T.C.
MARMARA UNIVERSITY
FACULTY OF ENGINEERING

CSE 315
DIGITAL DESIGN

TERM PROJECT

-ARMy-

Visual Servoing with a 2-link Planar Manipulator

Submitted By: Tekin Meriçli 200201221

Submitted To: Assoc. Prof. Murat Doğruel

2004
İstanbul,Göztepe

Table of Contents

I INTRODUCTION

The word 'robot' was coined by the Czech playwright Karel Capek (pronounced "chop'ek") from the Czech word for forced labor or serf. Capek was reportedly several times a candidate for the Nobel prize for his works and very influential and prolific as a writer and playwright.

The term 'robotics' refers to the study and use of robots. The term was coined and first used by the Russian-born American scientist and writer Isaac Asimov (born Jan. 2, 1920, died Apr. 6, 1992). Asimov wrote prodigiously on a wide variety of subjects. He was best known for his many works of science fiction. The most famous include *I Robot* (1950), *The Foundation Trilogy* (1951-52), *Foundation's Edge* (1982), and *The Gods Themselves* (1972), which won both the Hugo and Nebula awards.

The word 'robotics' was first used in *Runaround*, a short story published in 1942. *I, Robot*, a collection of several of these stories, was published in 1950. Asimov also proposed his three "Laws of Robotics", and he later added a 'zeroth law'.

- **Law Zero:** A robot may not injure humanity, or, through inaction, allow humanity to come to harm.
- **Law One:** A robot may not injure a human being, or, through inaction, allow a human being to come to harm, unless this would violate a higher order law.
- **Law Two:** A robot must obey orders given it by human beings, except where such orders would conflict with a higher order law.
- **Law Three:** A robot must protect its own existence as long as such protection does not conflict with a higher order law.

In this project, a simple robot arm is implemented. What this robot arm does is finding the positions of the red metal pieces that are scattered on the black surface, and collecting them on a point determined by the user. The details of the implementation are given in what follows.

II BACKGROUND

In this section, brief information about theoretical backgrounds of vision and manipulator systems are given.

II.I COMPUTER VISION BASICS

The goal of Computer Vision is to make useful decisions about real physical objects and scenes based on sensed images. Computer vision researches have been started as a branch of Artificial Intelligence. Eventually, as the researches on computer vision widens and deepens, it became a different research area. Mainly, Computer Vision (or Machine Vision) concentrates on interpreting an image obtained from a source (a camera, in general). Computer Vision has a very wide application area varying from object recognition in industrial manufacturing and robotic systems to interpreting satellite images for disaster area examination, from diagnosing cancer to reconstruction of 3D models of historical buildings by using taken photos of them.

A digital image, which is the input of a Computer Vision system, is a two dimensional matrix of atomic picture elements called “Pixels”. A pixel has some numerical values denoting the optical characteristics (such as intensity, color, etc.) of the corresponding point on the image. There are various formats for representing a pixel. If we have a black & white picture, we need only intensity or brightness information of the point. If we have a color image, we need the main color components specifying both intensity and color information. Some of the popular image formats in color image representation are RGB, YUV, HSI. RGB means Red, Green and Blue, consists of amounts of red, green and blue colors in a pixel. HSI means Hue, Saturation and Intensity. In YUV, Y is luminance and U and V are chromaticity values. In our work, we have used RGB color base so we will use RGB for explaining vision and pattern classification concepts. Since a pixel consists of three components, we can consider a pixel as a point in 3D space where axes are Red, Green and Blue components. If we use 24 bits per pixel, we can specify levels of red, green and blue between [0,255].

Usually, there are three main steps in a computer vision system:

- Color classification / quantization
- Region finding (region growing/merging)
- Object recognition

In color classification/quantization step, number of colors in the image is reduced to a smaller number in order to provide homogeneity in image and make region finding possible. There are many algorithms exists for both color classification and color quantization. Color classification is a supervised process which means we have to map input pixel values to a certain class code according by using a classification function learned from a train set. In color quantization, there is no need to a supervisor. In our work, we have used color classification for distinguishing pieces and manipulator arm from each other and from background.

In region finding step, either color classified or quantized image is used as input and regions (connected pixels of same class/color) are determined. There are two main approaches for region finding: Region clustering and region growing. In region clustering, neighboring pixel with more than a certain similarity are treated as regions. In region growing, regions are found by expanding the bounds of a region by adding similar neighbor pixels to the region. In our work, we have used a recursive region growing algorithm for determining pieces and manipulator arm.

There are many algorithms exists in the literature for object recognition by using the region-segmented image. Since we have only two types of interested objects that are distinguishable by their colors, we did not used an object recognition algorithm.

II.II A PRIMER ON PATTERN CLASSIFICATION

In many practical problems, we have some data and we want to have some information about the characteristics of the data. There are two main aspects on pattern recognition: Classification and Regression. In regression, we construct a mathematical model of the data by using a train set and try to foresee the characteristics of the future data. In classification, we train a classification system and try to find which class a data point belongs to. In this project, we have used a classification system for determining the pixels belonging to “Piece” class and “Manipulator” class.

II.III INVERSE KINEMATICS

In trajectory planning and calculation of robotic manipulators, there are two types of kinematics are used: Forward and Inverse Kinematics. Forward kinematics deals with the calculation of resultant position and orientation of the end-point of a manipulator with a set of given joint angle values. In contrary to that, inverse kinematics tries to find the candidate set of joint angle values that results in a given position and orientation of the end-point of the manipulator.

III PROPOSED APPROACH

ARMy project can be separated into three main subsections:

- Vision System
- Manipulator System
- Motor Controller

In this section, detailed descriptions of vision subsystem, kinematical model of manipulator and control architecture for stepper motors are given.

III.I VISION SYSTEM

Vision module of the project is used for finding the center points of the red pieces that are randomly distributed on a black surface, and determining the position of the arm head before and after each action of the manipulator.

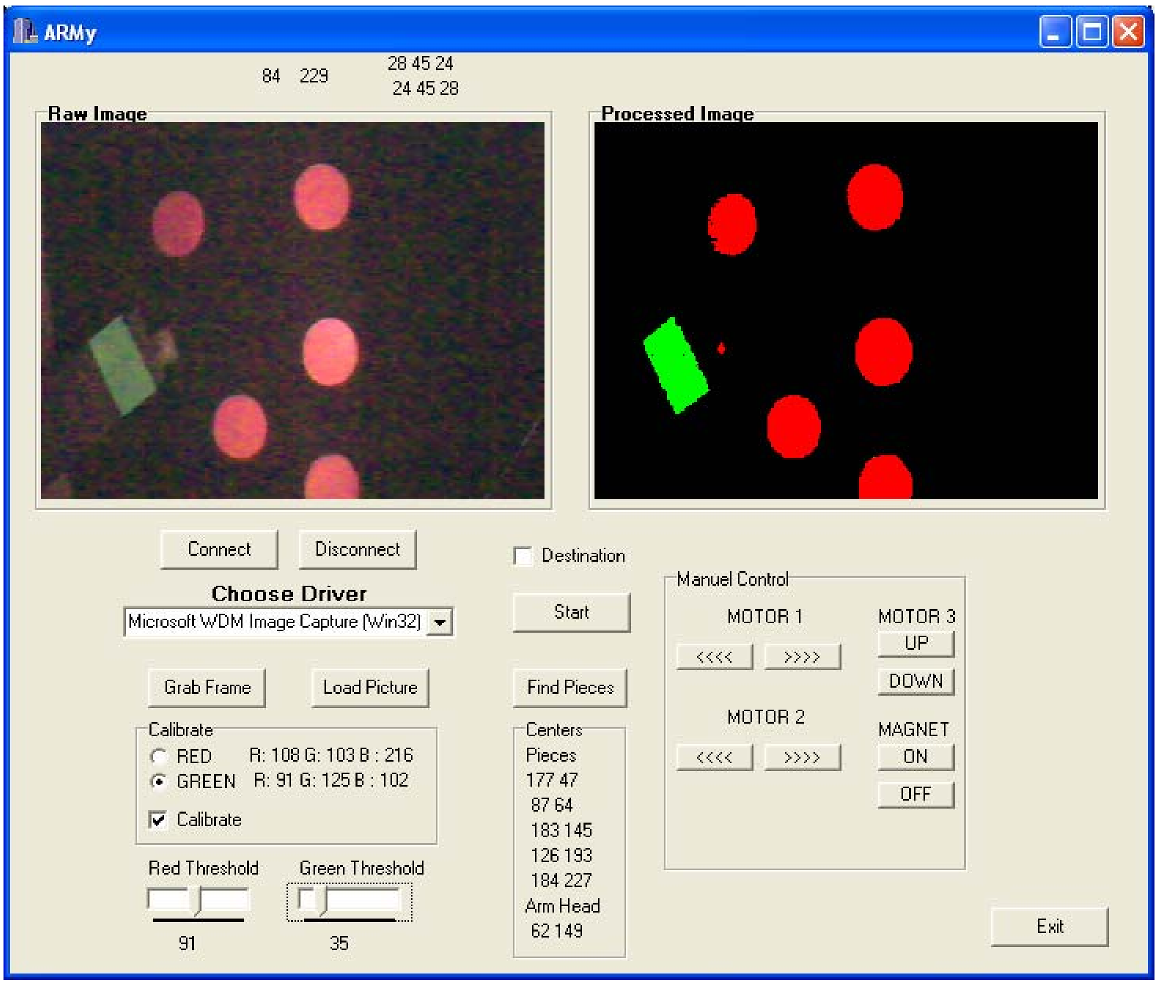


Figure 1 - Vision and manual control interface of the ARMy project

To find the center points of the pieces and the arm head, the system is first trained manually by selecting random pixels from the red and green regions. The light conditions are not the same for all the regions on the surface. Hence, when training the system, the pieces are scattered to different regions on the surface, and the arm head is located in different positions. For each position of the pieces and the arm head, one frame showing the current situation of the system is grabbed from the camera, and the system is trained again according to these new conditions.

During the training process, what is done is summing up the values of red, green and blue components of the selected pixels (red pixels when training for the pieces, and green pixels when training for the arm head), and then dividing the sum by the number of selected pixels to find the centroids of the nebulas consist of red and green pixels in the RGB space.

When the system is started, the raw image that is obtained by grabbing a new frame from the camera after each action of the manipulator is scanned, and for each pixel of the image, the distance of this pixel to the centroids of the nebulas is determined by using the formula,

$$d = \sqrt{(R - ROC)^2 + (G - GOC)^2 + (B - BOC)^2}$$

where d is distance, R,G, and B are the red, green and blue components of the pixel, respectively, and ROC, GOC, and BOC are red, green, and blue components of the centroid of one nebula, respectively. The new pixel is put into one of the two classes (red nebula or green nebula) according to the distance to the centroid of the nebula. If the distance is above some predefined threshold value (threshold values can be changed by using the threshold sliders in the user interface), the pixel is not put into that class. If it can not be classified as being red or green, then it is said to be in the “ignore” class. Figure 2 shows the distribution of the “red nebula” and the “green nebula” in the RGB space after scanning the image and classifying each pixel.

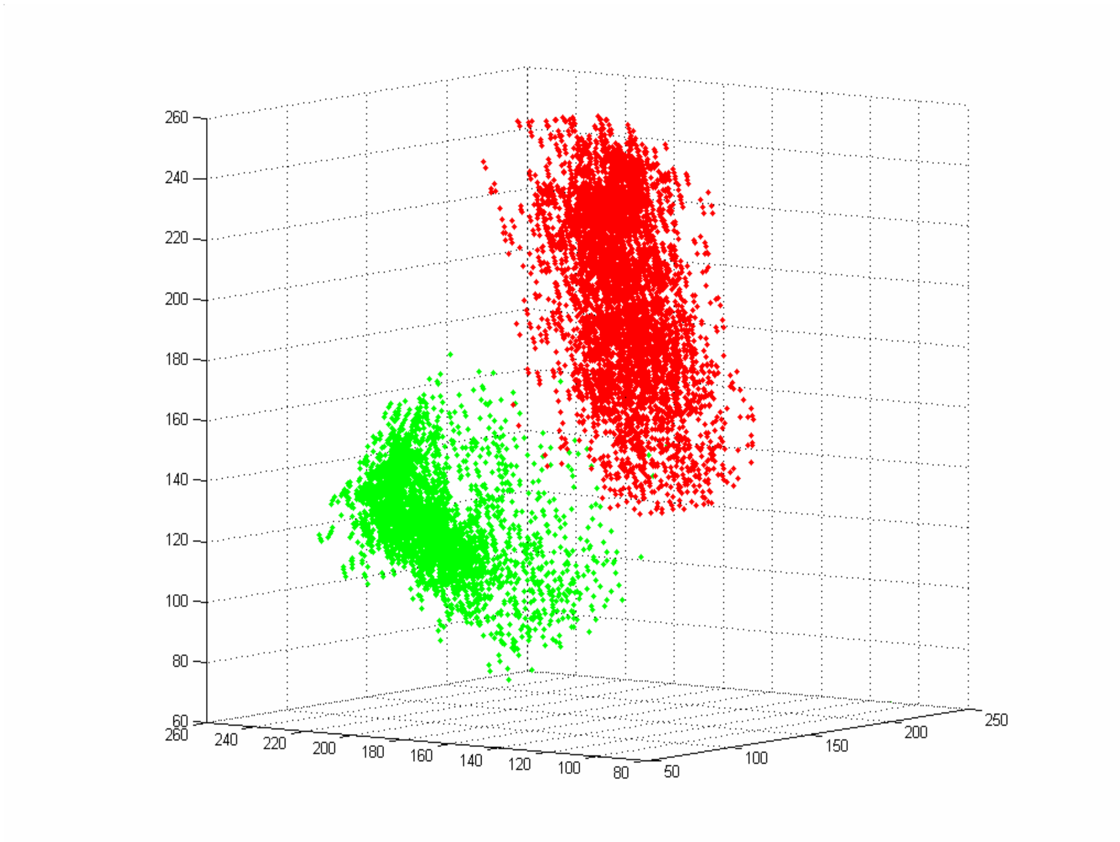


Figure 2 - Distribution of red and green pixels in RGB color space after training the system and classifying each pixel.

III.II KINEMATICAL MODEL

The manipulator used in this project is a simple two-link manipulator shown in Figure 3.

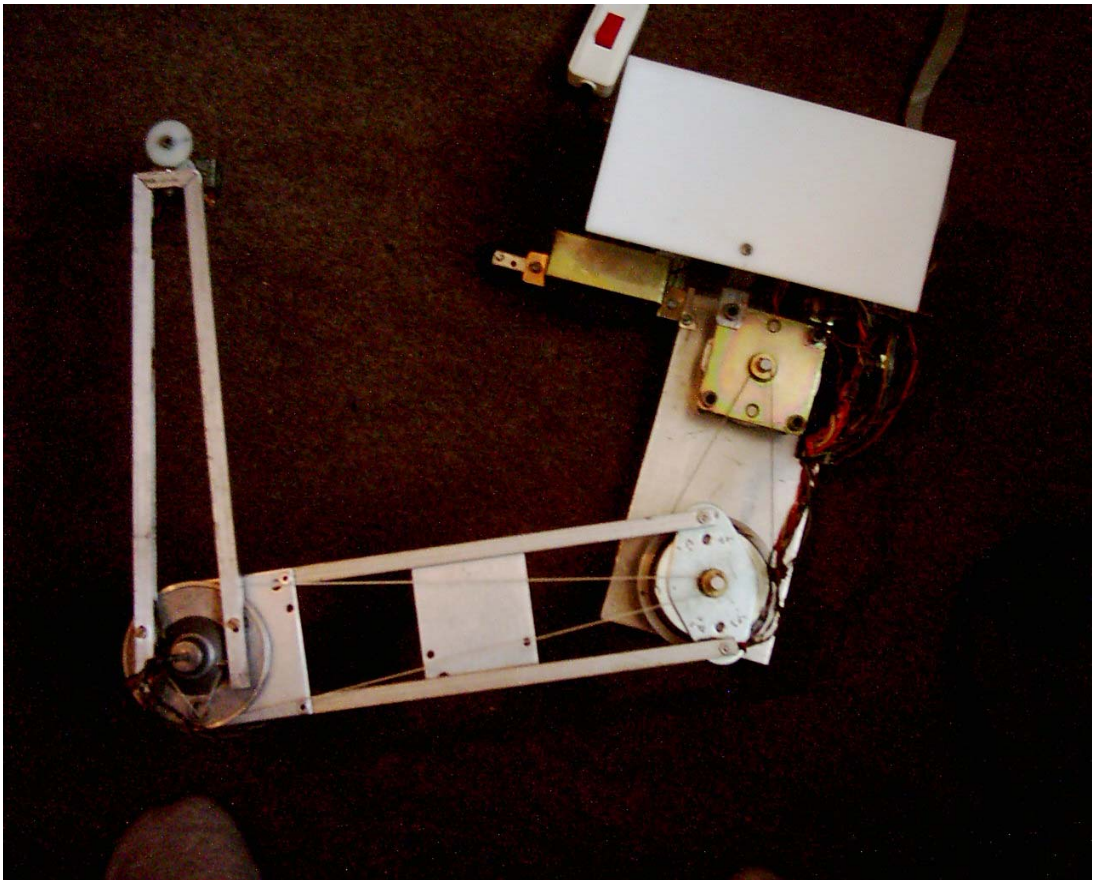


Figure 3 - Simple two-link manipulator used in this project.

For constructing the kinematical model of this manipulator, it is simplified to two thin connected lines. The kinematical model of the manipulator is constructed by using the law of cosines and law of sinus. This simplified model and calculations done for finding the joint angles are shown in Figure 4.

$$c^2 = a^2 + b^2 - 2ab\cos\alpha$$

$$(x^2 + y^2) = I_1^2 + I_2^2 - 2I_1 I_2 \cos\Theta_2$$

$$\cos\Theta_2 = \frac{I_1^2 + I_2^2 - x^2 - y^2}{2I_1 I_2}$$

$$I_1 = I_2 = I$$

$$\Theta_2 = \arccos(\frac{2I^2 - x^2 - y^2}{2I^2})$$

$$\frac{\sin(\Theta_1 - \alpha)}{I} = \frac{\sin\Theta_2}{\sqrt{x^2 + y^2}}$$

$$\Theta_1 = \arcsin(\frac{I\sin\Theta_2}{\sqrt{x^2 + y^2}}) + \alpha$$

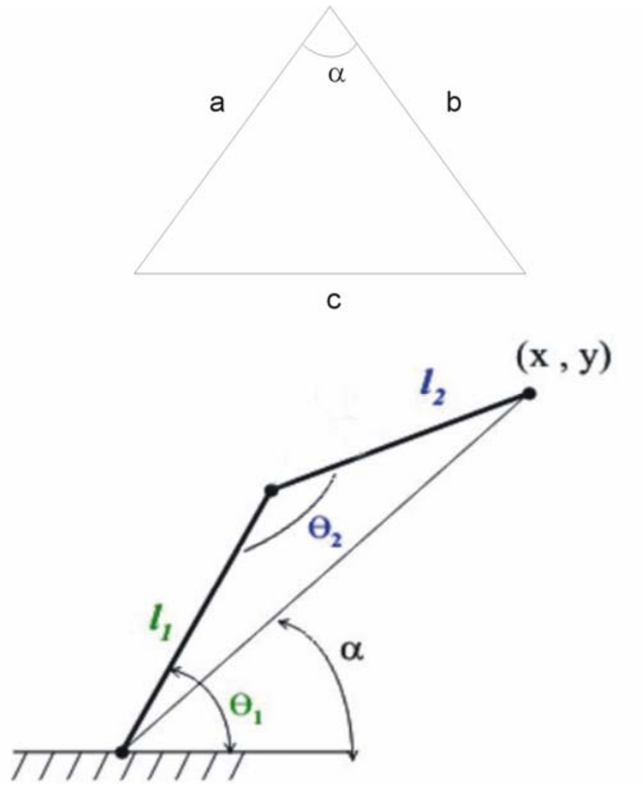


Figure 4 - Kinematical model of the manipulator.

For applying these calculations to the project, another application, which is called armSimulator, was developed. This application shows the joint angles and corresponding arm posture according to the coordinates of the mouse pointer. A snapshot from this application is shown in Figure 5.

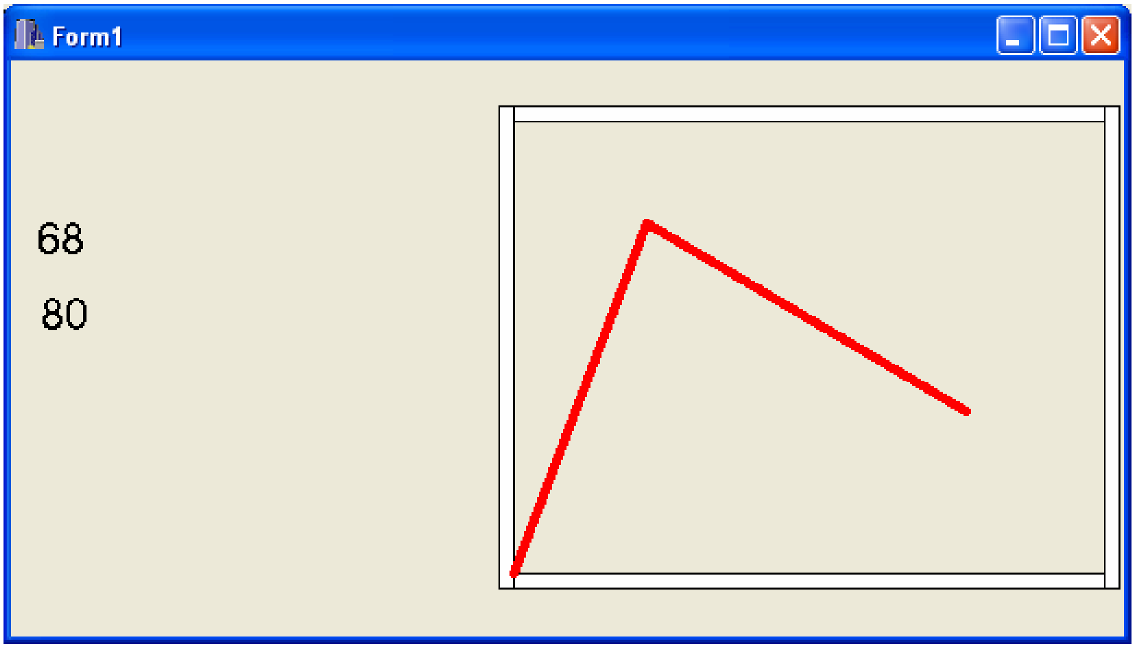


Figure 5 – A snapshot from armSimulator program.

III.III STEPPER MOTOR CONTROL & PARALLEL PORT USAGE

A stepper motor is a permanent magnet or variable reluctance dc motor that has the following performance characteristics:

1. rotation in both directions,

2. precision angular incremental changes,

3. repetition of accurate motion or velocity profiles,

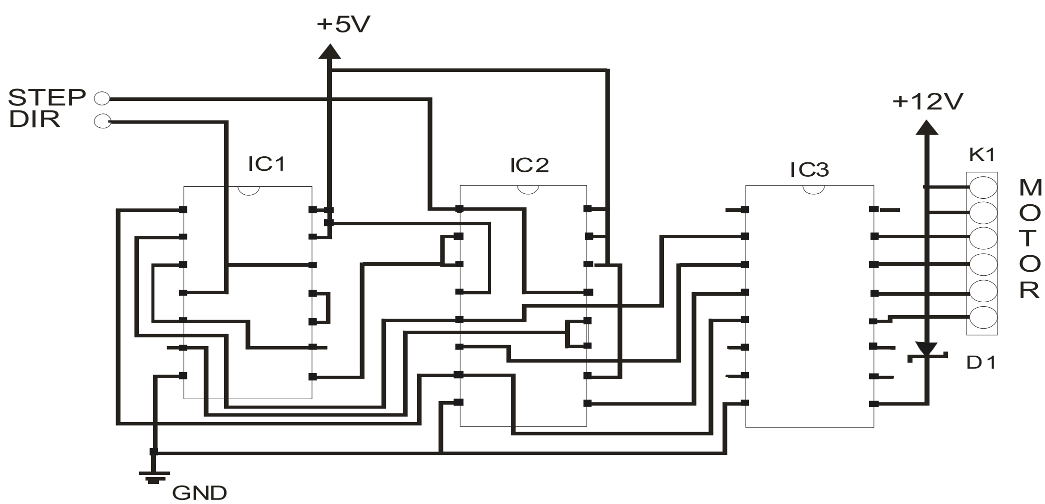
4. a holding torque at zero speed, and

5. capability for digital control.

A stepper motor can move in accurate angular increments known as steps in response to the application of digital pulses to an electric drive circuit from a digital controller. The number and rate of the pulses control the position and speed of the motor shaft. Generally, stepper motors are manufactured with steps per revolution of 12, 24, 72, 144, 180, and 200, resulting in shaft increments of 30, 15, 5, 2.5, 2, and 1.8 degrees per step.

Stepper motors are either **bipolar**, requiring two power sources or a switchable polarity power source, or **unipolar**, requiring only one power source. They are powered by dc current sources and require digital circuitry to produce the coil energizing sequences for rotation of the motor. Feedback is not always required for control, but the use of an encoder or other position sensor can ensure accuracy when it is essential. The advantage of operating without feedback is that a closed loop control system is not required. Generally, stepper motors produce less than 1 horsepower (746W) and are therefore frequently used in low-power position control applications.

In this project, a stepper motor controller/driver unit is used for decreasing the number of pins used for one motor from 4 to 2. The circuit diagram of this controller unit is shown in Figure 6.



IC1 74LS86
IC2 74LS112
IC3 ULN2003
D1 Zener 15V, 1W

Figure 6 - Contoller circuit diagram. The controller makes it possible to control a stepper motor with two pins; STEP and DIR.

The original IBM-PC's Parallel Printer Port had a total of 12 digital outputs and 5 digital inputs accessed via 3 consecutive 8-bit ports in the processor's I/O space.

- 8 output pins accessed via the **DATA Port**
- 5 input pins (one inverted) accessed via the **STATUS Port**
- 4 output pins (three inverted) accessed via the **CONTROL Port**
- The remaining 8 pins are grounded

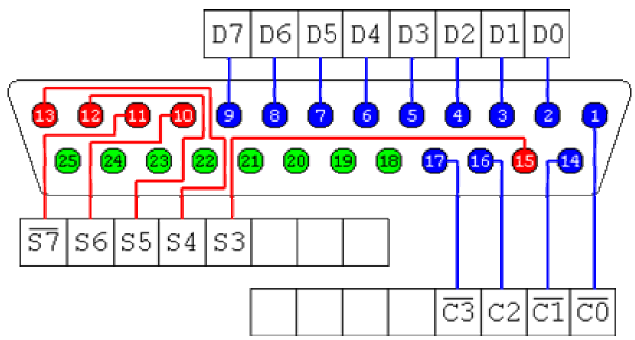


Figure 7 - 25-way Female D-Type Connector

In this project, only 7 data pins are used such as:

- D0 is for step control of the first motor.
- D1 is for direction control of the first motor.
- D2 is for step control of the second motor.
- D3 is for direction control of the second motor.
- D4 is for magnet carrying motor – up.
- D5 is for magnet carrying motor – down.
- D6 is for turning on/off the magnet.

III.IV ALGORITHM

Pseudo code of the algorithm used in the project is given below.

```
    GrabFrame          // grab one frame from the camera
    TrainTheSystem     /* train the system for red and green regions in the
                       grabbed image by clicking the desired colored
                       pixels.*/
    FindPieces ()
    Start
    For i = 1 to number of pieces {
        GoToXY (piece[i].x , piece[i].y)
        FindArmHead()
        If (distance from armHead center to piece center > threshold) {
            While (distance < threshold) {
                GoToXY (piece[i].x , piece[i].y)
                FindArmHead()
            }
        }
        Motor3.down()
        Magnet = ON
        Motor3.up()
        GoToXY (destination.x , destination.y)
        FindArmHead()
        If (distance from armHead center to destination center > threshold) {
            While (distance < threshold) {
                GoToXY (destination.x , destination.y)
                FindArmHead()
            }
        }
        Motor3.down()
        Magnet = OFF
        Motor3.up()
}
```

IV CONCLUSION

This project was helpful about learning general stepper motor control concepts, some image processing and pattern classification algorithms, inverse kinematics calculation techniques, and parallel port usage. The results obtained when the runs are satisfactory.

V REFERENCES

Linda G. Shapiro, George C. Stockman. Computer Vision. Prentice Hall, 2001.

Wolfram Stadler. Analytical Robotics and Mechatronics. McGraw – Hill Series in Electrical and Computer Engineering. 1995.