled by different methods. These methods are explained in following parts.

4.2.1 Armature Voltage Control

This speed control method is based on changing the armature voltage. The armature circuit resistance R_a and I_f are kept constant and armature voltage (terminal voltage) is varying to control the speed. As shown in the Equation 4.5b, speed increases when terminal voltage V increases. These relations are visualized in Figure 4-3.

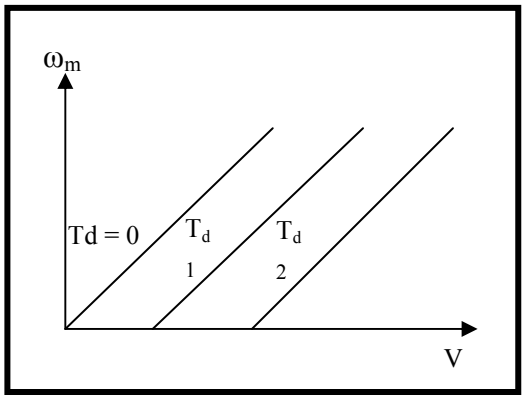
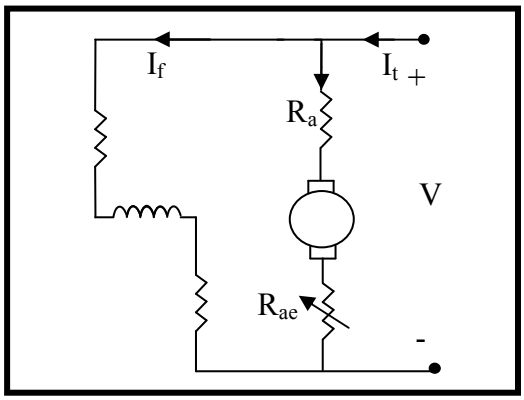


Figure 4-3: Speed - Voltage Characteristics

4.2.2 Armature Resistance Control

In this method, field current I_f and the armature terminal voltage V are holding constant at their rated values. Angular speed is controlled by changing resistance with a rheostat that is shown in Figure 4-5. The relationship between the rheostat resistance R_{ae} and the angular speed ω_m is shown in Figure 4-6.



Fi”gure 4-4: Armature Resistance Control Circuit Schema

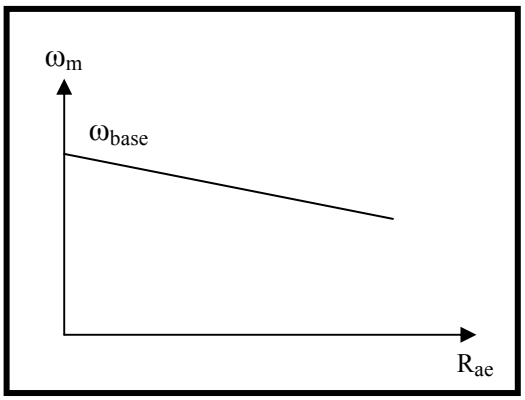


Figure 4-5: Speed - Armature Resistance Characteristic

4.2.3 Field Control

In this method, terminal voltage V and armature resistance R_a are fixed. Angular speed ω is controlled by changing the current I_f with a rheostat R_{fc} that is serial connected to circuit as show in Figure 4-7. The linear relationship between I_f and the speed ω_m is shown in Figure 4-8.

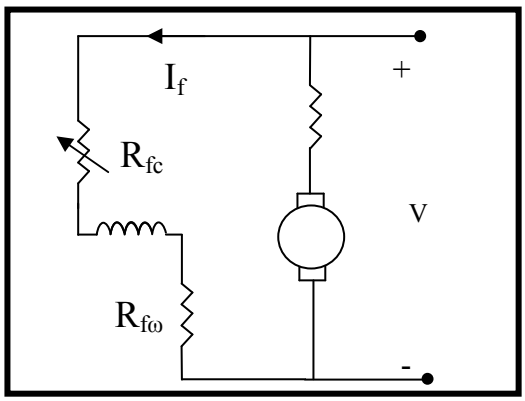


Figure 4-6: Field Control Circuit Schema

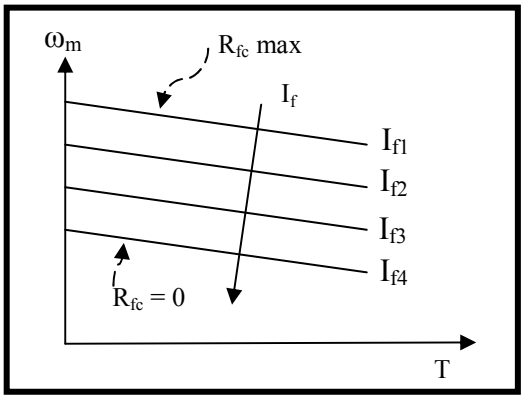


Figure 4-7: Speed – I_f and Torque Characteristic

In this system, armature voltage control method is used. Speed of motors ω_x and ω_y are controlled with the terminal voltage. It is clear that, speed of direct current motors is linear to the supply voltage if the voltage is constant for a while to get steady state. (Ra is neglected in Equation 4.5b)

$$V = K.n \tag{4.6}$$

$$V = K.\frac{1}{2.\pi}.\omega \tag{4.6a}$$

If an equation of a curve is given as y= f(x).

$$\frac{dy}{dx} = \frac{\frac{dy}{dt}}{\frac{dx}{dt}} = \frac{\vartheta_y}{\vartheta_x} \tag{4.7}$$

According to Equation 4.7 if an imaginary point is moving on the curve and its projection speed on x axis and y axis are represented ϑ_x and ϑ_y alternately.

$$\vartheta_{\mathrm{y}} = \frac{dy}{dx}\,\vartheta_{\mathrm{x}} \qquad \textbf{(4.7a)}$$

It was formulated in the Equation 4.7a the angular speed of motor y is $\frac{dy}{dx}$ times of the angular speed of motor x and curve that will be drawn y = f(x) will be obtained.
However in this control application, supply voltage is varying continuously and instantly so there is not enough time that is presented with t_0 to get steady state as shown in Figure 4-9. So it's impossible to express this relation with a simple linear equation. So a control method is needed to control this system.

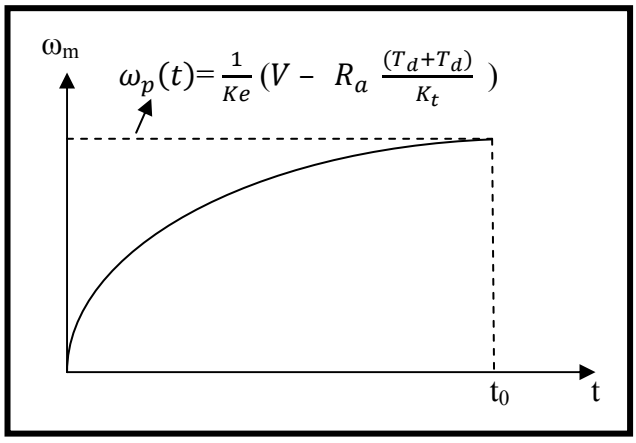


Figure 4-8: Change of Angular Speed in time t_o

4.3 CONTROL SYSTEMS

Automatic control systems are the systems that are intended to keep set values without human intervention. These systems were managed or regulated by mechanical or pneumatic mechanism in past but now by the development of technology, controllers become programmable microprocessor. These microprocessors make systems more flexible, stable.

In recent years automatic control systems are used at every area of human life. Especially in buildings air conditioning systems, at automobiles speed control systems, at industry, temperature, pressure, speed, humidity and chemical parameter control devices are examples of automatic control systems. These systems are all feedback systems that compare system outputs with set values to modify.

On-off control is an example of automatic control systems. It is a simple negative feedback control also cheap and effective when it is used on suitable systems. Because this system stops working when measuring value reaches the set value and waits until this value is out of the set limits. In addition on-of control system make the actuator work with full capacity or make them stop. Refrigerators, thermostats and most of the home type air condition system are on-off control.

On-off control is not an effective method for all control systems. For example motor control systems waste more power if motors stop and start up continuously. And also these start up and stop actions for several times damage the mechanical parts of the motor and cause problems. On the other hand on-off control systems are also not suitable for the precision necessary control systems like industrial control applications.

Another control system is the linear control system. Linear control systems use linear negative feedback to generate a control signal after mathematical calculations. Outputs of these systems are calculated with the difference between set values and the measuring values that is called error. According to the error, system works any capacity between full capacity and zero capacity. Consequently linear control systems are more complex than on-off control systems.

The proportional integral derivative (PID) controller is often mentioned as a ‘three-term’ linear controller. It is currently one of the most frequently used controllers in industry. In PID controller the control variable is generated from a term proportional to the error, a term which is the integral of the error, and a term which is the derivative of the error.

Proportional: Proportion term is obtained by the multiplying errors with the proportional gain K_p. This value has to be high because the actuator of system has to work immediately. Sometimes high proportional term may cause instable and a very low gain may cause the system to drift away.

Integral: Integral term is calculated by summing the multiplications of the errors and the integral gain K_i. This term always magnifies the errors to activate the actuator when error is too small. It increases the overshoot and accelerates the process.

Derivative: The derivative term is calculated by the multiplying the difference between the error and the previous error with the derivative gain K_d. Derivative term produces a corrective signal that makes output close to the set point by comparing the present error with previous error to determine if the difference decreases or not. It reduces overshot that is made by proportional term and integrative term, and makes the system more stable.

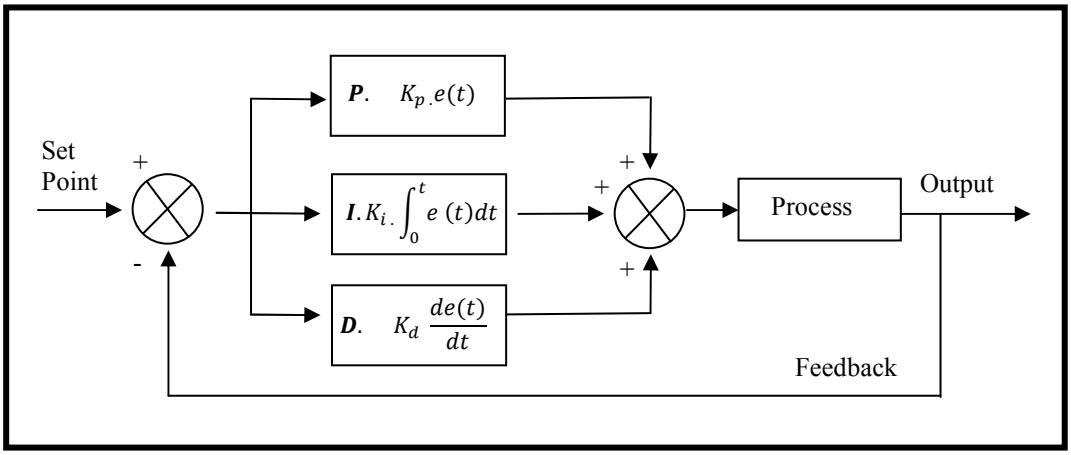


Figure 4-9: PID Block Diagram

4.4 CONTROL ALGORITHM

Total rotation angle of these motors are θ_x and θ_y are presented as x and y in this part because it is assumed there is a mechanism that converts the angular motion of motors to linear motion as explained before.

During the control process, PID position control algorithm is applied to system. Position data that is sent from the computer via RS-232 contains the coordinates that motors will reach. According to the control algorithm, motors are driven to these coordinates one by one as shown in the Figure 4-11.

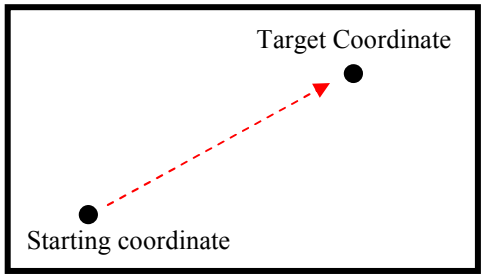


Figure 4-10: Point to Point Motion

Motion routes of the motors on Cartesian coordinate system consist of line segments between the coordinates that are sent by computer. When the computer sends a coordinate, microcontroller calculates the slope of the line segment that is formed by the starting coordinate and the target coordinate that comes from computer and formulates the equation of this line segment. This equation is in the form of y = m.x that is presented y = m.x + n, as the general equation of a line, in literature. Because in this control algorithm, equation of the line segments are produced relatively to the starting points of every line segments and these points are assumed as an origin point so the term n is eliminated and the equation becomes y = m.x that is expressed in $\Delta y = m.\Delta x$ where Δx = x-$x_{\text{starting point}}$ Δy= y-$y_{\text{starting point}}$ and $m = \frac{y_{target} - y_{starting}}{x_{target} - x_{starting}}$.

Then the distance $(x_{target} - x)$ that the motor x will take is computed. This is the error for the motor x and it is calculated continuously and instantly by comparing the present coordinate that comes from feedback with the target coordinates. For the motor y, the error term is the difference between its present coordinate (y) and the coordinate where the motor y should be $(\Delta x.m)$ according to the equation of the line segment and formulated as $(\Delta x.m - y)$. These errors are the inputs for the PID algorithm.

According to the PID algorithm proportional term is determined instantly by multiplying the K_p coefficient with the errors. Integral terms are determined by

summing the products of coefficient K_i and the errors. Derivation term is calculated by the coefficient K_d and the difference between present error and the preview error. Sum of these terms produce the voltages, V_x and V_y which decide the motor motions that include slowing down, getting faster or changing direction, and voltages are transmitted to motors. This control algorithm is visualized in Figure 4-12.

According to the control algorithm, slopes of the line segments are determined as explained before. When the line segments are vertical, in other words when the slope is infinite, it cannot be computed by microcontroller. In this case, slope is not calculated and only the motor y is powered.

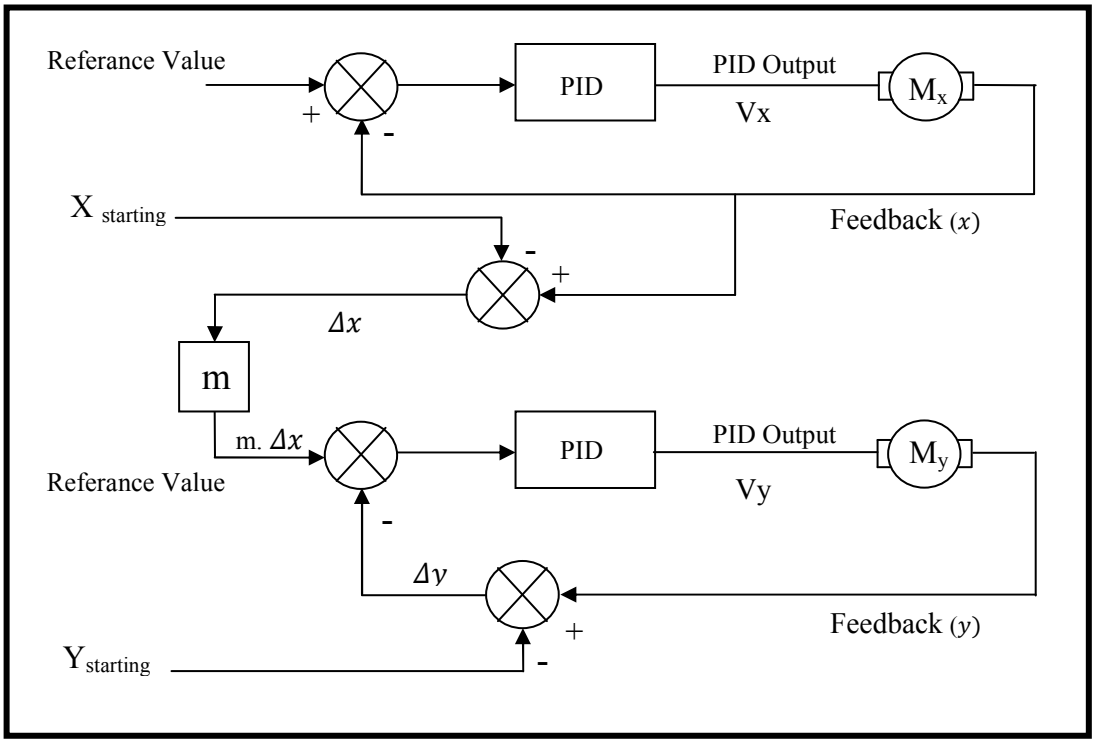


Figure 4-11: System Block Schema

4.5 EMERGENCY STOP ALGORITHM

Emergency stop is a protective algorithm that prevents the drawings against errors caused by the motor synchronizing problems. According to the algorithm when one of the motors stops because of any problem caused by the motor control card, frictions or

other disruptive effects, the other motor will stop too and waits for the other one until the disruptive effect disappears. If this effect is removed, both of the motors will start to move again.

As it was mentioned before, in this system the motor x is the master motor according to the control algorithm. So when it stops, motor y (slave motor) will stop normally. If the motor y stops, an algorithm that is based on the Equation 4.8 where Δy and Δx are the distance taken of the motor y and motor x respectively, m is the slope of a line segment becomes active and makes the motor x stop.

$$|\Delta_y - m.\Delta_x| >= \text{Dynamic Threshold Value} \quad \textbf{(4.8)}$$

The dynamic threshold value is based on the slope of the line and determined experimentally. While both of the motors are working, the value of $(\Delta_y - m.\Delta_x)$ is too small $(\Delta_y - m.\Delta_x) = 0$ in theory). But if the motor y is off, this difference becomes larger than the threshold and algorithm make the power of the motor x off. In that way both motors stop until $(\Delta_y - m.\Delta_x)$ is smaller than or equal to the threshold value.

4.6 SYSTEM MODEL

In this part, it is mentioned about the transfer function of the system and determining the PID controller coefficient.

4.6.1 Transfer Function of the System

Transfer function is the mathematical expression of the system and can be defined as the ratio of the output Laplace Transform to the input Laplace Transform assuming zero initial conditions. In this part, transfer function is also used to determine the PID coefficients.

$$T = j\alpha \quad \textbf{(4.9)}$$

$$T = js\omega \quad \textbf{(4.9a)}$$

$$T = K_t i_a \quad \textbf{(4.10)}$$

$$js\omega = K_t i_a \quad \textbf{(4.10a)}$$

$$i_a = \frac{1}{K_t} js\omega \quad \textbf{(4.10b)}$$

$$V = K_e\,\omega + L\,\frac{di_a}{dt} + R_a i_a \quad \textbf{(4.11)}$$

$$V = K_e\,\omega + R_a.i_a + Ls.i_a \quad \textbf{(4.11a)}$$

$$V = K_e s\,\theta + \frac{1}{K_t} jR_a s^2\theta + \frac{1}{K_t} Ljs^3\theta \quad \textbf{(4.11b)}$$

And equation is arranged

$$V = \theta(K_e s + \frac{jR_a s^2 + jLs^3}{K_t}) \quad \textbf{(4.11c)}$$

$$\frac{\theta}{V} = \frac{K_t}{jR_a s^2 + jLs^3 + K_e K_t.s} = \frac{K_t}{s\,(jLs^2 + jR_a s + K_e K_t)} \quad \textbf{(4.12)}$$

Equation 4.12 is the mathematical expression of the system and explains the relationship between the position and the terminal voltage of the motor except motor driver effect. In this system, output voltage of the microcontroller is amplified by the motor driver with $K_v = 2{,}4$. So the equation becomes as shown below in the Equation 7.12a.

$$\frac{\theta}{V_c} = \frac{2{,}4.K_t}{s\,(jLs^2 + jR_a s + K_e K_t)} \quad \textbf{(4.12a)}$$

is obtained as a transfer function.

4.6.2 Determining the PID Coefficient

According to the transfer function of the system, PID controller coefficients K_p, K_i and K_d are calculated using the motor characteristics. Characteristics of motor are in presented below.

$J = 5.7x\ 10^{-7}\ kg.m^2$

$R_a = 1.9\ \Omega$

$K_e = 13.369 \times 10^{-3}\ V.sec$

$K_t = 13.4 \times 10^{-3}\ N.m/A$

$L = 6.5 \times 10^{-5} H$

Optimum PID coefficients are determined using these characteristics experimentally in MATLAB. In these experiments, the best results were obtained with the coefficients $K_P = 15$, $K_d = 10$, $K_i = 000.1$. When a unit pulse is input of the system, it takes 0.001 seconds for system stable response as shown in the figure 4.13. This means that motors reach the desired speed in a short time.

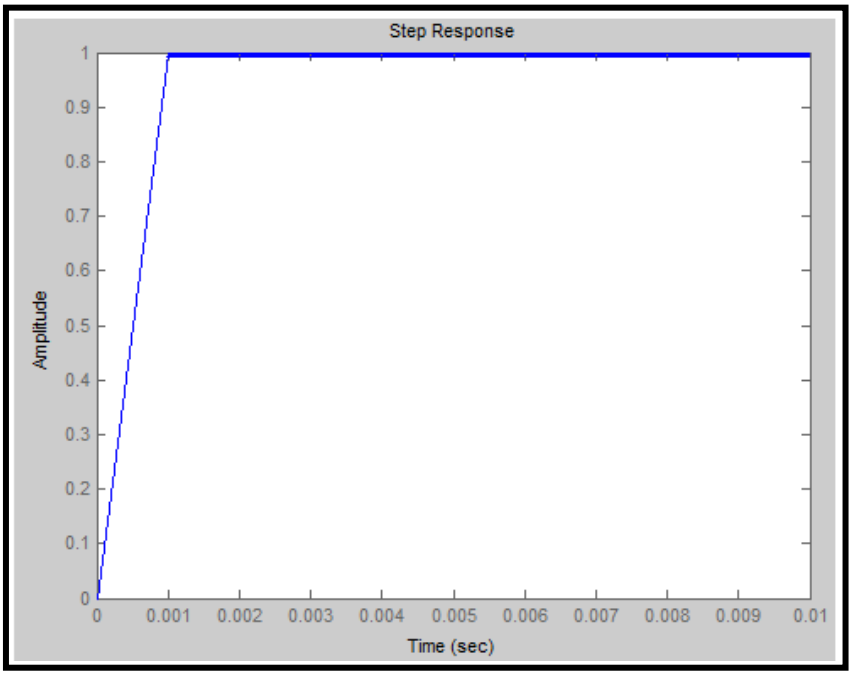


Figure 4-12: Step Response of PID System

5. VISUALIZING PLOTTER BEHAVIOR

The last step of this thesis is visualizing the drawings which are assumed to be plotted on the screen by virtual plotter. For this purpose, interface software is designed in Visual Basic. This software communicates the motor control circuit. It reads the text file and sends these data to the microcontroller and also catches the encoder signal and shows them on the plotter screen.

5.1 VIRTUAL PLOTTER

Virtual Plotter Software screen is introduced in Figure 5-1. Settings and features of this software are also explained below.

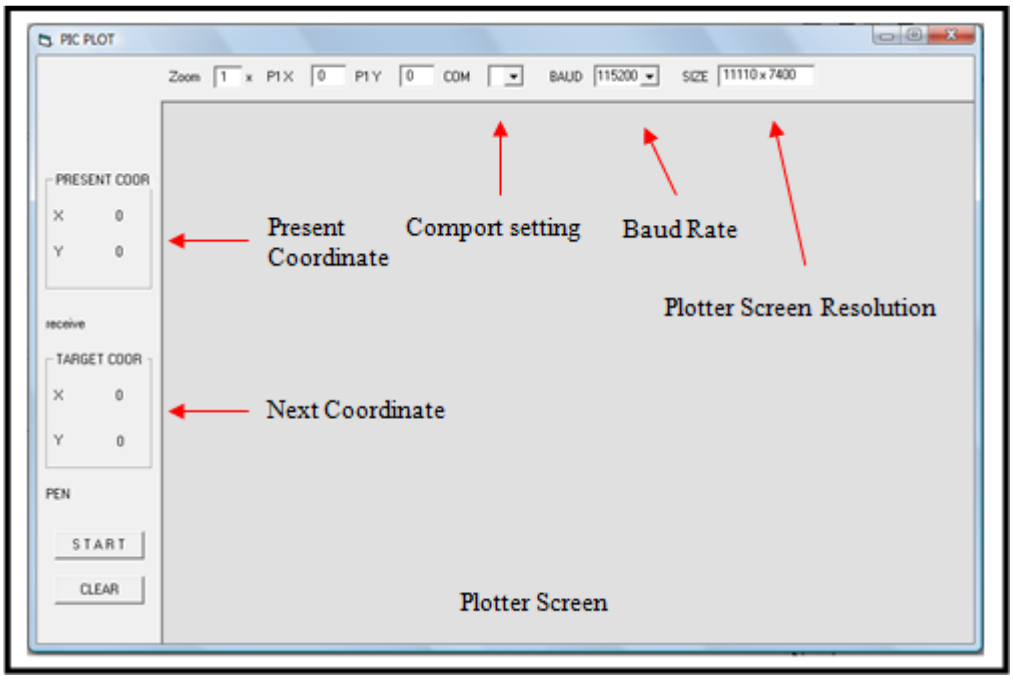


Figure 5-1: Virtual Plotter Software

Plotter Screen Resolution: This window shows the dimensions of the plotter screen that vary according to the computer screen dimensions.

Baud Rate: This setting is about the serial port communication speed. Default value is 256000 baud.

Comport Setting: This setting is the port selection option. Software decides the port which control card is connected automatically.

Present Coordinate: In this window, present coordinates of motors are located instantly.

Next Coordinate: In this window, the target position of the motor x and y are located. When motors reach these coordinates, new target coordinates show up.

Plotter Screen: Virtual Plotter screen is the Cartesian coordinate system that assumes left bottom corner as an origin point. This point is the default starting point of the plotter.

Pen: This window shows if the pen active or the plotter goes to coordinate without drawing. “PU” means, pen is not drawing, “PD” means pen is drawing.

P1X and P1Y: This is the setting that changes the origin point of Virtual Plotter screen.

5.2 ALGORITHM

This software reads the HPGL formatted text data that contains the xy coordinates and drawing commands. According to these commands “PU” or “PD”, software decides to draw or to move motors without drawing. The position data for the motor x and y are read and sent to the microcontroller. Microcontroller controls the motors and drives them to these coordinates. While motors are driven to these coordinates, microcontroller

sends “1” or “2” characters for the motor x, “3” or “4” characters for the motor y according to the their turning direction.

These signals are counted by the software and visualized on the plotter screen. When the motors reach the target point, microcontroller sends “+” character to the VB software. Then VB software sends new coordinates to microcontroller. Point to point motion is provided by this communication until all the coordinates are sent.

6. CIRCUIT DESIGN

In this section, parts of the control card, circuit components with connection schemas and their functions are explained.

6.1 VOLTAGE REGULATION

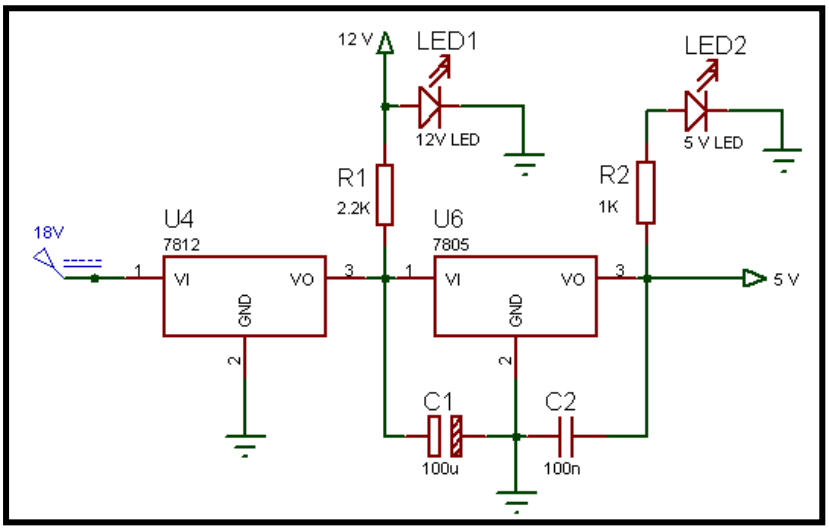


Figure 6-1: Voltage Regulation Circuit

This is the voltage regulation circuit of the control card. This part consist of two voltage regulators LM7812 and LM7805, on-off switch, power leds, and the capacitors that provide smooth signal, as shown in the Figure 6-1. Voltage regulators LM7805 and LM7812 that are the most popular component in regulation reduce the input voltage 18V, to the supply voltage of dc motors 12V and then to logic voltage 5V for the microcontroller and the other integrated circuits.

6.2 PIC18F452

In this system, PIC18F452 microcontroller is used. This microcontroller is 16-bit high speed microcontroller with the clock frequency 40 MHz, which is the most important property for this control system, and has a large program and data memories and also has many features like Hardware PWM (HPWM) module, portb change interrupts and special pins for serial communication.

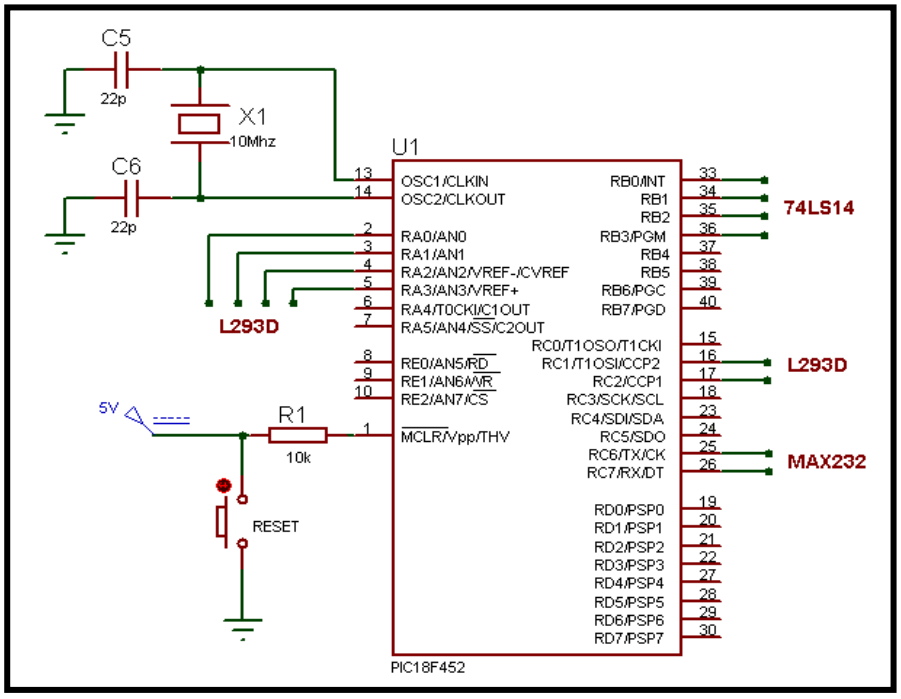


Figure 6-2: PIC18F452 Circuit

First pin is the MCLR pin and it is connected to the ground with the reset button and logic voltage with a pull up resistor.

Thirteenth and the 14th pins are for oscillator. In this circuit, 10 MHz oscillator is used. But its effect on PIC is 40 MHz with PLL feature that belongs to PIC18 family and makes the oscillator frequency four times faster.

First four pins of PORTB are connected to the Schmitt trigger 74ls14. These pins take the motor encoder pulses and these pulses create interrupts inside the PIC. According to

the algorithm, encoder pulses are counted and in that way the motor positions and the rotation direction of the motors are determined. .

First four pins of PORTA are connected to the motor driver L293D and employed to control the directions of the motors. According to the control algorithm PIC determines the direction of the motor rotation by changing the voltage (5V or 0V) on these pins.

Sixteenth and 17th pins of the PIC are also connected to the motor driver L293D. But these pins that are specialized for the hardware PWM, are employed for adjusting the speed of the motors according to the PID algorithm.

Twenty fifth and 26th pins of the PIC are connected to the MAX-232 that is used for serial communication. Reading and writing operations are applied with these pins that are specialized for serial connection.

6.3 PULSE WIDTH MODULATION

In this circuit, motor power is adjusted with the Pulse Width Modulation technique. Pulse Width Modulation (PWM) is an efficient technique for controlling analog circuits with the processor's digital outputs. In PWM applications, digital pulses in form of rectangle waves are transmitted to desired component as shown in the Figure 6-3. Voltage level is determined by the changing the width of high and low periods. Rate of high periods over whole period is the called duty cycle.

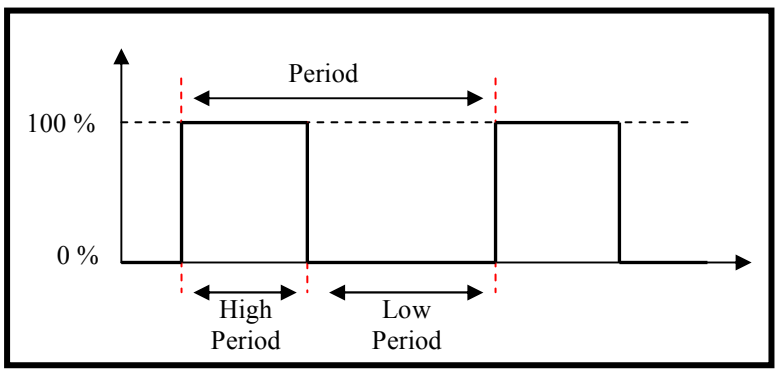


Figure 6-3: PWM Signal

Many microcontrollers include on-chip PWM units. For example, Microchip's PIC18F452 that is used in this thesis includes two PWM modules that can be activated by software.

6.4 READING ENCODER SIGNALS

Optical encoders are the electro mechanic devices that convert a mechanical position into electrical signal by means of a patterned disk or scale, a light source and photo sensitive elements. With proper interface electronics, position and speed information can be derived.

Typical encoder consists of the emitter side, detector side and the coded disk between these sides as shown in the Figure 6-4. The disk that has small holes on is attached to the motor shaft and does the circular motion with the shaft. In IR encoders, the emitters are the IR leds. While the disk is spinning, infrared light beam can pass through disk when the holes pass in front of the led, turn the IR phototransistor on and a raw analog signal is produced as an output on circuit as shown in Figure 6-5. While holes are not in front of the leds, infrared light beam cannot pass the disk and cannot reach the IR phototransistor. So signal is not produced.

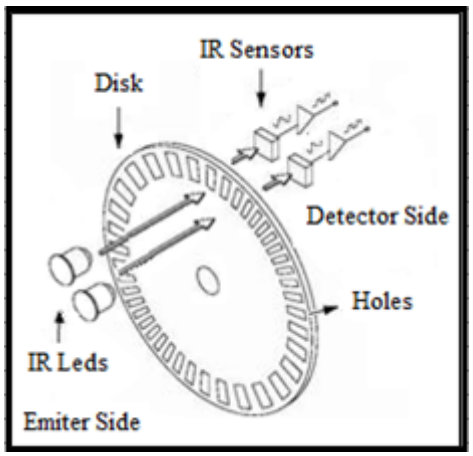


Figure 6-4: Encoder Structure

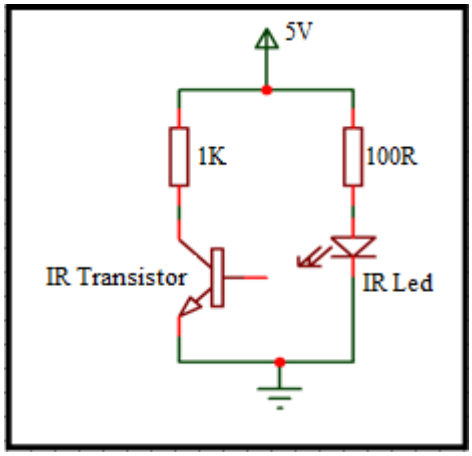


Figure 6-5: Encoder Circuit Schema

In this control system, two phase infrared encoders are used to determine the motor position and the shaft rotation direction. Two phase encoders have two emitters. So while the shaft is spinning, two square waves are produced and there is 90° phase shift between these pulses shown in the Figure 6-6. In this wise, rotation direction can be determined by the algorithm that is explained below.

6.4.1 Algorithm

This algorithm is used to determine the rotation direction of the motors as mentioned before. According to the algorithm Phase A is named as a base phase and for every positive edge of phase A, the value of phase B is determined. If its value is 1, it is assumed that motor x turns to the right and microcontroller sends the character "1" to the computer. Also present position for the coordinate x is incremented. Otherwise if the value of phase B is 0, motor x turns to the left and present coordinate of x is decremented

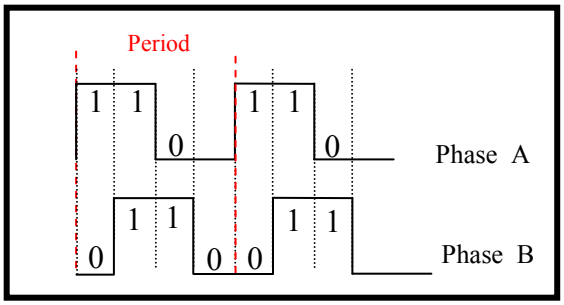


Figure 6-6: Encoder Signals

For the motor y, the same algorithm is implemented and rotation directions and positions of the motors are determined.

6.5 RS-232 SERIAL PORT COMMUNICATION

In this circuit, RS-232 serial port is used to provide communication between computer and the motor control card. It is necessary to use MAX-232 to convert the logic voltage

to the RS-232 standard voltage that can vary between -13 V and 13 V. Communication circuit as shown in Figure 6-7.

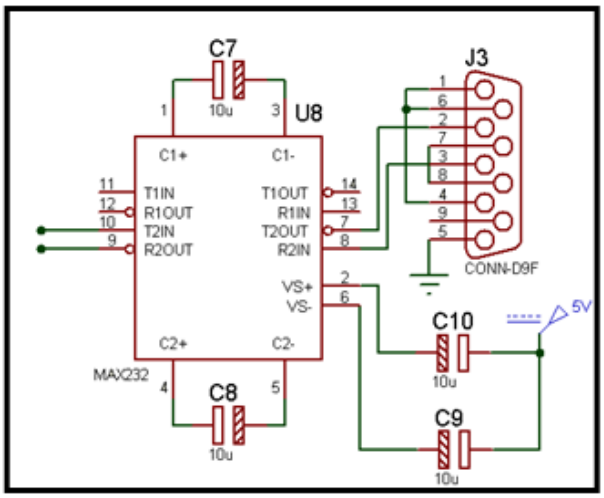


Figure 6-7: RS-232 Serial Port Circuit

6.6 SCHMITT TRIGGER

In this control circuit, motor encoders are analog IR encoders and have sensitive IR sensors to sense the shaft motion. These sensors are for the infrared light beams but they are heavily affected by the outside light (especially sun light). So in the day light it is hard to get signals correctly from the encoders.

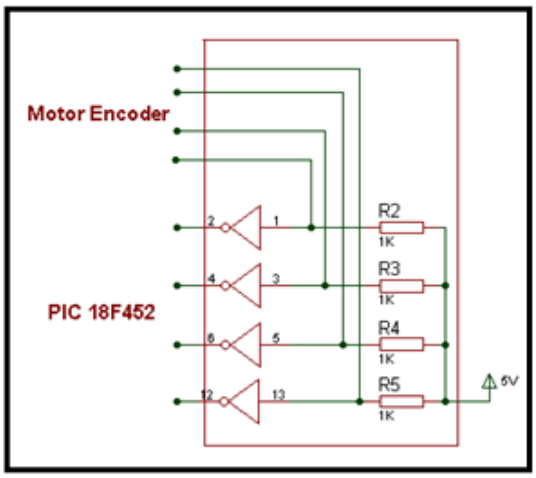


Figure 6-8:74LS14 Circuit Schema

So Schmitt Trigger 74LS14 that is an inverted analog to digital converter is located in circuit as shown in Figure 6-8. Input noisy analog pulses that come from encoders are quantized according to the threshold voltage and converted to smooth square wave as shown in Figure 6-9. In this way noises that will occur are eliminated and microcontroller get these smooth square wave pulses.

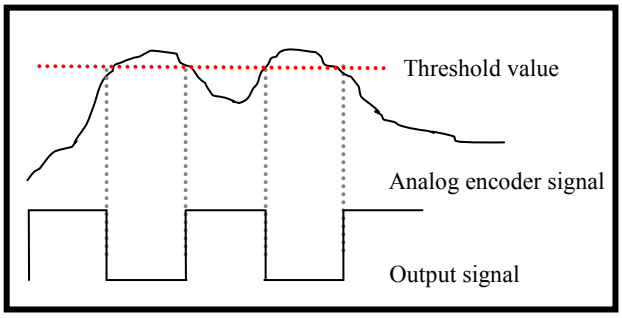


Figure 6-9: 74LS14 Input and Output Signals

Input pins are connected to the motor encoders. Noisy analog signals that come from encoders are transmitted to 74LS14.

Output pins are connected to the microcontroller. Output digital signals are transmitted to the microcontroller.

6.7 MOTOR DRIVER

The nominal voltage and current of microcontrollers are the logical voltage 5V and 20mA alternately. In this manner it is impossible to drive any motors that need more power. So driving circuits that consists of transistors and diodes or the motor drivers like L293D that includes H-bridge and back EMF diodes, are used for driving motors.

In this control circuit, L293D motor driver is used. Because its operating time and voltage range are suitable for this motor control circuit and especially with its feature

that makes the motor rotation direction easy to change is useful. Connection schema is shown in Figure 6-10.

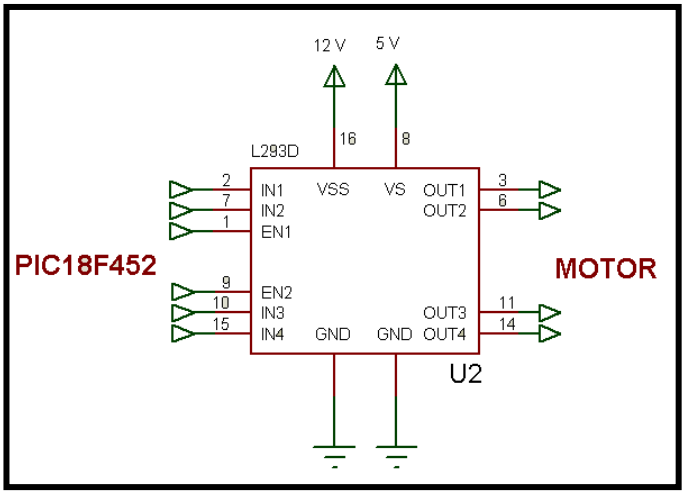


Figure 6-10: L293D Circuit Schema

First and 9th pins are the enable pins of driver and the PWM pulses are applied to these pins by microcontroller. PWM pulses determine the motor speed.

Sixteenth pin is the maximum supply voltage of motors and this voltage is operated according to the ratio of between the voltage on 8th pin and PWM pins.

Second, 7th pins and 10th, 15th pins that are connected to the microcontroller. Applied voltage on these pins, determine the direction of the motors.

Third, 6th pins and 11th, 14th pin are the output pins that are connected to motor x and the motor y alternately.

7. EXPERIMENTS

Algorithms that are explained at previous parts were applied to following images. First image is the line art of the Turkish Flag in Figure 7-1, and the second one (in Figure 7-2) is a technical drawing.

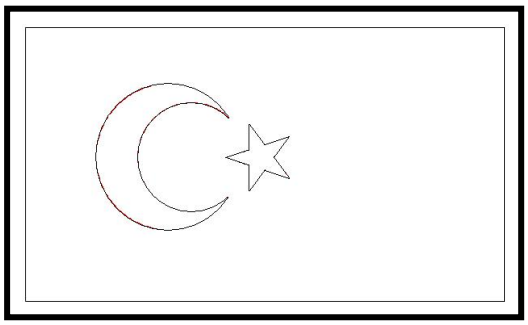


Figure 7-1: Input-1 Turkish Flag

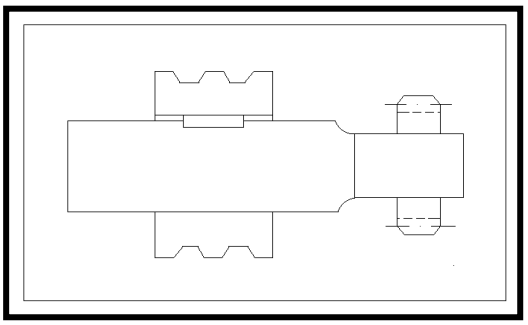


Figure 7-2: Input-2 Technical Drawing

After the vectorzation algoritm, error analysis algorithm is applied to vectorization outputs. As results of this algorithm, average error, maximum error and the standart deviation of error distribution are determined and then error distrubiton graph is plotted. These results are shown in Table 7-1. (Average and maximum error is shown in terms of pixels)

Criteria	Drawing 1	Drawing 2
Average Error	0.0991	0.0042
Max Error	2.0392	0.5547
Standart Daviation	0.2435	0.0386
Operating Time	76 sec.	35 sec

Table 7-1: Error Analysis Results

Spefications of the Computer

OS: 32 bit-Windows Vista

RAM: 3GB

Microproccessor: Intel® Core ™ Duo 2.0 GHz

According to the results in Table 7.1, it is clear that vectorization algorithm works with approximately zero errors and it is more successful on the drawings that contain linear figures. On the other hand, in the vectorization of nonlinear figures like Figure 7-1, error values are greater because this vectorization algorithm is based on the linear approximations on nonlinear figures. And also, it takes more time for algorithm to vectorize Figure 7-1 than Figure 7-2.

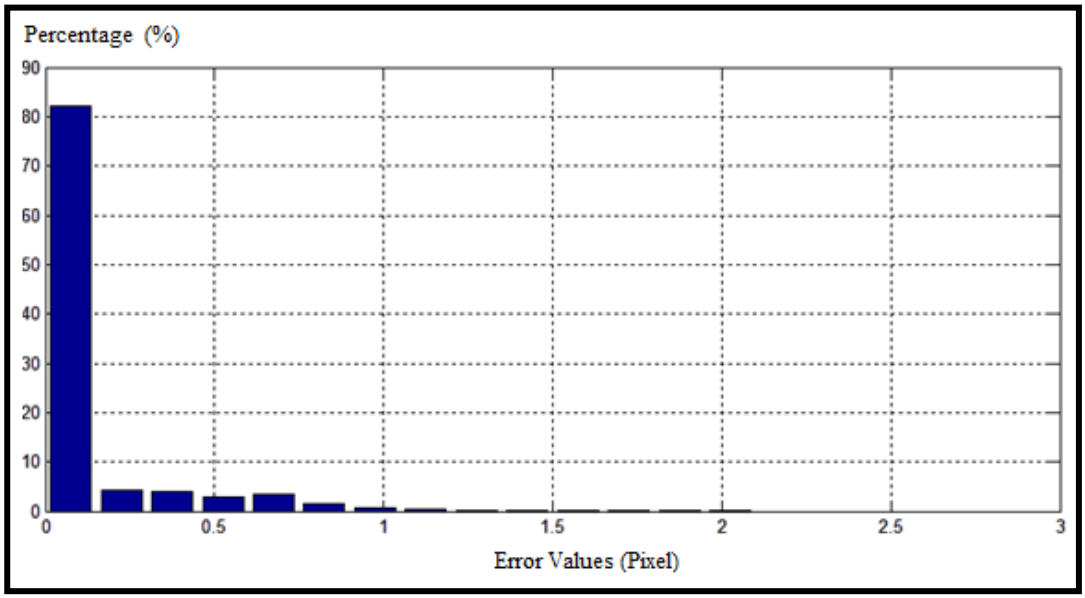


Figure 7-3: Error Distrubition Graph of Input-1

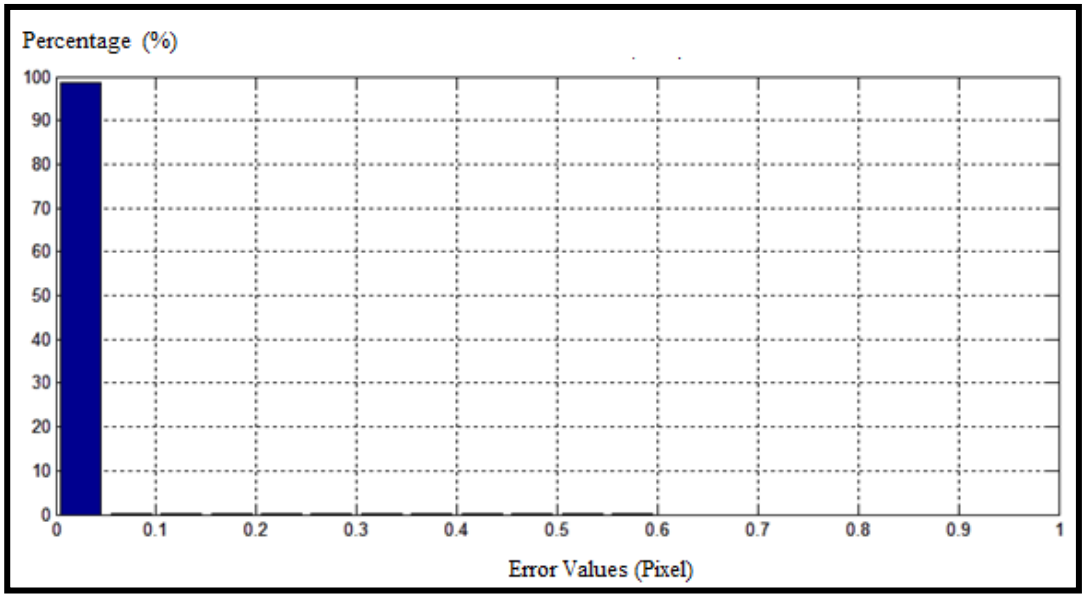


Figure 7-4: Error Distrubition Graph of Input-2

In these distribution graphs, y axis shows the percantage of the errors values. These error values are shown in the pixel by x axis. In Figure 7-3, errors values are between 0-2 pixel and the most number of errors are in the 0-0.125 error gap. In the figure 7-4, errors are between 0-0.6 pixel and the most number of error locates in the 0-0.05 error gap. At the end of machıne vision algorithm, motor control algorithm works and plots the drawings. Output drawings of system are shown in the Figure 7-5 and 7-6.

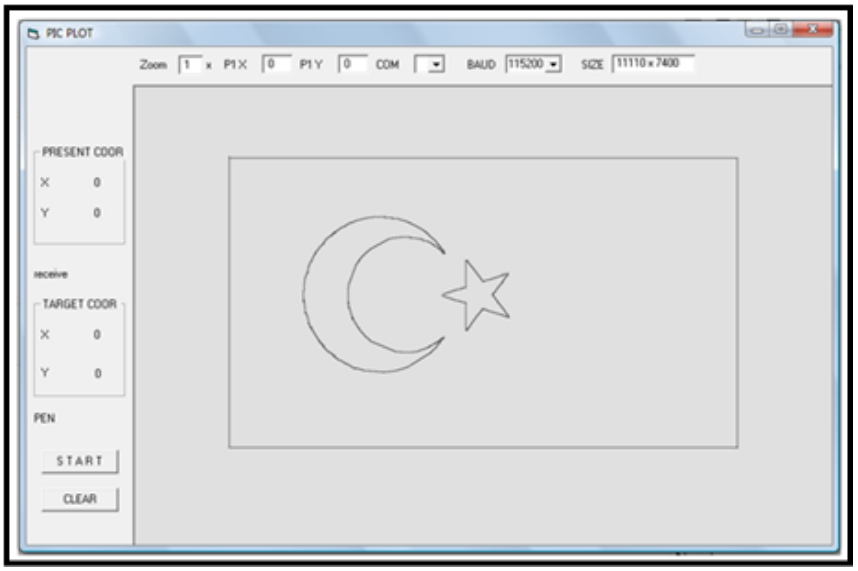


Figure 7-5: System Ouput-1

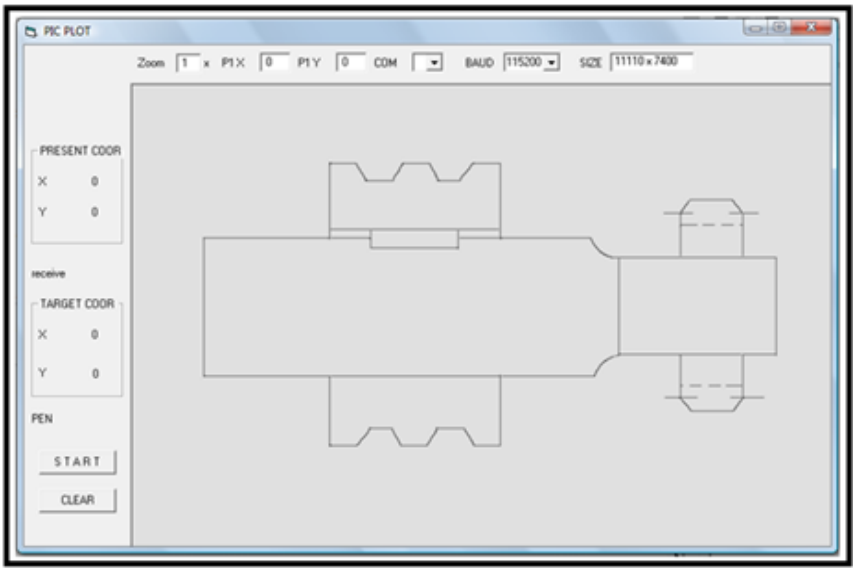


Figure 7-6: System Ouput-2

7. CONCLUSION AND FUTURE WORK

In this thesis, two axis gantry control system is developed for a virtual Cartesian pen plotter that does not have mechanical equipments except DC motors. For this system, motor control circuit is designed and machine vision software is developed as a driver for this system. This machine vision software converts a black and white raster based drawing to vector graphics format. This software includes vectorization algorithm and HPGL converter algorithm to operate images for plotters and there is an error analysis algorithm that examines the vectorization of drawings. Both of these algorithms work successfully especially at technical drawings, according to the results and error analysis. But this software will be more effective and successful when some additional algorithms are added. For example if the vectorization algorithm is assisted with a thinning algorithm, vectorization of thick lines will work faster. On the other hand if an algorithm that can detect the alphabetic letters or determine the color of the lines, vectorization algorithm will be excellent.

Another part of thesis is the motor control system. Two DC motors are controlled according to their positions and PID control method is applied. In that way, synchronized motor motion is provided. As it is clear that, for the precision necessary motor control applications, stepper motors are more preferable. But in this system, DC motor that is more preferred in industry is chosen. For the future work of this control algorithm is to use a third motor for the motion at the z-axis instead of providing motion virtually with software or a using a real mechanical X-Y table to realize this plotter.

The other part is the visualizing the virtual plotter output. For this purpose, Virtual Plotter Software that counts the encoder pulses and shows the position of the motors on own screen is developed. Other job of this software is sending the target coordinates

that are obtained by machine vision software to the microcontroller to achieve point to point motion. For the future work of this software may be, to make the display of Virtual Plotter Software more detailed although it is simple and easy to understand now.

The future work of complete system is, to develop an algorithm that takes a maximum error parameter from user and runs the machine vision and the motor control algorithms according to this error value. In this regard, user can define the maximum error value according to his/her purpose. This feature makes the system more useful.

REFERENCES

Boatto, L., 1992. An Interpretation System for Land Register Maps. *IEEE Computer 1992*; 25(7):25–32.

Chen, S. and Hsieh, T., 2007. Repetitive control design and implementation for linear motor machine tool. *International Journal of Machine Tools & Manufacture* 47 (2007) pp.1807–1816.

Chai, I. and Dori, D., 1992. Orthogonal Zig-Zag: An efficient method for extracting lines from engineering drawings. *In: Arcelli, Cordella, Sanniti, Visual Form. Plenum Press*, pp. 127–136.

Giam, S., Tan, K. & Huang ,S., 2007. Precision coordinated control of multi-axis gantry stages. *ISA Transactions* 46 (2007) pp.399–409.

Huang, P., Lin S., & Chen, Y., 1998. Real-coded Genetic Algorithm based Fuzzy Slide Mode Control For Precision Positioning.

Jimenez, J. and Navalon, JL., 1982. Some experiments in image vectorization. *IBM Journal of Research and Development*, pp. 724–734.

Koçer, B. and Yıldız, F., 2006. Türev tabanlı kenar çıkarma ile tam otomatik vektörizasyon. *J. Fac.Eng.Arch. Selcuk Univ*., v.21, n.3-4, 2006.

Lam, S., Lee, L., & Suen, C., 1992. Thinning methodologies: A comprehensive survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 1992; pp. 869–887.

Lin, X., Shimotsuji, S., Minoh, M., & Sakai, T., 1985. Efficient diagram understanding with characteristic pattern detection. *Computer Vision, Graphics and Image Processing,* pp. 84–106.

Liu, W. and Dori, D., 1996. Automated CAD Conversion with the machine drawing understanding system. *Proceedings 2nd IAPR Workshop on Document Analysis Systems, Malvern, PA*, pp. 241–259.

Liu, W. and Dori D, 1999. From Raster to Vectors. *Extracting Visual Information From Line Drawing, Pattern Analysis & Applications*.

Monagan, G. and Roosli, M., 1993. Appropriate base representation using a run graph *Proceedings 2nd International Conference on Document Analysis and Recognition Tsukuba, Japan*, pp. 623–626.

Ratcliffe, J., Hatonen, J., Lewin, P., Rogers, E., Harte, T., & Owens, D., 2005. P-type iterative learning control for systems that contain resonance. *International Journal of Adaptive Control and Signal Processing Int. J. Adapt. Control Signal Process* pp. 769–796.

Rieber, J. and Taylor D., 2004. Integrated control system and mechanical design of a compliant two-axes mechanism.

Shapiro, B., Pisa, J., & Sklansky, J., 1981. Skeleton generation from x-y boundary sequences. *Computer Graphics and Image Processing,* pp. 136–153.

Song, J., Cai, M., & Lyu, M, 2002. Graphics recognition from binary images: One step or two steps. *In International Conference on Pattern Recognition (ICPR)*, volume 3, pp. 135– 138.

Tamura, H.,1978. A comparison of line thinning algorithms from digital geometry viewpoint. Proc. *4th Int. Conf. Pattern Recognition*, pp. 715-719.

Torngren, M., Henriksson, D., Arzen, K.., Cervin, A., & Hanzalek, H., 2006. Tool supporting the co-design of control systems and their real-time implementation: Current status and future directions. *In IEEE International Symposium on Computer-Aided Control Systems Design*.

Yang, Z. and Red, E., 1996. On-line Cartesian trajectory control of mechanisms along complex curves', *Communications of the ACM.*

Zhang, T., and Suen, Z., 1984. A Fast Parallel Algorithm for Thinning Digital Patterns. *Communications of the ACM.*

.

APPENDICES

Appendix A

MikroC Code

```
#define YENC0 PORTB_st.F7
#define YENC1 PORTB_st.F6
#define XENC0 PORTB_st.F5
#define XENC1 PORTB_st.F4

#define XYONA PORTA.F0
#define XYONB PORTA.F1
#define YYONA PORTA.F2
#define YYONB PORTA.F3

#define START PORTA.F4

unsigned short olds,olds_2 ;
long x = 0;
long y = 0;
long old_y=0;                   // actual y value
long old_x=0;                   // actual x value
float  m = 0.000000 ;
unsigned long sayi[10]={0,0,0,0,0,0,0,0,0,0};
unsigned char indis = 0;
unsigned long dec_num[2];
unsigned short coordinate;

long pwm_1;
long pwm_2;
long pwm_2_limit ;
long pwm_1_limit;
unsigned char PORTB_st;

int sayac=0;
unsigned char direction_x =0;
unsigned char direction_y =0;

unsigned char direction_x_flag=0;
unsigned char direction_y_flag=0;

int counter = 0;

int flag=0;                     // interrupt flag
int usart_flag=0;               // usart interrupt flag

float y_p_err=0;
float y_i_err=0;
float y_i_err_total=0;
float y_d_err=0;
float y_err=0;
float y_prev_err=0;
float y_total_err=0;

float delta_y=0;
float delta_y_cal=0;
```

```
 float y_start=0;
```

Appendix A-cont'd

```
 float y_req=0;
 float old_y_req =0;
 long x_p_err=0;
 long x_i_err=0;
 long x_i_err_total=0;
 long x_d_err=0;
 long x_err=0;
 long x_prev_err=0;
 long x_total_err=0;

 long delta_x=0;
 long delta_x_cal=0;
 long x_start=0;
 long x_req=0;
 long old_x_req =0;

 float Kp_x;
 float Ki_x;
 float Kd_x;

 float Kp_y;
 float Ki_y;
 float Kd_y;
////////////////////////////////////////////////////////////////////////////////
 void read()
   {
    if (USART_Data_Ready())
    {
     coordinate = Usart_Read();
     sayi[indis] = coordinate ;
     indis++;
     dec_num[0] = 10000 *sayi[4]+1000 *sayi[3]  + 100 *sayi[2] + 10 *sayi[1] + sayi[0];
     dec_num[1] = 10000 *sayi[9]+1000 *sayi[8]  + 100 *sayi[7] + 10 *sayi[6] + sayi[5];

     if ( indis > 9 )
     {
     indis=0;
     sayi[0]=0;sayi[1]=0;sayi[2]=0;sayi[3]=0;sayi[4]=0;
     sayi[5]=0;sayi[6]=0;sayi[7]=0;sayi[8]=0;sayi[9]=0;
     usart_flag=1;
     }
    }
   }
////////////////////////////////////////////////////////////////////////////////

 void interrupt() {
   if(INTCON.RBIE==1)
   {
   PORTB_st=PORTB;
     if (XENC0==0){
     olds=0;
               }
     else if (olds==0)
     {
```

Appendix A-cont'd

```
 olds=1;                                          //  X encoder
        if (XENC1==1){
        flag=1;
        x=x+1;
     direction_x_flag=1;
        direction_x='1';
                  }
        else   {
        flag=1;
        x=x-1;
        direction_x_flag=1;
        direction_x='2';
             }
      }
////////////////////////////////////////////////////////////////////////////////
    if (YENC0==0)       {
       olds_2=0;
                  }
     else if (olds_2==0)
     {
     olds_2=1;
          if (YENC1==1) {
          y=y+1;
          flag=1;
          direction_y_flag=1;                     //  Y encoder
          direction_y='3';
                  }
          else  {
          y=y-1;
          flag=1;
          direction_y_flag=1;
          direction_y='4';
             }
      }
          INTCON.RBIF=0;
    }
  }
////////////////////////////////////////////////////////////////////////////////
short int sgn (long x)
{
     if (x<0) return (-1);                        //Signum Function
     else return (1);
}
////////////////////////////////////////////////////////////////////////////////
void main()
{
  short int y_prv_err_sign;
  short int x_prv_err_sign;

  PORTC = 0;
  TRISC = 0;

  PORTB = 0;
  TRISB = 240;
```

Appendix A-cont'd

```
  PORTA = 0;
  TRISA = 16;

  ADCON1 = 7;

  PORTD=0;
  TRISD=0;

  Usart_Init(256000);

  PWM1_Init(2500);                                  // main
  PWM2_Init(2500);

  XYONA=1;
  XYONB=0;
  YYONA=1;
  YYONB=0;

  PWM1_Start();
  PWM2_Start();

  INTCON.RBIF = 0;
  INTCON.RBIE = 1;
  INTCON.GIE  = 1;

  x_prv_err_sign = 1;
  y_prv_err_sign = 1;

////////////////////////////////////////////////////////////////////////////////
  while (1) {

      if  (direction_x_flag) {
      Usart_Write(direction_x);
      direction_x_flag=0;
      }
      if (direction_y_flag) {
      Usart_Write(direction_y);
      direction_y_flag=0;
      }
      if (USART_Data_Ready())                       // PID coefficient
      read();

      Kp_x=15;
      Ki_x=0.00001 ;
      Kd_x=0;

      Kp_y= 15 ;
      Ki_y=0.00001 ;
      Kd_y=10;

      pwm_2_limit = 255;
      pwm_1_limit = 20;
////////////////////////////////////////////////////////////////////////////////

      old_x_req = x_req;
```

Appendix A-cont'd

```
    old_y_req = y_req;
     if(usart_flag)                                  // Position calculations
     {
     x_req = dec_num[0];
     y_req = dec_num[1];
     flag=1;
     usart_flag=0;
     }
////////////////////////////////////////////////////////////////////////////////
     if (flag) {
     x_prev_err = x_err;
     y_prev_err = y_err;

     x_err = x_req - x;
     x_p_err = x_err * Kp_x;

     x_i_err = x_err * Ki_x;
     x_i_err_total = x_i_err + x_i_err_total;
     x_d_err =(x_err-x_prev_err)*Kd_x;

     x_total_err = ( x_p_err + x_i_err_total + x_d_err);

     if(abs(delta_y-m*delta_x) > 10*abs(m)+10) {
     pwm_1=0;
     PWM1_Change_Duty(pwm_1);                  // Motor X Control
     }
     else {

          if ( x_prv_err_sign != sgn (x_total_err)  )
          {
              x_prv_err_sign = sgn(x_total_err );
              XYONA = ~XYONA;
              XYONB = ~XYONB;
              x_total_err= x_total_err/2;
          }
          pwm_1 = abs(x_total_err);

          if( pwm_1 >= pwm_1_limit)
          pwm_1 = pwm_1_limit;

     PWM1_Change_Duty(pwm_1);
          }
////////////////////////////////////////////////////////////////////////////////
     if(x_req == x_start )  {
     y_err = y_req - y;
     Kp_y=15;
     Kd_y=10;
     pwm_2_limit = 15;
                    }
     else  {
     m = (y_req-y_start) / (x_req-x_start);
     delta_x = x - x_start;
     delta_y = y - y_start;
     delta_y_cal = m * delta_x;
```

Appendix A-cont'd

```
 y_err = delta_y_cal - delta_y;
          }
      y_p_err = y_err * Kp_y;

      y_i_err = y_err * Ki_y;
      y_i_err_total = y_i_err + y_i_err_total;
      y_d_err =(y_err-y_prev_err)*Kd_y;
                                                          // Motor Y Control
      y_total_err = ( y_p_err + y_i_err_total + y_d_err);
       if ( y_prv_err_sign != sgn (y_total_err)  )
          {
             y_prv_err_sign = sgn(y_total_err );
             YYONA = ~YYONA;
             YYONB = ~YYONB;
             y_total_err= y_total_err/2;
          }
          pwm_2 =abs(y_total_err);

          if( pwm_2 > pwm_2_limit )
          pwm_2= pwm_2_limit;

     PWM2_Change_Duty(pwm_2);
////////////////////////////////////////////////////////////////////////////
     if (((x == x_req ) && (y == y_req)) &&( (y!=old_y) || (x!=old_x))) {
     x_start = x;
     y_start = y;                                        // Receiving new coordinates
     old_x=x;
     old_y=y;
     Usart_Write ('+');                                 }

     flag=0;
     }
  }
  }
```

APPENDIX B

Visual Basic Code

```
Option Explicit

Private Declare Sub Sleep Lib "kernel32" (ByVal dwMilliseconds As Long)

Dim comno As Integer
Dim byt As String * 1
Dim p1xn As Long, p1yn As Long
Dim carpann As Single
Dim A As String
Dim B As String
Dim C As String


Dim X As Double, Y As Double
Dim newx As Double, newy As Double

Dim flag As Integer

Dim progname As String

Private Sub Check1_Click()

End Sub

Private Sub gonder_Click()
Dim i As Integer
Dim A As String
Dim B As String
Dim C As String
Input #1, C, A, B

 A = Format(Val(A), "00000")

 B = Format(Val(B), "00000")

Label12 = A
Label13 = B
Label11 = C
''''''''''''''''''''''''''''''
If C = "PU" Then              '
flag = 0                    '
End If                      '
If C = "PD" Then              '
flag = 1
End If
''''''''''''''''''''''''''''''
For i = 5 To 1 Step -1
          comport.Output = Chr(Val(Mid(A, i, 1)))
          Do While comport.OutBufferCount > 0
          Loop
          Sleep (10) ' For PIC to catch up
       Next i
For i = 5 To 1 Step -1
```

APPENDIX B-cont'd

```
  comport.Output = Chr(Val(Mid(B, i, 1)))
        Do While comport.OutBufferCount > 0
        Loop

        Sleep (10) ' For PIC to catch up
      Next i
End Sub

Private Sub Form_Load()

progname = "VIRTUAL PLOTTER SOFTWARE"

Call port_fill
If portlist.ListCount > 0 Then
   portlist.ListIndex = 0
   comno = portlist.Text
End If

On Error Resume Next

p1x = GetSetting(appname:=progname, section:="Startup", _
                 Key:="p1x", Default:="00000")
p1y = GetSetting(appname:=progname, section:="Startup", _
                 Key:="p1y", Default:="00000")
carpan = GetSetting(appname:=progname, section:="Startup", _
                 Key:="carpan", Default:="1")
Me.Top = GetSetting(appname:=progname, section:="Startup", _
                 Key:="Me_Top", Default:=Me.Top)
Me.Left = GetSetting(appname:=progname, section:="Startup", _
                 Key:="Me_Left", Default:=Me.Left)
Me.Width = GetSetting(appname:=progname, section:="Startup", _
                 Key:="Me_Width", Default:=Me.Width)
Me.Height = GetSetting(appname:=progname, section:="Startup", _
                 Key:="Me_Height", Default:=Me.Height)
comno = GetSetting(appname:=progname, section:="Startup", _
                 Key:="Com_No", Default:=1)
baud = GetSetting(appname:=progname, section:="Startup", _
                 Key:="Baud", Default:=256000)
p1xn = p1x
p1yn = p1y
carpann = carpan

comport.InputLen = 1

baud.ListIndex = 2

Open "z:\data.txt" For Input As #1          'Dosyadan okumak için

plot_calistir

comport.PortOpen = False
End Sub

Private Sub clear_Click()
```

APPENDIX B-cont'd

```
comport.PortOpen = False

Picture1.Cls

comport.PortOpen = True

X = 0#

Y = 0#
coordx = LTrim(Str(X))
coordy = LTrim(Str(Y))
newx = 0
newy = 0
A = 0
B = 0
C = 0
Picture1.SetFocus

End Sub

Private Sub gonder1_Click()

End Sub

Private Sub Image1_Click()

End Sub

Private Sub Label19_Click()

End Sub

Private Sub Label28_Click()

End Sub

Private Sub picture1_KeyDown(KeyCode As Integer, Shift As Integer)

   byt = Chr(KeyCode)
   If byt = " " Then
      comport.Output = Chr(48 + Int(Rnd() * 4 + 1))
      byt = ""
   End If

End Sub

Private Sub plot_calistir()

PIC_PLOT_FRM.Visible = True
Picture1.SetFocus

X = 0#
Y = 0#
coordx = LTrim(Str(X))
coordy = LTrim(Str(Y))
```

APPENDIX B-cont'd

```
Do While True
   DoEvents
   If comport.InBufferCount > 0 Or byt <> " " Then
      If byt = " " Then byt = comport.Input
      rcvdchr = byt
      If byt <> Chr(0) Then
         Select Case byt
            Case Is = "1", "a": newx = X + carpann
            Case Is = "2", "b": newx = X - carpann
            Case Is = "3", "c": newy = Y + carpann
            Case Is = "4", "d": newy = Y - carpann
            Case Is = "C": clear_Click: Beep
            Case Is = "+": gonder_Click

         End Select
         If flag = 1 Then
         Picture1.Line (X - p1xn, Picture1.Height - Y - 100 + p1yn)-(newx - p1xn, Picture1.Height - newy
- 100 + p1yn)
         End If
         X = newx
         Y = newy
         coordx = LTrim(Str(X))
         coordy = LTrim(Str(Y))
      End If
      byt = ""
   End If
Loop

End Sub

Private Sub Form_resize()
  If PIC_PLOT_FRM.Height > 1000 Then Picture1.Height = (PIC_PLOT_FRM.Height - 1000)
  If PIC_PLOT_FRM.Width > 1850 Then Picture1.Width = PIC_PLOT_FRM.Width - 1850
  p1cap.Top = PIC_PLOT_FRM.Height - 700
  wndsize = Str(Picture1.Width) & " x" & Str(Picture1.Height)
'  wndsize.Left = PIC_PLOT_FRM.Width - 1450
  Picture1.Cls
End Sub

Private Sub Form_Unload(Cancel As Integer)
   On Error Resume Next
   comport.PortOpen = False
   On Error GoTo 0

   SaveSetting appname:=progname, section:="Startup", Key:="p1x", setting:=p1x
   SaveSetting appname:=progname, section:="Startup", Key:="p1y", setting:=p1y
   SaveSetting appname:=progname, section:="Startup", Key:="carpan", setting:=carpan
   SaveSetting appname:=progname, section:="Startup", Key:="Me_Top", setting:=Me.Top
   SaveSetting appname:=progname, section:="Startup", Key:="Me_Left", setting:=Me.Left
   SaveSetting appname:=progname, section:="Startup", Key:="Me_Width", setting:=Me.Width
   SaveSetting appname:=progname, section:="Startup", Key:="Me_Height", setting:=Me.Height
   SaveSetting appname:=progname, section:="Startup", Key:="Com_No", setting:=comno
   SaveSetting appname:=progname, section:="Startup", Key:="Baud", setting:=baud.Text

   End
End Sub
```

APPENDIX B-cont'd

```
Private Sub p1x_lostfocus()
   p1xn = p1x
   p1x = Str(p1xn)
   Picture1.SetFocus
End Sub
Private Sub p1y_lostfocus()
   p1yn = p1y
   p1y = Str(p1yn)
   Picture1.SetFocus
End Sub
Private Sub carpan_lostfocus()
   carpann = carpan
   carpan = Str(carpann)
   Picture1.SetFocus
End Sub

Private Sub port_fill()
Dim i As Integer

On Error Resume Next
For i = 1 To 5
   comport.CommPort = i
   comport.PortOpen = True
   If Err = 0 Then
      portlist.AddItem (Str(i))
   End If
   comport.PortOpen = False
Next i
On Error GoTo 0
End Sub

Private Sub Picture2_Click()

End Sub

Private Sub portlist_Click()

If comport.PortOpen Then comport.PortOpen = False
comno = portlist.Text
comport.CommPort = comno
comport.PortOpen = True

End Sub
Private Sub baud_click()
Dim comsetting As String

On Error Resume Next
comport.PortOpen = False
If Err = 0 Then
   comsetting = baud.Text & ":N,8,1"
   comport.PortOpen = True
End If
On Error GoTo 0

End Sub
```

CURRICULUM VITAE

Onur Gökçe Öcal

Tel: 0 (284) 535 02 49
0 (535) 816 10 22
Adress: Can Ören Sitesi, B Blok, Daire 7, EDİRNE

Birth Date: 07.03.1984
Birth Place: İstanbul
Nationality: T.C.
Marital Status: Single

Educational :

2003 – 2007	**Bahçeşehir University** Electrical and Electronics Engineering
1999 – 2002	**High School of Science (Edirne)**
1995 – 1999	**Edirne Anatolian High School (Edirne)**
1990 – 1995	**Şükrüpaşa Primary School (Edirne)**

Military Service: No.

Foreign Languages:

English
(Advanced)
German
(Medium)
French
(Low Level)

Job Experience :

2007-2009	**Akaryakıt Pompaları San. Tic. Ltd. Şti. (İstanbul)** Research and Development Engineer
2006	**Bilkont Dış Ticaret ve Tekstil Sanayi A.Ş. (Kırklareli)** Electrical and Electronics Training
2005	**Zorlu Linen (Kırklareli)** Electrical and Electronics Training
2004	**Bahçeşehir University (İstanbul)** MS Office, Make your own PC topics in 20 work days

Professional Training:

2009 **Otomatik Kumanda ve Kontrol Sistemleri**

- Basics of Control Systems
- Designing Control Systems
- PLC Programming

2004 **SmartPro**

- Information Systems

Computer Knowledge:

* Visual Basic
* MATLAB
* MikroC
* Microsoft Windows 98/XP/Vista
* Word, Excel, Access
* Siemens Logo Comfort-V5
* PLC Siemens S7-200 and MicroWin40

Hobbies: Soccer, Cinema, Music, Swimming, Travelling.