GOVP1200203708

635.63
L293ㅅ
19

최 종
연구보고서

시설재배용 과채류(오이) 수확기 개발

The Development of Fruit-Vegetables(Cucumber) Harvester in the Greenhouse

연 구 기 관
성 균 관 대 학 교

농 림 부

제 출 문

농림부 장관 귀하

본 보고서를 "시설재배용 과채류(오이) 수확기 개발" 과제의 최종보고서로 제출합니다.

2001 . 10 . 31 .

주관연구기관명 : 성균관대학교
총괄연구책임자 : 이 대 원
협동연구기관명 : 건국대학교
협동연구책임자 : 성 시 흥

요 약 문

Ⅰ. 제 목

시설재배용 과채류(오이) 수확기 개발

Ⅱ. 연구개발의 목적 및 중요성

농산물 수입개방으로 외국산 농산물의 대량 유입은 우리 농업의 기반을 흔들고 있는 실정이다. 따라서 농산물의 고품질화 및 생산비 절감으로 대외 경쟁력을 높일 수밖에 없는 실정이다. 이를 위해서 농산물의 적기 수확은 재배기술 못지 않게 중요하다.

대부분의 과채류 수확은 현재 수작업으로 이루어지고 있다. 지상(地上) 1m 내외에서 결과(結果)하는 과채류의 경우 수작업시 많은 노동력을 필요로 한다. 특히 오이의 경우 연속된 단순 반복 작업으로 작업인의 피로도는 더욱 증가될 수밖에 없는 실정으로 오이 재배에 많은 어려움을 낳고 있다. 사람이 오이의 수확 작업을 하는 경우에는 과실 길이가 200 mm정도의 과실을 대상으로 하며, 한 손으로 과경 바로 아래 부분인 과실 상단 근처를 잡고 과경의 절단 위치를 눈으로 보거나 손으로 감지하여 다른 한 손으로 절단한다. 이러한 작업을 자동적으로 수행 할 수 있는 로봇 개발이 필요하다. 농업생산물의 경우 품종에 따라 약간의 차이는 있지만, 수확시기에 노동력 투하시간이 가장 많다. 과채류 중에서 오이의 경우도 마찬가지로 수확을 위해서 많은 노동력을 요구한다. 또한 최근의 농업 노동력은 현저하게 감소하고 있으며, 따라서 오이를 포함한 과채류의 적기 수확을 위해서 수확작업의 자동화가 절실히 요구되고 있으며, 이를 위해 본 연구에서는 최근 전기전자기계 기술을 기본으로 하는 메카트로닉스기술을 이용하여 오이수확을 위한 생력화된 시스템의 개발을 목적으로 하고

있다.

Ⅲ. 연구개발 내용 및 범위

본 연구에서는 오이수확을 위한 구동시스템, 그리퍼 및 엔드이펙터시스템, 매니퓰레이터시스템 영상처리시스템 및 종합시스템을 개발하였다. 각각의 시스템을 조합한 종합시스템의 현장 수확실험 결과 비교적 양호한 결과를 얻었다. 그러나 개발한 시스템만으로는 오이수확을 위한 현장보급이나 상품화에는 더 많은 연구가 필요하다. 따라서 본 연구에서 개발한 자동 오이수확기의 경우, 수확 작업외의 다른 단위작업에 대한 기능이 추가된 시스템으로 보완 발전되어야 농가에서의 수용이 가능할 것으로 판단되었다. 본 연구의 개발 내용을 구체적으로 살펴보면 다음과 같다.

1. 구동시스템

가. 오이수확기 개발을 위해서 먼저 작업기가 임의의 공간으로 이동하기 위해서는 구동시스템이 필요하다. 이를 위해 궤도형 구동시스템과 대차형 구동시스템 및 유인형 구동시스템을 설계 제작하여 각각의 성능에 대해서 분석하였다.

나. 시설재배공간에서 사용하고 있는 작업기를 무인으로 동작하도록 하기 위한 궤도용 제어시스템을 개발하였다. 이 시스템은 실제 온실 구조의 크기보다는 작게 설계제작하기 위하여 모델화 하였다. 작업기가 무인주행을 할 때 가장 정확하게 위치를 검출하기 위하여 고정경로 방식을 설정하였으며, 고정경로 방식의 단점인 경로의 한정성을 탈피하기 위하여 X-Y테이블 구동시스템으로 설계 제작하였다. 이 시스템은 시설재배용 천장레일에서 이동하면서 평평한 온실에서 작업기가 농산물을 취급하기에 적합하게 설계 제작한 적정한 구조로

판단된다.

다. 제자리 회전이 가능한 대차형 구동시스템의 경우 경로의 이동이 자유롭고, 작업공간이 좁은 시설내에서 작업이 원활하게 수행할 수 있었다. 그러나 동력 전달을 위한 모터의 구동에 한계를 가지고 있었으며, 바닥면의 불규칙함에 따른 기타 연계시스템의 공간상에서의 위치제어의 정확도에 한계를 가지고 있었다.

라. 유인형 구동시스템의 경우, 리미트 스위치에 의해 회전 설정 프로그램은 횡축과 종축의 경로 설정 가이드가 교차하는 지점에서 리미트 스위치로부터 전기적인 신호를 받으면 회전할 수 있도록 C 언어를 사용하여 구현하였다. 경로 설정 가이드에 따른 직선 이동 실험에서 지면 상태에 관계없이 전체 성공률이 54.4%로 측정되었다. 유인형 구동시스템의 경우, 작업 성능면에서는 양호하다고 할 수 있으나, 상부의 많은 방해물에 의해 온실내에서 적절한 작업을 수행할 수 있는 수확기에 응용하기에는 어려움이 있을 것으로 판단하였다.

2. 그리퍼, 엔드이펙터 시스템

가. 오이수확기에서 오이를 파지하고 수확하기 위해서 그리퍼 및 엔드이펙터를 개발하였으며, 이를 위해 오이의 물성측정을 분석하였다. 이를 바탕으로 수확기의 엔드이펙터 설계 및 제작에 이용하였다. 또한 수확기를 위한 엔드이펙터을 직접 설계 제작한 후 성능실험을 수행하였다.

나. 오이의 수확기 개발에 있어서 그리퍼 및 엔드이펙터의 정밀한 설계를 위해서 오이의 물성을 측정한 결과 함수율의 경우 다른 과채류와 비슷하게 나타났다. 경도를 포함한 물성값을 이용하여 수확기 개발의 기초자료로 활용할 수 있었다.

다. 그리퍼 및 엔드이펙터를 이용하여 작업성능시험을 행한 결과, 오이의 수확을 위해 절단과 파지을 하는데 소요되는 시간을 측정하였으며, 일분당 8.58개를 수확할 수 있었으며, 수확성공율은 42.7%이었다.

3. 매니퓰레이터 시스템

가. 오이 수확기의 매니퓰레이터를 개발을 위해 오이 수확에 적합한 링크구조와 메커니즘으로 매니퓰레이터와 모터 컨트롤러를 설계하였다. 본 매니퓰레이터는 DC모터로 작동되는데, 작동원리는 퍼지 논리이론을 적용하여 부드러운 작동이 가능하도록 하였다.

나. 매니퓰레이터의 사양은 3축 PRR(prismatic revolute revolute)형 매니퓰레이터이다. 그리고 각각의 모터는 회전관성모멘트가 실리지 않는 부분에 장착되고 동력은 타이밍 벨트를 이용하여 전달하는 방식을 취하였다. 이렇게 함으로써 매니퓰레이터의 작동성능을 높였고, 내구성 또한 증가되도록 하였다.

4. 영상처리시스템

가. 오이수확로봇 개발에 있어서 정확한 오이의 형상 및 위치를 인식하기 위하여 형상인식 알고리즘에 대한 연구를 목적으로 행하였다. 다양한 오이형상을 인식하기 위한 방법으로는 신경회로망의 연상메모리를 이용하여 오이의 특징형상을 인식하였다. 또한, 오이의 영상정보를 구별하기 위한 영상처리방법으로는 영상 이치화와 경계선 분할 추출에 대한 연구를 수행하였다. 오이의 형상인식은 이치화 영상을 직접 이용한 방식으로 수행하였다. 또한, 학습패턴에 대한 출력패턴을 조건화시킴으로서, 학습패턴과 유사한 출력패턴만을 검출하도록 알고리즘을 구현하였다. 형상인식은 실제영상에서 오이의 형상과 위치를 판정

할 수 있도록 알고리즘을 개발하였다.

나. 영상 알고리즘에서는 일정한 학습패턴의 수를 2개, 3개, 4개를 각각 기억시켜 샘플패턴 20개를 실험하여 연상시킨 결과, 학습패턴으로 복원된 출력패턴의 비율은 각각 65.0%, 45.0%, 12.5%로 나타났다. 이는 학습패턴의 수가 많을수록 수렴할 때, 다른 출력패턴으로 많이 검출되었다.

다. 오이의 특징형상 검출은 30×30간격으로 자동검출 되도록 처리하였다. 실제 영상에서 자동 검출로 처리한 결과, 오이인식의 처리시간은 약 0.5~1초/1개(패턴) 빠르게 검출되었다. 또한, 다섯 개의 실제 영상에서 실험한 결과, 학습패턴에 대한 다른 출력패턴은 96~99%의 제거율을 나타내었다. 오이로 인식된 출력패턴 중에서, 오검출된 출력패턴의 비율은 0.1~4.2%를 나타내었다.

라. 신경회로망을 이용하여 오이의 형상 및 위치를 인식할 수 있도록 알고리즘을 개발하였다. 오이의 위치측정은 실제영상에서 학습패턴과 유사한 출력패턴의 좌표를 가지고, 오이의 위치좌표를 추정할 수 있었다.

5. 종합시스템

가. 오이 수확을 위한 종합시스템의 설계 제작하였으며, 이를 이용하여 현장에서의 오이 수확실험을 수행하였다. 또한 각 구성요소 중에서 수확에 직접 영향을 주는 매니퓰레이터의 성능을 검증하였다. 작업기는 오이 수확에 적합한 링크구조와 메커니즘를 설계하였으며, 본 매니퓰레이터는 모터로 작동되는데, 작동원리는 퍼지논리 이론을 적용하여 부드러운 작동이 가능하도록 하였다.

나. 전체 수확을 시도한 오이 중에서 42개는 수확이 가능하였으며, 6개는 파지 및 절단이 불충분한 결과이었으며, 2개에 대해서는 잎과 줄기 등에 의한 인

식의 불충분 및 파지와 절단등이 복합적으로 작용하여 수확이 불가능하였다.

다. 3차원 공간상의 좌표에 대하여 매니퓰레이터의 10회 반복 위치 오차는 Z 축에 관계없이 0.1mm 내외로 정확하게 작동하는 것으로 나타나 수확작업에 적합한 것으로 나타났다.

라. 각 축을 제어함에 있어 동력원으로 모터를 사용하였다. 모터는 제어가 용이한 장점이 있다. 본 연구에서는 퍼지제어 이론을 적용 작업환경에 맞게 rule base를 작성하여 대응하도록 한 결과 비교적 정확한 위치제어가 가능하였다.

Ⅳ. 연구개발결과 및 활용에 대한 건의

본 연구에서는 오이 수확을 위한 로봇 개발을 하였다. 엔드이펙터, 매니퓰레이터, 영상장치 및 구동시스템이 현장 실험을 통하여 비교적 양호한 결과를 얻었다. 그러나 현장 보급 및 상품화를 위한 완전한 수확기 개발을 위해서는 작업환경에 존재하는 덩쿨손, 측지, 하위엽 및 기형과을 제거하는 기술이 수반되어야 할 것이다. 본 연구에서 개발한 내용을 기초로 하여 작업환경에 존재하는 문제점을 제거하는 기술이 병행되어야 할 것으로 판단된다.

SUMMARY

Ⅰ. TITLE

The Development of Fruit-Vegetables(Cucumber) Harvester in the Greenhouse

Ⅱ. Objectives and importance of development

Liberalization of imports for agricultural products made Korean agriculture weak. It is necessary for international competitiveness that agricultural products should be high quality and low-cost product. So, harvesting in time agricultural products is an important thing as a method of cultivation.

Most harvesting operation of fruit-vegetables is manual operation. fruit-vegetables which grows less than 1 meter from ground and needs lots of labors to harvest them. Specially, harvesting cucumber is simple but repetitional and boring operation which workers are easily tired. Harvesting a cucumber, the size of a cucumber of approximately 200mm. One hand graps its stem, the other hand cuts the throat between its fruit and its stem. This operation must be processed both robot manipulator and end-effector. Also the cucumber is required the more time in harvesting than any other operation. Since labor is decreasing recently, so automated harvesting is very important to harvest in time. So, in this study, with mechatronics technology which is based on both electricity and machinery technique, the robotic system for saving of labor system was designed and constructed.

Ⅲ. Contents of development

Harvesting system of cucumber proved to be reliable system for recognizing the position of a cucumber and cutting and gripping its cucumber. Its development involved the integration of a computer vision system and end effecters along with an PC computer. Software, written in visual C++, combined the functions of image capture, image processing, and control into programs. Two separate programs, one for the end effector, and one for image processing, were included in the software.

1. Operating system

In this study, a traveling control system was developed to transfer a machine without an operator in the working zone. The dimension of the system was modelized to design and construct smaller than that of real configuration of a greenhouse. For this system, the fixed path type was used to detect exact position during operating a manless machine, and the X-Y table actuator type to escape a unique path, which had the disadvantage in a fixed path type environment.

Based on the results of this research the following conclusions were made :

A. This system used two screws to move toward horizontal direction, and a plate to reach at any points in the working zone.

B. The software combined the functions of path selection and motor operation to control into one program. The path selection program was a menu driven

program written in Visual Basic, and the motor operation program was written in Borland C++ for actuating motors.

C. The path-select mode of the program was used by selecting the desired paths, and the user path-create mode by selecting a random path in the path-selection program.

D. The system proved to be a reliable system for operating a manless machine, since accuracy and precision to reach the positions were less than 1%.

Agricultural machine is currently operated by man power in the greenhouse, which is oppressively hot and humid, and is for a farmer not to work in comfortable circumstances. In the future, agricultural machine will not have to operate by man power, but it will need do by unmanned power. In order to put into the automatic and unmanned operation of agricultural machine, this system was designed and built to move through the fixed path in the greenhouse.

This system was composed of wires, a limit switch, an operating equipment, its software for automatizing a machine in the greenhouse. The wire was connected between the wall pillars, and the equipment was able to slide over the fixed path made of the wire, by rectilinear and rotational motion. Model cart was developed with a stepping motor to calculate on the success rate ot its operation with the system

As might be expected, this system with model cart was moved the paths with a success rate of 100% on the flat plane surface in our laboratory. However, on the flat sand plane or the other material plane, the success rate was not better than 80%. If the cart were well operated, the

success rate was would be 100%. Based on the results of this research, this system would be expected to operate well on the path made of a simple wire.

2. Gripper and end-effecter system

A. Gripper and end-effecter system were developed for grip cucumber and harvest. For this experiment, cucumber's physical properties were analyzed. Design and production of end-effecter for cucumber harvester was provided for those analysis.

B. Cucumber's properties were measured for precision of gripper and end-effecter. The results were similar to those of other vegetables. Properties including hardness of cucumber were used as basic data for development of harvester.

C. Operation efficiency of gripper and end-effecter was tested during experiment. Harvester was operated by griping and cutting. The results were recorded by time. The harvester could harvest 8.85 cucumber per minute and success rate was 42.7%.

3. Manipulator system

This study developed a manipulator for robotic harvester to harvest cucumber. The manipulator was designed and built for transferring an end-effector from a fixed point to a specified cucumber. Its development involved the integration of a manipulating system with a PC compatible,

DC motors, geared boxes, timing belts, and a motor controller board. Software, written in Quick basic, combined the functions of motor control with various circumstances. In order to move smoothly and rapidly the manipulator, it's shoulder link and elbow link were minimized by using rotational inertial moment without a motor and a geared box.

After 30 replications of exercising the manipulator, it was concluded that the precision values of the X, Y and Z axes were less than 0.5mm, 0.25mm and 0.35mm, respectively. The precision data indicated the manipulator was not missing any steps for the harvester to reach a target cucumber.

4. Image processing system

In this research, the algorithm pattern recognition for development of a robotic harvester has been conducted to find the shape and position of a cucumber fruit from an image in and out of the working zone. The image processing methods were performed to recognize the shapes and patterns of various kinds of cucumber in the greenhouse by using an associative memory of neural network. The thresholding and edge segmentation was used to enhance and extract the images of a cucumber fruit against the background.

Pattern recognition of a cucumber was conducted to detect directly the binary images by using thresholding method, which has the threshold level at the optimum intensity value. By restricting conditions of learning pattern, output patterns could be extracted from the same and similar input patterns by the algorithm. The algorithm of pattern recognition was developed to determine the position of the cucumber from a real image within working condition.

This study was developed the algorithm to recognize the 3D positions of a cucumber using neural network. The position of a cucumber fruit could be measured the output position which would be the same position of a sample pattern position.

5. Integrated system (Cucumber Harvester)

A. the integrated system, which harvests a cucumber, was designed and constructed. Performance test of the fully integrated system, conducted using the end-effector , manipulator and the imaging processing system. Motors, which are easily controled, were used to control each axis of the manipulator by using fuzzy control method.

B. Of a total of 42 cucumber fruits harvested, 6 fruits were insufficiently cut or grasped and 2 fruits them were not found the position which was cut or grasped by using the image processing methods.

C. In 3-D coordinates, ten replications of measuring the errors within 0.1mm regardless of z-axis were completed. These errors show that the manipulator is adequate for harvesting operation.

Ⅳ. The result and utilization of development

In this study, the robotic system for harvesting cucumber was developed. End-effector, manipulator, image processing system and controling system showed good performance. Based on the results of this research the following recommendations are made for further study: [1]

First, besides harvesting its fruit, its the oldest leaves, creeper and the youngest small side leaves are needed to be removed. [2] Second the its stem are needed to be fixed the line, which is used to for cucumber to be fixed. [3] Third, its creeper are needed to be removed. [4] Finally a robotic system with several operations could be developed for field working operations.

CONTENTS

목 차

제 1 장 서 론

제 1 절 서 설

노동 집약적이던 농업은 급속한 기계화로 인하여 노동력의 절감은 여러 분야에서 이루어지고 있지만, 아직까지도 농업은 수익성이 높지 않아 여전히 사람들로 하여금 노동을 기피되고 있는 산업중의 하나이다. 온실 자동화는 특용 작물 등의 고부가가치 상품에만 한정되어 있으므로 이를 개선해야 한다.

대부분의 기계 및 공정 등이 현재 점차적으로 무인자동화로 발전되고 있으며, 작업의 편리성을 강조하고 있다. 앞으로의 농업기계는 무인자동화 시스템의 도입으로 인하여 누구나 손쉽게 작업설정을 하고 기계 스스로 원활하게 작업을 수행할 수 있도록 개선되어야 한다. 온실내의 무인 자동화, 원활성, 편리성에 대한 연구가 필요한 실정이다.

본 연구에서는 주행경로의 설정이 용이하고 제어가 수월한 고정 경로 방식을 이용한 주행 시스템의 개발과 경로 설정 가이드를 지면에 설치할 경우에는 사람 또는 다른 작업기의 이동시에 불편을 초래하고, 마찰 등에 의한 파손의 문제가 발생할 수 있다. 따라서 지면 설치가 아닌 온실 상부에 설치하여 작업기를 구동시키는 시스템을 개발하고, 온실의 형태, 고랑의 위치 등에 무관하게 주행 경로의 설정을 제어할 수 있는 시스템을 개발하고, 경로설정 가이드와 주행 경로 고정 시스템을 이용하여 자동화 시스템을 개발하였다.

현재 엔드이펙터를 이용하여 수확기 개발을 시도하고 있으며, 부분적으로 상당한 수준에 도달한 분야도 있다. 엔드이펙터를 이용한 작업은 지루한 작업이 반복적으로 수행되는 어렵고 힘든 작업에 이용되고 있다. 농업 분야에도 단순 자동제어 시스템으로 해결될 수 없거나 복잡하고 지루한 반복작업이 지속되는 분야에서 농업용 엔드이펙터 개발이 필요한 실정이다.

이중에서 로봇의 엔드이펙터를 개발 할 때, 중요하게 고려할 요소는 작업할

대상물의 물리적 특성이다. 작업할 대부분의 농작물은 생명을 가지고 있지만, 산업용 대상물은 생명을 가지고 있지 않다는 점을 인식하여야 한다. 그러므로 일반적으로 공장에서 사용하는 산업용 로봇의 엔드이펙터의 몇 가지 기능만을 변화시켜 농업용 엔드이펙터로 이용할 수 있는 농업용 대상물은 제한되어 있다. 즉, 공업용 대상물은 대부분 정밀 작업이 요구되는 견고한 강체이지만, 농작물은 이동하거나 충격을 가하면 상처를 쉽게 받는 유연한 생명체이다. 그러므로 이러한 점들을 고려하여 엔드이펙터 개발을 하는 것이 필요하다.

대부분의 과채류 수확은 현재 수작업으로 이루어지고 있다. 지상(地上) 1m 내외에서 결과(結果)하는 과채류의 경우 수작업시 많은 노동력을 필요로 한다. 특히 오이의 경우 연속된 단순 반복 작업으로 작업인의 피로도는 더욱 증가될 수밖에 없는 실정으로 오이 재배에 많은 어려움을 낳고 있다. 사람이 오이의 수확 작업을 하는 경우에는 과실 길이가 200 mm정도의 과실을 대상으로 하며, 한 손으로 과경 바로 아래 부분인 과실 상단 근처를 잡고 과경의 절단 위치를 눈으로 보거나 손으로 감지하여 다른 한 손으로 절단한다. 이러한 작업을 자동적으로 절단할 수 있는 엔드이펙터의 개발이 필요하다.

시설원예에서 소요되는 노동력 중에서 가장 많은 부분을 차지하는 부분 중에 하나가 바로 수확작업이다. 시설원예 작물 중에서 오이의 수확작업은 많은 노동력이 투입되고 있다. 또한 시설원예에서 수확되는 오이의 생산량 및 수출 현황은 '97년에 IMF로 인하여 감소하였지만, 농림부의 농산물생산통계에 의하면 '99년 노지오이의 재배면적이 1,728ha이고, 시설오이 재배면적은 5,964ha이다 (농산물생산통계, 농림부, 2000). 이는 시설오이의 재배면적이 약 3.5배정도 많은 면적이며, 시설내 재배환경에 대한 생력화가 필요하며, 이에 따른 과다 노동력 투입을 해소하기 위해서는 오이수확작업의 자동화가 절실하다고 할 수 있다. 오이 수확작업의 자동화를 위해서는 먼저 작업기의 이동이 선행되어야 하며 이를 위해 X-Y테이블형 궤도시스템을 이용한 작업기의 접근성능에 관한 연구가 보고되었다(Lee et al. 1998). 3차원 시각을 센서를 이용한 미니토마토의 수확로봇에 대한 연구도 있었으며, 이 연구에 이용된 매니퓰레이터는 4자유

도를 가지고 있으며, 그리퍼는 흡입력을 이용하였으며, 3차원시각센서를 이용한 수확 실험에 유효하다고 보고하였다(Subrata et al. 1998).

오이 수확작업의 자동화를 이루기 위해서 필요한 기술적 방법은 다른 작물에 비하여 어렵다. 왜냐하면 첫째 오이는 한번 심은 줄기로부터 계속해서 오이가 열리고 연속적으로 관찰하여 수확하여야 하고, 둘째 넝쿨식물로써 줄기는 가늘고 잎은 크다는 특징이 있기 때문이다. 즉 오이 수확 작업 시 오이 줄기의 손상은 바로 오이 수확량의 감소로 이어지고, 잎이 크기 때문에 수확하려는 오이를 쉽게 구별해 내기가 어렵다(Lee et al. 1998). 이렇게 어려운 수확작업을 자동화하기 위하여 로봇 수확기가 필요한 실정이다. 이 로봇 수확기에 필요한 기술은 크게 4가지로 볼 수 있다.

첫째 오이를 인식하는 영상처리 시스템 개발이고 둘째는 위치가 파악된 오이를 파지(把持)하고 과경(果柄)을 절단하는 엔드이펙터의 개발이고 세 번째는 엔드이펙터를 오이에 근접하도록 이동시키고 수확된 오이를 이동시키는 매니퓰레이터의 개발이다. 마지막 네 번째는 수확된 오이를 이동시키는 자동대차이다. 본 연구는 앞에서 언급한 기술적 요소 가운데에서 오이 수확작업에 적합한 매니퓰레이터의 개발을 목적으로 하고 있다.

시설재배용 과채류 수확 로봇 개발은 우리나라 뿐 만 아니라, 전세계적으로 활발히 연구중이지만, 아직 한정적인 성능시험 정도로서 실용화 단계에 있지는 않다. 특히, 과채류 수확기 개발을 위한 영상처리부는 3차원 공간에 놓여 있는 과일의 개별적인 위치를 선별 및 인식하여야 하는 어려움이 있다. 영상처리 시스템을 이용하여 과일이나 과채류 등의 인식과 선별에 대한 효율성을 증가시키고 있지만, 과채류 영상은 대부분 복잡한 3차원 구조의 군락을 이루고 있어 적절한 영상분석이 매우 어렵다. 또한 주변의 광 조건에 의해 영상획득 및 보정 등의 과정이 요구되는 문제가 있다.

과채류 영상인식에는 카메라 외에 근적외선 , 레이저, 초음파 등 다양한 센서를 이용하여 3차원 영상처리 문제를 해결하는데 이용하고 있다. 또한 영상처리의 국한적인 방법에 지나치지 않고, 인공지능 분야인 신경회로망과 퍼지, 유

전자알고리즘 등의 장점을 살린 과채류 인식방법을 적용하고 있다. 이와 같은 자료는 아직까지 실용화하기에 미비한 상태이다.

오이수확 로봇 개발을 위한 영상처리 시스템은 수확할 대상체(오이)가 줄기나 잎의 색이 비슷하기 때문에, 칼라 카메라(Color camera)로는 과실을 식별하기가 불가능하다. 또한, 오이 영상은 대부분 잎에 의해 가려져 있기 때문에 오이에 대한 특징인자를 찾아내는 것이 현실적으로 어려운 실정이다.

그러므로 오이의 형상과 위치를 인식하기 위한 전단계 과정으로 신경회로망(Neural network)의 연상메모리(Associative memory)를 이용하여 오이의 형상정보를 인식하고자 한다. 신경회로망을 이용한 형상인식 기술은 학습패턴(Learning pattern)에 대한 통계적 모델 없이 직접 학습이 가능하고, 학습 패턴으로 대상체의 인식이 가능하다는 장점 때문에 강력한 패턴 인식 방법으로 인정받고 있다. 형상인식에 대한 연구는 학습패턴에 대한 출력패턴의 연상능력과 검출조건을 분석한 후 자동검출(Auto scanning)로 처리하여, 실제영상에서 오이의 형상과 위치정보를 인식하고자 한다.

농업관련분야의 신경회로망 이용은 센서의 잡음제거 및 일괄적 처리, 영상처리에 의한 작물의 인식 등 변동하는 작업환경에 대한 해결능력을 갖고 있다. 그러므로 본 연구에서는 오이수확기 개발을 위한 대상체의 인식 알고리즘에 신경회로망을 이용하여 오이인식의 가능성 여부를 구명하고자 한다.

제 2 절 연구개발의 목표 및 내용

1. 연구개발의 목표

농업생산물의 경우 품종에 따라 약간의 차이는 있지만, 수확시기에 노동력 투하시간이 가장 많다. 과채류 중에서 오이의 경우도 마찬가지로 수확을 위해서 많은 노동력을 요구한다. 또한 최근의 농업 노동력은 현저하게 감소하고 있으며, 따라서 오이를 포함한 과채류의 적기 수확을 위해서 수확작업의 자동화가 절실히 요구되고 있으며, 이를 위해 본 연구에서는 최근 전기전자기계 기술을 기본으로 하는 메카트로닉스기술을 이용하여 오이수확을 위한 생력화된 시스템의 개발을 목적으로 하고 있다.

2. 연구개발의 내용

가. 구동시스템

- 구동시스템의 기초연구
- 오이 성장에 따른 구동시스템의 적합성
- 재배양식에 따른 구동시스템 모델화
- 수확기의 크기에 대한 구동시스템 모델화
- 구동시스템의 설계 및 제작
- 구동시스템의 제어장치 제작
- 구동시스템의 안정도 측정
- 구동시스템의 실험 및 분석

나. 그리퍼, 엔드이펙터 시스템

- 시스템의 시작기 제작 및 기초실험
- 제어시스템 제작

- 절단부 설계 및 제작
- 파지부와 절단부의 부착
- 재배양식에 따른 시스템의 적응성 실험
- 온실에서 그리퍼 시스템의 실험 결과 분석

다. 매니퓰레이터 시스템

- 매니퓰레이터 시스템의 문헌조사 및 기초연구
- 구동장치 메카니즘 분석
- 재배양식에 따른 작동부의 기초설계
- 구동장치의 작동을 위한 프로그램 개발
- 기계장치 및 제어시스템 제작
- 위치 제어 알고리즘 개발
- 시작기 실험
- 온실에서 매니퓰레이터 시스템의 실험결과 분석

라. 영상처리시스템

- 영상처리 시스템의 문헌조사 및 기초연구
- 실내에서 대상물의 정적상태 인식 및 패턴인식
- 배경과 오이 인식 알고리즘 개발
- 오이 인식 및 좌표 추적을 위한 3차원 영상시스템 제작
- 원점으로부터 오이의 위치의 결정
- 영상처리시스템의 온실내 인식알고리즘 개발

마. 종합시스템

- 인터페이스된 종합시스템 설계 및 제작
- 종합 시작기의 실험과 분석
- 환경변화에 따른 영상처리 적응성 검토

오이 수확기의 최종 분석 및 평가

제 2 장 : 기초연구

제 1 절 서 론

농산물 수입개방으로 외국산 농산물의 대량 유입은 우리 농업의 기반을 흔들고 있다. 따라서 농산물의 고품질화 및 생산비 절감으로 대외 경쟁력을 높일 수 밖에 없는 현실에 와 있다. 이를 위해서 농산물의 적기 수확은 재배기술 못지 않게 중요하다. 농업생산액중에서 원예산업이 차지하는 비중이 계속증가하고 있으며, '95년말 현재 농업생산액의 38.9%를 점하고 있어 원예산업의 중요성이 커지고 있다. 이중 채소가 25.2%, 과수가 11.2%, 화훼가 2.0%를 차지하고 있고, 시설채소도 '90년 3.6%에서 '95년에는 8.5%로 급격히 성장하고 있다. 이러한 현상은 계속 이어질 전망이며, 원예산업의 발전은 수입개방으로 인한 경쟁력 약화를 극복할 수 있을 것으로 생각하며, 이를 위해 첨단기술의 접목을 통하여 농업생산물의 적기 수확이 필요 할 것으로 판단된다.

농산물의 적기 수확은 최근 농업인구의 변화 및 노동력을 감안한다면 결코 쉬운일이 아니며, 이를 위해서 농산물 수확에서의 자동화는 부족한 노동력을 해결할 수 있는 하나의 방법이며, 이를 기초로 궁극적으로는 농업생산시설의 무인화가 가능할 것으로 판단된다. 생물생산시설의 자동화를 위해서 먼저 다양한 생물의 생장 특상을 고려해 볼 때, 생장이나 수확에 영향을 미치지 않는다면, 생육방법을 자동화에 적합하게 인위적으로 변형하는 방법을 생각할 수 있다. D'Esnon(1985)은 유연한 손가락 8개가 전후로 180 °회전 할 수 있도록 되어있어 과경의 방향에 관계없이 대부분의 과실을 매니플레이터(Manipulator) 내로 들어오게 할 수 있도록 되어있다. 줄기와 잎이 팔의 이동에 장해가 되거나, 작동영역 외에 과실이 존재하는 경우는 자동적으로 팔을 끌어들인다. 近藤直[4]은 관절형 로봇으로서 그리퍼 끝이 플랜지에 시각부와 절단기를 가진 엔드이펙터를 부착한 것이다. 엔드이펙터의 경우 반원 링형의 절단기를 사용하여

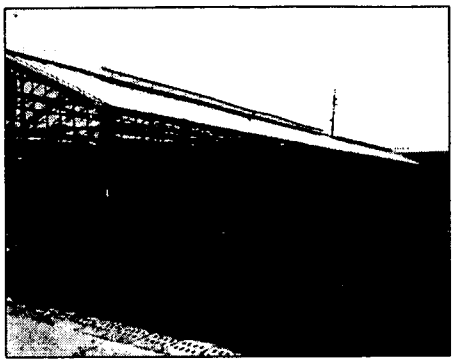
수확하도록 되어 있다. 그리퍼는 완만히 구부러진 그리퍼가 좌우로 부착되어 있고 과실을 미끄러지지 않도록 그리고 가능한 손상을 주지 않도록 손가락의 안쪽에 고무가 부착되어 있다.

따라서 본 연구에서는 오이수확기 개발을 위하여, 수확하고자 하는 오이를 보다 정확하게 인식할 수 있도록 영상처리 방법을 이용하며, 이를 위해서 영상을 통한 오이인식에 가장 많은 장애가 되는 잎을 단계적으로 제거하여 생육과 수확량에 차이와 낙과수를 관찰하여 보다 정확한 영상정보를 얻는데 기여하고자 한다. 이를 통하여 수확기의 보다 정확한 설계가 가능할 것으로 판단된다.

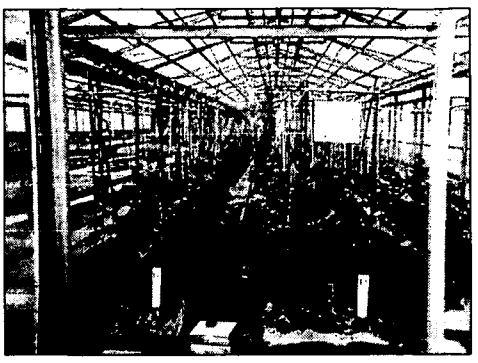
제 2 절 장치 및 방법

1. 실험장치

본 실험은 오이수확기 개발을 위해서 가장 중요한 기술 중에 하나는 오이인식을 위한 영상처리라 할 수 있다. 그러나 현실적으로 대부분의 오이는 잎에 의해 영상을 통한 인식에 방해를 받는다. 따라서 잎으로 인한 오이인식의 어려움을 해결하고자, 오이의 생육 및 수확량에 영향을 주지 않는 잎의 제거로 오이인식의 효율을 높이고자 한다. 또한 기계화작업을 위하여 관행적인 재배양식을 변형하여 생육과 수확량과의 차이를 관찰하고자 <Fig. 2-1>, <Fig. 2-2>와 같이 원예연구소내 2-PET온실에서 시행하였다. 실험온실은 동서동(東西棟)이며, 실험 대상오이는 은성백다다기종을 재배하여 행하였으며, 75주씩 8줄로 정식하여 그 중 75주를 본 실험의 대상오이로 정하였다. <Fig. 2-3>은 실험온실내의 오이의 재배형식 및 실험대상오이를 나타낸 그림이다. 오이는 1998년 2월 중순에 파종하여, 3월 중순에 정식하였으며, 수확은 4월 하순부터 시작하였다. 여기서 본 실험은 1998년 5월 5일부터 1998년 5월 28일까지 24일간을 실험기간으로 정하였다.



<Fig. 2-1> The fixture of an experimental greenhouse



<Fig. 2-2> Cultivation of cucumber in greenhouse

2. 실험방법

오이수확기 개발을 위하여 보다 정확한 오이의 영상을 얻기 위해서 오이의 불필요한 잎을 제거하였다. 이에 따른 오이의 잎제거가 생육 및 수확의 미치는 영향을 구명하고자 한다. 또한 오이의 정식간격을 24cm, 30cm의 두 가지 변수를 정하여 오이를 재배하였다. 그리고 최근 원예작물의 기계화를 위해서 재배양식과의 연관성 등을 고려하여, 오이의 성장을 3가지 형태의 유인방법에 따라 재배하였다. 여기서 3가지의 재배유인방법으로는 먼저, 통상 일반농가에서 재배하는 방식으로 오이를 생장시키면서 아래로 계속 줄기를 내리는 방법이며, 두 번째의 유인방법으로는 오이의 성장을 일정 높이까지는 수직으로 재배한 후 횡방향으로 오이의 줄기를 유인하는 직립후 횡유인방법을 택하였다. 마지막으로는 오이를 일정높이까지 수직으로 재배한 후 종방향(가로줄)으로 오이의 줄기를 유도하는 재배방법으로 선택하였다. 따라서 <Table 2-1>과 같이 3가지 형태의 잎제거방법, 두 형태의 정식간격, 3가지의 줄기 유인방식으로 실험을 수행하였다. 여기서 관행적인 재배방식의 경우에는 실험하우스 전면에서 볼 때, 수직으로 나란히 두 줄로 오이를 재배하였다, 다른 두가지방법의 줄기 유인방식은 실험하우스의 전면에서 볼 때, <Fig. 2-2>와 같이 V자 형태의 재

배방식을 택하여 기계화작업에 적절하도록 오이의 성장을 유도하였다. 이때 유인막대는 지면의 수평면과 약 78 °를 이루도록 하였다.

<Table 2-1> Experimental design

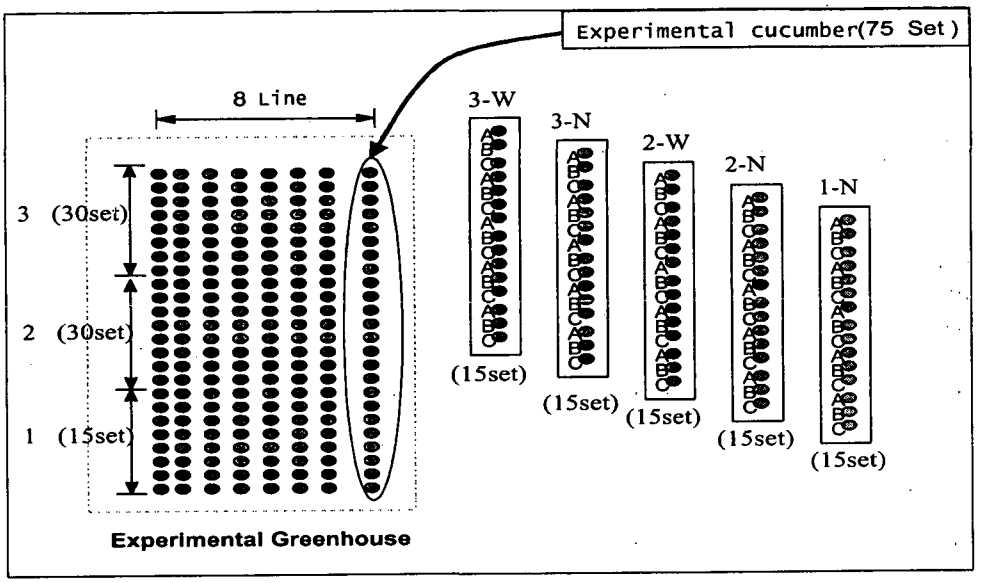
Induction method / Formal space / Method of leaf elimination	Normal farmhouse (1)	Induction cordon (V-cordon, V-training)system			
		Horizontal Induction after erection (2)		Horizontal line Induction after erection (3)	
	24cm (N)	24cm (N)	30cm (W)	24cm (N)	30cm (W)
Normal elimination (A)	1-N-A	2-N-A	2-W-A	3-N-A	3-W-A
One elimination from substructure (B)	1-N-B	2-N-B	2-W-B	3-N-B	3-W-B
Harvesting cucumber substructure overall elimination (C)	1-N-C	2-N-C	2-W-C	3-N-C	3-W-C

잎제거 방식으로는 <Fig. 2-3> <Table 2-1>에서와 같이 3가지의 형태로 제거하였다. 먼저 A는 수확시기가 된 오이만 수확한 후, 고사된 잎만 제거한 경우이며, B는 수확시기가 된 오이를 수확한 후, 줄기의 가장 밑 부분에 달린 잎을 하나씩 제거한 경우이다, 그리고 C는 수확시기가 된 오이를 수확한 후, 수확한 마디 이하의 잎을 모두 제거한 경우를 나타낸 것이다. 단, C의 경우 하부의 오이가 상부의 오이보다 늦은 경우에는 하부오이 마디의 잎은 수확시기까지 늦추어 제거하였다.

3. 분석방법

본 연구에서는 오이수확기를 위한 영상처리시스템의 정확한 오이영상 인식을 위해서 수행하였다. 따라서 부분적인 잎의 제거에 따른 오이의 생육과 수확량에 미치는 영향을 구명하고자 한다. 또한 재배정식간격과 줄기유인형태가

생육에 미치는 영향을 관찰하기 위해서 위의 <Table 2-1>의 실험설계에서 직립후횡유인 방법의 30주를 대상으로 전 실험 기간동안 2일 간격으로 오이의 전장(全長)을 측정하였다. 일반농가의 관행재배방식이나 직립후가로줄 유인방법의 경우에는 줄기유인을 위한 매듭의 문제에 따른 전장측정이 현실적으로 곤란하여 직립후횡유인 방법만을 택하여 전장측정실험을 행하였다.



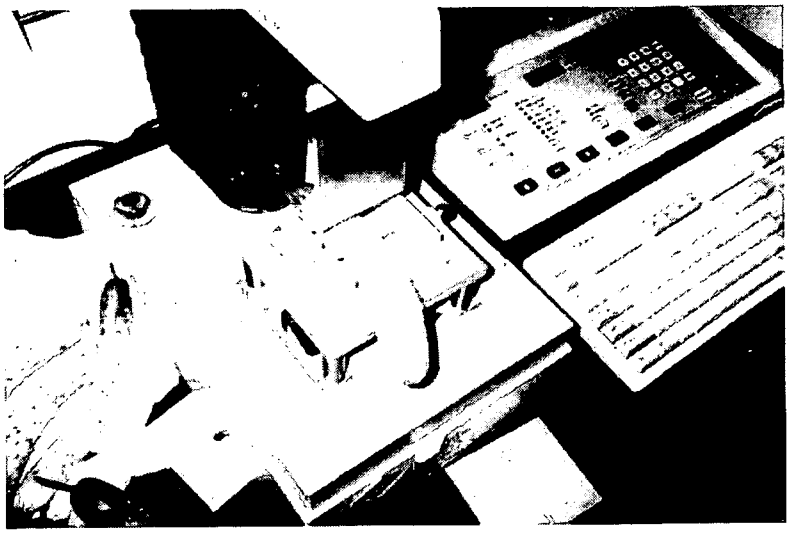
<Fig. 2-3> The Outline of Experimental Greenhouse

또한 잎의 제거가 생육에 미치는 영향을 알아보기 위해서 하나의 판별변수로 수확량과 수확한 오이의 1등급비율을 측정하였다. 이를 위해서 <Table 2-1>의 실험설계에서 각 실험구별로 각각 5주의 실험오이를 정하여 결과(結果)에서 10cm이하의 오이, 10cm에서 20cm이하의 오이, 그리고 20cm이상(수확)오이의 수를 2일 간격으로 측정하였다. 또한 수확된 오이의 등급은 상품가치에 있어서 매우 중요한 변수가 되므로 본 실험에서는 1등급의 비율을 관찰하여 각 실험별 차이를 관찰하였다. 오이의 1등급기준은 일본수출용 오이를 기준으로 길이가 22~25cm, 무게가 110~140이며 굽은 정도가 2cm이내이고 양

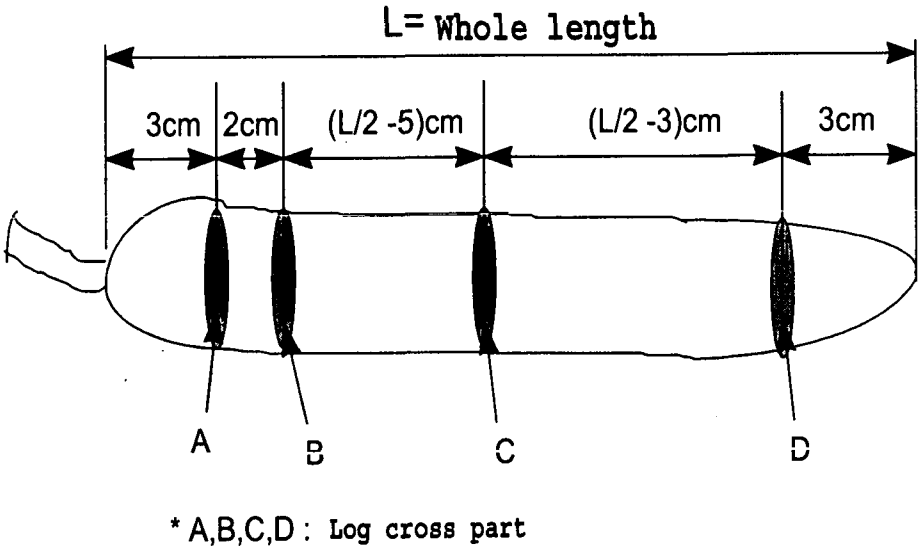
끝 부분이 굵거나 가늘지 않고, 상해가 없어야 하는 근거를 두고 판별하였다.

수확된 오이의 물성측정은 <Fig. 2-4>과 같은 SMS(Stable Micro Systems)사의 Texture analysers-XT.RA을 이용하여 경도(Hardness), 깨짐성(Fracturability), 탄력성(Springness), 점성(Gumminess)을 측정하였다. <Fig. 2-5>은 조직물성 측정기의 작동하는 형태을 나타내고 있다. 위의 물성측정시 오이의 측정 위치는 <Fig. 2-6>와 같이 4단면에서 임의의 5개 오이에 대해서 행하였다. 4부분(A: 과경에서 3cm, B: 과경에서 5cm, C: 가운데, D:꽃피는 지점에서 3cm)을 측정한 이유는 오이의 물성 측정에 있어 정확한 값을 얻기 위해서 하였다. 또한 과경에서 5cm이하의 거칠은 부분은 접촉하면 상품성이 저하되는 부분이다. A 와 B의 경우 그리퍼로 오이를 파지하는 부분이다. 오이의 함수율은 75℃에서 1주일동안 Dry-oven에서 건조한 후 생체무게와 비교하여 측정하였다.

<Fig. 2-4> Experimental equipment for measuring physical property



<Fig. 2-5> A cucumber to be measured its physical property by the experimental equipment



<Fig. 2-6> Four positions for a cucumber to be measured for its physical property

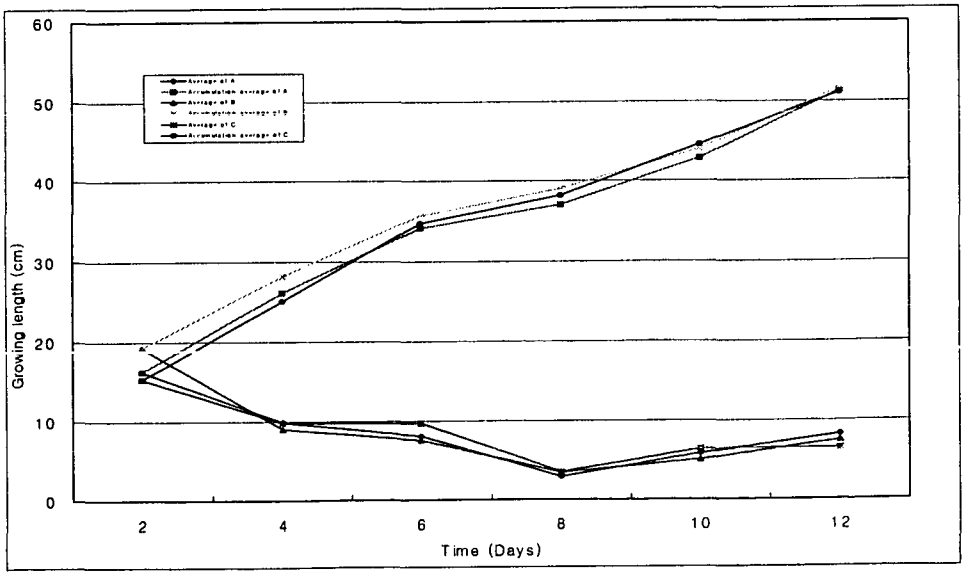
엔드이펙터의 성능을 알아보기 위하여 은성 백다다기오이를 대상으로 실험을 실시하였다.

제 3 절 결과 및 고찰

1. 전장측정

가. 잎제거 형태

본 연구에서는 오이의 영상인식이 가장 큰 장해가 되는 잎을 부분적으로 제거하여도 오이의 생육이나 수확에 영향을 미치지 않는 다면, 오이수확기 개발에 있어서 보다 효율이 높은 수확기 개발이 가능할 것으로 판단된다. 따라서 본 실험에서는 부분적인 오이잎제거가 오이의 생육에 어떠한 영향이 미치는지 알아보기 위하여 행하였다. 또한 오이의 정식간격 형태도 병행하여 오이의 생육에 미치는 영향을 판단하고자 실험하였다. 이를 위하여 앞 절의 실험설계에서 직립후횡유인 방법의 실험오이 30주를 대상으로 실험기간동안 오이의 잎제거형태별 전장측정 결과 <Fig. 2-7>와 같이 나타났다.



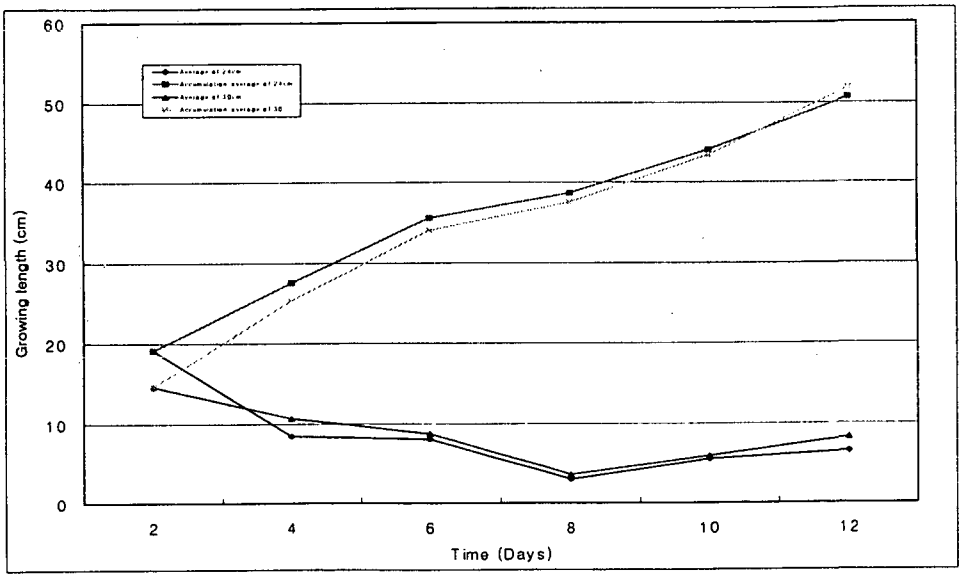
<Fig. 2-7> Overall length measured by two days

위의 결과에서 3가지의 잎제거 형태에 따른 오이생장의 정도를 살펴보면 일반적인 잎제거 형태인 A의 경우에는 12일간 평균 52.2cm성장하였으며, 2일마다 하부의 잎 하나를 제거한 B의 경우에는 평균 51.5cm가 자랐다. 수확한 오이의 하부잎을 모두 제거한 C의 경우에는 51cm 성장하였다. 여기서 3가지의 잎제거 형태에 따른 오이 생장은 모두 3% 이내로 거의 차이가 없는 것을 알 수 있다. 또한 각각의 측정일에서 살펴 보아도 전체적으로 성장의 차이가 3% 이내로 나타났으며, 일별 성장 정도의 차이는 <Fig. 2-7>의 곡선 유형을 살펴볼 때, 비슷한 형태로 나타났다. 여기서 부분적인 일별 성장의 차이는 잎의 제거에 따른 차이가 아닌 외부 기상이나 그 외 또 다른 요인에 의한 차이라 판단된다. 또한 8일째의 오이성장이 다른 날에 비해서 매우 적다는 것을 볼 수 있다. 오이는 빛을 좋아하는 호광성 채소로서 강우로 인해 외기온의 저하와 부족한 광원으로 인해 외부환경요인에 따른 영향으로 판단된다. 따라서 잎제거에 따른 오이의 생육에 미치는 영향을 구명하기 위해서는 보다 정확한 환경계측장비로 외부환경변수의 계측도 매우 중요할 것으로 판단된다.

나. 정식간격

일반적으로 식물의 정식간격은 수확량과 성장을 고려하여 적절하게 유지하여야 한다. 본 실험에서는 일반적으로 오이의 재배간격에 가장 많이 이용되고 있는 24cm, 30cm의 두 가지 형태의 정식간격을 기준으로 오이의 성장을 관찰하였다. 이러한 정식간격은 전체 수확량에 차이가 없다면, 기계화에 적당한 간격을 선정하여 본 연구의 궁극적인 목적인 오이수확기 개발에 있어서 보다 정확한 영상정보를 획득하여 수확기의 효율을 향상시키고자 하는데 이용하고자 한다. 따라서 본 실험에서는 두 가지형식의 재배정식 간격이 오이의 생육에 어떠한 영향을 미치는지 알아보기 위하여 행하였다. 이를 위하여 앞 절의 실험설계에서 직립후횡유인 방법의 실험오이 30주를 대상으로 실험기간동안 오이의 전장을 측정한 결과 <Fig. 2-8>와 같이 나타났다.

여기서 두가지의 재배형태에 따른 전장의 길이가 각각, 50.8cm, 51.8cm로 약 2%의 차이를 보이고 있다. 이러한 결과에서 정식간격은 전체 오이 성장에는 큰 영향을 주지 않은 것을 알 수 있다. 또한 정식간격과는 무관하게 이틀에 약 3~10cm씩 자란 것으로 나타났으며, 이러한 일별 성장은 두가지의 정식 간격에서 비슷한 형태를 보이고 있다. 이는 위에서도 언급하였듯이 외부의 환경요소에 따른 영향으로 판단되며, 정식 간격에 의한 결과는 아니라고 판단된다. 따라서 전장측정 결과 오이의 잎제거 형식이나 정식간격에 따른 전체 성장에는 영향을 주지 않는 것으로 판단되며, 수확량을 비롯한 다른 변수와 연관하여 오이의 수확에 용의한 기계화작업에 적절한 재배형식을 결정하는 것이 중요할 것으로 판단된다.



<Fig. 2-8> Overall length measured of formal space between 24 cm and 30cm by two days

2. 수확량과 1등급비율

가. 잎제거 형태

본 연구에서는 앞에서 언급하였듯이 궁극적으로는 오이의 보다 정확한 영상 정보를 얻고자 하는데 그 목적을 두고 있다. 이를 위해서 오이의 영상처리에 있어서 가장 크게 장해가 되는 것 중에 하나가 잎이다. 따라서 오이의 영상정보획득의 효율을 높이고자 먼저 오이의 부분적인 잎제거가 오이의 생육 및 수확량에 미치는 영향을 구명하고자 한다. 여기서 잎제거에 따른 영향을 알아보기 위한 판별변수의 하나로 수확량과 수확한 오이의 1등급 비율로 정하였다. 따라서 본 실험에서는 부분적인 오이 잎제거가 오이의 수확에 미치는 영향을 알아보기 위해서 3가지의 실험변수를 정하였다. 먼저 일반농가에서 관행적으로 고사된 잎만 제거하는 방법, 두 번째는 실험 기간동안 2일 간격으로 하부의 잎을 하나씩 제거하는 방법, 그리고 마지막으로 수확한 오이가 있으면 그 이하의 잎은 모두 제거하는 방식 등의 3가지 형태의 잎제거 방식으로 수확량과 1등급의 비율을 비교 관찰하였다. 수확량과 1등급의 비율의 측정은 앞절의 실험설계에서 전체 실험오이 75주를 대상으로 측정하였으며, 그 결과 <Table 2-2> 와 같이 나타났다.

<Table 2-2> Yield and the first grade ratio by method of leaf elimination

Method of leaf elimination	Actual test	Yield(number)	First grade ratio(%)
Normal elimination	1-N-A	23(5.8)	47.8
	2-N-A	33(4.7)	69.7
	2-W-A	44(6.3)	54.5
	3-N-A	29(4.1)	44.8
	3-W-A	36(5.1)	66.7
	Average	33.00	56.70
One elimination from substructure	1-N-B	23(5.8)	34.8
	2-N-B	25(3.6)	60.0
	2-W-B	38(5.4)	57.9
	3-N-B	35(5.0)	65.7
	3-W-B	34(4.9)	47.1
	Average	31.00	53.10
Harvesting cucumber substructure overall elimination	1-N-C	24(6.0)	45.8
	2-N-C	35(5.0)	54.3
	2-W-C	39(5.6)	64.1
	3-N-C	29(4.1)	51.7
	3-W-C	29(4.1)	65.5
	Average	31.20	56.28

( ) ; Average yield for 2 days of cucumber with 5 weeks

농업 생산물이 다 그러하듯이 오이도 토지의 생산성 측면에서 수확량을 증가시켜야 할 뿐 만 아니라, 수확된 농산물의 품질도 양호해야 한다. 따라서 오이의 수확량과 1등급 비율은 경제적인 측면에서도 중요한 판별변수 중에 하나이다. 오이는 2일 간격으로 약 20cm이상인 것을 수확하여 수량을 측정하였으며, 수확된 오이의 1등급기준은 일본수출용 오이를 기준으로 길이가 22~25cm, 무게가 110~140이며 굽은 정도가 2cm이내이고 양끝 부분이 굵거나 가늘지 않고, 상해가 없어야 하는 구거를 두고 판별하였다.

위의 결과에서 3가지의 잎제거 형태에 따른 수확량과 1등급의 비율을 살펴보면 일반농가에서 관행적으로 행하는 고사잎만 제거한 방법은 실험 기간동안 실험구에서 평균 33개의 오이를 수확하였으며, 평균 한주당 2.6개/일을 수확할 수 있었다. 그리고 수확된 오이 가운데 1등급은 56.7%를 점하고 있다. 두 번째로 2일 간격으로 하부잎을 한 개씩 제거한 경우의 수확량은 각 실험구에서 평

균 31개로 한주당 2.47개/일의 수확이 가능하였으며, 1등급비율은 53.1%로 나타났다. 마지막으로 3번째 잎제거 방식인 수확한 오이의 하부잎은 전부 제거한 경우에는 수확량이 각 실험구에서 평균 31.2개 였으며, 이는 한주당 2.48개/일이 수확 된 것이며, 1등급비율은 56.28%로 나타났다. 이것을 전체 수확량으로 환산하면 첫 번째의 경우에는 165개, 두 번째, 세 번째는 각각 155개, 156개가 수확된 것이다. 여기서 3가지의 잎제거 형태에 따른 수확량의 차이는 약 6%정도로 앞 절의 전장 측정에 비해서는 다소 높은 결과인 것을 알 수 있다. 상품가치로서의 수확된 오이의 1등급의 비율은 잎제거 형태에 따라 각각 56.7%, 53.1%, 56.28%를 보였다. 이는 수확량과 비슷한 경향을 보이고 있으며, 잎제거 형태에 따른 약간의 차이는 보이고 있지만 10% 이내로 큰 영향을 주지는 않을 것으로 판단된다. 또한 수확한 오이의 하부잎은 모두 제거하는 세 번째 잎제거 방식이 2일마다 1개씩 제거한 두 번째 방식보다 제거한 잎의 수는 많았음에도 수확량과 1등급비율은 높은 것으로 나타난 것을 보면, 잎제거가 수확량과 1등급비율에 크게 영향을 미치지 않는 것을 알 수 있다.

나. 정식 간격

본 실험에서는 일반적으로 오이의 재배간격에 널리 이용되고 있는 24cm; 30cm의 두 가지 형태의 정식간격을 기준으로 실험기간동안 오이의 수확량과 1등급비율을 관찰하였다. 두 가지형식의 재배정식 간격이 오이의 생육에 어떠한 영향이 미치는지 알아보기 위하여 수확량과 1등급의 비율의 측정은 앞장의 실험설계에서 전체 실험오이 75주를 대상으로 측정하였다. 그 결과 <Table 2-3>과 같이 나타났으며, 수확량과 1등급비율은 앞에서와 같이 2일 간격으로 약 20cm이상인 것을 수확하여 수량을 측정하였다. 수확된 오이의 1등급기준은 일본수출용 오이를 기준으로 길이가 22~25cm, 무게가 110~140이며 굽은 정도가 2cm이내이고 양끝 부분이 굵거나 가늘지 않고, 상해가 없어야 하는 근거를 두고 판별하였다.

위의 결과에서 2가지 형태의 정식간격에 따른 수확량과 1등급의 비율을 살펴보면 24cm의 경우에는 각 실험구에서 평균 32.66개의 수확량을 보였다. 이는 2.3개/일의 평균 수확을 의미하며, 그 가운데 1등급의 비율은 59.76%로 나타났다. 정식간격이 30cm의 경우에는 평균 각 실험구별 35개의 수확량을 보였으며, 이는 한주당 2.5개/일의 수확량을 의미하며, 여기서 1등급비율이 57.23%로 나타났다. 여기서 정식간격이 30cm의 경우가 수확량이 약 6%이상 높았다. 그러나 1등급비율을 기준으로 살펴보면 오히려 24cm의 정식간격에서 더 높은 것을 알 수 있다. 또한 이것을 1등급의 수확량으로 살펴보면 각각 19.5개, 20.0개로 거의 비슷한 결과라는 것을 알 수 있다. 따라서 정식간격에 따른 수확량은 24cm 경우가 약간 적었지만, 1등급의 개수를 비교하면 거의 차이가 없으므로 농가 전체를 기준으로 수확량을 환산한다면 오히려 정식간격이 좁은 것이 유리할 수도 있다. 그러나 여기서 기계화에 용의한 정식간격을 생각한다면 농가의 상황을 적절히 고려하여 결정하는 것이 좋을 것으로 판단된다.

<Table 2-3> Yield and the first grade ratio by formal space

Formal space	Actual test	Yield(number)	First grade ratio(%)
24cm	2-N-A	33 (4.7)	69.7
	2-N-B	25 (3.6)	60.0
	2-N-C	35 (5.0)	54.3
	3-N-A	29 (4.1)	44.8
	3-N-B	35 (5.0)	65.7
	3-N-C	39 (5.6)	64.1
	Average	32.67	59.77
30cm	2-W-A	44 (6.3)	54.5
	2-W-B	38 (5.4)	57.9
	2-W-C	29 (4.1)	51.7
	3-W-A	36 (5.1)	66.7
	3-W-B	34 (4.9)	47.1
	3-W-C	29 (4.1)	65.5
	Average	35.00	57.23

( ) ; Average yield for 2 days of cucumber with 5 weeks

다. 줄기유인 형식

앞에서도 언급하였듯이 오이의 영상처리에 있어서 가장 크게 장해가 되는 것 중에 하나가 잎이다. 또한 수확할 오이의 위치는 오이수확기 개발의 효율을 제고시킬 수 있는 방법중에 하나 일 것으로 판단하여, 3가지 형태의 줄기유인방법을 정하여 수확량과 1등급의 비율을 관찰하였다. 이러한 유인방법에 따른 수확량과 1등급의 비율의 측정은 앞장의 실험설계에서 전체 실험오이 75주를 대상으로 앞 절의 기준과 같이 측정하였으며, 그 결과 <Table 2-4>과 같이 나타났다.

여기서 3가지의 유인형태에 따른 수확량과 1등급의 비율을 살펴보면 일반 농가에서 관행적으로 행하는 유인방법의 경우 실험 기간동안 각 실험구에서 평균 23.33개의 오이를 수확하였다. 이것은 한주당 평균 2.9개/일을 수확하였다는 것을 의미한다. 그리고 수확된 오이 가운데 1등급은 42.8%를 점하고 있다. 두 번째로 직립후 횡유인의 경우 각 실험구에서 평균 31개로 한주당 2.2개/일의 수확이 가능하였으며, 1등급비율은 61.33%로 나타났다. 마지막으로 3번째 유인방법인 직립후 가로줄 유인의 경우 수확량이 각 실험구에서 평균 31개 였으며, 이는 한주당 2.2개/일이 수확 된 것이며, 1등급비율은 54.06%로 나타났다. 여기서, 첫 번째 관행적이 방법의 경우에 전체 수확량이 적은 이유는 실험시설내의 재배위치에 따른 입구와 내부의 기온차이 때문에 수확시기가 1주일정도 늦었다. 또한 전체 재배포기가 다른 방식에 비해서 1/2인 것을 고려하면, 한주당 수확량은 오히려 다른 유인방법에 비해서 높은 것을 알 수 있다. 그러나 1등급의 비율을 살펴 보면 관행적이 방법에 비해서 두 번째, 세 번째가 훨 씬 양호한 결과를 보이고 있다. 따라서 관행적인 방법에 비해서 유인방법을 달리한 두 가지의 형태가 수확량과 1등급 비율에서 좋은 결과를 보인 것으로 비추어 볼 때, 적절한 줄기의 유인은 오이의 생육과 수확에 유리할 것으로 판단된다. 이는 또한 직립후 횡유인과 직립후 가로줄유인의 경우 수확할 오이의 위치가 상부쪽에 위치하므로 오이의 영상정보 획득에 유리할 뿐 만 아니라, 기

계화작업도 여러 가지 측면에서 유리할 것으로 판단된다.

<Table 2-4> Yield and the first grade ratio by induction method

Induction method	Actual test	Yield(number)	First grade ratio(%)
Normal farmhouse	1-N-A	23(5.8)	47.8
	1-N-B	23(5.8)	34.8
	1-N-C	24(6.0)	45.8
	Average	23.33	42.80
Horizontal induction after erection (V-cordon)	2-N-A	33(4.7)	69.7
	2-N-B	25(3.6)	60.0
	2-N-C	35(5.0)	54.3
	Average	31.00	61.33
Horizontal line induction after erection (V-cordon)	3-N-A	29(4.1)	44.8
	3-N-B	35(5.0)	65.7
	3-N-C	29(4.1)	51.7
	Average	31.00	54.07

( ) ; Average yield for 2 days of cucumber with 5 weeks

3. 낙과수

가. 잎제거 형태

오이의 부분적인 잎제거가 오이의 생육 및 수확량에 미치는 영향을 구명하고자 판별변수의 하나로 낙과수를 측정하였다. 낙과는 오이가 개화하여 열매로 성장하면서 수확 할 수 있는 상태로 자라지 못한 것을 의미하며, 다른 판별변수와 마찬가지로 2일간격으로 측정하였다. 따라서 본 실험에서는 부분적인 오이 잎제거가 오이의 낙과에 미치는 영향을 알아보기 위해서 앞 절에서

언급한 3가지 형태의 잎제거 방식에 따른 낙과수를 측정하였다. 낙과수의 경우도 다른 판별변수와 마찬가지로 전체 실험오이 75주를 대상으로 측정하였으며, 그 결과 <Table 2-5>와 같이 나타났다.

위의 결과에서 3가지의 잎제거 형태에 따른 낙과수를 살펴보면 일반농가에서 관행적으로 행하는 고사잎만 제거한 방법은 실험기간동안 각 실험구에서 평균 11.4개의 낙과가 있었으며, 두 번째로 2일 간격으로 하부잎을 한 개씩 제거한 경우의 각 실험구에서는 평균 8.8개로 낙과가 있었으며, 마지막으로 3번째 잎제거 방식인 수확한 오이의 하부잎은 전부 제거한 경우에는 각 실험구에서 평균 8.8개의 낙과가 있었다. 이것을 전 실험기간 동안의 낙과수로 환산하면 각각 57개, 44개, 44개로 나타났다. 따라서 어떤 형태로든 부분적인 입제거가 오히려 낙과를 줄이는데 도움을 줄 수 있다는 것을 알 수 있다. 따라서 기계화작업 및 오이 영상정보획득 등의 측면에서 뿐 만 아니라, 생육의 측면에서도 적절한 잎의 제거는 통풍 및 양분의 고른 공급을 위해서 필요할 것으로 판단된다.

<Table 2-5> Number of dropped cucumber fruit by method of leaf elimination

Method of leaf elimination	Inducing method	Formal space	Acual test	Number of dropped cucumber fruit
Normal elimination	Normal farmhouse	24cm	1-1-A	7
	Horizontal inducing after erection	24cm	2-1-A	17
		30cm	2-2-A	9
	Horizontal line inducing after erection	24cm	3-1-A	18
		30cm	3-2-A	6
	Average			11.40
One elimination from substructure	Normal farmhouse	24cm	1-1-B	4
	Horizontal inducing after erection	24cm	2-1-B	14
		30cm	2-2-B	10
	Horizontal line inducing after erection	24cm	3-1-B	9
		30cm	3-2-B	7
	Average			8.80
Harvesting cucumber substructure overall elimination	Normal farmhouse	24cm	1-1-C	1
	Horizontal inducing after erection	24cm	2-1-C	12
		30cm	2-2-C	6
	Horizontal line inducing after erection	24cm	3-1-C	14
		30cm	3-2-C	11
	Average			8.80

나. 정식 간격

본 실험에서는 일반적으로 오이의 재배간격에 널리 이용되고 있는 24cm, 30cm의 두 가지 형태의 정식간격을 기준으로 실험기간동안 오이의 낙과수를 관찰하였다. 두 가지형식의 재배정식 간격이 오이의 생육에 어떠한 영향이 미치는지 알아보기 위하여 낙과수의 측정은 전체 실험오이 75주를 대상으로 측정하였으며, 그 결과 <Table 2-6>과 같이 나타났다.

위의 결과에서 2가지 형태의 정식간격에 따른 낙과수를 살펴보면, 24cm의 경우에는 각 실험구에서 평균 14개가 수확되지 못하고 낙과하였으며, 정식간격

이 30cm의 경우에는 평균 각 실험구별 8.16개의 오이열매가 낙과한 것을 알 수 있다. 여기서 정식간격이 24cm의 경우에 낙과수가 약 40%이상 많았으며, 이는 앞 절의 수확량과 비교해 볼 때, 같은 결과를 나타내고 있다. 따라서 전체 수확량에 차이를 주지 않는 다면 적절한 정식간격의 유지는 기계화 뿐 만 아니라 오이의 생육에도 더 효과적 일 것으로 판단된다.

<Table 2-6> Number of dropped cucumber fruit by formal space

Formal space	Inducing method	Actual test	Number of dropped cucumber fruit
24cm	Horizontal inducing after erection	2-1-A	17
		2-1-B	14
		2-1-C	12
	Horizontal line inducing after erection	3-1-A	18
		3-1-B	9
		3-1-C	14
	Average		14.00
30cm	Horizontal inducing after erection	2-2-A	9
		2-2-B	10
		2-2-C	6
	Horizontal line inducing after erection	3-2-A	6
		3-2-B	7
		3-2-C	11
	Average		8.17

다. 줄기유인 형식

본 실험에서는 먼저 일반농가에서 관행적으로 오이의 줄기가 성장하면 윗부분에 매듭을 하여 아래로 줄기를 내리는 방식, 두 번째는 일정높이까지는 위로 줄기를 유인한 다음, 매듭을 이용하여 기존방향에 90도 방향으로 횡유인 하였으며(직립후 횡유인방식), 마지막방법은 두 번째 방법과 마찬가지로 일정 높이까기 위로 유인한 다음, 다른 인접한 오이가 있는 방향과 나란히 가로줄로 유인하는 방법(직립후 가로줄유인) 등의 3가지 형태의 유인 방법으로 낙과수를 비교 관찰하였다. 이러한 유인방법에 따른 낙과수 측정은 앞장의 실험설계에서

처럼 전체 실험오이 75주를 대상으로 앞 절의 기준과 같이 측정하였으며, 그 결과 <Table 2-7>와 같이 나타났다.

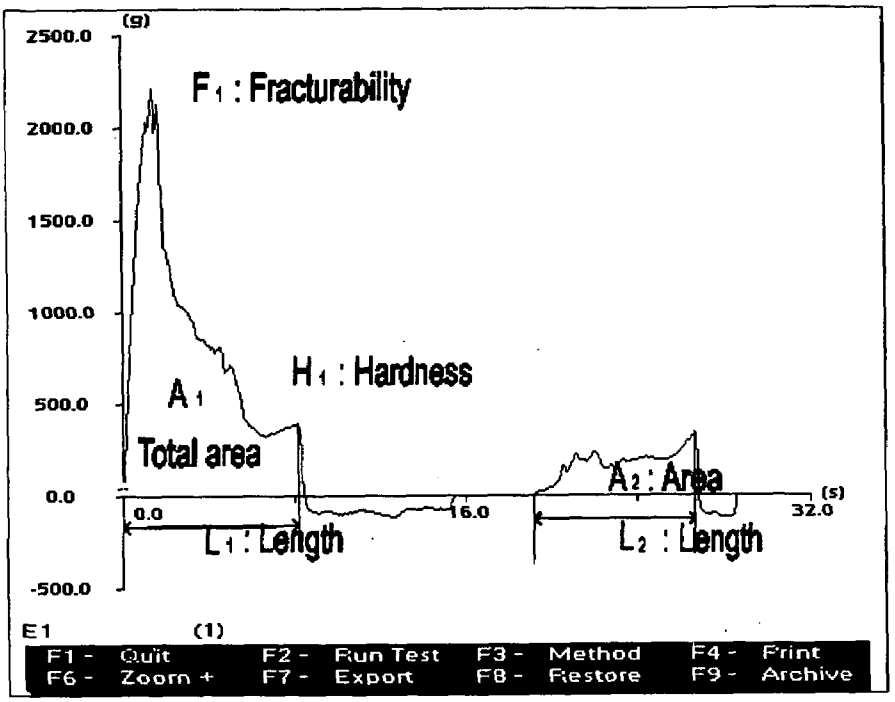
위의 결과에서 3가지의 유인형태에 따른 낙과수를 살펴보면 일반농가에서 관행적으로 행하는 방식의 경우에는 각 실험구에서 평균 4개의 낙과가 있었으며, 두 번째로 직립후 횡유인의 경우에는 각 실험구에서는 평균 14.33개의 낙과가 있었으며, 마지막으로 3번째 직립후 가로줄 유인의 경우에는 각 실험구에서 평균 13.66개의 낙과가 있었다. 여기서 두 번째, 세 번째에 비해서 관행적인 방법인 첫 번째 방식의 경우에 매우 적게 나타난 것을 알 수 있다. 앞 절에서 수확시기가 늦은 점과 포기수가 다른 유인방법에 비해서 1/2인 것을 고려한다면 전체적으로 영향을 줄만큼의 차이는 아니라고 판단된다. 이를 수확량과 같이 생각하면 큰 차이가 없다는 것을 알 수 있다.

<Table 2-7> Number of dropped cucumber fruit by inducing method

Inducing method	Formal space	Method of leaf elimination	Actual test	Number of dropped cucumber fruit
Normal farmhouse	24cm	Normal elimination	1-1-A	7
		One elimination from substructure	1-1-B	4
		Harvesting cucumber substructure overall elimination	1-1-C	1
	Average			4.00
Horizontal inducing after erection (V-cordon)	24cm	Normal elimination	2-1-A	17
		One elimination from substructure	2-1-B	14
		Harvesting cucumber substructure overall elimination	2-1-C	12
	Average			14.33
Horizontal line inducing after erection (V-cordon)	24cm	Normal elimination	3-1-A	18
		One elimination from substructure	3-1-B	9
		Harvesting cucumber substructure overall elimination	3-1-C	14
	Average			13.67

4. 물성측정

오이수확기 개발을 위한 가장 중요한 기술중에 하나는 그리퍼를 포함한 엔드이펙터부분의 설계 및 제작이라 할 수 있다. 보다 효율적인 수확을 위해서 정확한 엔드이펙터의 설계를 위해 본 실험에서는 오이의 무게, 길이, 단면적을 측정하였다. 조직물성측정기 TPA(Texture Profile Analysis)을 이용하여 깨짐성, 경도, 탄력성, 점성을 측정한 결과 <Table 2-8>과 같이 나타났다.



<Fig. 2-9> Graph of texture profile analysis

<Fig. 2-9>은 조직물성측정기 TPA(Texture Profile Analysis)을 이용하여 깨짐성(Fracturability), 경도(Hardness), 탄력성(Springness), 점성(Gumminess)을 측정한 것이다. 여기서, F1은 깨짐성은 힘을 가했을때 처음으로 하강하는 곡선을 나타낸다. 경도는 압축중 0점으로 떨어지는 최고점으로 나타나고, 탄력

성은 L2/L1을 이용하여 구할 수 있다. 점성은 경도 × 응집성(Cohesiveness)으로 나타난다. 여기서, 응집성은 A2/A1으로 구한다.

<Table 2-8> The physical properties of cucumber

Physical Property	Weight(g)	Length(cm)	Cross section(mm^2)	Cucumber moisture content (%,w.b)
Value	170.64	21.9	812.31	96.76
Calyx moisture content (%,w.b)	Fracturability(g)	Hardness(g)	Springness	Gumminess(g)
93.2	2302.36	549.31	0.98	190.89

앞장에서 언급하였듯이 물성측정에 사용된 5개의 오이의 무게의 평균은 170.64 g이며, 길이는 21.9 cm, 단면적은 812.31 mm^2으로 나타났다. 측정위치는 상부·하부에서 각각 3 cm 부근과 상부에서 5 cm, 중간부위 등 모두 4부분에서 깨짐성, 경도, 탄력성, 점성을 측정하였으며, 그 결과 경도의 경우 파지부 설계의 기초자료로서 의미가 있을 것으로 판단된다. 또한 함수율은 75℃에서 1주일 동안 건조 오븐(Dry-oven)에서 건조하여 건조전의 무게와 비교하여 측정한 결과 일반적인 과채류의 함수율과 마찬가지로 90%이상으로 나타난 것을 알 수 있다. 또한 엔드이펙터와 직접관련이 있는 과경의 함수율은 오이 자체의 함수율보다는 낮지만 비교적 높은 함수율을 나타내고 있다. 따라서 위의 물성값을 기초로 수확기 개발을 위한 기초연구자료로 이용 가능할 것으로 판단된다.

제 4 절 요약 및 결론

본 연구는 오이수확기 개발을 위하여 오이의 인식에 장해가 되는 잎의 부분적인 제거 및 기계화를 위한 재배양식이 오이의 생육과 수확량에 미치는 영향을 구명하고자 하였다. 세가지 형태의 잎제거 방식과 두가지 형태의 정식간격, 그리고 3가지 형태의 줄기 유인방식을 정하여 오이의 전장, 수확량과 1등급비율 및 낙과수를 측정한 결과 다음과 같은 결론을 얻었다.

1. 잎제거 형태에 따른 오이의 생육을 관찰하기 위해서 전장을 측정한 결과, 관행적인 제거방식의 경우 일정 실험기간동안 평균 52.2cm 성장하였다. 또한, 2일마다 하부잎 하나를 제거한 경우에는 평균 51.5cm 성장하였으며, 수확한 오이가 있는 경우 하부의 잎을 모두 제거한 경우에는 평균 51cm가 성장한 것으로 보아 평균 3% 이내의 차이를 보였다. 또한 24cm, 30cm의 정식간격에 따라서는 각각 평균 50.8cm, 51.8cm의 약 2%정도 성장에 차이를 보였다. 따라서 부분적인 잎제거는 오이의 성장에 큰 영향을 미치지 않는 것으로 판단된다.

2. 판별변수의 하나인 수확량과 1등급비율을 잎제거 형태에 따라 비교해 보면, 관행적인 제거방식의 경우, 평균 한주당 2.6개/일 를 수확하였으며, 1등급 비율은 56.7%로 나타났다. 또한 2일마다 하부잎 하나를 제거한 경우에는 평균 한주당 2.47개/일의 수확이 있었으며, 1등급 비율은 53.1%로 나타났다. 마지막으로 수확한 오이가 있는 경우 하부의 잎을 모두 제거한 경우에는 한주당 평균 2.48개/일를 수확하였으며, 1등급 비율이 56.28%로 나타났다. 따라서 수확량의 차이는 약 6%정도이고 1등급 비율도 10% 이내로 큰 영향은 없는 것으로 판단된다.

3. 오이의 정식간격에 따른 수확량과 1등급비율을 측정한 결과, 24cm의 경

우 한주당 2.3개/일의 평균수확이 되었으며, 1등급비율은 59.76%, 30cm의 경우 한주당 평균 2.5개/일을 수확하였으며, 1등급은 57.23%로 나타났다. 따라서 수확량은 24cm의 경우가 약간 적었지만 1등급의 비율로 환산하면 크게 차이가 없는 것을 알 수 있다. 따라서 농가전체 수확량을 기준으로 한다면 24cm가 유리하다고 판단된다.

4. 줄기유인 형식에 따라 수확량과 1등급비율을 살펴보면, 관행적인 방법은 평균 한주당 2.9개/일, 42.8%, 직립후횡유인은 평균 한주당 2.2개/일, 61.33%, 직립후 가로줄유인은 평균 한주당 2.2개/일, 54.06%로 나타났다. 따라서 수확량과 1등급을 동시에 고려하면, 큰 차이는 없지만, 직립후 횡유인방법과 직립후 가로줄유인의 경우는 오이가 지면에서부터 상대적으로 상부에 위치하므로 영상인식과 기계화 측면에서 유리할 것으로 생각된다.

5. 낙과수를 판별변수로 하여 잎제거 형태 비교해 보면, 관행적인 제거방식의 경우, 평균 11.4개, 2일마다 하부잎 하나를 제거한 경우와 수확한 오이가 있는 경우 하부의 잎을 모두 제거한 경우에는 평균 8.8개가 낙과하였다. 정식간격으로 살펴보면, 24cm의 경우 평균 14개, 30cm의 경우는 평균 8.16개가 낙과하였다. 또한 줄기 유인형태에 따라 낙과수를 비교하면, 관행적인 방법은 평균 4개, 직립후 횡유인은 14.33개, 가로줄유인은 13.66개로 나타났다. 여기서 관행적인 방법은 전체 수확기간의 적었기 때문이며, 이것을 고려하면 3가지 형태의 유인방법이 큰 자이가 없는 것으로 판단된다. 따라서 인위적인 잎제거는 낙과를 줄일 뿐 만 아니라 기계화측면과 생육에도 도움이 될 것으로 판단된다.

제 3 장 구동시스템 개발

제 1 절 서 론

시설재배농업의 작물재배 과정에서 자동화는 필수적이라 할 수 있으며, 작물 사이의 고랑들을 따라서 원하는 위치로 이동할 수 있는 주행 시스템의 개발은 요구되어지고 있는 기술이라 할 수 있다. 작업기의 자동주행이 이루어진다면 노동력의 감소 뿐 아니라 고온다습의 온실 내에서의 작업과정을 작업자가 외부에서 제어해 줄 수 있으며, 밀폐된 온실에 약제를 살포하는 등의 질식의 위험성이 있는 작업을 사람이 직접하지 않아도 될 것이다.

지금까지 온실내 작업기의 주행 시스템은 작업차에 직접 부착이 되는 방식이었다. 따라서 시설재배농업이 전과정 자동화 및 식물공장의 형태로 발전되어 나아갈 때 모든 작업기에 이러한 시스템을 부착한다면, 작업기의 제작과정이나 비용면에서 적지 않은 부담이 되리라 판단된다.

제 2 절 구동시스템의 연구동향

주행장치는 궤도형과 유인형 및 대차 시스템으로 나눌 수 있다. 시설재배의 경우 궤도형 및 유인형을 많이 사용하고 있다. 궤도형의 경우는 궤도를 바닥에 설치하는 방법이며, 유인형의 경우는 온실 상부에 유인 장치를 하는 것이다. 궤도형은 토양 슬립이 적어 방향 전환이 안정적인 궤도형 주행장치를 많이 연구하고 있으나, 초기 투자비가 많이 들고 작업자의 작업 공간 축소에 따른 안전사고의 위험이 따르고 있다. 유인형의 경우는 온실 상부에 레일 및 연마봉을 설치하고, 주행장치가 상부에 고정되어 주행되는 것이다. 이 경우 작업공간의 확보는 가능하지만, 초기 투자비가 많이 들고 기존의 농가에서 설치하는데 있

어서 어려움이 많다. 또한 채광이 좋지 않아 작물의 생육환경에 불리한 단점이 있다. 대차 시스템은 초기 투자비가 궤도형, 유인형에 비해 적게들고, 작업 형태에 따라 적용범위가 크지만 소프트웨어적으로 제어가 어려운 단점을 가지고 있다.

현재 농업분야에서 이용되고 있는 차량의 주행장치로는 휠형과 궤도형이 이용되고 있다. 특히 연약지에서 많이 운용되고 있는 농용 휠형차량은 높은 접지압 때문에 토양의 다짐에 의해 작물의 생산량을 감소시키는 것으로 보고되고 있고(Aura,1993), 수분을 전달하고 유지하는데 필요한 안정된 토양구조에 손상을 준다는 보고도 있다(Ayers, 1987).

Muro(1989)는 연약지에서 주행하는 불도저의 견인성능을 예측하기 위해 궤도의 토양반력은 접지면에 작용하는 접지압의 적분값과 같고 이것은 궤도장역의 수직문력과 같다고 가정하여 반복 계산에 의해 접지압력을 구하였다.

Culshaw(1988)는 농업용 트랙터에 사용된 공기타이어와 전통적인 강철 궤도의 상대적인 장점을 연구하였고, 각각의 장점을 혼합한 고무궤도의 가능성에 관해 조사하였다.

Wong(1984, 1986, 1988, 1989)은 연성 궤도형차량의 견인성능을 예측하는데 있어, 차량무게, 궤도장력, 궤도 폭,전륜 수, 전륜 직경 등 차량의 주요 설계변수뿐만아니라 토양의 압력-침하와 전단특성 및 반복하중과 반복전단 특성 등을 포함하는 궤도-토양간의 상호작용을 분석하여 궤도아래의 압력분포를 예측해서 연성 궤도형차량의 성능을 예측할 수 있는 수학적 모델을 개발하여 타당성을 증명하였다.

Dwyer (1993)등은 고무궤도형 차량의 견인성능을 예측하기 위해, 궤도를 완전 강체와 완전 연성인 두 가지 극단적인 경우로 궤도세를 가정한 후 각각의 수학적 모델로부터 견인력을 구하여 실험치와 비교 검토하였다.

Okello(1994)는 궤도를 두가지의 극단적인 경우로 가정한 Dwyer 등의 결과를 보완하여 실제적인 연성을 갖는 고무궤도를 대상으로 궤도장력을 고려한 수학적 모델을 개발하였다. 고무궤도-토양의 상호작용 형태로서 전륜 사이의

유연한 궤도부분들의 형태를 유한요소의 절점 좌표들에 의해 결정하고, 각각의 점에서의 토양변형을 모델에 포함시켜 모델의 의해 예측된 값과 측정된 값을 비교하여 고무궤도형 차량의 견인성능 예측 모델의 타당성을 입증하였다.

육묘 로봇은 Kutz(1987)등이 비닐하우스와 같은 원예시설 내에서 생산되고, 모의 육묘관리작업은 거의 인력에 의존하고 있고, 농업 종사자의 감소와 고령화에 대처하기 위하여 개발되었다. 시설 내에 궤도를 설치하고 제어는 마이크로 프로세서가 내장된 프로그래머블 컨트롤러의 제어프로그램에 따라 순차적으로 작동되도록 하였다.

약제살포 로봇은 작업자가 농약에 노출되는 문제를 해결하기 위해 개발되었다. 구동부는 이랑 사이를 주행하기 위해 호스를 감는 장치와 노즐이 탑재되어 있어 양쪽편에 있는 작물에 약재를 살포하도록 되어있고, 4륜 각각에 한 방향 클러치가 장착되어 있고 배터리로 구동된다. 클러치는 전진시에는 전륜, 후퇴시에는 후륜구동 방향으로 부착되어 있어 전후진 모두 전륜구동을 하게 하였다.

시비 로봇은 폭이 좁이 좁은 고무 차륜을 사용하였으며 전륜을 90 °틀어서 횡방향으로도 이동할 수 있는 4륜 구동차로 되어 있다. 제어부는 8비트 단일칩 마이크로 컴퓨터 3개를 이용하였다.

제초 로봇은 일본 홋카이도 농업시험장에서 작물과 잡초를 식별하여, 잡초만을 제거하는 연구를 하였다. 토양과 식물체의 분광반사특성을 이용하여 자율주행을 하면서 잡초를 제거하도록 하였다. 검출된 잡초는 유압 구동형 제초기구를 이용하여 칼쿠리에 의한 기계적인 제초방법을 적용하였다.

잔디 깍는 로봇은 예취한 곳의 경계를 인식하면서 주행하도록 되어 있다. 4륜을 가지고 있으며 전륜의 바퀴가 후륜의 바퀴보다 크기가 크다. 전륜은 직진성만을 가지고 있으며, 전방에 흑백 카메라를 설치하여 영상을 이치화 한 후 후륜의 바퀴를 이용하여 방향전환이 가능하도록 하였다.

토마토 수확 로봇은 近藤 直(1995)등이 연구를 하고 있다. 이것은 논두렁에 레일을 묻고 문모양의 주행대차를 포장 좌우로 걸치게 하여 양 논두렁의 레일

위를 주행하는 것이다. 작업기는 주행장치에 장착시키고, 컴퓨터로 위치를 제어하고, 배터리를 이용하여 일정거리를 주행한 후 정지하도록 되어있다.

오이 수확 로봇은 일본 동경대학(1997)에서 수평방향으로 전선을 사용하여 수평으로 유인하는 방법으로 재배방식을 변경하여 실내 실험을 하였다. 주행장치는 레일을 설치하고, 그 위를 주행하도록 되어있다.

밀감·오렌지 수확 로봇은 Harrell(1990)등이 개발하였다. 밀감·오렌지는 수확시기가 같기 때문에 진동을 이용하여 수확을 시도하였으나, 손상을 입는 과실이 많아 생식용 수확기로는 부적당하였다. 또한 수관이 큰 과수를 수확하기 위해서 매니퓰레이터의 크기가 커야하기 때문에 주행 장치에 장착하여 작업자가 수동으로 조종하였다.

포도 수확 로봇은 대부분의 재배가 선반식으로 하고 있기 때문에 작업자가 위를 향한 자세로 장시간 힘든 작업을 수행하는 문제가 있어 A.Sitticharеonchai(1989)등이 개발하였다. 주행장치는 포장이 불규칙한 경우가 많기 때문에 무한궤도형으로 하였다. 장치의 크기는 전장 23mm, 너비 1400mm, 매니퓰레이터까지의 높이는 420mm이다. 최대 속도는 2m/s까지이며, 무단변속으로의 사용도 가능하다. 주행을 위한 조향은 수동으로 하고 있다.

제 3 절 궤도형 구동시스템

작업기 자체와는 독립적으로 분리되는 순수 주행 시스템을 제작하여 경로를 설정해 주도록 할 필요가 있다. 위치검출을 정확히 하기 위해서는 고정경로 방식을 채택하는 것이 바람직하다. 고정경로의 가장 큰 단점인 경로의 고정성을 보완하기 위하여 X-Y 테이블 방식을 이용할 필요가 있으며, X-Y 테이블 방식을 사용함으로써 작업기는 정해진 지역에서 원하는 어느 위치로도 이동이 가능하고 사용자가 임의로 경로를 선택하여 사용할 수도 있다.

시설내에서 작물의 종류나 기타 다른 상황들로 인하여 고랑의 위치가 바뀔 경우에 고정경로 방식은 레일 등의 설치물의 고랑에 맞추어서 다시 설치해 주

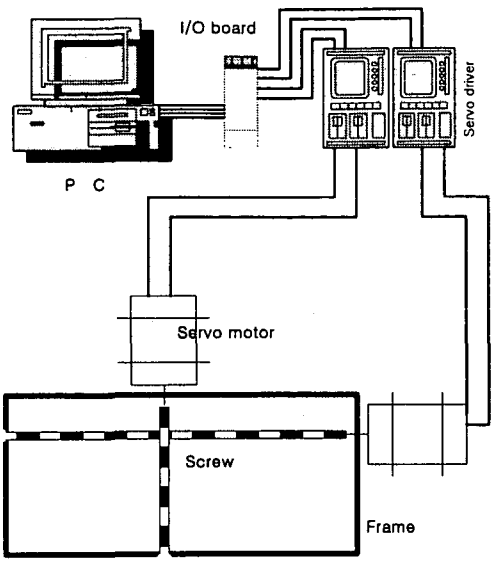
어야만 한다.

그러나 X-Y 테이블 방식을 사용할 경우에는 별도의 설비적인 변화없이 구동 소프트웨어 상의 설정치의 조절만으로도 온실 형태의 변화에 적용시킬 수 있다. X-Y 테이블은 바닥에 설치할 때 크기가 서로 다른 작업기의 운행이나 기타 다른 작업 을 수행할 경우 불편을 주기 때문에 천장에 설치하여 사용하는 방식을 고려하였다. 작업기는 천장의 X-Y 테이블에 연결되어 온실 바닥에서는 떨어져 작업을 수행하게 된다. 이 때 작업기는 구동부와 전력 공급부가 불필요하기 때문에 최대한 경량으로 제작할 수가 있어 작업기 자체 하중에 대해서는 특별한 경우를 제외하고는 주행 상에 문제가 없을 것으로 판단된다.

1. 궤도형 구동 시스템 제어장치 및 구동실험

가. 개요 및 구조

시설재배용 무인작업기를 자동주행 시킬때 가장 중요한 인자는 정확한 위치검출과 직진성 및 선회능력등을 들 수 있다. 본 연구에서는 직진성과 위치검출능력이 뛰어나고 선회시에 별도의 공간이 필요없으며, 시설의 윗 부분에 설치를 해주는 X-Y테이블 방식을 이용하였다. 본 연구에서 제작된 X-Y테이블형 주행 시스템은 모형으로 축소하여 제작하였으며 직사각형 프레임을 제작하여 그 위에 X-Y테이블을 설치하였다.



<Fig. 3-1> Outline of system

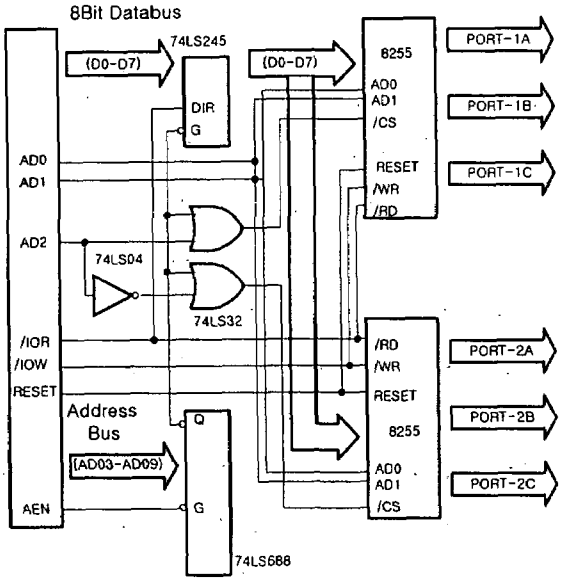
두 개의 스크류를 이용하여 두방향의 직진 주행을 할 수 있도록 설계하였다. 각 스크류 끝에 모터를 부착하여 스크류의 회전을 이용하여 구동시키도록 하였다. 모터의 구동은 PC로 직접 제어를 해 주었으며, 이를 위하여 8255칩을 이용한 입출력 보드를 사용하였다.

<Table 3-1> Specification of AC servo motor

Item	Sepcfication(Unit)
Model	Deawoo 01CA
Nominal power	100(Watt)
Rated torque	3.25(Kg · cm)
	0.32(Nm)
Rated velocity	3000(Rpm)
Maximum velocity	4500(Rpm)
Rated current	1.23(A)
Resolution	2000(Pulse/rev)

나. 모터

X-Y 테이블을 구동시키기 위하여 두 방향으로 제어를 해 주어야 하며, 각 방향으로의 구동은 AC 서보모터를 이용하였다. 각 모터에는 인코더가 부착되어 있으며, 독립적인 서보 드라이버를 사용하였다. 서보드라이버와 외부 컨트롤러와의 연결방법에는 펄스를 조절해 줌으로써 위치를 검출할 수 있는 펄스입력형과 아나로그 신호를 입력하여 속도를 조절해 줄 수 있는 속도입력형의 두가지 방법이 있다. 본 연구에서는 1회 구동시에 정속도회전으로 구동하는 것을 원칙으로 하였으며, 위치검출이 주요사항이므로 펄스입력형 방식으로 구성하였다.



<Fig. 3-2> Layout of I/O interface board

전원의 ON, OFF시에 각 회로의 전원 및 신호송신 사이에 일정간격 이상의 시간을 두어야 한다. 제어전원을 켠 후 정해진 시간간격이 경과되기 이전에 펄스입력신호를 주면 서보 드라이버에서 이를 무시하므로 정확한 위칠검출

을 위하여 서보온 신호의 입력 후 3초 이상의 간격을 두고 펄스입력신호를 주었다. 펄스열의 입력형태에는 FORWARD/REVERSE(CCW/CW) 형태와 PULSE/SIGN형태의 두 가지 종류가 있다. FORWARD/REVERSE형태는 반시계방향 회전시에는 B상 신호에 high를 주고 A상 신호에 펄스를 주며, 시계방향 회전시에는 A상 신호에 high를 주고 B상 신호에 펄스를 주면된다. PULSE/SIGN형태는 A상 신호에 펄스를 주며, B상 신호가 high일 때는 반시계방향 회전을하고 low일 때는 시계방향 회전을 한다. 본 연구에서는 PULSE/SIGN형태의 펄스열 입력형태를 취하였다.

다. 컴퓨터 입출력 및 모터 제어

모터의 제어를 PC에서 직접 제어해 줄 수 있도록 8255칩을 이용한I/O 인터페이스 보드를 제작하여 PC에 부착하였다. 입출력 보드에는 8255칩을 두 개 사용하였으며, 각 칩에는 A, B, C, D 4개의 데이타 버스가 있다. 각 데이터 버스는 8핀으로 구성이 되며, D_0~D_7은 PC의 메인보드와 입출력 보드간의 데이타 버스이고 A, B, C 는 입출력 보드와 외부와의 데이터 버스이다.

PC에서 사용자가 사용할 수 있는 포트의 어드레스는 200(Hex)~3FF(Hex)까지이며, 5개의 딥 스위치를 이용하여서 베이스 어드레스를 설정해 주도록 하였다.

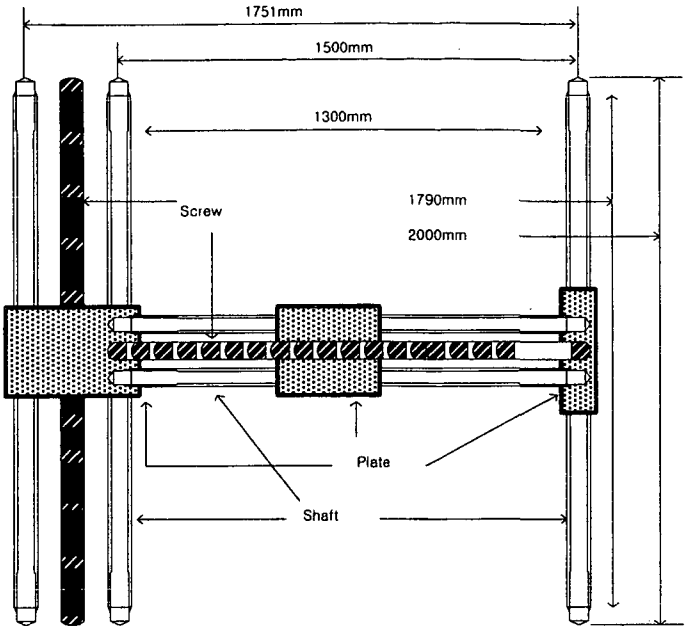
출력되는 모든 신호에 LS04 TTL 칩을 통과시켜 갑작스러운 주전원의 이상시에 모터와의 신호를 차단 시켜주도록 하였다. Servo ON 과 Servo OFF 신호는 24볼트의 전압이 필요하기 때문에 2803칩과 릴레이를 이용하였으며, 24볼트의 전원공급은 파워서플라이를 이용하였다.

라. X-Y 테이블

본 연구에서 사용된 X-Y 테이블은 실제 온실크기보다 작게 축소하여 제작

하였다. 두 방향으로의 위치 이동을 위하여 두 개의 스크류를 사용하였으며, 장축의 길이가 2000㎜, 단축의 길이가 1500㎜이다. 각 스크류의 지름은 20Ø이고, 1회전당 수평이동 거리는 20㎜이다. 스크류에 걸리는 하중을 최소화 시키고, 수평성을 유지시키기 위하여 각 방향의 스크류의 양쪽에 지름 20Ø의 샤프트를 수평 정렬시켰다. 각 샤프트에는 베어링이 들어있는 블록을 2개씩 장착하여 스크류와 연결시킬 수 있도록 하였다. 장축을 구동시키는 모터는 장축구동용 스크류바 끝에 설치하였으며, 단축 구동용 모터는 장축의 스크류와 샤프트블록에 연결되어 X방향으로 움직이는 플레이트를 제작하여 그 위에 설치하여 단축 스크류의 끝단과 연결 시켰다. 단축구동용 스크류와 샤프트 블록에 이동가능한 플레이트를 제작하였으며, 그 플레이트에 작업기를 부착시켜 사용할 수 있도록 하였다. 본 연구에서는 단축에 설치되는 플레이트의 위치를 이용하여 위치검출을 행하였으며, 그 플레이트의 위치가 곧 작업기의 위치가 된다.

모터와 스크류는 플렉서블커플링(Flexible Coupling)을 사용하여 연결시켜 모터축과 스크류를 보호도록 하였다. 모터가 회전을 하면 플렉서블커플링을 통하여 회전력이 스크류에 전달이 되고, 스크류가 이 회전력을 받아 회전을 하면 스크류 위에 설치한 플레이트가 수평 이동을 하게 된다.



<Fig. 3-3> Arrangement outline of screw, shaft and plate

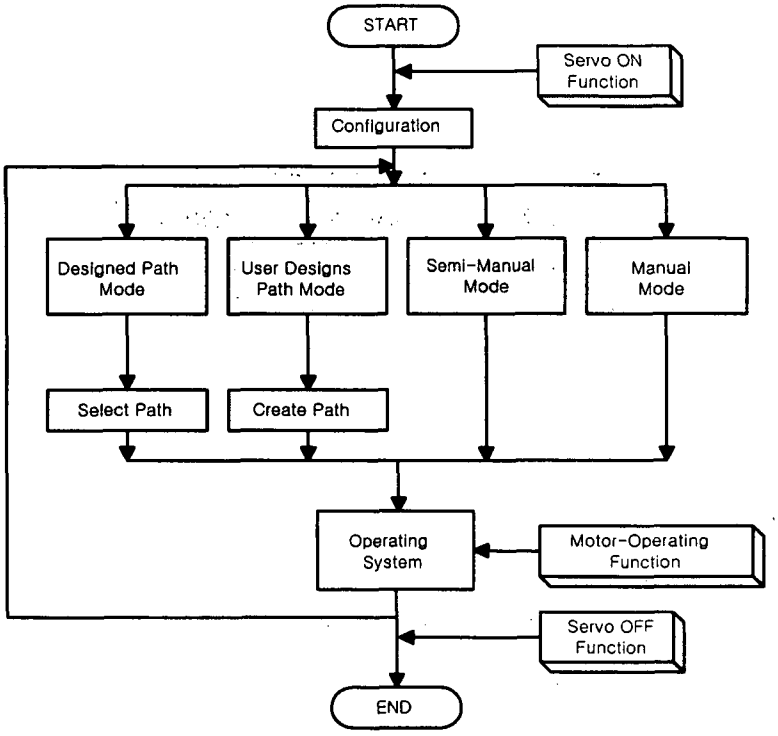
2. 구동 소프트웨어

가. 개요 및 구조

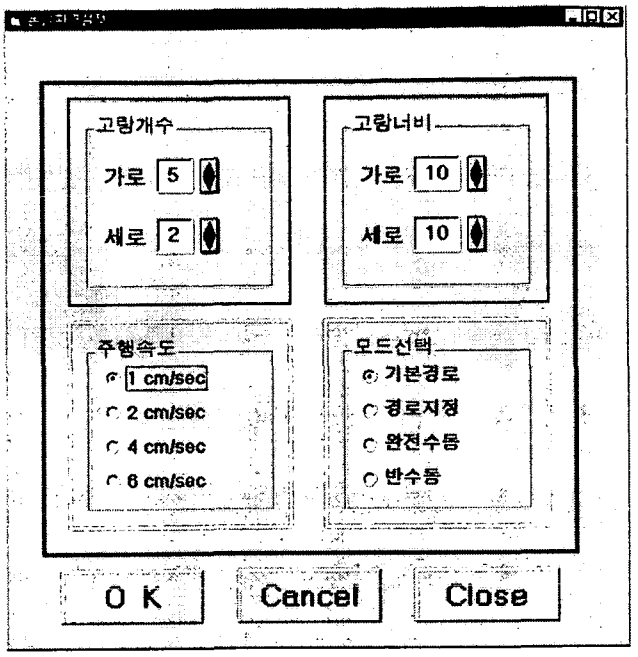
구동 소프트웨어는 크게 두 부분으로 분류될 수 있으며, 한 부분은 화면상의 디스플레이 및 환경설정, 경로지정, 모드선택등을 할 수 있는 경로 설정부이고 다른 한 부분은 모터의 구동을 위하여 직접 하드웨어 포트 입출력을 담당하는 모터 구동부이다. 본 연구에 있어서 메인부는 디자인과정이 쉽고 수정이 용이하며 사용이 간편한 비쥬얼 베이직을 사용하였고, 모터 구동부의 경우에는 하드웨어 포트 입출력이 용이하고 처리속도가 빠른 C++ 을 사용하여 작성하였다.

메인부와 모터 구동부간의 연결은 메인부에서 모터 구동부로의 함수를 호출

해주는 방식으로 하였으며, 이는 비쥬얼 베이직의 OLE 객체 삽입 기능과 외부영역 함수 호출 기능을 이용하여 작성하였다. 메인부에서 모터 구동부로의 함수 호출시 호출 인자로서 이동거리, 이동방향, 주행속도의 세가지를 전달하였다. 이동거리 및 주행속도는 메인 프로그램의 초기화면에 있는 환경설정 모드에서 사용자가 직접 설정을 할 수 있게 하였으며, 이동방향은 지정된 경로 이동 및 수동제어시 사용자가 실시간으로 설정해 주는 방향을 계산하여 전달하였다.



<Fig. 3-4> Flowchart of main program



<Fig. 3-5> Configuration mode(main mode)

모터 구동함수는 메인 프로그램에서 모터의 구동이 필요할 때 메인 프로그램으로부터 호출을 받아 실행이 되며, 모터구동 함수의 호출시에 필요한 호출 인자로는 이동거리, 주행속도, 이동방향등의 세 가지가 있다. 이동방향은 X-Y 테이블상에서 두 개의 축에 각각 2방향으로의 이동이 가능하기 때문에 4가지로 구분하여 이를 메인 프로그램 내에서 수치화시켜 인자로 전달하게 된다. 주행속도와 이동거리는 메인 프로그램의 환경설정 모드에서 사용자가 직접설정을 해 주도록 되어 있다. 이동거리는 메인 프로그램의 환경설정 모드에서 설정되어 있는 값을 직접 전달하게 되며, 모터 구동용 함수에서는 스크류 1회전당 수평이동거리와 모터의 회전수를 이용하여 계산하였다. 주행속도도 메인 프로그램의 환경설정 모드에서 설정해 주는 값을 직접 전달받게 되며, 모터 회전속도와 모터 1회전당 이동거리를 이용하여 계산하였다.

나. 환경설정

환경설정 모드에서는 시스템을 사용자가 원하는데로 구동할 수 있는 정보들을 입력해 주도록 하였다. 환경설정 부분에서 사용자가 지정해 줄 수 있는 인자들로는 가로·세로 고랑의 개수와 주행속도, 가로·세로 고랑사이의 너비 등의 세가지이다.

가로측 고랑의 개수는 일반적으로 인식되는 고랑의 개수를 말하며, 세로측 고랑의 개수는 연동온실에서와 같이 하나의 고랑이 온실의 끝에서 끝까지 연결되어 있지 않고 중간에 나뉘어 지는 단의 수를 의미한다. 가로 및 세로측의 고랑의 개수에 1을 더한 값이 실제 시스템이 가질수 있는 경로의 수이며, 이는 가로·세로 고랑의 외곽부분으로도 작업기가 진행할 수 있다는 전제로 하여 계산한 것이다. 본 연구에서는 고랑의 개수를 5개 까지로 한정을 하였으며, 필요에 따라서는 프로그램상의 상수값을 변화 시킴으로써 최대 고랑수를 늘리거나 줄일 수 있도록 하였다.

고랑사이의 너비는 일정하다는 것을 전제로 하였으며, 가로측과 세록측 고랑사이의 너비는 다르게 설정해 줄 수 있도록 하였다. 본 시스템에서는 가로측과 세로측의 최대 이동거리를 고려하여 고랑의 개수와 고랑사이의 너비를 곱한 값이 최대 이동거리를 넘지 않도록 하였다. 고랑사이의 너비는 센티미터 단위로 설정 할 수 있도록 하였다.

온실에서 작업을 하는 작업기의 경우에는 특별한 경우를 제외하고는 고속주행이 필요하지 않으며, 본 연구에서는 5㎝/sec 에서 15㎝/sec 사이의 주행속도를 갖고 구동될 수 있도록 하였다. 속도는 최소속도와 최대속도 사이에서 4수준으로 구동할 수 있도록 하였으며, 이 값은 환경설정 모드에서 사용자가 설정하여 사용할 수 있도록 하였다. 주행속도의 설정도 고랑개수의 설정과 마찬가지로 프로그램상의 상수를 변화시켜 최소속도와 최대속도를 변화시킬수 있도록 하였다. 모터의 회전속도는 RPM단위로 측정이 되며, 모터 1회전당 2㎝가 이동이 되므로 이를 고려하여 주행속도를 다음식을 통하여 계산하였다.

$$주행속도(cm/\sec) = 모터회전속도(RPM) \times \frac{1}{60} \times 2$$

환경설정 모드에서는 인자들의 설정을 해준 후에 곧바로 하부모드로의 이동이 가능하도록 하였고, 각 하부모드에서 작업이 종료된 후에는 다시 환경설정 모드로 되돌아 가도록 하였다. 다른 서브모드에서 귀환이 된 상태에서도 처음에 설정해 주었던 설정치들은 변하지 않으며, 그 상태에서 사용자가 원하는 서브모드로의 직접이동이 가능하도록 하였다. 고랑의 개수와 고랑사이의 너비등을 직접 설정하여 시스템에 바로 적용시킬 수가 있기 때문에 작업환경의 변화로 인한 시스템 교체의 필요없이 사용할 수 있도록 하였다.

다. 경로지정 모드

사용자가 시스템의 사용목적과 형태에 따라서 임의의 경로를 선택해 줄 경우에 일반적으로 많이 사용되는 경로를 프로그램내에 미리 저장을 해 두어서 이러한 경로를 선택하여 사용하는 경우에는 별도로 사용자가 경로를 일일이 지정할 필요가 없이 저장되어 있는 경로를 로드하여 이용할 수 있도록 하였다.

온실내에서 일반적으로 사용되는 작업기의 경로는 한쪽 방향에서 다른쪽 방향으로의 순차적일 경우가 대부분이며, 이러한 순차적인 경로를 네가지로 분류하여 프로그램내에 저장하여 두었다. 작업기의 진행방향을 가로측 기준과 세로측 기준으로 분류하였으며, 각 기준에서의 작업시작점을 좌측과 우측, 또는 상단과 하단으로 분류하였다. 종류로는 작업의 시작부분과 작업기의 초기 진행방향을 중심으로 "세로방향(좌측먼저)", "세로방향(우측먼저)", "가로방향(상단먼저)", "가로방향(하단먼저)"의 네가지의 형태로 저장시켜 놓았다.

기본적으로 저장되어 있는 경로 이외에도 사용자가 자주 사용하는 경로를 경로지정 모드안에 저장시켜 놓을 수 있도록 하여 사용자가 자주 사용하는 경로가 한가지가 아니고 여러 가지의 경우에도 특별한 조작없이 PC에서 경로를

선택하여 쉽게 사용할 수 있도록 하였다.

경로설정 모드에서는 화면에 환경설정 모드에서 설정을 해 주었던 가로측 및 세로측의 고랑개수를 이용하여 직사각 모양의 온실의 평면도를 이동경로 중심으로 개략적으로 나타내 주었다. 사용자가 선택한 경로를 화면에 이동 순서와 함께 표시하여 주었으며, 시물레이션 기능을 통하여 화면상으로 경로의 진행 방향과 순서를 점검할 수 있도록 하였다.

라. 사용자 경로설정 모드

사용자 경로설정 모드는 경로지정 모드에 저장되어 있지 않은 경로를 사용자가 임의로 지정하여 시스템을 구동시키고자 할 때 이용된다. 경로의 진행 단위는 가로측 및 세로측에서의 고랑과 고랑이 만나는 교차점들 간이며, 이동거리는 환경설정 모드에서 설정해 준 가로 및 세로고랑의 너비에 따르게 된다.

시작점으로 부터 다음 고랑들 사이의 교차점까지의 방향은 버튼을 조작함으로써 부분경로의 설정이 가능하며, 다음번의 부분경로의 설정시에는 이전단계에서 설정해 주었던 끝점이 다시 시작점이 되어 그 점으로부터의 부분경로를 방향설정 버튼으로부터 입력받게 된다. 부분경로를 설정해 주는 과정은 PC의 화면상으로 디스플레이가 되며, 각 방향으로의 선택전에 미리보기 기능으로 사용자가 화면을 통하여 정확한 경로를 선택해 나아가고 있는지를 알 수 있도록 하였다. 부분경로설정들을 통하여 모든 경로의 설정이 끝난 후에는 완성버튼(Complete Button)을 이용하여서 시스템의 주행준비 상태로 전환을 시킬 수가 있으며, 그 이후의 과정은 경로선택 모드와 동일하다.

사용자 경로설정 모드에서도 환경설정 모드에서 지정해 준 가로 및 세로측의 고랑개수를 이용하여 개략적으로 온실에서의 이동가능한 경로를 보여주며, 경로 선택이 완결된 후에 시물레이션 기능을 통하여 선택한 경로의 진행방향과 진행순서를 점검해 볼 수 있도록 하였다. 경로의 설정이 완결된 후에도 시스템을 직접 구동시키기 전에 경로의 수정이 필요한 경우에는 경로를 시작점

으로부터 다시 설정해 줄 수 있도록 하여 사용자의 조작 오류에 의한 수정에 드는 시간을 최대한 단축시켰으며, 간단한 마우스 조작만으로도 경로설정 및 수정이 가능하도록 하였다.

마. 반수동 모드

반수동 모드는 모든 경로를 설정해 주어서 작업이 완료될 때 까지 시스템이 자동적으로 이동을 시켜주는 방법과는 대조적으로 시스템을 동작시키면서 사용자가 그때그때의 상황에 맞게 실시간으로 방향을 설정해 줄 수 있도록 하였다. 주행속도는 환경설정 모드에서 설정해 주었던 속도를 이용해 등속도로 이동을 하게 되며, 사용자는 방향만을 설정해주면 된다.

작업기의 이동거리는 환경설정에서 설정해 놓은 고랑사이의 너비를 이용하고, 1회의 방향설정으로 이동하는 거리는 고랑과 고랑사이의 너비가 된다. 방향설정 버튼을 이용하여 이동방향을 결정해준 뒤 시작버튼을 사용하여 시스템을 이동시키며, 시스템이 직접 구동되기 전에 설정한 방향을 전후좌우의 4가지 방향으로 미리 디스플레이 해 주었다. 경로의 우측 끝단에서 우측으로 이동하는 방향버튼을 클릭하는 등의 작업기가 이동할 수 없는 방향으로의 선택이 이루어지면 경고음을 발생시키고 다시 이전의 상태로 복귀를 하도록 하였다.

작업을 진행하면서 전체의 경로를 순서에 맞게 예측하기 어려운 경우에 반수동 모드를 이용할 수 있으며, 작업의 진행정도와 기타 작업환경등을 고려하여 그때그때의 상황에 맞게 사용자가 직접 유관으로 확인을 하면서 국지적인 경로의 선택을 할 수 있도록 하였다. 사용자가 방향을 설정해 주면 시스템은 설정된 방향으로 다음의 교차점까지는 자동적으로 이동을 할 수 있도록 하였다.

바. 완전수동 모드

완전수동 모드는 이동거리를 시스템이 자동적으로 처리해 주지 않으며, 사용자가 직접 유관으로 확인하면서 정지시킬 위치에 다다랐을 때 정지를 시켜주어야 한다. 환경설정에서 설정해 주는 고랑의 개수와 고랑사이의 거리와는 무관하게 작동을 하며, 단지 사용자가 시작과 정지를 위하여 시스템에 전달하도록 하는 신호에 의해 구동이 된다.

주행속도는 환경설정 모드에서 설정해 주는 속도로 등속도 이동을 하게 된다. 반수동 모드의 경우에 사용자는 방향만을 설정해 주면되고 이동거리는 프로그램에 의해 자동으로 설정이 되는데 반하여 완전수동 모드에서는 사용자가 이동방향뿐 아니라 이동거리까지도 직접 제어를 해 주어야 한다.

사. 모터 구동 프로그램

모터 구동용 프로그램은 AC서보모터의 작동을 시작할 수 있게 하는 서보온(Servo ON)함수, AC서보모터의 작동을 정지시키는 서보오프(Servo OFF)함수와 직접 모터의 회전을 담당하게 되는 모터구동 함수로 분류하였다.

모터 서보온 함수는 메인 프로그램의 실행시에 환경설정 모드가 로드되는 것과 동시에 호출신호를 받아 실행되며, 시스템이 구동 되어질 수 있는 상태로 만들어 준다. 모터 서보오프 함수는 메인 프로그램의 종료 직전에 호출되어 시스템의 구동을 종료 시킨후 리턴이 되어 메인 프로그램이 종료될 수 있도록 하였다.

하드웨어의 베이스 어드레스(Base Address)를 200(Hex)로 지정을 해 주었으며 200(Hex)부터 203(Hex)까지는 장축을 구동시키는 모터의 신호 입출력 어드레스로 설정하였고, 204(Hex)부터 207(Hex)까지는 단축을 구동시키는 모터의 신호 입출력 어드레스로 설정하였다. 서보온/오프(Servo ON/OFF)어드레스는 200(Hex)번지의 신호 입력으로 양축의 모터를 동시에 제어해 주었으며,

그 이외의 입출력신호 설정번지와 펄스 입력번지는 양쪽 모터에 대하여 독립적으로 설정하였다.

아. 모터 구동 함수

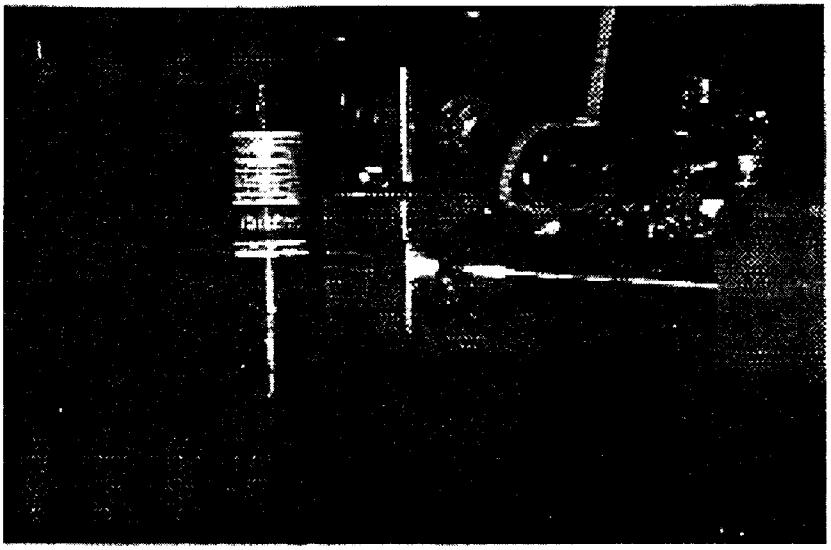
메인 프로그램에서 모터 구동 함수를 호출할 때에는 호출 인자로서 주행속도, 이동거리, 이동방향을 전달하게 된다. 모터 구동 함수가 호출되면 가장 먼저 호출인자들의 입력을 받게 된다. 입력받은 호출인자들을 사용하여 모터를 사용자가 원하는 속도와 거리, 방향으로 구동을 시키게 된다 모터 구동 함수에서 인자들에 특별한 처리를 취하지 않고 그 값들을 직접 사용하기 위하여 메인 프로그램에서 호출인자들을 송신해 줄 때 모터 구동 함수의 폼에 맞게 가공하여 호출하도록 하였다. 모터 1회전당 펄스수는 1000으로 설정을 하였으며, 필요에 따라서 이 값을 변경하려면 모터 구동 프로그램에서의 상수값 변화로 쉽게 늘리거나 줄일 수 있다. 이 때 서보 드라이버의 설정값도 함께 변화시켜 주어야 한다. 작업기의 갑작스러운 출발과 정지는 시스템에 충격을 가하게 되어 시스템의 고장 및 파손의 우려가 있다. 이를 방지하기 위하여 시스템을 출발시킬때의 가속과 정지시킬때의 감속을 행할 수 있는 프로그램을 작성하였다.

3. 실험 장치

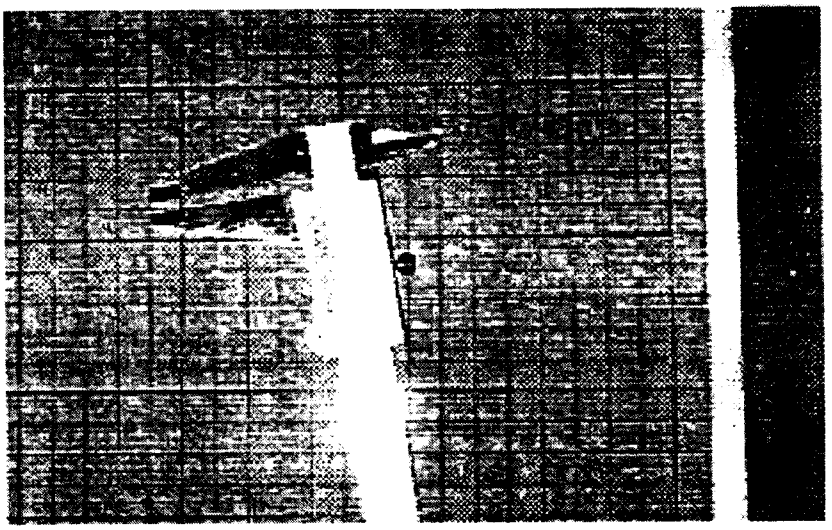
시스템의 구동시에 정확한 위치로의 이동이 이루어지는 여부와 경로를 이동하고 난 후에 다시 정위치로 정확히 귀환하는 것을 알아보아야 한다.

이를 위하여 작업기가 부착되는 위치에 직선으로 투사되는 레이저를 수직방향으로 장착시켜 작업기 부착용 플레이트가 이동할 때 레이저가 수직방향의 바닥으로 투사되도록 하였다. 바닥에는 1mm단위로 표시선을 두었으며, 50mm단위로 좌표를 표기하여 측정하는데 편리하도록 하였다. 좌표는 시스템의 출발

위치가 (0.0)이 되도록 하였다. 눈금판의 좌표는 (-20,-20)에서 시스템의 실제 이동거리인 (1500,1000)보다 큰 (1600,1100)까지로 하여 구동시에 실제 이동거리 밖으로 발생할지도 모르는 오차를 측정할 수 있도록 하였다. 1mm단위 이하로 발생하는 오차는 버어니어 켈리퍼스를 이용하여 0.1mm단위로 측정을 하였다. 측정위치는 눈금에서 투사된 레이저광의 중점을 중심으로 하였으며, 투사된 레이저광의 지름은 약 0.5mm이다.



<Fig. 3-6> Picture of laser process



<Fig. 3-7> The method of errors measuring

4. 실험 방법

1회 구동시에 두 방향을 모두 구동시켰다. 원하는 위치로의 이동시에는 X 방향으로 먼저 구동을 시킨 후에 Y방향으로 구동을 시켰다. 원점으로의 귀환시에는 진행해온 경로와 반대 방향을 가질 수 있도록 X방향으로 먼저 구동시켰다. 1회 측정시에 원하는 위치로의 이동과 귀환은 직사각형 방향으로의 진행이 된다. 지정위치로의 진행때는 직사각형의 상변과 우변을 차례로 그리게 되며, 귀환시에는 직사각형의 하변과 좌변을 차례로 그리게 된다.

X방향으로 2수준(x1,x2), Y방향으로 2수준(y1,y2)으로 하여 실험을 하였으며, X방향 2수준과 Y방향 2수준의 조합으로 총 4개의 위치를 지정하여 실험을 하였다. 각 위치로의 이동과 귀환실험은 구동 소프트웨어상의 환경설정모드에서 설정해 줄 수 있는 4가지의 속도로 각각 측정하였다. 따라서 총 16수준에서의 위치검증실험을 하였고, 각각의 경우에 10회 반복으로 구동실험을 하였다.

X방향으로의 실험거리는 300mm와 1500mm로 하였고, Y방향으로의 실험

거리는 250mm와 1000mm로 하였다. X방향과 Y방향의 조합으로는 (300,250), (300,1000), (1500,250), (1500,1000)의 4가지로 하였고 각각의 경우에 4가지 속도로 구동실험을 하였다. 속도는 5cm/sec, 8cm/sec, 12cm/sec, 15cm/sec로 하였고 각각의 속도로의 구동시에는 가속과 감속을 4cm씩 시켜주었다.

측정은 이동거리를 기준으로 하였다. 지정위치로 이동했을때의 위치를 측정하였고, 그 위치에서 다시 원점으로 귀환했을때의 위치를 측정하여 원점과 비교를 하였다. 각각의 경우에 있어서 발생하는 오차를 중심으로 정밀도를 측정하였으며, 이동거리와의 관계로써 정확도를 측정하였다. 10회 반복 측정시에 오차의 평균은 양수와 음수가 모두 발생할 수 있기 때문에 산술평균을 사용하지 않고, 평균편차를 이용하여 구하였다.

$$D = |d_i| = \frac{1}{n} \sum |d_i|$$

정밀도는 각각의 측정위치에서 발생하는 편차를 가지고 계산하였다. 정밀도는 표준편차를 이용하여 나타낼 수 있으며, 표준편차를 평균으로 나누면 구할 수 있다.

$$\frac{s}{D} = \frac{\left(\frac{1}{\sqrt{n-1}} \sqrt{\sum (x_i - \mu)^2}\right)}{D}$$

정확도는 실제 이동거리를 이동거리의 참값과 비교를 해 주면 구할 수 있다. 평균값과 참값의 차를 참값으로 나눈 후 100을 곱하면 구할 수 있다.

$$\frac{\text{편의}}{\text{참값}} * 100 = \frac{(\text{평균} - \text{참값})}{\text{참값}} * 100 = \frac{\text{오차}}{\text{참값}} * 100$$

5. 결과 및 고찰

각 위치로의 구동은 원점(0,0)에서 출발하였으며 각 표에서 괄호안의 수치는 귀환시에 발생하는 측정치들이다.

가. 편차의 측정

<Table 3-2> Average deviation on every assigned points

Velocity (cm/sec)	Average deviation at assigned point(mm)							
	(300,250)		(300,1000)		(1500,250)		(1500,1000)	
	X	Y	X	Y	X	Y	X	Y
50	0.42	0.29	2.52	0.27	2.59	0.91	2.45	0.68
	(0.39)	(0.31)	(2.57)	(0.33)	(2.60)	(0.85)	(2.46)	(0.65)
80	0.43	0.18	3.06	0.30	2.55	0.99	2.95	0.68
	(0.37)	(0.28)	(3.06)	(0.35)	(2.55)	(1.00)	(2.93)	(0.63)
120	0.27	0.10	2.34	0.26	2.35	1.19	2.97	0.63
	(0.30)	(0.14)	(2.37)	(0.38)	(2.29)	(1.18)	(2.97)	(0.57)
150	0.37	0.14	2.49	0.28	2.40	1.23	2.96	0.56
	(0.35)	(0.25)	(2.50)	(0.29)	(2.43)	(1.20)	(2.97)	(0.46)

각 측정위치에서의 편차는 Y축 방향의 경우에는 거의 전 구간에서 1mm 이하의 오차를 보였고, X축 방향의 경우에는 0.3mm에서 3.0mm사이의 편차를 보였다. 각 방향으로의 편차는 이동거리와는 상관이 없는 것으로 판단이 되며, X축 방향의 오차가 크게 나타나는 것은 눈금판의 제작과정과 바닥에의 고정과정에서 발생한 오차라고 판단된다.

X축의 편차가 같은 X좌표에서도 Y좌표에 따라서 차이를 보이는데 이는 X축과 Y축의 고정시에 축간의 직각 정렬이 정밀하게 맞추어지지 않아서 발생한 오차라 판단된다.

주행속도에 따른 편차의 변화는 거의 없었으며, 한 좌표에서의 속도에 따른 편차 발생의 차이는 X축과 Y축 방향 모두 0.6mm이내였고, 편차의 크기와 속

도와는 무관한 것으로 판단된다.

좀 더 정확한 측정장비를 사용하고 좀 더 정밀하게 양쪽 축간의 직각 정렬을 맞추어 준다면 현재 발생중인 오차를 상당부분 줄일 수 있을 것으로 사료된다.

나. 정밀도의 측정

<Table 3-3> Precision on every assigned points

Velocity (cm/sec)	Average deviation at assigned point(mm)							
	(300,250)		(300,1000)		(1500,250)		(1500,1000)	
	X	Y	X	Y	X	Y	X	Y
50	0.33	0.41	0.07	0.35	0.03	0.18	0.03	0.09
	(0.42)	(0.28)	(0.06)	(0.35)	(0.05)	(0.21)	(0.04)	(0.15)
80	0.36	0.67	0.04	0.25	0.04	0.15	0.04	0.22
	(0.49)	(0.28)	(0.06)	(0.27)	(0.03)	(0.15)	(0.05)	(0.24)
120	0.61	1.15	0.06	0.38	0.04	0.08	0.05	0.21
	(0.40)	(0.90)	(0.08)	(0.24)	(0.03)	(0.08)	(0.07)	(0.26)
150	0.26	0.83	0.04	0.28	0.03	0.13	0.05	0.19
	(0.49)	(0.66)	(0.04)	(0.33)	(0.05)	(0.11)	(0.05)	(0.31)

시스템의 정밀도 측정에 있어서는 거의 전 구간에서 1%미만의 값을 보였고, 특히 Y축 방향으로의 정밀도는 전구간에서 0.38% 이하의 값을 보이고 있다. 시스템의 정밀도는 상당히 좋은 것으로 판단이 되며, 현재 발생하고 있는 오차는 스크류와 각 베어링이 가지고 있는 고유의 허용오차가 크게 영향을 미치고 있는 것으로 사료된다. X축의 정밀도가 Y축의 정밀도보다 낮은 이유는 시스템의 제작시에 X축 방향으로의 정렬이 Y축 방향으로의 정렬보다 조금 부정확하기 때문인 것으로 판단된다. 주행속도의 변화가 정밀도에는 영향을 미치지 않는 것으로 판단이 되며, 시스템의 출발과 정지시에 사용한 가속과 감속 프로그램은 만족할 만한 것으로 판단된다.

좀 더 정밀한 축 정렬기술을 사용하고, 허용오차가 작은 스크류와 베어링을 사용한다면 현재보다 정밀도가 향상될 것으로 사료된다.

다. 정확도의 측정

시스템의 정확도 측정에 있어서도 정밀도와 마찬가지로 거의 전 구간에서 양쪽 축방향 모두 1%이하의 편차를 가지고 있다. 따라서 정확도도 상당히 좋은 것으로 판단이 된다.

한 위치(300,1000)에서의 정확도가 다른 위치에서 보다 상대적으로 많이 낮은 이유는 양쪽축간의 직각 정렬이 정확히 맞지 않아서 발생된 것이라 사료되며, 정확한 계측장비를 사용한다면 정확도가 더 좋아 질 것이라 사료된다.

한 위치(300,1000)를 제외한 전 구간에서는 0.5%이하의 편차를 보여 모터의 회전수 및 회전위치 제어에 있어서는 만족할 만 하다고 판단된다. 편차의 측정 및 정밀도의 측정과 마찬가지로 정확도의 경우에서도 속도에 따른 변화는 없었다.

귀환의 경우에 있어서 설정 위치로의 이동시와 원점으로의 귀환시의 이동거리에는 거의 차이가 나지 않았으며, 이는 모터가 정확한 회전 위치를 유지하면서 회전하는 것으로 판단이 된다. 편차의 측정과 정밀도의 측정에서도 마찬가지로 귀환시에 그 위치로의 이동시와의 오차가 발생하는 것은 시스템 제작시의 정확한 수평 및 수직 정렬과정에서 생긴 기구적 오차와 측정시에 생긴 측정오차 때문인 것으로 판단이 되며, 이동거리와 주행속도에는 관계가 없는 것으로 판단된다.

<Table 3-4> Exactness on every assigned points

Velocity (cm/sec)	Average deviation at assigned point(mm)							
	(300,250)		(300,1000)		(1500,250)		(1500,1000)	
	X	Y	X	Y	X	Y	X	Y
50	0.10	0.12	0.84	0.03	0.17	0.36	0.16	0.07
	(0.13)	(0.12)	(0.84)	(0.03)	(0.17)	(0.34)	(0.16)	(0.07)
80	0.14	0.07	1.02	0.03	0.17	0.40	0.20	0.07
	(0.12)	(0.11)	(1.02)	(0.04)	(0.17)	(0.40)	(0.20)	(0.06)
120	0.09	0.04	0.78	0.04	0.16	0.48	0.20	0.06
	(0.10)	(0.06)	(0.79)	(0.04)	(0.15)	(0.47)	(0.20)	(0.06)
150	0.12	0.06	0.83	0.03	0.16	0.49	0.20	0.06
	(0.11)	(0.10)	(0.83)	(0.03)	(0.16)	(0.48)	(0.20)	(0.05)

6. 요약 및 결론

본 연구에서는 시설재배에서 사용하고 있는 작업기를 무인으로 동작하도록 하기 위한 궤도용 제어시스템을 개발하였다. 이 시스템은 실제 온실 구조의 크기보다는 작게 설계제작하기 위하여 모델화 하였다. 작업기가 무인주행을 할 때 가장 정확하게 위치를 검출하기 위하여 고정경로 방식을 설정하였으며, 고정경로 방식의 단점인 경로의 한정성을 탈피하기 위하여 X-Y테이블 구동시스템으로 설계제작하였다. 이 시스템은 시설재배용 천장레일에서 이동하면서 평평한 온실에서 작업기가 농산물을 취급하기에 적합하게 설계제작한 적정한 구조로 사료 된다. 그 결과를 요약하면 다음과 같다.

1. 실제 온실에서와는 달리 구동축을 지지해 줄 수 있는 프레임을 모델화한 후 제작하였고, 레벨링 풋(Leveling Foot)을 이용하여 수평을 유지하도록 하였다. 두 개의 스크류를 이용하여 두 방향으로의 수평이동이 가능하도록 설계하였고, 이동 플레이트를 제작하여 시설내의 어느 위치로도 이동이 가능하도록 하였다. 모터의 구동은 각각의 독립적인 드라이버를 사용하였고, PC사이의 신호 입출력을 위한 I/O 보드를 직접 제작하였다.

2. 구동 소프트웨어는 경로설정 프로그램과 모터구동 프로그램으로 나누어 개발하였다. 경로설정 프로그램은 사용자 인터페이스 부분이며 비쥬얼 베이직을 사용하였고, 모터구동용 프로그램은 C++언어를 사용하였다. 두 프로그램 및 하드웨어의 인터페이스는 비쥬얼 베이직의 외부함수 호출기능을 이용하였다.

3. 경로설정 프로그램은 다섯부분으로 나누어 작성하였다. 경로선택모드에서는 주로 요구되는 경로를 선택하여 사용하도록 하였고, 사용자 경로설정모드에서는 사용자가 임의의 경로를 설정하도록 하였다. 사용자가 경로를 직접 설정해 줌으로써 고정경로 방식의 최대 단점인 경로의 한정성 문제를 해결하였다.

4. 위치검출 실험에서 정밀도와 정확도 모두 1% 이하로 측정이 되어 작업기의 무인주행에 무리가 없을 것으로 판단되며, 구동축의 설치때 수평 및 수직정렬을 정확이 맞추면 시스템의 정확도 및 정밀도가 0.5% 이내로 좋아질것으로 사료된다. 또한 시설내에서는 저속 주행으로 대부분의 작업이 가능하기 때문에 15cm/sec 이하의 저속으로 실험하였으며, 이 범위에서는 위치검출에 영향이 없는 것으로 나타났다.

제 4 절 유인형 구동시스템

1. 서 론

노동 집약적이던 농업은 급속한 기계화로 인하여 노동력의 절감은 여러 분야에서 이루어지고 있지만, 아직까지도 농업은 수익성이 높지 않아 여전히 사람들로 하여금 노동을 기피되고 있는 산업중의 하나이다. 온실 자동화는 특용 작물 등의 고부가가치 상품에만 한정되어 있으므로 이를 개선해야 한다.

대부분의 기계 및 공정 등이 현재 점차적으로 무인자동화로 발전되고 있으며, 작업의 편리성을 강조하고 있다. 앞으로의 농업기계는 무인자동화 시스템의 도입으로 인하여 누구나 손쉽게 작업설정을 하고 기계 스스로 원활하게 작업을 수행할 수 있도록 개선되어야 한다. 온실내의 무인 자동화, 원활성, 편리성에 대한 연구가 필요한 실정이다.

본 연구에서는 주행경로의 설정이 용이하고 제어가 수월한 고정 경로 방식을 이용한 주행 시스템의 개발과 경로 설정 가이드를 지면에 설치할 경우에는 사람 또는 다른 작업기의 이동시에 불편을 초래하고, 마찰 등에 의한 파손의 문제가 발생할 수 있다. 따라서 지면 설치가 아닌 온실 상부에 설치하여 작업기를 구동시키는 시스템을 개발하고, 온실의 형태, 고랑의 위치 등에 무관하게 주행 경로의 설정을 제어할 수 있는 시스템을 개발하고, 경로설정 가이드와 주행 경로 고정 시스템을 이용하여 자동화 시스템을 개발하였다. 이 개발된 시스템을 이용하여 실증 실험을 하였다.

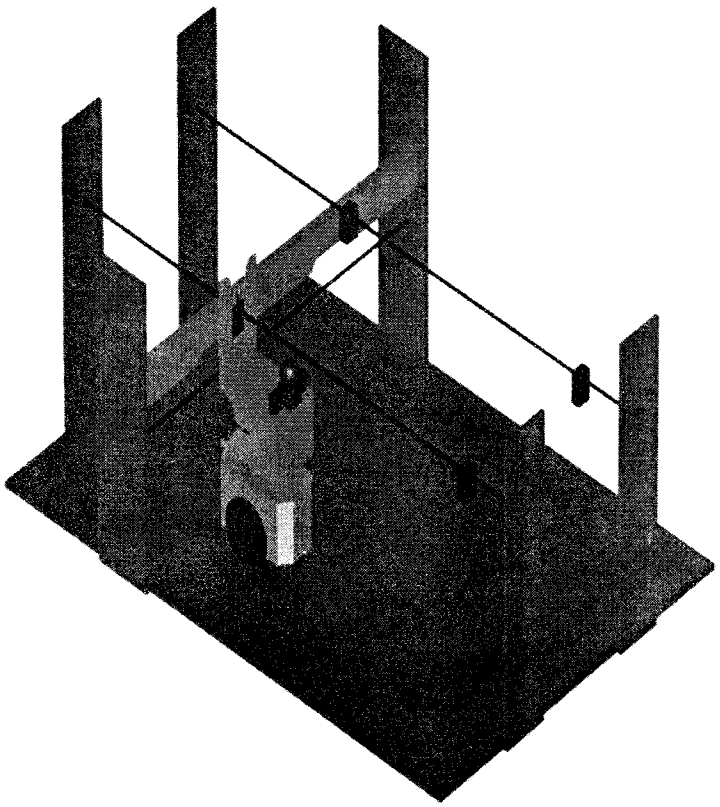
2. 재료 및 방법

가. 대차 시스템의 구조

주행 시스템은 온실상부에 경로 설정 가이드를 설치하고, 작업기에 주행경로

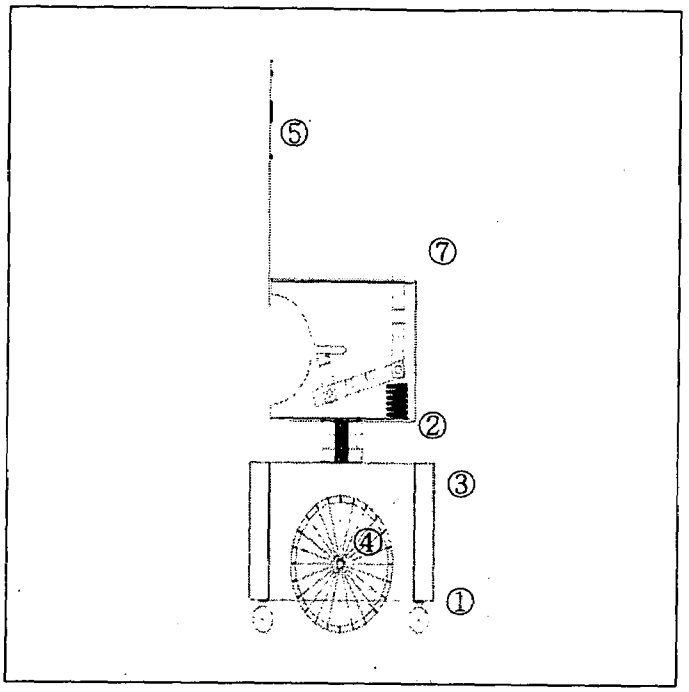
고정 시스템을 부착하는 방식을 채택하였다. 여기서 경로 설정 가이드는 대차가 온실 내부를 이동하는데 있어 일정한 방향으로 원활한 작동을 하도록 하는 것이고, 주행 경로 고정 시스템은 경로 설정 가이드에 부착되는 것이다. 따라서 주행경로 설정과 직진성 및 회전성이 뛰어나 작업 중에 위치 변동이 발생하지 않는다. 구동 시스템은 주행 경로 고정 시스템에 의하여 가이드에 고정되기 때문에 작동 중에 직진성이 향상된다. 또한 구동 시스템이 경로 설정 가이드에 의하여 설정된 경로에서 벗어날 경우 반강제적인 경로를 따라 작업을 수행한다. 제작된 주행경로 고정 시스템은 실제크기로 <Fig. 3-8>과 같이 제작했다.

본 시스템을 직접 온실 내부에 설치를 하는 경우에는 시설의 골격을 이용하여 경로 설정 가이드를 고정시킬 수 있으므로 별도의 외형 틀이 필요하지 않으나, 본 연구에서는 프레스 보드를 이용, 직사각형의 틀을 제작하여 경로설정 가이드를 고정했다. 실험에서는 실제 온실의 약 1/20 크기로 제작하였으며, 외형틀 모형의 사용을 줄이기 위하여 길이를 2,500㎜에서 1100㎜로 줄였다. 그리고 높이는 구동 시스템을 마이크로 마우스로 대신하였기 때문에 이 높이에 맞추어서 500㎜로 제작하였다.



<Fig. 3-8> Carriage system of fixation path formula

대차 시스템은 스테핑 모터를 이용하였고, 대차부의 활동성을 극대화하기 위하여 원칩제어 방법을 사용하였다. <Fig. 2-9>와 <Table 3-5>에서 장치에 대하여 각부의 위치 및 크기를 나타냈다.



<Fig. 3-9> Plan and size of carriage system

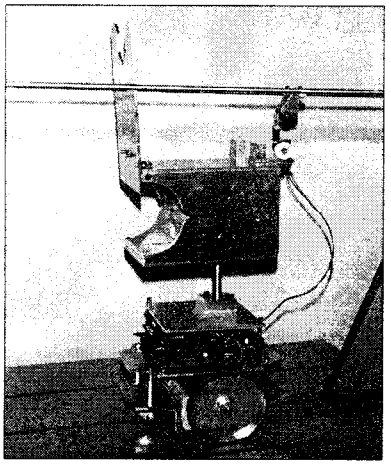
<Table 3-5> General description of carriage system

All standard	Carriage width	Carriage length ①	Carriage height ②
	148㎜	170㎜	125㎜
Part standard	Wheel radius ③	Support wheel radius④	Wheel circumference
	43.2㎜	13.5㎜	271.4㎜
Route fixation system	Horizontal ⑤	Vertical ⑥	Height ⑦
	144㎜	104㎜	106㎜
Route fixing guide	Radius	Height	
	5㎜	75㎜	

※ Wheel circumference : 271.4㎜(2×π×43.2)

<Fig. 3-10>은 실제 제작된 경로 설정 대차 시스템의 모습을 보여주고 있다.

경로 설정 가이드는 직경 4㎜의 연마봉을 사용하였다. 이는 대차 시스템의 직진성을 향상시켜 고랑 밖으로의 탈선을 방지하고, 주행경로를 고정해주는 역할을 한다. 따라서 외부로부터 하중이나 대차 및 주행경로 고정 시스템의 하중은 고려하지 않아도 된다. 가이드의 직경은 주행경로 고정시스템과 맞물리는 부위의 크기 및 연마봉의 처짐 정도를 고려하여 설정하였으며 주행경로 고정 시스템이 회전할 때에 가이드에 의하여 상부의 고정틀이 힘을 받아 눌렸을 때의 스프링의 반력을 초과할 수 있도록 하였다.

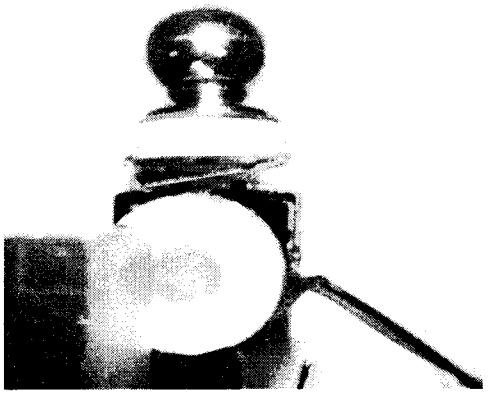


<Fig. 3-10> View of a route fixing guide

<Fig. 3-11>에서 리미트 스위치의 경우 횡축에 설치된 회전 점에 의하여 스위치가 작동이 되고 스위치로부터 전기적인 신호를 받아 구동시스템이 회전을 하도록 설계 제작하였다. 경로 설정이 'ㄷ'자 모양으로 지정되어 있으므로 그 경로에 맞추어 그 회전 값을 초기 우측방향으로 90° , 90° , 180° 회전 후에 그 역회전인 좌측방향으로 90° , 90° , 180° 회전이 가능하도록 초기 설정을 하였다. 차후 다른 형태의 경로에서 사용 시 그 데이터 값만 수정하면 어느 환경에서든지 적용시킬 수 있다.

리미트 스위치는 경로설정 가이드의 횡축과 종축이 교차하는 지점에서 회전

부에 의해 작동하게 된다. 따라서 전·후 또는 좌·우로의 어느 방향에서든지 신호의 입력이 가능하도록 제작하였다.



<Fig. 3-11> Limit switch

나. 구동 시스템

구동 시스템은 고정 경로 가이드를 따라 종·횡 방향으로 이동 방향을 전환한다. 이 때 정확한 90° ,180° 회전을 제어하기 위해 바퀴에 2개의 스테핑 모터를 사용하여 구동하였다. 사용한 스테핑 모터의 제원은 <Table 3-6>에 나타냈다.

<Table 3-6> Specification of stepping motor

Model No.	Holding torque (kgcm)	Current (A/phase)	Voltage (V)	Resistance (Ω /phase)	Rotor inertia (gcm^2)	Overall length (mm)	Weight (Kg)
NK266-02A	9	2	3.6	1.8	300	54	0.7

스테핑 모터의 제어를 하는데 모터 드라이브가 필요하여 이를 SLA7024로

사용하였다. 사용한 모터 드라이브의 제원은 <Table 3-7>에 나타냈다.

<Table 3-7> Specification of motor control drive

Model No.	Rating voltage	FET generating power inside press	Control voltage	T input voltage	Standard voltage	Generating power current	허용 손실
SLA7024	46	100	46	7	2	1.5	4.5

구동 시스템은 마이크로마우스를 이용하여 실험하였다. 구동 시스템의 구동을 제어하기 위하여 16비트 방식의 80C196마이크로 컨트롤러를 사용하였고 제원은 <Table 3-8>에 나타냈다.

<Table 3-8> Characteristic of main CPU

Model No.	Characteristic
80C196KC	488 byte RAM 20㎒ move 28 branch interept source/16 vector Total dual in serial port Behaviour structure 8bit/16bit bus sample/hold ability 8/9 bit A/D convert 232 byte resister file 5 branch 8 bit I/O port 4 branch 16bit software timer 16bit watch timer

다. 환경 설정

환경 설정부분에 있어서 주요 인자는 주행속도, 지면의 형태, 온실의 형태, 고랑 사이의 간격 등이 있다.

온실에서 작업을 하는 작업기의 경우에는 특별한 경우를 제외하고는 고속주행

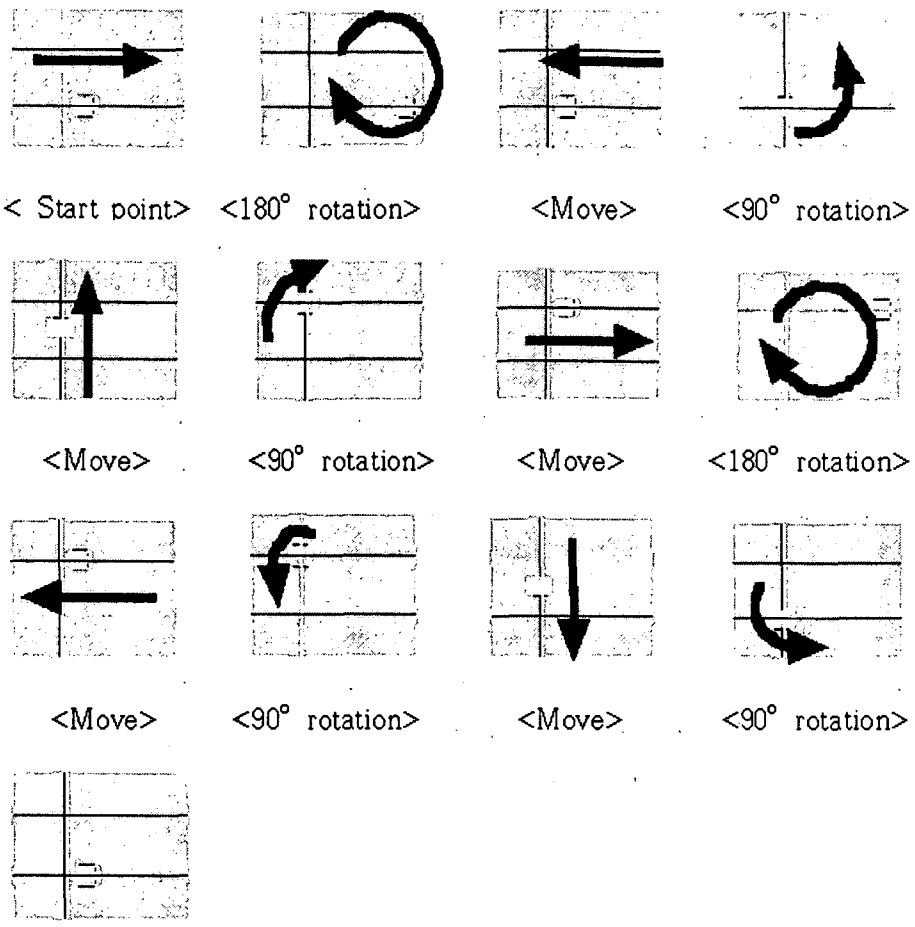
이 필요하지 않지만 각각의 작업에 따라 작업 속도가 다르므로 주행 속도 또한 작업속도에 의해 결정된다. 따라서 주행속도의 변화가 가능해야 하기 때문에 단계별로 속도 변화를 주어 온실 상부 궤도를 따라 이동할 때 지면의 상태에 따른 주행 경로 고정 시스템의 효율성과 적합성 여부를 측정하였다. 즉, 주행 속도에 따라 주행경로 고정 시스템이 구동 시스템의 주행시 직진성을 얼마나 보장할 수 있는가에 대한 실험을 하였다.

지면의 형태는 완전 평면일 경우와 온실 내부의 지면이 고르지 못한 상태 등을 감안하여 프레스 보드 위에서의 실험, 흙표면 위에서의 실험, 모래 위에서의 실험등 세 단계로 나누어 각각의 환경에서 주행경로 고정 시스템이 직진성 향상에 미치는 영향을 측정하였다.

라. 경로지정

고정 경로 방식은 사용자가 시스템의 사용 목적과 형태에 따라서 임의로 경로를 설정할 수 없도록 하였다. 그러나 온실 내부의 고랑사이 간격의 변화로 인하여 무인 자동화 대차 시스템의 경로설정에 변화를 주어야 할 경우에는 특별한 프로그램상의 변화 없이 경로 설정 가이드의 위치변화만으로 그 이동위치의 변화가 가능하도록 하였다.

경로 설정 가이드는 고정부분을 좌우로 이동이 용이하게 설계를 하였기 때문에 간단한 조작만으로 손쉽게 위치의 변화가 가능하도록 제작하였다. 경로 설정 가이드의 위치변환 방식을 <Fig. 3-12>에 나타냈다.



<Fig. 3-12> Moving program

마. 모터 구동 프로그램

본 연구에서는 궤도를 이용한 대차 시스템의 경로 설정에 중점을 두었으므로 구동 소프트웨어는 본 연구에 맞추어서 프로그램을 구현하였다. 모터 구동 프로그램은 IC96-C컴파일러를 이용하여 헥사 파일로 변환시킨 후 MAX-232를 이용하여 시리얼 케이블을 통해 구동 시스템의 메인보드의 RAM에 다운로드 시켜서 구동하였다.

모터 구동 프로그램은 시스템의 구동이 시작과 종료를 위해 모터 온·오프 함수, 한 스텝각(1.8°)만큼 스테핑 모터를 구동시키는 원스텝함수, 리미트 스위치로부터 입력을 받는 스위치체크 함수, 이동 방향을 전환하는 턴 함수 등으로 구성되어 있으며 이 함수들을 호출하여 스테핑 모터를 구동시키는 메인 함수로 구현하였다

모터를 구동시키는 메인 함수는 펄스수, 이동 방향, 리미트 스위치의 카운터 값을 인자들로 하여 모터를 구동시킨다. 이동 방향과 펄스수는 프로그램 상에서 수동적으로 수정하는 방법을 사용하였다. 리미트 스위치의 카운터 값은 대차 시스템의 이동 방향이 전환되어야 하는 지점에서 리미트 스위치로부터 실시간으로 직접 입력을 받았다 .

온실 내부에 이동 경로가 'ㄷ'형 또는 'ㅌ'형인데 본 실험에서는 'ㄷ'형을 채택하였다. 좌측 하단에서 대차 시스템의 구동을 시작하여 각 방향 전환 지점에서 180° , 90° , 90° , 180° , 90° , 90° 로 방향 전환을 하게 하였다.

대차 시스템은 리미트 스위치의 입력을 받고 펄스수에 따라 90° 또는 180° 로 제자리에서 방향만 전환 후에 전진하게 하였다. 90° ,180° 로 이동 방향을 전환하게 하는 펄스수는 다음과 같은 식에 의거하여 계산하였다.

$$1\text{펄스당 전진하는 거리} : \frac{\text{스텝각}}{360^\circ} \times \text{차바퀴둘레}$$

$$360^\circ \text{ 회전시} : 2 \times \pi \times \frac{\text{차축간의거리}}{2} \times 1\text{펄스당 전진하는 거리}$$

$$180^\circ \text{ 회전시} : 2\times\pi\times\frac{\text{차축간의거리}}{2}\times 1\text{펄스당 전진하는 거리}\times\frac{1}{2}$$

$$90^\circ \text{ 회전시} : 2\times\pi\times\frac{\text{차축간의거리}}{2}\times 1\text{펄스당 전진하는 거리}\times\frac{1}{4}$$

본 실험에서 대차 시스템의 차축간의 거리는 132㎜이고 1펄스당 전진하는 거리는 스테핑 모터의 스텝각이 1.8° 이므로 1.357㎜계산되었다. 대차가 180° 회전시에는 모터에 가해주는 펄스를 153으로 사용하였고 90° 회전시에는 76 펄스를 모터에 설정하였다.

바. 실험방법

시스템의 구동 시에 정확한 위치로의 이동이 이루어지는지의 여부와 정확한 지점에서 회전성 및 직진성 향상 여부를 알아보아야 한다.

실험은 속도의 변화를 세 단계(0.1m/s, 0.5m/s, 0.9m/s)로 분류하여 측정하였고, 지면의 상태(평면, 흙, 모래)를 변화시켜, 가능한 실제 온실과 같은 환경으로 조건을 맞추어 <Table 3-9>와 같이 설계하였다.

<Table 3-9> Experimental design

Ground / Velocity	Plane surface	Soil surface	Sand surface
Slow speed (0.1m/s)	TEST 1	TEST 2	TEST 3
Middle speed (0.5m/s)	TEST 4	TEST 5	TEST 6
Quick speed (0.9m/s)	TEST 7	TEST 8	TEST 9

3. 결과 및 고찰

가. 구동 시스템의 안정성

무인 자동화 대차 시스템은 부착되는 작업기의 종류에 따라 그 작업속도 및 주행 속도의 차이가 발생하므로 모터의 회전속도의 변화를 주어서 각 속도마다 시스템의 구동시에 정확한 위치로의 이동이 이루어지는 지의 여부와 정확한 지점에서의 회전성 여부를 알아보기 위하여 프로그램상의 모터 제어부분에서 그 속도를 달리하여 실험을 하였다. 또한 지면의 형태는 완전 평면일 경우와 온실 내부의 지면이 고르지 못한 상태 등을 감안하여 프레스 보드 위에서의 실험, 흙 표면 위에서의 실험, 모래 위에서의 실험등 세 단계로 나누어 각각의 환경에서 어떠한 반응을 나타내는지를 관찰하였으며 이를 통하여 온실이 위치한 지리적 특성 등을 고려하여 사용할 수 있도록 하였다.

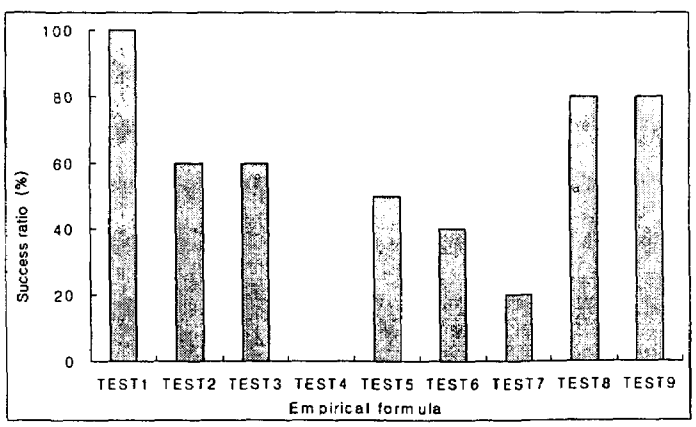
각 환경에 따라서 시설재배용 대차 시스템이 고정 경로를 따라 직선 이동 및 회전을 하면서 출발점으로 돌아오는데 까지의 성공 여부를 측정하였다. 실험은 각각 10회 이상 측정하여 그 평균치를 산출하였고, 이를 바탕으로 최적의 작업환경을 찾아냈다.

주행경로고정 시스템이 회전 지정부를 통과한 횟수를 N이라고 하였으며, 경로고정 시스템이 올바른 작동횟수를 B라고 하였다. 이때, N의 최대값은 6이고, B의 최대값은 1이다. 정상 작동상태는 B와 N이 모두 최대값을 가질 때라고 가정하였다.

구동 시스템은 전복에 대하여 스스로 회복능력이 없기 때문에 강제적으로 직진성을 향상시킬 수 없었다. 실험은 TEST 1에서 즉, 평면, 저속의 상태에서 100%의 정확한 측정치를 얻을 수 있었다.

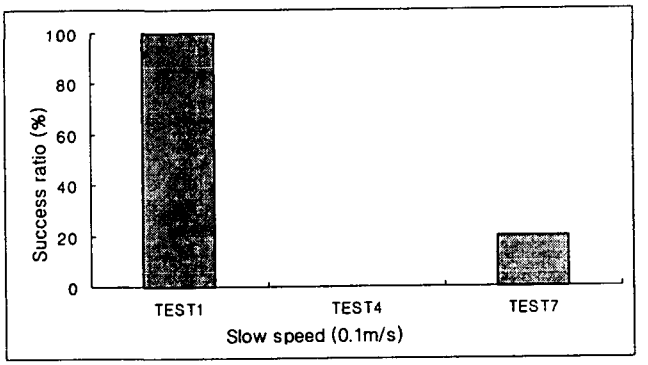
<Table 3-10> Success ratio of ground and velocity

Surface	Plane surface						Soil surface						Sand surface					
Velocity(%)	0.1		0.5		0.9		0.1		0.5		0.9		0.1		0.5		0.9	
Determination variable number	N	B	N	B	N	B	N	B	N	B	N	B	N	B	N	B	N	B
1	6	1	6	1	6	1	1	·	2	·	2	·	2	·	6	1	2	·
2	6	1	·	·	·	·	1	·	6	1	6	1	2	·	6	1	6	1
3	6	1	6	1	·	·	2	·	6	1	6	1	1	·	2	·	6	1
4	6	1	6	1	6	1	1	·	1	·	1	·	1	·	6	1	6	1
5	6	1	2	·	6	1	2	·	2	·	1	·	2	·	6	1	2	·
6	6	1	6	1	1	·	2	·	2	·	6	1	6	1	2	·	6	1
7	6	1	6	1	1	·	2	·	2	·	6	1	2	·	6	1	6	1
8	6	1	1	·	6	1	1	·	6	1	2	·	6	1	6	1	6	1
9	6	1	6	1	6	1	2	·	6	1	1	·	1	·	6	1	6	1
10	6	1	·	·	6	1	1	·	6	1	1	·	2	·	6	1	6	1
Success number	10		6		6		·		5		4		2		8		8	
%	100		60		60		0		50		40		20		80		80	



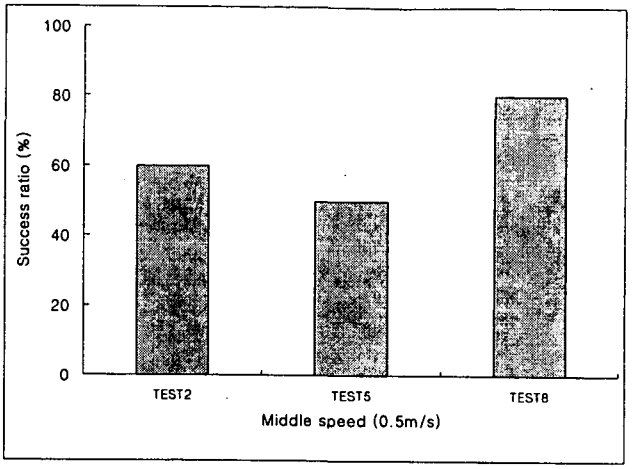
<Fig. 3-13> Success ratio of ground and velocity

<Fig. 3-14>에서 보여주듯 평면에서 저속으로 실험을 한 TEST 1에서 100%로 가장 높은 성공률을 보였으며, 흙 표면에서 저속으로 실험한 TEST 4와 TEST 7의 경우에는 슬립이 발생하여 성공률이 낮게 나타났다.



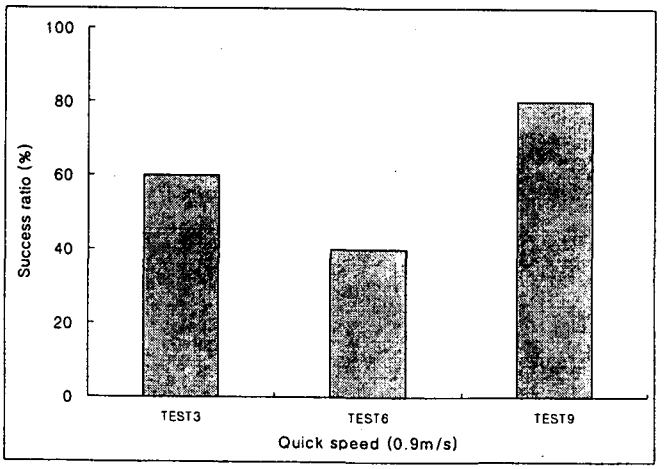
<Fig. 3-14> Success ratio of slow speed

<Fig. 3-15>에서 0.5m/s의 속도로 주행 성능을 실험한 결과 TEST 2, TEST 5, TEST 8의 경우 각각 60%, 50%, 80%의 성공률을 나타냈다. TEST 8의 실험을 통하여 중속으로 주행을 할 경우에는 저속보다는 균형성과 속도에 따른 견인력 향상으로 인한 안전성이 증가하는 것으로 나타났다.



<Fig. 3-15> Success ratio of middle speed

<Fig. 3-16>에서 0.9m/s의 고속에서 주행 성능을 실험한 결과 TEST 3, TEST 6, TEST 9의 경우 각각 60%, 50%, 80%로 나타났다. <Fig. 3-15>와 <Fig. 3-16>를 통하여 살펴보면 속도가 0.5m/s에서 0.9m/s증가해도 영향이 크게 않게 나타났다. <Fig. 3-16>를 통해 고속에서 지면 상태가 모래인 경우에 평면과 흙표면일 때 보다 균형성, 견인력 및 안정성이 증가하여 높은 성공률을 보였다.



<Fig. 3-16> Success ratio of quick speed

나. 궤도시스템의 문제점 분석

측정은 이동거리를 기준으로 하였다. 지정 위치로 이동했을 때의 위치를 측정하였고, 그 위치에서 다시 원점으로 귀환했을 때의 위치를 측정하여 원점과 비교를 하였다. 각각의 경우에 있어서 발생하는 오차를 중심으로 정밀도를 측정하였으며, 이동거리와의 관계로써 정확도를 측정하였다.

10회 반복 측정시에는 오차의 평균은 양수와 음수가 모두 발생할 수 있기 때문에 산술평균을 사용하지 않고, 평균편차를 이용하여 구했다.

$$D = |\overline{d_i}| = \frac{1}{n}\sum |d_i|$$

정밀도는 각각의 측정위치에서 발생하는 편차를 가지고 계산하였다. 정밀도는 표준편차를 이용하여 나타낼 수 있으며, 표준편차를 평균으로 나누어서 구했다.

$$\frac{s}{D} = \frac{\left(\frac{1}{\sqrt{n-1}}\sqrt{\sum(x_i-\mu)^2}\right)}{D}$$

정확도는 실제이동거리를 이동거리의 참값과 비교를 해 주면 구할 수 있다. 평균값과 참값의 차를 참값으로 나눈 후 100을 곱하여 구했다.

$$\frac{\text{편의}}{\text{참값}} \times 100 = \frac{(\text{평균} - \text{참값})}{\text{참값}} \times 100 = \frac{\text{오차}}{\text{참값}} \times 100$$

각 위치로의 구동은 원점(0.,0)에서 출발하였으며 각 표에서 괄호안의 수치는 귀환시에 발생하는 측정치들이다. 각 측정위치에서의 편차는 Y축 방향의 경우

에는 거의 전 구간에서 1mm이하의 오차를 보였고, X축 방향의 경우에는 약 0.3mm에서 3.0mm 사이의 편차를 보였다.

각 방향의 편차는 이동거리와는 상관이 없는 것으로 판단이 되며, X축 방향의 오차가 크게 나타나는 것을 눈금판의 제작과정과 바닥에의 고정과정에서 발생한 오차라고 판단되었다. X축의 편차가 같은 X좌표에도 Y좌표에 따라서 차이를 보이는데, 이는 X축과 Y축의 고정시에 축간의 직각 정렬이 정밀하게 맞추어지지 않아서 발생한 오차라 이고, 주행속도에 따른 편차의 변화는 거의 없었으며, 한 좌표에서의 속도에 따른 편차발생의 차이는 X축과 Y축 방향모두 0.6mm이내였고, 편차의 크기와 속도와는 무관한 것으로 판단하였다. 더 정확한 측정장비를 사용하고 더 정밀하게 양쪽 축간의 직각 정렬을 맞추어 준다면 발생중인 오차를 상당부분 줄일 수 있다.

시스템의 정밀도 측정에 있어서 거의 전 구간에서 1% 미만의 값을 보였고, 특히 Y축 방향으로의 정밀도는 전구간에서 0.38% 이하의 값을 보였다. 시스템의 정밀도는 상당히 좋은 것으로 판단되며, 현재 발생하고 있는 오차는 스크류와 각 베어링이 가지고 있는 고유의 허용오차가 크게 영향을 미치고 있는 것으로 판단되었다. X축의 정밀도가 Y축의 정밀도 보다 낮은 이유는 시스템의 제작시에 X축 방향으로의 정렬이 Y축 방향으로의 정렬보다 조금 부정확하기 때문이다.

주행속도의 변화가 정밀도에는 영향을 미치지 않는 것으로 판단이되며, 시스템의 출발과 정지시에 사용한 가속과 감속 프로그램은 만족 할만 한 것으로 판단되었다.

시스템의 정확도 측정에 있어서도 정밀도와 마찬가지로 거의 전 구간에서 양쪽 축방향 모두 1%이하의 편차를 가지고 있다. 따라서 정확도도 높게 나타났다. 한 위치 (300, 1000)에서의 정확도가 다른 위치에서 보다 상대적으로 많이 낮은 이유는 양쪽 축간의 직각 정렬이 정확히 맞지 않아서 발생된 것이라 사료되며, 정확한 계측장비를 사용한다면 정확도가 더 좋아질 것이다. 한 위치 (300, 1000)를 제외한 전 구간에서 0.5%이하의 편차를 보여 모터의 회전수 및

회전위치 제어에 있어서는 만족할 만 하다고 판단된다. 편차의 측정 및 정밀도의 측정과 마찬가지로 정확도의 경우에서도 속도에 따른 변화는 없었다.

귀환의 경우에 있어서 설정 위치로의 이동시와 원점으로의 귀환시의 이동거리에는 거의 차이가 나지 않았으며, 이는 모터가 정확한 회전 위치를 유지하면서 회전하는 것이다. 편차의 측정과 정밀도의 측정에서도 마찬가지로 귀환시에 그 위치로의 이동시와의 오차가 발생하는 것은 시스템 제작시의 정확한 수평 및 수직 정렬과정에서 생긴 기구적 오차와 측정시에 생긴 측정오차 때문인 것으로 판단이 되며, 이동거리와 주행속도에는 관계가 없는 것이다.

4. 요약 및 결론

본 연구에서는 시설재배에서 사용하고 있는 작업기를 무인으로 동작하도록 하기 위한 제어 시스템을 제작하였다. 실험을 위하여 실제의 온실을 제작할 수 없으므로 프레스 보드를 이용하여 경로 설정 가이드를 고정시켰으며, 직경 4㎜의 연마봉으로 경로 설정 가이드를 대신하였다. 경로 설정 가이드의 경우에는 실제 온실에서 사용할 경우 작업기 하중의 영향을 받지 않는다. PC로부터 입력의 경우 이동성능을 높이기 위하여 RAM으로 데이터를 전송하는 방법을 채택하였다. 또한 대차 시스템에 필요한 장비를 부착하여 사용할 수 있도록 제작하기 위하여 특정 작업기를 사용하지 않고 마이크로 마우스를 사용하여 그 이동 주행 여부를 실험하였다. 구동 소프트웨어는 리미트 스위치에 의한 회전 설정 프로그램과 모터 구동 프로그램의 두 부분으로 되어있다. 회전 설정 프로그램은 횡축과 종축의 경로 설정 가이드가 교차하는 지점에서 리미트 스위치로부터 전기적인 신호를 받으면 회전할 수 있도록 C 언어를 사용하여 구현하였다. 모터 구동용 프로그램은 C언어를 사용하였다. 경로 설정 가이드에 따른 직선 이동 실험에서 지면 상태에 관계없이 전체 성공률이 54.4%로 측정되었다. 각 작업마다의 작업속도를 고려하여 0.1㎧ ~ 0.9㎧ 까지 속도 변화를 주어 실험하였으며 0.1㎧의 경우에는 평면 상태에서 100%의 성공률을 나타냈다. 0.5㎧

와 0.9m/s 속도의 경우에는 모래 토양에서 속도의 증가로 인한 균형성, 견인력 및 안정성이 증가하여 성공률이 80%로 증가하였다. 대차 시스템은 자기 수복 능력을 갖춘 구동장치로 교체한다면 온실내 무인 주행 시스템으로 최적의 효과를 낼 수 있을 것이다.

제 5 절 대차형 구동시스템

1. 서 론

현재 농업분야에서 이용되고 있는 차량의 주행장치로는 휠형과 궤도형이 이용되고 있다. 특히 연약지에서 많이 운용되고 있는 농용 휠형차량은 높은 접지압 때문에 토양의 다짐에 의해 작물의 생산량을 감소시키는 것으로 보고되고 있고(Aura,1983), 수분을 전달하고 유지하는데 필요한 안정된 토양구조에 손상을 준다는 보고도 있다(Ayers, 1987).

휠형차량은 궤도형 차량에 비해 그 구조가 간단하고 내부 에너지의 손실이 적고, 단단한 지형에서 기동성이 좋은 장점을 갖고 있으나 연약한 지형에서는 높은 접지압으로 침하가 크게 발생하기 때문에 주행저항이 크게 나타나 견인성능면에서 비효율적이다. 이에 비해 궤도형차량은 구조가 복잡하고 내부 운동저항이 커서 에너지 손실은 많으나 휠형에 비해 같은 중량의 차량일 경우 접지압이 작으므로 연약지에서 높은 견인성능을 낼 수 있는 장점이 있다. 따라서 농업용으로 로외에서 사용하는 차량에는 이러한 궤도형 차량이 많이 사용되고 있다. 그러나 일반적으로 사용되고 있는 강성 궤도형 차량은 무게가 많이 나가고 주행부가 단단하기 때문에 골의 손상 및 표면토층의 파괴등 많은 문제점을 가지고 있다.

주행장치는 궤도형과 유인형 및 대차 시스템으로 나눌 수 있다. 시설재배의 경우 궤도형 및 유인형을 많이 사용하고 있다. 궤도형의 경우는 궤도를 바닥에 설치하는 방법이며, 유인형의 경우는 온실 상부에 유인 장치를 하는 것이다.

궤도형은 토양 슬립이 적어 방향 전환이 안정적인 궤도형 주행장치를 많이 연구하고 있으나, 초기 투자비가 많이 들고 작업자의 작업 공간 축소에 따른 안전사고의 위험이 따르고 있다. 유인형의 경우는 온실 상부에 레일 및 연마봉을 설치하고, 주행장치가 상부에 고정되어 주행되는 것이다. 이 경우 작업공간의 확보는 가능하지만, 초기 투자비가 많이 들고 기존의 농가에서 설치하는데 있어서 어려움이 많다. 또한 채광이 좋지 않아 작물의 생육환경에 불리한 단점이 있다. 대차 시스템은 초기 투자비가 궤도형, 유인형에 비해 적게들고, 작업 형태에 따라 적용범위가 크지만 소프트웨어적으로 제어가 어려운 단점을 가지고 있다.

2. 재료 및 방법

가. 실험 재료

대차 시스템은 고온 다습한 온실에서 작동하기 때문에 각 부품에 대한 설계는 드랄니늄과 ABS 수지를 이용하여 설계하였다. 대차 시스템에 사용된 모터는 일본 TAMAGAWA SEIKI사의 TS3134N9을 사용하였다. 모터의 토크는 대차와 매니퓰레이터 및 엔드이펙터의 부하하중을 60kgf로 하여 설계 부하를 계산하였다. 대차 시스템은 바퀴를 각각 구동할 수 있도록 4개의 모터를 두었으며, 중앙부에 방향전환이 가능하도록 모터를 설치하였다. 중앙부의 모터가 작동하면 링크로 연결된 각각의 바퀴에 동력이 전달되며, 4개의 바퀴는 바퀴 고정축이 힌지로 연결되어 있기 때문에 회전이 가능하도록 하였다. 바퀴는 이탈 및 고정을 위해서 바퀴 고정축과 모터 고정축을 두어 주행 및 회전 중에 모터의 부하 및 진동을 방지하였다. 온실내의 고랑을 따라 주행 중 방향전환은 미리 정해놓은 거리에 따라서 일정거리를 주행 후 자동으로 방향전환이 가능하도록 하였다. 거리는 모터의 회전수와 바퀴의 지름과의 관계를 이용하여 정하였다.

대차가 작업하는 지형은 포장이 되어 있는 도로가 아니기 때문에 4개의 구동 바퀴가 모두 지면에 접촉하는 것이 아닌 경우가 발생할 수 있기 때문에 3개의 바퀴만이 지면에 밀착할 경우를 가정하여 다음과 같은 방법으로 부하를 계산하였다.

$$W_o = \frac{(\text{부하하중} + \text{외력}) \times \text{안전율}}{\text{바퀴수}}$$

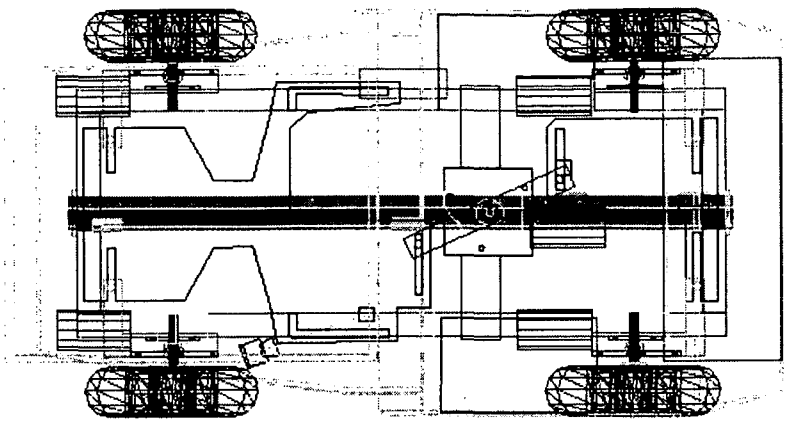
$$= \frac{(60 + 5) \times 1.5}{3} = 32.5\text{kgf}$$

바퀴의 경우 링크에 의해 힌지로 고정되어 있으면서 대차 시스템이 작동중 방향전환을 하는 거리에 가면 대차 시스템이 정지하게 되고 중앙부의 모터가 작동하면 4개의 바퀴가 정회전 90° , 180° 및 역회전 90° , 180° 로 회전이 가능하도록 설정을 하였다. 만약 다른 경로에서 사용 시 거리에 대한 데이터 값만 수정하면 어는 환경에서든지 적용시킬 수 있다. 대차의 경로설정은 고랑이 끝나는 지점까지의 거리를 PC에 저장하여 놓고 바퀴의 지름을 감안하여 모터의 회전수를 결정하였다. 환경 설정부분에 있어서 주요 인자는 주행속도, 지면의 형태, 온실의 형태, 고랑 사이의 간격 등이 있다.

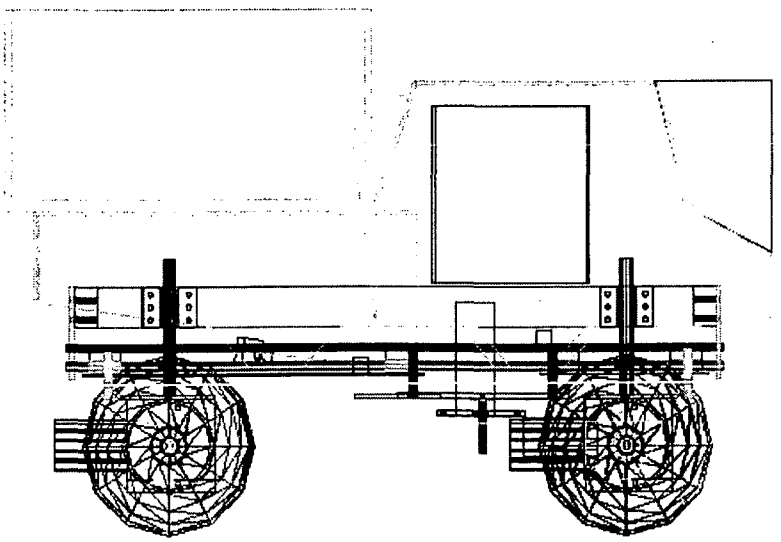
온실은 직사각형의 밑면을 기본으로 하고 있기 때문에 대부분의 경우 90° 및 180° 로 회전하면 큰 각도상의 문제점은 발생하지 않지만 경우에 따라서는 ±15° 정도의 오차를 보이는 온실도 있다. 무인 자동화 대차 시스템은 온실내의 상황에 따라서 'ㄷ'자 형태의 90° 또는 180° 의 회전을 원칙으로 하고 있지만, 상황에 따라서는 90° 및 180° 이외의 회전이 발생할 우려가 있기 때문에 고랑의 형태를 PC에 저장하면 상황에 맞추어서 대차 시스템이 회전할 수 있도록 되어있다.

본 연구에서는 대차 시스템의 경로 설정에 중점을 두었으므로 구동 소프트웨어는 Visual C++프로그램으로 구현하였다. 설계는 3dmax을 이용하였으며, 구동은 4개의 스템핑 모터를 이용하였다. <Fig. 3-17>은 정면도, <Fig. 3-18>

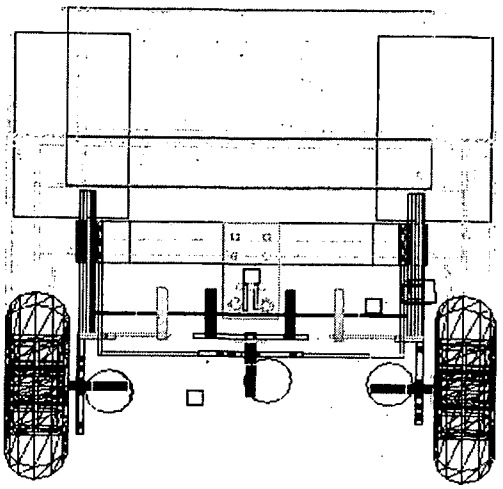
는 측면도, <Fig. 3-19>은 우측면도, <Fig. 3-20>는 입체도를 나타내고 있다



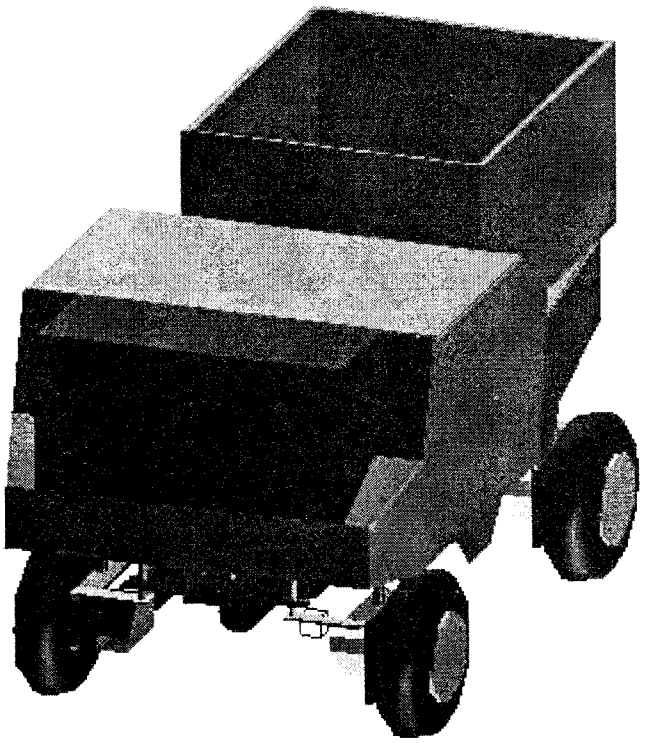
<Fig. 3-17> Top view



<Fig. 3-18> Front view



<Fig. 3-19> Right view



<Fig. 3-20> Perspective view

나. 실험방법

재배 양식에 따른 궤도 시스템은 환경보다는 시설에 의한 영향을 받기 때문에 대차 시스템을 이용하여 적응성을 알아보았다. 재배 양식에 따른 적응성은 <Table 3-11>와 같이 일반농가, 직립 후 횡 유인, 직립 후 가로줄 유인의 3가지 형태의 유인 방법과 0.1m/s, 0.5m/s, 1.0m/s의3가지 속도에 따라 각각의 경우에 대하여 통로를 10회 반복 횟수에 대하여 대차 시스템의 적응성에 대하여 실험 설계를 하였다.

유인방법 중 일반농가는 관행적으로 재배하는 방식으로서 2m의 높이에 정

식간격 24cm마다 비닐 끈을 늘어뜨려 여기에 덩굴을 매달아 놓고 성장에 따라 줄을 내려주어 재배하는 방법이다. 직립후 횡유인의 경우는 2m의 높이에 봉을 X축으로 설치하여 덩굴이 봉을 따라 성장하는 방법이다. 직립후 가로줄 유인의 경우는 직선으로 성장시킨뒤 X축으로 5° 이동시켜 재배하는 방법으로 2m의 높이에 20cm간격으로 철사줄을 연결하여 유인하는 방법이며 적심재배을 하여 측지로 부터 오이를 수확하는 재배방법이다.

<Table 3-11> Experimental design

Induction method / Velocity	Normal farmhouse	Horizontal Induction after erection	Horizontal line Induction after erection
0.1m/s	Test 1	Test 2	Test 3
0.5m/s	Test 4	Test 5	Test 6
1.0m/s	Test 7	Test 8	Test 9

스텝핑모터는 TAMAGAWA SEIKI사의 모델은 TS3134N9이며 토크는 150이고 24볼트의 3.1암페어를 가지고 있는 것을 사용하였다.

3. 결과 및 고찰

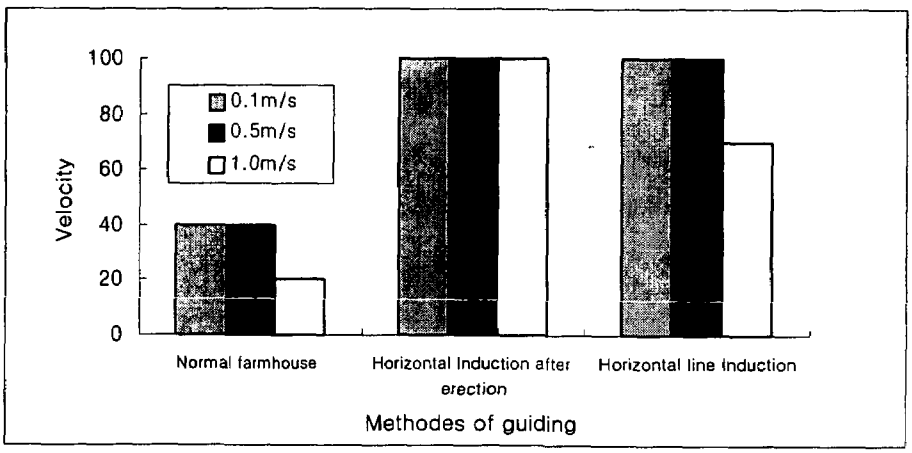
일반농가의 경우 지면이 일정하지 않아서 Test 7에서 0.1m/s의 저속에서 토양과 바퀴에 슬립이 발생하여 20%로 낮게 나타났다. 오이가 재배되는 이랑은 180~200cm이며 두둑의 경우 120~130cm이고 통로는 60~70cm가 일반적인 통로의 형태이나 일반농가에서 재배하는 경우 통로의 폭이 일정하지 않아서 측정값이 33.3%로 낮게 나타났다.

직립 후 횡 유인은 지면의 상태가 일정하여 속도에 관계없이 항상 100%로 가장 좋은 측정값을 나타냈다. 직립 후 가로줄 유인의 경우 일반 농가에 비해

서 통로가 일정하지만 직선으로 성장시킨 뒤 X축으로 5° 이동시켜 재배하기 때문에 채광이 좋지 못하여 1.0m/s의 고속의 경우에 카메라의 인식이 불량하여 70%로 나타났다

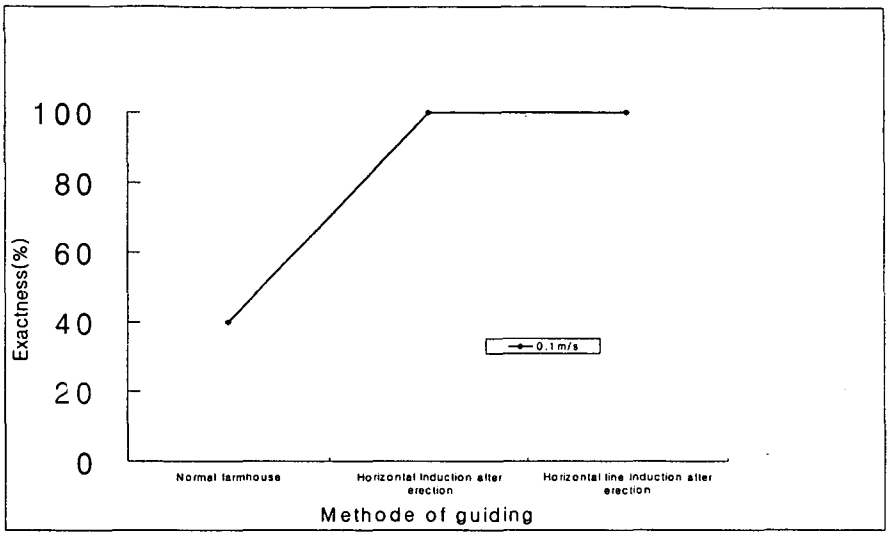
<Table 3-12> Indoors experiment for driving < unit : % >

Induction method / Velocity	Normal farmhouse	Horizontal Induction after erection	Horizontal line Induction after erection
0.1m/s	40	100	100
0.5m/s	40	100	100
1.0m/s	20	100	70



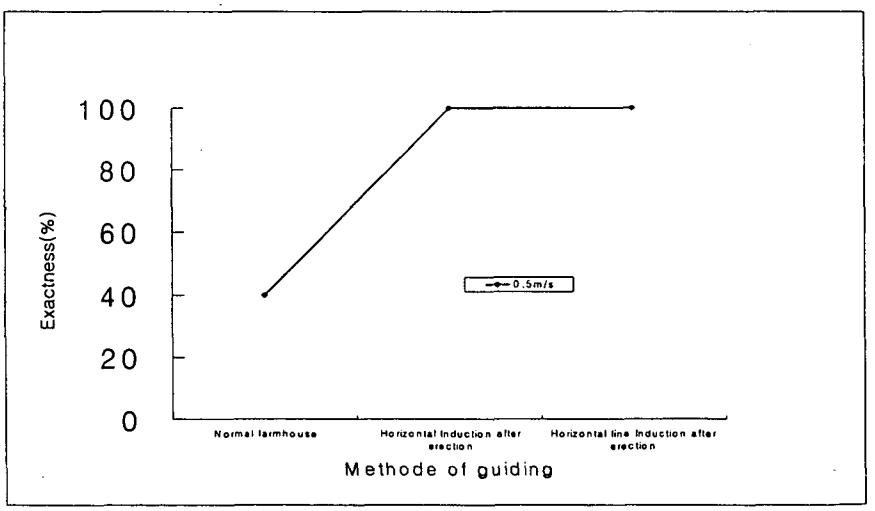
<Fig. 3-21> Exactness of velocities and methods of guiding

그림에서 일반농가의 경우 0.1m/s, 0.5m/s, 1.0m/s의 경우 40%, 40%, 20%로 낮게 나타났으나 직립 후 횡유인의 경우 속도에 관계없이 항상 일정하고 정확도가 높게 나타났다.



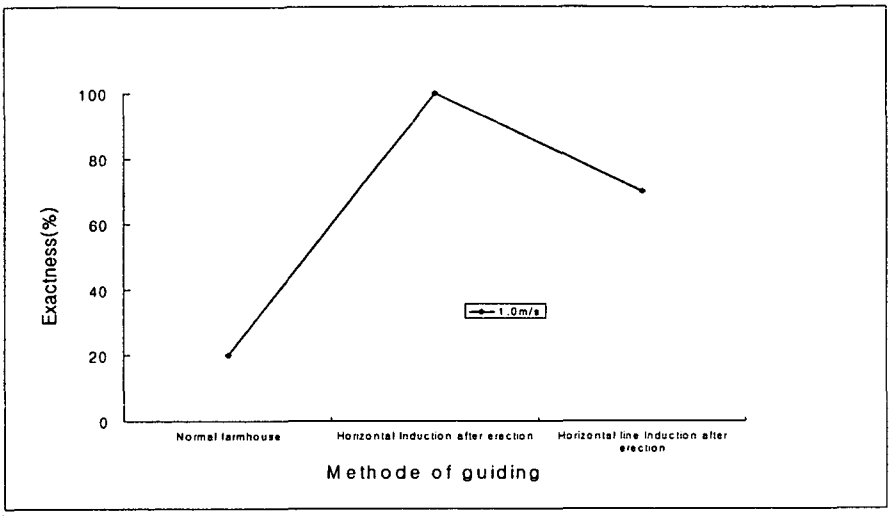
<Fig. 3-22> Methods of guiding and exactness on velocity 0.1m/s

본 실험에서 0.1m/s의 경우에 일반농가의 경우 40%로 재배 환경이 일정하지 못해서 낮게 나타났고, 직립 후 횡 유인과 직립 후 가로줄 유인의 경우에는 100%로 정확도가 높게 나타났다.



<Fig. 3-23> Methods of guiding and exactness on velocity 0.5m/s

0.5m/s 속도의 경우에 일반농가의 경우 40%로 재배 환경이 일정하지 못해서 낮게 나타났고, 직립 후 횡 유인과 직립 후 가로줄 유인의 경우에는 100%로 정확도가 높게 나타났다.



<Fig. 3-24> Methods of guiding and exactness on velocity 1.0m/s

1.0m/s 속도의 경우에 일반농가의 경우 재배 환경이 일정하지 못해서 지면과 바퀴에 대해서 속도의 증가로 슬립이 증가해서 20%로 낮게 나타났고, 직립 후 횡 유인의 경우에는 속도의 증가에 대하여 영향을 받지 않았지만 직립 후 가로줄 유인의 경우에는 속도의 증가로 인해서 카메라의 인식률의 저하로 70%로 나타났다.

4. 요약 및 결론

본 연구에서는 시설재배에서 작업기로 사용하기 위해 대차용 유인장치를 개발하였다. 이 시스템은 노면의 상태에 의한 슬립의 발생과 대차의 작업속도를 결정하여 적응성을 알아보기 위하여 하였다. 작업기의 사용을 위하여 일반

농가, 직립 후 횡 유인 및 직립 후 가로줄 유인에 대하여 실험한 결과 직립 후 횡 유인의 경우 대차용 유인장치가 작업하기에 적당한 재배 방법이라고 사료된다. 그 결과를 요약하면 다음과 같다.

1. 0.1m/s의 경우에 일반농가의 경우 40%로 재배 환경이 일정하지 못해서 낮게 나타났고, 직립 후 횡 유인과 직립 후 가로줄 유인의 경우에는 100%로 정확도가 높게 나타났다.

2. 0.5m/s 속도의 경우에 일반농가의 경우 40%로 재배 환경이 일정하지 못해서 낮게 나타났고, 직립 후 횡 유인과 직립 후 가로줄 유인의 경우에는 100%로 정확도가 높게 나타났다.

3 1.0m/s 속도의 경우에 일반농가의 경우 재배 환경이 일정하지 못해서 지면과 바퀴에 대해서 속도의 증가로 슬립이 증가해서 20%로 낮게 나타났고, 직립 후 횡 유인의 경우에는 속도의 증가에 대하여 영향을 받지 않았지만 직립 후 가로줄 유인의 경우에는 속도의 증가로 인해서 카메라의 인식률의 저하로 70%로 나타났다.

제 4 장 그리퍼, 엔드이펙트 시스템 개발

제 1 절 서 론

농산물 수입개방으로 외국산 농산물의 대량 유입은 우리 농업의 기반을 흔들고 있는 실정이다. 따라서 농산물의 고품질화 및 생산비 절감으로 대외 경쟁력을 높일 수밖에 없는 실정이다. 이를 위해서 농산물의 적기 수확은 재배기술 못지 않게 중요하다.

현재 엔드이펙터를 이용하여 수확기 개발을 시도하고 있으며, 부분적으로 상당한 수준에 도달한 분야도 있다. 엔드이펙터를 이용한 작업은 지루한 작업이 반복적으로 수행되는 어렵고 힘든 작업에 이용되고 있다. 농업 분야에도 단순 자동제어 시스템으로 해결될 수 없거나 복잡하고 지루한 반복작업이 지속되는 분야에서 농업용 엔드이펙터 개발이 필요한 실정이다.

이중에서 로봇의 엔드이펙터를 개발 할 때, 중요하게 고려할 요소는 작업할 대상물의 물리적 특성이다. 작업할 대부분의 농작물은 생명을 가지고 있지만, 산업용 대상물은 생명을 가지고 있지 않다는 점을 인식하여야 한다. 그러므로 일반적으로 공장에서 사용하는 산업용 로봇의 엔드이펙터의 몇 가지 기능만을 변화시켜 농업용 엔드이펙터로 이용할 수 있는 농업용 대상물은 제한되어 있다. 즉, 공업용 대상물은 대부분 정밀 작업이 요구되는 견고한 강체이지만, 농작물은 이동하거나 충격을 가하면 상처를 쉽게 받는 유연한 생명체이다. 그러므로 이러한 점들을 고려하여 엔드이펙터 개발을 하는 것이 필요하다.

대부분의 과채류 수확은 현재 수작업으로 이루어지고 있다. 지상(地上) 1m 내외에서 결과(結果)하는 과채류의 경우 수작업시 많은 노동력을 필요로 한다. 특히 오이의 경우 연속된 단순 반복 작업으로 작업인의 피로도는 더욱 증가될 수밖에 없는 실정으로 오이 재배에 많은 어려움을 낳고 있다. 사람이 오이의 수확 작업을 하는 경우에는 과실 길이가 200 mm정도의 과실을 대상으로 하

며, 한 손으로 과경 바로 아래 부분인 과실 상단 근처를 잡고 과경의 절단 위치를 눈으로 보거나 손으로 감지하여 다른 한 손으로 절단한다. 이러한 작업을 자동적으로 절단할 수 있는 엔드이펙터의 개발이 필요하다. 오이 수확기의 엔드이펙터를 개발하기 위하여 기초 연구로서 시도된 본 연구의 구체적인 연구 목적은 다음과 같다.

가. 엔드이펙터의 파지장치의 개발을 위하여 오이의 물리적 특성을 측정하여 분석한다.

나. 엔드이펙터 기술을 확립하여 과실의 파지와 과경(果柄)을 절단할 수 있는 엔드이펙터를 개발한다.

제 2 절 그리퍼 및 엔드이펙트 시스템의 연구동향

농업용 엔드이펙터의 절단장치는 가위, 절단기, 전정기, 바늘 형태 등이 쓰인다. 농산물을 잡기 위한 파지장치의 개발도 연구가 활발히 진행 중이다. 현재 가장 많이 연구되고 있는 농업용 엔드이펙터는 모종을 이식하기 위한 연구와 농산물 수확을 위한 연구이다.

절단장치의 연구를 살펴보면, Key(1985)는 양털을 깎는데 가위를 사용하였고, Clarke(1985)는 돼지고기를 자르는데 절단장치를 개발하였으며, Sevila (1985)는 포도덩굴을 전정 하는데 있어 전정도구를 개발하였다.

파지장치에 관한 연구를 살펴보면, Hoy(1986)는 감자를 잡기 위하여 압력센서를 가진 엔드이펙터, Trevelyan(1984)은 피부와의 거리를 검출하는 센서가 부착된 엔드이펙터, A. R. Frost(1993)는 젖소의 유방에 착유하기 위한 엔드이펙터를 개발하였다.

Hoy(1986)는 감자 출하를 위해 3개의 오목한 손가락을 가진 그리퍼를 설계했다. 손가락은 안에 공기가 들어있다. 촉각은 압력센서를 이용하여 피드백(Feedback)시키므로 감자와 꼭맞게 만들어진 금속의 지주가 잡는다. 오목한 손가락의 그리퍼는 감자 지름의 0.01inch까지 분해능을 가지고 있다.

Trevelyan(1984)은 양모를 깎는 작업에 필요한 비용 절감을 목적으로 개발하였다. 엔드이펙터의 구동은 유압을 사용하였으며, 연약하고 다치기 쉬운 양을 취급하기 때문에 피부에 접촉하면 바로 피할 수 있는 고속응답성과 높은 신뢰성을 가지고 있다. 엔드이펙터에는 피부와의 거리를 검출하는 센서가 부착되어 있고, 모터로 구동된다. 두당 소요시간은 24분으로 95%이상의 털을 깎을 수 있다.

A. R. Frost(1993)는 인간의 개입 없이 착유를 하기 위하여 착유기를 개발하였는데, 착유기는 고리형태로 연속적으로 배열, 선회 축을 가지고 있으며 엔드이펙터가 X, Y, Z로 움직이도록 구성되어 있다. 엔드이펙터의 착유컵 부착 시간은 실험을 통하여 누적 산출하였는데 약 80초가 소요되었고, 짧게 걸린 시

간은 약20초, 약90%가 소요되었으며, 30초의 경우 50%로 낮게 나타났다.

묘목을 이식하기 위한 엔드이펙터는 Hwang 등(1985), Kutz 등(1987), Ting(1988), 이(1990)등이 수행하였다.

Hwang(1985)등은 후추 묘목과 할라파(Jalapeno)의 줄기를 삽처럼 생긴 손가락으로 2개를 이식시킨다. Kutz(1987)등은 2개의 평행한 집게를 가진 엔드이펙터를 가지고 있다. 손가락은 높이가 30~50mm인 토마토와 금잔화(Marigold)의 묘목을 이식하는 실험을 위하여 설계되었다. Ting(1988)등은 바늘 형태의 엔드이펙터에 근접센서를 사용하여 이식하도록 설계하였다.

이(1990)는 담배 모종을 모판에서 정식하기 위하여 4개의 평판(Spade)을 이용하였다. 이 때 모종의 흙이 엔드이펙터에 부착되는 것을 방지하기 위하여 물통 속으로 통과하는 사이에 평판을 진동하도록 하였다. 류 등은 육모용 이식을 목적으로 엔드이펙터를 개발하였는데, 무게를 줄이기 위하여 알루미늄을 사용하여 제작하였다. 모종을 잡을 때에는 스테핑모터가 잎의 방향에 따라 회전한 후 공압실린더와 공압척을 순차적으로 작동하여 모종을 잡는다.

수확용 엔드이펙터의 개발은 매우 중요하다. 농산물의 과실과 과경의 형태에 알맞은 엔드이펙터의 개발이 필요하다. 많은 연구가 진행되고 있지만 실용적으로 사용되고 있는 경우는 거의 없는 실정이다. D'Esnon (1985)와 Chai 등(1989)은 포도 수확용, Harrell 등(1990)은 오렌지용 엔드이펙터, 近藤 直(1996)은 감귤과 토마토용, D'Esnon (1985)은 생식용 사과용 엔드이펙터를 개발하였다.

D'Esnon(1985)은 생식용 사과 수확을 목적으로 로봇을 개발하였는데, TV 카메라를 이용해서 과실을 검출하고 핸드부는 진공을 사용하여 과실을 흡착한 후 비틀어서 수확은 방법을 사용하였다. 광조건이 좋은 경우 울타리형으로 재배된 수형의 사과나무에서 50% 이상의 과일을 4초에 1개꼴로 수확가능하고 상처가 거의 없으며 75%는 사과 과경이 붙어 있었다. 유연한 손가락 8개가 전후로 180 °회전 할 수 있도록 되어있어 과경의 방향에 관계없이 대부분의 과실을 매니퓰레이터(Manipulator) 내로 들어오게 할 수 있도록 되어있다. 줄기

와 잎이 팔의 이동에 장해가 되거나, 작동영역 외에 과실이 존재하는 경우는 자동적으로 팔을 끌어들인다. 실험 결과 1개를 수확하는데 4초의 시간이 소요되었다.

Chai(1989)등은 포도 수확을 목적으로 계발하였으며, 그리퍼(Gripper)부의 파지력은 수확시 과방의 질량이 약 500g이므로, 수축의 마찰저항을 고려하여 10N으로 한다. 또 수확시 수축의 단면적은 약 30 mm^2이며, 수축의 절단저항을 고려하여 그리퍼(Gripper)부의 절단력은 100N으로 했다. 파지, 절단부의 아래쪽으로 과방을 밀어내는 기능을 부과하고 있다. 파지부와 절단부는 하나의 DC 모터와 용수철로, 밀어내는 부분은 모터의 회전운동을 래크와 피니언에 의해 직선운동으로 바꾸어 구동하고 있다.

Harrell(1990)등은 오렌지수확로봇을 개발하였다. 매니퓰레이터 선단에 카메라가 있고 12개의 엔드이펙터을 가지고 초당 6개의 과실을 수확하도록 고안하였으며, 하루에 4시간의 점검정비시간을 제외하고 20시간의 작업이 가능하였다.

近藤 直(1996)등은 오이 수확작업의 자동화를 목적으로 로봇수확에 적합한 재배양식을 구명하고 그 재배양식에 맞는 매니퓰레이터를 개발하였다. 특징은 로봇 수확에 적합하도록 과실과 경엽 등의 분리가 용이한 경사재배양식에서 수확실험을 하였으며, 매니퓰레이터는 직선운동의 1자유도와 회전 및 관절운동의 5자유도로 구성하였다. 재배된 오이의 붕(棚)의 각도가 크면 수확이 용이한 것으로 나타났다. 수확은 과실을 그리퍼 내의 접촉센서로 검출하면 파지부에서 과실을 약 70N의 힘으로 파지, 유지함과 동시에 센서, 절단부가 과실을 약 3N의 힘으로 잡고, 오이를 따라 상방향으로 미끄러짐으로써 지름을 검지 할 수 있다. 과경에서는 과실 지름이 급속하게 작아지기 때문에 그 위치를 마이크로스위치로 알아내어 과경을 절단하였다. 절단력은 8N~15N이다.

近藤 直(1996)은 하우스 안의 토마토 등의 과채류 수확을 목적으로 개발을 하였으며, 범용성을 갖도록 방제, 솎음 작업등도 매니퓰레이터와 엔드이펙터를 교환하면 가능하도록 제작하였다. 관절형 로봇으로서 그리퍼 끝이 플랜지에 시

각부와 절단기를 가진 엔드이펙터를 부착한 것이다. 엔드이펙터의 경우 반원링형의 절단기를 사용하여 수확하도록 되어 있다. 그리퍼는 완만히 구부러진 그리퍼가 좌우로 부착되어 있고 과실을 미끄러지지 않도록 그리고 가능한 손상을 주지 않도록 손가락의 안쪽에 고무가 부착되어 있다. 소형 DC모터로 그리퍼를 개폐하고, 전류를 제어함으로써 과실을 일정한 힘으로 파지하며, 매니퓰레이터를 구동하여 당기거나 구부려서 수확을 한다.

近藤 直(1996)은 감귤 등과 같은 큰 수관을 가지는 과실을 수확하는 목적으로 개발하였으며, 과실인식 및 위치검출은 카메라를 이용하였고 핸드는 고무재질의 3개의 손가락으로 파지 하도록 하였다.
고무제의 3개의 손가락을 가지고 있지만, 그리퍼의 상부에 있는 가위로 결과지를 절단하도록 되어 있다. 럽액츄레이터(rubactuator)와 손가락 끝을 잇는 철사를 럽액츄레이터의 수축력으로 잡아당기면 손가락이 매끄럽게 휘어져 과실을 파지 한다. 손가락은 위에 2개, 아래에 1개가 있고, 2개의 위에 있는 손가락 사이에 결과지를 넣어 과실을 파지하여 수확하였고 엔드이펙터의 구동원은 유압을 이용하였다.

제 3 절 실험재료 및 방법

1. 실험장치

가. 엔드이펙터의 설계기준

수확작업을 수행하는데는 엔드이펙터의 성능이 매우 중요하므로 다음과 같은 설계기준을 두고 설계하였다. 첫째, 공간상에서 원활하게 움직이며 작업을 하기 위해 엔드이펙터가 경량이어야 하고 둘째, 오이의 파지 및 과경의 절단시 작업 대상에 상해를 주지 않아야 한다. 셋째, 하우스상에서 작업을 하므로 습기에 대한 내부식성이어야 하고 넷째, 고장으로 인한 수리 및 소제(掃除)를 위해 구조가 간단하고, 작업 중 변형이 없어야한다.

나. 엔드이펙터 제작

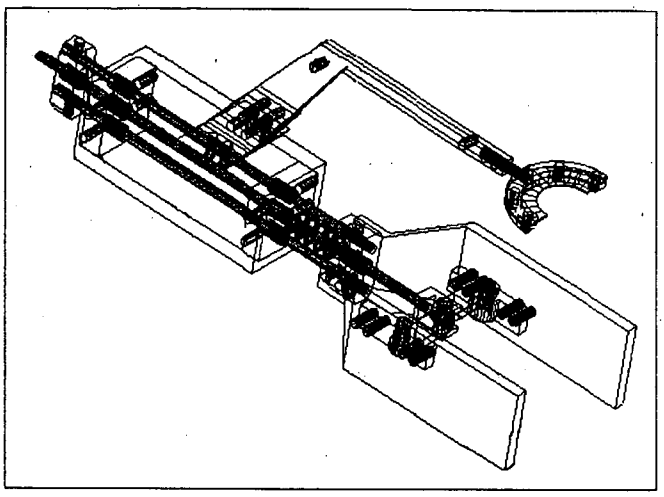
설계는 3차원설계 프로그램인 3dsmax를 이용하여 외날 엔드이펙터, 쌍날 엔드이펙터, 세날 엔드이펙터 및 회전날을 설계하였고, 실험실에 있는 CAM과 CNC를 이용하여 제작하였다.

1) 시스템 I

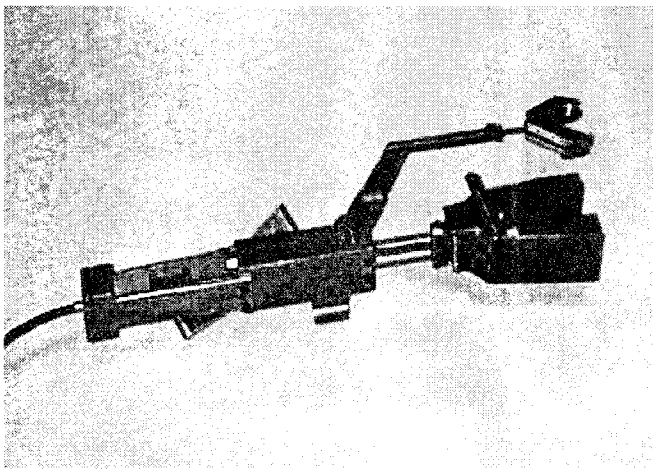
<Fig. 4-1>은 설계 제작된 엔드이펙터를 나타내며, (a)는 설계도이며 (b)는 제작된 엔드이펙터의 사진이다.

엔드이펙트는 3D-max을 이용하여 설계하였으며, 제작은 CNC 조각기을 사용하였다. 엔드이펙터는 그리퍼의 Y축 상부에 부착할 수 있도록 설계되었다. 비틀림을 방지하기 위하여 직선 가이드를 장착하였고, 파지시 저항력을 주기 위하여 5mm의 인장스프링을 사용하였다. 또한, 보완 및 수정을 용이하게 하기

위해서 탈부착이 가능하도록 3mm의 볼트로 고정하였다. 각 기계 요소들의 재질은 무게를 줄이고, 부식되지 않도록 하기 위하여 드랄니늄과 1mm의 스텐레스을 사용하여 제작하였다.



(a)

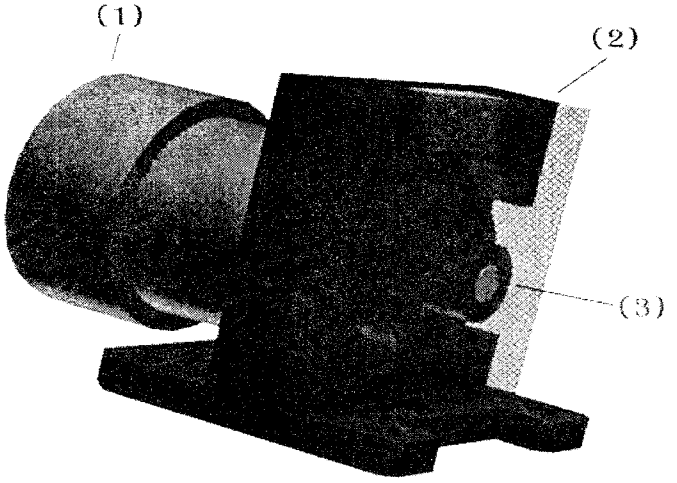


(b)

<Fig. 4-1> Single blade end-effector

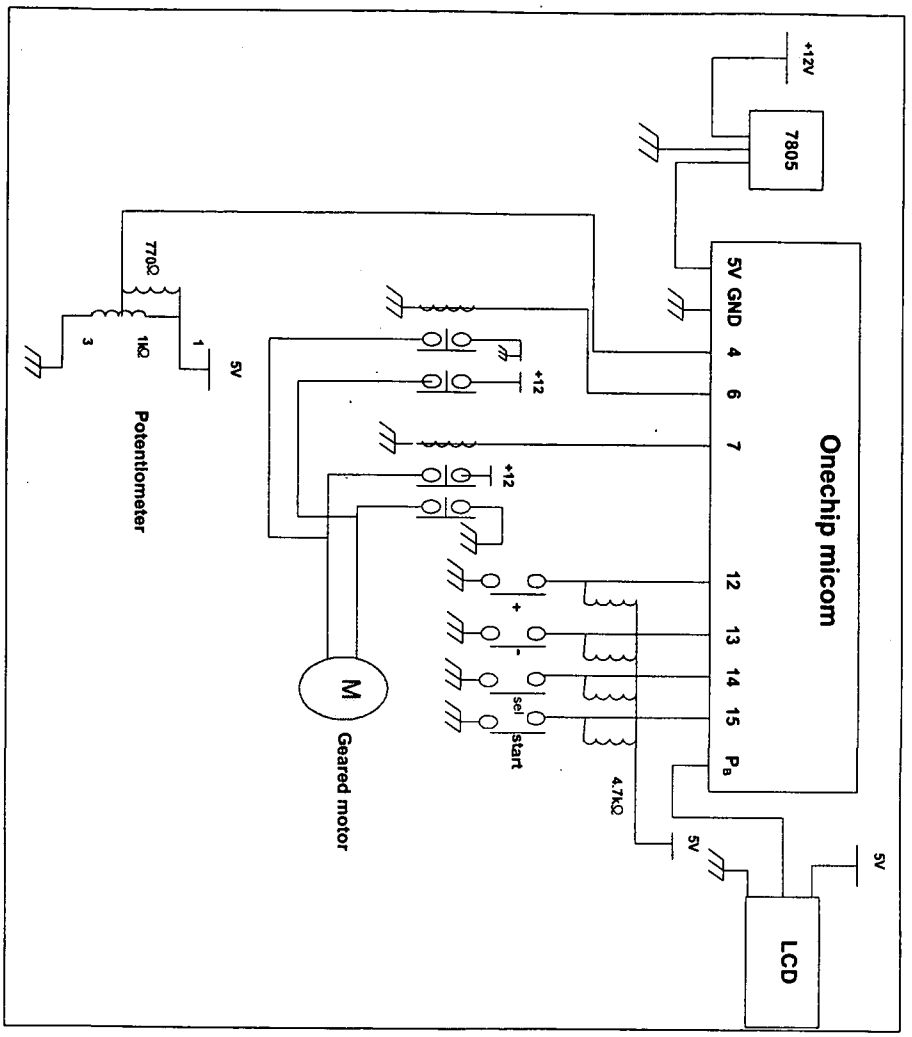
가) 제어장치

<Fig. 4-2>은 감속모터의 입체도를 나타낸 것이고, 감속모터는 문일에서 제작한 감속비가 75 : 1인 감속모터를 사용하였다. 모델명은 KG. D20이고 제어 방식은 12V, 60rpm의 감속모터가 회전하면 모터축에 연결된 풀리가 회전하면서 와이어를 당기게 된다. 이때 풀리에 연결된 1KΩ의 오차가 ±5%인 Sakae사의 모델명 22HP-10인 포텐시오 메타의 변환되는 저항값을 전압으로 변환하고 그 전압을 원칩 마이컴에 내장된 A/D컨버터를 통해 원칩 마이컴으로 입력함으로서 원칩 마이컴이 포텐시오 미터의 회전각을 입력받는다. 이렇게 원칩 마이컴으로 입력된 입력값을 토대로 와이어를 그리퍼 작동에 적합한 길이 만큼 끌어당기고 풀어주게 된다. 여기서 당기는 길이는 원칩 마이컴에 연결된 푸쉬(Push) 버튼을 통해 조절할 수 있게 프로그램 하였다.



(1) Geared motor (2) Box of wire pulley (3) Wire pulley

<Fig. 4-2> Geared motor



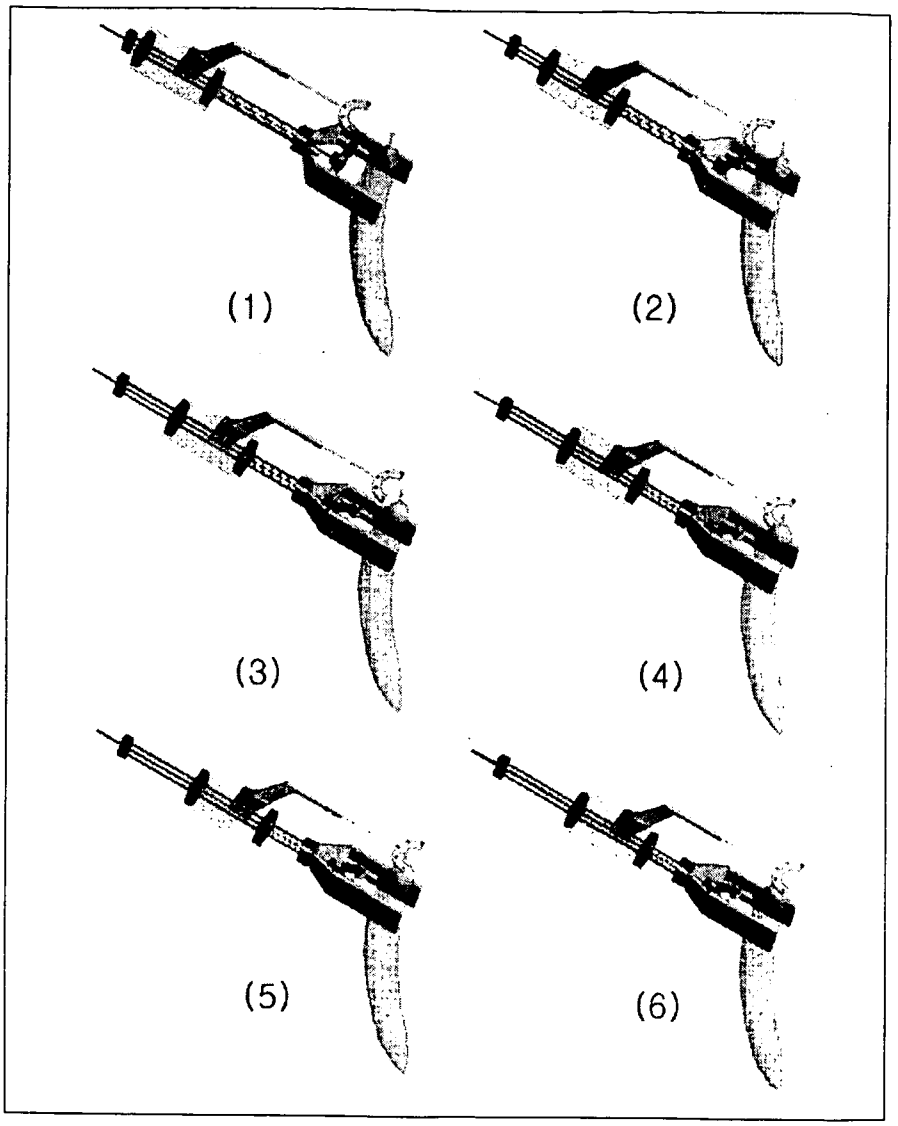
<Fig. 4-3> Geared motor to control a end-effector

나) 작동방법

오이를 파지하고 줄기를 절단하는 역할을 하는 엔드이펙터는 공간상에서

원하는 위치로 신속히 공간상을 이동하기 위해 가벼워야 하므로 주재료로 드랄니늄을 사용하였다. 부품은 인장 스프링과 내경 3mm의 부싱 그리고 오이 과경 절단을 위한 반원형 형태의 칼날, 작동부로의 동력전달을 위한 3mm의 연마봉을 사용하였다. 각 부재의 연결은 수리 및 손질하기 편하도록 3mm의 볼트로 구성되어 있다.

엔드이펙터는 12V, 60rpm의 감속모터(Geared motor)로 조정축(Control axis)을 당기면 이동축(Moving axis)이 안쪽으로 당겨지면서 고정핀(Fixation a pin)으로 결합되어 있는 연결판(Connection tube)을 당기고 파지부(Attaching Part)에 고정되어 있는 고정판(Fixing plate)에 힘이 가해져서 파지부가 오이를 파지한후 조정축에 당기는 힘을 더가하면 파지부가 당겨지면서 절단용 칼날(Cutting blade of knife)에 과경이 닿아서 절단되는 것이다. 절단후 인장력을 제거하면 인장스프링(Extension coil spring)에 의해 원점으로 이동하는 것이다.

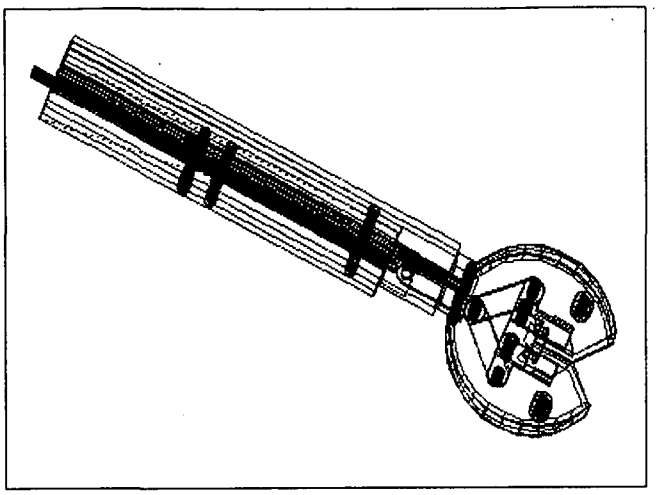


<Fig. 4-4> Operating flow picture of the end-effector I

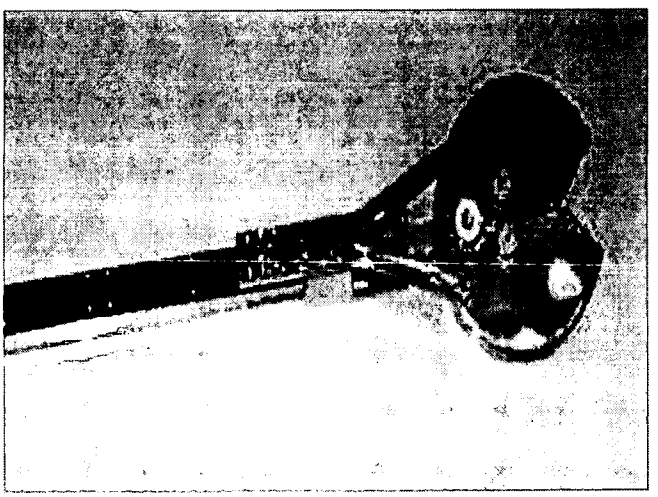
2) 시스템 II

<Fig. 4-5>은 설계 제작된 엔드이펙터를 나타내며, (a)는 설계도이며 (b)는

제작된 엔드이펙터의 사진이다.



(a)

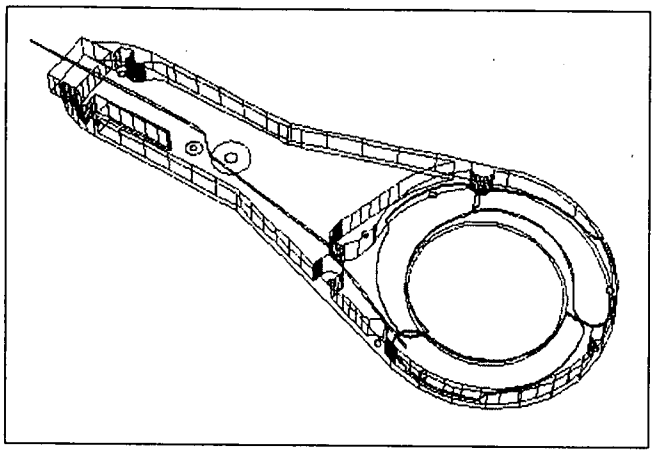


(b)

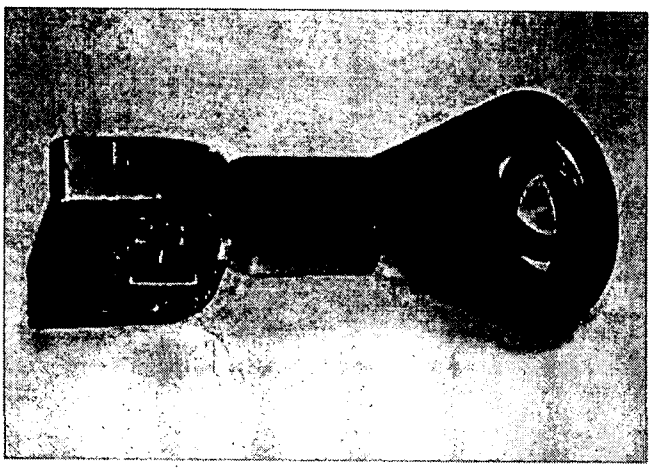
<Fig. 4-5> Double blade end-effector

3) 시스템 Ⅲ

<Fig. 4-6>은 설계 제작된 엔드이펙터를 나타내며, (a)는 설계도이며 (b)는 제작된 엔드이펙터의 사진이다.



(a)

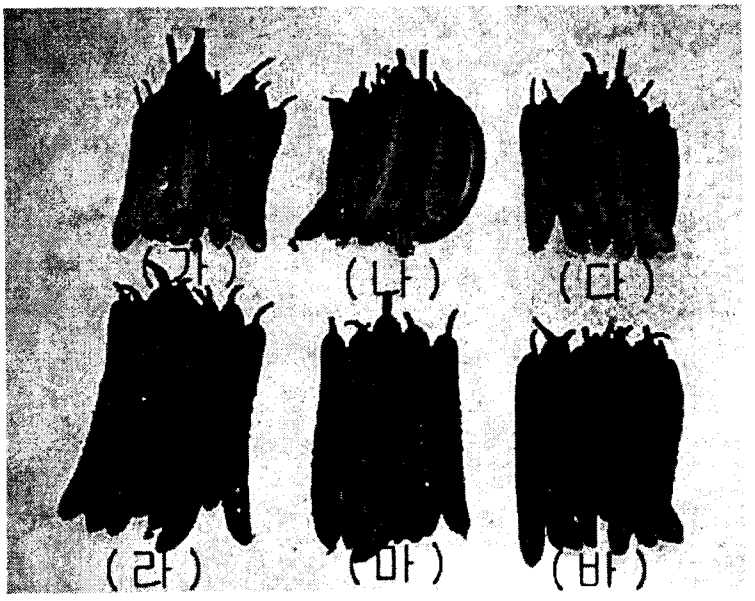


(b)

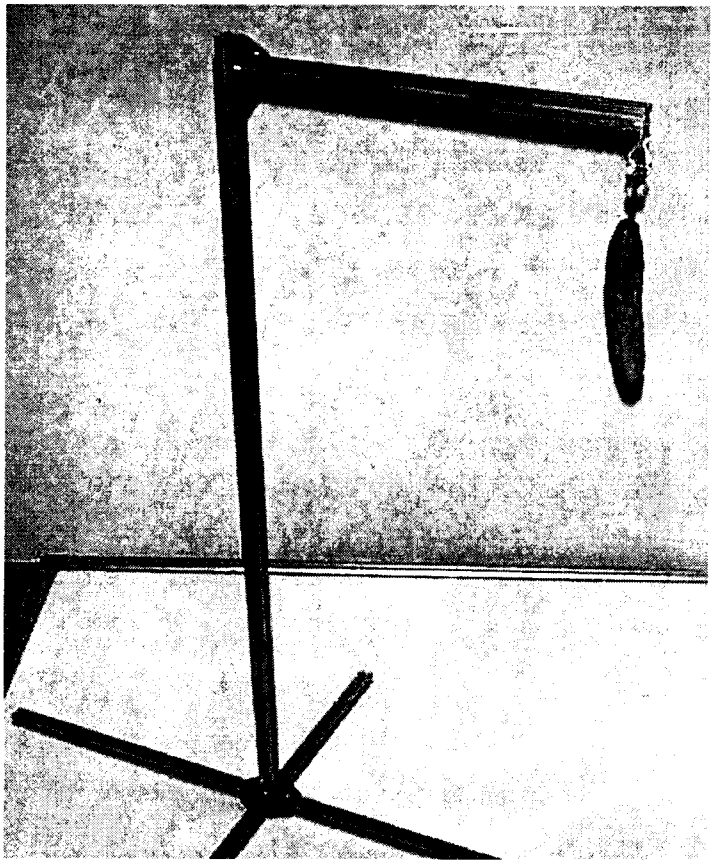
<Fig. 4-6> Triple blade end-effcetor

가) 실험재료 및 장치

과채류 과경의 절단정도를 알기 위하여 오이를 실험재료로 사용하였다. 본 실험에 사용된 오이는 시장에서 판매하는 은성백다다기(가, 나, 다)와 청장계오이(라, 마, 바)의 두 품종을 사용하였다. <Fig. 4-7>는 두 품종의 과경지름을 기준으로 실험에 사용된 오이(가: 5.5mm이상, 나: 4.5~5.5mm, 다: 4.5mm이하, 라:5.5mm이상, 마: 4.5~5.5mm, 바: 4.5mm이하)의 모습이다. <Fig. 4-8>는 성능 평가을 위해 제작된 실험장치이다. 이 실험장치는 가로 세로가 20×20mm인 정사각형 프로파일을 이용하여 장축 1100mm, 단축 420mm, 밑판 장축 800mm, 단축 600mm로 제작하였다. 단축 끝에 클립으로 오이를 매단 후 오이의 절단 실험을 수행하였다.



<Fig. 4-7> Cucumber for experiment



<Fig. 4-8> Equips for experiment

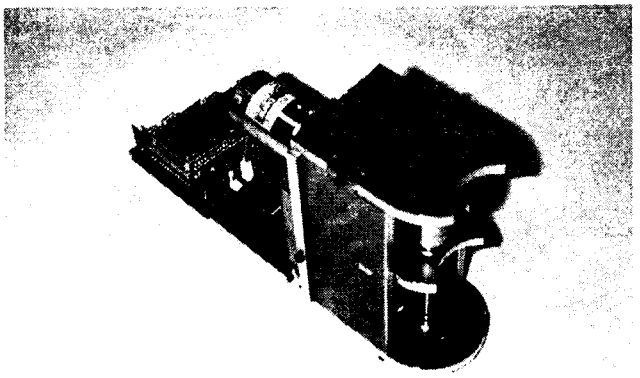
4) 시스템 Ⅳ

엔드이펙터는 3D-MAX를 이용하여 설계하였고, 드랄니늄을 이용하여 제작하였다. 또한 1 : 1의 정밀도로써, 각 구조물을 제작하였다. 간단하고 견고하도록 제작하였으며, 칼날은 고온·다습한 곳에서 부식이 되지 않도록 하였다.

픽 베이직 통한 모터제어 부위에서 동력을 주어 DC 24V, 110rpm의 모터에서 수평동력이 발생하게 된다. 이 동력은 축을 따라 베벨기어에 전달된 후

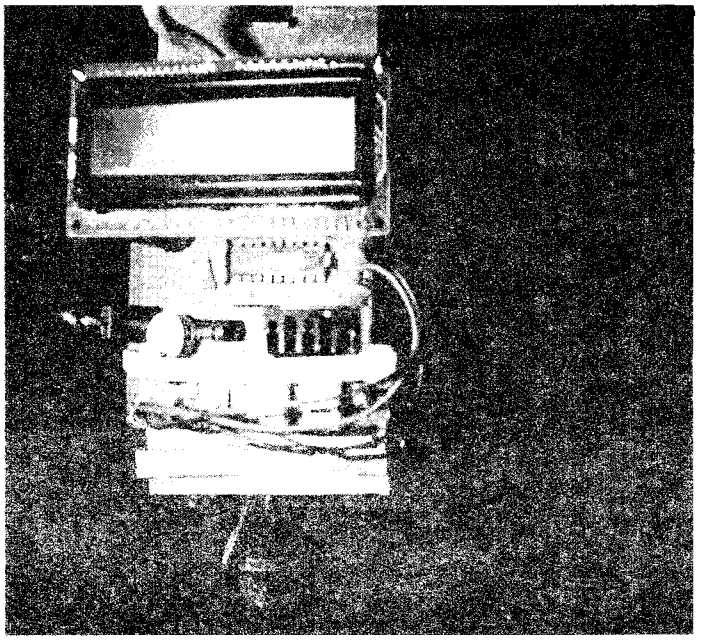
rpm증가를 위해 1 : 3 비율인 베벨기어을 최고 330rpm까지 전달하게 된다. 최종적으로 두 베벨기어를 통해 수평 동력이 수직 동력으로 전달되면 칼날에 의해 오이 과경이 절단되는 것이다.

엔드이펙터에 들어가는 재료는 칼날부위를 중심으로 하여, 지름 4.5cm의 칼날로 절단하였다. 엔드이펙터는 전체적으로, 간단한 구조로써 견고함을 유지하였고, 축과 베어링을 이용하여 전체적으로 회전 할 수 있도록 하였다. 모터는 제어장치를 이용하여 칼날속도를 66~330rpm까지 5단계로 나누었다.



<Fig. 4-9> The end effector

<Fig. 4-9>는 실제 제작된 엔드이펙터의 모습을 보여주고 있다. 엔드이펙터는 견고성을 고려하여 드랄니늄으로 제작하였고, 판의 연결은 3Ø텝을 내어 볼트로 체결하였다. 내부의 동력 전달에 있어서 타임벨트나 V벨트를 이용할 수도 있었으나, 보다 정확한 동력 전달을 위해 1 : 3 비율의 베벨기어를 선택하였다.



<Fig. 4-10> Motor controller

<Fig. 4-10>은 7805 원칩 마이크로 프로세서를 장착한 기판이다. 모터를 구동시키는데 DC 24V 전원을 필요로 하고, 기판 자체 전원으로 DC 5V를 사용하여 rpm을 증가·감속시키는 장치이다.

2. 실험방법

가. 시스템 Ⅰ,Ⅱ,Ⅲ의 실험방법

과경의 지름을 측정한 결과는 은성백다다기와 청장계 품종의 경우 평균지름이 각각 4.54mm와 5.51mm로 나타났다. <Table 4-1>은 2가지 품종에 대해서 3가지 형태의 엔드이펙터를 이용하여 3가지 수준 (과경의 지름이 4.5~5.5mm, 5.5mm이상, 4.5mm이하)에 대해서 실험하였다.

<Table 4-1> Experimental design I

Species / Diameter / Features	Model 1 : 1			Model 2 : 2		
	Over 5.5mm:1	4.5~5.5mm :2	Under 4.5mm :3	Over 5.5mm :1	4.5~5.5mm :2	Under 4.5mm :3
Single blade end-effector :A	1-1-A	1-2-A	1-3-A	2-1-A	2-2-A	2-3-A
Double blade end-effector :B	1-1-B	1-2-B	1-3-B	2-1-B	2-2-B	2-3-B
Triple blade end-effector :C	1-1-C	1-2-C	1-3-C	2-1-C	2-2-C	2-3-C

작업성능은 2종류의 오이에 대해 과경지름을 기준으로 3수준(5.5mm이상, 4.5~5.5mm, 4.5mm이하)으로 구분하여 3가지 형태의 엔드이펙터로 각 실험구마다 10개 총 180개의 오이를 사용하여 절단율을 측정하였다.

나. 시스템 Ⅳ의 실험방법

본 연구의 기계적인 변수로 엔드이펙터의 칼날 회전속도와 각도, 오이의 변수인 과경 직경과 오이무게를 변수로 설정했다. 그리고 이런 상황하에서 1초라는 시간을 기준으로 시간내에 오이줄기를 절단하면 성공 그렇지 않으면 실패로 간주하였다. 여기서 1초는 오이의 절단에 있어 충분한 시간이라고 보고 1초를 선정했다.

<Table 4-2> Experimental design II

Experiment / Variables	Level(rpm)	Condition	Angle	Direction
End-effector	Level 1	78/min	angle 1	Down to 15 °
	Level 2	132/min	angle 2	Right angle
	Level 3	198/min	angle 3	Up to 15 °
	Level 4	264/min	angle 4	Up to 30 °
	Level 5	330/min	angle 5	Up to 15 °
Cucumber	Group 1 -Diameter	~3.5mm	Group 1 -Weight	~150g
	Group 2 -Diameter	3.5mm~	Group 2 -Weight	150g~
Time during cutting	Not over 1 second	Success ○		
	Over 1 second	Failure ×		

먼저 오이를 따기 전에 오이 줄기를 버니어캘리퍼스로 측정하고, 엔드이펙터를 하향 15 °~상향 45 °로 움직이면서 rpm에 따라서 오이의 샘플을 4개씩 선정한 후 오이의 무게를 측정했다.

엔드이펙터의 성능을 보기 위한 방법은 절단에 대하여 상관관계가 있는지를 판단하기 위해서였다.

제 4 절 결과 및 고찰

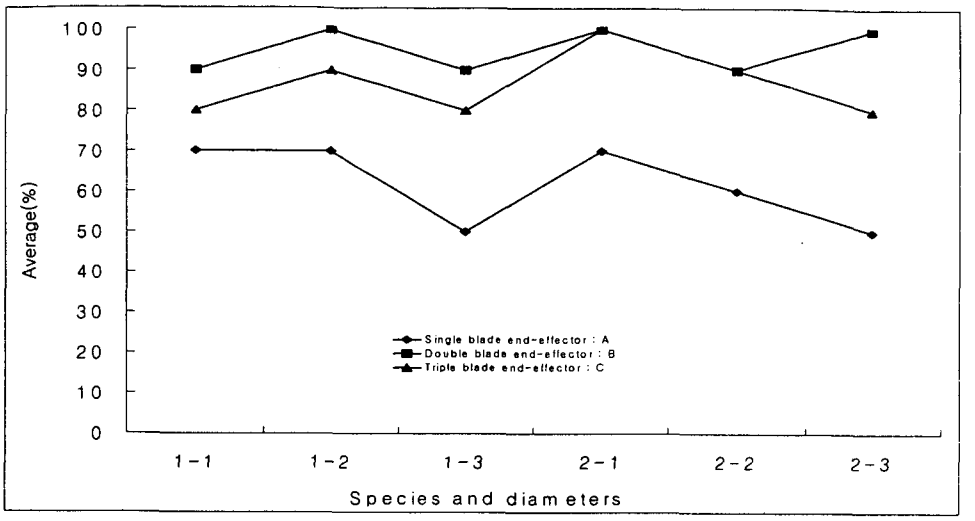
1. 시스템 Ⅰ, Ⅱ ,Ⅲ의 성능분석

오이수확기의 엔드이펙터 개발을 위한 실험에서는 엔드이펙터의 날의 수에 따라 은성백다다기와 청장계의 2품종에 대하여 과경지름의 3수준에 대하여 실험을 하였다. <Table 4-3>와 같이 은성백다다기와 청장계 오이의 평균 절단율은 각각 80.0%, 82.2%로 나타났다.

<Table 4-3> Successive cutting using end-effector <unit : %>

Species / Diameter / Features	Model 1 : 1			Model 2 : 2		
	5.5mm 이상:1	4.5~5.5mm :2	Under 4.5mm :3	Over 5.5mm :1	4.5~5.5mm :2	Under 4.5mm :3
Single blade end-effectro :A	70	70	50	70	60	50
Double blade end-effectro :B	90	100	90	100	90	100
Triple blade end-effectro :C	80	90	80	100	90	80

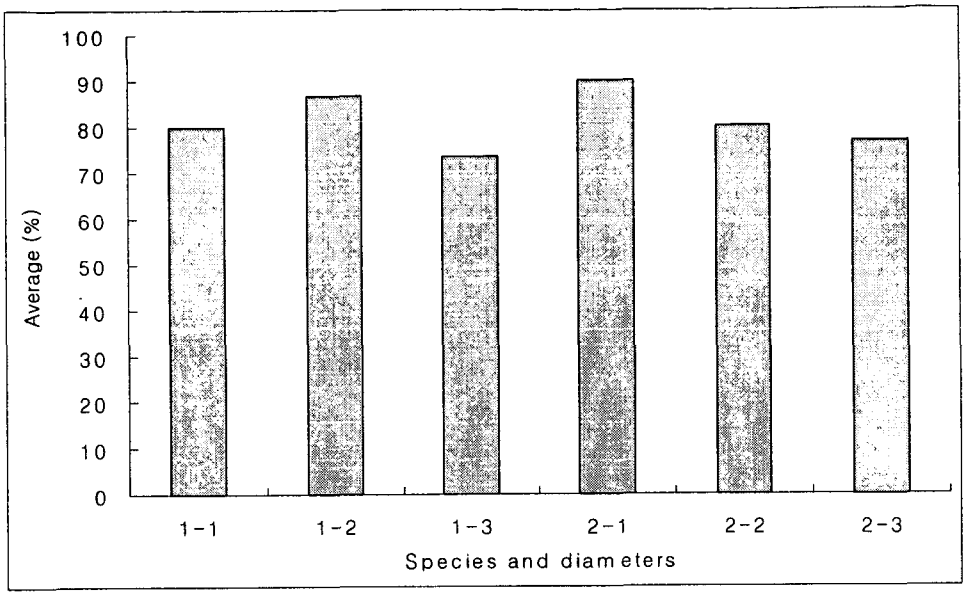
<Fig. 4-11>에서 B와 C는 오이과경의 지름에 관계없이 높은 절단율을 보인 반면 A는 절단율이 낮게 나타났다. 은성백다다기와 청장계 오이의 평균 지름은 33.3mm, 33.2mm로 나타났으며 <Table 4-2>의 실험설계에서 1-1, 1-2, 1-3, 2-1, 2-2, 2-3의 경우 평균 36.6mm, 33.4mm, 29.9mm, 35.6mm, 33.4mm, 30.5mm로 측정되었다.



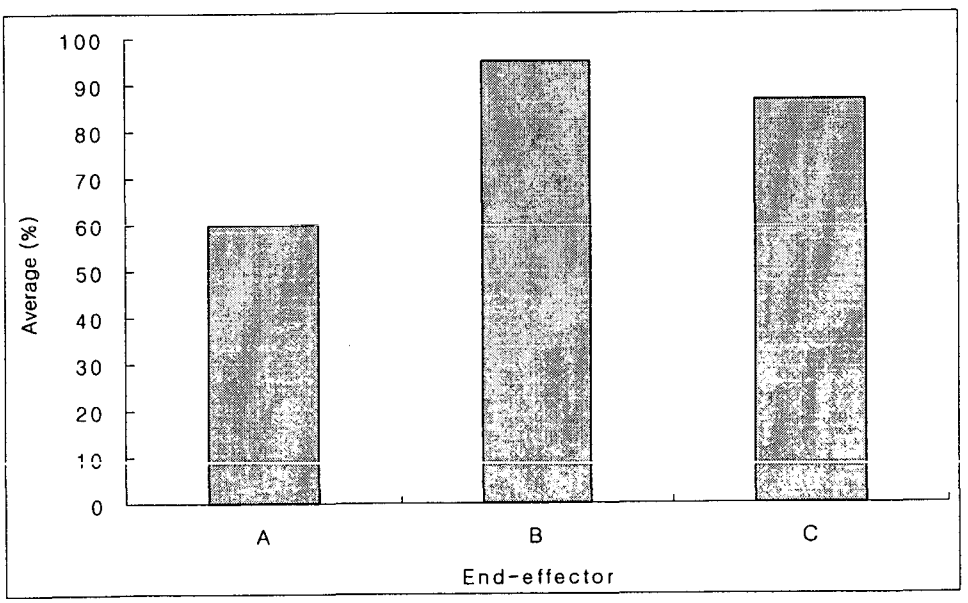
<Fig. 4-11> Cutting properties of diameter

2. 시스템 Ⅰ, Ⅱ ,Ⅲ의 성능 비교

절단율은 <Fig. 4-12>의 1-1, 1-2, 1-3, 2-1, 2-2, 2-3에서 각각 80%, 86.7%, 73.3%, 90%, 80%, 76,7%로 나타났다. 1-3과 2-3의 경우 절단율이 낮게 나타난 이유는 A의 경우 과경의 지름이 29mm이하에서는 절단율이 낮게 측정되었다. <Fig. 4-13>에나타난 바와 같이 외날 엔드이펙터(A), 쌍날 엔드이펙터(B), 세날 엔드이펙터(C)의 절단율은 61.7%, 95%, 86.7%이다. 이는 B의 절단율의 작업성능이 가장 좋다는 것을 의미한다. A의 경우는 절단시 엔드이펙터에 과경이 미끄러져서 절단율이 더욱 낮게 났다.



<Fig. 4-12> Cutting rate



<Fig. 4-13> Cutting rate of end-effector

3. 시스템 Ⅳ의 과경 지름에 따른 분류

오이의 수확작업을 하면서 과경과 오이무게의 상관성이 있는 것을 보았다. 그래서 오이 과경을 두 개의 Group으로 나누어 절단에 있어서 적합성을 찾기로 했다.

<Table 4-4> Petiole group 1 (~3.5mm)

rpm / angle	Level 1	Level 2	Level 3	Level 4	Level 5	Cucumber	Success
Down to 15 °	0/4	0/4	0/4	0/4	0/4	0/20	0%
Right angel	0/4	1/4	2/4	2/4	3/4	8/20	40%
Up to 15 °	0/4	1/4	3/4	3/4	3/4	10/20	50%
Up to 30 °	1/4	2/4	4/4	4/4	3/4	14/20	70%
Up to 45 °	1/4	1/4	3/4	3/4	3/4	11/20	55%
Cucumber	2/20	5/20	12/20	12/20	12/20	43/100	43%
Success	10%	25%	60%	60%	60%		

<Table 4-4>의 결과를 통하여 작업기 특성상, 하향 15 °에서는 수확작업을 할 수 없었다. 그러나 직각상태에서 칼날의 회전속도가 증가할수록 오이줄기 절단율이 높았지만, rpm 3단계이상에서는 rpm에 의해 절단율이 좌우되지는 않았다. <Table 4-5>에서 보는 것과 같이 상향 30 °와 rpm 3단계 이상의 조건을 충족시키면 100%로 절단이 가능했다. 1단계 rpm은 절단하는데 있어 저속의 회전에 의해 많은 시간이 소요되었으며 rpm 5단계에서는 빠른 회전속도에 오이 과경이 축에 걸리기도 하고, 오이 과실의 굽은 정도에 영향을 받아 칼날이 헛도는 경우가 있었다.

<Table 4-5> Petiole group 2 (3.5mm~)

rpm / angle	Level 1	Level 2	Level 3	Level 4	Level 5	Cucumber	Success
Down to 15 °	0/4	0/4	0/4	0/4	0/4	0/20	0%
Right angel	0/4	1/4	3/4	2/4	3/4	9/20	45%
Up to 15 °	0/4	1/4	3/4	3/4	3/4	10/20	50%
Up to 30 °	1/4	2/4	4/4	4/4	4/4	15/20	75%
Up to 45 °	1/4	1/4	4/4	3/4	3/4	12/20	60%
Cucumber	2/20	5/20	14/20	12/20	13/20	47/100	47%
Success	10%	25%	70%	60%	65%		

<Table 4-5>의 결과를 보면 위에서 아래로 내린 30 °의 상태는 칼날의 회전속도에 따라, 오이 줄기의 절단 성공률이 높게 나타났다. 그러나 3단계 이상 올라가거나 각도가 30 °이상 커지면 절단율이 변하였고, 불필요한 동력소모 및 작물의 안정성을 볼 때 회전속도 3단계 200rpm에서 각도가 30 °일 경우에 가장 적절한 상황으로 판단되었다. 결국 실험에서는 각도 30 °에 회전속도 3단계가 오이 수확작업에 가장 적합하였다.

4. 엔드이펙터를 이용한 수확의 적합성

백다다기 오이는 과경 직경이 3.0mm~4.1mm 였다. 또 상품가치를 갖는 무게는 140g~170g이였다. 엔드이펙터는 위로 30 ° 각도로 칼날의 회전속도가 200rpm에서 적정수준이었다.

간단한 구조와 견고하게 제작하였으므로 온실의 습기와 온도에 영향을 받지 않았으며, 오이 과경의 절단에 있어서도 날의 예리함과 회전속도에 의해 시간이 지연될 뿐 모든 대상 오이의 과경이 절단되었다.

제 5 절 요약 및 결론

엔드이펙터는 오이 수확로봇을 개발하는데 있어서 가장 중요한 요소 중에 하나이다. 엔드이펙터의 개발은 3가지 형태를 설계 제작하였고, 성능을 평가하기 위해 실험실내에서 절단율을 측정하여 결과를 비교 분석하였다. 은성백다다기와 청장계의 2가지 오이품종에 대해 날의 수를 1~3개의 엔드이펙터를 개발하여 비교 실험한 결과는 다음과 같다.

1. 엔드이펙터의 절단율은 날의 수를 1, 2, 3개로 했을 경우 61.7%, 95%, 86.7%로 나타났다.
2. 엔드이펙터의 날의 수가 2, 3개의 경우에는 오이과경의 지름과는 무관하게 나타났으나, 날의 수가 1개인 경우에는 파지부분의 지름이 29mm이하에서 61.7%로 낮게 나타났다.
3. 엔드이펙터 날의 수가 3개인 경우에는 오이를 수확하는데 시간적인 면에서 적합하지 않지만, 포도, 사과, 토마토 등의 형태를 가지고 있는 농산물 수확에 적합할 것으로 판단된다.
4. 동력전달은 베벨기어를 사용하였으며 이때, 모터에서 칼날까지 동력전달이 잘 되었다. 오이의 절단작업은 오이 과경의 법선 방향 30 °각도에서 절단율이 높게 나타났다.
5. 모터의 회전 속도가 200rpm에서 가장 효율적으로 절단되었다.

제 5 장 매니퓰레이터 시스템 개발

제 1 절 서 론

시설원예에서 소요되는 노동력 중에서 가장 많은 부분을 차지하는 부분이 바로 수확작업이다. 시설원예 작물중 오이의 경우 수확작업에 투입되는 노동력이 과다하다. 또한 시설원예로 수확되는 오이의 수확량이 농림부 통계로 94년에는 23만 톤 95년에는 26만톤 96년에는 299,401톤으로 매년 약13%의 수확량 증가를 보이고 있는 반면 수확작업은 아직까지도 원시적인 수준을 벗어나지 못하고 있다. 또한 시설원예농가수가 95년 통계에 의하면 71,411가구에 달하고 1ha 이상 경지면적을 가진 농가 수만 33,427가구에 달한다. 따라서 이 같은 과다 노동력 투입을 해소하기 위해서는 오이수확작업의 자동화가 절실하다고 할 수 있다. 오이 수확작업의 자동화를 이루기 위해서 필요한 기술적 방법은 다른 작물에 비에 어렵다. 왜냐하면 첫째 오이는 한번 심은 줄기로부터 계속해서 오이가 열리고 따라서 벼와 같이 한번에 수확하는 것이 아니고 고추처럼 연속적으로 관찰하여 익은 것만을 수확해야한다는 점이고, 둘째 넝쿨식물로써 줄기는 가늘고 잎은 크다는 특징이 있기 때문이다. 즉 오이 수확 작업 시 오이 줄기의 손상은 바로 오이수확량의 감소로 이어지고, 잎이 크기 때문에 수확하려는 오이를 쉽게 구별해 내기가 어렵다. 이렇게 어려운 수확작업을 자동화하기 위하여 필요한 기술은 크게 4가지로 볼 수 있는데 첫 번째는 오이를 인식하는 영상처리 프로그램의 개발이고 둘째는 위치가 파악된 오이를 파지(把持)하고 과경(果柄)을 절단하는 엔드이펙터의 개발이고 세 번째는 엔드이펙터를 오이에 근접하도록 이동시키고 수확된 오이를 이동시키는 매니퓰레이터의 개발이다. 마지막 네 번째는 수확된 오이를 모아 고랑사이로 이동시키는 자동대차이다. 본 연구는 앞에서 언급한 기술적 요소 가운데에서 오이 수확작업에 적합한 매니퓰레이터의 개발을 목적으로 하고 있다. 본 목적을 이루기 위한 구체적인

연구목적은 다음과 같다.

가. 골 사이로 이동하는 대차에 장착할 수 있기 위해 소형 경량인 매니퓰레이터를 개발한다.

나. 빠른 수확작업이 가능하도록 고속 작동이 가능하도록 고려된 매니퓰레이터를 설계한다.

다. 매니퓰레이터에 장착된 모터를 컨트롤하기 위한 소형 모터컨트롤러를 개발한다.

제 2 절 매니퓰레이터 시스템의 연구동향

수확용 매니퓰레이터의 개발은 Okayama대학(1995), Shimane대학 (1995), 전남대(1995) 등에서 연구를 하였다.

일본의 오까야마 대학에서 연구하여 오이수확기를 개발하였는데 여 기에 쓰인 매니퓰레이터는 1개의직선운동(prismatic) 기구와 5개의 회전 기구로 구성되어 있다. 이 매니퓰레이터의 링크 길이와 회전축의 위치는 오이 줄기에 따라 결정되었다. 이 매니퓰레이터는 5개의 자유도를 가진 매니퓰레이터로 크기를 소형화 하면서 작업반경 및 스트 로크 (stroke)를 증대시켰다.

시마네대학에서는 체리 토마토 수확기를 개발하였다. 여기에 사용된 매니퓰레이터는 토양 부분에 깔린 토마토을 수확하기 위해 X-Y table 형식의 매니퓰레이터를 개발 이용하였다. 고랑을 따라 직선 운동 시에는 대차의 바퀴추진력을 사용하고 좌우 이동 및 상하 이동은 직선 운동기구 를 이용 엔드이펙터를 원하는 위치로 이동 가능하도록 설계하였다.

오까야마대학에서 개발한 양배추 수확 로봇에서 사용된 메니플레이 터는 Melfa RV-P2S 모델을 사용하였다. 이 로봇은 DC 서보모터를 및 드라이버를 사용하였고 5자유도를 가지며 팔부분이 85mm stroke를 가지 며 10N absorption pad의 성능을 가졌다. 또한, 토마토 수확기를 개발 하였는데 여기에 사용된 매니퓰레이터는 7개의 자유도를 가진 로봇이다. 자유도가 많아서 제어가 복잡하긴하나 미묘하고 정교한 위치제어가 가능 한 장점을 가지고 있다. 프리스메틱 조인트는 주로 로테이션 조인트의 위치 결정 및 위치설정을 위해 사용하였다.

전남대에서는 육묘이식용 로봇을 개발하였는데 여기에 쓰인 메니플 레이터는 직교 좌표 3축 매니퓰레이터를 사용하였다. 직선 운동기구를 이 용 X-Y table 형식의 매니퓰레이터를 사용 위치결정 정밀도를 높이고 제어가 쉽도록 하였다.

제 3 절 실험재료 및 방법

1. 실험재료

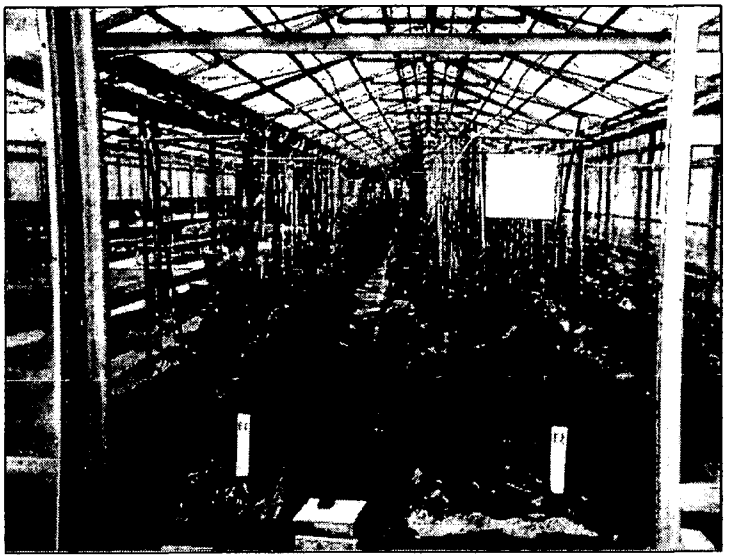
매니퓰레이터 설계에 있어 고려해야할 가장 중요한 것이 작업성능이다. 작업성능을 높이기 위해서는 매니퓰레이터의 각 관절에서 신속한 위치결정이 이루어져야 한다. 신속한 위치결정이 이루어지게 하기 위한 방법은 크게 모터의 출력을 높이는 방법과 매니퓰레이터의 각 링크구조물을 가볍게 만들어야 하는 것이다.

첫 번째 방법인 모터출력증가는 본 매니퓰레이터가 이송장치에 장착되어 축전지 전원으로 구동됨을 감안할 때 그 한계가 있다. 따라서 매니퓰레이터의 각 링크구조물을 가벼우면서도 견고하게 설계하고 회전 모멘트가 적게 걸리도록 모든 모터를 부착시키고 동력은 타이밍 벨트로 전달하는 방식을 취하였다. 또한 구조물은 가볍고 강성이 높은 재료로 제작되어야 하기 때문에 드랄리늄과 ABS수지를 사용하였다. 매니퓰레이터의 구동 원으로 DC 모터를 사용하였는데 이 모터의 사양은 정격 입력전압 DC 12V 최대 입력전류 3A로 감속비(53.9:1)의 감속기어를 장착하고 있다. 출력축의 출력 토오크는 60Kg/Cm 이고 백래쉬는 0.27° 이다.

모터 컨트롤러에 사용된 원칩컨트롤러로는 SCENIX사의 SX28AC를 사용하였다. 이칩은 50MHz의 고속클럭과 1명령이 1 펄스에 의해서 구동되는 초고속 원칩컨트롤러이다. 그리고 모터의 전류제어용 파워 TR로 SSR(SOLID STATE RELAY) 모델명 DN0502P를 사용하였다. 이 모델은 포토 인터럽터로 입력 축과 출력 축을 절연시켜서 출력 축의 과부하로부터 마이컴 회로를 보호해 준다. 모터축의 회전 위치와 속도를 측정하기 위하여 각 모터 회전자 축에 엔코더를 연결하였다. 이 엔코더의 사양은 360(PULSE/REV) ,Totem Pole 출력을 한다. 출력 상은 A상 B상과 원점출력 Z상이다.

<Fig. 5-1>은 오이가 시설 재배되고 있는 환경을 나타낸 사진이다. 골사

이의 간격이 60cm내외로 매우 협소하다. 이러한 작업공간에서 작업을 원활하게 수행하기 위해서 작고 유연한 재료를 이용한 매니퓰레이터가 설계되어야만 한다.



<Fig. 5-1> A cucumber attached to the stem around the leaves

2. 실험장치

가. 매니퓰레이터의 설계기준

수확작업을 수행하는데 있어 매니퓰레이터의 역할은 연약하고 상해를 입기 쉬운 오이에 엔드이펙터를 근접시키는 것이다. 이러한 궁극적인 목표 외에도 고속 작업이 가능하기 위하여 경량이어야 하고 단순 반복적인 작업에서도 대응 할 수 있도록 내구성도 있어야 한다. 특히 매니퓰레이터의 작업환경이 습도가 높은 온실 내부임을 가만할 때 내부식성이어야 한다.

온실 내부에 작물이 심어진 형태는 앞서 <Fig. 5-1>에서 보여진 바와 같이

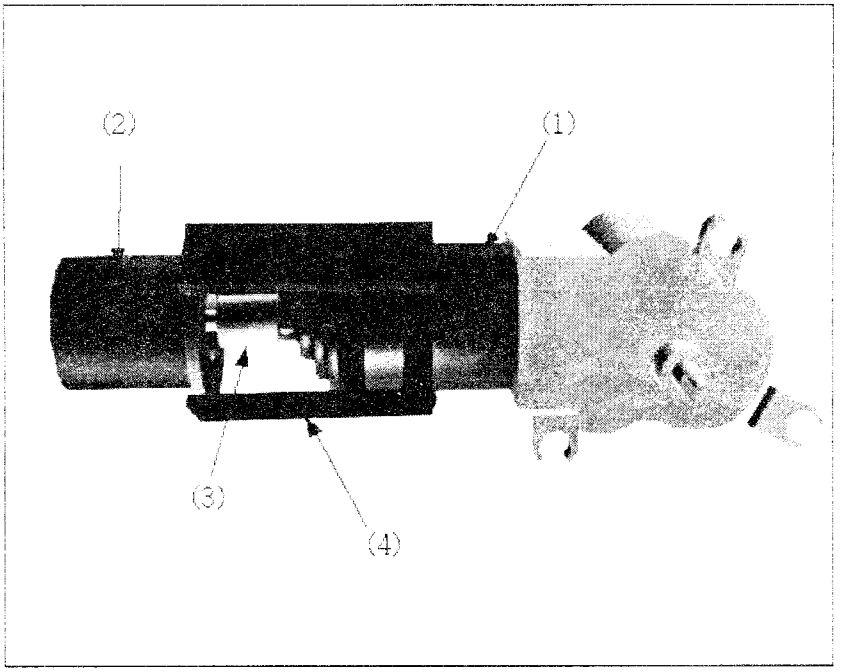
골을 따라 재배된다. 그리고 그 골은 간격은 작물의 수확량을 높이기 위해 60cm내외로 매우 협소하다. 이러한 작업환경에서 자유 자제로 작업이 가능하도록 매니퓰레이터를 제작하는 것도 설계기준에 포함된다. 앞에서 언급한 3가지의 설계 기준을 기반으로 본 매니퓰레이터를 설계 제작하였다.

나. 매니퓰레이터의 제작

3차원 설계 컴퓨터 프로그램을 이용하여 본 매니퓰레이터를 설계하였다. 각종 프레임 및 부품 제작은 CNC 조각기을 이용 1/100mm 의 정밀도로 제작하여 사용하였다.

모터부분은 <Fig. 5-2>에서 보는바와 같이 DC모터(1)의 회전자축에 엔코더(2)를 ABS 수지로 제작한 고정대(4)로 고정시킨 모습이다. 엔코더와 모터 회전축과는 프렉시블커플링을 이용하여 연결하였다.

감속DC모터의 감속율이 53.9:1 이고 엔코더의 사양이 360 (Pulse /Rev) 인 것을 감안할 때 감속이 이루어진 후 출력 축이 1바퀴 회전하면 엔코더에서 나오는 펄스의 양은 53.9 X 360 = 19404 개의 펄스가 출력된다. 따라서 Elbow link의 길이가 294mm 인 점을 감안할 때 한 펄스간격으로 움직이는 엘보우링크(Elbow link)의 이동거리는 294 X PI(3.141592) /19404 =0.0476 (mm)이다. 이는 매니퓰레이터의 엘보우링크 디비젼(Div ision)이 되겠다.

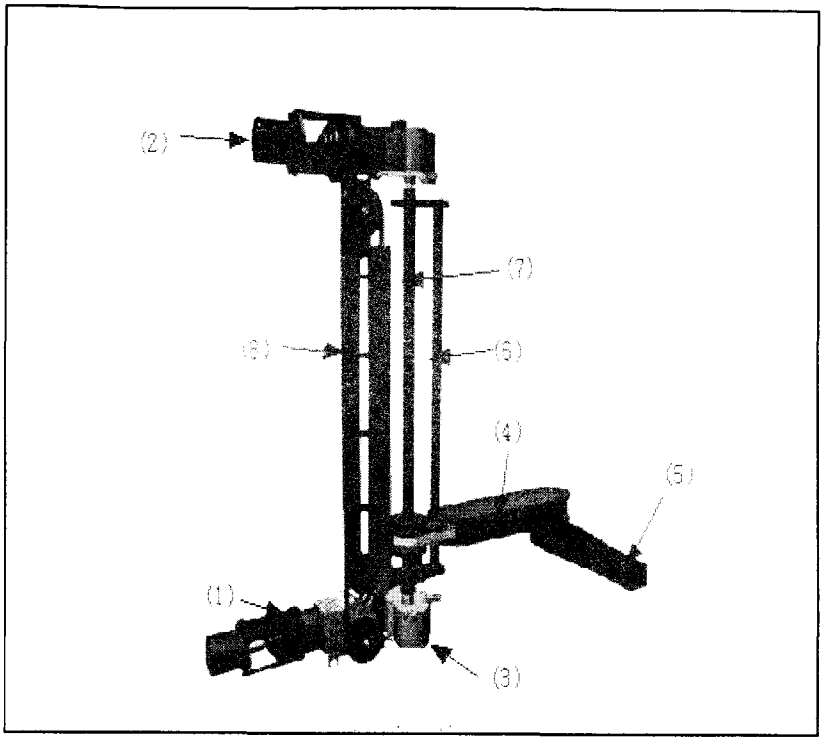


<Fig. 5-2> Motor connected by Encoder

하지만 감속기의 백래쉬가 0.27° 인 것을 감안한다면 0.683(mm)이다. 294 X PI X (0.27 / 365) = 0.683 (mm)이다. 여기서, 294는 엘보우링크의 길이(mm),PI는 기어 백레쉬, (0.27 / 365)는 엘보우 링크오차를 나타낸다. 따라서 엔코더의 디비전과 기어에 의한 오차와의 비는 14.35 : 1 정도로 설계하였다.

매니퓰레이터는 쇼울더 링크(shoulder link)와 엘보우 링크(elbow link) 그리고 이 두 링크를 상하로 이동시켜주는 슬라이딩 베어링으로 구성된다. 슬라이딩 베어링은 타이밍벨트 및 벨트풀리로 이송된다. 이때 이송되는 방향은 수직방향이다.

<Fig. 5-3>은 설계된 모습을 보여주는 그림이다.

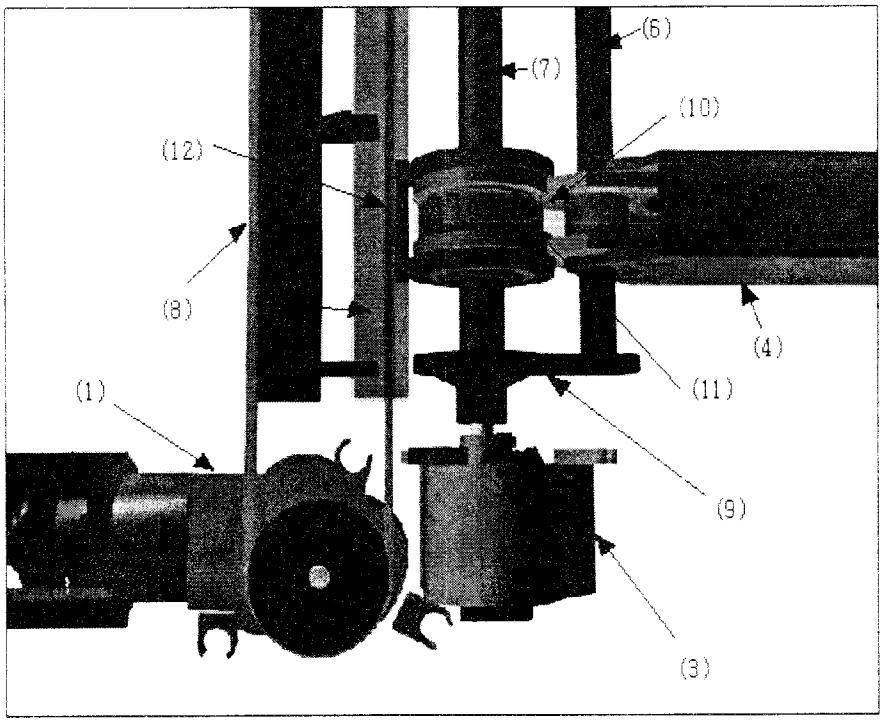


(1) Shoulder link and elbow link elevating motor (2) Elbow link spin DC motor (3) Shoulder link spin DC motor (4) Shoulder link (5) Elbow link (6) Ehoulder link spindle (7) Linear guide for elbow link (8) Timing belt for shoulder link and elbow link

<Fig. 5-3> Perspective view of manipulator

엘보우 링크를 회전시키기 위한 모터(2)의 구동 축이 회전하면 그 구동 축에 연결된 원통형의 직선 가이드(7)가 회전하고 직선가이드를 따라 왕복하는 베어링에 장착된 풀리가 회전하게 된다. 그러면 풀리에 연결된 타이밍벨트가 동력을 전달 엘보우링크가 회전하게 되는 것이다. 이를 좀더 자세히 설명하면 <Fig. 5-4>에서 모터(3) 이 회전하면 직선 슬라이드 축(7) 이 회전하고 그 축

에 끼워진 슬라이딩베어링에 키로 고정된 타이밍 벨트 풀리 (11)가 회전하여 타이밍벨트(10)를 당겨주게 되는 것이다. 그러면 타이밍 벨트(10) 에 연결된 풀리 즉, 엘보우 링크의 회전축에 고정된 풀리를 회전시키고 따라서 엘보우 링크가 회전하게 된다.

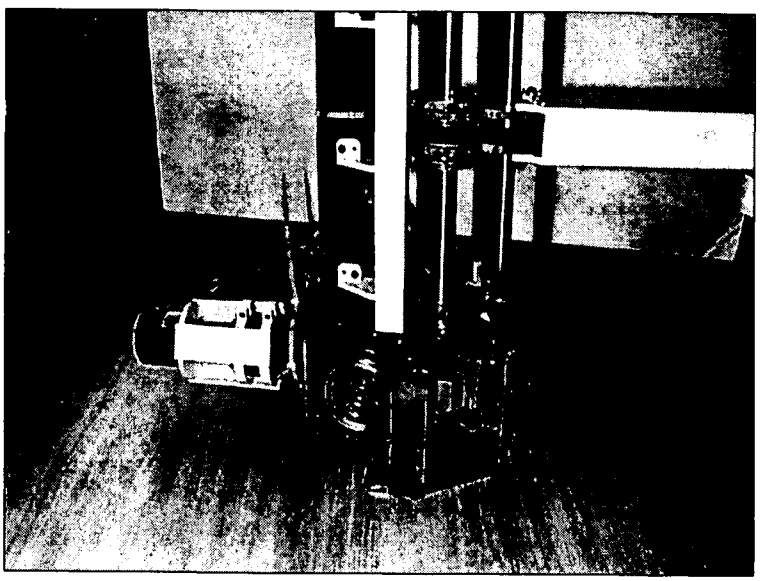


(1) Shoulder link and elbow link elevating motor (3) Shoulder link spin DC motor (4) Shoulder link (6) Shoulder link spindle (7) Linear guide for elbow link (8) Timing belt for shoulder link and elbow link (9) Spin frame for shoulder link which fixed on motor 3 (10) Timing belt for elbow link (11) Timing belt pooly for linear bearing (12) Vertical power system for timing belt 8

<Fig. 5-4> Perspective view of motor 1 and motor 3

다음으로 shoulder link 의 회전 원리에 대하여 설명하면 모터(3)의 출력 축이 회전하면 출력 축에 연결된 프레임(9) 가 회전하고 또한 축(6) 이 슬라이딩 가이드(7)를 회전중심으로 회전하게된다. 그러면 축(6)이 끼워져 있는 Shoulder link(4)도 걸려서 회전하게 되는 것이다.

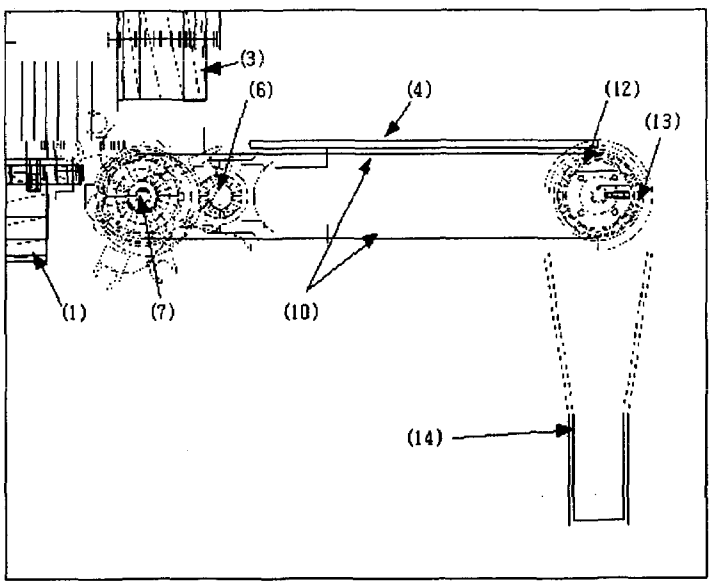
이렇게 함으로써 무게가 무거운 모터 및 기어부분을 작동부 즉 쇼울더 및 엘보우 링크에 장착하지 않아 매니퓰레이터 동작시 발생하는 관성 모멘트를 줄여 고속작동이 가능하도록 하였다.



<Fig. 5-5> Picture of motor 1 and motor 3

<Fig. 5-5>는 <Fig. 5-4>의 원리로 실제 제작한 매니퓰레이터의 모습을 확대한 사진이다. <Fig. 5-4>의 설계사양대로 실 척으로 제작하였으며 원활한 작동을 위해 각 축이 연결된 부분에는 볼 베어링을 삽입하였다.

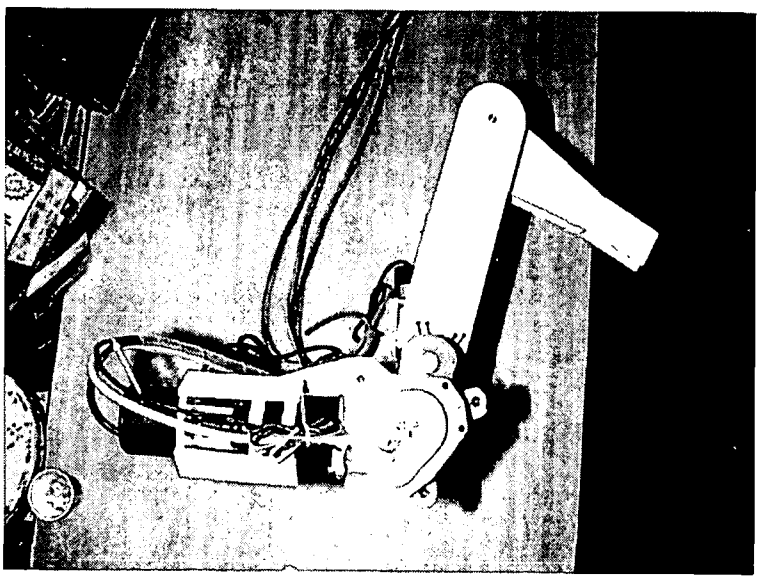
타이밍 벨트가 느슨하면 동작시 부드러움이 감소하고 오차가 커진다. 이를 방지하기 위하여 타이밍벨트가 걸쳐진 두 개의 풀리 중 한쪽에 나사를 이용하여 벨트가 팽팽해지는 방향으로 당기도록 제작하였다. 이렇게 함으로써 정교한 구동이 되도록 하였다. <Fig. 5-6>에서 타이밍벨트(10) 의 한쪽 풀리(11) 이고 다른 한쪽은 엘보우 링크의 회전축에 고정된 풀리인데 이는 엘보우 링크의 회전을 하기 위한 동력전달용이다. 즉 풀리(11)가 회전하면 타이밍벨트(10)가 따라서 회전하게되고 <Fig. 5-6>에서 엘보우 링크(14)에 풀리고정용 키(13)에 의해 고정된 풀리(12)를 회전시키고 따라서 엘보우 링크도 회전하게 된다. 즉 모터2의 회전력이 엘보우 링크의 회전력으로 전달되는 것이다.



(1) Shoulder link and elbow link elevating motor (3) Shoulder link spin DC motor (4) Shoulder link (6) Shoulder link spindle (7) Linear guide for elbow link (10) Timing belt for elbow link (12) Pully : fixed on elbow link and pin with timing belt 10 (13) Fixing key to spin elbow link 14

<Fig. 5-6> Top view of manipulator

이것 또한 작동부 즉 엘보우 링크 및 쇼울터 링크 에 모터와 같은 질량이 많이 나가는 부품을 장착시키기 않기 위하여 설계한 것이다. 이렇게 함으로써 보다 부드럽고 신속하게 매니퓰레이터가 작동할 수 있게 하였다.

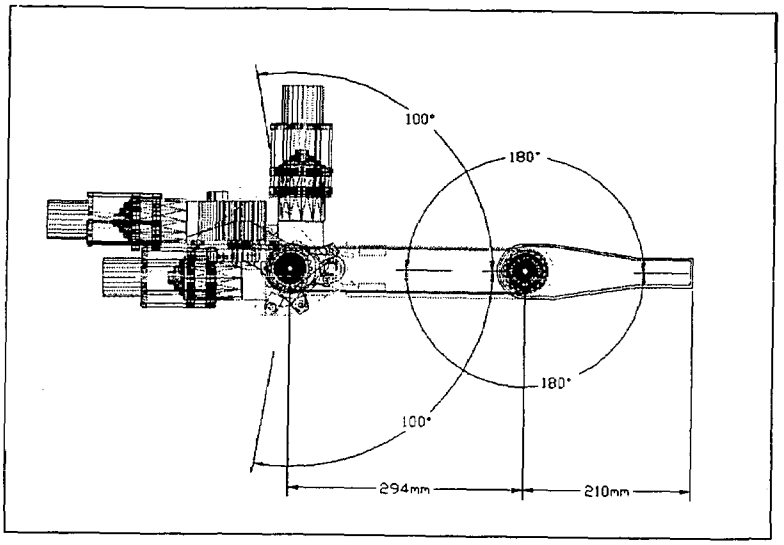


<Fig. 5-7> Top picture of manipulator

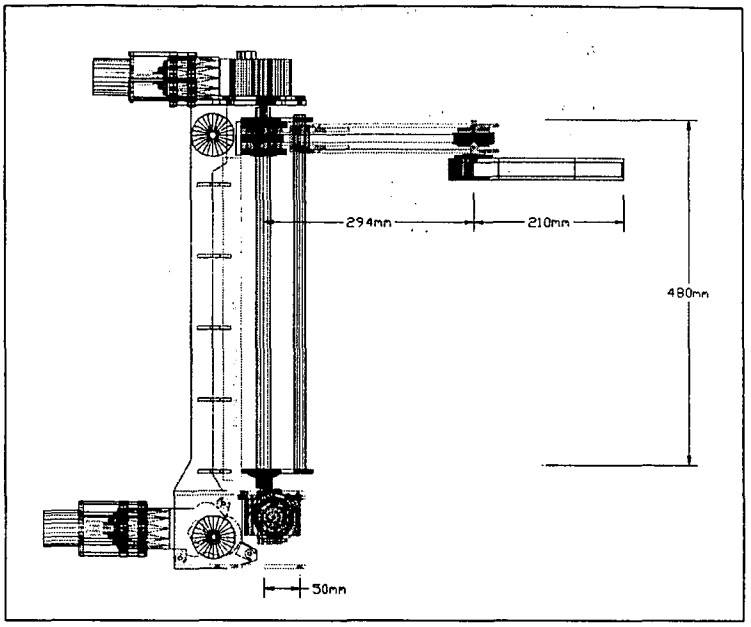
<Fig. 5-7>은 매니퓰레이터를 위에서 찍은 사진이다. 아랫부분에 작물이 놓여지게 된다. 쇼울더 링크 와 엘보우 링크는 지면에 대하여 평행하게 회전하게 되고 Z축 방향 즉, 지면 수직인 방향으로의 이동은 쇼울더 링크와 엘보우 링크가 동시에 <Fig. 5-6>의 슬라이딩 가이드(7)를 따라서 이송되게 된다. 이렇게 함으로써 Z축 방향의 위치결정을 정확하게 할 수 있다. Z 축의 이론상 이송속도 계산은 수직 이송용 벨트 풀리의 직경 × 모터출력축의 회전수 × 분당이송속도로 구했다.

62 (mm) × PI(3.141592) × 60 (rpm) = 11686.72(mm/min)

쇼울더 링크 및 엘보우 링크의 이론상 최대 회전속도는 모터 출력 축과 감속 없이 연결되어 있으므로 60rpm으로 설정했다.

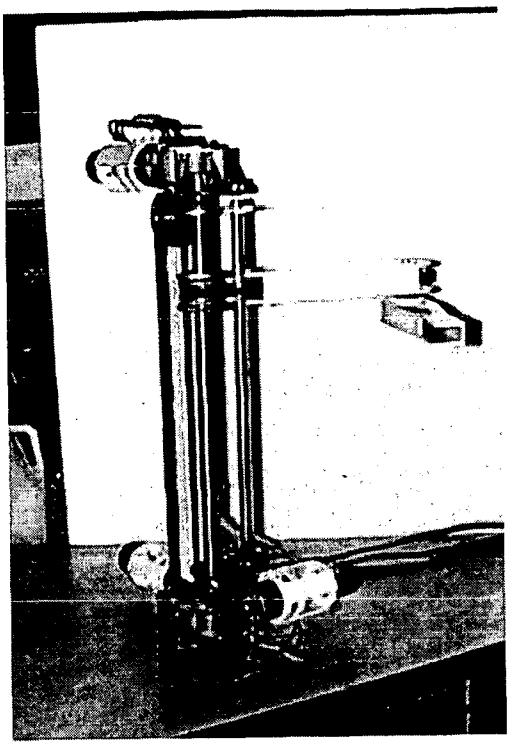


<Fig. 5-8> Dimension of top view



<Fig. 5-9> Dimension of side view

<Fig. 5-8> 과 <Fig. 5-9>는 매니퓰레이터의 작동범위 및 치수를 나타낸다.



<Fig. 5-10> Picture of manipulator

엘보우 링크 및 쇼울더 링크는 모두 5mm 의 ABS 수지를 이용해서 제작하였다. ABS수지는 가벼울 뿐만 아니라 가공이 쉽고 내구성이 있어 적합하다고 판단되었기 때문에 사용하였다. 이렇게 함으로써 링크의 무게를 최소화하여 소모 전력을 줄일 뿐만 아니라 매니퓰레이터의 제어를 쉽게 하고 작동도 신속하도록 하였다.

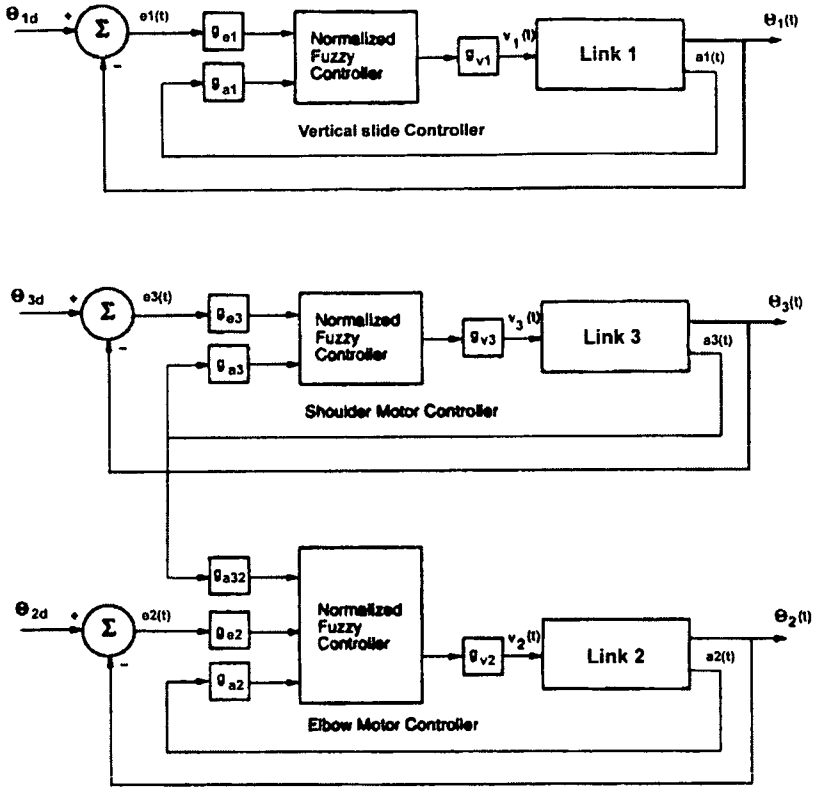
다. 제어장치

모터 제어용 알고리즘은 퍼지제어를 이용하였다. 오이의 수확작업은 질량이나 체적이 유동적이고 충격을 가하면 상처를 쉽게 받는 유연한 생명체이므

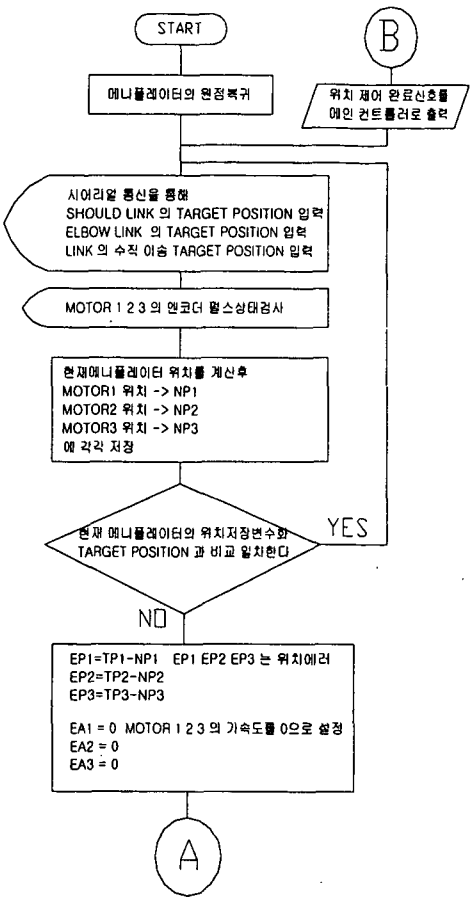
로 모터구동에 있어서도 이러한 점들을 고려하여 개발하는 것이 필요하다.

$$E_1^j \text{ and } A_1^k \; THEN \; V_1^m$$

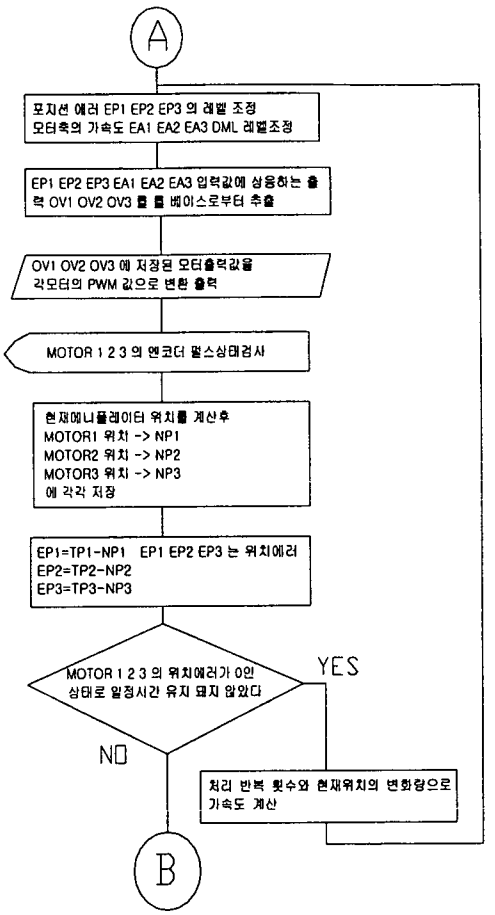
이식에서 E^j_1 은 위치오차를 의미하는 것으로 $e_i(t)=\Theta_{id}(t)-\Theta_i(t)$ 이다. 이는 가고자하는 목표위치 와 현재 링크가 취하고 있는 위치의 차이 값을 의미한다. A^k_1 은 링크간 연결 축에 연결되어있는 가속도계에서 입력되는 가속도 값을 의미하지만 본 논문의 회로에서는 엔코더에서 입력되어지는 위치 값을 시간 함수로 2번 미분하여 사용하였다. V^m_1은 엠프를 통해 출력되어 모터에 인가되는 전압을 의미하는 것으로 위 퍼지 규칙은 위치 오차 값과 모터 회전축의 각가속도 값을 기반으로 여러 개의 퍼지 규칙을 적용 유연하게 대응하는 출력 전압을 발생시키게 되는 것이다. 퍼지 제어기의 기본 회로도는 <Fig. 5-11>에 나타나 있다. 링크 1 은 매니퓰레이터의 수직방향 이송용 슬라이드 축을 의미하고 링크 2 는 모터 2가 연결된 링크로써 엘보우 링크에 해당된다. 마지막 링크 3는 모터 3 가 연결된 링크로써 쇼울더 링크에 해당한다. 링크 2 와 링크 3 는 각각의 운동이 서로 영향을 미친다 따라서 링크 2의 퍼지 제어기 입력중 한 개를 링크 3의 가속도 출력 값을 취하고 있다. 이렇게 함으로써 링크 2가 위치결정을 위해 움직일 경우 모터 3 에 적정전압이 인가되어 유연하게 작동되게된다. 전압의 인가 방식은 PWM (pulse width modulation)방식으로 하였다. 즉 높은 전압으로 인가할 때에는 ON펄스 폭은 크게 하고, OFF 펄스 폭을 작게 한다. 상대적으로 낮은 전압으로 인가할 때에는 OFF펄스 폭을 ON펄스 폭보다 길게 하는 것이다.



<Fig. 5-11> Fuzzy control system design for direct Fuzzy control design



<Fig. 5-12> Flow chart 1 of control program



<Fig. 5-13> Flow chart 2 of control program

<Fig. 5-12>와 <Fig. 5-13>은 제어 원리를 나타낸 프로그램 흐름도 이다. 처음 프로그램이 시작되면 먼저 매니퓰레이터의 모터 1, 2, 3을 원점으로 복귀시킨다. 이는 매니퓰레이터의 위치가 기억되는 변수의 값과 실제 매니퓰레이터의 위치를 일치시키는 작업으로 매우 중요한 작업이다. 만일 원점이 교차

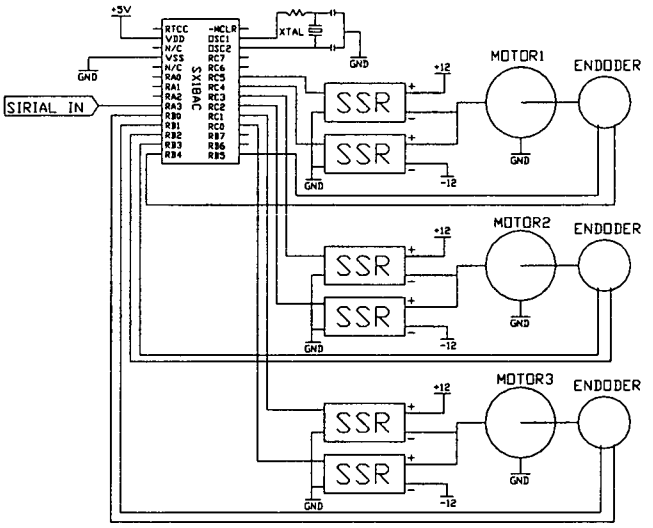
되지 않으면 작업시 일정량의 오차가 계속적으로 발생한다.

그 다음으로 처리되는 것은 메인 컨트롤러로부터 매니퓰레이터가 이동하여야할 목표지점을 통신을 통해 입력받는 작업이다. 이는 본 제어장치가 독립적으로 모든 처리를 하는 것이 아니고 단순히 주어지는 각도로 모터를 회전시키는 작업만을 퍼지이론을 통해서 수행한다는 것을 의미한다. 만일 입력되어지는 값이 없다면 매니퓰레이터는 동작하지 않는다. 이렇게 대기동작을 하는 부분은 <Fig. 5-12>에서 첫 번째 만나는 비교문이다. 이 비교문은 현재의 매니퓰레이터의 위치와 통신으로 주어지는 목표 위치와의 차이가 없을 경우 혹은 입력되어지는 위치 값이 없을 경우 계속적으로 입력 값을 기다리게 하는 비교문이다. 만일 입력되어지는 값은 없으나 외부 힘에 의해 매니퓰레이터가 움직여졌을 경우에는 이 루프를 벗어나게 된다. 왜냐하면 현재의 매니퓰레이터의 위치를 계속 검사하여 이전에 입력되어진 목표위치 즉 이전 값과 계속적으로 비교를 하기 때문이다. 따라서 첫 번째 루프에서 빠져 나올 수 있는 경우의 수는 위 두 가지 경우 이외에는 없다. 목표 값이 주어 졌을 경우에는 현재의 매니퓰레이터의 위치와 목표하는 위치가 틀리므로 비교 문에서 NO 방향으로 빠져나오게 된다. 그러면 다음 처리로 들어가게 된다. 다음 처리과정은 퍼지제어기의 두 입력값 즉 변위 오차(목표로 하는 위치 - 현재위치)와 각 가속도를 구하게 되는데 초기 정지상태의 매니퓰레이터에서 구동을 시작하는 것이기 때문에 가속도는 0으로 초기 값을 정하여 준다.

<Fig. 5-13>으로 처리가 넘어가면 퍼지제어 입력 값을 일정기준에 의한 레벨에 의해 7단계로 나누어진다. 이는 퍼지제어기의 퍼지룰 적용하기 쉽게 하기 위해서다. 즉 변위차를 -3, -2, -1, 0, 1, 2, 3 중에서 하나로 결정하고, 가속도를 마찬가지로 -3, -2, -1, 0, 1, 2, 3 중에서 맞는 레벨로 선택한다. 이렇게 단순화시킨 입력 값을 토대로 룰 베이스에서 결정된 출력값 즉 모터의 전압을 변수에 저장한다. 이 값을 기준으로 모터에 PWM 펄스를 출력한다. 이러한 작업을 목표위치에 도달할 때까지 반복하는데 원하는 위치에 갔다고 해서 바로 루프를 빠져 나올 경우 매니퓰레이터의 진동에 의한 오차를 벗어날

수 없다. 이를 방지하기 위하여 모터가 완전히 정지한 것을 확인하기 위하여 일정시간동안 목표위치에 서 있음을 확인한 후에 루프를 빠져 나오도록 비교문을 사용한다.

일정시간 목표위치에 있었음이 확인되면 <Fig. 5-12>의 초기 (B) 부분으로 이동 목표위치로 이동 완료했음을 메인 컨트롤러에게 알리고 다음 신호를 기다린다. <Fig. 5-14> 는 간략화 시킨 회로도를 나타낸다.



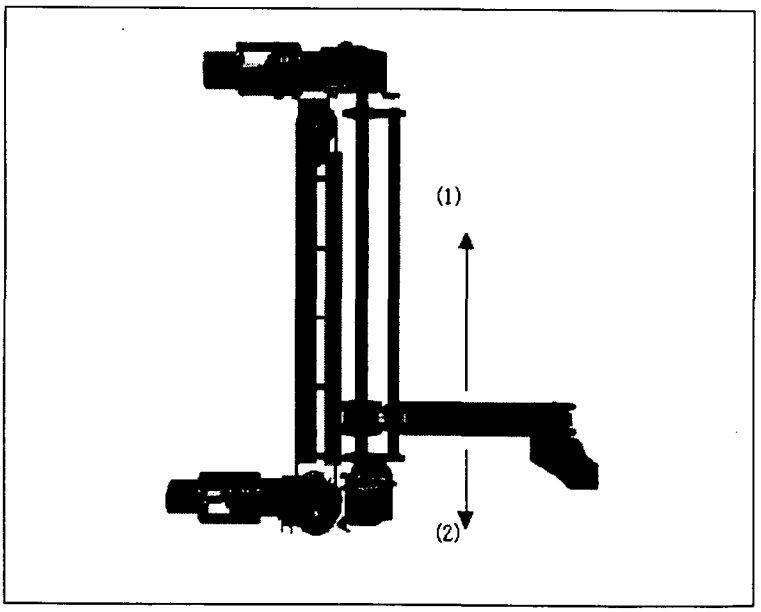
<Fig. 5-14> Circuit of motor controller

3. 작동방법

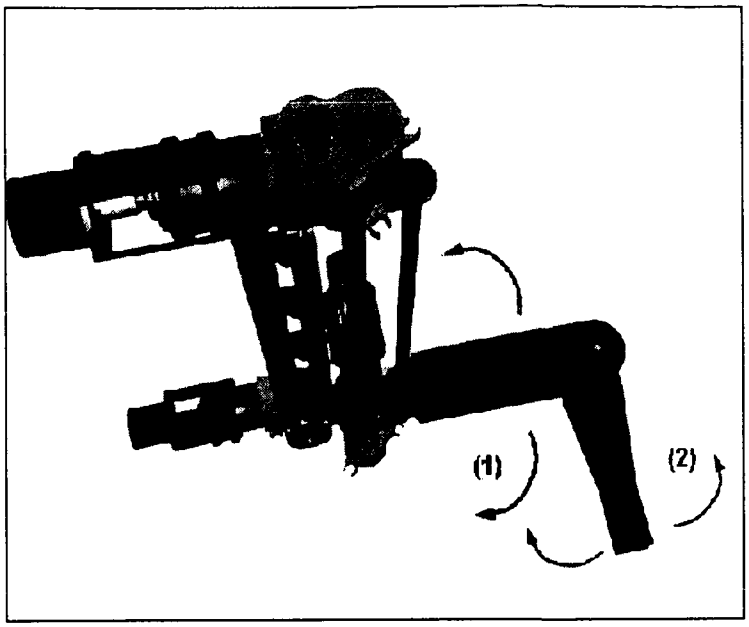
가. 매니퓰레이터 작동방법

메인 컨트롤러는 PC(Personal Computer)가 사용되었다. 실제 오이 수확기에서 필수적인 영상처리 작업을 수행하려면 PC가 사용되어야 하는데 그 이유

는 그래버 보드가 보통 PC에 부착되어 사용되도록 제작되어 나오기 때문이다. 메인 컨트롤러는 영상처리작업을 통해 얻어진 오이의 3차원위치를 토대로 행렬계산을 통해 매니퓰레이터를 구동시킬 수 있는 실제 모터 회전 량을 구하게 된다. 구하여진 모터의 위치 값을 시리얼 통신을 통해 모터 컨트롤러로 보낸다. 그러면 매니퓰레이터는 목표로 하는 위치 즉 오이로 엔드이펙터를 이동시키는 것이다. <Fig. 5-15>에 나타난 그림과 같이 설명하면 모터 1의 회전위치를 + 방향의 위치에러가 발생하도록 보내면 (1) 방향으로 - 의 위치에러가 발생하는 방향으로 목표위치를 보내면 (2) 방향으로 이송하게 된다. 이때 위치에러란 목표위치 - 현재위치가 된다. 목표 위치를 한 개씩 간격을 두고 보내는 것이 아니고 한꺼번에 연속적으로 모터 1, 2, 3 의 목표위치를 한꺼번에 보내면 각 모터가 동시에 그 위치로 회전하여 매니퓰레이터가 작동되게 된다. 모터 2의 회전에 의해 매니퓰레이터가 작동되는 방향은 + 방향의 위치에러가 발생하는 위치로 목표점을 주었을 때 <Fig. 5-16>에서 엘보우 링크가 (1)방향으로 위치에러값이 - 가 되는 목표 값을 주었을 때엔 (2) 방향으로 회전한다.



<Fig. 5-15> Direction of moving elbow link and shoulder link



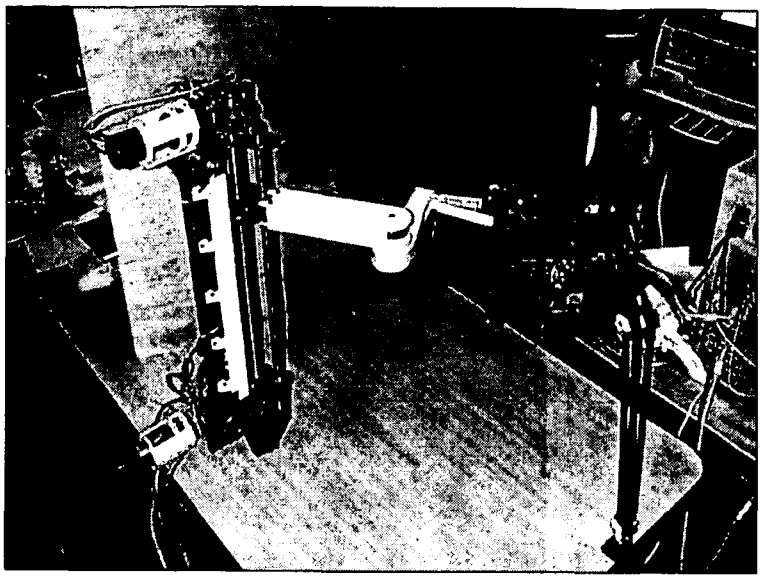
<Fig. 5-16> Direction of moving elbow link

4. 실험방법

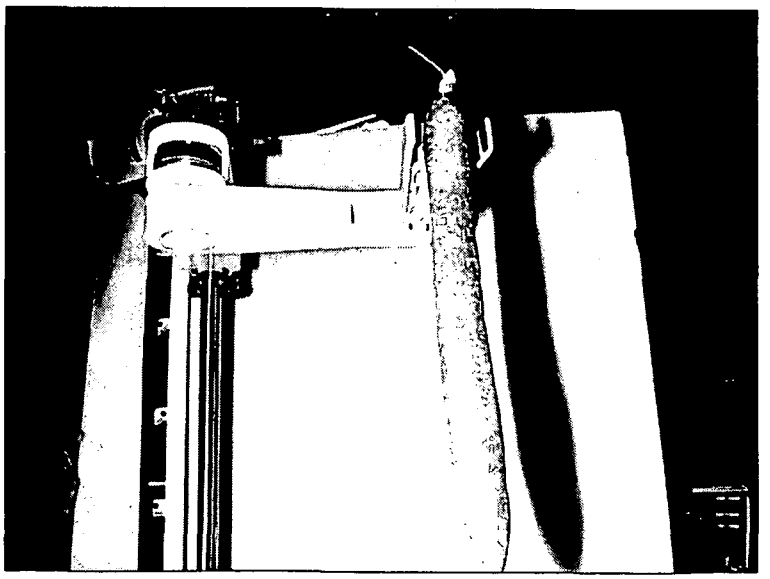
가. 실험설계

매니퓰레이터의 성능실험방법은 임의로 3차원 공간에 놓인 오이의 좌표를 매니퓰레이터의 컨트롤러에 기억시켜놓고 작동시켜서 임의로 제작해 장착한 엔드이펙터를 수확 가능한 위치로 이동시키는 것이다. 이러한 작업을 반복하면서 오이와 엔드이펙터의 위치에러를 측정 기록하고 기록된 데이터를 분석하는 방법으로 실험 설계하였다.

오이를 3차원 공간에서 위치변환을 하며 수확률이 높은 지점을 측정하고자 하였다. <Fig. 5-17>은 실험장치 및 실험방법을 보여주는 그림이다.

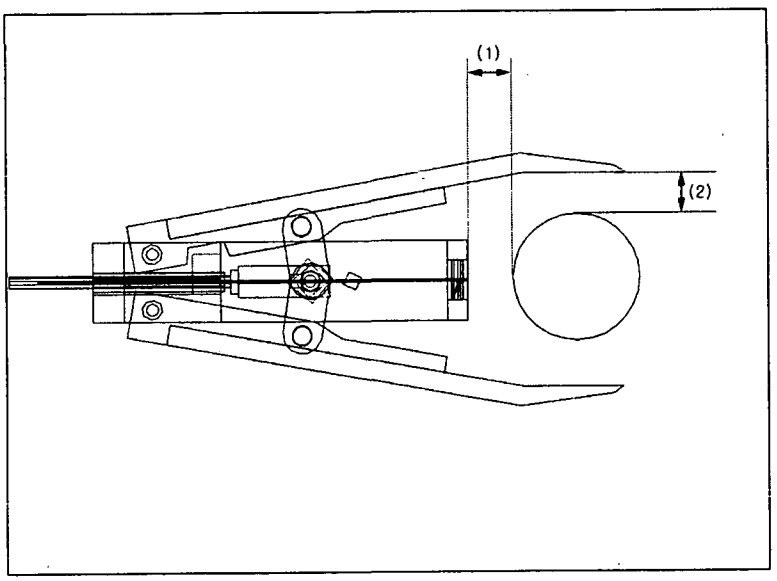


<Fig. 5-17> Method of test manipulator

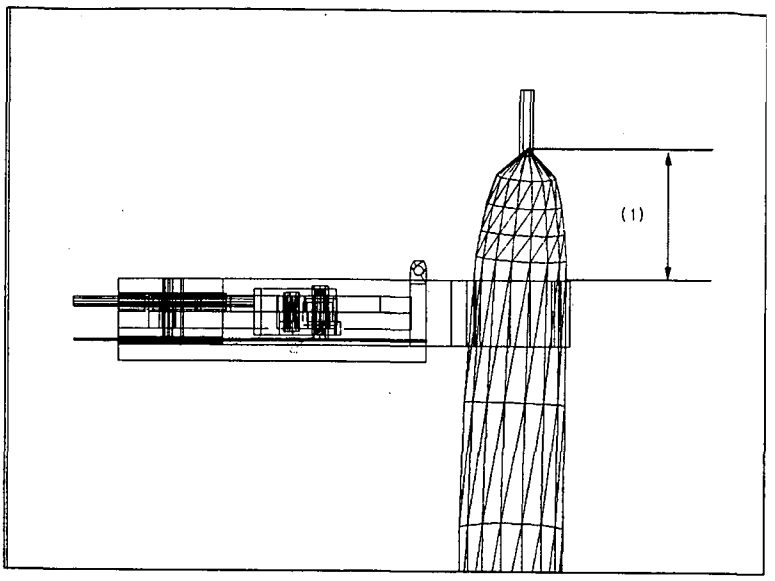


<Fig. 5-18> How to measure the distance

<Fig. 5-18>은 엔드이펙터가 반복동작을 했을 경우 오이로부터 얼마의 오차로 접근하는지를 측정하는 모습이다. 버니어캘리퍼스를 이용하여 X, Y, Z의 3방향으로 측정하였다. 오이를 한 부분에 고정하여 10차례 반복 측정한 뒤 오이를 임의의 다른 위치로 옮겨 10차례 다시 반복 측정을 하는 방식으로 10차례 측정을 하여 기록한 데이터를 통해 본 매니퓰레이터의 성능을 분석하였다. <Fig. 5-19>은 X변위 및 Y변위 측정부를 나타낸 그림이고 <Fig. 5-20>은 Z변위 측정부를 나타낸 그림이다.



<Fig. 5-19> How to measure the X and Y distance



<Fig. 5-20> How to measure the Z distance

나. 분석방법

매니퓰레이터 작동범위내의 임의의 위치에 오이를 두고 10회 반복하면서 측정한 데이터를 통해 매니퓰레이터의 오차를 계산하고 그 오차를 통해서 매니퓰레이터의 성능을 분석하였다.

또한 오차범위가 오이 수확작업에 미칠 영향에 대하여도 분석하였다.

제 4 절 결과 및 고찰

1. 반복 위치오차 측정

설계 제작한 매니퓰레이터의 원점에서 X, Y, Z축으로 300mm 떨어진 거리에 오이를 고정시켜 10회 반복 측정한 후, 오이를 원점에서 X, Y, Z축으로 400mm의 위치로 이동시켜 매니퓰레이터를 작동시키면서 엔드이펙터와 오이와의 거리를 측정하였다. 그 결과 <Table 5-1>와 같은 오차가 생겼으며, 구체적인 측정결과는 <Table 5-2>, <Table 5-3> <Table 5-4>와 같이 나타났다.

<Table 5-1> Maximum error of the measured distance between the tips of each position <unit : mm>

Maximum error / Measured No.	X axis		Y axis		Z axis	
	+	−	+	−	+	−
1	0.86	1.09	0.37	0.28	0.37	0.28
2	0.25	0.35	0.22	0.33	0.14	0.36
3	0.26	0.24	0.11	0.14	0.17	0.18

측정된 오차의 범위를 보면 각 좌표에서 1mm 이하로 나타난 것을 알 수 있다. 또한 매니퓰레이터의 원점에서 X, Y, Z축의 위치에 대해서 동일한 오차범위가 나타나는 것으로 보아 오이의 위치에 영향을 받지 않는 것으로 나타났다.

<Table 5-2> Maximum error of the distance for the measured No. 1

<unit : mm>

Axis / Number	X		Average	Y		Average	Z		Average
	300	400		300	400		300	400	
1	10.2	10.1	10.15	10.8	10.8	10.8	30.9	30.7	30.8
2	11.1	11.1	11.1	10.3	10.3	10.3	30	30.5	30.25
3	10.2	10.2	10.2	10.5	10.5	10.5	30.2	30.3	30.25
4	10.8	10.8	10.8	10.9	11	10.95	30.6	30.5	30.55
5	9.5	9.5	9.5	9.6	9.5	9.55	30.4	30.5	30.45
6	9.8	9.7	9.75	7.1	7.1	7.1	30.3	30.1	30.2
7	11.0	11.1	11.05	9.5	9.5	9.5	30.5	30.4	30.45
8	10.5	10.5	10.5	10.1	10	10.05	30.2	30.5	30.35
9	10.3	10.3	10.3	9.8	9.7	9.75	30.2	30.1	30.15
10	9.2	9.1	9.15	9.6	9.4	9.5	30.8	30.8	30.8
Average	10.26	10.24	10.25	9.82	9.78	9.80	30.41	30.44	30.43
Standard deviation	0.63	0.67	-	1.08	1.10	-	0.29	0.23	-
Standard error	0.20	0.21	-	0.34	0.35	-	0.09	0.07	-
Maximum error	+0.86 -1.09			+0.37 -0.28			+0.37 -0.28		

<Table 5-2>에서 Y, Z축은 최대오차가 +0.37 -0.28로 낮은 오차 범위에서 같은 수치를 나타냈다. X축은 +0.86 -1.09로 오차 범위가 Y, Z축에 비해 높게 나타났으나 엔드이펙터가 오이를 파지하기 위해 접근하는 데는 문제가 없었다. 표준편차는 X, Y, Z축의 300, 400의 경우 0.63, 0.67, 1.08, 1.10, 0.29, 0.23이었으며, 표준오차는 0.20, 0.21, 0.34, 0.35, 0.09, 0.07이었다. 다음의 식을 이용하여 T-검증을 한 결과 X, Y, Z축의 등분산은 0.07, 0.08, -0.26으로 나타났다.

<Table 5-3> Maximum error of the distance for the measured No. 2

<unit : mm>

Axis / Number	X		Average	Y		Average	Z		Average
	300	400		300	400		300	400	
1	11.4	11.2	11.3	8.3	8.4	8.35	28.1	28.2	28.15
2	12.0	11.8	11.9	8.6	8.6	8.6	28.6	28.7	28.65
3	11.6	11.5	11.55	8.9	8.7	8.8	28.5	28.5	28.5
4	11.4	11.4	11.4	8.3	8.2	8.25	28.6	28.4	28.5
5	11.6	11.7	11.65	8.8	8.7	8.75	28.7	28.6	28.65
6	11.8	11.9	11.85	8.5	8.6	8.55	28.4	28.4	28.4
7	11.7	11.7	11.7	8.6	8.5	8.55	28.6	28.5	28.55
8	11.9	11.6	11.75	8.4	8.5	8.45	28.5	28.5	28.5
9	11.6	11.5	11.55	8.9	8.8	8.85	28.6	28.7	28.65
10	11.8	11.8	11.8	8.6	8.6	8.6	28.7	28.5	28.6
Average	11.68	11.61	11.64	8.59	8.56	8.57	28.53	28.50	28.52
Standard deviation	0.20	0.21	-	0.22	0.17	-	0.18	0.15	-
Standard error	0.06	0.07	-	0.07	0.05	-	0.06	0.05	-
Maximum error	+0.25 -0.35			+0.22 -0.33			+0.14 -0.36		

<Table 5-3>와 <Table 5-4>에서 X, Y, Z축은 최대오차가 0.5mm내외로 오차범위가 낮게 나타났다. 이것은 수확작업을 하는데 있어서 엔드이펙터가 오이에 정확히 접근한다는 것을 나타내고 있다. 표준편차는 X, Y, Z축의 300, 400의 경우 0.20, 0.21, 0.22, 0.17, 0.18, 0.15였으며, 표준오차는 0.06, 0.07, 0.07, 0.05, 0.06, 0.05이었다. T-검증을 한 결과 X, Y, Z축의 등분산은 0.76, 0.34, 0.41로 나타났다.

<Table 5-4> Maximum error of the distance for the measured No. 3

<unit : mm>

Axis / Number	X		Average	Y		Average	Z		Average
	300	400		300	400		300	400	
1	11.8	11.8	11.8	8.6	8.6	8.6	25.7	25.9	25.8
2	11.5	11.5	11.5	8.9	8.8	8.85	25.6	25.6	25.6
3	11.9	11.9	11.9	8.4	8.5	8.45	25.9	25.8	25.85
4	11.6	11.4	11.5	8.5	8.5	8.5	25.8	25.6	25.7
5	11.5	11.6	11.55	8.7	8.4	8.55	25.6	25.6	25.6
6	11.6	11.5	11.55	8.5	8.6	8.55	25.7	25.7	25.7
7	11.7	11.7	11.7	8.8	8.6	8.7	25.3	25.6	25.45
8	11.4	11.4	11.4	8.5	8.6	8.55	25.8	25.7	25.75
9	11.8	11.9	11.85	8.6	8.5	8.55	25.5	25.7	25.6
10	11.6	11.7	11.65	8.8	8.7	8.75	25.7	25.5	25.6
Average	11.64	11.64	11.64	8.63	8.58	8.60	25.66	25.67	25.66
Standard deviation	0.16	0.19	-	0.16	0.11	-	0.17	0.12	-
Standard error	0.05	0.06	-	0.05	0.04	-	0.05	0.04	-
Maximum error	+0.26 -0.24			+0.11 -0.14			+0.17 -0.18		

<Table 5-4>에서 표준편차는 X, Y, Z축의 300, 400의 경우 0.16, 0.19, 0.16, 0.11, 0.17, 0.12이었으며, 표준오차는 0.05, 0.06, 0.05, 0.04, 0.05, 0.04이었다. T-검증을 한 결과 X, Y, Z축의 등분산은 ±0.00, 0.79, -0.15로 나타났다.

2. 매니퓰레이터의 수확작업시 적합성

매니퓰레이터가 목표위치에 도달되는 시간은 그 목표로 하는 위치에 따라 다르다. 하지만 각 모터의 정격부하시 회전수에 따라 계산된 값과 차이가 없게 나타나는 것으로 측정되었다. 즉 작업공간내의 한 점에서 다른 한 점으로 이동하는데 걸리는 최대 시간은 3초 미만이다. 이것은 매니퓰레이터의 상승 하강 시간 때문이다. 즉 Z축의 이동시간을 아래의 식과 같이 계산하였다.

$$62(mm) \times \pi(3.14592) \times 60(rpm) = 11686.72(mm/min)$$

Z축의 총 길이가 432mm임을 감안 할 때, Z축의 아래 끝점에서 윗 끝점으로의 이동시간은 432/11686.72 = 0.3696(min) 즉 2.21(sec)가 된다. 즉 목표위치로의 이동시간의 최대 소요시간은 2.21초가 된다. 소요시간을 줄이려면 상하이동시간을 단축시키는 방법이 있다. 이를 위해서 매니퓰레이터 상하 이송용 풀리의 직경을 크게 해주면 된다. 즉 영상처리장치, 오이 파지 및 줄기 절단장치, 대차장치의 성능과 속도를 고려하여 결정하여야 할 것으로 판정된다.

제 5 절 요약 및 결론

본 연구는 오이 수확기의 매니퓰레이터를 개발하는 것이다. 연구 목적은 오이 수확에 적합한 링크구조 와 메커니즘를 설계하는 것이다. 그래서 매니퓰레이터와 모터 컨트롤러를 설계하였다. 본 매니퓰레이터는 DC모터로 작동되는데, 작동원리는 퍼지 논리이론을 적용하여 부드러운 작동이 가능하도록하였다.

본 매니퓰레이터의 사양은 3축 PRR(prismatic revolute revolute)형 매니퓰레이터이다. 그리고 각각의 모터는 회전관성모멘트가 실리지 않는 부분에 장착되고 동력은 타이밍 벨트를 이용하여 전달하는 방식을 취하였다. 이렇게 함으로써 매니퓰레이터의 작동성능을 높였고, 내구성 또한 증가되도록 하였다. 주요 연구결과를 요약하면 다음과 같다.

1. 매니퓰레이터의 쇼울더링크 및 엘보우 링크에 모터 및 감속기를 장착시키지 않음으로 해서 회전 관성 모멘트를 최소화 시켰다. 이렇게 함으로써 쇼울더 링크 및 엘보우 링크 회전이 유연하게 작동되며, 또한 고속동작이 가능하였다.
2. 각 축을 제어함에 있어 동력원으로 모터를 사용하였다. 모터는 제어가 용이한 장점이 있다. 모터 제어방법에는 여러 가지가 있겠으나 본 연구에서는 퍼지제어이론을 적용 작업환경에 맞게 rule base를 작성하여 대응하도록 하였다.
3. 모터의 회전력을 각 링크에 전달하기 위해 감속을 하였다. 이때 감속용으로 웜기어를 사용하고 동력전달은 타이밍 벨트를 이용함으로서 저가의 실용적인 매니퓰레이터를 설계하였다.
4. 매니퓰레이터의 반복 오차를 측정한 결과 월드좌표계(World coordinate)에서 , X축으로 최대오차 +0.26mm -0.24mm , Y축으로 최대오차 +0.11mm -0.14mm , Z축으로 최대오차 +0.17mm -0.18mm 로 나타나 수확작업에 적합한 것으로 나타났다.

제 6 장 영상처리시스템 개발

제 1 절 서 론

컴퓨터 비젼(Computer vision)이란 투시된 영상들로부터 주어진 장면(scene)에 관한 유용한 정보를 추출하는 작업을 말한다. 즉, 여러 화소(pixel)들의 배열인 영상으로부터 물리적인 대상을 명확하고 의미 있게 기술하도록 하는 과정을 말한다. 시각 기관의 기능을 컴퓨터로 처리하는 컴퓨터시각(Computer Vision)의 연구에서는 컴퓨터로 하여금 영상을 획득하고 분석하여 필요한 정보를 얻게 한다. 즉, 명암도 영상(intensity image), 색채 영상(color image), 또는 거리정보 등의 시각 정보가 입력되었을 때 다양한 영상처리 기법을 이용하여 영상의 내용을 이해 또는 식별하도록 하고 있다. 컴퓨터 시각은 2차원적인 특성을 갖는 패턴을 인식하는 분야와 3차원 물체를 인식하는 분야로 나눌 수 있으며, 문자 인식, 목표물 추적, 원격 감지, 공장 자동화 그리고 로보트공학 등에 응용될 수 있다. 물체를 인식한다는 것은 이미 알려진 물체를 이해한다는 것일 수도 있으나 미지의 물체를 인식하려면 다양하게 물체의 모델들을 저장하여 입력된 영상과의 정합 과정을 통해서 인식할 수 있다.

일반적으로 실세계는 불변의 성질을 가진 3차원의 물체(object)들로 되어 있지만 입력된 영상은 불명확하거나 오인된 변형물도 많이 포함하고 있다. 따라서, 인간의 시각 시스템은 입력자료로부터 추출한 정보와, 실세계에 대한 가설 및 지식을 사용하여 불확실한 입력 영상을 이해하고 있다. 인간의 시각시스템과 같은 비전시스템을 구성하는 것은 다음과 같은 이유로 어렵다. 첫째, 하나의 장면이 많은 제약 조건하에서 영상으로 되기 때문에 영상 자체만으로는 그 장면을 회복하기에 충분한 정보가 제공되지 않으며, 실세계의 3차원 장면이 2차원 영상으로 투영될 때 깊이에 대한 정보가 없어지게 된다. 따라서 영상 자료가 갖는 모호성(ambiguity)을 해결하기 위하여 적절한 가설과 영상 자

료에서부터 얻어지는 측정값을 이용하게 되는데, 이러한 정보를 어떻게 활용할 것인가에 대한 연구가 어려운 실정이다. 둘째, 하나의 영상이 형성되는 과정에는 많은 요소들이 작용하기 때문에 어려움이 있다. 즉, 하나의 물체에 관한 영상은 그 물체의 표면 물질, 주변 환경, 발광체의 각도, 주변의 빛, 카메라의 각도와 특성 등에 의해 많은 영향을 받게 된다. 이러한 요소들은 한 화소의 명암도에 영향을 끼치므로, 어느 정도 영향을 미치는지 결정하는 것이 어렵다. 셋째, 일반적으로 영상을 이해하는 작업은 그 문제 영역에 관한 사전 지식을 요구한다. 대부분의 경우에 있어서 영상에서 관찰될 수 있는 특성은 매우 미약하며 인간의 시각 시스템 역시 찾고자 하는 물체에 관한 사전 지식을 활용하고 있다. 따라서 이러한 사전 지식없이 영상을 이해한다는 것은 불가능하다. 그러나 인간이 갖고있는 엄청난 양의 시각 세계에 대한 사전 정보를 시스템이 이해할 수 있도록 표현하는 작업은 매우 어려운 일이다. 넷째, 하나의 간단한 작업에 대해서 거대한 양의 정보를 처리해야 한다는 것이다. 이러한 방법은 간단한 물체일지라도 처리시간이 많이 요구되므로, 두드러진 물체의 특징들만을 이용하여 좀 더 간단하게 처리하는 방법을 찾아야 한다. 지금까지의 컴퓨터 시각 연구에서는 대부분 2차원 영상을 명암도에 따라 처리하는 기법을 사용하였으나, 물체가 놓여져 있는 상태나 물체 구조상의 복잡성, 그리고 조명과 그림자 등에 따라 물체의 특징을 추출하기가 용이하지 않다. 특히 작업 대상체의 형상인식이나 3차원적인 거리 정보 추출과 가공 작업을 위한 대상체의 형상인식 및 정보의 제공, 센서 장치의 수평 위치제어와 수직 위치제어, 거리제어 등의 형상제공시 3차원적인 해석이 더 용이하리라 본다.

제 2 절 영상처리 시스템의 연구동향

컴퓨터 시각장치에 의한 3차원 정보를 이용한 연구현황으로서는 최근에 많이 연구되어지고 있으며, 그 중에서 가장 일반적인 방법으로 사람의 눈과 비슷한 구조를 가진 스테레오 시각을 이용하는 방법이 있다. 이는 한 대의 카메라를 통해 얻어지는 3차원 형상 정보는 3차원 대상체가 2차원 평면상에 투사되어 얻어진 정보이므로 대상체가 갖는 3차원 공간 정보를 얻기 위한 접근법이 필요하다. 스테레오 시각(stereo vision)은 각각의 카메라를 통하여 2차원 영상을 얻고 이들간에 삼각측량법을 이용하여 3차원 정보를 산출하는 것이다. 그 방법으로서 두 단계가 있는데, 첫째 단계는 두 영상에서 선택된 점들의 대응관계(correspondence)를 가지는 가를 판단하고 그 점에서의 깊이정보를 계산하는 일이다. 둘째 단계는 보간법 등을 적용하여 영상의 모든 점에 대하여 깊이정보 맵을 구하고 또 그것을 이용하여 3차원 공간의 표면을 기술하는 것이다. 하지만 스테레오 방법의 단점으로서는 다음과 같은 단점을 내포하고 있는 실정이다. 1) 한쪽 카메라의 임의 점과 다른 카메라의 대응점 추출을 위한 영상 좌표계상의 매핑 문제. 2) 촛점거리 및 촛점각을 계산하기 쉽지 않다. 3) 획득된 거리영상은 감지 대상의 기하학적인 특징 때문에 물체의 심도정보뿐만 아니라 많은 잡음을 포함하고 있다. 그러므로 전처리 과정이 필요하다. 4) 연산시간이 길며 대상체에 대한 정확한 사전 정보가 요구되기 때문에 유연성 및 일반성이 결여되어 있다..

이러한 어려운 문제를 해결하기 위해 최근에는 고성능의 프로세서가 개발되어 계산시간문제가 해결되고 강력한 고성능 스테레오 모델이 개발됨으로써 특수 목적을 위한 실용적인 스테레오 비젼 시스템이 등장하고 있다.

대상체에 대한 3차원 정보를 얻기 위한 방법중 사람의 시각구성과 유사한 구조를 가진 스테레오 방법은 2개의 카메라 렌즈를 통하여 얻어진 영상을 이용하여 3차원 거리정보를 얻고, 표면의 재구성을 추출하는 것이다. 이는 각각의 영상에 존재하는 한점의 거리와 서로 다른 촛점에서 얻은 두 영상에서의

대응되는 상대적 차이에 의해 계산하며, 특징점들의 변위로부터 거리지도(depth map)을 얻어 3차원 표면을 재구상한다. 또한, 인간의 시각시스템에서 사용하는 자유도를 따라, 능동적인 스테레오 시각장치를 구현하는 연구가 최근에, 유럽과 일본에서 진행중이며, 많은 연구단체에서 이 분야에 서로다른 이론과 실험을 하고 있다. 액티브 비젼 연구는 지각시스템을 제어하여 스테레오 구성 배경에 나타내며, 각각의 서로다른 영상에 물체의 거리를 계산할 수 있도록 구현하여 대응배합이 초점절차에 의해 얻어질때마다 거리를 구할수 있도록 구현한다. 또한 카메라와 카메라렌즈의 컴퓨터 제어 사이에 각의 변화조절이 거리를 결정하는데 용이하도록 설계 및 알고리즘을 구현하여 지각 시스템을 고려한 내적 및 외적 매개변수의 변화에 용이하도록 연구가 진행중이다.

Enrico Grosso(1995)등은 능동적/동적 스테레오 시각 (Active / Dynamic Stereo Vision)에 대해 로봇작동을 위한 적절한 시각정보의 특징점문제에 대해 언급하였다. 그는 스테레오 이미지에서 변위는 배경에서 상대적 거리지도를 얻기 위해 광류를 결합시킨다. 배경에서 고정점으로부터 카메라의 거리에 의한 깊이는 중심뿐만이 아니라 상대적으로 측정된다. Narendra, Lynn Abbott (1993)은 능동시각(Active stereo): 변위구성(Integrating Disparity), 한계(Vergence), 초점(Focus), 검색(Aperture), 표면추정을 위한 계산에 대해 연구하였다. 그는 깊이정보의 근원으로서 스테레오 변위(disparity)를 강조하였다. 카메라의 한계와 렌즈촛점은 깊이의 복구를 위해 이용되며 이에 대한 구성(Integration)방법은 배경 기술을 전개하는 것을 바탕으로 능동적 이미지의 변수선택(한계, 초점, 검색, 그리고 줌)에 의해 이루어진다. 접근하는 두 단계로서는 시각적인 추적선택(visual target)과 표면복구(suface reconstruction)가 있다. 이 방법의 구현에 의한 실험 결과로서는 정확한 표면이 충분하게 시각적으로 상세하게 나타났다. 실행한 분석은 거의 2m의 거리에서 0.15%의 평균에러를 나타내며, 불연속적인 깊이를 포함한 배경에 대한 표면의 재구성은 더 복잡하게 구현된다. John Aloimonos(1987)등은 능동시각(Active Vision)의 구현에 있어, 기본적인 몇가지의 문제점을 나타내었다. 소위 능동(active)이라 불리

는 여러 종류의 활동은 센서장치의 기하학 매개변수를 제어하는데 목적이 있다. 능동의 목적은 지각처리(perceptual)결과의 질을 향상시키기 위해 관찰적 현상을 조종하게 다루는 것이다. 본 연구에서는 기본적인 비젼문제에서 수동적(passive)보다 능동적(active)관찰이 더 효과적인 방법으로 해결할 수 있는것을 증명하였다. 특히 영상내의 각 점에서의 밝기는 광원의 위치와 물체표면의 기울기로부터 얻어진다는 밝기 해석법을 지닌 명암의 형상복구(shape from shading)와 거리계산(depth computation), 등고선의 형상복구(shape from contour), 무늬상태의 형상복구(shape from texture), 움직임으로부터 기하학구조(structure from motion)은 수동적보다 능동적 관찰이 더 간단함을 보여준다. Jorge Dias, Carlos Paredes 등은(1998) 이동로봇과 인공시각을 가진 시스템을 이용하여 물체를 추적하기 위한 시뮬레이션 추적실험을 연구하였다. 그는 이동로봇(mobile robot)과 능동 시각시스템(active vision system)을 이용해서 평면에서 물체를 따라 추적하는 방법을 연구하였다. 이 해결방법은 시각시스템을 가지고 로봇을 제어하는 비쥬얼 피드백(visual feedback)을 이용하여 각각의 제어 시스템의 상호작용에 의해 처리되고 이동로봇의 시각적 상호작용 방법에 의해 수행되며, 이러한 두가지 시스템은 일정한 거리에 움직이는 물체를 따르는 것과 이동로봇에 관해 방향을 따르는 것으로 설계되어 있다. 능동시각시스템(active vision system)의 방향과 위치는 실시간으로 목표물(target)을 추적하기에 이용되는 제어에 대한 피드백 신호이다. 이 연구에서는 비쥬얼 피드백(visual feedback)과 시스템의 움직임을 위한 보정(compensation)을 이용하여 시각적 고정, 시각적 추적, 작동의 문제를 처리한다. 능동시각시스템(active vision system)으로 갖추어진 이동로봇은 목표물을 따라 움직이는 추적장치의 시스템 사이에서 문제를 처리한다. 이 시스템 구성은 두 관점을 가지고 있는데, 각각의 제어시스템 사이에서 상호작용과 시각시스템에 의해 제공된 일반적인 피드백 정보의 사용이다. 이 연구에서의 전체적인 해결방안은 이동로봇의 앞에 움직이는 목표물에 대한 추적 설계를 구현하기 위해 시각적 처리에 근거를 두는데 목적이 있다. 또한 시각시스템을 이용해서 물체의 추적을 실시

간(real-time)으로 처리하도록 하고 있다. 몇몇의 작업은 카메라 이미지상의 변화와 카메라 자세 변화와의 관계인 Jacobian을 가지고 로봇을 제어하는 비쥬얼 서보기구(visual servoing)-간접조속장치-에 의해 처리된다. 이 방법은 이미지상의 변화로부터 카메라 자세의 변화를 구할 수 있는 특징을 가지고 있다. Coombs(1992)는 양안시 시각(binocular vision)을 이용하여 실시간으로 시각처리를 구현하도록 연구하였다. 이 시스템은 매니퓰레이터 위에 양안시 시스템을 장착하여 두 이미지의 정보를 시각적처리로 제어되도록 설계하였다. 이러한 연구는 지연시간을 없애기 위한 전조적인 제어설계를 포함하고 있다.

Papanikolopoulos(1993)는 궤도가 이미지 평면에 평행한 물체를 추적하기 위해 매니퓰레이터에 카메라를 장착하는 방법으로 추적처리를 제안하였다. 시각시스템에 의해 충족된 정보는 광류(optical flow)를 기본으로 하는 알고리즘에 의해 처리되는데, 이는 실시간으로 물체를 추적하기에 충분한 정확도를 지니고 있으며, 윈도우의 이미지에서 효과적인 사용은 이러한 방법을 더 향상시킬수 있다고 하였다. Allen(1993)은 3차원적으로 물체를 따라 추적하는 방법으로 시각시스템은 두대의 카메라를 사용하여, 물체에 대해 매니퓰레이터의 시스템으로부터 계산되어진 스테레오이미지를 이용함으로써 실시간으로 계산되어진다. 일단 추적하는 단계가 이루어지면, 매니퓰레이터에 제어되는 시스템은 움직이는 물체를 구분하여 그것을 집게 된다.

최근에는 능동적 시각 원칙(active vision principles)을 기반으로 하여 시스템의 개발을 위해 전 유럽적인 계획(Large European Project)-Vision as Process Project-에 관여하고 있다. 시스템의 발달을 위한 연구의 수행으로는, 카메라 제어, 시각 반사광(ocular reflexes), 실시간 영상처리, 이미지추적, 지각처리방식화(perceptual grouping), 능동적 3차원 모델링(active 3-D Modeling), 그리고 물체인식 들의 연구가 수행되고 있다. 국내에 의해 보고된 연구로서는 김등(1998)은 능동 시각을 위한 자동 초점 시스템을 개발하였다. 그의 연구는 능동 시각 시스템에서의 자동 초점 조절을 위하여 기존의 적외선을 사용한 초점 조절방식이나, 아날로그 신호의 처리방식과 달리 디지털 영상을 바탕으로

한 초점 조절을 시도하였다. 초점 조절에 필수적인 초점값 산출을 위해 잡영에 민감한 일반적인 고대역 통과 필터를 개선한 이동평균 필터로 산출된 초점값은 CCD카메라의 초점조절에 있어 잡영의 영향을 덜 받으며, 하드웨어 구현을 간소화하도록 각종 초점 알고리즘을 실현할수 있는 시스템을 구현하였다. 심등(1994)은 움직이는 물체의 파지를 위하여 스테레오 카메라를 사용한 로봇 매니퓰레이터의 위치 및 자세 제어를 시뮬레이션을 통해 유효성을 확인하였다. 작업대상이 이동물체의 형상을 사전에 알고 있다는 가정하에서 스테레오 비젼과 이미지의 Jacobian을 이용하여 움직이는 물체의 파지를 위한 End-effector의 위치 및 방향제어를 시뮬레이션하였다. 시뮬레이션 결과에 의하면 이동하는 물체의 속도와 위치에 상관없이, 로봇 매니퓰레이터의 End-effector의 자세를 이동물체의 자세로 카메라 이미지상의 변화와 카메라 자세 변화와의 관계인 Jacobian을 가지고 로봇을 제어하는 Visual feedback servoing방법의 유효성을 확인하였다. 손등(1997)은 스테레오 영상처리를 이용한 토마토 위치검출 알고리즘을 개발하였는데. 이 연구에 의하면 토마토 수확로봇개발을 위한 시각구성으로서 스테레오 영상처리를 이용하여 토마토의 3차원 위치정보를 얻기 위해 먼저 토마토의 중심좌표를 계산하여 거리계산을 하도록 연구하였다. 또한 최적의 두 카메라간 거리 결정이 100mm로 구성할 때 카메라 공유면적이 가장 넓은 것으로 분석되었고, 조도 및 광원의 위치가 위치검출에 미치는 영향에서는 순광에 비해 역광에 대한 검출오차가 더 크고 조도에서는 100Lux이하에서 물체인식이 부정확하여 오차가 크게 나타난 것으로 분석되었다. 장등(1998)은 사과수확로봇의 개발을 위한 화상처리를 이용한 사과의 인식 알고리즘을 보고하였다. 대상물 인식을 위한 위치결정 알고리즘을 개발하였는데 색상함수를 구함으로써 대상물을 인식하고 장애물을 검출할 수 있도록 하였다. 사과 인식을 위한 변환된 결정함수 X=(R-G+256)/2 로 사과를 인식하고 상관계수를 사용한 템플레이트 매칭으로 사과의 인식과 좌표를 결정하여 장애물의 검출을 위해서 색상비율에 따라 가지와 잎의 장애물을 쉽게 검출할수 있다고 보고하였다. 특히 사과수확의 경우 사과의 색과 잎의 색은 확연히 다르기 때문에 사과

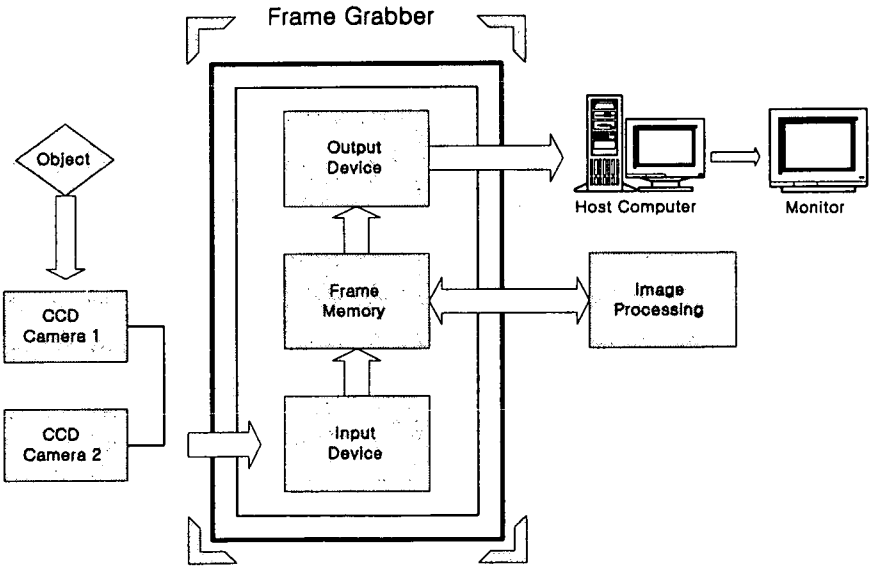
가 앞에 가려져 있더라도 각각의 색상비율에 의해 사과를 인식할수 있을 것으로 사료되지만 오이수확의 경우에 있어서는 거의 색상비율의 차이가 없기 때문에 오이수확에는 적용이 어려울것으로 사료된다. 위와 같이 스테레오비젼에 의한 방법은 사람의 인공시각과 유사한 형태로 해석하기 위해 능동시각시스템(Active stereo vision)에 대한 연구가 많이 이루어지고 있으나, 카메라에 의한 외적요소 및 초점대응거리 등의 많은 문제를 해결하고 있는 실정이다. 또한 카메라와 다른 센서 즉 초음파,레이저 등을 이용하여 물체의 3차원 정보추출에 관해 많은 연구가 이루어지고 있다.

제 3 절 장치 및 방법

1. 실험장치

가. 시스템 개요

일반적으로 사용된 영상처리 시스템의 기능 블록은 그림과 같이 영상신호 입력부, 영상처리부, 주 컴퓨터 및 영상출력부로 구성된다. 컬러 CCD카메라로부터 입력된 영상신호는 RS-170 형태의 아날로그 신호로 컴퓨터에 변환된다.



<Fig. 6-1> Block diagram of image processing system

나. 영상 입력부

입력 센서부에 해당하는 영상 입력장치로는 4.8mm렌즈를 부착한 Ikegami 사의 컬러 CCD 카메라(ICD-703) 2대를 사용한다. 본 연구에서 사용될 카메

라의 제원은 다음과 같다.

<Table 6-1> CCD camera specification

Item	Specification	
CCD camera (Model: ICD-703)	T V	NTSC standard 525Lines
		60 Field/sec 2:1
	Image sensor	1/3inch interline transfer CCD
	Pixel elements	771(H)×492(V)
	Resolution	460TV Lines
	S/N ratio	50dB(AGC OFF)

다. 신호 처리부

RS-170 12.28Mhz의 아날로그 신호 값으로 CCD 카메라에서 출력되는 영상 신호는 컴퓨터와의 인터페이스를 위하여 디지탈 신호값으로 변환되어야 한다. 영상변환은 영상처리보드의 디지타이져(digitizer)에 의해 디지탈 신호로 변환되는데 이 신호값을 이용하여 원하는 처리를 소프트웨어적으로 수행할 수 있게 된다. 컴퓨터 프로그램에 의해 수행된 영상처리 결과는 영상메모리 내부의 LUT(Look Up Table)를 조작함으로써 영상출력 전용의 모니터상에서 확인할 수 있다. 이를 위하여 본 연구에서는 Coreco사의 OCULSU TCI-SE PCI Frame Grabber(CORECO Inc., St. Laurent, CANADA) 를 사용한다. 이 장치는 A/D Convertor 가 내장된 Input Device, Processing Device, Frame Buffer Memory 가 포함된 Storage block, D/A Convertor가 내장된 Output Device 등으로 구성되어 있다. 본 연구에서 사용된 신호처리부의 제원은 <Table 6-2>와 같다.

<Table 6-2> OCULUS TCI-SE True Color Frame Grabber Specification

o IMAGE MEMORY	2MB
o IMAGE FRAME	Linear Memory
o DISPLAY	Uses Host Display
o OVERLAY MEMORY	None
o BIT DEPTH	15 or 24 bit per pixel
o SOURCE	RS-179, CCIR, NTSC, PAL
o Inputs	2 or 3 Composite inputs
o RESOLUTION	640×480×30 Frame/sec

라. 출력부

처리된 결과값과 영상 출력결과는 PC용 모니터로 출력한다. 이는 영상처리부를 통해 처리된 데이터를 R, G, B영상으로 출력한다. 본 연구에서 사용된 출력부의 제원은 다음과 같다.

<Table 6-3> VGA Color Monitor Specification

o Manufacturer	Samsung
o Model No.	SyncMaster 17GLi
o Resolution	1024×768
o Frequency	60Hz

2. 실험방법

가. 스테레오 비젼(Stereo Vision) 시스템

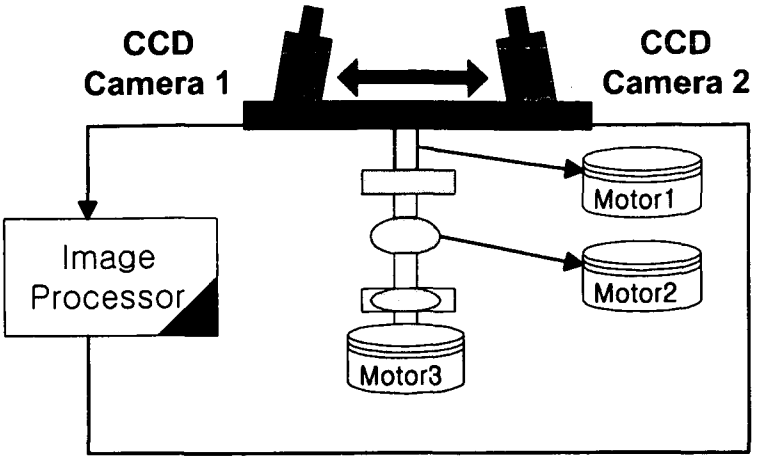
1) 이론적 배경

스테레오 비젼 연구는 지각시스템의 매개변수변화를 제어하는 것과 관련되

며 이 연구는 초기에 Pennsylvania대학에서 R. Bajcsy에 의해 제안되었다. 이 접근은 후에 E. Krotkow가 스테레오 구성을 구현하여 각각의 서로 다른 영상에 배경물체의 거리를 계산할 수 있도록 함으로써, 각각의 이미지에 대한 대응 매칭이 초점절차에 의해 얻어 질 때마다 거리를 구할 수 있도록 구현하였다. Krotkow에 의해 제시된 시스템은 카메라와 카메라렌즈의 컴퓨터 제어 사이에 각의 변화조절이 거리를 결정하는데 용이하도록 되어 있으며, 후에 이 연구는 지각 매개변수(sensory parameters)의 변화로부터 알고리즘을 제시하였다. 최근에 이 연구는 지각 시스템을 고려한 내적 및 외적 매개변수의 변화를 용이하게 실행하도록 연구가 진행중이다. 내적 매개변수(촛점, 초점거리,)는 모터가 달린 렌즈를 사용함으로 제어할 수 있고 외적 매개변수제어는 인간의 시각 시스템에서 사용하는 자유도를 따라서 나누어진다. 인간은 두 눈이 수평선상-시각범위(eye vergence), 수직적-시각 팬(eye panning)위치제어를 한다. 게다가 인간은 각각의 눈이 광학축(cyclotorsion)을 둘레로 회전할 수도 있다. 이러한 3 자유도는 시각구성의 조건으로 정의한다. 게다가 목(neck)의 움직임(pan and tilt)과 몸체 운동(3차원 공간 위치변화)은 시점 선택에서 사용될 수 있다.

2) 기하학 구조 (Vision system geometry)

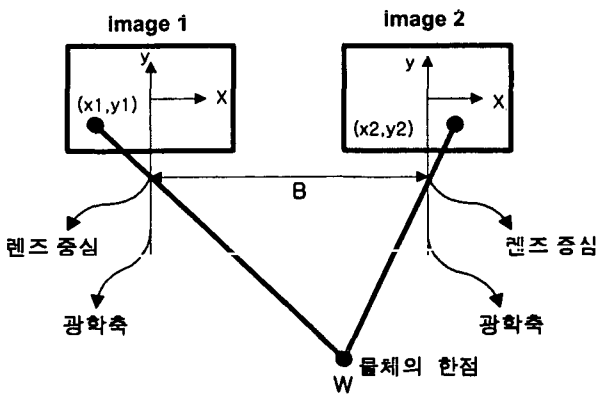
시각 시스템은 아래 그림과 같은 구조와 같이 카메라 2대를 장착하고 사람과 유사한 방식처럼 3자유도가 가능하게 설계되어진다. 능동 시각 시스템은 On-board컴퓨터에 연결하여 동적제어가 가능하게 설계되며 2대의 카메라로 이미지를 처리하도록 연결되어 진다. 본 연구의 능동시각시스템은 3대의 스테핑모터를 이용하였고 2대의 컬러CCD Camera는 각각의 카메라의 범위각제어(vergence control)가 가능하도록 설계하였으며, Head Tilt, Neck Pan이 작동되도록 구현하였다. 또한 이 시스템의 관리와 인터페이스는 컴퓨터에 의해 처리되고 주처리장치에 연결된다.



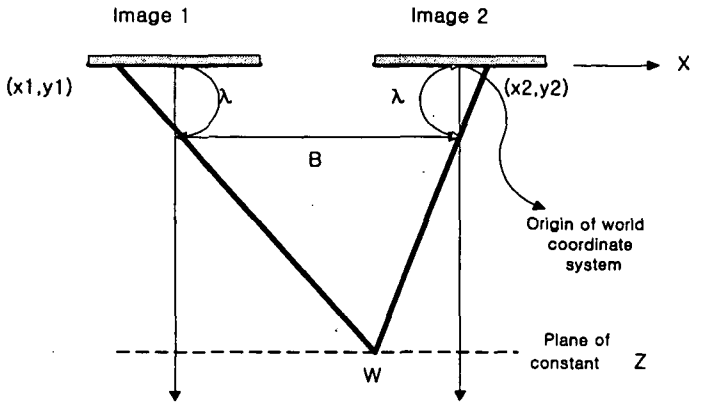
<Fig. 6-2> Stereo vision system

3) 거리정보 및 카메라의 범위각(Vergence angle)제어

가) 스테레오 영상의 거리정보추출



<Fig. 6-3> Stereo image processing



<Fig. 6-4> Stereo image processing

스테레오 영상은 한 개의 물체의 점에 대하여 두 개의 카메라를 이용하여 영상을 따로 얻는 것을 말하는데, 두 카메라 렌즈의 중심과 중심을 연결한 선을 기저선(Baseline)이라 하고 그 거리는 B이다. 두 카메라간의 초점 거리는 모두 λ이다. 두 카메라는 동일하다고 가정하고 각 카메라의 x,y축과 물체의 x,y축은 평행하다고 하면 영상면의 한 점 (x_1, y_1) 이고 물체의 첫 번째 영상의 좌표계와 동일한 것을 사용했을 때 그 점의 좌표를 $(u_1,\ v_1,\ w_1)$이라 하면

$$u_1 = \frac{x_1(\lambda - w_1)}{\lambda}$$

이다.

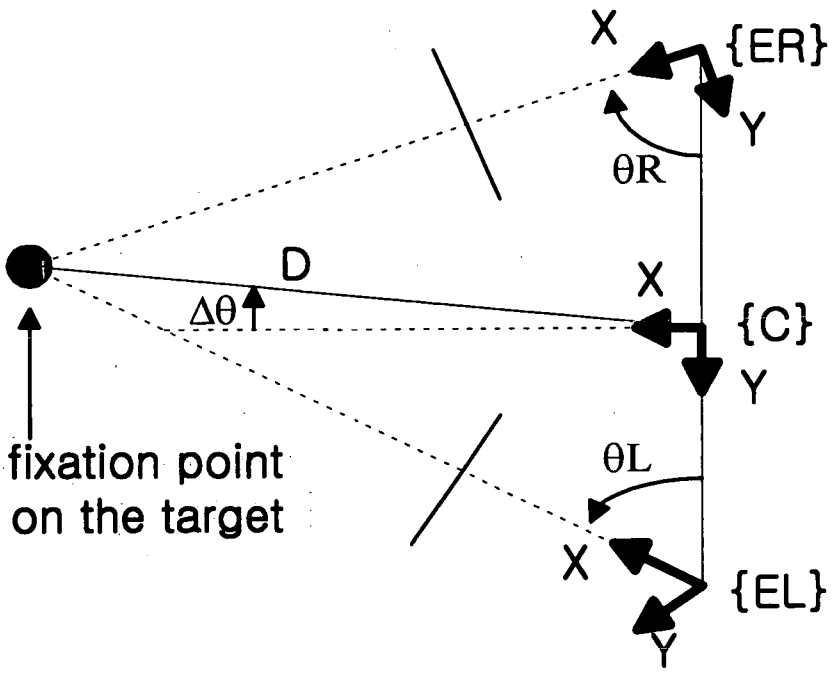
둘째 영상 좌표계의 영상 점의 좌표도 마찬가지로 (x_2, y_2)라 하고 물체점의 좌표를 $(u_2,\ v_2,\ w_2)$ 라 하면 $u_2 = \frac{x_2(\lambda - w_2)}{\lambda}$ 이다.

첫 영상면의 x-y좌표계는 둘째 영상면의 x-y좌표계와 평행하지만 x축상으로 렌즈의 거리 B 만큼 떨어져 있으므로

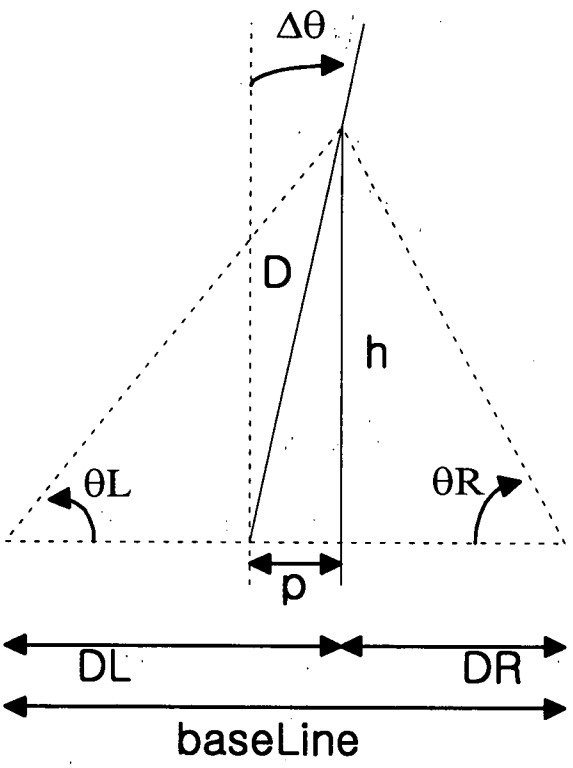
$u_2 = u_1 + B$ 이고 $u_2 = w_1 = w$ 이다.

이를 사용하면 $u_1 = \frac{x_1}{\lambda}(\lambda - w)$ $u_1 + B = \frac{x_2}{\lambda}(\lambda - w)$

을 얻으며, 양변을 각각 감산하여 $w = \lambda - \frac{\lambda B}{x_2 - x_1}$ 을 얻는다. 이것은 물체의 점에 대한 두 영상의 x좌표, x_1, x_2 를 알면 깊이 w를 구할 수 있다.



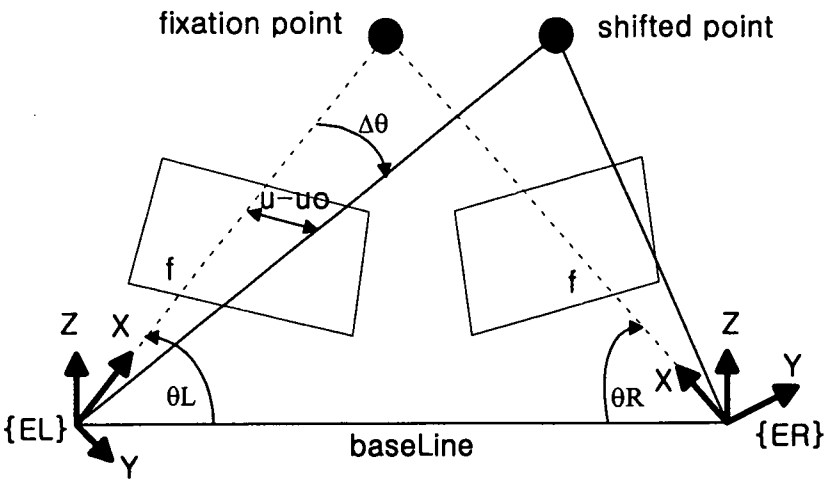
<Fig. 6-5> Angle and distance from point C on XY coordinate



<Fig. 6-6> Parameter

나) 카메라 범위각 제어

이미지에서 대상체의 위치정보는 비젼시스템의 위치와 방향제어에 이루어진다. 공간상에서 대상체의 위치변화는 이미지위치 역시 변화할 것이다. 이것은 두 개의 이미지 중심에서 대상체의 이미지투사를 하는 방법으로 제어하면 대상체의 거리를 구한다.



<Fig. 6-7> Control of camera angle

이미지 중심에서 대상체의 투사를 유지하는데에는 카메라각 θL and θR 이 변화할 것이다. 대상체의 변화시 수평적 부등(horizontal disparity) $\triangle u = (u - u0)$ 이며 x축의 기준점의 픽셀좌표를 u라 하면 각각의 카메라 각은 다음과 같이 변화할 것이다.

$$\triangle \theta = \arctan \frac{u - u0}{S_x f}$$

여기서 f는 초점길이(focus length)이고 S_x, S_y 는 카메라의 본질적 매개변수라 불리우는 비례인자(scale factor)이다. 물체의 추적위치는 그림에서 보여주고 있는 기준점에 따른 고정점(두 개의 매개변수)을 이용해서 {C}기준점에 대해 거리 D와 각 $\triangle \theta$에 의해 알수 있다. 보조매개변수에 의한 관계에서 D와 $\triangle \theta$값의 방정식은 다음과 같이 얻을 수 있다.

$$h = \tan(\theta R)\ D_r$$

$$h = \tan(\theta L)\ D_r$$

$$B = DL + DR$$

$$P = DL - \frac{B}{2}$$

{C}기준에 관련한 고정점의 거리 D와 $\triangle\theta$ 는 다음의 방정식에 의해 얻을 수 있다.

$$\triangle\theta = 90^0 - \arctan(\frac{h}{p})$$

$$D = \sqrt{h^2 + p^2}$$

만약 θL 이 θR과 같다면, 위의 관계들은 정확하지 않다. 이러한 경우에 $\triangle\theta$는 0이고, 거리 D는 다음과 같다.

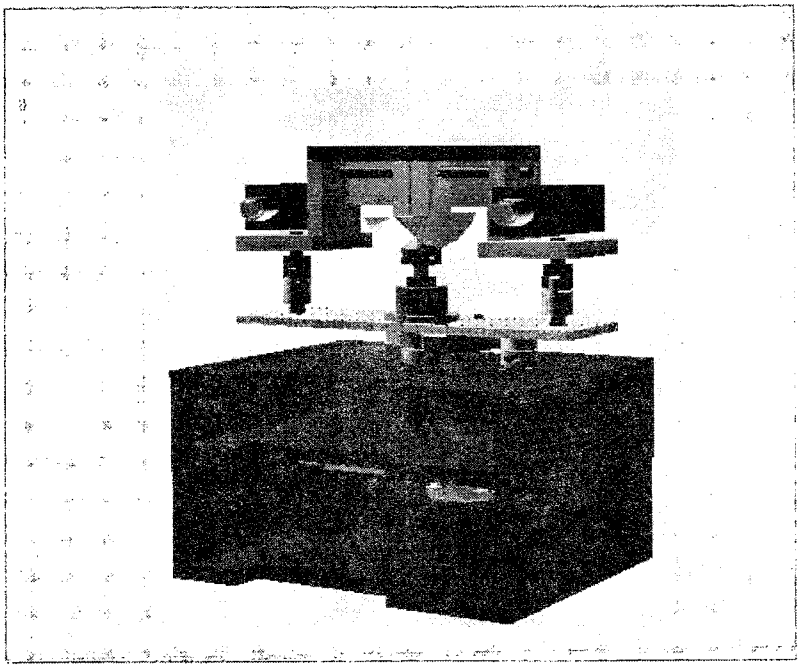
$$D = \frac{B}{2}\ \tan(\theta L)$$

위의 서술된 수학식 표현에서, 두 카메라가 고정점에서 이동점으로의 움직임은 스테레오 방법에 의해 유도된 수학식에서 θL과 θR의 각을 가지고 대상체의 두 이미지에서 거리의 변화 D는 다음과 같이 접근된다.

$$D = \frac{B}{2}\ \tan(\theta L)$$

위의 <Fig. 6-2>에서 Motor1은 카메라의 범위각제어(Camera Vergence angle control)역활을 하며 Motor2는 카메라의 Head Tilt, Motor3는 카메라의 Neck Pan 역할을 한다. 각각의 모터의 제어는 one-chip controller에 의해 제

어되도록 설계되어 있다. 또한 두 카메라간의 거리는 135mm로 설계하여, 일정거리 만큼 떨어진 두 카메라 영상의 인식영역에 있어서 두 카메라간 거리가 멀때보다 100~150mm사이에서 넓은 인식영역을 나타냈다. 본 연구의 BaseLine은 135mm이며 3개의 스테핑모터를 이용하여 회전가능하게 설계하였다.



<Fig. 6-8> Stereo system

나. 오이 영상처리

본 3차원 영상처리장치는 카메라의 PAN TILT 동작 이외에도 두 대 카메라의 포커스도 조절이 가능해 영상처리시 대상물체에 대한 DEEP IMFORMATION 추출은 물론 추출된 이미지에서 OBJECT외 배경의 삭제도 용이하도록 하였다. 제어장치 및 PIC 마이컴 제어용 프로그램 및 메인컴퓨터와의 통신프로그램도 완성되었으며, 앞으로 추출된 영상정보를 신속하게 처리할 수 있는 알고리즘을 개발이 필요할 것으로 판단된다.

1) 전처리 알고리즘

영상처리 알고리즘에서 전처리는 왜곡된 영상에서 보다 나은 영상데이타를 얻고자 하는데 있다. 전처리는 영상의 질을 개선하거나 영상을 특정한 응용목적에 알맞도록 변환시키는 등의 영상처리를 의미한다. 2대의 카메라에 의한 스테레오방법의 대상체 인식정보를 추출하기 위해 선행되어야 할 전처리 알고리즘으로서는 대상체와 배경을 분리함으로써 대상체에 대한 많은 정보를 얻기 위한 이치화 과정을 하고 경계검출 알고리즘을 사용할 것이다. 또한 두 카메라에 의한 스테레오 방법으로 대상체의 이미지를 그랩하여 이미지를 처리하기 위해, 배경으로부터 대상체 정보를 추출하는 알고리즘을 구현할 것이다 . 이는 두 대의 카메라로 추출하고자 하는 대상체를 그랩(Grab)하면 대상체 외의 배경의 이미지는 화소의 위치변화를 대상체보다 크게 나타난다. 두 대의 카메라에 의해 이미지의 대상체에서 큰 화소변화의 배경은 없애고 추출하고자 하는 특징점만을 형상정보로 얻기 위한 알고리즘을 구현하고자 한다.

가) 영상분할을 위한 이치화 방법

이치화를 보통 Binaruzation 또는 Thresholding 이라고 하며 크게 적용영

역에 따라 두가지로 구분하면 Global thresholding 와 Local thresholding으로 나눌수 있다. 전자는 처리 범위가 전체 측정윈도우에 해당하며 후자는 부분적인 측정윈도우를 지정함으로써 수행된다. 특히 특정영상을 분석하고 측정하고자 할때는 대상물체에 따라 효율적으로 물체를 분리할 수 있는 국부이치화 방법에 대한 연구가 중요하다.

입력영상을 f(x,y), 출력영상을 g(x,y) 라고 정의하면 화소의 좌표 x, y에 대한 이치화 기준은 경계값 T보다 클 경우를 1로 나타내고 반대의 경우는 0으로 정하게 된다. 여기서 0은 물체(또는 배경)을 1은 배경(또는 물체)을 나타낸다.

$$g(x,y) = \begin{cases} 1 & : \ f(x,y) \geq T \\ 0 & : \ f(x,y) < T \end{cases}$$

여기서 T는 Brightness level 로 배경부분과 추출할 패턴을 분리하는 이 변수를 Threshold value 라고 하는데 이 변수를 어떻게 화상의 히스토그램 분석에 활용하여 결정하는 것이 문제로 되어 있다. 이 변수는 Manual 조작에 의해 임의로 설정하여 줄 수도 있고 또는 2개의 Windows설정에 의해 경계치를 결정하는 Windows Thresholding Method 알고리즘과 가상의 모집단 분류에 의해 2모집단의 분산값을 최대로 하는 통계적 판별분석법을 이용하여 Threshold Value를 정할 수 있다. 본 연구에서는 정적 이미지를 처리하도록 bitmap함수로 구현하였고 동적 이미지에서는 그레버보드(Grabber Board)의 FrameBuffer 함수를 직접 이용하여 이미지 처리가 가능하게 이치화를 하도록 구현하였다.

나) 에지 중심(edge based)분할 방법

에지(Edge)는 서로 다른 명암도를 갖는 영역간의 경계에 위치하는 점이

다. 그러므로 영상함수 f(x)에 대해, 에지에서의 기울기 df/dx 는 명암이 균일한 지역에서의 기울기와 현저히 비교된다. 그러므로, 도함수(Derivative)를 계산하고 임계치(Threshold)를 사용하여 에지점(edge point)들을 검출할 수 있다. 그러나 잡음(noise)이 존재하는 경우에, 이 에지검출자는 많은 오인된 에지들을 검출하게 되는데 좀 더 나은 결과를 위해 화소단위로 에지크기를 측정하는 대신에 영역 단위로 에지크기를 측정하는 방법을 사용한다.

본 전처리 과정은 형상정보를 얻기 위해 이치화와 경계검출과정을 실행함으로써 좀더 물체정보를 정확히 검출하는데 목적이 있다. 특히 서로 다른 명암도를 갖는 영역간의 경계를 이치화 및 경계검출 알고리즘을 사용하면 더 나은 형상정보추출을 얻을 수 있다.

에지는 두 영역(region)의 경계에 위치하는 점들을 말하며, 영역간의 경계부분은 한 영상안에서 명암도의 불연속으로 나타난다. 이는 두 영역의 경계(boundary)에 의해 대상을 표현하기 위한 방법이다. 주로 영상에 있는 명암값(gray-level value)들로부터 직접 대상들의 경계를 찾아내기 위해 중간단계의 에지(edge)영상으로 변환(transform)하고나서 더 정확한 경계 영상을 구성하는 방법으로, 명암을 가지고 에지들을 찾아내는 것이 일반적인 방법이다.

에지 연산자들의 공통적인 특징은 연산자들이 최대 명암 변화의 방향(direction)과 이 변화의 정도를 나타내는 크기(magnitude)를 계산하는 것이다. 미분 연산자의 에지는 서로 다른 명암도를 갖는 영역간의 경계에 위치하는 점이다. 도함수를 계산하고 임계치(Threshold)를 사용하여 에지점들을 검출하게 되는데 좀 더 나은 결과를 위해 화소단위로 에지크기를 측정하는 대신에 영역 단위로 에지크기를 측정하는 방법을 사용한다. 또한 라플라시안은 선의 끝과 모퉁이에 특히 민감한 반응을 보이는데 경사진 곳의 견각(shoulder)에서 민감한 반응을 보이고, 그 곳에서 영점교차(zero-crossing)를 야기하며, 영점교차선의 경사도로서 에지크기를 나타낸다. Sobel 연산자는 해당 에지주변의 점을 에지로 검출하는 성질을 갖는다. 이러한 성질은 에지의 확대 및 축소를 야기시키며 Template Matching 알고리즘도 다소 유사한 특성을 갖는다. 또한

노이즈에 대해 상대적으로 강한 면을 볼 수 있다.

경계검출 방법은 우선 여러방향의 에지 마스크를 씌어 얻은 각 방향 에지 값들을 제곱의 합의 제곱근값이나 계산상 수월한 각 방향 에지 절대값의 합 또는 절대값의 최대값 등의 비선형 계산을 거친 후 임계값보다 큰지 작은지에 따라 그 점이 에지인지 아닌지를 나타내도록 2치(binary value)하여 검출되도록 한다. 대표적인 에지 연산자로는 Sobel 연산자, Laplacian 연산자, Roberts 연산자, Prewitt 연산자 등이 있으며 초기의 이미지를 Sobel, Laplacian, Roberts의 경계 추출알고리즘을 이용하여 구현해 보았다. 그림에서 보듯이 Sobel edge 알고리즘이 임계치 150에서 효율적으로 검출됨을 보여준다.

2) 칼라영상처리

수확기의 영상입력장치가 일반 산업용 로봇과 차이점은 대상체 개개의 형상, 크기, 색이 각기 다르고, 3차원적으로 불규칙하게 분포한다는 점, 자연 상태의 외부환경하에서 작업하기 때문에 조명조건(태양광의 직사, 그늘, 역광 등)이 불균일하고 가변적인 점 등을 들 수 있다.

대상물을 배경으로부터 분리해서 식별하고 고속으로 처리하기 위해 영상의 2 치화를 많이 사용한다. 흑백 카메라로 영상을 입력하는 경우, 2 치화 처리는 설정한 문턱값 밝기보다 더 밝은가 아닌가로 수행한다. 농업용 로봇을 이용한 작업의 경우, 실내에서 이미 거의 위치가 정해져 있는 묘에 대한 형상계측 등에서는 전술한 2치화 처리 방법도 가능하리라 생각된다. 그러나 야외 작업용의 수확용 로봇에서는 전술한 것처럼 조명 상태가 불규칙하고 가변적이어서 이 방법으로는 대상물의 식별이 곤란할 경우가 많다. 이와 같은 조건에서는 색정보를 이용한 식별이 유효하다. 특히 토마토, 귤 등은 수확시기에 배경을 이루는 줄기, 잎과는 명확한 색상의 차이가 있다.

컬러 영상신호는 적(R), 녹(G), 청(B)신호와 동기신호로 구성되어 있다. 이 컬러 영상신호로부터 대상물의 색 차이를 이용하여 작물의 2치영상을 얻는 방

법으로서, 색신호끼리 직접 비교하거나 RGB색신호를 이용하여 연산한 값을 문턱값과 비교하는 등의 방법이 이용되고 있다.

RGB색 신호로부터 얻어 사용하는 변수량에는 휘도(Y), 색상, 채도, 색차(R-Y, B-Y) 그리고 RGB 3원색 중 R값이 차지하는 비율 등이 있다. 그리고 휘도 Y는

$$Y = 0.30R + 0.59G + 0.11B$$

의 식으로 구해진다. 색상은 적, 청, 황의 색조를 나타낸다. 채도는 포화도라고도 부르며, 색의 선명한 정도를 나타낸다. 예를 들면 같은 청색에서도 하야스름한 청색과 선명한 청색이 있는데 선명한 청색을 채도가 크다고 한다. 연산·비교의 방법에는, 컬러 TV카메라로부터 입력되는 아날로그 영상신호를 연산증폭기를 통한 후 비교기로 비교하는 방법과 컬러 영상 입력장치를 이용하여 RGB 각 신호를 각기 A-D변환하여 영상메모리에 입력해서 컴퓨터나 영상처리 프로세서로 행하는 방법이 있다.

이외에 컬러 영상신호로부터 디코더를 사용하여 색상과 채도를 취출하고, 색상과 채도를 취출하고, 색상과 채도를 비교기로써 문턱값과 비교하여 오렌지의 2치영상을 입력하는 방법도 있다. RGB 3원색을 각각 A-D변환하고, 영상메모리에 입력하여 디지털 영상처리를 하는 방법도 있다. 작물의 식별방법으로서는, 주로 다음의 방법들이 이용되고 있다.

(1) 3원색 중 2색의 차, 예를 들면 R-G를 문턱값과 비교

(2) 영상신호 중에 차지하고 있는 특정 색신호의 비율, 예를 들면 R/(R+G+B)를 문턱값과 비교

(3) 색상, 채도를 연산하여 구하고, 문턱값과 비교

(4) NTSC방식의 TV방송에서 사용하고 있는 클로미넌스(Q,I)와 색차신호(R-Y, B-Y)를 연산으로 구하고, 문턱값과 비교

이와 같은 A-D변환된 RGB신호를 이용하여 디지털적으로 식별하는 방법은 하드웨어로써 하는 방식이 많은데, 대상물과 배경의 색에 다수 존재하거나 겹

쳐 보이는 것, 예를 들면 과수의 잎이나 솎아내기 전의 묘를 대상으로 하여 개개의 잎이나 묘를 식별하기 위해서는, 이것만으로는 불충분하여 고도의 영상처리 알고리즘이 필요하다. 이 경우, 2치화하고나서 메모리로 입력하기보다는 RGB 3원색 모두 A-D변환하는 편이 정보량이 많아 인식 가능성이 크다. 옥외작업의 경우 식물체는 다양하고 가변적인 자연환경하에서 복잡한 형상, 색, 그리고 크기 등을 가진다. 따라서 잎이나 가지 등과 같은 계통의 색을 띤 과실을 식별할 경우에는 색신호를 이용하는 방법만으로는 충분하다고 할 수 있다. 그러므로 일반적인 영상입력센서가 감도를 갖는 가시광영역에서부터 근적외영역까지의 분광반사특성을 조사하여 식별에 적합한 대상물 특유의 파장대역을 여러 개 취하여 식별하는 방법이 유효하다.

수확용 로봇의 경우 영상입력부 기능으로서 대상물 각 부위의 분광반사특성에 기초한 식별기능, 즉 잎의 형상, 생장상태, 영양상태, 종류, 줄기, 가지의 방향 그리고 병의 유무 등을 식별하는 것이 필요한 경우가 많다. 이들 정보의 대부분은 잎에 있고, 그 형상, 방향 등에 관한 정보를 여러 종류의 방법으로 간단히 수치화해서 기술할 수 있으면 인식능력은 커진다. 또한, 영상인식에 있어서는 3차원 공간의 대상체가 3차원으로 투영되어 나타나는 영상정보로부터 어떻게 거리에 관한 정보를 얻어 대상을 이해하는 데 결부시킬까 하는 것이 중요한 과제의 하나이다. 이때, 대상의 형상에 관한 특징을 미리 알고 있으면, 완전한 거리정보를 얻을 수 없는 경우도 그 형상의 특징과 규칙성을 이용한 인식이 가능하게 된다. 영상처리 및 인식의 주사순서에 있어서도 영상 전체를 거의 화소의 배열순으로 주사하는 것이 아니라, 대상물의 형상 특징으로부터 영상 중에서 대상이 존재하는 범위를 미리 추측해서 주사할 수 있으면, 불필요한 주사를 감소시키는 동시에 처리시간을 단축할 수 있으며 인식률도 높일 수 있다.

식물체의 잎이나 꽃 등의 배열은 규칙성을 가지는 경우가 많아서, 이것을 미리 지식 베이스로써 저장해 두면 신속하고 정확한 인식을 할 수 있다. 예를 들면 오이는 거의 2열 호생엽서(互生葉序)에 가깝고, 대상물고 시점의 상대적

인 위치관계, 시점의 방향, 카메라의 시야각 등을 알고 있으면, 대상물의 영상을 대략 추측할 수 있다. 반대로, 이 잎차례의 규칙성을 이용하면 영상으로부터 대상물과 시점의 위치관계 및 시점의 위치관계 및 시점의 방향 등의 추측도 어느 정도는 가능하다. 이와 같이 투영모델은 시점의 이동 및 전장에 따라 변화한다. 이 모델과 영상을 대응시킴으로써, 대상물과 시점의 위치관계 추정 및 줄기와 잎을 주사하는 알고리즘을 작성할 수 있다.

작물은 이랑 등을 따라 다수의 열로 재배되는 경우가 많으므로 우선, 대상이 되는 작물과 그 이외의 것을 식별할 필요가 있다. 이때 작물 이외의 토양, 배경 등의 식별에는 주로 분광반사특성을 이용하고, 작물끼리의 식별에는 거리정보를 이용하는 것이 적당하다고 생각된다. 다음에 영상 데이터를 2치화 혹은 다치화 처리를 하여 모멘트, 페레장비 등을 이용하여 대상의 주줄기, 선단, 하단 등의 검출 및 크기의 계측을 행한다. 그리고 투영 모델, 페레장비, 복잡도 등을 이용하여 시점과 대상물의 위치관계 및 방향을 추정한 후, 추정된 방향의 투영모델을 기초로 각 줄기와 잎을 주사하여 위치, 방향 등을 검출한다.

제 4 절 결과 및 고찰

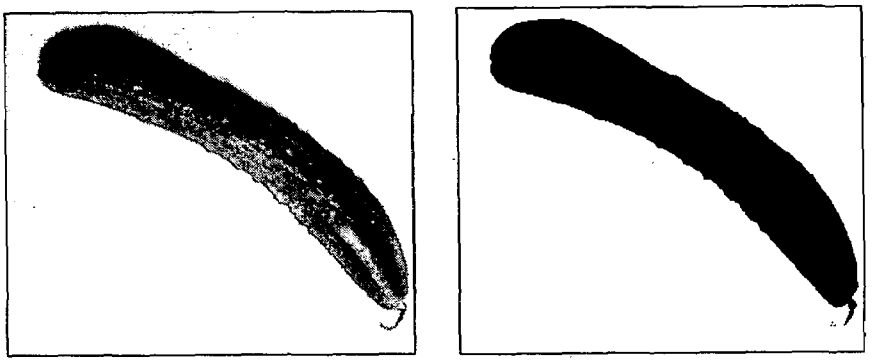
본 3차원 영상처리장치는 카메라의 PAN TILT 동작 이외에도 <Fig. 6-9>에서 보여지는 것과 같이 두 대 카메라의 포커스도 조절이 가능해 영상처리시 대상물체에 대한 DEEP IMFORMATION 추출은 물론 추출된 이미지에서 OBJECT외 배경의 삭제도 용이하도록 하였다. 제어장치 및 PIC 마이컴 제어용 프로그램 및 메인컴퓨터와의 통신프로그램도 완성되었으며, 앞으로 추출된 영상정보를 신속하게 처리 할 수 있는 알고리즘을 개발이 필요할 것으로 판단된다.

1. 전처리 알고리즘

영상처리 알고리즘에서 전처리는 왜곡된 영상에서 보다 나은 영상데이타를 얻고자 하는데 있다. 전처리는 영상의 질을 개선하거나 영상을 특정한 응용목적에 알맞도록 변환시키는 등의 영상처리를 의미한다. 2대의 카메라에 의한 스테레오방법의 대상체 인식정보를 추출하기 위해 선행되어야 할 전처리 알고리즘으로서는 대상체와 배경을 분리함으로써 대상체에 대한 많은 정보를 얻기 위한 이치화 과정을 하고 경계검출 알고리즘을 사용할 것이다. 또한 두 카메라에 의한 스테레오 방법으로 대상체의 이미지를 그랩하여 이미지를 처리하기 위해, 배경으로부터 대상체 정보를 추출하는 알고리즘을 구현할 것이다 . 이는 두 대의 카메라로 추출하고자 하는 대상체를 그랩(Grab)하면 대상체 외의 배경의 이미지는 화소의 위치변화를 대상체보다 크게 나타난다. 두 대의 카메라에 의해 이미지의 대상체에서 큰 화소변화의 배경은 없애고 추출하고자 하는 특징점만을 형상정보로 얻기 위한 알고리즘을 구현하고자 한다.

가. 영상분할을 위한 이치화 방법

이치화를 보통 Binaruzation 또는 Thresholding 이라고 하며 크게 적용영역에 따라 두가지로 구분하면 Global thresholding 와 Local thresholding으로 나눌수 있다. 전자는 처리 범위가 전체 측정윈도우에 해당하며 후자는 부분적인 측정윈도우를 지정함으로써 수행된다. 특히 특정영상을 분석하고 측정하고자 할때는 대상물체에 따라 효율적으로 물체를 분리할 수 있는 국부이치화 방법에 대한 연구가 중요하다.



(a) Original Image (b) Threshold image

<Fig. 6-9> Threshold image

입력영상을 f(x,y), 출력영상을 g(x,y) 라고 정의하면 화소의 좌표 x, y에 대한 이치화 기준은 경계값 T보다 클 경우를 1로 나타내고 반대의 경우는 0으로 정하게 된다. 여기서 0은 물체(또는 배경)을 1은 배경(또는 물체)을 나타낸다.

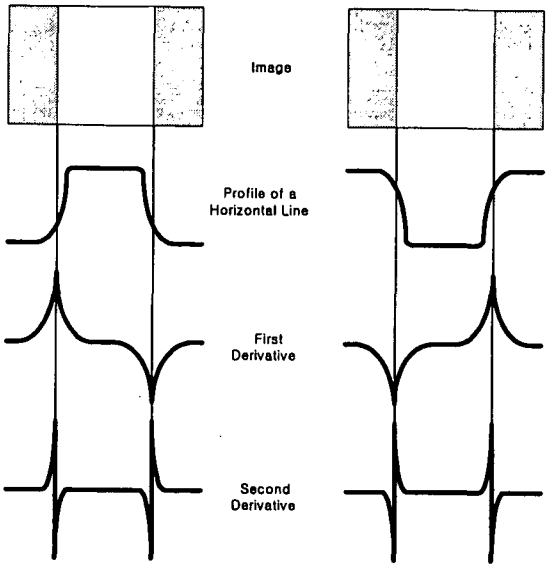
$$g(x,y) = \begin{cases} 1 & : \ f(x,y) \geq T \\ 0 & : \ f(x,y) < T \end{cases}$$

여기서 T는 Brightness level 로 배경부분과 추출할 패턴을 분리하는 이 변수를 Threshold value 라고 하는데 이 변수를 어떻게 화상의 히스토그램 분석에 활용하여 결정하는 것이 문제로 되어 있다. 이 변수는 Manual 조작에 의해 임의로 설정하여 줄 수도 있고 또는 2개의 Windows설정에 의해 경계치를 결정하는 Windows Thresholding Method 알고리즘과 가상의 모집단 분류에 의해 2모집단의 분산값을 최대로 하는 통계적 판별분석법을 이용하여 Threshold Value를 정할 수 있다. 본 연구에서는 정적 이미지를 처리하도록 bitmap함수로 구현하였고 동적 이미지에서는 그레버보드(Grabber Board)의 FrameBuffer 함수를 직접 이용하여 이미지 처리가 가능하게 이치화를 하도록 구현하였다.

나. 에지 중심(edge based)분할 방법

에지(Edge)는 서로 다른 명암도를 갖는 영역간의 경계에 위치하는 점이다. 그러므로 영상함수 f(x)에 대해, 에지에서의 기울기 df/dx 는 명암이 균일한 지역에서의 기울기와 현저히 비교된다. 그러므로, 도함수(Derivative)를 계산하고 임계치(Threshold)를 사용하여 에지점(edge point)들을 검출할 수 있다. 그러나 잡음(noise)이 존재하는 경우에, 이 에지검출자는 많은 오인된 에지들을 검출하게 되는데 좀 더 나은 결과를 위해 화소단위로 에지크기를 측정하는 대신에 영역 단위로 에지크기를 측정하는 방법을 사용한다.

본 전처리 과정은 형상정보를 얻기 위해 이치화의 경계검출과정을 실행함으로써 좀더 물체정보를 정확히 검출하는데 목적이 있다. 특히 서로 다른 명암도를 갖는 영역간의 경계를 이치화 및 경계검출 알고리즘을 사용하면 더 나은 형상정보추출을 얻을 수 있다.

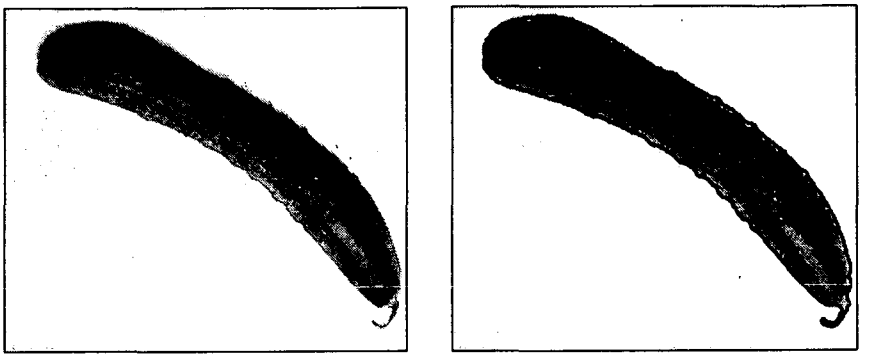


<Fig. 6-10> Edge Detection By Derivative Operators

에지는 두 영역(region)의 경계에 위치하는 점들을 말하며, 영역간의 경계 부분은 한 영상안에서 명암도의 불연속으로 나타난다. 이는 두 영역의 경계(boundary)에 의해 대상을 표현하기 위한 방법이다. 주로 영상에 있는 명암값(gray-level value)들로부터 직접 대상들의 경계를 찾아내기 위해 중간단계의 에지(edge)영상으로 변환(transform)하고나서 더 정확한 경계 영상을 구성하는 방법으로, 명암을 가지고 에지들을 찾아내는 것이 일반적인 방법이다.

에지 연산자들의 공통적인 특징은 연산자들이 최대 명암 변화의 방향(direction)과 이 변화의 정도를 나타내는 크기(magnitude)를 계산하는 것이다. 미분 연산자의 에지는 서로 다른 명암도를 갖는 영역간의 경계에 위치하는 점이다. 도함수를 계산하고 임계치(Threshold)를 사용하여 에지점들을 검출하게 되는데 좀 더 나은 결과를 위해 화소단위로 에지크기를 측정하는 대신에 영역 단위로 에지크기를 측정하는 방법을 사용한다. 또한 라플라시안은 선의 끝과 모퉁이에 특히 민감한 반응을 보이는데 경사진 곳의 견각(shoulder)에서 민감한 반응을 보이고, 그 곳에서 영점교차(zero-crossing)를 야기하며, 영점교

차선의 경사도로서 에지크기를 나타낸다. Sobel 연산자는 해당 에지주변의 점을 에지로 검출하는 성질을 갖는다. 이러한 성질은 에지의 확대 및 축소를 야기시키며 Template Matching 알고리즘도 다소 유사한 특성을 갖는다. 또한 노이즈에 대해 상대적으로 강한 면을 볼 수 있다. 아래의 그림은 배경과 물체가 복잡하지 않을때의 이미지이다.



(a) Original image (b) Edge detection image

<Fig. 6-11> Edge detection image

경계검출 방법은 우선 여러방향의 에지 마스크를 씌어 얻은 각 방향 에지 값들을 제곱의 합의 제곱근값이나 계산상 수월한 각 방향 에지 절대값의 합 또는 절대값의 최대값 등의 비선형 계산을 거친 후 임계값보다 큰지 작은지에 따라 그 점이 에지인지 아닌지를 나타내도록 2치(binary value)하여 검출되도록 한다. 대표적인 에지 연산자로는 Sobel 연산자, Laplacian 연산자, Roberts 연산자, Prewitt 연산자 등이 있으며 초기의 이미지를 Sobel, Laplacian, Roberts의 경계 추출알고리즘을 이용하여 구현해 보았다. 그림에서 보듯이 Sobel edge 알고리즘이 임계치 150에서 효율적으로 검출됨을 보여준다.

(a) Original Image

(b) Sobel edge detection: threshold value 150

(c) Laplacian edge detection: thrshold value 150

(d) Roberts edge detection: threshold value 150

<Fig. 6-12> Edge detection field experiment

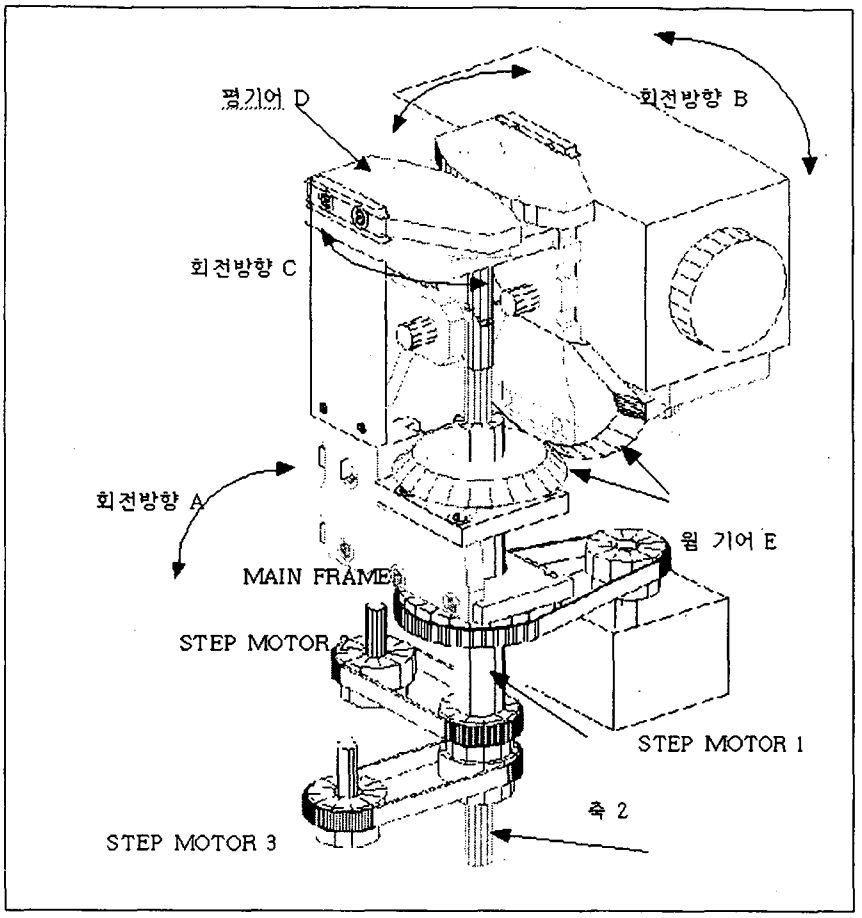
2. 영상처리시스템 제어장치

오이의 형상 및 위치를 검출하기 위한 영상처리시스템은 위치가 다른 2대의 카메라로부터 영상을 입력받아야 한다. 두 대 카메라 사이의 거리가 고정적이어야 하고 pan 및 tilt angle의 정확도가 중요하다. 본 시스템의 특징은 스테핑모터를 구동원으로 사용하여 구동드라이버를 간략화 시켰다.

가. 영상처리시스템의 장치 및 원리

본 장치의 구성은 스텝 모터 , 스텝모터 구동 드라이버 , 카메라 두 대 , 타이밍 벨트 타이밍 벨트 풀리 , 웜기어 , 원칩 마이컴 및 PC와 RS232 통신이 가능하도록 하는 통신포트 등으로 구성되어 있다.

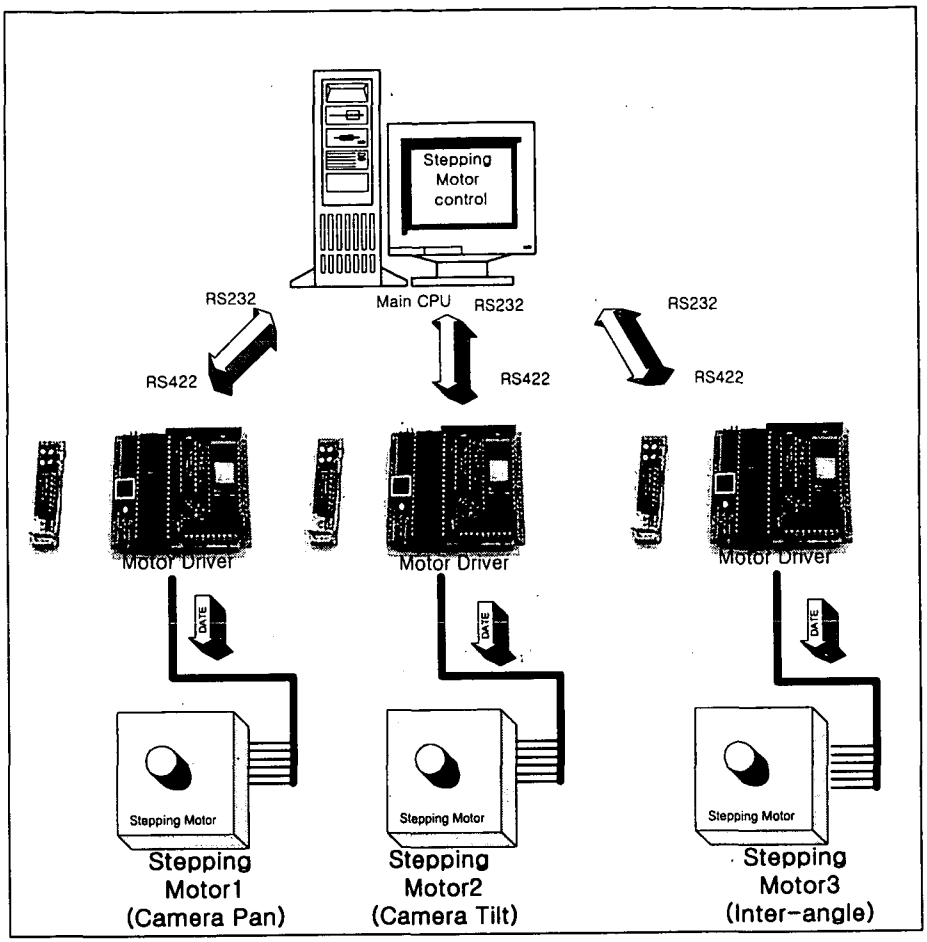
Stepping Motor 1의 축이 회전하면 타이밍벨트를 통해 카메라가 장착된 Main frame에 부착된 풀리를 돌려줌으로서 Main frame이 회전하여 카메라를 Pan 시키게 된다. 그리고 Stepping Motor 2의 축이 회전하면 풀리를 통해 축 2를 회전시키게 되고, 상단부에 고정된 웜기어 또한 회전하게 되어 Camera frame에 장착된 웜기어를 회전시켜 Camera를 Tilt 시켜준다. 마지막으로 Stepping Motor 3 출력축이 회전하게 되면 축 3이 회전하게 축 3 상단에 연결된 유니버셜 조인트를 거쳐 평기어 D를 회전시키고, 따라서 두 대의 Camera가 회전방향 C로 확대, 축소되면서 각도를 제어 할 수 있다.



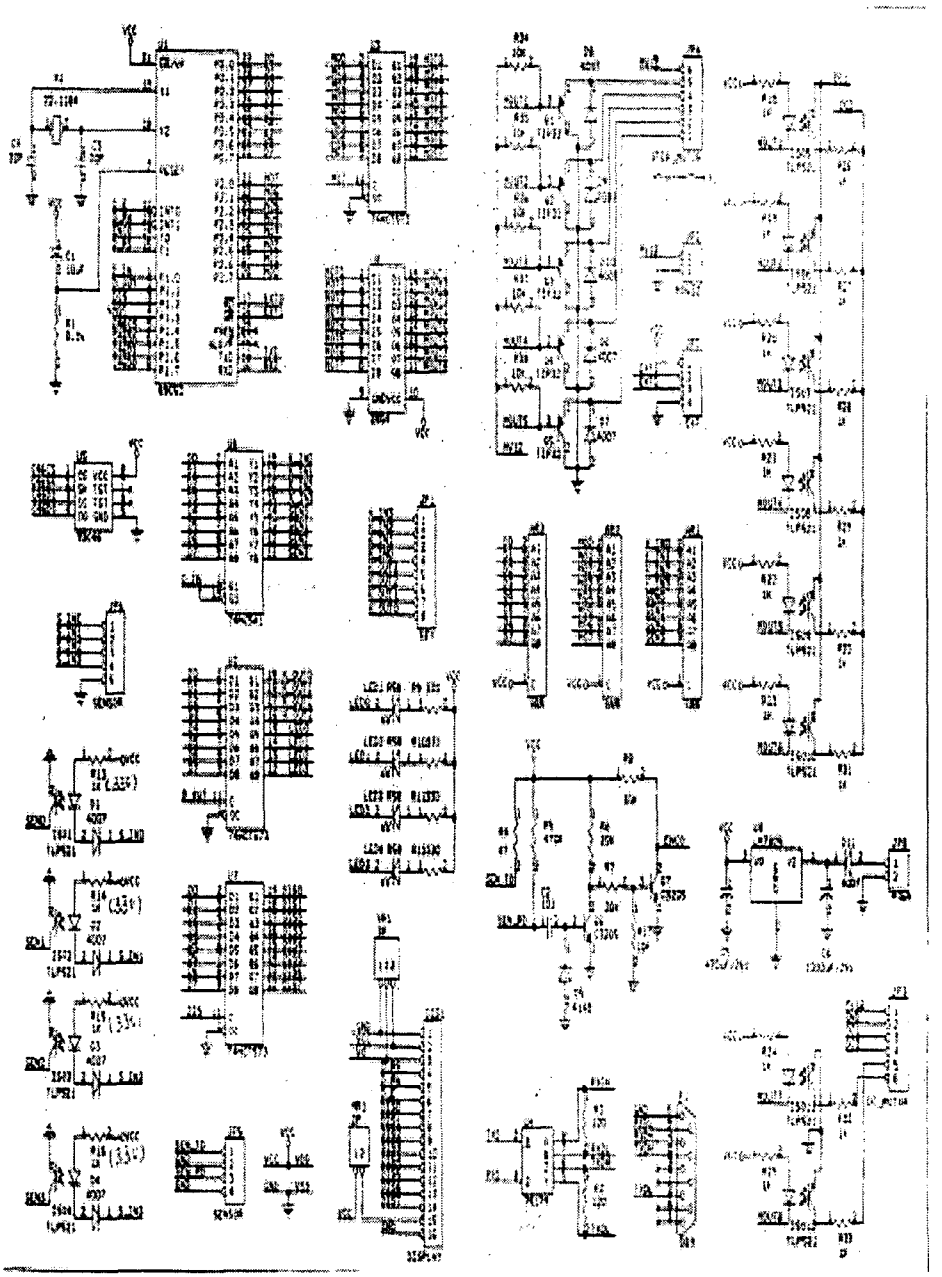
<Fig. 6-13> Stereo vision system

나. 스텝핑모터의 제어장치

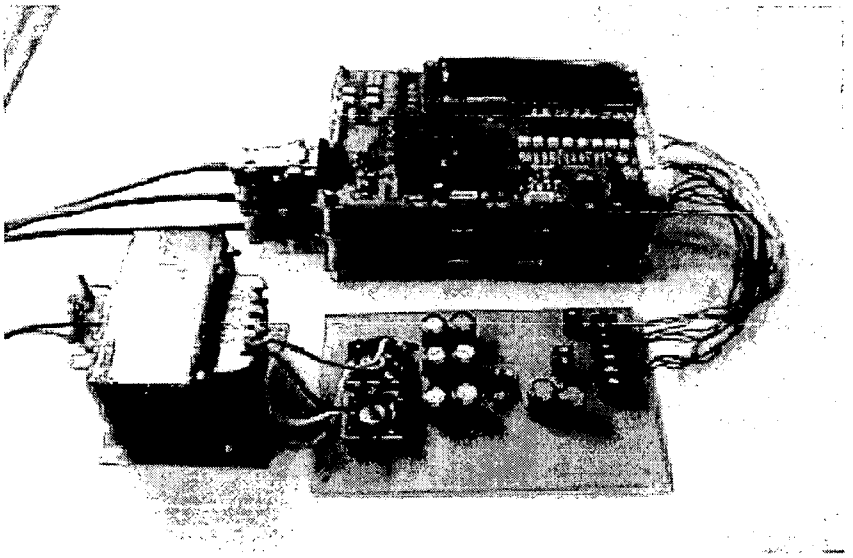
<Fig. 6-14>는 3개의 스텝핑모터의 모터콘트롤 보드와 PC의 연결을 개략적으로 보여주는 그림이다. <Fig. 6-15>은 모터제어를 위한 콘트롤 보드의 회로도를 나타낸 그림이며, <Fig. 6-16>은 제작한 보드 사진이다.



<Fig. 6-14> Three stepping motors



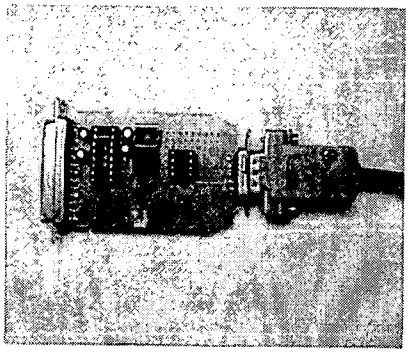
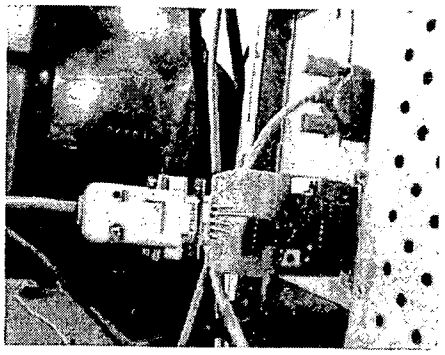
<Fig. 6-15> Motor control board circuit



<Fig. 6-16> Control board for stepping motors

위 시스템은 원칩 마이컴(ATMEL사)의 89C52칩을 이용하였고, PC와의 통신이 가능한 시리얼 포트가 있어 데이터의 전송을 할 수 있게 하였다. 메인 CPU인 89C52는 양방향성 입출력과 2개의 16비트 타이머/카운터, 5개의 인터럽트 등 제어응용에 적합한 8비트 CPU로써 EEP-ROM형태이므로 사용자의 프로그램을 개발에 용이하다.

<Fig. 6-17>는 통신을 위한 컨버터인 RS422로써 MAX232C칩과 75179칩을 사용하여 PC와의 시리얼 통신을 위해 제작된 모습이다. RS232C방식에서 내노이즈성과 전송능력을 개선하기 위해 차동증폭기를 사용한 RS422방식을 채택하였으며 멀티통신이 가능하도록 하였다. 또한 접촉센서인 리미트 스위치와 모터에 부착된 엔코더를 사용하여 위치제어와 속도제어가 용이하도록 설계되었으며, 현재 작동상태를 표시할 수 있도록 16 × 2 라인의 LCD를 부착하였다.

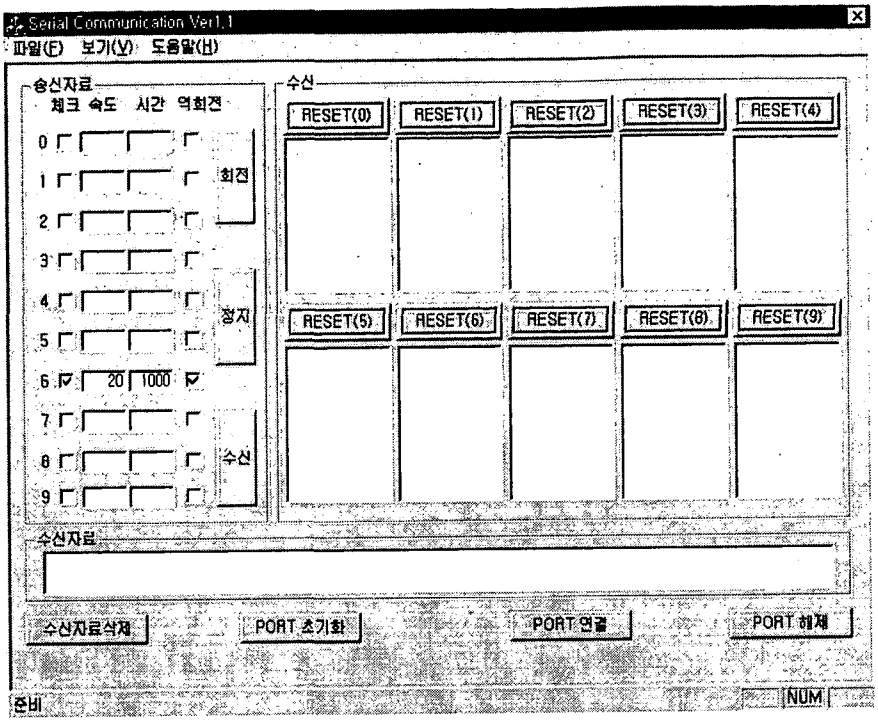



<Fig. 6-17> RS422 convertor

<Fig. 6-18>은 콘트롤 보드로 데이터를 전송하는 프로그램으로 Visual C++을 이용하여 제작하였다. 본 프로그램의 목적은 ROM의 번지와 속도, 거리(작동시간), 회전방향의 데이터를 RS422를 통해 콘트롤 보드로 전송해주는 것이다. 콘트롤 보드로 전송되는 통신프로토콜은 총 12 byte로 모두 7가지이며 <Table 6-4>와 같다.

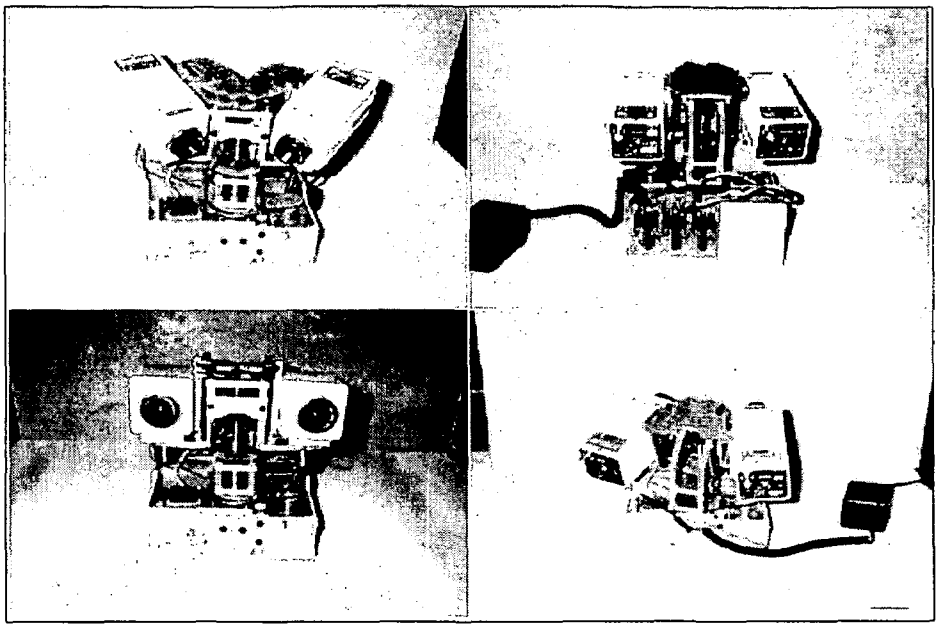
<Table 6-4> Trans-protocol

순서	PROTOCOL	SIZE
1	Start command	1byte
2	Address	1byte
3	Command	1byte
4	Speed Data	3byte
5	How long time	4byte
6	Up or Down	1byte
7	End command	1byte



<Fig. 6-18> Basic screen for control board

<Fig. 6-19>은 완성된 3차원 영상처리장치의 개발된 제어시스템으로 스텝 핑모터를 구동할 경우의 카메라 모양을 나타낸 그림이다.



<Fig. 6-19> 3-D image processing system

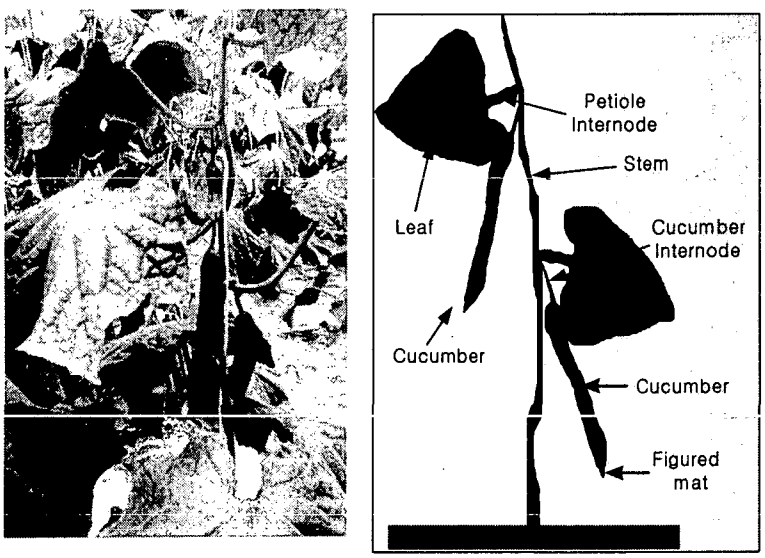
3. 오이 인식 영상처리 알고리즘

가. 실험 장치

영상은 칼라 CCD카메라로 부터 입력시키고, 영상신호는 RS-170 형태의 아날로그 신호로서 프레임 그래버(Frame grabber)에 의해 컴퓨터에 변환되어 입력한다. 본 연구에서 사용된 컴퓨터 사양은 Pentium Pro 200Mhz, RAM 64M, Video RAM 4M이다.

나. 실험 재료

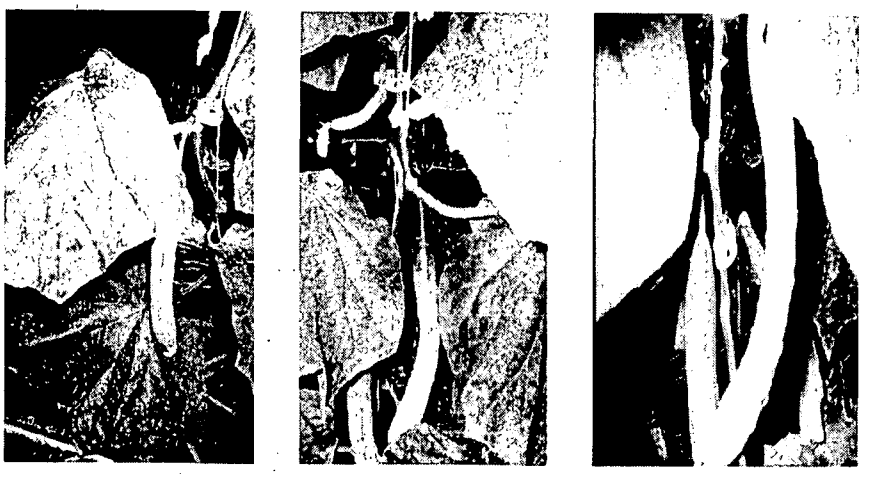
실험에 사용된 오이영상은 <Fig. 6-20>(a)와 같이 오이와 잎이 혼합되어 있으므로 정확한 오이의 형상이 나타나는 것은 쉽지 않고, 배경과 오이에 대한 색상차이 및 특징인자를 구별하기가 쉽지 않다.



(a) Shape (b) Structure

<Fig. 6-20> Cucumber in field

또한, 잎과 원 줄기사이의 마디를 잎꼭지(Petiole internode)로 표기하였다 <Fig. 6-20>의 영상들은 오이의 형상정보를 인식하기 위한 실험재료이다 <Fig. 6-21>에서 보는 바와 같이 실제영상은 대부분 잎과 줄기에 의해 오이를 구별하기가 용이하지 않다.



(a) (b) (c)

<Fig. 6-21> Variety of field cucumber

다. 영상인식 방법

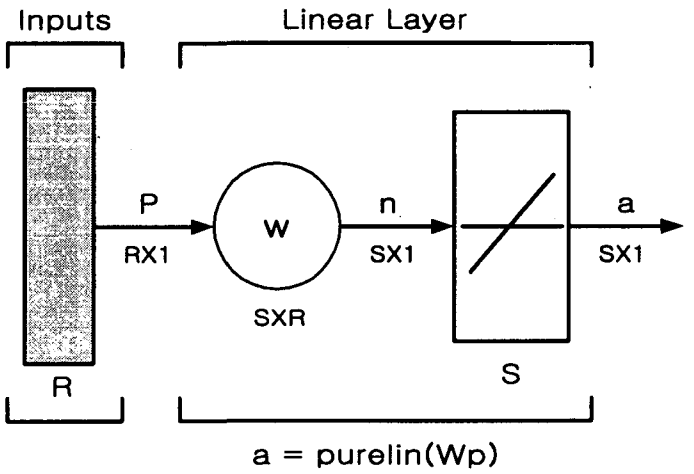
영상처리 과정에서 중요시되는 점은 영상에서 특징을 추출하는 과정이다. 이때 고려되는 특징 추출의 주요인자는 빛의 밝기와 색, 대상물체의 표면, 질감, 에지 등이 있다. 그러므로 전처리의 목적은 이러한 주요인자를 고려하여, 영상의 질을 개선하거나 특정한 응용 목적에 알맞도록 데이터를 얻는데 있다. 특히, 영상에서의 이치화와 에지 검출은 대상체에 대한 크기나 형태를 인식하는데 도움을 준다. 에지 검출은 근거리에 위치한 두 물체간의 상대적인 좌표를 인식해 내는데 효율적으로 사용되고 있다.

본 연구에서는 대상체(오이)의 인식정보를 추출하기 위해, 전처리 알고리즘

의 특성 중 문턱값(Thresholding)방법과 경계(Edge)검출 알고리즘을 사용하였다.

1) 연상 메모리의 수학적 모델

신경망 모델의 규칙은 가중치의 초기화 및 성능향상을 위한 가중치 변경 등에 의해 결정된다. 또한 몇 개의 입력 노드가 여러 개의 처리 노드에 국소적으로 연결되어 있기 때문에, 한두 개의 노드에 오류가 발생해도 전체적 성능에는 현저하게 영향을 주지 않는 특성이 있다.



<Fig. 6-22> Linear association

본 연구에서는 신경회로망 모델 중 구현이 간단하고 비교적 분류능력이 좋은 연상메모리 중에서 동질연상 메모리를 이용하였다. 연상메모리는 패턴의 일부분이 주어지고 전체 패턴의 모습을 알고자 할 때 사용할 수 있다. 그러므로 불완전한 패턴이나 왜곡된(distorted)패턴이 제시되었을 때, 학습패턴에 대해서 완전한 출력패턴으로 유추할 수 있는 특징을 가지고 있다.

<Fig. 6-22>은 선형연상기의 구조를 간단히 도식화한 그림이다. 패턴간의

연관성을 기억시키기 위한 학습단계는 순방향 신경망(Feed-forward network) 연상메모리 방식을 이용하였다. 여기서, 입력벡터 P와 연결강도 W의 관계에 얻은 출력벡터를 a라 하였고, 그 결과 식은 아래와 같이 얻을 수 있다. 여기서, Q는 기억할 학습패턴 쌍이다.

$$a = W p$$

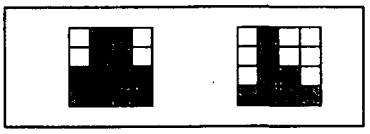
$$a_i = \sum_{j=1}^{Q} W_{ij} p_j$$

연상메모리는 크게 선형연상, 순환연상, 양방향연상 메모리 방식이 있다. 여기서 선형연상메모리는 이질, 동질연상메모리 방식이 있고, 순환연상메모리에는 홉필드모델 방식 등이 있다. 선형과 순환연상은 한 방향으로 연상 작용되는 모델이며, BAM(Bidirectional Associative Memory)은 양방향으로 연상작용이 가능한 연상메모리이다.

2) 인식조건 분석

동질연상 메모리의 수학적 모델은 학습패턴과 출력패턴의 직교치가 0일 때, 대부분 학습패턴으로 연상되지만, 대부분 0이 아닌 경우이다. 그러므로 학습패턴에 대한 출력패턴은 학습패턴으로 연상되는 경우와 학습패턴으로 복원중인 패턴, 국부 최소점 문제(Locial minimum problem)에 빠지는 패턴 등이 출력될 수 있다. 학습패턴에 대한 정확한 출력패턴의 복원방법은 수도역변환 법칙(Pseudo inverse rule)을 이용할 수 있다. 이 방법은 단계적으로 오차(Error)를 줄여나가는 절차로 복원효율을 더 높인다. 하지만 본 연구에서는 학습패턴에 대한 정확한 출력패턴이 아니라, 복원된 영상 중 학습패턴과 유사한 출력패턴을 인식하는 것으로 국한시켰다. 이와 같은 알고리즘을 개발하기 위해 패턴(4×4)의 마스크(Mask)를 샘플(Sample)하여 출력패턴을 분석하였다. 따라서, <Fig. 6-23>과 같은 출력패턴을 위한 학습패턴 두 개의 형태를 기억시킨다.

출력패턴은 <Fig. 6-24>와 같은 샘플패턴과 가중치를 곱하면 출력패턴을 생성한다. 여기서, 학습패턴 쌍은 양극성 데이터로 계산하였고, 패턴의 변환은 열과 열(Column by column)로 배열하였다.

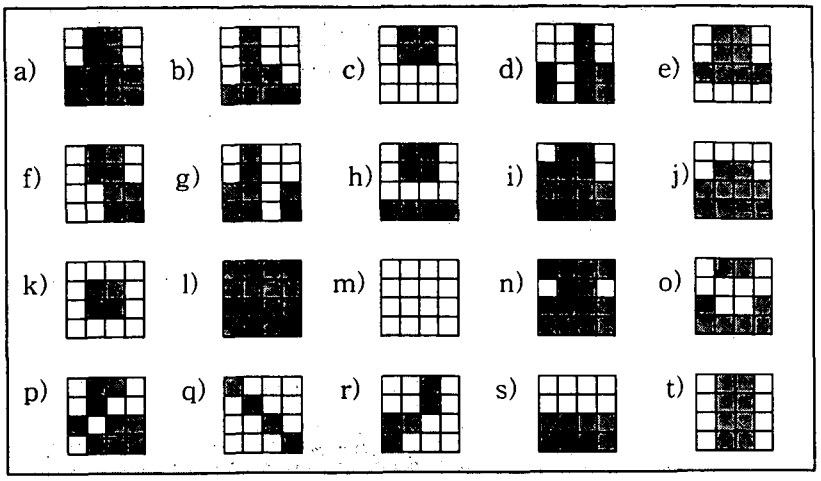


p1) p2)

<Fig. 6-23> Training pattern

패턴의 마스크를 샘플링하여 실험한 결과, <Fig. 6-25>의 출력결과와 같이 학습패턴은 완전한 복원이 일어나지 않은 출력패턴을 추출하였고, 학습패턴과 유사한 샘플패턴은 학습패턴으로 정확히 복원되었다. 그러므로, 학습패턴과 샘플패턴간의 유사관계에 의해서 학습패턴으로 복원이 진행중이거나, 학습패턴과 다른 출력패턴이 복원됨을 판단할 수 있다.

본 연구에서는 학습패턴과 유사한 출력패턴만을 검출하고자 한다. 이를 위해서, 학습패턴에 대한 출력패턴의 검출조건을 분석하였다. 그러므로 학습패턴과 유사한 출력패턴만을 검출하기 위해서는 학습패턴과 다른 패턴을 제거하기 위한 검출조건이 설정되어야 한다. <Table 6-5>은 출력패턴 결과를 나타내었다. <Table 6-5>에서 인식패턴(Recognition pattern)은 두 번의 검출조건에 만족한 출력패턴이고, 제거패턴(Removal pattern)은 두 번의 검출조건에 만족하지 않은 출력패턴이다.



<Fig. 6-24> Example of patterns

<Table 6-5> Output of training pattern

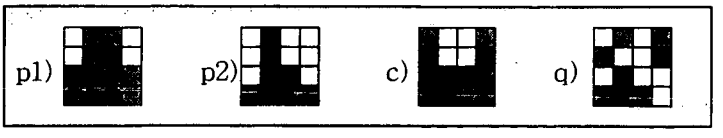
Test / Patterns	Sample patterns		Inclusion unit matrix			Removal unit matrix
			Pattern Value Sum	Hardlims Sum	Hamming Distance	Hardlims Sum
Recognition Patterns	p1	a)	112	8	0	8
		d)	64	8	0	8
		e)	64	8	0	8
		f)	64	8	0	8
		i)	92	8	0	8
		j)	88	8	0	8
		l)	32	8	0	8
		n)	72	8	0	0
		o)	80	8	0	8
		p)	76	8	0	8
		t)	64	8	0	8
	p2	b)	64	0	0	0
		k)	16	0	0	0
Removal Patterns	H · S	g)	64	4	2	8
		h)	64	4	2	8
		m)	-32	-8	16	-8
		r)	28	12	6	16
		s)	64	6	1	8
	H · D	c)	16	8	8	16
		q)	-16	0	6	8

여기서, 인식패턴은 학습패턴 p1과 p2에 대해 각각 두 검출조건을 만족한 경우로 분류하였고, 제거패턴은 각각 H·S(Hardlims Sum)조건과 H·D(Hamming Distance)조건에 의해 분류된 출력패턴을 나타낸다.

학습패턴에 대한 출력패턴의 복원율을 측정하기 위해, 단위 메트릭스(Unit matrix)를 포함하였을 때와 제거하였을때 양극치 합(H·S)을 나타내었다. 해밍거리(H·D)는 학습패턴과 샘플패턴간의 유사도를 측정하였다. <Table 6-5>에서 분석한 바와 같이, 학습패턴과 같은 출력패턴을 인식하기 위해 두 가지 검출조건을 설정하였다.

첫 번째는 학습패턴의 양극치 합을 기준으로 한다. 이 기준은 출력패턴이 학습패턴 H·S의 기준에 포함하는가에 따라, 출력패턴의 유무를 판별하는 것이다. 이 조건을 간략하게 H·S로 표기하였다. 이 조건의 장점은 간단한 조건식과 빠른 검출시간이 가능하고, 학습패턴과 다른 출력패턴을 많이 제거할 수 있다. 그러므로 H·S조건은 출력패턴의 분류를 효율적으로 처리할 수 있다. 하지만, 이 조건의 단점으로는 유사도 측정이 불가능하여, 완전하게 학습패턴과 다른 출력패턴을 제거할 수 없다.

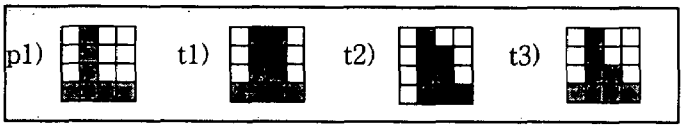
<Fig. 6-25>은 H·S조건에서 오검출된 출력패턴이다. 학습패턴 p1과 p2의 H·S는 8과 0으로, 출력패턴 c)와 q)도 같은 값을 갖는다. 그림과 같이, H·S조건은 학습패턴에 대한 다른 패턴이 출력되더라도 검출될 단점이 있다. <Fig. 6-25>과 같은 오검출을 제거하기 위해 두 번째 조건을 추가하였다.



<Fig. 6-25> Missed extraction on H·S conditions

두 번째는 해밍거리(H · D)를 적용한다. 이 검출조건은 H · S에 의해 오검출된 출력패턴만을 다시 제거한다. 해밍네트웍(Hamming net)중의 하나인 해밍거리는 각 비트별로 XOR연산하여, 같은 비트는 0으로, 다른 비트는 1로 계산한다. 즉, 해밍거리가 0이면 학습패턴과 같은 출력패턴이고, 값이 클수록 유사관계가 작아지기 때문에, 학습패턴과 다른 출력패턴으로 판단할 수 있다. 이 조건을 간략하게 H · D로 표기하였다.

<Fig. 6-26>는 학습패턴과 샘플패턴간의 H · D의 유사도 측정을 나타낸다. 학습패턴 p1)에 대한 샘플패턴 t1), t2), t3)의 H · D는 3, 2, 1 값을 갖는다. 그러므로 학습패턴 p1)과 가장 유사한 샘플패턴은 t3)로 판단할 수 있다.



<Fig. 6-26> Test of similarity on H · D condition

이와 같이, <Table 6-6>에서 학습패턴에 대한 출력패턴의 검출조건 결과는 다음과 같다. 첫째, 학습패턴의 H · S는 p1)은 8, p2)는 0이므로 H · S범위를 유연성(Flexibility)있게 아래 식과 같이 설정할 수 있다.

$$p1 = (7 \le H \cdot S \le 9)$$

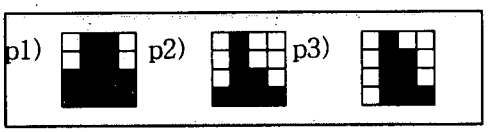
$$p2 = (-1 \le H \cdot S \le 1)$$

둘째, 위 식의 H · S 조건에 만족한 출력패턴은 다시 H · D조건에서 오검출을 제거한다. 아래 식과 같이 학습패턴 p1)과 p2)의 조건에서 검출여부를 판단한다. 두 조건을 하나라도 만족하지 않으면 출력패턴을 제거한다.

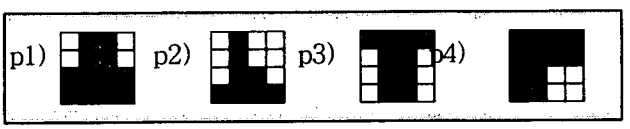
$$p1 = (0 \le H \cdot D \le 1)$$

$$p2 = (0 \le H \cdot D \le 1)$$

<Fig. 6-23>의 출력패턴의 결과에서 c), q)는 H · D조건에 만족하지 않는 출력패턴이고 g), h), m), r), s)패턴은 H · S조건에서 만족하지 않는 출력패턴이다. 이와 같이, 본 알고리즘은 연상기억에 있어 일정한 학습패턴들을 연결강도로 저장하였다가 샘플패턴이 주어질 때 학습패턴과 유사한 출력패턴을 검출하였다. <Fig. 6-27>,<Fig. 6-28>의 학습패턴 3개와 4개를 <Fig. 6-22>의 샘플패턴으로 실험한 결과는 <Table 6-7>에 나타내었다. 여기에서 학습패턴의 수가 많을수록 학습패턴과 같은 출력패턴의 수가 작아진다는 것을 판단할 수 있었다. 따라서 학습패턴과 유사한 출력패턴은 인식하고, 학습패턴과 다른 출력패턴은 제거하고자 한다. 본 절에서 인식조건에 대한 결과는 <Table 6-8>에 간략하게 나타내었다. 20개의 샘플패턴을 학습패턴 2개, 3개, 4개로 각각 실험할 때, 학습패턴에 대한 출력패턴수가 13개, 9개, 5개로 작아짐을 확인할 수 있다. 이와 같이, 너무 많은 학습패턴을 저장하고 있으면 수렴할 때 학습패턴에 대한 다른 패턴을 연상할 수 있다.



<Fig. 6-27> Three training patterns



<Fig. 6-28> Four training patterns

<Table 6-6> Output patterns

Learning pattern : 3						Learning pattern : 4					
Pattern	Sample patterns		PVS	H · S	H · D	Pattern	Sample patterns		PVS	H · S	H · D
Recogniti on patterns	p1	d)	20	4	0	Recognitio n patterns	p1	a)	112	8	0
		i)	32	4	0			g)	48	8	0
		l)	20	4	0			h)	80	8	0
		n)	28	4	0			i)	96	8	0
		o)	32	4	0		p2	b)	42	0	0
		p)	26	4	0	Removal patterns	H · S	c)	48	16	4
	p2	g)	-4	-2	0			e)	80	10	1
	p3	k)	12	0	0			f)	16	2	1
		q)	-4	0	0			m	-80	-4	10
Removal patterns	H · S	a)	36	2	1			n)	112	6	1
		c)	12	2	1			o)	104	6	1
		e)	20	2	1			p)	104	6	1
		f)	28	2	1			q)	-64	-8	8
		h)	28	2	1			r)	48	14	5
		j)	20	2	1			s)	-16	2	1
		t)	36	2	1		H · D	d)	-16	4	2
	H · D	b)	32	2	2			j)	48	4	3
		m	-20	-4	6			k)	16	8	4
		r)	8	16	6			l)	80	8	4
		s)	4	2	2			t)	112	8	5

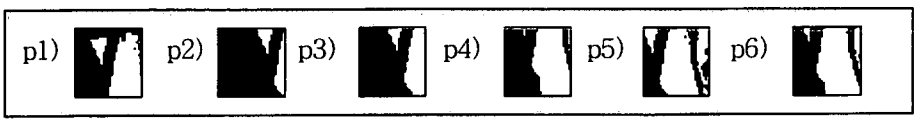
<Table 6-7> Results of training patterns

Learning pattern number	Sample pattern number	Recognition patterns		Removal patterns				
		Detection number	Detection rate	Number	H · S rate		H · D rate	
2	20	13	65.0%	7	5/7	71.4%	2/7	28.5%
3		9	45.0%	11	7/11	63.6%	4/11	36.7%
4		5	12.5%	15	10/15	66.6%	5/15	33.3%

라. 신경회로망을 이용한 형상인식 알고리즘

동질연상 메모리의 특성과 복원효율 및 검출조건을 분석하였다. 본 절에서는 전 절의 결과를 가지고, 실제 영상에서 오이의 형상정보를 인식하고자 한다. 따라서 오이인식을 판정할 수 있도록 인식알고리즘을 개발한다. 실제영상에서 오이의 형상인식은 <Table 6-30>와 같이 RGB영상을 이치화 영상으로 변환시킨다. 그리고 기억시키기 위한 학습패턴(학습데이터)은 오이의 과경 부분(30×30픽셀)으로 인식하고자 한다. 이러한 픽셀크기는 아래의 그림에서와 같이, 인식하기 위한 과경 부분을 임의로 스캔(Scan)하여 추출한 후 이를 학습패턴으로 기억시킨다.

<Fig. 6-29> Pattern extraction from the cucumber features



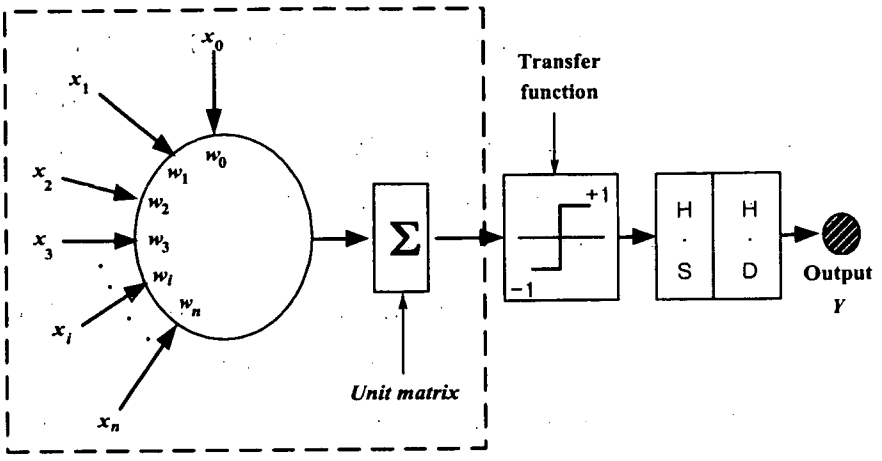
<Fig. 6-30> Auto-recognition of training patterns

오이의 특징부분을 인식하기 위한 학습패턴은 <Table 6-31>과 같이 p1~p6을 추출하여 학습시켰다. 오이의 과경 부분을 인식하기 위한 학습패턴의 검출조건 범위는 <Table 6-8>와 같다. 학습패턴에 대한 H·S와 H·D의 검출조건에 대한 범위설정은 학습패턴과 같은 출력패턴뿐만 아니라, 학습패턴과 유사한 출력패턴도 고려했기 때문이다.

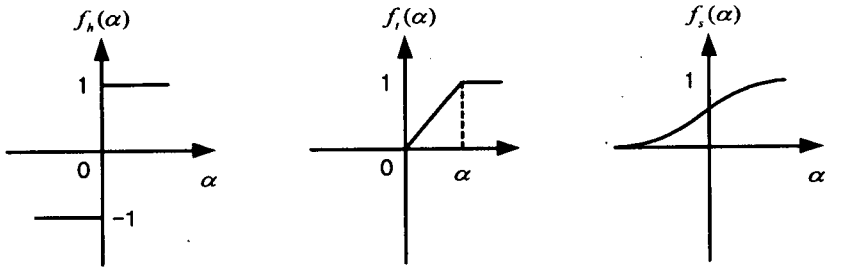
<Fig. 6-31>은 본 알고리즘의 순방향 신경망 구조를 간단히 나타낸다. 각 뉴런의 데이터 값(x_n)이 입력되면, 각 학습 데이터에 대한 가중치(w_n)의 합이 구해진다. 그 후에, 학습패턴에 대한 출력패턴의 복원율을 높이기 위해 대각선 요소, 즉 단위 메트릭스를 이용한다. 또한, 복원율을 더 향상시키기 위해 활성화 함수를 이용한 다음 연상된 패턴을 출력한다.

<Table 6-8> Factors of recognition condition from training patterns

Learning pattern	value	H · S Condition (+900 ~ -900)	H · D Condition
p1	92	80≤ H · S ≤110	0≤ H · D ≤50
p2	390	380≤ H · S ≤400	0≤ H · D ≤50
p3	219	210≤ H · S ≤240	0≤ H · D ≤50
p4	-54	-64≤ H · S ≤-44	0≤ H · D ≤50
p5	-312	-322≤ H · S ≤-300	0≤ H · D ≤50
p6	-232	-242≤ H · S ≤-222	0≤ H · D ≤50



<Fig. 6-31> Principle feed-forward network



(a) Hard Limite (b) Threshold Logic (c) Sigmoid

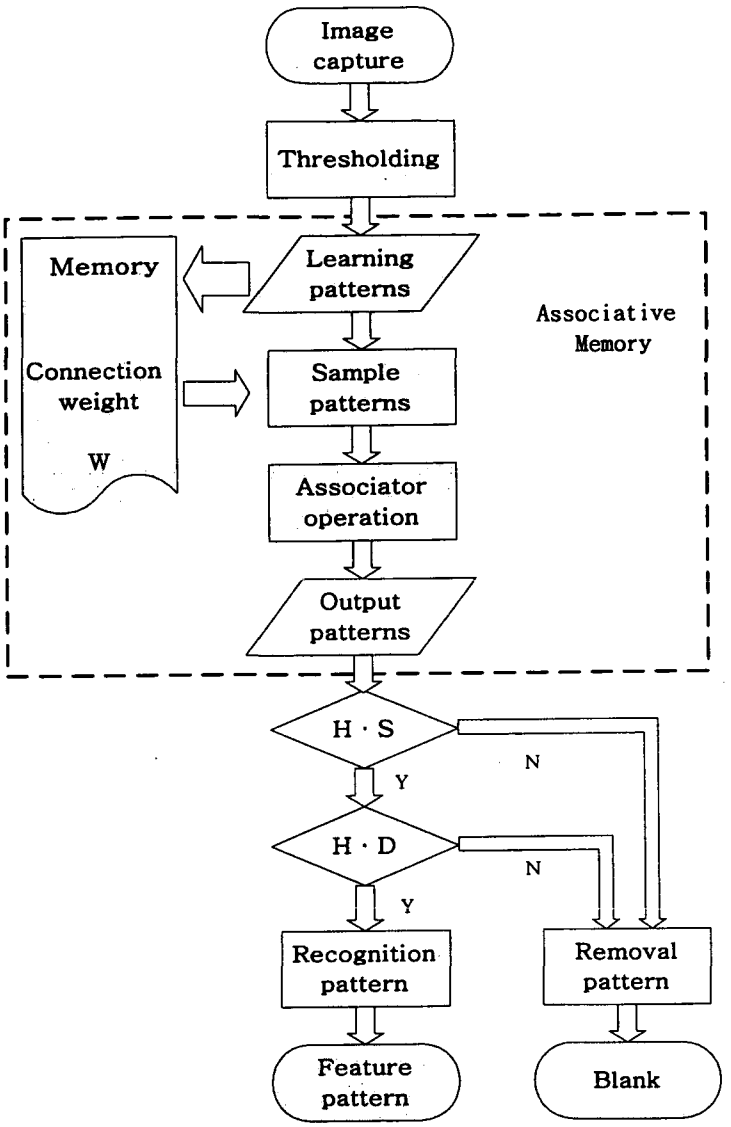
<Fig. 6-32> Nonlinear function

활성화함수(Activation function)는 단조 증가 함수로서, <Fig. 6-32>은 대표적인 비선형 함수들을 나타낸다. 본 연구에서는 계단함수 중 <Fig. 6-32>(a)인 양극성(Bipolar)이진 함수를 사용하였고, 수학적 표현은 다음 식과 같다. 여기서, 출력패턴을 α라 하면 이때, 값이 0보다 적을 경우에는 뉴런의 출력을 -1로 하고, α값이 0이거나, 0보다 클 때는 뉴런의 출력을 1로 변환시킨다.

$$f_h(\alpha) = \begin{cases} +1 : \alpha \geq 0 \\ -1 : \alpha < 0 \end{cases}$$

<Fig. 6-33>는 형상인식 알고리즘을 나타낸 것이며, 알고리즘의 순서에 대한 구체적인 설명은 다음과 같다.

첫째, 카메라에서 획득된 입력영상은 적절한 임계치로 이치화 시킨다.
둘째, 이치화된 영상에서 추출하고자 하는 인식부위, 즉 기억시키고자 하는 학습 패턴을 30×30픽셀 크기로 스캔한다.
셋째, 학습벡터에 대한 출력벡터의 가중치(W)를 구한다.
넷째, 연상메모리에 의해서 학습패턴에 대한 출력패턴을 연상한다. 또한, 출력패턴의 복원 효율을 높이기 위해 단위메트릭스를 이용한 후 양극성 데이터로 변환하여 학습패턴과 유사한 출력패턴을 연상한다.
다섯째, 학습패턴과 유사한 출력패턴만을 검출하기 위해 H·S조건을 이용하여 학습패턴과 다른 패턴을 제거하고, H·S조건에서 오검출된 출력패턴은 H·D조건으로 다시 제거한다
여섯째, H·S와 H·D조건에 모두 만족한 출력패턴은 특징형상으로 인식하고, 하나라도 만족하지 않은 출력패턴은 제거한다.

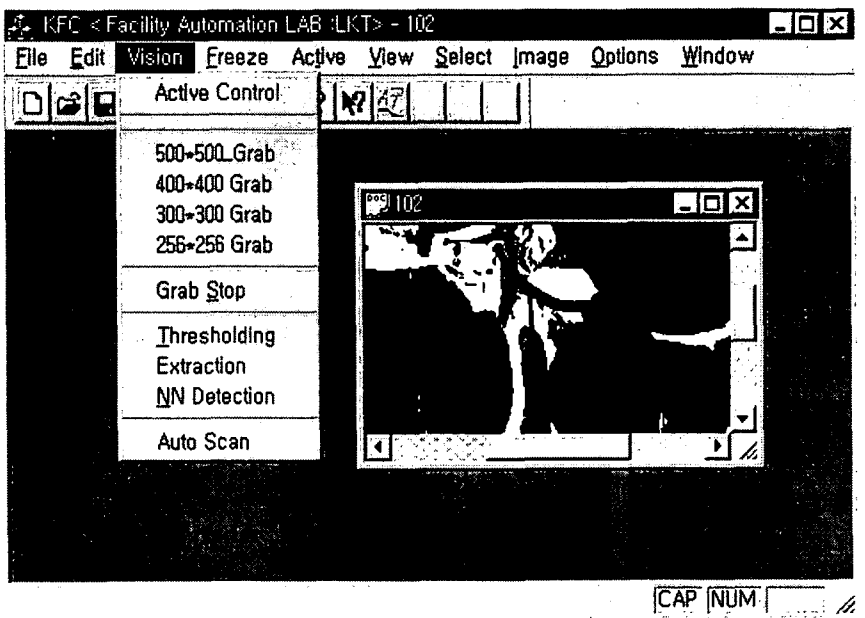


<Fig. 6-33> Flowchart of pattern recognition algorithm

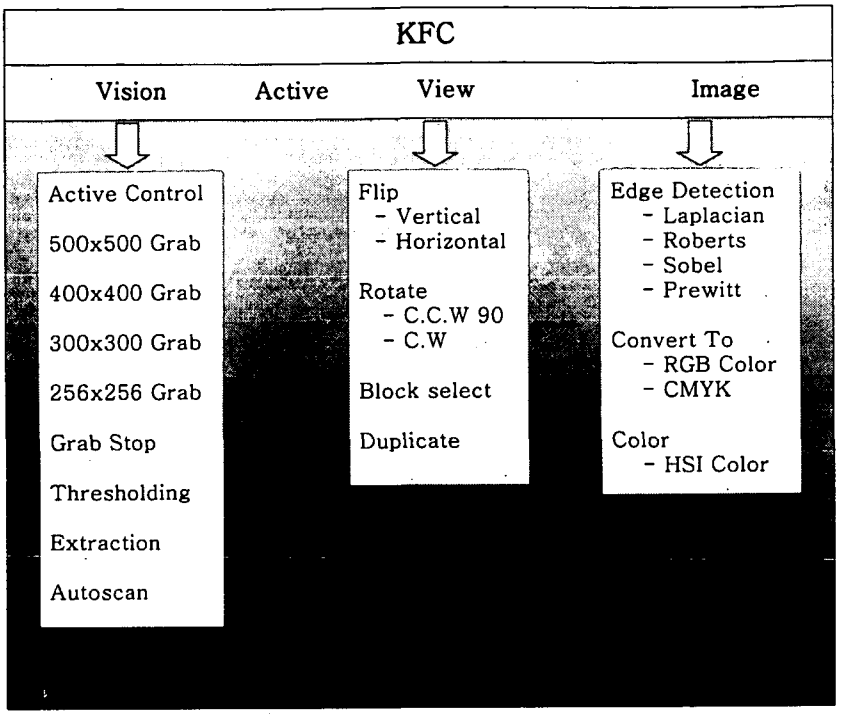
앞에서 개발한 알고리즘을 가지고, 오이의 형상을 구체적으로 인식하기 위해, 지식기반을 통한 형상검출의 컴퓨터모델(Knowledge based Feature detection for Computer model; KFC)프로그램을 개발하였다. 이 프로그램은

Visual C++을 이용하여 구현하였다.

<Fig. 6-34>은 KFC의 전체 메인프레임(Mainframe)화면을 나타낸 것이고, <Fig. 6-35>은 프로그램의 주요 구성부를 나타낸 그림이다. 비젼(Vision)부는 그랩(Grab)의 크기선택, 영상의 이치화, 패턴추출(Extraction) 및 자동검출(Auto scan)기능을 갖는다. 자동검출은 이미지에서 30×30간격으로 진행하고, 검출간격은 조정할 수 있게 구현하였다. 뷰(View)부는 영상을 편집할 수 있는 간단한 기능을 구현하였다. 이미지(Image)부는 각 연산자에 대한 에지 추출과 원 영상을 RGB, HSI, CMYK Color로 변환할 수 있도록 구현하였다.



<Fig. 6-34> KFC



<Fig. 6-35> Structure of KFC

마. 신경회로망을 이용한 형상인식

1) 특징형상 추출

오이의 특징형상 추출을 위한 학습패턴은 실제 영상에서 30×30의 픽셀크기로 스캔하여 학습시켰다. 학습패턴은 오이의 변형된 과경이나 비슷한 과경 등의 패턴을 학습패턴으로 기억시켰다. 이러한 학습패턴은 잎이나 줄기 등의 형태가 복합된 실제영상에서, 학습패턴에 대한 출력패턴의 결과를 분석하기 위함이다.

특징형상을 추출하기 위해, 학습패턴은 1열로 900개를 배열하고, 학습패턴들에 대한 가중치는 900×900를 메모리에 저장한다. 출력패턴을 생성하기 위해,

샘플패턴과 가중치를 곱하면 학습패턴에 대한 출력패턴을 연상하게 된다. 이러한 특징형상 추출의 연상시간은 연구에 사용된 컴퓨터 사양에서 약 1초의 시간이 필요하였다.

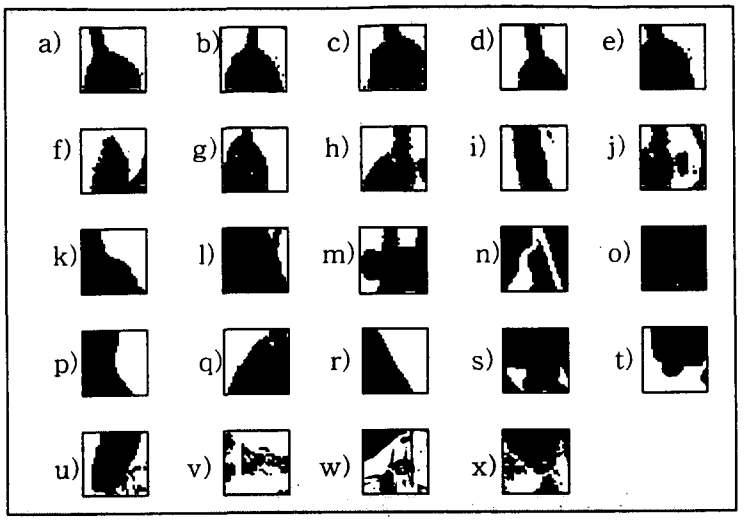
<Fig. 6-36>는 학습패턴에 대한 출력패턴을 연상하기 위해, 6개 패턴의 오이 과경 모양을 기억시켰다. <Fig. 6-37>은 실제 영상에서 <Fig. 6-38>의 학습패턴을 가지고, 출력패턴의 결과를 얻기 위해 실험한 24개의 샘플패턴이다. <Fig. 6-38>는 샘플패턴을 신경회로망에 의해 연상시킨 결과, 출력패턴을 나타낸 그림이다.

<Fig. 6-36> Images for training

<Fig. 6-38>에서 a)~f)패턴은 학습패턴과 유사한 출력패턴으로 검출되었고, g)~x)패턴은 학습패턴에 대한 다른 패턴으로 검출되었다. 또한 출력패턴 중에서 i), k), m), q), r), t), u), v), w)패턴은 학습패턴으로 복원중인 출력패턴으로 판단할 수 있고, h), l), o), p), s), x)패턴은 학습패턴에 대한 반전된 패턴임을 판단할 수 있다.

이러한 출력패턴을 통해서, 학습패턴과 샘플패턴간의 픽셀 분포차가 작으면 학습패턴과 비슷한 패턴으로 검출되고, 픽셀의 분포차가 크면 학습패턴과 다른 패턴으로 출력됨을 판단할 수 있었다. 즉, 오이의 과경부위 인식은 <Fig. 6-38>의 a)~f)패턴과 같이, 학습패턴과 같은 출력패턴을 검출하는 것이다.

이와 같이, 실제 영상에서 오이의 특징형상 검출은 <Fig. 6-37>의 출력패턴 결과를 가지고, 전체영상 크기에서 자동검출로 처리하고자 하였다. 이러한 자동검출 처리로, 오이의 학습패턴과 유사한 출력패턴을 인식하여, 오이의 형상 및 위치를 판정하고자 하였다.



<Fig. 6-37> Sample patterns of real images

따라서, 인식 알고리즘을 실제 영상에 적용한다면, 오이형상 검출은 학습패턴과 유사한 출력패턴을 복원하여, 학습패턴에 대한 출력패턴을 정확히 인식할 수 있을 것으로 판단하였다. 또한, 학습패턴으로 인식된 오이의 특징형상 위치를 추정함으로서, 오이의 위치 측정이 가능하리라 판단되었다.

2) 인식 알고리즘

본 연구에서는 신경회로망의 연상메모리를 이용하여 실제영상에서 오이의 형상을 인식하고자 하였다. 본 인식 알고리즘은 학습패턴과 유사한 출력패턴을 정확하게 연상하였지만, 많은 출력패턴들이 학습패턴에 대한 다른 패턴으로 복원되는 것을 확인할 수 있었다.

본 인식 알고리즘의 특성은 다음과 같다.

첫째, 본 신경회로망은 이치화된 영상을 직접 이용한 방식으로, 이치화 결과에 따라 영향을 직접적으로 받을 수 있다.

둘째, 연상기억에 있어서, 일정한 학습패턴들은 연결강도로 저장하였다가 샘플 패턴이 주어질 때, 학습패턴과 유사한 출력패턴을 찾아낸다.

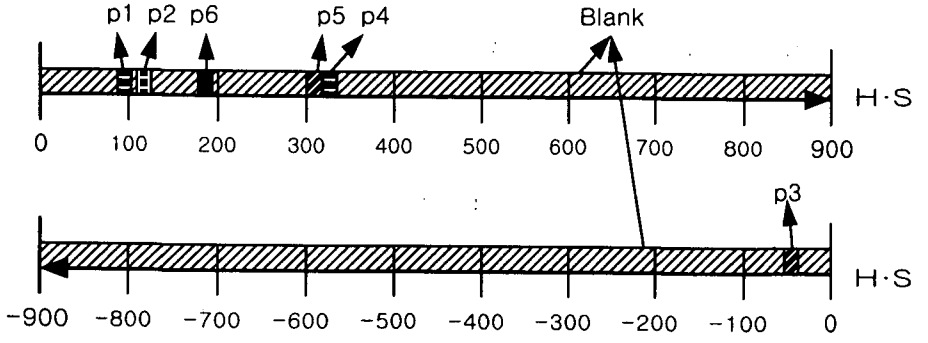
셋째, 다른 신경회로망과는 다르게 미리 가중치가 결정된다. 이러한 특성은 연상시간의 단축과 학습절차가 필요 없다.

넷째, 저장되어 있는 학습패턴 수가 많을수록, 학습된 패턴으로 정확히 복원된 출력패턴의 수는 한정적으로 검출되었다.

인식 알고리즘의 검출조건은 다음과 같다.

첫째, 학습패턴의 양극치 합(H · S)을 기준으로 출력패턴을 분류한다. H · S조건은 출력패턴이 학습패턴의 H · S값에 포함하는가에 따라 출력패턴의 유무를 판별한다. 이러한 검출조건은 빠른 검출시간과 많은 패턴의 판별이 가능하여, 효율적인 출력패턴을 분류할 수 있다. 하지만, 패턴간의 유사도 측정이 불가능하여, 학습패턴에 대한 다른 출력패턴이 검출될 단점이 있다.

둘째, 해밍거리(H · D)에 의한 출력패턴의 분류는 H · S조건에 의해 분류된 출력패턴만을 다시 측정한다. 이 방법은 XOR연산을 통해, 학습패턴에 대한 샘플 패턴의 유사도를 측정하여 검출하므로, 학습패턴과 다른 출력패턴을 완전히 분류할 수 있다.



<Fig. 6-38> H · S range of training patterns

<Fig. 6-38>은 <Table 6-9>에서의 H·S값에 대한 검출조건의 범위를 나타내었다. 그림과 같이, 특징형상(30×30)에서의 H·S값 범위는 +900에서 -900까지 값을 가지며, 이 때 H·S값이 +900일 때는 패턴이 흑색이고, -900일 때는 백색을 나타낸다. 그림에서 학습패턴의 H·S값은 p1~p6을 띠 형태로 표시하였다. 그 이외의 값에 출력패턴이 분포하면, 제거(Blank)한다. 이 조건범위 내에 만족하는 패턴은 H·D를 적용하여 다시 분류함으로서 오이형상을 인식시킨다. 다음의 <Table 6-9>는 <Fig. 6-36>의 학습패턴에 대한 출력패턴의 결과를 얻기 위해, 검출조건을 나타내었다.

<Table 6-9> Extraction conditions of training patterns

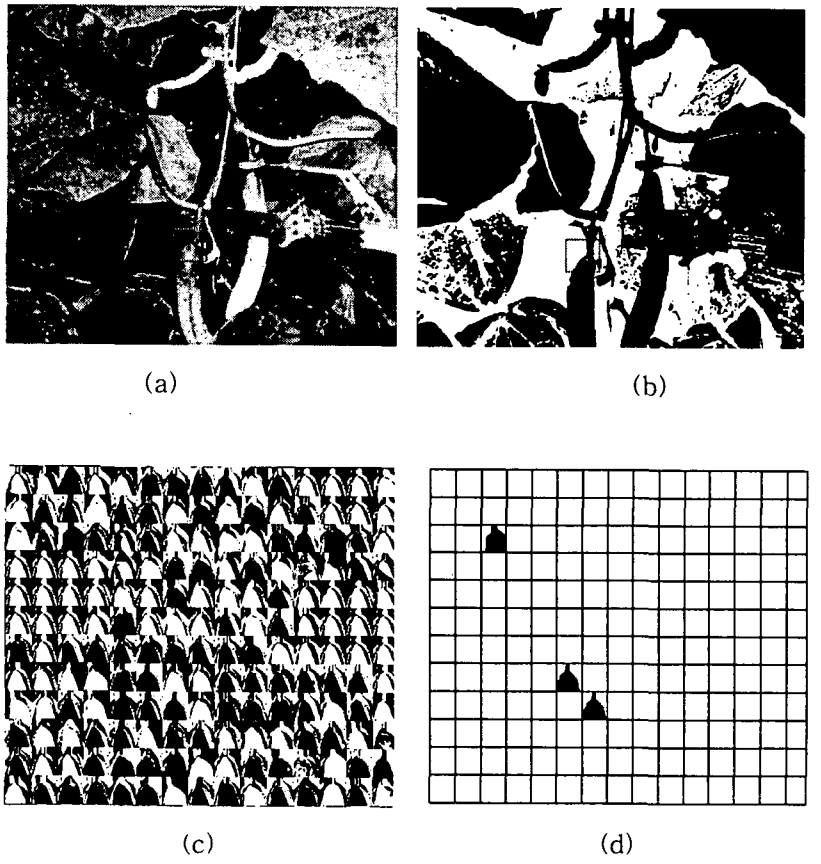
H·S value (+900 ~ -900)		H·S Condition	H·D Condition
Pattern 1	92	80≤ H·S ≤105	0≤ H·D ≤30
Pattern 2	114	106≤ H·S ≤130	0≤ H·D ≤30
Pattern 3	-42	-55≤ H·S ≤-30	0≤ H·D ≤30
Pattern 4	332	323≤ H·S ≤345	0≤ H·D ≤30
Pattern 5	312	300≤ H·S ≤322	0≤ H·D ≤30
Pattern 6	182	170≤ H·S ≤195	0≤ H·D ≤30
Beyond value		Blank	Blank

바. 실제 영상의 자동검출과 인식판정

본 절에서는 신경회로망의 인식알고리즘 특성과 검출조건의 분석을 기초로 실제영상에서 자동검출로 처리하여 적용하였다. 자동검출 처리는 오이의 특징형상 정보 인식에 대한 검출율을 판정하고자 하였다. 이러한 결과데이터를 통해서 본 알고리즘을 이용한 영상처리 시스템이 현장 적용에 가능한지를 구명하고자 하였다.

오이 인식을 위해서 <Fig. 6-35>의 학습패턴을 기억시켰고, 검출조건은 <Table 6-9>를 이용하여 조건화시켰다. <Fig. 6-39>(a)의 이미지 크기는 445

×363픽셀이며 오이의 인식부위를 사각형으로 표시하였다. 영상정보를 효율적으로 구분하기 위해, <Fig. 6-39>(b)에서 임계값은 150을 설정하여 이치화 하였다. <Fig. 6-39>(c)의 이미지는 연상메모리에 의해서 검출된 학습패턴에 대한 출력패턴을 나타낸다. 이러한 알고리즘에 의해 검출되는 출력패턴들은 30×30간격으로 오이인식을 자동검출로 처리하였다.



<Fig. 6-39> Recognition on real cucumber images at field (I)

즉, 검출조건을 설정하지 않으면 출력패턴들이 <Fig. 6-39>(c)처럼 모두 검출됨을 확인할 수 있다. 그러므로, 오이의 특징형상 인식은 <Table 6-9>의 조

건으로 출력패턴을 분류하여, 학습패턴과 유사한 출력패턴을 검출하였다. 이러한 검출조건은 인식하고자 하는 오이의 특징형상을 인식하기 위함이다. <Fig. 6-39>(d)는 학습패턴과 유사한 출력패턴을 인식한 결과이고, 학습패턴과 다른 출력패턴들은 제거됨을 보여준다.

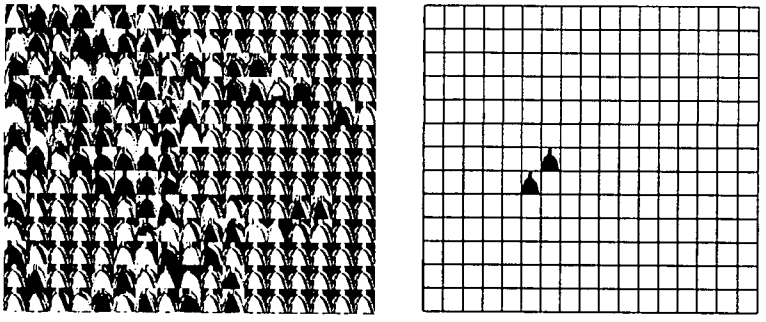
<Fig. 6-39>(d)에서, 사각형은 Visual C++의 Rectangle함수를 이용하여 검출 조건에 만족하지 않은 출력패턴을 사각형으로 처리하였다. <Fig. 6-39>(d)의 영상에서는, 본 연구 목적인 두 가지 정보를 얻을 수 있다.

첫째, 인식된 패턴은 학습패턴과 유사한 출력패턴으로서, 오이의 특징형상을 판정할 수 있다.

둘째, 인식된 특징형상의 좌표를 통해서, 오이의 과경 위치를 추정할 수 있다. <Fig. 6-39>(a)에서 오이의 특징형상과 <Fig. 6-39>(d)의 학습패턴에 대한 출력패턴은 오이의 과경에 인식되었다. 또한 실제영상에서 과경 부분의 좌표는 (196,261)이고, <Fig. 6-39>(d)의 출력패턴 중 과경 형상에 인식된 출력패턴의 좌표는 (192,255)에서 검출되어, 오이에 대한 위치정보를 측정할 수 있었다. 하지만 2개의 특징형상이 오검출되어 학습패턴으로 인식됨으로서, 실제영상에서 오이의 위치측정은 가능하나 정확한 위치정보는 오차를 포함하고 있음을 판단할 수 있었다.

<Fig. 6-35>의 학습패턴과 <Table 6-9>의 검출조건으로 다른 영상에서 오이 인식을 실험하였다. <Fig. 6-40>(a)의 영상 크기는 501×391픽셀이고, <Fig. 6-40>(b)의 이치화 영상은 임계값을 210으로 설정하였다.

(a) (b)



(c) (d)

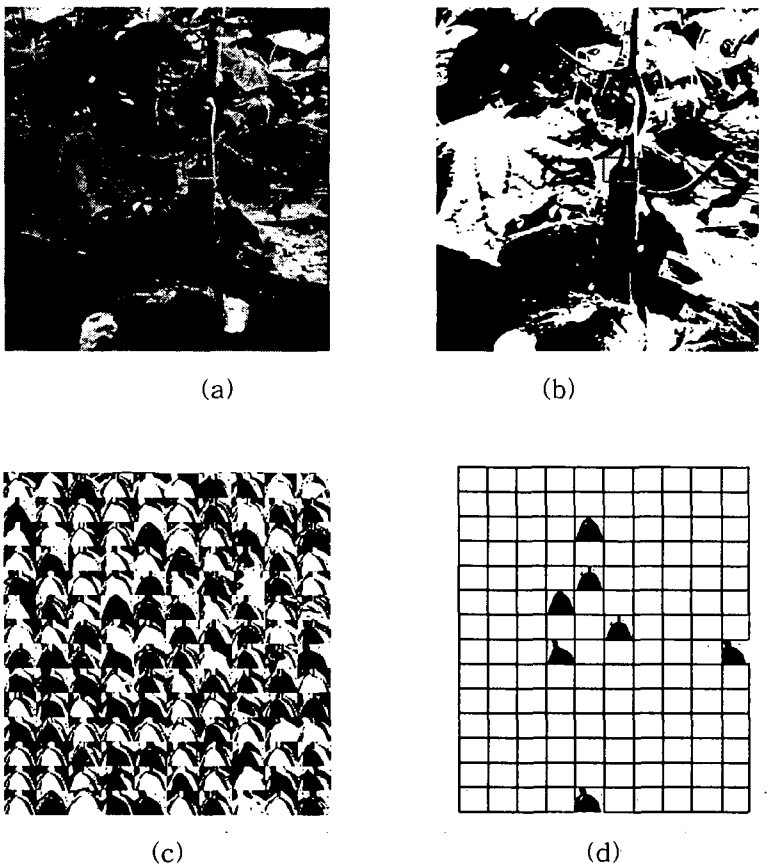
<Fig. 6-40> Recognition on real cucumber images at field (II)

<Fig. 6-40>에서 사각형으로 표시한 오이의 과경 좌표는 (171,209)이고, <Fig. 6-40>(d)에서 오이에 검출된 특징형상의 좌표는 (164,223)에서 인식되었다. 하지만, 검출조건에 의해 인식된 출력패턴은 1개가 더 연상됨으로서, 오검출을 나타내었다. 그 이유로는 이치화에 의한 영상정보가 학습패턴과 유사한 출력패턴으로 연상되었기 때문인 것으로 판단되었다.

<Fig. 6-41>도 마찬가지로 <Fig. 6-35>의 학습패턴과 <Table 6-9>의 검출조건으로 오이인식을 실험하였다. <Fig. 6-41>(a)의 영상크기는 300×421픽셀이고, <Fig. 6-41>(b)의 이치화 영상은 임계치를 80으로 설정하였다. <Fig. 6-41>(a)에서 사각형으로 표시한 과경의 좌표는 (170,191)이고, <Fig.

6-41>(d)에서 오이로 인식된 특징형상의 좌표는 (165,192)에서 검출하였다. <Fig. 6-41>(d)에서와 같이, 오이의 특징형상은 6개 패턴을 학습패턴과 같은 출력패턴으로 오검출하였다. 오검출로 인식된 주된 이유는 위와 같이, 임계치 설정에 의한 이치화 영상이 너무 복잡한 영상정보를 추출하였기 때문이다. 이러한 결과는 학습패턴에 대한 유사한 출력패턴을 연상하므로, 오검출이 많이 나타난 것으로 판단하였다. <Fig. 6-42>의 실제영상은 잎과 줄기로 혼합된 영상으로, 오이에 대한 특징형상의 인식율을 실험해 보고자 하였다. <Fig. 6-42>(a)의 영상크기는 297×421픽셀이고, <Fig. 6-42>(b)의 이치화영상은 임계치를 160으로 설정하였다. 학습패턴과 검출조건은 <Fig. 6-35>와 <Table 6-9>를 이용하여 학습패턴에 대한 출력패턴의 유무를 검출하였다.

<Fig. 6-42>(d)에서와 같이, 실제 영상에서 학습패턴에 대한 출력패턴은 1개의 특징형상을 오검출로 나타났다. 이러한 오검출은 <Fig. 6-41>의 오검출 분석과 마찬가지로 전처리에 의한 이치화 영상이 복잡한 영상정보를 추출하였기 때문임을 판단할 수 있었다.

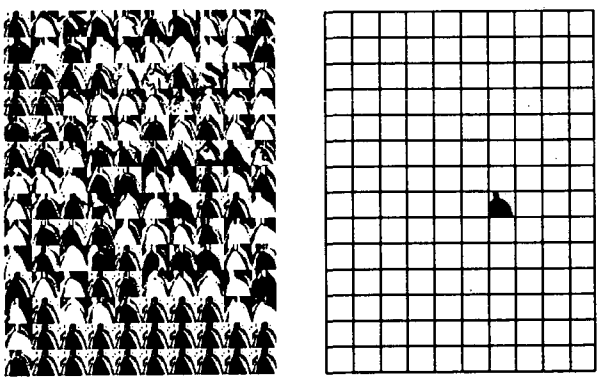


(a) (b)

(c) (d)

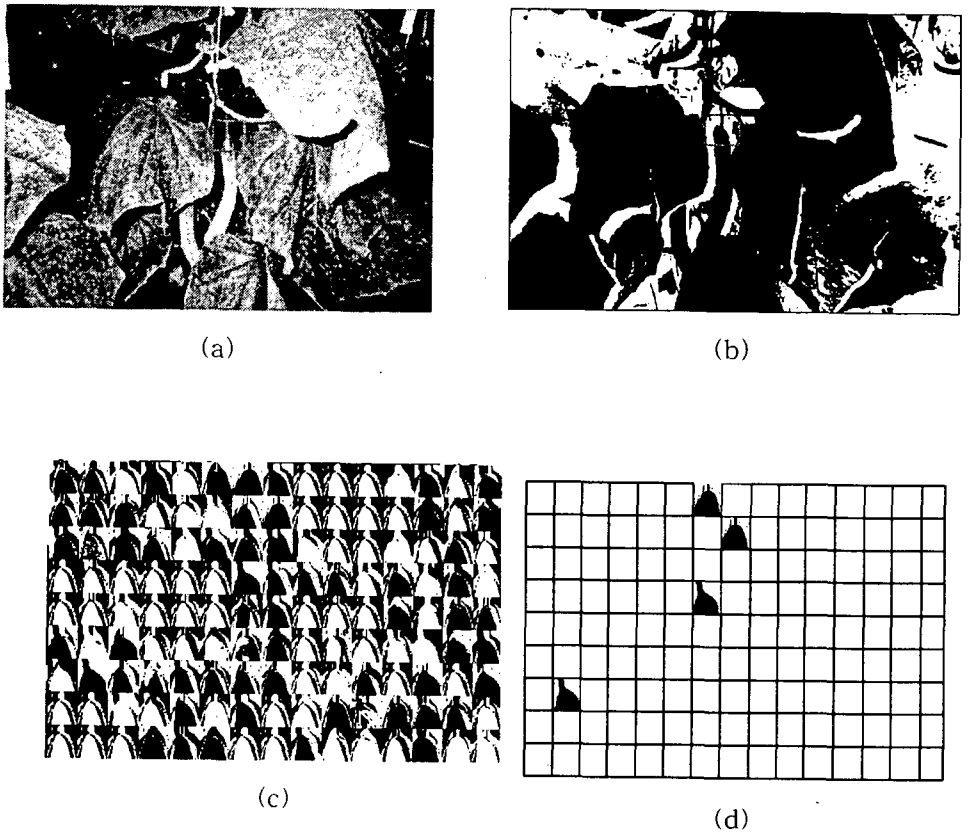
<Fig. 6-41> Recognition on real cucumber images at field (III)

(a) (b)



(c) (d)

<Fig. 6-42> Recognition on real cucumber images at field (IV)



(a) (b)

(c) (d)

<Fig. 6-43> Recognition on real cucumber images at field (V)

<Fig. 6-43>(a)의 실제영상 크기는 450×271픽셀이고, <Fig. 6-43>(b)의 이치화 영상은 임계치를 180으로 설정하였다. 학습패턴과 검출조건은 <Fig. 6-35>와 <Table 6-9>를 적용하였다. <Fig. 6-43>(a)에서 오이의 과경 좌표는 (211,102)이고, <Fig. 6-43>(d)에서 학습패턴과 유사한 출력패턴으로 검출된 과경 부위의 좌표는 (205,103)에서 비슷하게 인식되었다. <Fig. 6-43>(d)에서 학습패턴에 대한 출력패턴의 검출 결과는 3개의 특징형상을 오검출하였다. 그 이유는 <Fig. 6-42>의 오검출 분석과 같은 것으로 판단된다.

<Table 6-10>은 본 절에서 5개의 실제영상을 가지고 자동검출로 처리한 오이인식의 결과를 나타내었다. 여기서, No.1~4는 오이의 인식과 위치추정을

검출하기 위한 실제영상이고, No.5는 학습패턴에 대한 출력패턴의 유무를 판별하기 위한 실제영상이다. 표에서, 총검출수(Total detection numbers)는 전체영상에서 오이의 검출을 자동으로 검출한 수이다. 또한 검출된 수(Detected numbers)는 검출조건을 적용한 후 학습패턴과 유사한 출력패턴으로 인식한 수이다.

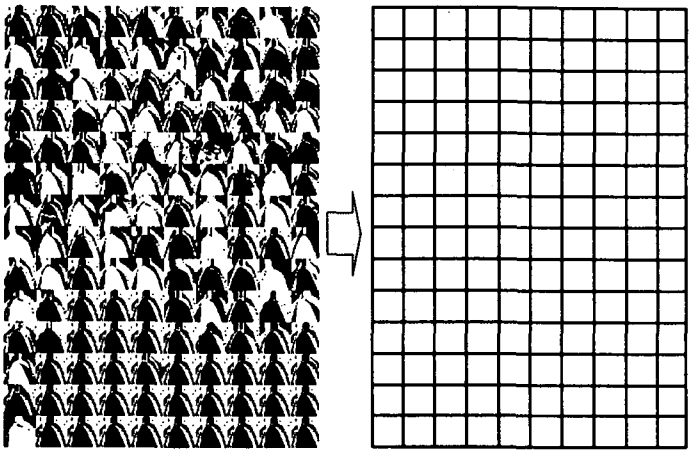
오이로 판정된 패턴 중에서, 오이의 과경 부위에 인식된 패턴은 인식수(Recognition number)로 나타내었고, 과경으로 오인식된 패턴은 오검출 패턴(False detection pattern)으로 표기하였다. 오검출 패턴은 오검출율과 제거율로 분류하여 나타내었다. 이러한 데이터를 통해서 본 알고리즘의 연상능력 및 검출조건에 대한 분류능력을 판단할 수 있었다.

<Table 6-10> Results of pattern recognition

No.	Image size (Pixels)	Threshold value	Total detection numbers	Recognition pattern		False detection pattern		
				Detected numbers	Recognition number	Number	False rate	Removal rate
1	445×363	170	15×12 (180)	4	1	3	1.6 %	98.4%
2	501×391	210	17×13 (221)	2	1	1	0.1 %	99.9%
3	300×421	80	10×14 (140)	7	1	6	4.2 %	95.8%
4	450×271	180	15×9 (135)	4	1	3	2.2 %	97.8%
5	297×421	160	10×14 (140)	1	-	1	0.1 %	99.9%

실제영상에서 오이의 인식결과, 오검출율(False rate)은 전체 영상에서 0.1~4.2%를 나타내었고, 학습패턴에 대한 다른 출력패턴은 96~99%의 제거율(Removal rate)을 나타내었다. 이러한 데이터는 자연상황의 복잡한 영상을 고려할 때, 만족할 만한 결과를 얻었다고 판단하였다. 또한, 오검출에 대한 주된 원인은 이치화 영상에서 불명확한 영상정보를 얻었기 때문인 것으로 판단되었다. 따라서 본 알고리즘은 이치영상을 직접 이용한 방식이기 때문에 임계치 결

과에 따라 영향을 직접적으로 받을 수 있었다. 이를 증명하기 위해서, <Fig. 6-44>의 결과를 나타내었다.



<Fig. 6-44> Image pattern recognition on critical value 140

<Fig. 6-44>은 <Fig. 6-42>의 이치화 영상에서 임계치를 160에서 140으로 설정한 후의 학습패턴에 대한 출력패턴의 결과를 나타낸다. 임계치가 160에서는 1개를 오검출하였으나, 임계치가 140에서는 무(無)검출되었다. 이와 같이 전처리에 있어서, 임계치 설정은 인식판정에 대한 중요한 변수임을 확인할 수 있었다. <Table 6-11>은 <Table 6-10>에서 임계치를 다르게 설정하였을 때, 오이의 검출율 결과를 나타내었다. 표에서 특히, 3번과 5번 영상의 임계치 설정이 <Table 6-10>의 임계치 설정보다 더 효율적으로 오이에 대한 인식율을 나타내었다. <Table 6-12>는 원 영상에서, 오이의 과경 위치와 오이로 인식된 출력패턴의 위치 좌표를 나타내었다. 여기서, 위치오차(Position error)는 실제 영상에서의 오이 과경과 인식패턴의 좌표차이를 나타낸다. 이와 같이, 실제영상에서 위치오차는 30×30간격으로 검출하였기 때문에 발생한 오차로 판단된다. 검출간격을 통한 위치추정은 간격 범위를 줄일수록 오차를 줄이므로, 정확한 오이형상의 위치추정이 가능할 것으로 판단된다.

<Table 6-11> Results of pattern recognition on each threshold value

No.	Image size (Pixels)	Threshold value	Total detects numbers	Recognition pattern		False detection pattern		
				Detected numbers	Recognition number	Number	false rate	Removal rate
1	445×363	150	15×12 (180)	5	1	4	2.6 %	97.4%
2	501×391	190	17×13 (221)	4	1	3	1.6 %	98.4%
3	300×421	60	10×14 (140)	5	1	4	2.9 %	97.1%
4	450×271	160	15×9 (135)	4	1	3	2.2 %	97.8%
5	297×421	140	10×14 (140)	0	-	0	0.0 %	100.0%

<Table 6-12> Errors of position from recognized images

No. of images	Image size (Pixels)	Pattern coordinate		
		Original image	Recognized image	Position error
1	445×363	(196,261)	(192,255)	(4,6)
2	501×391	(171,209)	(164,223)	(7,14)
3	300×421	(170,191)	(165,192)	(5,1)
4	450×271	(211,102)	(205,103)	(6,1)
5	297×421	-	-	-

제 5 절 요약 및 결론

오이수확로봇 개발에 있어서 정확한 오이의 형상 및 위치를 인식하기 위하여 형상인식 알고리즘에 대한 연구를 수행하였다. 다양한 오이형상을 인식하기 위한 방법으로는 신경회로망의 연상 메모리를 이용하여 오이의 특징형상을 인식하였다. 또한, 오이의 영상정보를 구별하기 위한 영상처리방법으로는 영상이치화와 경계선 분할 추출에 대한 연구를 수행하였다. 오이의 형상인식은 이치화 영상을 직접 이용한 방식으로 수행하였다. 또한, 학습패턴에 대한 출력패턴을 조건화시킴으로서, 학습패턴과 유사한 출력패턴만을 검출하도록 알고리즘을 구현하였다.

형상인식은 실제영상에서 오이의 형상과 위치를 판정할 수 있도록 알고리즘을 개발한 결과, 다음과 같은 결론을 얻었다.

1. 본 알고리즘에서는 일정한 학습패턴의 수를 2개, 3개, 4개를 각각 기억시켜 샘플패턴 20개를 실험하여 연상시킨 결과, 학습패턴으로 복원된 출력패턴의 비율은 각각 65.0%, 45.0%, 12.5%로 나타났다. 이는 학습패턴의 수가 많을수록 수렴할 때, 다른 출력패턴으로 많이 검출되었다.

2. 형상인식 알고리즘은 복원율을 높이기 위해서, 학습패턴의 양극치 합(H · S)에 대한 검출조건을 추가하였다. 그 결과, 학습패턴에 대한 다른 출력패턴을분류할 수 있었다. 이는 실제영상에서 H · S 검출조건은 63.6%~71.4%의 학습패턴에 대한 다른 출력패턴을 분류하였다. 따라서, H · S조건은 많은 패턴의 유무판별 및 빠른 검출시간이 가능하였다.

3. 인식효율을 높이기 위한 패턴의 유사도 측정은 해밍거리(H · D)를 적용하였다. H · D는 H · S 조건에 의해 분류된 출력패턴 중에서, 유사도를 측정하여 오검출된 출력패턴을 분류시켰다. H · D조건을 추가함으로서, 학습패턴과 다른

출력패턴을 28.5%~36.7% 제거하였으며, 이는 검출시간과 출력패턴의 분류에 대한 효율을 증대시켰다.

4. 오이의 특징형상 검출은 30×30간격으로 자동검출 되도록 처리하였다. 실제 영상에서 자동 검출로 처리한 결과, 오이인식의 처리시간은 약 0.5~1초/1개(패턴) 빠르게 검출되었다. 또한, 다섯 개의 실제 영상에서 실험한 결과, 학습패턴에 대한 다른 출력패턴은 96~99%의 제거율을 나타내었다. 오이로 인식된 출력패턴 중에서, 오검출된 출력패턴의 비율은 0.1~4.2%를 나타내었다.

5. 본 연구에서는 신경회로망을 이용하여 오이의 형상 및 위치를 인식할 수 있도록 알고리즘을 개발하였다. 오이의 위치측정은 실제영상에서 학습패턴과 유사한 출력패턴의 좌표를 가지고, 오이의 위치좌표를 추정할 수 있었다.

제 7 장 종합시스템 개발

제 1 절 서 설

시설원예에서 소요되는 노동력 중에서 가장 많은 비중을 차지하고 있는 것이 수확작업이다. 특히 오이의 경우 지상 1m 내외에서 수확작업을 하기 때문에 적당한 매니퓰레이터가 있어야 한다. 메니플레이터의 역할은 오이를 파지하고 줄기를 절단하는 역할을 하는 그리퍼를 공간상에서 원하는 위치로 정확하게 이동시키는 것이다. 이때 요구되어지는 사항은 먼저 신속히 공간상을 이동하기 위해 모든 구조물이 가벼워야 하며, 3차원 위치정확도는 5mm내외 어어야 한다.

즉 모터 및 감속기가 메니플레니터 무게의 대부분을 차지한다고 해도 과언이 아니다. 하중을 줄이기 위해 모터 및 감속기를 아래에 두어 작동 시 회전관성 모멘트를 줄었다. 상하의 이동은 스크류를 이용하여 정확하게 제어 할 수 있도록 하였다. 이렇게 함으로써 장치를 고속으로 작동하게 함과 동시에 내구성 향상에도 기여한다.

위에서 오이수확용 로봇개발을 위해서 각 시스템의 구조와 성능에 대해서 언급하였다. 본 장에서는 위의 각 시스템을 기본으로 종합시스템을 구축하였다.

제 2 절 실험재료 및 방법

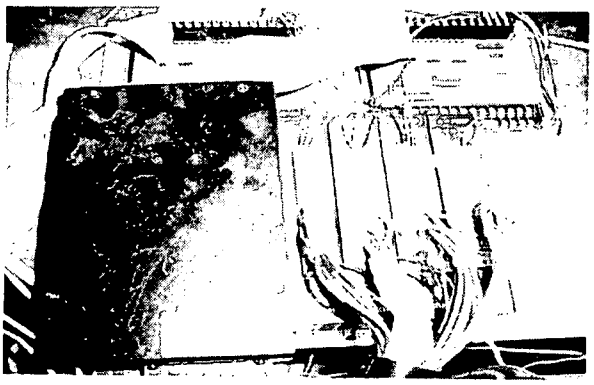
1. 실험장치

각 시스템의 구성요소들의 연결하여 종합시스템을 구축하였으며, <Fig 7-1>은 종합시스템의 전체 외형도를 나타낸 그림이다.

<Fig. 7-1> The whole system

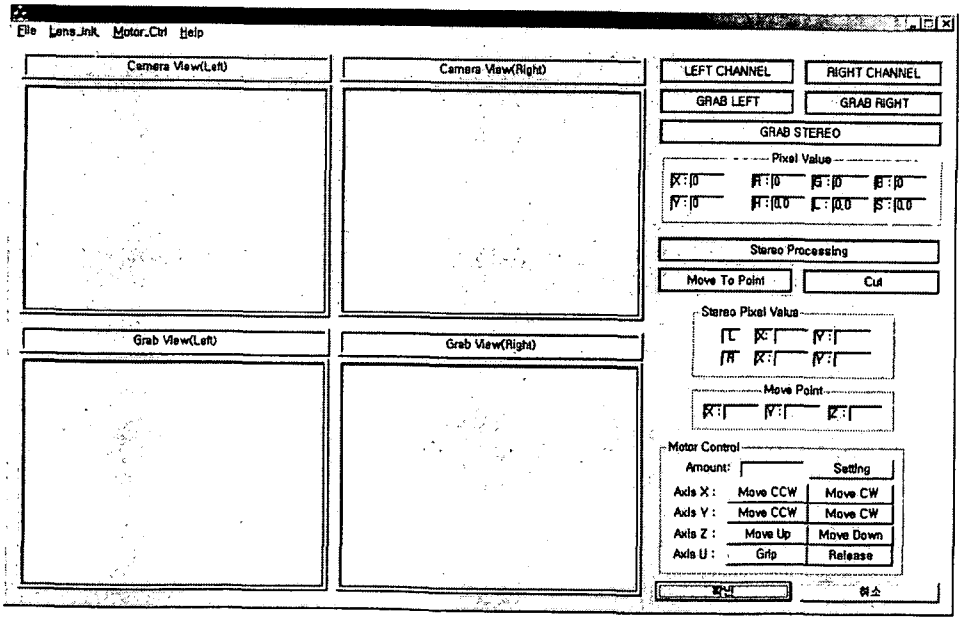
<Fig. 7-2>은 3차원 오이의 영상을 추출하기 위한 카메라이며, <Fig 7-3>는 매니퓰레이터를 구동하기 위한 모터 컨트롤러를 나타내고 있다.

<Fig. 7-2> Iimage processing system



<Fig. 7-3> The motor control for system operating

또한 영상처리를 통한 오이의 위치 결정을 위해 구축한 전체 프로그램의 본 화면을 <Fig. 7-4>에서 나타내었다.

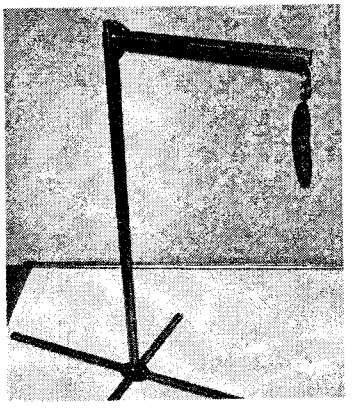


<Fig. 7-4> The main program for image processing

2. 실험방법

가. 실내실험

본 실험은 영상장치를 통하여 입력된 좌표로 매니퓰레이터가 실제 좌표로 정확하게 이동하는 것을 알아보기 위해서 카메라 영상의 원점을 기준으로 X, Y축으로 -100mm, 0mm, 100mm, Z축으로 -200mm, 0mm, 200mm이동하면서 위치별로 10회 반복 측정하였다. 또한 <Fig. 7-5>은 매니퓰레이터의 이동오차를 실험하기 위하여 제작한 실험 장치이다. 실험장치는 오이를 매단 후, 오이 과경을 절단하기 위하여 가로 세로가 20×20mm인 프로파일을 이용하여 장축 1100mm, 단축 420mm, 밑판 장축 800mm, 단축 600mm이며, 오이 과경의 결속을 위하여 장축의 끝에 클립을 장착하여 실내 실험을 하였다.



<Fig. 7-5> Equips for experiment

종합시스템의 작업성능을 검증하기 위해서 오이 270개를 대상으로 수확실험을 행하였다. 또한 오이 과경에 작업기가 정확하게 접근하는 것을 알고자, 지면으로부터 오이의 결과 위치 3가지와 모터의 rpm 속도에 따라 3가지의 각각의 경우에 대하여 과경 지름에 따른 3가지로 나누어서 매니퓰레이터의 최적

의 속도 및 정확도를 알아보고자 하였다.

오이는 지면으로부터 1m내에서 결과하기 때문에 지면으로부터 30cm, 60cm 90cm로 하여 최적의 수확 높이를 측정하고, 모터의 rpm에 따라 오이 과경에 접근하는 오차를 측정하고자 50rpm, 100rpm, 150rpm으로 하였다. 또한 rpm에 대하여 오이 과경에 따른 오차의 발생률을 측정하고자 과경 지름 4.5mm이하 4.5~5.5mm, 5.5mm이상의 3가지 경우에 대하여 하였다.

<Table 7-1>은 rpm에 따른 3가지와 rpm에 대한 오이 과경의 영향 3가지 지면으로부터 오이의 결과 위치에 따른 3가지로 나누어서 총 27개의 실험구를 두어 각각의 실험구에 대하여 10회 측정하였다.

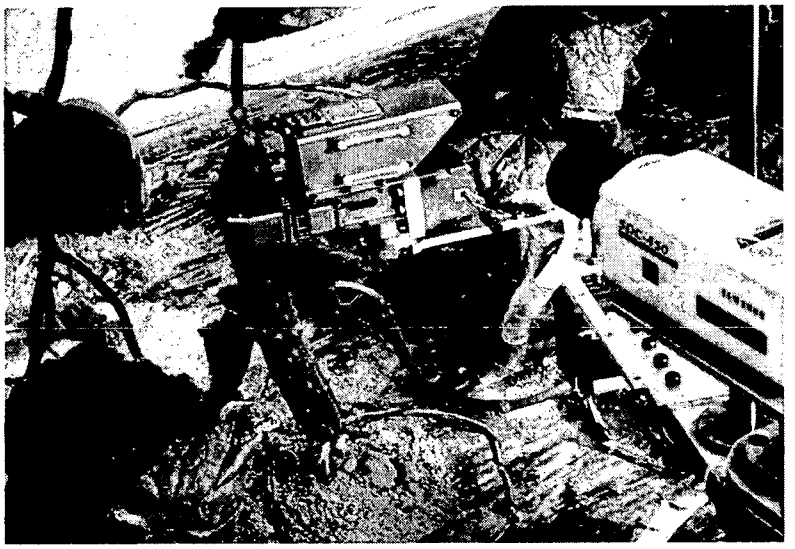
<Table 7-1> Experimental design

rpm / Diameter / Position	50 : 1			100 : 2			150 : 3		
	Over 5.5mm: 1	4.5~5.5 mm: 2	Under 4.5mm: 3	Over 5.5mm: 1	4.5~5.5 mm: 2	Under 4.5mm: 3	Over 5.5mm: 1	4.5~5.5 mm: 2	Under 4.5mm: 3
30cm : **A**	1-1-A	1-2-A	1-3-A	2-1-A	2-2-A	2-3-A	3-1-A	3-2-A	3-3-A
60cm : **B**	1-1-B	1-2-B	1-3-B	2-1-B	2-2-B	2-3-B	3-1-B	3-2-B	3-3-B
90cm : **C**	1-1-C	1-2-C	1-3-C	2-1-C	2-2-C	2-3-C	3-1-C	3-2-C	3-3-C

나. 현장실험

종합시스템의 현장실험을 위해서 경기도 수원시 탑동 소재 원예연구소를 방문하여 작업성능실험을 수행하였다. 종합시스템의의 성능은 3차원 공간에서 재배되고 있는 오이의 좌표를 영상처리에 의해 인식하고 매니퓰레이터의 컨트롤러에 기억되어 제작한 엔드이펙터를 수확 가능한 위치로 이동시키는 것이다. 이러한 작업을 반복하면서 오이와 엔드이펙터의 위치에러를 측정 기록하고, 기록된 데이터를 분석하는 방법으로 실험하였다. 오이를 3차원 공간에서 위치변

환을 하며 수확률이 높은 지점을 측정하고자 하였다. 실험을 통해서 엔드이펙터가 오이 과경에 어느 정도의 오차로 접근하는지를 버니어켈리퍼스를 이용하여 과경과 엔드이펙터의 간격을 측정하였다. <Fig. 7-6>은 현장에서 종합시스템의 작업관경을 나타낸 사진이다.

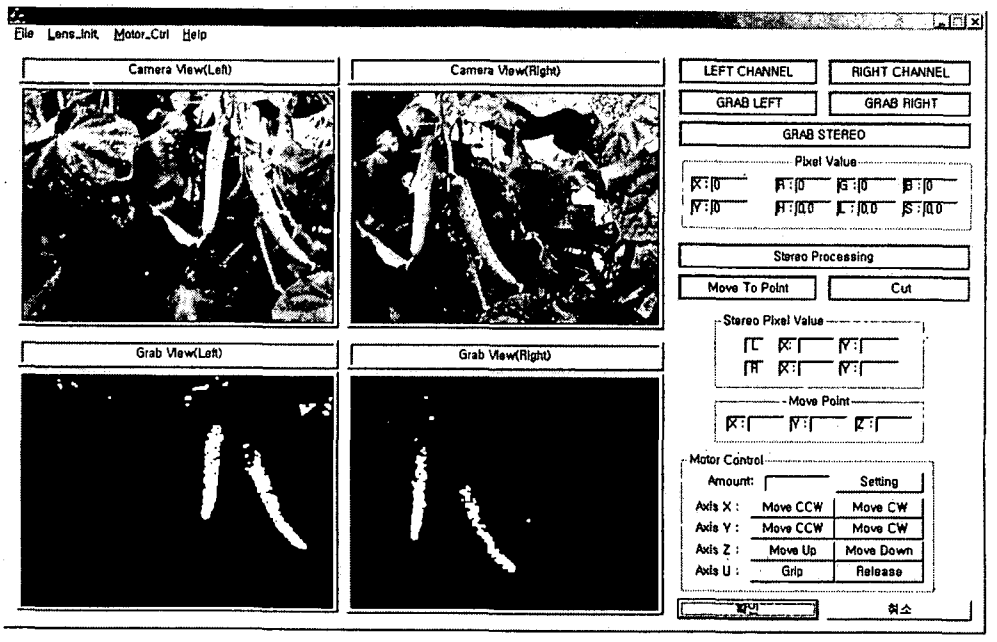


<Fig. 7-6> Operating on field

제 3 절 결과 및 고찰

1. 영상처리

<Fig. 7-7>는 현장에서 연상처리 된 결과를 나타낸 그림이다. 현장에서의 조명의 영향은 온실내부인 관계로 크기 않았으며, 오이의 과경 인식에는 큰 어려움이 없었다.



<Fig. 7-7> Program for the cucumber harvesting

2. 수확성능

개발한 오이수확기 종합시스템을 이용하여 원예 연구소에서 임의의 50개의 오이에 대해서 수확실험을 하였다. 그 결과 <Table 7-2>와 같이 나타났다.

<Table 7-2> The number of harvested cucumber

Total cucumbers	Harvested completely	Harvested incompletely	Did not be harvested
50	42	6	2
Rate(100%)	84 %	12 %	4 %

위의 결과에서 살펴보면 전체 수확을 시도한 오이 중에서 42개는 수확이 가능하였으며, 6개는 수확 적기가 끝난 시점이기 때문에 기형과에 의한 파지 및 절단이 불충분한 결과이었으며, 2개에 대해서는 잎과 줄기 등에 의한 인식의 불충분 및 파지와 절단등이 복합적으로 작용하여 수확이 불가능하였다.

3. 반복 위치에러 측정 및 정확성

매니퓰레이터의 반복오차를 측정하기 위하여 카메라 영상의 원점을 기준으로 X, Y축으로 -100mm, 0mm, 100mm, Z축으로 -200mm, 0mm, 200mm이동하면서 각 실험구 마다 10회 반복 측정하였다. <Table 7-3>은 카메라 원점에서 Z축으로 -200mm 이동했을 때 10회 반복오차이다. 오차의 측정은 카메라를 이용하여 엔드이펙터의 끝단과 모형 오이 과경의 중심간 거리를 Visual C++ 프로그램을 이용하여 계산하는 방법과 버어니어켈리퍼스를 이용하여 측정하는 방법을 병행하였다.

<Table 7-3> -200mm translation form Z axis < unit : mm >

Y axis	X axis / Travel / Replication error	X								
	Travel	-100			0			100		
		x	y	z	x	y	z	x	y	z
Y	-100	0.08	0.12	0.12	0.08	0.08	0.08	0.08	0.06	0.07
	0	0.07	0.07	0.09	0.02	0.06	0.04	0.07	0.06	0.04
	100	0.05	0.03	0.03	0.04	0.04	0.03	0.05	0.05	0.03

<Table 7-4> Zero offset of Z axis < unit : mm >

Y axis	X axis / Travel / Replication error	X								
	Travel	-100			0			100		
		x	y	z	x	y	z	x	y	z
Y	-100	0.09	0.08	0.12	0.08	0.06	0.09	0.07	0.07	0.07
	0	0.07	0.07	0.09	0.02	0.07	0.04	0.07	0.07	0.06
	100	0.06	0.03	0.07	0.04	0.06	0.03	0.05	0.07	0.03

<Table 7-5> 200mm translation form Z axis < unit : mm >

Y axis	X axis / Travel / Replication error	X								
	Travel	-100			0			100		
		x	y	z	x	y	z	x	y	z
Y	-100	0.07	0.08	0.12	0.07	0.04	0.09	0.06	0.06	0.05
	0	0.06	0.06	0.09	0.06	0.08	0.04	0.07	0.09	0.05
	100	0.04	0.05	0.07	0.04	0.04	0.06	0.04	0.05	0.06

<Table 7-4>는 카메라의 Z축으로 원점일 때 10회 반복 측정 오차이며, 표 3은 카메라 원점으로부터 Z축으로 200mm 이동했을때의 10회 반복 오차를 나타내고 있다. <Table 7-3><Table 7-4><Table 7-5>을 통하여 알 수 있는데 매니퓰레이터의 10회 반복 위치 오차는 0.1mm 내외로 정확하게 작동하는

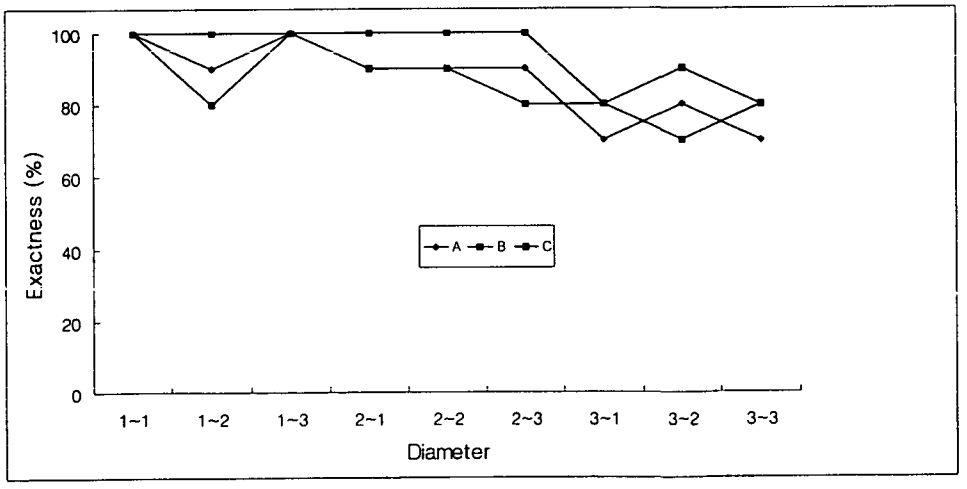
것으로 나타났다.

실내에서 종합시스템의 작업기가 오이 과경에 정확하게 접근하는가를 알아보기 위해서 실험한 결과 <Table 7-6>과 같은 결과를 얻었다. 정확성은 오이 과경을 기준으로 하여 매니퓰레이터가 작동하여 엔드이펙터가 과경으로 부터 1mm내에 있으면 성공한 것으로 보았다.

<Table 7-6> Exact approaching to the cucumber

rpm / Diameter / Position	50 : 1			100 : 2			150 : 3		
	Over 5.5mm: 1	4.5~5.5 mm: 2	Under 4.5mm: 3	Over 5.5mm: 1	4.5~5.5 mm: 2	Under 4.5mm: 3	Over 5.5mm: 1	4.5~5.5 mm: 2	Under 4.5mm: 3
30cm : A	100	90	100	90	90	90	70	80	70
60cm : B	100	100	100	100	100	100	80	90	80
90cm : C	100	80	100	90	90	80	80	70	80

실험을 통하여 오이의 결과 위치가 지면으로부터 60cm인 경우 50에서 100rpm으로 작동할 때 가장 정확하게 오이 과경에 접근하는 것을 알았다.

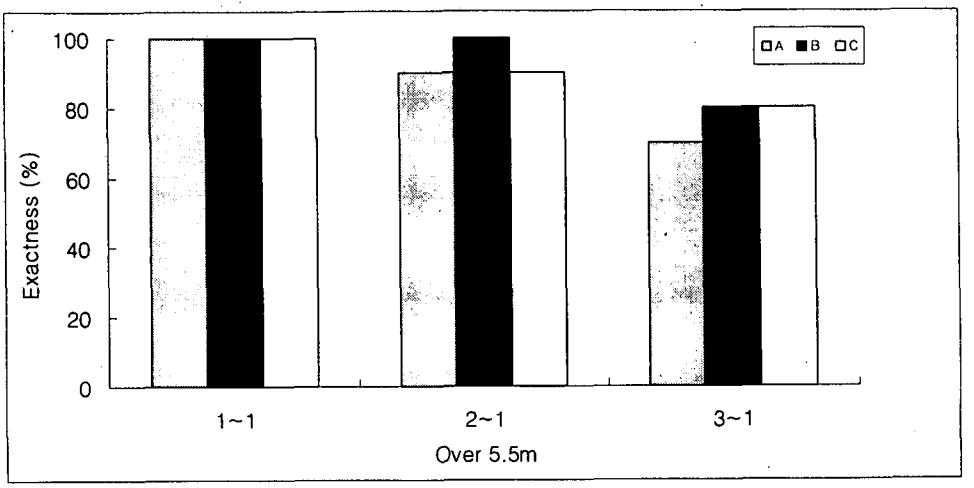


<Fig. 7-8> Exactness of approaching to the cucumber

<Fig. 7-8>에서 1-1-A, 1-2-B, 1-3-C의 경우에 100%의 정확성을 보였다. 그러나 3-1-A와 3-3-A 그리고 3-3-C의 경우에는 모터 회전수가 빠르기 때문에 70%로 낮게 나타난 것으로 판단된다.

4. 오이 과경과 모터 회전수의 관계

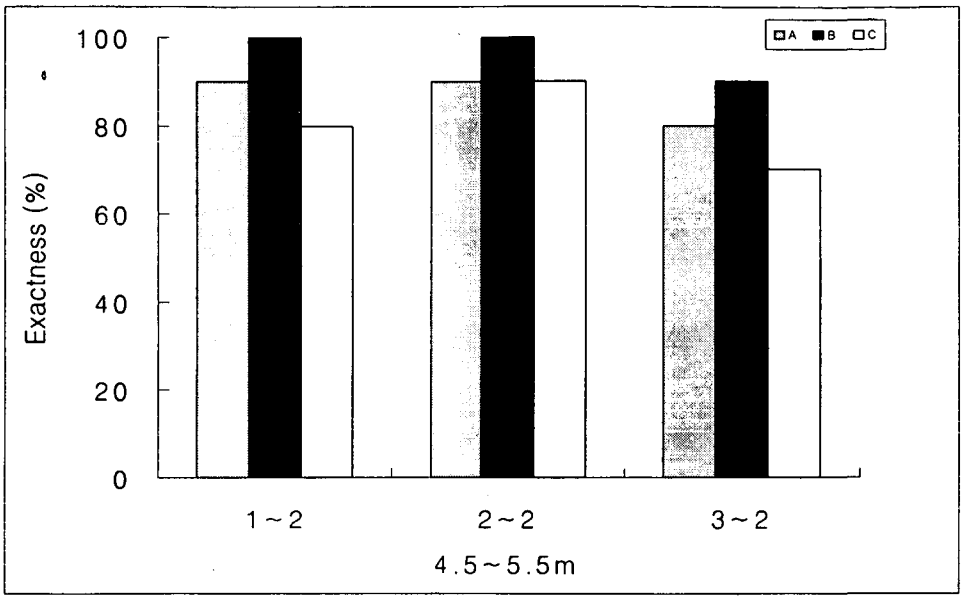
오이 과경에 따른 엔드이펙터의 크기와 모터 회전수를 결정하기 위하여 오이 과경 5.5mm와 모터 회전수의 관계를 통하여 오이 과경에 엔드이펙터가 가장 정확하게 접근하는 것을 알아보았다.



<Fig. 7-9> Exactness on diameter 5.5mm

<Fig. 7-9>에서 1-1-A, 1-1-B, 1-1-C, 2-1-B의 경우 100%의 정확성을 보였다. 그러나 2-1-A와 2-1-C의 경우 90%의 정확성을 보인 반면 3-1-B, 3-3-C는 80%, 3-1-A는 70%로 나타났다. 3-1의 경우 정확성이 낮게 나타난 것은 모터의 회전수가 빠르기 때문에 오이 과경을 벗어났기 때문이다. 1-1의 경우에는 모터의 회전수가 50rpm으로 늦은 속도에 의해서 오이과경에서 정확하게 정지했다. 오이 과경 4.5~5.5mm와 모터 회전수의 관계를 통하여 오이

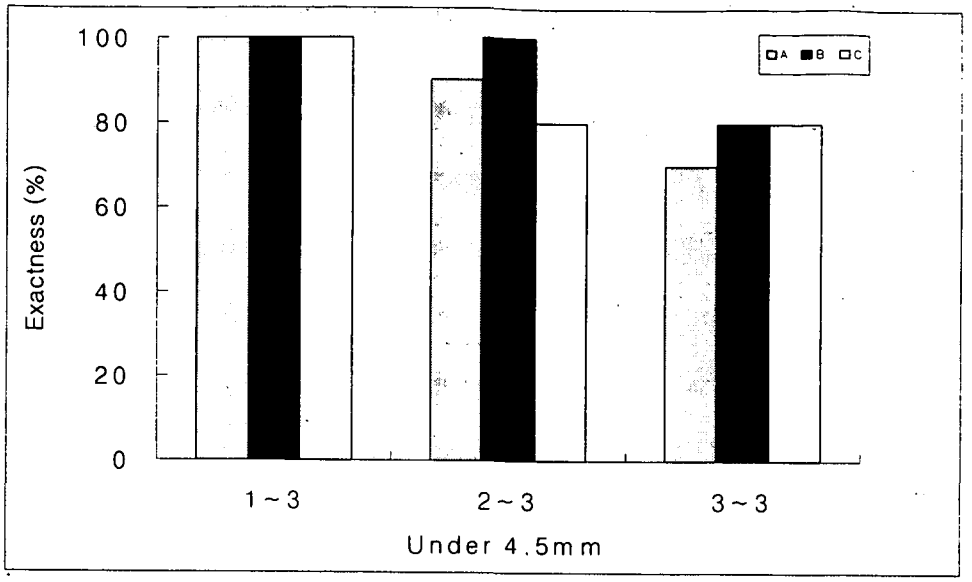
과경에 엔드이펙터가 가장 정확하게 접근하는 것을 알아보았다.



<Fig. 7-10> Exactness on diameter 4.5~5.5mm

<Fig. 7-10>에서 1-2-B, 2-2-B의 경우 !00%의 정확성을 나타냈다. 1-2-C는 80%, 3-2-C는 70%로 낮은 정확성을 나타냈다. 이것은 매니퓰레이터가 작동하는데 있어 지면이 일정하지 않기 때문에 지면으로부터 90cm 떨어진 경우에 작동범위를 벗어나서 발생된 것이다.

오이 과경 4.5mm와 모터 회전수의 관계를 통하여 오이 과경에 엔드이펙터가 가장 정확하게 접근하는 것을 알아보았다.

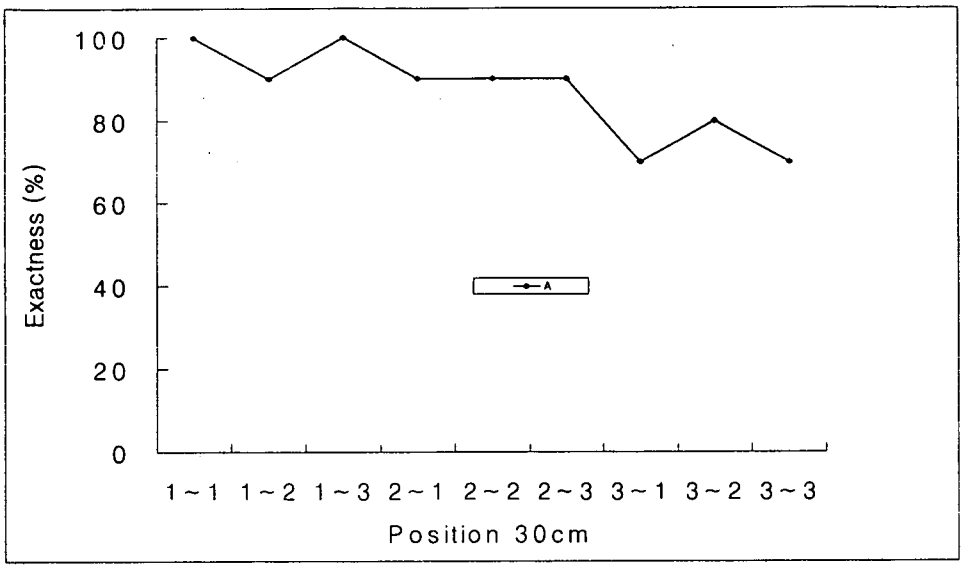


<Fig. 7-11> Exactness under diameter 4.5mm

<Fig. 7-11>에서 1-3-A, 1-3-B, 1-3-C, 2-3-B의 경우 100%의 정확성을 나타냈다. 3-3-A는 70%, 3-3-B, 3-3-C의 경우는 80%로 낮게 나타났다. 실험을 통하여 3-3의 경우 낮은 정확도를 나타낸 것은 모터의 회전수 가 빠르기 때문으로 판단된다.

5. 결과높이가 다른 오이 과경과 모터 회전수와의 관계

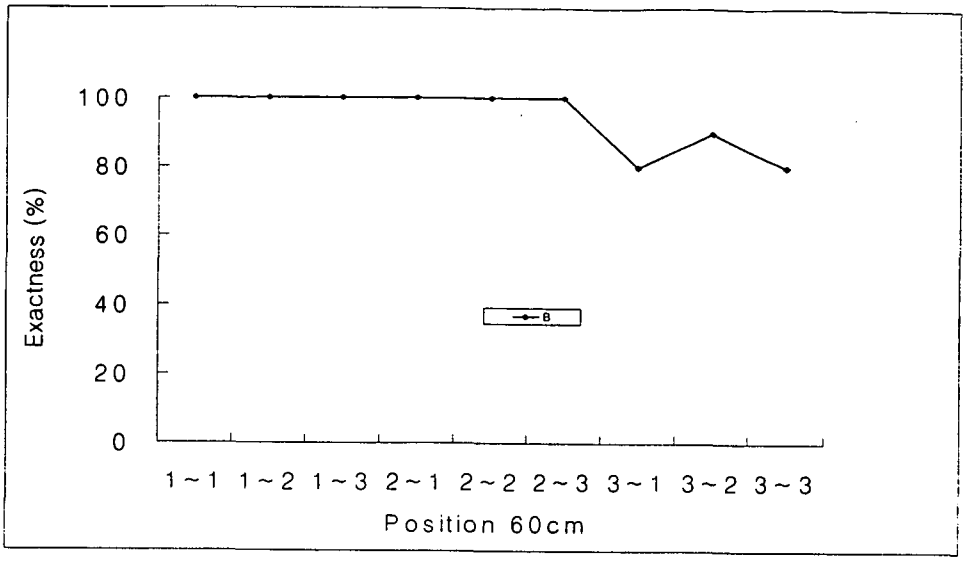
지면으로부터 30cm에 결과한 오이에 대하여 50, 100, 150rpm의 모터 회전수와의 관계를 알아보았다.



<Fig. 7-12> Exactness of approaching to 30cm height

<Fig. 7-12>에서 1-1-A, 1-1-3의 경우 100%의 정확성을 나타냈다. 3-1-A, 3-3-A의 경우 70%의 정확성을 나타냈다. 이것은 모터의 회전수가 빠르고 매니퓰레이터의 작동 범위를 벗어나기 때문에 나타난 것이다.

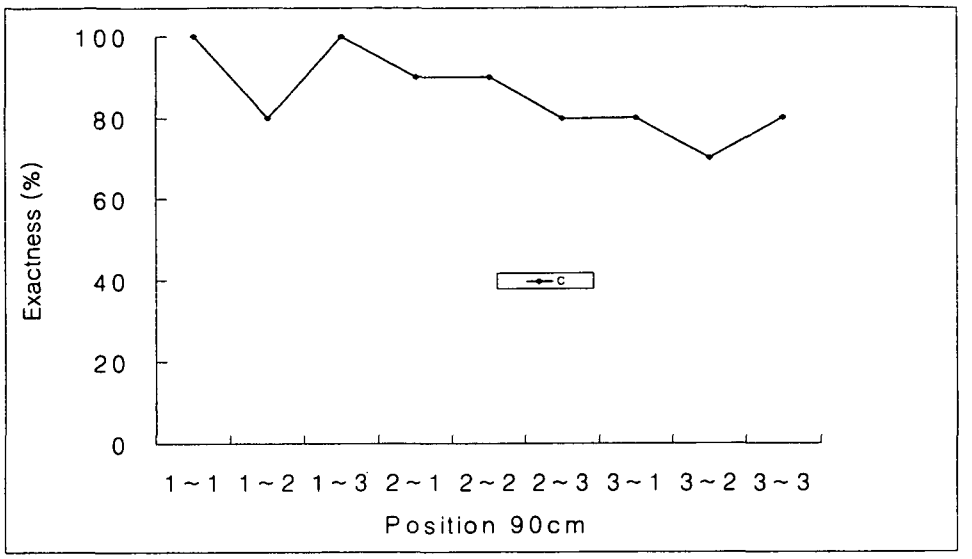
지면으로부터 60cm에 결과한 오이에 대하여 50, 100, 150rpm의 모터 회전수와의 관계를 알아보았다.



<Fig. 7-13> Exactness of approaching to height 60cm

<Fig. 7-13>에서 1-1-B, 1-2-B, 1-3-B, 2-1-B, 2-2-B, 2-3-B의 경우 100%의 정확성을 나타냈다. 3-1-B, 3-2-B, 3-3-B의 경우 80%, 90%, 80%의 정확성을 가지고 엔드이펙터가 오이 과경에 접근했다. 실험을 통하여 결과 높이가 60cm에서 가장 높은 정확성을 나타냈다. 3-1-B, 3-2-B, 3-3-B의 경우에는 매니퓰레이터가 모터의 회전수의 영향을 받는 것으로 나타났다.

지면으로부터 90cm에 결과한 오이에 대하여 50, 100, 150rpm의 모터 회전수와의 관계를 알아보았다.



<Fig. 7-14> Exactness of approaching to height 90cm

<Fig. 7-14>에서 1-1-C, 1-3-C의 경우에는 100%의 성공률을 보였다. 3-2-C의 경우 70%로 낮은 정확도를 나타냈다. C의 경우 지면으로부터 90cm에서 오이가 결과하기 때문에 매니퓰레이터의 작동하는 범위를 벗어나 정확성이 낮게 나타났다.

6. 매니퓰레이터의 문제점 분석 및 적합성

오이의 경우 지면에서 1m내외에서 결과하기 때문에 실험에서 30cm, 60cm, 90cm로 실험을 하였다. 온실의 지면이 일정하지 않기 때문에 경사각이 5° 정도 발생하였다. 이로 인해 매니퓰레이터가 오이 과경으로 접근하는데 있어 30cm, 90cm인 경우에는 정확도가 떨어졌다. 정확하게 오이 과경에 접근하기 위해서는 매니퓰레이터가 작동하기 전에 수평을 만들 수 있는 장치가 필요하다.

매니퓰레이터가 목표위치에 도달되는 시간은 그 목표로 하는 위치에 따라

다르다. 하지만 각 모터의 정격부하시 회전수에 따라 영향을 받는 것으로 나타나는 것으로 측정되었다. 실험을 통하여 최적의 모터 회전수는 50rpm이었으며, 오이의 결과 높이에 따라서 영향이 있었다. 오이가 결과는 높이가 60cm인 경우에는 모터 회전수가 50~100rpm이 최적의 결과를 얻을 수 있었다. 또한, 작업공간 안에서 한점에서 다른 한점으로 이동하는데 걸리는 최대 시간은 3초 미만이다. 이것은 매니퓰레이터의 상승 하강 시간 때문이다.

제 4 절 요약 및 결론

오이 수확을 위한 종합시스템의 설계 제작하였으며, 이를 이용하여 현장에서의 오이 수확실험을 수행하였다. 또한 각 구성요소 중에서 수확에 직접 영향을 주는 매니퓰레이터의 성능을 검증하였다. 작업기는 오이 수확에 적합한 링크구조와 메커니즘를 설계하였으며, 본 매니퓰레이터는 DC모터로 작동되는데, 작동원리는 퍼지논리 이론을 적용하여 부드러운 작동이 가능하도록 하였다. 그 결과 다음과 같은 결론을 얻을 수 있었다.

1. 위의 결과에서 살펴보면 전체 수확을 시도한 오이 중에서 42개는 수확이 가능하였으며, 6개는 파지 및 절단이 불충분한 결과이었으며, 2개에 대해서는 잎과 줄기 등에 의한 인식의 불충분 및 파지와 절단등이 복합적으로 작용하여 수확이 불가능하였다.

2. 각 축을 제어함에 있어 동력원으로 모터를 사용하였다. 모터는 제어가 용이한 장점이 있다. 본 연구에서는 퍼지제어 이론을 적용 작업환경에 맞게 rule base를 작성하여 대응하도록 한 결과 비교적 정확한 위치제어가 가능하였다.

4. 3차원 공간상의 좌표에 대하여 매니퓰레이터의 10회 반복 위치 오차는 Z축에 관계없이 0.1mm 내외로 정확하게 작동하는 것으로 나타나 수확작업에 적합한 것으로 나타났다.

5. 오이의 결과 위치가 지면으로부터 60cm인 경우 50에서 100rpm으로 작동할 때 가장 좋은 작업성능을 나타냈다. 모터의 회전수가 느릴수록 오이 과경에 접근하는 정확성이 높았다. 실험을 통하여 오이의 결과 높이에 따라서 모터의 회전수에 변화를 주어야 된다.

참고문헌

[1] Aura, E., 1983, Soil Compaction by the tractor in spring and its effect on soil porosity. Journal of Scientific Agricultural Society of Finland, Vol. 55

[2] Ayers, P. D., 1987. Moisture and density effects on soil shear strength parameters for coarse grained soils. Transaction of the ASAE, Vol. 30, No. 5, 1282~1287

[3] Bang W. Lee., Bing J. Sheu. 1991. Modified Hopfield Neural Networks for Retrieving the Optimal Solution. IEEE Trans. Neural Network. Vol. 8(1);75~83

[4] Chulhee Lee., David A. 1997. Decision Boundary Feature Extraction for Neural Networks. IEEE Trans. Neural Network. Vol.8(1);75~83

[5] Clarke, P. T. 1985. Automatic break up of pork carcasses. ASAE and SMS, Proceedings of the Agri-Mation I Conference & Exposition. 173~182

[6] Claus Neubauer. 1998. Evaluation of Convolutional Neural Networks for Visual Recognition. IEEE Trans. Neural Network, Vol.9(4);685~696

[7] Culshaw, D., 1988. Rubber tracks for traction. Journal of Terramechanics, Vol. 25, No. 1, 60~80

[8] Devlaemink, R. M. 1985. vision systems and robotics in food processing. ASAE and SME, Proceedings of the Agri-Mation I Conference & Exposition. 27~37

[9] Dwyer, M. J. and J. A. Okello and A. J. Scarett, 1993, A theoretical and experimental investigation of rubber tracks for agriculture. Journal of Terramechanics, Vol. 30, No. 4, 285~298

[10] Ekwue, E. I., R. J. Stone. 1995. Organic matter effects on the strength

properties of compacted agricultural soils. Transaction of the ASAE. vol. 38(2):325~332

[11] Frost. A. R., T. T. Mottram, M. J. Street, R. C. Hall, D. S. Spencer, C. J .Allen, 1993. A Field of a Teatcup Attachment Robot for an Automatic Miking System, J. agric. Engng Res 55. p.325~334

[12] Fu K. S., R. C. Gonzalez, C. S. G. Lee, Robotics ; Control, Sensing vision, and Intelligence, McGraw-Hill Book Company.

[13] Gopalasamy Athithan., Chandan Dasgupta. 1997. On the Problem of Spurious Patterns in Neural Associative Memory Models. IEEE Trans. Neural Network. Vol.8(6);1483~1491

[14] Hagan., Demuth., Beale. 1996. Neural Network Design. PWS Press.

[15] Harrell, R. C., P. D. Adsit. 1990. The Florida Robotic Grove-Lab. Trans of the ASAE Vol.8(33);391~399

[16] Henry A. Rowley. 1998. Neural Network-Based Face Detection. IEEE Trans. on Pattern Analysis and Machine Intelligence. Vol.20(1);23~38

[17] Hoy, Roger Michael, 1986. A Unique Hollow Finger Gripper Designed for Agricultural Robots, M.S. Thesis, Department of Biological and Agricultural Engineering, North Carolina State University, Raleigh, NC. 12 ~40

[18] Hwang, H., F. E. Sistler. 1985. The implementation of a robotic machine. ASAE and SME, Proceedings of the Agri-Mation I Conference & Exposition. 173~182

[19] Japanese Society of Agricultural Machinery, Proceedings of International Symposium on Automation and Robotics in Bioproduction and Processing, Vol. 3 , November 3~6, 1995

[20] John J. Craig , Introduction to Robotics ; Mechanics and Control, Addison Wesley, 1986

[21] Kah Kay. Sung., Tomaso Poggio. 1998. Example-Based Learning for View-Based Human Face Detection. IEEE Trans. on Pattern Analysis and Machine Intelligence. Vol.20(1);39~50

[22] Key, S. J. 1985. Productivity modeling and forecasting for automated shearing machinery. ASAE and SME, Proceedings of the Agri-Mation I Conference & Exposition. 200~209

[23] KIM Ki Dae, Saigehisa OZAKI Takayuki KOJIMA. Development of an automatic robot system for a vegetable factory. 1995. JSAM Vol. 1, November 3~6, 157~164

[24] Kutz, L. J., G. E. Miles, P. A. Hammer and G. W. Krutz. 1987. Robotic transplanting of bedding plants. Trans. of the ASAE 30(3). 586~590

[25] Lea, Dae-Weon, 1990. A Robotic and Vision System for Locating and Transferring Container Grown Tabacco Seedling, Ph. D THesis, Department of Biological and Agricultural Engineering, North Carolina State University, Raleigh, NC.

[26] Lee, Dae-Weon, W. F. McClure , 1993. A Robotic System for Transferring Tobacco Seedling, International Conference for Agricultural Machinery and Process Engineering, Vol. 3, 850 ~ 858

[27] Masaru HIFA, Seiji NAKAO, Toshio IWAO. Vision sensor for cherry tomato harvesting robot. 1995. JSAM Vol. 1, November 3~6, 13~20

[28] Mikell P. Groover, Mitchell Weiss, Roger N. Nagel , Nicholas G. Odrey, Industrial Robotics; Technology, Programming, and Applications, McGraw-Hill Book Company, 1986

[29] Mohaphatra, S. C., Dae-Weon Lee, W. F. McClure, 1990. 「Optimization of Plant Tissue Culture System, Biomedical Engineering: Opening New Doors」, New York University, New York University Press, 111 ~ 114

[30] Muro, T., 1993. Tractive and Braking performance of a flexible tracked tractor moving up and down weak sloped terrain. Jounal of Terramechanics, Vol. 32, No. 5, 245~261

[31] Naoshi KONDO, Yoshiaki NISHITSUJI, Yasunori SHIBANO, Kentaro MOHRI, Mitsuji MONTA, Hisaya YAMADA. 1995. Visual feedback control of pretty-tomato harvesting robot. JSAM Vol. 1, November 3~6, 181~187

[32] Naoshi. Kondo., Nishitsuji. Y., K. C. Ting. 1996. Visual Feedback guided tomato harvesting. Trans. of the ASAE. Vol.39(6);2331~2338.

[33] Naoshi. Kondo., Y. Shibano. 1993. Studies on Visual Sensor for Cucumber Fruit Detection(1). Environ. Control Biol. Vol.31(2);93-100 .

[34] Okello, J. A., 1994. Prediction and experimental validation of the feild tractive performance of a rubber track unit. Journal of Agricultural Engineering Research, Vol. 59, No. 2, 163~171

[35] Okello, J. A., 1994. The tractive performance of rubber tracks and tractor driving wheel type as influenced by design parameters. Journal of Agricultural Engineering Research, Vol. 59, No. 2, 33~43

[36] Ollis, M. A. Stenz. 1997. Vision-based Perception for an Automated Harvester. In Proceeding of the International Conference on Robotic System. 1838~1844

[37] Paul B. Watta., Kaining Wang. 1997. Recurrent Neural Nets as Dynamical Boolean Systems with Application to Associative Memory. IEEE Trans. Neural Network. Vol.8(6);1268~1280

[38] Pla, F., F.Juste, F. 1993. Color Segmentation based on a Light Reflection Model to Locate Citrus Fruits for Robotic Harvesting. Comput. Electron. Agric. Vol.9(3):53~70

[39] R. C. Gonzalez., R. E. Woods. 1992. Digital image processing.

Addison Wesley Inc.

[40] Seiichi ARIMA, Naoshi KONDO, Tateshi FUJIURA, Hiroshi NAKAMURA, Jun YAMASHITA. 1995. Basic studies on cucumber harvesting robot. JSAM Vol. 1, November 3~6, 195~202

[41] Sevila, F. 1985. A robot to prune the grapevine. ASAE and SME, Proceedings of the Agri-Mation I Conference & Exposition. 190~199

[42] Shigehiko HAYASHI, Osamu SAKAUE, Mituho SUGIMOTO. 1995. Application of robotics for vegetable production. JSAM Vol. 1, November 3~6, 189~194

[43] Shimizu. H., R. D. Heins. 1995. Computer vision based system for plant growth analysis Trans. of the ASAE. Vol.38(3);959~964

[44] Steve Lawrence. 1997. Face Recognition: A Convolutional Neural Network Approach. IEEE Trans. Neural Network. Vol.8(1);98~113.

[45] Susan. S., Young. 1998. Foveal automatic target recognition using a multiresolution neural network. IEEE Trans. on image processing, Vol.7 (8);1122~1135

[46] Ting, K. C., G. A. Giacomelli. 1988. In the greenhouse : robot automates plug transplanting. American Vegetable Grower, Volume 6(12) : 44~48

[47] Ting, K. C., G. A. Giacomelli. Shen, S. J., W. P. Kabala. 1988. End-effector development for robotic transplanting of seedlings. American Society of Agricultural Engineers, Paper No. 84~1544

[48] Todd Law., Hidenori Ioth. 1996. Image Filtering, Edge Detection and Edge tracing using Fuzzy reasoning. IEEE Trans. on Pattern Analysis and Machine Intelligence. Vol.18(5);481~491

[49] Virginio Cantonio., Stefano Levialdi., Vito Roberto.(eds) 1997. Artificial Vision. Academic Press.

[50] William Kleitz. Digital Electronics. 1996

[51] Wong, J. Y., 1984. On the study of wheel-soil interaction. Journal of Terramechanics, Vol. 21, No. 2, 117~131

[52] Wong. J. Y. and Preston-Thomas, J., 1986. Parametric analysis of tracked vehicle performance using an advanced computer simulation model. Proceedings of the Institution of Mechanical Engineers. Vol. 200(D2), No. 60, 101~114

[53] Wong. J. Y. and Preston-Thomas, J., 1988. Investigation into the effects of suspension characteristics and design parameters on the performance of tracked vehicles using an advanced computer simulation model. Proceedings of the Institution of Mechanical Engineers. Vol. 202(D3), No. 53, 143~161

[54] Wong. J. Y., 1989. Terramechanics and off-road vehicles. Elsevier Publishers.

[55] 고보연. 1996. 학습 패턴의 분포 특징을 고려한 다층퍼셉트론 신경회로망의 개선. 정보과학회논문지. Vol.23(9);963~971

[56] 권기영, 이민수. 1990. Hamming Net의 충돌 회피를 위한 Priority 적용에 관한 연구. 정보과학회논문지. Vol.17(1);149~152

[57] 김상엽, 문종섭. 1996. 양방향 연상 메모리의 개선된 학습 방법. 정보과학회 논문지. Vol.23(3);280~286

[58] 김태윤,김홍복, 1991.산업용 로봇,도서출판 생능 p.36~49

[59] 노상하 외. 1991. 영상처리 장치를 이용한 사과의 색택 판정. 한국농업기계학회지 Vol.16(3);272~280

[60] 류관희, 김기영, 이희환, 박정인, 1997. 육묘용 로봇 이식기의 개발(Ⅱ), 한국농업기계학회지 제22권 제3호. 325~332

[61] 류관희, 김기영, 이희환, 황호준, 1997. 육묘용 로봇 이식기의 개발(Ⅰ), 한국농업기계학회지 제22권 제3호. 317~324

[62] 류관희, 조성인, 황 헌, 최중섭, 1996. 생물생산을 위한 지능로봇공학, 문운당

[63] 박선호, 1991. FA 센서 응용백과 1, 영진출판사

[64] 손재룡 외. 1997. 스테레오영상처리를 이용한 토마토 위치검출알고리즘 개발. 한국농업기계학회 논문집. 409~414

[65] 양현승. 1992. 신경회로망과 컴퓨터 비젼. 정보과학회지. Vol.10(2);39~48

[66] 우운택, 정홍. 1991. 신경회로망을 이용한 매니퓰레이터의 시각제어. 한국정보과학회 논문지. Vol.18(2);174~183

[67] 이근철, 1994. 메카트로닉스를 위한 서보 기술 입문, 기전연구사

[68] 이근철, 이형수, 1998. 마이크로컴퓨터에 의한 모터의 제어,기전연구사, p219 235

[69] 이수희,노상하. 1998. 기계시각을 이용한 후지사과의 색 측정 .한국농업기계학회 논문집. 374~379

[70] 이종호, 박승제, 이중용, 1997. 고추수확기의 탈실장치 개발(Ⅰ), 한국농업기계학회지 제22권 제2호. 177~188

[71] 이충호. 1995. 컴퓨터 시각에 의한 범표고의 외관검색 및 자동선별시스템 개발. 성균관대학교 농업기계공학과 박사학위 논문

[72] 장익주, 김태한, 권기영, 1997. 사과 수확 로봇의 핸드 개발(Ⅰ), 한국농업기계학회지 제22권 제4호. 411~420

[73] 장익주,김태한,유경선. 1998. 사과수확 로봇의 개발(Ⅱ)-화상처리를 이용한 사과의 인식알고리즘 .한국농업기계학회 논문집. 481~486

[74] 정재영, 김문현. 1998. 다중 물체가 움직이는 동영상에서 자동적인 특징점 추출 및 추적. 정보과학논문지. Vol.25(3);562~579

[75] 진상호, 권준박, 1995. 로보트 제어 입문, 원창출판사

[76] 최용, 박환중, 홍종태, 전현종, 윤무경, 1998. 무·당근 수확기 개발, 한국농업기계학회 동계학술대회 논문집, Vol.3 No.1 49~55

[77] 최진영, 박현주. 1995. 신경회로망을 이용한 시스템 모델링 및 제어. 제어

자동화시스템공학회지. Vol.1(3)62～73

[78] 한국기계연구원, 1996. 6축 DD Robot 개발, 과학기술처 특정연구개발사업연구보고서

[79] 황 헌. 1991. 인공신경회로망 기술과 응용. 한국농업기계학회지. Vol. 16(1) : 90～99

[80] 황헌, 1990.생물자원 생산자동화 시스템과 로보틱스 기술, 농업생산시스템의 자동화와 첨단기술: 한국농업기계학회 심포지움 발표문, 149～182

[81] 近藤 直, 1996. 農業用ロボット開發の課題と展望(2) - ロボットハンドの研究開發の課題と展望, 日本農業機械學會誌, 58(1), 139-144

[82] 德田 勝, 並河 淸. 1996. 畵像處理によるスイカ果實の識別. 農業機械學會誌. Vol.57(2):13～20

[83] 日本工業技術센터. 1995. 컴퓨터 畵像處理入

[84] 張 樹魂 外. 1998. 赤外線熱畵像によるリンコの檢出に關する研究. 農業機械學會誌. Vol.60(1):69～76

[85] 竹田洋志, 並河 淸. 1998. スイカ果實識別のための畵像處理による標識の檢出. 農業機械學會誌. Vol.60(1):77～83

[86] 川 材 貞 夫, 1996. 로봇 제어 입문, 성안당

[87] 淸水 活, 大下誠一 外. 1991. 畵像處理による3次元曲線長さの非接触計測. 農業機械學會誌. Vol.53(4):85～92

[88] 淸水 活. 1998. 畵像計測による植物体3次元長さの計測. 農業機械學會誌. Vol.60(1):139～142

[89] 淸水 活. 大下誠一. 1993. 植物3次元形狀の非接触計測システムの開發. 農業機械學會誌. Vol.55(2):93～100

[90] 行本 修, 1996. 農業用ロボット開發の課題と展望(2) - 農業用車兩のロボット化の課題と展望 , 日本農業機械學會誌, 58(1), 133-138

<부 록> 종합시스템 운영 프로그램

엔드이펙터 구동 프로그램

```
'
'          AN001. KEYPAD USER I/F
'
           DIM I AS BYTE
           DIM J AS BYTE
           DIM STPO AS BYTE              '시작점 저장변수
           DIM POS1 AS BYTE              '정지위치 1번 저장변수
           DIM POS2 AS BYTE              '정지위치 2번 저장변수
           DIM SEL AS BYTE
           DIM M AS BYTE

           DIM N AS BYTE
           SEL=1
           M=0
           N =0


           STPO=EEREAD(&HFF0)                    '초기값 설정
           POS1=EEREAD(&HFF1)
           POS2=EEREAD(&HFF2)


           LCDINIT

           OUT 7,1

'          ====       원점 복귀 ===

5          I=ADIN(4)
           LOCATE 0,1
           PRINT DEC(I)
           IF I>STPO THEN
                    GOTO 5
           END IF
           OUT 7,0

           === LCD 초기화 ===
```

```
        LOCATE 0,0
        PRINT "PUSH START OR"
        LOCATE 1,1
        PRINT "SELECT BUTTON"

        === 작동 루푸 ===

ROOP:   I=KEYIN(14,100)
        IF I=0 THEN
                SEL=SEL+1
                M=1
        END IF

        === 스위치 입력값 처리 함수 ===

        IF SEL=1 AND M=1 THEN
                LOCATE 0,0
                PRINT "START POSITION"
                LOCATE 0,1
                PRINT "                "
                LOCATE 3,1
                PRINT DEC(STPO)
                M=0
        END IF
        IF SEL=2 AND M=1 THEN
                LOCATE 0,0
                PRINT "POSITION 1
                LOCATE 0,1
                PRINT "                "
                LOCATE 3,1
                PRINT DEC(POS1)
                M=0
        END IF
        IF SEL=3 AND M=1 THEN
                LOCATE 0,0
                PRINT "POSITION 2      "
                LOCATE 0,1
                PRINT "                "
```

```
            LOCATE 3,1
            PRINT DEC(POS2)
            M=0
END IF

IF SEL>3 THEN
            SEL=1
END IF
I=KEYIN(13)

=== 증가 감소 스위치 입력처리 함수 ---

IF I=0 AND SEL=1 THEN
            STPO=STPO-1
            LOCATE 4,1
            PRINT DEC(STPO)
            EEWRITE &HFF0,STPO
END IF
IF I=0 AND SEL=2 THEN
            POS1=POS1-1
            LOCATE 4,1
            PRINT DEC(POS1)
            EEWRITE &HFF1,POS1
END IF
IF I=0 AND SEL=3 THEN
            POS2=POS2-1
            LOCATE 4,1
            PRINT DEC(POS2)
            EEWRITE &HFF2,POS2
END IF
I=KEYIN(12)
IF I=0 AND SEL=1 THEN
            STPO=STPO+1
            LOCATE 4,1
            PRINT DEC(STPO)
            EEWRITE &HFF0,STPO
END IF
IF I=0 AND SEL=2 THEN
            POS1=POS1+1
```

```
                LOCATE 4,1
                PRINT DEC(POS1)
                EEWRITE &HFF1,POS1
        END IF
        IF I=0 AND SEL=3 THEN
                POS2=POS2+1
                LOCATE 4,1
                PRINT DEC(POS2)
                EEWRITE &HFF2,POS2
        END IF

        === 작동시작 스위치 입력 처리함수 ===

        I=KEYIN(15)
        IF I=0 AND N=1 THEN
                LOCATE 0,0
                PRINT "NOW ACTIVE    "
                GOSUB OPPL
                N=0
        END IF

        IF I=0 AND N=0 THEN
                LOCATE 0,0
                PRINT "NOW ACTIVE    "

                GOSUB OPP
                N=1
        END IF


GOTO ROOP

        === 모터 위치제어구동 릴레이 ON / OFF 함수 =

OPP:
        OUT 6,1
20      I=ADIN(4)
        GOSUB DISP
        IF I<POS1 THEN
```

```
                GOTO 20
        END IF
        OUT 6,0
        FOR I=1 TO 125
        FOR J-1 TO 1
        NEXT J
        NEXT I
        OUT 6,1
30      I=ADIN(4)
        GOSUB DISP
        IF I<POS2 THEN
                GOTO 30
        END IF
        OUT 6,0

RETURN

OPPL:
        OUT 7.1
90      I=ADIN(4)
        GOSUB DISP
        IF I>STPO THEN
                GOTO 90
        END IF
        OUT 7,0
        LOCATE 0,1
        PRINT "END OPERATION"
RETURN

        --- LCD DISPLAY 함수 ---

DISP:
        LOCATE 0,1
        PRINT DEC(I)
RETURN
```

```
// BPPRN.cpp : implementation file
//

#include "stdafx.h"
#include "GraphProcess.h"
#include "BPPRN.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// BPPRN dialog


BPPRN::BPPRN(CWnd* pParent /*=NULL*/)
        : CDialog(BPPRN::IDD, pParent)
{
        //{{AFX_DATA_INIT(BPPRN)
        m_OutBp1 = 0.0;
        m_OutBp2 = 0.0;
        m_OutBp3 = 0.0;
        m_OutBp4 = 0.0;
        m_OutBp5 = 0.0;
        m_TgtBp1 = 0.0;
        m_TgtBp2 = 0.0;
        m_TgtBp3 = 0.0;
        m_TgtBp4 = 0.0;
        m_TgtBp5 = 0.0;
        m_INU = 0;
        m_SNO = 0;
        m_OutBp6 = 0.0;
        m_OutBp7 = 0.0;
        m_OutBp8 = 0.0;
```

```
        m_OutBp9 = 0.0;
        m_OutBp10 = 0.0;
        m_TgtBp6 = 0.0;
        m_TgtBp7 = 0.0;
        m_TgtBp8 = 0.0;
        m_TgtBp9 = 0.0;
        m_TgtBp10 = 0.0;
        //}}AFX_DATA_INIT
}


void BPPRN::DoDataExchange(CDataExchange* pDX)
{
        CDialog::DoDataExchange(pDX);
        //{{AFX_DATA_MAP(BPPRN)
        DDX_Text(pDX, IDC_EDIT1, m_OutBp1);
        DDX_Text(pDX, IDC_EDIT2, m_OutBp2);
        DDX_Text(pDX, IDC_EDIT3, m_OutBp3);
        DDX_Text(pDX, IDC_EDIT4, m_OutBp4);
        DDX_Text(pDX, IDC_EDIT5, m_OutBp5);
        DDX_Text(pDX, IDC_EDIT6, m_TgtBp1);
        DDX_Text(pDX, IDC_EDIT7, m_TgtBp2);
        DDX_Text(pDX, IDC_EDIT8, m_TgtBp3);
        DDX_Text(pDX, IDC_EDIT9, m_TgtBp4);
        DDX_Text(pDX, IDC_EDIT10, m_TgtBp5);
        DDX_Text(pDX, IDC_EDIT11, m_INU);
        DDX_Text(pDX, IDC_EDIT12, m_SNO);
        DDX_Text(pDX, IDC_EDIT13, m_OutBp6);
        DDX_Text(pDX, IDC_EDIT14, m_OutBp7);
        DDX_Text(pDX, IDC_EDIT15, m_OutBp8);
        DDX_Text(pDX, IDC_EDIT16, m_OutBp9);
        DDX_Text(pDX, IDC_EDIT17, m_OutBp10);
        DDX_Text(pDX, IDC_EDIT18, m_TgtBp6);
        DDX_Text(pDX, IDC_EDIT19, m_TgtBp7);
        DDX_Text(pDX, IDC_EDIT20, m_TgtBp8);
        DDX_Text(pDX, IDC_EDIT21, m_TgtBp9);
        DDX_Text(pDX, IDC_EDIT22, m_TgtBp10);
        //}}AFX_DATA_MAP
}
```

```
BEGIN_MESSAGE_MAP(BPPRN, CDialog)
        //{{AFX_MSG_MAP(BPPRN)
                // NOTE: the ClassWizard will add message map macros here
        //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// BPPRN message handlers



// BPSET.cpp : implementation file
//

#include "stdafx.h"
#include "GraphProcess.h"
#include "BPSET.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// BPSET dialog


BPSET::BPSET(CWnd* pParent /*=NULL*/)
        : CDialog(BPSET::IDD, pParent)

        //{{AFX_DATA_INIT(BPSET)
        m_WorkName = _T("");
        m_InpU = 0;
        m_OutU = 0;
        m_HidU = 0;
        m_HidLay = 0;
```

```
        m_Alpha = 0.0;
        m_Eta = 0.0;
        m_MaxPErr = 0.0;
        m_MaxErr = 0.0;
        m_SmpNo = 0;
        m_MaxEpo = 0;
        m_PrnDat = 0;
        m_PrnErr = 0;
        m_ErrFile = _T("");
        m_OutFile = _T("");
        m_PrnTest = 0;
        //}}AFX_DATA_INIT
}


void BPSET::DoDataExchange(CDataExchange* pDX)
{
        CDialog::DoDataExchange(pDX);
        //{{AFX_DATA_MAP(BPSET)
        DDX_Text(pDX, IDC_EDIT1, m_WorkName);
        DDX_Text(pDX, IDC_EDIT2, m_InpU);
        DDX_Text(pDX, IDC_EDIT3, m_OutU);
        DDX_Text(pDX, IDC_EDIT4, m_HidU);
        DDX_Text(pDX, IDC_EDIT6, m_HidLay);
        DDX_Text(pDX, IDC_EDIT7, m_Alpha);
        DDX_Text(pDX, IDC_EDIT8, m_Eta);
        DDX_Text(pDX, IDC_EDIT9, m_MaxPErr);
        DDX_Text(pDX, IDC_EDIT10, m_MaxErr);
        DDX_Text(pDX, IDC_EDIT5, m_SmpNo);
        DDX_Text(pDX, IDC_EDIT11, m_MaxEpo);
        DDX_Text(pDX, IDC_EDIT12, m_PrnDat);
        DDX_Text(pDX, IDC_EDIT13, m_PrnErr);
        DDX_Text(pDX, IDC_EDIT14, m_ErrFile);
        DDX_Text(pDX, IDC_EDIT15, m_OutFile);
        DDX_Text(pDX, IDC_EDIT16, m_PrnTest);
        //}}AFX_DATA_MAP
```

```
BEGIN_MESSAGE_MAP(BPSET, CDialog)
        //{{AFX_MSG_MAP(BPSET)
                // NOTE: the ClassWizard will add message map macros here
        //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// BPSET message handlers



// BPTST.cpp : implementation file
//

#include "stdafx.h"
#include "GraphProcess.h"
#include "BPTST.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// BPTST dialog


BPTST::BPTST(CWnd* pParent /*=NULL*/)
        : CDialog(BPTST::IDD, pParent)
{
        //{{AFX_DATA_INIT(BPTST)
        m_TrainedFile = _T("");
        m_TestedFile = _T("");
        m_TsmpNo = 0;
        //}}AFX_DATA_INIT
}
```

```
void BPTST::DoDataExchange(CDataExchange* pDX)
{
        CDialog::DoDataExchange(pDX);
        //{{AFX_DATA_MAP(BPTST)
        DDX_Text(pDX, IDC_EDIT1, m_TrainedFile);
        DDX_Text(pDX, IDC_EDIT2, m_TestedFile);
        DDX_Text(pDX, IDC_EDIT3, m_TsmpNo);
        //}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(BPTST, CDialog)
        //{{AFX_MSG_MAP(BPTST)
                // NOTE: the ClassWizard will add message map macros here
        //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// BPTST message handlers


// GraphProcess.cpp : Defines the class behaviors for the application.
//

#include "stdafx.h"
#include "GraphProcess.h"
#include "GraphProcessDlg.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CGraphProcessApp

BEGIN_MESSAGE_MAP(CGraphProcessApp, CWinApp)
        //{{AFX_MSG_MAP(CGraphProcessApp)
```

```
		// NOTE - the ClassWizard will add and remove mapping macros here.
		//    DO NOT EDIT what you see in these blocks of generated code!
	//}}AFX_MSG
	ON_COMMAND(ID_HELP, CWinApp::OnHelp)
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CGraphProcessApp construction

CGraphProcessApp::CGraphProcessApp()
{
	// TODO: add construction code here,
	// Place all significant initialization in InitInstance
}

/////////////////////////////////////////////////////////////////////////////
// The one and only CGraphProcessApp object

CGraphProcessApp theApp;

/////////////////////////////////////////////////////////////////////////////
// CGraphProcessApp initialization

BOOL CGraphProcessApp::InitInstance()
{
	AfxEnableControlContainer();

	// Standard initialization
	// If you are not using these features and wish to reduce the size
	//  of your final executable, you should remove from the following
	//  the specific initialization routines you do not need.

#ifdef _AFXDLL
	Enable3dControls();			// Call this when using MFC in a shared DLL
#else
	Enable3dControlsStatic();	// Call this when linking to MFC statically
#endif

	CGraphProcessDlg dlg;
```

```
        m_pMainWnd = &dlg;
        int nResponse = dlg.DoModal();
        if (nResponse == IDOK)
        {
                // TODO: Place code here to handle when the dialog is
                //  dismissed with OK
        }
        else if (nResponse == IDCANCEL)
        {
                // TODO: Place code here to handle when the dialog is
                //  dismissed with Cancel
        }

        // Since the dialog has been closed, return FALSE so that we exit the
        //  application, rather than start the application's message pump.
        return FALSE;
}


// GraphProcessDlg.cpp : implementation file
//

#include "stdafx.h"
#include "GraphProcess.h"
#include "GraphProcessDlg.h"

#include <conio.h>
#include <ctype.h>
#include <math.h>
#include <malloc.h>

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

#define    Success_Exit    3
```

```
#define    Restart          2
#define    Failure_Exit     1
#define    Continue         0

// BP 관련재처 챔짐완 렐껐
double Nerr[20000];
float CurErr;
float *WgT[10000],*OuT[10000],*ErR[10000],*DeL[10000];
float TgeT[10000][50],InpT[10000][50],PErr[10000],OutP[10000][50],*Ptrr;
int   UNO[10];
int   RStop,CurEpo,nsnew,nsold,chkno;
char  WorkName[20];
int   Prn_Err;
float *Mem1,*Mem2,*Mem3,*Mem4;
FILE  *fp,*fp1,*fp2,*fp3,*fopen(),*fp4;

/////////////////////////////////////////////////////////////////////////////
// CAboutDlg dialog used for App About

class CAboutDlg : public CDialog
{
public:
            CAboutDlg();

// Dialog Data
            //{{AFX_DATA(CAboutDlg)
            enum { IDD = IDD_ABOUTBOX };
            //}}AFX_DATA

            // ClassWizard generated virtual function overrides
            //{{AFX_VIRTUAL(CAboutDlg)
            protected:
            virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
            //}}AFX_VIRTUAL

// Implementation
protected:
            //{{AFX_MSG(CAboutDlg)
            //}}AFX_MSG
```

```
        DECLARE_MESSAGE_MAP()
};

CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
{
        //{{AFX_DATA_INIT(CAboutDlg)
        //}}AFX_DATA_INIT
}

void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
        CDialog::DoDataExchange(pDX);
        //{{AFX_DATA_MAP(CAboutDlg)
        //}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
        //{{AFX_MSG_MAP(CAboutDlg)
                // No message handlers
        //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CGraphProcessDlg dialog

CGraphProcessDlg::CGraphProcessDlg(CWnd* pParent /*=NULL*/)
        : CDialog(CGraphProcessDlg::IDD, pParent)
{
        //{{AFX_DATA_INIT(CGraphProcessDlg)
        //}}AFX_DATA_INIT
        // Note that LoadIcon does not require a subsequent DestroyIcon in Win32
        m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);

        m_pImageDev = NULL;
}

CGraphProcessDlg::~CGraphProcessDlg()
{
        if (m_pImageDev != NULL)
```

```
        {
                delete m_pImageDev;
                m_pImageDev = NULL;
        }
        delete m_pStereoLSource;
        delete m_pStereoRSource;
}

void CGraphProcessDlg::DoDataExchange(CDataExchange* pDX)
{
        CDialog::DoDataExchange(pDX);
        //{{AFX_DATA_MAP(CGraphProcessDlg)
        DDX_Control(pDX, IDC_REALCOOR_ZVALUE_EDIT, m_ctrlReCoZEdit);
        DDX_Control(pDX, IDC_REALCOOR_YVALUE_EDIT, m_ctrlReCoYEdit);
        DDX_Control(pDX, IDC_REALCOOR_XVALUE_EDIT, m_ctrlReCoXEdit);
        DDX_Control(pDX, IDC_IMAGECOOR_ZVALUE_EDIT, m_ctrlImCoZEdit);
        DDX_Control(pDX, IDC_IMAGECOOR_YVALUE_EDIT, m_ctrlImCoYEdit);
        DDX_Control(pDX, IDC_IMAGECOOR_XVALUE_EDIT, m_ctrlImCoXEdit);
        DDX_Control(pDX, IDC_THRESHOLD_BVALUE_EDIT, m_ctrlThreshBlueEdit);
        DDX_Control(pDX, IDC_THRESHOLD_GVALUE_EDIT, m_ctrlThreshGreenEdit)
        DDX_Control(pDX, IDC_THRESHOLD_RVALUE_EDIT, m_ctrlThreshRedEdit);
        DDX_Control(pDX, IDC_YVALUE_EDIT, m_ctrlYValueEdit);
        DDX_Control(pDX, IDC_XVALUE_EDIT, m_ctrlXValueEdit);
        DDX_Control(pDX, IDC_RVALUE_EDIT, m_ctrlRValueEdit);
        DDX_Control(pDX, IDC_GVALUE_EDIT, m_ctrlGValueEdit);
        DDX_Control(pDX, IDC_BVALUE_EDIT, m_ctrlBValueEdit);

        //}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CGraphProcessDlg, CDialog)
        //{{AFX_MSG_MAP(CGraphProcessDlg)
        ON_WM_SYSCOMMAND()
        ON_WM_PAINT()
        ON_WM_QUERYDRAGICON()
        ON_BN_CLICKED(IDC_OVERLAY_DEV, OnOverlayDev)
        ON_BN_CLICKED(IDC_PREVIEW_DEV, OnPreviewDev)
        ON_COMMAND(IDM_VIDEOCOMP_DLG, OnVideocompDlg)
        ON_COMMAND(IDM_VIDEOFORMAT_DLG, OnVideoformatDlg)
```

```
        ON_COMMAND(IDM_VIDEOSOURCE_DLG, OnVideosourceDlg)
        ON_COMMAND(ID_ABOUT_DLG, OnAboutDlg)
        ON_BN_CLICKED(IDC_STEREOL_BUTTON, OnStereolButton)
        ON_BN_CLICKED(IDC_STEREOR_BUTTON, OnStereorButton)
        ON_BN_CLICKED(IDC_STEREO_PROCESS_BUTTON, OnStereoProcessButton)
        ON_WM_MOUSEMOVE()
        ON_BN_CLICKED(IDC_STEREO_THRESHOLD_BUTTON, OnStereoThresholdButton)
        ON_BN_CLICKED(IDC_STEREO_BLOCKTH_BUTTON, OnStereoBlockthButton)
        ON_BN_CLICKED(IDC_CALIBRATION_LEFT_DEV, OnCalibrationLeftDev)
        ON_BN_CLICKED(IDC_CALIBRATION_RIGHT_DEV, OnCalibrationRightDev)
        ON_COMMAND(ID_BMPFILE_OPEN, OnBmpfileOpen)
        ON_COMMAND(ID_BMPFILE_SAVE, OnBmpfileSave)
        ON_BN_CLICKED(IDC_BPLN_BTN, OnBiBplnBtn)
        ON_BN_CLICKED(IDC_BPOUT_BTN, OnBiBpoutBtn)
        //}}AFX_MSG_MAP
END_MESSAGE_MAP()

int CGraphProcessDlg::DoModal()
{
        // TODO: Add your specialized code here and/or call the base class
        return CDialog::DoModal();
}

/////////////////////////////////////////////////////////////////////////////
// CGraphProcessDlg message handlers

BOOL CGraphProcessDlg::OnInitDialog()
{
        CDialog::OnInitDialog();

        // Add "About..." menu item to system menu.

        // IDM_ABOUTBOX must be in the system command range.
        ASSERT((IDM_ABOUTBOX & 0xFFF0) == IDM_ABOUTBOX);
        ASSERT(IDM_ABOUTBOX < 0xF000);

        CMenu* pSysMenu = GetSystemMenu(FALSE);
        if (pSysMenu != NULL)
        {
```

```
                CString strAboutMenu;
                strAboutMenu.LoadString(IDS_ABOUTBOX);
                if (!strAboutMenu.IsEmpty())
                {
                        pSysMenu->AppendMenu(MF_SEPARATOR);
                        pSysMenu->AppendMenu(MF_STRING,          IDM_ABOUTBOX,
strAboutMenu);
                }
        }

        // Set the icon for this dialog.  The framework does this automatically
        //  when the application's main window is not a dialog
        SetIcon(m_hIcon, TRUE);                         // Set big icon
        SetIcon(m_hIcon, FALSE);                // Set small icon

        // TODO: Add extra initialization here
        m_pCameraWnd = GetDlgItem(IDC_CAMERA_MON);
        m_pStereoLWnd = GetDlgItem(IDC_STEREOLEFT_MON);
        m_pStereoRWnd = GetDlgItem(IDC_STEREORIGHT_MON);

        m_pImageDev = new CImageDev(m_pCameraWnd);
        m_pStereoLSource = new CImageSource();
        m_pStereoRSource = new CImageSource();

        int w, h;
        w = m_pImageDev->GetSizeWidthCapWindow();
        h =     m_pImageDev->GetSizeHeightCapWindow();

        m_pImageDev->Overlay();

        return TRUE;  // return TRUE  unless you set the focus to a control
}

void CGraphProcessDlg::OnSysCommand(UINT nID, LPARAM lParam)
{
        if ((nID & 0xFFF0) == IDM_ABOUTBOX)
        {
                CAboutDlg dlgAbout;
                dlgAbout.DoModal();
```

```
        }
        else
        {
                CDialog::OnSysCommand(nID, lParam);
        }
}

// If you add a minimize button to your dialog, you will need the code below
//  to draw the icon.  For MFC applications using the document/view model,
//  this is automatically done for you by the framework.

void CGraphProcessDlg::OnPaint()
{
        if (IsIconic())
        {
                CPaintDC dc(this); // device context for painting

                SendMessage(WM_ICONERASEBKGND, (WPARAM) dc.GetSafeHdc(), 0)

                // Center icon in client rectangle
                int cxIcon = GetSystemMetrics(SM_CXICON);
                int cyIcon = GetSystemMetrics(SM_CYICON);
                CRect rect;
                GetClientRect(&rect);
                int x = (rect.Width() - cxIcon + 1) / 2;
                int y = (rect.Height() - cyIcon + 1) / 2;

                // Draw the icon
                dc.DrawIcon(x, y, m_hIcon);
        }
        else
        {
                CDialog::OnPaint();
        }
}

// The system calls this to obtain the cursor to display while the user drags
//  the minimized window.
HCURSOR CGraphProcessDlg::OnQueryDragIcon()
```

```
{
        return (HCURSOR) m_hIcon;
}

void CGraphProcessDlg::OnOverlayDev()
{
        // TODO: Add your control notification handler code here
        m_pImageDev->Overlay();
}

void CGraphProcessDlg::OnPreviewDev()
{
        // TODO: Add your control notification handler code here
        m_pImageDev->Preview();
}

void CGraphProcessDlg::OnVideocompDlg()
{
        // TODO: Add your command handler code here
        m_pImageDev->VideoCompressionDlg();
}

void CGraphProcessDlg::OnVideoformatDlg()
{
        // TODO: Add your command handler code here
        m_pImageDev->VideoFormatDlg();
}

void CGraphProcessDlg::OnVideosourceDlg()
{
        // TODO: Add your command handler code here
        m_pImageDev->VideoSourceDlg();
}

void CGraphProcessDlg::OnAboutDlg()
{
        // TODO: Add your command handler code here
        CAboutDlg dlgAbout;
        dlgAbout.DoModal();
```

```
}


void CGraphProcessDlg::OnStereolButton()
{
        // TODO: Add your control notification handler code here
        CDC * pCamDC;
        CDC * pStereoLeftDC;

        pCamDC = m_pCameraWnd->GetDC();
        pStereoLeftDC = m_pStereoLWnd->GetDC();
        int i, j;

        m_pImageDev->Preview();
        m_pImageDev->GrabFrame();

        for (j=0; j<240; j++)
        {
                for (i=0; i<320; i++)
                {
                        m_pStereoLSource->m_cOrgColor[i][j] = pCamDC->GetPixel(i, j);
                        m_pStereoLSource->m_cRstColor[i][j]
m_pStereoLSource->m_cOrgColor[i][j];
                }
        }

        m_pImageDev->Overlay();

        for(j=0; j<240; j++)
        {
                for(i=0; i<320; i++)
                {
                        pStereoLeftDC->SetPixel(i, j, m_pStereoLSource->m_cRstColor[i][j])
                }
        }

        ReleaseDC(pCamDC);
        ReleaseDC(pStereoLeftDC);
```

```
void CGraphProcessDlg::OnStereorButton()
{
	// TODO: Add your control notification handler code here
	CDC * pCamDC;
	CDC * pStereoRightDC;

	pCamDC = m_pCameraWnd->GetDC();
	pStereoRightDC = m_pStereoRWnd->GetDC();
	int i, j;

	m_pImageDev->Preview();
	m_pImageDev->GrabFrame();

	for (j=0; j<240; j++)
	{
		for (i=0; i<320; i++)
		{
			m_pStereoRSource->m_cOrgColor[i][j] = pCamDC->GetPixel(i, j);
			m_pStereoRSource->m_cRstColor[i][j]
m_pStereoRSource->m_cOrgColor[i][j];
		}
	}

	m_pImageDev->Overlay();

	for(j=0; j<240; j++)
	{
		for(i=0; i<320; i++)
		{
			pStereoRightDC->SetPixel(i, j, m_pStereoRSource->m_cRstColor[i][j])
		}
	}

	ReleaseDC(pCamDC);
	ReleaseDC(pStereoRightDC);
}
```

```
void CGraphProcessDlg::OnStereoProcessButton()
{
	// TODO: Add your control notification handler code here
	CDC * pCamDC;
	CDC * pStereoLeftDC;
	CDC * pStereoRightDC;

	pCamDC = m_pCameraWnd->GetDC();
	pStereoLeftDC = m_pStereoLWnd->GetDC();
	pStereoRightDC = m_pStereoRWnd->GetDC();

	m_pImageDev->Preview();
	m_pImageDev->GrabFrame();

	int i, j;

	for (j=0; j<240; j++)
	{
		for (i=0; i<320; i++)
		{
			m_pStereoLSource->m_cOrgColor[i][j] = pCamDC->GetPixel(i, j)
			m_pStereoLSource->m_cRstColor[i][j]
m_pStereoLSource->m_cOrgColor[i][j];
		}
	}

	m_pImageDev->Overlay();

	// Start
	m_pStereoLSource->HistogramEqColor();
	m_pStereoLSource->SubBinaryColor();
	// End

	//Start of Thining
	m_pStereoLSource->Thining();
	//End of Thining

	if (MessageBox("Right Camera ON", "Wait", MB_OK | MB_ICONQUESTION) == IDOK)
	{
```

```
                    m_pImageDev->Preview();
                    m_pImageDev->GrabFrame();

                    for (j=0; j<240; j++)
                    {
                              for (i=0; i<320; i++)
                              {
                                        m_pStereoRSource->m_cOrgColor[i][j]
pCamDC->GetPixel(i, j);
                                        m_pStereoRSource->m_cRstColor[i][j]
m_pStereoRSource->m_cOrgColor[i][j];
                              }
                    }
                    // Start
//                  m_pStereoRSource->HistogramEqColor();
                    m_pStereoRSource->SubBinaryColor();
                    // End

                    //Start of Thining
                    m_pStereoRSource->Thining();
                    //End of Thining

                    m_pImageDev->Overlay();
          }

          for(j=0; j<240; j++)
          {
                    for(i=0; i<320; i++)
                    {
                              pStereoLeftDC->SetPixel(i, j, m_pStereoLSource->m_cRstColor[i][j]);
                    }
          }

          for(j=0; j<240; j++)
          {
                    for(i=0; i<320; i++)
                    {
                              pStereoRightDC->SetPixel(i, j, m_pStereoRSource->m_cRstColor[i][j])
                    }
```

```
        }

        ReleaseDC(pCamDC);
        ReleaseDC(pStereoLeftDC);
        ReleaseDC(pStereoRightDC);
}

void CGraphProcessDlg::OnMouseMove(UINT nFlags, CPoint point)
{
        // TODO: Add your message handler code here and/or call default
        CPoint      ptStereoLeft = point;
        CPoint      ptStereoRight = point;

        MapWindowPoints(m_pStereoLWnd, &ptStereoLeft, 1);
        MapWindowPoints(m_pStereoRWnd, &ptStereoRight, 1);

        m_pStereoLWnd->GetClientRect(&m_rectStereoL);
        m_pStereoRWnd->GetClientRect(&m_rectStereoR);

        CString     szX;
        CString     szY;
        CString     szR;
        CString     szG;
        CString     szB;

        if ((m_rectStereoL.left <= ptStereoLeft.x && m_rectStereoL.right > ptStereoLeft.x) &&
                (m_rectStereoL.top <= ptStereoLeft.y && m_rectStereoL.bottom > ptStereoLeft.y))
        {
                szX.Format("%d", ptStereoLeft.x);
                szY.Format("%d", ptStereoLeft.y);
                szR.Format("%d",
GetRValue(m_pStereoLSource->m_cRstColor[ptStereoLeft.x][ptStereoLeft.y]));
                szG.Format("%d",
GetGValue(m_pStereoLSource->m_cRstColor[ptStereoLeft.x][ptStereoLeft.y]));
                szB.Format("%d",
GetBValue(m_pStereoLSource->m_cRstColor[ptStereoLeft.x][ptStereoLeft.y]));
                m_ctrlXValueEdit.SetWindowText(szX);
                m_ctrlYValueEdit.SetWindowText(szY);
```

```
                m_ctrlRValueEdit.SetWindowText(szR);
                m_ctrlGValueEdit.SetWindowText(szG);
                m_ctrlBValueEdit.SetWindowText(szB);
        }
        else if ((m_rectStereoR.left <= ptStereoRight.x && m_rectStereoR.right > ptStereoRight.x) &&
                (m_rectStereoR.top <= ptStereoRight.y && m_rectStereoR.bottom > ptStereoRight.y))
        {
                szX.Format("%d", ptStereoRight.x);
                szY.Format("%d", ptStereoRight.y);
                szR.Format("%d"
GetRValue(m_pStereoRSource->m_cRstColor[ptStereoRight.x][ptStereoRight.y]));
                szG.Format("%d"
GetGValue(m_pStereoRSource->m_cRstColor[ptStereoRight.x][ptStereoRight.y]));
                szB.Format("%d"
GetBValue(m_pStereoRSource->m_cRstColor[ptStereoRight.x][ptStereoRight.y]));
                m_ctrlXValueEdit.SetWindowText(szX);
                m_ctrlYValueEdit.SetWindowText(szY);
                m_ctrlRValueEdit.SetWindowText(szR);
                m_ctrlGValueEdit.SetWindowText(szG);
                m_ctrlBValueEdit.SetWindowText(szB);
        }
        else
        {
                m_ctrlXValueEdit.SetWindowText("0");
                m_ctrlYValueEdit.SetWindowText("0");
                m_ctrlRValueEdit.SetWindowText("0");
                m_ctrlGValueEdit.SetWindowText("0");
                m_ctrlBValueEdit.SetWindowText("0");
        }

        CDialog::OnMouseMove(nFlags, point);
}

void CGraphProcessDlg::OnStereoThresholdButton()
{
        // TODO: Add your control notification handler code here
        CDC * pStereoLeftDC;
        CDC * pStereoRightDC;
```

```
pStereoLeftDC = m_pStereoLWnd->GetDC();
pStereoRightDC = m_pStereoRWnd->GetDC();

int ThreshRed, ThreshGreen, ThreshBlue;
CString szRed = "0", szGreen = "0", szBlue = "0";
int i, j;

m_ctrlThreshRedEdit.GetWindowText(szRed);
ThreshRed = atoi((const char *)szRed);

m_ctrlThreshRedEdit.GetWindowText(szGreen);
ThreshGreen = atoi((const char *)szGreen);

m_ctrlThreshRedEdit.GetWindowText(szBlue);
ThreshBlue = atoi((const char *)szBlue);

//      m_pStereoLSource->HistogramEqColor();
        m_pStereoLSource->BinaryColor(ThreshRed, ThreshGreen, ThreshBlue);

//      m_pStereoRSource->HistogramEqColor();
        m_pStereoRSource->BinaryColor(ThreshRed, ThreshGreen, ThreshBlue);

for (j=0; j<240; j++)
{
        for (i=0; i<320; i++)
        {
                pStereoLeftDC->SetPixel(i, j, m_pStereoLSource->m_cRstColor[i][j]);
        }
}

for (j=0; j<240; j++)
{
        for (i=0; i<320; i++)
        {
                pStereoRightDC->SetPixel(i, j, m_pStereoRSource->m_cRstColor[i][j])
        }
}

ReleaseDC(pStereoLeftDC);
```

```
        ReleaseDC(pStereoRightDC);



void CGraphProcessDlg::OnStereoBlockthButton()

        // TODO: Add your control notification handler code here
        CDC * pCamDC;
        CDC * pStereoLeftDC;
        CDC * pStereoRightDC;

        pCamDC = m_pCameraWnd->GetDC();
        pStereoLeftDC = m_pStereoLWnd->GetDC();
        pStereoRightDC = m_pStereoRWnd->GetDC();

        int i, j;
        m_pImageDev->Preview();
        m_pImageDev->GrabFrame();

        for (j=0; j<240; j++)
        {
                for (i=0; i<320; i++)
                {
                        m_pStereoLSource->m_cOrgColor[i][j] = pCamDC->GetPixel(i, j)
                        m_pStereoLSource->m_cRstColor[i][j]
m_pStereoLSource->m_cOrgColor[i][j];
                }
        }

        m_pImageDev->Overlay();

        //Start Of Block Binary
        m_pStereoLSource->HistogramEqColor();
        m_pStereoLSource->BlockBinary();

        if (MessageBox("Right Camera ON", "Wait", MB_OK | MB_ICONQUESTION) == IDOK)
        {
                m_pImageDev->Preview();
                m_pImageDev->GrabFrame();
```

```
                    for (j=0; j<240; j++)
                    {
                              for (i=0; i<320; i++)
                              {
                                        m_pStereoRSource->m_cOrgColor[i][j]
pCamDC->GetPixel(i, j);
                                        m_pStereoRSource->m_cRstColor[i][j]
m_pStereoRSource->m_cOrgColor[i][j];
                              }
                    }

                    m_pStereoRSource->HistogramEqColor();
                    m_pStereoRSource->BlockBinary();
                    m_pImageDev->Overlay();
          }

          //End Of Block Binary

          for (j=0; j<240; j++)
          {
                    for (i=0; i<320; i++)
                    {
                              pStereoLeftDC->SetPixel(i, j, m_pStereoLSource->m_cRstColor[i][j]);
                    }
          }

          for (j=0; j<240; j++)
          {
                    for (i=0; i<320; i++)
                    {
                              pStereoRightDC->SetPixel(i, j, m_pStereoRSource->m_cRstColor[i][j])
                    }
          }

          ReleaseDC(pCamDC);
          ReleaseDC(pStereoLeftDC);
          ReleaseDC(pStereoRightDC);
```

```
void CGraphProcessDlg::OnCalibrationLeftDev()
{
	// TODO: Add your control notification handler code here

}

void CGraphProcessDlg::OnCalibrationRightDev()
{
	// TODO: Add your control notification handler code here

}


void CGraphProcessDlg::OnBmpfileOpen()
{
	// TODO: Add your command handler code here
	CFileDialog dlgBmpOpen(TRUE, "BMP", NULL, OFN_ALLOWMULTISELECT
OFN_FILEMUSTEXIST,
		"BMP Files(*.bmp)|*.bmp|All Files(*.*)|*.*||", NULL);
	if (dlgBmpOpen.DoModal() == IDOK)
	{
		CString strTemp;
		POSITION pos = dlgBmpOpen.GetStartPosition();

		while (pos)
		{
			strTemp += dlgBmpOpen.GetNextPathName(pos);
			strTemp += "\r\n";
		}
	}
}

void CGraphProcessDlg::OnBmpfileSave()
{
	// TODO: Add your command handler code here
	CFileDialog dlgBmpSave(FALSE, "BMP", NULL, OFN_FILEMUSTEXIST,
		"BMP Files(*.bmp)|*.bmp|All Files(*.*)|*.*||", NULL);
	if (dlgBmpSave.DoModal() == IDOK)
	{
```

```
                    CString strTemp;

                    strTemp += dlgBmpSave.GetFileName()
            }
}

////////////////////////////////////// BP
void CGraphProcessDlg::BPLearn(CString filename)
{
            long RanDom = 568731L ;
            int RStop,i,j,c,k,S1,S2;
            char  FN[20],VF[20],WF[20],EFile[20];
            int   m,n,p,Bias,index,chk_no,flag;
            float NET,Net_Out;  //float
            float PResult;

            CString szMsg;
            BPSET objDlg;
            BPPRN objDlg3;
            CClientDC objDC(this);          //DC
            int nSaveDC=objDC.SaveDC();
            fp4=fopen("epoch.log","w");
   //*****************************************
            objDlg.m_WorkName=filename;
            objDlg.m_InpU=101;
            objDlg.m_OutU=8;
            objDlg.m_SmpNo=10;
            objDlg.m_HidLay=1;
            objDlg.m_HidU=8;
            objDlg.m_Alpha=(float)0.9;
            objDlg.m_Eta=(float)0.6;
            objDlg.m_MaxPErr=(float)0.005;
            objDlg.m_MaxErr=(float)0.0005;
            objDlg.m_MaxEpo=5000;
            objDlg.m_PrnDat=0;
            objDlg.m_PrnErr=1;
            objDlg.m_ErrFile="BPERR";
            objDlg.m_OutFile="TEST";
            objDlg.m_PrnTest=1;
```

```
if(objDlg.DoModal()==IDCANCEL) return;
        m_WorkName=objDlg.m_WorkName;
        m_InpU=objDlg.m_InpU;
        m_OutU=objDlg.m_OutU;
        m_SmpNo=objDlg.m_SmpNo;
        m_HidLay=objDlg.m_HidLay;
        m_HidU=objDlg.m_HidU;
        m_Alpha=objDlg.m_Alpha;
        m_Eta=objDlg.m_Eta;
        m_MaxPErr=objDlg.m_MaxPErr;
        m_MaxErr=objDlg.m_MaxErr;
        m_MaxEpo=objDlg.m_MaxEpo;
        m_PrnDat=objDlg.m_PrnDat;
        m_PrnErr=objDlg.m_PrnErr;
        m_ErrFile=objDlg.m_ErrFile;
        m_OutFile=objDlg.m_OutFile;
        m_PrnTest=objDlg.m_PrnTest;
//      objDC.RestoreDC(int this);

        strcpy(FN,m_WorkName);
        strcat(FN,".dat");
        if((fp=fopen(FN,"r"))==NULL)
                {
            szMsg.Format(" %",FN);
                AfxMessageBox(szMsg);
            return;//exit(0);
            }
        for ( i=0; i<m_SmpNo;i++)
                {
            for (j=0; j<m_InpU;j++)
                        {
                        fscanf(fp,"%f",&InpT[i][j]);
                        if (m_PrnDat==1)
                                {
                                szMsg.Format("%f  ",InpT[i][j])
                                AfxMessageBox(szMsg);
                                }
                }
```

```
            for (j=0; j<m_OutU;j++)
                                {
                                fscanf(fp,"%f",&TgeT[i][j]);
                                if (m_PrnDat==1)
                                        {
                                        szMsg.Format("%f  ",TgeT[i][j])
                                        AfxMessageBox(szMsg);
                                        }
                        }
            }
        if((i=fclose(fp)) != 0)
                        {
            szMsg.Format(" %d",i);
                        AfxMessageBox(szMsg);
            return;//exit(0);
            }
        strcpy(EFile,m_ErrFile);
        strcat(EFile,".dat");
        if (m_PrnErr==1)
                        {
                        if ((fp3=fopen(EFile,"w")) == NULL)
                                {
                                AfxMessageBox;
                                return;//exit(0);
                                }
                        }
for (i=0; i<m_HidLay; i++)
                        {
                        UNO[i+1]=m_HidU;
            }
UNO[m_HidLay+1]=m_OutU;
UNO[0]=m_InpU;
        S1 = 0;                 //Dynamic Memory allocation
        S2 = 0;
        UNO[m_HidLay+2]=0;
        for (i=0;i<(m_HidLay+2);i++)
                        {
                        S1 +=(UNO[i]+1)*UNO[i+1];
                        S2 +=(UNO[i]+1);
```

```
        Mem1=(float *) calloc(S1+1,sizeof(float));
        Mem2=(float *) calloc(S2+1,sizeof(float));
        Mem3=(float *) calloc(S2+1,sizeof(float));
        Mem4=(float *) calloc(S1+1,sizeof(float));

        WgT[0]=Mem1;
        OuT[0]=Mem2;
        ErR[0]=Mem3;
        DeL[0]=Mem4;

        for (i=1;i<(m_HidLay+1);i++)
    {
                WgT[i]=WgT[i-1]+UNO[i]*(UNO[i-1]+1);
                DeL[i]=DeL[i-1]+UNO[i]*(UNO[i-1]+1);
                }
        for (i=1;i<(m_HidLay+2);i++)
                {
                OuT[i]=OuT[i-1]+UNO[i-1]+1;
                ErR[i]=ErR[i-1]+UNO[i-1]+1;
                }
                                                                        //Threshold
        for (i=0;i<m_HidLay+1;i++)
                {
                *(OuT[i]+UNO[i]) = 1.0 ;
                }
        do  {
                for (j=0;j<m_HidLay+1;j++)
                        {
                        for (i=0 ; i<(UNO[j]+1) * UNO[j+1] ; i++)
                                {
                                RanDom = 15625L * RanDom + 22221L ;
                                PResult=(float)pow(2.,15.);
                                *(WgT[j]+i) = (float)(((RanDom >> 16) & 0x7FFF) /PResult
0.5);

                                *(DeL[j] + i) = 0.0;
                                }
                        }
                nsold=0;
```

```
CurEpo=0;
chk_no=1;
RStop=Continue;                    //continue=0

do          {
            CurErr = 0.0;
            for(i=0; i<m_SmpNo;i++)    //upward calculation
                        {
                        for (m=0;m<m_InpU;m++) *(OuT[0]+m) = InpT[i][m];
                        for (m=1;m<m_HidLay+2;m++)
                                    {
                                    for (n=0;n<UNO[m];n++)
                                                {
                                                NET=0.0;
                                                for(p=0;p<UNO[m-1]+1;p++)
                                                            {
                                                            Bias=(UNO[m-1]+1)*n+p
                                                                        NET          +=
*(WgT[m-1]+Bias)*(*(OuT[m-1]+p));

                                                            }
                                                *(OuT[m]+n)
(float)(1/(1+exp(-NET)));//sigmoid->tanh
                                    }
                                    }
                        for     (n=0;     n<UNO[m_HidLay+1];n++)     OutP[i][n]
*(OuT[m_HidLay+1]+n);
                        for(m=0;m<UNO[m_HidLay+1];m++)
                                    {
                                    Net_Out= *(OuT[m_HidLay+1] +m);

*(ErR[m_HidLay+1]+m)=(TgeT[i][m]-Net_Out)*(1-Net_Out)*Net_Out;
                                    }
                        for (m=m_HidLay+1;m>=1;m--)
                                    {
                                    for(n=0;n<UNO[m-1]+1; n++)
                                                {
                                                *(ErR[m-1]+n)= 0.0;
                                                for (p=0; p<UNO[m]; p++)
                                                            {
```

```
                                                    Bias=(UNO[m-1]+1)*p+n;

*(DeL[m-1]+Bias)=(float)(m_Alpha*(*(ErR[m]+p))*(*(OuT[m-1]+n))+m_Eta*(*(DeL[m-1]+Bias)));
                                                                        *(ErR[m-1]+n)   +
*(ErR[m]+p)*(*(WgT[m-1]+Bias));
                                                    }

*(ErR[m-1]+n)=*(ErR[m-1]+n)*(1-*(OuT[m-1]+n))*(*(OuT[m-1]+n));
                                                    }
                                        }

//weight modification
                                        for (m=1;m<m_HidLay+2;m++)
                                                {
                                                for (n=0;n<UNO[m];n++)
                                                        {
                                                for(p=0;p<UNO[m-1]+1;p++)
                                                                {
                                                                B    i    a    s
(UNO[m-1]+1)*n+p;
                                                                *(WgT[m-1]+Bias)    +
*(DeL[m-1]+Bias);
                                                        }
                                                        }
                                        }
                                        PErr[i] = 0.0;
                                        for (m=0; m<UNO[m_HidLay+1];m++)
                                                {
                                                                PErr[i]
(float)fabs((TgeT[i][m]-*(OuT[m_HidLay+1]+m)));
                                                }
                                        CurErr +=PErr[i]*PErr[i];
                                        }

//normalized error
                                CurErr = (float)(0.5*CurErr)/(float)m_SmpNo;
                                Nerr[CurEpo]=CurErr;
                                objDC.RestoreDC(nSaveDC);

                                CurEpo++;
```

```
            if(CurEpo==chk_no*30)  chk_no++;
            if (CurEpo>=m_MaxEpo) RStop=Failure_Exit;
            nsnew=0;
            flag=1;
            for (i=0; (i<m_SmpNo)&&(flag==1);i++)
                        {
                        if (PErr[i] <= m_MaxPErr) nsnew++;
                        else flag = 0;
                        }
            if (flag == 1) RStop=Success_Exit;
            else if (CurErr<=m_MaxErr) RStop=Success_Exit;
            else RStop=Continue;
} while (RStop == Continue);

            if(m_PrnErr==1)
                        {
                        fprintf(fp4,"%d\n",CurEpo);
                        for(i=0;i<=CurEpo;i++)
                                    {
                                    fprintf(fp3,"%f\n",Nerr[i]);
                                    }
                        if ((c=fclose(fp3))!=0)
                                    {
                                    szMsg.Format(" %d", c);
                                    AfxMessageBox(szMsg);
                                    }
                        fclose(fp4);
                        }

for (i=0;i<m_SmpNo;i++)
            {                                   //upward calculation
            for (m=0;m<m_InpU;m++) *(OuT[0]+m) = InpT[i][m];
            for (m=1;m<m_HidLay+2;m++)
                        {
                        for (n=0;n<UNO[m];n++)
                                    {
                                    NET=0.0;
                                    for(p=0;p<UNO[m-1]+1;p++)
                                                {
```

```
                                                    Bias=(UNO[m-1]+1)*n+p;
                                                NET += *(WgT[m-1]+Bias)*(*(OuT[m-1]+p));
                                                    }
                                                *(OuT[m]+n)  =  (float)(1/(1+exp(-NET)))
//sigmoid function
                                        }
                                        }
                                for     (n=0;   n<UNO[m_HidLay+1];n++)      OutP[i][n]
*(OuT[m_HidLay+1]+n);
                                }
                        for (i=0;i<m_HidLay+1;i++)
                                {
                                index = 0;
                                }
                        szMsg.Format("(%d)-(%d)-(%d)%d, %f",UNO[0],UNO[1],UNO[2],CurEpo,CurErr);
                        AfxMessageBox(szMsg);
                        m_SNO=0;
                if(m_PrnTest==1)
                        {
                        for(i=0;i<=m_SmpNo;i++)
                                {
                                //if(objDlg3.DoModal()==IDCANCEL) return;
                                objDlg3.m_INU=m_OutU;
                                m_INU=objDlg3.m_INU;
                                for(j=0;j<m_OutU;j++)
                                        {
                                        if(m_OutU==1)
                                                {
                                                objDlg3.m_OutBp1=OutP[i][j];
                                                objDlg3.m_OutBp2=0;
                                                objDlg3.m_OutBp3=0;
                                                objDlg3.m_OutBp4=0;
                                                objDlg3.m_OutBp5=0;
                                                objDlg3.m_OutBp6=0;
                                                objDlg3.m_OutBp7=0;
                                                objDlg3.m_OutBp8=0;
                                                objDlg3.m_OutBp9=0;
                                                objDlg3.m_OutBp10=0;
```

```
                m_OutBp1=objDlg3.m_OutBp1;
                m_OutBp2=objDlg3.m_OutBp2;
                m_OutBp3=objDlg3.m_OutBp3;
                m_OutBp4=objDlg3.m_OutBp4;
                m_OutBp5=objDlg3.m_OutBp5;
                m_OutBp6=objDlg3.m_OutBp6;
                m_OutBp7=objDlg3.m_OutBp7;
                m_OutBp8=objDlg3.m_OutBp8;
                m_OutBp9=objDlg3.m_OutBp9;
                m_OutBp10=objDlg3.m_OutBp10;

                objDlg3.m_TgtBp1=TgcT[i][j];
                objDlg3.m_TgtBp2=0;
                objDlg3.m_TgtBp3=0;
                objDlg3.m_TgtBp4=0;
                objDlg3.m_TgtBp5=0;
                objDlg3.m_TgtBp6=0;
                objDlg3.m_TgtBp7=0;
                objDlg3.m_TgtBp8=0;
                objDlg3.m_TgtBp9=0;
                objDlg3.m_TgtBp10=0;
                m_TgtBp1=objDlg3.m_TgtBp1;
                m_TgtBp2=objDlg3.m_TgtBp2;
                m_TgtBp3=objDlg3.m_TgtBp3;
                m_TgtBp4=objDlg3.m_TgtBp4;
                m_TgtBp5=objDlg3.m_TgtBp5;
                m_TgtBp6=objDlg3.m_TgtBp6;
                m_TgtBp7=objDlg3.m_TgtBp7;
                m_TgtBp8=objDlg3.m_TgtBp8;
                m_TgtBp9=objDlg3.m_TgtBp9;
                m_TgtBp10=objDlg3.m_TgtBp10

        }
        if(m_OutU==2)
                {
                objDlg3.m_OutBp1=OutP[i][j];
                objDlg3.m_OutBp2=OutP[i][j+1];
                objDlg3.m_OutBp3=0;
                objDlg3.m_OutBp4=0;
```

```
                objDlg3.m_OutBp5=0;
                objDlg3.m_OutBp6=0;
                objDlg3.m_OutBp7=0;
                objDlg3.m_OutBp8=0;
                objDlg3.m_OutBp9=0;
                objDlg3.m_OutBp10=0;
                m_OutBp1=objDlg3.m_OutBp1;
                m_OutBp2=objDlg3.m_OutBp2;
                m_OutBp3=objDlg3.m_OutBp3;
                m_OutBp4=objDlg3.m_OutBp4;
                m_OutBp5=objDlg3.m_OutBp5;
                m_OutBp6=objDlg3.m_OutBp6;
                m_OutBp7=objDlg3.m_OutBp7;
                m_OutBp8=objDlg3.m_OutBp8;
                m_OutBp9=objDlg3.m_OutBp9;
                m_OutBp10=objDlg3.m_OutBp10;
                objDlg3.m_TgtBp1=TgeT[i][j];
                objDlg3.m_TgtBp2=TgeT[i][j+1];
                objDlg3.m_TgtBp3=0;
                objDlg3.m_TgtBp4=0;
                objDlg3.m_TgtBp5=0;
                objDlg3.m_TgtBp6=0;
                objDlg3.m_TgtBp7=0;
                objDlg3.m_TgtBp8=0;
                objDlg3.m_TgtBp9=0;
                objDlg3.m_TgtBp10=0;
                m_TgtBp1=objDlg3.m_TgtBp1;
                m_TgtBp2=objDlg3.m_TgtBp2;
                m_TgtBp3=objDlg3.m_TgtBp3;
                m_TgtBp4=objDlg3.m_TgtBp4;
                m_TgtBp5=objDlg3.m_TgtBp5;
                m_TgtBp6=objDlg3.m_TgtBp6;
                m_TgtBp7=objDlg3.m_TgtBp7;
                m_TgtBp8=objDlg3.m_TgtBp8;
                m_TgtBp9=objDlg3.m_TgtBp9;
                m_TgtBp10=objDlg3.m_TgtBp10
                }
if(m_OutU==3)
                {
```

```
objDlg3.m_OutBp1=OutP[i][j];
objDlg3.m_OutBp2=OutP[i][j+1];
objDlg3.m_OutBp3=OutP[i][j+2];
objDlg3.m_OutBp4=0;
objDlg3.m_OutBp5=0;
objDlg3.m_OutBp6=0;
objDlg3.m_OutBp7=0;
objDlg3.m_OutBp8=0;
objDlg3.m_OutBp9=0;
objDlg3.m_OutBp10=0;
m_OutBp1=objDlg3.m_OutBp1;
m_OutBp2=objDlg3.m_OutBp2;
m_OutBp3=objDlg3.m_OutBp3;
m_OutBp4=objDlg3.m_OutBp4;
m_OutBp5=objDlg3.m_OutBp5;
m_OutBp6=objDlg3.m_OutBp6;
m_OutBp7=objDlg3.m_OutBp7;
m_OutBp8=objDlg3.m_OutBp8;
m_OutBp9=objDlg3.m_OutBp9;
m_OutBp10=objDlg3.m_OutBp10

objDlg3.m_TgtBp1=TgeT[i][j];
objDlg3.m_TgtBp2=TgeT[i][j+1]
objDlg3.m_TgtBp3=TgeT[i][j+2]
objDlg3.m_TgtBp4=0;
objDlg3.m_TgtBp5=0;
objDlg3.m_TgtBp6=0;
objDlg3.m_TgtBp7=0;
objDlg3.m_TgtBp8=0;
objDlg3.m_TgtBp9=0;
objDlg3.m_TgtBp10=0;
m_TgtBp1=objDlg3.m_TgtBp1;
m_TgtBp2=objDlg3.m_TgtBp2;
m_TgtBp3=objDlg3.m_TgtBp3;
m_TgtBp4=objDlg3.m_TgtBp4;
m_TgtBp5=objDlg3.m_TgtBp5;
m_TgtBp6=objDlg3.m_TgtBp6;
m_TgtBp7=objDlg3.m_TgtBp7;
m_TgtBp8=objDlg3.m_TgtBp8;
```

```
                m_TgtBp9=objDlg3.m_TgtBp9;
                m_TgtBp10=objDlg3.m_TgtBp10
                }
if(m_OutU==4)
                {
                objDlg3.m_OutBp1=OutP[i][j];
                objDlg3.m_OutBp2=OutP[i][j+1];
                objDlg3.m_OutBp3=OutP[i][j+2];
                objDlg3.m_OutBp4=OutP[i][j+3];
                objDlg3.m_OutBp5=0;
                objDlg3.m_OutBp6=0;
                objDlg3.m_OutBp7=0;
                objDlg3.m_OutBp8=0;
                objDlg3.m_OutBp9=0;
                objDlg3.m_OutBp10=0;
                m_OutBp1=objDlg3.m_OutBp1;
                m_OutBp2=objDlg3.m_OutBp2;
                m_OutBp3=objDlg3.m_OutBp3;
                m_OutBp4=objDlg3.m_OutBp4;
                m_OutBp5=objDlg3.m_OutBp5;
                m_OutBp6=objDlg3.m_OutBp6;
                m_OutBp7=objDlg3.m_OutBp7;
                m_OutBp8=objDlg3.m_OutBp8;
                m_OutBp9=objDlg3.m_OutBp9;
                m_OutBp10=objDlg3.m_OutBp10
                objDlg3.m_TgtBp1=TgeT[i][j];
                objDlg3.m_TgtBp2=TgeT[i][j+1]
                objDlg3.m_TgtBp3=TgeT[i][j+2]
                objDlg3.m_TgtBp4=TgeT[i][j+3]
                objDlg3.m_TgtBp5=0;
                objDlg3.m_TgtBp6=0;
                objDlg3.m_TgtBp7=0;
                objDlg3.m_TgtBp8=0;
                objDlg3.m_TgtBp9=0;
                objDlg3.m_TgtBp10=0;
                m_TgtBp1=objDlg3.m_TgtBp1;
                m_TgtBp2=objDlg3.m_TgtBp2;
                m_TgtBp3=objDlg3.m_TgtBp3;
                m_TgtBp4=objDlg3.m_TgtBp4;
```

```
                m_TgtBp5=objDlg3.m_TgtBp5;
                m_TgtBp6=objDlg3.m_TgtBp6;
                m_TgtBp7=objDlg3.m_TgtBp7;
                m_TgtBp8=objDlg3.m_TgtBp8;
                m_TgtBp9=objDlg3.m_TgtBp9;
                m_TgtBp10=objDlg3.m_TgtBp10
                }
if(m_OutU==5)
                {
                objDlg3.m_OutBp1=OutP[i][j];
                objDlg3.m_OutBp2=OutP[i][j+1];
                objDlg3.m_OutBp3=OutP[i][j+2];
                objDlg3.m_OutBp4=OutP[i][j+3];
                objDlg3.m_OutBp5=OutP[i][j+4];
                objDlg3.m_OutBp6=0;
                objDlg3.m_OutBp7=0;
                objDlg3.m_OutBp8=0;
                objDlg3.m_OutBp9=0;
                objDlg3.m_OutBp10=0;
                m_OutBp1=objDlg3.m_OutBp1;
                m_OutBp2=objDlg3.m_OutBp2;
                m_OutBp3=objDlg3.m_OutBp3;
                m_OutBp4=objDlg3.m_OutBp4;
                m_OutBp5=objDlg3.m_OutBp5;
                m_OutBp6=objDlg3.m_OutBp6;
                m_OutBp7=objDlg3.m_OutBp7;
                m_OutBp8=objDlg3.m_OutBp8;
                m_OutBp9=objDlg3.m_OutBp9;
                m_OutBp10=objDlg3.m_OutBp10;

                objDlg3.m_TgtBp1=TgeT[i][j];
                objDlg3.m_TgtBp2=TgeT[i][j+1];
                objDlg3.m_TgtBp3=TgeT[i][j+2];
                objDlg3.m_TgtBp4=TgeT[i][j+3];
                objDlg3.m_TgtBp5=TgeT[i][j+4];
                objDlg3.m_TgtBp6=0;
                objDlg3.m_TgtBp7=0;
                objDlg3.m_TgtBp8=0;
                objDlg3.m_TgtBp9=0;
```

```
                objDlg3.m_TgtBp10=0;
                m_TgtBp1=objDlg3.m_TgtBp1;
                m_TgtBp2=objDlg3.m_TgtBp2;
                m_TgtBp3=objDlg3.m_TgtBp3;
                m_TgtBp4=objDlg3.m_TgtBp4;
                m_TgtBp5=objDlg3.m_TgtBp5;
                m_TgtBp6=objDlg3.m_TgtBp6;
                m_TgtBp7=objDlg3.m_TgtBp7;
                m_TgtBp8=objDlg3.m_TgtBp8;
                m_TgtBp9=objDlg3.m_TgtBp9;
                m_TgtBp10=objDlg3.m_TgtBp10
                }
if(m_OutU==6)
                {
                objDlg3.m_OutBp1=OutP[i][j];
                objDlg3.m_OutBp2=OutP[i][j+1];
                objDlg3.m_OutBp3=OutP[i][j+2];
                objDlg3.m_OutBp4=OutP[i][j+3];
                objDlg3.m_OutBp5=OutP[i][j+4];
                objDlg3.m_OutBp6=OutP[i][j+5];
                objDlg3.m_OutBp7=0;
                objDlg3.m_OutBp8=0;
                objDlg3.m_OutBp9=0;
                objDlg3.m_OutBp10=0;
                m_OutBp1=objDlg3.m_OutBp1;
                m_OutBp2=objDlg3.m_OutBp2;
                m_OutBp3=objDlg3.m_OutBp3;
                m_OutBp4=objDlg3.m_OutBp4;
                m_OutBp5=objDlg3.m_OutBp5;
                m_OutBp6=objDlg3.m_OutBp6;
                m_OutBp7=objDlg3.m_OutBp7;
                m_OutBp8=objDlg3.m_OutBp8;
                m_OutBp9=objDlg3.m_OutBp9;
                m_OutBp10=objDlg3.m_OutBp10

                objDlg3.m_TgtBp1=TgeT[i][j];
                objDlg3.m_TgtBp2=TgeT[i][j+1]
                objDlg3.m_TgtBp3=TgeT[i][j+2]
                objDlg3.m_TgtBp4=TgeT[i][j+3]
```

```
                objDlg3.m_TgtBp5=TgeT[i][j+4];
                objDlg3.m_TgtBp6=TgeT[i][j+5];
                objDlg3.m_TgtBp7=0;
                objDlg3.m_TgtBp8=0;
                objDlg3.m_TgtBp9=0;
                objDlg3.m_TgtBp10=0;
                m_TgtBp1=objDlg3.m_TgtBp1;
                m_TgtBp2=objDlg3.m_TgtBp2;
                m_TgtBp3=objDlg3.m_TgtBp3;
                m_TgtBp4=objDlg3.m_TgtBp4;
                m_TgtBp5=objDlg3.m_TgtBp5;
                m_TgtBp6=objDlg3.m_TgtBp6;
                m_TgtBp7=objDlg3.m_TgtBp7;
                m_TgtBp8=objDlg3.m_TgtBp8;
                m_TgtBp9=objDlg3.m_TgtBp9;
                m_TgtBp10=objDlg3.m_TgtBp10
                }
if(m_OutU==7)
                {
                objDlg3.m_OutBp1=OutP[i][j];
                objDlg3.m_OutBp2=OutP[i][j+1];
                objDlg3.m_OutBp3=OutP[i][j+2];
                objDlg3.m_OutBp4=OutP[i][j+3];
                objDlg3.m_OutBp5=OutP[i][j+4];
                objDlg3.m_OutBp6=OutP[i][j+5];
                objDlg3.m_OutBp7=OutP[i][j+6];
                objDlg3.m_OutBp8=0;
                objDlg3.m_OutBp9=0;
                objDlg3.m_OutBp10=0;
                m_OutBp1=objDlg3.m_OutBp1;
                m_OutBp2=objDlg3.m_OutBp2;
                m_OutBp3=objDlg3.m_OutBp3;
                m_OutBp4=objDlg3.m_OutBp4;
                m_OutBp5=objDlg3.m_OutBp5;
                m_OutBp6=objDlg3.m_OutBp6;
                m_OutBp7=objDlg3.m_OutBp7;
                m_OutBp8=objDlg3.m_OutBp8;
                m_OutBp9=objDlg3.m_OutBp9;
                m_OutBp10=objDlg3.m_OutBp10
```

```
                objDlg3.m_TgtBp1=TgeT[i][j];
                objDlg3.m_TgtBp2=TgeT[i][j+1]
                objDlg3.m_TgtBp3=TgeT[i][j+2]
                objDlg3.m_TgtBp4=TgeT[i][j+3]
                objDlg3.m_TgtBp5=TgeT[i][j+4]
                objDlg3.m_TgtBp6=TgeT[i][j+5]
                objDlg3.m_TgtBp7=TgeT[i][j+6]
                objDlg3.m_TgtBp8=0;
                objDlg3.m_TgtBp9=0;
                objDlg3.m_TgtBp10=0;
                m_TgtBp1=objDlg3.m_TgtBp1;
                m_TgtBp2=objDlg3.m_TgtBp2;
                m_TgtBp3=objDlg3.m_TgtBp3;
                m_TgtBp4=objDlg3.m_TgtBp4;
                m_TgtBp5=objDlg3.m_TgtBp5;
                m_TgtBp6=objDlg3.m_TgtBp6;
                m_TgtBp7=objDlg3.m_TgtBp7;
                m_TgtBp8=objDlg3.m_TgtBp8;
                m_TgtBp9=objDlg3.m_TgtBp9;
                m_TgtBp10=objDlg3.m_TgtBp10
                }
if(m_OutU==8)
                {
                objDlg3.m_OutBp1=OutP[i][j];
                objDlg3.m_OutBp2=OutP[i][j+1];
                objDlg3.m_OutBp3=OutP[i][j+2];
                objDlg3.m_OutBp4=OutP[i][j+3];
                objDlg3.m_OutBp5=OutP[i][j+4];
                objDlg3.m_OutBp6=OutP[i][j+5];
                objDlg3.m_OutBp7=OutP[i][j+6];
                objDlg3.m_OutBp8=OutP[i][j+7];
                objDlg3.m_OutBp9=0;
                objDlg3.m_OutBp10=0;
                m_OutBp1=objDlg3.m_OutBp1;
                m_OutBp2=objDlg3.m_OutBp2;
                m_OutBp3=objDlg3.m_OutBp3;
                m_OutBp4=objDlg3.m_OutBp4;
                m_OutBp5=objDlg3.m_OutBp5;
```

```
                m_OutBp6=objDlg3.m_OutBp6;
                m_OutBp7=objDlg3.m_OutBp7;
                m_OutBp8=objDlg3.m_OutBp8;
                m_OutBp9=objDlg3.m_OutBp9;
                m_OutBp10=objDlg3.m_OutBp10

                objDlg3.m_TgtBp1=TgeT[i][j];
                objDlg3.m_TgtBp2=TgeT[i][j+1]
                objDlg3.m_TgtBp3=TgeT[i][j+2]
                objDlg3.m_TgtBp4=TgeT[i][j+3]
                objDlg3.m_TgtBp5=TgeT[i][j+4]
                objDlg3.m_TgtBp6=TgeT[i][j+5]
                objDlg3.m_TgtBp7=TgeT[i][j+6]
                objDlg3.m_TgtBp8=TgeT[i][j+7]
                objDlg3.m_TgtBp9=0;
                objDlg3.m_TgtBp10=0;
                m_TgtBp1=objDlg3.m_TgtBp1;
                m_TgtBp2=objDlg3.m_TgtBp2;
                m_TgtBp3=objDlg3.m_TgtBp3;
                m_TgtBp4=objDlg3.m_TgtBp4;
                m_TgtBp5=objDlg3.m_TgtBp5;
                m_TgtBp6=objDlg3.m_TgtBp6;
                m_TgtBp7=objDlg3.m_TgtBp7;
                m_TgtBp8=objDlg3.m_TgtBp8;
                m_TgtBp9=objDlg3.m_TgtBp9;
                m_TgtBp10=objDlg3.m_TgtBp10
                }
if(m_OutU==9)
                {
                objDlg3.m_OutBp1=OutP[i][j];
                objDlg3.m_OutBp2=OutP[i][j+1];
                objDlg3.m_OutBp3=OutP[i][j+2];
                objDlg3.m_OutBp4=OutP[i][j+3];
                objDlg3.m_OutBp5=OutP[i][j+4];
                objDlg3.m_OutBp6=OutP[i][j+5];
                objDlg3.m_OutBp7=OutP[i][j+6];
                objDlg3.m_OutBp8=OutP[i][j+7];
                objDlg3.m_OutBp9=OutP[i][j+8];
                objDlg3.m_OutBp10=0;
```

```
                    m_OutBp1=objDlg3.m_OutBp1;
                    m_OutBp2=objDlg3.m_OutBp2;
                    m_OutBp3=objDlg3.m_OutBp3;
                    m_OutBp4=objDlg3.m_OutBp4;
                    m_OutBp5=objDlg3.m_OutBp5;
                    m_OutBp6=objDlg3.m_OutBp6;
                    m_OutBp7=objDlg3.m_OutBp7;
                    m_OutBp8=objDlg3.m_OutBp8;
                    m_OutBp9=objDlg3.m_OutBp9;
                    m_OutBp10=objDlg3.m_OutBp10

                    objDlg3.m_TgtBp1=TgeT[i][j];
                    objDlg3.m_TgtBp2=TgeT[i][j+1]
                    objDlg3.m_TgtBp3=TgeT[i][j+2]
                    objDlg3.m_TgtBp4=TgeT[i][j+3]
                    objDlg3.m_TgtBp5=TgeT[i][j+4]
                    objDlg3.m_TgtBp6=TgeT[i][j+5]
                    objDlg3.m_TgtBp7=TgeT[i][j+6]
                    objDlg3.m_TgtBp8=TgeT[i][j+7]
                    objDlg3.m_TgtBp9=TgeT[i][j+8]
                    objDlg3.m_TgtBp10=0;
                    m_TgtBp1=objDlg3.m_TgtBp1;
                    m_TgtBp2=objDlg3.m_TgtBp2;
                    m_TgtBp3=objDlg3.m_TgtBp3;
                    m_TgtBp4=objDlg3.m_TgtBp4;
                    m_TgtBp5=objDlg3.m_TgtBp5;
                    m_TgtBp6=objDlg3.m_TgtBp6;
                    m_TgtBp7=objDlg3.m_TgtBp7;
                    m_TgtBp8=objDlg3.m_TgtBp8;
                    m_TgtBp9=objDlg3.m_TgtBp9;
                    m_TgtBp10=objDlg3.m_TgtBp10
                    }
f(m_OutU==10)
                    {
                    objDlg3.m_OutBp1=OutP[i][j];
                    objDlg3.m_OutBp2=OutP[i][j+1];
                    objDlg3.m_OutBp3=OutP[i][j+2];
                    objDlg3.m_OutBp4=OutP[i][j+3];
                    objDlg3.m_OutBp5=OutP[i][j+4];
```

```
                objDlg3.m_OutBp6=OutP[i][j+5];
                objDlg3.m_OutBp7=OutP[i][j+6];
                objDlg3.m_OutBp8=OutP[i][j+7];
                objDlg3.m_OutBp9=OutP[i][j+8];
                objDlg3.m_OutBp10=OutP[i][j+9];
                m_OutBp1=objDlg3.m_OutBp1;
                m_OutBp2=objDlg3.m_OutBp2;
                m_OutBp3=objDlg3.m_OutBp3;
                m_OutBp4=objDlg3.m_OutBp4;
                m_OutBp5=objDlg3.m_OutBp5;
                m_OutBp6=objDlg3.m_OutBp6;
                m_OutBp7=objDlg3.m_OutBp7;
                m_OutBp8=objDlg3.m_OutBp8;
                m_OutBp9=objDlg3.m_OutBp9;
                m_OutBp10=objDlg3.m_OutBp10;

                objDlg3.m_TgtBp1=TgeT[i][j];
                objDlg3.m_TgtBp2=TgeT[i][j+1];
                objDlg3.m_TgtBp3=TgeT[i][j+2];
                objDlg3.m_TgtBp4=TgeT[i][j+3];
                objDlg3.m_TgtBp5=TgeT[i][j+4];
                objDlg3.m_TgtBp6=TgeT[i][j+5];
                objDlg3.m_TgtBp7=TgeT[i][j+6];
                objDlg3.m_TgtBp8=TgeT[i][j+7];
                objDlg3.m_TgtBp9=TgeT[i][j+8];
                objDlg3.m_TgtBp10=TgeT[i][j+9]
                m_TgtBp1=objDlg3.m_TgtBp1;
                m_TgtBp2=objDlg3.m_TgtBp2;
                m_TgtBp3=objDlg3.m_TgtBp3;
                m_TgtBp4=objDlg3.m_TgtBp4;
                m_TgtBp5=objDlg3.m_TgtBp5;
                m_TgtBp6=objDlg3.m_TgtBp6;
                m_TgtBp7=objDlg3.m_TgtBp7;
                m_TgtBp8=objDlg3.m_TgtBp8;
                m_TgtBp9=objDlg3.m_TgtBp9;
                m_TgtBp10=objDlg3.m_TgtBp10;
                }
objDlg3.m_SNO=i+1;
m_SNO=objDlg3.m_SNO;
```

```
                                                    }

                                        }
                                }

        } while(RStop==Restart);        /* 2 */
        ////////////////////////////
        if(RStop==Failure_Exit)  /* 1 */
                {
            AfxMessageBox("\n ");
                }
        strcpy(VF,m_OutFile);
        strcat(VF,"_v.dat");
        if ((fp1 = fopen(VF,"w+")) == NULL)
                {
                AfxMessageBox("");
                return;//exit(0);
                }
        fprintf(fp1,"%d          %d          %d          %f          %f          %d
%d\n",m_SmpNo,m_OutU,m_InpU,m_Alpha,m_Eta,m_HidLay,m_MaxEpo);
        for (i=0;i<m_HidLay+2;i++)
                {
            fprintf(fp1,"%d ",UNO[i]);
                }
        fprintf(fp1,"\n%d %f",CurEpo,CurErr);
        fprintf(fp1,"\n");
        for(i=0;i<m_SmpNo;i++)
                {
                for(j=0;j<m_OutU;j++) fprintf(fp1,"%f       ",OutP[i][j]);
                fprintf(fp1,"\n");
                }
        if ((c=fclose(fp1)) !=0)
                {
                szMsg.Format(" %d",c);
                AfxMessageBox(szMsg);
                return;//exit(0);
                }
        strcpy(WF,m_OutFile);
        strcat(WF,"_w.dat");
```

```
        if ((fp2 = fopen(WF,"w+")) == NULL)
            {
            AfxMessageBox(" ");
            return;//exit(0);
            }
        k=0;
        for (i=0;i<(m_HidLay+1);i++)
                {
                for (j=0;j< ((UNO[i]+1)*UNO[i+1]);j++)
                        {
                        if(k==8)
                                {
                        k=0;
                        fprintf(fp2,"\n");
                                }
                        fprintf(fp2,"%f ",*(WgT[i]+j))
                        k++;
                }
                }
        if ((c=fclose(fp2))!=0)
                {
                szMsg.Format("(_W.dat) %d", c);
                AfxMessageBox(szMsg);
                }
}

void CGraphProcessDlg::BPOut()
{
        int    i,j,c,m,n,p;
        char VF[20],WF[20];
        int S1,S2,Bias ;
        float *Mem1,*Mem2,*Mem3,*Mem4,NET;

        int Res[16];
        TestRes=0;

        BPTST objDlg;
        CString szMsg, tr;
```

```
fp4=fopen("NNSIM.out","w");
m_TrainedFile="TEST";
//if(objDlg.DoModal()==IDCANCEL) return;
//m_TrainedFile=objDlg.m_TrainedFile;
//m_TestedFile=objDlg.m_TestedFile;
m_TsmpNo=1;//objDlg.m_TsmpNo;

strcpy(VF,m_TrainedFile);
strcat(VF,"_v.dat");
if ((fp1=fopen(VF,"r"))==NULL)
        {
        AfxMessageBox(" *_V.");
        return;//exit(0);
        }

fscanf(fp1,"%d%d%d%f%f%d%d",&m_SmpNo,&m_OutU,&m_InpU,&m_Alpha,&m_Eta,&m_HidLay,&m_MaxEpo
);

for (i=0;i<m_HidLay+2;i++)
        {
        fscanf(fp1,"%d",&UNO[i]);
        }
if ((c=fclose(fp1)) !=0)  AfxMessageBox(" %d",c);

S1=0;
S2=0;
UNO[m_HidLay+2]=0;
szMsg.Format("HIDDEN LAYER IS %d",m_HidLay);
//AfxMessageBox(szMsg);
for(i=0;i<(m_HidLay+2); i++)
        {
        S1 += (UNO[i] +1) * UNO[i+1];
        S2 += (UNO[i] +1);
        }
Mem1=(float *) calloc(S1+1,sizeof(float));
Mem2=(float *) calloc(S2+1,sizeof(float));
Mem3=(float *) calloc(S2+1,sizeof(float));
Mem4=(float *) calloc(S1+1,sizeof(float));
WgT[0]=Mem1;
```

```
    OuT[0]=Mem2;
    ErR[0]=Mem3;
    DeL[0]=Mem4;
    for (i=1;i<(m_HidLay+1);i++)
{
            WgT[i] = WgT[i-1]+UNO[i]*(UNO[i-1]+1);
            DeL[i] = DeL[i-1] +UNO[i]*(UNO[i-1]+1);
            }
    for (i=1;i<(m_HidLay+2);i++)
            {
            OuT[i]  = OuT[i-1]  + UNO[i-1] + 1 ;
            ErR[i] = ErR[i-1] + UNO[i-1] + 1 ;
            }
    for (i=0;i<m_HidLay+1;i++)
            {
            *(OuT[i]+UNO[i]) = 1.0 ;
            }
    strcpy(WF,m_TrainedFile);
    strcat(WF,"_w.dat");
    if ((fp2 = fopen(WF,"r")) == NULL)
       {
       AfxMessageBox("*_W.dat ");
       return;//exit(0);
       }
    for (i=0;i<m_HidLay+1;i++)
            {
        for (j=0; j < (UNO[i]+1) * UNO[i+1]; j++)
                        {
                        fscanf(fp2,"%f",(WgT[i]+j));
                        szMsg.Format("%f\n",*(WgT[i]+j))
//                      AfxMessageBox(szMsg);
            }
            }
    if((c=fclose(fp2)) !=0)
            {
            szMsg.Format(" NNSIM.OUT %d",c);
            AfxMessageBox(szMsg);
            }
    if ((fp1=fopen(m_TestedFile,"r")) ==NULL)
```

```
                {
                AfxMessageBox("");
                return;//exit(0);
                }
        for(i=0;i<m_TsmpNo;i++)
                {
                for (m=0;m<m_InpU;m++)
                        {
                fscanf(fp1,"%f",&InpT[i][m]);
                        }
                }
        for(i=0;i<m_TsmpNo;i++)
                {
                for (m=0;m<m_InpU;m++) *(OuT[0]+m) = InpT[i][m];
                for (m=1;m<m_HidLay+2;m++)
                        {
                        for (n=0;n<UNO[m];n++)
                                {
                                NET=0.0;
                                for(p=0;p<UNO[m-1]+1;p++)
                                        {
                                        Bias=(UNO[m-1]+1)*n+p;
                                NET += *(WgT[m-1]+Bias)*(*(OuT[m-1]+p));
                                        }
                                *(OuT[m]+n) = (float)(1/(1+exp(-NET)));
                                }
                        }
                for (n=0; n<UNO[m_HidLay+1];n++) OutP[i][n] = *(OuT[m_HidLay+1]+n);
//      forward calculation end

                for (m=0;m<m_OutU;m++)
                        {
                                szMsg.Format("\n      sample      %d      output      %d
%f",i,m,*(OuT[m_HidLay+1]+m));
                        //AfxMessageBox(szMsg);
                        fprintf(fp4,"\n sample %d output %d = %f",i,m,*(OuT[m_HidLay+1]+m))
                        if( *(OuT[m_HidLay+1]+m) < 0.5) Res[m]=0;
                        else Res[m]=1;
                        TestRes+=(int)(Res[m]*pow(2,m));
```

```
                    }
                }
        tr.Format("\n   : %d .", TestRes);
        AfxMessageBox(tr);
        //AfxMessageBox("\nOutputs have been generated ");
        if ((i=fclose(fp1)) != 0)
                {
                szMsg.Format("\nFile cannot be closed %d",i)
                AfxMessageBox(szMsg);
                }
        fclose(fp4);


}

////////////////////////////////////////// BP
void CGraphProcessDlg::OnBiBpInBtn()
{
        CString filename="bpin2";
        BPLearn(filename);
}
////////////////////////////////////////// BP
void CGraphProcessDlg::OnBiBpoutBtn()
{
        m_TestedFile="bp2.dat";
        BPOut();
        m_BpBi=TestRes;
}


// imageDev.cpp: implementation of the CImageDev class.
//
//////////////////////////////////////////////////////////////////////

#include "stdafx.h"
#include "imageDev.h"

#ifdef _DEBUG
#undef THIS_FILE
```

```
static char THIS_FILE[]=__FILE__;
#define new DEBUG_NEW
#endif

//////////////////////////////////////////////////////////////////
// Construction/Destruction
//////////////////////////////////////////////////////////////////

CImageDev::CImageDev(CWnd * ghWndMain) {
        m_gwPalFrames = DEF_PALNUMFRAMES;
        m_gwPalColors = DEF_PALNUMCOLORS;

        m_nWidth = 320;
        m_nHeight = 240;

        TCHAR   achDeviceName[80];
        TCHAR   achDeviceVersion[100];
        TCHAR   achBuffer[100];
        WORD    wDriverCount = 0;
        WORD    wIndex;
        DWORD   dwError;

    // First create the capture window
    m_ghWndCap.m_hWnd = capCreateCaptureWindow((LPTSTR)TEXT("Capture Window"),
                                WS_CHILD | WS_VISIBLE,
                                0, 0, m_nWidth, m_nHeight,
                                (HWND)ghWndMain->m_hWnd, (int)0) ;

        // Try to connect one of the MSVIDEO drivers
        capDriverDisconnect(m_ghWndCap.m_hWnd);

        for (wIndex = 0 ; wIndex < MAXVIDDRIVERS ; wIndex++) {
                if (capGetDriverDescription(wIndex,
                        (LPTSTR)achDeviceName, sizeof(achDeviceName)/ sizeof(TCHAR),
                        (LPTSTR)achDeviceVersion, sizeof(achDeviceVersion)/sizeof(TCHAR))) {

                        // There is such a driver in the "system.ini" file.
                        // Append driver name to "Options" list in menu
                        wsprintf(achBuffer,     TEXT("&%d       %s"),       wIndex
```

```
(LPTSTR)achDeviceName)

                                        if (wDriverCount++ == 0) {
                                                // Only if no other driver is already connected
                                                dwError     =     capDriverConnect(m_ghWndCap.m_hWnd
wIndex);
                                                if (dwError) {
                                                        m_gwDeviceIndex = wIndex ;
                                                } // end if
                                        } // end if
                                } // end if
                        } // end for
                        capDriverGetCaps(m_ghWndCap.m_hWnd, &m_gCapDriverCaps, sizeof(CAPDRIVERCAPS)) ;

        // Get video format and adjust capture window
        capGetStatus(m_ghWndCap.m_hWnd, &m_gCapStatus, sizeof(CAPSTATUS)) ;

        // Start preview by default
        capPreviewRate(m_ghWndCap.m_hWnd, MS_FOR_15FPS) ;
        capPreview(m_ghWndCap.m_hWnd, FALSE);
                capOverlay(m_ghWndCap.m_hWnd, FALSE);
} // end CImageDev::CImageDev

CImageDev::~CImageDev() {
                capDriverDisconnect(m_ghWndCap.m_hWnd);
} // end CImageDev::~CImageDev

CWnd *      CImageDev::getCapWnd() {
                return &m_ghWndCap;
} // end CImageDev::getCapWnd

void CImageDev::Preview() {
                // Toggle Preview
                capGetStatus(m_ghWndCap.m_hWnd, &m_gCapStatus, sizeof(CAPSTATUS));
                capPreview(m_ghWndCap.m_hWnd, !m_gCapStatus.fLiveWindow);
} // end CImageDev::preview

void CImageDev::Overlay() {
                // Toggle Overlay
```

```
        capGetStatus(m_ghWndCap.m_hWnd, &m_gCapStatus, sizeof(CAPSTATUS))
        capOverlay(m_ghWndCap.m_hWnd, !m_gCapStatus.fOverlayWindow);

} // end CImageDev::overLay

int CImageDev::GetSizeWidthCapWindow()
{
        return m_nWidth;
} // end CImageDev::getSizeWidthCapWindow

int CImageDev::GetSizeHeightCapWindow()
{
        return m_nHeight;
} // end CImageDev::getSizeHeightCapWindow

void CImageDev::VideoFormatDlg()
{
        capDlgVideoFormat(m_ghWndCap.m_hWnd);
}

void CImageDev::VideoSourceDlg()
{
        capDlgVideoSource(m_ghWndCap.m_hWnd);
}

void CImageDev::VideoCompressionDlg()
{
        capDlgVideoCompression(m_ghWndCap.m_hWnd);
}

void CImageDev::GrabFrame()
{
        capGrabFrameNoStop(m_ghWndCap.m_hWnd);
}

void CImageDev::PalletteAuto()
{
        capPaletteAuto(m_ghWndCap.m_hWnd, 0, 256);
}
```

```
// ImageSource.cpp: implementation of the CImageSource class
//
//////////////////////////////////////////////////////////////////////

#ifdef _DEBUG
#undef THIS_FILE
static char THIS_FILE[]=__FILE__;
#define new DEBUG_NEW
#endif

//////////////////////////////////////////////////////////////////////
// Construction/Destruction
//////////////////////////////////////////////////////////////////////
#include "stdafx.h"
#include "imageSource.h"

CImageSource::CImageSource()
{

}

CImageSource::~CImageSource()
{

}

int CImageSource::GetOrgRValue(int i, int j)
{
        return GetRValue(m_cOrgColor[i][j]);
}

int CImageSource::GetOrgGValue(int i, int j)
{
        return GetGValue(m_cOrgColor[i][j]);
}

int CImageSource::GetOrgBValue(int i, int j)
```

```
{
        return GetBValue(m_cOrgColor[i][j]);
}

int CImageSource::GetOrgGrayValue(int i, int j)
{
        return (int)((GetOrgRValue(i, j) + GetOrgBValue(i, j) + GetOrgGValue(i, j))/3)
}

int CImageSource::GetRstRValue(int i, int j)
{
        return GetRValue(m_cRstColor[i][j]);
}

int CImageSource::GetRstGValue(int i, int j)
{
        return GetGValue(m_cRstColor[i][j]);
}

int CImageSource::GetRstBValue(int i, int j)
{
        return GetBValue(m_cRstColor[i][j]);
}

int CImageSource::GetRstGrayValue(int i, int j)
{
        return (int)((GetRstRValue(i, j) + GetRstBValue(i, j) + GetRstGValue(i, j))/3);
}

int CImageSource::GetTmpRValue(int i, int j)
{
        return GetRValue(m_cTmpColor[i][j]);
}

int CImageSource::GetTmpGValue(int i, int j)
{
        return GetGValue(m_cTmpColor[i][j]);
}
```

```
int CImageSource::GetTmpBValue(int i, int j)
{
        return GetBValue(m_cTmpColor[i][j]);
}

int CImageSource::GetTmpGrayValue(int i, int j)
{
        return (int)((GetTmpRValue(i, j) + GetTmpBValue(i, j) + GetTmpGValue(i, j))/3)
}

int CImageSource::GetBufRValue(int i, int j)
{
        return GetRValue(m_cBufColor[i][j]);
}

int CImageSource::GetBufGValue(int i, int j)
{
        return GetGValue(m_cBufColor[i][j]);
}

int CImageSource::GetBufBValue(int i, int j)
{
        return GetBValue(m_cBufColor[i][j]);
}

int CImageSource::GetBufGrayValue(int i, int j)
{
        return (int)((GetBufRValue(i, j) + GetBufBValue(i, j) + GetBufGValue(i, j))/3);
}

void CImageSource::InitTempColor()
{
        int i, j;

        for (j=0; j<240; j++)
        {
                for (i=0; i<320; i++)
                {
                        m_cTmpColor[i][j] = RGB(0, 0, 0);
```

```
                }
        }
}

void CImageSource::InitBuffColor()
{
        int i, j;

        for (j=0; j<240; j++)
        {
                for (i=0; i<320; i++)
                {
                        m_cBufColor[i][j] = RGB(0, 0, 0);
                }
        }
}

void CImageSource::ResultColor()
{
        int i, j;

        for (j=0; j<240; j++)
        {
                for (i=0; i<320; i++)
                {
                        m_cRstColor[i][j] = m_cTmpColor[i][j
                }
        }
}

void CImageSource::TempColor()
{
        int i, j;

        InitTempColor();

        for (j=0; j<240; j++)
        {
                for (i=0; i<320; i++)
```

```
                {
                        m_cTmpColor[i][j] = m_cRstColor[i][j];
                }
        }
}

void CImageSource::BuffColor()
{
        int i, j;

        InitBuffColor();

        for (j=0; j<240; j++)
        {
                for (i=0; i<320; i++)
                {
                        m_cBufColor[i][j] = m_cRstColor[i][j];
                }
        }
}

void CImageSource::Binary(int theshhold)
{
        int i, j;

        for (j=0; j<240; j++)
        {
                for (i=0; i<320; i++)
                {
                        if (GetRstGrayValue(i, j) > theshhold)
                        {
                                m_cRstColor[i][j] = RGB(255, 255, 255)
                        }
                        else
                        {
                                m_cRstColor[i][j] = RGB(0, 0, 0);
                        }
                }
        }
```

```
}

void CImageSource::BinaryColor(int ThreshRed, int ThreshGreen, int ThreshBlue)
{
	int i, j;
	int RedValue, GreenValue, BlueValue;

	for (j=0; j<240; j++)
	{
		for (i=0; i<320; i++)
		{
			RedValue = GetRstRValue(i, j);
			GreenValue = GetRstGValue(i, j);
			BlueValue = GetRstBValue(i, j);
			if (RedValue < ThreshRed && GreenValue < ThreshGreen &&
BlueValue < ThreshBlue)
			{
				m_cRstColor[i][j] = RGB(RedValue, GreenValue, BlueValue);
			}
			else
			{
				m_cRstColor[i][j] = RGB(255, 255, 255);
			}
		}
	}
}

void CImageSource::SubBinary()
{
	int i, j, SubPixel;

	for (j=0; j<240; j++)
	{
		for (i=0; i<320; i++)
		{
			SubPixel = abs(GetRstGrayValue(i, j) - GetRstGrayValue(i+1, j));
			if (SubPixel > 15)
			{
				m_cRstColor[i][j] = RGB(255, 255, 255);
```

```
                    }
                    else
                    {
                        m_cRstColor[i][j] = RGB(0, 0, 0);
                    }
                }
            }
}

void CImageSource::SubBinaryColor()
{
            int i, j;
            int RedValue = 0, GreenValue = 0, BlueValue = 0, GrayValue = 0;
            int RedSub, GreenSub, BlueSub, RedValueSub, GreenValueSub, BlueValueSub

            for (j=0; j<240; j++)
            {
                for (i=0; i<320; i++)
                {
                    RedValue = GetRstRValue(i, j);
                    RedValueSub = GetRstRValue(i+1, j);
                    RedSub = abs(RedValue - RedValueSub);

                    GreenValue = GetRstGValue(i, j);;
                    GreenValueSub = GetRstGValue(i+1, j);;
                    GreenSub = abs(GreenValue - GreenValueSub);

                    BlueValue = GetRstBValue(i, j);;
                    BlueValueSub = GetRstBValue(i+1, j);;
                    BlueSub = abs(BlueValue - BlueValueSub);
                    if (RedSub > 15 || GreenSub > 15 || BlueSub > 15)
                    {
                        m_cRstColor[i][j] = RGB(255, 255, 255);
                    }
                    else
                    {
                        m_cRstColor[i][j] = RGB(0, 0, 0);
                    }
                }
```

```
        }
}

void CImageSource::BlockBinary()
{
        int i, j, x, y;
        int result = 0, mask = 0;
        int sum = 0, count = 0;

        InitTempColor();

        for (j=0; j<231; j++)
        {
                for (i=0; i<311; i++)
                {
                        for (y=0; y<9; y++)
                        {
                                for(x=0; x<9; x++)
                                {
                                        mask += GetRstGrayValue(i+x, j+y)
                                }
                        }
                        result = mask / 81;
                        mask = 0;
                        if (GetRstGrayValue(i+5, j+5) > result)
                        {
                                m_cTmpColor[i+5][j+5] = RGB(255, 255, 255);
                        }
                        else
                        {
                                m_cTmpColor[i+5][j+5] = RGB(0, 0, 0);
                        }
                        result = 0;
                }
        }

        ResultColor();
}
```

```
void CImageSource::BlockBinaryColor()
{
        int i, j, x, y;
        int Rresult = 0, Rmask = 0;
        int Gresult = 0, Gmask = 0;
        int Bresult = 0, Bmask = 0;
        int Rsum = 0, Rcount = 0;
        int Gsum = 0, Gcount = 0;
        int Bsum = 0, Bcount = 0;

        InitTempColor();

        for (j=0; j<231; j++)
        {
                for (i=0; i<311; i++)
                {
                        for (y=0; y<9; y++)
                        {
                                for(x=0; x<9; x++)
                                {
                                        Rmask += GetRstRValue(i+x, j+y);
                                        Gmask += GetRstGValue(i+x, j+y);
                                        Bmask += GetRstBValue(i+x, j+y);
                                }
                        }
                        Rresult = Rmask / 81;
                        Gresult = Gmask / 81;
                        Bresult = Bmask / 81;
                        Rmask = 0;
                        Gmask = 0;
                        Bmask = 0;
                        if (GetRstRValue(i+5, j+5) > Rresult)
                        {
                                m_cTmpColor[i+5][j+5] = RGB(255, GetRstGValue(i+5, j+5)
GetRstBValue(i+5, j+5));
                        }
                        else if (GetRstRValue(i+5, j+5) <= Rresult)
                        {
                                m_cTmpColor[i+5][j+5] = RGB(0, GetRstGValue(i+5, j+5)
```

```
GetRstBValue(i+5, j+5));
                                }
                                else if (GetRstGValue(i+5, j+5) > Gresult)
                                {
                                        m_cTmpColor[i+5][j+5] = RGB(GetRstRValue(i+5, j+5), 255
GetRstBValue(i+5, j+5));
                                }
                                else if (GetRstGValue(i+5, j+5) <= Gresult)
                                {
                                        m_cTmpColor[i+5][j+5] = RGB(GetRstRValue(i+5, j+5), 0
GetRstBValue(i+5, j+5));
                                }
                                else if (GetRstBValue(i+5, j+5) > Bresult)
                                {
                                        m_cTmpColor[i+5][j+5] = RGB(GetRstRValue(i+5, j+5)
GetRstGValue(i+5, j+5), 255);
                                }
                                else if (GetRstBValue(i+5, j+5) <= Bresult)
                                {
                                        m_cTmpColor[i+5][j+5] = RGB(GetRstRValue(i+5, j+5)
GetRstGValue(i+5, j+5), 0);
                                }
                                Rresult = 0;
                                Gresult = 0;
                                Bresult = 0;
                        }
                }

                ResultColor();
}

void CImageSource::HistogramEq()
{
        int i, j, z;
        int k = 0, sum = 0, TotalPixel = 0;

        for (i=0; i<256; i++)
        {
                HistoGray[i] = 0;
```

```
                Sum_Of_HistoGray[i] = 0;
        }

        for (j=0; j<240; j++)
        {
                for (i=0; i<320; i++)
                {
                        k = GetRstGrayValue(i, j);
                        HistoGray[k] = HistoGray[k] + 1;
                }
        }

        for (i=0; i<256; i++)
        {
                sum = sum + HistoGray[i];
                Sum_Of_HistoGray[i] = sum;
        }

        TotalPixel = 320 * 240;

        for (j=0; j<240; j++)
        {
                for (i=0; i<320; i++)
                {
                        k = GetRstGrayValue(i, j);
                        z = (int)(Sum_Of_HistoGray[k] * (255.0 / TotalPixel))

                        m_cRstColor[i][j] = RGB(z, z, z);
                }
        }
}

void CImageSource::HistogramEqColor()
{
        int i, j, r, g, b;
        int rk = 0, gk = 0, bk = 0;
        int rsum = 0, gsum = 0, bsum = 0;
        int TotalPixel = 0;
```

```
for (i=0; i<256; i++)
{
        HistoRed[i] = 0;
        HistoGreen[i] = 0;
        HistoBlue[i] = 0;

        Sum_Of_HistoRed[i] = 0;
        Sum_Of_HistoGreen[i] = 0;
        Sum_Of_HistoBlue[i] = 0;
}

for (j=0; j<240; j++)
{
        for (i=0; i<320; i++)
        {
                rk = GetRstRValue(i, j);
                HistoRed[rk] = HistoRed[rk] + 1;
                gk = GetRstGValue(i, j);
                HistoGreen[gk] = HistoGreen[gk] + 1
                bk = GetRstBValue(i, j);
                HistoBlue[bk] = HistoBlue[bk] + 1;
        }
}

for (i=0; i<256; i++)
{
        rsum = rsum + HistoRed[i];
        Sum_Of_HistoRed[i] = rsum;
        gsum = gsum + HistoGreen[i];
        Sum_Of_HistoGreen[i] = gsum;
        bsum = bsum + HistoBlue[i];
        Sum_Of_HistoBlue[i] = gsum;
}

TotalPixel = 320 * 240;

for (j=0; j<240; j++)
{
        for (i=0; i<320; i++)
```

```
                {
                        rk = GetRstRValue(i, j);
                        gk = GetRstGValue(i, j);
                        bk = GetRstBValue(i, j);
                        r = (int)(Sum_Of_HistoRed[rk] * (255.0 / TotalPixel));
                        g = (int)(Sum_Of_HistoGreen[gk] * (255.0 / TotalPixel))
                        b = (int)(Sum_Of_HistoBlue[bk] * (255.0 / TotalPixel));

                        m_cRstColor[i][j] = RGB(r, g, b);
                }
        }
}

void CImageSource::HistogramSt()
{
        int i, j, z;
        int lowthresh = 0, highthresh = 255;
        int LookUpTable[256];
        float scalefactor;

        for (i=0; i<256; i++)
        {
                HistoGray[i] = 0;
        }

        for (j=0; j<240; j++)
        {
                for (i=0; i<320; i++)
                {
                        HistoGray[GetRstGrayValue(i, j)]++;
                }
        }

        for (i=0; i<256; i++)
        {
                if (HistoGray[i])
                {
                        lowthresh = i;
                        break;
```

```
            }
        }

        for (i=255; i>0; i--)
        {
            if (HistoGray[i])
            {
                highthresh = i;
                break;
            }
        }

        for (i=0; i<lowthresh; i++)
        {
            LookUpTable[i] = 0;
        }

        for (i=highthresh; i>0; i--)
        {
            LookUpTable[i] = 255;
        }

        scalefactor = (float)(255.0 / (highthresh - lowthresh));

        for (i=lowthresh; i<highthresh; i++)
        {
            LookUpTable[i] = (int)((i - lowthresh) * scalefactor)
        }

        for (j=0; j<240; j++)
        {
            for (i=0; i<320; i++)
            {
                z = LookUpTable[GetRstGrayValue(i, j)]
                m_cRstColor[i][j] = RGB(z, z, z);
            }
        }
}
```

```
void CImageSource::IIistogramStColor()
{
        int i, j, r, g, b;
        int lowthreshr = 0, highthreshr = 255;
        int lowthreshg = 0, highthreshg = 255;
        int lowthreshb = 0, highthreshb = 255;
        int LookUpTableRed[256];
        int LookUpTableGreen[256];
        int LookUpTableBlue[256];
        float rscalefactor;
        float gscalefactor;
        float bscalefactor;

        for (i=0; i<256; i++)
        {
                HistoRed[i] = 0;
                HistoGreen[i] = 0;
                HistoBlue[i] = 0;
        }

        for (j=0; j<240; j++)
        {
                for (i=0; i<320; i++)
                {
                        HistoRed[GetRstRValue(i, j)]++;
                        HistoGreen[GetRstGValue(i, j)]++
                        HistoBlue[GetRstBValue(i, j)]++;
                }
        }

        for (i=0; i<256; i++)
        {
                if (HistoRed[i])
                {
                        lowthreshr = i;
                        break;
                }
        }
```

```
for (i=0; i<256; i++)
{
        if (HistoGreen[i])
        {
                lowthreshg = i;
                break;
        }
}

for (i=0; i<256; i++)
{
        if (HistoBlue[i])
        {
                lowthreshb = i;
                break;
        }
}

for (i=255; i>0; i--)
{
        if (HistoRed[i])
        {
                highthreshr = i;
                break;
        }
}

for (i=255; i>0; i--)
{
        if (HistoGreen[i])
        {
                highthreshg = i;
                break;
        }
}

for (i=255; i>0; i--)
{
        if (HistoBlue[i])
```

```
            {
                        highthreshb = i;
                        break;
            }
}

for (i=0; i<lowthreshr; i++)
{
            LookUpTableRed[i] = 0;
}

for (i=0; i<lowthreshg; i++)
{
            LookUpTableGreen[i] = 0;
}

for (i=0; i<lowthreshb; i++)
{
            LookUpTableBlue[i] = 0;
}

for (i=highthreshr; i>0; i--)
{
            LookUpTableRed[i] = 255;
}

for (i=highthreshg; i>0; i--)
{
            LookUpTableGreen[i] = 255;
}

for (i=highthreshb; i>0; i--)
{
            LookUpTableBlue[i] = 255;
}

rscalefactor = (float)(255.0 / (highthreshr - lowthreshr));
gscalefactor = (float)(255.0 / (highthreshg - lowthreshg))
bscalefactor = (float)(255.0 / (highthreshb - lowthreshb))
```

```
        for (i=lowthreshr; i<highthreshr; i++)
        {
                LookUpTableRed[i] = (int)((i - lowthreshr) * rscalefactor);
        }

        for (i=lowthreshg; i<highthreshg; i++)
        {
                LookUpTableGreen[i] = (int)((i - lowthreshg) * gscalefactor);
        }

        for (i=lowthreshb; i<highthreshb; i++)
        {
                LookUpTableBlue[i] = (int)((i - lowthreshb) * bscalefactor);
        }

        for (j=0; j<240; j++)
        {
                for (i=0; i<320; i++)
                {
                        r = LookUpTableRed[GetRstRValue(i, j)];
                        g = LookUpTableGreen[GetRstGValue(i, j)];
                        b = LookUpTableBlue[GetRstBValue(i, j)];
                        m_cRstColor[i][j] = RGB(r, g, b);
                }
        }
}

void CImageSource::SobelMask()
{
        int i, j, x, y;
        int CenterValueX = 0, CenterValueY = 0;
        int sum = 0;
        int SobelMaskX[3][3] = {-1, 0, 1,
                                                        -2, 0, 2,
                                                        -1, 0, 1};
        int SobelMaskY[3][3] = {1, 2, 1,
                                                        0, 0, 0,
                                                        -1, -2, -1}
```

```
InitTempColor();

for (j=0; j<237; j++)
{
        for (i=0; i<317; i++)
        {
                for (y=0; y<3; y++)
                {
                        for(x=0; x<3; x++)
                        {
                                CenterValueX += GetRstGrayValue(i+x, j+y) *
SobelMaskX[x][y];

                                CenterValueY += GetRstGrayValue(i+x, j+y) *
SobelMaskY[x][y];

                        }
                }
                sum = abs(CenterValueX) + abs(CenterValueY);
                if (sum > 255)
                {
                        sum = 255;
                }

                m_cTmpColor[i+1][j+1] = RGB(sum, sum, sum);
                CenterValueX = 0;
                CenterValueY = 0;
                sum = 0;
        }
}

ResultColor();
for(j=0; j<240; j++)
{
        for(i=0; i<320; i++)
        {
                if (GetRstGrayValue(i, j) < 100)
                {
                        m_cRstColor[i][j] = RGB(0, 0, 0);
                }
```

```
                                else
                                {
                                        m_cRstColor[i][j] = RGB(255, 255, 255);
                                }
                        }
                }
}

void CImageSource::PrewittMask()
{
        int i, j, x, y;
        int CenterValueX = 0, CenterValueY = 0;
        int sum = 0;
        int PrewittMaskX[3][3] = {-1, 0, 1,
                                  -1, 0, 1,
                                  -1, 0, 1};
        int PrewittMaskY[3][3] = {1, 1, 1,
                                  0, 0, 0,
                                  -1, -1, -1};

        InitTempColor();

        for (j=0; j<237; j++)
        {
                for (i=0; i<317; i++)
                {
                        for (y=0; y<3; y++)
                        {
                                for(x=0; x<3; x++)
                                {
                                        CenterValueX += GetRstGrayValue(i+x, j+y) *
PrewittMaskX[x][y];
                                        CenterValueY += GetRstGrayValue(i+x, j+y) *
PrewittMaskY[x][y];
                                }
                        }
                        sum = abs(CenterValueX) + abs(CenterValueY);
                        if (sum > 255)
```

```
                {
                        sum = 255;
                }

                m_cTmpColor[i+1][j+1] = RGB(sum, sum, sum);
                CenterValueX = 0;
                CenterValueY = 0;
                sum = 0;
        }
    }

    ResultColor();
    for(j=0; j<240; j++)
    {
        for(i=0; i<320; i++)
        {
            if (GetRstGrayValue(i, j) < 25)
            {
                m_cRstColor[i][j] = RGB(0, 0, 0);
            }
            else
            {
                m_cRstColor[i][j] = RGB(255, 255, 255);
            }
        }
    }
}

void CImageSource::RobertMask()
{
    int i, j, x, y;
    int CenterValueX = 0, CenterValueY = 0;
    int sum = 0;
    int RobertMaskX[3][3] = {0, 0, -1,
                                                    0, 1, 0,
                                                    0, 0, 0};
    int RobertMaskY[3][3] = {-1, 0, 0,
                                                    0, 1, 0,
                                                    0, 0, 0};
```

```
InitTempColor();

for (j=0; j<237; j++)
{
        for (i=0; i<317; i++)
        {
                for (y=0; y<3; y++)
                {
                        for(x=0; x<3; x++)
                        {
                                CenterValueX += GetRstGrayValue(i+x, j+y) *
RobertMaskX[x][y];
                                CenterValueY += GetRstGrayValue(i+x, j+y) *
RobertMaskY[x][y];
                        }
                }
                sum = abs(CenterValueX) + abs(CenterValueY);
                if (sum > 255)
                {
                        sum = 255;
                }

                m_cTmpColor[i+1][j+1] = RGB(sum, sum, sum);
                CenterValueX = 0;
                CenterValueY = 0;
                sum = 0;
        }
}

ResultColor();
for(j=0; j<240; j++)
{
        for(i=0; i<320; i++)
        {
                if (GetRstGrayValue(i, j) < 25)
                {
                        m_cRstColor[i][j] = RGB(0, 0, 0);
                }
```

```
                    else
                    {
                        m_cRstColor[i][j] = RGB(255, 255, 255);
                    }
                }
            }
}

void CImageSource::LaplacianMask()
{
        int i, j, x, y;
        int CenterValue = 0;
        int sum = 0;
        int LaplacianMask[3][3] = {-1, -1, -1,
                                   -1, 8, -1,
                                   -1, -1, -1};

        InitTempColor();

        for (j=0; j<237; j++)
        {
            for (i=0; i<317; i++)
            {
                for (y=0; y<3; y++)
                {
                    for(x=0; x<3; x++)
                    {
                        CenterValue += GetRstGrayValue(i+x, j+y) *
LaplacianMask[x][y];
                    }
                }
                sum = abs(CenterValue);
                if (sum > 255)
                {
                    sum = 255;
                }

                m_cTmpColor[i+1][j+1] = RGB(sum, sum, sum);
                CenterValue = 0;
```

```
                    sum = 0;
                }
            }

            ResultColor();
            for(j=0; j<240; j++)
            {
                for(i=0; i<320; i++)
                {
                    if (GetRstGrayValue(i, j) < 40)
                    {
                        m_cRstColor[i][j] = RGB(0, 0, 0);
                    }
                    else
                    {
                        m_cRstColor[i][j] = RGB(255, 255, 255)
                    }
                }
            }
}

void CImageSource::Thining()
{
            int i, j;
            int ca = 0, cb = 0, cc = 0, cd = 0, total = 0;
            int npl = 0, spl = 0, hv = 0;
            int count = 0, check = 0, flag = 0, cz = 0;

            do
            {
                for (j=1; j<239; j++)
                {
                    for (i=1; i<319; i++)
                    {
                        m_cTmpColor[i][j] = RGB(0, 0, 0);
                    }
                }

                flag = 0;
```

```
check = count % 2;
count++;

for (j=1; j<239; j++)
{
        for (i=1; i<319; i++)
        {
                if (GetRstGrayValue(i, j) == 255)
                {
                        ca = 0;
                        cb = 0;
                        cc = 0;
                        cd = 0;
                        sp1 = 0;

                        np1 = GetRstGrayValue(i-1, j-1) +
GetRstGrayValue(i-1, j)
                                + GetRstGrayValue(i-1, j+1) +
GetRstGrayValue(i, j-1) +
                                + GetRstGrayValue(i, j+1) +
GetRstGrayValue(i+1, j-1)
                                + GetRstGrayValue(i+1, j) +
GetRstGrayValue(i+1, j+1);
                        if (np1 >= 2*255 && np1 <= 6*255)
                        {
                                ca = 0;
                        }
                        else
                        {
                                ca = 1;
                        }

                        if (GetRstGrayValue(i-1, j) == 0 &&
GetRstGrayValue(i-1, j+1) == 255)
                        {
                                sp1++;
                        }
                        if (GetRstGrayValue(i-1, j+1) == 0 &&
GetRstGrayValue(i, j+1) == 255)
```

```
{
        spl++;
}
if (GetRstGrayValue(i, j+1) == 0 &&
GetRstGrayValue(i+1, j+1) == 255)
{
        spl++;
}
if (GetRstGrayValue(i+1, j+1) == 0 &&
GetRstGrayValue(i+1, j) == 255)
{
        spl++;
}
if (GetRstGrayValue(i+1, j) == 0 &&
GetRstGrayValue(i+1, j-1) == 255)
{
        spl++;
}
if (GetRstGrayValue(i+1, j-1) == 0 &&
GetRstGrayValue(i, j-1) == 255)
{
        spl++;
}
if (GetRstGrayValue(i, j-1) == 0 &&
GetRstGrayValue(i-1, j-1) == 255)
{
        spl++;
}
if (GetRstGrayValue(i-1, j-1) == 0 &&
GetRstGrayValue(i-1, j) == 255)
{
        spl++;
}
if (spl == 1)
{
        cb = 0;
}
else
{
```

```
                    cb = 1;
                }

                if (check == 0)
                {
                    cc = GetRstGrayValue(i-1, j) *
GetRstGrayValue(i, j+1) * GetRstGrayValue(i+1, j);
                    cd = GetRstGrayValue(i, j+1) *
GetRstGrayValue(i+1, j) * GetRstGrayValue(i, j-1);
                }
                else
                {
                    cc = GetRstGrayValue(i-1, j) *
GetRstGrayValue(i, j+1) * GetRstGrayValue(i, j-1);
                    cd = GetRstGrayValue(i-1, j) *
GetRstGrayValue(i+1, j) * GetRstGrayValue(i, j-1);
                }

                total = ca || cb || cc || cd;
                if (total)
                {
                    m_cTmpColor[i][j] = RGB(255, 255,
255);
                }
                else
                {
                    flag = 1;
                }
            }
        }
        for (j=1; j<239; j++)
        {
            for (i=1; i<319; i++)
            {
                m_cRstColor[i][j] = m_cTmpColor[i][j];
            }
        }
    } while (flag);
```

```
	for (j=1; j<239; j++)
	{
		for (i=1; i<319; i++)
		{
			hv = 0;
			if (GetRstGrayValue(i, j) == 255)
			{
				if (GetRstGrayValue(i-1, j) == 255 && GetRstGrayValue(i
j+1) == 255)
				{
					hv++;
				}
				if (GetRstGrayValue(i, j+1) == 255 && GetRstGrayValue(i+1
j) == 255)
				{
					hv++;
				}
				if (GetRstGrayValue(i+1, j) == 255 && GetRstGrayValue(i
j-1) == 255)
				{
					hv++;
				}
				if (GetRstGrayValue(i, j-1) == 255 && GetRstGrayValue(i-1
j) == 255)
				{
					hv++;
				}

				if (hv == 1)
				{
					m_cRstColor[i][j] = RGB(0, 0, 0);
				}
			}
		}
	}

void CImageSource::CannyMask()
```

```
int i, j, x, y;
int c = 0, cc = 0;
int CenterValue = 0;
int CannyMask[5][5] = {2, 4, 5, 4, 2,
                       4, 9, 12, 9, 4,
                       5, 12, 15, 12, 5,
                       4, 9, 12, 9, 4,
                       2, 4, 5, 4, 2};

InitTempColor();

for (j=0; j<240; j++)
{
        for (i=0; i<320; i++)
        {
                for (y=0; y<5; y++)
                {
                        for(x=0; x<5; x++)
                        {
                                CenterValue += GetRstGrayValue(i+x, j+y) *
CannyMask[x][y];
                        }
                }
                if (CenterValue > 255)
                {
                        CenterValue = 255;
                }

                m_cTmpColor[i+2][j+2] = RGB((int)CenterValue, (int)CenterValue
(int)CenterValue);
                CenterValue = 0;
        }
}

for (j=1; j<240; j++)
{
        for (i=1; i<320; i++)
```

```
                {
                        cc = -GetRstGrayValue(i-1, j-1) - 2 * GetRstGrayValue(i-1, j) -
GetRstGrayValue(i-1, j+1);
                        cc += GetRstGrayValue(i+1, j-1) + 2 * GetRstGrayValue(i+1, j) +
GetRstGrayValue(i+1, j+1);
                        c = abs(cc);

                        cc = -GetRstGrayValue(i-1, j-1) - 2 * GetRstGrayValue(i, j-1) -
GetRstGrayValue(i+1, j-1);
                        cc += GetRstGrayValue(i-1, j+1) + 2 * GetRstGrayValue(i, j+1) +
GetRstGrayValue(i+1, j+1);
                        c += abs(cc);

                        if (c > 255)
                        {
                                c = 255;
                        }
                        else if (c < 100)
                        {
                                c = 0;
                        }
                        m_cTmpColor[i][j] = RGB((int)c, (int)c, (int)c);
                }
        }

        ResultColor();
        for(j=0; j<240; j++)
        {
                for(i=0; i<320; i++)
                {
                        if (GetRstGrayValue(i, j) < 100)
                        {
                                m_cRstColor[i][j] = RGB(0, 0, 0);
                        }
                        else
                        {
                                m_cRstColor[i][j] = RGB(255, 255, 255);
                        }
                }
```

```
        }
}

void CImageSource::Erosion()
{
        int i, j, x, y;
        int sum = 0;
        int min = 1000;

        int Temp[3][3] = {0, 0, 0, 0, 0, 0, 0, 0, 0};
        int Mask[3][3] = {255, 255, 255, 255, 255, 255, 255, 255, 255};

        InitTempColor();
        for (j=0; j<240; j++)
        {
                for (i=0; i<320; i++)
                {
                        for (y=0; y<3; y++)
                        {
                                for(x=0; x<3; x++)
                                {
                                        Temp[x][y] = GetRstGrayValue(i+x, j+y)
                                }
                        }
                        for (y=0; y<3; y++)
                        {
                                for(x=0; x<3; x++)
                                {
                                        sum = Temp[x][y] + Mask[x][y];
                                        if (min > sum)
                                        {
                                                min = sum;
                                        }
                                }
                        }
                        m_cTmpColor[i+1][j+1] = RGB(min, min, min);

                        sum = 0;
                        min = 1000;
```

```
                    }
            }

            ResultColor();
}

void CImageSource::SobelMaskColor()
{
            int i, j, x, y;
            int RCenterValueX = 0, RCenterValueY = 0;
            int GCenterValueX = 0, GCenterValueY = 0;
            int BCenterValueX = 0, BCenterValueY = 0;
            int Rsum = 0, Gsum = 0, Bsum = 0;
            int SobelMaskX[3][3] = {-1, 0, 1,
                                                                        -2, 0, 2,
                                                                        -1, 0, 1};
            int SobelMaskY[3][3] = {1, 2, 1,
                                                                        0, 0, 0,
                                                                        -1, -2, -1};

            InitTempColor();

            for (j=0; j<237; j++)
            {
                        for (i=0; i<317; i++)
                        {
                                    for (y=0; y<3; y++)
                                    {
                                                for(x=0; x<3; x++)
                                                {
                                                            RCenterValueX += GetRstRValue(i+x, j+y) *
SobelMaskX[x][y];
                                                            RCenterValueY += GetRstRValue(i+x, j+y) *
SobelMaskY[x][y];

                                                            GCenterValueX += GetRstGValue(i+x, j+y) *
SobelMaskX[x][y];
                                                            GCenterValueY += GetRstGValue(i+x, j+y) *
SobelMaskY[x][y];
```

```
                                        BCenterValueX  +=  GetRstBValue(i+x,  j+y)
SobelMaskX[x][y];
                                        BCenterValueY  +=  GetRstBValue(i+x,  j+y)
SobelMaskY[x][y];
                                }
                        }
                        Rsum = abs(RCenterValueX) + abs(RCenterValueY);
                        Gsum = abs(GCenterValueX) + abs(GCenterValueY);
                        Bsum = abs(BCenterValueX) + abs(BCenterValueY);
                        if (Rsum > 255)
                        {
                                Rsum = 255;
                        }
                        if (Gsum > 255)
                        {
                                Gsum = 255;
                        }
                        if (Bsum > 255)
                        {
                                Bsum = 255;
                        }

                        m_cTmpColor[i+1][j+1] = RGB(Rsum, Gsum, Bsum);
                        RCenterValueX = 0;
                        RCenterValueY = 0;
                        GCenterValueX = 0;
                        GCenterValueY = 0;
                        BCenterValueX = 0;
                        BCenterValueY = 0;
                        Rsum = 0;
                        Gsum = 0;
                        Bsum = 0;
                }
        }

        ResultColor();
}
```

```
void CImageSource::Dilation()
{
        int i, j, x, y;
        int sum = 0;
        int max = -1000;

        int Temp[3][3] = {0, 0, 0, 0, 0, 0, 0, 0, 0};
        int Mask[3][3] = {0, 0, 0, 0, 0, 0, 0, 0, 0};

        InitTempColor();

        for (j=0; j<240; j++)
        {
                for (i=0; i<320; i++)
                {
                        for (y=0; y<3; y++)
                        {
                                for(x=0; x<3; x++)
                                {
                                        Temp[x][y] = GetRstGrayValue(i+x, j+y)
                                }
                        }
                        for (y=0; y<3; y++)
                        {
                                for(x=0; x<3; x++)
                                {
                                        sum = Temp[x][y] + Mask[x][y];
                                        if (max < sum)
                                        {
                                                max = sum;
                                        }
                                }
                        }
                        m_cTmpColor[i+1][j+1] = RGB(max, max, max);

                        sum = 0;
                        max = -1000;
                }
        }
```

```
        ResultColor();
}

void CImageSource::LcaningRed()
{
        int i, j, x, ptX;
        int pt1, pt2;
        int HistoX1, HistoY1, HistoX2, HistoY2;
        int leaningV;

        InitTempColor();

        for (j=0; j<240; j++)
        {
                for (i=0; i<320; i++)
                {
                        pt1 = GetRstRValue(i, j);
                        pt2 = GetRstRValue(i+1, j);
                        for (x=0; x<256; x++)
                        {
                                if (pt1 == x)
                                {
                                        HistoX1 = pt1;
                                        HistoY1 = HistoRed[pt1]
                                        break;
                                }
                        }
                        for (x=0; x<256; x++)
                        {
                                if (pt2 == x)
                                {
                                        HistoX2 = pt2;
                                        HistoY2 = HistoRed[pt2]
                                        break;
                                }
                        }
                        ptX = HistoX2 - HistoX1;
```

```
                        if (ptX == 0)
                        {
                                leaningV = 0;
                                if (leaningV <= 0)
                                {
                                        m_cTmpColor[i][j] = RGB(0, 0, 0);
                                }
                                else
                                {
                                        m_cTmpColor[i][j] = RGB(255, 255, 255);
                                }
                        }
                        else
                        {
                                leaningV = (HistoY2 - HistoY1) / (HistoX2 - HistoX1)
                                if (leaningV <= 0)
                                {
                                        m_cTmpColor[i][j] = RGB(0, 0, 0);
                                }
                                else
                                {
                                        m_cTmpColor[i][j] = RGB(255, 255, 255);
                                }
                        }
                }
        }

        ResultColor();
}

void CImageSource::LeaningGreen()
{
        int i, j, x, ptX;
        int pt1, pt2;
        int HistoX1, HistoY1, HistoX2, HistoY2;
        int leaningV;

        InitTempColor();
```

```
for (j=0; j<240; j++)
{
        for (i=0; i<320; i++)
        {
                pt1 = GetRstGValue(i, j);
                pt2 = GetRstGValue(i+1, j);
                for (x=0; x<256; x++)
                {
                        if (pt1 == x)
                        {
                                HistoX1 = pt1;
                                HistoY1 = HistoGreen[pt1];
                                break;
                        }
                }
                for (x=0; x<256; x++)
                {
                        if (pt2 == x)
                        {
                                HistoX2 = pt2;
                                HistoY2 = HistoGreen[pt2];
                                break;
                        }
                }
                ptX = HistoX2 - HistoX1;

                if (ptX == 0)
                {
                        leaningV = 0;
                        if (leaningV <= 0)
                        {
                                m_cTmpColor[i][j] = RGB(0, 0, 0);
                        }
                        else
                        {
                                m_cTmpColor[i][j] = RGB(255, 255, 255)
                        }
                }
                else
```

```
                    {
                        leaningV = (HistoY2 - HistoY1) / (HistoX2 - HistoX1)
                        if (leaningV <= 0)
                        {
                            m_cTmpColor[i][j] = RGB(0, 0, 0);
                        }
                        else
                        {
                            m_cTmpColor[i][j] = RGB(255, 255, 255);
                        }
                    }
                }
            }

            ResultColor();
}

void CImageSource::LeaningBlue()
{
            int i, j, x, ptX;
            int pt1, pt2;
            int HistoX1, HistoY1, HistoX2, HistoY2;
            int leaningV;

            InitTempColor();

            for (j=0; j<240; j++)
            {
                for (i=0; i<320; i++)
                {
                    pt1 = GetRstBValue(i, j);
                    pt2 = GetRstBValue(i+1, j);
                    for (x=0; x<256; x++)
                    {
                        if (pt1 == x)
                        {
                            HistoX1 = pt1;
                            HistoY1 = HistoBlue[pt1];
                            break;
```

```
                }
            }
            for (x=0; x<256; x++)
            {
                if (pt2 == x)
                {
                    HistoX2 = pt2;
                    HistoY2 = HistoBlue[pt2];
                    break;
                }
            }
            ptX = HistoX2 - HistoX1;

            if (ptX == 0)
            {
                leaningV = 0;
                if (leaningV <= 0)
                {
                    m_cTmpColor[i][j] = RGB(0, 0, 0);
                }
                else
                {
                    m_cTmpColor[i][j] = RGB(255, 255, 255);
                }
            }
            else
            {
                leaningV = (HistoY2 - HistoY1) / (HistoX2 - HistoX1)
                if (leaningV <= 0)
                {
                    m_cTmpColor[i][j] = RGB(0, 0, 0);
                }
                else
                {
                    m_cTmpColor[i][j] = RGB(255, 255, 255);
                }
            }
        }
    }
```

```
        ResultColor();
}

void CImageSource::LeaningGray()
{
        int i, j, x, ptX;
        int pt1, pt2;
        int HistoX1, HistoY1, HistoX2, HistoY2;
        int leaningV;

        InitTempColor();

        for (j=0; j<240; j++)
        {
                for (i=0; i<320; i++)
                {
                        pt1 = GetRstGrayValue(i, j);
                        pt2 = GetRstGrayValue(i+1, j);
                        for (x=0; x<256; x++)
                        {
                                if (pt1 == x)
                                {
                                        HistoX1 = pt1;
                                        HistoY1 = HistoGray[pt1]
                                        break;
                                }
                        }
                        for (x=0; x<256; x++)
                        {
                                if (pt2 == x)
                                {
                                        HistoX2 = pt2;
                                        HistoY2 = HistoGray[pt2]
                                        break;
                                }
                        }

                        ptX = HistoX2 - HistoX1;
```

```
                        if (ptX == 0)
                        {
                                leaningV = 0;
                                if (leaningV <= 0)
                                {
                                        m_cTmpColor[i][j] = m_cRstColor[i][j];
                                }
                                else
                                {
                                        m_cTmpColor[i][j] = RGB(255, 255, 255);
                                }
                        }
                        else
                        {
                                leaningV = (HistoY2 - HistoY1) / (HistoX2 - HistoX1)
                                if (leaningV <= 0)
                                {
                                        m_cTmpColor[i][j] = m_cRstColor[i][j];
                                }
                                else
                                {
                                        m_cTmpColor[i][j] = RGB(255, 255, 255);
                                }
                        }
                }
        }

        ResultColor();
}

void CImageSource::HistogramColor()
{
        int i, j;
        int rk = 0, gk = 0, bk = 0;

        for (i=0; i<256; i++)
        {
                HistoRed[i] = 0;
```

```
                HistoGreen[i] = 0;
                HistoBlue[i] = 0;
        }

        for (j=0; j<240; j++)
        {
                for (i=0; i<320; i++)
                {
                        rk = GetRstRValue(i, j);
                        listoRed[rk] = HistoRed[rk] + 1;
                        k = GetRstGValue(i, j);
                        listoGreen[gk] = HistoGreen[gk] + 1
                        ok = GetRstBValue(i, j);
                        HistoBlue[bk] = HistoBlue[bk] + 1;
                }
        }
}

void CImageSource::HistogramGray()
{
        int i, j;
        int k = 0;

        for (i=0; i<256; i++)
        {
                HistoGray[i] = 0;
        }

        for (j=0; j<240; j++)
        {
                for (i=0; i<320; i++)
                {
                        k = GetRstGrayValue(i, j);
                        HistoGray[k] = HistoGray[k] + 1;
                }
        }
}

void CImageSource::ForwardFft()
```

```
{

}

void CImageSource::Opening()
{
        Erosion();
        Dilation();
}

void CImageSource::Closing()
{
        Dilation();
        Erosion();
}

void CImageSource::LineYScan()
{
        int i, j;
        int nCount = 0;

        for (i=0; i<320; i++)
        {
                for (j=0; j<240; j++)
                {
                        if (GetRstRValue(i, j) == 255 && GetRstGValue(i, j) == 255 &&
GetRstBValue(i, j) == 255)
                        {
                                nCount++;
                        }
                }
                if (nCount >= 30)
                {
                        for (j=0; j<240; j++)
                        {
                                m_cRstColor[i][j] = RGB(0, 0, 0);
                        }
                }
                nCount = 0;
```

```
		}
	}

	void CImageSource::LineXScan()
	{
		int i, j;
		int nCount = 0;

		for (j=0; j<240; j++)
		{
			for (i=0; i<320; i++)
			{
				if (GetRstRValue(i, j) == 255 && GetRstGValue(i, j) == 255 &&
	GetRstBValue(i, j) == 255)
				{
					nCount++;
				}
			}
			if (nCount >= 30)
			{
				for (j=0; j<240; j++)
				{
					m_cRstColor[i][j] = RGB(0, 0, 0);
				}
			}
			nCount = 0;
		}
	}

	void CImageSource::DilationColor()
	{
		int i, j, x, y;
		int sum = 0;
		int max = -1000;

		int Temp[3][3] = {0, 0, 0, 0, 0, 0, 0, 0, 0};
		int Mask[3][3] = {0, 0, 0, 0, 0, 0, 0, 0, 0};

		InitTempColor();
```

```
for (j=0; j<240; j++)
{
        for (i=0; i<320; i++)
        {
                for (y=0; y<3; y++)
                {
                        for(x=0; x<3; x++)
                        {
                                Temp[x][y] = GetRstRValue(i+x, j+y)
                        }
                }
                for (y=0; y<3; y++)
                {
                        for(x=0; x<3; x++)
                        {
                                sum = Temp[x][y] + Mask[x][y];
                                if (max < sum)
                                {
                                        max = sum;
                                }
                        }
                }
                m_cTmpColor[i+1][j+1] = RGB(max, 0, 0);

                sum = 0;
                max = -1000;
        }
}

sum = 0;
max = -1000;
for (j=0; j<240; j++)
{
        for (i=0; i<320; i++)
        {
                for (y=0; y<3; y++)
                {
                        for(x=0; x<3; x++)
```

```
                    {
                        Temp[x][y] = GetRstGValue(i+x, j+y);
                    }
                }
                for (y=0; y<3; y++)
                {
                    for(x=0; x<3; x++)
                    {
                        sum = Temp[x][y] + Mask[x][y];
                        if (max < sum)
                        {
                            max = sum;
                        }
                    }
                }
                m_cTmpColor[i+1][j+1] = RGB(GetTmpRValue(i+1, j+1), max, 0)

                sum = 0;
                max = -1000;
            }
        }

        sum = 0;
        max = -1000;
        for (j=0; j<240; j++)
        {
            for (i=0; i<320; i++)
            {
                for (y=0; y<3; y++)
                {
                    for(x=0; x<3; x++)
                    {
                        Temp[x][y] = GetRstBValue(i+x, j+y);
                    }
                }
                for (y=0; y<3; y++)
                {
                    for(x=0; x<3; x++)
                    {
```

```
                                        sum = Temp[x][y] + Mask[x][y];
                                        if (max < sum)
                                        {
                                                max = sum;
                                        }
                                }
                        }
                        m_cTmpColor[i+1][j+1]   =   RGB(GetTmpRValue(i+1,   j+1)
GetTmpGValue(i+1, j+1), max);

                        sum = 0;
                        max = -1000;
                }
        }

        ResultColor();
}

void CImageSource::ChainCode()
{
        //chain code vector
        // ------------------------------------------------------------
        // |1(i-1, j-1) | 2(i, j-1) | 3(i+1, j-1) |
        // ------------------------------------------------------------
        // |8(i-1, j)   | (i, j)    | 4(i+1, j)   |
        // ------------------------------------------------------------
        // |7(i-1, j+1) | 6(i, j+1) | 5(i+1, j+1) |
        // ------------------------------------------------------------
        int i, j;
        int c1;
        int vector = 1;
        int GrayValue = 0;

        for (j=0; j < 240; j++)
        {
                for (i=0; i < 320; i++)
                {
                        GrayValue = GetRstGrayValue(i, j);
                        if (GrayValue == 255)
```

```
                    {
                        m_cRstColor[i][j] = RGB(255,0,0);
                        break;
                    }
                }
                if (GrayValue == 255)
                {
                    break;
                }
            }

            for (c1 = 0; c1 < 76800; c1++)
            {
                if (i > 319 || j > 239)
                {
                    break;
                }
                switch (vector)
                {
                case 1:
                    {
                        GrayValue = GetRstGrayValue(i-1, j);
                        if (GrayValue == 255)
                        {
                            m_cRstColor[i-1][j] = RGB(255,0,0);
                            i = i - 1;
                            vector = 8;

                        }
                        else if (GetRstRValue(i-1, j) == 255 && GetRstGValue(i-1
j) == 0 && GetRstBValue(i-1, j) == 0)
                        {
                            break;
                        }
                        else
                        {
                            GrayValue = GetRstGrayValue(i-1, j-1);
                            if (GrayValue == 255)
                            {
```

```
                                        m_cRstColor[i-1][j-1]
RGB(255,0,0);

                                        i = i - 1;
                                        j = j - 1;
                                        vector = 1;

                                }
                                else if (GetRstRValue(i-1, j-1) == 255 &&
GetRstGValue(i-1, j-1) == 0 && GetRstBValue(i-1, j-1) == 0)
                                {
                                        break;
                                }
                                else
                                {
                                        GrayValue = GetRstGrayValue(i,
j-1);
                                        if (GrayValue == 255)
                                        {
                                                m_cRstColor[i][j-1] =
RGB(255,0,0);
                                                j = j - 1;
                                                vector = 2;

                                        }
                                        else if (GetRstRValue(i, j-1) == 255
&& GetRstGValue(i, j-1) == 0 && GetRstBValue(i, j-1) == 0)
                                        {
                                                break;
                                        }
                                        else
                                        {
                                                GrayValue =
GetRstGrayValue(i+1, j-1);
                                                if (GrayValue == 255)
                                                {
m_cRstColor[i+1][j-1] = RGB(255,0,0);
                                                        i = i + 1;
                                                        j = j - 1;
```

```
                                                    vector = 3;
                                                }
                                                else if
(GetRstRValue(i+1, j-1) == 255 && GetRstGValue(i+1, j-1) == 0 && GetRstBValue(i+1, j-1) == 0)
                                                {
                                                    break;
                                                }
                                                else
                                                {
                                                    GrayValue =
GetRstGrayValue(i+1, j);
                                                    if
(GrayValue == 255)
                                                    {

m_cRstColor[i+1][j] = RGB(255,0,0);
                                                    i
= i + 1;
vector = 4;

                                                    }
                                                    else if
(GetRstRValue(i+1, j) == 255 && GetRstGValue(i+1, j) == 0 && GetRstBValue(i+1, j) == 0)
                                                    {

break;
                                                    }
                                                    else
                                                    {

GrayValue = GetRstGrayValue(i+1, j+1);
if (GrayValue == 255)
{
        m_cRstColor[i+1][j+1] = RGB(255,0,0);
```

```
            i = i + 1;

            j = j + 1;

            vector = 5;


}

else if (GetRstRValue(i+1, j+1) == 255 && GetRstGValue(i+1, j+1) == 0 && GetRstBValue(i+1, j+1) == 0)

{

            break;

}

else

{

            GrayValue = GetRstGrayValue(i, j+1);

            if (GrayValue == 255)

            {

                        m_cRstColor[i][j+1] = RGB(255,0,0);

                        j = j + 1;

                        vector = 6;
```

```
        else if (GetRstRValue(i, j+1) == 255 && GetRstGValue(i, j+1) == 0 && GetRstBValue(i, j+1) ==
0)

        {

                break;

        }

        else

        {

                GrayValue = GetRstGrayValue(i-1, j+1);

                if (GrayValue == 255)

                {

                        m_cRstColor[i-1][j+1] = RGB(255,0,0);

                        i = i - 1;

                        j = j + 1;

                        vector = 7;



                }

                else if (GetRstRValue(i-1, j+1) == 255 && GetRstGValue(i-1, j+1) == 0 &&
GetRstBValue(i-1, j+1) == 0)

                {

                        break;

                }
```

```
                }

}
                                                                    }
                                                                }
                                                            }
                                                        }
                                                    }
                                                    break;
                                                }
                                    case 2:
                                                {
                                                    GrayValue = GetRstGrayValue(i-1, j-1);
                                                    if (GrayValue == 255)
                                                    {
                                                        m_cRstColor[i-1][j-1] = RGB(255,0,0);
                                                        i = i - 1;
                                                        j = j - 1;
                                                        vector = 1;

                                                    }
                                                    else if (GetRstRValue(i-1, j-1) == 255 &&
GetRstGValue(i-1, j-1) == 0 && GetRstBValue(i-1, j-1) == 0)
                                                    {
                                                        break;
                                                    }
                                                    else
                                                    {
                                                        GrayValue = GetRstGrayValue(i, j-1);
                                                        if (GrayValue == 255)
                                                        {
                                                            m_cRstColor[i][j-1] = RGB(255,0,0);
                                                            j = j - 1;
                                                            vector = 2;

                                                        }
                                                        else if (GetRstRValue(i, j-1) == 255 &&
GetRstGValue(i, j-1) == 0 && GetRstBValue(i, j-1) == 0)
```

```
{
        break;
}
else
{
        GrayValue = GetRstGrayValue(i+1,
j-1);
        if (GrayValue == 255)
        {
                m_cRstColor[i+1][j-1] =
RGB(255,0,0);
                i = i + 1;
                j = j - 1;
                vector = 3;

        }
        else if (GetRstRValue(i+1, j-1) ==
255 && GetRstGValue(i+1, j-1) == 0 && GetRstBValue(i+1, j-1) == 0)
        {
                break;
        }
        else
        {
                GrayValue
GetRstGrayValue(i+1, j);
                if (GrayValue == 255)
                {

m_cRstColor[i+1][j] = RGB(255,0,0);
                        i = i + 1;
                        vector = 4;

                }
                else
(GetRstRValue(i+1, j) == 255 && GetRstGValue(i+1, j) == 0 && GetRstBValue(i+1, j) == 0)
                {
                        break;
                }
                else
```

```
                                                                {
                                                                    GrayValue =
GetRstGrayValue(i+1, j+1);
                                                                    i f
(GrayValue == 255)
                                                                    {

m_cRstColor[i+1][j+1] = RGB(255,0,0);
                                                                        i
= i + 1;
                                                                        j
= j + 1;

vector = 5;

                                                                    }
                                                                    else if
(GetRstRValue(i+1, j+1) == 255 && GetRstGValue(i+1, j+1) == 0 && GetRstBValue(i+1, j+1) == 0)
                                                                    {

break;
                                                                    }
                                                                    else
                                                                    {

GrayValue = GetRstGrayValue(i, j+1);

if (GrayValue == 255)

{

        m_cRstColor[i][j+1] = RGB(255,0,0);

        j = j + 1;

        vector = 6;
```

```
}
else if (GetRstRValue(i, j+1) == 255 && GetRstGValue(i, j+1) == 0 && GetRstBValue(i, j+1) == 0)
{
        break;
}
else
{
        GrayValue = GetRstGrayValue(i-1, j+1);
        if (GrayValue == 255)
        {
                m_cRstColor[i-1][j+1] = RGB(255,0,0);
                i = i - 1;
                j = j + 1;
                vector = 7;
        }
        else if (GetRstRValue(i-1, j+1) == 255 && GetRstGValue(i-1, j+1) == 0 && GetRstBValue(i-1
j+1) == 0)
        {
                break;
```

```
        }
        else
        {
                GrayValue = GetRstGrayValue(i-1, j);
                if (GrayValue == 255)
                {
                        m_cRstColor[i-1][j] = RGB(255,0,0);
                        i = i - 1;
                        vector = 8;

                }
                else if (GetRstRValue(i-1, j) == 255 && GetRstGValue(i-1, j) == 0 &&
GetRstBValue(i-1, j) == 0)
                {
                        break;
                }
        }
}
                                                                        }
                                                                }
                                                        }
                                                }
                                        }
```

```
                    break;
                }
            case 3:
                {
                    GrayValue = GetRstGrayValue(i, j-1);
                    if (GrayValue == 255)
                    {
                        m_cRstColor[i][j-1] = RGB(255,0,0);
                        j = j - 1;
                        vector = 2;

                    }
                    else if (GetRstRValue(i, j-1) == 255 && GetRstGValue(i,
j-1) == 0 && GetRstBValue(i, j-1) == 0)
                    {
                        break;
                    }
                    else
                    {
                        GrayValue = GetRstGrayValue(i+1, j-1);
                        if (GrayValue == 255)
                        {
                            m_cRstColor[i+1][j-1]
RGB(255,0,0);

                            i = i + 1;
                            j = j - 1;
                            vector = 3;

                        }
                        else if (GetRstRValue(i+1, j-1) == 255 &&
GetRstGValue(i+1, j-1) == 0 && GetRstBValue(i+1, j-1) == 0)
                        {
                            break;
                        }
                        else
                        {
                            GrayValue = GetRstGrayValue(i+1,

                            if (GrayValue == 255)
```

```
                                                    {
                                                            m_cRstColor[i+1][j]     =
RGB(255,0,0);
                                                            i = i + 1;
                                                            vector = 4;

                                                    }
                                                    else if (GetRstRValue(i+1, j) == 255
&& GetRstCValue(i+1, j) == 0 && GetRstBValue(i+1, j) == 0)
                                                    {
                                                            break;
                                                    }
                                                    else
                                                    {
                                                            GrayValue               =
GetRstGrayValue(i+1, j+1);
                                                            if (GrayValue == 255)
                                                            {

m_cRstColor[i+1][j+1] = RGB(255,0,0);
                                                                    i = i + 1;
                                                                    j = j + 1;
                                                                    vector = 5;

                                                            }
                                                            else                    if
(GetRstRValue(i+1, j+1) == 255 && GetRstGValue(i+1, j+1) == 0 && GetRstBValue(i+1, j+1) == 0)
                                                            {
                                                                    break;
                                                            }
                                                            else
                                                            {
                                                                    GrayValue =
GetRstGrayValue(i, j+1);
                                                                    i               f
(GrayValue == 255)
                                                                    {

m_cRstColor[i][j+1] = RGB(255,0,0);
```

```
                                                                    }
= j + 1;

vector = 6;

                                                                    }
                                                                    else        if
(GetRstRValue(i, j+1) == 255 && GetRstGValue(i, j+1) == 0 && GetRstBValue(i, j+1) == 0)
                                                                    {

break;
                                                                    }
                                                                    else
                                                                    {

GrayValue = GetRstGrayValue(i-1, j+1);

if (GrayValue == 255)

{

        m_cRstColor[i-1][j+1] = RGB(255,0,0);

        i = i - 1;

        j = j + 1;

        vector = 7;


}

else if (GetRstRValue(i-1, j+1) == 255 && GetRstGValue(i-1, j+1) == 0 && GetRstBValue(i-1, j+1) == 0)

{

        break;
```

```
}
else
{
        GrayValue = GetRstGrayValue(i-1, j);
        if (GrayValue == 255)
        {
                m_cRstColor[i-1][j] = RGB(255,0,0);
                i = i - 1;
                vector = 8;
        }
        else if (GetRstRValue(i-1, j) == 255 && GetRstGValue(i-1, j) == 0 && GetRstBValue(i-1, j)
0)
        {
                break;
        }
        else
        {
                GrayValue = GetRstGrayValue(i-1, j-1);
                if (GrayValue == 255)
```

```
                {
                        m_cRstColor[i-1][j-1] = RGB(255,0,0);
                        i = i - 1;
                        j = j - 1;
                        vector = 1;

                }
                else if (GetRstRValue(i-1, j-1) == 255 && GetRstGValue(i-1, j-1) == 0 &&
GetRstBValue(i-1, j-1) == 0)
                {
                        break;
                }
        }
}
                                                                        }
                                                                }
                                                        }
                                                }
                                        }
                                        break;
                                }
                        case 4:
                                {
                                        GrayValue = GetRstGrayValue(i+1, j-1);
                                        if (GrayValue == 255)
                                        {
                                                m_cRstColor[i+1][j-1] = RGB(255,0,0);
```

```
                    i = i + 1;
                    j = j - 1;
                    vector = 3;

                }
                else if (GetRstRValue(i+1, j-1) == 255 &&
GetRstGValue(i+1, j-1) == 0 && GetRstBValue(i+1, j-1) == 0)
                {
                    break;
                }
                else
                {
                    GrayValue = GetRstGrayValue(i+1, j);
                    if (GrayValue == 255)
                    {
                        m_cRstColor[i+1][j] = RGB(255,0,0);
                        i = i + 1;
                        vector = 4;

                    }
                    else if (GetRstRValue(i+1, j) == 255 &&
GetRstGValue(i+1, j) == 0 && GetRstBValue(i+1, j) == 0)
                    {
                        break;
                    }
                    else
                    {
                        GrayValue = GetRstGrayValue(i+1,
j+1);
                        if (GrayValue == 255)
                        {
                            m_cRstColor[i+1][j+1] =
RGB(255,0,0);
                            i = i + 1;
                            j = j + 1;
                            vector = 5;

                        }
                        else if (GetRstRValue(i+1, j+1) ==
```

```
255 && GetRstGValue(i+1, j+1) == 0 && GetRstBValue(i+1, j+1) == 0)
                                                                {
                                                                        break;
                                                                }
                                                                else
                                                                {
                                                                        GrayValue
GetRstGrayValue(i, j+1);
                                                                        if (GrayValue == 255)
                                                                        {
m_cRstColor[i][j+1] = RGB(255,0,0);
                                                                                j = j + 1;
                                                                                vector = 6;
                                                                        }
                                                                        else if (GetRstRValue(i,
j+1) == 255 && GetRstGValue(i, j+1) == 0 && GetRstBValue(i, j+1) == 0)
                                                                        {
                                                                                break;
                                                                        }
                                                                        else
                                                                        {
                                                                                GrayValue =
GetRstGrayValue(i-1, j+1);
                                                                                i f
(GrayValue == 255)
                                                                                {
m_cRstColor[i-1][j+1] = RGB(255,0,0);
                                                                                        i
= i - 1;
                                                                                        j
= j + 1;
vector = 7;
                                                                                }
                                                                                else if
```

```
(GetRstRValue(i-1, j+1) == 255 && GetRstGValue(i-1, j+1) == 0 && GetRstBValue(i-1, j+1) == 0)
                                                                                    {
break;
                                                                                    }
                                                                                    else
                                                                                    {
GrayValue = GetRstGrayValue(i-1, j);
if (GrayValue == 255)
{
        m_cRstColor[i-1][j] = RGB(255,0,0);
        i = i - 1;
        vector = 8;

}
else if (GetRstRValue(i-1, j) == 255 && GetRstGValue(i-1, j) == 0 && GetRstBValue(i-1, j) == 0)
{
        break;
}
else
{
        GrayValue = GetRstGrayValue(i-1, j-1);
        if (GrayValue == 255)
```

```
{
        m_cRstColor[i-1][j-1] = RGB(255,0,0);
        i = i - 1;
        j = j - 1;
        vector = 1;

}
else if (GetRstRValue(i-1, j-1) == 255 && GetRstGValue(i-1, j-1) == 0 && GetRstBValue(i-1
j-1) == 0)
{
        break;
}
else
{
        GrayValue = GetRstGrayValue(i, j-1);
        if (GrayValue == 255)
        {
                m_cRstColor[i][j-1] = RGB(255,0,0);
                j = j - 1;
                vector = 2;
```

```
                }

                else if (GetRstRValue(i, j-1) == 255 && GetRstGValue(i, j-1) == 0 &&
GetRstBValue(i, j-1) == 0)

                {

                        break;

                }

        }

}
                                                                        }
                                                                }
                                                        }
                                                }
                                        }
                                        break;
                        }
                        case 5:
                                {
                                        GrayValue = GetRstGrayValue(i+1, j);
                                        if (GrayValue == 255)
                                        {
                                                m_cRstColor[i+1][j] = RGB(255,0,0);
                                                i = i + 1;
                                                vector = 4;

                                        }
                                        else if (GetRstRValue(i+1, j) == 255 && GetRstGValue(i+1
j) == 0 && GetRstBValue(i+1, j) == 0)
                                        {
                                                break;
                                        }
```

```
                    else
                    {
                        GrayValue = GetRstGrayValue(i+1, j+1);
                        if (GrayValue == 255)
                        {
                            m_cRstColor[i+1][j+1]
RGB(255,0,0);
                            i = i + 1;
                            j = j + 1;
                            vector = 5;

                        }
                        else if (GetRstRValue(i+1, j+1) == 255 &&
GetRstGValue(i+1, j+1) == 0 && GetRstBValue(i+1, j+1) == 0)
                        {
                            break;
                        }
                        else
                        {
                            GrayValue = GetRstGrayValue(i,
j+1);
                            if (GrayValue == 255)
                            {
                                m_cRstColor[i][j+1] =
RGB(255,0,0);
                                j = j + 1;
                                vector = 6;

                            }
                            else if (GetRstRValue(i, j+1) == 255
&& GetRstGValue(i, j+1) == 0 && GetRstBValue(i, j+1) == 0)
                            {
                                break;
                            }
                            else
                            {
                                GrayValue =
GetRstGrayValue(i-1, j+1);
                                if (GrayValue == 255)
```

```
                                                    {
m_cRstColor[i-1][j+1] = RGB(255,0,0);
                                                        i = i - 1;
                                                        j = j + 1;
                                                        vector = 7;
                                                    }
                                                    else if
(GetRstRValue(i-1, j+1) == 255 && GetRstGValue(i-1, j+1) == 0 && GetRstBValue(i-1, j+1) == 0)
                                                    {
                                                        break;
                                                    }
                                                    else
                                                    {
                                                        GrayValue =
GetRstGrayValue(i-1, j);
                                                        if
(GrayValue == 255)
                                                        {
m_cRstColor[i-1][j] = RGB(255,0,0);
                                                            i
= i - 1;
vector = 8;
                                                        }
                                                        else if
(GetRstRValue(i-1, j) == 255 && GetRstGValue(i-1, j) == 0 && GetRstBValue(i-1, j) == 0)
                                                        {
break;
                                                        }
                                                        else
                                                        {
GrayValue = GetRstGrayValue(i-1, j-1);
```

```
if (GrayValue == 255)
{
        m_cRstColor[i-1][j-1] = RGB(255,0,0);
        i = i - 1;
        j = j - 1;
        vector = 1;

}
else if (GetRstRValue(i-1, j-1) == 255 && GetRstGValue(i-1, j-1) == 0 && GetRstBValue(i-1, j-1) == 0)
{
        break;
}
else
{
        GrayValue = GetRstGrayValue(i, j-1);
        if (GrayValue == 255)
        {
                m_cRstColor[i][j-1] = RGB(255,0,0);
                j = j - 1;
                vector = 2;
```

```
}
else if (GetRstRValue(i, j-1) == 255 && GetRstGValue(i, j-1) == 0 && GetRstBValue(i, j-1) ==
0)
{
        break;
}
else
{
        GrayValue = GetRstGrayValue(i+1, j-1);
        if (GrayValue == 255)
        {
                m_cRstColor[i+1][j-1] = RGB(255,0,0);
                i = i + 1;
                j = j - 1;
                vector = 3;
        }
        else if (GetRstRValue(i+1, j-1) == 255 && GetRstGValue(i+1, j-1) == 0 &&
GetRstBValue(i+1, j-1) == 0)
```

```
                {
                        break;
                }
        }
}
                                                                        }
                                                                }
                                                        }
                                                }
                                        }
                                        break;
                                }
                        case 6:
                                {
                                        GrayValue = GetRstGrayValue(i+1, j+1);
                                        if (GrayValue == 255)
                                        {
                                                m_cRstColor[i+1][j+1] = RGB(255,0,0);
                                                i = i + 1;
                                                j = j + 1;
                                                vector = 5;

                                        }
                                        else if (GetRstRValue(i+1, j+1) == 255 &&
GetRstGValue(i+1, j+1) == 0 && GetRstBValue(i+1, j+1) == 0)
                                        {
                                                break;
                                        }
                                        else
                                        {
                                                GrayValue = GetRstGrayValue(i, j+1);
                                                if (GrayValue == 255)
                                                {
                                                        m_cRstColor[i][j+1] = RGB(255,0,0);
                                                        j = j + 1;
```

```
                    vector = 6;

                }
                else if (GetRstRValue(i, j+1) == 255 &&
GetRstGValue(i, j+1) == 0 && GetRstBValue(i, j+1) == 0)
                {
                    break;
                }
                else
                {
                    GrayValue = GetRstGrayValue(i-1,
j+1);
                    if (GrayValue == 255)
                    {
                        m_cRstColor[i-1][j+1] =
RGB(255,0,0);
                        i = i - 1;
                        j = j + 1;
                        vector = 7;

                    }
                    else if (GetRstRValue(i-1, j+1) ==
255 && GetRstGValue(i-1, j+1) == 0 && GetRstBValue(i-1, j+1) == 0)
                    {
                        break;
                    }
                    else
                    {
                        GrayValue =
GetRstGrayValue(i-1, j);
                        if (GrayValue == 255)
                        {
m_cRstColor[i-1][j] = RGB(255,0,0);
                            i = i - 1;
                            vector = 8;

                        }
                        else if
```

```
(GetRstRValue(i-1, j) == 255 && GetRstGValue(i-1, j) == 0 && GetRstBValue(i-1, j) == 0)
                                                                    {
                                                                        break;
                                                                    }
                                                                    else
                                                                    {
                                                                        GrayValue =
GetRstGrayValue(i-1, j-1);
                                                                        i f
(GrayValue == 255)
                                                                        {

m_cRstColor[i-1][j-1] = RGB(255,0,0);
                                                                            i
= i - 1;
                                                                            j
= j - 1;
vector = 1;
                                                                        }
                                                                        else if
(GetRstRValue(i-1, j-1) == 255 && GetRstGValue(i-1, j-1) == 0 && GetRstBValue(i-1, j-1) == 0)
                                                                        {

break;
                                                                        }
                                                                        else
                                                                        {

GrayValue = GetRstGrayValue(i, j-1);
if (GrayValue == 255)
{
        m_cRstColor[i][j-1] = RGB(255,0,0);
        j = j - 1;
```

```
            vector = 2;


}
else if (GetRstRValue(i, j-1) == 255 && GetRstGValue(i, j-1) == 0 && GetRstBValue(i, j-1) == 0)
{
            break;
}
else
{
            GrayValue = GetRstGrayValue(i+1, j-1);
            if (GrayValue == 255)
            {
                        m_cRstColor[i+1][j-1] = RGB(255,0,0);
                        i = i + 1;
                        j = j - 1;
                        vector = 3;


            }
            else if (GetRstRValue(i+1, j-1) == 255 && GetRstGValue(i+1, j-1) == 0 && GetRstBValue(i+1
j-1) == 0)
```

```
        {
                break;
        }
        else
        {
                GrayValue = GetRstGrayValue(i+1, j);
                if (GrayValue == 255)
                {
                        m_cRstColor[i+1][j] = RGB(255,0,0);
                        i = i + 1;
                        vector = 4;

                }
                else if (GetRstRValue(i+1, j) == 255 && GetRstGValue(i+1, j) == 0 &&
GetRstBValue(i+1, j) == 0)
                {
                        break;
                }
        }
}
```

```
                                                                    }
                                                                }
                                                            }
                                                        }
                                                        break;
                                                    }
                                                case 7:
                                                    {
                                                        GrayValue = GetRstGrayValue(i, j+1);
                                                        if (GrayValue == 255)
                                                        {
                                                            m_cRstColor[i][j+1] = RGB(255,0,0);
                                                            j = j + 1;
                                                            vector = 6;

                                                        }
                                                        else if (GetRstRValue(i, j+1) == 255 && GetRstGValue(i,
j+1) == 0 && GetRstBValue(i, j+1) == 0)
                                                        {
                                                            break;
                                                        }
                                                        else
                                                        {
                                                            GrayValue = GetRstGrayValue(i-1, j+1);
                                                            if (GrayValue == 255)
                                                            {
                                                                m_cRstColor[i-1][j+1]
RGB(255,0,0);

                                                                i = i - 1;
                                                                j = j + 1;
                                                                vector = 7;

                                                            }
                                                            else if (GetRstRValue(i-1, j+1) == 255 &&
GetRstGValue(i-1, j+1) -- 0 && GetRstBValue(i-1, j+1) == 0)
                                                            {
                                                                break;
                                                            }
```

```
                                        else
                                        {
                                                GrayValue = GetRstGrayValue(i-1,
j);
                                                if (GrayValue == 255)
                                                {
                                                        m_cRstColor[i-1][j]
RGB(255,0,0);
                                                        i = i - 1;
                                                        vector = 8;

                                                }
                                                else if (GetRstRValue(i-1, j) == 255
&& GetRstGValue(i-1, j) == 0 && GetRstBValue(i-1, j) == 0)
                                                {
                                                        break;
                                                }
                                                else
                                                {
                                                        GrayValue
GetRstGrayValue(i-1, j-1);
                                                        if (GrayValue == 255)
                                                        {

m_cRstColor[i-1][j-1] = RGB(255,0,0);
                                                                i = i - 1;
                                                                j = j - 1;
                                                                vector = 1;

                                                        }
                                                        else if
(GetRstRValue(i-1, j-1) == 255 && GetRstGValue(i-1, j-1) == 0 && GetRstBValue(i-1, j-1) == 0)
                                                        {
                                                                break;
                                                        }
                                                        else
                                                        {
                                                                GrayValue =
GetRstGrayValue(i, j-1);
```

```
                                                                    i          f
(GrayValue == 255)
                                                                    {

m_cRstColor[i][j-1] = RGB(255,0,0);
                                                                              j
= j - 1;

vector = 2;

                                                                    }
                                                                    else       if
(GetRstRValue(i, j-1) == 255 && GetRstGValue(i, j-1) == 0 && GetRstBValue(i, j-1) == 0)
                                                                    {

break;
                                                                    }
                                                                    else
                                                                    {

GrayValue = GetRstGrayValue(i+1, j-1);

if (GrayValue == 255)

{

        m_cRstColor[i+1][j-1] = RGB(255,0,0);

        i = i + 1;

        j = j - 1;

        vector = 3;



}

else if (GetRstRValue(i+1, j-1) == 255 && GetRstGValue(i+1, j-1) == 0 && GetRstBValue(i+1, j-1) == 0)
```

```
{
            break;
}
else
{
            GrayValue = GetRstGrayValue(i+1, j);
            if (GrayValue == 255)
            {
                        m_cRstColor[i+1][j] = RGB(255,0,0);
                        i = i + 1;
                        vector = 4;

            }
            else if (GetRstRValue(i+1, j) == 255 && GetRstGValue(i+1, j) == 0 && GetRstBValue(i+1, j)
0)
            {
                        break;
            }
            else
            {
```

```
                    GrayValue = GetRstGrayValue(i+1, j+1);

                    if (GrayValue == 255)

                    {

                            m_cRstColor[i+1][j+1] = RGB(255,0,0);

                            i = i + 1;

                            j = j + 1;

                            vector = 5;



                    }

                    else if (GetRstRValue(i+1, j+1) == 255 && GetRstGValue(i+1, j+1) == 0 &&
GetRstBValue(i+1, j+1) == 0)

                    {

                            break;

                    }

            }

}
                                                                                    }
                                                                            }
                                                                    }
                                                            }
                                                    }
                                                    break;
                                            }
                    case 8:
```

```
{
        GrayValue = GetRstGrayValue(i-1, j+1);
        if (GrayValue == 255)
        {
                m_cRstColor[i-1][j+1] = RGB(255,0,0);
                i = i - 1;
                j = j + 1;
                vector = 7;

        }
        else if (GetRstRValue(i-1, j+1) == 255 &&
GetRstGValue(i-1, j+1) == 0 && GetRstBValue(i-1, j+1) == 0)
        {
                break;
        }
        else
        {
                GrayValue = GetRstGrayValue(i-1, j);
                if (GrayValue == 255)
                {
                        m_cRstColor[i-1][j] = RGB(255,0,0);
                        i = i - 1;
                        vector = 8;

                }
                else if (GetRstRValue(i-1, j) == 255 &&
GetRstGValue(i-1, j) == 0 && GetRstBValue(i-1, j) == 0)
                {
                        break;
                }
                else
                {
                        GrayValue = GetRstGrayValue(i-1,
j-1);
                        if (GrayValue == 255)
                        {
                                m_cRstColor[i-1][j-1] =
RGB(255,0,0);
                                i = i - 1;
```

```
                    j = j - 1;
                    vector = 1;

                }
                else if (GetRstRValue(i-1, j-1) ==
255 && GetRstGValue(i-1, j-1) == 0 && GetRstBValue(i-1, j-1) == 0)
                {
                    break;
                }
                else
                {
                    GrayValue
GetRstGrayValue(i, j-1);
                    if (GrayValue == 255)
                    {

m_cRstColor[i][j-1] = RGB(255,0,0);
                        j = j - 1;
                        vector = 2;

                    }
                    else if (GetRstRValue(i,
j-1) == 255 && GetRstGValue(i, j-1) == 0 && GetRstBValue(i, j-1) == 0)
                    {
                        break;
                    }
                    else
                    {
                        GrayValue =
GetRstGrayValue(i+1, j-1);
                        i f
(GrayValue == 255)
                        {

m_cRstColor[i+1][j-1] = RGB(255,0,0);
                            i
= i + 1;
                            j
= j - 1;
```

```
vector = 3;
                                                                    }
                                                                    else        if
(GetRstRValue(i+1, j-1) == 255 && GetRstGValue(i+1, j-1) == 0 && GetRstBValue(i+1, j-1) == 0)
                                                                    {

break;
                                                                    }
                                                                    else
                                                                    {

GrayValue = GetRstGrayValue(i+1, j);

if (GrayValue == 255)

{

        m_cRstColor[i+1][j] = RGB(255,0,0);

        i = i + 1;

        vector = 4;



}

else if (GetRstRValue(i+1, j) == 255 && GetRstGValue(i+1, j) == 0 && GetRstBValue(i+1, j) == 0)

{

        break;

}

else
```

```
{
        GrayValue = GetRstGrayValue(i+1, j+1);
        if (GrayValue == 255)
        {
                m_cRstColor[i+1][j+1] = RGB(255,0,0);
                i = i + 1;
                j = j + 1;
                vector = 5;

        }
        else if (GetRstRValue(i+1, j+1) == 255 && GetRstGValue(i+1, j+1) == 0 && GetRstBValue(i+1
j+1) == 0)
        {
                break;
        }
        else
        {
                GrayValue = GetRstGrayValue(i, j+1);
                if (GrayValue == 255)
                {
```

```
                    m_cRstColor[i][j+1] = RGB(255,0,0);

                    j = j + 1;

                    vector = 6;



                }

                else if (GetRstRValue(i, j+1) == 255 && GetRstGValue(i, j+1) == 0 &&
GetRstBValue(i, j+1) == 0)

                {

                    break;

                }

        }

}
                                                                        }
                                                                    }
                                                                }
                                                            }
                                                        }
                                                        break;
                                                    }
                }
                if (GetRstRValue(i, j) == 255 && GetRstGValue(i, j) == 0 && GetRstBValue(i, j) ==
0)
                {
                    break;
                }
        }
}
```

```
// CucumberProcess.cpp : Defines the class behaviors for the application.
//

#include "stdafx.h"
#include "CucumberProcess.h"
#include "CucumberProcessDlg.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CCucumberProcessApp

BEGIN_MESSAGE_MAP(CCucumberProcessApp, CWinApp)
        //{{AFX_MSG_MAP(CCucumberProcessApp)
                // NOTE - the ClassWizard will add and remove mapping macros here
                //    DO NOT EDIT what you see in these blocks of generated code!
        //}}AFX_MSG
        ON_COMMAND(ID_HELP, CWinApp::OnHelp)
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CCucumberProcessApp construction

CCucumberProcessApp::CCucumberProcessApp()
{
        // TODO: add construction code here,
        // Place all significant initialization in InitInstance
}

/////////////////////////////////////////////////////////////////////////////
// The one and only CCucumberProcessApp object

CCucumberProcessApp theApp;
```

```
/////////////////////////////////////////////////////////////////////////////
// CCucumberProcessApp initialization

BOOL CCucumberProcessApp::InitInstance()
{
        AfxEnableControlContainer();

        // Standard initialization
        // If you are not using these features and wish to reduce the size
        //  of your final executable, you should remove from the following
        //  the specific initialization routines you do not need.

#ifdef _AFXDLL
        Enable3dControls();                     // Call this when using MFC in a shared DLL
#else
        Enable3dControlsStatic();       // Call this when linking to MFC statically
#endif

        CCucumberProcessDlg dlg;
        m_pMainWnd = &dlg;
        int nResponse = dlg.DoModal();
        if (nResponse == IDOK)
        {
                // TODO: Place code here to handle when the dialog is
                //  dismissed with OK
        }
        else if (nResponse == IDCANCEL)
        {
                // TODO: Place code here to handle when the dialog is
                //  dismissed with Cancel
        }

        // Since the dialog has been closed, return FALSE so that we exit the
        //  application, rather than start the application's message pump.
        return FALSE;
}
```

```
// CucumberProcessDlg.cpp : implementation file
//

#include "stdafx.h"
#include "CucumberProcess.h"
#include "CucumberProcessDlg.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CAboutDlg dialog used for App About

class CAboutDlg : public CDialog
{
public:
        CAboutDlg();

// Dialog Data
        //{{AFX_DATA(CAboutDlg)
        enum { IDD = IDD_ABOUTBOX };
        //}}AFX_DATA

        // ClassWizard generated virtual function overrides
        //{{AFX_VIRTUAL(CAboutDlg)
        protected:
        virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
        //}}AFX_VIRTUAL

// Implementation
protected:
        //{{AFX_MSG(CAboutDlg)
        //}}AFX_MSG
        DECLARE_MESSAGE_MAP()
};
```

```
CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
{
	//{{AFX_DATA_INIT(CAboutDlg)
	//}}AFX_DATA_INIT
}

void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialog::DoDataExchange(pDX);
	//{{AFX_DATA_MAP(CAboutDlg)
	//}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
	//{{AFX_MSG_MAP(CAboutDlg)
		// No message handlers
	//}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CCucumberProcessDlg dialog

CCucumberProcessDlg::CCucumberProcessDlg(CWnd* pParent /*=NULL*/)
	: CDialog(CCucumberProcessDlg::IDD, pParent)
{
	//{{AFX_DATA_INIT(CCucumberProcessDlg)
		// NOTE: the ClassWizard will add member initialization here
	//}}AFX_DATA_INIT
	// Note that LoadIcon does not require a subsequent DestroyIcon in Win32
	m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);

	m_pImageDev = NULL;
}

CCucumberProcessDlg::~CCucumberProcessDlg()
{
	if (m_pImageDev != NULL)
	{
		delete m_pImageDev;
```

```
                m_pImageDev = NULL;
        }
        delete m_pImageSource;
}

void CCucumberProcessDlg::DoDataExchange(CDataExchange* pDX)
{
        CDialog::DoDataExchange(pDX);
        //{{AFX_DATA_MAP(CCucumberProcessDlg)
                // NOTE: the ClassWizard will add DDX and DDV calls here
        //}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CCucumberProcessDlg, CDialog)
        //{{AFX_MSG_MAP(CCucumberProcessDlg)
        ON_WM_SYSCOMMAND()
        ON_WM_PAINT()
        ON_WM_QUERYDRAGICON()
        ON_BN_CLICKED(IDC_OVERLAY_DEV, OnOverlayDev)
        ON_BN_CLICKED(IDC_PREVIEW_DEV, OnPreviewDev)
        ON_COMMAND(IDM_VIDEOCOMP_DLG, OnVideocompDlg)
        ON_COMMAND(IDM_VIDEOFORMAT_DLG, OnVideoformatDlg)
        ON_COMMAND(IDM_VIDEOSOURCE_DLG, OnVideosourceDlg)
        ON_COMMAND(ID_ABOUT_DLG, OnAboutDlg)
        ON_BN_CLICKED(IDC_GRAB_BUTTON, OnGrabButton)
        ON_COMMAND(ID_BMPFILE_OPEN, OnBmpfileOpen)
        ON_COMMAND(ID_BMPFILE_SAVE, OnBmpfileSave)
        //}}AFX_MSG_MAP
END_MESSAGE_MAP()

////////////////////////////////////////////////////////////////////////////
// CCucumberProcessDlg message handlers

BOOL CCucumberProcessDlg::OnInitDialog()
{
        CDialog::OnInitDialog();

        // Add "About..." menu item to system menu.
```

```
	// IDM_ABOUTBOX must be in the system command range.
	ASSERT((IDM_ABOUTBOX & 0xFFF0) == IDM_ABOUTBOX);
	ASSERT(IDM_ABOUTBOX < 0xF000);

	CMenu* pSysMenu = GetSystemMenu(FALSE);
	if (pSysMenu != NULL)
	{
		CString strAboutMenu;
		strAboutMenu.LoadString(IDS_ABOUTBOX);
		if (!strAboutMenu.IsEmpty())
		{
			pSysMenu->AppendMenu(MF_SEPARATOR);
			pSysMenu->AppendMenu(MF_STRING, IDM_ABOUTBOX
strAboutMenu);
		}
	}

	// Set the icon for this dialog.  The framework does this automatically
	//  when the application's main window is not a dialog
	SetIcon(m_hIcon, TRUE);			// Set big icon
	SetIcon(m_hIcon, FALSE);		// Set small icon

	// TODO: Add extra initialization here
	m_pCameraWnd = GetDlgItem(IDC_CAMERA_MON);
	m_pViewWnd = GetDlgItem(IDC_GRAB_MON);

	m_pImageDev = new CImageDev(m_pCameraWnd);
	m_pImageSource = new CImageSource();

	int w, h;
	w = m_pImageDev->GetSizeWidthCapWindow();
	h =	m_pImageDev->GetSizeHeightCapWindow();

	m_pImageDev->Overlay();

	return TRUE;  // return TRUE  unless you set the focus to a control


void CCucumberProcessDlg::OnSysCommand(UINT nID, LPARAM lParam)
```

```
{
        if ((nID & 0xFFF0) == IDM_ABOUTBOX)
        {
                CAboutDlg dlgAbout;
                dlgAbout.DoModal();
        }
        else
        {
                CDialog::OnSysCommand(nID, lParam);
        }
}

// If you add a minimize button to your dialog, you will need the code below
//  to draw the icon.  For MFC applications using the document/view model,
//  this is automatically done for you by the framework.

void CCucumberProcessDlg::OnPaint()
{
        if (IsIconic())
        {
                CPaintDC dc(this); // device context for painting

                SendMessage(WM_ICONERASEBKGND, (WPARAM) dc.GetSafeHdc(), 0)

                // Center icon in client rectangle
                int cxIcon = GetSystemMetrics(SM_CXICON);
                int cyIcon = GetSystemMetrics(SM_CYICON);
                CRect rect;
                GetClientRect(&rect);
                int x = (rect.Width() - cxIcon + 1) / 2;
                int y = (rect.Height() - cyIcon + 1) / 2;

                // Draw the icon
                dc.DrawIcon(x, y, m_hIcon);
        }
        else
        {
                CDialog::OnPaint();
        }
```

```
}

// The system calls this to obtain the cursor to display while the user drags
//  the minimized window.
HCURSOR CCucumberProcessDlg::OnQueryDragIcon()
{
        return (HCURSOR) m_hIcon;
}

void CCucumberProcessDlg::OnOverlayDev()
{
        // TODO: Add your control notification handler code here
        m_pImageDev->Overlay();
}

void CCucumberProcessDlg::OnPreviewDev()
{
        // TODO: Add your control notification handler code here
        m_pImageDev->Preview();
}

void CCucumberProcessDlg::OnVideocompDlg()
{
        // TODO: Add your command handler code here
        m_pImageDev->VideoCompressionDlg();
}

void CCucumberProcessDlg::OnVideoformatDlg()
{
        // TODO: Add your command handler code here
        m_pImageDev->VideoFormatDlg();
}

void CCucumberProcessDlg::OnVideosourceDlg()
{
        // TODO: Add your command handler code here
        m_pImageDev->VideoSourceDlg();
}
```

```
void CCucumberProcessDlg::OnAboutDlg()
{
        // TODO: Add your command handler code here
        CAboutDlg dlgAbout;
        dlgAbout.DoModal();
}

void CCucumberProcessDlg::OnGrabButton()
{
        // TODO: Add your control notification handler code here
        CDC * pCamDC;
        CDC * pGrabDC;

        pCamDC = m_pCameraWnd->GetDC();
        pGrabDC = m_pViewWnd->GetDC();
        int i, j;

        m_pImageDev->Preview();
        m_pImageDev->GrabFrame();

        for (j=0; j<240; j++)
        {
                for (i=0; i<320; i++)
                {
                        m_pImageSource->m_cColor[i][j] = RGB(0, 0, 0);
                }
        }

        for (j=0; j<240; j++)
        {
                for (i=0; i<320; i++)
                {
                        m_pImageSource->m_cColor[i][j] = pCamDC->GetPixel(i, j)
                }
        }

        for(j=0; j<240; j++)
        {
                for(i=0; i<320; i++)
```

```
                {
                        pGrabDC->SetPixel(i, j, m_pImageSource->m_cColor[i][j]);
                }
        }

        m_pImageDev->Overlay();

        ReleaseDC(pCamDC);
        ReleaseDC(pGrabDC);
}


void CCucumberProcessDlg::OnBmpfileOpen()
{
        // TODO: Add your command handler code here
        CFileDialog    dlgBmpOpen(TRUE,    "BMP",    NULL,    OFN_ALLOWMULTISELECT
OFN_FILEMUSTEXIST,
                "BMP Files(*.bmp)|*.bmp|All Files(*.*)|*.*||", NULL);
        if (dlgBmpOpen.DoModal() == IDOK)
        {
                CString strTemp;
                POSITION pos = dlgBmpOpen.GetStartPosition();

                while (pos)
                {
                        strTemp += dlgBmpOpen.GetNextPathName(pos);
                        strTemp += "\r\n";
                }
        }
}

void CCucumberProcessDlg::OnBmpfileSave()
{
        // TODO: Add your command handler code here
        CFileDialog dlgBmpSave(FALSE, "BMP", NULL, OFN_FILEMUSTEXIST,
                "BMP Files(*.bmp)|*.bmp|All Files(*.*)|*.*||", NULL);
        if (dlgBmpSave.DoModal() == IDOK)
        {
                CString strTemp;
```

```
                    strTemp += dlgBmpSave.GetFileName();
            }
}


// imageDev.cpp: implementation of the CImageDev class.
//
//////////////////////////////////////////////////////////////////////

#include "stdafx.h"
#include "imageDev.h"

#ifdef _DEBUG
#undef THIS_FILE
static char THIS_FILE[]=__FILE__;
#define new DEBUG_NEW
#endif

//////////////////////////////////////////////////////////////////////
// Construction/Destruction
//////////////////////////////////////////////////////////////////////

CImageDev::CImageDev(CWnd * ghWndMain) {
            m_gwPalFrames = DEF_PALNUMFRAMES;
            m_gwPalColors = DEF_PALNUMCOLORS;

            m_nWidth = 320;
            m_nHeight = 240;

            TCHAR    achDeviceName[80];
            TCHAR    achDeviceVersion[100];
            TCHAR    achBuffer[100];
            WORD     wDriverCount = 0;
            WORD     wIndex;
            DWORD    dwError;

    // First create the capture window
    m_ghWndCap.m_hWnd = capCreateCaptureWindow((LPTSTR)TEXT("Capture Window"),
```

```
                                        WS_CHILD | WS_VISIBLE,
                                        0, 0, m_nWidth, m_nHeight,
                                        (HWND)ghWndMain->m_hWnd, (int)0) ;

        // Try to connect one of the MSVIDEO drivers
        capDriverDisconnect(m_ghWndCap.m_hWnd);

        for (wIndex = 0 ; wIndex < MAXVIDDRIVERS ; wIndex++) {
                if (capGetDriverDescription(wIndex,
                        (LPTSTR)achDeviceName, sizeof(achDeviceName)/ sizeof(TCHAR),
                        (LPTSTR)achDeviceVersion, sizeof(achDeviceVersion)/sizeof(TCHAR))) {

                        // There is such a driver in the "system.ini" file.
                        // Append driver name to "Options" list in menu
                        wsprintf(achBuffer,     TEXT("&%d     %s"),     wIndex
(LPTSTR)achDeviceName) ;

                        if (wDriverCount++ == 0) {
                                // Only if no other driver is already connected
                                dwError     =     capDriverConnect(m_ghWndCap.m_hWnd
wIndex);
                                if (dwError) {
                                        m_gwDeviceIndex = wIndex ;
                                } // end if
                        } // end if
                } // end if
        } // end for
        capDriverGetCaps(m_ghWndCap.m_hWnd, &m_gCapDriverCaps, sizeof(CAPDRIVERCAPS)) ;

    // Get video format and adjust capture window
    capGetStatus(m_ghWndCap.m_hWnd, &m_gCapStatus, sizeof(CAPSTATUS)) ;

    // Start preview by default
    capPreviewRate(m_ghWndCap.m_hWnd, MS_FOR_15FPS) ;
    capPreview(m_ghWndCap.m_hWnd, FALSE);
        capOverlay(m_ghWndCap.m_hWnd, FALSE);
} // end CImageDev::CImageDev

CImageDev::~CImageDev() {
```

```
        capDriverDisconnect(m_ghWndCap.m_hWnd);
} // end CImageDev::~CImageDev

CWnd *    CImageDev::getCapWnd() {
        return &m_ghWndCap;
} // end CImageDev::getCapWnd

void CImageDev::Preview() {
        // Toggle Preview
        capGetStatus(m_ghWndCap.m_hWnd, &m_gCapStatus, sizeof(CAPSTATUS))
        capPreview(m_ghWndCap.m_hWnd, !m_gCapStatus.fLiveWindow);
} // end CImageDev::preview

void CImageDev::Overlay() {
        // Toggle Overlay
        capGetStatus(m_ghWndCap.m_hWnd, &m_gCapStatus, sizeof(CAPSTATUS))
        capOverlay(m_ghWndCap.m_hWnd, !m_gCapStatus.fOverlayWindow);

} // end CImageDev::overLay

int CImageDev::GetSizeWidthCapWindow()
{
        return m_nWidth;
} // end CImageDev::getSizeWidthCapWindow

int CImageDev::GetSizeHeightCapWindow()
{
        return m_nHeight;
} // end CImageDev::getSizeHeightCapWindow

void CImageDev::VideoFormatDlg()
{
        capDlgVideoFormat(m_ghWndCap.m_hWnd);
}

void CImageDev::VideoSourceDlg()
{
        capDlgVideoSource(m_ghWndCap.m_hWnd);
}
```

```
void CImageDev::VideoCompressionDlg()
{
        capDlgVideoCompression(m_ghWndCap.m_hWnd);
}

void CImageDev::GrabFrame()
{
        capGrabFrameNoStop(m_ghWndCap.m_hWnd);
}

void CImageDev::PalletteAuto()
{
        capPaletteAuto(m_ghWndCap.m_hWnd, 0, 256);
}

// ImageSource.cpp: implementation of the CImageSource class
//
//////////////////////////////////////////////////////////////////////

#include "stdafx.h"
#include "ImageSource.h"

#ifdef _DEBUG
#undef THIS_FILE
static char THIS_FILE[]=__FILE__;
#define new DEBUG_NEW
#endif

//////////////////////////////////////////////////////////////////////
// Construction/Destruction
//////////////////////////////////////////////////////////////////////
CImageSource::CImageSource()
{

}
CImageSource::~CImageSource()
{
}
```

매니퓰레이터 구동 프로그램

```
#include <stdio.h>        /* required header for printf, putchar */
#include <io51.h>         /* required header for output, bit_set, etc. */

typedef unsigned char BYTE;
typedef idata unsigned char BYTE1;
typedef unsigned int WORD;

#define EXT_ADR(address)    (((char *)0x010000)[address])
#define REST_KEY(address)    (((char *)0x030000)[address])

#define D_IN        P1.0
#define D_OUT       P1.1
#define LCD_DIS     P1.2
#define STEP_MOT    P1.3
#define C46_CS      P1.4
#define C46_SK      P1.5
#define C46_DI      P1.6
#define C46_DO      P1.7

BYTE usec, msec, sec, hlf_sec;
WORD  Mcnt;
BYTE EnCnt, Adr;

BYTE CmpSpeed;
BYTE Speed;
BYTE KeyReadTime;
BYTE KeyBog, MotorWaite;
BYTE IntTemp;
BYTE y_motor_speed, x_motor_speed;
BYTE RxdBuf[20], RxdT;

WORD x_motor_data, y_motor_data;
bit RcvFull, MotorOn, UpDown;
bit LcdDisBit;

extern void lcd_init(void);
```

```
extern void lcd_set(void);
extern void lcd_clear(void);

extern void lcd_gotoxy(char x, char y);
extern void lcd_putch(char ch);
extern void lcd_cursoron(void);
extern void lcd_cursoroff(void);

interrupt[0x0B] void T0_int (void);

void diplay(void);
void inty_speed(void);
void intx_speed(void);

void start_int(void);
void SendTxd(BYTE Aval);
void timer_init(void);
void RunMotor(void);
void DataDisPlay(void);
void RcvSnd(void);

interrupt [0x23] void SCON_int (void)
{
        BYTE Aval;
        EA =0;
        if(RI == 1){
                RI = 0;
                Aval = SBUF;
                if(Aval == 0x1e) RxdT = 0, RcvFull = 0;
                if(RxdT < 19) RxdBuf[RxdT++] = Aval;
                if(Aval == 0x1f) RcvFull = 1;
        }

        if(TI == 1){
                TI = 0;
        }

        EA =1;
}
```

```
void timer_com_init(void)
{
        TMOD  = 0x21;                   /* timer0 : 3, timer1 : 2          */

        TL0 = 200;
        TH0 = 0xff;
        /* TH1 = 0xfd; * 9600 burate 11.0592 */
        TH1  = 0xFA;
        /* PCON = 0;  burate */
        ES = 1;     /* serial interupt enable */
        TR0 =1;    /* run timer  */
        TR1 = 1;
        PS = 1;
        ET1 = 1;
        SCON = 0x50;
        ET0 = 1; /* timer1 interrupt */
        EX1 =0;
        EA  = 1;       /* enable inteerut */
}


interrupt[0x0B] void T0_int (void)

/*
        real-time clock process
        conter value for  software timer
        fosc=6mhz, fcontes=6mhz/12 =0.5mhz,pcount=2usec
        hense counts # for 100msec is 100,000/2=50,000
        conter valuc = -50,000
*/
 {
        EA =0;
        TR0 = 0;
        if(Mcnt) Mcnt--;

        EnCnt++;
        /* 1msec */
```

```
        ++usec;
        if(usec >= 8){
        /* 4msec */
          usec = 0;
          ++msec;

/*
          if(TL0 > Speed)     {     if(CmpSpeed) CmpSpeed--;              }
          else if(TL0 < Speed){           if(CmpSpeed < 68) CmpSpeed++; }
          Aval = TL0;
          TL0 = 0;
 */

        }

        if(msec >= 10){
        /* 40msec */
         msec = 0;
         ++sec;
         ++hlf_sec;
        }

        if(sec >= 10){
         /* 400 msec */
         sec = 0;
        /*  SBUF = CmpSpeed; */
        }

        if(MotorWaite)  MotorWaite--;
        else{MotorWaite=10;                                    RunMotor();}

        if(KeyReadTime)   KeyReadTime--;
        TL0 = 0x248;
        TH0 = 0xff;
        TR0 = 1;
        EA =1;
}

void RunMotor(void)
```

```
{
        BYTE  i, n;
        Speed &=  0xfc;
        if(MotorOn -- 0){
                P2 = 0;
                STEP_MOT = 1;  STEP_MOT = 0;
        } else  {
                n = Speed;
                IntTemp = 0;
                for(i=0; i<8; i++){
                  if(n & 1) IntTemp |= 1;
                  n >>= 1;  IntTemp <<= 1;
                }
                if(UpDown){ P2 = IntTemp | 1; STEP_MOT = 1; STEP_MOT = 0;
                } else    { P2 = IntTemp | 2; STEP_MOT = 1; STEP_MOT = 0;}
        }

}

void start_int(void)
{
        P1.4 = 0;
        RcvFull = 0;
        RxdT = 0;
        MotorOn = 0;
        Speed = 1;
        P1 = 1;
        CmpSpeed = 12;
        Mcnt = 30;
}

void DeselSpeed(void)
{

  while(1){
  /*
          P2 = Speed;
          STEP_MOT = 1;  STEP_MOT = 0;
        */
```

```
            if(Speed <= y_motor_speed) break;
            Speed--;
            Mcnt = 2000;
            while(Mcnt);
  }
/*  if(Speed == 0) INT0 = 0, INT1 = 0; */
}

void AcselSpeed(void)
{

  if(y_motor_speed > 64) y_motor_speed = 64;
  while(1){
  /*
             P2 = Speed;
            STEP_MOT = 1;  STEP_MOT = 0;
           */
            if(Speed >= y_motor_speed) break;
            Speed++;
            Mcnt = 2000;
            while(Mcnt);
  }

}

void ExtRcv(void)
{
          BYTE i;
/*  commuicatuon protocol
          1) start command  '0x1e'
                    * 1byte
          2) Adrress        0;
                    * 0 = 0x30
                    * 1byte
                    * '1' = Step motor
                    * '2' = Dc motor

          3) command        " A --> Z "
                    * 1byte
```

```
                * 'A' = start;
                * 'B' = Stop;

        4) Speed Data      " 000 --> 999 "
                * 000 = 0x30, 0x30, 0x30
                * 3byte
        5) how long time  " 0000 --> 9999 "
                * 000 = 0x30, 0x30, 0x30
                * 4byte

        6) UP or down      " 1 or 0     "
                *  0  =  0x30
                *  1  =  0x31

        7) End command     '0x1f'
*/
  if(0x1e != RxdBuf[0]) return;
  i = 1;
  Adr = 0x32; /* DC motor Address 2 */
  if(Adr != RxdBuf[1]) return;
  RcvSnd();
  switch(RxdBuf[2]){

          case 'A':
                        LcdDisBit = 1;
                        y_motor_speed =  (RxdBuf[3] & 0x0f) * 100;
                        y_motor_speed +=  (RxdBuf[4] & 0x0f) * 10;
                        y_motor_speed +=  (RxdBuf[5] & 0x0f);

                        y_motor_data +=  (RxdBuf[6] & 0x0f) * 1000;
                        y_motor_data +=  (RxdBuf[7] & 0x0f) * 100;
                        y_motor_data +=  (RxdBuf[8] & 0x0f) * 10;
                        y_motor_data +=  (RxdBuf[9] & 0x0f);


                        if(RxdBuf[10] == 0x30){
                                if(UpDown == 0){ Speed = 0; MotorOn = 0; Mcnt = 10000
while(Mcnt);}
                                UpDown = 1;
```

```
                                } else {
                                        if(UpDown == 1){ Speed = 0; MotorOn = 0; Mcnt = 10000
while(Mcnt);}
                                        UpDown = 0;
                                }

                                MotorOn = 1;
                                DataDisPlay();
                                if(y_motor_speed < Speed)       DeselSpeed();
                                else if(y_motor_speed > Speed) AcselSpeed();

        break;

        case 'B':
                                MotorOn = 0;
                                LcdDisBit = 1;
                                DataDisPlay();
        break;

  }

 RxdBuf[0] = 0;

}

void SendTxd(BYTE Aval)
{
                while(Mcnt);
                SBUF = Aval;
                Mcnt = 30;

}

void ReadKey(void)
{
        BYTE Aval;

        if(KeyReadTime){ return;}
        P0.0 = 1; P0.1 = 1;
```

```
            Aval = P0;
            Aval &= 3;
            if(Aval == KeyBog){ return;}
            KeyReadTime = 100;
            KeyBog = Aval;


            if((Aval & 1) == 0){ MotorOn = 1;
                                                                                MotorWaite
= 100; UpDown-1; if(Speed) Speed--; SendTxd(Speed);return;

            if((Aval & 2) == 0){ MotorWaite = 100; if(Speed < 24) Speed++;

                        SendTxd(Speed);return;


}

/*
void TestMotor(void)
{
 BYTE i;
 WORD n;
  INT1 =0;
  while(1){
                        INT0 =1;
                        for(i=0; i<30; i++){
                                    if(i == CmpSpeed){ INT0 =0;
                                    }
                        }
                        ReadKey();
                        .
  }
}
*/

void LcdString(char *Ptr)
{
            while(1){
```

```
            if(*Ptr == 0) break;
                        lcd_putch(*Ptr); Ptr++;

          }

}

void SendString(char *Ptr)
{
          while(1){
            if(*Ptr == 0) break;
                        SendTxd(*Ptr); Ptr++;

          }

}

void RcvSnd(void){
          lcd_gotoxy(0, 0);
          RxdBuf[12] = 0;
  /*      LcdString(&RxdBuf[0]); */
          SendString(&RxdBuf[0]);
}



void DataDisPlay(void)
{
  int Aval;

  if(LcdDisBit == 0) return;

  LcdDisBit = 0;
  lcd_clear();
  lcd_gotoxy(0, 0);

  if(MotorOn){
          if(UpDown) LcdString(" Motor Up Run ");
          else       LcdString("Motor Down Run");
```

```
            lcd_gotoxy(0, 1);
            LcdString("Speed : ");
            Aval = y_motor_speed;
            Aval %= 1000;
            lcd_putch((Aval /100) + 0x30);
            Aval %= 100;
            lcd_putch((Aval / 10) + 0x30);
            lcd_putch((Aval % 10) + 0x30);

  } else {
            LcdString("  Motor Ready ");
  }

}


void main(void)
{
            BYTE i;
            start_int();
            timer_com_init();
            lcd_init();

            for(i=0; i<20; i++){
                        SendTxd(i+0x30);
            }

            UpDown = 0;
            LcdDisBit = 0;
            lcd_set();
            lcd_gotoxy(0, 0);
            LcdString("Tuin electroinc!");
            lcd_gotoxy(0, 1);
            LcdString("!Tel:02-354-5324 ");

loop:

            if(RcvFull){ RcvFull = 0; ExtRcv();}
```

```
/*
            ReadKey();
*/

 /*
            if(RcvFull) RcvFull = 0, RcvSnd();
 */
goto loop;

}



#include <stdio.h>        /* required header for printf, putchar */
#include <io51.h>         /* required header for output, bit_set, etc. */

#define LCD_DATA    P0
#define LCD_RS      T1
#define LCD_RW      INT1
#define LCD_E       INT0
#define LCD_DIS     P1.2

extern void lcd_init(void);
extern void lcd_set(void);
extern void lcd_clear(void);

extern void lcd_gotoxy(char x, char y);
extern void lcd_putch(char ch);
extern void lcd_cursoron(void);
extern void lcd_cursoroff(void);

void lcd_plus(void);


void lcd_plus(void)
{
            unsigned int i;
            LCD_DIS = 1; LCD_DIS = 0;
```

```
        LCD_E = 1;
        for(i=0; i< 10; i++) {
                if (i == 4) LCD_E = 0;
        }
}

void lcd_init(void)
{
        LCD_RS   = 0;
        LCD_RW   = 0;
        LCD_DATA = 0x38;
        lcd_plus();

        LCD_RS   = 0;
        LCD_RW   = 0;
        LCD_DATA = 0x38;
        lcd_plus();

        LCD_RS   = 0;
        LCD_RW   = 0;
        LCD_DATA = 0x0e;
        lcd_plus();

        LCD_RS   = 0;
        LCD_RW   = 0;
        LCD_DATA = 0x0e;
        lcd_plus();

        LCD_RS   = 0;
        LCD_RW   = 0;
        LCD_DATA = 0x06;
        lcd_plus();
}

void lcd_set(void)
{
        char i;
```

```
        lcd_gotoxy(0, 0);
        for (i=0; i < 16; i++) lcd_putch(0xff);

        lcd_gotoxy(0, 1);
        for (i=0; i < 16; i++) lcd_putch(0xff);
}

void lcd_clear(void)
{
        LCD_RS   = 0;
        LCD_RW   = 0;
        LCD_DATA = 1;
        lcd_plus();
}

void lcd_gotoxy(char x, char y)
{
        char i;

        if (y == 0) i = x;
        else        i = x + 0x40;

        LCD_RS   = 0;
        LCD_RW   = 0;
        LCD_DATA = i | 0x80;
        lcd_plus();
}

void lcd_putch(char ch)
{
        LCD_RS   = 1;
        LCD_RW   = 0;
        LCD_DATA = ch;
        lcd_plus();
}

void lcd_cursoron(void)
{
        LCD_RS = 0;
```

```
        LCD_RW = 0;
        LCD_DATA = 0xf;
        lcd_plus();
}


void lcd_cursoroff(void)
{
        LCD_RS   = 0;
        LCD_RW   = 0;
        LCD_DATA = 0xc;
        lcd_plus();
}



#include <io51.h>       /* required header for output, bit_set, etc. */
#include "MOTOR.H"

/* serial eeprom memory allocation */

#define MOTOR_POWER                 P1.0
#define PROM_CS             P1.1
#define PROM_CK             P1.2
#define PROM_DI             P1.3
#define PROM_DO             P1.4

#define FOW_MOTOR       INT1
#define BAK_MOTOR       INT0

#define SW_INC_MOTOR    P0.0
#define SW_DEC_MOTOR        P0.1
#define LEFT_MOTOR      P0.2
#define RIGHT_MOTOR     P0.3
#define LED_1               P0.4
#define LED_2               P0.5
#define LED_3               P0.6
#define LED_4               P0.7
```

```
/* coin mode step */


bit  SECBIT, MOTOR_ON, Mode_set, TimeBit;
bit  PowerBit;
bit  RcvFull;

BYTE PROM_MSB;
BYTE PROM_LSB;

BYTE RealSpeed;
BYTE MYCODE;
BYTE      INTCNT;                                   /*    interrupt    counter
          */

WORD USRCNT, SwReadTime;
WORD RPM_1000, RPM_100, SecTime;

BYTE RPM_10, Led_Control;
BYTE KeyMode, LedMode;


unsigned      char      SYSEROR;                        /*    user    umer
                        */
BYTE   SECCNT;                                          /*  1  mSec  timer
                  */
BYTE MsecCut;

BYTE     COINCNT;                                   /*  coin   signal   counter
          */
BYTE COIN_POS;



/* sio variable  */
char      SIO_POS;                              /*    sio    rx    data    pos
          */
```

```
BYTE SIO_BUF[8];                /* sio rx buffer max 16 bytes                */

/* BYTE  TestB; */

interrupt[0x1B] void T1_int (void);
void init_timer(void);
void init_para(void);
void Delay(WORD cnt);

void init_main(void);
void check_coin(void);
void read_eprom(char mode);
void write_eprom(char mode);

/* ********************************************************** */
/* eeprom 93c46 routine                                       */
/* ********************************************************** */

void C46_addr(char mode);
void eprom_getch(char addr);
void eprom_putch(char addr);
void C46_pulse(void);

extern void lcd_init(void);
extern void lcd_set(void);
extern void lcd_clear(void);

extern void lcd_gotoxy(char x, char y);
extern void lcd_putch(char ch);
extern void lcd_cursoron(void);
extern void lcd_cursoroff(void);


/* ********************************************************** */
/* cw trans routine                                           */
/* ********************************************************** */
void sio_send(char cmd, char len);
void sio_tx(char ch);
```

```
/*
void sio_tx1(char ch);
*/

void sio_txins(char ch);
void sio_recv(void);
/*
void check_sum(char ch);
void splite_checksum(void);
*/

interrupt[0x1B] void T1_int (void)
{


        MsecCut++;
        if(MsecCut >= 60){
                MsecCut = 0;
                if(TL0 > RealSpeed){
                        MOTOR_POWER = 0;
                } else if(TL0 < RealSpeed){
                        if(PowerBit) MOTOR_POWER = 1
                }
                TL0 = 0;
        }


        INTCNT++;
        INTCNT &= 3;

        if(!INTCNT) {
                TR0 = 0;
                SECBIT++;

                if (SECBIT) {
                        if(USRCNT) USRCNT--;
                        if(COINCNT) COINCNT--;
                        if(SwReadTime) SwReadTime--;
                        if(SecTime) SecTime--;
```

```
                    if(!SecTime) TimeBit = 1;
                    else         TimeBit = 0;
                }



            TR0 = 1;
        }
        TH0 = 0x48;         /* timer init.          */
}

void init_timer(void)
{
        TMOD  = 0x27;               /* timer0 : 3, timer1 : 2          */
        TCON  = 0;

        TH0   = 0x48;               /* timer 0 active 0.1 mSec                 */
                                                /*    OSC      22.1184    MHz
        */

        TH1   = 0xFA;       /* baud rate 9600 bps                  */
        SCON                        0x50;               /*          sio          mode1,
        */
        PCON = 0x00;                /* sio timer 1/2 divide                    */
        TI    = 0;
        RI    = 0;

        TR0   = 0;                  /* run timer0 disable                      */
        TR1   = 1;                  /* run timer1 enable                       */

        ET0   = 0;          /* timer0 interrupt disable        */
        ET1   = 1;                  /* timer1 interrupt enable                 */
        ES    = 0;                  /* sio    interrupt disable                */
        EA    = 1;
}

void init_para(void)
```

```
{
        INTCNT    = 0;
        SECCNT    = 0;
        USRCNT    = 0;
        INT0 = 0;
        INT1 = 0;

        SecTime = 100;
        TimeBit = 0;
        MOTOR_ON = 0;
        Led_Control = 2;
        LedMode = 1;
        SwReadTime = 200;
        RealSpeed  = 10;
        T1 = 1;
}

void Delay(WORD cnt)
{
        USRCNT = cnt;
        while(USRCNT);
}

void LcdString(BYTE *str)
{
  while(0 != *str){
         lcd_putch(*str);
         *str++;
  }
}

void Motor_control(void)
{
        bit ValBit, Val_spd;
        WORD Led_delay;
        Val_spd = 0;
        if(!LEFT_MOTOR | !RIGHT_MOTOR) {ValBit = 0, Mode_set = 0;

                if(KeyMode == 1){
```

```
if(!SwReadTime){
        SwReadTime = 200;

        if(!SW_DEC_MOTOR){Val_spd = 1;
                if(Led_Control > 1){
                        Led_Control--;
                        Led_delay = Led_Control * 5;}
        }

        if(!SW_INC_MOTOR){Val_spd = 1;
                if(Led_Control < 99){
                        Led_Control++;
                        Led_delay = Led_Control * 5;}
        }

        if(Val_spd){
                Val_spd = 0;
                lcd_clear();
                lcd_cursoroff();
                lcd_gotoxy(0, 0);
/*              LcdString("Test Borad");    */
                if(!LEFT_MOTOR)     LcdString("Led     Test
Left");
                else            LcdString("Led Test   Right");
                lcd_gotoxy(0, 1);
                LcdString("Speed Numver ");
                lcd_putch((Led_Control / 10) + 0x30);
                lcd_putch((Led_Control % 10) + 0x30);
        }
        if(TimeBit)  SecTime = Led_delay, LedMode++;

        if(LedMode > 4)  LedMode = 1;
        if(!LEFT_MOTOR) {
                switch(LedMode){
                        case 1:    LED_1 = 1, LED_2 = 1,
LED_3 = 1, LED_4 = 0;
                        break;
                        case 2:    LED_1 = 1, LED_2 = 1,
LED_3 = 0, LED_4 = 1;
```

```
                                        break;
                                        case 3:   LED_1 = 1, LED_2 = 0,
LED_3 = 1, LED_4 = 1;
                                        break;
                                        case 4:   LED_1 = 0, LED_2 = 1,
LED_3 = 1, LED_4 = 1;
                                        break;
                                }
                        }
                        else    {
                                switch(LedMode){
                                        case 1:   LED_1 = 0, LED_2 = 1
LED_3 = 1, LED_4 = 1;
                                        break;
                                        case 2:   LED_1 = 1, LED_2 = 0
LED_3 = 1, LED_4 = 1;
                                        break;
                                        case 3:   LED_1 = 1, LED_2 = 1
LED_3 = 0, LED_4 = 1;
                                        break;
                                        case 4:   LED_1 = 1, LED_2 = 1
LED_3 = 1, LED_4 = 0;
                                        break;
                                }
                        }
                }
        }

        if(KeyMode == 2){
                if(!SwReadTime){
                        SwReadTime = 200;

                        if(!LEFT_MOTOR) {FOW_MOTOR = 1, BAK_MOTOR =
0;}
                        else            {FOW_MOTOR = 0, BAK_MOTOR = 1;}

                        if(!ValBit)  {PowerBit = 1;}
                        else         {PowerBit= 0, MOTOR_POWER = 0;}
```

```
if(!SW_DEC_MOTOR){
        if(RealSpeed) RealSpeed--, Val_spd = 1;
}

if(!SW_INC_MOTOR){
        if(RealSpeed < 60) RealSpeed++, Val_spd = 1;
}

RPM_1000 = RealSpeed * 7;

if(Val_spd){
        Val_spd = 0;
        lcd_clear();
        lcd_cursoroff();
        lcd_gotoxy(0, 0);
/*      LcdString("Test Borad");    */
        if(!LEFT_MOTOR)   LcdString("Motor   Test
Left");
        else              LcdString("Motor Test Right");
        lcd_gotoxy(0, 1);
        LcdString("Speed ");
        lcd_putch((RealSpeed / 10) + 0x30);
        lcd_putch((RealSpeed % 10) + 0x30);

        LcdString(" RPM");
        Val_spd = 0;
        if(RPM_1000 / 1000)  {
                lcd_putch((RPM_1000  /  1000)  +
0x30);
                Val_spd = 1;
        }
        else      lcd_putch(' ');

        RPM_100 = RPM_1000 % 1000;
        if(RPM_100 / 100) {
                lcd_putch((RPM_100 / 100) + 0x30);
                Val_spd = 1;
        }
```

```
                                        else {
                                                if(!Val_spd) lcd_putch(' ');
                                                else         lcd_putch('0');
                                        }

                                        RPM_10 = RPM_1000 % 100;
                                        if(RPM_10 / 10)  {
                                                lcd_putch((RPM_10 / 10) + 0x30)
                                                Val_spd = 1;
                                        }
                                        else {
                                                if(!Val_spd) lcd_putch(' ');
                                                else      lcd_putch('0');
                                        }

                                        lcd_putch((RPM_10 % 10) + 0x30);

                                }
                        }
                }
        }
        else        {PowerBit= 0, MOTOR_POWER = 0, Mode_set = 1;
                                LED_1 = 1, LED_2 = 1, LED_3 = 1, LED_4 = 1;
}

/*
void Led_control(void)
{

}
*/

void main(void)
{
        bit ValBit;
        KeyMode = 0;

        init_main();
```

```
        lcd_init();
        lcd_set();
        Delay(500);

        lcd_clear();
        lcd_cursoroff();
        lcd_gotoxy(0, 0);
        LcdString(" SUNG KYUN KWAN ");
        lcd_gotoxy(0, 1);
        LcdString("-- UNIVERSITY --");

Loop:

        Motor_control();

        if(Mode_set) {
                if(!SwReadTime){
                        SwReadTime = 300;
                        ValBit = SW_DEC_MOTOR;
                        if(!ValBit) KeyMode++;
                }
                if(KeyMode > 4) KeyMode = 1;

                switch(KeyMode){

                        case 1:
                                lcd_clear();
                                lcd_cursoroff();
                                lcd_gotoxy(0, 0);
                                LcdString("Led    Control");
                                lcd_gotoxy(0, 1);
                                LcdString("Press Quit key")
                                while(SW_DEC_MOTOR);

                        break;

                        case 2:
                                lcd_clear();
                                lcd_cursoroff();
```

```
                    lcd_gotoxy(0, 0);
                    LcdString("Motor Control");
                    lcd_gotoxy(0, 1);
                    LcdString("Press Quit key");
                    MOTOR_ON = 1;
                    while(SW_DEC_MOTOR);

                break;

                case 3:
                    lcd_clear();
                    lcd_cursoroff();
                    lcd_gotoxy(0, 0);
                    LcdString("Sreial Transmit")
                    lcd_gotoxy(0, 1);
                    LcdString("Press Quit key");
                    while(SW_DEC_MOTOR);

                    break;


                    case 4:
                    lcd_clear();
                    lcd_cursoroff();
                    lcd_gotoxy(0, 0);
                    LcdString("Sreial Reccive");
                    lcd_gotoxy(0, 1);
                    LcdString("Press Quit key");
                    while(SW_DEC_MOTOR);

                    break;

            }
        }

goto Loop;

}
```

```
void init_main(void)
{

        init_para();
        init_timer();

        TR0 = 1;

}



/* ********************************************************** */
/* eeprom 93c46 routine                                       */
/* ********************************************************** */
void C46_addr(char mode)
{
        char i;

        /* cprom_mode(); */
        PROM_CS  = 1;
        PROM_DI  = 0;
        C46_pulse();

        PROM_DI = 1;                                /* start bit */
        C46_pulse();

        /* cprom_addr(); */
        for(i = 0; i < 8; i++) {
                if (mode & 0x80) PROM_DI = 1;
                else             PROM_DI = 0;
                C46_pulse();
                mode <<= 1;
        }
}

void eprom_getch(char addr)
{
```

```
        char i;

        /* ******************* */
        /* serial eeprom  READ */
        /* ******************* */
        C46_addr(addr | 0x80);
        /* eprom_rdata(); */
        PROM_DO  = 1;

        PROM_MSB  = 0;
        PROM_LSB  = 0;
        for (i = 0; i < 8; i++) {
                PROM_MSB <<= 1;
                C46_pulse();
                if(PROM_DO)  PROM_MSB |= 1;
        }

        for (i = 0; i < 8; i++) {
                PROM_LSB <<= 1;
                C46_pulse();
                if(PROM_DO)  PROM_LSB |= 1;
        }

        PROM_CS = 0;
}

/* ******************* */
/* serial eeprom Write */
/* ******************* */
void eprom_putch(char addr)
{
        char i;

        C46_addr(0x30);
        PROM_CS = 0;

        C46_addr(addr | 0x40);
        /* eprom_wdata(); */
        for (i= 0; i < 8; i++) {
```

```
                if (PROM_MSB & 0x80) PROM_DI = 1
                else                 PROM_DI = 0;
                C46_pulse();
                PROM_MSB <<= 1;
        }

        for (i= 0; i < 8; i++) {
                if (PROM_LSB & 0x80) PROM_DI = 1;
                else                 PROM_DI = 0;
                C46_pulse();
                PROM_LSB <<= 1;
        }

        /* eprom_busy();        */
        PROM_CS  = 0;
        PROM_DI = 1;
        C46_pulse();

        PROM_CS = 1;
        USRCNT  = 30;
        while(USRCNT) {
                C46_pulse();
                if (PROM_DI) break;
        }
        PROM_CS = 0;

        C46_addr(0);
        PROM_CS = 0;
}

void C46_pulse(void)
{
        PROM_CK = 1;
        PROM_CK = 0;
}


void sio_tx(char ch)
{
```

```
        SBUF = ch;
        while(!TI);
        TI = 0;
}

void sio_txins(char ch)
{
        if((ch == 0xAB)||(ch == 0x18)) {
                SBUF = 0x18;
                while(!TI);
                TI = 0;
                ch ^= 0x40;
        }
        SBUF = ch;
        while(!TI);
        TI = 0;
}

void sio_recv(void)
{
        unsigned char ch;

        if(RI){
                ch  = SBUF;
                RI  = 0;
                if(SIO_POS < 7) SIO_BUF[SIO_POS++] = ch
                if((ch == 'h') && (ch == 'H')) RcvFull = 1;
        }
}



// MainFrm.cpp : implementation of the CMainFrame class
//

#include "stdafx.h"
#include "Serial.h"

#include "MainFrm.h"
```

```
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CMainFrame

IMPLEMENT_DYNCREATE(CMainFrame, CFrameWnd)

BEGIN_MESSAGE_MAP(CMainFrame, CFrameWnd)
        //{{AFX_MSG_MAP(CMainFrame)
        ON_WM_CREATE()
        ON_COMMAND(ID_SERIAL_HELP, OnSerialHelp)
        //}}AFX_MSG_MAP
END_MESSAGE_MAP()

static UINT indicators[] =
{
        ID_SEPARATOR,           // status line indicator
        ID_INDICATOR_CAPS,
        ID_INDICATOR_NUM,
        ID_INDICATOR_SCRL,
};

/////////////////////////////////////////////////////////////////////////////
// CMainFrame construction/destruction

CMainFrame::CMainFrame()
{
        // TODO: add member initialization code here

}

CMainFrame::~CMainFrame()
{
}
```

```
int CMainFrame::OnCreate(LPCREATESTRUCT lpCreateStruct)
{
	if (CFrameWnd::OnCreate(lpCreateStruct) == -1)
		return -1;

	/*if (!m_wndToolBar.CreateEx(this, TBSTYLE_FLAT, WS_CHILD | WS_VISIBLE | CBRS_TOP
		| CBRS_GRIPPER | CBRS_TOOLTIPS | CBRS_FLYBY | CBRS_SIZE_DYNAMIC) ||
		!m_wndToolBar.LoadToolBar(IDR_MAINFRAME))
	{
		TRACE0("Failed to create toolbar\n");
		return -1;		// fail to create
	}*/

	if (!m_wndStatusBar.Create(this) ||
		!m_wndStatusBar.SetIndicators(indicators,
		  sizeof(indicators)/sizeof(UINT)))
	{
		TRACE0("Failed to create status bar\n");
		return -1;		// fail to create
	}

	// TODO: Delete these three lines if you don't want the toolbar to
	//  be dockable
	/*m_wndToolBar.EnableDocking(CBRS_ALIGN_ANY);
	EnableDocking(CBRS_ALIGN_ANY);
	DockControlBar(&m_wndToolBar);*/

	return 0;
}

BOOL CMainFrame::PreCreateWindow(CREATESTRUCT& cs)
{
	if( !CFrameWnd::PreCreateWindow(cs) )
		return FALSE;
	// TODO: Modify the Window class or styles here by modifying
	//  the CREATESTRUCT cs
	cs.style=WS_SYSMENU;
	//cs.style=WS_THICKFRAME;
```

```
        return TRUE;
}

/////////////////////////////////////////////////////////////////////////////
// CMainFrame diagnostics

#ifdef _DEBUG
void CMainFrame::AssertValid() const
{
        CFrameWnd::AssertValid();
}

void CMainFrame::Dump(CDumpContext& dc) const
{
        CFrameWnd::Dump(dc);
}

#endif //_DEBUG

/////////////////////////////////////////////////////////////////////////////
// CMainFrame message handlers


void CMainFrame::OnSerialHelp()
{
        // TODO: Add your command handler code here
        AfxGetApp()->WinHelp(HID_START);
}



void CMainFrame::ActivateFrame(int nCmdShow)
{
        // TODO: Add your specialized code here and/or call the base class

        WINDOWPLACEMENT wp;
        wp.rcNormalPosition=CRect(100,10,805,570);
        SetWindowPlacement(&wp);
```

```
        CFrameWnd::ActivateFrame(nCmdShow);
}


// PortSet.cpp : implementation file
//

#include "stdafx.h"
#include "Serial.h"
#include "PortSet.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CPortSet dialog


CPortSet::CPortSet(CWnd* pParent /*=NULL*/)
        : CDialog(CPortSet::IDD, pParent)
{
        //{{AFX_DATA_INIT(CPortSet)
                // NOTE: the ClassWizard will add member initialization here
        //}}AFX_DATA_INIT
}


void CPortSet::DoDataExchange(CDataExchange* pDX)
{
        CDialog::DoDataExchange(pDX);
        //{{AFX_DATA_MAP(CPortSet)
                // NOTE: the ClassWizard will add DDX and DDV calls here
        //}}AFX_DATA_MAP
}
```

```
BEGIN_MESSAGE_MAP(CPortSet, CDialog)
        //{{AFX_MSG_MAP(CPortSet)
        //}}AFX_MSG_MAP
END_MESSAGE_MAP()



// PortSetDlg.cpp : implementation file
//

#include "stdafx.h"
#include "Serial.h"
#include "PortSetDlg.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CPortSetDlg dialog


CPortSetDlg::CPortSetDlg(CWnd* pParent /*=NULL*/)
        : CDialog(CPortSetDlg::IDD, pParent)
{
        //{{AFX_DATA_INIT(CPortSetDlg)
                // NOTE: the ClassWizard will add member initialization here
        //}}AFX_DATA_INIT
        m_nPort=1;
        m_nBaudRate=9600;
}


void CPortSetDlg::DoDataExchange(CDataExchange* pDX)
{
        CDialog::DoDataExchange(pDX);
        //{{AFX_DATA_MAP(CPortSetDlg)
```

```
                    // NOTE: the ClassWizard will add DDX and DDV calls here
        //}}AFX_DATA_MAP
}


BEGIN_MESSAGE_MAP(CPortSetDlg, CDialog)
        //{{AFX_MSG_MAP(CPortSetDlg)
        ON_BN_CLICKED(IDC_BAUD1, OnBaud1)
        ON_BN_CLICKED(IDC_BAUD2, OnBaud2)
        ON_BN_CLICKED(IDC_BAUD3, OnBaud3)
        ON_BN_CLICKED(IDC_COM1, OnCom1)
        ON_BN_CLICKED(IDC_COM2, OnCom2)
        ON_BN_CLICKED(IDC_COM3, OnCom3)
        ON_BN_CLICKED(IDC_COM4, OnCom4)
        //}}AFX_MSG_MAP
END_MESSAGE_MAP()

////////////////////////////////////////////////////////////////////////////
// CPortSetDlg message handlers

void CPortSetDlg::OnBaud1()
{
        // TODO: Add your control notification handler code here
        m_nBaudRate=9600;
}

void CPortSetDlg::OnBaud2()
{
        // TODO: Add your control notification handler code here
        m_nBaudRate=19200;
}

void CPortSetDlg::OnBaud3()
{
        // TODO: Add your control notification handler code here
        m_nBaudRate=38400;
}

void CPortSetDlg::OnCom1()
```

```
{
        // TODO: Add your control notification handler code here
        m_nPort=1;
}

void CPortSetDlg::OnCom2()
{
        // TODO: Add your control notification handler code here
        m_nPort=2;
}

void CPortSetDlg::OnCom3()
{
        // TODO: Add your control notification handler code here
        m_nPort=3;
}

void CPortSetDlg::OnCom4()
{
        // TODO: Add your control notification handler code here
        m_nPort=4;
}

UINT CPortSetDlg::GetPort()
{
        return m_nPort;
}

UINT CPortSetDlg::GetBaudRate()
{
        return m_nBaudRate;
}



// ProDlg.cpp : implementation file
//

#include "stdafx.h"
```

```
#include "Serial.h"
#include "ProDlg.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CProDlg dialog


CProDlg::CProDlg(CWnd* pParent /*=NULL*/)
        : CDialog(CProDlg::IDD, pParent)
{
        //{{AFX_DATA_INIT(CProDlg)
                // NOTE: the ClassWizard will add member initialization here
        //}}AFX_DATA_INIT
}


void CProDlg::DoDataExchange(CDataExchange* pDX)
{
        CDialog::DoDataExchange(pDX);
        //{{AFX_DATA_MAP(CProDlg)
                // NOTE: the ClassWizard will add DDX and DDV calls here
        //}}AFX_DATA_MAP
}


BEGIN_MESSAGE_MAP(CProDlg, CDialog)
        //{{AFX_MSG_MAP(CProDlg)
                // NOTE: the ClassWizard will add message map macros here
        //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CProDlg message handlers
```

```
// Serial.cpp : Defines the class behaviors for the application.
//

#include "stdafx.h"
#include "Serial.h"

#include "MainFrm.h"
#include "SerialDoc.h"
#include "SerialView.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CSerialApp

BEGIN_MESSAGE_MAP(CSerialApp, CWinApp)
        //{{AFX_MSG_MAP(CSerialApp)
        ON_COMMAND(ID_APP_ABOUT, OnAppAbout)
                // NOTE - the ClassWizard will add and remove mapping macros here.
                //    DO NOT EDIT what you see in these blocks of generated code!
        //}}AFX_MSG_MAP
        // Standard file based document commands
        ON_COMMAND(ID_FILE_NEW, CWinApp::OnFileNew)
        ON_COMMAND(ID_FILE_OPEN, CWinApp::OnFileOpen)
        // Standard print setup command
        ON_COMMAND(ID_FILE_PRINT_SETUP, CWinApp::OnFilePrintSetup)
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CSerialApp construction

CSerialApp::CSerialApp()
{
```

```
	// TODO: add construction code here,
	// Place all significant initialization in InitInstance
}

/////////////////////////////////////////////////////////////////////////////
// The one and only CSerialApp object

CSerialApp theApp;

/////////////////////////////////////////////////////////////////////////////
// CSerialApp initialization

BOOL CSerialApp::InitInstance()
{
	AfxEnableControlContainer();

	// Standard initialization
	// If you are not using these features and wish to reduce the size
	//  of your final executable, you should remove from the following
	//  the specific initialization routines you do not need.

#ifdef _AFXDLL
	Enable3dControls();			// Call this when using MFC in a shared DLL
#else
	Enable3dControlsStatic();	// Call this when linking to MFC statically
#endif

	// Change the registry key under which our settings are stored.
	// TODO: You should modify this string to be something appropriate
	// such as the name of your company or organization.
	SetRegistryKey(_T("Local AppWizard-Generated Applications"));

	LoadStdProfileSettings();  // Load standard INI file options (including MRU)

	// Register the application's document templates.  Document templates
	//  serve as the connection between documents, frame windows and views.

	CSingleDocTemplate* pDocTemplate;
	pDocTemplate = new CSingleDocTemplate(
```

```
                IDR_MAINFRAME,
                RUNTIME_CLASS(CSerialDoc),
                RUNTIME_CLASS(CMainFrame),       // main SDI frame window
                RUNTIME_CLASS(CSerialView));
        AddDocTemplate(pDocTemplate);

        // Parse command line for standard shell commands, DDE, file open
        CCommandLineInfo cmdInfo;
        ParseCommandLine(cmdInfo);

        // Dispatch commands specified on the command line
        if (!ProcessShellCommand(cmdInfo))
                return FALSE;

        // The one and only window has been initialized, so show and update it.
        m_pMainWnd->ShowWindow(SW_SHOW);
        m_pMainWnd->UpdateWindow();

        return TRUE;
}


/////////////////////////////////////////////////////////////////////////////
// CAboutDlg dialog used for App About

class CAboutDlg : public CDialog
{
public:
        CAboutDlg();

// Dialog Data
        //{{AFX_DATA(CAboutDlg)
        enum { IDD = IDD_ABOUTBOX };
        //}}AFX_DATA

        // ClassWizard generated virtual function overrides
        //{{AFX_VIRTUAL(CAboutDlg)
        protected:
        virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
```

```
        //}}AFX_VIRTUAL

// Implementation
protected:
        //{{AFX_MSG(CAboutDlg)
                // No message handlers
        //}}AFX_MSG
        DECLARE_MESSAGE_MAP()
};

CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
{
        //{{AFX_DATA_INIT(CAboutDlg)
        //}}AFX_DATA_INIT
}

void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
        CDialog::DoDataExchange(pDX);
        //{{AFX_DATA_MAP(CAboutDlg)
        //}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
        //{{AFX_MSG_MAP(CAboutDlg)
                // No message handlers
        //}}AFX_MSG_MAP
END_MESSAGE_MAP()

// App command to run the dialog
void CSerialApp::OnAppAbout()
{
        CAboutDlg aboutDlg;
        aboutDlg.DoModal();
}

/////////////////////////////////////////////////////////////////////////////
// CSerialApp message handlers
```

```
// SerialDoc.cpp : implementation of the CSerialDoc class
//

#include "stdafx.h"
#include "Serial.h"

#include "SerialDoc.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CSerialDoc

IMPLEMENT_DYNCREATE(CSerialDoc, CDocument)

BEGIN_MESSAGE_MAP(CSerialDoc, CDocument)
        //{{AFX_MSG_MAP(CSerialDoc)
                // NOTE - the ClassWizard will add and remove mapping macros here.
                //    DO NOT EDIT what you see in these blocks of generated code!
        //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CSerialDoc construction/destruction

CSerialDoc::CSerialDoc()
{
        // TODO: add one-time construction code here
        for(int a=0;a<10;a++)
                m_MotorData[a]=_T("");
}

CSerialDoc::~CSerialDoc()
{
```

```
}

BOOL CSerialDoc::OnNewDocument()
{
        if (!CDocument::OnNewDocument())
                return FALSE;

        // TODO: add reinitialization code here
        // (SDI documents will reuse this document)

        return TRUE;
}



/////////////////////////////////////////////////////////////////////////////
// CSerialDoc serialization

void CSerialDoc::Serialize(CArchive& ar)
{
        if (ar.IsStoring())
        {
                // TODO: add storing code here
        }
        else
        {
                // TODO: add loading code here
        }
}

/////////////////////////////////////////////////////////////////////////////
// CSerialDoc diagnostics

#ifdef _DEBUG
void CSerialDoc::AssertValid() const
{
        CDocument::AssertValid();
}
```

```
void CSerialDoc::Dump(CDumpContext& dc) const
{
        CDocument::Dump(dc);
}
#endif //_DEBUG

/////////////////////////////////////////////////////////////////////////////
// CSerialDoc commands



// SerialView.cpp : implementation of the CSerialView class
//

#include "stdafx.h"
#include "Serial.h"

#include "SerialDoc.h"
#include "SerialView.h"

#include "MainFrm.h"
#include "PortSetDlg.h"
#include "ProDlg.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__
#endif

/////////////////////////////////////////////////////////////////////////////
// CSerialView

IMPLEMENT_DYNCREATE(CSerialView, CFormView)

BEGIN_MESSAGE_MAP(CSerialView, CFormView)
        //{{AFX_MSG_MAP(CSerialView)
        ON_BN_CLICKED(IDC_PORTCONN, OnPortconn)
        ON_BN_CLICKED(IDC_PORTINIT, OnPortinit)
```

```
    ON_BN_CLICKED(IDC_PORTRELE, OnPortrele)
    ON_BN_CLICKED(IDC_R_M0, OnRM0)
    ON_BN_CLICKED(IDC_R_M1, OnRM1)
    ON_BN_CLICKED(IDC_R_M2, OnRM2)
    ON_BN_CLICKED(IDC_R_M3, OnRM3)
    ON_BN_CLICKED(IDC_R_M4, OnRM4)
    ON_BN_CLICKED(IDC_R_M5, OnRM5)
    ON_BN_CLICKED(IDC_R_M6, OnRM6)
    ON_BN_CLICKED(IDC_R_M7, OnRM7)
    ON_BN_CLICKED(IDC_R_M8, OnRM8)
    ON_BN_CLICKED(IDC_R_M9, OnRM9)
    ON_BN_CLICKED(IDC_START, OnStart)
    ON_BN_CLICKED(IDC_STOP, OnStop)
    ON_BN_CLICKED(IDC_CLEARRECIVE, OnClearrecive)
    ON_BN_CLICKED(IDC_M0_CHK, OnM0Chk)
    ON_BN_CLICKED(IDC_M1_CHK, OnM1Chk)
    ON_BN_CLICKED(IDC_M2_CHK, OnM2Chk)
    ON_BN_CLICKED(IDC_M3_CHK, OnM3Chk)
    ON_BN_CLICKED(IDC_M4_CHK, OnM4Chk)
    ON_BN_CLICKED(IDC_M5_CHK, OnM5Chk)
    ON_BN_CLICKED(IDC_M6_CHK, OnM6Chk)
    ON_BN_CLICKED(IDC_M7_CHK, OnM7Chk)
    ON_BN_CLICKED(IDC_M8_CHK, OnM8Chk)
    ON_BN_CLICKED(IDC_M9_CHK, OnM9Chk)
    ON_BN_CLICKED(IDC_RECIDATA, OnRecidata)
    //}}AFX_MSG_MAP
    // Standard printing commands
    ON_COMMAND(ID_FILE_PRINT, CFormView::OnFilePrint)
    ON_COMMAND(ID_FILE_PRINT_DIRECT, CFormView::OnFilePrint)
    ON_COMMAND(ID_FILE_PRINT_PREVIEW, CFormView::OnFilePrintPreview)
    ON_MESSAGE(WM_COMM_READ,OnCommunication)
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CSerialView construction/destruction

HWND hCommWnd;

CString strTemp;
```

```
CSerialView::CSerialView()
	: CFormView(CSerialView::IDD)
{
	//{{AFX_DATA_INIT(CSerialView)
	m_m1chk = FALSE;
	m_m2chk = FALSE;
	m_m3chk = FALSE;
	m_m4chk = FALSE;
	m_m5chk = FALSE;
	m_m6chk = FALSE;
	m_m7chk = FALSE;
	m_m8chk = FALSE;
	m_m9chk = FALSE;
	m_m9time = _T("");
	m_m8time = _T("");
	m_m7time = _T("");
	m_m6time = _T("");
	m_m5time = _T("");
	m_m4time = _T("");
	m_m3time = _T("");
	m_m2time = _T("");
	m_m1time = _T("");
	m_m0time = _T("");
	m_m0speed = _T("");
	m_m1speed = _T("");
	m_m2speed = _T("");
	m_m3speed = _T("");
	m_m4speed = _T("");
	m_m5speed = _T("");
	m_m6speed = _T("");
	m_m7speed = _T("");
	m_m9speed = _T("");
	m_m3rot = FALSE;
	m_m2rot = FALSE;
	m_m1rot = FALSE;
	m_m0rot = FALSE;
	m_m4rot = FALSE;
	m_m5rot = FALSE;
	m_m6rot = FALSE;
```

```
		m_m7rot = FALSE;
		m_m8speed = _T("");
		m_m8rot = FALSE;
		m_m9rot = FALSE;
		m_m0chk = FALSE;
		//}}AFX_DATA_INIT
		// TODO: add construction code here

		m_nPort=0;
		m_nBaudRate=0;
		bPortInit=FALSE;

		m_DataChar=_T("");
		bDial=FALSE;
		m_SendData=_T("");
		for(int a=0;a<10;a++)
		{
			m_MotorData[a]=_T("");

		}
		m_command=0;
}

CSerialView::~CSerialView()
{

}

void CSerialView::DoDataExchange(CDataExchange* pDX)
{
		CFormView::DoDataExchange(pDX);
		//{{AFX_DATA_MAP(CSerialView)
		DDX_Control(pDX, IDC_RECIVE, m_recive);
		DDX_Control(pDX, IDC_DATALIST9, m_datalist9)
		DDX_Control(pDX, IDC_DATALIST8, m_datalist8)
		DDX_Control(pDX, IDC_DATALIST7, m_datalist7)
		DDX_Control(pDX, IDC_DATALIST6, m_datalist6)
		DDX_Control(pDX, IDC_DATALIST5, m_datalist5)
		DDX_Control(pDX, IDC_DATALIST4, m_datalist4)
```

```
DDX_Control(pDX, IDC_DATALIST3, m_datalist3)
DDX_Control(pDX, IDC_DATALIST2, m_datalist2)
DDX_Control(pDX, IDC_DATALIST1, m_datalist1)
DDX_Control(pDX, IDC_DATALIST0, m_datalist0)
DDX_Check(pDX, IDC_M1_CHK, m_m1chk);
DDX_Check(pDX, IDC_M2_CHK, m_m2chk);
DDX_Check(pDX, IDC_M3_CHK, m_m3chk);
DDX_Check(pDX, IDC_M4_CHK, m_m4chk);
DDX_Check(pDX, IDC_M5_CHK, m_m5chk);
DDX_Check(pDX, IDC_M6_CHK, m_m6chk);
DDX_Check(pDX, IDC_M7_CHK, m_m7chk);
DDX_Check(pDX, IDC_M8_CHK, m_m8chk);
DDX_Check(pDX, IDC_M9_CHK, m_m9chk);
DDX_Text(pDX, IDC_M9_TIME, m_m9time);
DDV_MaxChars(pDX, m_m9time, 4);
DDX_Text(pDX, IDC_M8_TIME, m_m8time);
DDV_MaxChars(pDX, m_m8time, 4);
DDX_Text(pDX, IDC_M7_TIME, m_m7time);
DDV_MaxChars(pDX, m_m7time, 4);
DDX_Text(pDX, IDC_M6_TIME, m_m6time);
DDV_MaxChars(pDX, m_m6time, 4);
DDX_Text(pDX, IDC_M5_TIME, m_m5time);
DDV_MaxChars(pDX, m_m5time, 4);
DDX_Text(pDX, IDC_M4_TIME, m_m4time);
DDV_MaxChars(pDX, m_m4time, 4);
DDX_Text(pDX, IDC_M3_TIME, m_m3time);
DDV_MaxChars(pDX, m_m3time, 4);
DDX_Text(pDX, IDC_M2_TIME, m_m2time);
DDV_MaxChars(pDX, m_m2time, 4);
DDX_Text(pDX, IDC_M1_TIME, m_m1time);
DDV_MaxChars(pDX, m_m1time, 4);
DDX_Text(pDX, IDC_M0_TIME, m_m0time);
DDV_MaxChars(pDX, m_m0time, 4);
DDX_Text(pDX, IDC_M0_SPEED, m_m0speed);
DDV_MaxChars(pDX, m_m0speed, 3);
DDX_Text(pDX, IDC_M1_SPEED, m_m1speed);
DDV_MaxChars(pDX, m_m1speed, 3);
DDX_Text(pDX, IDC_M2_SPEED, m_m2speed);
DDV_MaxChars(pDX, m_m2speed, 3);
```

```
        DDX_Text(pDX, IDC_M3_SPEED, m_m3speed);
        DDV_MaxChars(pDX, m_m3speed, 3);
        DDX_Text(pDX, IDC_M4_SPEED, m_m4speed);
        DDV_MaxChars(pDX, m_m4speed, 3);
        DDX_Text(pDX, IDC_M5_SPEED, m_m5speed);
        DDV_MaxChars(pDX, m_m5speed, 3);
        DDX_Text(pDX, IDC_M6_SPEED, m_m6speed);
        DDV_MaxChars(pDX, m_m6speed, 3);
        DDX_Text(pDX, IDC_M7_SPEED, m_m7speed);
        DDV_MaxChars(pDX, m_m7speed, 3);
        DDX_Text(pDX, IDC_M9_SPEED, m_m9speed);
        DDV_MaxChars(pDX, m_m9speed, 3);
        DDX_Check(pDX, IDC_M3_ROT, m_m3rot);
        DDX_Check(pDX, IDC_M2_ROT, m_m2rot);
        DDX_Check(pDX, IDC_M1_ROT, m_m1rot);
        DDX_Check(pDX, IDC_M0_ROT, m_m0rot);
        DDX_Check(pDX, IDC_M4_ROT, m_m4rot);
        DDX_Check(pDX, IDC_M5_ROT, m_m5rot);
        DDX_Check(pDX, IDC_M6_ROT, m_m6rot);
        DDX_Check(pDX, IDC_M7_ROT, m_m7rot);
        DDX_Text(pDX, IDC_M8_SPEED, m_m8speed);
        DDX_Check(pDX, IDC_M8_ROT, m_m8rot);
        DDX_Check(pDX, IDC_M9_ROT, m_m9rot);
        DDX_Check(pDX, IDC_M0_CHK, m_m0chk);
        //}}AFX_DATA_MAP
}

BOOL CSerialView::PreCreateWindow(CREATESTRUCT& cs)
{
        // TODO: Modify the Window class or styles here by modifying
        //  the CREATESTRUCT cs

        return CFormView::PreCreateWindow(cs);
}

void CSerialView::OnInitialUpdate()
{
        CFormView::OnInitialUpdate();
        GetParentFrame()->RecalcLayout();
```

```
        ResizeParentToFit();

}

/////////////////////////////////////////////////////////////////////////////
// CSerialView printing

BOOL CSerialView::OnPreparePrinting(CPrintInfo* pInfo)
{
        // default preparation
        return DoPreparePrinting(pInfo);
}

void CSerialView::OnBeginPrinting(CDC* /*pDC*/, CPrintInfo* /*pInfo*/)
{
        // TODO: add extra initialization before printing
}

void CSerialView::OnEndPrinting(CDC* /*pDC*/, CPrintInfo* /*pInfo*/)
{
        // TODO: add cleanup after printing
}

void CSerialView::OnPrint(CDC* pDC, CPrintInfo* /*pInfo*/)
{
        // TODO: add customized printing code here
}

/////////////////////////////////////////////////////////////////////////////
// CSerialView diagnostics

#ifdef _DEBUG
void CSerialView::AssertValid() const
{
        CFormView::AssertValid();
}

void CSerialView::Dump(CDumpContext& dc) const
{
```

```
	CFormView::Dump(dc);
}

CSerialDoc* CSerialView::GetDocument() // non-debug version is inline
{
	ASSERT(m_pDocument->IsKindOf(RUNTIME_CLASS(CSerialDoc)));
	return (CSerialDoc*)m_pDocument;
}
#endif //_DEBUG

/////////////////////////////////////////////////////////////////////////////
// CSerialView message handlers

void CSerialView::OnPortconn()
{
	// TODO: Add your control notification handler code here
	if(m_nPort==0)
	{
		MessageBox("Port를 선택하시오!","Port초기화",MB_ICONSTOP | MB_OK)
		return;
	}


	CString Temp;
	Temp.Format("COM%d",m_nPort);
	if(!m_ComuPort.OpenPort(Temp,m_nBaudRate,m_nPort))
	{
		MessageBox("Port 초기화 실패","Port 초기화",MB_ICONSTOP | MB_OK);
		return;
	}
	else
	{
		Temp.Format("Port COM%d 초기화 성공",m_nPort);
		ProcessMessage(Temp);
		bPortInit=TRUE;
	}
}

void CSerialView::OnPortinit()
```

```
{
        // TODO: Add your control notification handler code here
        CPortSetDlg dlg;
        if(dlg.DoModal()!=IDOK)
                return;
        m_nPort=dlg.GetPort();
        m_nBaudRate=dlg.GetBaudRate();
}

void CSerialView::OnPortrele()
{
        // TODO: Add your control notification handler code here
        m_nPort=0;
        m_nBaudRate=0;
        CString Temp;
        Temp.Format("Port COM%d 연결을 해제합니다!",m_nPort);
        ProcessMessage(Temp);
        m_ComuPort.ClosePort();
        bPortInit=FALSE;
}

void CSerialView::ProcessMessage(CString msg)
{
        ((CMainFrame *)AfxGetMainWnd())->m_wndStatusBar.SetWindowText(msg)
}

void CSerialView::OnDraw(CDC* pDC)
{
        // TODO: Add your specialized code here and/or call the base class
        CSerialDoc *pDoc=GetDocument();
        ASSERT_VALID(pDoc);
        hCommWnd=m_hWnd;
        if(bDial==FALSE)
        {
                CProDlg dlg;
                if(dlg.DoModal()==IDOK)
                        Invalidate(FALSE);
        }
        bDial=TRUE;
```

```
}

void CSerialView::OnStart()
{
	// TODO: Add your control notification handler code here
	CSerialDoc *pDoc=GetDocument();
	UpdateData();
	m_SendData=_T("");
	Message(m_nPort,bPortInit);

	m_command=1;
	CString Temp;
	Temp.Format("Port COM%d로 Data를 전송합니다!",m_nPort);
	ProcessMessage(Temp);

	//#0 Motor 회전Data//
	if(m_m0chk==TRUE)
		m_MotorData[0]=ConvertData("0",m_command,m_m0speed,m_m0time,m_m0rot);

	else
		m_MotorData[0]=_T("");


	//#1 Motor 회전Data//
	if(m_m1chk==TRUE)
		m_MotorData[1]=ConvertData("1",m_command,m_m1speed,m_m1time,m_m1rot)
	else
		m_MotorData[1]=_T("");

	//#2 Motor 회전Data//
	if(m_m2chk==TRUE)
		m_MotorData[2]=ConvertData("2",m_command,m_m2speed,m_m2time,m_m2rot)
	else
		m_MotorData[2]=_T("");

	//#3 Motor 회전Data//
	if(m_m3chk==TRUE)
		m_MotorData[3]=ConvertData("3",m_command,m_m3speed,m_m3time,m_m3rot)
	else
```

```
            m_MotorData[3]=_T("");

//#4 Motor 회전Data//
if(m_m4chk==TRUE)
            m_MotorData[4]=ConvertData("4",m_command,m_m4speed,m_m4time,m_m4rot)
else
            m_MotorData[4]=_T("");

//#5 Motor 회전Data//
if(m_m5chk==TRUE)
            m_MotorData[5]=ConvertData("5",m_command,m_m5speed,m_m5time,m_m5rot)
else
            m_MotorData[5]=_T("");

//#6 Motor 회전Data//
if(m_m6chk==TRUE)
            m_MotorData[6]=ConvertData("6",m_command,m_m6speed,m_m6time,m_m6rot)
else
            m_MotorData[6]=_T("");

//#7 Motor 회전Data//
if(m_m7chk==TRUE)
            m_MotorData[7]=ConvertData("7",m_command,m_m7speed,m_m7time,m_m7rot)
else
            m_MotorData[7]=_T("");

//#8 Motor 회전Data//
if(m_m8chk==TRUE)
            m_MotorData[8]=ConvertData("8",m_command,m_m8speed,m_m8time,m_m8rot)
else
            m_MotorData[8]=_T("");

//#9 Motor 회전Data//
if(m_m9chk==TRUE)
            m_MotorData[9]=ConvertData("9",m_command,m_m9speed,m_m9time,m_m9rot)
else
            m_MotorData[9]=_T("");

for(int a=0;a<10;a++)
```

```
                m_SendData+=m_MotorData[a];

        m_ComuPort.WriteComm((unsigned char *)(LPCTSTR)m_SendData,m_SendData.GetLength())

}

/////////자료 수신시 처리 함수////////////

LONG CSerialVicw::OnCommunication(UINT Port,LONG lParam)
{

        //strTemp=_T("");
        BYTE aByte;

        int size=(m_ComuPort.m_QueueRead).GetSize();

        for(int i=0;i<size;i++)
        {
                (m_ComuPort.m_QueueRead).GetByte(&aByte);
                if(aByte!=NULL)
                {
                        m_DataChar+=(CString)aByte;
                        strTemp+=(CString)aByte;
                }

                else
                {
                        i--;
                        size--;
                }

        }

        if(strTemp.GetLength()>8*12)
        {
                m_recive.AddString(strTemp.Mid(0,8*12));
                strTemp.Delete(0,8*12);
        }
```

```
m_recive.AddString(strTemp);

CString imsi;

if(m_DataChar.GetLength()>=12)
{

        int count=m_DataChar.GetLength()/12;
        for(int a=0;a<count;a++)
        {

                while(m_DataChar[0]!=0x1E)
                {
                        m_DataChar.Delete(0,1);
                        if(m_DataChar.GetLength()<12)
                                return 0;
                }

                imsi=m_DataChar.Left(12);
                if(imsi.Left(1)==0x1E && imsi.Right(1)==0x1F)
                {
                        StrogeData(imsi);
                        m_DataChar.Delete(0,12);
                }

                switch(atoi(imsi.Mid(1,1)))
                {
                case 0:
                        m_datalist0.ResetContent();
                        ListDisplay("0");
                        break;
                case 1:
                        m_datalist1.ResetContent();
                        ListDisplay("1");
                        break;
                case 2:
                        m_datalist2.ResetContent();
                        ListDisplay("2");
                        break;
```

```
                    case 3:
                        m_datalist3.ResetContent();
                        ListDisplay("3");
                        break;
                    case 4:
                        m_datalist4.ResetContent();
                        ListDisplay("4");
                        break;
                    case 5:
                        m_datalist5.ResetContent();
                        ListDisplay("5");
                        break;
                    case 6:
                        m_datalist6.ResetContent();
                        ListDisplay("6");
                        break;
                    case 7:
                        m_datalist7.ResetContent();
                        ListDisplay("7");
                        break;
                    case 8:
                        m_datalist8.ResetContent();
                        ListDisplay("8");
                        break;
                    case 9:
                        m_datalist9.ResetContent();
                        ListDisplay("9");
                        break;
                    }
                }
            }

            return 0;
}
```

///////송신 자료를 수신할 수 있는 Data Type으로 변환////////

CString CSerialView::ConvertData(CString motorno, int command=1, CString mspeed=_T(""), CString mtime=_T(""), BOOL mrot=FALSE)

```
{
        CString str,SendData;
        SendData=_T("");
        SendData+=0x1E;

        switch(command)
        {
        case 1:
                {
                        SendData+=motorno;
                        SendData+=_T("A");
                        switch(mspeed.GetLength())
                        {
                        case 1:
                                str=_T("00");
                                break;
                        case 2:
                                str=_T("0");
                                break;
                        case 3:
                                str=_T("");
                                break;
                        }
                        if(mspeed==_T(""))
                                str=_T("000");
                        SendData+=str;

                        SendData+=mspeed;

                        if(mtime==_T(""))
                                str=_T("0000")
                        switch(mtime.GetLength())
                        {
                        case 1:
                                str=_T("000");
                                break;
                        case 2:
                                str=_T("00");
```

```
                    break;
                case 3:
                    str=_T("0");
                    break;
                case 4:
                    str=_T("");
                    break;
                }
                if(mtime==_T(""))
                    str=_T("0000");
                SendData+=str;
                SendData+=mtime;

                str= (mrot==FALSE) ? _T("0") : _T("1")
                SendData+=str;
                SendData+=0x1F;
                return SendData;
            }
        case 2:
            {
                SendData+=motorno;
                SendData+=_T("B00000000");
                SendData+=0x1F;
                return SendData;
            }
        case 3:
            {
                SendData+=motorno;
                SendData+=_T("C00000000");
                SendData+=0x1F;
                return SendData;
            }
        }
        return _T("");
}

void CSerialView::StrogeData(CString data)
{
        CSerialDoc *pDoc=GetDocument();
```

```
        int datano;
        datano=atoi(data.Mid(1,1));
        if(pDoc->m_MotorData[datano].GetLength()==12*10)
                pDoc->m_MotorData[datano].Delete(0,12);
        pDoc->m_MotorData[datano]+=data;
}


void CSerialView::ListDisplay(CString RMotorNo)
{
        CSerialDoc *pDoc=GetDocument();


        //if(m_DataList.GetCount()>=10)
        //        m_DataList.DeleteString(0);
        int n=atoi(RMotorNo);
        if(pDoc->m_MotorData[n]==_T(""))
                return;
        int count=pDoc->m_MotorData[n].GetLength()/12;
        for(int a=0;a<count;a++)
        {
                CString imsi;
                imsi=pDoc->m_MotorData[n].Mid(a*12,12)
                switch(n)
                {
                case 0:
                        m_datalist0.AddString(imsi);
                        break;
                case 1:
                        m_datalist1.AddString(imsi);
                        break;
                case 2:
                        m_datalist2.AddString(imsi);
                        break;
                case 3:
                        m_datalist3.AddString(imsi);
                        break;
                case 4:
                        m_datalist4.AddString(imsi);
```

```
                    break;
                case 5:
                    m_datalist5.AddString(imsi);
                    break;
                case 6:
                    m_datalist6.AddString(imsi);
                    break;
                case 7:
                    m_datalist7.AddString(imsi);
                    break;
                case 8:
                    m_datalist8.AddString(imsi);
                    break;
                case 9:
                    m_datalist9.AddString(imsi);
                    break;
                }
        }
}

void CSerialView::OnRM0()
{
        // TODO: Add your control notification handler code here
        CSerialDoc *pDoc=GetDocument();
        m_datalist0.ResetContent();
        pDoc->m_MotorData[0]=_T("");
}

void CSerialView::OnRM1()
{
        // TODO: Add your control notification handler code here
        CSerialDoc *pDoc=GetDocument();
        m_datalist1.ResetContent();
        pDoc->m_MotorData[1]=_T("");
}

void CSerialView::OnRM2()
{
        // TODO: Add your control notification handler code here
```

```
        CSerialDoc *pDoc=GetDocument();
        m_datalist2.ResetContent();
        pDoc->m_MotorData[2]=_T("");
}

void CSerialView::OnRM3()
{
        // TODO: Add your control notification handler code here
        CSerialDoc *pDoc=GetDocument();
        m_datalist3.ResetContent();
        pDoc->m_MotorData[3]=_T("");
}

void CSerialView::OnRM4()
{
        // TODO: Add your control notification handler code here
        CSerialDoc *pDoc=GetDocument();
        m_datalist4.ResetContent();
        pDoc->m_MotorData[4]=_T("");
}

void CSerialView::OnRM5()
{
        // TODO: Add your control notification handler code here
        CSerialDoc *pDoc=GetDocument();
        m_datalist5.ResetContent();
        pDoc->m_MotorData[5]=_T("");
}

void CSerialView::OnRM6()
{
        // TODO: Add your control notification handler code here
        CSerialDoc *pDoc=GetDocument();
        m_datalist6.ResetContent();
        pDoc->m_MotorData[6]=_T("");
}

void CSerialView::OnRM7()
{
```

```
	// TODO: Add your control notification handler code here
	CSerialDoc *pDoc=GetDocument();
	m_datalist7.ResetContent();
	pDoc->m_MotorData[7]=_T("");
}

void CSerialView::OnRM8()
{
	// TODO: Add your control notification handler code here
	CSerialDoc *pDoc=GetDocument();
	m_datalist8.ResetContent();
	pDoc->m_MotorData[8]=_T("");
}

void CSerialView::OnRM9()
{
	// TODO: Add your control notification handler code here
	CSerialDoc *pDoc=GetDocument();
	m_datalist9.ResetContent();
	pDoc->m_MotorData[9]=_T("");
}

void CSerialView::OnStop()
{
	// TODO: Add your control notification handler code here
//	CSerialDoc *pDoc=GetDocument();
	UpdateData();

	Message(m_nPort,bPortInit);
	m_command=2;
	CString Temp;
	Temp.Format("Port COM%d로 Data를 전송합니다!",m_nPort)
	ProcessMessage(Temp);

	m_SendData=_T("");
	//#0 Motor 정지Data//
	if(m_m0chk==TRUE)
		m_MotorData[0]=ConvertData("0",m_command);
	else
```

```
            m_MotorData[0]=_T("");

//#1 Motor 정지Data//
if(m_m1chk==TRUE)
            m_MotorData[1]=ConvertData("1",m_command)
else
            m_MotorData[1]=_T("");

//#2 Motor 정지Data//
if(m_m2chk==TRUE)
            m_MotorData[2]=ConvertData("2",m_command)
else
            m_MotorData[2]=_T("");

//#3 Motor 정지Data//
if(m_m3chk==TRUE)
            m_MotorData[3]=ConvertData("3",m_command);
else
            m_MotorData[3]=_T("");

//#4 Motor 정지Data//
if(m_m4chk==TRUE)
            m_MotorData[4]=ConvertData("4",m_command);
else
            m_MotorData[4]=_T("");

//#5 Motor 정지Data//
if(m_m5chk==TRUE)
            m_MotorData[5]=ConvertData("5",m_command);
else
            m_MotorData[5]=_T("");

//#6 Motor 정지Data//
if(m_m6chk==TRUE)
            m_MotorData[6]=ConvertData("6",m_command);
else
            m_MotorData[6]=_T("");

//#7 Motor 정지Data//
```

```
        if(m_m7chk==TRUE)
                m_MotorData[7]=ConvertData("7",m_command);
        else
                m_MotorData[7]=_T("");

        //#8 Motor 정지Data//
        if(m_m8chk==TRUE)
                m_MotorData[8]=ConvertData("8",m_command);
        else
                m_MotorData[8]=_T("");

        //#9 Motor 정지Data//
        if(m_m9chk==TRUE)
                m_MotorData[9]=ConvertData("9",m_command);
        else
                m_MotorData[9]=_T("");

        for(int a=0;a<10;a++)
                m_SendData+=m_MotorData[a];

        m_ComuPort.WriteComm((unsigned char *)(LPCTSTR)m_SendData,m_SendData.GetLength())

}

void CSerialView::Message(UINT nPort, BOOL PortInit)
{
        if(nPort==0)
        {
                MessageBox("Port를 Setting하십시오!","전송",MB_ICONSTOP | MB_OK);
                return;
        }

        if(!PortInit)
        {
                MessageBox("Port가 연결되지 않았습니다!","전송",MB_ICONSTOP | MB_OK);
                return;
        }

}
```

```
void CSerialView::OnClearrecive()
{
        // TODO: Add your control notification handler code here
        strTemp=_T("");
        m_recive.ResetContent();

}

void CSerialView::OnM0Chk()
{
        // TODO: Add your control notification handler code here
        GetDlgItem(IDC_M0_SPEED)->SetFocus();
}

void CSerialView::OnM1Chk()
{
        // TODO: Add your control notification handler code here
        GetDlgItem(IDC_M1_SPEED)->SetFocus();
}

void CSerialView::OnM2Chk()
{
        // TODO: Add your control notification handler code here
        GetDlgItem(IDC_M2_SPEED)->SetFocus();
}

void CSerialView::OnM3Chk()
{
        // TODO: Add your control notification handler code here
        GetDlgItem(IDC_M3_SPEED)->SetFocus();
}

void CSerialView::OnM4Chk()
{
        // TODO: Add your control notification handler code here
        GetDlgItem(IDC_M4_SPEED)->SetFocus();
}
```

```
void CSerialView::OnM5Chk()
{
        // TODO: Add your control notification handler code here

        GetDlgItem(IDC_M5_SPEED)->SetFocus();
}

void CSerialView::OnM6Chk()
{
        // TODO: Add your control notification handler code here
        GetDlgItem(IDC_M6_SPEED)->SetFocus();
}

void CSerialView::OnM7Chk()
{
        // TODO: Add your control notification handler code here
        GetDlgItem(IDC_M7_SPEED)->SetFocus();
}

void CSerialView::OnM8Chk()
{
        // TODO: Add your control notification handler code here
        GetDlgItem(IDC_M8_SPEED)->SetFocus();
}

void CSerialView::OnM9Chk()
{
        // TODO: Add your control notification handler code here
        GetDlgItem(IDC_M9_SPEED)->SetFocus();
}

void CSerialView::OnRecidata()
{
        // TODO: Add your control notification handler code here
        UpdateData();

        Message(m_nPort,bPortInit);
        m_command=3;
        CString Temp;
```

```
Temp.Format("Port COM%d로 Data를 전송합니다!",m_nPort)
ProcessMessage(Temp);

m_SendData=_T("");
//#0 Motor 수신Data//
if(m_m0chk==TRUE)
        m_MotorData[0]=ConvertData("0",m_command);
else
        m_MotorData[0]=_T("");

//#1 Motor 수신Data//
if(m_m1chk==TRUE)
        m_MotorData[1]=ConvertData("1",m_command);
else
        m_MotorData[1]=_T("");

//#2 Motor 수신Data//
if(m_m2chk==TRUE)
        m_MotorData[2]=ConvertData("2",m_command);
else
        m_MotorData[2]=_T("");

//#3 Motor 수신Data//
if(m_m3chk==TRUE)
        m_MotorData[3]=ConvertData("3",m_command);
else
        m_MotorData[3]=_T("");

//#4 Motor 수신Data//
if(m_m4chk==TRUE)
        m_MotorData[4]=ConvertData("4",m_command);
else
        m_MotorData[4]=_T("");

//#5 Motor 수신Data//
if(m_m5chk==TRUE)
        m_MotorData[5]=ConvertData("5",m_command);
else
        m_MotorData[5]=_T("");
```

```
//#6 Motor 수신Data//
if(m_m6chk==TRUE)
        m_MotorData[6]=ConvertData("6",m_command);
else
        m_MotorData[6]=_T("");

//#7 Motor 수신Data//
if(m_m7chk==TRUE)
        m_MotorData[7]=ConvertData("7",m_command);
else
        m_MotorData[7]=_T("");

//#8 Motor 수신Data//
if(m_m8chk==TRUE)
        m_MotorData[8]=ConvertData("8",m_command);
else
        m_MotorData[8]=_T("");

//#9 Motor 수신Data//
if(m_m9chk==TRUE)
        m_MotorData[9]=ConvertData("9",m_command);
else
        m_MotorData[9]=_T("");

for(int a=0;a<10;a++)
        m_SendData+=m_MotorData[a];

m_ComuPort.WriteComm((unsigned char *)(LPCTSTR)m_SendData,m_SendData.GetLength())

}

}
```