GOVP1200515333

최 종
연구보고서

중량과 재배관리, 수확 및 동시선별 적재작업을 위한 다목적 생력 작업시스템 개발

Development of Modular Type Multi-purpose Tele-operative Robot System for Watermelon Cultivation, Harvesting, Sorting, and Loading

연 구 기 관

성균관대학교

농 림 부

최 종 보 고 서

2002년도 농림기술개발사업에 의하여 완료한 중량과 재배관리, 수확 및 동시선별 적재작업을 위한 다목적 생력작업시스템 개발에 관한 연구의 최종보고서를 별첨과 같이 제출합니다.

첨부 : 1. 최종보고서 10부

2. 최종보고서 디스켓 1매

2002. 10. .

주관연구기관 : 성균관대학교

총괄연구책임자 : 황 헌 (인)

주관연구기관장 : 성균관대학교 총장 직인

농 림 부 장 관 귀 하

제 출 문

농림부 장관 귀하

본 보고서를 "중량과 재배관리, 수확 및 동시선별 적재작업을 위한 다목적 생력 작업시스템 개발" 과제의 최종보고서로 제출합니다.

2002. 10. .

주관연구기관명 : 성균관대학교

총괄연구책임자 : 황 헌
연 구 원 : 최 창현
연 구 원 : 김 영길
연 구 원 : 김 시찬
연 구 원 : 임 동혁
연 구 원 : 최 태현

요 약 문

Ⅰ. 제 목

중량과 재배관리, 수확 및 동시선별 적재작업을 위한 다목적 생력작업시스템 개발

Ⅱ. 연구개발의 목적 및 필요성

국내 시설재배의 경우, 육묘작업, 시비(施肥)작업, 정지(整地)작업, 파종(播種)작업, 관수(灌水)작업, 수확작업, 선별작업, 포장출하작업 등의 다양한 작업공정에 대한 생력화(Automation) 연구가 개별 작업단위로 활발히 추진되어 오고 있다.
하지만, 이들 개개 단위작업의 특성은 효율적인 생력화(Automation) 시스템의 개발 및 적용에 문제가 되고 있다. 또한 단위 작업공정별 전용 작업기계의 종류가 다양해지고 단위 작업기계의 가동률은 일반 산업기계와 비교하여 현저히 떨어지기 때문에 시설내의 작업 기계화 및 생력화(Automation) 실현에 장애가 되고 있다. 특히 일부 작업에 있어서는 대상물의 인식·판별에 대한 실시간 처리 기술의 어려움, 대상작목에 따라 요구되는 취급설비의 고 정밀성, 가변적인 작업환경 등 생력화(Automation) 작업설비의 개발에 있어 기술적 문제를 포함하여, 경제성, 실용성, 유지보수 측면에서 큰 어려움이 있다. 따라서 시설재배 또는 노지(露地)재배의 생력화(Automation)를 위한 작업기 및 설비의 개발은 경제성, 실용성 그리고 작업의 난이도 및 기계 가동률 등의 문제점을 효율적으로 해결할 수 있도록 새로운 각도에서의 접근이 필요하다.

시설재배 하의 수박(반촉성(促成)) 재배의 경우 작업별 해당 노동 투하시간은 10a 당 수확에 58.9시간, 선별 및 포장이 26.4시간, 접목 작업에 43시간, 정식(이식)에 46.1시간, 묘판 관리에 45.7시간 그리고 제초에 26시간이 투여되고 있다. 노동투하측면에서 큰 비중을 차지하는 작업은 수확, 선별 및 포장, 정식(이식), 접목, 묘판 관리, 제초 작업 등이다. 대부분의 작업이 시설 내에서 이루어지고 있으나, 작업공간의 한정성, 기기의 이동성, 다양한 설비 등으로 인하여 실질적 적용사례는 극히 드문 실정이다. 따라서 시설재배를 통한 과채류 생산에 있어서 참외 및 수박과 같은 중량과의 경우에는 수확,

재배관리 등의 작업에 현실 적용성을 고려한 생력화(Automation) 설비를 개발하여 부족한 노동력 및 힘든 작업을 대체하고 전반적인 작업생산성을 향상시키는 것이 절실하다.

특히, 최근에는 과거 7, 80년대와 다르게 환경에 대한 인식이 확산되어 유기농업, 친환경농업, 정밀농업 등 환경친화적인 농업작업에 대한 연구가 활발하게 진행되고 있다. 시설농업의 경우 설비투자비가 높아 재배공간은 고밀도화 되어 있으며 생산성을 높이기 위해 식물의 생육상태를 최적화 하도록 환경을 제어하는 것이 일반적이다. 한정된 작업 공간 및 환경적인 요인(병 및 충해의 유입 방지, 작물의 최적 생육을 위한 시설내 이산화탄소, 습도, 질소 등) 으로 인하여 대개는 작업자의 시설내 출입을 최소화하고 있으며, 때로는 작업자의 건강 및 쾌적한 작업환경의 확보가 어렵다. 작업자의 작업환경을 개선하고 작물재배에 소요되는 환경유해 물질을 최소화하는 자동화 작업 시스템의 개발은 생산성 확보, 생산물의 고품질화와 더불어 매우 중요하다.

향후, 시설생산 농가의 육성을 위해서는 노동집약적 작업을 생력화(Automation)할 수 있는 설비개발을 통하여 시설생산 농가의 노동력 부족 및 고 임금 실태에 대처하여야 한다. 특히 시설재배를 통한 작물생산 작업은 다양한 작업공적으로 구성되어 있고 이들 공정의 대부분이 단기간의 집약적 노동을 요구하고 있다. 시설내의 개별적 작업을 생력화하기 위한 기계 또는 시스템에 대한 연구 개발이 시도되었으나 경제성 및 작업 효율성 측면에서 전혀 기대에 미치지 못하고 있는 실정이다. 따라서, 실질적으로 작업 생산성을 향상시킬 수 있고 전반적인 시설재배 작업에 투여되는 기계비용을 최소화할 수 있는 효율적인 다목적 생력(Multi-purpose Task Automation) 작업기계의 개발이 필수적이다. 이러한 다목적 생력작업 설비의 개발을 통하여 작업생산성을 향상시킴으로서 국내 생산 과채류의 국제 경쟁력을 확보할 수 있고 국내 시설농업 경영자의 경영수지를 개선할 수 있다.

이러한 필요성을 바탕으로 본 과제의 연구개발 목적을 다음과 같이 설정하였다.

시설 내 적용성을 고려한 수박의 재배관리, 수확 및 동시선별 적재작업을 위한 모듈(module)화 된 다목적 원격 생력작업 시스템 개발

위의 연구 목적에 의거하여 기계 적응성, 효용성 및 경제성, 산업화 가능성을 고려하여 아래와 같이 세부 연구 목적을 설정하여 연구개발을 수행하였다.

1. 모듈(Module)화 개념의 작업장치 개발을 통하여 다양한 단위작업 공정에 대처할 수 있도록 함으로써 개발 시스템의 작업 가동률을 극대화시키고 전체 작업의 생력화에 소요되는 기계비용을 최소화하며 각 작업별 작업성, 생산성 및 작업정밀도를 향상시킨다.

2. 방제(防除)작업과 같이 인체에 유해한 작업환경에서도 작업을 가능하게 하고 국지적(Site-Specific) 개념의 정밀농법에 의거 방제물질의 사용을 최소화하는 동시에 효율적인 방제가 가능하도록 하는 환경보전적 작업시스템을 개발한다.

3. 작업자의 작업환경 개선 및 다수 시스템을 원격지에서의 중앙 통제하는 원격 유/무인 시스템 제어에 입각한 Tele-operative 로봇시스템을 개발한다.

4. 선별작업의 경우, 수확시 크기, 무게, 색깔 등의 품질 관련요소를 판정하여 선별적재가 가능하도록 하여 적기 수확을 가능하게 하고 상품의 고부가가치화와 농가의 수익증대에 기여하는 시스템을 개발한다.

5. 신기술의 축적을 통하여 외국기술에의 의존을 탈피하고 국내기술의 자립에 기여하며 국내 농업기계 제조업체의 시설생산 관련 설비개발능력을 촉진시키고 국내개발 생력화(Automation) 설비의 해외수출에 기여한다.

Ⅲ. 연구개발 내용 및 범위

본 과제에서 목표로 하는 연구 개발의 내용 및 범위는 다음과 같다.

○ 본 연구과제는 현재 생산체계로 볼 때 실질적으로 적용성이 높고, 타 작목에 비해 고부가가치는 있으나 중량으로 인하여 비교적 수작업에 의한 취급이 어려운 중량과 중 수박을 대상으로 연구한다. 수박의 수확작업 생력화를 주목적으로 재배관리, 방제(防除), 재초 및 선별작업 등에도 작업선단부를 단순 교체함으로써 공통적으로 이용할 수 있는 다목적 생력작업 시스템을 개발한다. 또한 주변기기 및 작업시스템에 필요한 설비 및 요소기술을 연구 개발한다.

○ 수작업 위주의 수확, 관리(돌리기), 이송운반 작업에 필요한 모듈(module)형의 간이 생력화(Automation) 단위장치를 포함하여 작업 선단부 교체가 가능한 모듈(module)형 시설재배 작업용 다기능 생력 작업시스템을 개발한다.

○ 본 연구는 인식·판단 기술의 개발, 원격조작에 의한 작업기술 및 장치 개발을 통하여 농업 생산기술의 혁신을 추진하는 동시에, 실험실 수준의 기초기술 확보를 뛰어넘어, 경제성과 적정 작업기능성에 바탕을 둔 실질적인 현장 투입용 작업기 및 생력화(Automation) 작업설비의 개발연구를 수행한다. 다목적 생력작업 시스템과 관련 주변기기 및 작업설비의 개발 연구에 대한 개괄적 연구 내용은 다음과 같다.

- 수확 및 관리(돌리기), 이송을 위한 시설재배 작업공정의 분석과 경제성과 현장 투입성을 고려한 개발 작업시스템의 구체적 작업 기능 사양 설정 및 작업공간, 가반(Load) 중량, 동작궤적(軌迹), 동작제어, 동특성, 작업속도, 동력, 이동성, 조작성을 고려한 구동원 설정 구축 및 작동장치 구성

- 작업기능 및 조작성과 내구성 그리고 작업안전성을 고려한 정밀 직교 3축 작업기를 탑재한 갠트리 시스템, 수확 후 이송장치, 작업기와 선단 작업부의 동력전달장치, 기구부, 구동부, 제어부, 조작부 설계와 단위 시작장치 구성 및 제작

작업지시 제어 및 작업구동 소프트웨어 개발과 라이트 펜 또는 터치 패드 제어 시스템의 구축 및 설계 제작

터치스크린에 의한 인터페이스 및 수확, 수박 관리(돌리기), 제초, 방제(防除), 선별, 관수(灌水) 등 단위 작업별 작업 프로그래밍 소프트웨어 개발 및 구동 하드웨어 구축.

원격조작을 위한 무선 칼라 카메라시스템 구축 및 국부(局部)영상처리 시스템 개발

원격 조작신호의 무선 송수신 처리 및 무선 작업기 구동 시스템 구축

자동 작업기에 간편하게 장/탈착할 수 있는 모듈(module)형 수확용 그리퍼(Gripper), 꼭지 절단 장치, 수박 자세 변환용 그리퍼(Gripper) 기능설정 및 개념설계와 장치 제작

수확시 수박의 무게(로드 셀) 및 크기(영상처리) 등의 계측이 가능한 실시간 품질측정 및 판정을 위한 신호처리 시스템 구축과 품질판정 알고리즘 개발

다목적 생력 작업시스템의 작업모듈(module) 장, 탈착 성능, 작업별 성능시험 및 현장적응성 보완 연구

Ⅳ. 연구개발 결과 및 활용에 대한 건의

본 연구는 중량과 재배관리, 수확 및 동시선별 적재작업을 위한 다목적 생력작업시스템 개발에 관한 연구로 국부(局部) 영상처리 기술, 원격 제어 및 시스템 관리기술, 모듈(module)화 된 작업 도구 개발 및 운용 기술 등을 이용하여 시설 내에서 수박의 재배관리를 생력화(Automation) 하게 하였다. 그리고 이러한 개발 기술을 통하여 다른 중량과의 적용성 및 생산품의 고품질화를 확보 할 수 있도록 하였다.

다음은 세부적인 연구결과이다.

○ 시설 내의 작업환경 및 작부 체계에 대한 연구 및 작업 시스템의 현장 적응성을 높이는 주변 환경 구축을 위한 연구를 수행하였다.

○ 시설 내 자연광 하에서 정확히 목표물의 외형 즉 외곽선 추정 둘레 찾기, 높이 추정 등을 위한 국부(局部) 영상처리 알고리즘(Local image processing algorithm)을 개발을 위한 연구를 수행하였고, 유/ 무인 국부(局部)영상처리 알고리즘 및 시스템을 개발하였다.

○ 터치스크린 인터페이스를 이용한 작업교시(敎示) 및 작업영역 설정, 시스템 설정을 수행하기 위한 소프트웨어 및 사용자 인터페이스를 개발하였다.

○ 모듈(module)화 작업 장치의 정확한 목표물 추적을 위하여 분산제어 시스템 구축에 관한 연구를 수행하고, 시스템 제어 소프트웨어를 개발하였다.

○ 방제(防除), 시비(施肥), 제초, 가지치기, 돌리기, 수확을 위한 모듈(module)화 작업 장치에 대한 연구를 수행하여, 모듈(module)을 쉽게 교체할 수 있는 전기, 전자적 인터페이스와 기계적 인터페이스를 구축하였다. 그리고 가지치기 모듈(module), 수박 돌리기 모듈(module), 수확을 위한 전동 및 진공, 공압을 이용한 그리퍼(Gripper) 모듈(module)을 개발하였다.

○ 원격지에서의 재배관리 및 시스템 통제를 위한 아날로그 및 디지털 원격 무선 제어시스템을 설계, 구축하였고, 시스템을 통재할 수 있는 통신 프로토콜 및 통신 소프트웨어를 적용/개발하였다.

○ 수확한 수박의 중량 및 외형 선별을 위한 중량 선별 시스템 및 국부(局部) 영상 처리 알고리즘 연구를 통하여 품질 선별 시스템을 구축하였다.

○ 1, 2차에 걸친 시작기의 제작과 각 모듈(module)별 설계 및 제작을 통하여 갠트리(Gantry)형 로봇 Arm 및 레일 구동형 대차로 구성된 작업 시스템을 개발하였다. 시스템의 각 부분은 사용자 직접제어 및 무선 통신을 이용하여 원격제어가 가능토록 하였으며, 개별 동작 및 작업별 프로파일 동작을 수행하여 현장 적응성을 높였다.

본 연구는 상기와 같이 중량 과채류에 대한 통합 재배관리용 로봇작업 시스템을 개발하였고, 실내 실험을 통해 양호한 결과를 얻었다. 그러나 현장 보급 및 상품화를 위해서는 시설 내에서 지속적인 적응성 연구와 및 선단작업부의 기능 개선연구와 작물의 생육모델 개발, 시공간 생육 데이터베이스 확보, 작물환경에 따른 생육상태의 정량화 감시 등 개발한 시스템을 이용한 작물생육 측면에서의 연구가 향후 수행되어야 할 것이다.

SUMMARY

I. Title

Development of Modular Type Multi-purpose Tele-operative Robot System for Watermelon Cultivation, Harvesting, Sorting, and Loading

II. Objectives and necessity

There have been some research and developments attempts to develop the automated machinery or system to handle each unit process among various processes required for the greenhouse cultivation. Especially, most researches were focused to pesticide and sorting. Some efforts have been done in automating grafting process of fruit bearing vegetables and other plants.

Though many research and development efforts have been made world-wide including Korea toward automating individual process of greenhouse operation, they were not successful in the real field operation. Some of them showed just the feasibility of application through the laboratory experiment. Though some of the major failures in field application were difficulty of real time processing in task identification, too much complex system requirement for farmers to handle, technical difficulties in adapting the system to various and variable environment, and lack of the robustness of the system and complexity of the system maintenance. Furthermore, the required system cost was too high and machine or system operating time for a unit individual task was too short compared to the overall plant growth period. It was the main motivation of this research. Purpose of this research was developing a newly concept machine or system which can

solve efficiently most of the obstacles mentioned above in automating greenhouse plant production process, while maximizing system operation time and reducing total system investment.

In Korea, in cultivating greenhouse watermelon, harvesting required 58.9 hours of labor time per 10ar. And it required 26.4 hours of labor for sorting and packaging, 43 hours for grafting, 45.7 hours for seedbed administration, 46.1 hours for planting, and 26 hours for weeding. Considering labor requirements of individual task during plant growth, harvesting, pruning, and weeding are most time consuming. Though these tasks were performed inside the greenhouse, limitation of the space availability for a machine, travel space availability of usual cultivating machine and variety of machinery corresponding to each individual task became bottlenecks in developing automated machinery systems or machines. Therefore, it is necessary to develop an innovated system to solve those intrinsic barriers of greenhouse cultivation. Watermelon was chosen as a representative plant of weighted fruit bearing vegetables to implement and realize the proposed system research and development.

These days, differently from 1970s and 80s', the issue of environment preservation was spreaded over the nation. Researches and developments on organic farming, environment friendly cultivation, and precision farming have been widely spreaded and actively performed. Greenhouse cultivation often requires high investment of system. Because of that, density of planting is high and precise control of plant environment such as atmosphere and nutrition is usually being exercised. It is known to be desirable to maintain the greenhouse as a closed space separated from outside to prevent migration of bad insects and diseases from outside. Besides, the optimum atmosphere for plants are usually not good for human health because of the excessive CO_2. And it is highly required to minimize the application of chemical pesticide and fertilizer. The proposed multi-purpose task automating system should cope with not only above problems efficiently but also quality

maximization and improvement of overall production rate.

In reality, automating agricultural field operation often requires real time information processing such as job environment recognition including object, adaptive decision-making, etc. There have been many research efforts to substitute those human functions and various sophisticated and expensive technologies have been introduced. So far, however, it is still far from the success especially for the outdoor field operation. Moreover, producing agricultural products such as fruits, vegetables, and crops requires many diverse processes, which require various machines to automate each process. Compared with humans versatility, those machines have been developed for the individual process with specialized function. These may help to automate each unit process successfully to some degree but resulted into the increase of the total production cost and decrease of the machine operation.

According to the previously mentioned necessities, the purpose of this research was formed as following:

Final Goal: **Development of Modular Type Multi-purpose Tele-operative Robot System for Watermelon Cultivation, Harvesting, Sorting, and Loading**

Considering greenhouse adaptation of machinery, machine operating efficiency, economical efficiency, and feasibility of commercialization, subcategories of research targets were set up as following.

1. Through the development of modular type tools, system is adapted to various tasks resulting into the improvement of machine operation time, reduction of the system cost required for individual task automation, and improvement of the task handling efficiency.

2. System is developed to be environment friendly, while keeping farmer or

operator off the cultivation site and introducing site-specific precision application of pesticide or fertilizer including water.

3. System adopts tele-operative concept through developing wireless remote control of the system, data and signal communication, and operator friendly man-machine control interface.

4. System is capable of sorting during cultivation and harvesting process via processing size, color, and weight of watermelon resulting into farm profit increase and production of high value added watermelon.

5. Via accumulating innovative technology and introducing high end technology to agriculture in Korea, the increased international competitiveness in agricultural machinery business and export of system and technology are to be expected.

III. Research content and scope

Content and scope of research and development of this project are as following:

○ Watermelon was chosen as a representative fruit bearing vegetables in this research. Watermelon is high value added vegetable, but is difficult to handle because of its weight. Through this research, modular type Multi-task Tele-operative robot system for watermelon cultivation, harvesting, sorting, and loading is developed. And technology related to peripherals and system element required to develop the proposed system is also studied.

○ Modular type unit system such as drive controller, wireless remote data transmission, image acquisition device and so on including end-tool exchanger are developed.

○ This project pursues developing system which can be applied to the field while satisfying the needs of economical efficiency and functionality of machine to perform various cultivation tasks including harvest successfully. The goal is not limited to the acquisition of laboratory level basic technology but aims to develop the practical multi-purpose high end system which utilizes frontier technologies such as recognition and decision finding technology under variable environment and tele-operative task command and data communication technology. Following shows brief but overall sub contents of this project.

- Characteristics of individual tasks such as cultivation, harvest, and loading is analyzed. Functional specification of machine to be developed such as workspace, load capacity, motion trajectory, dynamic characteristics, operating speed, required power, and maneuverability are set up based on the defined job function. Driving source and mechanism are selected and built.

- Robotic gantry loaded with a precise 3-axis manipulator, transport device, power transmission device, kinematic structure, power drive and control mechanism is developed based on the machine function determined from task specification, manipulability, and endurance.

- Design and building task command control unit, driving hardware and control software, touch pad based remote control system and interface unit are performed. Software module for an individual task is developed.

- Wireless remote image acquisition system is built and local image processing algorithm is developed. Wireless remote data and control signal transmission unit is built with remote driving system.

- Modular type detachable various end-grippers for each task such as

harvest, pesticide, stem cutting, attitude alternator, and pruning are designed and built.

- Quality evaluation is done via real time measuring of the weight, size, and color of watermelon during harvest.

- Performance and field adaptability test of the developed system for individual task and modification is done.

Ⅳ. Results and Suggestion

In this research, tele-operative system automation, which is a new concept of automation for bio-production was proposed and prototype of a multi-purpose modular type tele-operative robotic system was developed. The proposed system showed practical and feasible way of automation for the volatile bio-production process. Based on the proposition, job environment recognition with object identification was performed using computer vision system. Processing an outdoor camera image to extract some useful information is very difficult and requires heavy computing because of the light variation, complexity of the background and so on. The environment sensitivity was a key barrier to robotize the field operations in bio-production. A man-machine interactive hybrid decision-making, system drive control with wireless task command control, and wireless data and signal transmission, which utilized a concept of tele-operation was developed to overcome limitations in recognizing the complex and variable environment while improving robustness of machine operation.

Following shows summary of results obtained from this research.

○ Mechanization research on integrated systematic cultivation of watermelon was performed, which allows practicability, labor saving, production efficiency, and economical efficiency.

○ Local image processing which overcomes variation of natural light in object recognition and job execution.

○ Touch screen based man-machine interface was developed. Software modules for task teaching, task type and space specification, system set up, and graphic user interface were developed.

○ Distributed control scheme was introduced to trace target object precisely and stereo vision system was built and tested for target recognition.

○ Mechanical and electro-electrical interfaces for the modular type detachable end tools were designed and developed. Modular end-tools for various tasks such as pesticide, fertilizing, weeding, pruning, turning, and harvest with stem cutting were developed.

○ Hardware and software module for remote wireless data and signal transmission, communication protocol, system control module were developed.

○ Two prototypes were built and tested. One is for out-field and the other one was for greenhouse. 3-axis rectangular manipulator was mounted on the gantry type rail driven base system. Each unit of system was controlled through remote command control, which is specified by user and transmitted via radio communication. Through individual remote task operation, function of the system and field adaptability were tested.

Major deficiencies of current automation scheme including various robots for bio-production include the lack of task adaptability and real time processing, low job performance for diverse tasks, and the lack of robustness of task results, high system cost, failure of the credit from the operator, and so on. This paper proposed a scheme that could solve the current limitation of task abilities of conventional computer controlled automatic system. The proposed scheme is the

man-machine hybrid automation via tele-operation which can handle various bio-production processes. And it was classified into two categories. One category was the efficient task sharing between operator and CCM(computer controlled machine). The other was the efficient interface between operator and CCM.

As an interface system between operator and CCM, a touch pad screen mounted on the monitor and remotely captured imaging system were used. Object indication was done by the operator's finger touch to the captured image using the touch pad screen. A certain size of local image processing area was specified after the touch was made. And image processing was performed with the specified local area to extract desired features of the object. An MS Windows based interface software was developed using Visual C++6.0. The software was developed with four modules such as remote image acquisition module, task command module, local image processing module and 3D coordinate extraction module.

Developed system showed the feasibility of real time processing, robust and precise object identification, and adaptability of various job and environments through selected sample tasks. Researches related to plant science such as growth modeling, building spatio-temporal plant growth database, and quantitative monitoring of plant growth state are highly recommended utilizing the developed system. For further research, it is required to have a performance test and end-tool modification according to the continuous greenhouse site experimentation research in order to commercialize the developed system.

CONTENTS

목 차

제 1 장 연구개발과제의 개요

제 1 절 연구목표

1. 최종 연구 목표

중량과 재배관리, 수확 및 동시선별 적재작업을 위한 다목적 생력작업 시스템 개발

2. 최종 연구 개발 사업 세부 목표

가. 본 연구과제는 우선적으로, 현재 생산체계로 볼 때 실제 적용성이 높고, 타 작목에 비해 고부가가치가 있으나 중량으로 인하여 비교적 취급이 어려운 중량과(수박 등)를 대상으로 연구한다. 수박의 수확작업을 목표로 하며 재배관리, 방제(防除), 제초 및 선별작업 등에도 작업 선단부의 단순교체로써 공통적으로 이용할 수 있는 다목적 생력작업(Automation work) 시스템을 개발하고 주변기기 및 작업 시스템에 필요한 설비 및 요소기술을 연구 개발한다.

나. 수작업 위주의 수확, 관리(돌리기), 이송운반 작업에 필요한 모듈(module)형의 간이 생력화(Automation) 단위장치를 포함하여 작업 선단부 교체가 가능한 모듈(module)형 시설재배 작업용 다기능 생력 작업시스템을 개발한다.

다. 본 연구는 인식·판단 기술의 개발, 원격조작에 의한 작업기술 및 장치 개발을 통하여 농업 생산기술의 혁신을 추진하나, 실험실 수준의 기초기술 확보를 뛰어넘어, 경제성과 적정 작업기능성에 바탕을 둔 실질적인 현장 투입용 작업기 및 생력화(Automation) 작업설비의 개발연구를 수행한다. 다목적 생력작업 시스템과 관련 주변기기 및 작업설비의 개발 연구에 대한 개괄적 연구 내용은 다음과 같다.

1) 수확 및 관리(돌리기), 이송을 위한 시설재배 작업공정의 분석과 경제성과 현장 투입성을 고려한 개발 작업시스템의 구체적 작업기능사양의 설정 및 작업 공간, 가반(Load)중량, 동작궤적(軌迹), 동작제어, 동특성, 작업속도, 동력, 이동성, 조작성을 고려한 구동원 설정 구축 및 작동장치 구성.

2) 작업기능 및 조작성과 내구성 그리고 작업안전성을 고려한 정밀 직교 3축 작업기를 탑재한 매거진과 매거진 이송용 2축 구동 프레임, 이송 및 선별용 컨베이어, 작업기와 선단 작업부의 동력전달장치, 기구부, 구동부, 제어부, 조작부 설계와 단위 시작장치 구성 및 제작.

3) 작업지시 제어 및 작업구동 소프트웨어 개발과 터치패드 원격제어 시스템의 구축 및 설계 제작.

4) 터치패드에 의한 인터페이스 및 수확, 수박 관리(돌리기), 제초(除草), 방제(防除), 선별(選別), 관수(灌水) 등 단위작업별 작업프로그래밍 소프트웨어 개발 및 구동 하드웨어 구축.

5) 원격 조작을 위한 칼라 카메라 시스템 구축 및 국부(局部) 영상처리 시스템 개발.

6) 원격 조작신호의 무선 송수신 처리 및 작업기 구동 시스템 구축.

7) 시설 내 작업조작 간편성을 고려한 안테나형 베이스 설계와 컨베이어 이송 시스템 및 차량탑재용 시작기 제작.

8) 자동 작업기에 간편하게 장착(裝着), 탈착(脫着)할 수 있는 모듈(module)형 수확용 그리퍼(Gripper), 꼭지 절단장치, 수박 자세변환용 그리퍼(Gripper) 기능설정 및 개념설계와 장치 제작.

9) 수확과 동시에 수박의 무게(로드셀) 및 크기(영상처리)등의 실시간 품질측정 및 판정을 위한 신호처리 시스템 구축과 관련 기구부 및 품질판정 알고리즘 개발.

10) 수확 후 이송장치 구축.

11) 다목적 생력 작업시스템의 작업모듈(module) 장착, 탈착 성능 및 기능시험.

12) 작업별 성능시험 및 현장적응성 보완 연구.

제 2 절 연구의 필요성

1. 기술적 측면

가. 국내 시설재배의 경우, 육묘(育苗)작업, 시비(施肥)작업, 정지(整地)작업, 파종(播種)작업, 관수(灌水)작업, 수확작업, 선별(選別)작업, 포장출하작업 등의 다양한 작업공정에 대한 생력화(Automation) 연구가 개별 작업단위로 활발히 추진되어오고 있다. 하지만, 이들 개개 단위작업의 특성은 효율적인 생력화(Automation) 시스템의 개발 및 적용에 문제가 되고 있다. 또한 단위 작업공정별 전용 작업기계의 종류가 다양해지고 단위 작업기계의 가동률은 일반 산업기계와 비교하여 현저히 떨어지기 때문에 시설내의 작업 기계화 및 생력화 실현에 장애가 되고 있다.

나. 특히 일부 작업에 있어서는 대상물의 인식·판별에 대한 실시간 처리 기술의 어려움, 대상작목에 따라 요구되는 취급설비의 고 정밀성, 가변적인 작업환경 등 생력화 작업설비의 개발에 있어 기술적 문제를 포함하여, 경제성, 실용성, 유지보수 측면에서 큰 어려움이 있다.

다. 따라서 시설재배 또는 노지(露地)재배의 생력화를 위한 작업기 및 설비의 개발

은 경제성, 실용성 그리고 작업의 난이도 및 기계 가동률 등의 문제점을 효율적으로 해결할 수 있도록 새로운 각도에서의 접근이 필요하다.

라. 시설재배의 수박(반촉성(促成))의 노동투하시간은 표 1-2-1과 같다. 수박의 경우 작업별 해당 노동 투하시간은 10a 당 수확에 58.9시간, 선별 및 포장이 26.4시간, 접목 작업에 43시간, 정식(이식)에 46.1시간, 묘판 관리에 45.7시간 그리고 제초에 26시간이 투여되고 있다.

마. 표 1-2-1에서 볼 수 있듯이 수박(반촉성(促成))재배 작업에서 노동투하측면에서 큰 비중을 차지하는 작업은 수확, 선별 및 포장, 정식(이식), 접목, 묘판 관리, 제초 작업 등이다. 대부분의 작업이 시설 내에서 이루어지고 있으나, 작업공간의 한정성, 기기의 이동성, 다양한 설비 등으로 인하여 실질적 적용사례는 극히 드문 실정이다.

바. 따라서 시설재배를 통한 과채류 생산에 있어서 참외 및 수박과 같은 중량과의 경우 수확, 재배관리 등의 작업에 있어 현실 적용성을 고려한 생력화(Automation) 설비의 개발을 통하여 부족한 노동력 및 힘든 작업을 대체하고 전반적으로 작업생산성을 향상시키는 것이 절실하다.

사. 시설 재배시의 재배관리 및 수확작업에 있어 다양한 재배 형태에 능동적으로 대처할 수 있는 다목적 생력화(Automation) 작업기를 개발하여 시설재배를 통한 중량과의 생산에 요구되는 대부분 작업에 있어 일괄적으로 생산성을 향상시키고 다목적 기능을 통한 작업기계의 가동률을 높여 단위 생력화 시스템의 작업능률을 극대화 할 수 있다.

아. 시설 재배의 관리 및 수확 선별작업을 위한 다목적 생력화 작업기의 개발을 통하여 현시점에서 기술적용이 곤란한 인식·판단 기술을 새로운 관점에서 재고하여 시설 재배에서의 기계화 실용도를 높일 수 있으며, 이를 통하여 첨단기술의 실질적 농업적용이 가능해지며 시설 재배분야의 생력화 기술을 선도하게 될 것이다.

<Table 1-2-1> Labour Time for Watermelon Cultivation

작업명	노동투하시간(h/10a)
종자예조 및 소독	1.2
묘판준비 및 설치	14.6
파 종	17.3
묘판 관리	45.7
경운 정지(整地)	12.5
퇴비 및 기비(基肥)주기	14.0
정 식(이 식)	46.1
추비(秋肥) 주기	10.1
병충해 방제(防除)	18.7
제 초	26.0
피복 및 복토	17.6
적심(摘心)적아(摘芽)(접목)	43.0
물 관 리	16.2
온도 관리	24.5
수 확	58.9
하우스설치 및 관리	69.0
선별 및 포장	26.4
운반 및 저장	21.1
기 타	3.2
합 계	486.2

자료 : 작목별 작업단계별 노동력 투하시간. ('94, 농촌진흥청)

2. 경제·산업적 측면

가. 시설 생산농의 육성을 위해서는 노동집약적 작업의 생력화(Automation) 설비개발을 통하여 시설 생산농가의 노동력 부족 및 고임금 실태에 대처하여야 하고 실질적으로 작업생산성을 향상시킬 수 있고 기계 비용을 절감할 수 있는 효율적인 다목적 생력화 작업기계의 개발이 필수적이다.

나. 생력화 설비의 개발을 통하여 생산성을 향상시킴으로서 국내 생산 과채류의 국제 경쟁력을 확보할 수 있고 국내 시설농업 경영자의 경영수지를 개선할 수 있다.

다. 개발한 다목적 생력화 작업시스템은 모듈(module)화 개념으로 개발하여 각 단위작업 공정에의 적용이 용이하여 저렴한 기계설비 비용으로 각 작업별 생산성 및 작업정밀도를 향상시키고 고품질 과채류의 생산에 기여한다.

라. 방제(防除)작업의 경우 개발한 다목적 생력작업 시스템으로부터 간단하게 작업선단부를 교체함으로써 자동살포가 가능해 지고 필요한 곳에만 직접 살포하는 정밀방제(防除)의 실현이 가능하다. 이를 통하여 농약의 살포량을 절감시키고 환경보전적 영농(營農)을 가능하게 한다.

마. 제초(除草)작업의 경우 역시 개발한 다목적 생력작업 시스템으로부터 간단하게 작업선단부를 교체하여 잡초의 선별적 자동제초가 가능하다. 이로써 제초제의 절감 또는 제초제 사용을 억제할 수 있다. 따라서 환경보전적 영농을 가능하게 한다.

바. 선별(選別)작업의 경우 역시 수확 시 크기, 무게, 색깔 등의 품질 관련요소를 판정하여 선별 적재가 가능하여 상품의 고부가가치화와 농가의 수익증대에 기여한다.

사. 장기적으로 볼 때 신기술의 축적을 통하여 외국기술에의 의존을 탈피하고 기술

자립에 기여하며 국내 농업기계 제조업체의 설비 개발능력을 촉진시키며 국내 개발 생력화(Automation) 설비의 해외 수출을 통하여 농업기계 산업체의 활성화를 촉진하게 된다.

3. 사회·문화적 측면

가. 농업기술의 첨단화로 농업이 사양산업이라는 고정관념을 탈피하는데 기여한다.

나. 시설 원예 작업의 3D((Dirty, Difficult, Dangerous) 인식 탈피 및 작업기피 성향을 개선하게 된다.

다. 국민들의 관심을 유발시켜 첨단 농업으로서의 인식을 제고하고 향후 우수인력의 농업분야 유입을 촉진한다.

라. 단순 노동이라는 농작업의 관념을 일신시키고 전문 기술자에 의한 기술농업으로서 위치를 공고히 하게 된다.

제 3 절 연구개발 방법 및 설계

1. 연구 개발의 설계 인자

가. 본 연구과제의 목표인 시설재배에서의 중량 과채류의 재배관리, 제초, 방제(防除) 및 수확 선별작업을 위한 다목적 생력 작업시스템 개발을 위하여 다음과 같은 설계 인자를 우선적으로 검토하여 연구를 수행하였다.

1) 시설재배 과채류 생산작업을 위한 생력화(Automation) 설비 및 기계의 개발에 있어서 고려할 사항

2) 수확, 재배관리, 제초 및 방제(防除) 등 각 단위 작업 공정이 독특한 작업특성을 가지고 있고 작업여건 상 가변성이 있는 환경 그리고 재식(栽植)밀도가 높은 시설 내에서 작업을 수행하여야 하는 작업여건으로 인하여 실질적으로 단위작업별 전용기 특히나 무인 전용기의 개발은 기술적 난이도 문제는 배제하더라도 전체 재배작업체계에서 볼 때 생력화 기계설비비의 비중이 높아져 경제성이 없을 뿐 아니라 각 작업별 단기간 작업특성으로 인하여 단위작업별 전용기계의 가동효율이 떨어지고 유지보수의 문제점 등 현장 적용에 있어서 상당한 문제점을 내포하고 있다. 따라서 새로운 개념의 생력 기계화 연구를 통한 작업기계의 개발이 현 시점에서 필요하다.

3) 작업자의 기계조작 및 관리기술을 고려한 생력화설비를 개발한다.

4) 시설 내 과채류 생산의 전체 작업공정에 고려한 생력화설비.
생력화 설비의 가동시간 및 작업능률을 고려하여 가능한 한 노동집약적(수확 및 선별 등)이고 인체에 유해(제초 및 방제(防除) 등)한 작업에 적용될 수 있도록 한다.

5) 실질적인 노동력 절감을 고려하여 설비를 개발한다.
단위작업을 위한 전용 무인 작업기의 경우, 현장 적용성 측면에서는 감시자로서의 노동력 투하가 필수적이다. 따라서 기술적 난이도를 떠나서 실질적인 노동력 절감 효과가 떨어지는 문제점이 있다. 본 연구과제에서 제안하는 생력화(Automation) 설비는 작업의 일관화 및 연계작업성을 고려하여 노동력 절감을 최대화하고 설비의 효용성을 극대화하는 시스템으로 개발한다.

6) 생력화 설비의 제조 및 수요를 고려하여 대상 작목의 종류에 유연하게 대처할 수 있는 작업 범용성을 갖도록 한다.

7) 유지 보수가 어렵거나, 시설 환경(습도, 온도, 분진 등)에 따른 내구성이 약한 단위장치 또는 부품의 사용을 가급적 지양한다.

8) 고가의 부품 및 정교한 취급이 요구되는 부품은 가능한 한 사용하지 않는다.

9) 시설의 안전성 및 작업자의 작업 안전성을 고려한다.

10) 시설농의 생산기반을 고려한 적정 설계를 수행한다.

11) 국내 시설농가의 생력화(Automation) 설비 및 기계구입에 따른 경제적 여유를 고려한다.

제 2 장 국내외 기술개발 현황

제 1 절 국내외 관련 연구 현황

1. 국내 현황

가. 표 2-1-1는 수박과 참외에 대한 생산작업 체계를 보여주며 작업에 따른 기계화 정도를 보여주고 있다.

<Table 2-1-1> Production Work System of Watermelon and How Mechanized Works are on Going

작목	작업체계 및 작업별 기계화 정도													
수박	파종	→	접목	→	접목 후 관리	→	정식	→	측지정리 및 자만정리	→				
	(*) 인력, 파종기		(-) 인력		(x) 인력		(-) 인력		(x) 인력					
	추비	→	교배	→	유인	→	제초	→	측지정리 및 과실정돈	→	수확	→	선별 및 포장	
	(-) 인력		(x) 인력		(x) 인력		(-) 인력		(x) 인력		(-) 인력		(-) 인력	

* : 기계화가 이루어진 작업, - : 기계화가 요구되는 작업
x : 기계화가 어려운 작업

나. 농업 생산설비와 작업기계의 개발연구 측면에서 국내의 연구실태와 선진외국을 총괄적으로 비교하면, 우리나라의 농작업기계는 극히 수도작 중심으로 개발 실용화되고 있을 뿐이며, 전작(田作)이나 시설생산에 있어서는 제한된 품목의 작업기만이 수입 또는 국산화되어 있는 실정이다. 따라서 전작 및 원예 분야의 생력화(Automation)를 위한 설비개발 및 기계화와 자동화 기술연구가 향후 필요하다.

다. 수확작업의 경우, 농업 기계화 연구소에서 토마토 수확용 로봇의 기초 연구가 이루어지고 있고, 성균관대학교 생명공학부 바이오메카트로닉스 연구실에서 시작기 형태로 매니퓰레이터(manipulator)의 연구개발을 추진하였다. 그러나 작업여건을 고려한 현장 투입 측면에서는 기능을 비롯한 현장 적용성, 작업 다양성 등 여러 가지 측면에서 앞으로 많은 연구가 수행될 필요가 있다.

라. 본 연구팀은 1994년에 실험실 수준의 초음파 센서를 장착하고 PC에 의해 주행이 제어되는 2 륜 차동 구동 전기모터에 의해 구동되는 Mobile 로봇을 개발하였으며, 1997년 2 륜 차동 구동형 소형 상용 Mobile 로봇을 이용한 원격제어 및 관련 영상처리 소프트웨어를 개발했고, 동년에 호접에 의한 과채류 자동접목장치, 건조 표고버섯 전자동 등급판정 및 선별장치를 개발하여 선별기 수준의 영상 처리 기술 및 시스템 설계 기술, 기초 무선 원격 제어 기술을 확보하였다.

마. 국내외적으로 다양한 시설 재배에서의 수확, 재배관리, 수확 동시선별 및 이송, 정밀방제(防除) 및 제초 등의 작업에 단일 설비로서 공통적으로 적용할 수 있는 다목적 생력 작업기 개발 연구는 현재까지 개발되지 않았다.

2. 국외 현황

가. 포도과수원의 규모가 대규모인 프랑스와 미국의 경우에는 작업자가 운전하여 조작하는 전용기 형태의 작업기들이 개발되어 보급되고 있는데 1차 전정(剪定) 기계, 포도의 곁가지 절단 및 줄기유인(유인 후 클립으로 붙들어 맴)용 기계, 포도밭의 좌우와 위에 있는 가지를 절단하는 기계, 포도주 가공용 포도를 훑어 수확하는 포도 수확기 등이 개발되어 실용화되고 있다.

나. 농작업용 로봇의 연구개발에 있어서는, 로봇 시비(施肥)기(Mitsubishi Co. Ltd, 1989; Matsuo, et. al., 1989), 무인 주행트랙터(Ikumot et. al., 1989), 원격제어 잔디깍기 로봇(Kubota Co. Ltd), 무인 콤바인 수확기(Kito, 1986), 로봇 이앙기(Yamashita et.al., 1990) 등의 메카트로닉스 부문 첨단기술을 이용한 연구가 활발히 추진되고 있다.

다. 무인 작업화 혹은 로봇을 이용한 과수 분야의 작업 시스템 개발은 선진국에서도 아직은 실험실 수준 내지 국부(局部)적인 성능시험 정도로 아직은 실용화되고 있지는 않으나, 일본을 중심으로 농업 생산 분야의 특성을 고려한 컴퓨터 시각 시스템과 정보처리기술 등의 개발을 통하여 단위 작업별로 미흡하나마 실용화 연구를 시도하고 있다. 하지만 현장 투입은 여러 가지 문제점으로 향후 10년간 어려울 것으로 판단된다.

라. 과일 수확의 경우 프랑스의 CEMAGRAF에서 사과 수확용 로봇을 실험실 수준으로 개발한 바 있으며, 일본의 Okayama 대학에서 실험실 수준의 포도, 딸기, 오이 등의 수확을 위하여 선단의 End-effector 개발을 중심으로 기존의 산업용 로봇을 이용한 로봇 수확시작기를 개발하였다. 하지만 현장 적용측면에서는 아직 요원한 실정이다.

제 2 절 연구개발 분야의 발전방향 분석

1. 개발 분야의 발전 방향 전망

가. 시설 생산용 생력화(Automation) 설비의 개발 요구는 계속적으로 증대할 것으로 본다. 하지만 국내외적으로 생력화 설비의 연구방향 설정에 지속적인 문제가 있었다. 즉, 단위작업 공정에 전용으로 투입되는 전자동 작업기를 개발하고자 하는 연구방향은 개발 및 실용화 단계 측면에서 매우 비현실적이고 기술적 측면에서, 경제성 측면에서, 그리고 안전성과 작업성 측면에서 대단히 많은 문제점을 안고 있다. 그러므로 본 연구팀은 앞으로의 연구 개발의 발전 방향은 다음과 같이 단계별로 전망한다.

① 수작업
② 간이자동화 장치
③ 반자동화 장치

④ 전용 자동화 장치

⑤ 무인 자동조작과 유인작업의 혼합형태의 범용작업기

- 부분적 작업환경 인식 및 보조형태의 자동 의사 결정 기능

⑥ 무인 자동조작과 자율주행, 유인감시 혼합형태의 범용작업기

- 국부(局部)적 환경 인식 및 국부(局部)적 자동 의사 결정 기능

⑦ 무인 전자동 범용 작업기

- 환경 인식 및 자동 의사결정

현재 시설 재배 생력화(Automation) 설비의 개발 방향은 아래 기술한 내용 중 단계 5에 맞추어 단계 6으로 개발이 추진되어야 함에도 불구하고 국내외적으로 단계 7에 주로 연구 초점을 맞추어, 연구개발의 성과가 미흡하고 기술적 어려움 뿐 아니라 경제적 사회적 측면에서도 연구개발의 문제가 제기되고 있다.

제 3 장 연구개발 수행 내용 및 결과

제 1 절 기초연구

1. 수박의 재배의 생태적 특성과 재배환경

가. 작형(作形)

1) 작형의 분류

수박은 성수기인 6~8월에 가장 많이 소비되며, 그 소비에 맞추어 아래 표와 같이 시기별로 5가지 작형으로 나누어진다.

<Table 3-1-1> Cultivation Type of Watermelon

작 형	파 종 기	정 식 기	수 확 기	성출하기
촉성재배	11하-12상	1중-1하	3하-4하	3중-4중
반촉성재배	12하-1상	2상-2중	4하-5하	4중-6중
조숙재배	2중-3하	3하-5상	6상-7중	5중-7중
억제재배	7하-8상	8하-9상	11상-12하	11중-12중

2) 촉성(促成) 재배

촉성(促成) 재배는 3~4월 출하를 목표로 하는 작형으로 전 생육기간을 시설내에서 재배한다.

3) 반촉성(反促成) 재배

반촉성(反促成) 재배는 5월에 출하를 목표로 하는 작형으로 육묘(育苗)기 생육전반에 가온하여 재배하고, 정식 후부터는 무 가온 재배하는 방법이다.

4) 터널 조숙(早熟) 재배

6월 하순~7월 중순에 출하는 작형으로 2월 중순~3월 하순에 하우스에서 파종(播種) 및 육묘(育苗)하여 3월 하순~5월 상순에 노지(露地)의 터널에 정식한다.

5) 노지(露地) 재배

4월중~5월 하순에 묘상에 육묘(育苗)하여 정식하거나 노지(露地)에 바로 직파하는 가장 일반적인 작형이다.

6) 억제 재배

7~8월에 파종(播種)하여 11월~12월에 수확하는 작형으로 내서성과 내한성을 갖춘 뿌리가 강한 품종이 좋다.

나. 수박의 생육 시기 및 주요 작업

수박의 생육은 발아기, 육묘(育苗)기, 생육기, 착과(着果)기, 과실 비대(肥大)기, 수확기로 나누어진다. 다음 그림은 수박의 재배력을 나타낸다.

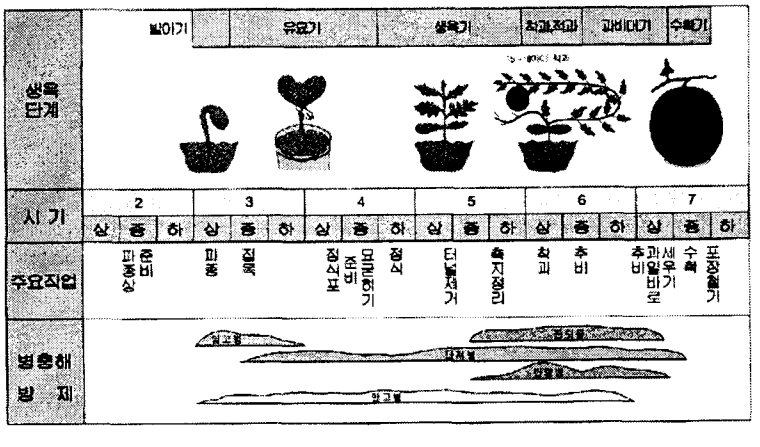

Fig. 3-1-1 Cultivation Calendar of Watermelon

1) 육묘(育苗)기

파종(播種) 후 30~45일이 필요하며 시설 재배의 경우 연작피해가 심각하므로 필히 접목을 통하여 결점을 보완해야 한다.

2) 생육기

생육기는 30일이 소요되며 이시기동안 정식 및 곁가지정리, 추비가 이루어진다. 보통 시설재배의 경우 포기사이의 거리를 30Cm~45Cm로 정식하고 덩굴 수는 2~3개로 하며 1 주당 1개를 착과(着果) 시킨다.
다음 그림은 정식된 묘의 사진이다.

Fig. 3-1-2 Watermelon seedlings planted in the field.

3) 착과(着果)기

착과(着果)기는 15일 정도이며, 이 때 개화(開花), 교배(交配) 및 적과(摘果), 곁가지 정리를 한다. 곁가지의 경우 첫 마디부터 5~6절까지와 착과(着果)절의 아래, 위절에 4~5 절 나오는 것이 보통인데 많이 나올 경우는 저온 육묘(育苗)이다. 덩굴 정리 작업은 정식 후 아들 줄기가 40~50Cm 정도 자랐을 때 원하는 수만큼의 아들 줄기를 남기고 모두 제거하며, 곁가지는 아들 줄기의 각 잎에서 발생하는데 조기에 이를 제거해야 암꽃이 충실해진다.

다음 그림은 곁가지의 사진이다.

Fig. 3-1-3 Side branch of watermelon.

다음 그림은 수박 덩굴의 가꾸는 모양에 대한 그림이다.

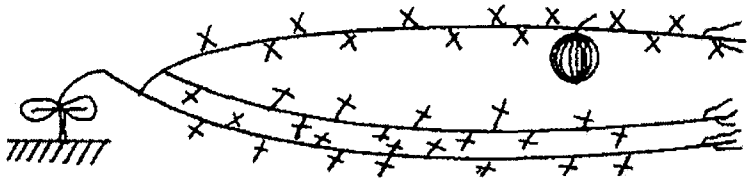

Fig. 3-1-4 Runner shape of watermelon.

4) 과실 비대(肥大)기

과실 비대(肥大)기는 과실 비대(肥大)기와 성숙기로 나눌 수 있는데 비대(肥大)기는 15~20일 정도이고, 성숙기는 20~30일 정도이다. 이 때의 주요 작업은 추비, 곁가지 정리, 과실 정돈 및 관수(灌水)가 있다. 여기서 과실 정돈은 수박 자리 잡기, 돌리기 작업 등을 말한다.

과실 비대(肥大)기의 초기는 교배(交配) 후 35~45일 까지 인데 비대(肥大) 초

기와 중기로 나눌 수 있다. 다음으로 성숙기는 교배(交配) 후 35~45일 이후는 성숙기로 그림 3-1-8과같이 자리부를 마련하여 돌리기 작업과 같은 과실 관리가 필요하다. 다음 그림은 과실 비대(肥大)기 및 성숙기의 과실 사진이다.

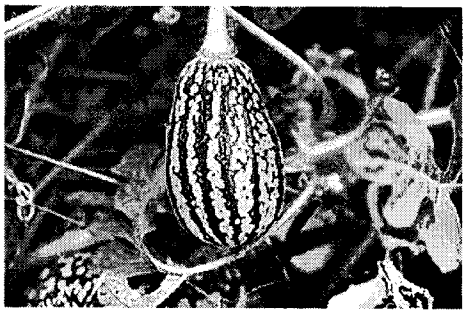

Fig. 3-1-5 Fault flesh (early age).

Fig. 3-1-6 Fault flesh (middle age)

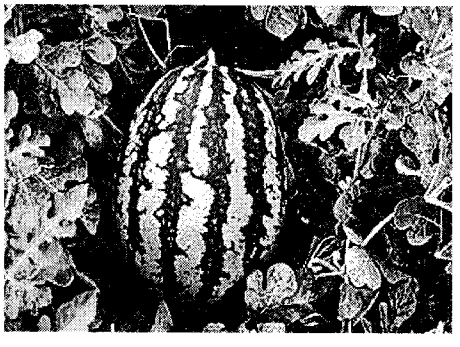

Fig. 3-1-7 Period of maturity.

Fig. 3-1-8 Watermelon sitter.

5) 수확기

가) 수확 시기 결정

수박의 수확시기는 품종, 착과(着果)위치, 토질, 재배시기, 기상조건 등에 따라 차이가 나는데 매일의 평균기온의 합계인 적산온도를 통해서 수확시기를 결정한다. 다음 표는 수확 시기 결정을 위한 품종별 적산온도를 나타내었다.

<Table 3-1-2> Temperature for Deciding a Harvesting Time

소과종 품종	중과종 품종	중대과종품종	만생종 품종
700~800℃	900	1,100~1,200	1,200

위의 표를 바탕으로 수확시기를 착과(着果) 후 일수로 계산하면 촉성(促成), 반촉성(促成) 지배의 경우 40~45일, 조숙(早熟)재배 35~40일 이다. 보통 수확은 이 일수에서 3~4일 전에 행하며, 하루 중 과실의 온도가 올라가지 않은 시간인 오전 9~10경에 수확한다.

나) 수확 적기 판단 기준

과실의 성숙이란 내부적으로는 착색이 양호하고, 종자 역시 고유의 색깔을 띠며, 당도가 상승하고, 씹는 감이 좋아져서 식미가 우수해지는 것을 말한다. 외관적으로는 품종 고유의 과피(果皮)색을 나타내며 상품성이 증대되는 것을 말한다. 과실의 성숙은 착과(着果) 후의 일수, 적산온도, 그리고 외관을 보고 판정하는데 관행적으로 행해져오는 숙도 판정법은 다음과 같다.

① 과형(果形) : 발육이 나쁜 과실은 판정하기 어렵지만 발육이 충실한 과실이 성숙하면 어깨 부분이 펴지고 과경(果莖) 주변에 결이 생긴다.

② 소리 : 과피(果皮)의 두께, 경도, 공동의 유무, 과실의 대소에 따라 다르지만 대체로 미숙한 것은 금속음이 나고 성숙한 것은 탁음이 난다. 지나치게 낮은 탁음이 나는 경우에는 육질 악변과일 가능성이 많으므로 주의해야 한다.

③ 과피(果皮)의 색 : 과피(果皮)색은 햇빛, 수분, 질소의 효과 등에 따라 다르지만 대체로 과실이 성숙함에 따라 과피(果皮)에 윤기가 나며 호피무늬가 진해지고 선명해진다. 또 과경(果莖)부의 털이 없어지고 성숙한 색으로 된다.

④ 탄력 : 품종, 과실, 크기, 착과(着果)절위, 재배시기 등에 따라 차이가 있으나 성숙된 것은 과경부(果莖部)와 화흔부(花痕部)를 눌러보면 탄력이 있고 과피(果皮)도 누르면 탄력이 있다.

2. 품종별 특성

가. 수박의 형태학적 분류 방법

형태학적인 분류방법은 엽색(葉色), 과피(果皮) 줄무늬, 과중(果重), 과육(果肉)색, 당도(糖度)로 구분된다.

나. 작형별 주요 등록 품종

1) 촉성(促成) 및 반촉성(促成) 재배

대감 수박(홍농 종묘), 빛나 수박, 일출 수박(중앙 종묘), 환호성 수박(서울), 내고향 수박, 맛 수박(한농), 금천 수박(농우), 한들 수박(농우), 기찬 수박(농진), 꿀단지 수박(청원), 단샘 수박(고려)

2) 노지(露地) 재배

오림피아 수박(홍농), 무지개 수박(한농), 왕장수박(농우), 빅토리아수박(농진)

3) 터널 조숙(早熟) 재배

삼복꿀수박, 감로수박(홍농), 대상수박(중앙), 단비수박(중앙), 백두산수박(서울), 달고나 수박(서울), 참다라 수박(한농), 달 수박(농우), 강남수박(농진), 달덩이수박(청원), 금강산수박(제일)

4) 소과종

귀공자 수박(홍농). 옥동자 수박(중앙), 복 수박(서울), 은초롱 수박(한농), 엄지수박(한농), 조생단 수박(농우), 홍단수박(농진)

다. 주요 품종별 특성

<Table 3-1-3> Characteristics about Main Species

품 종	특 성	비 고
달 고 나	- 저온기 착과(着果) 비대(肥大) 양호함. - 공동과 및 기형과 발생이 적고 수송성 및 저장성 강함.	서울 종묘
복 수 박	- 소과종으로 과피(果皮)가 얇고 당도가 높음. - 세력이 강해 착과(着果)가 어려움. - 1주당 3 ~ 4개 정도 착과(着果)시킴.	서울 종묘
갈 채	- 과형(果形)은 높은 구형이고 저온 하에서도 비대(肥大)성이 좋음. - 저온 하에서도 꽃가루 발생이 좋고 꽃가루양도 많음. - 완숙이 되어도 꼭지 함몰이 적어 상품성이 우수함. - 육질은 섬유질이 적고, 과피(果皮)경도가 강하여 열과 발생이 적음.	홍농 종묘
아폴로수박	- 꼭지 함몰이 적은 타원형 중과종 - 호피무늬 선명하고, 과육(果肉)색은 홍색이며 육질이 치밀하고 당도 높다	농우 종묘
삼복꿀수박	- 중과종으로 과형(果形)은 원형과 타원형. - 착과(着果) 후 배꼽이 작아 열과 발생이 거의 없음. - 과피(果皮)는 얇고 수송성이 강하고 당도 축적 및 숙기가 빠름.	홍농 종묘
대감수박	- 육질이 치밀하고 당도가 높으며 기형과 및 공동과 발생이 비교적 적고 수송성이 강함. - 저온기 재배 시 꽃가루 발생이 잘 안되어 수분수로 금성 수박이 좋고, 대목은 강력 참박이 좋음.	
온누리 수박	- 저온기 착과(着果) 양호, 재배관리에 따라 대과 5 ~10kg	
금배수박	- 높은 구형으로 과육은 홍색 육질 연하고 당도 높음. - 과피(果皮)는 얇으나 열과는 잘 안 생김. 초세 좋고 내병성 강함.	농우 종묘

3. 시설 재배 환경 분석

가. 시설 하우스의 형태

처음에 하우스의 형태는 벽이 없는 지붕형태와 터널형에서 발전하여 현재에 들어 차츰 대형화되었다.

1) 대형 터널 하우스(반원형 하우스)

우리나라 시설원예의 초기형태로 모양은 그림 3-1-9의 (a)와 같이 반원형이며 주 골조자재는 대나무이다. 하우스의 규격은 보통 너비가 4.2~5.4m 정도이며, 높이는 1.2~2.0m 정도로 낮으나 길이는 50~100m 로 길며, 소규모 단동형태를 갖는다.

2) 지붕형 하우스

그림 3-1-9의 (b)의 형태로 목재 골조자재로 소형 단동과 중형 단동형으로 시설되었으나 골조의 차광율이 높고, 내구성이 낮아 골조자재를 철재로 개량하고, 외 피복 자재를 연질 피복 자재에서 경질판으로 개량되어 사용되고 있다. 이 하우스는 화훼 재배에 많이 이용된다.

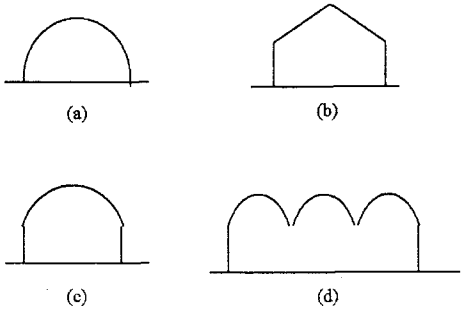

Fig. 3-1-9 Form of House

3) 아치(Arch)형 하우스

그림 3-1-9의 (c)와 (d)의 형태와 같은 아치(Arch)형 하우스의 골조자재는 철재 파이프가 사용되며, 크기에 따라 구분되며, 단동(a) 또는 연동(b)으로 시설된다. 이 형태의 시설이 현재 계속해서 증가되고 있다.

나. 농가 보급형 자동화 하우스 표준 설계 규격

농촌 진흥청에서 2001년 발표한 농가 보급형 자동화 하우스 표준 설계서에 의하면 1-2W 각관 A형, 1-2W 각관 B형, 1-2W 서까래 보강형, 1-2W 기본형과 같이 4개가 있다.

주요 규격을 보면 폭 7m~7.5m, 측고 3.0m, 동고 4.8m, 길이 50m, 연동수 3연동으로 아래 그림은 1-2W 각관 A형의 설계 도면이다.

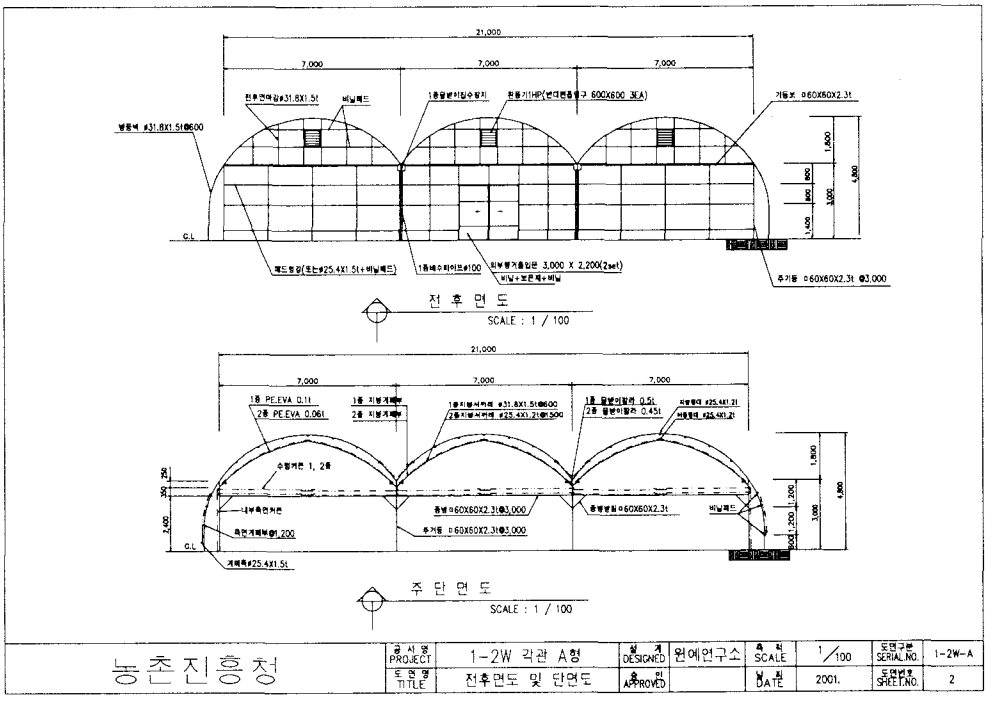

Fig. 3-1-10 1-2W At Angle Pipe A Type Conservatory Design

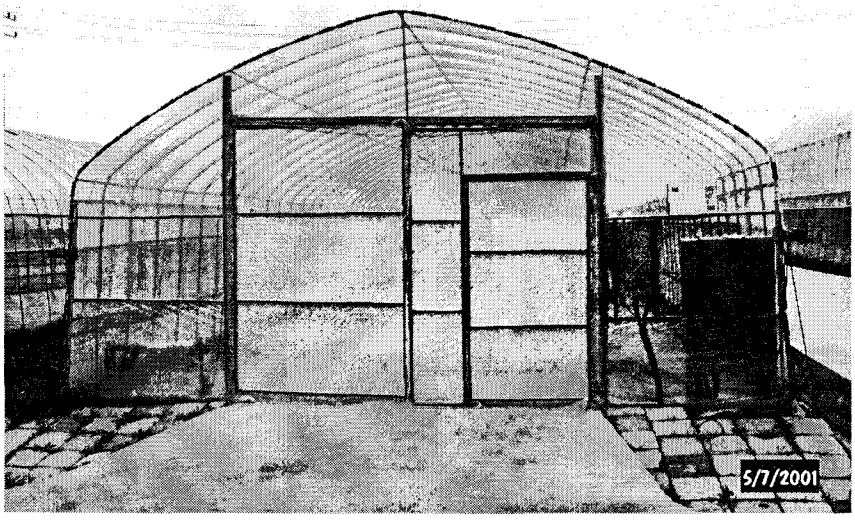

Fig. 3-1-11 Experimental vinyl house.

다. 시설 내부 형태 분석

수박의 이랑은 중앙부를 높게 하고 경사지게 만들어 이랑위에 물이 고이지 않게 하여야 하며, 이랑의 높이는 가급적 높게 만드는 것이 좋다. 배수로도 중앙부는 높게 양쪽 가장자리는 낮게 만들어 골에 물이 고이지 않도록 하여야 한다.

이랑의 너비는 270~300cm 정도로 하는 것이 일반적인데 지역에 따라 이보다 약간 좁게 만드는 경우도 있다. 또 이랑의 경우 검은색 필름 멀칭(mulching)을 하는 것이 보통인데 보온 및 보습효과가 있다. 그리고 멀칭(mulching)은 지온을 높여서 뿌리 자람을 좋게 하고, 토양 수분을 보존하여 관수(灌水)를 적게 할 수 있고, 하우스 표면에 물방울을 덜 맺게 하는 효과도 있다.

다음으로 고랑의 경우 45~55cm 정도의 너비로 조성한다. 그림 3-1-12는 실험용 온실의 내부 사진이다. 여기서 이랑과 고랑의 일반적인 형태를 볼 수 있다.

Fig. 3-1-12 Interior of conservatory.

제 2 절 모듈(module)형 작업 시스템 개발

1. 서론

본 연구에서는 시설 내 수박의 종합적인 재배관리를 위하여 관수(灌水), 방제(防除), 가지자르기, 돌리기, 수확용 그리퍼(Gripper)를 모듈(module) 형태로 개발하여 주 시스템에 쉽게 장 탈착할 수 있게 하였다. 종합 재배 관리는 하나의 단위 장치를 통해서 이루어 질 수 없으므로 종래에는 개별적인 단위 장치에 의존하였다. 시스템 최적화 설계를 연구를 통해 이런 각 단위 장치들 간의 공통적인 부분을 정리하였으며, 그 결과 모듈(module)형으로 장, 탈착할 수 있는 구조를 선정하게 되었다.

먼저 관수(灌水), 방제(防除) 작업의 경우 시설 내 관수(灌水) 및 방제(防除) 설비를 통해 작업이 이루어 졌으나 이 설비는 시설 내 환경 변화에 따라 쉽게 변경하기 어려우며, 부분별 관수(灌水) 및 방제(防除) 작업을 할 수가 없어 최적화된 작업 수행이 어렵다. 그리고 환경적 및 경제적인 측면에서도 농약과 같은 화학 약품들을 적기 적소에 살포할 수 없으므로 많은 문제점을 내포하고 있다. 본 연구에서는 이러한 관수(灌水) 및 방제(防除) 설비를 종합 재배관리 시스템에 도입하여 상기의 문제점을 해결하는 방법을 모색하였다. 그 결과 개발된 시스템의 정밀 매거진을 통해 정밀 관수(灌水) 및 방제(防除)를 할 수 있는 하드웨어 및 소프트웨어 환경을 개발할 수 있었다.

다음으로, 가지자르기 및 돌리기, 수확작업의 경우 다량의 노동투입이 되는 재배 과정 중 하나이다. 현재 일본 교토 대학에서 자동화된 수확 작업 시스템이 연구되었으나, 가지 지르기 및 돌리기와 같이 여러 작업을 할 수 있는 시스템은 존재하지 않는다. 다시 말하면 현재 이러한 작업은 순수 인력을 통해 이루어지고 있다. 그러므로 본 연구에서는 정밀 매거진을 장착한 재배 관리 시스템을 통해 이러한 작업을 할 수 있는 시스템을 개발하였다.

2. 가지자르기 그리퍼(Gripper) 개발

가. 작업 환경

가지자르기 작업의 경우 정식 초기에부터 착과(着果) 후 영양상태에 따라 지속적으로 이루어지는 작업으로 그 작업 시기와 위치가 과중에 많은 영향을 미치며, 그 노동 투하량도 크다. 여기서 가지자르기 작업은 아들 줄기의 정리 작업과 곁가지 정리 작업을 뜻한다.

본 연구에서는 작업자의 1차 교시(敎示) 및 국부(局部) 영상처리를 통하여 곁가지의 위치 좌표와 자세 정보를 추출하고 정밀매거진에 장착된 로봇 팔 및 자르기 그리퍼(Gripper)를 통하여 곁가지 제거 작업을 수행하게 하였는데, 곁가지 제거 작업은 가급적 갈라진 마디에서 절단해 주어야 하며, 조기에 제거를 하여야 한다.

곁가지 굵기는 자람에 따라 많이 다른데 본 연구에서는 조기에 제거함을 목적으로 4㎜~8㎜를 기준으로 하였다.

나. 그리퍼(Gripper) 구조

가지자르기 그리퍼(Gripper)는 다음과 같이 구성하여 설계 제작 하였다.

- 절단 칼날
- 공압식(空壓式) 칼날 구동부
- 로봇 선단 연결부
- 센서 및 공압 솔레노이드

먼저 절단 칼날은 그림과 같이 상용품의 절단 가위 날을 사용하였으며, 절단 가위의 칼날 부분만 분리하여 공압 그리퍼(Gripper)와 링크 기구로 연결하였다.

절단 칼날의 구동부인 공압 그리퍼의 행정(Stroke)은 10.0㎜이고, 링크 기구를 통하여 절단 칼날 끝단부의 행정을 49.44㎜가 되게 설계하여 가지를 충분하게 자를 수 있게 하였다.

칼날 구동부는 그림과 같이 공압 그리퍼 및 연결 링크로 구성하였다. 그리고 로봇 선단 연결부는 로봇 선단의 높이가 수박이 있을 경우에 맞추어 설계되었고, 가지자

르기는 이랑 표면 근처에서 이루어지므로 이것을 고려하여 높이 보정을 하기위하여 설계, 제작 하였다.

다음으로 칼날의 동작 여부를 알기 위해 근접센서를 사용하여 그리퍼(Gripper)의 동작을 모니터 할 수 있게 하였으며, 공압 배선 및 센서 배선을 롤(Roll)형으로 하여 Z 축 및 회전축의 움직임에 적응 할 수 있도록 하였다.

그리고 칼날은 사용 빈도에 따라 무디어 질 수 있으므로 손쉽게 교환할 수 있게 하였다.

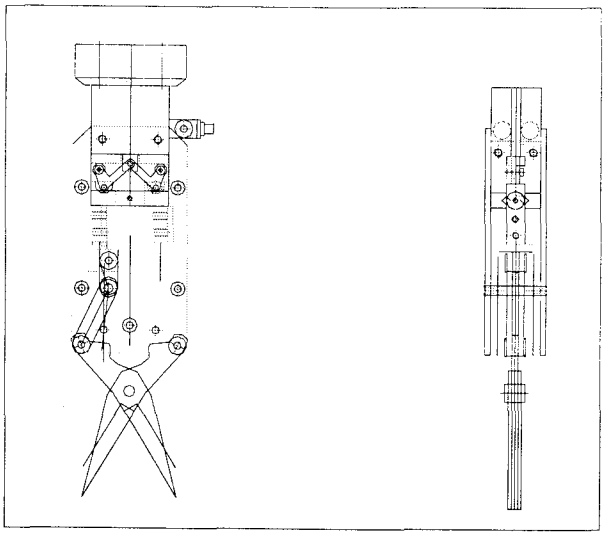

Fig. 3-2-1 Air pressure gripper for pruning.

다. 링크 설계

설계된 그리퍼(Gripper)의 절단날은 상용 절단 가위의 칼날부를 분리하여 링크 기구를 통하여 공압 그리퍼(Gripper)에 연결된다.

먼저 링크 기구 설계를 위해 절단날의 운동 궤적(軌迹)을 알아야 하는데 가지 절단을 충분하게 하기 위하여 아래 그림과 같이 칼날 끝의 행정(Stroke)을 결정하였다. 절단날은 그림에서와 같이 교차되었을 때 교차 한계를 갖는 구조로 되어있어 완전하게 교차되지 않게 되어있다. 이것은 칼날 끝이 교차로인하여 무디어지지 않게 하기 위해서이다.

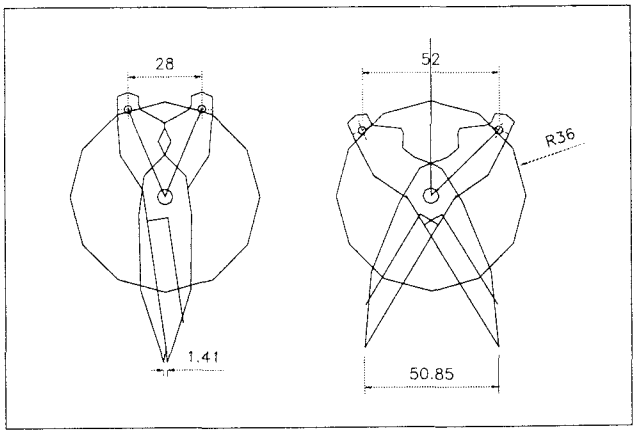

Fig. 3-2-2 Trajectory and stroke of the blade of a Cutter.

절단날은 그림과 같이 끝 부분의 행정(Stroke)이 49.44㎜ 그 반대쪽은 24.0㎜이다. 그리고 운동 궤적(軌迹)은 절단날 중심축을 기준으로 반경이 36㎜의 원호를 따라 이동하게 되는데, 절단날을 폈을 때와 교차시켰을 때 8.27㎜만큼 수직 길이가 길어진다. 그러므로 본 개발에서는 링크 기구를 그림과 같이 공압 그리퍼(Gripper)의 10㎜ 행정(Stroke)을 24㎜의 행정(Stroke)으로 바꾸는 구조를 갖게 하였다.

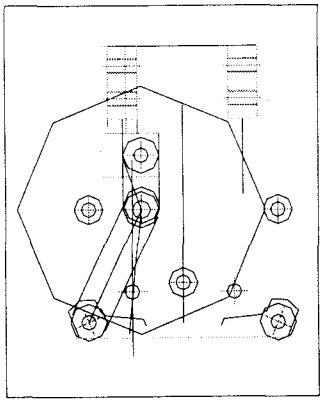

Fig. 3-2-3 Structure of link.

라. 결과 및 고찰

곁가지 자르기 그리퍼(Gripper)를 실제 곁가지 및 수박의 꼭지에 대하여 절단실험을 수행하였다. 곁가지 및 꼭지의 절단 작업은 매우 잘 이루어졌으며, 절단날이 스텐레스 재질이여서 줄기의 수액으로부터 산화 되지 않았으나, 많은 작업을 할 경우 수액이 건조하여 절단을 하기위한 힘이 증가하였다. 꼭지의 절단의 경우 수액(水液)이 많지 않아 그런 경우가 적게 나타났으나 어린 곁가지의 경우 작업량이 수확에 비하여 상대적으로 많기 때문에 일정 작업을 수행한 후 수액(水液)에 대한 절단날의 세척이 필요하다고 사료된다.

3. 돌리기 그리퍼(Gripper) 개발

가. 작업 환경

수박 돌리기 작업은 수확 10~15일 전에 지면에 부착된 과피(果皮)가 황화되므로 착색을 위해 수박을 눕힌 상태에서 돌리기 작업을 한다. 한번에 돌리는 량은 약 100° 정도로 꼭지부에 무리가 가하지 않는 범위에서 수행하여야 한다. 복수박의 경우 열매 싸기와 병행하여 돌리기를 수행하는데 이 때는 고른 당도(糖度)를 갖게 하기 위함으로 수정 후 10~15일 경과 후 작업한다.

수박 돌리기 작업은 과실의 중(中) 비대(肥大)기에서 후(後) 비대(肥大)기에 이루어지므로 과중은 보통 출하 시기의 70%~90%정도이다. 그리고 수박 자리를 마련한 경우와 없이 재배하는 경우가 있는데 개발한 그리퍼(Gripper)는 수박의 자리이동이 없게 설계하였으며, 과중이 6kg인 수박을 기준으로 설계하였다.

나. 1차 돌리기 그리퍼(Gripper) 시작기

1) 구조

1차 돌리기 작업용 그리퍼(Gripper) 시작기의 구조는 견인용 진공 패드 기구, 돌리기용 진공 패드 및 모터, 링크 구조, 로봇 선단(先端) 연결부, 센서 및 진공 솔레노

이드로 그림과 같은 구조로 구성하였다. 먼저 견인용 진공 패드 기구는 수박 외의 줄기 및 잎의 손상을 최소화하기 위하여 사용하며, 진공 패드와 상하 이동을 위한 공압 실린더로 구성하였다.

다음으로, 돌리기용 진공 패드는 수평방향 돌리기 회전축을 결정짓는 기구로 수박의 양쪽을 고정한다. 그리고 돌리기용 진공 패드 중 하나의 축을 모터에 연결하여 돌리기 작업을 수행하게 하였다. 로봇 선단 연결부는 4개의 나사로 고정할 수 있게 하였으며, 공압, 진공, 센서의 배선은 나선형으로 하여 축의 움직임에 따른 배선의 스트레스를 최소화하였다.

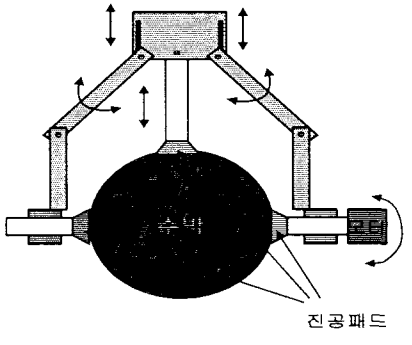

Fig. 3-2-4 Gripper to turn watermelon.

2) 돌리기 작업

수박의 하부(대지 접지면)가 위로 올라오도록 꼭지를 중심축으로 회전시켜야하며, 꼭지 및 주변의 줄기 및 잎에 영향을 주지 않아야 하고, 과피(果皮)의 손상이 없어야 한다. 회전방법으로는 먼저 꼭지 및 주변줄기, 잎에 영향을 주지 않게 하기 위하여 과실을 지상에서 일정높이 까지 견인하여 그리퍼(Gripper)로부터 주변을 분리하여야 한다. 과피(果皮)에 손상을 최대한 억제하고자 진공 패드를 이용하여 견인하는 방법을 선택하였으며, 견인 높이는 줄기 손상을 고려하여 150mm로 하였다.

돌리기 작업의 순서는 먼저 견인작업을 하기 위하여 견인 진공 패드와 그리퍼(Gripper) 뭉치가 수박에 접근하고(a), 견인(b)을 한 후 돌리기 작업을 위해 꼭지를 축으로 돌리기용 진공 패드 및 그리퍼(Gripper)를 이용하여 수박을 잡고(c), 돌리기 진공 패드를 축으로 공압 모터를 이용하여 수박을 돌리는(d) 순서로 작업이 이루어지게 하였다.

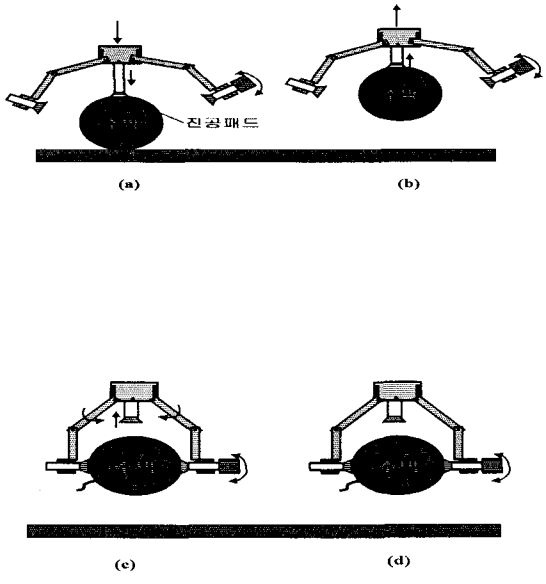

Fig. 3-2-5 Order of watermelon turning.

3) 진공 흡착 패드 기구

진공흡착패드는 장기간 사용 후에도 기밀(氣密)이 유지될 수 있어야 하며, 기밀이 샐 경우 이것을 감지하여 소정의 조치를 할 수 있어야 된다. 그리고 패드가 수박에 접근할 때 패드 기구로 인한 과피(果皮)의 손상이 없어야 한다. 이러한

조건에 만족하게 하기 위하여 진공 패드는 불소(Fluorine)재질과 직경 Ψ125인 것을 선정하였으며, 견인용에는 평형을 회전축에는 평형립 부착을 선정하였다. 그리고 패드 부착기구는 과피(果皮) 손상을 줄이고자 Z축(수직축) 방향으로 스프링을 이용한 완충기구가 있는 모델을 선정하였다. 견인용 진공 패드 기구의 모델은 단해공압의 ZPX125HF-B01-B12를 선정하였으며, 회전축에는 동사의 ZPT50CFK10을 선정하였다.

4) 결과 및 고찰

수박 돌리기 그리퍼(Gripper)의 1차 시작기의 성능을 시험하기 위하여 그리퍼(Gripper) 단독 실험 및 다목적 작업기에서의 실험을 행하였다. 단독 실험의 경우 돌리기 작업은 원활하게 수행 되었으나 기구가 복잡하고, 기구의 중량이 커 조작하기가 용이하지 않았다.

다목적 작업기에서의 실험에서는 기구의 크기로 인하여 카메라 뷰(View) 영역의 상당부분을 가려 작업구간의 특정영역에서만 작업을 수행할 수 있었다. 그리고 돌리기 그리퍼(Gripper)의 중량과 과중으로 인하여 로봇 암의 움직임이 빠를 경우 진동과 로봇에 기계적인 무리를 가할 수 있었다.

그러므로 본 연구에서는 모듈(module)형 작업 시스템의 규모는 정밀매거진의 정적 및 동적인 부하 이하를 가져야 하며, 영상처리를 통하여 작업이 수행되므로 작업 구간이 1차 시작기보다 커서 모듈(module)형 작업기가 카메라 뷰 영역을 가리지 않게 하여야 한다는 결론을 내렸다. 즉 작업 구간의 크기를 확대하고, 모듈(module)형 작업 시스템의 규모는 가급적 간단하게 구성해야 한다.

다. 2차 돌리기 그리퍼(Gripper) 시작기

1) 서론

1차 돌리기 그리퍼(Gripper) 시작기에서 기구의 복잡성과 크기 및 무게의 요인으

로 실제 시스템에 장착했을 때 동작상의 문제가 발생하였다. 그러므로 2차 시작기는 다음과 같은 기준으로 설계를 하였다.

가) 작고, 단순할 것

나) 구동원이 단순할 것.

다) 기계적인 기능 일부를 소프트웨어로 전환할 것

위의 설계 기준을 바탕으로 단일 진공 패드를 이용한 그리퍼(Gripper)를 설계했으며, 작업 프로세싱을 개선함으로 돌리기 작업을 위한 메커니즘을 최소화하였다.

2) 구조

돌리기 그리퍼(Gripper) 2차 시작기는 그림과 같이 진공 패드, 볼 조인트(Joint) 결합기구, 스프링 완충장치, 진공 기구, 센서로 구성하였다. 먼저 진공 패드는 직경이 110㎜이며, 벨로우즈(Bellows)의 행정(Stroke)은 22㎜의 고무 재질이며, 볼 조인트는 수박의 중심축으로부터 옵셋(Offset)된 지점을 견인할 때 수박 표면과 진공 패드가 올바른 접촉을 할 수 있도록 하기 위하여 전 방향으로 30° 휠 수 있으며, 회전도 할 수 있게 하였으며, 스프링 완충장치는 진공 패드가 수박 표면을 일정 압력 이하로 누를 수 있게 하기 위하여 설정하였다.

2차 돌리기 그리퍼(Gripper)의 연결 하우징(Housing)의 내부에 근접센서를 장착하여 진공 패드가 일정 깊이 이상 수박을 누르는 것을 감지함으로 수박과 그리퍼(Gripper)가 접촉되었음을 제어 시스템이 알 수 있게 하였다.

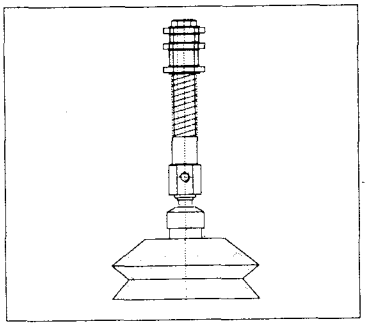

Fig. 3-2-6 Vacuum pad.

3) 돌리기 작업

1차 시작기의 경우 꼭지의 위치가 측면 진공 패드와 간섭이 일어날 가능성이 있으므로 2차 시작기에서는 그림과 같이 소프트웨어적인 반복 견인을 통하여 돌리기 작업을 수행하게 하였다.

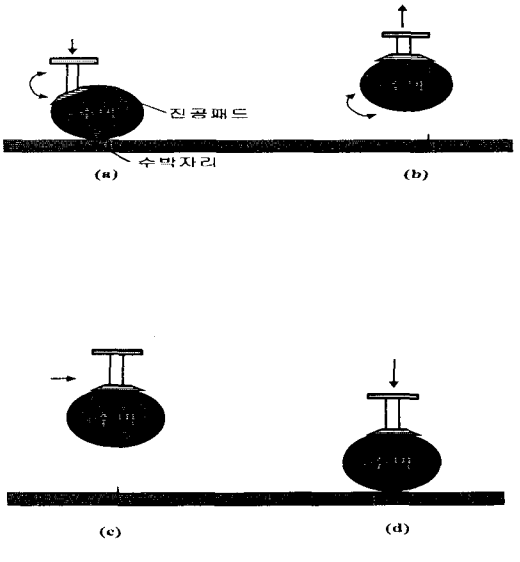

Fig. 3-2-7 Order of watermelon turning

작업 순서는 다음과 같다.

가) 작업자의 작업물 및 꼭지부 교시(敎示)
나) 영상처리를 통한 작업물의 형상 검출 (크기, 자세, 중심)
다) 작업물의 중심에서 일정량 편심된 지점을 흡착패드로 견인
라) 편심량 만큼 중심점 보상 후 놓기
마) 지정된 각도에 이를 때까지 다), 라)를 반복 수행

4) 견인(牽引) 실험

수박의 견인실험을 하기 위한 공시재료는 대과종 3개, 중과종 3개를 이용하였다. 중량은 3kg~8kg이였으며, 형태는 원형, 타원형 이였다.
실험 방법은 그림과 같이 꼭지와 배꼽을 축으로 하여 수직축과 수평축을 기준으로 0°, 15°, 30° 에 대하여 견인 실험을 수행하였다.

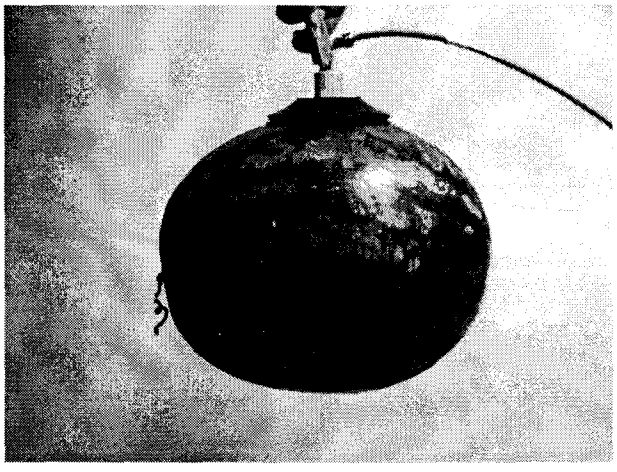

Fig. 3-2-8 Vertical traction (large size).

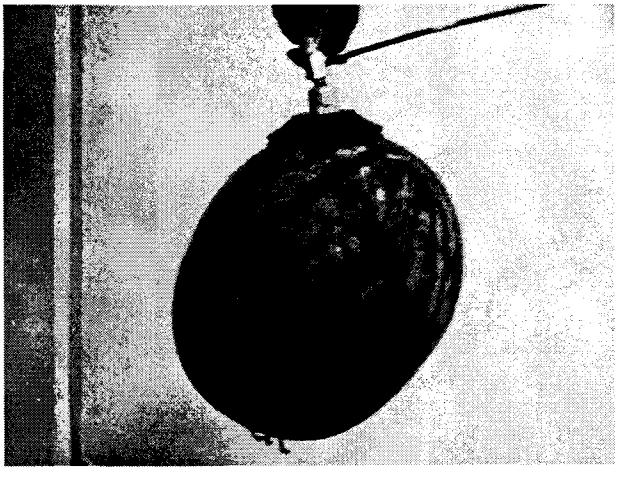

Fig. 3-2-9 Off-axis traction (large size).

실험 결과 견인이 완벽하게 수행되었으나, 진공배선이 길 경우 진공도가 떨어짐을 통하여 견인이 되지 않았으며, 30°가 넘으면 진공 패드가 수박 표면에 완전하게 접촉되지 않아 견인이 잘되지 않았다.

라. 결과 및 고찰

2차 시작기의 수박 돌리기의 경우 3번의 돌리기 과정을 반복함으로 돌리기 작업을 종료할 수 있었다. 한 번의 돌리기 작업에 평균 10초 정도의 시간이 소요되었으나, 로봇 암의 가감속의 최적화 및 수박 꼭지의 강도를 조사함을 통하여 보다 최적화 될 수 있을 것이다.

진공 패드를 이용한 견인 실험에서 소형 인젝션 형의 진공 발생기만으로도 충분히 견인을 할 수 있었으며, 수확의 경우 중량물을 보다 신속하게 이송하기 위해서는 진공 패드의 형태 및 크기에 따른 실험 및 진공발생기의 용량 최적화 선정이 수반되어야 될 것이다.

4. 수확용 그리퍼(Gripper) 개발

가. 기초자료

수박의 수확은 착과(着果) 후 보통 35~45일 후에 작업하는데, 수확 시기는 품종, 재배시기, 재배조건에 따라 변화가 크므로 보통 착과(着果)날짜를 표시하여 일정한 일수가 지나면 수확하는 방법을 일반적으로 사용한다. 또 다른 방법은 수박의 성숙에 필요한 적산(積算)온도를 계산함을 통하여 수확시기를 판단하는데 보통 800~1200°C 정도에 수확하지만, 소과종과 대과종, 조생종, 만생종, 크기, 기후조건 등에 따라 차이가 많이 난다. 그러므로 수확시기를 결정하는 가장 좋은 방법은 착과(着果)일을 표시하고 성숙 일수가 지났을 때 표본채취를 통해서 수확 작업을 수행하는 것이다.

김(1997)의 경우 반발특성을 이용하여 숙도를 판정하는 센서를 개발하였으며, 이(1998)의 경우 초음파를 이용하여 수박의 내부 결함 판정에 대한 연구를 하였다. 그리고 최(2002)의 경우 타공법을 이용하여 수박의 숙도를 판정하는 연구를 하였다

나. 1차 시작기

1) 서론

수박을 수확하기 위해서는 꼭지를 일정길이 간격으로 절단을 해야 한다. 그러나 꼭지의 상하 위치가 일정치 않고, 과경(果徑) 및 과형(果形)이 일정하지 않으므로 이러한 것에 적응을 하여 절단 작업을 수행해야 한다. 다음으로 절단된 수박을 수확물 적재장소로 옮기기 위한 기구가 필요하다. 이 때 수확물의 중량이 매우 크므로 신속하게 수확물을 옮기는 기술이 필요하다.

1차 시작기에서는 꼭지의 절단 방법을 칼날 교차식으로 하고 이송 기구는 진공패드를 이용하였다. 진공 패드를 통한 이송 방법은 수박 돌리기 그리퍼(Gripper) 개발 시 견인 성능과 이동성능을 측정한 바 진공발생기의 성능 및 수확물 이송 방법에 따라 달라지나 매우 강인한 흡착력을 갖고 있음을 알았다.

2) 구성

수확용 그리퍼(Gripper)의 구성은 크게 꼭지 절단부와 견인부로 나누었다. 먼저 꼭지 절단부의 경우 절단칼날, 칼날 가이드, 구동부로 세부적으로 나누었고, 견인부는 진공 패드, 진공 발생장치로 나누었다.

꼭지 절단부의 절단칼날은 수박의 꼭지가 있는 전 범위를 자를 수 있도록 200 ㎜의 스테인리스(Stainless) 강으로 제작하였으며, 그 끝단부에 전방향의 구름 롤러를 장착하여 수박표면을 따라 칼날을 유도할 수 있게 하였다. 다음으로 칼날 가이드의 경우 칼날이 수박표면을 따라 유도될 수 있도록 하는 기능을 갖는데 스프링 및 자중을 이용한 장력 장치로 이를 구현하게 하였다.

1차 시작기의 꼭지를 절단하는 방법은 공압실린더를 이용한 칼날 교차 방법으로 꼭지의 위치가 지면에 수직축을 기준으로 다양하게 분포되기 때문에 넓은 범위에서 절단이 이루어 질수 있도록 설계 제작하였다. 다음으로 견인부의 경우 수박돌리기 그리퍼(Gripper) 2차 시작기에 사용된 동일한 진공 패드를 사용하였으며, 다만 목부분이 고정된 형태를 갖게 설계하였다.

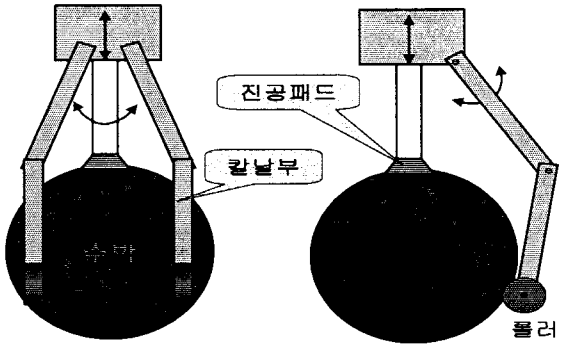

Fig. 3-2-10 Stem cutting gripper.

3) 동작 방법

수확용 그리퍼(Gripper) 1차 시작기의 동작방법은 다음과 같다.

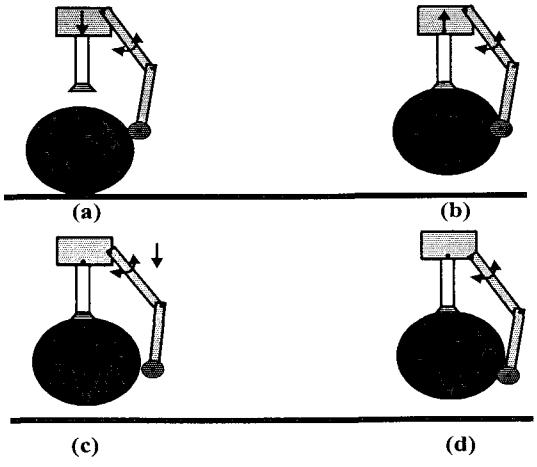

Fig. 3-2-11 Order of stem cutting.

가) 작업자의 1차 수확물 교시(敎示) 및 꼭지부 방향 교시(敎示)
나) 국부(局部) 영상처리를 통한 외형 및 좌표 인식
다) 로봇 암(Arm) 작업위치로 이동
라) Z축의 이동 및 견인 (a), (b)
마) 절단날을 통한 꼭지 절단(c)
바) 적재부로 이송 (d)

4) 결과 및 고찰

1차 시작기의 수확용 그리퍼(Gripper)의 성능은 꼭지부의 절단이 가장 중요하였다. 수확물 이송 작업의 경우 진공 패드의 성능 및 이송 방법에 따라 다르지만 견인 성능 실험을 통해 매우 양호한 결과를 얻을 수 있었다. 그러나 꼭지의 절

단의 경우 꼭지의 위치가 일정하지 않기 때문에 수박 표면을 따라 수직축으로 넓은 범위를 절단해야 한다.

본 시작기에서는 200㎜ 스테인리스 절단날을 이용하여 꼭지 절단작업을 수행하였는데, 절단날 가이드 및 칼날의 정밀가공이 어려워 절단작업이 잘 이루어지지 않았다. 절단날이 정교하게 교차하여야 절단이 되는데 날의 길이가 길기 때문에 잘 교차되지 않았고, 절단을 위한 공압 실린더의 힘도 부족하였다.

다. 2차 시작기

1) 서론

1차 시작기의 실험 결과 칼날의 구조와 칼날 교차 방법에 대한 새로운 방향의 연구를 수행한 결과 가지자르기용 그리퍼(Gripper)의 칼날과 공압그리퍼(Gripper)를 사용한 2차 시작기를 개발하였다.

먼저, 넓은 범위에서 절단이 이루어지게 하기 위하여 수박의 표면에 칼날이 유도되는 시점부터 칼날은 연속적으로 유도가 종료되는 시점까지 동작하게 함으로 작은 칼날을 이용하고도 넓은 범위에서 절단이 이루어 질 수 있도록 하였다. 그리고 절단 날 가이드도 단순히 자중을 이용하여 절단날을 수박 표면에 유도될 수 있도록 하여 그 기구도 간략화 시켰다.

2) 구성

수확용 그리퍼(Gripper) 2차 시작기의 구조는 다음과 같이 구성하였다.

가) 소형 절단 기구 및 공압 그리퍼(Gripper)
나) 절단 기구 가이드
다) 수확물 이송용 진공 패드 기구
라) 진공 발생 장치 및 공압 솔레노이드, 센서

먼저, 소형 절단 기구는 가지자르기 그리퍼(Gripper)에서 로봇 선단부와 연결되는 연결봉을 제외한 칼날부와 링크 기구, 공압 그리퍼(Gripper)를 이용하였으며, 절단 기구 가이드는 경량의 알루미늄 재질을 이용하여 소형 절단 기구를 수박 표면을 따라 유도되게 하였는데, 1차 시작기와 달리 과종에 따라 곡률이 다르게 설계된 가이드를 통하여 절단기구 가이드를 최적화 하였다. 다음으로 수박 표면을 따라 일정 압력으로 유도되게 하여 절단기구가 원활하게 작업을 수행하게 하기 위한 압력기구를 가이드 및 절단기구의 무게를 이용하여 단순화 시켰다.

다음으로 수확물 이송용 진공 패드 기구는 1차 시작기와 동일게 구성하였다.
그림 3-2-12 는 2차 시작기의 개략도이고, 그림 은 2차 시작기의 제작 도면이다. 여기서 기준 수박의 형태는 구형이며, 직경은 340㎜로 하였다.

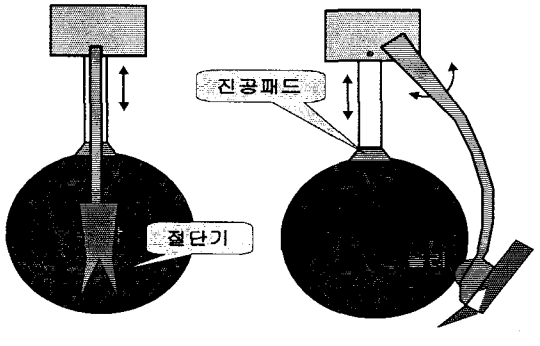

Fig. 3-2-12 Schematic diagram of harvest gripper (2_{nd} prototype).

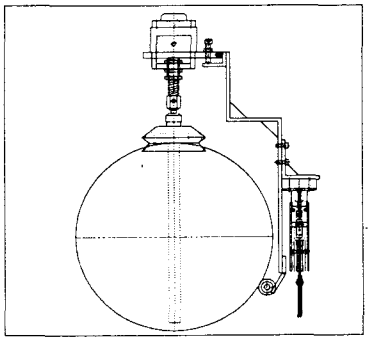

Fig. 3-2-13 Drawing of the 2nd prototype gripper.

3) 동작 방법

그림 3-2-14(a)와 같이 진공 패드를 수박에 접근 시키면서 절단기의 연속 절단을 시작한다. 그리고 (b) 와 같이 진공 패드가 표피에 접촉되면 공압 실린더를 통하여 수박을 견인한 후 (c)와 같이 구름 가이드를 통하여 절단기를 수박표면을 따라 이동하면서 연속 절단작업이 이루어진다. 이때 절단기의 각도는 진공 패드의 움직임 량에 따라 변화되게 하여 절단기의 절단면이 수박의 표피와 수평이 되도록 설계하였다.

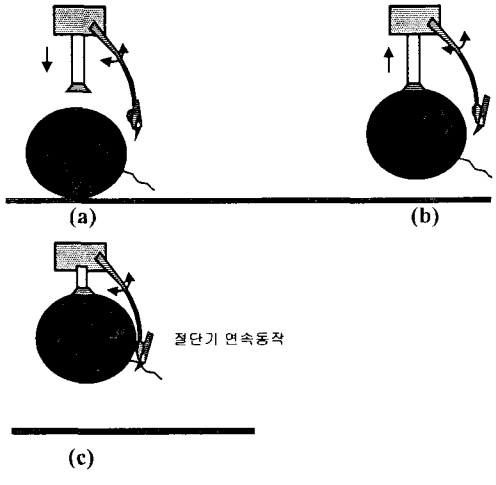

Fig. 3-2-14 Order of cutting (2_{nd} prototype).

라. 결과 및 고찰

수확을 위한 그리퍼(Gripper)를 1, 2차에 걸쳐 설계 제작하였다. 수확작업을 위한 그리퍼(Gripper)의 주요 기능은 꼭지 자르기와 견인 및 적재부 이송인데, 견인 및 이송의 경우 진공 패드의 성능은 양호한 결과를 얻었다. 그러나 꼭지 자르기의 경우에 있어서 꼭지 위치가 수박의 중심축에 대하여 수평하게 있지 않고 넓은 범위에 걸쳐 존재하므로 다양한 방법을 통하여 연구를 수행하였으며, 설계 제작하였다.

그 결과, 1차 시작기에서는 절단날의 교차 정밀도와 크기에서 문제가 발생하였으며, 2차 시작기의 경우 연속적인 절단작업으로 인하여 절단 시간이 다소 걸렸다. 실험을 통하여 340mm 구형 모형을 통하여 절단 시간을 측정한 결과 스텝 당 0.5초가 소요 되었으며, 전체 절단 횟수는 1 스탭 당 Z축의 이동거리를 10mm로 하였을 경우 20번 절단을 수행하여 총 10초가 소요되었다.

종합하면, 꼭지부 절단 성능은 다소 시간이 소요되었으나 양호하였으며, 견인성능은 매우 양호 하였다. 단 중량물의 이송의 경우 관성의 영향으로 저속으로 실험이 수행되었다. 여기서 성능 개선을 위해서는 매거진 이송부의 정밀 가감속 제어 및 경로 제어를 통하여 이송을 수행하면 관성의 영향을 적게 받음으로 이송에 필요한 소요 시간을 감소시킬 수 있을 것으로 사료된다.

제 3 절 영상처리 시스템 개발

1. 서론

최근, 작업환경이 가변적이고 작업 대상체의 외관 형상 및 내부 품질 특성이 정형화되어 있지 않은 생물생산 작업의 생력화(Automation)를 위한 연구 개발이 세계적으로 활발히 진행되고 있다(N. Kondo, 1998). 이러한 연구 개발에 있어 산업용 로봇 기술을 이용한 시스템 구동 및 제어, 컴퓨터 영상처리, 초음파 및 레이저를 이용한 영역 추출과 환경인식, 바이오센서를 이용한 대상체 특성 파악, 신경회로망, 퍼지(Fuzzy) 추론 및 전문가 시스템 등의 인공지능을 이용한 정보처리 기술 등이 중요한 역할을 하고 있다. 언급한 기술들은 가변적이고 복잡한 작업환경과 애매모호한 특성을 갖는 생물학적 대상체를 취급하는 생물생산 작업공정의 생력화(Automation)에 있어 기술적인 가능성을 제시하여 왔으나, 많은 경우 실제 작업에 적용하는데 있어서는 기술적인 측면에서 그리고 전반적인 효율성 측면에서 한계점을 표출해 왔다.

생물생산 작업을 수행하는 작업자의 역할을 기능적 측면에서 단순화하면 다음의 세 가지로 분류할 수 있다. 작업계획, 작업수행 및 작업예측, 작업상태 감시 및 작업성 판단 등을 총괄하는 뇌의 정보처리 기능과 작업상태와 주변 환경정보 획득을 위한 시각, 촉각, 후각, 미각, 청각 등의 감각기관을 이용한 정보 획득기능 그리고 마지막으로 뇌의 정보처리 결과에 의하여 동작하는 신체의 관절 및 근육 등의 구동기능이다.

생물생산 공정의 생력화(Automation)를 위하여 요구되는 언급한 작업자의 여러 기능에 있어서, 부분적으로는 인간의 작업기능을 대체할 수 있는 시스템이 개발되었다. 하지만 작업계획과 적절한 작업수행 및 작업예측, 작업상태의 감시 및 작업성의 판단 등을 총괄하는 뇌의 정보처리 기능에 있어서는 컴퓨터 하드웨어와 소프트웨어 기술의 급속한 발달에도 불구하고 아직은 요원한 실정이다. 이러한 시점에서, 생물생산 작업의 생력화(Automation)를 위한 기술 개발 연구는 생물생산 작업에 있어

작업자를 완전히 대체하기 보다는 작업자의 작업 편이성을 높여 전반적인 작업 생산성을 높이는 방향으로 수행되어야 한다. 이는 작업자가 작업수행을 하는데 있어 쉽게 작업피로를 느끼고 작업하기 꺼려하는 부분을 대체하는 시스템을 개발하는 것을 포함하며, 한편으로는 작업자의 작업 편이성 즉 뇌의 기능과 감각기능 그리고 구동 기능을 원활하게 수행할 수 있도록 보조하는 시스템 기술의 개발이 필요하다는 것을 의미한다.

특히, 시스템 개발을 통한 작업보조 및 부분적 작업 대체에 있어서, 작업자가 쉽게 할 수 있는 그리고 기능적으로 기존의 컴퓨터 기술보다는 월등한 기능을 보이는 부분에 대해서는 작업자의 도움을 받고 단순 반복 운동 또는 인간의 정보처리 특성상 구현하기 어려운 정밀도가 관여된 부분에 대해서는 컴퓨터-기계시스템으로 대체하는 것이 필요하다. 즉, 현장 적응성을 고려한 생물생산 작업의 생력화(Automation)를 위해서는 첫째, 작업자와 컴퓨터-기계간의 효율적 작업 배분이 필요하다. 둘째, 효율적 작업배분을 능률적으로 수행하기 위한 작업자와 컴퓨터-기계 시스템 사이의 편리한 작업 인터페이스 구축이 필요하다.

본 연구에서는 생물생산 작업의 생력화(Automation)를 위하여, 작업자와 컴퓨터-기계 사이의 효율적 작업배분과 작업 인터페이스 구축 개념을 컴퓨터 영상처리를 이용한 대상체의 인식 이라는 과제로 접근하였다. 생물생산 작업을 생력화(Automation)하는데 있어 많은 공정들이 컴퓨터 영상처리 기술을 이용하고 있으며 특히, 3차원 공간내의 대상체를 취급하거나 조작하는 데 있어서는 대상체의 인식 및 인식한 대상체의 정확한 위치 정보 즉, 3차원 좌표 추출이 요구된다. 이러한 문제를 해결하기 위한 기존의 접근 방법은 컴퓨터에 의한 영상처리 기술에 전적으로 의존함으로써 대상체를 인식하는 과제의 경우, 복잡한 작업 환경 하에서는 실시간 영상처리가 어렵고 가변적인 주변 환경과 대상체의 애매 모호성으로 인하여 인식정밀도의 안정성 확보가 문제가 되고 있다.

하지만, 이러한 문제는 작업자와 컴퓨터-기계의 작업배분과 효율적 인터페이스를 통하여 쉽게 해결할 수 있다. 즉 작업자의 도움을 통하여 대상체 인식에 대한 안정성과 강건성을 확보하고 컴퓨터-기계 시스템의 정밀성을 조합하는 방법으로 문제

를 해결할 수 있다. 그리고 작업자와 컴퓨터-기계 시스템간의 인터페이스는 원격 영상과 터치 패드를 이용한 간편한 입출력 방식으로 해결할 수 있다. 또한 복잡하고 가변적인 작업 환경과 대상체의 애매 모호성 문제는 작업자가 원격으로 송신되는 유무선 영상을 통하여 지시함으로써 해결할 수 있다. 보다 정밀하게 작업 대상체를 인식하고 작업 대상체의 위치좌표를 추출하는 것은 컴퓨터-기계 시스템이 수행하는 것이다.

본 연구에서는 원격영상 시스템과 터치 패드를 이용한 작업지시 입/출력 방식을 구축하였고 3차원 작업공간내의 작업 대상체를 인식하고 3차원 공간 좌표를 추출하는 시스템을 개발하였다. 개발한 알고리즘은 영상 보정을 통한 3차원 좌표추출에 관한 알고리즘과 자연광 하 및 반자동 범위에서의 수박 외형 추출, 줄기의 외형 추출에 관한 것이다. 또한, 작업자와 컴퓨터-기계간의 작업배분 및 인터페이스를 통한 작업 적응성을 구현하였다.

2. 스테레오 비전 시스템

가. 개발 배경

사람이 가지는 오감 중 시각을 컴퓨터로 대체 하는 컴퓨터 시각 기술은 10여 년간 하드웨어류의 급속적인 발전과 저가화를 계기로 고도의 정확성을 요하는 반도체 분야의 품질 검사로부터 일반 산업 분야의 다양한 작업 공정은 물론 농림, 축산, 수산 분야에 이르기까지 다양하게 활용되고 있다. 농업 생산의 첨단 자동화 중 영상 처리 기술은 우리나라뿐만 아니라 전 세계적으로 활발히 연구 중이지만, 아직 한정적인 성능 시험으로 실용화 단계에 있지는 않다. 특히 농산물 수확을 위한 영상 처리는 3차원 공간에 놓여 있는 대상물의 개별적인 위치를 선별 및 인식하여야 하는 어려움이 있다.

최근, 컴퓨터 시각 시스템을 이용하여 농산물 등의 인식과 선별에 대한 효율성을

증가시키고 있지만 대부분의 농산물 영상은 복잡한 3차원 구조의 군락을 이루고 있어 적절한 영상 분석이 매우 어렵다. 또한, 주변의 광 조건에 의해 영상 획득 및 보정 등의 과정이 요구되는 문제가 있다. 그러나 이러한 3차원 공간상의 좌표만 정확하게 측정할 수 있다면 농업 자동화에 지대한 역할을 할 것이다. 농산물의 수확의 적기를 맞춘다는 것은 농업에 있어 필연적인 것이다.

그러나 현실적으로 농업 인구의 감소와 고령화 등으로 인하여 수확 적기를 맞출 수 없는 부분이 있는 것이 현실이다. 3차원 공간상의 정확한 좌표를 추출하여 로봇 매니퓰레이터(Manipulator)에 그 좌표를 인식시켜, 과실 등의 수확을 로봇이 할 수 있게 할 수 있으리라 생각된다. 컴퓨터 시각 시스템을 이용하여 2차원 영상정보를 3차원 형상정보로 바꾸는 방법은 크게 세 가지로 나눌 수 있다.

첫째 한 대의 카메라를 사용하여 얻은 영상의 음영을 이용하여 깊이에 대한 정보를 얻는 형상 인식법(Shape Form Shading Technique)이 있다. 이것은 다른 방법에 비해 알고리즘이 복잡하고 형상의 깊이 인식은 정확성이 떨어지는 단점을 지니고 있어 활용에 어려움이 많다. 둘째, 한 대의 카메라와 레이저 구조광을 이용하여 3차원 정보를 측정하는 방법인데, 레이저 구조광은 일반적인 조명과는 달리 단파장이고, 직진성을 가지고 있다. 이러한 특성을 이용하여 기준면에 투사되는 레이저 구조광이 대상체에 투사될 때 나타나는 형상변이를 이용하여 대상체의 3차원 형상 정보를 얻을 수 있다. 이 방법을 사용하면 일반 조명으로 인한 빛의 퍼짐 현상이 없어 대상체의 영상을 정확하게 인식할 수 있고, 레이저 구조광 이외에는 조명을 사용하지 않으므로 잡음이 적은 이점이 있다. 세 번째 방법은 인간의 시각처럼 두 대의 카메라를 이용하여 3차원 정보를 얻는 스테레오 시각(Stereo Vision)법이 있다. 이 방법은 두 대의 카메라를 통해 얻은 각각의 정보를 이용하여 삼각 측량을 실시하여 3차원 정보를 산출하는데, 두 카메라간의 대응점 추출 알고리즘이 복잡하여 처리 시간이 길어지는 단점이 있다. 또한 카메라 2대를 이용하므로 가격이 비싼 것이 단점으로 적용된다. 보다 효율적이고 경제적인 영상 처리를 위하여 카메라 1대를 이용하여 2대의 효과를 낼 수 있는 방법을 고려하여 본 연구를 하게 되었다.

2차원의 영상 정보만을 획득하여 3차원의 좌표를 표현하기에는 많은 오차 요인(카메라 렌즈의 곡률 왜곡, 거리의 변화에 따른 화소값들의 변화량 등등)들이 포함되어 있어 정확한 좌표를 구하기 매우 어려운 현실이지만 적정한 오차의 범위에서 3차원 형상 정보에 대하여 유효한 정보를 얻을 수 있으리라 사료되어 본 연구를 위한 시스템을 개발하였다.

나. 배경 이론

1) 문헌개요

컴퓨터 시각을 이용하여 3차원 정보를 획득하기 위한 많은 연구들이 근간에 많이 행해져 왔다. 1대의 카메라를 사용하는 스테레오 비전 방법은 2대보다 가격이 저렴하고 처리 속도가 빠른 장점이 있다. 3차원 정보 획득을 위해서는 2차원의 영상 정보를 3차원 기준 좌표계의 정보로 변환시켜 주는 카메라 보정이 필요하다. 보정의 정확도는 3차원 정보 획득의 정확도에 많은 영향을 미친다. 따라서 카메라 보정에 관한 많은 연구들이 수행되었다.

Tsai(1986)는 카메라의 초점거리, 렌즈의 왜곡이나 영상 인식 파라미터들의 내부적인 요소뿐만 아니라 기준 좌표계 시스템에서 대상체와의 상대적인 관계를 표현하는 카메라의 위치(Position)나 자세(Orientation)등의 외부적인 요소의 산출에도 효과적인 2단계 기법(2-Stage Technique)을 개발하였다.

Hung(1988)은 로봇에 장착된 카메라를 보정하기 위하여 Off-line 단계와 On-line 단계로 나누어 처리하는 2단계 기법을 사용하였는데, Off-line 단계는 카메라, 로봇, 로봇-카메라(혹은 팔과 눈) 등의 관계를 이용하며, On-line 단계에서는 초기치를 계산하는 단계와 초기치를 이용하여 재보정하는 단계로 이루어져 있다.

Kearmey(1989)등은 기하학적 구속 원리를 적용하여 미세 조정이 가능한 4개의 독립된 보정 장치로부터 카메라의 중심과 초점거리 같은 불변하는 파라미터와

기준 좌표계 상의 카메라 위치나 자세 등에 의해 결정되는 변하는 파라미터의 두 가지 파라미터를 추출하였는데, 이것을 연계시키기 위한 비선형 방정식을 계산하기 위하여 종속되지 않은 기지의 목표점들을 이용하였다.
Liu(1990)등은 선형 알고리즘의 경우 최소 8개의 라인 혹은 6점의 대응을, 비선형 알고리즘의 경우 적어도 3개의 라인이나 3점의 대응을 이용하여 2차원 영상과 3차원 공간상의 직선이나 점의 대응을 결정하는 카메라 보정을 실시하였다.

Lee(1991)는 카메라 측정 시 발생하는 렌즈의 비선형 오차를 신경회로망에 의해 학습하여 보정하는 알고리즘의 개발을 수행하였다.
Wang(1991)등은 카메라의 자세, 위치, 초점거리 등의 파라미터를 보정하기 위하여 투사된 영상으로부터 기준 평면의 소멸선을 생성하는 3쌍의 평행면으로 구성된 평판 육각형을 이용하였다.

Heikkila(1997) 등은 종래의 2단계 카메라 보정 방법에 서클 형태로 표현되어 발생하는 왜곡을 보정하는 단계와 왜곡된 영상 좌표를 바로잡아주는 2단계를 추가한 4단계 보정 방법을 개발하였다.

Huynh(1997) 등은 동일 라인 상의 3개의 점들로 이루어진 동일 평면에 존재하지 않는 4개의 집합을 이용하여 각 광원의 변화에 구애되지 않고 평면들마다 동일하게 적용할 수 있는 4×3 영상-기준 좌표 변환 행렬을 산출하였다.

2) 보정 이론

영상처리 기법에 있어서 가장 중요한 부분이 카메라 보정을 행하는 부분이다. 실제의 좌표 혹은 크기를 결정해 주는 인자를 찾아 방정식으로 표현하고 카메라에 의한 영상을 실제의 모델로 일치 시켜주는 작업이 카메라 보정이다.
본 연구에서는 2차원 영상 좌표계를 3차원 기준 좌표계로 변환하는데 따른 보정을 하였다.

인간의 시각을 기계로 대체하기 위한 영상처리 기술은 대체로 그림 3-3-1과 같

은 카메라 모델로 표현된다.

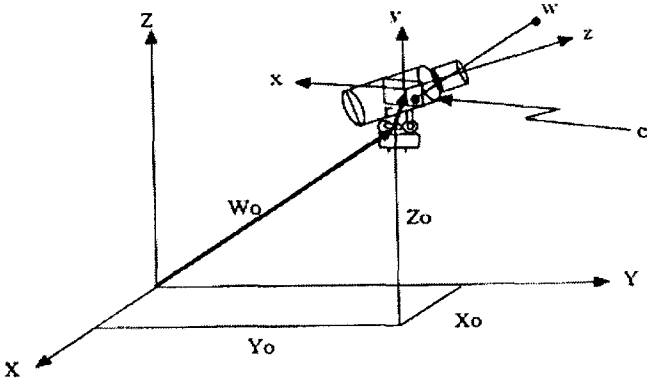

Fig. 3-3-1 Geometrical Modeling of Camera

기준좌표계(X, Y, Z) 상에서 각각 X_0, Y_0, Z_0 만큼 이동한 위치에 카메라 좌표계인 (u, v, w)가 존재하고, 임의의 3차원 공간상에 점 **w**는 카메라 영상면의 점 **c** 로 바뀐다. 카메라는 고정장치의 Pan-Tilt에 의하여 원점이 이동되는데 이러한 이동에 따라 기준 좌표계의 원점은 영상 좌표계로 W_0만큼 이동되고, 영상면은 Pan-Tilt에 의하여 **r**만큼 이동된다. 이러한 기준 좌표계와 영상좌표계간의 관계는 다음식과 같이 표현할 수 있다.

$$C_h = PW_h \quad \text{----------식(3-3-1)}$$

여기서 , C_h : 영상 좌표

P : 변환 행렬

W_h : 기준 좌표

카메라로부터 얻어지는 대상체의 2차원 영상좌표를 기준 좌표계상의 3차원 공간 좌표로 변환하는 방법이 투사 변환 혹은 영상화 변환이라고 하는데 이것은 3차원 공간의 점을 평면에 투사시키는 수학적 방법이다. 영상화 과정의 모형은 그림 3-3에서 보는 바와 같다.

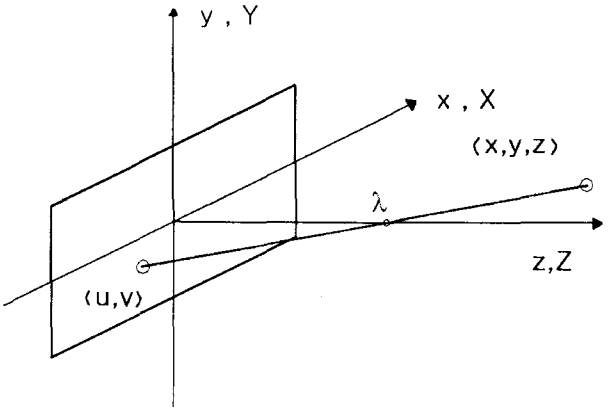

Fig. 3-3-2 Geometrical projected model of image formation process.

카메라의 영상면이 x-y평면에 놓여져 있고, z축은 카메라로부터의 거리를 나타낸다. 렌즈의 초점거리를 λ라고 하면, 원거리에 놓여 있는 물체는 렌즈가 카메라 영상면에서 λ만큼 떨어져 있을 때 정확한 영상을 맺는다.

공간상의 한 점의 좌표가 (x, y, z)라 할 때, z > λ라 하면 이 점은 렌즈의 앞쪽에 있음을 시사한다. Fig 3-3-2로부터 닮은꼴 삼각형을 이용하여 (u, v)점과 (x, y, z)점과의 관계 다음 식과 같이 얻을 수 있다.

$$\frac{u}{\lambda} = -\frac{x}{z-\lambda} \qquad \frac{v}{\lambda} = -\frac{y}{z-\lambda}$$ ---------- 식(3-3-2)

이들로부터 다음과 같은 결과를 얻는다.

$$u = \frac{\lambda x}{\lambda - z} \qquad v = \frac{\lambda y}{\lambda - z}$$ ---------- 식(3-3-3)

이러한 결과는 분모에 변수 z가 존재하므로 비선형이다. 이런 식을 계산하기 위해서 행렬화 하는 것이 유용한데, 이를 위해 동차 좌표계로 바꾸어 주는 것이

필요하다. 기준 좌표계의 한 점 (x, y, z)은 동차좌표계에서는 (kx, ky, kz, k)로 정의 되는데, 여기서 k는 투영변수를 의미한다. 따라서 동차 좌표계의 한 점으로부터 직교좌표계의 한 점으로 변환시키고자하면, 동차좌표계의 처음의 3요소를 마지막 요소로 나누어주면 된다. 기준좌표계의 한 점 **w** 가 다음과 같은 벡터로 주어지면

$$W = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

이에 대응하는 동차좌표계의 점 $\mathbf{W}_h$는

$$W_h = \begin{bmatrix} kx \\ ky \\ kz \\ k \end{bmatrix}$$ 이다.

투사변환행렬 P는 다음과 같다.

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -\frac{1}{\lambda} & 1 \end{bmatrix}$$

식 3-1로부터 C_h 좌표가 얻어진다.

$$C_h = PW_h$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -\frac{1}{\lambda} & 1 \end{bmatrix} \begin{bmatrix} kx \\ ky \\ kz \\ k \end{bmatrix} = \begin{bmatrix} kx \\ ky \\ kz \\ k - \frac{k}{\lambda z} \end{bmatrix}$$

영상면의 좌표 C_h를 기준좌표로 옮기려면 C_h의 마지막 요소로 3개의 요소를 나눈다. 즉, C_h에 대응하는 점 **c** 는

$$c = \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} \frac{\lambda x}{\lambda - z} \\ \frac{\lambda y}{\lambda - z} \\ \frac{\lambda z}{\lambda - z} \end{bmatrix} \quad \text{----------식(3-3-4)}$$

이다. 이 때 **c** 의 첫 번째 두 개의 요소가 **(u, v)**에 해당한다.

P의 역변환 P^{-1}은 영상면의 한 점을 다시 공간상의 한 점으로 바꾸어 준다. P^{-1}은 다음과 같다.

$$P^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & \frac{1}{\lambda} & 1 \end{bmatrix}$$

영상면의 한 점 $c = (u_0, v_0, 0)$을 공간상의 한 점으로 환원시키면

$C_h = \begin{bmatrix} ku_0 \\ kv_0 \\ 0 \\ k \end{bmatrix}$ 이다.

C_h에 대응하는 공간상의 한 점 Q_h는

$Q_h = P^{-1}C_h = \begin{bmatrix} ku_0 \\ kv_0 \\ 0 \\ k \end{bmatrix}$ 와 같으며

기준좌표계에서 **Q** 는

$$Q = \begin{bmatrix} u_0 \\ v_0 \\ 0 \end{bmatrix}$$ 이다.

이것은 원하는 바가 아니다. 왜냐하면 어떤 공간상의 점이 모두 z = 0 으로 주어지기 때문이다. 본래의 투사식 식(3-3-3)으로부터

$$x = \frac{u(\lambda - z)}{\lambda}$$

$$y = \frac{v(\lambda - z)}{\lambda}$$

이므로 (x_0, y_0)에 대응하는 공간의 점은

$$x = \frac{u_0(\lambda - z)}{\lambda}$$

$$y = \frac{v_0(\lambda - z)}{\lambda}$$ ----------식(3-3-5)

으로 주어진다. 이로부터 한 가지 정보가 있어야 공간상의 점을 산출할 수 있음을 알 수 있다. 즉, z 값이 주어져야 한다. c 의 z 성분을 0 대신 k_0라 하면,

$$C_h = \begin{bmatrix} ku_0 \\ kv_0 \\ k_w \\ k \end{bmatrix}$$ 이므로

$W_h = P^{-1}C_h$를 계산하면

$$W_h = \begin{bmatrix} ku_0 \\ kv_0 \\ k_w \\ k+\frac{k_w}{\lambda} \end{bmatrix}$$ 이다.

이를 기준좌표로 변환하면

$$w = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \frac{\lambda u_0}{\lambda+w} \\ \frac{\lambda v_0}{\lambda+w} \\ \frac{\lambda w}{\lambda+w} \end{bmatrix}$$ ----------식(3-3-6)

다시 말하면 역투사 변환을 하기 위하여 w의 정보가 주어져야 한다. 이러한 원리에 의해서 2차원 영상좌표를 3차원 공간좌표로 변환할 수 있는데 이러한 방법에서는 일반적으로 좌표변환에 영향을 끼치는 두 가지 요소를 고려하여야 한다.

첫째로는 카메라 렌즈의 초점거리, 영상면의 중심좌표, 화소좌표가 갖는 실제 측도 등의 내부인자이고, 둘째는 카메라 고정에 의한 자세로부터 발생하는 Pan-Tilt 각도 및 카메라의 위치정보 등의 내부인자이다.
그러나 본 연구에서는 이러한 인자들을 전혀 고려치 않고 원하는 결과를 얻을 수 있는 방법을 사용하였다. 카메라 자체를 하나의 측정 장치로 사용하여 같은 평면에 존재하지 않는 기지의 6점에 대한 영상 정보를 가지고 실제의 3차원 공간 좌표를 추출하는 방법이다. 거리(depth)정보를 알고 있는 기지의 6점에 대한 화소좌표를 영상평면에서 측정한 후, 2차원 영상 좌표계와 3차원 기준 좌표계를 대응시킴으로써 보정을 실시하였다.
3차원 기준 좌표계 상의 점 W_h는 4×4 선형 좌표변환행렬 P=[P_{ij}]를 이용하여 소실되는 미지의 **z** 정보 **w**를 무시하고 식(3-3-1)을 전개하여 다음 식을 얻는다.

$Ku = P_{11}x + P_{12}y + P_{13}z + P_{14}$ ----------식(3-3-7)

$Kv = P_{21}x + P_{22}y + P_{23}z + P_{24}$ ----------식(3-3-8)

$K = P_{41}x + P_{42}y + P_{43}z + P_{44}$ ----------식(3-3-9)

그리고 식(3-3-9)를 식(3-3-7)과 식(3-3-8)에 대입하여 다음 식을 얻는다.

$P_{11}x+P_{12}y+P_{13}z-P_{41}ux-P_{42}uy-P_{43}uz-P_{44}u+P_{14}=0$ ---식(3-3-10)

$P_{21}x+P_{22}y+P_{23}z-P_{41}ux-P_{42}vy-P_{43}uz-P_{44}v+P_{14}=0$ ---식(3-3-11)

3차원 공간상의 한 점과 이에 대응되는 영상좌표로부터 식 (3-3-10)과 식 (3-3-11)을 얻을 수 있으므로 6점의 공간 좌표와 대응되는 영상좌표로부터 12개의 식을 얻을 수 있다. P_{44}는 1 이므로 이를 행렬식으로 표현하면 다음과 같다.

$$\begin{bmatrix} x_1 & y_1 & z_1 & 1 & 0 & 0 & 0 & 0 & -u_1x_1 & -u_1y_1 & -u_1z_1 \\ 0 & 0 & 0 & 0 & x_1 & y_1 & z_1 & 1 & -v_1x_1 & -v_1y_1 & -v_1z_1 \\ & & & & & & \vdots & & & & \\ x_6 & y_6 & z_6 & 1 & 0 & 0 & 0 & 0 & -u_6x_6 & -u_6y_6 & -u_6z_6 \\ 0 & 0 & 0 & 0 & x_6 & y_6 & z_6 & 1 & -v_6x_6 & -v_6y_6 & -v_6z_6 \end{bmatrix} \begin{bmatrix} P_{11} \\ P_{12} \\ \vdots \\ P_{42} \\ P_{43} \end{bmatrix} = \begin{bmatrix} u_1 \\ v_1 \\ \vdots \\ u_6 \\ v_6 \end{bmatrix}$$

---------- 식(3-3-12)

식(3-3-12)를 AP = B 로 두면, A의 의사 역행렬을 B와 곱하여 원하는 변환 행렬 P를 얻는다.

$P = \{(A^T \times A)^{-1} \times A^T\} \times B$ ---------- 식(3-3-13)

식(3-13)의 결과로부터 얻은 행렬 P의 값을 식(3-3-10),식(3-3-11)에 대입하여 x, y를 구한다.

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} R_{11} & R_{12} \\ R_{21} & R_{22} \end{bmatrix} \times \begin{bmatrix} K_1 \\ K_2 \end{bmatrix} \quad \text{----------식(3-3-14)}$$

여기서, $R_{11} = P_{11} - P_{41} \times u$

$R_{12} = P_{12} - P_{42} \times u$

$R_{21} = P_{21} - P_{41} \times v$

$R_{22} = P_{22} - P_{42} \times v$

$K_1 = -P_{13} \times z + P_{43} \times u \times z + u - P_{14}$

$K_2 = -P_{22} \times z + P_{43} \times v \times z + v - P_{24}$

본 연구에서는 3차원 공간상의 6점의 정보를 얻기 위해서 보정차트를 제작하였다. 카메라 렌즈에서 530mm떨어진 지점을 기준점으로 하고, 보정차트 중심을 XY축의 원점으로 하는 3차원 기준좌표계를 설정하였다. 기준점으로부터 100mm씩 뒤쪽으로 이동하면 각 지점의 영상 좌표를 측정하였다

보정 차트의 중심은 카메라를 통해 얻는 영상 중앙점의 좌표(u=320, v=240)와 일치시켰으며, 차트의 전 구간에 임의의 6점을 선정하였다.
530~1030mm까지 6점에 대해 이미 알고 있는 3차원 정보 (x, y, z)와 그 때의 화소 좌표값(u, v)을 식 (3-3-13)에 대입하여 변환행렬 P를 구하고, 식 (3-3-14)를 이용하여 실제 x, y 좌표와의 관계를 측정하였다.

3) 3차원 공간의 좌표 측정 방법

본 연구는 특정물을 대상으로 하기 이전에 예비적으로 좌표 정보를 얻기 위한 것이기 때문에 작은 점을 기준으로 한다. 한 점에서 인식한 2개의 형상좌표 (u_1, v_1, u_2, v_2)를 이용하여 3차원상에 있는 점의 좌표를 구하기 위한 방법으로 2가

지가 있다.

첫째, 기하학적 성질과 카메라 모델링을 이용한 3차원 좌표 측정 방법
둘째, 카메라 자체를 측정 장치한 3차원 좌표 측정 방법 등이 있다.

가) 기하학적 성질과 카메라 모델링을 이용한 3차원 좌표 측정 방법

그림 3-3-3에서와 같이 스테레오 영상화 기법은 공간상의 하나의 점 **p**에 대해 두 개의 분리된 영상을 얻는 것이다.
두 렌즈 중심 간의 거리를 베이스 라인(Base Line : B) 이라고 한다. 그리고 목표는 영상점 (u_1, v_1)과 (u_2, v_2)를 갖는 점 **p** 의 좌표 (X, Y, Z)를 찾는 것이다. 여기서 카메라는 동일한 것이고 카메라에 의한 두 좌표계는 단지 각각의 원점이 다를 뿐 완전히 일직선상에 있다고 가정한다. 카메라와 실 좌표계를 일치시키면, 영상의 **xy** 평면은 실 좌표계의 **XY**평면과 일직선상에 있다. 이 가정 하에서 **p** 의 Z좌표계는 정확히 두 카메라 좌표계와 같다. 카메라를 실 좌표계와 일치시키기 위하여 그림 3-3-3을 위에서 본 투영도는 그림 3-3-4와 같다.

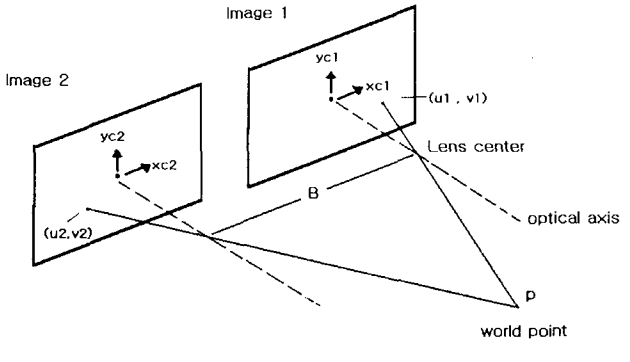

Fig. 3-3-3 Geometrical model for acquisition of stereo image.

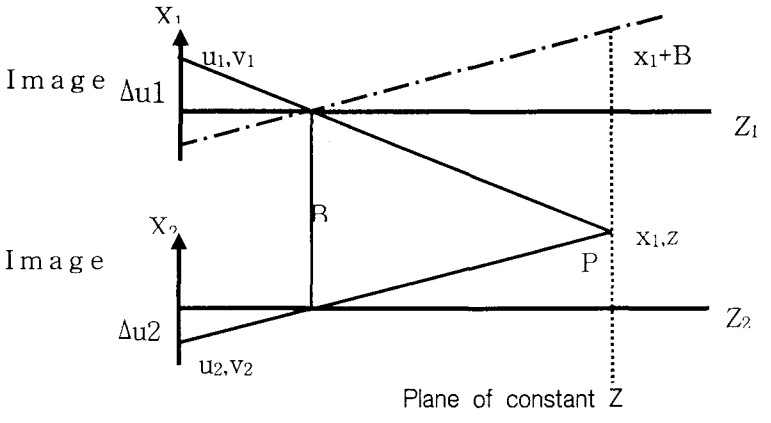

Fig. 3-3-4 Projected model for acquisition of stereo image.

식 3-3-3에 의해 **p**는 아래 좌표를 갖는 선상에 놓이게 된다.

$$x_1 = \frac{u_1(\lambda - z_1)}{\lambda} \qquad \text{----------식(3-3-15)}$$

여기서 **x**, **z**의 아래 첨자는 첫 번째 카메라가 실 좌표계의 원점으로 움직였다는 것을 의미한다. 두 번째 카메라와 **p** 는 따라오지만, 그림 3-3-4에서와 같이 상대적인 배열은 그대로 유지한다. 그 대신에 만약 두 번째 카메라가 실 좌표계의 원점으로 움직인다면, **p** 는 아래 좌표를 갖는 선상에 있게 된다.

$$x_2 = \frac{u_2(\lambda - z_2)}{\lambda} \qquad \text{----------식(3-3-16)}$$

그러나 카메라가 이동하였고 **p**의 **z** 좌표가 두 카메라 좌표계와 같기 때문에

$$x_2 = x_1 + B \qquad \text{----------식(3-3-17)}$$

$z_2 = z_1 = z$ ---------- 식(3-3-18)

이다. 여기서 **B** 는 베이스라인 거리이다.

식(3-3-17)과 식(3-3-18)을 식(3-3-15)과 식(3-3-16)에 대입하면

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

$B + x_1 = \frac{u_2(\lambda - z)}{\lambda}$ 이다.

위식을 대입하여 풀면

$z = \lambda - \frac{\lambda B}{u_2 - u_1}$ 이다. ----------식(3-3-19)

위 식은 대응하는 영상 좌표 u_1과 u_2의 차이가 결정되어질 수 있고 베이스라인과 초점거리를 알 때 **p**의 **z** 좌표를 계산할 수 있다. 실 좌표 X,Y 는 (u_1, v_1) 또는 (u_2, v_2)를 이용하여 식(3-3-17)을 대입하여 구할 수 있다.

그러나 이론적인 이런 식들을 실제로 적용하기 어려움이 있다.
첫째, 카메라의 초점을 알기가 어렵다는 것이다. 각 CCD카메라 고유의 특성을 가지고 있고 더욱 중요한 렌즈마다 카메라 초점이 다르다는 것이다. 둘째, 위 식의 기하학적 방법을 사용하기 위하여 x, B, u의 닮은꼴 삼각형을 이용하였으나 실제로 적용할 때 단위가 다르므로 형상좌표 u를 실제 기준좌표인 x 단위(mm)로 변경하여야 한다. 그러나 거리에 따라 1 pixel당의 거리(Depth)가 다르므로 이 관계식을 세우는 것 또한 매우 번거롭다.

위와 같은 문제점으로 인하여 외부적인 요인과 내부적인 요인을 생략하고 계산할 수 있는 방법을 사용하였다. 즉 카메라 자체를 측정 장치로 한 3차원 좌표 측정 방법이다. 카메라의 기준점에서 보정식을 세워 좌표변환행렬 **P**를 구한다. 또한 40mm씩 이동한 거리에서 좌표변환행렬 P′를 구한다. 이방법의 장점은 카메라에 상대 좌표를 인식시켜 줄 수 있는 장점이 있다. 즉 보정식에 의하여 알기 힘든 절대 좌표를 이미 알고 있는 상대 좌표로 인식시킬 수 있어 보다 쉽게 좌표를 찾을 수 있는 장점이 있다.

나) 카메라 자체를 측정 장치로 한 3차원 좌표 측정 방법

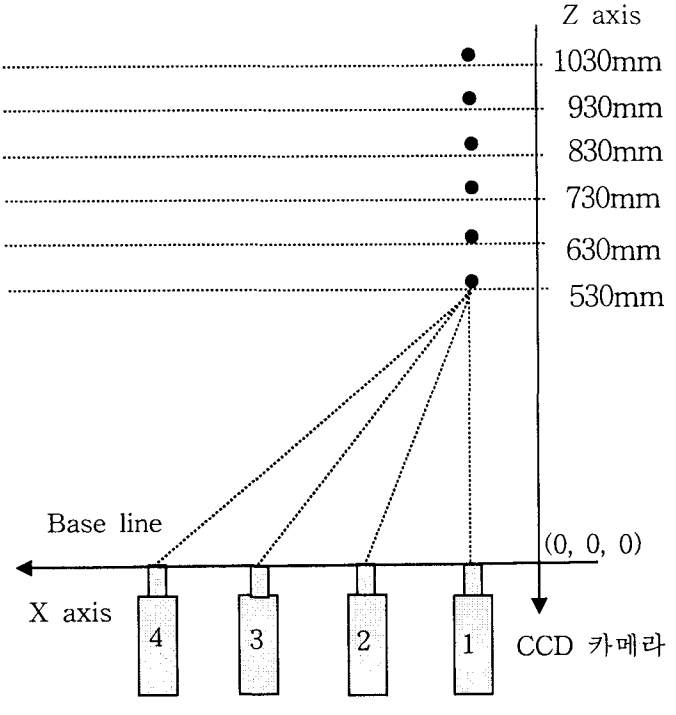

Fig. 3-3-5 Schematic diagram of image calibration of moving camera (X axis).

각각의 카메라 보정식으로부터 카메라 1의 변환식은

$$(A_{11}-A_{41}U_i)X_i+(A_{12}-A_{42}U_i)Y_i+(A_{13}-A_{43}U_i)Z_i=U_i-A_{14}$$
$$\Rightarrow \alpha_1X_i+\alpha_2Y_i+\alpha_3Z_i = \tau_1$$
$$(A_{21}-A_{42}V_i)X_i+(A_{22}-A_{42}V_i)Y_i+(A_{23}-A_{43}V_i)Z_i=V_i-A_{24}$$
$$\Rightarrow \alpha_4X_i+\alpha_5Y_i+\alpha_6Z_i = \tau_2$$

----------식 (3-3-20)

또한 40mm만큼씩 이동한 지점에서의 카메라 2의 변환식은

$$(B_{11}-B_{41}U_i)X_i+(B_{12}-B_{42}U_i)Y_i+(B_{13}-B_{43}U_i)Z_i=U_i-B_{14}$$
$$\Rightarrow \beta_1X_i+\beta_2Y_i+\beta_3Z_i = \tau_3$$
$$(B_{21}-B_{42}V_i)X_i+(B_{22}-B_{42}V_i)Y_i+(B_{23}-B_{43}V_i)Z_i=V_i-B_{24}$$
$$\Rightarrow \beta_4X_i+\beta_5Y_i+\beta_6Z_i = \tau_4$$

----------식 (3-3-21)

이다. 식 (3-3-20)과 식 (3-3-21)을 행렬로 표현하면

$$\begin{bmatrix} \alpha_1 & \alpha_2 & \alpha_3 \\ \alpha_4 & & \end{bmatrix} \quad \begin{matrix} \alpha_5 & \alpha_6 \\ & \beta_1 \end{matrix} \quad \begin{matrix} \beta_2 & \beta_3 \\ & \end{matrix}$$

----------식(3-3-22)

$$\| \qquad \| \qquad \|$$
$$A \qquad W \qquad C$$

이것을 간단히 표현하면, A×W = C 이다.
이 식의 의사 역행렬을 구하면

$$W = \{(A^T\times A)^{-1} \times A^T\} \times C$$

----------식(3-3-23)

공간상에 있는 임의의 점 W의 좌표는 W의 두 카메라에 맺힌 U, V점에 대한 화소값을 측정하여 식 (3-3-23)에 대입함으로서 구할 수 있다.

다. 실험 장치 및 재료

1) 구성

본 연구에서 구축한 실험장치는 크게 컴퓨터 영상처리 시스템, CCD 카메라 이동장치로 구성된다. 이러한 두 부분의 주요 기능을 블록도로 그림 3-3-6에 나타내었다.

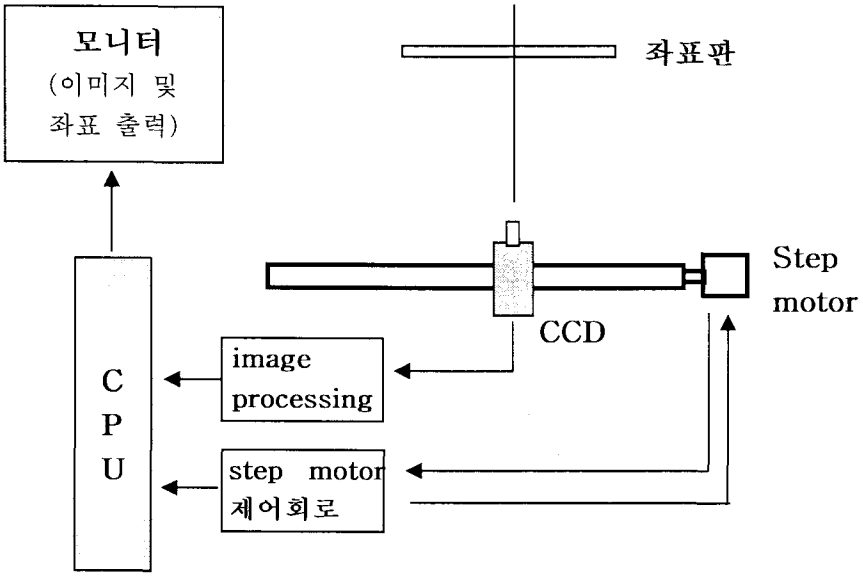

Fig. 3-3-6 Block diagram of system for 3-D coordinate measurement.

제작한 측정 시스템은 그림 3-3-7과 같으며 시스템의 골격은 알루미늄 프로파일(규격 40×40mm)을 사용하였다. 이 시스템의 전체 크기는 좌표 변환 장치를

제외한 상태에서 1180×940×770mm (길이×폭×높이) 이고 좌표변환 장치는 카메라 렌즈 중앙에서부터 530mm에서 1030mm까지 이동 가능하도록 만들었다.

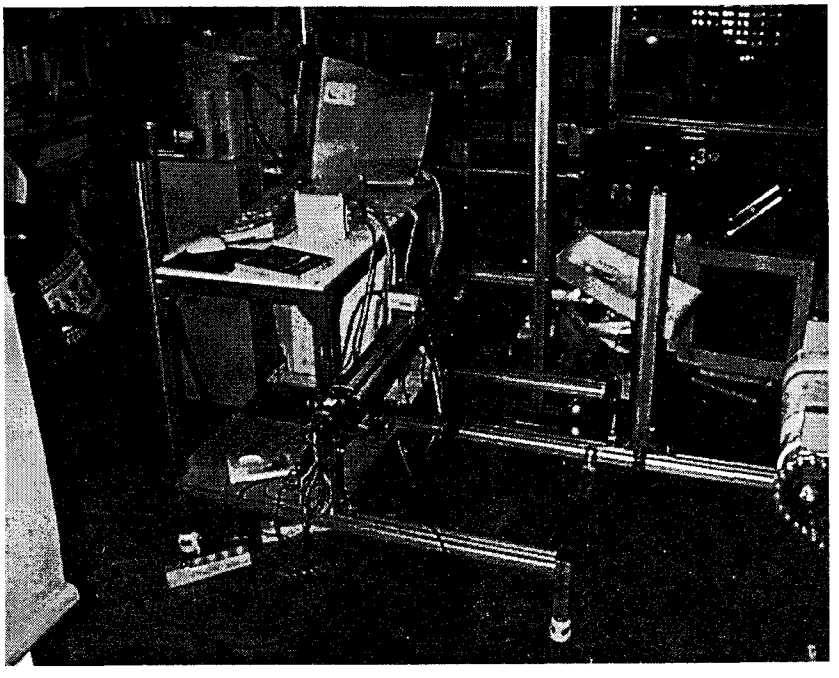

Fig. 3-3-7 Figure of System

2) 컴퓨터 영상처리 시스템

본 연구에서 사용한 컴퓨터 영상 처리 시스템은 영상 입력부, 프레임 그래버, 영상 처리에 사용되는 컴퓨터, 그리고 영상 출력부로 구분된다. CCD 카메라를 이용하여 대상체의 3차원 정보를 2차원 영상 정보를 입력받아 프레임 그래버 (Frame Grabber)에 저장하고, 이때 저장된 정보를 컴퓨터에서 처리하여 컬러 모니터에 화면으로 출력하는 동시에 계산된 3차원 형상 정보인 3차원 공간 좌표를 모니터에 출력하는 작업을 행한다.

가) 영상 입력부

측정하고자 하는 대상체의 2차원 영상 정보를 입력하는 센서로 본 연구에서 사용한 장치는 그림 3-3-8에 나타낸 Super Fine사의 CV-950 CCD 카메라이다. 이것을 그림 3-3-9과 같이 2상 STEP모터 및 볼 스크류로 구동되는 linear motion에 장착하였다.

좌표값을 변화시켜 다양한 거리를 구할 수 있는 좌표 이동 장치를 구성하였다. CCD 카메라 렌즈에서 최소 530mm부터 최대 1030mm까지 이동할 수 있도록 설치하였다. 변화하면서 x, y의 좌표값은 변화하지 않도록 프로파일로 고정하도록 설치하였고, 이동한 거리를 측정하기 위하여 CCD 카메라의 이미지 영상면으로부터의 거리를 알 수 있는 거리를 표시하였다.

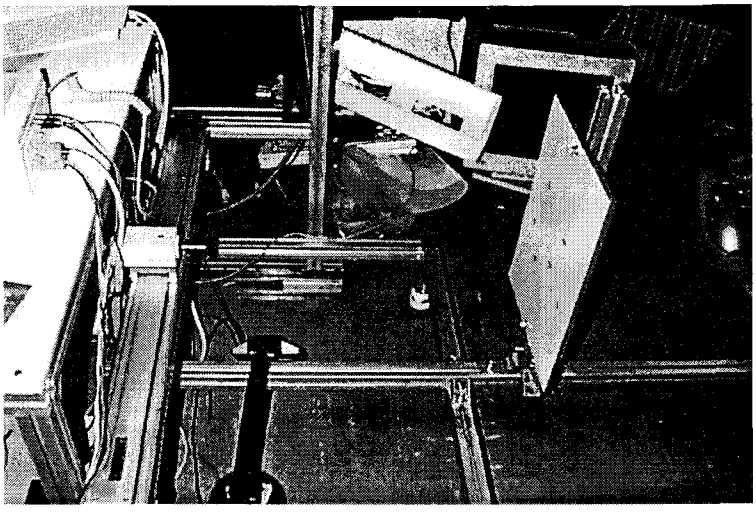

Fig. 3-3-8 CV-950 CCD camera.

나) 영상 처리 컴퓨터

카메라로부터 입력받은 영상 정보를 처리하기 위한 프레임 그래버를 설치하였다. Table 3-3-1에는 IBM PC 호환형 컴퓨터의 주요 사양을 나타낸 것이

다.

<Table 3-3-1> Specification of IBM PC Exchange Type Computer

구 성	모 델
Type	IBM 586 PC Compactible
CPU	Pentium 166 MHz (intel)
RAM	64 MB
HDD	6.4 GB (quentum)
O / S	windows 98 (Microsoft)
Programming Language	Visual C^{++}

다) 프레임 그래버

카메라에서 받아들인 아날로그 영상정보를 디지털 정보로 변환하는 장치로 본 연구에서는 CORECO 사의 BANDIT 제품을 사용하였다.

다음 그림은 영상을 획득하기 위한 영상처리 시스템의 기능 블록도이다.

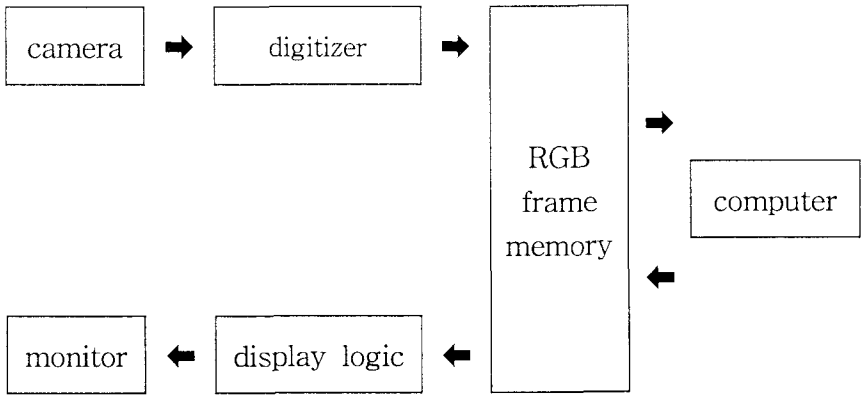

Fig. 3-3-9 Function block diagram of image treatment system.

라) 영상 출력부

카메라의 영상을 출력하고 개발된 소프트웨어를 이용해 처리되는 영상을 출력하는 장치로 15인치 컬러 모니터를 사용하였다.

3) CCD 카메라 이동장치

고정된 좌표점을 촬영하기 위하여 정밀도를 갖고 CCD 카메라를 이동시킬 수 있는 이동 장치를 제작하였다.

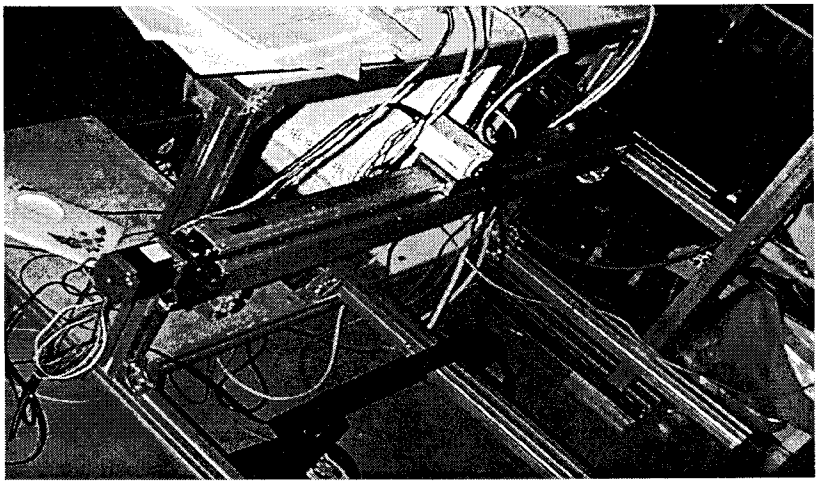

Fig. 3-3-10 System to move CCD camera.

이동 장치를 구성하기 위하여 Step 모터와 제어장치를 별도로 구성하였다. 제어장치는 Step 모터의 안정적인 구동 및 CCD 카메라의 이동 거리를 제어하는 중요한 요소로 영상처리 알고리즘에서 같이 제어되도록 프로그래밍을 하여 이미지 획득과 함께 제어 할 수 있도록 하였다.

가) Step 모터

정확한 거리 이동을 제어하고, 거리 계산을 쉽게 하기 위하여 Step 모터를 사용하였다. Step 모터의 주요 사양은 Table 3-3-2와 같다.

<Table 3-3-2> Specification of Step Motor Movement System

구 성	특 징
구동방식	유니폴라 방식
여자방식	2상
Step angle Deg	1.8°
Volt (v/phase)	4.5
Current	2A
Holding Torque	13.5 kg/cm

나) Step 모터 제어 system

본 연구에서는 Step 모터를 제어하기 위하여 아래 그림과 같이 전용 제어 장치를 만들었다.

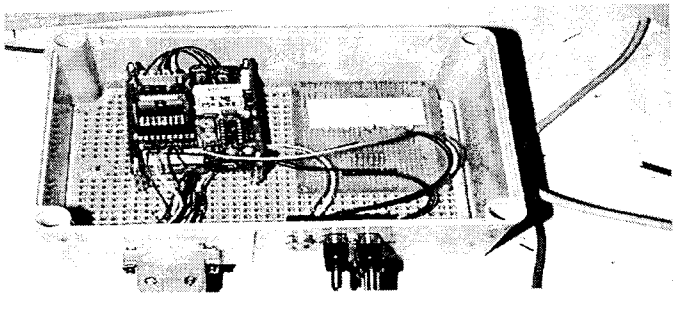

Fig. 3-3-11 Controller for step motor.

제어 장치의 회로 구성도는 그림 3-3-12과 같다.

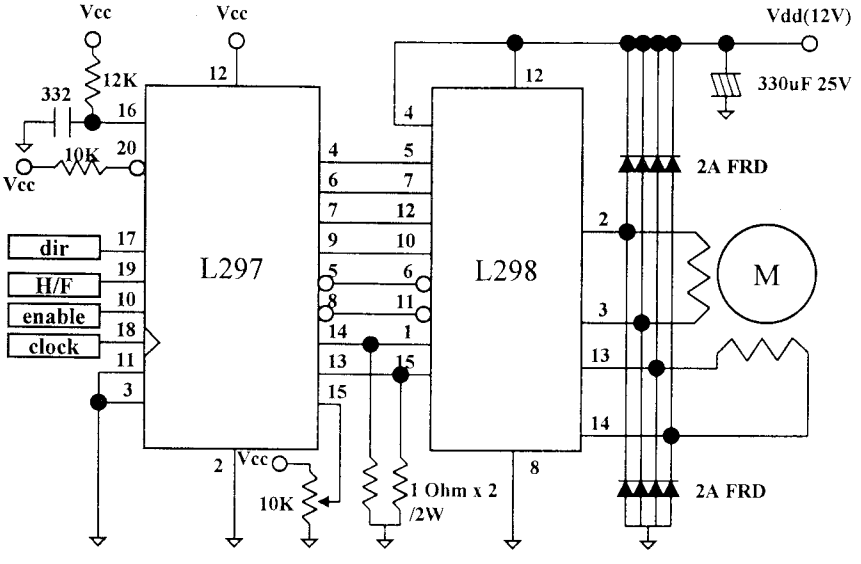

Fig. 3-3-12 Organization diagram of circuit for step motor control.

라. 소프트웨어 및 실험 방법

1) 실험 소프트웨어

본 연구에서는 3차원 정보를 얻기 위한 실험 장치와 소프트웨어를 개발했는데, 먼저 미리 알고 있는 좌표값과 영상을 통한 영상 획득 시스템에 대한 보정식을 세우고 이를 기준으로 하여 카메라 자체를 측정 장치로 한 3차원 좌표 측정하는 알고리즘을 개발하였다. 그림 3-3-16의 흐름도에 간략히 표시한 알고리즘은 크게 영상획득, 이치화, 세선화, 3차원 기준 좌표계로의 변환 등의 4단계로 구분하였다.

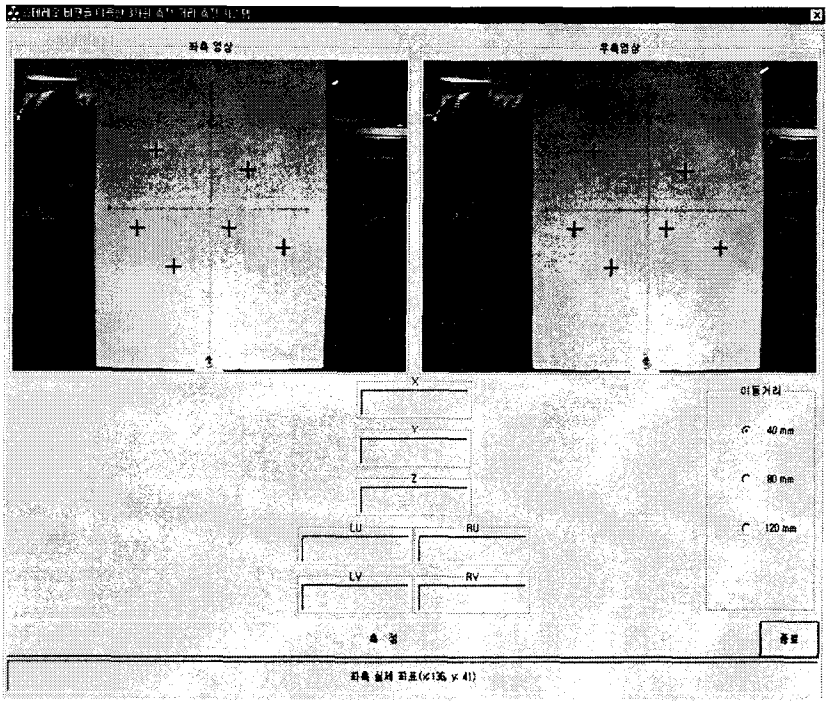

Fig. 3-3-13 Initial screen menu.

위의 그림은 스테레오 비전을 이용한 3차원 좌표 획득 소프트웨어의 초기화면이다. 본 실험에서는 그림에서 보여 지는 임의의 +형 무늬를 선택하게 되면 선택되어진 좌표의 일정 영역을 분리하여 이치화, 세선화 처리를 거쳐 정확한 + 무늬의 중심점을 찾게 된다. 그 뒤 그 중심점에 대한 실제 x, y, z 좌표를 3차원 정보 추출 알고리즘을 통하여 산출하게 하였다.

특히 알고자 하는 3차원 정보에 대하여 작업자가 미리 선택하므로 영상 획득, 영상처리, 정보 추출의 계산량 및 처리 시간을 대폭 줄이게 함으로 동일 영상에 대한 다중 좌표추출을 실시간으로 할 수 있도록 하였다.

그림을 통하여 처리과정을 설명하면 그림의 우측 화면에 CCD 카메라가 이동하기 전에 영상을 획득하여 표시하고, CCD 카메라를 40mm이동 시킨 후 제 2의 영상을 획득하여 이 두 화면에서 선택되어진 대상점을 영상처리를 통하여 2차

원 좌표인 (u, v) 좌표를 구하고 보정식을 통한 계산식에 의해 3차원 좌표를 추출한다.

그림 3-3-14은 목표점을 선택하였을 때 나타난 화면이다.

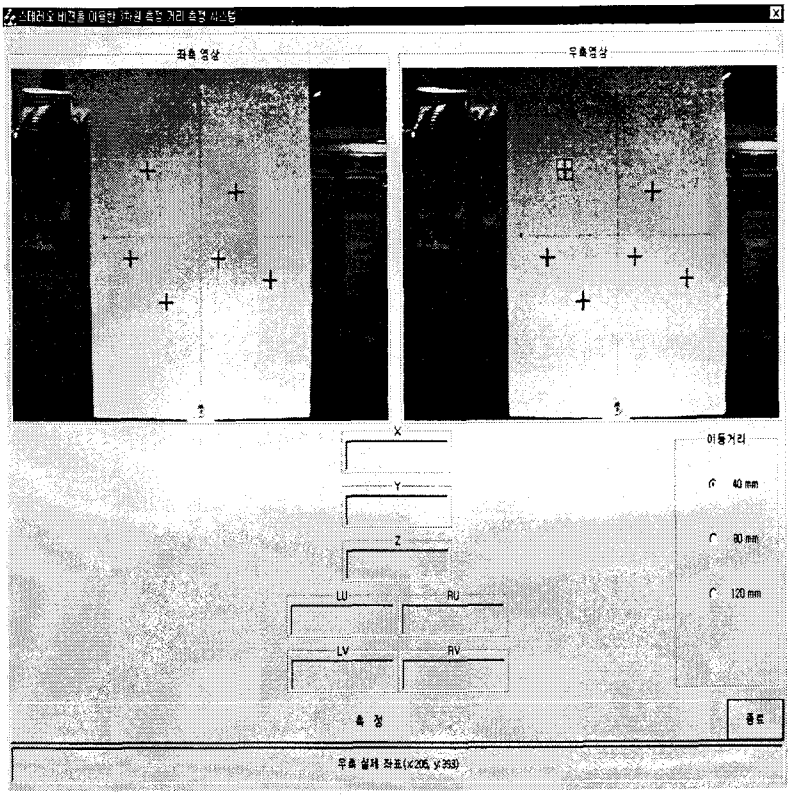

Fig. 3-3-14 Screen for Object Selection

그림 3-3-15은 이치화, 세선화 과정을 거쳐 3차원 좌표 추출한 화면이다.

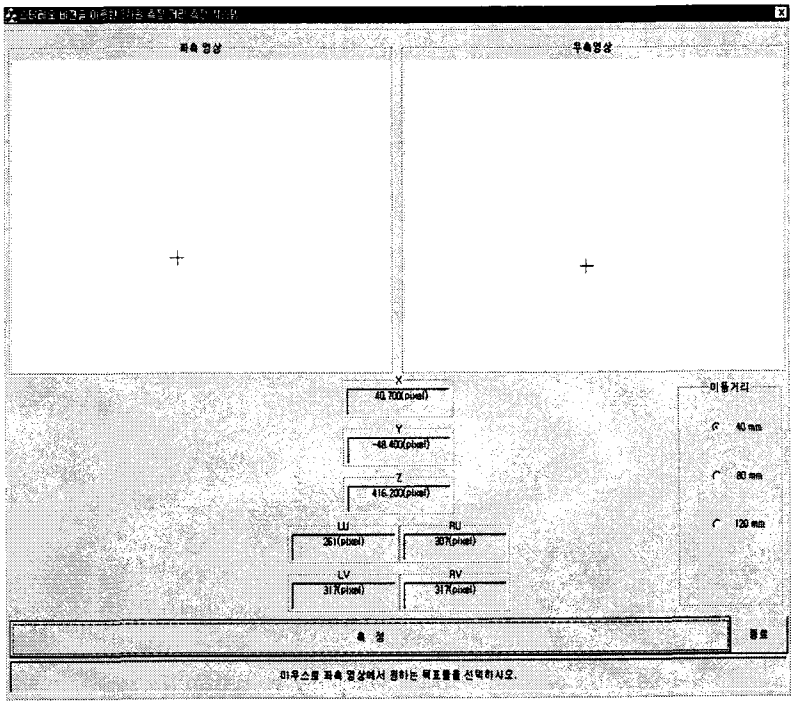

Fig. 3-3-15 Results screen.

아래의 그림은 스테레오 비전을 이용한 3차원 정보 추출을 위한 알고리즘에 대한 순서도이다.

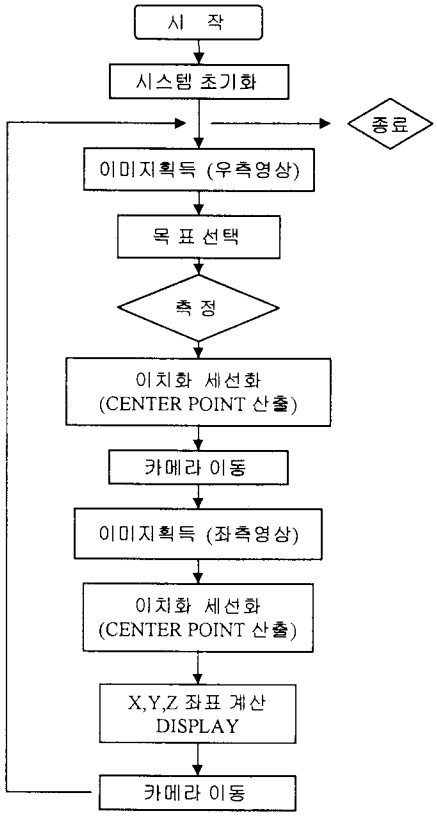

Fig. 3-3-16 Flow chart of 3D coordinate measurement algorithm.

2) 실험 방법

알고 있는 거리와 실제 x, y 좌표를 알고 있는 + 형 무늬가 있는 보정 차트를 통하여 3차원 보정정보를 카메라 렌즈로부터 530mm부터 1030mm까지를

100mm씩 증가시키며 영상을 획득하고, 좌우 이동은 40mm, 80mm, 120mm 이동하며 그룹별로 이 영상의 샘플 점들의 영상좌표와 기준 좌표계의 정보를 수집하여 좌우 이동 거리별 보정식을 적용하여 본 실험 시스템의 카메라 좌표 변환 행렬값을 구하였다.

그리고 임의의 좌표점이 있는 차트를 이용하여 3차원 정보추출을 수행하였으며, 실제 좌표값과 계산된 좌표값과의 비교를 통하여 실험을 수행하였다.

마. 결과 및 고찰

아래의 행렬값은 위 실험을 통한 카메라의 보정값이다.

Camera 1 $P = \begin{bmatrix} -13.2050 & 0.2240 & 6.0584 & 329.09656 \\ -0.2238 & -13.5145 & 4.5524 & 239.0965 \\ -0.0005 & 0.0006 & 0.0190 & 1.0 \end{bmatrix}$

Camera 2 $P = \begin{bmatrix} 20.9129 & -1.2480 & -10.3199 & -506.8957 \\ 0.8508 & 19.7546 & -7.9804 & 395.4855 \\ 0.0025 & -0.0040 & -0.0326 & 1.0 \end{bmatrix}$

Camera 3 $P = \begin{bmatrix} -33.6446 & 0.3939 & 16.5154 & 3182.9530 \\ 0.1757 & -35.8541 & 12.6212 & 203.6020 \\ 0.0018 & 0.0009 & 0.0524 & 1.0 \end{bmatrix}$

Camera 4 $P = \begin{bmatrix} -46.8242 & 1.7471 & 21.8443 & 5722.2133 \\ -1.6293 & -45.5356 & 16.7585 & 63.8888 \\ -0.0054 & 0.0041 & 0.1688 & 1.0 \end{bmatrix}$

위의 보정값은 위에서부터 카메라가 움직이지 않았을 때, 40mm 이동, 80mm 이동, 120mm 이동한 결과이다.

아래 표는 이 보정값을 통하여 80mm 이동 후 획득한 영상 정보에 적용시킨 결과이다.

<Table 3-3-3> Result of Calculation

Node	Original coordinate			Computed coordinate		
	X	Y	Z	X	Y	Z
1	50	-75	530	50.007765	-75.07791	530.45471
2	-50	50	630	-52.40182	50.917021	639.06933
3	-25	-25	730	-24.9993	-24.9552	729.01952
4	75	75	830	75.054372	74.800143	829.53916
5	-100	-50	930	-98.63514	-49.72315	924.55307
6	100	-25	1030	99.821667	-25.19822	1038.2456

위의 표에서와 같이 먼 거리의 영상 일수록 계산된 결과에 대한 오차가 커짐을 알 수 있었다. 이러한 오차들은 영상의 각 Pixel간의 표현력에서 주요하게 나타난 것으로 사료된다.

3. 터치스크린을 이용한 원격 국부(局部) 영상처리 시스템 개발

가. 서론

농작업에서의 영상처리 기술의 도입은 현재까지 많은 분야에서 활발하게 수행되고 있다. 적용 범위는 주로 선별 시스템에 많이 도입됐고, 수확 시스템, 공정 자동화 시스템 등 다양하게 수행되었다. 공정 자동화나 선별 시스템의 경우 광조건, 환경조건 등이 영상처리 시스템 도입에 용이하여 그 적용이 많았고, 그 결과도 어느 정도의 수준에 이르렀다. 그러나 수확 및 재배관리, 사육 관리 등과 같은 현장 작업의 경우 영상처리를 하기 위한 시스템 구현이 어려울 뿐 아니라 그 처리 정보가 많고, 제약사항이 많아 적용에 어려움이 있어 그 결과가 미흡한 실정이다.

그리고 기존의 영상 처리 시스템은 그 처리량으로 말미암아 주로 흑백 또는 그레이스케일(Gray scale)로 영상처리를 수행하였다. 그러나 현재에 이르러서는 프로세서의 성능 향상이 비약적으로 발전한 것과 같이 영상처리를 수행할 주변 하드웨어와 소프트웨어의 도움으로 컬러 영상처리 기술의 적용이 확대되고 있다. 컬러 영상처리의 경우 흑백의 영상처리와 달리 색상정보를 이용하므로 보다 정교한 프로세싱이 가능하고 다양한 결과를 얻을 수 있으나 그 정보량이 많고, 색상 정보의 상호 왜곡과 조명의 왜곡으로 인하여 실시간 영상처리를 하기가 어렵다.

본 연구에서는 작업자가 1차 교시(敎示)를 수행하고, 교시(敎示)한 좌표를 기준으로 일정 영역만 영상처리를 수행하게 함으로 표적이 되는 목표물을 오류 없이 찾을 수 있게 하였다. 그리고 1차 교시(敎示)를 통한 영상의 국부(局部) 처리가 가능하게 하였고, 이를 통하여 영상처리 알고리즘을 간단하게 구성할 수 있게 하였다. 작업자가 현장의 작업 환경 및 공정을 감안하여 교시(敎示)를 수행함으로 현재 영상처리 기술의 한계점을 보완할 수 있게 하였다.

선별과 같이 한정된 환경 하에서 프로세싱을 수행하는 경우와는 다르게 시설 환경에 투입되는 시스템과 같이 현장 적응 시스템일 경우 환경조건의 제약으로 오류를 범하기가 매우 쉽다. 그리고 광 조건의 가변성으로 인하여 시스템은 그 환경에 실시

간으로 적응하기 또한 어렵다.

다음으로 작업자의 1차 작업 교시(敎示)를 통하므로 시스템과 사용자간의 인터페이스가 매우 중요한데, 이는 과거의 DOS 시절과 현재의 Windows 환경을 통하여 쉽게 알 수 있다. 본 연구에서는 이러한 관점을 중심으로 터치스크린(Touch screen) 시스템을 도입하여 작업자가 직관적으로 작업 교시(敎示)를 수행하여 보다 효율적인 작업이 이루어 질 수 있도록 하였다.

다음으로, 작업자가 작업현장에서 시스템을 운용할 경우 다수의 시스템을 통제하기가 어렵고, 작업 환경 또한 열악하다. 그리고 작업 관련 지식과 경험을 바탕으로 하는 작업과 단순한 작업을 하는 경우 보통 노동력 투입은 대부분 단순작업에 치우치게 된다. 그러므로 작업자의 부가 가치를 높이기 위해서는 경험과 지식을 갖춘 전문가는 중앙 통제 시스템에서 전체를 총괄하고, 단순 작업의 경우 최소한의 인력을 통하여 작업을 수행함이 필요하다. 또한 작업 수행을 결정할 때 필요로 하는 통계적 자료나 전문 지식 또는 보조 자료를 재배관리 시스템에서 제공하게 함으로 보다 안정적인 재배관리가 이루질 수 있게 할 필요가 있다.

이러한 개념을 바탕으로 원격지에서 다수의 시스템을 중앙통제하고, 현장에서 중앙통제 명령을 시스템이 수행하게 함으로 위의 필요성을 충족시키게 하였다. 이 때 중앙 통제시스템은 각 단위별 작업 공정 및 누적 작업량, 시기 등과 같은 정보를 작업자에게 제공한다.

본 연구에서는 작업자의 1차 작업 교시(敎示)를 위한 터치스크린 시스템을 구현하고, 원격 명령을 수행하고, 시스템을 모니터링 할 수 있는 원격 무선 제어 시스템을 구현하였으며, 원격지에서 1차 교시(敎示)된 좌표 정보 및 작업 정보를 통하여 시스템을 운용할 수 있게 하였다. 그리고 교시(敎示)된 좌표 및 작업 정보를 바탕으로 작업 목표물의 영상을 획득하며, 그 획득된 영상을 처리하여 목표물의 형상, 자세, 위치를 추출하게 하는 영상처리 알고리즘을 개발하였다.

나. 배경 이론

1) 문헌개요

컬러 컴퓨터 시각을 이용하여 시설과 같은 자연광 내에서의 대상체 정보를 획득하기 위하여 근간에 많은 연구가 행해졌다. 그러나 환경적인 제약에 따라 그 효용성이 낮아 실용화 되기 위해서는 많은 연구와 시간이 필요하다고 사료된다.

손(1998)은 컬러 스테레오 영상을 이용하여 토마토 수확용 로봇을 개발하였는데, 대상체를 찾기 위해 인공광과, 컬러 이미지 정보의 R-Channel에서 Green Channel의 정보를 제거한 영상을 통하여 프로세싱을 수행하였다.

최(1999)등은 사과의 결점 판정을 위해 인공광과 컬러 영상을 이용하였으며, 이 때 컬러의 R, G, B 정보와 밝기정보를 갖는 영상을 이용하였고, 개체를 메시(Mesh)화 시킴으로 처리결과의 정밀도를 향상시켰다.

류(2000)는 식물공장용 포트묘의 보식 시스템을 위해 인공광과, 컬러 영상의 HSI 정보를 이용하여 재배홈통과 포트의 재질 변화에 독립된 영상처리 시스템을 개발하였다.

이(2001)는 사육장 내의 자연광 하에서 젖소의 체중 측정을 위한 영상처리 시스템을 개발하였다. 젖소와 배경 영상에서 이미지 빼기연산을 통하여 개체를 추출하였다.

손(2001)은 컬러의 HSI 및 YIQ 색 좌표계의 영상정보를 통해서 잎 및 육묘(育苗) 상자를 분리함으로로 육묘(育苗)의 결주 및 불량모 인식 시스템을 개발하였다.

최(2002)등은 컬러의 상호관계를 통하여 절화의 줄기 굵기, 길이, 개화 정도를 정량화 시키는 선별 시스템을 개발하였다.

2) 배경 이론

가) 터치스크린(Touchscreen)

터치스크린은 화면에 나타난 문자, 그림 및 모니터의 특정 위치에 손 또는 물체가 접촉하면 그 위치를 감지하고, 소프트웨어를 통해 터치스크린이 장착된 시스템에 그 위치를 알려주는 시스템을 말한다. 이러한 터치스크린을 이용한

시스템은 안내 시스템, 금융 시스템, 광고 시스템, 의료 장비, 자동화 장비 등과 같이 다양한 분야에서 사용자 인터페이스로 사용되고 있으며 그 범위가 확대되어 가고 있다. 이러한 터치스크린의 사용은 키보드나 마우스에 익숙하지 못한 사용자에게 부담감을 상당히 줄일 수 있다.

터치스크린은 이러한 직관적인 인터페이스를 제공하기도 하며, 외부 오염에 강하며, 긁힘, 충격에 강하여 실수로 옷이나 다른 물질로 터치하여도 이를 인식하지 않으므로 사용자 오류를 회피할 수 있다. 그리고 모니터와 완전 통합되어 있으므로 설치의 제약이 적다.
터치스크린 시스템의 구성은 크게 터치센서 패널, 제어기, 소프트웨어로 되어 있다. 터치스크린은 작업자의 접촉을 감지하는 센서의 역할을 하며, 제어기는 터치스크린의 종류에 따라 스크린을 제어하여 그 접촉 좌표를 컴퓨터에 전송하는 기능을 한다. 다음으로 소프트웨어는 전송된 좌표값을 컴퓨터 마우스 또는 키보드 신호와 연동 되게 하는 역할을 한다. 즉 터치스크린은 컴퓨터 마우스와 같은 역할을 수행하게 된다.
터치스크린의 국내외 연구개발 동향은 크게 유리패널형, 필름형으로 구분되며, 그 중 필름형은 압력식 터치스크린이라고 하며 5선 압력식, 4선 압력식, 그리고 디지털 매트릭스 방식이 있는데 향후 가장 많이 사용될 방법으로 예상된다.

(1) 종류 및 원리

(가) 접촉식 정전 용량식(Capacitive Overlay)

강화 유리 양면에 투명한 전도성 코팅을 한 터치스크린의 한 면 가장자리에 전압을 걸어 주고 반대 면에서 정전 용량을 검출하는 방식으로 사람의 정전 용량을 이용한다. 접촉면에 작업자의 접촉이 있을 때 정전 용량이 바뀜을 감지하여 접촉 위치를 알아낸다. 가벼운 접촉에도 잘 반응하고 분해능이 높지만 이물질이 터치 면에 있을 경우 오류를 발생시키는 단점이 있다.

(나) 압력식 저항막 방식(Resistive Overlay)

그림과 같이 유리나 투명 플라스틱 판위에 저항막을 코팅하고 그 위에 전도성 필름을 입힌 구조로 되어 있으며, 저항막과 전도성 필름 사이에는 절연 봉이 설치되어 상호 접촉이 되지 않게 되었다. 작업자의 접촉이 있으면 전도성 필름과 저항막이 압력에 의해 접촉되며 그로 인한 저항값의 변화를 읽어 접촉 좌표를 감지한다. 이 방식은 손가락이 아닌 볼펜이나 압력을 가할 수 있는 물체의 접촉도 감지할 수 있다. 그리고 접촉면이 오염되더라도 압력이 가해지지 않으므로 이물질로 인한 오류가 없는 장점이 있다. 종류로는 4선식 저항막 방식과 5선식 저항막 방식이 있다.

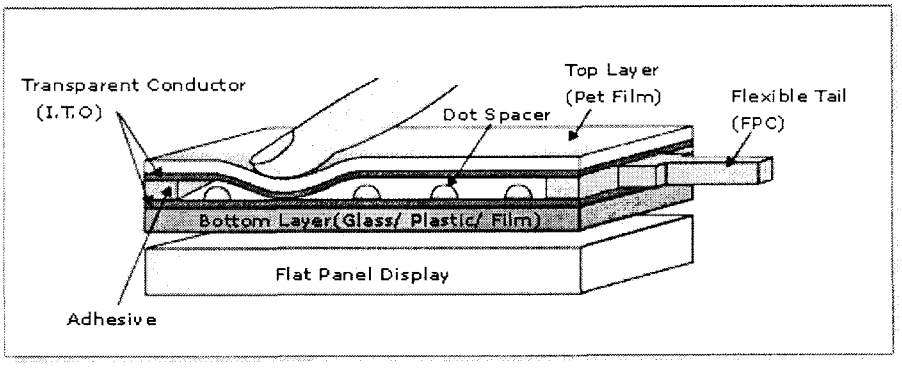

Fig. 3-3-17 Resistive overlay.

(다) 적외선 광 방식(Infrared beam)

접촉면이 없는 방식으로 스크린의 한쪽에서 적외선 광을 조사하며, 반대쪽에서 수광하는 구조의 방식이다. 이 방식은 접촉면이 없으므로 반영구적으로 사용할 수 있으나 외적인 환경 영향을 받기 쉽다.

(라) 표면 초음파 전도 방식(Surface acoustic wave)

그림 3-3-18과 같이 x축과 y축에 초음파를 보내고 받는 장치와 행과 열에

초음파 반사면을 두어 트랜스미터(Transmitter)에서부터 일정시간 간격으로 발사되는 초음파를 반사하여 최종 리시버에 도달하게 하는 구조이다. 스크린에 접촉이 가해지면 반사되는 초음파는 차단되므로 다음 반사면에 도달하지 못하는데 이때의 시간을 계산함으로 접촉면에 대한 좌표를 구할 수 있는 원리이다.

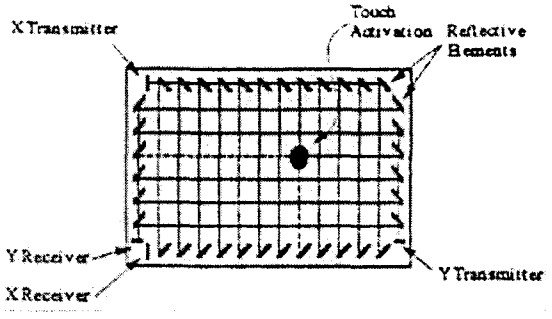

Fig. 3-3-18 Surface Acoustic Wave

(마) 유형별 장단점

<Table 3-3-4> Merits and Demerits of Type

종류	장점	단점
저항막 방식	높은 해상도 응답속도 빠름 장착이 쉽다.	파손위험이 있다. 광 투과율이 낮다.
정전용량 방식	높은 해상도 내구성 높음 응답속도 빠름	인체 외 감지 불가능 장착 어려움 광 투과율이 낮다.
초음파 방식	높은 해상도 높은 광 투과율 응답속도 빠름	오염에 대단히 약하다.
적외선 방식	높은 해상도 높은 광 투과율	부피가 크다. 오동작 가능 응답속도가 느리다.

(2) 5선식 저항막 방식의 구조와 동작원리

시설 내 작업환경 및 사용자 환경을 고려하면 고 신뢰성, 고 내구성이 요구되며, 외적인 오염으로부터 안정적인 동작을 해야 하므로 본 연구에서는 5선식 저항막 방식의 터치스크린을 사용하였다.

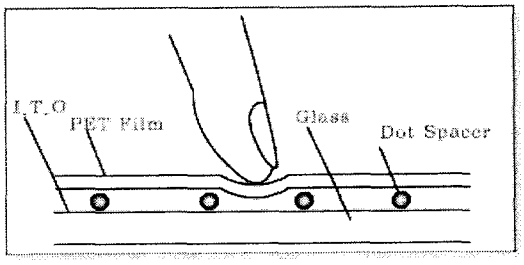

Fig. 3-3-19 Five line of method resistive overlay.

먼저 5선식 터치스크린의 구조를 보면 그림3-3-19과 같이 I.T.O(Indium tin oxide), 유리기판, 도트 스페이서(Dot Spacer), 폴리에스텔 커버로 이루어져 있으며, 유리 기판은 저항막이 코팅되어있고, 커버에는 전도 코팅이 되어있다. 도트 스페이서는 커버와 유리 기판이 접촉되지 않도록 유지시키는 기능을 한다.

그림3-3-20은 4선식과 5선식의 차이를 나타내는 그림이다. 4선식의 경우 상하, 좌우에 전극을 두고 커버는 각 저항값에 대한 전압만을 측정하는 원리를 갖는다.

5선식의 원리는 다음과 같다. 커버를 누르면 커버 안쪽에 균일하게 형성된 투명전극 Z는 눌린 위치에서 배면의 유리기판상에 균일하게 도포된 투명한 저항막과 접촉한다. 여기서 각각 x1, x2, y1, y2의 위치 즉 점 S가 눌려졌을 때 유리기판의 A, C전극에 전압 Vo를 가하고 B, D전극을 GND로 해서 X축 방향으로 전하를 주면, 투명전극 Z의 전위 Vz는 이 전위차 Vo를 접촉점 S로부터 X축의 양단까지의 거리비로 일정한 비율에 따라 고르게 나눈 값이 된다. 그러므로 투명전극 Z의 전압을 알면 x1과 x2의 비율을 알 수 있다. 같은 원리로 A, B에 Vo를 C, D에 GND를 가하면 Y축에 대한 좌표도 알

수 있다.

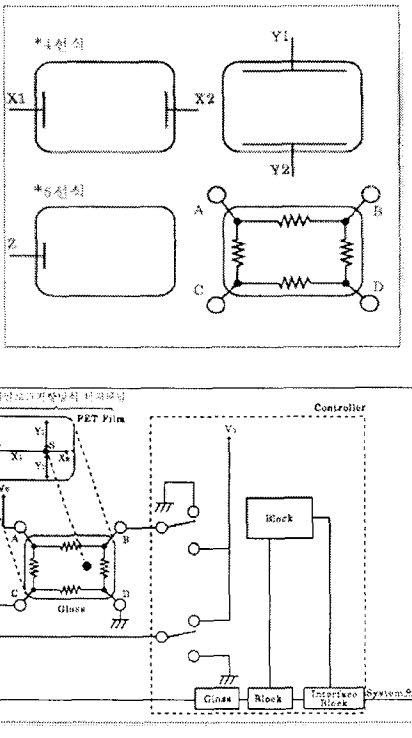

Fig. 3-3-20 Principal method of five line.

나) 원격 무선 제어 시스템

원격 제어 기술은 산업분야, 가전분야, 의료분야 등 많은 곳에 적용되어 작업자가 현장에서 하기 어렵거나 작업의 편의를 위해 많은 기여를 하고 있다.

특히 유선이 아닌 무선 원격 제어의 경우 설치의 제약을 극복할 수 있고, 다중 제어가 가능하여 하나의 통제 시스템을 통하여 다수의 지역 시스템을 관리 할 수 있다.

본 연구에서는 시설의 특수성을 감안하여 다중 무선 원격 제어 시스템을 구축하였고, 영상과 데이터 정보를 분리함으로 전송 속도 및 상호 간섭을 배제하였다.

영상 송수신 시스템은 NTSC 신호를 송수신하며, 데이터 송수신 시스템은

RS232 신호체계를 갖는 디지털 데이터를 송수신한다.

(1) 무선 송수신 시스템 기초 자료

(가) 국내 무선 시스템의 할당 주파수 대역 및 채널 수

<Table 3-3-5> Frequency Allocation of Domestic R/F Environment

용 도			할당주파수 대역	채널수/총대역		비 고
방송	표준 방송용		531-1602 ㎑	120 채널		
	FM 방송용		88.1-107.9 ㎒	100 채널		
	TV 방송용		57-85 ㎒	5채널	59 채널	
			177-213 ㎒	7채널		
			473-749 ㎒	47채널		
	위성 방송용		상향 14.5448-14.73628 ㎓ 하향 11.7466-11.93846 ㎓	6 채널		12㎓대 위성방송
항공 이동 업무	전용주파수대		2854-17965 ㎑ 2851-21997 ㎑ 3023-18023.5 ㎒			2-17 ㎒대 2-22 ㎒대 3-18 ㎒대
	ILS (계기착륙시스템용)		로칼라이자 108-111.9㎒ 그라이드패스 334.7-331.1㎒			
	VOR/DME 또는 VORTAC용		지상송신주파수 1025-1150 ㎒ 기상송신주파수 962-1213 ㎒ VOR ILS 108.1-117.9 ㎒	126 채널		
해상 이동 업무	전용주파수대		4,6,8,12,16,18/19,22,25/26 ㎒대	4 - 27.5 ㎒		
	무선전화용		4 ㎒대 6 ㎒대 8 ㎒대 12 ㎒대 16 ㎒대 18/19 ㎒대 22 ㎒대 25/26 ㎒대	29 채널 8 채널 37 채널 41 채널 56 채널 15 채널 53 채널 10 채널		
	무선전신용		4, 6, 8, 12, 16, 22, 25/26 ㎒대			채널폭 0.5 ㎑
	해상이동무선통신 업무용 주파수		선박국 156.025-157.425㎒ 해안국 156.375-162.025㎒	28 채널		
기타 무선국용	생활무선국용		26.965-27.405 ㎒	40 채널		
	코드 없는 전화기	3㎽이하	고정 장치 46.510-46.970㎒ 휴대 장치 49.695-49.970㎒	15 채널		
		10㎽ 이하	고정 장치 914.0125-914.9875㎒ 휴대 장치 959.0125-959.9875㎒	40 채널		
	간이 무선국용		422-423.9875 ㎒	160 채널		

(나) 주요 통신 서비스용 주파수 분배

<Table 3-3-6> Frequency Division of Major Telecommunication.

업무별	주파수대역(㎒)	전파형식	공중선전력	채 널 수	공고번호
이동전화	기지국:869-894 이동국:824-849	40KOF9X 1M32G7W	20W	832ch/20ch	공고제43호 ('91. 5. 9)
주파수 공용통신	기지국:851-899 이동국:806-821	16KOF(G)	75W	자가용:200ch 사업용:400ch	공고제43호 ('91. 5. 9)
	기지국:389.5-399.5 이동국:371.5-381.5	16KOF(G) 또 는 8K5OF(G)	75W	자가용:400ch (25㎑:200ch) 사업용:400ch (25㎑:200ch)	공고제21호 ('96. 2.26)
무 선 데 이 터 통신	기지국:938-940 이동국:898-900	8K50G7W	3W	사업용:160ch	공고제102호 ('97. 9.13)
개인 휴대통신	기지국:1840-1870 이동국:1750-1780	1M32G7W	20W이하	30ch	공고제160호 ('95.10.30)
무선호출용	162.43~164.33(26파) 167.25~169.15(27파) 322-328.6(264파)	16KOF(G)	70W이하 150W이하	317ch	공고제86호 ('92. 8.11)
발신전용 휴대전화	910 - 914	8OKOF9X	10㎽이하	40ch	공고제11호 ('96. 1. 26)

(다) 허가 및 신고가 필요 없는 무선국용 주파수

① 데이터 전송용 특정 소출력 무선기기

㉮ 용도, 주파수, 전파형식, 공중선 전력, 점유 주파수 대폭

다음의 표3-3-7에서 219.000(224.000)㎒ 및 424.7000㎒는 채널제어용 주파수이고 나머지 주파수는 통신용 주파수이고, 괄호 안의 주파수는 복신 또는 반복

신인 경우 송신(또는 수신)주파수대에 대응하는 수신(또는 송신)주파수이다.

<Table 3-3-7> Specification of the Data Transfer Frequency

장치명(용도)	주파수(㎒)	전파형식	공중선전력	점유주파수대폭
데이터전송	219.000(224.000) 219.025(224.025) 219.050(224.050) 219.075(224.075) 219.100(224.100) 219.125(224.125)	F(G)1D F(G)2D	10㎽이하	16㎑이하
	424.7000 424.7125 424.7250 424.7375 424.7500 424.7625 424.7750 424.7875 424.8000 424.8125 424.8250 424.8375 424.8500 424.8625 424.8750 424.8875 424.9000 424.9125 424.9250 424.9375 424.9500	F(G)1D F(G)2D	10㎽이하	8.5㎑이하

㉮ 발사 전파의 허용 편차

발사전파의 주파수 허용 편차는 다음과 같다.

200㎒대의 전파를 사용하는 것은 할당 주파수의 $\pm 7\times10^{-6}$이하 이며, 400㎒대의 전파를 사용하는 것은 할당 주파수의 $\pm 4\times10^{-6}$이하이다.

㉯ 반송파 감지시스템

반송파 감지 장치는 제 3의 무선 통신 시스템 간의 상호 간섭을 배제하기 위한 장치로 2㎶이상의 다른 특정 소출력 무선국의 전파를 수신한 경우에는 그 무선국의 발사 전파와 동일 주파수의 전파를 발사할 수 없게 하는 시스템이다.

㉲ 코드 식별 기억장치

데이터전송용 특정 소출력 무선기기는 다른 기기의 오동작을 방지하고 다른 기기의 신호에 의한 오동작을 일으키지 않도록 기기별 코드식별 기억장치를 갖추어야 한다.

② 영상 전송용 특정 소출력 무선기기

㉮ 용도, 주파수, 전파형식, 공중선전력, 점유주파수대폭

㉯ 발사 전파의 주파수 허용 편차

영상 전송 시스템에 있어서 발사 주파수 허용 편차는 50×10^{-6} 이하여야 한다.

<Table 3-3-8>Specification of Frequency for Image Transfer

장치명(용도)	주파수(㎒)	전파형식	공중선전력	점유주파수대폭
영상전송	2410.0 2430.0 2450.0 2470.0	A2F F2F A9W F9W	10㎽이하	16㎒이하

(2) 송수신 시스템

영상 송수신은 2.4㎓대의 주파수를 이용하며, 주로 NTSC나 PAL 방식의 아날로그 신호를 송수신한다. 현재에 들어서는 영상 압축 및 고속 무선 전송기술이 발달하여 디지털 영상 무선 송수신 시스템도 개발되었으나 가장 일반적인 아날로그 방식의 영상 송수신 시스템을 본 개발에 적용하였다.
다음으로 제어 데이터 송수신은 219㎒, 424.7㎒대의 주파수를 사용하며, 주로 FM 전송방식을 사용하고, 제어 시스템과 인터페이스가 좋게 하기 위하여 RS232C 규격을 따르는 입출력 인터페이스를 사용한다.

영상 및 데이터 송신 시스템은 최종 출력 신호가 영상, 오디오 또는 데이터

신호의 변조된 신호, 즉 고주파 신호가 출력되며, 수신 시스템은 이 고주파 신호를 다시 복조한 후 각 신호별 정보를 출력한다. 여기서 영상 송수신 시스템과 데이터 송수신 시스템을 비교하면 단지 정보의 대역폭(Band Width) 및 반송파(Carrier)의 주파수에서의 차이가 있고, 신호의 형태가 아날로그와 디지털의 차이가 있을 뿐 무선 송수신 자체의 원리는 같다.

(가) 송신기(Transmitter)

영상 송신기의 내부 블록도를 예를 들면 아래의 그림과 같이 출력 신호는 음성신호와 영상신호로 2가지를 혼합한 신호가 전송되는데, 먼저 음성신호를 좌우 각각 6.0㎒, 6.5㎒로 변조(Modulation)하고 영상 신호와 혼합기를 통해서 합한다. 그리고 VCO를 거쳐 2.4㎓로 FM 변조되고, 출력 증폭기에서 이 신호가 증폭되며, 안테나를 통해서 영상을 전송한다. 여기서 VCO는 Voltage Controlled Oscillator의 약자로 전압제어 발진기를 의미하며, 송신 시스템의 송신주파수와 수신시스템의 발진 주파수를 Synthesizer의 인가전압에 의해 안정되게 발진시키는 가변 주파수 발진회로 모듈(module)이며, PLL은 Phase Locked Loop의 약자로 송신 시스템의 불안정, 주변 환경의 불안정으로 인한 신호 주파수의 흔들림을 막아주어 정확한 주파수로 전송할 수 있게 하고, 또 특정 주파수로 주파수 가변의 역할을 한다.

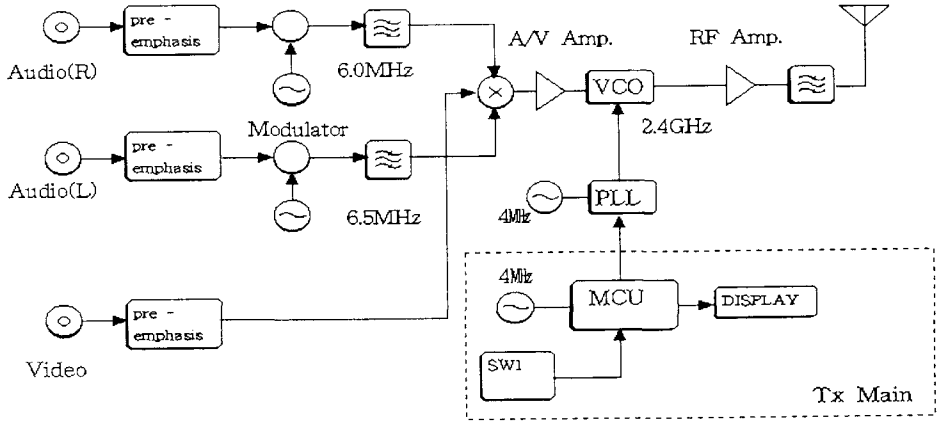

Fig. 3-3-21 Image Transmitter Block Diagram

(나) 수신기(Receiver)

영상수신기는 송신기의 변조기(Modulator)와 반대개념인 복조기(Demodulator)를 이용하여 송신된 2.4㎓의 신호를 영상과 음성신호로 복조한다.
그 구성은 먼저 안테나로부터 송신된 2.4㎓대의 신호를 수신하여 고주파 증폭 및 필터를 거친 후 PLL 및 VCO를 통하여 주파수 동기를 맞추고 고주파에서 IF/기저대역으로 주파수를 내려주는 주파수 하향 변환 믹서(Mixer)부를 통해서 채널을 분리한다. 이러한 방식을 슈퍼헤테로다인(Super Heterodyne)방식이라고 한다. 다음으로 복조기를 통해서 중간 주파수(IF: Intermediate frequency)를 영상신호와 음성신호를 복조하며, 그 신호에서 기저신호(Base band signal) 즉 영상 신호와 음성신호를 분리하여 각각 출력한다. 그림에서 MCU(Micro Controller Unit)부분은 채널 선택을 위한 PLL의 제어장치이다.

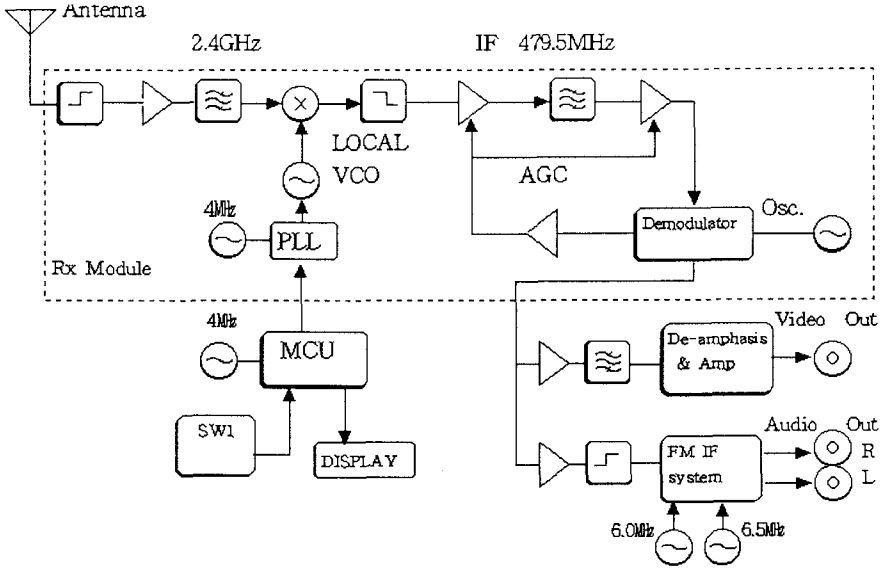

Fig. 3-3-22 Image receiver bock diagram.

다) 컬러 영상처리

(1) 컬러 영상 정보

컬러 영상정보를 인식하는 방법은 세 가지 변수로 나눌 수 있는데 색상(Hue), 채도(Saturation), 명도(Intensity)로 나눌 수 있다.

먼저 색상은 녹색이나 노랑색과 같이 색깔들을 의미하는데 이것은 빛의 파장에 기인한다. 430 ~ 480㎚의 파장은 청색이며, 노랑색은 570 ~ 600㎚의 범위에 걸쳐 있고, 610㎚ 이상의 파장은 적색으로 분류된다. 흑색, 회색, 그리고 흰색은 색깔을 가진다고는 하지만 여러 가지 파장대의 빛을 포함하고 있으므로 색상은 없다.

채도는 백색으로 희석되지 않은 색깔의 정도를 말한다. 순수한 색상에 첨가되는 자연 색깔의 양이 증가할수록 채도는 증가한다. 채도는 흔히 색깔이 얼마나 순수한가를 의미하기도 하는데 채도가 약한 색깔은 색이 바래거나 희미해져 보이며, 채도가 강한 색깔은 뚜렷하게 보인다. 붉은색은 채도가 가장 높은 색인 반면, 분홍색은 채도가 떨어지는 색이다. 순수한 색깔은 100%의 채도를 가지며 백색이 전혀 섞이지 않은 것이다. 흰빛과 순수한 색깔은 혼합하여 0%에서 100%의 채도를 만든다.

명도는 빛이 물체에 반사되어 느껴지는 강도이다. 이것은 백색에서부터 회색을 거쳐 흑색까지의 모든 범위를 의미한다. 그래서 흔히 이러한 범위를 명암도(gray level)라고 한다. 유사한 용어로 CRT와 같은 스스로 빛을 내는 물체의 감지 강도를 의미하는 명도(Brightness)가 있다. 대비(Contrast)는 영상의 가장 어두운 영역으로부터 가장 밝은 영역의 범위이다. 다음 식은 대비에 대한 표현식이다.

$$Contrast = \frac{Imax - Imin}{Imax + Imin}$$

여기서 Imax와 Imin은 명암도의 최대치와 최소치이다.
높은 대비를 가지는 영상들은 어두운 영역과 밝은 영역의 범위가 큰데, 좋은 대비를 가진 영상이 광도를 잘 표현한다고 할 수 있다.

영상의 대비가 증가하면 그 영상에서의 표현이 더욱 상세해지는데, 이것은 영상에서의 정보의 총량은 전혀 증가하지 않는 순수한 지각 작용이다. 우리의 지각 작용은 순수한 광도의 강도에 민감하기 보다는 광도의 대비에 더 민감하다.

예를 들어 커다란 사각형의 중간에 작은 사각형이 같은 광도를 가지고 있다고 해도 그 주변의 밝기에 따라 그 밝기가 동일하다는 느낌을 주지 않는데, 이러한 현상은 그 영역에서 배경의 밝기 강도에 크게 의존하기 때문이다.
서로 다른 광도가 인접해 있으면 Mach 밴드 효과라고 알려져 있는 현상이 발생하는데, 광도가 급격히 변화하는 데에 대해 경계부분을 강조하여 보이는 현상을 말한다. 밴드(1865년 Mach에 의해 발견)의 밝기는 오로지 하나만의 광도가 아닌 하나 이상의 광도에 의존하는 함수가 될 수 있다.

아래 그림은 서로 다른 명도를 갖는 바(Bar)에 대한 실제 명도값과 명도값 인식강도에 대한 그래프이다.

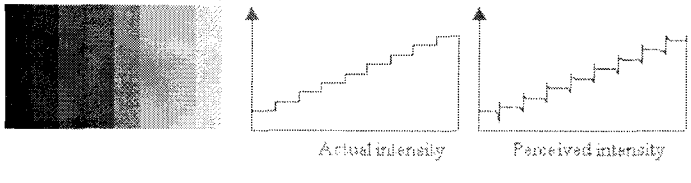

Fig. 3-3-23 Intensity

(2) 컬러(Color)의 표현

컬러 공간은 어떤 컬러와 다른 컬러들과의 관계를 표현하는 방법으로, 각각의 영상처리 시스템은 목적에 따라 서로 다른 컬러 공간을 사용한다. 출판

에 관련해서는 CMY 컬러 공간을 사용하며, 컬러 CRT 모니터와 컴퓨터 그래픽 시스템들은 RGB 컬러 공간을 사용한다. 그리고 색상, 채도, 명도를 각각 다루어야 하는 시스템들은 HSI컬러 공간을 사용한다.

(가) RGB 컬러공간

RGB 컬러공간은 모형은 아래 그림과 같이 각 축의 모서리가 3원색인 3차원 입방체로 표현될 수 있는데, 여기서 원점은 흑색이며, 반대쪽 꼭지점은 백색이다. 각 3원색의 색 표현은 256가지로 표현되며, RGB를 조합하여 자연색을 표현한다.

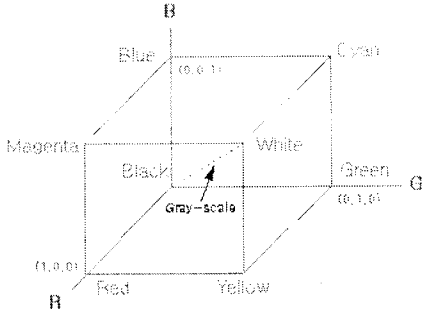

Fig. 3-3-24 RGB color space.

RGB 컬러공간은 컴퓨터 그래픽스에서 많이 응용되나 많은 영상처리 알고리즘들은 명암도만을 사용하므로 컬러 영상처리에서는 HSI가 더 적합하다.

다음 식은 RGB값을 명암도값으로 변환하는 NTSC(National Television System Committee) 표준식이다.

$$\text{명암도} = 0.288R + 0.587G + 0.114B$$

또는 $$\text{명암도} = 0.333R + 0.333G + 0.333B$$

(나) CMY/CMYK 컬러공간

CMY 컬러공간은 청록색(Cyan), 자홍색(Magenta), 그리고 노랑색(Yellow)으로 구성된다. 이것은 RGB 컬러공간과 반대의 공간이며 청록색, 자홍, 노랑은 빨강, 초록, 파랑 에 대한 각각의 보색(Complement)이다. 청록, 자홍, 노랑은 감할 수 있는 원색으로 이 원색들은 흰색으로부터 감산되어 원하는 색깔이 만들어진다. 청록색은 빨강색을 흡수하고, 자홍색은 초록색을, 그리고 노랑색은 파랑색을 흡수한 것이다. 그래서 영상에서 노랑색과 청록색을 증가시키거나 자홍색(초록의 보색)을 감소시켜 청록색을 증가시킨다.

RGB와 CMY가 보색의 관계에 있기 때문에 두 공간 사이의 변환은 쉽다. RGB에서 CMY로 변환하기 위해서는 다음 식과 같이 흰색에 대한 보수를 취한다.

CMY 컬러공간은 청록색(cyan), 자홍색(magenta), 그리고 노랑색(yellow)으로 구성된다. 이것은 RGB 컬러공간과 반대의 공간이며 청록색, 자홍, 노랑은 빨강, 초록, 파랑 각각의 보색(complement)이다. 청록, 자홍, 노랑은 감할 수 있는 원색으로 알려져 있다. 이 원색들은 흰색으로부터 감산되어 원하는 색깔이 만들어진다. 청록색은 빨강색을 흡수하고, 자홍색은 초록색을, 그리고 노랑색은 파랑색을 흡수한 것이다. 그래서 영상에서 노랑색과 청록색을 증가시키거나 자홍색(초록의 보색)을 감소시켜 청록색을 증가시킨다.

RGB와 CMY가 보색의 관계에 있기 때문에 두 공간 사이의 변환은 쉽다. RGB에서 CMY로 변환하기 위해서는 다음 식과 같이 흰색에 대한 보수를 취한다.

$$C = 1.0 - R$$
$$M = 1.0 - G$$
$$Y = 1.0 - B$$

다음은 CMY에서 RGB로 변환하는 경우이다.

$$R = 1.0 - C$$
$$G = 1.0 - M$$
$$B = 1.0 - Y$$

여기서 이 수식과 컬러공간들은 평준화되어 있다고 가정한다.
모든 값들은 0.0과 1.0 사이의 값이다.
다음 그림은 CMY와 RGB 컬러공간과의 상호 관계에 대한 그림이다.

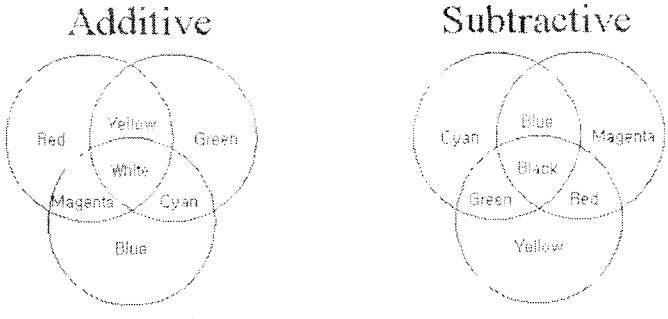

Fig. 3-3-25 Additive and Subtractive

CMYK 컬러공간은 청록, 자홍, 노랑의 세 가지 컬러에 검정색을 더한 색깔이 처리될 수 있는 컬러공간으로, 검정(black, K)은 다른 세 가지 컬러들의 조합에 의해 만들어 지는 것보다 순수한 검정색이 더욱 좋기 때문에 프린트 처리를 할 때 독자적으로 사용되며, 순수한 검정색은 뛰어난 대비를 제공한다. CMY에서 CMYK로 변환은 다음과 같다.

$$K = \min(C, M, Y)$$
$$C = C - K$$
$$M = M - K$$
$$Y = Y - K$$

그리고 CMYK에서 CMY로의 변환은 C, M, Y 요소에 각각 검은 요소를 더해주면 된다.

(다) HSI 컬러공간

색상, 채도 그리고 명도라는 세 가지 특성들이 컬러를 설명하는 데 사용되기 때문에 이것을 표현하는 컬러 모델을 HSI 컬러공간이라고 한다. HSI 컬러공간을 사용할 때는 어떤 컬러를 만들어 내기 위해서 몇 퍼센트의 파랑색이나 녹색이 필요한지 알 필요가 없다. 진한 빨강색을 분홍색으로 바꾸기 위해 단순히 채도를 조절하면 되며, 어두운 것을 밝게 하려면 명도를 조절하면 된다.

많은 영상처리 응용시스템이 HSI 컬러 모형을 사용한다. 특히 머신비전은 서로 다른 물체들의 컬러를 식별하는데 HSI 컬러공간을 주로 사용하는데, 히스토그램 연산, 명도 변환, 회선과 같은 영상처리 알고리즘들은 오직 영상의 명도에 대해서만 연산을 한다. 이러한 연산들은 영상이 HSI 컬러공간으로 되어 있는 것일수록 다루기가 더욱 쉽다.

HSI는 그림과 같이 원통 모양의 좌표계로 모형화 되어 있다.

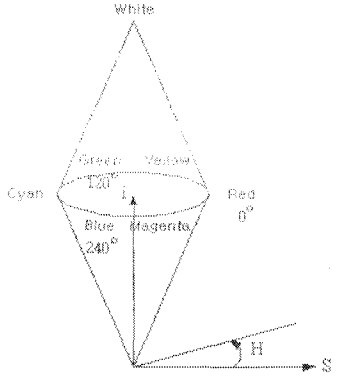

Fig. 3-3-26 HSI Color Space

색상은 0°에서 360°의 범위를 가진 각도로 표현되며, 채도는 0에서 1까지의 범위를 가지는 반지름에 해당하고, 명도는 z 축에 해당하는데 0일 때는 검정색을, 1일 때는 흰색을 나타낸다. S=0일 때, 컬러는 명도 I의 명암도만 가지

고 채도가 없는 회색조의 컬러를 의미한다. 그리고 S=1일 때 그 컬러는 원색을 갖는다고 할 수 있다.

다음으로 HSI 컬러공간에서의 색상의 조절은 각도에 따라 0°에서는 빨강색, 120°에서는 녹색, 240°에서는 파랑색, 그리고 360°에서는 다시 빨강색으로 변한다. I=0일 때, 컬러는 검정색이어서 H는 정의 되지 않는다. S=0일 때, 컬러는 명암도 등급이 된다. H는 이러한 경우에도 정의되지 않는다. I를 조절함으로써, 컬러는 어둡게 또는 밝게 될 수 있다. S=1로 유지하고 I를 조절하면, 그 컬러의 농도를 변화시킬 수 있다.

다음 수식은 RGB 공간에서 HSI로 변환하는 변환식이다.

$$H = \cos^{-1}\left[\frac{\frac{1}{2}[(R-G)+(R-B)]}{\sqrt{(R-G)^2+(R-B)(G-B)}}\right]$$

$$S = 1-\frac{3}{(RGB)}[\min(R,G,B)]$$

$$I = \frac{1}{3}(R+G+B)$$

(라) 이치화(Binarization)

이치영상으로의 변환을 위한 경계값의 선정은 영상처리에 의한 대상물의 외형측정에 가장 중요하다. 경계값 T를 자동으로 선정하는 방법으로는 히스토그램의 모드에 따른 분리도를 이용하는 방법, 단순 영상통계법, 모멘트 보존법, 최대 엔트로피 방법, 윈도우확장법 등이 있다.

대상물체에 따라 히스토그램 분포형태는 차이가 있으나 가장 일반적인 형태는 물체와 배경이 두 영역으로 구분되는 이중모드(Bi-mode)이다. 그밖에 히스토그램의 분포가 한 개의 봉우리를 이루는 단일모드(Uni-mode)와 3개 이상의 봉우리를 갖는 다중모드(Multi-mode)로 구분하게 된다.

윈도우 확장에 의한 이치화는 대상체의 배경부에 해당하는 윈도우의 크기를 조절하여 경계값 T를 추출하는 방법으로서 측정윈도우의 크기가 작을수록 효율적이며 알고리즘이 단순하고 처리속도가 빠른 장점을 가진다.

일반적으로 대상물과 배경의 구분이 명확한 영상에 대한 히스토그램 모드는 이중모드 형태를 갖는데, 이 경우 히스토그램 모드의 분리성을 이용하여 최대 깊이를 추출함으로써 영상을 이치화 할 수 있다. 대상체와 배경만을 분리하면 경계 내부의 세부적 특징추출은 불가능하다. 그러므로 목적하는 방향에 따라 이치화 방법을 복합시켜 사용해야한다.

① 윈도우 확장법

윈도우 확장에 의한 이치화는 측정윈도우를 확장하여 확장 전후의 히스토그램의 화소수 변화를 비교하는 방법이다. 두개의 히스토그램은 어느 밝기를 기준으로 모드의 변화가 생기게 되며 노이즈를 감안하여 적절한 경계값을 설정하여 이를 초과하는 화소밝기를 이치화를 기준값 T로 설정하게 된다. 확장전의 히스토그램의 높이(화소의 갯수)를 나타내는 식을 $y_s(x_g)$라 하고 확장된 히스토그램을 $y_l(x_g)$라 하면 이치화 기준값 T는 다음과 같은 식으로 정의된다.

$$\text{경계값} = \max(y_l(x_g) - y_s(x_g)) \quad , \quad 0 \le g \le 255$$

이 방법은 단순히 대상체와 배경만을 효과적으로 분리시켜 줄뿐 대상체 경계내부의 세부적 특징추출은 불가능하며 조명이 항상 일정하게 유지되어야 한다는 것이 전제조건이다. 따라서 조명상태가 수시로 바뀌게 되면 노이즈에 의한 영향으로 경계값을 제대로 찾을 수 없게 된다.

② P-tile 방법

이 방법은 배경이 밝고 대상이 어두울 때 적용하는 방법이다. 대상영역이 차

지하는 비율을 가정하고 경계값을 산출한다. 즉 이치화 된 영상에서 대상이 차지하는 영역이 최소한 (100-P)%를 포함하는 가장 높은 다치 영상으로 정의한다. 따라서 전체에서 P%를 포함하는 대상체의 이치화 경계값은 전체 화소수에서 최소한 P%를 허용하는 가장 큰 화소값으로 구할 수 있다.

(3) 최대 깊이 탐색법

최대 깊이 탐색법(Maximum Depth Searching Thresholding)은 이중모드의 히스토그램을 가진 물체와 배경을 분리하기 위한 방법으로 히스토그램에서 y축에 해당하는 화소수를 계산하여 이 값의 변화를 부호화하여 최대가 되는 두개의 변곡점을 구하는 방법이다. 여기서 변곡점은 최대 높이를 갖는 두 개의 정상점(Peak Point)을 나타내며 이것을 가지고 두 점 y_{1n} 과 y_{2n}을 연결하는 직선식을 구할 수 있다.

이치화 기준값 T는 y_{1n} 과 y_{2n}에 대응하는 x_{1g}에서 x_{2g} 까지 구간에서 이때의 화소밝기인 Gray값을 나타내는 x_g에 대해서, $y_n(x_g)$과 실제 히스토그램의 분포를 나타내는 $y_h(x_g)$ 사이에서 최대편차를 나타내는 x_g를 구하여 결정하게 된다. 직선식 $y_n(x_g)$의 기울기 a와 절편 b는 다음과 같이 구한다.

$$a = \frac{y_{2n} - y_{1n}}{x_{2g} - x_{1g}}$$

$$b = \frac{x_{2g} \cdot y_{1n} - x_{1g} \cdot y_{2n}}{x_{2g} - x_{1g}}$$

따라서 두 정상점을 연결하는 직선식 $y_n(x_g)$은 다음과 같이 쓸 수 있다.

$$y_n(x_g) = a \cdot x_g + b$$

이치화 경계값 T는 직선식 $y_n(x_g)$과 실제 히스토그램의 $y_h(x_g)$의 최대편차

를 나타내는 화소밝기 x_g를 기준으로 정하게 되며 x_g는 최대깊이 탐색에 의한 이치화 경계값을 나타낸다.

$$\text{경계값} = \max(y_n(x_g) - y_h(x_g))$$

(마) 영상완화

영상의 잡음성분은 체인코딩과 같은 탐색 알고리즘의 적용 시 프로세싱의 장애를 일으키며 측정정밀도를 저하시키므로 제거할 필요가 있다. 잡음은 불균일한 조명으로 인해 측정대상체의 경계부가 불규칙하게 나타나는 경우와 이물질에 의해 생기는 고립점이 대표적인 유형이다. 그 외에도 구멍, 모서리부의 훼손 등이 있다. 잡음성분의 제거를 위한 영상완화 알고리즘은 3×3 마스크를 설정한 후 논리 연산식을 이용하여 제거하였다.

각각의 잡음에 대한 논리 연산식은 표 2-2에 나타냈다. 논리 연산식 *M*은 참(true)이 되면 기준화소 ⓟ의 값은 1로 변환되어 화소를 보충하며 거짓(false)으로 판명되면 0이 되어 기준 화소는 제거된다.

<Table 3-3-9> Logic Operator by Each Noise Form

노이즈 형태	논리연산자 M
Hole & Notch	ⓟ||ⓑ&&ⓖ&&{ⓓ||ⓔ}||ⓓ&&ⓔ&&{ⓑ||ⓖ}
Isolated point & Bump	ⓟ&&{(ⓐ||ⓑ||ⓓ)&&(ⓔ||ⓖ||ⓗ)&&(ⓑ||ⓒ||ⓓ)&& (ⓓ||ⓕ||ⓖ)}
Uper right corner	!{ⓟ} &&{ⓓ&&ⓕ&&ⓖ}&& !{ⓐ||ⓑ||ⓒ||ⓔ||ⓗ}||ⓟ
Upper left corner	!{ⓟ} &&{ⓔ&&ⓖ&&ⓗ}&& !{ⓐ||ⓑ||ⓒ||ⓓ||ⓕ}||ⓟ
Lower right corner	!{ⓟ} &&{ⓐ&&ⓑ&&ⓓ}&& !{ⓒ||ⓔ||ⓕ||ⓖ||ⓗ}||ⓟ
Lower left corner	!{ⓟ}&&{ⓑ&&ⓒ&&ⓔ}&& !{ⓐ||ⓓ||ⓕ||ⓖ||ⓗ}||ⓟ

(바) 측도 설정

화소크기의 측도설정은 입력영상을 이치영상으로 변환하여 이미 길이를 알고 있는 샘플의 가로 및 세로의 화소수를 컴퓨터에 의해 계수하여 구한다. 측정 오차를 줄이기 위해 모눈종이에 등 간격의 점을 찍고 그 중심점을 카메라의 중심화소에 정확하게 일치시켰다. 그리고 영상을 획득하여 이치화 시키고 경계값에 따른 단위화소의 크기를 나타내는 CX(수평방향)와 CY(수직방향)는 샘플의 실제길이를 앞에서 측정한 평균화소수로 나누어서 구하게 된다.

$$CX = \frac{\text{샘플의 수평길이}(X)}{\text{화소 갯수}}$$

$$CY = \frac{\text{샘플의 수직길이}(Y)}{\text{화소 갯수}}$$

이 때 렌즈에 따라 카메라의 중심을 기준으로 영상 왜곡이 생길 수 있다.

다. 국부(局部) 영상처리 알고리즘 (Local image processing algorithm) 개발

1) 서론

보통 영상처리를 수행할 경우 전체 영상에 대하여 일괄적으로 수행하는 방법과, 다중 프레임(Frame)처리를 하는 방법, 한 프레임에 대한 부분적인 처리를 수행하는 방법이 있다. 기존의 머신비전(Machine Vision)에서 영상처리 알고리즘의 경우 영상처리 프로세싱 과정은 일반적으로 다음과 같다.

가) 영상획득
나) 전처리
다) 이미지 프로세싱
라) 정보획득

위의 순서로 획득된 영상에서 필요로 하는 정보를 획득하게 되는데, 이러한 일

련의 프로세싱이 컴퓨터에 의해서 자동으로 이루어진다.

만약 획득된 영상에서 원하고자 하는 목표가 없을 경우 프로세싱 오류를 나타내고 이것에 대한 오류 관리 프로세싱이 별도로 있어야 한다. 그리고 획득된 영상에서 목표가 있더라도 복잡한 영상정보로 인하여 목표물을 찾을 수 없을 경우에도 프로세싱 오류를 범할 수 있다. 그리고 자연광 하에서는 광 조건이 수시로 변화하게 되는데 이 경우에도 환경의 영향을 통하여 프로세싱 오류를 범한다. 이러한 경우 외에도 다양한 종류의 원인으로 프로세싱 오류가 발생된다. 그래서 대부분의 영상처리 알고리즘들은 특수한 목적, 한정된 환경, 한정된 목표를 가질 수밖에 없다. 그리고 위의 오류 보정이나, 전 영역에서의 프로세싱은 실시간 영상처리를 수행할 수 없게 하거나, 프로세싱이 불가능하게도 한다. 그러므로 이러한 문제점을 해결하기 위하여 반자동의 개념을 도입하여 작업자의 1차 교시(敎示)를 통한 국부(局部)영상처리 알고리즘을 제안한다.

본 연구에서 제안하는 국부(局部) 영상처리 알고리즘은 작업자의 1차 교시(敎示)에 의해서 프로세싱 위치가 결정되고, 그 위치를 중심으로 목표물에 대한 영상처리를 수행하는 방법이다. 이 방법과 기존의 영상처리 기법과의 차이는 작업자가 1차 교시(敎示)를 수행함으로 영상처리의 시작점을 설정하는 것이다. 이 1차 교시(敎示)를 통하여 프로세싱을 수행할 경우 기존의 알고리즘이 수행할 수 없는 자연광 상태나, 배경과 목표물의 구분이 명확하지 않은 경우 등에 적용 될 수 있으며, 작업영역을 전체 프레임의 일부로 한정 지을 수 있기 때문에 프로세싱 속도와 신뢰도를 확보할 수 있다.

2) 장치 및 방법

가) 실험 장치

(1) 시스템 개요

아래 그림은 본 연구에 사용된 실험 장치에 대한 블록도이다. 장치의 주요 구성은 카메라, 무선 영상 송/수신기, 프레임 그래버, 컴퓨터, 터치스크린을

장착한 모니터이다.

영상 정보 획득 과정을 보면 먼저 대상체에 대한 영상정보를 컬러 CCD 카메라를 통해 획득하고, 획득된 영상정보(NTSC)를 2.4㎓대의 무선 영상 전송 모듈(module)을 통해 무선 전송을 한다. 그리고 전송된 영상 정보를 무선 영상 수신 모듈(module)을 통해 수신하여 프레임 그래버를 통해 수신된 영상을 프레임 메모리에 저장한다. 다음으로 메모리에 저장된 영상을 터치스크린이 장착된 모니터를 통해 디스플레이 시킨다.

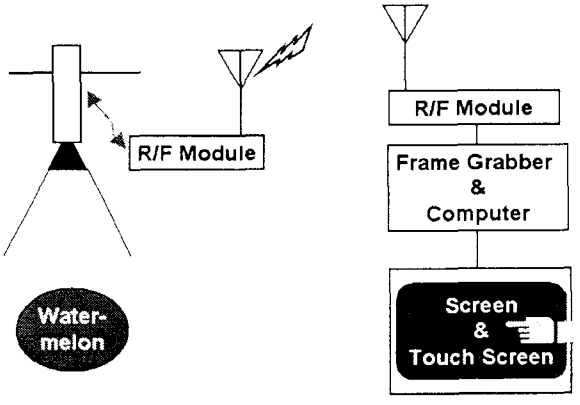

Fig. 3-3-27 R/F Module Summary Diagram

(2) 영상 입력부

영상 입력 카메라로 Super Fine사의 NTSC 출력의 컬러 카메라를 사용하였으며, 사용된 렌즈는 4.8mm 자동 초점식이다. 카메라의 화이트 밸런스는 수동으로 하였으며, 셔터 속도는 1/60초로 하였다.

(3) 무선 영상 송/수신기

카메라의 NTSC 신호를 허가받지 않고 사용할 수 있는 주파수대인 2.4㎓대

의 전자파로 무선 전송하기 위한 시스템으로 RF-Korea사의 시스템을 사용하였다. 송신기는 RTS-112이고, 수신기는 RRS-212이다. 주요 특징은 다음과 같다.

<Table 3-3-10> Specification of R/F Data Modem

Output Level	10mW	수신 Level	-85 dBm
주파수 대역	2.410~2.470GHz	영상 출력	1 Volt p-p
변조 방식	FM(Video/Audio)	음성 출력	1 Volt p-p
영상 입력	1 Volt p-p	잡음 (Noise)	3.5dB
음성 입력	1 Volt p-p	입력 전원	DC12V 500mA
영상입력 Impedance	75 ohms	소비 전력	Approx 4.8W
음성입력 Impedance	600 ohms		
입력 전원	DC 9V 700mA		
소비 전력	Approx 4.8W		

(4)프레임 그래버(Frame Grabber)

프레임 그래버는 아날로그 영상을 수신 받아 프레임 버퍼에 저장하고, 컴퓨터에서 영상처리를 수행하게 할 수 있는 인터페이스로 본 연구에서는 Matrox사의 Meteor2/4를 사용하였다. 다음은 Meteor2/4의 주요 제원이다.

(가) PCI, Compact PCI or PCI-Plus 방식
(나) NTSC, PAL, RS-170, CCIR 신호 입력
(다) 12개의 Video 입력과 Trigger 입력
(라) Grab한 영상을 실시간으로 Host System또는 VGA Memory로 전송
(마) 4MB Imaging Buffer
(바) 5 or 12 VDC power output
(사) RS-232 Serial port

(아) MJEPG codec module(lossy or lossless) 장착 가능

(자) MIL/ActiveMIL, MIL-Lite/ActiveMIL-Lite, Intellicam and Inspector 사용

(차) Windows 98/NT/2000

(5) 컴퓨터 및 프로그래밍 언어

<Table 3-3-11> Specification of IBM PC Exchange Type Computer

구 성	모 델
Type	IBM 686 PC Compactible
CPU	Pentium 800 MHz (intel)
RAM	256 MB
HDD	30 GB (quentum)
O / S	windows 2000 (Microsoft)
Programming Language	Visual Basic 6.0, Visual C 6.0

(6) 터치스크린(Touch Screen) 및 모니터

본 연구에서 사용된 터치스크린은 ELO Touch사의 AccuTouch이며 모니터는 15" LCD 모니터이다. 터치스크린 및 모니터의 주 사양은 다음과 같다.

(가) 크기 : 15"

(나) 해상도 : 4096 x 4096

(다) 광 투과도 : 90%

(라) 반응 시간 : 30㎳

(마) 수명 : 포인트 당 5000만 번 터치

(바) 방식 : 5선 압력 저항막식

(사) 컨트롤러 : RS-232C 데이터 전송방식

(아) 소프트웨어 : 마우스 에뮬레이션

(자) 모니터 형식 : 15" LCD 타입 (1024 x 768)

(7) 공시 재료

실험 환경에 맞추어 수확 시기의 수박을 대상으로 하였다. 영상 획득 환경은 오후 1시경, 맑은 날이며 대감 수박(흥농 종묘)을 대상으로 하였다.

Fig. 3-3-28 Daegam species(Heungnong Co.).

나) 실험 방법

(1) 원격 영상 획득(Remote Image Acquisition)

먼저 영상 처리 시스템과 5m ~ 30m 사이의 거리에 카메라와 2.4㎓대 FM 영상 송신장치를 설치하고, 영상처리 시스템에서는 영상 수신기와 프레임 그래버, 터치스크린, 모니터를 장착한 퍼스널 컴퓨터를 설치하였다. 카메라로부터 무선 입력되는 수박영상을 모니터에 나타나게 하여 영상 왜곡 등을

관찰하였다. 디스플레이 되는 영상을 Grab하여 Bitmap(*.BMP) image format으로 저장하고, 저장된 영상을 그림과 같이 화면에 출력하였다.

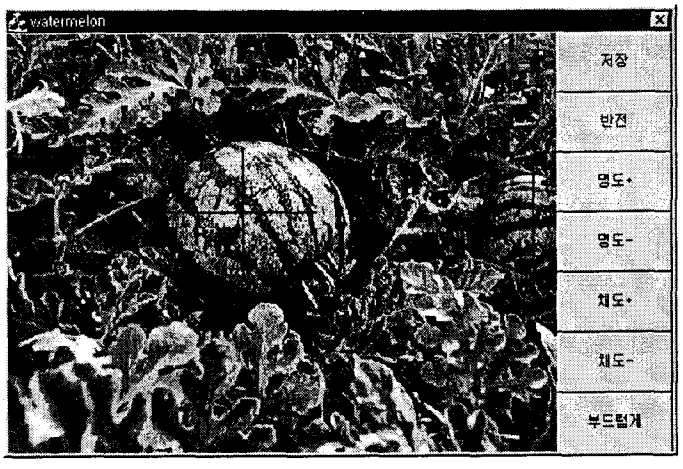

Fig. 3-3-29 Dialog screen menu.

화면에 나타난 영상에 대하여 터치스크린을 통하여 수박을 1차 교시(敎示)한 후 교시(敎示)점을 적색 + 마크를 하여 국부(局部) 영상처리 수행을 위한 시작점을 설정하였다.

(2) 국부(局部) 영상 처리 (Local Image Processing)

(가) 영역 설정

1차 교시(敎示)된 교시(敎示)점을 중앙점으로 설정하고, 200 x 200 Pixel 영역을 국부(局部) 영상 처리 영역으로 설정하였다. 이 때 국부(局部)영상처리 영역은 사용자가 임의로 조정할 수 있게 하였다.

(나) 전처리(Preprocessing)

설정된 영역에 대한 컬러 RGB 영상을 HSI 영상으로 변환 한 후 색도 정보와 명암도 정보를 이용하여 회색조 영상으로 변환하였다.

Fig. 3-3-30 Grey scale image.

(다) 필터링(Filtering)

회색조 영상에 대한 복잡도와 영상의 잡음을 제거하기 위하여 미디언(Median) 필터링을 수행하고, 히스토그램 평활화(Histogram Equalization)를 수행함으로 수박과 배경간의 대비를 극대화 시켰다. 그림 3-3-31는 미디언 필터를 수행 한 후 결과 영상이며, 그림 3-3-32는 히스토그램 평활화를 수행한 후의 결과영상이다.

Fig. 3-3-31 Median filtering image.

Fig. 3-3-32 Histogram equalization image.

(라) 외형 추출(Edge Extraction)

Sobel 필터로 영상의 각 경계선을 추출 하였고, 추출된 Gray scale 영상에 대한 이치화를 수행하여 수박 및 잎에 대한 외형을 추출하였다. 다음으로 영상

의 열기(Opening)와 닫기(Closing)를 수행하여 고립점과 잡음을 제거하였다. 그림 3-3-33은 경계를 추출하고, 고립점과 잡음을 제거한 영상을 반전시킨 영상이다.

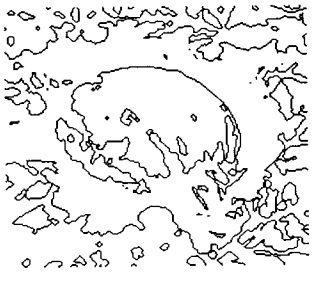

Fig. 3-3-33 Resulting image after external form abstraction.

(마) 영상 분할 및 외형 정보 추출

수박과 배경을 분할하기 위하여 교시(敎示)점을 중심으로 곡률이 가장 큰 경계선을 추출 한 후 추출된 경계선의 중심과 교시(敎示)점과의 평균 좌표를 중심으로 100 x 100 Pixel 영역을 분할하였다. 여기서 곡률이 가장 큰 경계선이 수박의 외형 일부이다. 분할된 영역을 기준으로 수박의 일부 외형 정보로부터 보간법을 이용하여 수박의 원형도(圓形度)를 알아내었다.

아래 그림은 교시(敎示)점을 중심으로 한 곡률 반경이 큰 선을 추출하고, 추출된 라인을 기준으로 반경의 중심을 찾아낸 화면이다.

Fig. 3-3-34 Image division and results of external form detection.

(3) 실험 방법

동일 대상체에 대하여 5분 간격으로 영상을 획득하고, 획득된 영상으로부터 국부(局部)영상처리를 수행 하였다. 이 과정을 5회 반복하여 수박의 반경과 중심좌표를 산출하고, 각 실험의 처리시간을 프레임 그래버의 소프트웨어로 계측하였다.

3) 결과 및 고찰

본 실험을 통하여 수시로 변화하는 광 환경 하에서 무선으로 원격지의 영상을 획득하고, 획득된 영상에서 1차적인 작업자 교시(敎示)를 통하여 국부(局部) 영상처리 알고리즘을 수행하였다.

실측한 수박의 반경은 249.5㎜ 이며, 픽셀 당 길이는 2.03㎜이고, 중심점(x′, y′)은 1차 교시(敎示)를 통해 설정된 영상처리 영역에서의 좌표이다. 설정된 영역에서의 1차 교시(敎示)점의 좌표는 (100, 100)이며, 교시(敎示)점은 처음 교시(敎示)한 좌표를 기준으로 영상처리를 수행하였다.

다음으로 처리 시간은 Matrox사의 Mil-Lite의 처리시간 계측을 위한 함수를 통하여 영상이 Grab 되는 순간부터 좌표 및 반경 결과가 나올 때까지의 시간이다. 수행된 결과는 다음과 같다.

<Table 3-3-12> Result of Experiments

횟수	반경(㎜)	중심점 (x′, y′),(Pixel, Pixel)	처리 시간 (Sec)
1	237.0	96, 99	0.32
2	250.5	100, 98	0.33
3	248.9	98,105	0.29
4	255.3	103, 96	0.33
5	250.8	99, 98	0.32

실험의 결과를 통하여 반경의 변화 폭은 다소 있었지만 작업 장치가 접근하는 좌

표인 중심점의 결과가 양호하게 나왔음을 알 수 있다. 그리고 처리 시간은 영상을 Grab하는 시간이 30%정도였고, 실제 영상 처리 시간은 100ms 이내였으므로 복합 영상에 대한 처리 시간이 매우 빠름을 알 수 있었다.

결과를 정리하면 다음과 같다. 작업자의 1차 교시(敎示)를 통하므로 작업자가 실수하지 않으면 교시(敎示)된 영역 내에는 반드시 목표물이 있으므로, 오류 처리 과정이 필요 없고, 교시(敎示) 좌표를 기준으로 한 부분 영역에서의 프로세싱을 수행하므로 수행 속도가 대폭 빨라졌다. 다음으로 교시(敎示)점이 보통 목표물의 중심점 부근이므로 이 교시(敎示)점을 가상 중심점으로 간주하고 영상처리 함으로 수박과 같은 원형 또는 특징이 있는 대상체의 경우 프로세싱 오차를 대폭 줄일 수 있었다.

다음으로 수시로 광조건 및 대상체 품종의 변하더라도 작업자의 1차 교시(敎示) 및 작업 환경설정을 즉시 변경할 수 있으므로 종래의 기계시각에 전적으로 의존한 시스템에 비해 적응성이 높다. 그리고 만약 잎 또는 다른 외적인 영향으로 인하여 프로세싱의 오류가 났을 경우에 작업에 따라 교시(敎示)점을 중심으로 작업을 수행하게 할 수 있어 영상처리 오류로 인한 작업성 저하가 없음을 알 수 있었다.

제 4 절 1 차 시작기 개발

1. 서론

중량과 재배관리, 수확 및 동시선별 적재작업을 위한 다목적 생력작업시스템 개발을 위하여 본 연구에서는 1차 시작기를 통하여 기초 실험 및 문제점을 파악하고자 하였다. 설계의 기준은 다음과 같이 하였다.

- 작업 체계에 따른 개별 작업 분석을 통한 모듈(module)형 장착 작업 장치를 설계 제작 하였다.
- 시설 내 작업 환경을 고려하여 적절한 구동형태, 기구 구조, 기구 규모를 산출하고 이를 기준으로 설계하였다.
- 개별 단위 장치 구축을 위한 기능 사양을 결정하고 그 결정된 요인을 분석함으로 개별 단위 장치를 설계 제작 하였다.
- 각 단위 장치의 요인 실험을 할 수 있는 공통된 구조를 갖는 시스템 설계에 중점을 두어 설계하였다.

본 연구에서 1차 시작기의 유효 작업 공간 설정은 소규모 시설 재배를 기준으로 시스템을 설계하였는데 그 내용은 다음과 같다.

- 작형 : 땅에서 눕혀서 재배 -> 1800mm × 90mm ~ 120mm
- 이랑 너비 : 1800mm
- 주당 너비 : 90mm ~ 120mm
- 기준 과경(果徑) : Ψ350mm
- 기준 과중 : 6kg 기준
- 유효 작업 공간 :
  - 이랑 방향 2000mm
  - 터널 방향 500mm
  - 높이 방향 450mm

위의 설계기준을 통하여 시작기를 제작하여 모듈(module)별 기능사양을 보완하였으며, 유선을 통한 직접 시스템 제어를 할 수 있는 제어기 및 소프트웨어를 개발하였다. 1차 시작기를 제작하기 전 단위 장치의 실험을 위한 간략화 된 시스템을 먼저

설계 제작하였으며, 이를 통하여 1차 시작기의 각 장치의 설계기준을 재정리하였다.

2. 시스템 구성

가. 실험용 정밀 매거진

실험용 정밀 매거진은 터치스크린을 통한 시스템 제어 실험과 매거진 성능 실험, 매거진 제어 시스템 구축을 위한 실험, 모듈(module)형 단위 작업 장치 실험, 대차 구동부를 제외한 간이 시작기 설계를 위하여 설계 제작 하였다.

시스템 구성은 크게 정밀 매거진부, 정밀 매거진 소구간 이송부, 서보 제어부, 시스템 총괄 제어부, 소프트웨어로 구성하였다.

1) 정밀 매거진

가) 기초 설계 요인

(1) 유효 작업 영역

정밀 매거진의 단위 작업 영역은 수박의 재식 거리를 반영하여 450㎜ × 450㎜ × 400㎜(이랑방향, 터널 방향, 높이, 1구간)로 구성 하였으며, 전체 작업 영역에 대한 구간은 이랑 방향으로 4구간으로 나누어 총 1800㎜의 이랑방향을 담당하도록 하였다. 즉 하나의 작업 구간은 터널방향으로 450㎜, 이랑방향으로 1800㎜임을 의미한다. 그리고 매거진의 정밀 위치제어와 매거진 자체를 4 구간으로 나누어 이동하는 것은 별도로 구성하였는데, 이는 정밀제어 구간이 클수록 고가의 요소 부품 및 제어장치가 투입되어야 하므로 이것을 피하기 위해서 독립하여 설계하였다. 또한 각 작업 특성에 맞게 매거진을 이동하게 함으로 불필요한 이동이 없게 하기 위함이다.

(2) 구동부

시스템 구동부는 크게 정밀 작업 영역에 대한 정밀 매거진 구동부와 정밀 매거진의 구간 이송을 위한 구동부, 터널 방향 이동을 위한 대차 이동부로 나누어 구성하였다.

먼저 정밀 작업 영역의 이동은 서보 제어를 통해 정밀 위치제어가 가능하도록 하였으며, 매거진의 소구간 이송의 경우 구간별 기준을 중심으로 작업이 행해지도록 하였으므로 그다지 정밀 위치제어가 필요치 않고, 향후 시설 내에서 독립 구동이 필요함으로 그 구동 전원을 축전지로 해야 함으로 감속기 부착형 직류 전동기 및 양쪽 가장자리의 한계를 위한 리밋 스위치를 사용하였다. 그리고 터널 방향 이동은 실험실 내에서 실험을 수행해야 함으로 설계 대상에서 제외 시켰다.

나) 시스템 구성

(1) 정밀 매거진부

정밀 매거진 부는 종단의 모듈(module)형 작업기를 작업위치로 정밀하고 신속하게 이동시켜 주어야 함으로 서보 제어 기구를 갖는 YAMAHA Robot사의 FXY-Series 형 X-Y 로봇을 이용하여 이 로봇을 본 시스템에 맞게 그림과 같이 재설계 및 제작을 하였다.

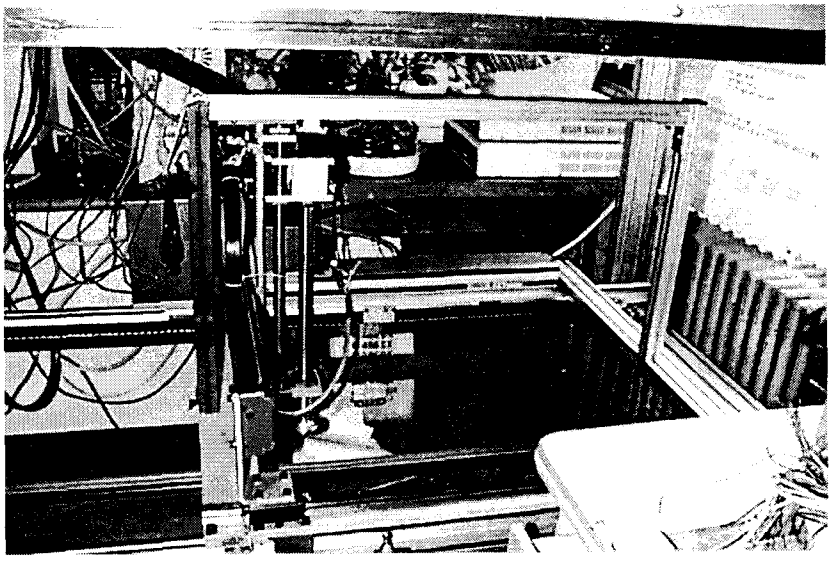

Fig. 3-4-1 Precision magazine XY robot and work transport system.

먼저 이랑의 X, Y 평면의 이동은 X 축 400W, Y 축 200W, 0.01㎜ 위치 정밀도를 갖는 DC Servo Motor를 장착한 YAMAHA FXY 로봇이 정밀 위치 이동을 하게 하였고, 모듈(module)형 단위 작업 장치를 장착할 수 있는 로봇 선단부의 Z축 이동과 회전운동을 할 수 있게 하기 위하여 그림과 같이 Tamagawa사의 2상 스테핑 모터를 사용하였다. 사용모터는 다음과 같다.

Z축: Japan Servo 사의 KP86SM2-501(22kg/cm),
Kuroda사의 lead 5㎜ 볼나사 감속
회전축: Oriental motor(japan), PK264A1(10kg/cm), 18:1 감속

먼저 Z축의 경우 볼나사를 이용한 정밀 이동 및 감속 이동을 할 수 있게 하였고, 회전축의 경우 유성치차 감속기를 이용하여 감속 회전을 하게 함으로 소요동력을 얻게 하였다. 그리고 스테핑 모터를 Z축과 회전축에 사용함으로 단순하게 위치 제어를 할 수 있게 하였다. 그리고 Z축의 경우 전력 공급이 중단되었을 경우 축 처짐이 발생하는데, 통상 이를 방지하게 하기 위하여 기계식 브

레이크를 사용한다. 그러므로 기구가 복잡해지기 쉽다. 스테핑 모터의 경우 홀딩 토크가 전원이 공급되지 않아도 기본적으로 크므로 이를 이용함으로 기구가 간단해진다.

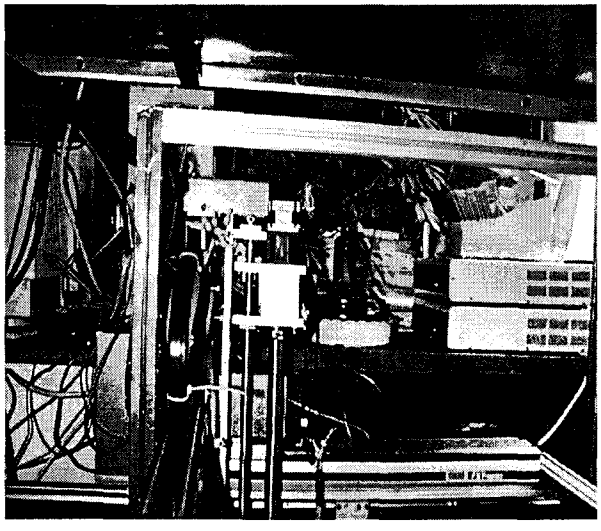

Fig. 3-4-2 Z Axis and rotation drive mechanism.

다음으로 각 축의 동작 범위는 다음과 같다.

- X axis: 450mm
- Y axis: 450mm
- Z axis: 400mm
- Roll : 300°

위의 각 축의 동작 범위에서 Z축의 동작 범위는 매거진의 작업 선단부와 이랑 사이의 높이와 모듈(module)형 작업 장치의 길이, 작업 대상체(수박)의 크기를 고려하여 산출하였다.

다음으로 정밀 매거진의 설계 가반 하중은 6 Kg/f로 하였는데 이는 수박의 대과종의 과중을 고려하여 결정하였다.

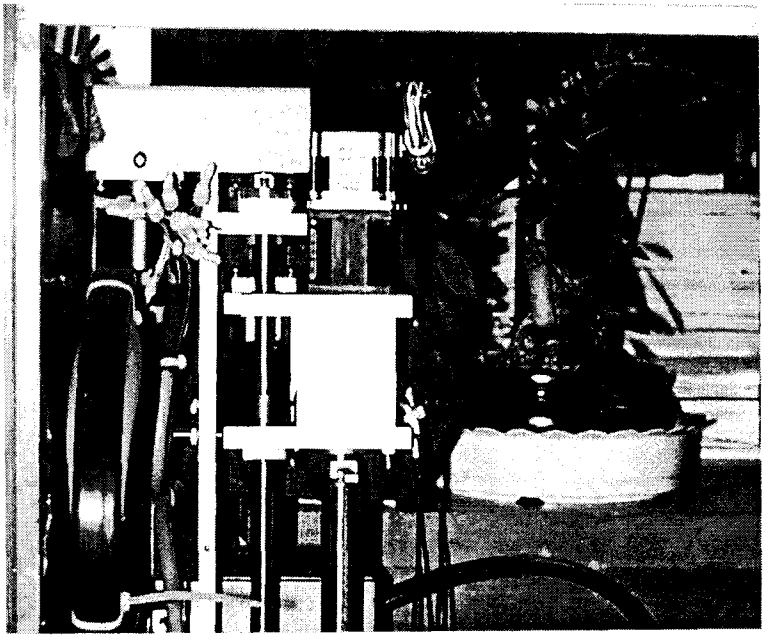

Fig. 3-4-3 Z Axis and rotation drive mechanism (detail).

(2) 정밀 매거진의 구간 이송부

구간 이송용 기구는 구동 모터 및 감속기, 동력 전달 기구로 구성되는데, 정밀 매거진 시스템 전체를 이랑 방향으로 4개로 나누어진 소구간 간의 이송을 하기위한 장치이다.

장치의 구동원으로는 성신 모터(S960GB-12, 60W, 감속비 30:1)의 감속기 장착형 AC 모터를 사용하였고, 동력 전달 기구는 장력 조절 기구를 포함한 타이밍 벨트 구동기구로 하였다. 이 때 모터는 향후 시스템 전체를 축전지를 통한 DC-AC 변환기로 구동할 것을 감안하여 결정하였다. 구간 이송기구의 급작스런 출발과 정지는 정밀 매거진에 영향을 줄 수 있다. 그래서 본 실험장치는 전기적으로 소프트 스타트와 정지를 할 수 있게 하였고, 타이밍벨트 동력전달기구를 통하여 다소 유연한 동작을 수행할 수 있도록 하였다.

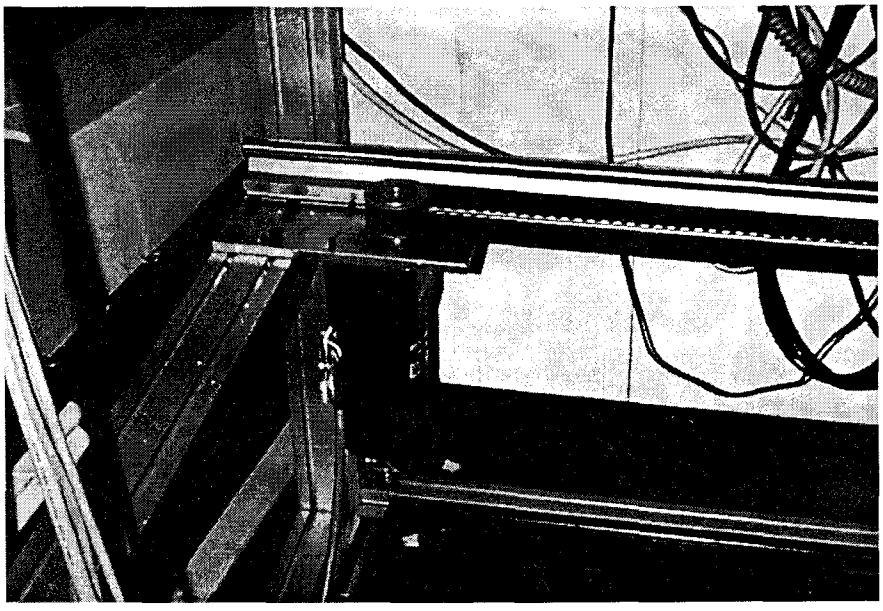

Fig. 3-4-4 DC motor for work magazine transfer and power transmission system.

(3) 서보 제어기 및 구동기, Digital I/O 장치

(가) 제어기 사양

서보 제어기는 YAMAHA 사의 DRC-2의 X, Y 복합 구동 및 제어기를 사용하였다. 이 제어기의 사양은 다음과 같다.

- 구동력 : X축 400W, Y축 200W
- 위치 감지 방식 : 리졸버(Resolver)형 위치 센서
- 모터 제어 : 전류제어, 가감속 제어, 프로파일(Profile) 제어
- 최고 속도 : 1000㎜/s, 가감속 0.2㎨
- 통신 방식 : RS232C를 통한 외부 제어기와의 통신
- 부가 장치 : 16CH Digital In, 16CH Digital Out
- 전원 : AC220V, 60Hz

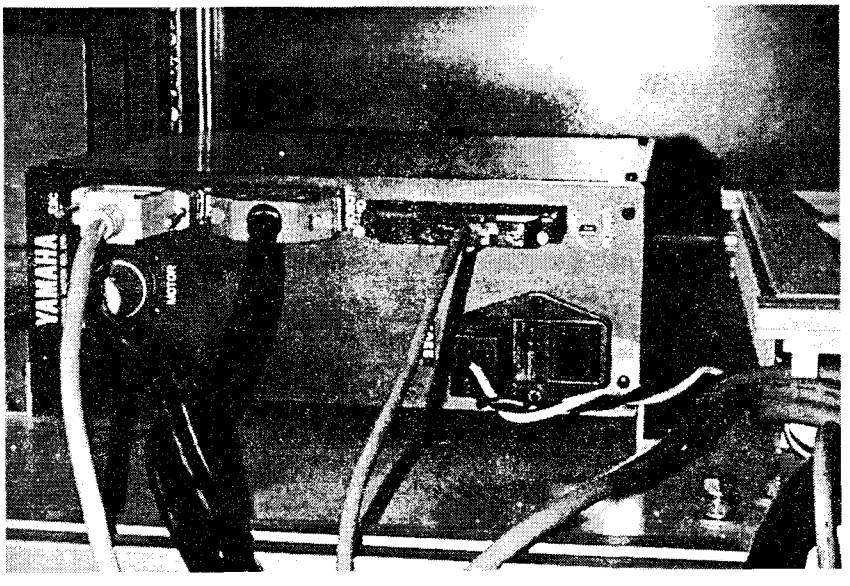

Fig. 3-4-5 Robot controller and digital I/O device.

(나) 외부 입출력

DRC-2의 디지털 입출력 포트를 이용하여 Z축과 회전축, 솔레노이드, 공압 기기, 진공 기기, 외부 센서 입력을 할 수 있도록 그림과 같이 시스템 I/O를 구성하였다. 스테핑 모터의 경우 정전류 구동방식의 구동 드라이버를 사용하였으며, 그 사양은 다음과 같다.

- 전원 : 12~50V DC 3A
- 구동 방식 : 정전류 초퍼(Chopper) 방식
- 구동 방법 : Pulse(속도), Signal(방향), Brake, Power Down
- 상 여자 방법 : 2상 여자, 1-2상 여자

솔레노이드 및 공압, 진공 기기들은 DRC-2의 디지털 출력을 직접 사용하여 릴레이(Relay)를 동작 시킴으로 구동되게 하였으며, 센서류는 디지털 입력단에 직결하였다.

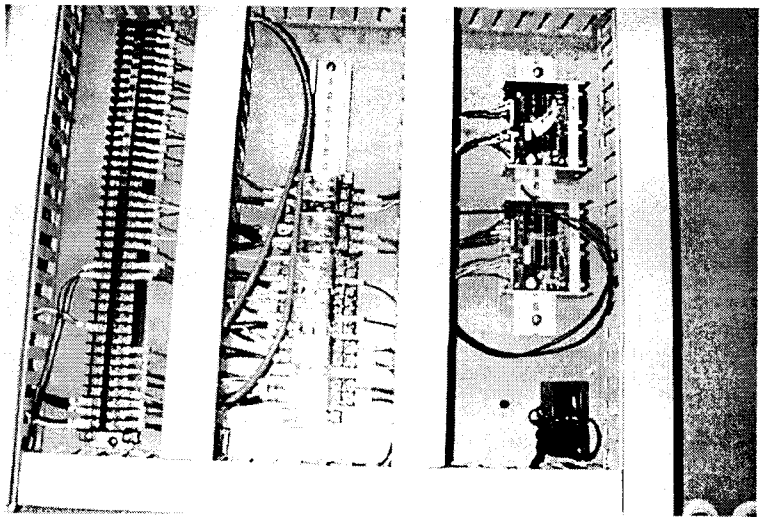

Fig. 3-4-6 Control part (step motor device, relay, terminal).

(다) 통신 프로토콜 (Communication Protocol)

실험장치와 주 제어기인 PC간의 명령 전송 체계는 다음 표와 같이 구성하였다. 명령 체계의 구성은 크게 출력 명령과 입력 명령으로 나누어지며, 제어기 시스템 파라미터 제어 및 프로파일 구동 경로 및 간단한 코드도 입력할 수 있게 하였다.

<Table 3-4-1> Ordering Form for RS 232 Communication

구분	명 령	기 능
출력	@ORG	로봇을 원점복귀시킴
	@RESET	로봇의 상태를 리셋(초기화) 시킴
	@SRVO NO	서보 모터 ON/OFF : NO -> 0 : OFF, 1 : ON
	@X+, @X-	로봇의 X 축 방향 1스탭 이동 : +: 증가, -: 감소 (1스탭: 0.01mm)
	@Y+, @Y-	로봇의 Y 축 방향 1스탭 이동 : +: 증가, -: 감소 (1스탭: 0.01mm)
	@MOVD X_POS,Y_POS,SPEED	로봇을 X_POS 및 Y_POS(원점에 대한 절대좌표, 실수)로 입력되어진 SPEED(0~100%, 100%->3000rpm)로 이동
	@DO OUTPUT_PORT_NO,NO	범용 디지털 출력 포트의 상태를 ON/OFF 시킴 NO-> 0 : OFF, 1 : ON
입력	@?XPOS	로봇의 원점에 대한 현재의 X축 방향의 절대좌표를 읽음
	@?YPOS	로봇의 원점에 대한 현재의 Y축 방향의 절대좌표를 읽음
	@?DI INPUT_PORT_NO	범용 디지털 입력포트의 상태를 읽음 리턴 값이 0이면 OFF, 1이면 ON
	@?DO OUTPUT_PORT_NO	범용 디지털 출력포트의 상태를 읽음 리턴 값이 0이면 동작정지, 1이면 동작 중
	@READ DIO	전체 디지털 입/출력 포트의 상태를 읽음

다) 시스템 소프트웨어

시스템 소프트웨어는 터치스크린을 통한 시스템 제어 실험과 매거진 성능 실험, 매거진 제어 시스템 구축을 위한 실험, 모듈(module)형 단위 작업 장치 실험을 위하여 개발하였다.

(1) 정밀 매거진 및 외부 입출력 제어

다음 그림은 정밀 매거진 제어 및 외부 입출력 제어를 위한 프로그램 실행 화면이다.

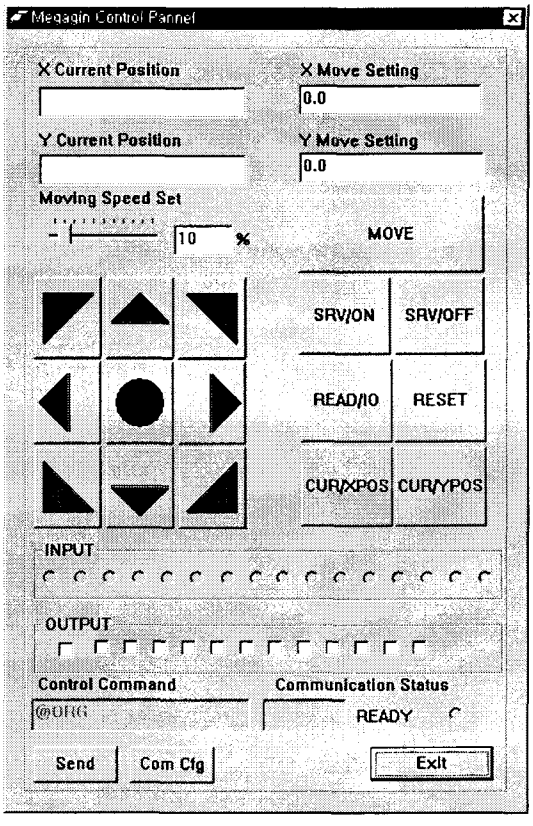

Fig. 3-4-7 Main control panel for precision magazine drive.

정밀 매거진 구동용 주 제어 패널은 XY 로봇의 속도, 최종 위치, 현재 위치, 외부 입, 출력 상황 또는 출력 등을 하나의 창에서 수행할 수 있게 하였다. 그리고 터치스크린을 통하여 기본 조정을 할 수 있도록 그래픽하게 제어 버튼을 구성함으로 터치스크린의 성능을 평가 할 수 있게 하였으며, 직접 명령 입력 창을 두어 서보 제어기의 내부 파라메타 및 복합 명령을 수행

할 수 있게 하여 시스템을 관리 할 수 있게 하였다.

프로그램 개발환경은 Windows 2000, Visual C++ 6.0을 사용하였다. 시스템 인터페이스는 윈도우 기반 GUI를 통한 판넬(Panel) 제어 방식을 채택하였으며, 초음파 방식 터치스크린을 사용함으로 직관적인 MMI(Man Machine Interface)를 구현하였다.

(2) 터치스크린을 이용한 영상 획득 및 알고리즘 실험

컬러 카메라로부터 입력 받는 영상 데이터(NTSC 신호)는 프레임 그래버를 통해서 주 제어 시스템으로 입력되어진다. 입력되어지는 데이터를 시스템 소프트웨어로 적기에 획득하고, 획득되어진 영상의 특정 영역을 설정함으로 국부(局部)영상 처리가 시작된다. 본 프로그램은 이러한 과정을 터치스크린을 통해서 일괄적으로 수행하게 하였다.
다음 그림은 영상획득 프로그램의 실행 화면이다.

Fig. 3-4-8 Image input window.

영상획득을 위한 영상처리 시스템은 다음과 같이 구성하였다.

- 주 제어기 : 컴퓨터(Intel 호환기종)

- 영상 처리 시스템 :
  ① 영상 획득 장치 : Coreco Co. Bandit Frame Grabber
  ② 카메라 : Color CCD Camera (Super Fine Color CV950)
  ③ 렌즈 : 4.8mm 자동 초점식 렌즈
- 터치 패널형 작업 지시 시스템 :
  ① 종류 : 14 " 초음파 방식 터치스크린(ELO touch Co. IntelliTouch)
  ② 방식 : 초음파식, Bus Controller 장착형

나. 작업기 프레임(Frame) 및 모듈(module)형 작업기 장착용 정밀 작업 매거진

1) 작업기 프레임

작업기 프레임은 시설 내에서 정밀 매거진의 기초(Base)로 매거진의 정밀 이동을 보장할 수 있는 구조이며, 수확 및 다양한 작업기의 부속 장비를 장착할 수 있도록 강건성이 확보되어야 한다. 그래서 당초 무한궤도(無限軌道)형 구동 시스템을 갖춘 프레임을 설계하였으나, 시설의 환경 및 주행 안정성과 작업 안정성을 위하여 차체가 시설의 고랑에 설치한 레일 위를 주행하는 방식으로 그림과 같이 재설계하였다.

차체의 전체 외형은 갠트리(Gantry) 형태로 설계하여 기계적으로 안정성을 도모 하였으며, 프레임의 강도를 갖게 하기 위하여 2층 구조의 보강프레임을 사용하여 작업자 및 정밀 매거진이 프레임 및 수확물 수납공간, 제어 시스템, 구동 시스템, 모듈(module)형 작업 시스템 및 부속기구를 그 위에 장착될 수 있도록 구성하였다.

레일구동부와 정밀 매거진, 작업자 공간은 제외한 실제 프레임의 설계 외형 치수는 2000㎜ × 2000㎜ × 645㎜(W × D × H)이다. 설계 외형 치수의 산출 근거는 소규모 시설에서의 이랑 너비에 맞추어 프레임의 폭을 결정하였으며, 프레임의 깊이는 정밀 매거진의 외형과 수확물 수납을 위한 공간을 고려하였다. 그리고 프레임의 높이는 레일과 이랑 사이의 높이 차이와 과경(果徑) 그리고 선단

작업부 치수, Z 축 유효 이동거리를 고려하여 결정하였다.

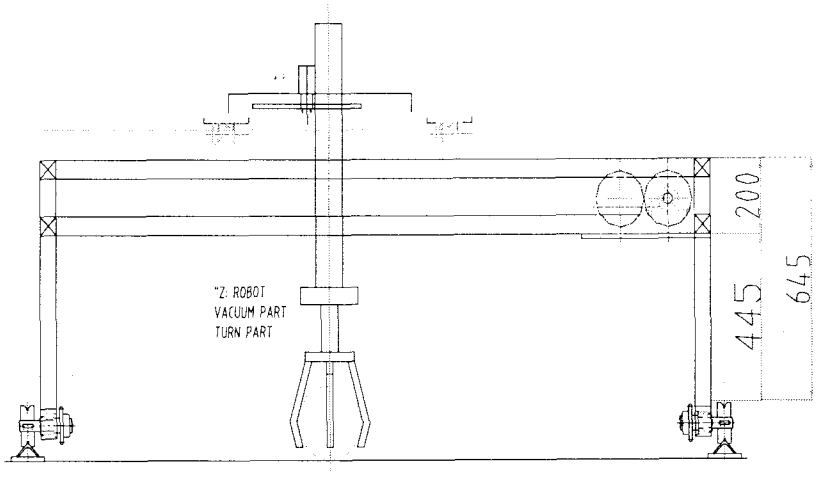

Fig. 3-4-9 Drawing of system frame (front view).

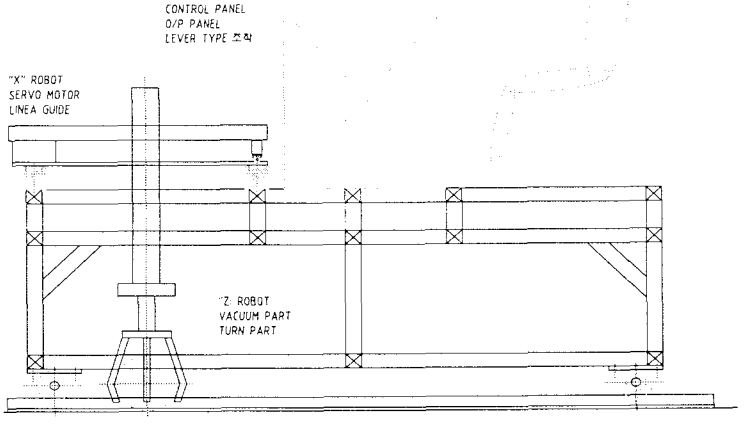

Fig. 3-4-10 Drawing of system frame (side view).

시스템이 진행하는 방향을 기준으로 그림과 같이 전반부에 정밀 매거진을 탑재하였고, 후반부의 한쪽은 시스템 제어장치, 차체 구동 시스템, 작업 제어 판넬 및 작업의자를 탑재하였으며, 반대쪽은 수확물 수납공간 또는 모듈(module)형 작업 시스템의 부속 기구물을 장착할 수 있게 하였다.
다음 그림은 실제 제작된 시작 1호기이다.

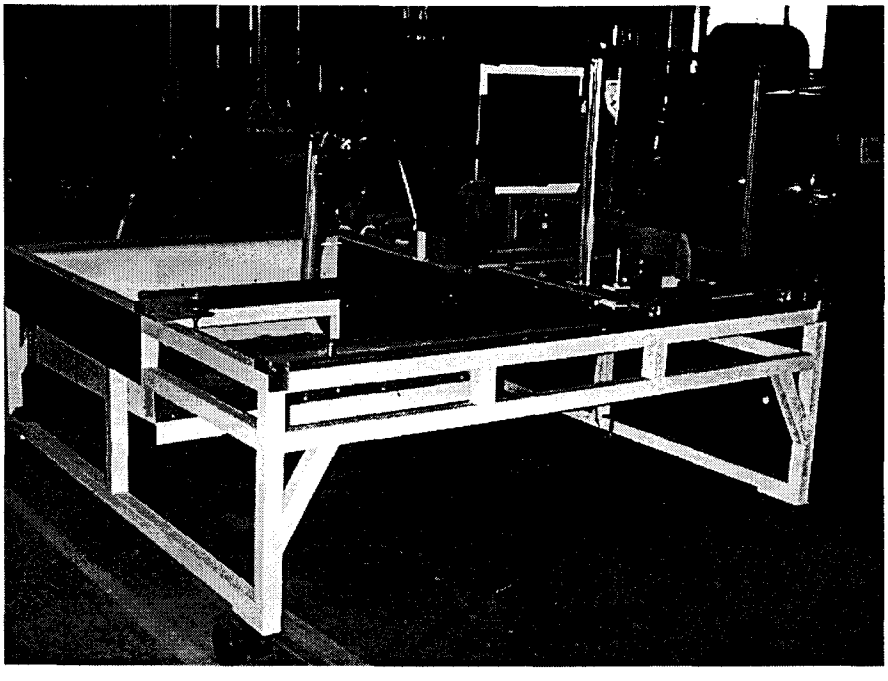

Fig. 3-4-11 The 1st prototype of cultivation management system.

2) 대차 구동 시스템

대차 구동 시스템은 그림과 같이 ∧형강을 레일로 사용한 레일 구동형으로 설계하였다. 구동 바퀴는 기어 감속기가 장착된 1㎾의 AC 모터의 동력을 체인을 통하여 전달하게 하였다. 이 때 대차 구동용 AC 모터는 인버터를 통해서 속도 및 정/역 회전이 제어되며, 체인(Chain)은 장력 조절 기구를 통하여 장력이 조정되게 하였다.

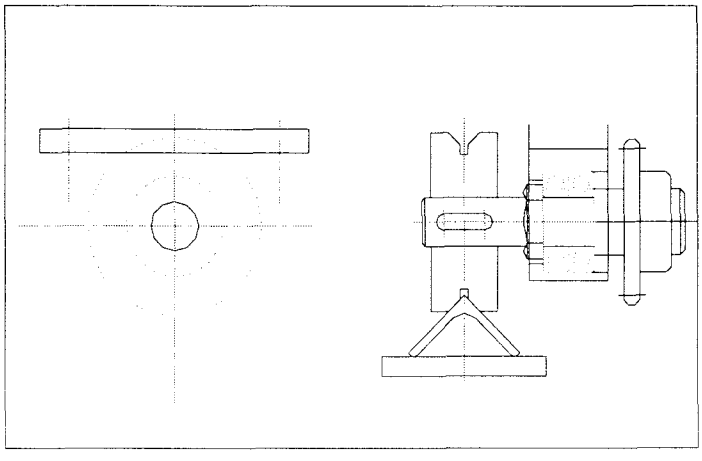

Fig. 3-4-12 Drawing of rail traveling unit.

아래 사진은 레일 주행형 대차 바퀴 기구이다. 여기서 리밋 스위치를 통하여 터널 방향으로 주행 한계(走行限界)를 감지하게 하였다.

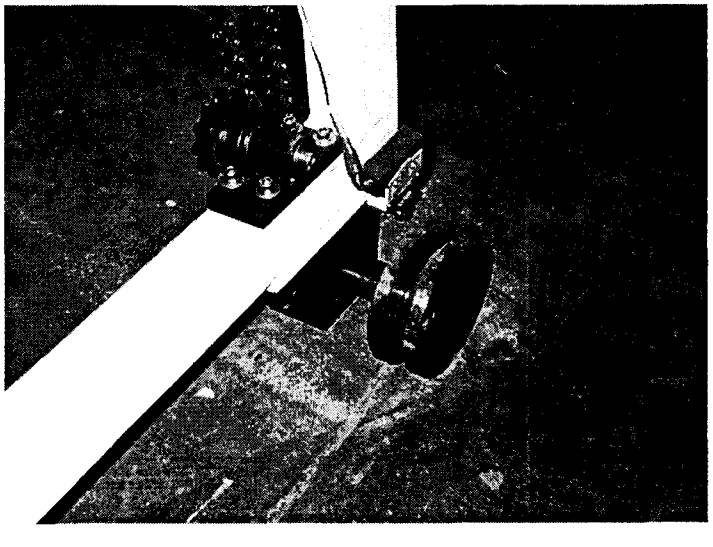

Fig. 3-4-13 Rail drive wheel.

대차의 주행은 1kW 기어 감속기 및 브레이크가 장착된 3상 AC 모터를 사용하였는

데, 그 구동은 그림과 같이 인버터를 통해 동력이 제어되게 하였다.

속도제어, 가/감속량의 조절은 작업자가 직접 파라미터를 조절을 통하여 변경시킬 수 있게 함으로 시스템 결함을 통하여 대차 주행의 안정성을 기하였다. 주 제어기에서는 모터의 정/역회전과 정지를 제어 할 수 있는데, 전/후 작업 한계에 대한 리밋은 인버터에 직결하여 주 제어기가 통제하지 않아도 자주적으로 정지할 수 있게 하였다. 단 리밋의 상황은 주 제어기가 알 수 있게 하여 작업자가 비상 상황에 대처할 수 있게 하였다.

다음으로 터널 방향의 작업 구간 이동은 구간별 마킹(Marking)을 인식할 수 있도록 리밋 센서를 장착하여 센서의 감지를 통하여 이동하게 하였다. 주행 중 구간 식별용 마킹이 감지되었을 때 인버터를 통해 감속 정지하게 하였다. 단 사용자가 마킹을 무시하고 주행하도록 명령하면 마킹에 의해서 정지하지 않고 연속으로 주행하며, 사용자의 정지 명령을 통하여 정지하게 하였다.

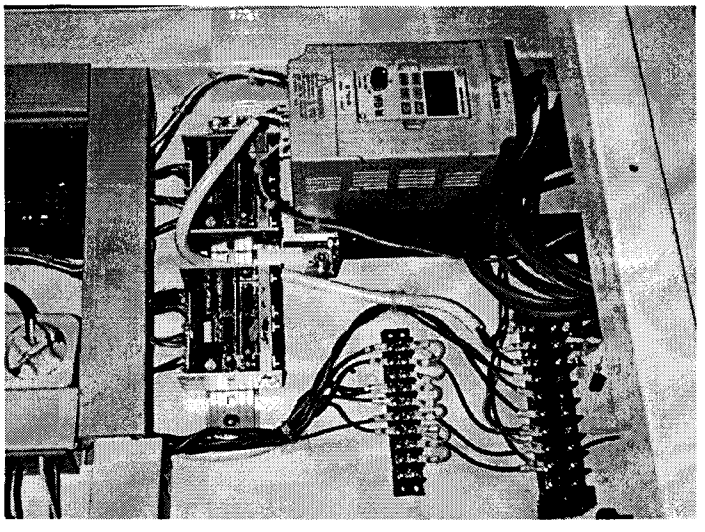

Fig. 3-4-14 Inverter system for cart drive.

구동 인버터는 DELTA사의 VFDO0723A 1㎾ 3상 인버터를 사용하였다.

3) 모듈(module)형 작업기 장착용 정밀 작업 매거진 및 부속 기구

모듈(module)형 작업기를 장착할 수 있는 정밀 작업 매거진의 개략도는 아래 그림과 같이 갠트리타입이며, X축 및 Y축 구동은 DC 서보 모터를 사용하였으며, Z축과 회전축은 스테핑 모터를 사용하였다. 사용 재원은 실험용 정밀매거진과 동일하며, 이 매거진을 차체의 전반부에 설치하여 운용하게 하였다.

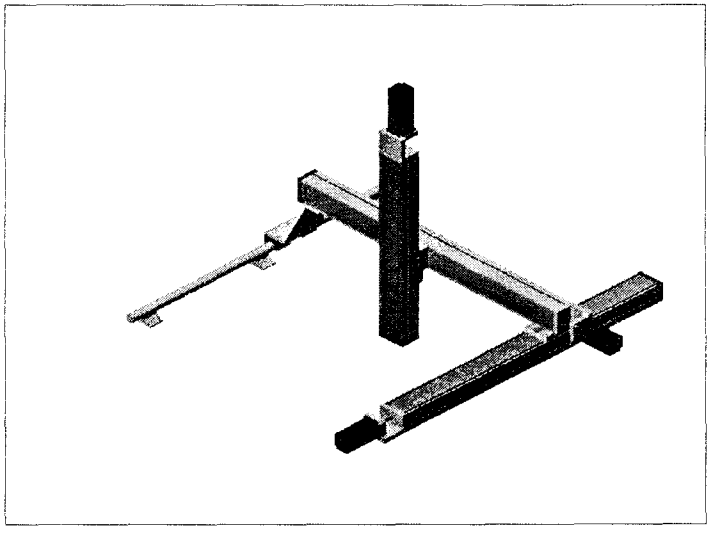

Fig. 3-4-15 3D drawing of precision magazine.

정밀 매거진 이송부의 재원은 실험 시작기와 동일하며, 이송 구간을 결정하는 검출 기구를 매 이송 구간에 설치하였다. 여기서 이 검출 기구는 이송 구간의 변경을 할 수 있도록 조절 기구를 설치하였다.

정밀 매거진의 분리를 용이하게 하기 위하여 시스템 제어기와 매거진 간의 전원, 신호선, 동력선, 공압선, 진공선 등을 분리할 수 있도록 중간 분리 기구를 두었으며, 직선 운동으로 인하여 이러한 배선들이 손상이 가지 않게 하기 위하여 각 축의 직선운동 부분에 케이블베이어(Cable Veyor)를 설치하였다.

다음 그림은 정밀 매거진의 전면과 측면의 사진이며 수확용 진공 그리퍼(Gripper)

가 장착되었으며, 매거진 이송 기구를 포함하여 차체 프레임에 설치된 모습이다

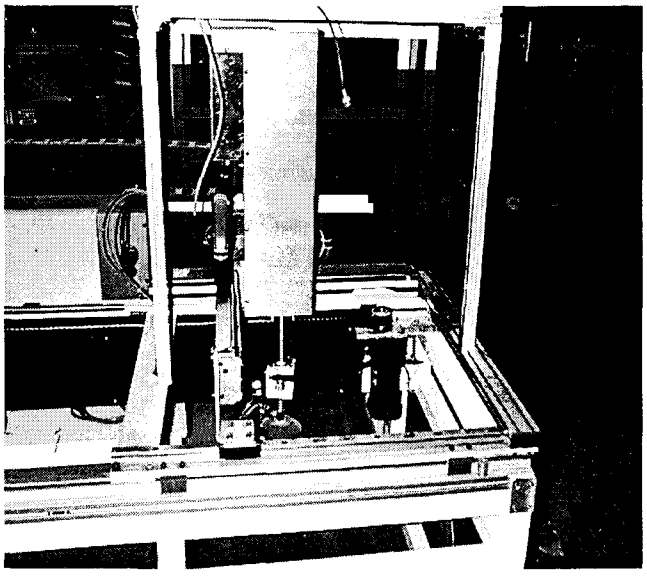

Fig. 3-4-16 Precision magazine and robot arm (front).

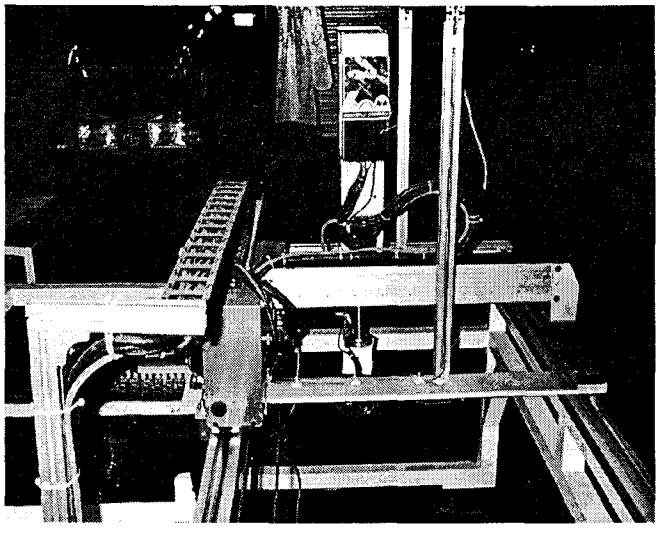

Fig. 3-4-17 Precision magazine and robot arm (side).

다음 그림은 돌리기 작업용 그리퍼(Gripper)가 장착된 사진이다.
여기서 그리퍼(Gripper)와 정밀 매거진의 선단부를 4개의 나사로 연결할 수 있게 하여 타 모듈(Module)형 작업 장치를 작업별로 쉽게 교체할 수 있게 하였으며, 공압, 진공, 센서, 동력선 등을 나선형으로 함으로 선단부의 수직이동에 대해 유연한 접속을 할 수 있게 하였다.

그리고 매거진 선단부에 여러 종류의 모듈러형(Modular type) 작업기가 장착되므로 각 장치마다 소요되는 사양이 다르므로 각 모듈(module)별 사양을 고려하여 공통 및 전용 커넥터를 마련하였으며, 쉽게 장/탈착을 할 수 있게 하였다.

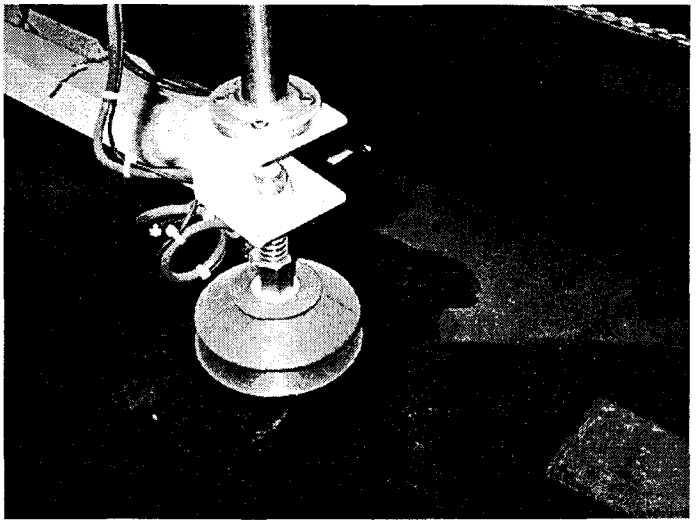

Fig. 3-4-18 Modular-type gripper for watermelon turning.

다음으로 진공 기구 및 공압 기구를 제어할 수 있게 그림과 같이 벤츄리(Venturi)형 진공 인젝터(Injector) 및 공압 솔레노이드를 사용하였다.
진공 인젝터의 사양은 다음과 같다.
- 최대 진공도 : 712.5㎜Hg
- 사용 압력 : 5 kg/㎠
- 형식 : Venturi Type Injector
- 구성 : 공기 필터, 제어용 솔레노이드(DC12V 120mA), 소음기

공압 솔레노이드는 수확물 받이 전후진 실린더 구동용으로 복동 솔레노이드를 사용하였으며, 모듈(Module)형 작업장치에 사용할 수 있도록 범용 단동 솔레노이드를 장착하였다. 각 공압용 솔레노이드의 구동은 DC 24V이며, 서보 제어기의 범용 출력 장치와 직결하여 제어하였다.

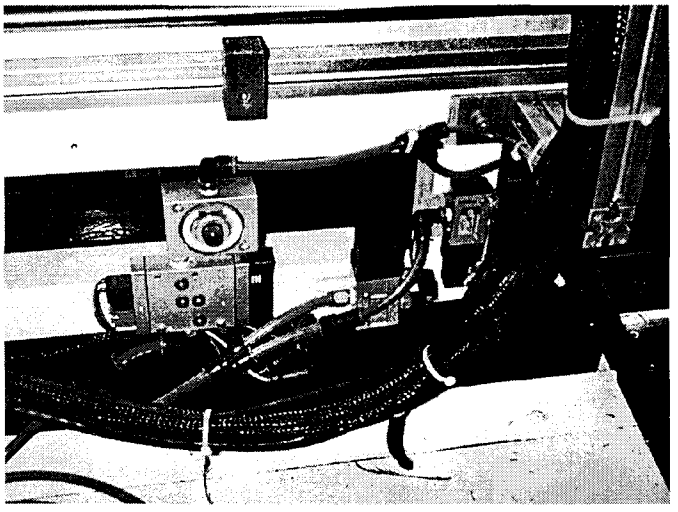

Fig. 3-4-19 Venturi way vacuum pump and air valve.

다. 영상처리 시스템

영상처리 시스템은 카메라, 프레임그래버, 영상 처리용 프로세서인 컴퓨터로 구성하였으며, 1차 시작기에는 유선으로 영상데이터(NTSC)를 전송하고, 전송된 데이터를 프레임 그래버에서 수신하여 터치스크린이 장착된 모니터에 표시하고 표시된 내용에 따라 작업자가 터치(Touch)를 통하여 시스템을 제어하게 하였다.

먼저 카메라의 설치를 위해서 정밀 매거진의 프레임에 40㎜ × 40㎜ 프로파일로 정밀매거진 사진에서 볼 수 있는 것과 같이 카메라 프레임을 설치하여 매거진이 구간 이송이 될 때 카메라 프레임이 같이 이송이 되며, 로봇 암의 이동이 있더라도 카메라 프레임은 매거진 프레임을 기준으로 고정되게 하였고, 카메라 뷰

(View)에서 매거진 구간 전 대역을 볼 수 있게 조절하였다.

다음으로 영상을 획득하는 시점에 로봇 암(Arm)이 카메라 뷰 상에 있지 않게 하기 위하여 영상획득 시점에 로봇 암을 원점 위치로 이동 시킨 후 영상획득 작업을 수행하게 하였다.

1차 시작기의 영상처리 시스템은 실험용 정밀매거진 제작에서 사용된 동일한 사양의 시스템으로 구성하였다. 다음 그림은 영상처리 시스템의 블록도이다.

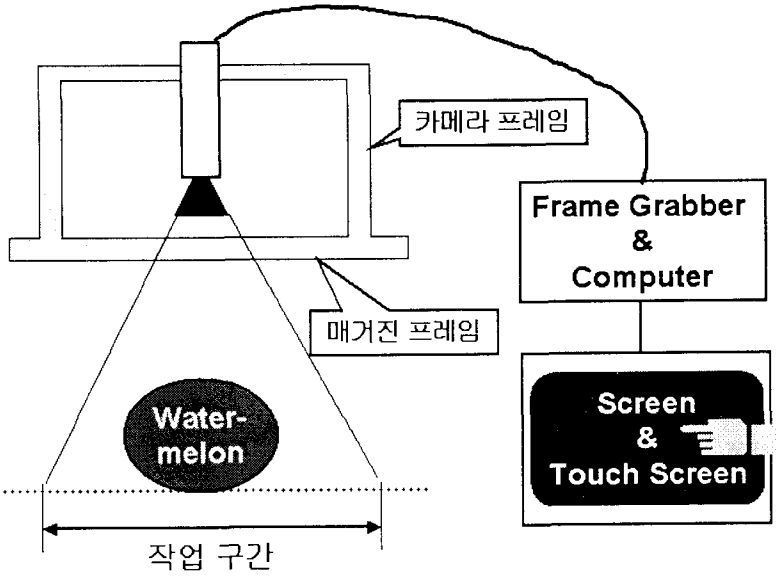

Fig. 3-4-20 Block diagram of the 1st prototype image processing system.

라. 모듈(module)형 단위 작업 장치

1) 수박 돌리기 그리퍼(Gripper)

가) 가반 하중

수박의 과중은 성숙과의 경우 종류에 따라 2.5~7kg정도이다. 현재 재배되는 수박의 종류는 대부분 대과종을 재배하는데 보통 4~6kg이며 향후 더 커질 것으로 예상된다. 이것을 그리퍼(Gripper) 및 정밀 매거진에서 다룰 수 있어야 하므로 설계기준은 안전율을 고려해서 6kgf로 설정하였다.

나) 돌리기 작업

수박의 하부(대지 접지면)가 위로 올라오도록 하기위해서는 꼭지와 끝 부분을 중심축으로 회전 시켜야 하며, 꼭지 및 주변의 줄기 및 잎에 영향을 주지 않아야 하고, 과피(果皮)의 손상이 없어야 한다.

회전방법으로는 먼저 꼭지 및 주변줄기, 잎에 영향을 주지 않게 하기 위하여 과실을 지상에서 일정높이 까지 견인하여 그리퍼(Gripper)로부터 주변을 분리하여야 한다. 과피(果皮)에 손상을 최대한 억제하고자 진공 패드를 이용하여 견인하는 방법을 선택하였으며, 견인 높이는 줄기 손상을 고려하여 150mm로 하였다. 다음으로 돌리기 작업은 기구를 단순하게 하고, 과피(果皮)의 손상을 억제하기 위하여 그림과 같이 과실의 측면에 2개의 진공 패드를 이용하는 방법을 선택하였으며, 공압을 이용하여 패드를 중심축으로 하여 회전시키는 구조로 하였다.

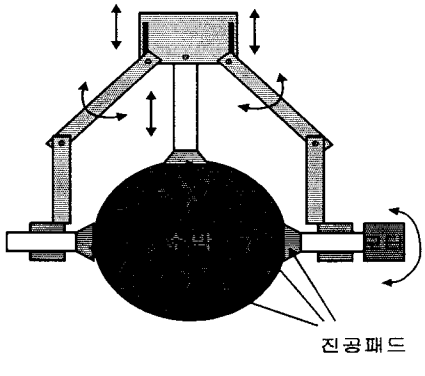

Fig. 3-4-21 Schematic diagram of gripper for watermelon rolling (1st prototype).

다음 그림은 돌리기 작업의 순서이다.

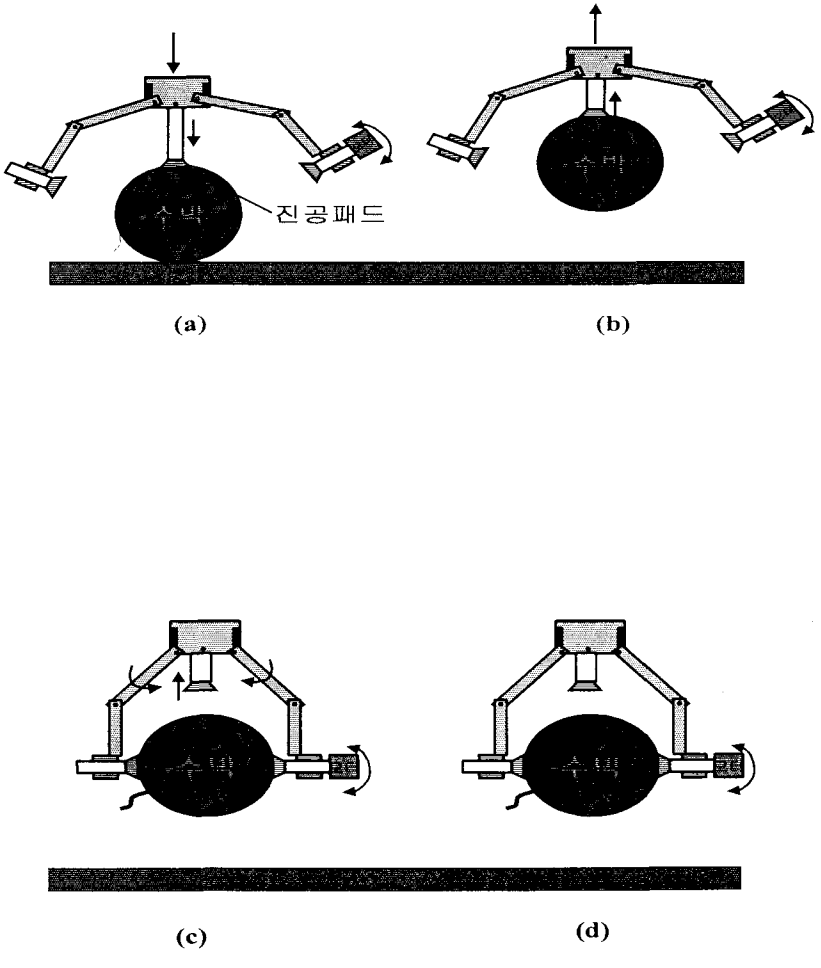

Fig. 3-4-22 Oder of turning watermelon.

먼저 견인작업을 하기 위하여 견인 진공 패드와 그리퍼(Gripper) 뭉치가 (a)와 같이 수박에 접근하고, (b)와 같이 견인을 한후 돌리기 작업을 위해 (c)와 같

이 꼭지를 축으로 돌리기용 진공 패드 및 그리퍼(Gripper)를 이용하여 수박을 잡고, (d)와 같이 돌리기 진공 패드를 축으로 공압모터를 이용하여 수박을 돌리는 순서로 작업이 이루어지게 하였다.

이 때 작업자가 터치스크린을 통하여 수박의 위치와 꼭지의 위치를 교시(敎示)함으로써 그리퍼(Gripper)의 회전 자세를 갖게 하여 돌리기 그리퍼(Gripper)가 꼭지부와 끝 쪽에 진공 패드를 부착시킬 수 있게 하였다.

다) 진공 흡착 패드 기구

진공 흡착 패드는 장기간 사용 후에도 기밀(氣密)이 유지될 수 있어야 하며, 기밀이 샐 경우 이것을 감지하여 소정의 조치를 할 수 있어야 된다. 그리고 패드가 수박에 접근할 때 패드 기구로 인한 과피(果皮)의 손상이 없어야 한다.

이러한 조건을 만족케 하기 위하여 진공 패드는 고무재질의 Ψ110를 선정하였으며, 견인용에는 평형을 회전축에는 평형립 부착형을 선정하였다. 그리고 패드 부착기구는 과피(果皮) 손상을 줄이고자 Z축 방향으로 스프링을 이용한 완충기구가 있는 모델을 선정하였으며, 진공 패드가 수박에 접근함으로 스프링 기구가 눌려지면 스프링 기구 상부에 장착된 감지 센서가 스프링의 눌린 량을 감지할 수 있게 하였으며, 이 때 패드의 누름을 중지시키게 하였다. 그리고 패드(Pad)의 목은 전 방향으로 약간의 회전을 할 수 있게 하여 수박의 중심좌표에서 다소 벗어난 위치로 패드가 접근하더라도 진공도도 유지시키고, 과피(果皮)의 손상도 억제시키게 하였다.

견인용 진공 패드 기구의 모델은 단해공압의 ZPX125HF-B01-B12를 선정하였으며, 회전축에는 동사의 ZPT50CFK10을 선정하였다.

2) 수박 수확용 그리퍼(Gripper)

가) 꼭지 절단

수박을 수확하기 위해서는 꼭지를 일정길이 간격으로 절단을 해야 한다. 그러나 꼭지의 상하 위치가 일정치 않고, 과경(果徑) 및 과형(果形)이 일정하지 않으므로 이러한 것에 적응을 하여 절단 작업을 수행해야 한다.

1차 시작기에서는 그림과 같이 칼날 교차식으로 수박 꼭지부의 줄기를 절단하는 방법을 사용하였다. 꼭지 절단 장치는 칼날 가이드, 칼날, 전방향 구름 롤러, 견인용 패드로 구성하였으며, 칼날 가이드는 수박 중심방향으로 스프링 장력 장치를 두어 수박표면을 구름 롤러가 일정 압력으로 접촉하면서 구를 수 있도록 하였으며, 절단 위치에 도달했을 때 절단을 위해 양 칼날을 교차시키기 위해 공압 실린더를 사용하였다. 그리고 절단 칼날의 끝부분에 전방향 구름롤러를 장착하여 과경(果徑)과 형상에 적응하여 칼날이 구를 수 있게 하였고, 견인용 패드가 수박을 지면으로부터 분리시키며 칼날을 꼭지가 존재하는 범위로 유도하게 한다. 다음으로 공압 실린더를 통하여 양 쪽 칼날이 상호 교차되게 하여 줄기를 절단하게 하였다. 이 때 꼭지부 줄기 절단 작업기가 자세를 갖도록 하기 위하여 작업자가 수박의 중심과 꼭지의 방향교시(敎示)를 해야 한다.

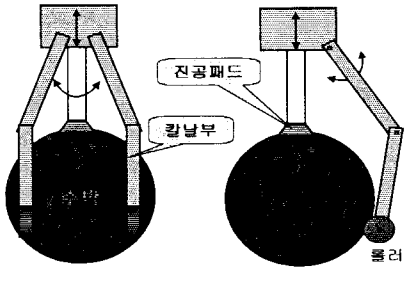

Fig. 3-4-23 Gripper for stem cutting.

나) 절단 범위

수박 꼭지의 줄기는 지면으로부터 다양한 높이에 있게 되는데 본 시작기에서는 수박의 수평축을 중심으로 상, 하 35°범위를 절단할 수 있게 설계 제작하였다.

다) 절단 순서

먼저 그림 (a), (b)와 같이 진공 패드를 이용하여 수박을 견인한 후 (c)와 같이 구름 가이드를 절단 범위로 하강시킨다. 그리고 (d)와 같이 칼날을 교차하여 절단한다.

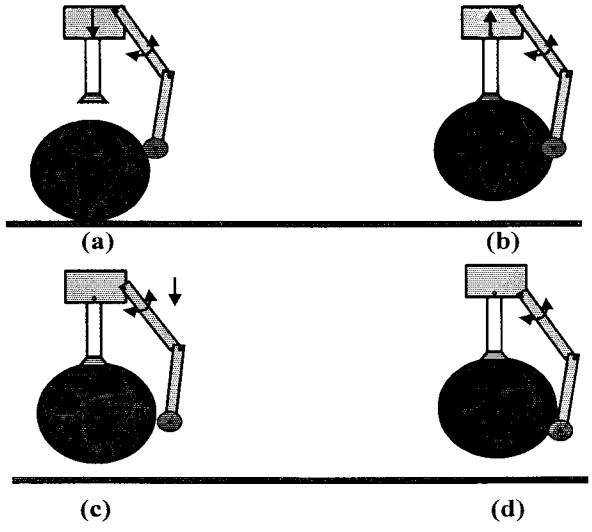

Fig. 3-4-24 Order of stem cutting.

마. 전체 제어 시스템

1) 제어 시스템 구성

1차 시작기의 시스템 제어부는 크게 터치스크린을 이용한 GUI형 MMI(Man Machine Interface) 부분과 XY 로봇의 서보 위치제어 및 구동부와 디지털 I/O, 그리고 드라이버 부분으로 나누어 그림과 같이 구성하였다. 먼저 MMI 부분은 5선 압력 저항식 터치스크린이 장착된 15" LCD Panel과 컴퓨터로 구성되어 있으며, 작업자의 터치를 통하여 1차 시작기의 시스템을 자동 또는 수동으로 통제할 수 있게 하였다.

다음으로 XY 로봇 위치제어 및 디지털 I/O 부분은 YAMAHA사의 DRC-2 로봇 제어기로 구성하여 MMI와 직렬통신(RS232C)으로 연결시켰다. 주 제어기의 모든 명령은 직렬 통신을 통하여 이루어지게 하여 향후 무선 원격제어에 대응할 수 있도록 하였다. DRC-2는 내부에 XY Position Controller와 가감속 제어기, 범용 디지털 I/O가 내장되어있다.

대차 구동용 인버터, Z축 및 회전축을 위한 스테핑 모터 드라이버, 릴레이 및 입력 스위치 및 센서들은 범용 디지털 I/O에 연결하여 각 장치를 구동할 수 있게 하였다.

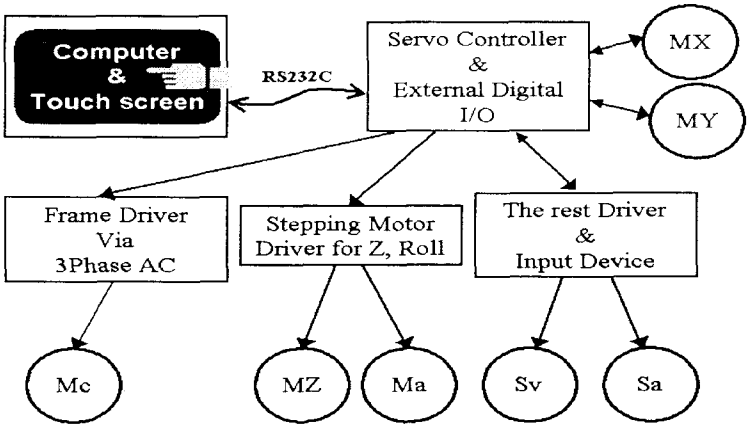

Fig. 3-4-25 Block diagram of control system.

다음으로 대차 구동용 3상 인버터는 강건한 정지를 위하여 저항식 브레이크를 사용할 수 있도록 하였으며, 스테핑 모터 드라이버는 정전류 초퍼(Chopper) 구동을 통하여 속도 및 효율을 높였으며, 정지 시 저전류를 모터에 인가할 수 있도록 하여 모터가 가열되는 것을 막았다. 공압 솔레노이드, 소형 모터 구동, 진공펌프용 솔레노이드 등의 구동에는 릴레이를 사용하였는데, 접점 보호를 위해 써지(Surge) 보호회로를 장착하였다. 그리고 입력 신호들은 DRC-2의 입력단에 직결하여 외부 센서의 상태를 알 수 있게 하였다.

2) 제어반

시스템 제어반은 크게 제어부와 드라이버부, 전원부, 릴레이 및 배전부로 구분하여 제작하였는데 소신호부와 대신호부, 전원부를 구별하여 배치함으로 상호 전기적인 오류를 발생하지 않도록 하였다. 그리고 열적인 보호를 하기 위하여 열원이 있는 드라이버부를 통풍이 쉬운 곳에 배치하였다. 각종 배선은 배선용 덕트(Duct)에 수납하였으며, 신호배선과 전력선은 분리시켜 배선하였다. 다음 그림은 시스템 제어반의 사진이다.

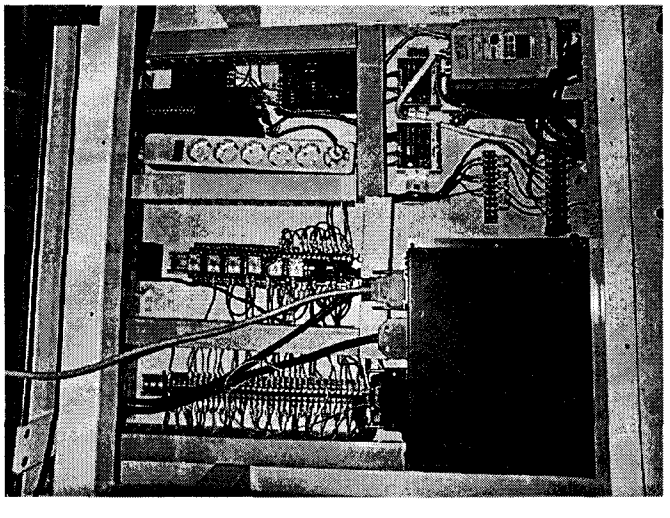

Fig. 3-4-26 Controller.

3) 모터 구동부

그림3-4-27은 인버터 및 스테핑 모터 드라이버부이며, 그림3-4-28은 XY 로봇 제어부이다. 이 부분은 열원으로 열이 잘 통할 수 있게 통풍구 근처에 배치하였으며, XY 로봇 제어부는 내부에 소신호 제어부와 대신호 모터드라이버 부분이 같이 있지만 상호 간섭에 대한 충분한 설계가 되어있다.

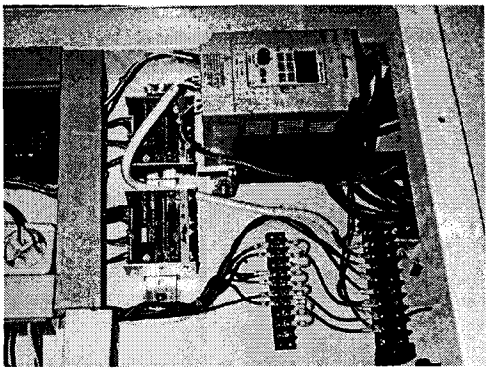

Fig. 3-4-27 Motor driver parts.

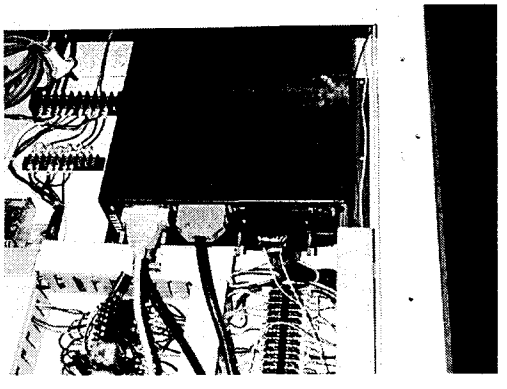

Fig. 3-4-28 DRC-2 control parts.

가) 대차 구동부 회로

여기서 릴레이 R1과 R2의 일부 접점을 이용하여 운전 중에는 브레이크를 풀고, 정지 시 브레이크를 작동시키게 하였고, 나머지 R1의 접점과 R2의 접점을 이용하여 모터의 정/역 동작을 수행하게 하였다.

다음 그림은 대차 구동용 인버터 및 모터, 브레이크 구동을 위한 회로도이다.

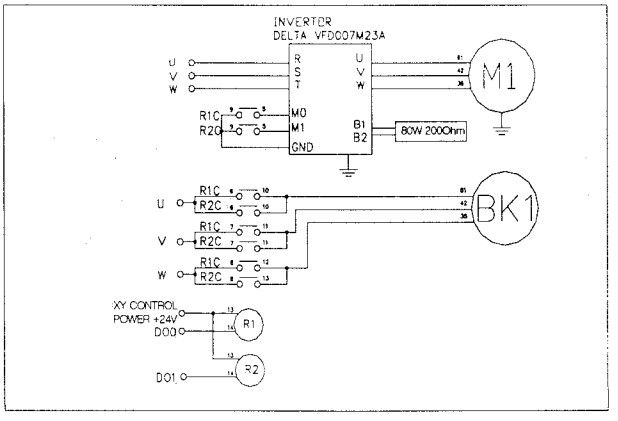

Fig. 3-4-29 Circuit for cart drive.

나) 정밀 매거진 이송 회로

정밀 매거진 이송을 위한 구동원은 Reversable AC 모터로 릴레이 R3과 R4를 통하여 브레이크 및 모터를 제어할 수 있게 아래 그림과 같이 회로를 구성하였다.

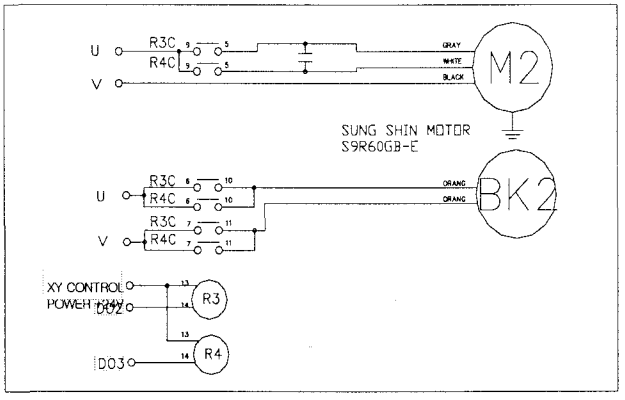

Fig. 3-4-30 Circuit for precision magazine transfer.

다) Z축 및 회전축 구동회로

Z축 및 회전축 구동을 위해 2개의 스테핑 모터를 사용하였다. 각 축의 스테핑

모터는 전용 드라이버를 이용하여 구동되어지는데, 스테핑 모터 드라이버는 제어용 전원과 구동용 전원을 아래 회로와 같이 분리시켜 구동부와 제어 신호부 간의 전기적 간섭을 배제시켰다.

스테핑 모터 드라이버의 제어 신호 입력단은 광 절연기 (Photo Isolator)를 장착하여 외부와 내부간의 전기적 절연을 시켜 외부 제어기의 오동작을 막게 하였으며, 모터 구동은 정전류 구동을 하여 속도 제어를 용이하게 하였으며, 보다 적은 전력과 열적 소비를 막게 하였다. 스테핑 모터의 경우 고속, 고토크 회전이 어려운 구조로 되어 있으므로 모터 코일의 시정수를 낮춤으로 보다 고속 제어를 할 수 있게 드라이버를 구성하였다.

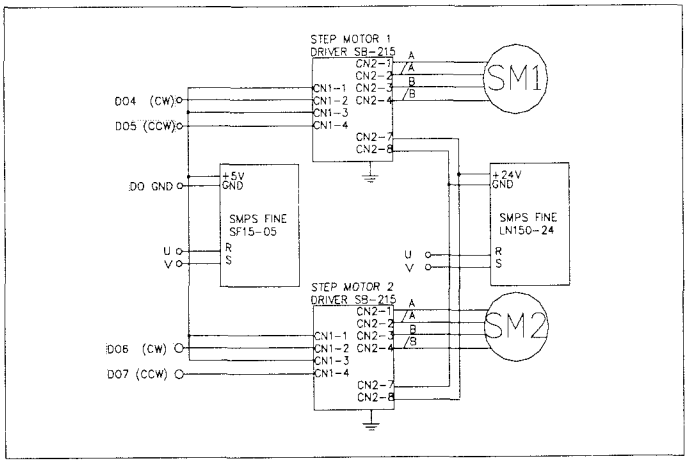

Fig. 3-4-31 Circuit for Z and motor drive for rotation Axis.

바. 시스템 소프트웨어

1차 시작기의 시스템 소프트웨어는 컬러 카메라를 통해 입력되는 작업 구간에 대한 영상신호를 모니터에 표시하고, 터치스크린을 통하여 작업자가 작업을 지시하면 획득된 영상을 바탕으로 정밀 매거진의 위치를 이동시키고, 모듈(module)형 작업기를 통하여 명령되어진 작업기를 수행하게 하였다.
각종 명령 버튼 및 작업 지시는 터치스크린을 통해 수행될 수 있게 효율적으로

구성하였으며, 각종 명령 체계는 작업자가 직관적이고 가장 단순한 조작을 통해 수행할 수 있도록 최적화 하였으며, 각 작업마다 필수적인 메뉴 구성과 버튼 구성으로 작업자의 오 조작을 방지하게 하였다. 먼저 시스템 소프트웨어의 주 메뉴는 다음과 같게 구성하였다.

- 시스템 수동 제어 : 로봇의 위치 및 속도의 수동 제어, 각종 출력장치의 수동 제어, 입력 데이터 모니터 기능을 가짐. 시스템 제어기에 사용자가 직접 명령을 입력할 수 있음.
- 수확 작업 모드 : 수확 작업을 수행하기 위한 일련의 작업을 작업자의 1차교시(敎示)를 통하여 수행하게 함.
- 방제(防除) 작업 모드 : 정밀 방제(防除)작업을 작업자 교시(敎示)를 통하여 일괄적으로 수행함.

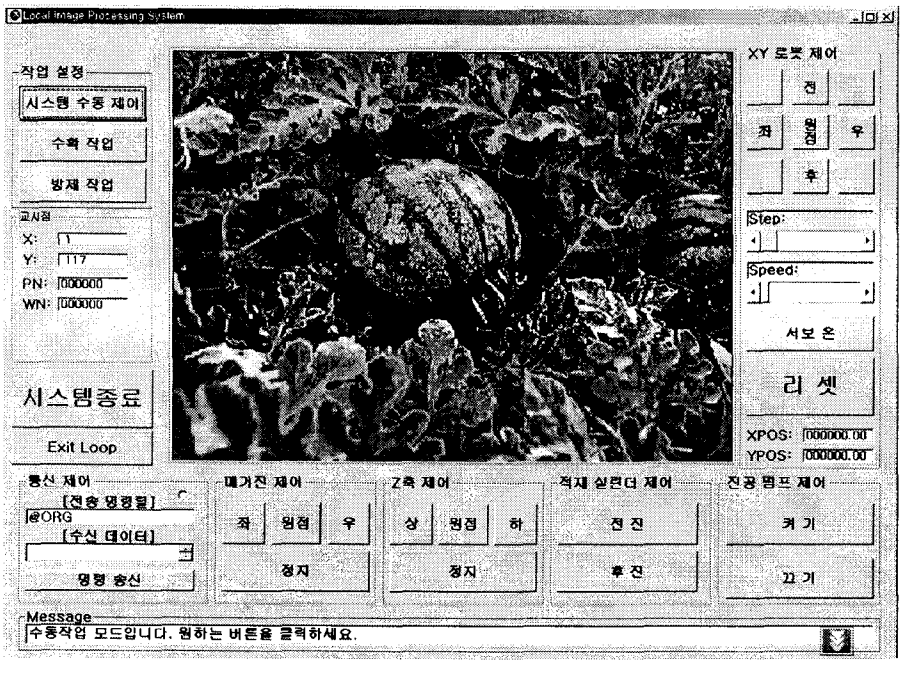

Fig. 3-4-32 Main program practice screen of the 1st prototype (touch pad screen).

1) 시스템 수동 제어

시스템 수동 제어 모드는 크게 매거진 제어와 서보 시스템 제어, 매거진 이송 제어, 각종 입/출력 제어로 아래 블록도와 같이 구성하였다.
시스템 수동제어는 시스템의 점검, 작업 환경의 변경, 카메라 보정을 위한 로봇 이송 등을 할 수 있게 하는 작업 모드이다. 그러나 필요에 따라 작업 과정에서도 임의의 시스템 조작이 필요한 경우 작업자의 의지에 따라 시스템을 조작 할 수 있게 하여 시스템 운용에 융통성을 부여하게 했다.

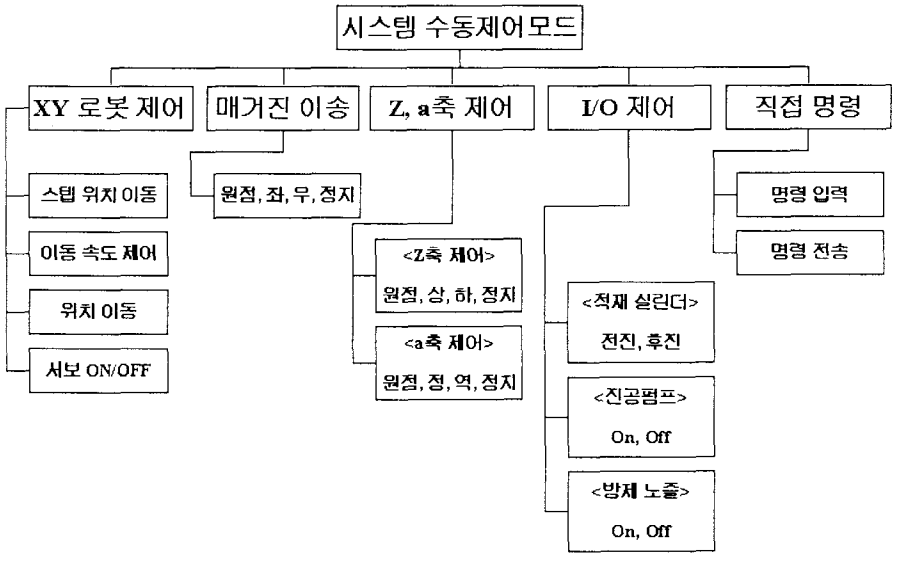

Fig. 3-4-33 Block diagram of the system control mode (1st prototype).

2) 수확 작업 모드

수확 작업 모드는 수박을 수확하는 일련의 작업을 작업자의 수박위치 및 꼭지부 위치 교시(敎示)를 통하여 수행하게 하는 작업모드이다.
그림3-4-34는 수확 작업 모드의 순서도이다. 여기서 작업자의 작업 명령에 따라 작업이 수행되는데, 먼저 작업 구간별로 수박 위치 교시(敎示) 및 꼭지 방향 교시(敎示)가 완료되면 교시(敎示) 되어진 좌표를 산출하고 각 목표물에 대한 수확 작업을 수행하여 수확물을 적재 장치를 통해 적재를 수행하게 하였다.

보통 수박 수확의 경우 한 작업 구간에 1개의 수확물이 존재하게 되므로 현재 작업 구간에서 1차 교시(敎示)가 이루어지면 그 좌표를 산출 및 저장하고 다음 구간으로 이동하게 하였다.

만약 작업자의 명령이 영역 이동일 경우에는 현재 작업 영역의 작업을 중단하고 다음 작업 영역으로 이동하게 하였으며, 작업자의 명령이 모드 종료이면 현재 영역에서 교시(敎示)한 데이터를 모두 무시하고 매거진 및 로봇을 원점복귀 시킨 후 수확모드를 종료하게 하였다.

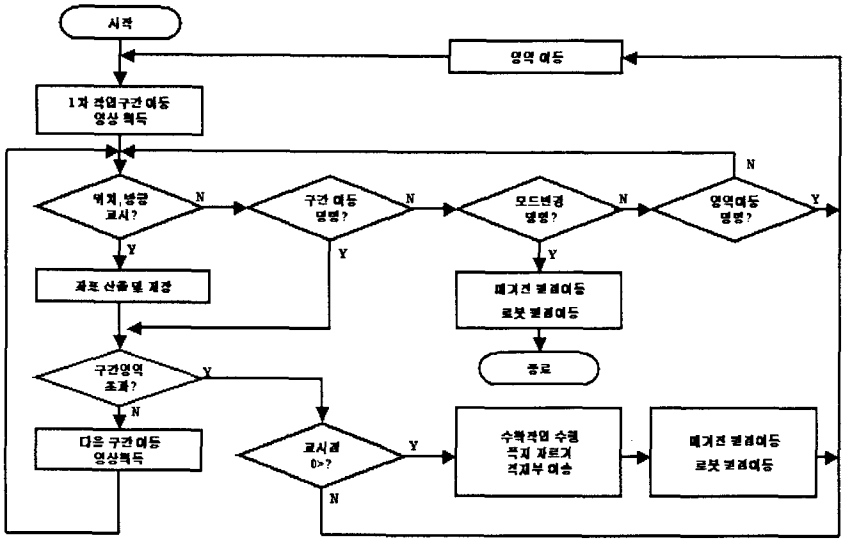

Fig. 3-4-34 Flowchart of harvest work (1st prototype).

그리고 각 작업 구간에서 작업 대상이 없을 경우 작업자가 구간 이동 명령을 수행할 수 있게 하여 교시(敎示)점 없이 다음 작업 구간으로 이동할 수 있게 하였으며, 전 작업 구간에서 교시(敎示)점이 없을 경우 수확 작업을 수행하지 않고, 매거진 및 로봇을 원점복귀 시키고 다음 작업 영역으로 대차를 이동 시키게 하였다. 수확 작업은 처음 교시(敎示)점을 시작으로 작업을 진행하는데 진행 방법을 다음과 같게 하였다.

① 로봇을 원점 위치로 이동 시킨 후 교시(敎示)점이 정의된 작업 구간으로 매거진을 이송시킨다.
② 매거진 이송이 완료되면 로봇을 통하여 수확 작업용 그리퍼(Gripper) 수확물 중앙점으로 이동시킨다.
③ 회전축의 회전을 통하여 그리퍼(Gripper)의 자세를 정렬한 후 Z축 및 진공패드의 동작을 통하여 수확물을 일정 높이로 상승 시키고, 절단 칼을 아래로 내려 절단 영역에 꼭지부 줄기가 오게 한다.
④ 절단 칼을 상호 교차 시켜 줄기를 절단한다.
⑤ 로봇을 안전 위치로 이동 시킨 후 매거진을 적재함 위치로 이동시킨다.
⑥ 적재함 받이를 전진 시키고 로봇을 적재함 받이 위치로 이동 시킨 후 진공펌프를 끔으로 수확물을 적재함 받이에 놓이게 한다.
⑦ 적재함 받이를 후진시킴으로 수확작업 종료

여기서 작업 구간이란 정밀 매거진에 장착된 로봇이 매거진 이송 없이 작업할 수 있는 공간을 말하며, 구간 이동은 매거진 이송부를 통해 대차의 진행방향과 수직으로 이동을 하는 것을 말한다.

그리고 작업 영역이란 대차의 움직임이 없고, 매거진 또는 로봇이 이동하여 작업할 수 있는 공간을 말하며, 영역 이동이란 대차를 이동하여 새로운 작업공간으로 이동함을 말한다.

3) 방제(防除) 작업

방제(防除) 작업 모드는 약제를 정밀하게 살포하는 일련의 작업을 작업자의 방제(防除)위치 교시(敎示)를 통하여 수행하게 하는 작업 모드이다. 아래 그림은 방제(防除) 작업 모드의 순서도이다. 여기서 작업자의 작업 명령에 따라 작업이 수행되는데, 먼저 작업구간별로 방제(防除) 위치 교시(敎示)가 완료되면 교시(敎示)되어진 좌표를 산출하고 각 목표물에 대한 방제(防除)작업을 현재 작업 영역에 대하여 일괄적으로 수행하였다.

보통 방제(防除)의 경우 한 작업 구간에 1개 이상의 방제(防除) 교시(敎示)점이 존재하게 되므로 구간 이동 명령을 통하여 현재 작업 구간에서 모든 교시(敎示)가 끝나게 하였다. 구간 이동 명령을 하면 현 구간에서 교시(敎示)된 모든 좌표를 산출 및 저장하고 다음 구간으로 이동하게 하였다. 만약 영역 이동일 경우에는 현재 작업영역의 작업을 중단하고 교시(敎示)된 데이터를 지운 후 작업 영역으로 이동하게 하였으며, 작업자의 명령이 모드 종료이면 현재 영역에서 교시(敎示)한 데이터를 모두 무시하고 매거진 및 로봇을 원점복귀 시킨 후 방제(防除) 모드를 종료하게 하였다.

방제(防除) 작업은 처음 교시(敎示)점을 시작으로 작업을 진행하는데 진행 방법을 다음과 같게 하였다.

- 로봇을 원점 위치로 이동 시킨 후 교시(敎示)점이 정의된 작업 구간으로 매거진을 이송시킨다.
- 매거진 이송이 완료되면 로봇을 통하여 방제(防除) 위치로 노즐(Nozzle)을 이동시키는데, 각 구간에서 경로를 최적화 시킨 좌표데이터를 통하여 방제(防除) 작업이 이루어지게 하였다.
- 방제(防除) 위치로 노즐을 이동시키고, 노즐 제어용 솔레노이드를 작동시킴으로 방제(防除) 작업을 수행한다.
- 현재 구간에서의 작업이 종료되면 로봇을 원점 복귀 시켜도 다음 구간으로 매거진을 이송하여 방제(防除) 작업을 계속 수행한다.
- 전 영역에 대하여 방제(防除) 작업이 완료되면 로봇 및 매거진을 원점복귀 시키고, 다음 작업 영역으로 대차를 이동시킴으로 방제(防除) 작업 완료

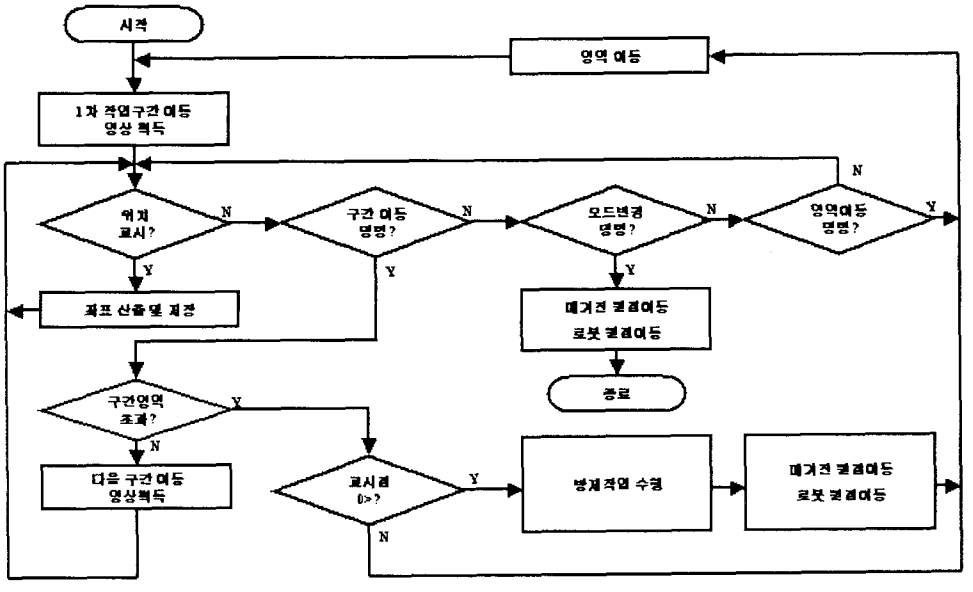

Fig. 3-4-35 Flowchart of pesticide (1st prototype).

3. 성능 시험

가. 시험 환경

다목적 생력 시스템 1차 시작기의 성능시험은 수확 작업 시험, 방제(防除) 작업 시험으로 나누어 실시하였다. 성능 시험을 하기 위하여 실험실의 인공 재배환경을 구축하고, 실내 인공광 환경(형광등) 하에서 실시하였다.

공시 재료로는 대과종 3개, 중과종 3개를 이용하였으며 중량은 3kg~8kg이였으며, 형태는 원형, 타원형을 사용하였다. 인공 재배 환경은 바닥면을 갈색 카페트를 이용하여 토양을 대신하였고, 줄기 및 잎은 장식용 넝쿨을 사용하여 실제 현장과 같은 복잡영상을 만들었다.

나. 수확 작업 성능 시험 결과 및 고찰

수확작업 성능 시험 항목은 수확 시간으로 하였다. 영상 획득 및 1차 교시(敎示)

시간을 제외한 순수한 작업기 동작 시간을 측정하였다. 시험 결과를 요약하면 다음과 같다.

대과종의 경우 꼭지 절단 후 적재함까지의 이송속도를 빠르게 할 수 없었는데 이것은 과중과 수확 작업 그리퍼(Gripper)의 관성으로 인하여 로봇 암에서 진동이 발생하였다. 그로 인하여 진공 패드로부터 수박이 이탈되는 현상이 발생하였으므로 저속 운전을 하였다. 대과종의 경우 수확작업 소요시간은 30초 정도 이며, 중과종의 경우 좀더 빠른 이송을 할 수 있어서 20초 정도 소요되었다. 전체 시간을 구간별로 나누어 보면 꼭지 자르기에 5초가 소요되었으며, 그 나머지 시간은 이송 시간이다. 이송시간도 2가지로 나누어지는데 로봇 암의 이동과 매거진 전체가 이송하는 것이 있다. 대과종의 경우 로봇 암 이동에 13초가 소요되었으며, 매거진 이송에 12초가 소요되었다. 그리고 중과종의 경우 로봇 암 이송에 8초가 소요되었으며, 매거진 이송에 7초가 소요되었다. 여기서 로봇 암의 경우 Z축의 이동이 가장 길었으며, XY 로봇의 경우 매거진이 이송될 때 동시에 작동하게 함으로 별도의 시간이 소요되지 않았다. 그리고 매거진 이송의 경우 가/감속 제어를 하지 않아 고속으로 동작 시킬 수가 없었다. 작업기 단위별 시험에서 꼭지 절단작업이 원활하게 수행되지 못하여 실제 현장에서의 실험은 할 수 없었다.

로봇 이동과 매거진 이동이 별도로 구성되어 제어 체계가 복잡하였으며, 수확용 그리퍼(Gripper) 시스템이 고 중량이라 관성에 의한 영향도 컸다. 향후, 그리퍼(Gripper) 시스템의 중량 및 꼭지 절단 방식을 개선할 필요가 있었으며, 정밀 매거진의 경우도 작업 영역이 협소하여 카메라 뷰에 작업 장치가 영향을 주었으므로 영역의 확대도 필요하다고 사료된다.

제 5 절 2 차 시작기 개발

1. 서론

중량과 재배관리, 수확 및 동시선별 적재작업을 위한 다목적 생력작업시스템 개발을 위하여 본 연구에서는 2차 시작기를 통하여 1차 시작기의 문제점을 개선하고, 보다 현장 적응성을 높이고자 하였다. 설계의 기준은 다음과 같이 하였다.

- 1차 시작기에서 설계 제작 및 실험한 모듈형 단위작업 시스템의 성능을 개선하고, 작업 체계에 대한 재분석을 통한 현장 적응성을 높였다.
- 중규모의 시설 환경 하에서 작업 할 수 있는 시스템을 개발하였다.
- 무선 원격 제어 및 국부 영상처리의 수행을 통한 통합 제어 시스템 구축의 기본 사항에 대한 연구를 수행하고 이를 토대로 시스템을 설계 제작 실험을 수행하였다.

본 연구에서 2차 시작기의 유효 작업 공간 설정은 중규모 시설 재배 표준인 단동 아치형을 기준으로 시스템을 설계 하였는데 그 내용은 다음과 같다.

- 작형 : 땅에서 눕혀서 재배 -> 2800mm x 45mm~ 60mm
- 이랑 너비 : 3000mm
- 주당 너비 : 45mm ~ 65mm
- 기준 과경(果徑) : Ψ350mm
- 기준 과중 : 6kg 기준
- 유효 작업 공간 :
  - 이랑 방향 2600mm
  - 터널 방향 750mm
  - 높이 방향 600mm

위의 설계기준을 통하여 2차 시작기를 제작하여 모듈 별 기능사양을 보완하였으며

무선 원격 제어를 통한 주 제어 시스템의 MMI 및 원격 통신 프로토콜을 구축하고, 원격지의 제어 시스템 및 소프트웨어를 개발하였다. 2차 시작기를 설계, 제작 및 실험을 통하여 설계기준을 정하였으며, 추가로 원격 무선 통합 제어 시스템을 구축하였다.

2. 시스템 구성

가. 작업기 프레임(Frame) 및 모듈형 작업기 장착용 정밀 작업 매거진

1) 작업기 프레임

본 연구에서 2차 시작기로 개발된 작업기 프레임은 1차 시작기와 달리 수확 작업과 같은 중량물을 다루는 작업에서 로봇 선단의 동적인 안정성을 확보하기 위해서 X축의 전 영역을 하나의 축으로 하였으며, 서보 모터가 이 축을 담당하게 하였다.

1차 시작기에서는 X축의 일부분을 정밀 매거진에 의해 동작되었으며, 전체 영역의 이동은 구간으로 나누어 정밀 매거진 이송부가 이동하게 하였기 때문에 로봇과 매거진 이송이 분리되어 동작하게 하였다. 이렇게 됨으로 매거진 이송에서 동적인 안정성을 얻을 수 없었고, 그로 인하여 중량물의 처리가 어려웠다. 2차 시작기에서는 X축 전 방향에 대한 서보 구조를 갖게 프레임을 구성하였다.

로봇 암의 구조도 전방 개방형으로 하여 모듈형 작업기가 작업기 프레임과의 간섭을 최소화 하게 하였다. 그래서 프레임의 구조도 로봇 암의 지지구조에 맞추어 균형을 잡을 수 있도록 베이스(Base)부를 돌출되게 하였다. 프레임의 규모는 고랑의 너비를 단동 아취형 표준 시설에 기준하여 설정하였으며, 구동 시스템은 1차 시작기와 같이 레일 구동형으로 하였다.

차체의 전체 외형은 그림과 같이 갠트리(Gantry) 형태로 설계하여 기계적으로 안정성을 도모 하였으며, 1차 시작기보다 넓은 영역에서의 작업을 수행해야 하므로 보다 큰 프레임의 강도를 갖게 하기 위하여 80×160㎜의 프로파일(Profile)

을 이용하여 3층 구조로 보강하였다. 레일구동부와 정밀 매거진을 제외한 실제 프레임의 설계 외형 치수는 3300㎜ × 1600㎜ × 1600㎜(W × D × H)이다.

설계 외형 치수의 산출 근거는 단동 아치형 시설 하우스의 이랑 너비에 맞추어 프레임의 폭을 결정하였으며, 프레임의 깊이는 정밀 매거진의 외형과 수확물 수납을 위한 공간 및 기계적인 균형을 고려하여 결정하였다. 그리고 프레임의 높이는 레일과 이랑 사이의 높이 차이와 과경(果徑) 그리고 선단 작업부 치수, Z축 유효 이동거리를 고려하여 결정하였다.

그리고 1차 시작기와 달리 작업자가 원격 무선으로 시스템을 제어하는 구조이므로 작업자의 탑승공간은 마련하지 않았다. 다음 그림은 2차 시작기에 대한 3차원 도면이다.

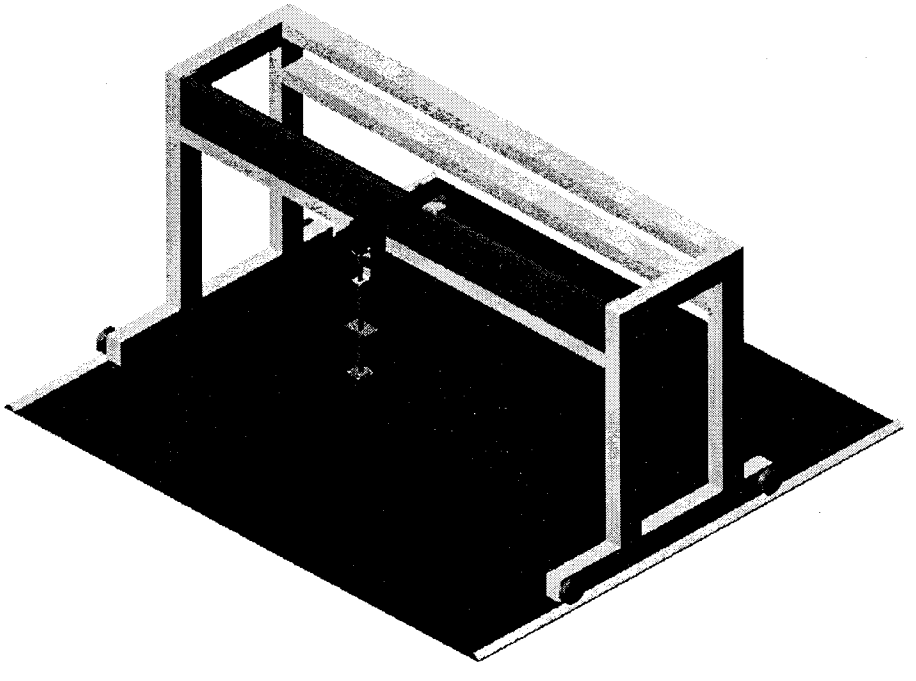

Fig. 3-5-1 3D schematic of 2nd prototype.

프레임의 공간별 구조는 다음과 같다. 프레임은 높이 방향으로 2개 공간, 너비 방향으로 3개 공간으로 나누었는데 아래층의 한 쪽을 수확물 수납을 위한 공간 또는 모듈형 작업 장치의 부속 장치를 설치할 수 있는 공간으로 사용할 수 있게 하였고, 가운데 부분은 수확물을 유도하여 수확물 수납공간으로 이송하는 시스템을 설치하였으며, 반대쪽은 축전지 및 독립 전원기구, 공압 펌프 등을 설치할 수 있게 하였다. 그리고 위층은 대차 구동용 모터 및 동력 전달기구, 공압 솔레노이드, 시스템 제어기, 무선 설비 등과 같은 제어 장치를 설치하게 설계하였다.

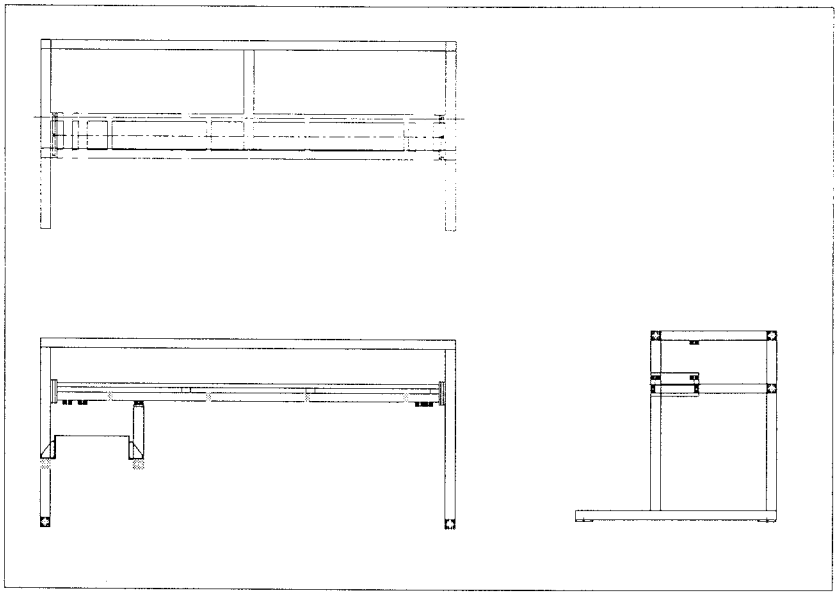

Fig. 3-5-2 Schematic drawing of main frame (2nd prototype).

그리고 프레임이 대형이므로 시설 내 설치 및 원거리 이동시 문제를 발생시킬 수 있으므로 기능별로 각 부분을 분할하여 분해 및 조립을 할 수 있게 하였다. 각 부분은 다음과 같다. 프레임 측면 지지부, 로봇 지지부, 수납부, 축전지 및 진공펌프 설치부, 수확물 유도부로 나누어 분해 조립을 할 수 있게 하였다.

각 구성은 기준 결합 구조물을 만들어 상호 조립 될 때 분해 및 조립으로 인하여 발생 될 수 있는 시스템이 틀어지는 현상을 막을 수 있고, 손쉽게 분해, 조립할 수 있다. 다음 그림은 2차 시작기 프레임의 사진이다.

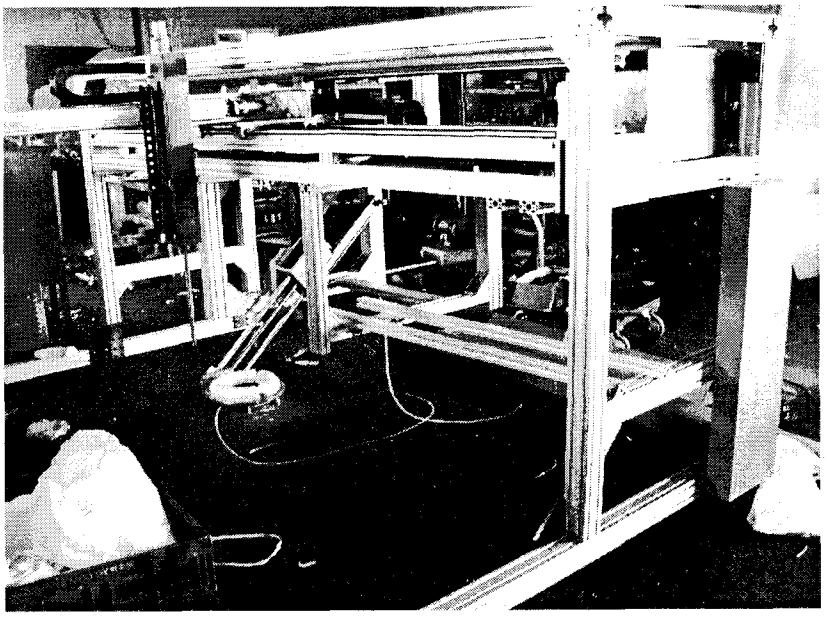

Fig. 3-5-3 Frame of 2nd prototype.

2) 대차 구동 시스템

대차 구동 시스템은 그림과 같이 레일(Rail) 사용하여 그 위를 주행하게 설계하였다. 시스템의 중량으로 인하여 레일의 침하를 막기 위하여 아래와 같은 구조의 레일을 사용하였으며, 구동 바퀴는 1차 시작기와 같은 기어 감속기가 장착된 1kW의 AC 모터의 동력을 체인을 통하여 전달하게 하였다. 이 때 대차 구동용 AC 모터는 인버터를 통해서 속도 및 정/역 회전이 제어되며, 체인(Chain)은 장력 조절 기구를 통하여 장력이 조정되게 하였다.

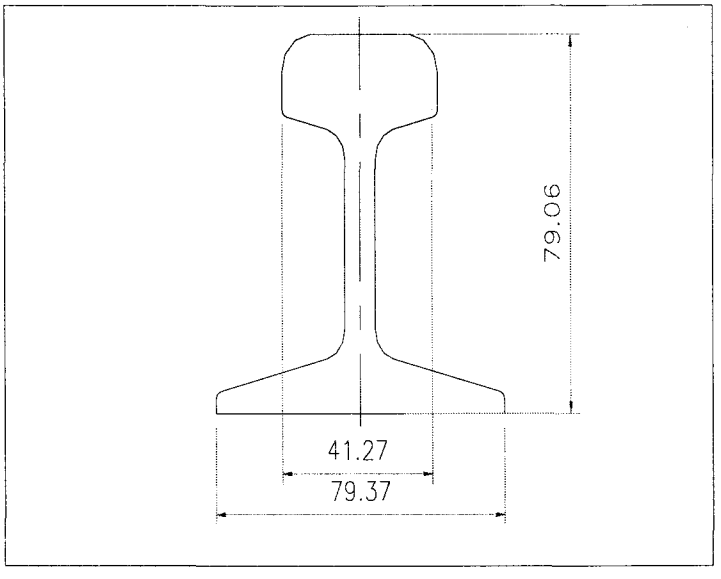

Fig. 3-5-4 Schematic diagram of rail.

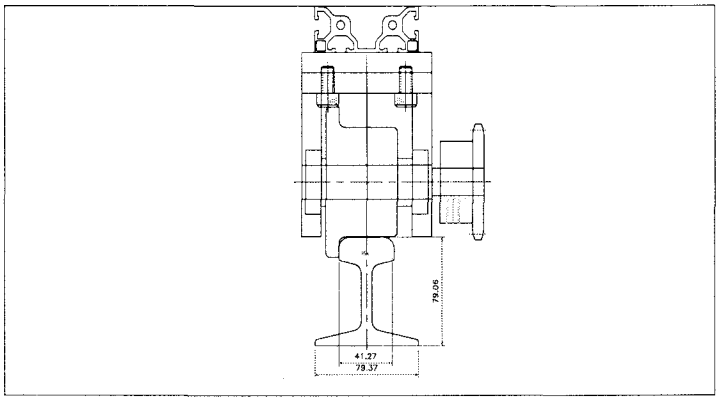

Fig. 3-5-5 Schematic diagram of wheel structure.

그림 3-5-6은 레일 주행형 대차 구동 기구의 사진이다.

그림 (a)는 구동모터 및 체인 장력조절기구, 체인, 구동 바퀴가 설치된 사진이고 그림(b)는 구동모터 및 장력조절기구, 모듈 분리 및 감속비를 조절할 수 있는 기

어이다. 다음 그림 (c)는 레일 위를 구동할 수 있는 구동 바퀴이다.

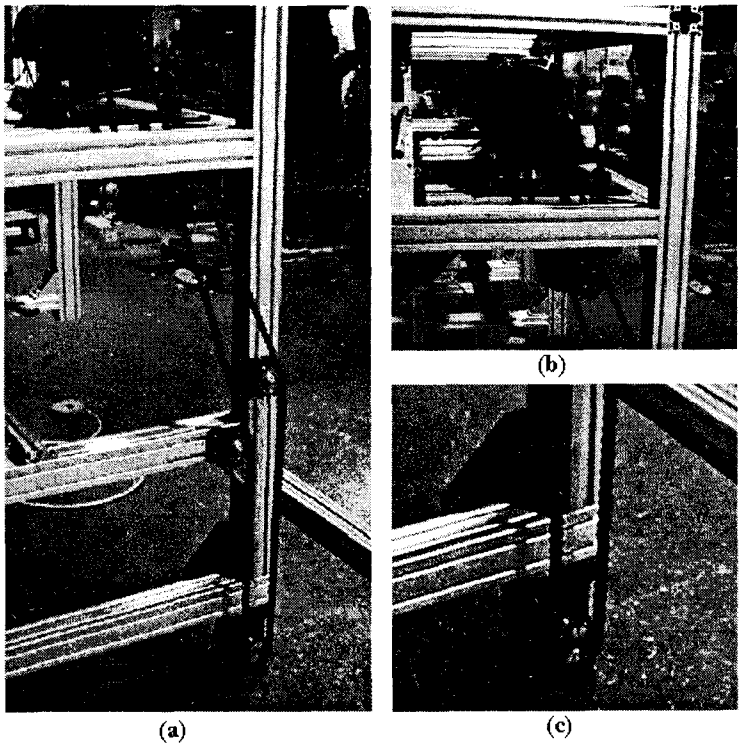

Fig. 3-5-6 Wheel structure.

대차의 주행 및 주행한계, 구간주행 구조는 1차 시작기와 동일하다. 단 주행용 모터의 브레이크는 제거하였으며, 소프트웨어적인 정지를 할 수 있도록 하였다.

3) 수확물 이송 기구

수확물 이송기구는 수박 수확 시 로봇 암으로부터 수확되어지는 수확물을 적재함으로 이송하는 장치인데 높이 문제로 인하여 로봇 암이 수확물을 직접 적재함에 적재 할 수 없기 때문에 본 장치가 필요하다. 본 시스템에서는 프레임의 가운데 부분에 장착하여 수확물 이송 경로를 최단거리로 하였으며, 차체의 방향에 상관없이 시설 내 이랑에서 작업을 수행 할 수 있게 하였다. 다음 그림은 수

확물 이송기구에 대한 사진이다.

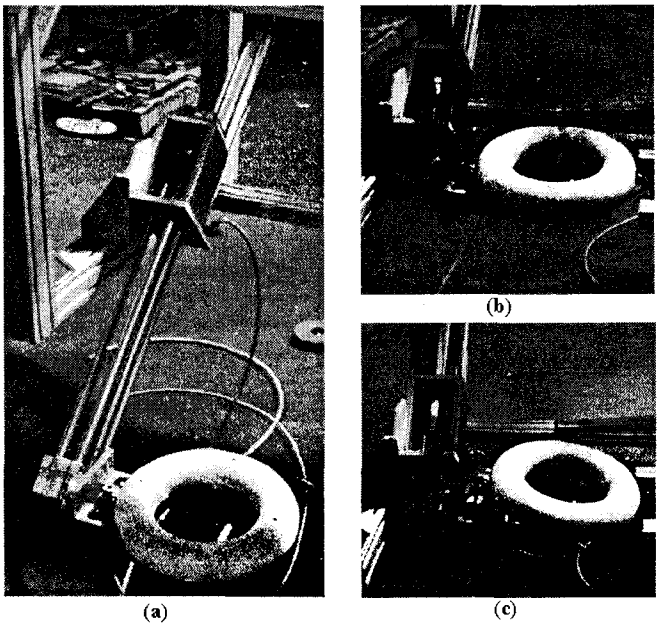

Fig. 3-5-7 Watermelon lifting system for harvest.

수확물 이송 장치의 동작은 다음과 같게 하였다. 수확물이 수확되어 로봇 암으로부터 수확물 이송장치로 이동하면 그림(a)와 같이 이송장치는 아래로 내려와 수확물을 실을 수 있는 자세를 갖는다. 다음으로 수확물이 이송장치에 올려지면, 로봇 암은 안전지대로 이동하고, 이송 장치는 그림(b)와 같이 수확물을 적재함이 있는 곳으로 수확물을 들어 올린다. 그리고 적재함으로 수확물을 보내기 위해 공압 실린더를 통하여 그림(c)와 같이 동작한다.
수확물 이송장치의 구동은 2개의 공압 실린더를 통해 동작하는데, 하나는 수확물을 들어 올리는데 소요되고, 또 하나는 적재함으로 수확물을 굴리는데 사용하였다.

다음으로 이송 장치는 수확물을 올려놓을 수 있는 적재부, 적재함까지 수확물을 들어올리기 위한 들어올림 기구, 적재함으로 수확물을 굴리기 위한 굴림 기구로

구성하였다.

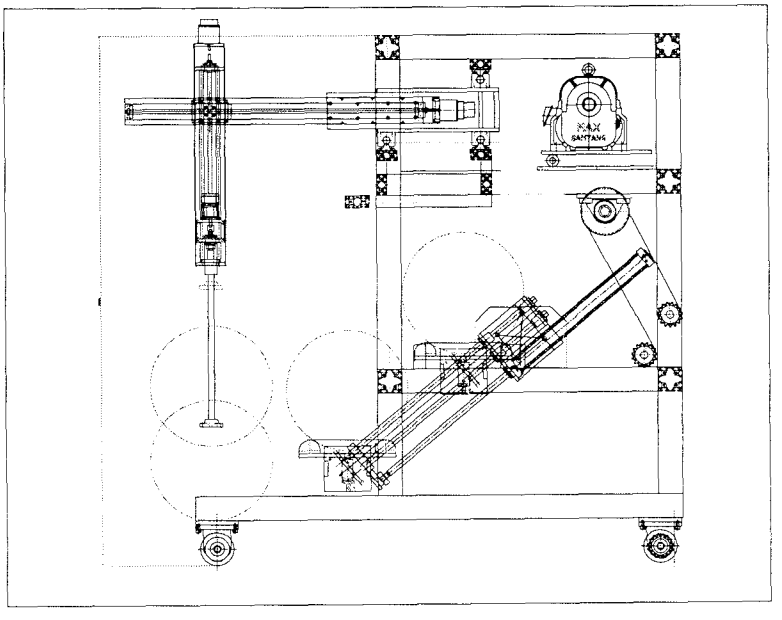

Fig. 3-5-8 Schematic diagram of watermelon lifting unit for harvest.

위 그림3-5-8은 수확 후 수확물의 이송 경로를 나타낸 것이다. 그림에서 수확물의 크기는 지름이 340㎜를 기준으로 하였으며, 무게는 6㎏으로 하였다.

들어올리기를 위한 실린더의 행정은 그림에서와 같이 모듈형 수확용 그리퍼가 수확물을 파지(把持)하고, 로봇 암이 Z축 상 최상위로 들어 올린 후의 높이에 맞추어 수확물 이송장치의 적재부의 최하위 위치와 수확물 이송장치의 적재부가 적재함 높이 까지 도달 했을 때의 위치를 기준으로 설정하였다. 그리고 수확물 이송장치의 적재부 형태 및 크기는 340㎜의 수박을 안전하게 올려놓기 위해 아래 도면과 같이 설계 제작 하였다.

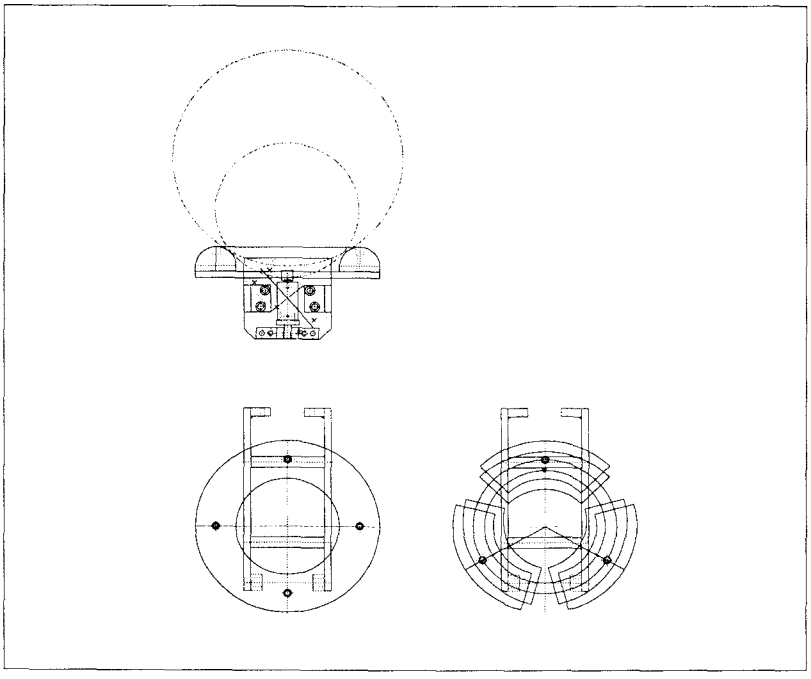

Fig. 3-5-9 Schematic diagram of waremelon loading parts.

적재부의 재질은 플라스틱(Plastic)이며, 반원형 링 구조를 갖게 하였으며, 수박의 크기에 따라 링의 반경을 조절 할 수 있게 하였다.

4) 모듈형 작업기 장착용 정밀 작업 로봇 암 및 부속 기구

모듈형 작업기를 장착할 수 있는 정밀 작업 로봇 암의 개략도(槪略圖)는 아래 그림과 같이 프레임 전면부에 부착할 수 있게 설계하였으며, X축, Y축 및 Z축의 구동은 AC 서보 모터를 사용하였으며, 회전축은 스테핑 모터를 사용하였다. 사용 모터는 다음과 같다.

- X축 : 삼성 서보 모터(CSM-10BB2ANT-3 with CSDJ-10BX2 Driver
- Y축 : 삼성 서보 모터(CSM-04BB2ANT-3 with CSDJ-04BX2 Driver
- Z축 : 삼성 서보 모터(CSM-02BB1ABT-3 with CSDJ-02BX2 Driver

- 회전축 : Oriental motor(japan), PK264A1(10kg/cm), 18:1 감속

서보모터의 각 축은 1회전 당 2048스텝(Step)으로 구분되어 디지털 제어되게 설계되어졌다.

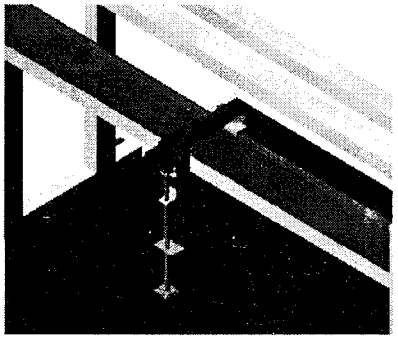

Fig. 3-5-10 3D Schematic diagram of 2nd prototype magazine.

정밀 작업 로봇 암의 동작 한계는 광센서를 이용하여 그 끝점을 감지하게 하였다. 사용 광센서는 OMRON사의 eesx6747 광센서로 위치 감지 정밀도가 0.0255㎜이므로 X축의 원점 센서로 그 기능을 겸하게 사용하였다.
다음 그림은 리밋(Limit) 및 원점센서로 사용된 광센서의 설치 모습이다.

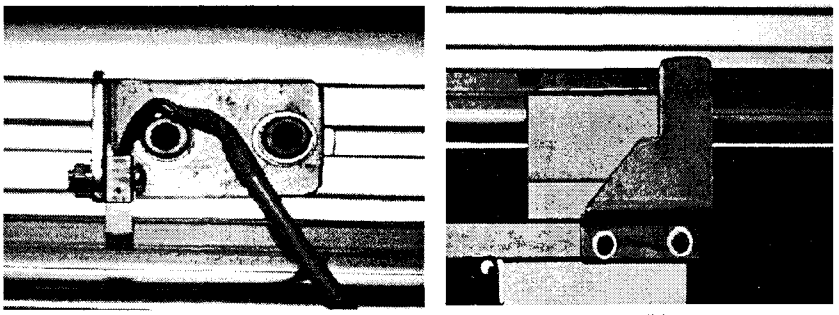

Fig. 3-5-11 Optic sensor and detecting object for area limit.

2차 시작기는 주 프레임부터 기능별로 모듈화 시켜서 조립 및 분해를 용이하게 설계 제작 하였는데, 로봇 암의 경우에도 프레임과 분리를 위해 각 축을 커넥터 처리하여 분해를 쉽게 할 수 있게 설계하였다. 그리고 직선 운동으로 인하여 이러한 배선들이 손상이 가지 않게 하기 위하여 각 축의 직선운동 부분에 케이블 베이어(Cable Veyor)를 설치하였다.

각 축은 원점 감지를 위하여 X 축에서 사용되어진 것과 동일한 광센서를 원점위치에 설치하였으며, 동력 전달 기구들은 기구의 보호 및 분진 및 안전을 위해 커버(Cover)를 설치하였다. X축의 동력 전달은 타이밍 벨트(Timing Belt) 및 감속기를 이용하였으며, Y축 및 Z축은 볼나사(Ball Screw)를 이용하여 동력전달을 했다. 그리고 회전축은 유성치차 감속기를 통해서 동력이 전달되도록 설계 제작 하였다. X축의 감속기는 10:1의 감속비를 갖으며, 모터 축과 90° 방향이 전환되어 동력 전달을 한다.

Y축 및 Z축의 볼나사 감속기는 리드(Lead)가 5㎜로 설계하였다. 다음 그림은 로봇 암의 여러 면의 사진이며 수박 돌리기용 진공 그리퍼가 장착된 모습이다.

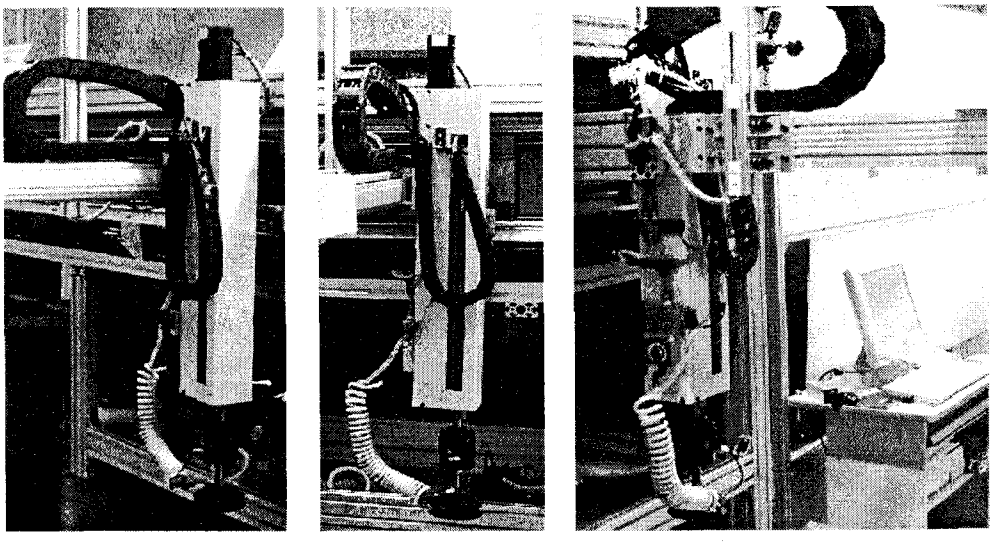

Fig. 3-5-12 Robot arm with gripper (2nd prototype).

다음 그림은 돌리기 작업용 그리퍼가 장착된 사진이다.

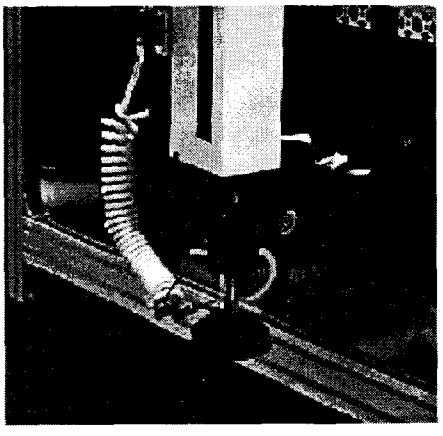

Fig. 3-5-13 Gripper for rolling.

여기서 그리퍼와 로봇 암의 선단부를 클립(Clip)식으로 연결할 수 있게 하여 타 모듈형 작업 장치를 작업별로 쉽게 교체할 수 있게 하였으며, 공압, 진공, 센서, 동력선 등을 나선형으로 함으로 선단부의 수직이동에 대해 유연한 접속을 할 수 있게 하였다.

그리고 매거진 선단부에 여러 종류의 모듈러형(Modular Type) 작업기가 장착되므로 각 장치마다 소요되는 사양이 다르므로 각 모듈별 사양을 고려하여 공통 및 전용 커넥터를 마련하였으며, 쉽게 장/탈착을 할 수 있게 하였다.
그리고 진공 발생기의 경우 형식은 1차 시작기와 동일하지만 설치 위치를 아래 그림과 같이 Z축 베이스에 설치함으로 호스(Hose)의 길이를 최대한 짧게 할 수 있도록 하여 진공 손실을 줄였다.

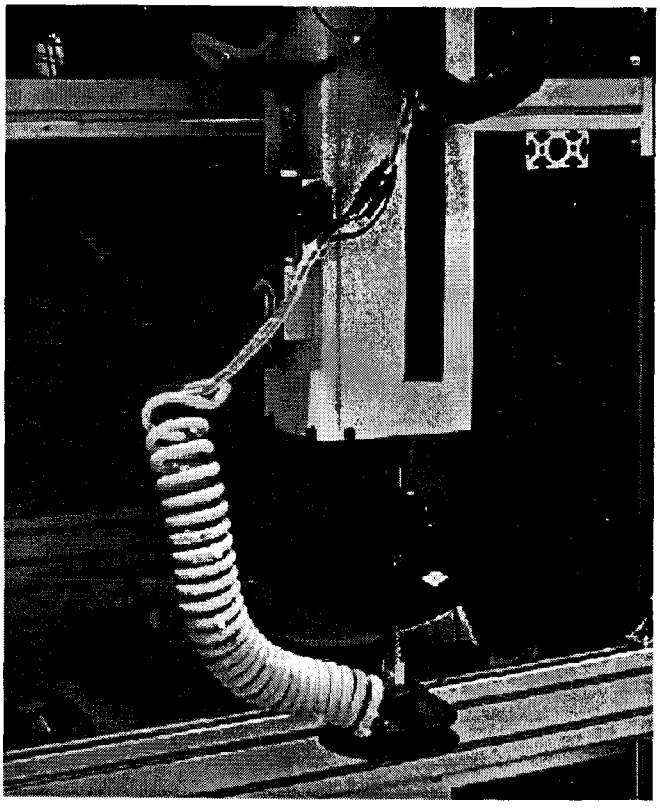

Fig. 3-5-14 Vacuum generator via multi-venturi.

라. 영상처리 시스템

영상처리 시스템은 카메라, 무선 영상 송/수신기, 프레임그래버, 영상 처리용 프로세서인 컴퓨터로 구성하였으며, 2차 시작기에서는 무선으로 영상데이터(NTSC)를 전송하고, 전송된 데이터를 프레임 그래버에서 수신하여 터치스크린이 장착된 모니터에 표시하고 표시된 내용에 따라 작업자가 터치(Touch)를 통하여 무선으로 시스템을 제어하게 하였다.

먼저 카메라의 설치를 위해서 그림과 같이 로봇 암의 Y축 프레임에 40㎜ × 40㎜ 프로파일로 카메라 설치 프레임을 장치하여 로봇 암이 X축 방향으로 이동 될 때

카메라 프레임이 같이 이송되며, 카메라 뷰(View)에서 구간 전 대역을 볼 수 있게 조절하였다.

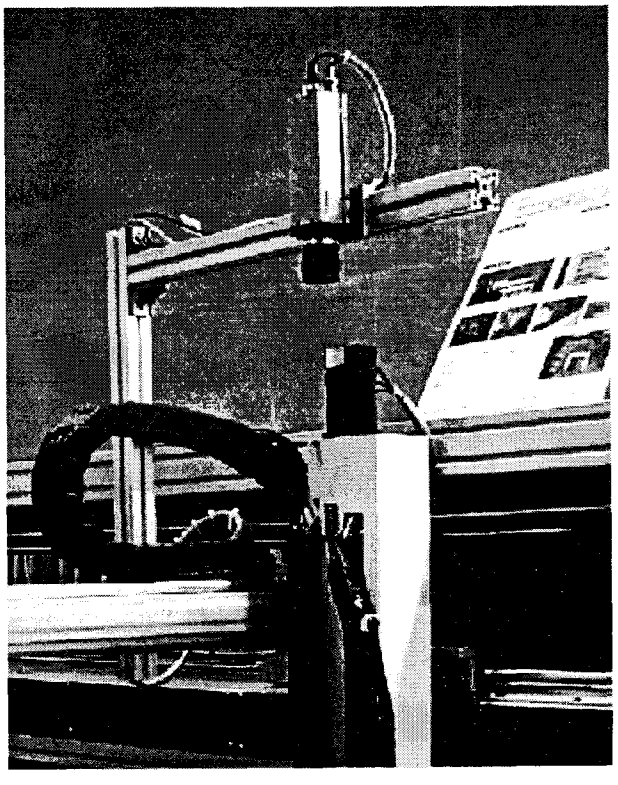

Fig. 3-5-15 Color camera with camera frame.

다음으로 영상을 획득하는 시점에 로봇 암(Arm)이 카메라 뷰 상에 있지 않게 하기 위하여 영상획득 시점에 로봇 암을 원점 위치로 이동 시킨 후 영상획득 작업을 수행하게 하였다.

2차 시작기의 영상처리 시스템은 실험용 정밀 매거진 제작에서 사용된 동일한 사양의 시스템과 무선 송/수신 모듈을 사용하여 구성하였다. 다음 그림은 영상처리 시스템의 무선 송/수신 모듈과 데이터 전송용 무선 모뎀의 사진이다.

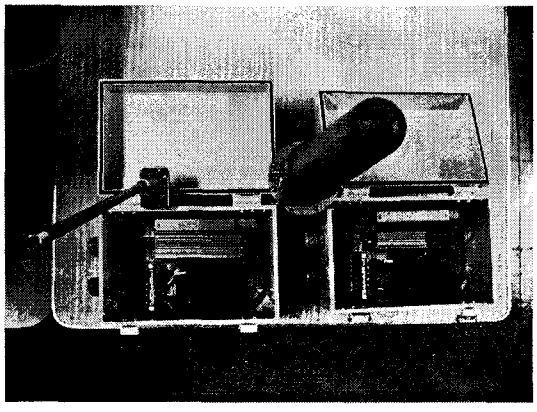

Fig. 3-5-16 R/F system for image and data signal.

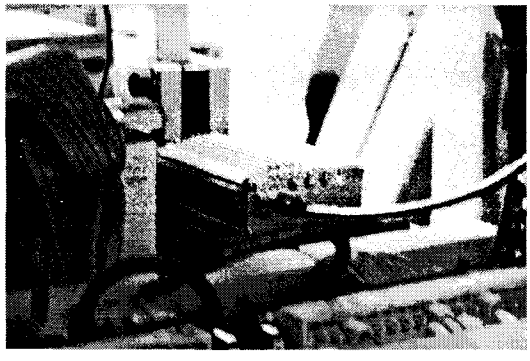

Fig. 3-5-17 R/F module for image.

마. 모듈형 단위 작업 장치

1) 수박 돌리기 그리퍼 2차 시작기

1차 돌리기 그리퍼 시작기에서 기구의 복잡성과 크기 및 무게의 요인으로 실제 시스템에 장착했을 때 동작상의 문제가 발생하였다. 그래서 2차 시작기는 기계적 기능의 일부를 소프트웨어로 구현하여 그림과 같이 소형이며 단순하게 설계하고 제작하였다.

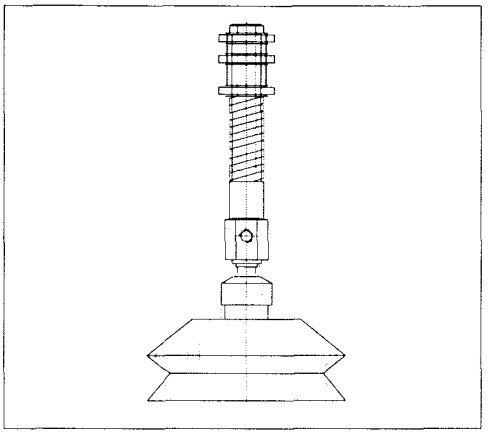

Fig. 3-5-18 Sketch of gripper for rolling.

작업 순서는 다음과 같게 소프트웨어를 통해 반복 작업으로 돌리기를 수행하게 하였다.

① 작업자의 작업물 및 꼭지부 교시
② 영상처리를 통한 작업물의 형상 검출 (크기, 자세, 중심)
③ 작업물의 중심에서 일정량 편심된 지점을 흡착패드로 견인
④ 편심량 만큼 중심점 보상 후 놓기
⑤ 지정된 각도에 이를 때까지 ③, ④를 반복 수행

2) 수박 수확용 그리퍼 2차 시작기

1차 시작기의 실험 결과 칼날의 구조와 칼날 교차 방법 및 크기에 대해서 문제점이 발생하여 그림과 같이 상용 칼날을 이용한 소형 꼭지 절단 장치를 설계 제작하고, 소프트웨어적으로 절단 범위를 넓혔다.

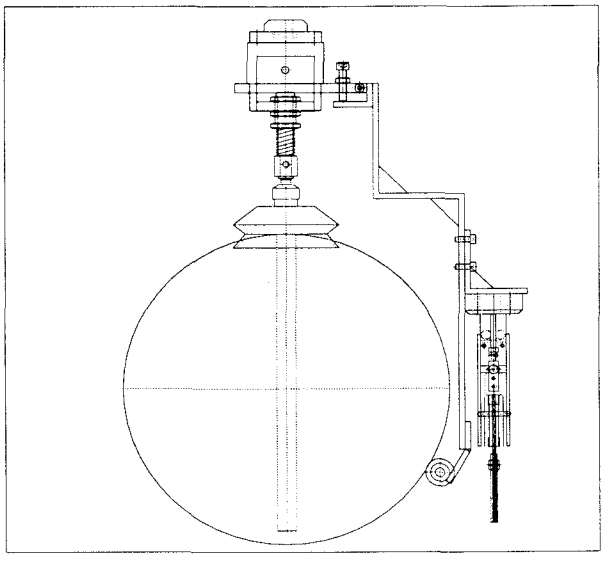

Fig. 3-5-19 Sketch of Gripper for Harvesting

동작 순서는 다음과 같다.

① 진공 패드를 수박에 접근 시키면서 절단기의 연속 절단 시작
② 진공 패드가 표피에 접촉되면 진공 패드로 수박을 파지한다.
③ 파지가 완료되면 Z축을 통한 견인 및 수박 적재부 이송장치로 수박을 이송한다.

바. 전체 제어 시스템

1) 제어 시스템 구성

2차 시작기의 시스템 제어부는 크게 터치스크린을 이용한 GUI형 MMI(Man Machine Interface) 부분과 로봇의 서보 위치제어 및 구동부와 디지털 I/O, 그리고 드라이버 부분으로 나누어 그림과 같이 구성하였다.
먼저 MMI 부분은 5선 압력 저항식 터치스크린이 장착된 15"형 LCD Panel과 컴퓨터로 구성되어 있으며, 작업자의 터치를 통하여 무선으로 시스템을 자동 또는 수동으로 통제할 수 있게 하였다.

다음으로 로봇 암의 위치제어 및 디지털 I/O 부분은 Nais사의 AFP2211 모션 제어기로 구성하여 MMI와 무선 직렬통신(RS232C)으로 연결시켰다. 주 제어기의 모든 명령은 직렬 통신을 통하여 무선으로 이루어지게 하여 무선 원격제어를 수행 하도록 하였다. AFP2211은 4축 Pulse형 위치제어를 수행하는 기능을 가졌으며, 여기에 디지털 I/O 모듈을 연결하여 그림과 같이 구성하였다. 대차 구동용 인버터, 릴레이 및 입력 스위치 및 센서들은 범용 디지털 I/O에 연결하여 각 장치를 구동할 수 있게 하였다.

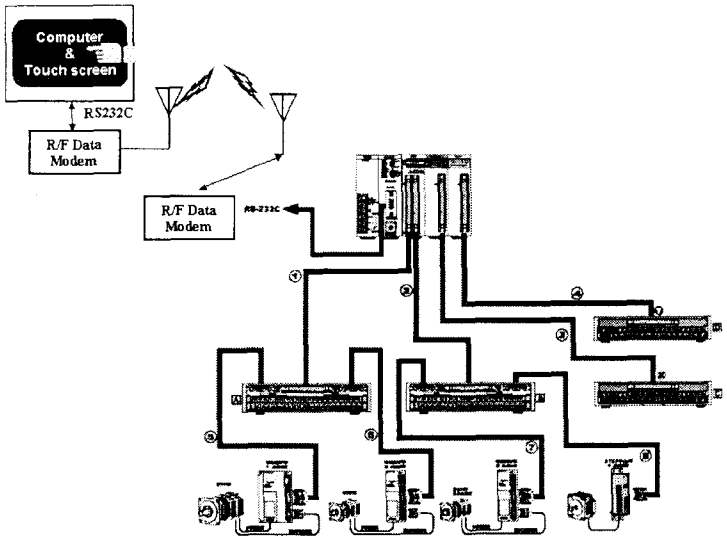

Fig. 3-5-20 Block Diagram for System Control

2) 제어반

시스템 제어반은 크게 무선 모듈부, 제어부와 드라이버부, 전원부, 릴레이 및 배전부로 구분하여 제작하였는데 소신호부와 대신호부, 전원부를 구별하여 배치함으로 상호 전기적인 오류를 발생하지 않도록 하였다. 그리고 열적인 보호를 하기 위하여 열원이 있는 드라이버부를 통풍이 쉬운 곳에 배치하였다. 각종 배선은 배선용 덕트(Duct)에 수납하였으며, 신호배선과 전력선은 분리시켜 배선하였다. 특히 대차 구동용 인버터와 공압 솔레노이드는 별도의 공간에 장착하여 상

호 간섭을 최대한 억제시켰다.

다음 그림은 시스템 제어반의 사진이다.

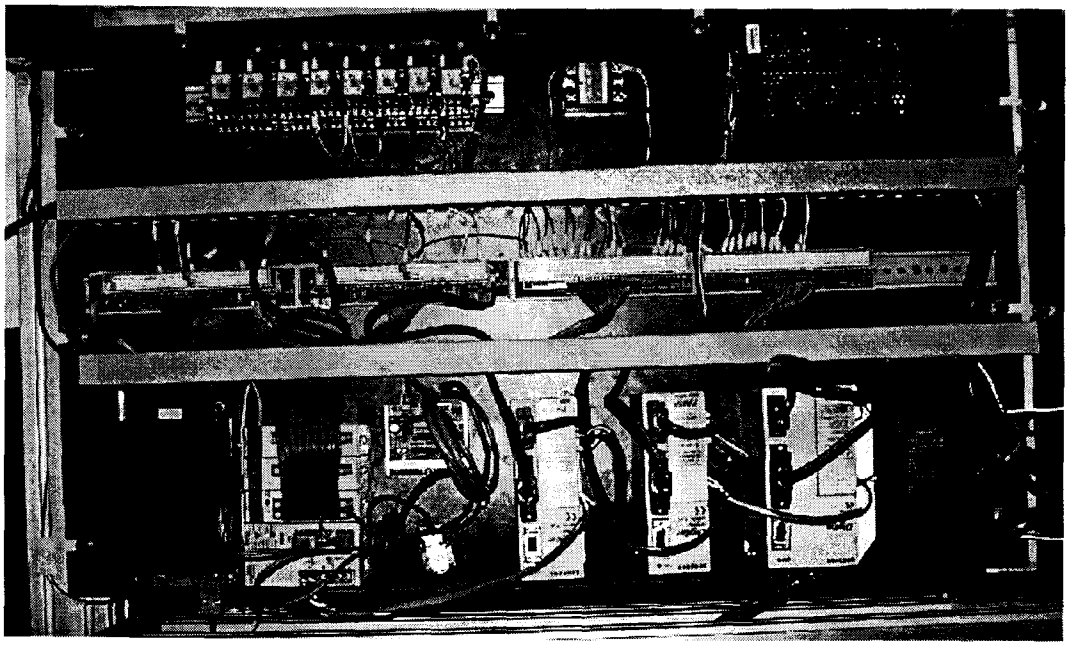

Fig. 3-5-21 Figure of System Controller Circuit(2nd Prototype)

3) 시스템 제어부

시스템 제어부는 Main CPU 모듈과 4축 펄스형 위치제어 모듈, 디지털 입력모듈, 디지털 출력 모듈로 구성하였다.

먼저 Main CPU 모듈은 무선으로 원격지에 있는 주 제어용 MMI와 RS232C 규격의 직렬 통신을 통해 명령 및 데이터를 송/수신하여 그 명령에 따라 작업 시스템을 전반적으로 통제하는 역할을 담당할 수 있게 하기 위하여 직렬 통신이 가능하고, 위치 제어 및 디지털 입출력이 가능하도록 그림과 같이 구성하였다.

시스템 제어부는 펄스형 4축 위치 제어 기능과 32개의 디지털 입력, 32개의 디지털 출력을 할 수 있다. 여기서 디지털 출력의 장치는 Power FET를 이용한 반도체형으로 되어 있어서 고속 출력이 가능하다. 그리고 디지털 입력 장치는 광 절연 되어있어서 외부의 외란으로부터 디지털 I/O 모듈 및 Main CPU 모듈을 보호할 수 있다.

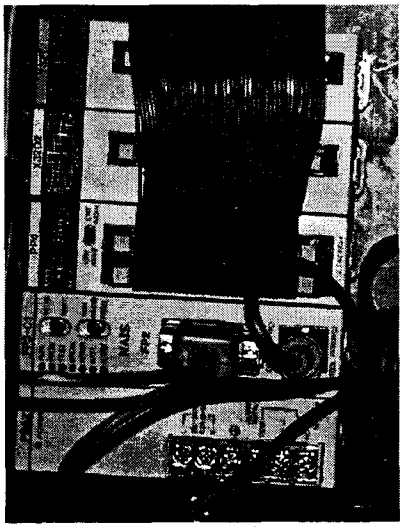

Fig. 3-5-22 Main system controller(2nd prototype).

4) 모터 구동부

가) 로봇 암 구동부

X, Y, Z 로봇 암을 구동하는 서보모터의 드라이버와 회전축을 담당하는 스테핑 모터의 드라이버를 아래와 같이 구성하여 대차 프레임의 상단에 장착하였다. X, Y, Z축 서보모터는 광학식 엔코더(Encoder)를 센서로 하여 모터의 회전 방향 회전량, 속도를 감지하며, 드라이버는 이 신호를 읽어 입력된 위치제어 펄스의 수와 비교함으로 로봇 암의 자세를 제어하게 한다. 사용된 서보 드라이버의 사양은 다음과 같다.

X축 : 1㎾ 드라이버, 2048Pulse/Rev 위치제어, AC Servo Driver
X축 : 400W 드라이버, 2048Pulse/Rev 위치제어, AC Servo Driver
X축 : 200W 드라이버, 2048Pulse/Rev 위치제어, AC Servo Driver
Z축 : 정전류 쵸퍼(Chopper)형 스테핑 모터 드라이버, 2, 1-2상제어

Fig. 3-5-23 Driver for X, Y, Z, Roll Motor

나) 대차 구동부 회로

대차 구동부 회로는 1차 시작기와 동일하며, 브레이크 회로가 생략하고, 소프트웨어로 브레이크를 동작시켜 부드러운 주행과 정지를 수행하게 하였다.

5) 전원부

2차 시작기에서의 전원부는 1차 시작기와 달리 밧데리를 이용한 구동을 하도록 하였다. 사용 밧데리는 12V 200AH × 3개를 사용하였으며, 3㎾ DC-AC 인버터, 밧데리 충전장치로 그림과 같이 구성하여 시스템의 하부 수납장소에 장치하여 사용하였다. 본 밧데리로 연속해서 6시간동안 시스템을 동작 시킬 수 있었으며, 충전 시간은 8시간이 소요되었다.

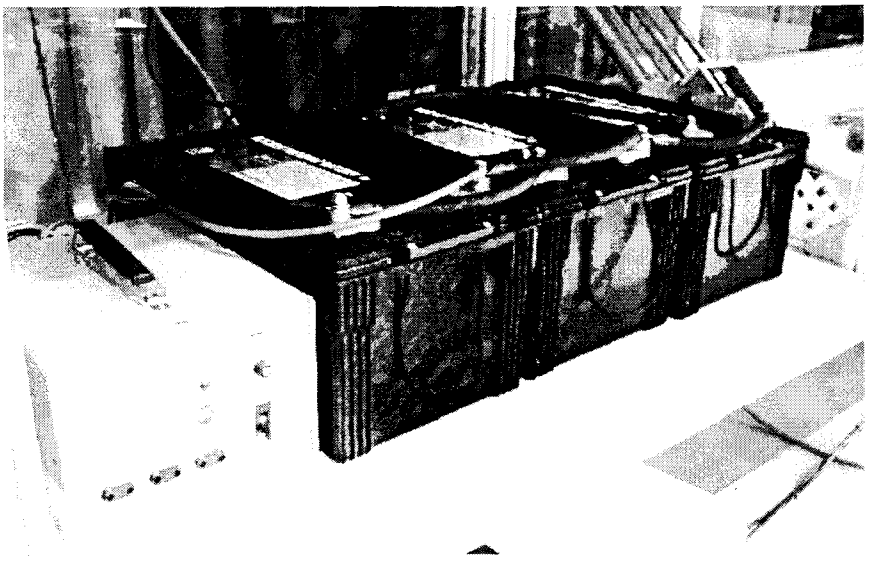

Fig. 3-5-24 Battery power supply.

사. 시스템 소프트웨어

2차 시작기의 시스템 소프트웨어는 컬러 카메라를 통해 입력되는 작업 구간에 대한 영상신호를 무선으로 원격지에 있는 MMI로 보내면 이 영상신호를 수신하여 프레임 그래버로 메모리에 저장 및 모니터에 표시하고, 터치스크린을 통하여 작업자가 작업을 무선 데이터 모뎀으로 지시하면 획득된 영상을 바탕으로 로봇 암의 위치를 이동시키고, 모듈형 작업기를 통하여 명령되어진 작업기를 수행하게 하였다.

각종 명령 버튼 및 작업 지시는 터치스크린을 통해 수행될 수 있게 효율적으로 구성하였으며, 각종 명령 체계는 작업자가 직관적이고 가장 단순한 조작을 통해 수행할 수 있도록 최적화 하였으며, 각 작업마다 필수적인 메뉴 구성과 버튼 구성으로 작업자의 오 조작을 방지하게 하였다. 먼저 로봇 제어 및 디지털 I/O 제어를 위하여 그림과 같이 전용 제어프로그램을 개발하여 시스템이 오류 없이 동작 할 수 있도록 실험을 할 수 있게 하였다. 주 프로그램의 시스템 점검 모드로

들어가면 본 프로그램이 동작하게 하였다.

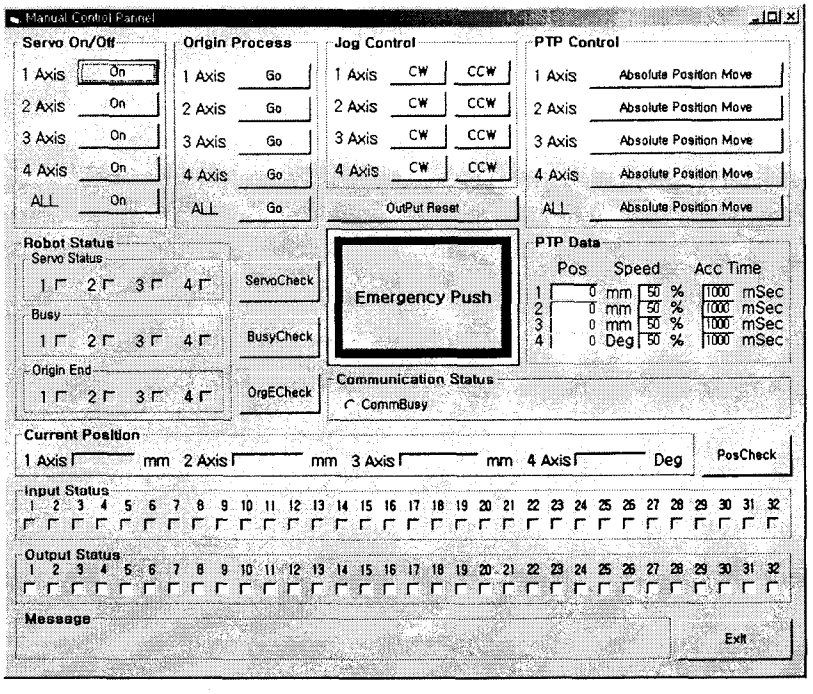

Fig. 3-5-25 Robot Control Program

이 프로그램을 통하여 제어 시스템과 주 제어 시스템간의 무선 통신을 수행하였으며, 이 때 전용 프로토콜(Protocol)을 통하여 시스템에 명령을 하게 하였다.

아래 그림은 2차 시작기의 주 프로그램의 실행 모습이다.

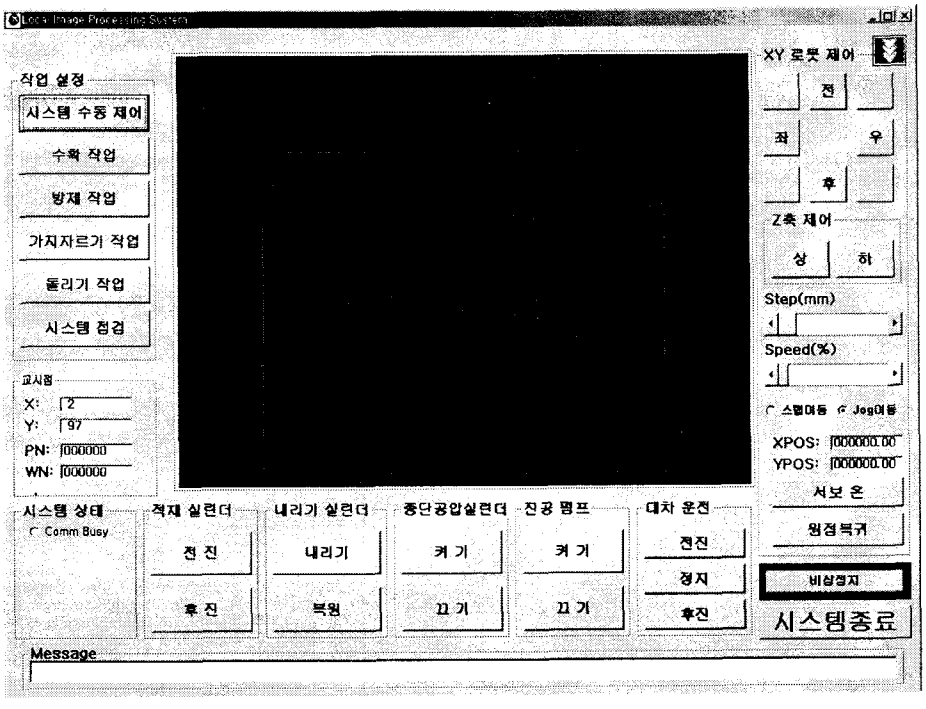

Fig. 3-5-26 Main Program (2nd Prototype)

시스템 소프트웨어의 주 메뉴는 다음과 같게 구성하였다.

- 시스템 수동 제어 : 로봇의 위치 및 속도의 수동 제어, 각종 출력장치의 수동 제어, 입력 데이터 모니터 기능을 가짐. 시스템 제어기에 사용자가 직접 명령을 입력할 수 있음.
- 수확 작업 모드 : 수확 작업을 수행하기 위한 일련의 작업을 작업자의 1차교시를 통하여 수행하게 함.
- 방제 작업 모드 : 정밀 방제 작업을 작업자 교시를 통하여 일괄적으로 수행함.
- 가지자르기 작업 모드: 곁가지 제거작업을 수행하는 작업모드이다.
- 돌리기 작업 모드 : 수박을 돌리기를 위한 작업 모드이다.
- 시스템 점검 모드 : 무선으로 영상 및 데이터를 송/수신하므로 무선 원격 제어 시스템 점검 모드를 두어 원격지에서 무선으로 시스템의 다양한 기능을 제어하게 하는 모드이다.

1) 시스템 수동제어

시스템 수동 제어 모드는 크게 로봇 시스템 제어, 적재 작업 제어, 각종 입출력 제어, 직접 무선명령으로 아래 블록도와 같이 구성하였다. 시스템 수동제어는 시스템의 점검, 작업 환경의 변경, 카메라 보정을 위한 로봇 이송 등을 할 수 있게 하는 작업 모드이다. 그러나 필요에 따라 작업 과정에서도 임의의 시스템 조작이 필요한 경우 작업자의 의지에 따라 시스템을 조작 할 수 있게 하여 시스템 운용에 융통성을 부여하게 했다.

1차 시작기와 달리 로봇 암에 대한 직접명령이 추가되었다.

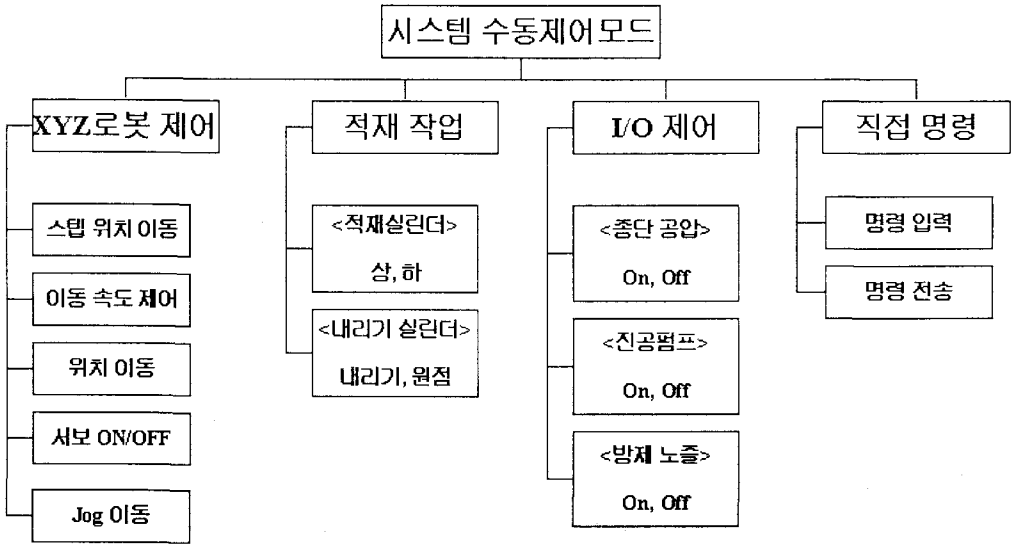

Fig. 3-5-27 Block Diagram of Manual Mode(2nd Prototype)

2) 수확 작업 모드

수확 작업 모드는 수박을 수확하는 일련의 작업을 작업자의 수박 위치 및 꼭지의 위치 교시를 통하여 수행하게 하는 작업모드이다.

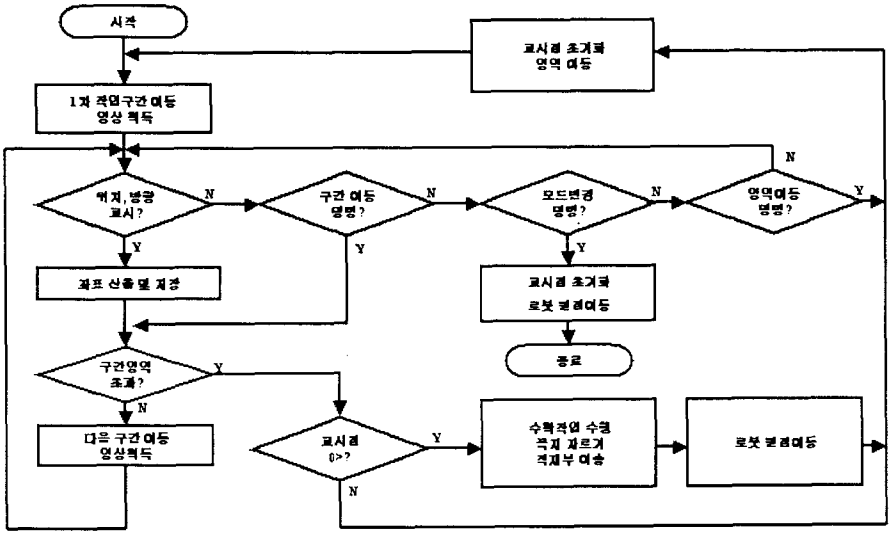

Fig. 3-5-28 Block Diagram of Harvesting Mode

위 그림은 수확 작업 모드의 순서도이다. 여기서 작업자의 작업 명령에 따라 작업이 수행되는데, 먼저 작업구간별로 수박 위치 교시 및 꼭지 방향 교시가 완료되면 교시되어진 좌표를 산출하고 각 목표물에 대한 수확작업을 수행하여 수확물을 적재 장치를 통해 적재를 수행하게 하였다.
1차 시작기와 작업 방법은 동일하나 매거진의 이동이 없이 로봇 암으로만 구간을 이동함이 다르다.

수확 작업은 처음 교시점을 시작으로 작업을 진행하는데 진행 방법을 다음과 같게 하였다.

- 로봇을 원점 위치로 이동 시킨 후 교시점이 정의된 작업 구간으로 로봇을 이동시킨다.
- 로봇을 통하여 수확 작업용 그리퍼를 수확물 중앙점으로 이동시킨다.
- 회전축의 회전을 통하여 그리퍼의 자세를 정렬한 후 Z축을 통하여 하강하면서 수박 표면으로 유도되는 절단날을 작동시킴으로 꼭지 자르기를 수행한다.
- 진공 패드가 완전히 흡착되면 수박을 견인하여 수박 적재함 이송장치로 이송

한다. 이 때 수박 적재함 이송장치는 하강한다.

- 하강된 적재함 받이에 로봇 암을 위치시키고 진공을 끔으로 수박을 적재함 받이에 내려놓는다.
- 수박 적재함 이송장치를 후진시키고 수박을 적재함으로 유도함으로 수확 작업 완료.

여기서 작업 구간이란 한 번에 영상처리를 할 수 있는 영역을 말한다.
그리고 작업 영역이란 대차의 움직임이 없고, 로봇 암이 이동하여 작업할 수 있는 공간을 말하며, 영역 이동이란 대차를 이동하여 새로운 작업 공간으로 이동함을 말한다.

3) 방제, 돌리기, 가지자르기 작업

1차 시작기와 달리 돌리기 작업과 가지자르기 작업을 추가하였다. 가지자르기와 돌리기 작업은 방제작업과 같은 프로세싱을 수행하는데, 다만 작업기 동작만 상호 다르게 구성하였다. 방제, 돌리기, 가지자르기 작업 모드는 작업자의 방제위치 교시를 통하여 수행하게 되는 작업모드이다.

Fig. 3-5-29는 방제, 돌리기, 가지자르기 작업 모드의 순서도이다. 여기서 작업자의 작업 명령에 따라 작업이 수행되는데, 먼저 작업구간별로 방제위치 교시가 완료되면 교시되어진 좌표를 산출하고 각 목표물에 대한 작업을 현재 작업 영역에 대하여 일괄적으로 수행하였다. 보통 한 작업 구간에 1개 이상의 교시점이 존재하게 되므로 구간 이동 명령을 통하여 현재 작업 구간에서 모든 교시가 끝나게 하였다. 구간 이동 명령을 하면 현 구간에서 교시된 모든 좌표를 산출 및 저장하고 다음 구간으로 이동하게 하였다. 만약 영역 이동일 경우에는 현재 작업영역의 작업을 중단하고, 교시된 데이터를 지운 후 다음 작업 영역으로 이동하게 하였으며, 작업자의 명령이 모드 종료이면 현재 영역에서 교시한 데이터를 모두 무시하고 매거진 및 로봇을 원점복귀 시킨 후 방제모드를 종료하게 하였다.

2차 시작기의 작업은 처음 교시점을 시작으로 작업을 진행하는데 진행 방법을 다음과 같게 하였다.

- 로봇 암을 원점 위치로 이동 시킨 후 교시점이 정의된 작업 구간으로 이송 시킨다.
- 로봇 암의 이송이 완료되면 로봇을 통하여 모듈형 작업 장치를 이동시키는데, 전체 구간에서 경로를 최적화 시킨 좌표데이터를 통하여 방제 작업이 이루어지게 하였다.
- 작업위치로 작업기를 이동시키고, 작업기 제어용 솔레노이드를 작동시킴으로 작업을 수행한다.
- 전 영역에 대하여 방제 작업이 완료되면 로봇 암을 원점복귀 시키고, 다음 작업 영역으로 대차를 이동시킴으로 방제 작업 완료

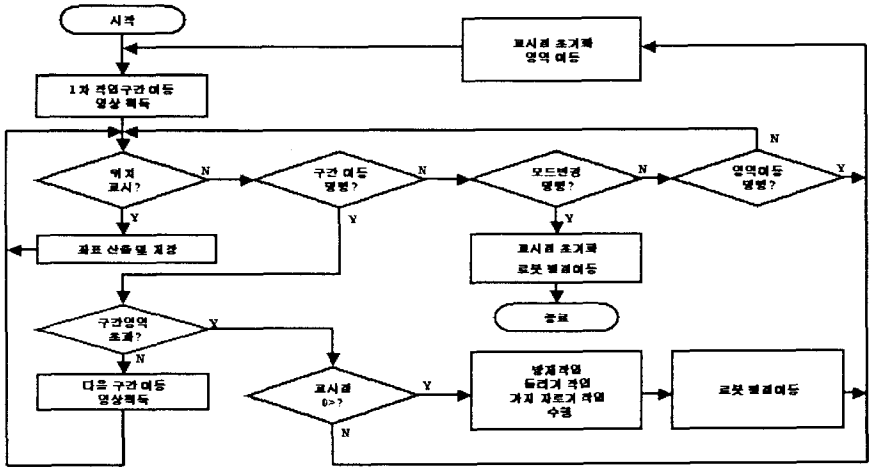

Fig. 3-5-29 Block Diagram of Job Processing (2nd Prototype)

4) 시스템 점검 모드

시스템 점검 모드는 원격으로 시스템을 제어 할 때 무선 데이터 프로토콜에 의해서 명령 전송 및 데이터 송/수신이 이루어지는데 이 때 명령 전송이 제대로

수행되는지 모니터 할 수 있고, 데이터 교환이 원활이 수행되는지에 대한 점검을 하기위하여 개발 되었다.
이 모드는 로봇 암의 원점 복귀, 서보 On/Off, Jog 컨트롤, 로봇 암의 Point to Point 위치 이동, 로봇 상태, 외부 디지털 입/출력 상태 모니터 및 명령, 비상정지와 같은 기능을 갖게 하여 로봇 시스템을 전반적으로 제어할 수 있게 하였다. 원격 데이터 전송 포맷은 직렬 전송 방식이고, 프로토콜은 MEWTOCOL 명령 체계를 갖게 하였다.

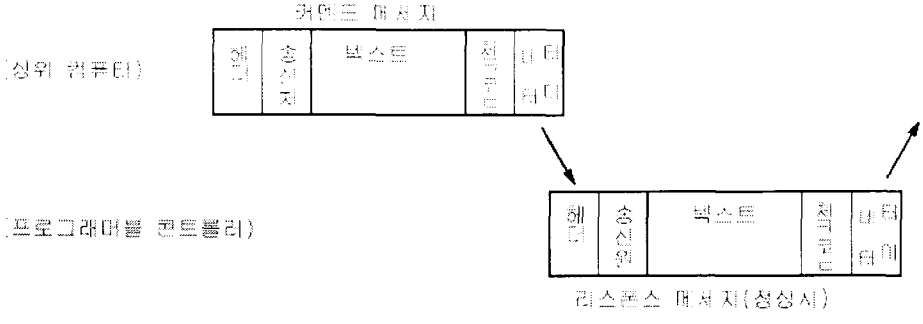

Fig. 3-5-30 Format of MEWTOCOL protocol.

주 제어기의 데이터 포맷은 그림에서와 같은 구조를 가지게 하였으며, 명령 송신 결과를 즉시 수신 받을 수 있어 올바른 송/수신이 이루어졌는지를 검사하게 하였다.

다음 표는 제어코드이다. 체크 코드는 전체 명령에 대한 2진값의 합에 대한 마지막 바이트 값이며, 이 값을 통하여 데이터 전송이 재대로 수행되었는지 알 수 있으며, 터미네이터는 메시지 종료를 나타낸다. 모든 명령은 텍스트 형태로 전송된다.

<Table 3-5-1> Code of Control for MEWTOCOL

명칭	캐릭터	ASCII코드	설명
헤더	% 또는 <	25H또는 3CH	메세지의 개시를 나타낸다.
커멘드	#	23H	커멘드 · 메세지인 것을 나타낸다.
리스폰스(정상)	$	24H	정상적인 리스폰스 · 메세지인 것을 나타낸다.
리스폰스(이상)	!	21H	에러시의 리스폰스 · 메세지인 것을 나타낸다.
터미네이터	C_R	0DH	메세지의 종료를 나타낸다.
딜리미터	&(+C_R)	26H	복수 프레임에 분할할 때의 구별을 나타낸다.

3. 성능 시험

가. 시험 환경

다목적 생력 시스템 2차 시작기의 성능시험은 수확 작업 시험, 방제 작업 시험, 돌리기 작업 시험, 가지자르기 시험으로 나누어 실시하였다. 성능 시험을 하기 위하여 실험실의 인공 재배 환경을 구축하고, 실내의 인공광(형광등) 하에서 실시하였다.

공시 재료로는 대과종 4개, 중과종 2개를 이용하였으며 중량은 3kg~8kg이였으며, 형태는 원형, 타원형을 사용하였다. 인공 재배 환경은 바닥면을 갈색 카페트를 이용하여 토양을 대신하였고, 줄기 및 잎은 장식용 넝쿨을 사용하여 실제 현장과 같은 복잡영상을 만들었다.

나. 수확 작업 성능 시험 결과 및 고찰

수확 작업 성능 시험 항목은 수확 시간으로 하였다. 영상 획득 및 1차 교시 시간을 제외한 순수한 작업기 동작 시간을 측정하였다. 시험 결과를 요약하면 다음과 같다.

2차 시작기는 1차 시작기와 달리 X축 전 구간을 서보 제어를 하였으므로 S자형 가/감속을 수행하였다. 그 결과 1차 시작기보다 빠른 성능을 보였으며, 보다 안정적인 동작을 할 수 있었다. 그리고 진공 발생장치를 로봇 암에 부착하여 보다 근거리에서 진공을 발생시키게 하였으므로 진공도 저하로 인한 성능 저하를 막았다. 그 결과 보다 중량물이더라도 파지를 잘 할 수 있었고, 이송 속도 또한 빠르게 할 수 있었다.

대과종, 중과종 동(同)이 수확작업 소요시간은 15초 정도에 처리하였다. 전체 시간을 구간별로 나누어 보면 꼭지 자르기에 10초가 소요되었으며, 그 나머지 시간은 이송 시간이다. 로봇 암 이동에 5초가 소요되었다. 여기서 로봇 암의 동작을 X, Y, Z축이 동시에 동작할 수 있게 하여 작업 시간을 줄이도록 하였다.

돌리기 작업의 경우 2차 시작기의 돌리기 성능은 1개의 수박을 돌리는데 평균 10초가 소요되었다. 이는 구조를 단순하게 한 대신 소프트웨어적으로 3회 반복 돌리기를 수행한 결과이다. 가지자르기 및 방제의 경우 하나의 가지를 자르는데 평균 4초가 소요되었으며, 이 때 로봇 암의 이송시간이 3초가 소요되었다.

2차 시작기의 전체 작업을 크게 교시작업 및 영상처리, 구간이송, 실제 작업으로 나눌 수 있는데 교시작업의 경우 작업자의 능률에 따라 달라질 수 있으며, 영상처리의 경우 0.3초가 소요되었으며, 구간이동은 평균 1초 정도 소요되었다. 그리고 실제 작업의 경우 수확작업이 15초, 돌리기 작업이 10초, 가지자르기 및 방제 작업의 경우 4초가 소요됨을 실험을 통해 알 수 있었다.

시스템 실험 결과 전체 시스템 프로세싱에 대한 연구와 하우스 간 이동, 다중 시스템 제어 등에 대한 연구를 통하여 실제 시설에 적용될 수 있으리라 사료된다.

제 4 장 결론 및 요약

수박에 대한 영상처리를 이용한 무선 원격지에서의 재배관리 시스템을 2차에 걸쳐 설계 제작 하였으며, 이를 이용하여 수확, 방제, 돌리기, 가지자르기 실험을 수행하였다.

1차 시작기에서는 터치스크린을 이용한 MMI의 구현과 영상처리 기술, 기초 실험이 수행되었고, 2차 시작기에서는 1차 시작기에서의 단점을 보완하면서 무선 원격 제어 기술을 접목시켰다. 그리고 모듈형 작업기에 대한 성능 평가를 하였다.

그 결과 다음과 같은 결론을 얻을 수 있었다.

- 작업자의 1차 교시를 통한 현장에서 영상처리 시스템개발의 결과로 기존의 전자동 영상처리 시스템에서 불가능하였던 현장 적응성을 알 수 있었다. 실제 작업의 경우 영상처리 시간은 평균 0.3초에 이루어졌다.
  수박의 경우와 같이 배경과 수박간의 구분이 명확하지 않는 작업 대상체에도 잘 동작하였으므로 타 작물에 대한 적용도 손쉽게 이룰 수 있음을 알 수 있었다.

- 무선 원격 제어 시스템의 구축 및 터치스크린을 통한 시스템 제어, 미래 지향적인 인터페이스 구축을 통하여 작업 환경의 대폭적인 개선 가능성을 얻었으며, 향후 통합 시스템 제어를 구축하여 재배 정보, 시스템 관리, 시설 관리를 종합적으로 수행할 수 있으리라 사료된다.

- 시설 내에서 재배 작업 전반을 수행할 수 있는 대차 시스템의 개발을 통하여 전용기가 아닌 모듈형의 작업기 장착의 적용에 대한 가능성을 얻었다.
  그리고 아치형 표준 시설에서의 다목적 재배관리 작업 시스템의 기초 모델을 정립 할 수가 있었다.

- 모듈형의 다양한 작업 장치의 개발을 통하여 앞으로 다양한 작업에 본 시스템을

적용시켜 종합적인 재배관리를 단일 시스템으로 이룰 수 있는 가능성을 얻었다. 수확 작업의 경우 평균 15초, 돌리기 작업의 경우 평균 10초, 방제, 가지자르기 작업의 경우 평균 4초의 성능을 보였으므로 현장 적용이 충분하다고 사료된다.

제 5 장 참고문헌

[1] 진홍청. 1994. 작목별 작업단계별 노동력 투하시간.

[2] 박원규. 1992. 한국의 농업구조개선과 농업기계화 방향. 농림축산 과학기술 심포지움.

[3] 이종호, 박승제, 김철수, 이중용, 김용현. 1993. 고추수확기 개발을 위한 기초 연구. 한국농업기계학회 18(2):pp. 110-121.

[4] 황헌. 1992. 채소용 이식기 개발을 위한 기초연구. 한국농업기계학회 산학협동 보고서.

[5] 김기대. 1996. 시설재배용 로봇 개발 현황과 발전: 일본의 개발 사례 중심으로. 농업용 로봇 연구개발 동향과 전망 pp.61-75

[6] Bernacki, H., J. Haman and Cz. Kanafojski. 1967. Agricultural Machines, Theory and Construction. Vol. 1. U.S. Department of Agriculture and National Science Foundation, Washington D.C.

[7] Hwang, H. and F. Sistler. 1986. A Robotic Pepper Transplanter. Trans. of ASAE Vol.2 No.1 pp.2-4.

[8] Ikumoto, O et. al. 1989. Autonomous Travelling System for Agricultural Vehicle, Proceedings of The 2nd Advanty Symposium on Vehicle Automation, pp.109-112.

[9] Ito, N. 1983. Application of Agricultural Robots in Japan, Robotics and Intelligence Machines in Agriculture, Proceeding of the First International Conference on Robotics and Intelligent Machines in Agriculture, pp. 63-75.

[10] Ito, K et. al. 1990. An automonic Guidance System úf a Lawn Tractor, Proceeding of The 1st Advanced Symposium of Vehicle Automation, pp.5-8.

[11] Jane. H. Pejsa and James E. Orrok. 1983. Intelligent Robot System: Potential Agricultural Applications, Robotics and Intelligent Machines in Agriculture, pp.104-111.

[12] Kanetoh, I. 1976. Driverless Combine Harvester, Grain and Forage Harvesting Conference Proceedings. ASAE.

[13] Kito, K. 1981. Development of the Microcomputerized Driverless Combine The Bulletin of the Faculty of Agriculture, Mie University, No.63, pp.235-255.

[14] Kito, K. 1986. Optimum Control of Rice Combine Harvester, Ph.D thesis, Kobe University.

[15] Kobayashi, K. 1991. Development of a Grafting Robot for Fruit-Vegetables. Plant cell technology 3(6), pp.477-482.

[16] Krutz, Gary W., Lester Thompson and Paul Claar 1984. Design of Agricultural Machinery. John Wiley & Sons, New York, NY.

[17] Matsuo, Y et. al. 1989. Research on Automonous Traveling System in IAM(Institute of Agricultural Mechanization), Proceeding of The 2nd Advanty Symposium on Vehicle Automation, pp.113-116.

[18] Matthews, G. A. 1979. Pesticides Application Methods. Longman Group Ltd.

[19] Mitsubishi Co. Ltd. Oct, 1989. R&D Report on Robotic Fertilizer for Rice.

[20] Okuyama, S. 1989. Some Examples of the Successful Developments of Agricultural Robots, The 38th Symposium Text for Japan Association of Industrial Robots, pp.56-60.

[21] Roberts, W. J. 1988. Int'l Symposium in High Technology in Protected Cultivation. YANMAR Noki Co. Ltd. 1980. Fully Automated Driverless Combine Harvester "ELECTRO".

[22] Yamashita, M et. al. 1990. Fuzzy Steering Control for Rice Translater, Journal

of Kansai Branch of JSAM(Japan Society of Agricultural Machinery), No.68, in press.

[23] 양동훈 1999. 수박재배기술, 주요작물재배기술. pp.77-97, 흥농종묘

[24] 유성철 1995. 시설 채소 재배, 오성출판사

[25] 농촌 진흥청, 1988. 시설 원예, 표준영농교본-4, 농촌진흥청

[26] 홍규현. 1995. 상품성이 높은 수박 생산을 위한 품종 선택. 새농사 95, 3월호)

[27] 이상규 1998. 과채류 접목재배 기술 체계화 연구 : 접목방법과 대목의 종류가 수박의 생육 및 품질에 미치는 영향, 시험연구보고서 채소.화훼.시설.환경편 pp.171-178, 1998, 농촌진흥청 원예연구소

[28] 농촌 진흥청 2001. 농가 보급형 자동화 하우스 표준 설계서, 농촌진흥청

[29] 이대원 1996. 머니퓰레이터 및 엔드이펙터 개발 현황과 연구과제, 농업용 로봇 연구 개발 동향과 전망, pp.33-55, 농촌진흥청

[30] 황헌, 1990, 생물자원 생산 자동화 시스템과 로보틱스 기술, 농업생산 시스템의 자동화와 첨단기술, 한국 농업기계학회 심포지움 발표문 pp.149-182

[31] Mark E. Rosheim 1989 Robot Wrist Actuators, WILEY

[32] Mitsuji Monta 2000. Robots for Bioproduction System, Bio-Robotics II 2nd International Workshop. pp.1-11, IFAC

[33] Satoru Sakai, Michihisa Iida 2000 Heavy Vegetable Harvesting Robot, Bio-Robotics II 2nd International Workshop. pp.47-50, IFAC

[34] Qixin Cao, M.Nagata,.. 2000, Basic Study on Strawberry Harvesting Robot, Bio-Robotics II 2nd International Workshop. pp.57-64, IFAC

[35] T.Fujiura, K.Ueda, S.H.Chung, N.Kondo 2000 Vision System for Cucumber-Harvesting Robot, Bio-Robotics II 2nd International Workshop.

pp.65-69, IFAC

[36] Roger Y. Tasi, 1986. An effcicnt and accurate camera calibration technique of 3d machine vision, IEEE, pp. 364-374

[37] Yi Ping Hung,1988. A simple real-time method for calibrating a camera mounted on a rabot for three dimension machine vision, SPIE, Vol 1005 pp. 12-19

[38] J.K Kearmey, 1989. Camera calibration using geometric constraints, IEEE, pp.672-679

[39] Yuncai Liu, Thomas S. Hunang, 1990. Determination of camera location from 2-d to 3-d line and point correspondences, IEEE Transactions, Vol. 12, No1, January, pp.28-37

[40] 이충호,1991, 컴퓨터시각에 의한 측정기술 및 측정오차의 분석 및 보정, 성균관대학교 석사학위논문

[41] Ling-Ling Wang and Wen-Hsiang Tsai, 1991. Camera calibration by vanishing lines for 3-d computer vision, IEEE Transactions, Vol. 13 No 4, April, pp.370-376

[42] Janne Heikkila and Olli Silven, 1997. A four~step camera calibration procedure wiht implicit image correction, IEEE, pp.1106-1112

[43] Du Q. Huynh, 1997. Calibration of a structured light system a prohective approach, IEEE, pp.225-230

[44]. 황헌, 장영창, 임동혁, 1998. 컴퓨터 시각과 레이저 구조광을 이용한 물체의 3차원 정보추출, 한국농업기계학회지 제 23권 제 4호, pp. 381~390

[45] 이혁동,김기대,김찬수 1999. 조직배양제 이식 로봇 시스템의 개발(II)-기계시각 시스템 , 한국 농업 기계학회지 제 24권 제 1호, pp.41~49

[46] 배영환, 구현모 1999. 영상처리에 의한 장미 선별 한국 농업 기계학회지 제 24권 제 1호, pp.67~74

[47] 스테핑 모터의 제어회로 설계 1992 도서출판 세운

[48] Gonzalez and woods 1998. Digital image processing pp.48~68

[49] 이충호 1995. 컴퓨터 시각에 의한 건표고의 외관 검색 및 자동 선별 시스템 개발. 성균관대학교 박사학위논문

[50] 이화조 1996. 농업용 로봇 개발을 위한 첨단 산업용 로봇 기술의 동향과 전망, 농업용로봇 연구 개발 동향과 전망, pp.105-119, 농촌진흥청

[51] 손재룡 외. 1998. 영상처리에 의한 수확용 토마토 인식 시스템 개발, 농경.농기계 논문집 40(2), pp.200-208

[52] 최규홍 외. 1999. 컬러 영상처리를 이용한 사과의 결점 판정, 한국농업기계학회 1999년 동계 학술대회 논문집 Vol.4(1) pp.575-580

[53] 류찬석,류관희 2000. 식물공장용 포트묘 보식 시스템 개발, 한국농업기계학회 2000년 동계학술대회 논문집 Vol.5(1) pp.251-259

[54] 이대원, 김현태, 2001. 젖소 체중측정을 위한 영상처리 시스템, 한국축산시설환경학회지 Vol.7(3) pp.183-190

[55] 손재룡, 2001. 칼라 영상처리에 의한 결주 및 불량모 인식, 한국농업기계학회지 Vol.26(3) pp.253-262

[56] 최승묵 외. 2002. 컴퓨터시각을 이용한 절화류 정밀 선별시스템 개발, 한국농업기계학회 2002년 하계학술대회 논문집 Vol.7(2) pp.298-303

[57] 중앙전파관리소, 2001. 전파법령

[58] D. M. Pozar, 1990, Microwave Engineering, Addison Wesley

<부록>

1. 로봇 제어 프로그램

가. 통신 프로그램

```
Public Sub chkOutput_Click(Index As Integer)
    If Index > 3 And CheckOutputFlag = True Then
        OutputFlag = True
        OutputIndex = Index
    End If
End Sub

Private Sub Output(Index As Integer)
    chkOutput(Index).Enabled = False
    If Index < 16 And Index > 3 Then
        Command = "WCSY010"
        If chkOutput(Index).Value = 1 Then
            OutDataStr = Hex(Index) + "1"
        Else
            OutDataStr = Hex(Index) + "0"
        End If
    ElseIf Index > 15 Then
        Command = "WCSY011"
        If chkOutput(Index).Value = 1 Then
            OutDataStr = Hex(Index - 16) + "1"
        Else
            OutDataStr = Hex(Index - 16) + "0"
        End If
    End If
    CommandStr = CommandHeader + Command + OutDataStr + BCC
    lblMessage = CommandStr
    Call SendData
    Call CheckResponse
    If InStr(ResponseStr, "$") = 0 Then
        If Index > 3 Then
            If chkOutput(Index).Value = 1 Then
                chkOutput(Index).Value = 0
            Else
                chkOutput(Index).Value = 1
            End If
        End If
    End If
    chkOutput(Index).Enabled = True
End Sub

Public Sub cmdAbsMove_Click(Index As Integer)
    AbsMoveFlag = True
    AbsMoveIndex = Index
End Sub

Private Sub AbsMove(Index As Integer)
    cmdAbsMove(Index).Enabled = False
    Command = "WD"
    Select Case Index
        Case 0: OutDataStr = CalAccTime(0, txtAccTime(0))
                CommandStr = CommandHeader + Command + AccTime1DT + OutDataStr + BCC
        Case 1: OutDataStr = CalAccTime(1, txtAccTime(1))
                CommandStr = CommandHeader + Command + AccTime2DT + OutDataStr + BCC
```

```
        Case 2: OutDataStr = CalAccTime(2, txtAccTime(2))
                CommandStr = CommandHeader + Command + AccTime3DT + OutDataStr + BCC
        Case 3: OutDataStr = CalAccTime(3, txtAccTime(3))
                CommandStr = CommandHeader + Command + AccTime4DT + OutDataStr + BCC
        Case 4: OutDataStr = CalAccTime(0, txtAccTime(0)) + CalAccTime(1, txtAccTime(1)) + CalAccTime(2, txtAccTime(2)) +
CalAccTime(3, txtAccTime(3))
                CommandStr = CommandHeader + Command + AccTimeADT + OutDataStr + BCC
    End Select
    Call SendData
    Call CheckResponse
    Command = "WD"
    Select Case Index
        Case 0: OutDataStr = CalSpeed(0, txtPTPSpeed(0))
                CommandStr = CommandHeader + Command + Speed1DT + OutDataStr + BCC
        Case 1: OutDataStr = CalSpeed(1, txtPTPSpeed(1))
                CommandStr = CommandHeader + Command + Speed2DT + OutDataStr + BCC
        Case 2: OutDataStr = CalSpeed(2, txtPTPSpeed(2))
                CommandStr = CommandHeader + Command + Speed3DT + OutDataStr + BCC
        Case 3: OutDataStr = CalSpeed(3, txtPTPSpeed(3))
                CommandStr = CommandHeader + Command + Speed4DT + OutDataStr + BCC
        Case 4: OutDataStr = CalSpeed(0, txtPTPSpeed(0)) + CalSpeed(1, txtPTPSpeed(1)) + CalSpeed(2, txtPTPSpeed(2)) + CalSpeed(3
txtPTPSpeed(3))
                CommandStr = CommandHeader + Command + SpeedADT + OutDataStr + BCC
    End Select
    Call SendData
    Call CheckResponse
    Command = "WD"
    Select Case Index
        Case 0: OutDataStr = CalPos(0, txtPTPPos(0))
                CommandStr = CommandHeader + Command + PTPPos1DT + OutDataStr + BCC
        Case 1: OutDataStr = CalPos(1, txtPTPPos(1))
                CommandStr = CommandHeader + Command + PTPPos2DT + OutDataStr + BCC
        Case 2: OutDataStr = CalPos(2, txtPTPPos(2))
                CommandStr = CommandHeader + Command + PTPPos3DT + OutDataStr + BCC
        Case 3: OutDataStr = CalPos(3, txtPTPPos(3))
                CommandStr = CommandHeader + Command + PTPPos4DT + OutDataStr + BCC
        Case 4: OutDataStr = CalPos(0, txtPTPPos(0)) + CalPos(1, txtPTPPos(1)) + CalPos(2, txtPTPPos(2)) + CalPos(3, txtPTPPos(3))
                CommandStr = CommandHeader + Command + PTPPosADT + OutDataStr + BCC
    End Select
    Call SendData
    Call CheckResponse
    Command = "WCS"
    Select Case Index
        Case 0: CommandStr = CommandHeader + Command + PTPRun1ON + BCC
        Case 1: CommandStr = CommandHeader + Command + PTPRun2ON + BCC
        Case 2: CommandStr = CommandHeader + Command + PTPRun3ON + BCC
        Case 3: CommandStr = CommandHeader + Command + PTPRun4ON + BCC
        Case 4: Command = "WCP4"
                CommandStr = CommandHeader + Command + PTPRunAON + BCC
    End Select
    lblMessage = CommandStr
    Call SendData
    Call CheckResponse
    If InStr(ResponseStr, "$") Then
        Command = "WCP4"
        CommandStr = CommandHeader + Command + PTPRun1OFF + PTPRun2OFF + PTPRun3OFF + PTPRun4OFF + BCC
        Call SendData
        Call CheckResponse
        Select Case Index
```

```
            Case 0: XPOS = lblCurPos(0)
            Case 1: YPOS = lblCurPos(1)
            Case 2: ZPOS = lblCurPos(2)
            Case 3: APOS = lblCurPos(3)
            Case 3: XPOS = lblCurPos(0): YPOS = lblCurPos(1): ZPOS = lblCurPos(2): APOS = lblCurPos(3)
        End Select
    End If
    cmdAbsMove(Index).Enabled = True
End Sub

Public Sub cmdBusyCheck_Click()
    BusyCheckFlag = True
End Sub

Private Sub BusyCheck()
    cmdBusyCheck.Enabled = False
    CommandStr = CommandHeader + BusyCheckCmdStr + BCC
    lblMessage = CommandStr
    Call SendData
    Call CheckResponse
    If InStr(ResponseStr, "$") Then
        If Mid(ResponseStr, 7, 1) <> "0" Then
            chkBusy(0).Value = 1
        Else
            chkBusy(0).Value = 0
        End If
        If Mid(ResponseStr, 8, 1) <> "0" Then
            chkBusy(1).Value = 1
        Else
            chkBusy(1).Value = 0
        End If
        If Mid(ResponseStr, 9, 1) <> "0" Then
            chkBusy(2).Value = 1
        Else
            chkBusy(2).Value = 0
        End If
        If Mid(ResponseStr, 10, 1) <> "0" Then
            chkBusy(3).Value = 1
        Else
            chkBusy(3).Value = 0
        End If
    End If
    cmdBusyCheck.Enabled = True
End Sub

Public Sub cmdEmergency_Click()
    EmergencyFlag = True
End Sub

Private Sub Emergency()
    cmdEmergency.Enabled = False
    If EmgFlag = True Then
        CommandStr = CommandHeader + EmergencyOffCmdStr + BCC
        lblMessage = CommandStr
        Call SendData
        If CheckResponse = True Then
            cmdEmergency.Caption = "Emergency Push"
            EmgFlag = False
        End If
```

```
    Else
        CommandStr = CommandHeader + EmergencyOnCmdStr + BCC
        lblMessage = CommandStr
        Call SendData
        Call CheckResponse
        If CheckResponse = True Then
            cmdEmergency.Caption = "Emergency Release"
            EmgFlag = True
        End If
    End If
    cmdEmergency.Enabled = True
End Sub

Public Sub cmdJogCCW_MouseDown(Index As Integer, Button As Integer, Shift As Integer, X As Single, Y As Single)
    If Button = 1 Then
        JogCCW_MouseDownFlag = True
        JogCCW_MouseDownIndex = Index
    End If
End Sub

Private Sub JogCCW_MouseDown(Index As Integer)
    Command = "WCS"
    Select Case Index
        Case 0: CommandStr = CommandHeader + Command + JogCCW1ON + BCC
        Case 1: CommandStr = CommandHeader + Command + JogCCW2ON + BCC
        Case 2: CommandStr = CommandHeader + Command + JogCCW3ON + BCC
        Case 3: CommandStr = CommandHeader + Command + JogCCW4ON + BCC
    End Select
    lblMessage = CommandStr
    Call SendData
    Call CheckResponse
End Sub

Public Sub cmdJogCCW_MouseUp(Index As Integer, Button As Integer, Shift As Integer, X As Single, Y As Single)
    If Button = 1 Then
        JogCCW_MouseUpFlag = True
        JogCCW_MouseUpIndex = Index
    End If
End Sub

Private Sub JogCCW_MouseUp(Index As Integer)
    Command = "WCS"
    Select Case Index
        Case 0: CommandStr = CommandHeader + Command + JogCCW1OFF + BCC
        Case 1: CommandStr = CommandHeader + Command + JogCCW2OFF + BCC
        Case 2: CommandStr = CommandHeader + Command + JogCCW3OFF + BCC
        Case 3: CommandStr = CommandHeader + Command + JogCCW4OFF + BCC
    End Select
    lblMessage = CommandStr
    Call SendData
    Call CheckResponse
End Sub

Public Sub cmdJogCW_MouseDown(Index As Integer, Button As Integer, Shift As Integer, X As Single, Y As Single)
    If Button = 1 Then
        JogCW_MouseDownFlag = True
        JogCW_MouseDownIndex = Index
    End If
End Sub
```

```
Private Sub JogCW_MouseDown(Index As Integer)
    Command = "WCS"
    Select Case Index
        Case 0: CommandStr = CommandHeader + Command + JogCW1ON + BCC
        Case 1: CommandStr = CommandHeader + Command + JogCW2ON + BCC
        Case 2: CommandStr = CommandHeader + Command + JogCW3ON + BCC
        Case 3: CommandStr = CommandHeader + Command + JogCW4ON + BCC
    End Select
    lblMessage = CommandStr
    Call SendData
    Call CheckResponse
End Sub

Public Sub cmdJogCW_MouseUp(Index As Integer, Button As Integer, Shift As Integer, X As Single, Y As Single)
    If Button = 1 Then
        JogCW_MouseUpFlag = True
        JogCW_MouseUpIndex = Index
    End If
End Sub

Private Sub JogCW_MouseUp(Index As Integer)
    Command = "WCS"
    Select Case Index
        Case 0: CommandStr = CommandHeader + Command + JogCW1OFF + BCC
        Case 1: CommandStr = CommandHeader + Command + JogCW2OFF + BCC
        Case 2: CommandStr = CommandHeader + Command + JogCW3OFF + BCC
        Case 3: CommandStr = CommandHeader + Command + JogCW4OFF + BCC
    End Select
    lblMessage = CommandStr
    Call SendData
    Call CheckResponse
End Sub

Public Sub cmdOrgEndCheck_Click()
    OrgEndCheckFlag = True
End Sub

Private Sub OrgEndCheck()
    cmdOrgEndCheck.Enabled = False
    CommandStr = CommandHeader + OrgEndCheckCmdStr + BCC
    lblMessage = CommandStr
    Call SendData
    Call CheckResponse
    'lblMessage = ResponseStr
    If InStr(ResponseStr, "$") Then
        If Mid(ResponseStr, 7, 1) <> "0" Then
            chkOrgEnd(0).Value = 1
        Else
            chkOrgEnd(0).Value = 0
        End If
        If Mid(ResponseStr, 8, 1) <> "0" Then
            chkOrgEnd(1).Value = 1
        Else
            chkOrgEnd(1).Value = 0
        End If
        If Mid(ResponseStr, 9, 1) <> "0" Then
            chkOrgEnd(2).Value = 1
        Else
            chkOrgEnd(2).Value = 0
```

```
        End If
        If Mid(ResponseStr, 10, 1) <> "0" Then
            chkOrgEnd(3).Value = 1
        Else
            chkOrgEnd(3).Value = 0
        End If
    End If
    cmdOrgEndCheck.Enabled = True
End Sub

Public Sub cmdOrigin_click(Index As Integer)
    OriginFlag = True
    OriginIndex = Index
End Sub

Private Sub Origin(Index As Integer)
    cmdOrigin(Index).Enabled = False
    Command = "WCS"
    Select Case Index
        Case 0: CommandStr = CommandHeader + Command + HomeSet1ON + BCC
        Case 1: CommandStr = CommandHeader + Command + HomeSet2ON + BCC
        Case 2: CommandStr = CommandHeader + Command + HomeSet3ON + BCC
        Case 3: CommandStr = CommandHeader + Command + HomeSet4ON + BCC
        Case 4: Command = "WCP4"
                CommandStr = CommandHeader + Command + HomeSet1ON + HomeSet2ON + HomeSet3ON + HomeSet4ON + BCC
    End Select
    lblMessage = CommandStr
    Call SendData
    Call CheckResponse
    If InStr(ResponseStr, "$") Then
        Command = "WCP4"
        CommandStr = CommandHeader + Command + HomeSet1OFF + HomeSet2OFF + HomeSet3OFF + HomeSet4OFF + BCC
        Call SendData
        Call CheckResponse
        Select Case Index
            Case 0: XPOS = 0#: lblCurPos(0) = CStr(XPOS)
            Case 1: YPOS = 0#: lblCurPos(1) = CStr(YPOS)
            Case 2: ZPOS = 0#: lblCurPos(2) = CStr(ZPOS)
            Case 3: APOS = 0#: lblCurPos(3) = CStr(APOS)
            Case 3: XPOS = 0#: YPOS = 0#: ZPOS = 0#: APOS = 0#
                    lblCurPos(0) = CStr(XPOS)
                    lblCurPos(1) = CStr(YPOS)
                    lblCurPos(2) = CStr(ZPOS)
                    lblCurPos(3) = CStr(APOS)
        End Select
    End If
    cmdOrigin(Index).Enabled = True
End Sub

Public Sub cmdOutReset_Click()
    OutResetFlag = True
End Sub

Private Sub OutReset()
    CommandStr = CommandHeader + OutResetCmdStr + BCC
    lblMessage = CommandStr
    Call SendData
    Call CheckResponse
End Sub
```

```
Public Sub cmdPosCheck_Click()
    PosCheckFlag = True
End Sub

Private Sub PosCheck()
    Dim i As Byte
    Dim j As Byte
    Dim CalData As Long
    Dim DataStr(10) As String
    cmdPosCheck.Enabled = False
    Command = "RD"
    CommandStr = CommandHeader + Command + CurPosADT + BCC
    lblMessage = CommandStr
    Call SendData
    Call CheckResponse
    If InStr(ResponseStr, "$") Then
        j = 0
        For i = StartDataPos To StartDataPos + 28 Step 4
            DataStr(j) = ConvPLCData(Mid(ResponseStr, i, 4))
            j = j + 1
        Next i
        lblCurPos(0) = CLng("&H" + DataStr(1) + DataStr(0)) / Axis1PosRatio
        lblCurPos(1) = CLng("&H" + DataStr(3) + DataStr(2)) / Axis1PosRatio
        lblCurPos(2) = CLng("&H" + DataStr(5) + DataStr(4)) / Axis1PosRatio
        lblCurPos(3) = CLng("&H" + DataStr(7) + DataStr(6)) / Axis1PosRatio
    End If
    cmdPosCheck.Enabled = True
End Sub

Public Sub cmdServo_Click(Index As Integer)
    ServoFlag = True
    ServoIndex = Index
End Sub

Private Sub Servo(Index As Integer)
    Dim SwDataStr As String
    cmdServo(Index).Enabled = False
    Command = "WCS"
    Select Case Index
        Case 0:
                If cmdServo(Index).Caption = "On" Then
                    SwDataStr = Servo1ON
                Else
                    SwDataStr = Servo1OFF
                End If
        Case 1:
                If cmdServo(Index).Caption = "On" Then
                    SwDataStr = Servo2ON
                Else
                    SwDataStr = Servo2OFF
                End If
        Case 2:
                If cmdServo(Index).Caption = "On" Then
                    SwDataStr = Servo3ON
                Else
                    SwDataStr = Servo3OFF
                End If
        Case 3:
                If cmdServo(Index).Caption = "On" Then
```

```
                SwDataStr = Servo4ON
            Else
                SwDataStr = Servo4OFF
            End If
        Case 4:
            If cmdServo(Index).Caption = "On" Then
                SwDataStr = ServoAON
            Else
                SwDataStr = ServoAOFF
            End If
    End Select
    Select Case Index
        Case 0 To 3: CommandStr = CommandHeader + Command + SwDataStr + BCC
        Case 4: Command = "WCP4"
            CommandStr = CommandHeader + Command + SwDataStr + BCC
    End Select
    lblMessage = CommandStr
    Call SendData
    Call CheckResponse
    If InStr(ResponseStr, "$") Then
        Select Case Index
            Case 0:
                If cmdServo(Index).Caption = "On" Then
                    cmdServo(Index).Caption = "Off"
                Else
                    cmdServo(Index).Caption = "On"
                End If
            Case 1:
                If cmdServo(Index).Caption = "On" Then
                    cmdServo(Index).Caption = "Off"
                Else
                    cmdServo(Index).Caption = "On"
                End If
            Case 2:
                If cmdServo(Index).Caption = "On" Then
                    cmdServo(Index).Caption = "Off"
                Else
                    cmdServo(Index).Caption = "On"
                End If
            Case 3:
                If cmdServo(Index).Caption = "On" Then
                    cmdServo(Index).Caption = "Off"
                Else
                    cmdServo(Index).Caption = "On"
                End If
            Case 4:
                If cmdServo(Index).Caption = "On" Then
                    cmdServo(Index).Caption = "Off"
                Else
                    cmdServo(Index).Caption = "On"
                End If
                Call cmdServoCheck_Click
        End Select
    End If
    cmdServo(Index).Enabled = True
End Sub

Public Sub cmdServoCheck_Click()
    ServoCheckFlag = True
```

```
End Sub

Private Sub ServoCheck()
    cmdServoCheck.Enabled = False
    CommandStr = CommandHeader + ServoCheckCmdStr + BCC
    lblMessage = CommandStr
    Call SendData
    Call CheckResponse
    If InStr(ResponseStr, "$") Then
        If Mid(ResponseStr, 7, 1) <> "0" Then
            cmdServo(0).Caption = "Off"
            chkServo(0) = 1
        Else
            cmdServo(0).Caption = "On"
            chkServo(0) = 0
        End If
        If Mid(ResponseStr, 8, 1) <> "0" Then
            cmdServo(1).Caption = "Off"
            chkServo(1) = 1
        Else
            cmdServo(1).Caption = "On"
            chkServo(1) = 0
        End If
        If Mid(ResponseStr, 9, 1) <> "0" Then
            cmdServo(2).Caption = "Off"
            chkServo(2) = 1
        Else
            cmdServo(2).Caption = "On"
            chkServo(2) = 0
        End If
        If Mid(ResponseStr, 10, 1) <> "0" Then
            cmdServo(3).Caption = "Off"
            chkServo(3) = 1
        Else
            cmdServo(3).Caption = "On"
            chkServo(3) = 0
        End If
        If Mid(ResponseStr, 7, 4) <> "0000" Then
            cmdServo(4).Caption = "Off"
        Else
            cmdServo(4).Caption = "On"
        End If
    End If
    cmdServoCheck.Enabled = True
End Sub

Private Sub Form_Load()
    PortNum = 1
    BaudRate = 19200
    frmRS232.PortOpen
    OnProcess = False
    EmgFlag = False
    CheckOutputFlag = True
    TimeOut = False
    OutputFlag = False
    AbsMoveFlag = False
    BusyCheckFlag = False
    EmergencyFlag = False
    JogCCW_MouseDownFlag = False
```

```
    JogCCW_MouseUpFlag = False
    JogCW_MouseDownFlag = False
    JogCW_MouseUpFlag = False
    OrgEndCheckFlag = False
    OriginFlag = False
    OutResetFlag = False
    PosCheckFlag = False
    ServoFlag = False
    ServoCheckFlag = False

    'Call cmdServoCheck_Click
    InDataOld = "0"
End Sub


Public Sub cmdExit_Click()
    cmdExit.Enabled = False
    Call cmdOutReset_Click
    cmdExit.Enabled = True
    frmManualControl.Hide
End Sub

Private Sub Form_Unload(Cancel As Integer)
    frmRS232.MSComm1.PortOpen = False
End Sub

Private Function CheckResponse() As Boolean
    tmrTimeOut.Enabled = True
    TimeOut = False
    Do
        DoEvents
        If TimeOut = True Then Exit Do
    Loop Until InStr(ResponseStr, vbCr)
    tmrTimeOut.Enabled = False
    If InStr(ResponseStr, "!") Then
        lblMessage = "Communication ERROR!!"
        CommErrorFlag = True
    ElseIf InStr(ResponseStr, "$") Then
        'lblMessage = "Communication OK!!"
        CommErrorFlag = False
    End If
    If TimeOut = False And CommErrorFlag = False Then
        CheckResponse = True
    Else
        CheckResponse = False
    End If
End Function

Private Sub UpdateIOStatus()
    Dim i As Byte
    Dim j As Byte
    Dim CalData As Long
    Dim DataStr(10) As String
    If InStr(ResponseStr, "$") Then
        j = 0
        For i = StartDataPos To StartDataPos + 4 Step 4
            DataStr(j) = ConvPLCData(Mid(ResponseStr, i, 4))
            j = j + 1
        Next i
```

```
    End If
    For i = 0 To 1
        CalData = Val("&H" + DataStr(i))
        For j = 1 To 16
            If ((CalData Mod (2 ^ j)) \ (2 ^ (j - 1))) Then
                If i < 2 Then
                    chkInput((j - 1) + (i * 16)).Value = 1
                Else
                    chkOutput((j - 1) + ((i - 2) * 16)).Value = 1
                End If
            Else
                If i < 2 Then
                    chkInput((j - 1) + (i * 16)).Value = 0
                Else
                    chkOutput((j - 1) + ((i - 2) * 16)).Value = 0
                End If
            End If
        Next j
    Next i
End Sub

Private Sub ReadIO()
    Dim InData As String
    If optCommBusy = False Then
        optCommBusy = True
        Command = "RCC"
        CommandStr = CommandHeader + Command + ReadIn + BCC
        Call SendData
        Call CheckResponse
        If CommErrorFlag = True Then
            InData = Mid(ResponseStr, StartDataPos, 8)
            If InDataOld <> InData Then Call UpdateIOStatus
            InDataOld = InData
        End If
        optCommBusy = False
    End If
End Sub

Private Sub txtJogSpeed_Change()
    If Val(txtJogSpeed) > 100 Then
        txtJogSpeed = "100"
        lblMessage = "Too High Speed!! Down Please!!"
    End If
End Sub

Private Sub tmrWritePLC_Timer()
    If optCommBusy = False Then
        If OutputFlag = True Then
            optCommBusy = True
            Call Output(OutputIndex)
            OutputFlag = False
        End If
        If AbsMoveFlag = True Then
            optCommBusy = True
            Call AbsMove(AbsMoveIndex)
            AbsMoveFlag = False
        End If
        If BusyCheckFlag = True Then
            optCommBusy = True
```

```
    Call BusyCheck
    BusyCheckFlag = False
End If
If EmergencyFlag = True Then
    optCommBusy = True
    Call Emergency
    EmergencyFlag = False
End If
If JogCCW_MouseUpFlag = True Then
    optCommBusy = True
    Call JogCCW_MouseUp(JogCCW_MouseUpIndex)
    JogCCW_MouseUpFlag = False
End If
If JogCCW_MouseDownFlag = True Then
    optCommBusy = True
    Call JogCCW_MouseDown(JogCCW_MouseDownIndex)
    JogCCW_MouseDownFlag = False
End If
If JogCW_MouseUpFlag = True Then
    optCommBusy = True
    Call JogCW_MouseUp(JogCW_MouseUpIndex)
    JogCW_MouseUpFlag = False
End If
If JogCW_MouseDownFlag = True Then
    optCommBusy = True
    Call JogCW_MouseDown(JogCW_MouseDownIndex)
    JogCW_MouseDownFlag = False
End If
If OrgEndCheckFlag = True Then
    optCommBusy = True
    Call OrgEndCheck
    OrgEndCheckFlag = False
End If
If OriginFlag = True Then
    optCommBusy = True
    Call Origin(OriginIndex)
    OriginFlag = False
End If
If OutResetFlag = True Then
    optCommBusy = True
    Call OutReset
    OutResetFlag = False
End If
If PosCheckFlag = True Then
    optCommBusy = True
    Call PosCheck
    PosCheckFlag = False
End If
If ServoFlag = True Then
    optCommBusy = True
    Call Servo(ServoIndex)
    ServoFlag = False
End If
If ServoCheckFlag = True Then
    optCommBusy = True
    Call ServoCheck
    ServoCheckFlag = False
End If
optCommBusy = False
```

```
    End If
End Sub


Private Sub tmrTimeOut_Timer()
    TimeOut = True
    lblMessage = "Communication Time Out!!  Please Check Comm Module or Port
End Sub

Private Sub txtPTPPos_Change(Index As Integer)
    Select Case Index
        Case 0:
            If Val(txtPTPPos(Index)) > XLimitPos Then
                txtPTPPos(Index) = "0"
                lblMessage = "Error !!  Area Over Set!!"
            End If
        Case 1:
            If Val(txtPTPPos(Index)) > YLimitPos Then
                txtPTPPos(Index) = "0"
                lblMessage = "Error !!  Area Over Set!!"
            End If
        Case 2:
            If Val(txtPTPPos(Index)) > ZLimitPos Then
                txtPTPPos(Index) = "0"
                lblMessage = "Error !!  Area Over Set!!"
            End If
        Case 3:
            If Val(txtPTPPos(Index)) > ALimitPos Then
                txtPTPPos(Index) = "0"
                lblMessage = "Error !!  Area Over Set!!"
            End If
    End Select
End Sub

Private Sub txtPTPSpeed_Change(Index As Integer)
    If Val(txtPTPSpeed(Index)) > 100 Then
        txtPTPSpeed(Index) = "100"
        lblMessage = "Too High Speed!! Down Please!!"
    End If
End Sub

Option Explicit

Private Sub cmdClearInDataBuffer_Click()
    lblReceivedData.Caption = ""
End Sub

Private Sub cmdConfigCom_Click()
    MSComm1.PortOpen = False
    frmConfigCom.Show 1
    Call PortOpen
End Sub

Public Sub cmdSendData_Click()
    If MSComm1.PortOpen = False Then
        Call PortOpen
    End If
    'lblReceivedData = ""
    'MSComm1.Output = txtSendData.Text + vbCr
```

```
    ResponseStr = ""
    MSComm1.Output = CommandStr + vbCr
    'txtSendData.Text = ""
'    txtSendData.SetFocus
End Sub

Private Sub Form_Load()
    PortNum = 1
    BaudRate = 19200
   ' Call PortOpen
End Sub

Public Sub PortOpen()
   MSComm1.CommPort = PortNum
   MSComm1.Settings = CStr(BaudRate) + ",n,8,1"
   lblComStatus.Caption = "포트번호: COM" + Str(PortNum) + ",  BaudRate: " + Str(BaudRate) + "BPS
   ' 입력된 문자를 버퍼 전체에서 읽을 수 있도록 알려줍니다.
   MSComm1.InputLen = 0
   ' 통신 포트를 엽니다.
   MSComm1.PortOpen = True
End Sub

Private Sub Form_Unload(Cancel As Integer)
    MSComm1.PortOpen = False
End Sub

Private Sub MSComm1_OnComm()
    Dim OneChar As String

    Select Case MSComm1.CommEvent

    Case comEvReceive
        OneChar = MSComm1.Input
        'lblReceivedData.Caption = lblReceivedData.Caption + OneChar
        ResponseStr = ResponseStr + OneChar
  End Select
End Sub

Private Sub txtSendData_KeyPress(KeyAscii As Integer)
    If KeyAscii = vbKeyReturn Then
        MSComm1.Output = txtSendData.Text + vbCr
        lblReceivedData = ""
        txtSendData.Text = ""
    End If
End Sub

Private Sub cmdCancel_Click()
    frmConfigCom.Hide
End Sub

Private Sub cmdSet_Click()
    PortNum = cboPortNum.ListIndex + 1
    Select Case cboBaudRate.ListIndex
    Case 0
        BaudRate = 19200
    Case 1
        BaudRate = 9600
    Case 2
        BaudRate = 4800
```

```
    Case 3
        BaudRate = 2400
    End Select
    frmConfigCom.Hide
End Sub

Private Sub Form_Activate()
    cboPortNum.ListIndex = PortNum - 1
    Select Case BaudRate
    Case 19200
        cboBaudRate.ListIndex = 0
    Case 9600
        cboBaudRate.ListIndex = 1
    Case 4800
        cboBaudRate.ListIndex = 2
    Case 2400
        cboBaudRate.ListIndex = 3
    End Select
End Sub

Global Const DisplayMainTop = 30
Global Const SystemInit = 1
Global Const ManualCon = 2
Global Const Harvest = 3
Global Const Spray = 4
Global Const IdleMode = 5
Global Const InitMsg = "시스템을 초기화하는 중 입니다."
Global Const JobMsg = "작업설정 박스에서 원하는 작업을 선택하십시요."
Global Const ManJobMsg = "수동작업 모드입니다. 원하는 버튼을 클릭하세요."
Global Const TeachMsg = "영상화면에서 작업 대상물을 선택하거나 이동버튼을 이용하여 영역변경을 하세요
Global Const JobEndMsg = "작업이 끝났습니다. 다음작업을 위하여 대차이동 또는 작업모드를 선택하세요."


Global Const CommandHeader = "%01#"
Global Const ResponseOKHeader = "%01$"
Global Const ResponseErrorHeader = "%01!"
Global Const BCC = "**"

Global Const Cur1PosDT = "D0009000091"
Global Const Cur2PosDT = "D0009200093"
Global Const Cur3PosDT = "D0009400095"
Global Const Cur4PosDT = "D0009600097"
Global Const CurPosADT = "D0009000097"

Global Const Speed1DT = "D0030000301"
Global Const Speed2DT = "D0030200303"
Global Const Speed3DT = "D0030400305"
Global Const Speed4DT = "D0030600307"
Global Const SpeedADT = "D0030000307"

Global Const AccTime1DT = "D0035000351"
Global Const AccTime2DT = "D0035200353"
Global Const AccTime3DT = "D0035400355"
Global Const AccTime4DT = "D0035600357"
Global Const AccTimeADT = "D0035000357"

Global Const PTPPos1DT = "D0036000361"
Global Const PTPPos2DT = "D0036200363"
Global Const PTPPos3DT = "D0036400365"
```

```
Global Const PTPPos4DT = "D0036600367"
Global Const PTPPosADT = "D0036000367"

Global Const ReadIn = "R00500051"

Global Const Busy1 = "R0300"
Global Const Busy2 = "R0320"
Global Const Busy3 = "R0340"
Global Const Busy4 = "R0360"

Global Const JogCCW1ON = "R00001"
Global Const JogCW1ON = "R00011"
Global Const JogCCW1OFF = "R00000"
Global Const JogCW1OFF = "R00010"

Global Const JogCCW2ON = "R00501"
Global Const JogCW2ON = "R00511"
Global Const JogCCW2OFF = "R00500"
Global Const JogCW2OFF = "R00510"

Global Const JogCCW3ON = "R01001"
Global Const JogCW3ON = "R01011"
Global Const JogCCW3OFF = "R01000"
Global Const JogCW3OFF = "R01010"


Global Const JogCCW4ON = "R01501"
Global Const JogCW4ON = "R01511"
Global Const JogCCW4OFF = "R01500"
Global Const JogCW4OFF = "R01510"

Global Const HomeSet1ON = "R00021"
Global Const HomeSet2ON = "R00521"
Global Const HomeSet3ON = "R01021"
Global Const HomeSet4ON = "R01521"
Global Const HomeSet1OFF = "R00020"
Global Const HomeSet2OFF = "R00520"
Global Const HomeSet3OFF = "R01020"
Global Const HomeSet4OFF = "R01520"

Global Const Servo1ON = "R00061"
Global Const Servo2ON = "R00561"
Global Const Servo3ON = "R01061"
Global Const Servo4ON = "R01561"
Global Const ServoAON = "R00061R00561R01061R01561"
Global Const Servo1OFF = "R00060"
Global Const Servo2OFF = "R00560"
Global Const Servo3OFF = "R01060"
Global Const Servo4OFF = "R01560"
Global Const ServoAOFF = "R00060R00560R01060R01560"

Global Const PTPRun1ON = "R00041"
Global Const PTPRun2ON = "R00541"
Global Const PTPRun3ON = "R01041"
Global Const PTPRun4ON = "R01541"
Global Const PTPRunAON = "R00041R00541R01041R01541"
Global Const PTPRun1OFF = "R00040"
Global Const PTPRun2OFF = "R00540"
Global Const PTPRun3OFF = "R01040"
```

```
Global Const PTPRun4OFF = "R01540"
Global Const PTPRunAOFF = "R00040R00540R01040R01540"

Global Const StartDataPos = 7

Global Const Axis1SpeedRatio = 800
Global Const Axis2SpeedRatio = 1200
Global Const Axis3SpeedRatio = 400
Global Const Axis4SpeedRatio = 5

Global Const Axis1PosRatio = 89.5 ' 2048/23
Global Const Axis2PosRatio = 410 '2048/5
Global Const Axis3PosRatio = 512
Global Const Axis4PosRatio = 20 ' 7200Pulse per 1rev

Global Const ServoCheckCmdStr = "RCP4Y0100Y0101Y0102Y0103"
Global Const BusyCheckCmdStr = "RCP4R0300R0320R0340R0360"
Global Const OrgEndCheckCmdStr = "RCP4X0008X0018X0028X0038"
Global Const EmergencyOffCmdStr = "WCP4Y00450Y00550Y00650Y00750"
Global Const EmergencyOnCmdStr = "WCP4Y00451Y00551Y00651Y00751"
Global Const OutResetCmdStr = "WCCY0010001100000000"

Global Const XLimitPos = 2150
Global Const YLimitPos = 700
Global Const ZLimitPos = 450
Global Const ALimitPos = 360

' IO define
Global Const VacummValve = 8
Global Const CutCylValve = 9
Global Const WheelMotorF = 16
Global Const WheelMotorB = 17
Global Const LoadCyl = 18
Global Const UnloadCyl = 20


Public Function BCDtoDEC(BCDData As String) As Long
    Dim i As Long
    Dim CalData As Long
    Dim ResultData As Long
    ResultData = 0
    For i = 0 To 3
        CalData = Val("&H" + Mid(BCDData, i + 1, 1))
        If (CalData \ 8) = 1 Then ResultData = ResultData + (2 ^ (15 - (i * 4)))
        If ((CalData Mod 8) \ 4) = 1 Then ResultData = ResultData + (2 ^ (14 - (i * 4)))
        If ((CalData Mod 4) \ 2) = 1 Then ResultData = ResultData + (2 ^ (13 - (i * 4)))
        If (CalData Mod 2) = 1 Then ResultData = ResultData + (2 ^ (12 - (i * 4)))
    Next i
    BCDtoDEC = ResultData
End Function

Public Function ConvPLCData(ConvData As String) As String
    Dim HighData As String
    Dim LowData As String
    LowData = Mid(ConvData, 1, 2)
    HighData = Mid(ConvData, 3, 2)
    ConvPLCData = HighData + LowData
End Function
```

```
Public Function CalSpeed(Index As Integer, SpeedDataStr As String) As String
    Dim CalData As Long
    Dim CalDataStr(3) As String
    Select Case Index
        Case 0: CalData = Val(SpeedDataStr) * Axis1SpeedRatio + 10000
        Case 1: CalData = Val(SpeedDataStr) * Axis2SpeedRatio + 10000
        Case 2: CalData = Val(SpeedDataStr) * Axis3SpeedRatio + 10000
        Case 3: CalData = Val(SpeedDataStr) * Axis4SpeedRatio + 500
    End Select

    CalDataStr(0) = Conv2WDPLCData(Hex(CalData))
    CalDataStr(1) = ConvPLCData(Mid(CalDataStr(0), 1, 4))
    CalDataStr(2) = ConvPLCData(Mid(CalDataStr(0), 5, 4))
    CalSpeed = CalDataStr(2) + CalDataStr(1)
End Function

Public Function CalAccTime(Index As Integer, AccTimeStr As String) As String
    Dim CalData As Long
    Dim CalDataStr(3) As String
    Select Case Index
        Case 0: CalData = Val(AccTimeStr)
        Case 1: CalData = Val(AccTimeStr)
        Case 2: CalData = Val(AccTimeStr)
        Case 3: CalData = Val(AccTimeStr)
    End Select

    CalDataStr(0) = Conv2WDPLCData(Hex(CalData))
    CalDataStr(1) = ConvPLCData(Mid(CalDataStr(0), 1, 4))
    CalDataStr(2) = ConvPLCData(Mid(CalDataStr(0), 5, 4))
    CalAccTime = CalDataStr(2) + CalDataStr(1)
End Function

Public Function CalPos(Index As Integer, PosDataStr As String) As String
    Dim CalData As Double
    Dim CalDataStr(3) As String
    Select Case Index
        Case 0: CalData = Val(PosDataStr) * Axis1PosRatio
        Case 1: CalData = Val(PosDataStr) * Axis2PosRatio
        Case 2: CalData = Val(PosDataStr) * Axis3PosRatio
        Case 3: CalData = Val(PosDataStr) * Axis4PosRatio
    End Select

    CalDataStr(0) = Conv2WDPLCData(Hex(CalData))
    CalDataStr(1) = ConvPLCData(Mid(CalDataStr(0), 1, 4))
    CalDataStr(2) = ConvPLCData(Mid(CalDataStr(0), 5, 4))
    CalPos = CalDataStr(2) + CalDataStr(1)
End Function

Public Function Conv2WDPLCData(DataStr As String) As String
    Dim i As Integer
    Dim DataLen As Integer
    DataLen = Len(DataStr)
    For i = 1 To 8 - DataLen
        DataStr = "0" + DataStr
    Next i
    Conv2WDPLCData = DataStr
End Function

Public Sub SendData()
```

```
    ResponseStr = ""
    If frmRS232.MSComm1.PortOpen = True Then frmRS232.MSComm1.Output = CommandStr + vbC
End Sub


Option Explicit
Dim TPPX As Long
Dim TPPY As Long
Dim buffer As String
Dim ServoSw As Boolean
Dim MouseDown As Boolean

Dim JobIndex As Byte
Dim MagaTurn As Boolean
Dim Zturn As Boolean
Dim PointData(19, 1, 3) As Long
Dim PointNo As Integer
Dim WinNo As Integer
Dim MulX As Double
Dim MulY As Double
Dim Speed As Byte
Dim Step As Double

Private Sub cmdEmergency_Click()
    If frmManualControl.cmdEmergency.Caption = "Emergency Push" Then
        shpEmergency.BackColor = vbYellow
        cmdEmergency.Caption = "비상복귀"
    Else
        shpEmergency.BackColor = vbRed
        cmdEmergency.Caption = "비상정지"
    End If
    EmergencyFlag = True
End Sub

Private Sub cmdExit_Click()
    frmManualControl.cmdExit_Click
    End
End Sub

Private Sub cmdManualControl_Click()
    frmManualControl.Show (1)
End Sub

Private Sub cmdOriginRun_Click()
    frmManualControl.cmdOrigin_click (4)
End Sub

Private Sub cmdZUp_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
    If Button = 1 Then
        JogCW_MouseDownFlag = True
        JogCW_MouseDownIndex = 2
    End If
End Sub

Private Sub cmdZUp_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    If Button = 1 Then
        JogCW_MouseUpFlag = True
        JogCW_MouseUpIndex = 2
    End If
```

```
End Sub

Private Sub cmdZDown_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single
    If Button = 1 Then
        JogCCW_MouseDownFlag = True
        JogCCW_MouseDownIndex = 2
    End If
End Sub

Private Sub cmdZDown_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    If Button = 1 Then
        JogCCW_MouseUpFlag = True
        JogCCW_MouseUpIndex = 2
    End If
End Sub

Private Sub Form_Load()
    Show
    Call FormAdjust
'    Digitizer.GrabContinuous
    'ServoSw = False
    'txtMessage.Text = JobMsg
    'PointNo = 0
    'WinNo = 0
    'MulX = 0.7    '////////////////////////////////////////
    'MulY = 0.7    '////////////////////////////////////////
    'Step = 0#
    'Speed = 0
    'JobIndex = IdleMode
End Sub

Private Sub Form_Unload(Cancel As Integer)
'    Digitizer.Halt
End Sub

Private Sub FormAdjust()
    Dim i As Byte
    TPPX = Screen.TwipsPerPixelX
    TPPY = Screen.TwipsPerPixelY
    With MainScreen
        .Top = 0
        .Left = 0
        .Height = Screen.Height
        .Width = Screen.Width
    End With
    DisplayMain.Top = DisplayMainTop
    DisplayMain.Left = (Screen.Width / TPPX) \ 2 - 320
    With fraImageDisplay
        .Top = DisplayMain.Top - 10
        .Left = DisplayMain.Left - 5
        .Width = DisplayMain.Width + 10
        .Height = DisplayMain.Height + 15
    End With
    With shpWorkArea
        .Top = 107 * TPPX
        .Left = 95 * TPPX
        .Width = 450 * TPPX
        .Height = 360 * TPPY
    End With
```

```
    For i = 0 To 19
        With R(i)
            .Width = 50 * TPPX
            .Height = 50 * TPPY
            .Visible = False
        End With
    Next i
End Sub
Private Sub UnvisibleTeachRect()
    Dim i As Byte
    For i = 0 To 19
        R(i).Visible = False
    Next i
End Sub

Private Sub cmdManualCon_Click()
    JobIndex = ManualCon
    Call VisibleManualCon
    txtMessage.Text = ManJobMsg
End Sub

Private Sub VisibleManualCon()
    fraZ.Visible = True
    fraUnload.Visible = True
    fraLoad.Visible = True
    fraVac.Visible = True
    fraCar.Visible = True
    fraEndAir.Visible = True
    fraRobot.Visible = True
End Sub

Private Sub UnVisibleManualCon()
    fraZ.Visible = False
    fraUnload.Visible = False
    fraLoad.Visible = False
    fraVac.Visible = False
    fraCar.Visible = False
    fraEndAir.Visible = False
    fraRobot.Visible = False
End Sub

Private Sub EnableManualCon()
    fraZ.Enabled = True
    fraUnload.Enabled = True
    fraLoad.Enabled = True
    fraVac.Enabled = True
    fraCar.Enabled = True
    fraEndAir.Enabled = True
    fraRobot.Enabled = True
End Sub

Private Sub DisableManualCon()
    fraZ.Enabled = False
    fraUnload.Enabled = False
    fraLoad.Enabled = False
    fraVac.Enabled = False
    fraCar.Enabled = False
    fraEndAir.Enabled = False
    fraRobot.Enabled = False
```

```
End Sub

Private Sub cmdHarvest_Click()
    Dim i As Integer
    Dim Xdata As Double
    Dim Ydata As Double
    JobIndex = Harvest
    Call UnVisibleManualCon
    Call VisibleHarvest
    WinNo = 0
    PointNo = 0
    Call UnvisibleTeachRect
    Call ClrBuffer
    txtMessage.Text = InitMsg
    Sleep (1000)
    ServoFlag = True: ServoIndex = 4
    Do
       frmManualControl.cmdBusyCheck_Click
       DoEvents
    Loop
    '////////////////////////////////////////완료동작 확인 추가

    'Call cmdZO_Click
    txtMessage.Text = TeachMsg
    Digitizer.Grab
    'cmdMoveLoc.Enabled = True
    For i = 1 To 4
        Do
            DoEvents
            If WinNo = i Then
                Digitizer.GrabContinuous
                PointNo = 0
                If i = 3 Then
                  '    cmdMoveLoc.Caption = "실  행"
                End If
                Call UnvisibleTeachRect
                Exit Do
            End If
           ' If ExitLoop = True Then
           '      ExitLoop = False
           '      Exit Do
           ' End If
        Loop
    Next i
    'cmdMoveLoc.Caption = "영역이동"
   ' cmdMoveLoc.Enabled = False
    For i = 0 To 19
        If PointData(i, 0, 3) <> 0 And PointData(i, 1, 3) <> 0 Then
            Xdata = CDbl(PointData(i, 0, 3))
            Ydata = CDbl(PointData(i, 1, 3))
            txtPN.Text = i
            txtWN.Text = 3
            txtX.Text = Xdata
            txtY.Text = Ydata

          '  txtSendData.Text = "@MOVD " & Format(Str(Xdata), "0.00") & "," & Format(Str(Ydata), "0.00") & ",30
          '  Call cmdSendData_Click
            Call cmdVacOn_Click
            Sleep (1000)
```

```
        '   txtSendData.Text = "@MOVD 200.00,200.00,5"
        '   Call cmdSendData_Click
        '  Call cmdLoadF_Click
           Sleep (1000)
        '  txtSendData.Text = "@MOVD 0.00,0.00,5"
      '    Call cmdSendData_Click
      '    Call cmdVacOff_Click
           Sleep (500)
      '    Call cmdLoadB_Click
    End If
Next i
 Call cmdMagaL_Click
For i = 0 To 19
    If PointData(i, 0, 2) <> 0 And PointData(i, 1, 2) <> 0 Then
        Xdata = CDbl(PointData(i, 0, 2))
        Ydata = CDbl(PointData(i, 1, 2))
        txtPN.Text = i
        txtWN.Text = 2
        txtX.Text = Xdata
        txtY.Text = Ydata

       ' txtSendData.Text = "@MOVD " & Format(Str(Xdata), "0.00") & "," & Format(Str(Ydata), "0.00") & ",30'
       ' Call cmdSendData_Click
        Call cmdVacOn_Click
        Sleep (1000)
       ' txtSendData.Text = "@MOVD 0.00,0.00,5"
       ' Call cmdSendData_Click
       ' Call cmdLoadF_Click
       ' Call cmdMagaR_Click
        Call cmdVacOff_Click
        Sleep (500)
       ' Call cmdLoadB_Click
       ' Call cmdMagaL_Click
    End If
Next i
 Call cmdMagaL_Click
For i = 0 To 19
    If PointData(i, 0, 1) <> 0 And PointData(i, 1, 1) <> 0 Then
        Xdata = CDbl(PointData(i, 0, 1))
        Ydata = CDbl(PointData(i, 1, 1))
        txtPN.Text = i
        txtWN.Text = 1
        txtX.Text = Xdata
        txtY.Text = Ydata

       ' txtSendData.Text = "@MOVD " & Format(Str(Xdata), "0.00") & "," & Format(Str(Ydata), "0.00") & ",30'
       ' Call cmdSendData_Click
        Call cmdVacOn_Click
      '   txtSendData.Text = "@MOVD 0.00,0.00,5"
      '   Call cmdSendData_Click
      '   Call cmdLoadF_Click
      '   Call cmdMagaR_Click
       '  Call cmdMagaR_Click
        Call cmdVacOff_Click
        Sleep (500)
       ' Call cmdLoadB_Click
       ' Call cmdMagaL_Click
       ' Call cmdMagaL_Click
    End If
```

```
    Next i
    'Call cmdMagaL_Click
    For i = 0 To 19
        If PointData(i, 0, 0) <> 0 And PointData(i, 1, 0) Then
            Xdata = CDbl(PointData(i, 0, 0))
            Ydata = CDbl(PointData(i, 1, 0))
            txtPN.Text = i
            txtWN.Text = 0
            txtX.Text = Xdata
            txtY.Text = Ydata

           ' txtSendData.Text = "@MOVD " & Format(Str(Xdata), "0.00") & "," & Format(Str(Ydata), "0.00") & ",30
           ' Call cmdSendData_Click
            Call cmdVacOn_Click
           ' txtSendData.Text = "@MOVD 0.00,0.00,5"
           ' Call cmdSendData_Click
           ' Call cmdLoadF_Click
           ' Call cmdMagaR_Click
           ' Call cmdMagaR_Click
           ' Call cmdMagaR_Click
           ' Call cmdVacOff_Click
            Sleep (500)
           ' Call cmdLoadB_Click
           ' Call cmdMagaL_Click
           ' Call cmdMagaL_Click
           ' Call cmdMagaL_Click
        End If
    Next i
    txtMessage.Text = JobEndMsg
    JobIndex = IdleMode
End Sub
Private Sub VisibleHarvest()
    'fraMagaMove4Job.Visible = True
End Sub



Private Sub cmdSpray_Click()
    Dim i As Integer
    Dim j As Integer
    Dim Xdata As Double
    Dim Ydata As Double
    JobIndex = Spray
    Call UnVisibleManualCon
    Call VisibleHarvest
    WinNo = 0
    PointNo = 0
    Call UnvisibleTeachRect
    Call ClrBuffer
    txtMessage.Text = InitMsg
    Sleep (1000)
    Call cmdMagaO_Click
    Call cmdServo_Click
    Call cmdO_Click
    '////////////////////////////////////////완료동작 확인 추가

    'Call cmdZO_Click
    txtMessage.Text = TeachMsg
    Digitizer.Grab
```

```
    cmdMoveLoc.Enabled = True
    For i = 1 To 4
        Do
            DoEvents
            If WinNo = i Then
                Digitizer.GrabContinuous
                PointNo = 0
                If i = 3 Then
                    cmdMoveLoc.Caption = "실  행"
                End If
                Call UnvisibleTeachRect
                Exit Do
            End If
            If ExitLoop = True Then
                ExitLoop = False
                Exit Do
            End If
        Loop
    Next i
    cmdMoveLoc.Caption = "영역이동"
    cmdMoveLoc.Enabled = False
    For j = 3 To 0 Step -1
        For i = 0 To 19
            If PointData(i, 0, j) <> 0 And PointData(i, 1, j) <> 0 Then
                Xdata = CDbl(PointData(i, 0, j))
                Ydata = CDbl(PointData(i, 1, j))
                txtPN.Text = i
                txtWN.Text = j
                txtX.Text = Xdata
                txtY.Text = Ydata

                txtSendData.Text = "@MOVD " & Format(Str(Xdata), "0.00") & "," & Format(Str(Ydata), "0.00") & ",30
                Call cmdSendData_Click
                Call cmdVacOn_Click
                Sleep (1000)
                Call cmdVacOff_Click
            End If
        Next i
        Call cmdMagaL_Click
    Next j
    txtSendData.Text = "@MOVD 0.00,0.00,5"
    Call cmdSendData_Click
    txtMessage.Text = JobEndMsg
    JobIndex = IdleMode
End Sub

Private Sub ClrBuffer()
    Dim i, j As Byte
    For j = 0 To 3
        For i = 0 To 19
                PointData(i, 0, j) = 0
                PointData(i, 1, j) = 0
        Next i
    Next j
End Sub

Private Sub cmdMoveLoc_Click()
    If WinNo < 3 Then
        Call cmdMagaR_Click
```

```
    End If
    WinNo = WinNo + 1
End Sub


'-------------------------------------------------------------
'로봇 제어 명령
'-------------------------------------------------------------
'-------------------------------------------------------------
' I/O Control
'-------------------------------------------------------------

Private Sub cmdLoadCylB_Click()
    frmManualControl.chkOutput(LoadCyl).Value = 0
End Sub

Private Sub cmdLoadCylF_Click()
    frmManualControl.chkOutput(LoadCyl).Value = 1
End Sub

Private Sub cmdUnloadCylOn_Click()
    frmManualControl.chkOutput(UnloadCyl).Value = 1
End Sub

Private Sub cmdUnloadCylOFF_Click()
    frmManualControl.chkOutput(UnloadCyl).Value = 0
End Sub

Private Sub cmdEndCylOn_Click()
    frmManualControl.chkOutput(UnloadCyl).Value = 1
End Sub

Private Sub cmdEndCylOff_Click()
    frmManualControl.chkOutput(UnloadCyl).Value = 0
End Sub

Private Sub cmdVacOn_Click()
    frmManualControl.chkOutput(UnloadCyl).Value = 1
End Sub

Private Sub cmdVacOff_Click()
    frmManualControl.chkOutput(UnloadCyl).Value = 0
End Sub

Private Sub cmdCarRunF_Click()
    frmManualControl.chkOutput(WheelMotorB).Value = 0
    frmManualControl.chkOutput(WheelMotorF).Value = 1
End Sub

Private Sub cmdCarStop_Click()
    frmManualControl.chkOutput(WheelMotorB).Value = 0
    frmManualControl.chkOutput(WheelMotorF).Value = 0
End Sub

Private Sub cmdCarRunB_Click()
    frmManualControl.chkOutput(WheelMotorF).Value = 0
    frmManualControl.chkOutput(WheelMotorB).Value = 1
End Sub
```

```
Private Sub cmdServo_Click()
    frmManualControl.cmdServo_Click (4)
End Sub

Private Sub hscSpeed_Change()
    Speed = hscSpeed.Value
    lblSpeed.Caption = "Speed :" & Str(Speed) + "%"
End Sub

Private Sub hscStep_Change()
    Step = hscStep.Value / 2#
    lblStep.Caption = "Step : " & Format(Step, "0.00") + "mm"
End Sub

Private Sub tmrGetMessage_Timer()
   txtMessage = frmManualControl.lblMessage
   optCommBusy = frmManualControl.optCommBusy
End Sub

Private Sub DisplayMain_MouseDown(Button As Integer, Shift As Integer, X As Long, Y As Long
    Dim tmpx As Double
    Dim tmpy As Double
    If JobIndex = Harvest Or JobIndex = Spray Then
        If Button = 1 And PointNo < 20 Then
            tmpx = (545 - X) * MulX
            tmpy = ((465 - Y) * MulY) + 100
            If tmpx >= 0 And tmpx < 350 And tmpy >= 0 And tmpy < 350 Then
                PointData(PointNo, 0, WinNo) = tmpx
                PointData(PointNo, 1, WinNo) = tmpy
                txtPN.Text = PointNo
                txtWN.Text = WinNo
                txtX.Text = tmpx
                txtY.Text = tmpy
                R(PointNo).Top = (Y - 25) * TPPY
                R(PointNo).Left = (X - 25) * TPPX
                R(PointNo).Visible = True
                PointNo = PointNo + 1
            Else
                txtMessage.Text = "허용범위초과"
            End If
        ElseIf Button = 2 And PointNo > 0 Then
            PointData(PointNo - 1, 0, WinNo) = 0
            PointData(PointNo - 1, 1, WinNo) = 0
            R(PointNo - 1).Visible = False
            PointNo = PointNo - 1
        End If
    End If
End Sub

Private Sub DisplayMain_MouseMove(Button As Integer, Shift As Integer, X As Long, Y As Long)
    txtX.Text = Str(X)
    txtY.Text = Str(Y)
End Sub
```

나. 스테레오 비전

```
// Math.cpp: implementation of the Math class.
//
//////////////////////////////////////////////////////////////////////

#include "stdafx.h"
#include "watermelon.h"
#include "CalMath.h"
#include <math.h>

#ifdef _DEBUG
#undef THIS_FILE
static char THIS_FILE[]=__FILE__;
#define new DEBUG_NEW
#endif

//////////////////////////////////////////////////////////////////////
// Construction/Destruction
//////////////////////////////////////////////////////////////////////

CalMath::CalMath()
{

}

CalMath::~CalMath()
{

}

void CalMath::MatrixTrans33(double matrix[3][3],double Omatrix[3][3])
{
    int i,j;
    for(i=0;i<3;i++)
        for(j=0;j<3;j++)
        {
            Omatrix[j][i]=matrix[i][j];
        }
}

void CalMath::MatrixTrans43(double matrix[4][3],double Omatrix[3][4])
{
    int i,j;
    for(i=0;i<4;i++)
        for(j=0;j<3;j++)
        {
            Omatrix[j][i]=matrix[i][j];
        }
}

void CalMath::MatrixTrans44(double matrix[4][4],double Omatrix[4][4])
{
    int i,j;
    for(i=0;i<4;i++)
        for(j=0;j<4;j++)
        {
            Omatrix[j][i]=matrix[i][j];
        }
```

```
}

void CalMath::MatrixMul3441(double matrix1[3][4],double matrix2[4][1],double Omatrix[3][1])
{
    int i;
    for(i=0;i<3;i++)
        {
            Omatrix[i][0]=  matrix1[i][0]*matrix2[0][0]
                            +matrix1[i][1]*matrix2[1][0]
                            +matrix1[i][2]*matrix2[2][0]
                            +matrix1[i][3]*matrix2[3][0];
        }
}

void CalMath::MatrixMul3334(double matrix1[3][3],double matrix2[3][4],double Omatrix[3][4])
{
    int i,j;
    for(i=0;i<3;i++)
        for(j=0;j<4;j++)
        {
            Omatrix[i][j]=  matrix1[i][0]*matrix2[0][j]
                            +matrix1[i][1]*matrix2[1][j]
                            +matrix1[i][2]*matrix2[2][j];
        }
}

void CalMath::MatrixMul3443(double matrix1[3][4],double matrix2[4][3],double Omatrix[3][3])
{
    int i,j;
    for(i=0;i<3;i++)
        for(j=0;j<3;j++)
        {
            Omatrix[i][j]=  matrix1[i][0]*matrix2[0][j]
                            +matrix1[i][1]*matrix2[1][j]
                            +matrix1[i][2]*matrix2[2][j]
                            +matrix1[i][3]*matrix2[3][j];
        }
}
void CalMath::MatrixMul4444(double matrix1[4][4],double matrix2[4][4],double Omatrix[4][4])
{
    int i,j;
    for(i=0;i<4;i++)
        for(j=0;j<4;j++)
        {
            Omatrix[i][j]=  matrix1[i][0]*matrix2[0][j]
                            +matrix1[i][1]*matrix2[1][j]
                            +matrix1[i][2]*matrix2[2][j]
                            +matrix1[i][3]*matrix2[3][j];
        }

}

void CalMath::MatrixInverse33(double matrix[3][3],double Omatrix[3][3])
{
    int         i,j,k;
    int         numrow=3,irow;
    double      dtemp,pivot;
    double      a[3][6];
    double      a_1[3][3];
```

```
    /*..Initialize..*/
    for(i=0;i<3;i++){
        for(j=0;j<3;j++)    a_1[i][j]=0.;
        for(j=0;j<6;j++) a[i][j]=0.;
    }

    /*..Input Matrix Component..*/
    for(i=0;i<numrow;i++){
        for(j=0;j<numrow;j++){
            a[i][j]=matrix[i][j];
            if(i==j) a[i][j+numrow]=1.;
            else     a[i][j+numrow]=0.;
        }
    }

    /*..pivot..*/
    for(i=0;i<numrow-1;i++){
        irow=i;
        for(k=i+1;k<numrow;k++){
            if(fabs(a[irow][i])<fabs(a[k][i])) irow=k;
        }
        for(j=0;j<2*numrow;j++){
            dtemp=a[i][j];
            a[i][j]=a[irow][j];
            a[irow][j]=dtemp;
        }
    }

    /*..Gauss-Jordan..*/
    for(i=0;i<numrow;i++){
        pivot=a[i][i];
        for(j=0;j<2*numrow;j++) a[i][j]/=pivot;
        for(k=0;k<numrow;k++){
            if(k==i) continue;
            dtemp=a[k][i];
            for(j=i;j<2*numrow;j++){
                a[k][j]-=dtemp*a[i][j];
            }
        }
    }

    /*..Inverse Matrix..*/
    for(i=0;i<numrow;i++)
    for(j=0;j<numrow;j++) a_1[i][j]=a[i][j+numrow];

    /*..Output Result..*/
    printf("\nInverse Matrix is...\n");
    for(i=0;i<numrow;i++){
        for(j=0;j<numrow;j++) Omatrix[i][j]=a_1[i][j];
    }
}

void CalMath::MatrixPseudo(double matrix1[4][3], double matrix2[4][1], double Omatrix[3][1]
{
    double OM1[3][4],OM2[3][3],OM3[3][3],OM4[3][4];
    MatrixTrans43(matrix1, OM1);
    MatrixMul3443(OM1,matrix1,OM2);
    MatrixInverse33(OM2,OM3);
    MatrixMul3334(OM3,OM1,OM4);
```

```
    MatrixMul3441(OM4,matrix2,Omatrix);
}

// Image.cpp : implementation file
//

#include "stdafx.h"
#include "Image.h"
#include "ImagePixel.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/*************************************************
                생성자, 초기화 함수
*************************************************/

CImage::CImage()
{
    m_Size          = CSize(1,1);
    m_hImage = NULL;
    m_pPal          = NULL;
    m_hUndoImage = NULL;
}

CImage::CImage( CImage &Src )
{
    m_Size          = CSize(1,1);
    m_hImage = NULL;
    m_pPal          = NULL;
    m_hUndoImage = NULL;

    *this = Src;
}

CImage& CImage::operator=( CImage &Src )  // Right side is the argumen
{
    Free();
    m_hImage = (HDIB) ::CopyHandle( Src.GetHandle() );
    m_Size = Src.GetSize();
    m_pPal = new CPalette;
    CreateDIBPalette();
    return (*this);
}

BOOL CImage::InitDIB(BOOL bCreatePalette)
{
    // 이미지의 가로, 세로 크기 설정
    LPSTR pDIB = (LPSTR)::GlobalLock((HGLOBAL) m_hImage);
    m_Size = CSize((int) ::DIBWidth(pDIB), (int) ::DIBHeight(pDIB));
    ::GlobalUnlock((HGLOBAL) m_hImage);

    if(bCreatePalette)
    {
        if(m_pPal != NULL) delete m_pPal;
        // 팔레트 생성
```

```
        m_pPal = new CPalette;
        if (CreateDIBPalette() == NULL)
        {
            // 팔레트를 가지지 않는 경우
            delete m_pPal;
            m_pPal = NULL;
            return FALSE;
        }
    }
    return TRUE;
}

void CImage::SetHandle(HANDLE hHandle)
{
    m_hImage = (HDIB)hHandle;
}

BOOL CImage::Create(int width, int height, int depth)
{
    LPBITMAPINFOHEADER lpbi ;
    BYTE        *lpPal;
    DWORD       dwSizeImage;
    int         i;

    dwSizeImage = height*(DWORD)((width*depth/8+3)&~3);

    if(depth == 24)
        m_hImage= (HDIB)GlobalAlloc(GHND,sizeof(BITMAPINFOHEADER)+dwSizeImage);
    else
        m_hImage= (HDIB)GlobalAlloc(GHND,sizeof(BITMAPINFOIIEADER)+dwSizeImage + 1024)

    if (m_hImage == NULL)
        return FALSE;

    lpbi = (LPBITMAPINFOHEADER)GlobalLock(m_hImage);
    lpbi->biSize            = sizeof(BITMAPINFOHEADER) ;
    lpbi->biWidth           = width;
    lpbi->biHeight          = height;
    lpbi->biPlanes          = 1;
    lpbi->biBitCount        = depth;
    lpbi->biCompression     = BI_RGB ;
    lpbi->biSizeImage       = dwSizeImage;
    lpbi->biXPelsPerMeter   = 0 ;
    lpbi->biYPelsPerMeter   = 0 ;
    lpbi->biClrUsed         = 0 ;
    lpbi->biClrImportant    = 0 ;

    lpPal = (BYTE *) lpbi;
    if (depth == 8)
    {
        lpbi->biClrUsed = 256;

        DWORD offDest = sizeof(BITMAPINFOHEADER);
        for(i = 0; i < 256; i++)
        {
            lpPal[offDest++] = (BYTE)i;
            lpPal[offDest++] = (BYTE)i;
            lpPal[offDest++] = (BYTE)i;
            lpPal[offDest++] = 0x00;
```

```
        }
    }

    InitDIB(FALSE);
    return TRUE;
}

BOOL CImage::CreateDIBPalette()
{
    LPLOGPALETTE lpPal;        // pointer to a logical palette
    HANDLE hLogPal;            // handle to a logical palette
    HPALETTE hPal = NULL;      // handle to a palette
    int i;                     // loop index
    WORD wNumColors;           // number of colors in color table
    LPSTR lpbi;                // pointer to packed-DIB
    LPBITMAPINFO lpbmi;        // pointer to BITMAPINFO structure (Win3.0)
    LPBITMAPCOREINFO lpbmc;    // pointer to BITMAPCOREINFO structure (old)
    BOOL bWinStyleDIB;         // flag which signifies whether this is a Win3.0 DIB
    BOOL bResult = FALSE;

    /* if handle to DIB is invalid, return FALSE */

    if (m_hImage == NULL)
      return FALSE;

  lpbi = (LPSTR) ::GlobalLock((HGLOBAL) m_hImage);

  /* get pointer to BITMAPINFO (Win 3.0) */
  lpbmi = (LPBITMAPINFO)lpbi;

  /* get pointer to BITMAPCOREINFO (old 1.x) */
  lpbmc = (LPBITMAPCOREINFO)lpbi;

  /* get the number of colors in the DIB */
  wNumColors = ::DIBNumColors(lpbi);

  if (wNumColors != 0)
  {
        /* allocate memory block for logical palette */
        hLogPal = ::GlobalAlloc(GHND, sizeof(LOGPALETTE)
                                      + sizeof(PALETTEENTRY)
                                      * wNumColors);

        /* if not enough memory, clean up and return NULL */
        if (hLogPal == 0)
        {
            ::GlobalUnlock((HGLOBAL) m_hImage);
            return FALSE;
        }

        lpPal = (LPLOGPALETTE) ::GlobalLock((HGLOBAL) hLogPal);

        /* set version and number of palette entries */
        lpPal->palVersion = PALVERSION;
        lpPal->palNumEntries = (WORD)wNumColors;

        /* is this a Win 3.0 DIB? */
        bWinStyleDIB = IS_WIN30_DIB(lpbi);
        for (i = 0; i < (int)wNumColors; i++)
```

```
        {
            if (bWinStyleDIB)
            {
                lpPal->palPalEntry[i].peRed = lpbmi->bmiColors[i].rgbRed;
                lpPal->palPalEntry[i].peGreen = lpbmi->bmiColors[i].rgbGreen;
                lpPal->palPalEntry[i].peBlue = lpbmi->bmiColors[i].rgbBlue;
                lpPal->palPalEntry[i].peFlags = 0;
            }
            else
            {
                lpPal->palPalEntry[i].peRed = lpbmc->bmciColors[i].rgbtRed;
                lpPal->palPalEntry[i].peGreen = lpbmc->bmciColors[i].rgbtGreen
                lpPal->palPalEntry[i].peBlue = lpbmc->bmciColors[i].rgbtBlue;
                lpPal->palPalEntry[i].peFlags = 0;
            }
        }

        /* create the palette and get handle to it */
        bResult = m_pPal->CreatePalette(lpPal);
        ::GlobalUnlock((HGLOBAL) hLogPal);
        ::GlobalFree((HGLOBAL) hLogPal);
    }

    ::GlobalUnlock((HGLOBAL) m_hImage);

    return bResult;
}

/*********************************************************
                    소멸자, 정리 함수
*********************************************************/

void CImage::Free()
{
    if( m_hImage )
    {
        if( GlobalFree( m_hImage ) != NULL)
        {
            TRACE("Can't free handle in CImage::Free()");
        }
        m_hImage = NULL;
    }
    if( m_hUndoImage )
    {
        if( GlobalFree( m_hUndoImage ) != NULL)
        {
            TRACE("Can't free handle in CRawImage::Free()");
        }
        m_hUndoImage = NULL;
    }

    if(m_pPal != NULL)
    {
        delete m_pPal;
        m_pPal = NULL;
    }
}

/*********************************************************
```

```
                    이미지 정보를 얻는 함수
*******************************************************/

int CImage::GetBitCount()
{
    if (m_hImage == NULL) return -1;
    LPBITMAPINFOHEADER lpbi;
    lpbi = (LPBITMAPINFOHEADER) ::GlobalLock((HGLOBAL) m_hImage );
    return lpbi->biBitCount;
}



/*******************************************************
                          그리기
*******************************************************/

BOOL CImage::Draw(HDC hDC, LPRECT lpDIBRect, LPRECT lpDCRect)
{
    LPSTR   lpDIBHdr;      // BITMAPINFOHEADER를 가리킬 포인터
    LPSTR   lpDIBBits;     // DIB 비트를 가리킬 포인터
    BOOL        bSuccess=FALSE;   // Success/fail 플래그
    HPALETTE    hPal=NULL;        // DIB 팔레트
    HPALETTE    hOldPal=NULL;     // 이전 팔레트

    // 메모리 고정
    lpDIBHdr  = (LPSTR) ::GlobalLock((HGLOBAL) m_hImage);
    // DIB 비트가 저장되어 있는 곳의 주소를 얻음
    lpDIBBits = ::FindDIBBits(lpDIBHdr);

    // 팔레트를 얻어 DC에 선택
    if(m_pPal != NULL)
    {
        hPal = (HPALETTE) m_pPal->m_hObject;
        hOldPal = ::SelectPalette(hDC, hPal, TRUE);
    }

    ::SetStretchBltMode(hDC, COLORONCOLOR);

    if ((RECTWIDTH(lpDCRect)  == RECTWIDTH(lpDIBRect)) &&
       (RECTHEIGHT(lpDCRect) == RECTHEIGHT(lpDIBRect)))
        // 원래 크기로 그대로 출력하는 경우
        bSuccess = ::SetDIBitsToDevice(hDC, // hDC
            lpDCRect->left,                 // DestX
            lpDCRect->top,                  // DestY
            RECTWIDTH(lpDCRect),            // nDestWidth
            RECTHEIGHT(lpDCRect),           // nDestHeight
            lpDIBRect->left,                // SrcX
            (int)DIBHeight(lpDIBHdr) - lpDIBRect->top -   RECTHEIGHT(lpDIBRect),        // SrcY
            0,                              // nStartScan
            (WORD)DIBHeight(lpDIBHdr),      // nNumScans
            lpDIBBits,                      // lpBits
            (LPBITMAPINFO)lpDIBHdr,         // lpBitsInfo
            DIB_RGB_COLORS);                // wUsage
    else // 확대 또는 축소하여 출력하는 경우
        bSuccess = ::StretchDIBits(hDC,   // hDC
            lpDCRect->left,                 // DestX
            lpDCRect->top,                  // DestY
            RECTWIDTH(lpDCRect),            // nDestWidth
```

```
                RECTHEIGHT(lpDCRect),           // nDestHeight
                lpDIBRect->left,                // SrcX
                lpDIBRect->top,                 // SrcY
                RECTWIDTH(lpDIBRect),           // wSrcWidth
                RECTHEIGHT(lpDIBRect),          // wSrcHeight
                lpDIBBits,                      // lpBits
                (LPBITMAPINFO)lpDIBHdr,         // lpBitsInfo
                DIB_RGB_COLORS,                     // wUsage
                SRCCOPY);                       // dwROP

    // 메모리 놓아줌
   ::GlobalUnlock((HGLOBAL) m_hImage);
    // DC 복원
    if (hOldPal != NULL) ::SelectPalette(hDC, hOldPal, TRUE);
    return bSuccess;
}

/*************************************************
                Undo 처리를 위한 함수
*************************************************/
BOOL CImage::Undo()
{
    HDIB hTemp;

    // 백업이 존재하면 백업과 이미지 핸들을 교환
    if(m_hUndoImage)
    {
        hTemp = m_hImage;
        m_hImage = m_hUndoImage;
        m_hUndoImage = hTemp;
        return TRUE;
    }
    else
        return FALSE;
}

BOOL CImage::PrepareUndo()
{
    // 이미 백업이 존재하면 이를 해제함
    if(m_hUndoImage)
    {
        ::GlobalFree((HGLOBAL)m_hUndoImage);
        m_hUndoImage = NULL;
    }

    // 이미지를 통째로 복사하여 백업을 만듦
    if ((m_hUndoImage = (HDIB)::CopyHandle((HGLOBAL)m_hImage)) == NULL)
    {
        m_hUndoImage = NULL;
        return FALSE;
    }
    return TRUE;
}

/*************************************************
                파일 읽어오기, 저장하기
*************************************************/

BOOL CImage::Save(LPCTSTR lpszFileName)
```

```
{
    CString filetype;
    filetype = lpszFileName;
    filetype.MakeUpper();

    if(filetype.Find(".BMP") > -1) return SaveBMP(lpszFileName);
    else return FALSE;
}

BOOL CImage::Load(LPCTSTR lpszFileName)
{
    CString filetype;
    filetype = lpszFileName;
    filetype.MakeUpper();

    if(filetype.Find(".BMP") > -1) return LoadBMP(lpszFileName);
    else return FALSE;
}

/*********************************************************
  실제 그래픽 파일을 읽어오는 루틴은 다음 파일에 있음

    ImageFileBmp.cpp : BMP 파일 (LoadBMP, SaveBMP)
    ImageFileGif.cpp : GIF 파일 (LoadGIF, SaveGIF)
    ImageFileTif.cpp : TIFF 파일 (LoadTIF, SaveTIF)
    ImageFileJpg.cpp : JPEG 파일 (LoadJPG, SaveJPG)

*********************************************************/


/*********************************************************
                    DIB와 관련된 전역 함수
*********************************************************/

LPSTR WINAPI FindDIBBits(LPSTR lpbi)
{
    return (lpbi + *(LPDWORD)lpbi + ::PaletteSize(lpbi));
}


DWORD WINAPI DIBWidth(LPSTR lpDIB)
{
    LPBITMAPINFOHEADER lpbmi;  // pointer to a Win 3.0-style DIB
    LPBITMAPCOREHEADER lpbmc;  // pointer to an other-style DIB

    /* point to the header (whether Win 3.0 and old) */

    lpbmi = (LPBITMAPINFOHEADER)lpDIB;
    lpbmc = (LPBITMAPCOREHEADER)lpDIB;

    /* return the DIB width if it is a Win 3.0 DIB */
    if (IS_WIN30_DIB(lpDIB))
        return lpbmi->biWidth;
    else  /* it is an other-style DIB, so return its width */
        return (DWORD)lpbmc->bcWidth;
}


DWORD WINAPI DIBHeight(LPSTR lpDIB)
```

```
{
    LPBITMAPINFOHEADER lpbmi;  // pointer to a Win 3.0-style DIB
    LPBITMAPCOREHEADER lpbmc;  // pointer to an other-style DIB

    /* point to the header (whether old or Win 3.0 */

    lpbmi = (LPBITMAPINFOHEADER)lpDIB;
    lpbmc = (LPBITMAPCOREHEADER)lpDIB;

    /* return the DIB height if it is a Win 3.0 DIB */
    if (IS_WIN30_DIB(lpDIB))
        return lpbmi->biHeight;
    else  /* it is an other-style DIB, so return its height */
        return (DWORD)lpbmc->bcHeight;
}


WORD WINAPI PaletteSize(LPSTR lpbi)
{
   /* calculate the size required by the palette */
   if (IS_WIN30_DIB (lpbi))
       return (WORD)(::DIBNumColors(lpbi) * sizeof(RGBQUAD));
   else
       return (WORD)(::DIBNumColors(lpbi) * sizeof(RGBTRIPLE));
}


WORD WINAPI DIBNumColors(LPSTR lpbi)
{
    WORD wBitCount;  // DIB bit count

    /*  If this is a Windows-style DIB, the number of colors in the
     *  color table can be less than the number of bits per pixel
     *  allows for (i.e. lpbi->biClrUsed can be set to some value).
     *  If this is the case, return the appropriate value.
     */

    if (IS_WIN30_DIB(lpbi))
    {
        DWORD dwClrUsed;

        dwClrUsed = ((LPBITMAPINFOHEADER)lpbi)->biClrUsed;
        if (dwClrUsed != 0)
            return (WORD)dwClrUsed;
    }

    /*  Calculate the number of colors in the color table based on
     *  the number of bits per pixel for the DIB.
     */
    if (IS_WIN30_DIB(lpbi))
        wBitCount = ((LPBITMAPINFOHEADER)lpbi)->biBitCount;
    else
        wBitCount = ((LPBITMAPCOREHEADER)lpbi)->bcBitCount;

    /* return number of colors based on bits per pixel */
    switch (wBitCount)
    {
```

```
        case 1:
            return 2;

        case 4:
            return 16;

        case 8:
            return 256;

        default:
            return 0;
    }
}


/*******************************************************
                클립보드를 위한 전역 함수
*******************************************************/

HGLOBAL WINAPI CopyHandle (HGLOBAL h)
{
    if (h == NULL)
        return NULL;

    DWORD dwLen = ::GlobalSize((HGLOBAL) h);
    HGLOBAL hCopy = ::GlobalAlloc(GHND, dwLen);

    if (hCopy != NULL)
    {
        void* lpCopy = ::GlobalLock((HGLOBAL) hCopy);
        void* lp     = ::GlobalLock((HGLOBAL) h);
        memcpy(lpCopy, lp, dwLen);
        ::GlobalUnlock(hCopy);
        ::GlobalUnlock(h);
    }

    return hCopy;
}

// ImageAdjust.cpp: implementation of the CImageAdjust class.
//
//////////////////////////////////////////////////////////////////////

#include "stdafx.h"
#include "Image.h"
#include "ImageAdjust.h"
#include "ImagePixel.h"

#ifdef _DEBUG
#undef THIS_FILE
static char THIS_FILE[]=__FILE__;
#define new DEBUG_NEW
#endif

//////////////////////////////////////////////////////////////////////
// Construction/Destruction
//////////////////////////////////////////////////////////////////////

CImageAdjust::CImageAdjust()
```

```
{
    m_pImage = NULL;
}

CImageAdjust::~CImageAdjust()
{

}

CImageAdjust::CImageAdjust(CImage &Image)
{
    m_pImage = &Image;
    m_nWidth = m_pImage->GetWidth();
    m_nHeight = m_pImage->GetHeight();
    m_nDepth = m_pImage->GetBitCount();
}

void CImageAdjust::Invert()
{
    ASSERT(m_pImage);
    ASSERT(m_pImage->GetHandle() != NULL);

    if(m_nDepth == 8) InvertGray();
    else InvertColor();
}

void CImageAdjust::InvertGray()
{
    CPixelPtr ptr(*m_pImage);

    int i, j;

    for(j=0 ; j<m_nHeight ; j++)
    {
        for(i=0 ; i<m_nWidth ; i++)
        {
            ptr[j][i] =~ptr[j][i];
        }
    }
}

void CImageAdjust::InvertColor()
{
    CColorPixelPtr ptr(*m_pImage);

    int i, j;

    for(j=0 ; j<m_nHeight ; j++)
    {
        for(i=0 ; i<m_nWidth ; i++)
        {
            ptr[j][i].R =~ptr[j][i].R;
            ptr[j][i].G =~ptr[j][i].G;
            ptr[j][i].B =~ptr[j][i].B;
        }
    }
}

void CImageAdjust::Brightness(int nAdjustStep)
```

```
{
    ASSERT(m_pImage);
    ASSERT(m_pImage->GetHandle() != NULL);

    if(m_nDepth == 8) BrightnessGray(nAdjustStep);
    else BrightnessColor(nAdjustStep);
}

void CImageAdjust::BrightnessGray(int nAdjustStep)
{

    CPixelPtr ptr(*m_pImage);
    CPixel p;

    for(int j=0 ; j<m_nHeight ; j++)
    {
        for(int i=0 ; i<m_nWidth ; i++)
        {
            p << ptr[j][i];
            p += nAdjustStep;
            p >> ptr[j][i];
        }

    }
}

void CImageAdjust::BrightnessColor(int nAdjustStep)
{
    CColorPixelPtr ptr(*m_pImage);
    CColorPixel p;

    for(int j=0 ; j<m_nHeight ; j++)
    {
        for(int i=0 ; i<m_nWidth ; i++)
        {
            p << ptr[j][i];
            p += nAdjustStep;
            p >> ptr[j][i];
        }
    }
}

void CImageAdjust::Contrast(int nAdjustStep)
{
    ASSERT(m_pImage);
    ASSERT(m_pImage->GetHandle() != NULL);

    if(m_nDepth == 8) ContrastGray(nAdjustStep);
    else ContrastColor(nAdjustStep);
}

void CImageAdjust::ContrastGray(int nAdjustStep)
{

    CPixelPtr ptr(*m_pImage);
    CPixel p;

    for(int j=0 ; j<m_nHeight ; j++)
    {
```

```
            for(int i=0 ; i<m_nWidth ; i++)
            {
                p << ptr[j][i];
                p += (p-128)*nAdjustStep/128;
                p >> ptr[j][i];
            }
        }
}

void CImageAdjust::ContrastColor(int nAdjustStep)
{
    CColorPixelPtr ptr(*m_pImage);
    CColorPixel p;

    for(int j=0 ; j<m_nHeight ; j++)
    {
        for(int i=0 ; i<m_nWidth ; i++)
        {
            p << ptr[j][i];
            p += (p-128)*nAdjustStep/128;
            p >> ptr[j][i];
        }
    }
}


#include "stdafx.h"
#include "Image.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

#define DIB_HEADER_MARKER   ((WORD) ('M' << 8) | 'B')
//////////////////////////////////////////////////////////////////////////////
// CCImage load/save bmp

BOOL CImage::SaveBMP(LPCTSTR lpszFileName)
{
    CFile file;
    CFileException fe;
    BITMAPFILEHEADER bmfHdr;
    LPBITMAPINFOHEADER lpBI;
    DWORD dwDIBSize;

    // 쓰기 모드로 파일 열기
    if (!file.Open(lpszFileName, CFile::modeCreate | CFile::modeWrite | CFile::shareDenyWrite, &fe)) return FALSE;

    // 메모리 핸들이 유효한지 확인
    if (m_hImage == NULL) return FALSE;

    // 메모리 고정
    lpBI = (LPBITMAPINFOHEADER)::GlobalLock((HGLOBAL)m_hImage);
    if (lpBI == NULL) return FALSE;

    // 비트맵 파일 헤더 정보를 설정
    bmfHdr.bfType = DIB_HEADER_MARKER;  // "BM"
```

```
    dwDIBSize = *(LPDWORD)lpBI + ::PaletteSize((LPSTR)lpBI);
    if((lpBI->biCompression==BI_RLE8) || (lpBI->biCompression==BI_RLE4))
        dwDIBSize += lpBI->biSizeImage;
    else
    {
        DWORD dwBmBitsSize;  // Size of Bitmap Bits only
        dwBmBitsSize = WIDTHBYTES((lpBI->biWidth)*((DWORD)lpBI->biBitCount)) * lpBI->biHeight;
        dwDIBSize += dwBmBitsSize;
        lpBI->biSizeImage = dwBmBitsSize;
    }

    bmfHdr.bfSize = dwDIBSize + sizeof(BITMAPFILEHEADER);
    bmfHdr.bfReserved1 = 0;
    bmfHdr.bfReserved2 = 0;
    bmfHdr.bfOffBits=(DWORD)sizeof(BITMAPFILEHEADER)+lpBI->biSize + PaletteSize((LPSTR)lpBI);
    TRY
    {
        // 비트맵 파일 헤더를 파일에 쓰기
        file.Write((LPSTR)&bmfHdr, sizeof(BITMAPFILEHEADER));
        // 나머지 데이터를 파일에 쓰기
        file.WriteHuge(lpBI, dwDIBSize);
    }
    CATCH (CFileException, e)
    {
        ::GlobalUnlock((HGLOBAL) m_hImage);
        THROW_LAST();
    }
    END_CATCH

    // 메모리 풀어줌
    ::GlobalUnlock((HGLOBAL) m_hImage);
    return TRUE;
}


BOOL CImage::LoadBMP(LPCTSTR lpszFileName)
{
    CFile file;
    CFileException fe;
    LPSTR pDIB;
    DWORD dwBitsSize;
    BITMAPFILEHEADER bmfHeader;

    // 읽기 모드로 파일 열기
    if(!file.Open(lpszFileName, CFile::modeRead|CFile::shareDenyWrite, &fe))
        return FALSE;

    // 파일의 길이를 구함
    dwBitsSize = file.GetLength();

    // 파일 헤더 읽기
    if(file.Read((LPSTR)&bmfHeader, sizeof(bmfHeader))!=sizeof(bmfHeader))
        return FALSE;

    // BMP 파일임을 나타내는 "BM" 마커가 있는지 확인
    if (bmfHeader.bfType != DIB_HEADER_MARKER)
        return FALSE;

    // 메모리 할당
```

```
    if((m_hImage = (HDIB)::GlobalAlloc(GMEM_MOVEABLE | GMEM_ZEROINIT, dwBitsSize)) == NULL) return FALSE

    // 메모리 고정
    pDIB = (LPSTR) ::GlobalLock((HGLOBAL) m_hImage);

    // 파일 읽기
    if (file.ReadHuge(pDIB, dwBitsSize - sizeof(BITMAPFILEHEADER)) != dwBitsSize - sizeof(BITMAPFILEHEADER) )
    {
        ::GlobalUnlock((HGLOBAL) m_hImage);
        ::GlobalFree((HGLOBAL) m_hImage);
        return FALSE;
    }

    // 메모리 풀어줌
    ::GlobalUnlock((HGLOBAL) m_hImage);

    // DIB 초기화
    InitDIB();

    return TRUE;
}

// ImageFiltering.cpp: implementation of the CImageFiltering class.
//
//////////////////////////////////////////////////////////////////////

#include "stdafx.h"
#include "Image.h"
#include "ImageFiltering.h"
#include "ImagePixel.h"

#ifdef _DEBUG
#undef THIS_FILE
static char THIS_FILE[]=__FILE__;
#define new DEBUG_NEW
#endif

//////////////////////////////////////////////////////////////////////
// Construction/Destruction
//////////////////////////////////////////////////////////////////////

CImageFiltering::CImageFiltering()
{
    m_pImage = NULL;
}

CImageFiltering::CImageFiltering(CImage &Image)
{
    m_pImage = &Image;
    m_nWidth = m_pImage->GetWidth();
    m_nHeight = m_pImage->GetHeight();
    m_nDepth = m_pImage->GetBitCount();
}

CImageFiltering::~CImageFiltering()
{

}
```

```
void CImageFiltering::Blur()
{
    int pnFilter[9] = {1, 3, 1, 3, 4, 3, 1, 3, 1};
    Filtering(pnFilter, _T("부드럽게 하기"));
}

void CImageFiltering::Sharpening()
{
    int pnFilter[9] = {0, -1, 0, -1, 5, -1, 0, -1, 0};
    Filtering(pnFilter, _T("날카롭게 하기"));
}

void CImageFiltering::Laplacian()
{
    int pnFilter[9] = {-1, -1, -1, -1, 8, -1, -1, -1, -1};
    Filtering(pnFilter, _T("경계선 추출"));
}

void CImageFiltering::Embossing()
{
    int pnFilter[9] = {1, 1, 0, 1, 0, -1, 0, -1, -1};
    Filtering(pnFilter, _T("엠보싱"));
}

void CImageFiltering::Filtering(int * pnFilter, CString strProgressBar)
{
    ASSERT(m_pImage);
    ASSERT(m_pImage->GetHandle() != NULL);
    ASSERT(m_pImage->GetUndoHandle() != NULL);

    if(m_nDepth == 8) FilteringGray(pnFilter, strProgressBar);
    else FilteringColor(pnFilter, strProgressBar);
}

void CImageFiltering::FilteringGray(int * pnFilter, CString strProgressBar)
{
    int i, j, tot=0;
    CPixelPtr ptrSorce(m_pImage->GetUndoHandle());
    CPixelPtr ptrDest(*m_pImage);
    CPixel p;

    for(i=0 ; i<8 ; i++)
        tot += pnFilter[i];

    for(j=1 ; j<m_nHeight-1 ; j++)
    {
        for(i=1 ; i<m_nWidth-1 ; i++)
        {
            p = ptrSorce[j-1][i-1]*pnFilter[0] +
                ptrSorce[j-1][ i ]*pnFilter[1] +
                ptrSorce[j-1][i+1]*pnFilter[2] +
                ptrSorce[ j ][i-1]*pnFilter[3] +
                ptrSorce[ j ][ i ]*pnFilter[4] +
                ptrSorce[ j ][i+1]*pnFilter[5] +
                ptrSorce[j+1][i-1]*pnFilter[6] +
                ptrSorce[j+1][ i ]*pnFilter[7] +
                ptrSorce[j+1][i+1]*pnFilter[8];
            if(tot != 0 && tot != 1) p /= tot;
            p >> ptrDest[j][i];
```

```
        }
    }
}

void CImageFiltering::FilteringColor(int * pnFilter, CString strProgressBar)
{
    int i, j, tot=0;
    CColorPixelPtr ptrSorce(m_pImage->GetUndoHandle());
    CColorPixelPtr ptrDest(*m_pImage);
    CColorPixel s[9];
    CColorPixel p;

    for(i=0 ; i<8 ; i++)
        tot += pnFilter[i];

    for(j=1 ; j<m_nHeight-1 ; j++)
    {
        for(i=1 ; i<m_nWidth-1 ; i++)
        {
            s[0] = ptrSorce[j-1][i-1];
            s[1] = ptrSorce[j-1][ i ];
            s[2] = ptrSorce[j-1][i+1];
            s[3] = ptrSorce[ j ][i-1];
            s[4] = ptrSorce[ j ][ i ];
            s[5] = ptrSorce[ j ][i+1];
            s[6] = ptrSorce[j+1][i-1];
            s[7] = ptrSorce[j+1][ i ];
            s[8] = ptrSorce[j+1][i+1];

            p = s[0]*pnFilter[0]+
                s[1]*pnFilter[1]+
                s[2]*pnFilter[2]+
                s[3]*pnFilter[3]+
                s[4]*pnFilter[4]+
                s[5]*pnFilter[5]+
                s[6]*pnFilter[6]+
                s[7]*pnFilter[7]+
                s[8]*pnFilter[8];
            if(tot != 0 && tot != 1) p /= tot;
            p >> ptrDest[j][i];
        }
    }
}


// ImageGeometry.cpp: implementation of the CImageGeometry class.
//
//////////////////////////////////////////////////////////////////////

#include "stdafx.h"
#include "Image.h"
#include "ImageGeometry.h"
#include "ImagePixel.h"

#ifdef _DEBUG
#undef THIS_FILE
static char THIS_FILE[]=__FILE__;
#define new DEBUG_NEW
#endif
```

```
//////////////////////////////////////////////////////////////////////
// Construction/Destruction
//////////////////////////////////////////////////////////////////////

CImageGeometry::CImageGeometry()
{
    m_pImage = NULL;
}

CImageGeometry::~CImageGeometry()
{

}

CImageGeometry::CImageGeometry(CImage &Image)
{
    m_pImage = &Image;
    m_nWidth = m_pImage->GetWidth();
    m_nHeight = m_pImage->GetHeight();
    m_nDepth = m_pImage->GetBitCount();
}

void CImageGeometry::FlipVertical()
{
    ASSERT(m_pImage);
    ASSERT(m_pImage->GetHandle() != NULL);
    ASSERT(m_pImage->GetUndoHandle() != NULL);

    if(m_nDepth == 8) FlipVerticalGray();
    else FlipVerticalColor();
}

void CImageGeometry::FlipVerticalGray()
{

    BYTE *temp;

    CPixelPtr ptr(*m_pImage);
    temp = new BYTE [m_nWidth];

    for(int j=0 ; j<m_nHeight/2 ; j++)
    {
        memcpy(temp, ptr[j], m_nWidth);
        memcpy(ptr[j], ptr[m_nHeight-j-1], m_nWidth);
        memcpy(ptr[m_nHeight-j-1], temp, m_nWidth);

    }
    delete temp;
}

void CImageGeometry::FlipVerticalColor()
{

    BYTE *temp;

    CColorPixelPtr ptrColor(*m_pImage);
    temp = new BYTE [m_nWidth*3];

    for(int j=0 ; j<m_nHeight/2 ; j++)
```

```
    {
        memcpy(temp, ptrColor[j], m_nWidth*3);
        memcpy(ptrColor[j], ptrColor[m_nHeight-j-1], m_nWidth*3);
        memcpy(ptrColor[m_nHeight-j-1], temp, m_nWidth*3);

    }
    delete temp;
}

void CImageGeometry::FlipHorizontal()
{
    ASSERT(m_pImage);
    ASSERT(m_pImage->GetHandle() != NULL);
    ASSERT(m_pImage->GetUndoHandle() != NULL);

    if(m_nDepth == 8) FlipHorizontalGray();
    else FlipHorizontalColor();
}

void CImageGeometry::FlipHorizontalGray()
{


    CPixelPtr ptr(*m_pImage);
    BYTE g;

    for(int j=0 ; j<m_nHeight ; j++)
    {
        for(int i=0 ; i<m_nWidth/2; i++)
        {
            g = ptr[j][i];
            ptr[j][i] = ptr[j][m_nWidth-i-1];
            ptr[j][m_nWidth-i-1] = g;
        }

    }
}

void CImageGeometry::FlipHorizontalColor()
{


    CColorPixelPtr ptrColor(*m_pImage);
    CColorPixel a, b;

    for(int j=0 ; j<m_nHeight ; j++)
    {
        for(int i=0 ; i<m_nWidth/2 ; i++)
        {
            a << ptrColor[j][i];
            b << ptrColor[j][m_nWidth-i-1];
            a >> ptrColor[j][m_nWidth-i-1];
            b >> ptrColor[j][i];
        }

    }
}
```

```
void CImageGeometry::TurnRight()
{
    ASSERT(m_pImage);
    ASSERT(m_pImage->GetHandle() != NULL);
    ASSERT(m_pImage->GetUndoHandle() != NULL);

    if(m_nDepth == 8) TurnRightGray();
    else TurnRightColor();
}

void CImageGeometry::TurnRightGray()
{

    GlobalFree(m_pImage->GetHandle());
    m_pImage->Create(m_nHeight, m_nWidth, m_nDepth);
    m_pImage->InitDIB(FALSE);

    CPixelPtr Destptr(*m_pImage);
    CPixelPtr ptr(m_pImage->GetUndoHandle());

    for(int j=0 ; j<m_nWidth ; j++)
    {
        for(int i=0 ; i<m_nHeight ; i++)
            Destptr[j][m_nHeight-i-1] = ptr[i][j];

    }
}

void CImageGeometry::TurnRightColor()
{

    GlobalFree(m_pImage->GetHandle());
    m_pImage->Create(m_nHeight, m_nWidth, m_nDepth);
    m_pImage->InitDIB(FALSE);

    CColorPixelPtr DestptrColor(*m_pImage);
    CColorPixelPtr ptrColor(m_pImage->GetUndoHandle());

    for(int j=0 ; j<m_nWidth ; j++)
    {
        for(int i=0 ; i<m_nHeight ; i++)
            memcpy(&DestptrColor[j][m_nHeight-i-1], &ptrColor[i][j], 3);

    }
}


void CImageGeometry::TurnLeft()
{
    ASSERT(m_pImage);
    ASSERT(m_pImage->GetHandle() != NULL);
    ASSERT(m_pImage->GetUndoHandle() != NULL);

    if(m_nDepth == 8) TurnLeftGray();
    else TurnLeftColor();
}

void CImageGeometry::TurnLeftGray()
{
```

```
    GlobalFree(m_pImage->GetHandle());
    m_pImage->Create(m_nHeight, m_nWidth, m_nDepth);
    m_pImage->InitDIB(FALSE);

    CPixelPtr Destptr(*m_pImage);
    CPixelPtr ptr(m_pImage->GetUndoHandle());

    for(int j=0 ; j<m_nWidth ; j++)
    {
        for(int i=0 ; i<m_nHeight ; i++)
            Destptr[m_nWidth-j-1][i] = ptr[i][j];

    }
}

void CImageGeometry::TurnLeftColor()
{


    GlobalFree(m_pImage->GetHandle());
    m_pImage->Create(m_nHeight, m_nWidth, m_nDepth);
    m_pImage->InitDIB(FALSE);

    CColorPixelPtr DestptrColor(*m_pImage);
    CColorPixelPtr ptrColor(m_pImage->GetUndoHandle());

    for(int j=0 ; j<m_nWidth ; j++)
    {
        for(int i=0 ; i<m_nHeight ; i++)
            memcpy(&DestptrColor[m_nWidth-j-1][i], &ptrColor[i][j], 3);

    }
}


////////////////////////////////////////////////////////////////////////////
//  PixelPointer.cpp

#include "stdafx.h"
#include "ImagePixel.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

//////////////////////////////////////////////////////////////////////
// CColorPixel
CColorPixel::CColorPixel()
{

}

CColorPixel::~CColorPixel()
{

}
```

```
//////////////////////////////////////////////////////////////////
// CColorPixelPtr
CColorPixelPtr::CColorPixelPtr(CImage &Im)
{
    m_nHeight = Im.GetHeight();        // 이미지의 가로 길이
    int nWidth = Im.GetRealWidth();   // 이미지의 세로 길이

    // 이미지의 세로 길이 만큼 포인터 배열 할당
    m_pPtr = new LPCOLOR [m_nHeight];

    // 메모리 블록을 고정
    m_hHandle = Im.GetHandle();
    LPSTR lpDIBHdr  = (LPSTR)::GlobalLock((HGLOBAL) m_hHandle);

    // 헤더 다음에 나오는 첫번째 픽셀의 포인터를 얻음
    LPSTR lpDIBBits = (LPSTR)::FindDIBBits(lpDIBHdr);

    // 가로 길이 만큼씩 진행하며 m_pPtr에 이미지의
    // 각 줄의 첫번째 픽셀의 주소를 뒤집어서 저장
    for(int i=m_nHeight-1 ; i>=0 ; i--)
    {
        m_pPtr[i] = (LPCOLOR)lpDIBBits;
        lpDIBBits += nWidth;
    }
}

CColorPixelPtr::CColorPixelPtr(HDIB hHandle)
{
    // 메모리 블록을 고정
    LPSTR lpDIBHdr  = (LPSTR) ::GlobalLock((HGLOBAL) hHandle);
    m_hHandle = hHandle;

    m_nHeight = ::DIBHeight(lpDIBHdr);    // 이미지의 세로 길이
    int nWidth = WIDTHBYTES(::DIBWidth(lpDIBHdr)*24);    // 이미지의 가로 길

    // 이미지의 세로 길이 만큼 포인터 배열 할당
    m_pPtr = new LPCOLOR [m_nHeight];

    // 헤더 다음에 나오는 첫번째 픽셀의 포인터를 얻음
    LPSTR lpDIBBits = (LPSTR)::FindDIBBits(lpDIBHdr);

    // 가로 길이 만큼씩 진행하며 m_pPtr에 이미지의
    // 각 줄의 첫번째 픽셀의 주소를 뒤집어서 저장
    for(int i=m_nHeight-1 ; i>=0 ; i--)
    {
        m_pPtr[i] = (LPCOLOR)lpDIBBits;
        lpDIBBits += nWidth;
    }
}

CColorPixelPtr::~CColorPixelPtr()
{
    // 메모리 블록을 풀어줌
    GlobalUnlock((HGLOBAL) m_hHandle);
    TRY
    {
        // 할당 받았던 포인터 배열을 해제
        delete [] m_pPtr;
    }
```

```
    CATCH (CMemoryException, e)
    {
        THROW_LAST();
    }
    END_CATCH
}

////////////////////////////////////////////////////////////////////
// CPixel
CPixel::CPixel()
{

}

CPixel::~CPixel()
{

}

CPixel::CPixel(int i)
{
    I = i;
}

////////////////////////////////////////////////////////////////////
// CColorPixelPtr

CPixelPtr::CPixelPtr(CImage &Im)
{
    m_nHeight = Im.GetHeight();        // 이미지의 가로 길이
    int nWidth = Im.GetRealWidth();   // 이미지의 세로 길이

    // 이미지의 세로 길이 만큼 포인터 배열 할당
    m_pPtr = new BYTE *[m_nHeight];

    // 메모리 블록을 고정
    m_hHandle = Im.GetHandle();
    LPSTR lpDIBHdr  = (LPSTR) ::GlobalLock((HGLOBAL) m_hHandle);

    // 헤더 다음에 나오는 첫번째 픽셀의 포인터를 얻음
    BYTE *lpDIBBits = (BYTE *) ::FindDIBBits(lpDIBHdr);

    // 가로 길이 만큼씩 진행하며 m_pPtr에 이미지의
    // 각 줄의 첫번째 픽셀의 주소를 뒤집어서 저장
    for(int i=m_nHeight -1 ; i>=0 ; i--)
    {
        m_pPtr[i] = lpDIBBits;
        lpDIBBits += nWidth;
    }
}


CPixelPtr::CPixelPtr(HDIB hHandle)
{
    // 메모리 블록을 고정
    LPSTR lpDIBHdr  = (LPSTR) ::GlobalLock((HGLOBAL) hHandle);
    m_hHandle = hHandle;

    m_nHeight = ::DIBHeight(lpDIBHdr);   // 이미지의 세로 길이
```

```
    int nWidth = WIDTHBYTES(::DIBWidth(lpDIBHdr)*8);     // 이미지의 가로 길

    // 이미지의 세로 길이 만큼 포인터 배열 할당
    m_pPtr = new BYTE *[m_nHeight];

    // 헤더 다음에 나오는 첫번째 픽셀의 포인터를 얻음
    BYTE *lpDIBBits = (BYTE *)::FindDIBBits(lpDIBHdr);

    // 가로 길이 만큼씩 진행하며 m_pPtr에 이미지의
    // 각 줄의 첫번째 픽셀의 주소를 뒤집어서 저장
    for(int i=m_nHeight-1 ; i>=0 ; i--)
    {
        m_pPtr[i] = lpDIBBits;
        lpDIBBits += nWidth;
    }
}

CPixelPtr::~CPixelPtr()
{
    // 메모리 블록을 풀어줌
    GlobalUnlock((HGLOBAL) m_hHandle);
    TRY
    {
        // 할당 받았던 포인터 배열을 해제
        delete [] m_pPtr;
    }
    CATCH (CMemoryException, e)
    {
        THROW_LAST();
    }
    END_CATCH
}


// ImageProcess.cpp: implementation of the CImageProcess class.
//
//////////////////////////////////////////////////////////////////////

#include "stdafx.h"
#include "watermelon.h"
#include "ImageProcess.h"
#include "Image.h"
#include "ImagePixel.h"

#ifdef _DEBUG
#undef THIS_FILE
static char THIS_FILE[]=__FILE__;
#define new DEBUG_NEW
#endif

//////////////////////////////////////////////////////////////////////
// Construction/Destruction
//////////////////////////////////////////////////////////////////////

CImageProcess::CImageProcess()
{
    m_pImage = NULL;
}
```

```
CImageProcess::CImageProcess(CImage &Image)
{
    m_pImage = &Image;
    m_nWidth = m_pImage->GetWidth();
    m_nHeight = m_pImage->GetHeight();
    m_nDepth = m_pImage->GetBitCount();
}

CImageProcess::~CImageProcess()
{

}

// 주 이치화 함수
void CImageProcess::Threshold(CRect SRect,int CutValue)
{
    ASSERT(m_pImage);
    ASSERT(m_pImage->GetHandle() != NULL);

    if(m_nDepth == 8) ThresholdGray(SRect,CutValue);
    else ThresholdColor(SRect,CutValue);
}

//그레이 스케일 영상의 이치화 함수
void CImageProcess::ThresholdGray(CRect SRect,int CutValue)
{
    CPixelPtr ptr(*m_pImage);

    int i, j;

    for(j=0 ; j<m_nHeight ; j++)
    {
        for(i=0 ; i<m_nWidth ; i++)
        {
            if(i>=SRect.left && i<=SRect.right && j>=SRect.top && j<= SRect.bottom)
            {
                if(ptr[j][i]>CutValue)    ptr[j][i] =255;
                else                      ptr[j][i] =0;
            }
            else ptr[j][i] =255;
        }
    }
}


// 컬러 영상의 이치화 함수
void CImageProcess::ThresholdColor(CRect SRect,int CutValue)
{
    CColorPixelPtr ptr(*m_pImage);

    int i, j;

    for(j=0 ; j<m_nHeight ; j++)
    {
        for(i=0 ; i<m_nWidth ; i++)
        {
            if(i>=SRect.left && i<=SRect.right && j>=SRect.top && j<= SRect.bottom)
            {
                if(ptr[j][i].R>CutValue)   ptr[j][i].R =255;
```

```
                else                    ptr[j][i].R =0;
                if(ptr[j][i].G>CutValue)  ptr[j][i].G =255;
                else                    ptr[j][i].G =0;
                if(ptr[j][i].B>CutValue)  ptr[j][i].B =255;
                else                    ptr[j][i].B =0;
            }
            else
            {
                ptr[j][i].R =255;
                ptr[j][i].G =255;
                ptr[j][i].B =255;
            }
        }
    }
}


// 컬러 영상의 그레이 스케일 변환 함수
void CImageProcess::ConvertGray(CRect SRect)
{
    CColorPixelPtr ptr(*m_pImage);

    int i, j;

    for(j=0 ; j<m_nHeight ; j++)
    {
        for(i=0 ; i<m_nWidth ; i++)
        {
            if(i>=SRect.left && i<=SRect.right && j>=SRect.top && j<= SRect.bottom
            {
                ptr[j][i].R=ptr[j][i].G=ptr[j][i].B=(ptr[j][i].R + ptr[j][i].G + ptr[j][i].B)/3;
            }
            else ptr[j][i].R=ptr[j][i].G=ptr[j][i].B=255;
        }
    }
}

// 히스토 그램을 위한 밝기값 계수
void CImageProcess::CalHisto(BYTE pixel[256])
{
    CColorPixelPtr ptr(*m_pImage);
    int i,j,k;
    for(i=0;i<256;i++)pixel[i]=0;
    for(j=0 ; j<m_nHeight ; j++)
    {
        for(i=0 ; i<m_nWidth ; i++)
        {
            k=(ptr[j][i].R + ptr[j][i].G + ptr[j][i].B)/3;
            pixel[k]+=1;
        }
    }
}

/////////중심점 찾기 함수
bool CImageProcess::CalUV(CRect SRect,int point[2])
{
    CColorPixelPtr ptr(*m_pImage);
    int i,j;
    BYTE img [482*642];
```

```
    for(j=SRect.top;j<=SRect.bottom;j++)
        for(i=SRect.left;i<=SRect.right;i++)
        {
            if( !ptr[j][i].R
                && (!ptr[j][i-1].R || !ptr[j+1][i-1].R || !ptr[j-1][i-1].R)
                && (!ptr[j][i+1].R || !ptr[j+1][i+1].R || !ptr[j-1][i+1].R)
                && (!ptr[j-1][i].R || !ptr[j-1][i+1].R || !ptr[j-1][i-1].R)
                && (!ptr[j+1][i].R || !ptr[j+1][i+1].R || !ptr[j+1][i-1].R))
            {
                point[0]=i;
                point[1]=j;
                return 1;
            }
        }
        point[0]=0;
        point[1]=0;
        return 0;
}

// 세선화  함수 ////////////////////////////////////////////////
void CImageProcess::DeleteA(BYTE img[], BYTE timg[],int cx,int cy)
{
    int i, j;

    for(i=1;i<cy+1;i++)
    for(j=1;j<cx+1;j++)
        if(timg[(i  )*(m_nWidth+1)+(j  )])
        {
            img[(i  )*(m_nWidth+1)+(j  )] = 0;
            timg[(i  )*(m_nWidth+1)+(j  )] = 0;
        }
}

int CImageProcess::Nays(BYTE img[], int i, int j)
{

    int k,l,N=0;

    for(k=i-1;k<=i+1;k++)
    for(l=j-1;l<=j+1;l++)
        if(k!=i || l!=j)
            if(img[(k  )*(m_nWidth+1)+(l  )] >=1) N++;

    return N;
}

int CImageProcess::Connect(BYTE img[], int i, int j)
{

     int n=0;

     if( img[(i  )*(m_nWidth+1)+(j+1)] >= 1 && img[(i-1)*(m_nWidth+1)+(j+1)] == 0 ) n
     if( img[(i-1)*(m_nWidth+1)+(j+1)] >= 1 && img[(i-1)*(m_nWidth+1)+(j  )] == 0 ) n
     if( img[(i-1)*(m_nWidth+1)+(j  )] >= 1 && img[(i-1)*(m_nWidth+1)+(j-1)] == 0 ) n
     if( img[(i-1)*(m_nWidth+1)+(j-1)] >= 1 && img[(i  )*(m_nWidth+1)+(j-1)] == 0 ) n
     if( img[(i  )*(m_nWidth+1)+(j-1)] >= 1 && img[(i+1)*(m_nWidth+1)+(j-1)] == 0 ) n
     if( img[(i+1)*(m_nWidth+1)+(j-1)] >= 1 && img[(i+1)*(m_nWidth+1)+(j  )] == 0 ) n
     if( img[(i+1)*(m_nWidth+1)+(j  )] >= 1 && img[(i+1)*(m_nWidth+1)+(j+1)] == 0 ) n
     if( img[(i+1)*(m_nWidth+1)+(j+1)] >= 1 && img[(i  )*(m_nWidth+1)+(j+1)] == 0 ) n
```

```
    return n;
}

void CImageProcess::Thinning(CRect SRect)
{
    CColorPixelPtr ptr(*m_pImage);
    int i,j,N;
    BOOL OK;
    BYTE img [482*642];
    BYTE timg[482*642];
    BYTE Timg[482*642];
    for(i=SRect.top-1;i<SRect.bottom+2;i++)
    {
        img[(i   )*(m_nWidth+1)] = 0;
        img[(i   )*(m_nWidth+1)+(m_nWidth+1)] = 0;
    }
    for(j=SRect.left-1;j<SRect.right+2;j++)
    {
        img[j] = 0;
        img[m_nHeight+1+j] = 0;
    }
     for(i=SRect.top;i<SRect.bottom+1;i++)
        for(j=1;j<m_nWidth+1;j++)
              Timg[i*(m_nWidth+1)+j]=ptr[i-1][j-1].R;

    //영상의 검은색을 1, 배경을 0로 바꿈.
    for(i=SRect.top;i<SRect.bottom+1;i++)
    for(j=SRect.left;j<SRect.right+1;j++)
    {
        if(Timg[i*(m_nWidth+1)+j]>0) img[(i   )*(m_nWidth+1)+(j   )] = 0; //white
        else                         img[(i   )*(m_nWidth+1)+(j   )] = 1; // black
        timg[(i   )*(m_nWidth+1)+(j   )] = 0;                          // timg 초기화
    }

    while(OK)
    {
        OK = 0;
        // first sub-iteration
        for(i=SRect.top;i<SRect.bottom+1;i++)
        for(j=SRect.left;j<SRect.right+1;j++)
        {
            if(img[(i   )*(m_nWidth+1)+(j   )] != 1) continue;

           //영상의 중앙 부분을 중심으로 1인 영상의 개수 계산.
            N = Nays(img, i, j);

            if((N>=2 && N<=6) && Connect(img, i,j) == 1)
            {
                if((img[(i   )*(m_nWidth+1)+(j+1)]*img[(i-1)*(m_nWidth+1)+(j   )]*img[(i   )*(m_nWidth+1)+(j-1)]) == 0 &&
                    (img[(i-1)*(m_nWidth+1)+(j   )]*img[(i+1)*(m_nWidth+1)+(j   )]*img[(i   )*(m_nWidth+1)+(j-1)])== 0)
                {
                    timg[(i   )*(m_nWidth+1)+(j   )] = 1;
                    OK = 1;
                }
            }
        }
        DeleteA(img, timg, m_nWidth, m_nHeight);
        if(OK == 0) break;
```

```
        // Second sub-iteration
        for(i=SRect.top;i<SRect.bottom+1;i++)
        for(j=SRect.left;j<SRect.right+1;j++)
        {
            if(img[(i  )*(m_nWidth+1)+(j  )] != 1) continue;

            N = Nays(img, i, j);

            if((N>=2 && N<=6) && Connect(img,i,j) == 1)
            {
                if((img[(i-1)*(m_nWidth+1)+(j  )]*img[(i  )*(m_nWidth+1)+(j+1)]*img[(i+1)*(m_nWidth+1)+(j  )]) == 0 &&
                    (img[(i  )*(m_nWidth+1)+(j+1)]*img[(i+1)*(m_nWidth+1)+(j  )]*img[(i  )*(m_nWidth+1)+(j-1)]) == 0)
                {
                    timg[(i  )*(m_nWidth+1)+(j  )] = 1;
                    OK = 1;
                }
            }
        }
        DeleteA(img, timg, m_nWidth, m_nHeight);
    }
    // 1을 0로 0를 255로 다시 환원
    for(i=SRect.top;i<SRect.bottom+1;i++)
        for(j=SRect.left;j<SRect.right+1;j++)
        if(img[(i  )*(m_nWidth+1)+(j  )] > 0)
         {
              ptr[i-1][j-1].R =0;
              ptr[i-1][j-1].G =0;
              ptr[i-1][j-1].B =0;
         }
        else
         {
              ptr[i-1][j-1].R =255;
              ptr[i-1][j-1].G =255;
              ptr[i-1][j-1].B =255;
         }

}


// watermelonDlg.cpp : implementation file
//

#include "stdafx.h"
#include "watermelon.h"
#include "watermelonDlg.h"

#include "Image.h"
#include "ImagePixel.h"
#include "ImageAdjust.h"
#include "ImageFiltering.h"
#include "ImageProcess.h"
#include "CalMath.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
```

```
/////////////////////////////////////////////////////////////////////////////
// CWatermelonDlg dialog

CWatermelonDlg::CWatermelonDlg(CWnd* pParent /*=NULL*/)
    : CDialog(CWatermelonDlg::IDD, pParent)
{
    //{{AFX_DATA_INIT(CWatermelonDlg)
    //}}AFX_DATA_INIT
    // Note that LoadIcon does not require a subsequent DestroyIcon in Win32
    m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
}

void CWatermelonDlg::DoDataExchange(CDataExchange* pDX)
{
    CDialog::DoDataExchange(pDX);
    //{{AFX_DATA_MAP(CWatermelonDlg)
    DDX_Control(pDX, IDC_TEXTRV, m_TextRV);
    DDX_Control(pDX, IDC_TEXTRU, m_TextRU);
    DDX_Control(pDX, IDC_TEXTLV, m_TextLV);
    DDX_Control(pDX, IDC_TEXTLU, m_TextLU);
    DDX_Control(pDX, IDC_RV, m_RV);
    DDX_Control(pDX, IDC_RU, m_RU);
    DDX_Control(pDX, IDC_LV, m_LV);
    DDX_Control(pDX, IDC_LU, m_LU);
    DDX_Control(pDX, IDC_STEP, m_Step);
    DDX_Control(pDX, IDC_Z, m_Z);
    DDX_Control(pDX, IDC_Y, m_Y);
    DDX_Control(pDX, IDC_X, m_X);
    DDX_Control(pDX, IDC_TEXTZ, m_TextZ);
    DDX_Control(pDX, IDC_TEXTY, m_TextY);
    DDX_Control(pDX, IDC_TEXTX, m_TextX);
    DDX_Control(pDX, IDC_TEXT3, m_Text3);
    DDX_Control(pDX, IDC_TEXT2, m_Text2);
    DDX_Control(pDX, IDC_TEXT1, m_Text1);
    DDX_Control(pDX, IDC_RADIO40, m_Radio40);
    DDX_Control(pDX, IDC_RADIO80, m_Radio80);
    DDX_Control(pDX, IDC_RADIO120, m_Radio120);
    DDX_Control(pDX, IDC_BOXMESSAGE, m_BoxMessage);
    DDX_Control(pDX, IDC_MEASURE, m_Measure);
    DDX_Control(pDX, IDOK, m_OK);
    DDX_Control(pDX, IDC_BOXMAIN, m_BoxMain);
    DDX_Control(pDX, IDC_BOXPIC2, m_BoxPic2);
    DDX_Control(pDX, IDC_BOXPIC1, m_BoxPic1);
    //}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CWatermelonDlg, CDialog)
    //{{AFX_MSG_MAP(CWatermelonDlg)
    ON_WM_PAINT()
    ON_WM_QUERYDRAGICON()
    ON_WM_SIZE()
    ON_BN_CLICKED(IDC_MEASURE, OnMeasure)
    ON_WM_LBUTTONDOWN()
    ON_WM_MOUSEMOVE()
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CWatermelonDlg message handlers
```

```
BOOL CWatermelonDlg::OnInitDialog()
{
    CDialog::OnInitDialog();

    // 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
    if(!LoadImage(1,"image1.bmp"))return FALSE;
    if(!LoadImage(2,"image2.bmp"))return FALSE;
    m_Radio40.SetCheck(1);
    DisplayMessage(1);
    SendMessage(WM_SYSCOMMAND, SC_MAXIMIZE);
    return TRUE;  // return TRUE  unless you set the focus to a control
}

// 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 CWatermelonDlg::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
    {
        DrawBoarder();
        if(!m_Image1.IsDataNull())
        {
            Refresh(1);

        }
        if(!m_Image2.IsDataNull())
        {
            Refresh(2);

        }

        CDialog::OnPaint();
    }
}
```

```
// The system calls this to obtain the cursor to display while the user drag
//  the minimized window.
HCURSOR CWatermelonDlg::OnQueryDragIcon()
{
    return (HCURSOR) m_hIcon;
}

void CWatermelonDlg::OnSize(UINT nType, int cx, int cy)
{
    CDialog::OnSize(nType, cx, cy);
    DrawBoarder();
    if(!m_Image1.IsDataNull())
    {
        Refresh(1);
    }
    if(!m_Image2.IsDataNull())
    {
        Refresh(2);
    }
}

void CWatermelonDlg::OnOK()
{
    // TODO: Add extra validation here

    CDialog::OnOK();
}

////////////////////////////////////////////////////////////////////////////
int CWatermelonDlg::LoadImage(unsigned char sw, LPCTSTR lpszFileName)
{
    switch(sw){
    case 1:if(!m_Image1.Load (lpszFileName))return FALSE;
        break;
    case 2:if(!m_Image2.Load (lpszFileName))return FALSE;
        break;
    }
    return TRUE;  // return TRUE  unless you set the focus to a control
}

void CWatermelonDlg::Refresh(unsigned char sw)
{
    CClientDC hDC(this);
    CRect rcDIB,rcDest;
    rcDIB.top = rcDIB.left = 0;
    switch(sw){
    case 1:
        rcDIB.right =m_Image1.GetWidth ();
        rcDIB.bottom = m_Image1.GetHeight();
        GetClientRect(rcDest);
        rcDest.left  +=10;
        rcDest.right -=rcDest.right/2+10;
        rcDest.top   +=40;
        rcDest.bottom-=rcDest.bottom/2-10;
        m_Image1.Draw (hDC,&rcDIB,&rcDest);
        mulx=(float)(rcDIB.right/ (float)(rcDest.right-rcDest.left));
        muly=(float)(rcDIB.bottom/(float)(rcDest.bottom-rcDest.top));
        break;
    case 2:
```

```
            rcDIB.right =m_Image2.GetWidth ();
            rcDIB.bottom = m_Image2.GetHeight();
            GetClientRect(rcDest);
            rcDest.left  +=rcDest.right/2+10;
            rcDest.right -=10;
            rcDest.top   +=40;
            rcDest.bottom-=rcDest.bottom/2-10;
            m_Image2.Draw (hDC,&rcDIB,&rcDest);
            break;
        }
}

void CWatermelonDlg::DrawBoarder()
{
    CRect rcDest;
    GetClientRect(rcDest);
    if(m_BoxMain.GetSafeHwnd ()!=NULL)
        m_BoxMain.SetWindowPos
            (NULL,rcDest.left,rcDest.top,rcDest.right,rcDest.bottom,NULL);
    if(m_BoxPic1.GetSafeHwnd ()!=NULL)
        m_BoxPic1.SetWindowPos
            (NULL,rcDest.left+6, rcDest.top+20,rcDest.right/2-11,rcDest.bottom/2-5,NULL);
    if(m_BoxPic2.GetSafeHwnd ()!=NULL)
        m_BoxPic2.SetWindowPos
            (NULL,rcDest.right/2+6,rcDest.top+20,rcDest.right/2-11,rcDest.bottom/2-5,NULL)
    if(m_OK.GetSafeHwnd ()!=NULL)
        m_OK.SetWindowPos
            (NULL,rcDest.right-(rcDest.right-rcDest .left)/13,
                rcDest.bottom- ((rcDest.bottom-rcDest.top)/17*2),
                (rcDest.right-rcDest .left)/14,
                (rcDest.bottom-rcDest.top)/18,NULL);
    if(m_Measure.GetSafeHwnd ()!=NULL)
        m_Measure.SetWindowPos
            (NULL,5,rcDest.bottom- ((rcDest.bottom-rcDest.top)/17*2),
                rcDest.right-((rcDest.right-rcDest .left)/14+10),
                (rcDest.bottom-rcDest.top)/18,NULL);
    if(m_BoxMessage.GetSafeHwnd ()!=NULL)
        m_BoxMessage.SetWindowPos
            (NULL,5,rcDest.bottom- ((rcDest.bottom-rcDest.top)/17),
                rcDest.right-10,(rcDest.bottom-rcDest.top)/18-5,NULL);
    if(m_X.GetSafeHwnd ()!=NULL)
        m_X.SetWindowPos
            (NULL,rcDest.right-((rcDest.right-rcDest .left)/14*8),
            rcDest.bottom- ((rcDest.bottom-rcDest.top)/40*20),
            ((rcDest.right-rcDest .left)/14*2),
            (rcDest.bottom-rcDest.top)/15,NULL);
    if(m_Y.GetSafeHwnd ()!=NULL)
        m_Y.SetWindowPos
            (NULL,rcDest.right-((rcDest.right-rcDest .left)/14*8),
            rcDest.bottom- ((rcDest.bottom-rcDest.top)/40*17),
            ((rcDest.right-rcDest .left)/14*2),
            (rcDest.bottom-rcDest.top)/15,NULL);
    if(m_Z.GetSafeHwnd ()!=NULL)
        m_Z.SetWindowPos
            (NULL,rcDest.right-((rcDest.right-rcDest .left)/14*8),
            rcDest.bottom- ((rcDest.bottom-rcDest.top)/40*14),
            ((rcDest.right-rcDest .left)/14*2),
            (rcDest.bottom-rcDest.top)/15,NULL);
    if(m_RU.GetSafeHwnd ()!=NULL)
```

```
    m_RU.SetWindowPos
        (NULL,rcDest.right-((rcDest.right-rcDest .left)/14*7),
        rcDest.bottom- ((rcDest.bottom-rcDest.top)/40*11),
        ((rcDest.right-rcDest .left)/14*2),
        (rcDest.bottom-rcDest.top)/15,NULL);
f(m_RV.GetSafeHwnd ()!=NULL)
    m_RV.SetWindowPos
        (NULL,rcDest.right-((rcDest.right-rcDest .left)/14*7),
        rcDest.bottom- ((rcDest.bottom-rcDest.top)/40*8),
        ((rcDest.right-rcDest .left)/14*2),
        (rcDest.bottom-rcDest.top)/15,NULL);
f(m_LU.GetSafeHwnd ()!=NULL)
    m_LU.SetWindowPos
        (NULL,rcDest.right-((rcDest.right-rcDest .left)/14*9),
        rcDest.bottom- ((rcDest.bottom-rcDest.top)/40*11),
        ((rcDest.right-rcDest .left)/14*2),
        (rcDest.bottom-rcDest.top)/15,NULL);
f(m_LV.GetSafeHwnd ()!=NULL)
    m_LV.SetWindowPos
        (NULL,rcDest.right-((rcDest.right-rcDest .left)/14*9),
        rcDest.bottom- ((rcDest.bottom-rcDest.top)/40*8),
        ((rcDest.right-rcDest .left)/14*2),
        (rcDest.bottom-rcDest.top)/15,NULL);
f(m_TextX.GetSafeHwnd ()!=NULL)
    m_TextX.SetWindowPos
        (NULL,rcDest.right-((rcDest.right-rcDest .left)/14*8-5),
        rcDest.bottom- ((rcDest.bottom-rcDest.top)/40*20-15),
        ((rcDest.right-rcDest .left)/14*2-10),
        (rcDest.bottom-rcDest.top)/17-13,NULL);
f(m_TextY.GetSafeHwnd ()!=NULL)
    m_TextY.SetWindowPos
        (NULL,rcDest.right-((rcDest.right-rcDest .left)/14*8-5),
        rcDest.bottom- ((rcDest.bottom-rcDest.top)/40*17-15),
        ((rcDest.right-rcDest .left)/14*2-10),
        (rcDest.bottom-rcDest.top)/17-13,NULL);
f(m_TextZ.GetSafeHwnd ()!=NULL)
    m_TextZ.SetWindowPos
        (NULL,rcDest.right-((rcDest.right-rcDest .left)/14*8-5),
        rcDest.bottom- ((rcDest.bottom-rcDest.top)/40*14-15),
        ((rcDest.right-rcDest .left)/14*2-10),
        (rcDest.bottom-rcDest.top)/17-13,NULL);
f(m_TextRU.GetSafeHwnd ()!=NULL)
    m_TextRU.SetWindowPos
        (NULL,rcDest.right-((rcDest.right-rcDest .left)/14*7-5),
        rcDest.bottom- ((rcDest.bottom-rcDest.top)/40*11-15),
        ((rcDest.right-rcDest .left)/14*2-10),
        (rcDest.bottom-rcDest.top)/17-13,NULL);
f(m_TextRV.GetSafeHwnd ()!=NULL)
    m_TextRV.SetWindowPos
        (NULL,rcDest.right-((rcDest.right-rcDest .left)/14*7-5),
        rcDest.bottom- ((rcDest.bottom-rcDest.top)/40*8-15),
        ((rcDest.right-rcDest .left)/14*2-10),
        (rcDest.bottom-rcDest.top)/17-13,NULL);
if(m_TextLU.GetSafeHwnd ()!=NULL)
    m_TextLU.SetWindowPos
        (NULL,rcDest.right-((rcDest.right-rcDest .left)/14*9-5),
        rcDest.bottom- ((rcDest.bottom-rcDest.top)/40*11-15),
        ((rcDest.right-rcDest .left)/14*2-10),
        (rcDest.bottom-rcDest.top)/17-13,NULL);
```

```
    if(m_TextLV.GetSafeHwnd ()!=NULL)
        m_TextLV.SetWindowPos
            (NULL,rcDest.right-((rcDest.right-rcDest .left)/14*9-5),
            rcDest.bottom- ((rcDest.bottom-rcDest.top)/40*8-15),
            ((rcDest.right-rcDest .left)/14*2-10),
            (rcDest.bottom-rcDest.top)/17-13,NULL);
    if(m_Step.GetSafeHwnd ()!=NULL)
        m_Step.SetWindowPos
            (NULL,rcDest.right-((rcDest.right-rcDest .left)/14*2),
            rcDest.bottom- ((rcDest.bottom-rcDest.top)/40*20-10),
            ((rcDest.right-rcDest .left)/14*2-10),
            (rcDest.bottom-rcDest.top)/17*6-13,NULL);
    if(m_Text1.GetSafeHwnd ()!=NULL)
        m_Text1.SetWindowPos
            (NULL,rcDest.right-((rcDest.right-rcDest .left)/14*1+10),
            rcDest.bottom- ((rcDest.bottom-rcDest.top)/40*17),
            ((rcDest.right-rcDest .left)/14*1-15),
            (rcDest.bottom-rcDest.top)/17-13,NULL);
    if(m_Text2.GetSafeHwnd ()!=NULL)
        m_Text2.SetWindowPos
            (NULL,rcDest.right-((rcDest.right-rcDest .left)/14*1+10),
            rcDest.bottom- ((rcDest.bottom-rcDest.top)/40*14),
            ((rcDest.right-rcDest .left)/14*1-15),
            (rcDest.bottom-rcDest.top)/17-13,NULL);
    if(m_Text3.GetSafeHwnd ()!=NULL)
        m_Text3.SetWindowPos
            (NULL,rcDest.right-((rcDest.right-rcDest .left)/14*1+10),
            rcDest.bottom- ((rcDest.bottom-rcDest.top)/40*11),
            ((rcDest.right-rcDest .left)/14*1-15),
            (rcDest.bottom-rcDest.top)/17-13,NULL);
    if(m_Radio40.GetSafeHwnd ()!=NULL)
        m_Radio40.SetWindowPos
            (NULL,rcDest.right-((rcDest.right-rcDest .left)/14*1+30),
            rcDest.bottom- ((rcDest.bottom-rcDest.top)/40*18-10),
            30,
            (rcDest.bottom-rcDest.top)/17-13,NULL);
    if(m_Radio80.GetSafeHwnd ()!=NULL)
        m_Radio80.SetWindowPos
            (NULL,rcDest.right-((rcDest.right-rcDest .left)/14*1+30),
            rcDest.bottom- ((rcDest.bottom-rcDest.top)/40*15-10),
            30,
            (rcDest.bottom-rcDest.top)/17-13,NULL);
    if(m_Radio120.GetSafeHwnd ()!=NULL)
        m_Radio120.SetWindowPos
            (NULL,rcDest.right-((rcDest.right-rcDest .left)/14*1+30),
            rcDest.bottom- ((rcDest.bottom-rcDest.top)/40*12-10),
            30,
            (rcDest.bottom-rcDest.top)/17-13,NULL);
}

//////////////////////
void CWatermelonDlg::Sharp()
{
    m_Image2.PrepareUndo ();
    CImageFiltering image(m_Image2);
    BeginWaitCursor();
    image.Sharpening();
    Refresh(2);
    EndWaitCursor();
```

```
}

void CWatermelonDlg::Edge()
{
    m_Image2.PrepareUndo ();
    CImageFiltering image(m_Image2);
    BeginWaitCursor();
    image.Laplacian();
    Refresh(2);
    EndWaitCursor();
}

void CWatermelonDlg::Embos()
{
    m_Image2.PrepareUndo ();
    CImageFiltering image(m_Image2);
    BeginWaitCursor();
    image.Embossing();
    Refresh(2);
    EndWaitCursor();
}

void CWatermelonDlg::Threshold(bool sw, CRect SRect)
{
    m_Image1.PrepareUndo ();
    CImageProcess image1(m_Image1);
    m_Image2.PrepareUndo ();
    CImageProcess image2(m_Image2);
    BeginWaitCursor();
    switch(sw)
    {
    case 0:   image2.Threshold(SRect,CutValue);
              Refresh(2);
              break;
    case 1: image1.Threshold(SRect,CutValue);
              Refresh(1);
              break;
    }
    EndWaitCursor();
}

void CWatermelonDlg::Thinning(bool sw,CRect SRect)
{
    CImageProcess image1(m_Image1);
    CImageProcess image2(m_Image2);
    BeginWaitCursor();
    switch(sw)
    {
    case 0:   image2.Thinning(SRect);
              Refresh(2);
              break;
    case 1: image1.Thinning(SRect);
              Refresh(1);
              break;
    }
    EndWaitCursor();
}

void CWatermelonDlg::CalUV(bool sw,CRect SRect)
```

```
{
    CImageProcess image1(m_Image1);
    CImageProcess image2(m_Image2);
    BeginWaitCursor();
    switch(sw)
    {
    case 0:    image2.CalUV(SRect,CenterPointR);
//             Refresh(2);
               break;
    case 1: image1.CalUV(SRect,CenterPointL);
//             Refresh(1);
               break;
    }
    EndWaitCursor();
}

void CWatermelonDlg::ConvertGray(bool sw, CRect SRect)
{
    m_Image1.PrepareUndo ();
    CImageProcess image1(m_Image1);
    m_Image2.PrepareUndo ();
    CImageProcess image2(m_Image2);
    BeginWaitCursor();
    switch(sw)
    {
    case 0:    image2.ConvertGray(SRect);
               Refresh(2);
               break;
    case 1: image1.ConvertGray(SRect);
               Refresh(1);
               break;
    }
    EndWaitCursor();
}
void CWatermelonDlg::CalHisto(bool sw)
{
    int i;
    char buf[100];
    CImageProcess image1(m_Image1);
    CImageProcess image2(m_Image2);
    BeginWaitCursor();
    switch(sw)
    {
    case 0: image2.CalHisto(Histo);
            break;
    case 1: image1.CalHisto(Histo);
            break;
    }
    for(i=0;i<256;i++)
    {
        if(Histo[i]>=8)
        {
            CutValue=i;
            sprintf(buf,"Cutting value=%3d",CutValue);
            m_BoxMessage.SetWindowText(buf);
            break;
        }
    }
```

```
    EndWaitCursor();
}

void CWatermelonDlg::CalXYZ(double P1[3][4],double P2[3][4],int point1[2],int point2[2],double X[3][1])
{
    double A[4][3],C[4][1];

    A[0][0]=P1[0][0]-P1[2][0]*(double)point1[0];
    A[0][1]=P1[0][1]-P1[2][1]*(double)point1[0];
    A[0][2]=P1[0][2]-P1[2][2]*(double)point1[0];

    A[1][0]=P1[1][0]-P1[2][0]*(double)point1[1];
    A[1][1]=P1[1][1]-P1[2][1]*(double)point1[1];
    A[1][2]=P1[1][2]-P1[2][2]*(double)point1[1];

    A[2][0]=P2[0][0]-P2[2][0]*(double)point2[0];
    A[2][1]=P2[0][1]-P2[2][1]*(double)point2[0];
    A[2][2]=P2[0][2]-P2[2][2]*(double)point2[0];

    A[3][0]=P2[1][0]-P2[2][0]*(double)point2[1];
    A[3][1]=P2[1][1]-P2[2][1]*(double)point2[1];
    A[3][2]=P2[1][2]-P2[2][2]*(double)point2[1];

    C[0][0]=(double)point1[0]-P1[0][3];
    C[1][0]=(double)point1[1]-P1[1][3];
    C[2][0]=(double)point2[0]-P2[0][3];
    C[3][0]=(double)point2[1]-P2[1][3];

    m_CalMath.MatrixPseudo(A,C,X);
}

void CWatermelonDlg::DrawHisto()
{
    CPen pen;
    int i;
    pen.CreatePen(PS_SOLID,1,RGB(127,127,127));
    CClientDC dc(this);
    CPen *pOldPen = (CPen *)dc.SelectObject(&pen);
    for(i=0;i<256;i++)
    {
        dc.MoveTo (10+i,600);
        dc.LineTo (10+i,600-Histo[i]);
    }
    dc.MoveTo(10,600);
    dc.LineTo(266,600);
    for(i=0;i<26;i++)
    {
        dc.MoveTo(10+i*10,600);
        dc.LineTo(10+i*10,605);
    }
    dc.MoveTo(10+CutValue,600);
    dc.LineTo(10+CutValue,500);
    dc.SelectObject(pOldPen);
}
//////////////////////


void CWatermelonDlg::OnMouseMove(UINT nFlags, CPoint point)
{
```

```
    // TODO: Add your message handler code here and/or call default
    char buf[200];
    CRect rcDest1,rcDest2;

    GetClientRect(rcDest1);
    rcDest1.left   +=10;  rcDest1.right -=rcDest1.right/2+10;
    rcDest1.top    +=40; rcDest1.bottom-=rcDest1.bottom/2-10;

    GetClientRect(rcDest2);
    rcDest2.left   +=rcDest2.right/2+10; rcDest2.right -=10;
    rcDest2.top    +=40; rcDest2.bottom-=rcDest2.bottom/2-10;

    if( point.x>=rcDest1.left && point.x<=rcDest1.right &&
        point.y>=rcDest1.top  && point.y<=rcDest1.bottom)
    {
        MeasureUV(0,point);
        sprintf(buf,"좌측 실제 좌표(x:%3d, y:%3d)",RealUV.x,RealUV.y);
        m_BoxMessage.SetWindowText(buf);
        FocusIndex=0;
    }
    else if( point.x>=rcDest2.left && point.x<=rcDest2.right &&
        point.y>=rcDest2.top  && point.y<=rcDest2.bottom)
    {
        MeasureUV(1,point);
        sprintf(buf,"우측 실제 좌표(x:%3d, y:%3d)",RealUV.x,RealUV.y);
        m_BoxMessage.SetWindowText(buf);
        hCursor=AfxGetApp()->LoadStandardCursor(IDC_CROSS);
        SetCursor(hCursor);
        FocusIndex=1;
    }
    else
    {
        DisplayMessage(1);
        FocusIndex=0;
    }
    CDialog::OnMouseMove(nFlags, point);
}

void CWatermelonDlg::OnMeasure()
{
    // TODO: Add your control notification handler code here
    char buf[20];
    int CenterL[2],CenterR[2];
    double Position[3][1];
    double P1[3][4]={{-13.2050,  0.2240 ,  6.0584 ,  329.09656}
                    ,{-0.2238 , -13.5145,  4.5524 ,  239.0965 }
                    ,{-0.0005 ,  0.0006 ,  0.0190 ,  1.0}};
    double P2[3][4]={{20.9129 , -1.2480 , -10.3199, -506.8957}
                    ,{0.8508  ,  19.7546, -7.9804 ,  395.4855}
                    ,{0.0025  , -0.0040 , -0.0326 ,  1.0}};
    double P3[3][4]={{-33.6446,  0.3939 ,  16.5154,  3182.9530}
                    ,{ 0.1757 , -35.8541,  12.6212,  203.6020 }
                    ,{-0.0018 ,  0.0009 ,  0.0524 ,  1.0}};
    double P4[3][4]={{-46.8242, 1.7471  , 21.8443 , 5722.2133}
                    ,{-1.6293 , -45.5356, 16.7585 ,  63.8888}
                    ,{0.0054  , -0.0041 , -0.1688 ,  1.0}};

    MoveLength=0;
    ConvertGray(0,ScanRect);
```

```
CalHisto(0);
Threshold(0,ScanRect);
DrawHisto();
Thinning(0,ScanRect);
CalUV(0,ScanRect);
sprintf(buf,"%3d(pixel)",CenterPointR[0]);
m_TextRU.SetWindowText(buf);
sprintf(buf,"%3d(pixel)",CenterPointR[1]);
m_TextRV.SetWindowText(buf);
if      (m_Radio40.GetCheck()==1)
{
    MoveLength=40;
    ScanRect.left  +=20;
    ScanRect.right+=50;
}
else if(m_Radio80.GetCheck()==1)
{
    MoveLength=80;
    ScanRect.left  +=45;
    ScanRect.right+=100;
}
else if(m_Radio120.GetCheck()==1)
{
    MoveLength=120;
    ScanRect.left  +=70;
    ScanRect.right+=140;
}
sprintf(buf,"MoveLength=%3d",MoveLength);
m_BoxMessage.SetWindowText(buf);
ConvertGray(1,ScanRect);
CalHisto(1);
Threshold(1,ScanRect);
Thinning(1,ScanRect);
CalUV(1,ScanRect);
sprintf(buf,"%3d(pixel)",CenterPointL[0]);
m_TextLU.SetWindowText(buf);
sprintf(buf,"%3d(pixel)",CenterPointL[1]);
m_TextLV.SetWindowText(buf);
CenterL[0]=CenterPointL[0]-320;
CenterL[1]=CenterPointL[1]-230;
CenterR[0]=CenterPointR[0]-320;
CenterR[1]=CenterPointR[1]-230;
if      (m_Radio40.GetCheck()==1)
{
    CalXYZ(P1,P2,CenterR,CenterL,Position);
}
else if(m_Radio80.GetCheck()==1)
{
    CalXYZ(P1,P3,CenterR,CenterL,Position);
}
else if(m_Radio120.GetCheck()==1)
{
    CalXYZ(P1,P4,CenterR,CenterL,Position);
}
sprintf(buf,"%8.3f(mm)",Position[0][0]);
m_TextX.SetWindowText(buf);
sprintf(buf,"%8.3f(mm)",Position[1][0]);
m_TextY.SetWindowText(buf);
sprintf(buf,"%8.3f(mm)",Position[2][0]);
```

```
    m_TextZ.SetWindowText(buf);
}



void CWatermelonDlg::DisplayMessage(unsigned char sw)
{
    switch(sw)
    {
    case 1: m_BoxMessage.SetWindowText("마우스로 좌측 영상에서 원하는 목표물을 선택하시오
        break;
    }
}

void CWatermelonDlg::OnLButtonDown(UINT nFlags, CPoint point)
{
    // TODO: Add your message handler code here and/or call default
    char buf[100];
    if(FocusIndex==1)
    {
        CPen pen;
        pen.CreatePen(PS_SOLID,1,RGB(255,0,0));
        CClientDC dc(this);
        CPen *pOldPen = (CPen *)dc.SelectObject(&pen);
        dc.MoveTo (point.x-10,point.y-10);
        dc.LineTo (point.x+10,point.y-10);
        dc.LineTo (point.x+10,point.y+10);
        dc.LineTo (point.x-10,point.y+10);
        dc.LineTo (point.x-10,point.y-10);
        dc.SelectObject(pOldPen);
        MeasureUV(1,point);
        ScanRect.left   =RealUV.x-20;
        ScanRect.top    =RealUV.y-20;
        ScanRect.right  =RealUV.x+20;
        ScanRect.bottom=RealUV.y+20;
    }
    CDialog::OnLButtonDown(nFlags, point);
}

void CWatermelonDlg::MeasureUV(BOOL sw,CPoint point)
{

    CRect rcDest;
    GetClientRect(rcDest);
    switch(sw)
    {
    case 0:   rcDest.left   +=10;   rcDest.right  -=rcDest.right /2+10;
              rcDest.top    +=40;  rcDest.bottom-=rcDest.bottom/2-10;
              break;
    case 1:   rcDest.left   +=rcDest.right/2+10; rcDest.right  -=10;
              rcDest.top    +=40;  rcDest.bottom-=rcDest.bottom/2-10;
              break;
    }
    RealUV.x=(LONG)((point.x-rcDest.left)*mulx);
    RealUV.y=(LONG)((point.y-rcDest.top) *muly);
}
```

다. 영상처리

```
#include <windows.h>
#include <windowsx.h>
#include "resource.h"
#include "bdpro.h"
#include "global.h"

// Global Variables
static     HWND hParenthWnd;                                    // parent application HWND
RECT CurCaptureRect;
static BD_VW_INPUT_CONTROL SaveInControl;      // Video Window Input for saving

// Function prototypes
static LRESULT SetAcqROI( RECT* rcAcqROI);
static BOOL DisplayPositionAcq(HWND hWnd, BD_VW_INPUT_CONTROL* InControl);
void UpdateTitleAcqScaling ( void);

////////////////////////////////////////////////////////////////////////////
// Name:    DlgAcqROI
//
// Declaration:
//
// BOOL CALLBACK DlgAcqROI( hwnd, message, wParam, lParam)
// HWND hwnd;
// UINT message;
// WPARAM wParam;
// LPARAM lParam;
//
// Input:
//         hwnd:                Identifies the Dialog Box that receives the message
//         message:          Specifies the Message Number
//         wParam:          Specifies 16 bits of additional message-dependent information
//         lParam:          Specifies 32 bits of additional message-dependent information
//
// Output:      NONE
//
// Return Value:
//
//       0 (FALSE) : Function did not Process Message
//       1 (TRUE)  : Function Processed Message
//
// Description:
//       This Callback Function interactively modifies the Acquisition
//       ROI.
//
////////////////////////////////////////////////////////////////////////////

#ifdef __BORLANDC__
   #pragma argsused
#endif


BOOL CALLBACK DlgAcqROI( HWND hwnd, UINT message, WPARAM wParam, LPARAM lParam)
   {
     LRESULT lResult;

   // Process Message
   switch (message)
      {
```

```
    case WM_INITDIALOG:      // Initialize Dialog Box

        // Get Video Input Control (to be modified)
        IResult = BD_GetVideoWindowInputControl( &BDBoard.InControl);
        if (IResult)
        {
            PrintBanditErrorMessage(hwnd, IResult, "ERROR after calling BD_GetVideoWindowInputControl() , file \"bdacqroi.c\"")
            return FALSE;
        }

        SaveInControl =  BDBoard.InControl; // save video input control

        // Acquisition dialog box is available only when not fit-to-screen is available
        EnableMenuItem(BDBoard.hMenu,    ID_OPTIONS_DISPLAY_FITTOSCREEN,    MF_GRAYED    |    MF_DISABLED
MF_BYCOMMAND);
        EnableMenuItem(BDBoard.hMenu, ID_ACQUISITION_CAMERA, MF_GRAYED | MF_DISABLED | MF_BYCOMMAND);

        UpdateTitleAcqRoi( hwnd, BDBoard.InControl.Destination.x, BDBoard.InControl.Destination.y
                                ,BDBoard.InControl.AcqRoi.xlen, BDBoard.InControl.AcqRoi.ylen);

    break;

    case WM_CLOSE:              // Destroy Modeless Dialog Box
      if (BDBoard.hwndAcqROIDialog)
        {
            DestroyWindow( hwnd);
            BDBoard.hwndAcqROIDialog = 0;
        }
        EnableMenuItem(BDBoard.hMenu, ID_OPTIONS_DISPLAY_FITTOSCREEN, MF_ENABLED | MF_BYCOMMAND);
        EnableMenuItem(BDBoard.hMenu, ID_OPTIONS_DISPLAY_ZOOM, MF_ENABLED | MF_BYCOMMAND);
        EnableMenuItem(BDBoard.hMenu, ID_ACQUISITION_CAMERA, MF_ENABLED | MF_BYCOMMAND);

    break;

    case WM_COMMAND:
      switch (GET_WM_COMMAND_ID(wParam, lParam))
        {

            // Define the ACQ ROI
        case IDC_ACQ_ROI_DEFINE:
          {
                RECT rc;

                ShowWindow( BDBoard.hwndAcqROIDialog, SW_RESTORE);
                EnableWindow(GetDlgItem(hwnd, IDC_ACQ_ROI_SAVE), FALSE);
                EnableWindow(GetDlgItem(hwnd, IDC_ACQ_ROI_DEFAULT), FALSE);
                EnableWindow(GetDlgItem(hwnd, IDC_ACQ_ROI_CLOSE), FALSE);
                EnableWindow(GetDlgItem(hwnd, IDC_ACQ_ROI_CANCEL), FALSE);
                if (GetCaptureRect(hParenthWnd, hwnd, &rc))  // window coordinate
                {
                    CurCaptureRect = rc;
                    SetAcqROI(&CurCaptureRect);
                }

                EnableWindow(GetDlgItem(hwnd, IDC_ACQ_ROI_SAVE), TRUE);
                EnableWindow(GetDlgItem(hwnd, IDC_ACQ_ROI_DEFAULT), TRUE);
                EnableWindow(GetDlgItem(hwnd, IDC_ACQ_ROI_CLOSE), TRUE);
                EnableWindow(GetDlgItem(hwnd, IDC_ACQ_ROI_CANCEL), TRUE);
                EnableWindow(GetDlgItem(hwnd, IDC_ACQ_ROI_MOVE), TRUE);
```

```
                    DrawMenuBar(hParenthWnd);
                    UpdateScreen(hParenthWnd, &rc);
            }

                UpdateTitleBar();
                UpdateTitleAcqScaling();
                SetFocus(hwnd); // Acquisition Dialog Box has focus
            break;

                // Set to default
        case IDC_ACQ_ROI_DEFAULT:
                // disable Acquisition Default
                EnableWindow(GetDlgItem(hwnd, IDC_ACQ_ROI_DEFAULT), FALSE);
                SetAcqRoiToDefault(hwnd);
                EnableWindow(GetDlgItem(hwnd, IDC_ACQ_ROI_SAVE), TRUE);
                UpdateTitleAcqScaling();
                UpdateTitleBar();
            break;

                // Save the current acquisition ROI
        case IDC_ACQ_ROI_SAVE:
                SaveAcquisition();
            break;

                case IDC_ACQ_ROI_CANCEL:
                    {
                    BOOL bGrab;

                    bGrab = BDBoard.bGrabOn;
                    if (bGrab)
                        StopGrab(BDBoard.hMenu);

                    BDBoard.InControl = SaveInControl;
                    BDBoard.InControl.AcqScale.x = (BDBoard.ScalingPercentage * BDBoard.InControl.AcqRoi.xlen)/100;
                    BDBoard.InControl.AcqScale.y = (BDBoard.ScalingPercentage * BDBoard.InControl.AcqRoi.ylen)/100;
                    BDBoard.InControl.VideoRoi.xlen = BDBoard.InControl.VideoScale.x = BDBoard.InControl.AcqScale.x;
                    BDBoard.InControl.VideoRoi.ylen = BDBoard.InControl.VideoScale.y = BDBoard.InControl.AcqScale.y;

                    lResult = _BD_SetVideoWindowInputControl( &BDBoard.InControl);
                    if (lResult)
                    {
                        PrintBanditErrorMessage(hwnd, lResult, "ERROR after calling _BD_SetVideoWindowInputControl(), file
\"bdacqroi.c\"");
                    }
                    // Close the modeless dialog box, fall through no "break;" statement

                    if (bGrab)
                        StartGrab(BDBoard.hMenu);
                    }
                    // Destroy Modeless Dialog Box
        case IDC_ACQ_ROI_CLOSE:    // OK
                if (BDBoard.hwndAcqROIDialog)
                  {
                    DestroyWindow( hwnd);
                    BDBoard.hwndAcqROIDialog = 0;

                        EnableMenuItem(BDBoard.hMenu,        ID_OPTIONS_DISPLAY_FITTOSCREEN,        MF_ENABLED
MF_BYCOMMAND);
                        EnableMenuItem(BDBoard.hMenu, ID_OPTIONS_DISPLAY_ZOOM, MF_ENABLED | MF_BYCOMMAND);
```

```
                }
                    UpdateTitleBar();
                    UpdateTitleAcqScaling();

            break;

                case IDC_ACQ_ROI_MOVE:

                    // have the user get the impression that the button is chosen
                    EnableWindow(GetDlgItem(hwnd, IDC_ACQ_ROI_DEFINE), FALSE);
                    EnableWindow(GetDlgItem(hwnd, IDC_ACQ_ROI_SAVE), FALSE);
                    EnableWindow(GetDlgItem(hwnd, IDC_ACQ_ROI_DEFAULT), FALSE);
                    EnableWindow(GetDlgItem(hwnd, IDC_ACQ_ROI_CLOSE), FALSE);
                    EnableWindow(GetDlgItem(hwnd, IDC_ACQ_ROI_CANCEL), FALSE);

                    DisplayPositionAcq(hParenthWnd, &BDBoard.InControl);

                    EnableWindow(GetDlgItem(hwnd, IDC_ACQ_ROI_DEFINE), TRUE);
                    EnableWindow(GetDlgItem(hwnd, IDC_ACQ_ROI_SAVE), TRUE);
                    EnableWindow(GetDlgItem(hwnd, IDC_ACQ_ROI_DEFAULT), TRUE);
                    EnableWindow(GetDlgItem(hwnd, IDC_ACQ_ROI_CLOSE), TRUE);
                    EnableWindow(GetDlgItem(hwnd, IDC_ACQ_ROI_CANCEL), TRUE);
                    SetFocus(hwnd); // Acquisition ROI gets focus
                    break;
          }
            ClipCursor(NULL);
        break;

      default:
        return( FALSE);
      }
   return( TRUE);
   }


////////////////////////////////////////////////////////////////////////////
// CmdAcqROI
//
//   Description:
//        Returns a pointer to a window handle of the ROI dialog, used to verify
//   if the dialog is still active
//
//   Input:
//        hwnd:                    handle of the window
//
//   Output:
//        CmdAcqROI          Pointer to a window handle
//
//
//   Returns:
//         Pointer to a window handle
//
////////////////////////////////////////////////////////////////////////////


HWND *CmdAcqROI(HWND hwnd)
   {
   if( BDBoard.hwndAcqROIDialog)
      {
```

```
        // if the dialog box is already open, then set focus to that dialog box
        SetFocus(BDBoard.hwndAcqROIDialog);
      return NULL;
      }
   else
      {
      // Create Modeless Dialog Box
         hParenthWnd = hwnd; //store it
      BDBoard.hwndAcqROIDialog = CreateDialogParam( GetWindowInstance(hwnd), MAKEINTRESOURCE(IDD_ACQROI), hParenthWnd
(DLGPROC) DlgAcqROI, 0L);
      return &(BDBoard.hwndAcqROIDialog);
      }
   }

//////////////////////////////////////////////////////////////////////////
// SetAcqROI
//
//    Description:
//         Set Acquisition Video Input Control according to the size of the rectangle defined
//         by user (rectangle rcAcqROI)
//
//    Input:
//         rcAcqROI              Pointer to ROI rectangle
//
//
//    Returns:
//         BD_OK is successful, otherwise error code
//
//////////////////////////////////////////////////////////////////////////

LRESULT SetAcqROI( RECT* rcAcqROI)
{
    LRESULT lResult;
    BOOL bGrab;
    BD_SIZE CamSize;

    lResult = BD_GetAcqInputStandardSize( &CamSize);

    if (lResult)
        return FALSE;

    // It is necessary to stop the grab, otherwise, it will leave some unwanted traces
    // image on the previous ACQ ROI
    bGrab = BDBoard.bGrabOn;
    if (bGrab)
        StopGrab(BDBoard.hMenu);

    // Setting Video Input Control when in normal mode, not when Horizontal Flipping
    if (BDBoard.CurHorizontalFlip)
    {
        SetHorizontalFlip( !BDBoard.CurHorizontalFlip);
        BDBoard.InControl.AcqRoi.x = BDBoard.InControl.Destination.x =
            BDBoard.VW_Size.x   -   (rcAcqROI->left   +   BDBoard.xCurrentScroll*BDBoard.wZoomFactor)/BDBoard.wZoomFactor
(rcAcqROI->right-rcAcqROI->left);
    }
    else
        BDBoard.InControl.AcqRoi.x = BDBoard.InControl.Destination.x =
            (rcAcqROI->left + BDBoard.xCurrentScroll*BDBoard.wZoomFactor)/BDBoard.wZoomFactor;
```

```
    BDBoard.InControl.AcqRoi.y = BDBoard.InControl.Destination.y = (rcAcqROI->top
BDBoard.yCurrentScroll*BDBoard.wZoomFactor)/BDBoard.wZoomFactor;

    BDBoard.InControl.AcqRoi.xlen = (rcAcqROI->right - rcAcqROI->left)/BDBoard.wZoomFactor;
    BDBoard.InControl.AcqRoi.ylen = (rcAcqROI->bottom - rcAcqROI->top)/BDBoard.wZoomFactor;
    BDBoard.InControl.AcqScale.x = (BDBoard.ScalingPercentage * BDBoard.InControl.AcqRoi.xlen)/100;
    BDBoard.InControl.AcqScale.y = (BDBoard.ScalingPercentage * BDBoard.InControl.AcqRoi.ylen)/100;
    BDBoard.InControl.VideoRoi.xlen = BDBoard.InControl.VideoScale.x = BDBoard.InControl.AcqScale.x;
    BDBoard.InControl.VideoRoi.ylen = BDBoard.InControl.VideoScale.y = BDBoard.InControl.AcqScale.y;

    lResult = _BD_SetVideoWindowInputControl( &BDBoard.InControl);

    // restore the setting of Horizontal Flip
    if (BDBoard.CurHorizontalFlip)
    {
        SetHorizontalFlip( BDBoard.CurHorizontalFlip);
    }

    if (lResult) // should never get here
        MessageBox(GetFocus(), "_BDSetVideoWindowInputControl", "ERROR BDACQROI.C", MB_ICONSTOP | MB_OK);

    // restore the grab if it is already ON
    if (bGrab)
        StartGrab(BDBoard.hMenu);

    UpdateTitleBar(); // due to VideoScale.x and VideoScale.y have been changed
    UpdateTitleAcqScaling();
    UpdateScreen(hParenthWnd, &CurCaptureRect);
    return lResult;
}


////////////////////////////////////////////////////////////////////////////
// SetAcqRoiToDefault
//
//   Description:
//        Set video input control to standard size
//
//   Output:
//        hWnd              application window handle
//
// Note:
//        function prototype is declared in bdpro.h (used also in bdpro.c)
//
//   Returns:
//        BD_OK if successful, error code otherwise
//
////////////////////////////////////////////////////////////////////////////

LRESULT SetAcqRoiToDefault(HWND hWnd)
{
    LRESULT lResult;

    lResult = BD_GetVideoWindowInputControl( &BDBoard.InControl);

    if (!lResult)
    {
        BDBoard.InControl.AcqRoi.x = BDBoard.InControl.Destination.x = 0;
        BDBoard.InControl.AcqRoi.y = BDBoard.InControl.Destination.y = 0;
```

```
        BDBoard.InControl.AcqRoi.xlen = BDBoard.VW_Size.x;
        BDBoard.InControl.AcqRoi.ylen = BDBoard.VW_Size.y;
        BDBoard.InControl.VideoRoi.xlen = BDBoard.InControl.VideoScale.x = BDBoard.VW_Size.x;
        BDBoard.InControl.VideoRoi.ylen = BDBoard.InControl.VideoScale.y = BDBoard.VW_Size.y;
        BDBoard.InControl.AcqScale.x = (BDBoard.ScalingPercentage * BDBoard.InControl.AcqRoi.xlen)/100;
        BDBoard.InControl.AcqScale.y = (BDBoard.ScalingPercentage * BDBoard.InControl.AcqRoi.ylen)/100;
        BDBoard.InControl.VideoRoi.xlen = BDBoard.InControl.VideoScale.x = BDBoard.InControl.AcqScale.x
        BDBoard.InControl.VideoRoi.ylen = BDBoard.InControl.VideoScale.y = BDBoard.InControl.AcqScale.y
    }

    if (!lResult)
        lResult = _BD_SetVideoWindowInputControl( &BDBoard.InControl);

    // Update Child Window text
    UpdateTitleAcqRoi(hWnd, BDBoard.InControl.Destination.x, BDBoard.InControl.Destination.y,
                              BDBoard.InControl.AcqRoi.xlen, BDBoard.InControl.AcqRoi.ylen);
    return lResult;
}


////////////////////////////////////////////////////////////////////////////
// SaveAcquisition
//
//   Description:
//          Save acquisition video input control to BDPRO.INI
//
//   Returns:
//       none
//
////////////////////////////////////////////////////////////////////////////

void SaveAcquisition()
{
    char str[_MAX_FNAME];
    char str2[20];

    BUILDSECTION(str);

    wsprintf(str2, "%d", BDBoard.InControl.AcqRoi.x);
    WritePrivateProfileString(str, DEF_IN_ACQROI_X, str2, InitFile);

    wsprintf(str2, "%d", BDBoard.InControl.AcqRoi.y);
    WritePrivateProfileString(str, DEF_IN_ACQROI_Y, str2, InitFile);

    wsprintf(str2, "%d", BDBoard.InControl.AcqRoi.xlen);
    WritePrivateProfileString(str, DEF_IN_ACQROI_XLEN, str2, InitFile);

    wsprintf(str2, "%d", BDBoard.InControl.AcqRoi.ylen);
    WritePrivateProfileString(str, DEF_IN_ACQROI_YLEN, str2, InitFile);

    wsprintf(str2, "%d", BDBoard.InControl.VideoRoi.x);
    WritePrivateProfileString(str, DEF_IN_VIDEOROI_X, str2, InitFile);

    wsprintf(str2, "%d", BDBoard.InControl.VideoRoi.y);
    WritePrivateProfileString(str, DEF_IN_VIDEOROI_Y, str2, InitFile);

    wsprintf(str2, "%d", BDBoard.InControl.Destination.x);
    WritePrivateProfileString(str, DEF_IN_DESTINATION_X, str2, InitFile);
```

```
    wsprintf(str2, "%d", BDBoard.InControl.Destination.y);
    WritePrivateProfileString(str, DEF_IN_DESTINATION_Y, str2, InitFile);

}

////////////////////////////////////////////////////////////////////////////
// DisplayPositionAcq
//
// Description:
//      Find the screen coordinate of Video Input Control Destination.x and
//      Destination.y
//
//  Parameters:
//          hwnd                    Handle of the parent window.
//          InControl:          pointer to Video Input Control
//
//  Note:
//      Destination.x and Destination.y will be validated and possibly changed according
//      to AcqRoi.x and AcqRoi.xlen (_BD_SetVideoInputControl() will modify)
//
//  Returns:
//      TRUE if successful, FALSE otherwise.
////////////////////////////////////////////////////////////////////////////

BOOL DisplayPositionAcq(HWND hWnd, BD_VW_INPUT_CONTROL* InControl)
{
    BOOL ok=TRUE;                   // boolean value to control the while loop
    MSG msg;                            // Windows message
  static RECT  rc;              // Application's Window Rectangle
    BOOL retVal;
    POINT ptBeg;
    POINT ptEnd;
    POINT   ptCursor;
    static BOOL bFirst=TRUE;  // boolean value to avoid repainting the screen for the first time
    HCURSOR hOldCursor;

    SetFocus(hWnd);

#if !_DEBUG
    SetCapture( hWnd);          // get focus of the mouse
#endif

    // Get the Application Window rectangle
    BD_GetClientRect( hWnd, &rc);
    ClientToScreen( hWnd, (POINT*) &rc.left);
    ClientToScreen( hWnd, (POINT*) &rc.right);

#if !_DEBUG
    ClipCursor(&rc);
#endif

    // Waiting for user to click left mouse button
    // That is the location where Acq ROI is placed
        while (ok)
        {
            // Get Client rectangle for repaint
            GetClientRect(hWnd, &rc);
            hOldCursor = SetCursor(LoadCursor(NULL, IDC_CROSS));
            GetMessage( &msg, NULL, 0, 0);
```

```
switch( msg.message)
{
    case WM_LBUTTONDOWN:
        {
            // Get the position where to put the image
            if (BDBoard.CurHorizontalFlip)
            {
                InControl->Destination.x = BDBoard.VW_Size.x - (LOWORD (msg.lParam)+
BDBoard.xCurrentScroll*BDBoard.wZoomFactor)/BDBoard.wZoomFactor - InControl->AcqRoi.xlen;
            }
            else
            {
                InControl->Destination.x = (LOWORD (msg.lParam)+
BDBoard.xCurrentScroll*BDBoard.wZoomFactor)/BDBoard.wZoomFactor;
            }
            InControl->Destination.y = (HIWORD (msg.lParam)+
BDBoard.yCurrentScroll*BDBoard.wZoomFactor)/BDBoard.wZoomFactor;
            ok = FALSE;
        }

        break;

    case WM_MOUSEMOVE:
        {
            // Do not update the screen for the first time
            if (!bFirst)
            {
            InvalidateRect( hWnd, &rc, TRUE);
            UpdateScreen( hWnd, &rc);
            }
            else
                bFirst=FALSE;

            ptBeg.x = LOWORD (msg.lParam);
            ptBeg.x = ptBeg.x - (ptBeg.x % 2); // make sure to be even to avoid distorted image
            ptBeg.y = HIWORD (msg.lParam);


            ptEnd.x = ptBeg.x + BDBoard.InControl.AcqRoi.xlen * BDBoard.wZoomFactor;
            ptEnd.y = ptBeg.y + BDBoard.InControl.AcqRoi.ylen * BDBoard.wZoomFactor;

            // autoscroll when the beginning point of the image is near the border of the client area
            if (ptBeg.x < (rc.left + MIN_PIXELS_TO_SCROLL))
                SendMessage(hWnd, WM_HSCROLL, SB_PAGEUP, 0L);

            if (ptBeg.y < (rc.top + MIN_PIXELS_TO_SCROLL))
                SendMessage(hWnd, WM_VSCROLL, SB_PAGEUP, 0L);

            if (ptBeg.x > (rc.right - MIN_PIXELS_TO_SCROLL))
                SendMessage(hWnd, WM_HSCROLL, SB_PAGEDOWN, 0L);

            if (ptBeg.y > (rc.bottom - MIN_PIXELS_TO_SCROLL))
                SendMessage(hWnd, WM_VSCROLL, SB_PAGEDOWN, 0L);

            DrawBoxOutline (hWnd, ptBeg, ptEnd, WHITE_PEN);
            // may have difference with VideoWindowInControl since Width needs to be a multiple of 4
            UpdateTitleAcqRoi( BDBoard.hwndAcqROIDialog, (ptBeg.x+BDBoard.xCurrentScroll)/BDBoard.wZoomFactor,
(ptBeg.y + BDBoard.yCurrentScroll)/BDBoard.wZoomFactor
                ,BDBoard.InControl.AcqRoi.xlen, BDBoard.InControl.AcqRoi.ylen);
```

```
            }
            break;

        // if Escape key is hit, the operation will be canceled
        case WM_KEYDOWN:
            switch (msg.wParam)
            {
                case VK_ESCAPE:
                    {
                        EnableWindow(GetDlgItem(hWnd, IDC_ACQ_ROI_SAVE), TRUE);
                        EnableWindow(GetDlgItem(hWnd, IDC_ACQ_ROI_DEFAULT), TRUE);
                        EnableWindow(GetDlgItem(hWnd, IDC_ACQ_ROI_CLOSE), TRUE);
                        EnableWindow(GetDlgItem(hWnd, IDC_ACQ_ROI_CANCEL), TRUE);
                        EnableWindow(GetDlgItem(hWnd, IDC_ACQ_ROI_MOVE), TRUE);
                        ok = FALSE; // quit the while loop
                    }
                    break;

                case VK_RETURN:
                    {
                        POINT pt;
                        GetCursorPos(&pt); // pt in screen coordinate
                        ScreenToClient(hWnd, &pt);
                        if (BDBoard.CurHorizontalFlip)
                        {
                            InControl->Destination.x = BDBoard.VW_Size.x - (LOWORD (msg.lParam)
BDBoard.xCurrentScroll*BDBoard.wZoomFactor)/BDBoard.wZoomFactor - InControl->AcqRoi.xlen;
                        }
                        else
                        {
                            InControl->Destination.x = (pt.x
BDBoard.xCurrentScroll*BDBoard.wZoomFactor)/BDBoard.wZoomFactor;
                        }
                        InControl->Destination.y = (pt.y
BDBoard.yCurrentScroll*BDBoard.wZoomFactor)/BDBoard.wZoomFactor;

                        ok = FALSE; // quit the while loop
                    }
                    break;

                case VK_RIGHT:
                    {
                    InvalidateRect( hWnd, &rc, TRUE);
                    UpdateScreen( hWnd, &rc);

                    ptBeg.x += GRANUL_X;
                    ptBeg.x = ptBeg.x - (ptBeg.x % 2); // make sure to be even to avoid distorted image

                    ptEnd.x = ptBeg.x + BDBoard.InControl.AcqRoi.xlen * BDBoard.wZoomFactor;
                    ptEnd.y = ptBeg.y + BDBoard.InControl.AcqRoi.ylen * BDBoard.wZoomFactor;

                    DrawBoxOutline (hWnd, ptBeg, ptEnd, WHITE_PEN);
                    UpdateTitleAcqRoi( BDBoard.hwndAcqROIDialog, (ptBeg.x
BDBoard.xCurrentScroll)/BDBoard.wZoomFactor, (ptBeg.y + BDBoard.yCurrentScroll)/BDBoard.wZoomFactor
                        ,BDBoard.InControl.AcqRoi.xlen, BDBoard.InControl.AcqRoi.ylen);
                    ptCursor = ptBeg;
                    ClientToScreen( hWnd, &ptCursor);
                    SetCursorPos( ptCursor.x, ptCursor.y);
                    }
```

```
                    break;

                    case VK_LEFT:
                        {
                        InvalidateRect( hWnd, &rc, TRUE);
                        UpdateScreen( hWnd, &rc);

                        ptBeg.x -= GRANUL_X;
                        ptBeg.x = ptBeg.x - (ptBeg.x % 2); // make sure to be even to avoid distorted image

                        ptEnd.x = ptBeg.x +  BDBoard.InControl.AcqRoi.xlen * BDBoard.wZoomFactor;
                        ptEnd.y = ptBeg.y +  BDBoard.InControl.AcqRoi.ylen * BDBoard.wZoomFactor;

                        DrawBoxOutline ( hWnd, ptBeg, ptEnd, WHITE_PEN);
                        UpdateTitleAcqRoi(              BDBoard.hwndAcqROIDialog,              (ptBeg.x
BDBoard.xCurrentScroll)/BDBoard.wZoomFactor, (ptBeg.y + BDBoard.yCurrentScroll)/BDBoard.wZoomFactor
                                                ,BDBoard.InControl.AcqRoi.xlen, BDBoard.InControl.AcqRoi.ylen);
                        ptCursor = ptBeg;
                        ClientToScreen( hWnd, &ptCursor);
                        SetCursorPos( ptCursor.x, ptCursor.y);
                        }
                    break;

                    case VK_UP:
                        {
                        InvalidateRect( hWnd, &rc, TRUE);
                        UpdateScreen( hWnd, &rc);

                        ptBeg.y -= GRANUL_X;
                        ptEnd.x = ptBeg.x +  BDBoard.InControl.AcqRoi.xlen * BDBoard.wZoomFactor;
                        ptEnd.y = ptBeg.y +  BDBoard.InControl.AcqRoi.ylen * BDBoard.wZoomFactor;

                        DrawBoxOutline (hWnd, ptBeg, ptEnd, WHITE_PEN);
                        UpdateTitleAcqRoi(              BDBoard.hwndAcqROIDialog,              ptBeg.x/BDBoard.wZoomFactor,
ptBeg.y/BDBoard.wZoomFactor
                                                ,BDBoard.InControl.AcqRoi.xlen, BDBoard.InControl.AcqRoi.ylen);
                        ptCursor = ptBeg;
                        ClientToScreen( hWnd, &ptCursor);
                        SetCursorPos( ptCursor.x, ptCursor.y);
                        }
                    break;

                    case VK_DOWN:
                        {
                        InvalidateRect( hWnd, &rc, TRUE);
                        UpdateScreen( hWnd, &rc);

                        ptBeg.y += GRANUL_X;
                        ptEnd.x = ptBeg.x +  BDBoard.InControl.AcqRoi.xlen * BDBoard.wZoomFactor;
                        ptEnd.y = ptBeg.y +  BDBoard.InControl.AcqRoi.ylen * BDBoard.wZoomFactor;

                        DrawBoxOutline (hWnd, ptBeg, ptEnd, WHITE_PEN);
                        UpdateTitleAcqRoi(              BDBoard.hwndAcqROIDialog,              (ptBeg.x
BDBoard.xCurrentScroll)/BDBoard.wZoomFactor, (ptBeg.y + BDBoard.yCurrentScroll)/BDBoard.wZoomFactor
                                                ,BDBoard.InControl.AcqRoi.xlen, BDBoard.InControl.AcqRoi.ylen);
                        ptCursor = ptBeg;
                        ClientToScreen( hWnd, &ptCursor);
                        SetCursorPos( ptCursor.x, ptCursor.y);
                        }
```

```
                    break;


                    default:
                        break;

                }
            } // end of switch
        } // end of while loop


        ClipCursor(NULL);
        SetCursor(LoadCursor(NULL, hOldCursor));
        ReleaseCapture(); // release the focus of the mouse

        InControl->AcqScale.x = (BDBoard.ScalingPercentage * InControl->AcqRoi.xlen)/100
        InControl->AcqScale.y = (BDBoard.ScalingPercentage * InControl->AcqRoi.ylen)/100
        InControl->VideoRoi.xlen = InControl->VideoScale.x = InControl->AcqScale.x;
        InControl->VideoRoi.ylen = InControl->VideoScale.y = InControl->AcqScale.y;


        if (BDBoard.CurHorizontalFlip)
        {
            SetHorizontalFlip( !BDBoard.CurHorizontalFlip);
        }

        // Set Video Input Control when it is in Normal Mode (not in Horizontal Flip)
        retVal = (_BD_SetVideoWindowInputControl( InControl) == BD_OK);

        // make sure that it is current Horizontal Flip
        if (BDBoard.CurHorizontalFlip)
        {
            SetHorizontalFlip( BDBoard.CurHorizontalFlip);
        }

        // erase last rectangle drawn and update the screen
        GetClientRect(hWnd, &rc); // get full rectangle including status bar
        UpdateScreen(hWnd, &rc);
    return retVal;
}

#ifdef __BORLANDC__
#pragma warn -par
#endif

//////////////////////////////////////////////////////////////////////////////
// UpdateTitleAcqRoi
//
//  Description:
//          Update title of the Acquisition Scaling dialog box
//
//      hwnd:                   Acquisition Roi Child window handle
//      DestinationX:  Acquisition Video ROI Destination X coordinate
//      DestinationY:  Acquisition Video ROI Destination Y coordinate
//      AcqRoiXlen:         Acquisition Video ROI length
//      AcqRoiYlen:         Acquisition Video ROI width
//  Returns:
//      none
//
```

```
////////////////////////////////////////////////////////////////////////////

void UpdateTitleAcqRoi( HWND hwnd, LONG DestinationX,  LONG DestinationY, LONG AcqRoiXlen, LONG AcqRoiYlen)
{
    char szBuffer[256];
    char szBuffer2[14];

    GetWindowText(hwnd, szBuffer2, sizeof(szBuffer2)-1); // get the title of child window
    wsprintf(szBuffer, "%s ROI:(%d,%d), WxH:(%d,%d)", szBuffer2, DestinationX, DestinationY, AcqRoiXlen, AcqRoiYlen);
    SetWindowText(hwnd, szBuffer);
}

////////////////////////////////////////////////////////////////////////////
// UpdateTitleAcqScaling
//
//   Description:
//              Update title of the Acquisition Scaling dialog box
//
//   Returns:
//        none
//
////////////////////////////////////////////////////////////////////////////

void UpdateTitleAcqScaling ( void)
{
    char szBuffer[256];
    BD_VW_INPUT_CONTROL InControl;

    // Get Current Input Control
    BD_GetVideoWindowInputControl( &InControl);

    if ( BDBoard.hwndAcqScalingDialog)
    {
        wsprintf( szBuffer, "%d", (BDBoard.ScalingPercentage * InControl.AcqRoi.xlen)/100);
        SetWindowText( GetDlgItem( BDBoard.hwndAcqScalingDialog, IDC_SCALING_WIDTH), szBuffer);
        wsprintf( szBuffer, "%d", (BDBoard.ScalingPercentage * InControl.AcqRoi.ylen)/100);
        SetWindowText( GetDlgItem( BDBoard.hwndAcqScalingDialog, IDC_SCALING_HEIGHT), szBuffer);
    }
}


#include <stdio.h>
#include <windowsx.h>
#include "custom.h"
#include "bdpro.h"
#include "bdapi.h"
#include "global.h"
#include "resource.h"
#include "ckbrush.h"
#include "image.h"
#include <commctrl.h>


#if !WIN32
#ifndef    GET_WM_COMMAND_ID
   #define GET_WM_COMMAND_ID(wp, lp)        (wp)
#endif
#endif
```

```
BDBOARD BDBoard;
char InitFile[_MAX_PATH];

static const char MAINCLASS[]   = "BDPRO";
static const char MAINCAPTION[] = "BDPRO";
static HANDLE hFileMap;    // File Handle to control one instance

static HWND    *phAcqImg;          // Pointer to handle of Image dialog box
static HWND    *phZoom;                // Pointer to handle of Zoom dialog box
static HWND      *phAcqROI;            // Pointer to handle of Acquisition ROI dialog box
static HWND      *phVideoControl;   // Handle of Video Control dialog box
static HWND      *phAcqScaling;        // Pointer to handle of Acquisition Scaling dialog box

static BOOL WriteState(char *Busy);
static BOOL ReadState(char *Busy);
static LRESULT ResetVideoOutputWindow(void);



// These variables are required for horizontal scrolling

static int xMinScroll=0;                // minimum horizontal scroll value
static int xCurrentScroll=0;            // current horizontal scroll value
static int xMaxScroll=0;                // maximum horizontal scroll value

// These variables are required for vertical scrolling

static int yMinScroll=0;                // minimum vertical scroll value
static int yCurrentScroll=0;            // current vertical scroll value
static int yMaxScroll=0;                // maximum vertical scroll value
static int HorizontalStepIncrement;// size of horizontal step change
static int VerticalStepIncrement;  // size of vertical step change

//BOOL   index;

SCROLLINFO si;                                  // structure contains scroll bar parameters

static int iMaxZoom=-1;          // value to control the Zoom in full-screen, will have value from 0 to 2
static LRESULT SetZoomRightArrow( DWORD iInc);
static LRESULT SetZoomLeftArrow( DWORD iInc);
static LRESULT SetZoomUpArrow(  DWORD iInc);
static LRESULT SetZoomDownArrow( DWORD iInc);
static LRESULT SetZoomIn(void);
static LRESULT SetZoomOut(void);
void FitScreen(HWND hWnd);

#ifdef __BORLANDC__
#pragma warn -par
#endif


//////////////////////////////////////////////////////////////////////////////
//   WinMain()
//////////////////////////////////////////////////////////////////////////////
int PASCAL WinMain(HINSTANCE hinst, HINSTANCE hinstPrev, LPSTR szCmdLine, int iCmdShow)
{
int iReturn = 0;
int i=50000,j=100;
```

```
    if(InitApp(hinst))
    {
        if(InitInst(hinst, hinstPrev, iCmdShow))
        {
     //     SendMessage(BDBoard.hWnd, WM_SYSCOMMAND, SC_MAXIMIZE,"");
            while(i){
                  while(j)j--;
                  j=100;
                  i--;
            }
            SetAcqRoiToDefault(BDBoard.hWnd);
            StartGrab(BDBoard.hMenu);
            i=5000;
            j=100;
            while(i){
                  while(j)j--;
                  j=100;
                  i--;
            }
            StopGrab(BDBoard.hMenu);
            SendMessage(BDBoard.hWnd, WM_LBUTTONDOWN, MK_LBUTTON,"")
//          index=FALSE;
            iReturn = MsgLoop();
      /*    i=5000;
            j=100;
            while(i){
                  while(j)j--;
                  j=100;
                  i--;
            }*/
            DestroyWindow(BDBoard.hWnd);
        }
            else if( BDBoard.hWnd)
                 DestroyWindow( BDBoard.hWnd);

        ShutdownInst();
    }
    ShutdownApp(hinst);

    return(iReturn);
}

#ifdef __BORLANDC__
#pragma warn .par
#endif

////////////////////////////////////////////////////////////////////////////
// InitApp()
//
// Application initialization
////////////////////////////////////////////////////////////////////////////
BOOL InitApp(HINSTANCE hinst)
{
    WNDCLASS wc;

    if(GetClassInfo(hinst, MAINCLASS, &wc))
        return(TRUE);

    // Define window class
```

```
    wc.style          = CS_HREDRAW | CS_VREDRAW;
    wc.lpfnWndProc    = WindowProc;
    wc.cbClsExtra     = 0;
    wc.cbWndExtra     = 0;
    wc.hInstance      = hinst;
    wc.hIcon          = LoadIcon(hinst, MAKEINTRESOURCE(IDI_MAIN));
    wc.hCursor        = LoadCursor(NULL, IDC_ARROW);
    wc.hbrBackground  = NULL;
    wc.lpszMenuName   = "IDR_MENU1";

    wc.lpszClassName = MAINCLASS;

    // Register window
    return(RegisterClass(&wc) != 0);
}

////////////////////////////////////////////////////////////////////////////
// InitInst()
//
// Instance initialization
//
// Returns:
//         TRUE if successful, FALSE otherwise
////////////////////////////////////////////////////////////////////////////
BOOL InitInst(HINSTANCE hinst, HINSTANCE hinstPrev, int iCmdShow)
{
    DWORD dwKeyingColor;
    LRESULT lResult;
    char  drive[_MAX_DRIVE], path[_MAX_DIR], ext[_MAX_EXT] = ".INI", fname[_MAX_FNAME];
    char str[_MAX_FNAME];
    RECT rcClient;

    if(hinstPrev)
      return(FALSE);

    // Create an filemapping to be shared by all BDPRO application
    hFileMap = (HANDLE) CreateFileMapping ( (HANDLE) 0xFFFFFFFF,
                                                        NULL,
                                                        PAGE_READWRITE,
                                                        0,
                                                        1,
                                                        "BANDITSHARED");

    if (hFileMap != NULL)
    {
        if (GetLastError() == ERROR_ALREADY_EXISTS)
        {
            // if the board is already being used by previous BDPRO demo program, return FALSE
            char Busy;
            BOOL bVal;
            bVal = ReadState(&Busy);
            if (Busy || !bVal)
            {
                MessageBox(NULL, "BANDIT board has already been used.", NULL, MB_OK | MB_ICONSTOP
                return FALSE;
            }
        }
        else
        {
```

```
            // First time user
            char Busy=1;
            BOOL bVal;
            bVal = WriteState(&Busy);
            if (!bVal)
            {
                MessageBox(NULL, "Could not write to memory-mapped file. Please reboot the system.", NULL, MB_OK
MB_ICONSTOP);
                return FALSE;
            }
        }
    }

  BDBoard.hinst = hinst;

  lResult = BD_OpenDriver();
   if( lResult)
    {
       PrintBanditErrorMessage(NULL, lResult, "ERROR after calling BD_OpenDriver(), file \"bdpro.c\"");
    return( FALSE);
    }

  // Get the executable file path
  GetModuleFileName(hinst, InitFile, sizeof(InitFile));
  // Create the initialization file name
  _splitpath(InitFile, drive, path, fname, NULL);
  _makepath(InitFile, drive, path, APP_TITLE, ext);

  if( NULL == (BDBoard.hWnd = CreateWindowEx( WS_EX_RIGHTSCROLLBAR , MAINCLASS, MAINCAPTION,
                              WS_OVERLAPPEDWINDOW | WS_HSCROLL | WS_VSCROLL,
                              0, 0,
                              CW_USEDEFAULT,CW_USEDEFAULT,
                              NULL, NULL, hinst, NULL)))
  {
     MessageBox(NULL, "Unable to open main window!", NULL,
              MB_ICONSTOP | MB_OK);
     return(FALSE);
  }

  // Get a handle of the Main Menu
  BDBoard.hMenu = GetMenu( BDBoard.hWnd);

   if( !InitBanditToDefaults( BDBoard.hWnd, InitFile))
   {
    MessageBox( BDBoard.hWnd, "Error in initializing Bandit!", NULL,
                MB_ICONSTOP | MB_OK);
    return(FALSE);
   }

   // Allocate the Offscreen Video Buffer, and display it
   lResult = AllocateVideoBuffer( TRUE);
   if( lResult) return( FALSE);

   //set up keying
   BD_GetColorKey(&dwKeyingColor);
   CreateColorKeyBrush(dwKeyingColor);

   // Initialize the settings of Horizontal Flip and Vertical Flip
   SetHorizontalFlip( BDBoard.CurHorizontalFlip);
```

```
    SetVerticalFlip( BDBoard.CurVerticalFlip);

    ShowWindow(BDBoard.hWnd, iCmdShow);

    // Initialize Video Input Control according to BDPRO.INI, has to be called after calling AllocateVideoBuffer()
    lResult = _BD_SetVideoWindowInputControl(&BDBoard.InControl);
    if( lResult)
        {
        // This should never happen
        PrintBanditErrorMessage(BDBoard.hWnd, lResult, "ERROR after calling _BD_SetVideoWindowInputControl(), file \"bdpro.c\"");
        return FALSE;
        }

    ResizeOutputControlVideoWindow( BDBoard.hWnd);
    UpdateTitleBar();

    UpdateWindow(BDBoard.hWnd);

    BUILDSECTION(str);

//  Adjust.HorizontalShift = GetPrivateProfileInt(str, DEF_VIDIN_ADJUST_HSHIFT, DEFAULT_VIDIN_ADJUST_HSHIFT, InitFile);
//  Adjust.HorizontalLength =GetPrivateProfileInt(str, DEF_VIDIN_ADJUST_HLENSHIFT, DEFAULT_VIDIN_ADJUST_HLENSHIFT,
InitFile);
//  Adjust.VerticalShift = GetPrivateProfileInt(str, DEF_VIDIN_ADJUST_VSHIFT, DEFAULT_VIDIN_ADJUST_VSHIFT, InitFile);
//  if( BD_SetVideoInputAdjust( &Adjust))
//  {
//      MessageBox( BDBoard.hWnd, "BD_SetVideoInputAdjust() failed", "ERROR in INIT.C", MB_OK|MB_ICONSTOP);
//      return FALSE;
//  }
    FitScreen(BDBoard.hWnd);

    //GetWindowRect(BDBoard.hWnd, &rcClient);
    SendMessage( BDBoard.hWnd, WM_SIZE, 0, MAKELPARAM( (WORD) (1023), (WORD) (767) ));
    InitCommonControls();
    CAP_Size.x=10;
    CAP_Size.y=10;
    return(TRUE);
}

int MsgLoop(void)
{
    MSG msg;
    BOOL indx=1;

    while(1)
    {
        indx=GetMessage(&msg, NULL, 0, 0);
        if(!indx)break;
        TranslateMessage(&msg);
        DispatchMessage(&msg);
    }
    return((int)msg.wParam);
}

void ShutdownInst(void)
{
        BD_HideVideoWindow();
    BD_CloseDriver();
        if (hFileMap)
```

```
        {
            CloseHandle(hFileMap);
            hFileMap = NULL;
        }
}

void ShutdownApp(HINSTANCE hinst)
{
   WNDCLASS wc;

   if(GetClassInfo(hinst, MAINCLASS, &wc))
      UnregisterClass(MAINCLASS, hinst);    //Unregister window
}


LRESULT PASCAL EXPORT WindowProc(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam)
{
    BOOL bChecked; // boolean value to see whether Status Bar option is checked or not

   switch(uMsg)
   {
        case WM_CHAR:
            switch (LOWORD(wParam))
            {
                case VK_ESCAPE: // Escape
                {
                        LONG lStyle;
                        BOOL bChecked;

                        if (!BDBoard.bFullScreen)
                            break;
                        else
                            BDBoard.bFullScreen = FALSE; // so that irtual keys and WM_SIZE message can work properly

                        // restore old menu
                        SetMenu(hWnd, BDBoard.hMenu);

                        xCurrentScroll = 0;
                        yCurrentScroll = 0;
                        xMinScroll = 0;
                        yMinScroll = 0;
                        iMaxZoom = -1; // value for ZOOM factor in full-screen

                        // restore old caption
                        lStyle = GetWindowLong(hWnd, GWL_STYLE);
                        lStyle |= WS_OVERLAPPEDWINDOW;

                        SetWindowLong(hWnd, GWL_STYLE, lStyle);
                        MoveWindow( hWnd, BDBoard.rcWndClient.left, BDBoard.rcWndClient.top, BDBoard.OutControl.VideoZoom.x
GetSystemMetrics(SM_CXBORDER) + GetSystemMetrics(SM_CXDLGFRAME),
                                    BDBoard.OutControl.VideoZoom.y + GetSystemMetrics(SM_CYCAPTION)
GetSystemMetrics(SM_CYMENU) + GetSystemMetrics(SM_CYBORDER) + GetSystemMetrics(SM_CYDLGFRAME)
BDBoard.StatusBarHeight, TRUE);
                        BDW_MoveVideoWindow( BDBoard.rcWndClient.left + GetSystemMetrics(SM_CXBORDER)
GetSystemMetrics(SM_CXDLGFRAME),
                            BDBoard.rcWndClient.top + GetSystemMetrics(SM_CYCAPTION) + GetSystemMetrics(SM_CYMENU)
GetSystemMetrics(SM_CYBORDER) + GetSystemMetrics(SM_CYDLGFRAME) + BDBoard.StatusBarHeight);
                        SetWindowPos ( hWnd, HWND_NOTOPMOST, 0, 0, 0, 0, SWP_NOMOVE | SWP_NOSIZE);
```

```
                    if (BDBoard.hwndZoomDialog != NULL)
                        ShowWindow(BDBoard.hwndZoomDialog, SW_SHOW);

                    // Restore Video Window Control (destroyed when full screen is used)
                    BD_SetVideoWindowOutputControl( &BDBoard.OutControl);

                    if (BDBoard.hwndStatusBar != NULL)
                    {
                        bChecked    =    GetMenuState(BDBoard.hMenu,    ID_OPTIONS_DISPLAY_SHOWSTATUSBAR,
MF_BYCOMMAND);
                        if (bChecked & MF_CHECKED)
                            ShowWindow(BDBoard.hwndStatusBar, SW_SHOW);
                        else
                            ShowWindow(BDBoard.hwndStatusBar, SW_HIDE);
                    }

                    UpdateTitleBar();

                    // Restore showing cursor after returning from full-screen mode
                    ShowCursor(TRUE);
                    InvalidateRect(hWnd, NULL, TRUE);
                    UpdateWindow(hWnd);
                }
            } // end of switch
            return 0;

            case WM_KEYDOWN:
                switch (wParam)
                {
                    case VK_ADD: // Numeric keypad +
                            if (BDBoard.bFullScreen)
                            {
                                iMaxZoom ++; // has value 0, 1, 2, will be decremented by VK_SUBTRACT
                                if (iMaxZoom > 2)
                                {
                                    iMaxZoom = 2;
                                    return FALSE;
                                }
                                SetZoomIn(); // iMaxZoom has value from 0 to 2
                            }
                        break;

                    case VK_SUBTRACT: // Numeric keypad -
                            if (BDBoard.bFullScreen)
                                {
                                    if (iMaxZoom < -1)
                                    {
                                        iMaxZoom = -1;
                                        return FALSE;
                                    }
                                iMaxZoom --; // has value 0, 1, 2, will be incremented by VK_ADD
                                SetZoomOut();
                                }

                        break;

                    case VK_RIGHT:
                        {
                        if (BDBoard.bFullScreen)
```

```
            SetZoomRightArrow( 1);
        else
            SendMessage(hWnd, WM_HSCROLL, SB_PAGEDOWN, 0L
        break;
        }

case VK_LEFT:
        {
        if (BDBoard.bFullScreen)
            SetZoomLeftArrow( 1);
        else
            SendMessage(hWnd, WM_HSCROLL, SB_PAGEUP, 0L);
        break;
        }

case VK_UP:  // Up Arrow
        {
        if (BDBoard.bFullScreen)
            SetZoomDownArrow( 1);
        else
            SendMessage(hWnd, WM_VSCROLL, SB_LINEUP, 0L);
        break;
        }

case VK_DOWN:  // Down Arrow
        {
        if (BDBoard.bFullScreen)
            SetZoomUpArrow( 1);
        else
            SendMessage(hWnd, WM_VSCROLL, SB_LINEDOWN, 0L);
        break;
        }

case VK_PRIOR:  // Page Up
        {
        if (BDBoard.bFullScreen)
            SetZoomDownArrow( 8);
        else
            SendMessage(hWnd, WM_VSCROLL, SB_PAGEUP, 0L);
        break;
        }

case VK_NEXT:  // Page Down
        {
        if (BDBoard.bFullScreen)
            SetZoomUpArrow( 8);
        else
            SendMessage(hWnd, WM_VSCROLL, SB_PAGEDOWN, 0L)
        break;
        }

case VK_HOME: // Home key
        {
        if (BDBoard.bFullScreen)
            SetZoomRightArrow( 8);
        else
            SendMessage(hWnd, WM_HSCROLL, SB_PAGEUP, 0L);
        break;
        }
```

```
            case VK_END:
                {
                if (BDBoard.bFullScreen)
                    SetZoomLeftArrow( 8);
                else
                    SendMessage(hWnd, WM_HSCROLL, SB_PAGEDOWN, 0L);
                break;
                }

            default:
                break;
        }


case WM_CREATE:
    {
    // Each time the window switches mode from fullscreen to normal size, WM_CREATE is sent
    // Needs only to call once
    static BOOL bOnce = TRUE;

    if (bOnce)
        {
            AddCustomMenu( hWnd);
            InitStatusBar( hWnd);
            bOnce = FALSE;
        }
    }
    break;

case WM_PAINT:
    {
    LRESULT lResult;
    RECT rcBlack;
    HDC hdc;
    BD_VW_OUTPUT_CONTROL_LIMITS_EX Limits;

    lResult = PaintWithColorKeyBrush(hWnd);

    if (BDBoard.bFullScreen)
        {
            int nMaxWidth;
            int nMaxHeight;

            nMaxWidth = GetSystemMetrics (SM_CXSCREEN);
            nMaxHeight = GetSystemMetrics( SM_CYSCREEN);

            hdc = GetDC (hWnd);
            lResult = BD_GetVideoWindowOutputControlLimitsEx( &Limits);

            if (!lResult)
            {
                // Fill out the right rectangle with black
                rcBlack.left = Limits.VideoZoom.MaxX;
                rcBlack.top = 0;
                rcBlack.right = nMaxWidth;
                rcBlack.bottom = nMaxHeight;
                PatBlt(hdc, rcBlack.left, rcBlack.top, rcBlack.right - rcBlack.left,
                rcBlack.bottom - rcBlack.top, BLACKNESS);
```

```
                    // Fill out the bottom rectangle with black
                    rcBlack.left = 0;
                    rcBlack.top = Limits.VideoZoom.MaxY;
                    rcBlack.right = nMaxWidth;
                    rcBlack.bottom = nMaxHeight;
                    PatBlt(hdc, rcBlack.left, rcBlack.top, rcBlack.right - rcBlack.left,
                    rcBlack.bottom - rcBlack.top, BLACKNESS);
                    ReleaseDC (hWnd, hdc);
                }

            }
        return lResult;
        }

    case WM_MOVING:
        return  (LimitMove ( lParam));

    case WM_MOVE:
        return(BDW_MoveVideoWindow( LOWORD(lParam), HIWORD(lParam)));

    case WM_DISPLAYCHANGE:
        {
            if (BDBoard.bGrabOn)
            {
                StopGrab( BDBoard.hMenu);
                BD_ClearVideoWindow();
            }
            MessageBox( hWnd, "Changing monitor resolution while running is not allowed. Please run BDPRO again.", "FATAL
ERROR",  MB_OK | MB_ICONSTOP);
            PostMessage (hWnd, WM_CLOSE, 0, 0);
        }

        break;

  case WM_SIZE:
        {
            BOOL bChecked;
            LRESULT lResult;

            lResult = BDW_SizeVideoWindow( hWnd, LOWORD( lParam), HIWORD( lParam));

            bChecked = GetMenuState(BDBoard.hMenu, ID_OPTIONS_DISPLAY_SHOWSTATUSBAR, MF_BYCOMMAND);
            if ((bChecked & MF_CHECKED)  && BDBoard.hwndStatusBar)
                {
                    ResizeStatusBar(BDBoard.hwndStatusBar, lParam, TRUE);
                    UpdateStatusBar(BDBoard.hwndStatusBar);
                }
            return lResult;
        }

        // Windows session is about to be ended
  case WM_QUERYENDSESSION:
    WMDestroy();
        return TRUE;

    case WM_CLOSE:
  case WM_DESTROY:
    return(WMDestroy());
   case WM_LBUTTONDOWN:
```

```
    if((LOWORD(wParam)==MK_LBUTTON))
        {
                    static      HWND hParenthWnd;
                    static RECT CurCaptureRect;
                    static RECT rcClient;
                    RECT rc;

//          StopGrab(BDBoard.hMenu);

                    GetClientRect(hWnd, &rc);
                    GetCaptureRect2(BDBoard.hWnd, hWnd, &CurCaptureRect,lParam);
                    SaveImageRect(hWnd, &CurCaptureRect,lParam);
                    GetClientRect(hParenthWnd, &rcClient);
                    InvalidateRect(hParenthWnd, &rcClient, TRUE);
                    UpdateWindow(hParenthWnd);
                    ClipCursor(NULL);
//          index=TRUE;
//      }
//  else if((LOWORD(wParam)==MK_LBUTTON)&&index=='TRUE)
//      {
//          StartGrab(BDBoard.hMenu);
//          index=FALSE;
        }
    else return(DefWindowProc(hWnd, uMsg, wParam, lParam));
    break;

    case WM_MOUSEMOVE:
        {
        POINT pt;

        // if full-screen mode, do not update the position of the status bar
        if (BDBoard.bFullScreen)
            break;

        // the position of the mouse will be displayed on the status bar
        pt.x = LOWORD(lParam) + (xCurrentScroll * BDBoard.wZoomFactor);
        pt.y = HIWORD(lParam) + (yCurrentScroll * BDBoard.wZoomFactor);

        bChecked = GetMenuState(BDBoard.hMenu, ID_OPTIONS_DISPLAY_SHOWSTATUSBAR, MF_BYCOMMAND
        if (bChecked & MF_CHECKED)
            {
                UpdateStatusBarCursor(BDBoard.hwndStatusBar, &pt);
                UpdateStatusBar(BDBoard.hwndStatusBar);
            }
        }
        break;

    case WM_HSCROLL:
        {

   int xNewPos;    // new position

   switch (LOWORD(wParam)) {

      // User clicked the shaft left of the scroll box

      case SB_PAGEUP:
          xNewPos = xCurrentScroll - VerticalStepIncrement;
          break;
```

```
    // User clicked the shaft right of the scroll box

    case SB_PAGEDOWN:
        xNewPos = xCurrentScroll + VerticalStepIncrement;
        break;

    // User clicked the left arrow

    case SB_LINEUP:
        xNewPos = xCurrentScroll - GRANUL_X;
        break;

    // User clicked the right arrow

    case SB_LINEDOWN:
        xNewPos = xCurrentScroll + GRANUL_X;
        break;

    // User dragged the scroll box

    case SB_THUMBPOSITION:
            case SB_THUMBTRACK:
        xNewPos = HIWORD(wParam);
        break;

    default:
        xNewPos = xCurrentScroll;
}

// New position must be between 0 and the screen width

xNewPos = max(0, xNewPos);
xNewPos = min(xMaxScroll, xNewPos);

// If the current position does not change, do not scroll

if (xNewPos == xCurrentScroll)
    break;

// Reset the current scroll position
xCurrentScroll = xNewPos;

// Reset the scroll bar

si.cbSize = sizeof(si);
si.fMask  = SIF_POS;
si.nPos   = xCurrentScroll;
SetScrollInfo(hWnd, SB_HORZ, &si, TRUE);

   // Update mouse position when the user hits horizontal scroll ba
   BDW_MoveVWOHorizontal(xCurrentScroll);
   {
           POINT pt;
           pt.x = xCurrentScroll;
           pt.y = yCurrentScroll;
           UpdateStatusBarCursor(BDBoard.hwndStatusBar, &pt);
   }
   BDBoard.xCurrentScroll = xCurrentScroll;
```

```
} // WM_HSCROLL
    break;

case WM_VSCROLL: {

    int yNewPos;    // new position

    switch (LOWORD(wParam)) {

        // User clicked the shaft above the scroll box

        case SB_PAGEUP:
            yNewPos = yCurrentScroll - HorizontalStepIncrement;
            break;

        // User clicked the shaft below the scroll box

        case SB_PAGEDOWN:
            yNewPos = yCurrentScroll + HorizontalStepIncrement;
            break;

        // User clicked the top arrow

        case SB_LINEUP:
            yNewPos = yCurrentScroll - GRANUL_Y;
            break;

        // User clicked the bottom arrow

        case SB_LINEDOWN:
            yNewPos = yCurrentScroll + GRANUL_Y;
            break;

        // User dragged the scroll box

        case SB_THUMBPOSITION:
              case SB_THUMBTRACK:
            yNewPos = HIWORD(wParam);
            break;

        default:
            yNewPos = yCurrentScroll;
    }

    // New position must be between 0 and the screen height

    yNewPos = max(0, yNewPos);
    yNewPos = min(yMaxScroll, yNewPos);

    // If the current position does not change, do not scroll

    if (yNewPos == yCurrentScroll)
        break;

    // Reset the current scroll position

    yCurrentScroll = yNewPos;
```

```
    // Reset the scroll bar

    si.cbSize = sizeof(si);
    si.fMask  = SIF_POS;
    si.nPos   = yCurrentScroll;
    SetScrollInfo(hWnd, SB_VERT, &si, TRUE);
       BDW_MoveVWOVertical(yCurrentScroll);

} // WM_VSCROLL

      // Update mouse position when the user hits vertical scroll bar
       {
              POINT pt;
              pt.x = xCurrentScroll;
              pt.y = yCurrentScroll;
              UpdateStatusBarCursor(BDBoard.hwndStatusBar, &pt);
          }
       BDBoard.yCurrentScroll = yCurrentScroll;
    break;

     case WM_COMMAND:      // message: command from application menu
          {
              switch( GET_WM_COMMAND_ID(wParam, lParam))
              {

                  case ID_FILE_NEW:
                       BD_ClearVideoWindow();
                       break;

                  case ID_FILE_OPEN:
                       OpenImage(hWnd);
                       break;

                  case ID_FILE_SAVE:{
                       static      HWND hParenthWnd;
                       static RECT CurCaptureRect;
                       static RECT rcClient;
                       RECT rc;
                       GetClientRect(hWnd, &rc);
                       GetCaptureRect(GetParent(hWnd), hWnd, &CurCaptureRect
                       SaveImageRect(hWnd, &CurCaptureRect,lParam);
                       GetClientRect(hParenthWnd, &rcClient);
                       InvalidateRect(hParenthWnd, &rcClient, TRUE);
                       UpdateWindow(hParenthWnd);
                       ClipCursor(NULL);
                       //phAcqImg = CmdAcqImg(hWnd);
                                       }
                       break;

                  case ID_FILE_EXIT:       // Exit the Application
                       DestroyWindow(hWnd);
                       break;

                  case ID_OPTIONS_GRAB:
                  case ID_IMAGE_GRAB:
                       StartGrab(BDBoard.hMenu);
                  //   index=FALSE;
                       break;
                  case ID_IMAGE_FREEZE:
```

```
                case ID_OPTIONS_FREEZE:
                    StopGrab(BDBoard.hMenu);
                //  index=TRUE;
                    break;

                case ID_ACQUISITION_VIDEOCONTROL:   // set video control parameters
                    phVideoControl = CmdVideoControl(hWnd);
                    break;


                case ID_ACQUISITION_SCALING:  // set horizontal and vertical scaling
                    phAcqScaling = CmdAcqScaling(hWnd);
                    break;

                case ID_HELP_ABOUT:
                    DialogBox(GetWindowInstance(hWnd), MAKEINTRESOURCE(IDD_ABOUT), hWnd, (DLGPROC) DlgAbout);
                    break;

                case ID_ACQUISITION_CAMERA:   // select camera input and format
                    DialogBox(GetWindowInstance(hWnd),      MAKEINTRESOURCE(IDD_CAMSEL),      hWnd,      (DLGPROC)
DlgCameraSelect);
                    DrawMenuBar(hWnd);   // refresh the menu
                    break;

                case ID_OPTIONS_DISPLAY_HORIZONTALFLIP:
                    if (BDBoard.CurHorizontalFlip)
                    {
                        BDBoard.CurHorizontalFlip = FALSE;
                        SetHorizontalFlip( BDBoard.CurHorizontalFlip);
                        CheckMenuItem(BDBoard.hMenu,   ID_OPTIONS_DISPLAY_HORIZONTALFLIP,   MF_UNCHECKED   |
MF_BYCOMMAND);
                    }
                    else
                    {
                        BDBoard.CurHorizontalFlip = TRUE;
                        SetHorizontalFlip( BDBoard.CurHorizontalFlip);
                        CheckMenuItem(BDBoard.hMenu,    ID_OPTIONS_DISPLAY_HORIZONTALFLIP,    MF_CHECKED    |
MF_BYCOMMAND);
                    }
                    UpdateStatusBarHorizontalFlip(BDBoard.hwndStatusBar, BDBoard.CurHorizontalFlip);

                    break;

                case ID_OPTIONS_DISPLAY_SHOWSTATUSBAR:
                    {
                        BOOL bChecked;
                        RECT rcWnd;
                        int nNewWidth, nNewHeight;

                        GetWindowRect ( BDBoard.hWnd, &rcWnd);
                        nNewWidth = (rcWnd.right  - rcWnd.left);

                        bChecked      =      GetMenuState(BDBoard.hMenu,      ID_OPTIONS_DISPLAY_SHOWSTATUSBAR,
MF_BYCOMMAND);
                        if (bChecked & MF_CHECKED)
                        {
                            ShowWindow( BDBoard.hwndStatusBar, SW_HIDE);
                            CheckMenuItem(          BDBoard.hMenu,          ID_OPTIONS_DISPLAY_SHOWSTATUSBAR,
MF_UNCHECKED | MF_BYCOMMAND);
```

```
                    nNewHeight = (rcWnd.bottom - rcWnd.top) - BDBoard.StatusBarHeight;
                    BDBoard.StatusBarHeight = 0;
                }
                else
                {
                    ShowWindow( BDBoard.hwndStatusBar, SW_SHOW);
                    BDBoard.StatusBarHeight = GetStatusBarHeight( BDBoard.hwndStatusBar);
                    CheckMenuItem( BDBoard.hMenu, ID_OPTIONS_DISPLAY_SHOWSTATUSBAR, MF_CHECKED |
MF_BYCOMMAND);
                    nNewHeight = (rcWnd.bottom - rcWnd.top) + BDBoard.StatusBarHeight;
                }
                    // Resize app window such that status window will fit high-resolution mode
                    SetWindowPos(BDBoard.hWnd,
                                 NULL,
                                 0,              // Xpos
                                 0,              // YPos
                                 nNewWidth,      // Width
                                 nNewHeight,     // Height
                                 SWP_NOMOVE   |
                                 SWP_NOZORDER);
                                 // | SWP_NOACTIVATE);
                    InvalidateRect (BDBoard.hWnd, NULL, TRUE);
                    UpdateWindow (BDBoard.hWnd);
            }
            break;

        case ID_OPTIONS_DISPLAY_FITTOSCREEN:
            {
                RECT rcClient;
                GetClientRect(hWnd, &rcClient);
                if (BDBoard.bFitToScreen)
                {
                    BDBoard.bFitToScreen = FALSE;
                    CheckMenuItem(BDBoard.hMenu,  ID_OPTIONS_DISPLAY_FITTOSCREEN,  MF_UNCHECKED  |
MF_BYCOMMAND);
                    EnableMenuItem(BDBoard.hMenu,      ID_OPTIONS_DISPLAY_ZOOM,      MF_ENABLED      |
MF_BYCOMMAND);
                    EnableMenuItem(BDBoard.hMenu, IDM_POSITION_ROIS, MF_ENABLED | MF_BYPOSITION);
                }
                else
                {

                    BDBoard.bFitToScreen = TRUE;
                    ResetVideoOutputWindow();
                    CheckMenuItem(BDBoard.hMenu,  ID_OPTIONS_DISPLAY_FITTOSCREEN,  MF_CHECKED  |
MF_BYCOMMAND);
                    EnableMenuItem(BDBoard.hMenu, ID_OPTIONS_DISPLAY_ZOOM, MF_GRAYED | MF_DISABLED
| MF_BYCOMMAND);
                    EnableMenuItem(BDBoard.hMenu, 4, MF_GRAYED | MF_DISABLED | MF_BYPOSITION);
                    SetAcqRoiToDefault(BDBoard.hWnd);
                }
                // Update window, title bar, status bar
                DrawMenuBar(hWnd);   // refresh the menu
                SendMessage( hWnd, WM_SIZE, 0, MAKELPARAM( (WORD) (rcClient.right - rcClient.left), (WORD)
(rcClient.bottom - rcClient.top) ));

            }
```

```
                    break;

                case ID_OPTIONS_DISPLAY_ZOOM:
                    phZoom = CmdZoom(hWnd);
                    break;

                case ID_OPTIONS_VIDEOINADJUST:
                    ViewVideoInAdjust( BDBoard.hWnd);
                    break;

                case ID_ROIS_ACQUISITION:
                    SetWindowFullSize( hWnd); // ie.640x480 in NTSC, 768x576 in PAL
                    phAcqROI = CmdAcqROI(hWnd);
                    break;

                case ID_OPTIONS_DISPLAY_FULLSCREEN:
                    SetWindowToFullScreen(hWnd);
                    break;

                default:
                    if( !CustomControl( hWnd, GET_WM_COMMAND_ID(wParam, lParam)
                        return( DefWindowProc(hWnd, uMsg, wParam, lParam));
                    }
                break;
                }
        default:
            return(DefWindowProc(hWnd, uMsg, wParam, lParam));
    }

    return( 0);
}


////////////////////////////////////////////////////////////////////////////
// Standard Windows WM_DESTROY message handler
////////////////////////////////////////////////////////////////////////////
LRESULT WMDestroy(void)
{
    // destroy Zoom Dialog Box if still active
    if (phZoom && *phZoom)
    {
        SendMessage( (HWND) (*phZoom), WM_CLOSE, 0, 0L);
        phZoom = NULL;
    }

    // destroy Image ROI Dialog Box if still active
    if (phAcqImg && *phAcqImg)
    {
        SendMessage( (HWND) (*phAcqImg), WM_CLOSE, 0, 0L);
        phAcqImg = NULL;
    }

    // destroy Acquisition ROI Dialog Box if still active
    if (phAcqROI && *phAcqROI)
    {
        SendMessage( (HWND) (*phAcqROI), WM_CLOSE, 0, 0L);
        phAcqROI = NULL;
    }
```

```
    // destroy Acquisition ROI Dialog Box if still active
    if (phAcqScaling && *phAcqScaling)
    {
        SendMessage( (HWND) (*phAcqScaling), WM_CLOSE, 0, 0L);
        phAcqROI = NULL;
    }

    DestroyColorKeyBrush();

    if (hFileMap)
    {
        CloseHandle(hFileMap);
        hFileMap = NULL;
    }

    // Destroy status bar window
    if (BDBoard.hwndStatusBar)
    {
        DestroyWindow( BDBoard.hwndStatusBar);
        BDBoard.hwndStatusBar = NULL;
    }

    // Destroy Modeless Video Control Dialog Box
    if (BDBoard.hwndVideoControlDialog)
    {
        DestroyWindow( BDBoard.hwndVideoControlDialog);
        BDBoard.hwndVideoControlDialog = NULL;
    }

    // Destroy Modeless Scaling Dialog Box
    if (BDBoard.hwndAcqScalingDialog)
    {
        DestroyWindow( BDBoard.hwndAcqScalingDialog);
        BDBoard.hwndAcqScalingDialog = NULL;
    }

    // Destroy Modeless Video Input Adjust Dialog Box
    if (BDBoard.hWndVideoInAdjust)
    {
        DestroyWindow( BDBoard.hWndVideoInAdjust);
        BDBoard.hWndVideoInAdjust = NULL;
    }

    ReleaseCustomResources();

  PostQuitMessage(0);
  return(0);
}


BOOL StartGrab(HANDLE hmenu)
{
    LRESULT lResult;
    DWORD dwStatus;

    if (BD_GetAcqStatus(&dwStatus) == BD_OK)
        if (!(dwStatus & BD_ACQSTATUS_VIDEO_PRESENT))
            {
            MessageBox(BDBoard.hWnd, "No video signal detected", "WARNING", MB_ICONINFORMATION | MB_OK)
```

```
            return FALSE;
            }

    lResult = BD_AcquireVideo();
    if( lResult == BD_OK)
    {
        EnableMenuItem( hmenu, IDM_POSITION_GRAB, MF_DISABLED | MF_GRAYED | MF_BYPOSITION);
        EnableMenuItem( hmenu, IDM_POSITION_FREEZE, MF_ENABLED | MF_BYPOSITION);
        EnableMenuItem( hmenu, ID_ACQUISITION_CAMERA, MF_DISABLED | MF_GRAYED | MF_BYCOMMAND)
        EnableMenuItem( hmenu, ID_FILE_NEW, MF_GRAYED | MF_DISABLED | MF_BYCOMMAND);
        EnableMenuItem( hmenu, ID_FILE_OPEN, MF_GRAYED | MF_DISABLED | MF_BYCOMMAND);
        EnableMenuItem( hmenu, ID_FILE_SAVE, MF_GRAYED | MF_DISABLED | MF_BYCOMMAND);
        DrawMenuBar( BDBoard.hWnd);
        BDBoard.bGrabOn = TRUE;
        return TRUE;
    }
    else
    {
        PrintBanditErrorMessage ( BDBoard.hWnd, lResult, "ERROR after calling BD_AcquireVideo(), file \"bdpro.c\"");
        return FALSE;
    }
}


BOOL StopGrab(HANDLE hmenu)
{
    LRESULT lResult;
    char Msg[200];

    lResult = BD_FreezeVideo();
    if( lResult == BD_OK)
    {
        EnableMenuItem( hmenu, IDM_POSITION_GRAB, MF_ENABLED | MF_BYPOSITION);
        EnableMenuItem( hmenu, IDM_POSITION_FREEZE, MF_DISABLED | MF_GRAYED | MF_BYPOSITION);
        // Camera is enabled only when ROI dialog box is not active
        if (!IsWindow (BDBoard.hwndAcqROIDialog))
            EnableMenuItem( hmenu, ID_ACQUISITION_CAMERA, MF_ENABLED | MF_BYCOMMAND);
        EnableMenuItem( hmenu, ID_FILE_NEW, MF_ENABLED | MF_BYCOMMAND);
        EnableMenuItem( hmenu, ID_FILE_OPEN, MF_ENABLED | MF_BYCOMMAND);
        EnableMenuItem( hmenu, ID_FILE_SAVE, MF_ENABLED | MF_BYCOMMAND);
        DrawMenuBar( BDBoard.hWnd);
        BDBoard.bGrabOn = FALSE;
        return TRUE;
    }
    else
    {
        wsprintf( Msg, "Error Freezing Video: ERR #%ld", lResult);
        MessageBox(NULL, Msg, NULL, MB_ICONSTOP | MB_OK);
        return FALSE;
    }
}

void UpdateTitleBar(void)
{
    char buf[100];
    BD_VW_INPUT_CONTROL VWI;
    BD_VW_OUTPUT_CONTROL VWO;
    BD_GetVideoWindowInputControl(&VWI);
    BD_GetVideoWindowOutputControl(&VWO);
```

```
    wsprintf(buf, "영상수집창 (Roi:%dx%d,Scale:%dx%d,Dsp:%dx%d)", VWI.AcqRoi.xlen, VWI.AcqRoi.ylen,
        VWI.VideoScale.x, VWI.VideoScale.y, VWO.VideoZoom.x, VWO.VideoZoom.y);
    SetWindowText(BDBoard.hWnd, buf);
}

////////////////////////////////////////////////////////////////////////////
//  AllocateVideoBuffer
//
//  Description:
//      Allocate video buffer size according to the actual size of the camera size.
//      The program will crop the actual size of the camera to fit into the monitor
//      resolution
//
// Input:
//      bFirst: TRUE if this is the first time to call this function, FALSE otherwise
//
//  Output:
//      TRUE is successful, FALSE otherwise
//
////////////////////////////////////////////////////////////////////////////

LRESULT AllocateVideoBuffer( BOOL bFirst)
{
    LRESULT lResult = BD_OK;
    BD_SIZE CamSize;        // Camera Image Size
    BD_SIZE Size;           // Video Buffer Image Size
    BD_SIZE OldSize;        // Previous Video Buffer Image Size
    BOOL bScalingModified = FALSE;

    // Keep Current Size of Video Buffer if it is NOT the first time
    if (!bFirst)
    {
        OldSize = BDBoard.VW_Size;
        lResult = BD_DestroyVideoWindow();
    }

    if ( !lResult)
        // Get Size of Image from Camera
        lResult = BD_GetAcqInputStandardSize( &CamSize);

    if (CamSize.x > (DWORD) GetSystemMetrics( SM_CXSCREEN))
    {
        CamSize.x = (DWORD) GetSystemMetrics( SM_CXSCREEN);
        bScalingModified = TRUE;
        CamSize.y = CamSize.x * 3 / 4;  // keep aspect ratio
    }

    if( !lResult)
    {
        int i;
        BOOL bReduce = FALSE;

        for (i = 0, Size = CamSize; i < MAX_REDUCED_SIZE; i ++)
        // Create Offscreen Video Buffer: preferred Full Size of Camera Image
        {
            lResult = BD_CreateVideoWindow( BDBoard.hWnd, BD_VW_NO_COLORKEY_DRAW, Size.x, Size.y)

            if (lResult == BD_ERR_FBMEMORY)
```

```
        {
            Size.x >>= 1;
            Size.y >>= 1;
            bReduce = TRUE;
        }
        else
            break;
    }

    // if by some reasons Create Video Window fails, try to use old values
    if (lResult && !bFirst)
        {
            // Re-create original Image Size
            lResult = BD_CreateVideoWindow( BDBoard.hWnd, BD_VW_NO_COLORKEY_DRAW, OldSize.x, OldSize.y);
            bScalingModified = TRUE;
        }

    if( lResult)
    {
        PrintBanditErrorMessage ( BDBoard.hWnd, lResult, "ERROR");
        return( lResult);
    }
    else if( bReduce)
    {
    //    char Msg[256];

    //    wsprintf( Msg, "Not enough memory for Camera Image Size (%d x %d). Created reduced Video Buffer: (%d x %d)
    //                        CamSize.x, CamSize.y, Size.x, Size.y);
    //    MessageBox(NULL, Msg, "Warning", MB_ICONWARNING | MB_OK);

    }
    K_Size.x=Size.x;K_Size.y=Size.y;
}

// For some special cases, monitor resolution is smaller than the camera size
// Need to update Acq Roi Scaling percentage, cannot use the old values
if ( !lResult && bScalingModified)
{
    char szBuffer[256];
    WORD wOldPercentage;        // recalculate scaling percentage
    BD_VW_INPUT_CONTROL InControl;

    wOldPercentage = BDBoard.ScalingPercentage;
    lResult = BD_GetVideoWindowInputControl( &InControl);
    if (!lResult)
    {
        // if the resolution is smaller than the size of the camera, then crop it to the size of monitor resolution
        InControl.AcqRoi.xlen = Size.x;
        InControl.AcqRoi.ylen = Size.y;

        BDBoard.ScalingPercentage = (WORD) (100L * InControl.AcqScale.x / InControl.AcqRoi.xlen);

        if (wOldPercentage != BDBoard.ScalingPercentage)
        {
            int nMaxWidth, nMaxHeight;

            lResult = BD_GetAcqInputStandardSize( &CamSize);

            if( lResult)
```

```
                {
                    PrintBanditErrorMessage ( BDBoard.hWnd, lResult, "ERROR");
                    return( lResult);
                }

                nMaxWidth = GetSystemMetrics (SM_CXSCREEN);
                nMaxHeight = GetSystemMetrics (SM_CYSCREEN);

                wsprintf (szBuffer, "Display resolution = %dx%d.\nYour acquisition has been changed from %dx%d to %dx%d.",
                    nMaxWidth, nMaxHeight, CamSize.x, CamSize.y, nMaxWidth, nMaxHeight );
                MessageBox (BDBoard.hWnd, szBuffer, "WARNING", MB_OK | MB_ICONWARNING);
            }
            lResult = _BD_SetVideoWindowInputControl( &InControl);
        }
        else
        {
            return lResult;
        }
    }

    if( lResult)
    {
        PrintBanditErrorMessage ( BDBoard.hWnd, lResult, "ERROR after calling BD_CreateVideoWindow(), file \"bdpro.c\"");
        return( lResult);
    }

    if( !lResult)
    {
        RECT rcWnd, rcClnt;

        BDBoard.VW_Size = Size;

        // Create Window with a client area equal to the Video Window
        GetWindowRect( BDBoard.hWnd, &rcWnd);
        GetClientRect( BDBoard.hWnd, &rcClnt);

        ClientToScreen( BDBoard.hWnd, (POINT far *)&rcClnt.left);
        ClientToScreen( BDBoard.hWnd, (POINT far *)&rcClnt.right);

        BDW_MoveVideoWindow( rcClnt.left, rcClnt.top);

        if ( GetSystemMetrics( SM_CXSCREEN) > VGA_RESOLUTION_X)
        {
            MoveWindow( BDBoard.hWnd, bFirst ? 0:rcWnd.left, bFirst ? 0:rcWnd.top,
                (int) (BDBoard.VW_Size.x + (rcClnt.left - rcWnd.left) + ( rcWnd.right - rcClnt.right)),
                (int)(BDBoard.VW_Size.y + (rcClnt.top - rcWnd.top) + ( rcWnd.bottom - rcClnt.bottom)) + BDBoard.StatusBarHeight,
                TRUE);
        }
        else // largest possible window, assume task bar height is equal to the height of the menu
        {
            MoveWindow( BDBoard.hWnd, bFirst ? 0:rcWnd.left, bFirst ? 0:rcWnd.top,
                (int) (BDBoard.VW_Size.x + (rcClnt.left - rcWnd.left) + ( rcWnd.right - rcClnt.right))
                        - (GetSystemMetrics( SM_CXHSCROLL) + 2*GetSystemMetrics( SM_CXDLGFRAME)),
                (int)(BDBoard.VW_Size.y + (rcClnt.top - rcWnd.top) + ( rcWnd.bottom - rcClnt.bottom))
                    -   (GetSystemMetrics(   SM_CYHSCROLL)+   2*GetSystemMetrics(SM_CYMENU)   +   GetSystemMetrics(
SM_CYCAPTION) + 2*GetSystemMetrics( SM_CYDLGFRAME)),
                TRUE);
        }
```

```
        BD_ShowVideoWindow();
        DrawMenuBar(BDBoard.hWnd);    // refresh the menu
    }
    else
    {
        MessageBox(NULL, "Could not resize Video Buffer", NULL, MB_ICONSTOP | MB_OK)
    }

    return( lResult);
}

////////////////////////////////////////////////////////////////////////////
// ReadState
//
// Description:
//        Read the state of the board from memory-mapped file to see if it is being used
//
// Returns:
//        TRUE if successful, FALSE otherwise
//
////////////////////////////////////////////////////////////////////////////
BOOL ReadState(char *Busy)
{
  char * lpView = (char *) MapViewOfFile(hFileMap, FILE_MAP_READ , 0, 0, sizeof (char));
   if (lpView != NULL)
   {
        // Read the state of the board
        memcpy(Busy, (char*) lpView, 1);
        UnmapViewOfFile((char *) lpView);
        return TRUE;
   }
   return FALSE;
}

////////////////////////////////////////////////////////////////////////////
//   WriteState
//
// Description:
//        Update the state of the board by writing 1 to memory-mapped file so that
//        it cannot being used by another application
//
// Returns:
//        TRUE if successful, FALSE otherwise
//
////////////////////////////////////////////////////////////////////////////
BOOL WriteState(char *Busy)
{
  char * lpView = (char *) MapViewOfFile(hFileMap, FILE_MAP_WRITE, 0, 0, sizeof (char));
   if (lpView != NULL)
   {
        // Find the state and update it
        memcpy((char*) lpView, Busy, sizeof (char));
        UnmapViewOfFile((char *) lpView);
        return TRUE;
   }
   return FALSE;
}

////////////////////////////////////////////////////////////////////////////
```

```
//    PrintBanditErrorMessage
//
// Description:
//         Print the message box to indicate the error message number and message
//
//    Input:
//         hwnd:                    handle of the window that displays the message box
//         ErrorMsgNumber:   Bandit driver message number
//         lpCaption:          title of message box, ie. WARNING, ERROR
//
// Returns:
//         nothing
//
////////////////////////////////////////////////////////////////////////////
void PrintBanditErrorMessage(HANDLE hwnd, long ErrorMsgNumber, LPCSTR lpCaption)
{
     char Msg[200];
     char Msg2[255];
     BD_GetErrorMessage( ErrorMsgNumber, Msg, sizeof(Msg));
     wsprintf(Msg2, "%s (ErrorNo=%ld)", Msg, ErrorMsgNumber);
     if (!hwnd)
          MessageBox(GetFocus(), Msg2, lpCaption, MB_OK | MB_ICONSTOP);
     else
          MessageBox(hwnd, Msg2, lpCaption, MB_OK | MB_ICONSTOP);
}

////////////////////////////////////////////////////////////////////////////
// InitializeScrollBar
//
//    Description:
//
//
// Inputs:
//         hwnd:              window handle.
//         xNewSize:     Width of client area
//         yNewSize:     Height of client area
//         bReset:         boolean value to control the reset xCurrentScroll and yCurrentScroll
//
//
// Returns:  nothing
//
// Note: bReset is set to TRUE whenever BD_SetVideoOutputControl is called
//               before calling InitializeScrollBar
//
////////////////////////////////////////////////////////////////////////////
void InitializeScrollBar(HWND hwnd, int xNewSize, int yNewSize, BOOL bReset)
{
     BD_SIZE VWSize;
     LRESULT lResult;
     BD_VW_OUTPUT_CONTROL OutControl;

     lResult = BD_GetVideoWindowOutputControl( &OutControl);

     if (bReset)
     {
          xCurrentScroll = 0;
          yCurrentScroll = 0;
          BDBoard.xCurrentScroll = 0;
          BDBoard.yCurrentScroll = 0;
```

```
            OutControl.VideoRoi.x = 0;
            OutControl.VideoRoi.y = 0;
            lResult = BD_SetVideoWindowOutputControl( &OutControl);
            if (lResult)
            {
                PrintBanditErrorMessage(BDBoard.hWnd, lResult, "ERROR after calling BD_SetVideoWindowOutputControl(), file
\"bdpro.c\"");
                return;
            }
        }

        if (!lResult)
            lResult = BD_GetVideoWindowSize( &VWSize);

        if (lResult)
            return;

        // The horizontal scrolling range is defined by
        // (size of the video window) - (width of client area)
        // The current horizontal scroll value remains within the
        // horizontal scrolling range.

        if (BDBoard.bFitToScreen || (BDBoard.bFullScreen && GetSystemMetrics(SM_CXSCREEN) == VGA_RESOLUTION_X))
            xMaxScroll = 0;     // remove scrolling bar if fit to screen
        else
            xMaxScroll = max(VWSize.x - xNewSize/BDBoard.wZoomFactor, 0);

        xCurrentScroll = min(xCurrentScroll, xMaxScroll);
        si.cbSize = sizeof(si);
        si.fMask = SIF_RANGE | SIF_POS;

        si.nMin = xMinScroll;
        si.nMax = xMaxScroll;
        si.nPage = 0;
        si.nPos = xCurrentScroll;
        SetScrollInfo (hwnd, SB_HORZ, &si, TRUE);

        // The vertical scrolling range is defined by
        // (size of the video window) - (width of client area)
        // The current vertical scroll value remains within the
        // vertical scrolling range.

        if (BDBoard.bFitToScreen || (BDBoard.bFullScreen && GetSystemMetrics(SM_CXSCREEN) == VGA_RESOLUTION_X))
            yMaxScroll = 0;     // remove scrolling bar if fit-to-screen
        else
        {
            yNewSize = yNewSize - BDBoard.StatusBarHeight;
            yMaxScroll = max(VWSize.y - yNewSize/BDBoard.wZoomFactor, 0);
        }

        yCurrentScroll = min(yCurrentScroll, yMaxScroll);
        si.cbSize = sizeof(si);
        si.fMask = SIF_RANGE | SIF_POS;
        si.nMin = yMinScroll;
        si.nMax = yMaxScroll;
        si.nPage = 0;
        si.nPos = yCurrentScroll;
        SetScrollInfo (hwnd, SB_VERT, &si, TRUE);
```

```
    // calculate the horizontal and vertical increment
    HorizontalStepIncrement = 4*GRANUL_X;
    VerticalStepIncrement = 4*GRANUL_Y;
}


////////////////////////////////////////////////////////////////////////////
// ResetVideoOutputWindow
//
//   Description:
//              action to be done when the user switches from !bFitToScreen to bFitToScreen
//
// Note:
//        When Not FitToScreen is used, the horizontal and vertical bar have modified the valu
//        of Video Output Control VideoRoi.x and VideoRoi.y
//
// Inputs:
//        Nothing
//
// Returns:  error code of BD_GetVideoWindowInputControl or BD_GetVideoWindowOutputContr
//
//
////////////////////////////////////////////////////////////////////////////

LRESULT ResetVideoOutputWindow()
{
    LRESULT lResult;
    BD_VW_OUTPUT_CONTROL OutControl;
    BD_VW_INPUT_CONTROL InControl;

    if (!BDBoard.bFitToScreen)
        return FALSE;

    BDBoard.wZoomFactor = 1;

    // Get current output control
    lResult = BD_GetVideoWindowOutputControl( &OutControl);

    if( !lResult)
        lResult = BD_GetVideoWindowInputControl( &InControl);

    if( !lResult)
    {
        // Set the display rectangle to the desired size
        OutControl.VideoRoi.x = InControl.VideoRoi.x;
        OutControl.VideoRoi.y = InControl.VideoRoi.y;
    }

    lResult = _BD_SetVideoWindowOutputControl( &OutControl);

    return lResult;
}


////////////////////////////////////////////////////////////////////////////
// SetZoomIn / SetZoomOut
//
//   Description:
//              Zoom In / Zoom out by changing Video Window Output Control
```

```
//
// Inputs:
//        iInc:       value for speeding up the scrolling (large value for Home, End, PageUp, PageDown key
//
//   Note:
//        Used only in fullscreen mode
//
// Returns:  error code of BD_GetVideoWindowInputControl or BD_GetVideoWindowOutputControl
//
////////////////////////////////////////////////////////////////////////////////

LRESULT SetZoomIn()
{
    BD_VW_OUTPUT_CONTROL OutControl;
    LRESULT lResult;

    lResult = BD_GetVideoWindowOutputControl( &OutControl);

    if (!lResult)
    {
        OutControl.VideoRoi.xlen = (OutControl.VideoRoi.xlen >> 1);
        OutControl.VideoRoi.ylen = (OutControl.VideoRoi.ylen >> 1);
        lResult = _BD_SetVideoWindowOutputControl( &OutControl);
    }

    return lResult;
}


LRESULT SetZoomOut()
{
    BD_VW_OUTPUT_CONTROL OutControl;
    LRESULT lResult;
    lResult = BD_GetVideoWindowOutputControl( &OutControl);

    if (!lResult)
    {
        OutControl.VideoRoi.xlen = (OutControl.VideoRoi.xlen << 1);
        OutControl.VideoRoi.ylen = (OutControl.VideoRoi.ylen << 1);
        lResult = _BD_SetVideoWindowOutputControl( &OutControl);
    }

    return lResult;
}

////////////////////////////////////////////////////////////////////////////////
// SetZoomRightArrow, SetZoomLeftArrow, SetZoomUpArrow, SetZoomDownArrow
//
//   Description:
//              Scroll image right / left / Up / Down (corresponding to the function names)
//
// Inputs:
//        iInc:       value for speeding up the scrolling
//                        large value for Home, End, PageUp, PageDown keys
//                        small value for Up, Down, Right, Left arrow keys
//
//   Note:
//        Used only in fullscreen mode
//
```

```
// Returns:  error code of BD_GetVideoWindowInputControl or BD_GetVideoWindowOutputCont
//
////////////////////////////////////////////////////////////////////////////

LRESULT SetZoomRightArrow( DWORD iInc)
{
    BD_VW_OUTPUT_CONTROL OutControl;
    LRESULT lResult;
    lResult = BD_GetVideoWindowOutputControl( &OutControl);

    if (!lResult)
    {
        OutControl.VideoRoi.x += (iInc*GRANUL_X);
        lResult = _BD_SetVideoWindowOutputControl( &OutControl);
    }

    UpdateWindow(BDBoard.hWnd);

    return lResult;
}

LRESULT SetZoomLeftArrow( DWORD iInc)
{
    BD_VW_OUTPUT_CONTROL OutControl;
    LRESULT lResult;

    lResult = BD_GetVideoWindowOutputControl( &OutControl);

    if (OutControl.VideoRoi.x < (iInc*GRANUL_X))
        return FALSE;

    if (!lResult)
    {
        OutControl.VideoRoi.x -= (iInc*GRANUL_X);
        lResult = _BD_SetVideoWindowOutputControl( &OutControl);
    }

    UpdateWindow(BDBoard.hWnd);

    return lResult;
}

LRESULT SetZoomUpArrow( DWORD iInc)
{
    BD_VW_OUTPUT_CONTROL OutControl;
    LRESULT lResult;

    lResult = BD_GetVideoWindowOutputControl( &OutControl);

    if (!lResult)
    {
        OutControl.VideoRoi.y += (iInc*GRANUL_Y);
        lResult = _BD_SetVideoWindowOutputControl( &OutControl);
    }

    UpdateWindow(BDBoard.hWnd);

    return lResult;
}
```

```
LRESULT SetZoomDownArrow( DWORD iInc)
{
    BD_VW_OUTPUT_CONTROL OutControl;
    LRESULT lResult;

    lResult = BD_GetVideoWindowOutputControl( &OutControl);

    if (OutControl.VideoRoi.y < (iInc*GRANUL_Y))
        return FALSE;

    if (!lResult)
    {
        OutControl.VideoRoi.y -= (iInc*GRANUL_Y);
        lResult = _BD_SetVideoWindowOutputControl( &OutControl);
    }

    UpdateWindow(BDBoard.hWnd);

    return lResult;
}

void FitScreen(HWND hWnd)
{
    RECT rcClient;
    GetClientRect(hWnd, &rcClient);
    if (BDBoard.bFitToScreen)
    {
        BDBoard.bFitToScreen = FALSE;
        CheckMenuItem(BDBoard.hMenu, ID_OPTIONS_DISPLAY_FITTOSCREEN, MF_UNCHECKED | MF_BYCOMMAND);
        EnableMenuItem(BDBoard.hMenu, ID_OPTIONS_DISPLAY_ZOOM, MF_ENABLED | MF_BYCOMMAND);
        EnableMenuItem(BDBoard.hMenu, IDM_POSITION_ROIS, MF_ENABLED | MF_BYPOSITION);
    }
    else
    {

        BDBoard.bFitToScreen = TRUE;
        ResetVideoOutputWindow();
        CheckMenuItem(BDBoard.hMenu, ID_OPTIONS_DISPLAY_FITTOSCREEN, MF_CHECKED | MF_BYCOMMAND);
        EnableMenuItem(BDBoard.hMenu, ID_OPTIONS_DISPLAY_ZOOM, MF_GRAYED | MF_DISABLED | MF_BYCOMMAND);
        EnableMenuItem(BDBoard.hMenu, 4, MF_GRAYED | MF_DISABLED | MF_BYPOSITION);
        SetAcqRoiToDefault(BDBoard.hWnd);
    }
        // Update window, title bar, status bar
    DrawMenuBar(hWnd);    // refresh the menu
    SendMessage( hWnd, WM_SIZE, 0, MAKELPARAM( (WORD) (rcClient.right - rcClient.left), (WORD) (rcClient.bottom - rcClient.top)
));

}



#include <windows.h>
#include <windowsx.h>
#include <string.h>
#include <memory.h>
#include <commdlg.h>
#include <stdio.h>
```

```
#include "bdapi.h"
#include "global.h"
#include "imgapi.h"
#include "bdpro.h"
#include "image.h"

// Function prototypes
static void BuildFilter(char *szFilter, UINT bufLen);
static BOOL TrimImage(BYTE *lpYUVbuffer, IMGSIZE Destination, IMGSIZE Source);
static BOOL ReAdjustVWOutput(HWND hwnd, BYTE *lpYUVbuffer, BD_WINDOW * VW, BD_WINDOW * BW, FRAMESIZEINFO
sFrameSizeInfo);
static BOOL GetPosition(HWND hwnd, BYTE* lpYUVbuffer, FRAMESIZEINFO sFrameSizeInfo);


////////////////////////////////////////////////////////////////////////////
// SaveImage
//
//   Description:
//        Saves an image grabbed in the current device. It uses a "saveas" common dialog box
//
//   Parameters:
//        hwnd          Handle of the parent window
//
//   Returns:
//        TRUE if successful, FALSE otherwise
////////////////////////////////////////////////////////////////////////////
BOOL SaveImage(HWND hwnd)
    {

    BOOL                retVal = TRUE;
    OPENFILENAME   file;
    char                szFilter[256],        // file filter
                        szFile[256],          // file name
                        szExt[4] = "tif",     // default extension
                        *pExt;                // pointer to the file extension

    DWORD dwSize;
    DWORD dwFormat;
    BD_WINDOW VW;          // ROI in Video Window
    BD_WINDOW BW;          // ROI in System Buffer
    void *lpYUVbuffer;
    void *lpRGBbuffer;
    FRAMECAPINFO sFrameCapInfo;
    FILE *fp;
    int i=1;


    memset(&file, 0, sizeof(OPENFILENAME)); // initialize the structure with 0s

    file.lStructSize = sizeof(OPENFILENAME);
    file.hwndOwner = hwnd;

    // prepare the filter
    BuildFilter(szFilter, sizeof(szFilter));

    file.lpstrFilter = szFilter; // set the filters

    szFile[0] = '\0';
    file.lpstrFile = "image.bmp";//szFile;
```

```
file.nMaxFile = sizeof(szFile);
file.Flags = OFN_OVERWRITEPROMPT | OFN_PATHMUSTEXIST | OFN_HIDEREADONLY;
file.lpstrDefExt = szExt;

// get file name
if (1)//GetSaveFileName(&file))
    {
        SetCursor(LoadCursor(NULL, IDC_WAIT));

        // Read Video Window
        VW.x = 0;
        VW.y = 0;
        VW.xlen =  BDBoard.VW_Size.x;
        VW.ylen =  BDBoard.VW_Size.y;

        BW.x = 0;
        BW.y = 0;
        BW.xlen = BDBoard.VW_Size.x;
        BW.ylen = BDBoard.VW_Size.y;


        dwSize = BDBoard.VW_Size.x * BDBoard.VW_Size.y; // full

        BD_GetVideoWindowFormat( &dwFormat);

        if( dwFormat != BD_FMT_CCIR_422)
            return( FALSE);

        // Allocate memory for the system memory buffer that will
        // receive the video data transferred from the frame buffer.
        lpYUVbuffer = GlobalAllocPtr(GHND, dwSize*2);
        if( !lpYUVbuffer)
        {
            return(FALSE);
        }

        if( BD_ReadVideoWindow( lpYUVbuffer, BDBoard.VW_Size.x, &BW, &VW))
            return( FALSE);

        lpRGBbuffer = GlobalAllocPtr(GHND, dwSize*3); // 4 bytes of YUV vs 6 bytes of RGB
        if( !lpRGBbuffer)
        {
            return(FALSE);
        }

        if( BD_ConvertBuffer( lpYUVbuffer, dwFormat, lpRGBbuffer, BD_FMT_RGB_888_24, dwSize
            return( FALSE);

                        fp=fopen("image.txt","w");

                        while(i<100){
                        fprintf(fp,"%2x  ",lpRGBbuffer);
                        i++;
                        }
                        fclose(fp);
        // Fill up the structure of information to send to IMGAPI.DLL
        sFrameCapInfo.wFromWidth = (WORD) BDBoard.VW_Size.x;
        sFrameCapInfo.wToWidth = (WORD) BDBoard.VW_Size.x;
        sFrameCapInfo.wFromHeight = (WORD) BDBoard.VW_Size.y;
```

```
            sFrameCapInfo.wToHeight = (WORD) BDBoard.VW_Size.y;
            strcpy(sFrameCapInfo.FileFullName, file.lpstrFile);

            // find the file extension
            pExt = strrchr((char *) szFile, (int) '.');

            if(stricmp(pExt, ".tif") == 0)
                {
                sFrameCapInfo.wFileFormat=TIF_24BIT;
                retVal = WriteTIFFile(sFrameCapInfo, lpRGBbuffer);
                }
            else if(stricmp(pExt, ".bmp") == 0)
                {
                sFrameCapInfo.wFileFormat = BMP_24BIT;
                retVal = WriteBMPFile(sFrameCapInfo, lpRGBbuffer);
                }

            else if(stricmp(pExt, ".tga") == 0)
                {
                sFrameCapInfo.wFileFormat = TGA_24BIT;
                retVal = WriteTGAFile(sFrameCapInfo, lpRGBbuffer);
                }
            else
                MessageBox(hwnd, "Unrecognized File format.", "WARNING", MB_OK | MB_ICONEXCLAMATION

            if (lpYUVbuffer)
            {
                GlobalUnlockPtr( lpYUVbuffer);
                GlobalFree( GlobalPtrHandle( lpYUVbuffer));
            }

            if ( lpRGBbuffer)
            {
                GlobalUnlockPtr( lpRGBbuffer);
                GlobalFree( GlobalPtrHandle( lpRGBbuffer));
            }

            if( !retVal)
                MessageBox(hwnd, "Could not save image", "WARNING", MB_OK | MB_ICONEXCLAMATION);
            else
            {
                char buf[288];
                wsprintf(buf, "File %s was saved successfully.", file.lpstrFile);
                MessageBox(hwnd, buf, "Image Saving", MB_OK);
            }

        }
        return retVal;
    }


////////////////////////////////////////////////////////////////////////////
// OpenImage
//
// Description:
//      This routine loads a file image in the video windows. It uses a "open"
//      common dialog box.
//
//  Parameters:
```

```
//              hwnd            Handle of the parent window.
//
//  Returns:
//      TRUE if successful, FALSE otherwise.
////////////////////////////////////////////////////////////////////////////

BOOL OpenImage(HWND hwnd)
    {
    BOOL                retVal = TRUE;
    OPENFILENAME   file;
    char                szFilter[256],      // file filter
                        szFile[256],        // file name
                        szExt[4] = "bmp";   // default extension

    BYTE                *lpYUVbuffer;       // pointer to buffer to hold YUV image
    DWORD dwSize;
    DWORD dwFormat;
    FRAMESIZEINFO  sFrameSizeInfo;
    HCURSOR     hCursor;

    memset(&file, 0, sizeof(OPENFILENAME)); // initialize the structure with 0s

    file.lStructSize = sizeof(OPENFILENAME);
    file.hwndOwner = hwnd;

    // prepare the filter
    BuildFilter(szFilter, sizeof(szFilter));

    // set the filters
    file.lpstrFilter = szFilter;

    szFile[0] = '\0';
    file.lpstrFile= "f:\\bandit\\test\\image.bmp";//szFile;
    file.nMaxFile = sizeof(szFile);
    file.Flags = OFN_PATHMUSTEXIST | OFN_HIDEREADONLY;
    file.lpstrDefExt = szExt;

    // get file name
    if (1)//GetOpenFileName(&file))
        {
            hCursor = SetCursor(LoadCursor(NULL, IDC_WAIT));

            // Read file image and Write to Video Window

            // Make sure the signal will be YUV422
            BD_GetVideoWindowFormat( &dwFormat);
            if( dwFormat != BD_FMT_CCIR_422)
                retVal = FALSE;

            // Get the size of the file image
            if (retVal)
            {
                if (!GetImageSize (hwnd,file.lpstrFile, &sFrameSizeInfo))
                {
                    MessageBox(hwnd, "Cannot get Image size", NULL, MB_OK | MB_ICONEXCLAMATION
                    retVal = FALSE;
                }
            }
```

```
        // Abort if the size of the image is smaller than the video memory of the bandit board
        if (retVal)
        {
            if ((sFrameSizeInfo.wImageWidth > (WORD) BDBoard.VW_Size.x) || (sFrameSizeInfo.wImageHeight > (WORD
BDBoard.VW_Size.y))
            {
                char buf[100];
                wsprintf(buf, "Image is larger than the Video Window (%ldx%ld) size of the board.", BDBoard.VW_Size.x
BDBoard.VW_Size.y);
                MessageBox(hwnd, buf, NULL, MB_OK | MB_ICONSTOP );
                return FALSE;
            }
            else
                // Calculate the size of the file image
                dwSize = sFrameSizeInfo.wImageWidth * sFrameSizeInfo.wImageHeight;
        }

        if (retVal)
        {
            // Allocate memory for YUV according to the size of the file image
            lpYUVbuffer = GlobalAllocPtr(GHND, dwSize*2);
            if (!lpYUVbuffer)
                retVal = FALSE;
            else
                // initialized with 0s
                memset(lpYUVbuffer, 0, sizeof(dwSize*2));
        }

        if (retVal)
        {
            // Allocate memory for RGB according to the size of the file image
            lpRGBbuffer = GlobalAllocPtr(GHND, dwSize*3);
            if (!lpRGBbuffer)
                retVal = FALSE;
            else
                // initialized with 0s
                memset(lpRGBbuffer, 0, sizeof(dwSize*3));
        }

        if (retVal)
        {
            // Get image from file
            retVal = GetImage(hwnd, file.lpstrFile, lpRGBbuffer);
        }

        if (retVal)
        {
            // conversion RGB to YUV422 and display
            if( BD_ConvertBuffer( lpRGBbuffer, BD_FMT_RGB_888_24, lpYUVbuffer, BD_FMT_CCIR_422, dwSize))
                retVal = FALSE;
        }

        // Waiting for user to click left mouse button
        // That is the location where the image is placed
        if (retVal)
            GetPosition(hwnd, lpYUVbuffer, sFrameSizeInfo);

        if (lpYUVbuffer)
        {
```

```
                GlobalUnlockPtr( lpYUVbuffer);
                GlobalFree(GlobalPtrHandle( lpYUVbuffer));
            }
            if (lpRGBbuffer)
            {
                GlobalUnlockPtr( lpRGBbuffer);
                GlobalFree(GlobalPtrHandle( lpRGBbuffer));
            }

        }
    SetCursor( hCursor);
    return retVal;
    }

////////////////////////////////////////////////////////////////////////////
// BuildFilter
//
// Description:
//      Builds the filter needed by the function SaveImage and OpenImage.
//
//  Parameters:
//  szFilter        Buffer that receives the string
//  bufLen          Buffer length
//
//  Returns:
//      none
//
////////////////////////////////////////////////////////////////////////////
static void BuildFilter(char *szFilter, UINT bufLen)
{
    char p[] = "All Files\0*.tif;*.tga;*.bmp\0TIFF Files\0*.tif\0TGA Files\0*.tga\0BMP Files\0*.bmp\0"; // temporary pointe

    memcpy(szFilter, p, sizeof(p) < bufLen ? sizeof(p) : bufLen);
}

////////////////////////////////////////////////////////////////////////////
//
// SaveImageRect
//
//  Description:
//      saves a "rectangle image" defined on the screen. It uses a "saveas" common dialog box
//
//
//  Parameters:
//      hwnd            Handle of the parent window
//      rc              pointer to rectangle of image to be saved
//
//  Returns:
//      TRUE if successful, FALSE otherwise
//
////////////////////////////////////////////////////////////////////////////
BOOL SaveImageRect(HWND hwnd, RECT *rc, LPARAM lParam)
    {
    BOOL                retVal = TRUE;
    OPENFILENAME   file;
    char                szFilter[256],      // file filter
                        szFile[256],        // file name
                        szExt[4] = "bmp",   // default extension
                        *pExt;              // pointer to the file extension
```

```
    DWORD dwSize;
    DWORD dwFormat;
    BD_WINDOW VW;           // ROI in Video Window
    BD_WINDOW BW;           // ROI in System Buffer
    float p_scale_x, p_scale_y;
    void *lpYUVbuffer;
    void *OLDlpRGBbuffer;
    FRAMECAPINFO sFrameCapInfo;
    FILE *fp;
    long i,j, flen;
    BYTE                *tmpRGBbuffer;          // pointer to buffer to hold RGB image
    RECT rcClient;
    memset(&file, 0, sizeof(OPENFILENAME)); // initialize the structure with 0s

    file.lStructSize = sizeof(OPENFILENAME);
    file.hwndOwner = hwnd;

    // prepare the filter
    BuildFilter(szFilter, sizeof(szFilter));

    file.lpstrFilter = szFilter; // set the filters

    szFile[0] = '\0';
    file.lpstrFile = "image.bmp";
    file.nMaxFile = sizeof(szFile);
    file.Flags = OFN_OVERWRITEPROMPT | OFN_PATHMUSTEXIST | OFN_HIDEREADONLY;
    file.lpstrDefExt = szExt;

    // force the width of the saved image to be an even number and also the starting position to capture
    // is an even number since IMGAPI does not support odd width pixel and YUV signal always need 2 pix
    // to get the right color

/*  GetClientRect(hwnd, &rcClient); // get client are for repainting purpose
    p_scale_x=(float)((float)(K_Size.x)/(float)(rcClient.right));
    p_scale_y=(float)((float)(K_Size.y)/(float)(rcClient.bottom));
    rc->left = rc->right = (LONG)((float)LOWORD (lParam)*p_scale_x)-CAP_Size.x; // in client coordinates
    rc->top  = rc->bottom= (LONG)((float)HIWORD (lParam)*p_scale_y)-CAP_Size.y;
    if(rc->left<0)rc->left=0;if(rc->top<0)rc->top=0;
    rc->right =  (LONG)((float)LOWORD (lParam)*p_scale_x)+CAP_Size.x;
    rc->bottom = (LONG)((float)HIWORD (lParam)*p_scale_y)+CAP_Size.y;
    if(rc->right> (LONG)((float)rcClient.right*p_scale_x)) rc->right= (LONG)((float)rcClient.right*p_scale_x);
    if(rc->bottom>(LONG)((float)rcClient.bottom*p_scale_y))rc->bottom=(LONG)((float)rcClient.bottom*p_scale_y

    if (rc->right%2)
        (rc->right < (WORD) BDBoard.VW_Size.x)? (rc->right++) : rc->right--;
    while (rc->left%2)
        rc->left++;
*/
    // get file name
    if (1)//GetSaveFileName(&file))
        {
            SetCursor(LoadCursor(NULL, IDC_WAIT));

            // Read Video Window
            VW.x = 0;//rc->left;
            VW.y = 0;//rc->top;
            VW.xlen =  640;//rc->right - rc->left;
            VW.ylen =  480;//rc->bottom - rc->top;
```

```
BW.x = 0;
BW.y = 0;
BW.xlen = VW.xlen;
BW.ylen = VW.ylen;

dwSize = VW.xlen * VW.ylen;

BD_GetVideoWindowFormat( &dwFormat);

if( dwFormat != BD_FMT_CCIR_422)
     return( FALSE);

// Allocate memory for the system memory buffer that will
// receive the video data transferred from the frame buffer.
lpYUVbuffer = GlobalAllocPtr(GHND, dwSize*2);
if( !lpYUVbuffer)
{
     return(FALSE);
}

if( BD_ReadVideoWindow( lpYUVbuffer, VW.xlen, &BW, &VW))
{
     retVal = FALSE;
}

lpRGBbuffer = GlobalAllocPtr(GHND, dwSize*3); // 4 bytes of YUV vs 6 bytes of RGB
if( !lpRGBbuffer)
{
     return(FALSE);
}

if( BD_ConvertBuffer( lpYUVbuffer, BD_FMT_CCIR_422, lpRGBbuffer, BD_FMT_RGB_888_24, dwSize))
     return( FALSE);

          fp=fopen("image.txt","w");
                    tmpRGBbuffer=OLDlpRGBbuffer=lpRGBbuffer;
                    flen=VW.xlen*VW.ylen;
                    for(i=0;i<VW.ylen;i++){
                         for(j=0;j<VW.xlen;j++){
                              rgb[i][j].b=*(tmpRGBbuffer);
                              rgb[i][j].g=*(tmpRGBbuffer+1);
                              rgb[i][j].r=*(tmpRGBbuffer+2);
                              fprintf(fp,"{%2x,%2x,%2x}",*(tmpRGBbuffer),*(tmpRGBbuffer+1),*(tmpRGBbuffer+2
                              //if(rgb[i][j].b>=0x5f)     *(lpRGBbuffer   )=0xff;
                              //else                           *(lpRGBbuffer   )=0x00;
                              //if(rgb[i][j].g>=0x5f)     *(lpRGBbuffer+1)=0xff;
                              //else                           *(lpRGBbuffer+1)=0x00;
                              //if(rgb[i][j].r>=0x5f)     *(lpRGBbuffer+2)=0xff;
                              //else                           *(lpRGBbuffer+2)=0x00;
                              *(lpRGBbuffer   )=rgb[i][j].b;
                              *(lpRGBbuffer+1)=rgb[i][j].g;
                              *(lpRGBbuffer+2)=rgb[i][j].r;
                              lpRGBbuffer+=3;
                              tmpRGBbuffer+=3;
                         }
                         fprintf(fp,"\n");
                    }
                    fclose(fp);
                    lpRGBbuffer=OLDlpRGBbuffer;
```

```
            // Fill up the structure of information to send to IMGAPI.DLL
            sFrameCapInfo.wFromWidth = (WORD) VW.xlen;
            sFrameCapInfo.wToWidth =  (WORD) VW.xlen;
            sFrameCapInfo.wFromHeight = (WORD) VW.ylen;
            sFrameCapInfo.wToHeight = (WORD) VW.ylen;
            strcpy(sFrameCapInfo.FileFullName, file.lpstrFile);

            // find the file extension
            pExt = strrchr((char *) szFile, (int) '.');

            //if(stricmp(pExt, ".tif") == 0)
            //    {
            //    sFrameCapInfo.wFileFormat=TIF_24BIT;
            //    retVal = WriteTIFFile(sFrameCapInfo, lpRGBbuffer);
            //    }
            //else if(stricmp(pExt, ".bmp") == 0)
            //    {
                    sFrameCapInfo.wFileFormat=BMP_24BIT;
                    retVal = WriteBMPFile(sFrameCapInfo, lpRGBbuffer);
            //    }

//          else if(stricmp(pExt, ".tga") == 0)
//                  {
//                  sFrameCapInfo.wFileFormat=TGA_24BIT;
//                  retVal = WriteTGAFile(sFrameCapInfo, lpRGBbuffer);
//                  }
//          else
//                  MessageBox(hwnd, "Unrecognized File format.", "WARNING", MB_OK | MB_ICONEXCLAMATION)


            GlobalUnlockPtr( lpYUVbuffer);
            GlobalFree(GlobalPtrHandle( lpYUVbuffer));

            GlobalUnlockPtr( lpRGBbuffer);
            GlobalFree(GlobalPtrHandle( lpRGBbuffer));

            if( !retVal)
                MessageBox(hwnd, "Could not save image", "WARNING", MB_OK | MB_ICONEXCLAMATION);
            else
            {
            //  char buf[_MAX_PATH];
            //  wsprintf(buf, "File %s was saved successfully.", file.lpstrFile);
            //  MessageBox(hwnd, buf, "Image Saving", MB_OK);
            }

        }
        return retVal;
    }


////////////////////////////////////////////////////////////////////////////
// TrimImage
//
//  Description:
//      Trims the image according to the destination image size
//
//
//  Input:
//      Source              Structure contains the size of the source file image
```

```
//        Destination          Structure contains the size of the destination rectangle where the image
//                                  is placed
//
//    Output:
//        lpYUVbuffer          Pointer contains the file image being trimmed by the destination
//                                  rectangle
//        Source:              size of the image want to be displayed (file image)
//        Destination:     size of the image to be displayed (on the screen, in most cases,
//                                  it will be small than Source size
//
//    Note:
//        lpYUVbuffer will be updated according to the size of the destination image size
//
//    Returns:
//        TRUE if successful, FALSE otherwise.
//
////////////////////////////////////////////////////////////////////////////////

BOOL TrimImage(BYTE *lpYUVbuffer, IMGSIZE Source, IMGSIZE Destination)
{
    WORD i, j, k;
    WORD wWidth;                    // difference of width between source and destination size
                                        // the data of this part will be discarded
    WORD wLoopWidth;                // control the width of data to be kept
    WORD wLoopHeight;               // control the height of data to be kept
    BOOL retVal=TRUE;
    BYTE *lpYUVbufferTmp;       // a working copy of lpYUVbuffer
    BYTE *lpTmp;
    BYTE *lpSave;


    // Validation
    if (Source.wWidth > Destination.wWidth)
    {
        wWidth = (WORD) (Source.wWidth - Destination.wWidth); // use type casting for Borland
        wLoopWidth = Destination.wWidth;
        if (Source.wHeight > Destination.wHeight)
            wLoopHeight = Destination.wHeight;
        else
            wLoopHeight = Source.wHeight;
    }
    else
    {
        wWidth = 0;
        wLoopWidth = Source.wWidth;
        if (Source.wHeight > Destination.wHeight)
            wLoopHeight = Destination.wHeight;
        else
            wLoopHeight = Source.wHeight;
    }

    // Allocate memory for temporary working pointer
    if (retVal)
    {
        lpTmp=GlobalAllocPtr(GHND, (wLoopWidth+wWidth)*wLoopHeight*2);
        if (!lpTmp)
        {
            MessageBox(GetFocus(), "Could not allocate memory", NULL, MB_OK | MB_ICONSTOP
            retVal = FALSE;
```

```
        }
        else
        {
            lpSave = lpTmp;                          // make a copy of pointer for deallocate late
            lpYUVbufferTmp = lpYUVbuffer;        // make a copy of pointer to file image size
        }
    }

    if (retVal)
    {
        // scan the lines of the image
        for (i=0; i<wLoopHeight; i++)
        {
            // Signal is YUV, so we need 2 bytes per pixel
            for (j=0; j<wLoopWidth*2; j++)
            {
                *lpTmp++ = *lpYUVbufferTmp++;
            }
            for (k=0; k<wWidth*2; k++)
                // increment the pointer only, data are not needed
                lpYUVbufferTmp++;
        }
    }

    if (retVal)
    {
        // update data
        lpTmp = lpSave;
        for (i=0; i<wLoopHeight; i++)
            for (j=0; j<wLoopWidth*2; j++)
                    *lpYUVbuffer++ = *lpTmp++;
    }

    GlobalUnlockPtr( lpSave);
   GlobalFree(GlobalPtrHandle( lpSave));

    return retVal;
}


////////////////////////////////////////////////////////////////////////////
// ReAdjustVWOutput
//
//    Description:
//        positions the image in the Frame buffer, not based on what we can see on the screen
//
//    Input:
//        Source              Structure contains the size of the source file image
//        Destination         Structure contains the size of the destination rectangle where the image
//                                is placed
//
//    Output:
//        lpYUVbuffer         Pointer contains the file image
//
//    Returns:
//        TRUE if successful, FALSE otherwise.
//
////////////////////////////////////////////////////////////////////////////
```

```
BOOL ReAdjustVWOutput(HWND hwnd, BYTE *lpYUVbuffer, BD_WINDOW * VW, BD_WINDOW * BW, FRAMESIZEINFO
sFrameSizeInfo)
{

    BD_VW_OUTPUT_CONTROL VWO;
    BOOL bTrim;
    BOOL retVal = TRUE;

    BD_GetVideoWindowOutputControl(&VWO);

    if (VWO.VideoZoom.x != 0)
    {
        VW->x = (VWO.VideoRoi.xlen*VW->x)/VWO.VideoZoom.x;
    }

    if (VWO.VideoZoom.y != 0)
    {
        VW->y = (VWO.VideoRoi.ylen*VW->y)/VWO.VideoZoom.y;
    }

    // force VW->x to be even, otherwise may get wrong color (YUYV video signal)
    if (VW->x%2)
        VW->x--;

    bTrim = (((VW->x + sFrameSizeInfo.wImageWidth) > VWO.VideoRoi.xlen) ||
                ((VW->y + sFrameSizeInfo.wImageHeight) > VWO.VideoRoi.ylen ));

    if (bTrim)
    {
         IMGSIZE Dst, Src;

         Dst.wWidth = (WORD) ( BDBoard.VW_Size.x - VW->x);
         Dst.wHeight = (WORD) (BDBoard.VW_Size.y - VW->y);

         Src.wWidth = sFrameSizeInfo.wImageWidth;
         Src.wHeight = sFrameSizeInfo.wImageHeight;
         if (!TrimImage(lpYUVbuffer, Src, Dst))
         {
              MessageBox(hwnd, "Failed to trim the image", "ERROR", MB_OK | MB_ICONSTOP);
              retVal = FALSE;
         }
    }

    BW->x = 0;
    BW->y = 0;
    if (bTrim)
    {
        if ((BDBoard.VW_Size.x - VW->x) < (DWORD) sFrameSizeInfo.wImageWidth)
            VW->xlen = BW->xlen = BDBoard.VW_Size.x - VW->x;
        else
            VW->xlen = BW->xlen = sFrameSizeInfo.wImageWidth;

        if ((BDBoard.VW_Size.y - VW->y) < (DWORD) sFrameSizeInfo.wImageHeight)
            VW->ylen = BW->ylen = BDBoard.VW_Size.y - VW->y;
        else
            VW->ylen = BW->ylen = sFrameSizeInfo.wImageHeight;
    }
    else
    {
```

```
        VW->xlen = BW->xlen = sFrameSizeInfo.wImageWidth;
        VW->ylen = BW->ylen = sFrameSizeInfo.wImageHeight;
    }

    return retVal;
}


////////////////////////////////////////////////////////////////////////////
// GetPosition
//
// Description:
//      Find the coordinate to place the image
//
//  Parameters:
//          hwnd                    Handle of the parent window.
//          lpYUVbuffer:    pointer to YUV Video Window buffer
//          sFrameSizeInfo:  image frame information
//
// Note:
//      For more efficient repainting, calculate the size of the rectangle, make it
//      larger by 1 pixel, and send message WM_PAINT or call UpdateWindow(hwnd)
//
//  Returns:
//      TRUE if successful, FALSE otherwise.
////////////////////////////////////////////////////////////////////////////

BOOL GetPosition(HWND hWnd, BYTE* lpYUVbuffer, FRAMESIZEINFO sFrameSizeInfo)
{
    LRESULT lResult;
    BOOL ok = TRUE;             // boolean value to control the while loop
    MSG msg;                    // Windows message
  static RECT  rc;              // Application's Window Rectangle
    BD_WINDOW VW;               // ROI in Video Window
    BD_WINDOW BW;               // ROI in System Buffer
    BOOL retVal = TRUE;
    static POINT ptBeg;
    static POINT ptEnd;
    static BOOL bFirst=TRUE;  // boolean value to avoid repaint the screen for the first time

    if (!lpYUVbuffer)
        return FALSE;

    VW.x = 0;
    VW.y = 0;

    SetCapture( hWnd);          // get focus of the mouse

    // Get the Application Window rectangle not including the status bar (if enabled)
    BD_GetClientRect(hWnd, &rc);
    ClientToScreen( hWnd, (POINT*) &rc.left);
    ClientToScreen( hWnd, (POINT*) &rc.right);

    ClipCursor(&rc);

    // Waiting for user to click left mouse button
    // That is the location where the image is placed
        while (ok)
        {
```

```
// Get Client rectangle for repaint
BD_GetClientRect(hWnd, &rc);
SetCursor(LoadCursor(NULL, IDC_CROSS));
GetMessage( &msg, NULL, 0, 0);
switch( msg.message)
{
    case WM_LBUTTONDOWN:
        {
            // Get the position where to put the image
            VW.x = LOWORD (msg.lParam);
            VW.y = HIWORD (msg.lParam);
            // force the position to a valid position, 4 pixels
            if (VW.x%2)
                VW.x++;
            ok = FALSE;
        }
        break;

    case WM_MOUSEMOVE:
        {
            BD_VW_OUTPUT_CONTROL OutControl;
            POINT pt;    // used for VideoRoi.x and VideoRoi.y cursor display

            BD_GetVideoWindowOutputControl (&OutControl);

            pt.x = LOWORD(msg.lParam) + OutControl.VideoRoi.x;
            pt.y = HIWORD(msg.lParam) + OutControl.VideoRoi.y;

            ptBeg.x = LOWORD (msg.lParam);
            ptBeg.y = HIWORD (msg.lParam);
            ptEnd.x = ptBeg.x + sFrameSizeInfo.wImageWidth;
            ptEnd.y = ptBeg.y + sFrameSizeInfo.wImageHeight;

            // Do not update the screen for the first time
            if (!bFirst)
            {
                InvalidateRect(hWnd, &rc, TRUE);
                SendMessage(hWnd, WM_PAINT, 0, 0L);
            }
            else
                bFirst=FALSE;

            // BDBoard.wZoomFactor has value 1, 2 and 4 for x1, x2, x4 Zoom facto
            ptEnd.x += sFrameSizeInfo.wImageWidth * (BDBoard.wZoomFactor / 2);
            ptEnd.y += sFrameSizeInfo.wImageHeight * (BDBoard.wZoomFactor / 2);

            //
            // scroll horizontal and vertical bars if not enough space on the screen
            // Use only the beginning point to control the scrollbars
            // If use both points Beginning and End points, there is a problem when
            // the image is equal to the screen size
            if (ptBeg.x < (rc.left + MIN_PIXELS_TO_SCROLL))
            {
                SendMessage(hWnd, WM_HSCROLL, SB_PAGEUP, 0L);
            }

            if (ptBeg.x > (rc.right - MIN_PIXELS_TO_SCROLL))
            {
                SendMessage(hWnd, WM_HSCROLL, SB_PAGEDOWN, 0L);
```

```
                    }

                    if (ptBeg.y < (rc.top + MIN_PIXELS_TO_SCROLL))
                    {
                        SendMessage(hWnd, WM_VSCROLL, SB_PAGEUP, 0L);
                    }

                    if (ptBeg.y > (rc.bottom - MIN_PIXELS_TO_SCROLL))
                    {
                        SendMessage(hWnd, WM_VSCROLL, SB_PAGEDOWN, 0L);
                    }

                    // draw the imported image according to the Zoom factor
                    DrawBoxOutline (hWnd, ptBeg, ptEnd, WHITE_PEN);
                    if (BDBoard.hwndStatusBar)
                    {
                        UpdateStatusBarCursor (BDBoard.hwndStatusBar, &pt);
                        // display the size of file image on the status bar
                        UpdateStatusBarImageSize (BDBoard.hwndStatusBar, sFrameSizeInfo.wImageWidth
FrameSizeInfo.wImageHeight);
                    }
                }
                break;

            // if Escape key is hit, image will be placed at the top left corner
            // User may have difficulty when placing the image at the top left corner
            case WM_KEYDOWN:
                if (msg.wParam == '\x1B') // Escape
                {
                    VW.x = 0;
                    VW.y = 0;
                    ok = FALSE; // quit the while loop
                }
                else if (msg.wParam == '\x0D') // Enter key
                {
                    POINT pt;
                    GetCursorPos(&pt); // pt in screen coordinate
                    ScreenToClient(hWnd, &pt);
                    VW.x = pt.x;
                    VW.y = pt.y;
                    ok = FALSE; // quit the while loop
                }
                break;
        } // end of switch
    } // end of while loop


    ClipCursor(NULL);
    ReleaseCapture(); // release the focus of the mouse

    // BDBoard.xCurrentScroll implies OutputControl.VideoRoi.x
    // BDBoard.yCurrentScroll implies OutputControl.VideoRoi.y
    if (!BDBoard.bFitToScreen)
    {
        VW.x += BDBoard.xCurrentScroll * BDBoard.wZoomFactor;
        VW.y += BDBoard.yCurrentScroll * BDBoard.wZoomFactor;
    }

    // erase last rectangle drawn and update the screen
```

```
        GetClientRect(hWnd, &rc); // get full rectangle including status bar
        InvalidateRect(hWnd, &rc, TRUE);
        SendMessage(hWnd, WM_PAINT, 0, 0L);
        ReAdjustVWOutput(hWnd, lpYUVbuffer, &VW, &BW, sFrameSizeInfo);
        lResult = BD_WriteVideoWindow( lpYUVbuffer, VW.xlen, &BW, &VW);
        if (lResult)
        {
            PrintBanditErrorMessage(hWnd, lResult, "ERROR after calling BD_WriteVideoWindow(), file \"image.c\"");
            retVal = FALSE;
        }

    return retVal;
}
```

2. 영상처리 VB code

```
Public Sub ThresholdGray(Sx As Long, Sy As Long, Ex As Long, Ey As Long, cutvalue As Byte)
'흑백 배열에 대한 이치화, cutvalue를 기준으로 이치화
'회색조로 변환된 영상 데이터에 대한 이치화, 이 때cutvalue는 문턱값이다.
    Dim x As Long
    Dim y As Long
    For y = Sy To Ey - 1
        For x = Sx To Ex - 1
            If ImageArrayGray(x, y) > cutvalue Then
                ImageArrayGray(x, y) = 255
            Else
                ImageArrayGray(x, y) = 0
            End If
        Next x
    Next y
End Sub

Public Sub ExtractBeef(Sx As Long, Sy As Long, Ex As Long, Ey As Long)
' 칼라 이치화, R,G,B의 색상정보의 상호 관계를 기준으로 한 이치화
' 칼라 색상 정보에서 두드러진 색상정보를 획득하고자 할 때 사용한다.
    Dim x As Long
    Dim y As Long
    Dim R As Long
    Dim G As Long
    Dim B As Long
    Dim tmpdata As Long
    For y = Sy To Ey - 1
        For x = Sx To Ex - 1
            R = CLng(ImageArrayColor(x, y, RedBand)) + CLng(ImageArrayColor(x + 1, y, RedBand)) + CLng(ImageArrayColor(x, y + 1,
RedBand)) + CLng(ImageArrayColor(x + 1, y + 1, RedBand))
            G = CLng(ImageArrayColor(x, y, GreenBand)) + CLng(ImageArrayColor(x + 1, y, GreenBand)) + CLng(ImageArrayColor(x, y
+ 1, GreenBand)) + CLng(ImageArrayColor(x + 1, y + 1, GreenBand))
            B = CLng(ImageArrayColor(x, y, BlueBand)) + CLng(ImageArrayColor(x + 1, y, BlueBand)) + CLng(ImageArrayColor(x, y + 1,
BlueBand)) + CLng(ImageArrayColor(x + 1, y + 1, BlueBand))
            R = R \ 4
            G = G \ 4
            B = B \ 4
            If x > 630 Or y > 470 Or x < 10 Or y < 10 Then
                    ImageArrayColor(x, y, RedBand) = 0
                    ImageArrayColor(x, y, GreenBand) = 0
                    ImageArrayColor(x, y, BlueBand) = 0
```

```
            ElseIf Abs(R - G) < 10 And Abs(R - B) < 10 And Abs(G - B) < 10 Then
                    ImageArrayColor(x, y, RedBand) = 0
                    ImageArrayColor(x, y, GreenBand) = 0
                    ImageArrayColor(x, y, BlueBand) = 0
            Else
            ' 각 RGB값의 차이가 25 이상일 때
                If (R > 50 And R < 120) And (Abs(G - B) < 10) And (R - Abs(G - B) > 10) Then 'R이G보다 클때
                    ImageArrayColor(x, y, RedBand) = 255
                    ImageArrayColor(x, y, GreenBand) = 0
                    ImageArrayColor(x, y, BlueBand) = 0
                Else
                    ImageArrayColor(x, y, RedBand) = 0
                    ImageArrayColor(x, y, GreenBand) = 0
                    ImageArrayColor(x, y, BlueBand) = 0
                End If
            End If
        Next x
    Next y
End Sub

Public Sub Numbering(Sx As Long, Sy As Long, Ex As Long, Ey As Long)
' 칼라 이치화, R,G,B의 색상정보의 상호 관계를 기준으로 한 이치화
' 칼라 색상 정보에서 두드러진 색상정보를 획득하고자 할 때 사용한다.
    Dim x As Long
    Dim y As Long
    Dim Red As Long
    Dim Green As Long
    Dim Blue As Long
    Dim tmpdata As Long
    Dim tmpArray(640, 480) As Long
    Dim NumHisto(100000) As Long
    Dim i As Long
    Dim j As Long
    Dim k As Long
    Dim Min As Long
    Dim Max As Long
    k = 1
    Min = 0
    For y = Sy + 1 To Ey - 1
        For x = Sx + 1 To Ex - 1
            If ImageArrayColor(x, y, RedBand) = 255 Then
                If tmpArray(x - 1, y - 1) = 0 And _
                   tmpArray(x, y - 1) = 0 And _
                   tmpArray(x - 1, y) = 0 And _
                   tmpArray(x + 1, y) = 0 And _
                   tmpArray(x - 1, y + 1) = 0 And _
                   tmpArray(x, y + 1) = 0 And _
                   tmpArray(x + 1, y + 1) = 0 Then
                    tmpArray(x, y) = k
                    k = k + 1
                Else
                    Min = 100000
                    For i = -1 To 1
                        For j = -1 To 1
                            If (tmpArray(x + i, y + j) < Min) And (tmpArray(x + i, y + j) <> 0) Then
                                Min = tmpArray(x + i, y + j)
                            End If
                        Next j
                    Next i
```

```
                tmpArray(x, y) = Min
                For i = -1 To 1
                    For j = -1 To 1
                        If tmpArray(x + i, y + j) <> 0 Then
                            If tmpArray(x + i, y + j) <> Min Then
                                tmpArray(x + i, y + j) = Min
                            End If
                        End If
                    Next j
                Next i
            End If
        Else
            tmpArray(x, y) = 0
        End If
    Next x
Next y

For y = Sy To Ey - 1
    For x = Sx To Ex - 1
        If (tmpArray(x, y) <> tmpArray(x, y + 1)) And (tmpArray(x, y) <> 0) And (tmpArray(x, y + 1)) Then
            If tmpArray(x, y) > tmpArray(x, y + 1) Then
                Min = tmpArray(x, y + 1)
                Max = tmpArray(x, y)
            Else
                Max = tmpArray(x, y + 1)
                Min = tmpArray(x, y)
            End If
            For j = Sy To Ey - 1
                For i = Sx To Ex - 1
                    If tmpArray(i, j) = Max Then
                        tmpArray(i, j) = Min
                    End If
                Next i
            Next j
        End If
    Next x
Next y

For y = Sy To Ey - 1
    For x = Sx To Ex - 1
        If tmpArray(x, y) <> 0 Then
            NumHisto(tmpArray(x, y)) = NumHisto(tmpArray(x, y)) + 1
        End If
    Next x
Next y
Max = 0
For x = 1 To 100000
    If NumHisto(x) > Max Then
        Max = NumHisto(x)
    End If
Next x
MainScreen.txtMessage = Str(Max)
For y = Sy To Ey - 1
    For x = Sx To Ex - 1
        If (tmpArray(x, y) <> 0) And (NumHisto(tmpArray(x, y)) = Max) Then
            ImageArrayColor(x, y, 0) = 255
            ImageArrayColor(x, y, 1) = 0
            ImageArrayColor(x, y, 2) = 0
        ElseIf (tmpArray(x, y) <> 0) And (NumHisto(tmpArray(x, y)) <> Max) Then
```

```
                ImageArrayColor(x, y, 0) = 0
                ImageArrayColor(x, y, 1) = 0
                ImageArrayColor(x, y, 2) = 0
            Else
                ImageArrayColor(x, y, 0) = 0
                ImageArrayColor(x, y, 1) = 0
                ImageArrayColor(x, y, 2) = 0
            End If
        Next x
    Next y
End Sub


Public Sub FindEdge(Sx As Long, Sy As Long, Ex As Long, Ey As Long, SourceArray() As Byte, TargetArray() As Byte, ObjectPoint() As PointData)
    Dim x As Long
    Dim y As Long
    Dim StartPosition As StartPos
    Dim tmpArray(639, 479, 2) As Byte

    StartPosition = FindStart(Sx, Sy, Ex, Ey, SourceArray)
    EdgeTracking StartPosition, RedChannel, SourceArray, SourceArray, ObjectPoint
End Sub

Public Function FindStart(Sx As Long, Sy As Long, Ex As Long, Ey As Long, SourceArray() As Byte) As StartPos
    Dim x As Long
    Dim y As Long
    For y = Sy To Ey - 1
        For x = Sx To Ex - 1
            If SourceArray(x, y, 0) = 255 Then
                FindStart.StartPosX = x
                FindStart.StartPosY = y
                Exit Function
            End If
        Next x
    Next y
End Function


Public Sub EdgeTracking(StartPosition As StartPos, band As Channel, SourceArray() As Byte, TargetArray() As Byte, ObjectPoint() As PointData)
    Dim x As Long
    Dim y As Long
    Dim sPosOrgX As Long
    Dim sPosOrgY As Long
    Dim DirX As Long
    Dim DirY As Long
    Dim sPosX As Long
    Dim sPosY As Long
    Dim cnt As Long
    Dim indx As Boolean
'   Dim ObjectPoint(10000) As PointData
    Dim tmpArray(639, 479, 2) As Byte
    Dim i As Long
    Dim j As Long
    Dim k As Long
    cnt = 0
    indx = False
```

```
For x = StartPosition.StartPosX To StartPosition.StartPosX + 10
    For y = StartPosition.StartPosY To StartPosition.StartPosY + 10
        If band <> DefAllChannel Or band <> AllChannel Then
            If SourceArray(x, y, band) = 255 Then
                TargetArray(x, y, 1) = 255
                TargetArray(x, y, 0) = 0
                TargetArray(x, y, 2) = 0
                sPosOrgX = x
                sPosOrgY = y
                indx = True
                Exit For
            End If
        End If
    Next y
    If indx = True Then Exit For
Next x
sPosX = sPosOrgX
sPosY = sPosOrgY
MainScreen.txtMessage = Str(sPosOrgX) + ", " + Str(sPosOrgY)
If SourceArray(sPosX - 1, sPosY - 1, band) = 255 Then
    DirX = -1
    DirY = -1
ElseIf SourceArray(sPosX, sPosY - 1, band) = 255 Then
    DirX = 0
    DirY = -1
ElseIf SourceArray(sPosX + 1, sPosY - 1, band) = 255 Then
    DirX = 1
    DirY = -1
ElseIf SourceArray(sPosX + 1, sPosY, band) = 255 Then
    DirX = 1
    DirY = 0
ElseIf SourceArray(sPosX + 1, sPosY + 1, band) = 255 Then
    DirX = 1
    DirY = 1
ElseIf SourceArray(sPosX, sPosY + 1, band) = 255 Then
    DirX = 0
    DirY = 1
ElseIf SourceArray(sPosX - 1, sPosY + 1, band) = 255 Then
    DirX = -1
    DirY = 1
End If
sPosX = sPosX + DirX
sPosY = sPosY + DirY
sPosX = sPosOrgX
sPosY = sPosOrgY
tmpArray(sPosX, sPosY, 0) = 0
tmpArray(sPosX, sPosY, 1) = 255
tmpArray(sPosX, sPosY, 2) = 0
TargetArray(sPosX, sPosY, 0) = 0
TargetArray(sPosX, sPosY, 1) = 255
TargetArray(sPosX, sPosY, 2) = 0
'TargetArray(sPosX, sPosY, 1) = 255
ObjectPoint(0).x = sPosX
ObjectPoint(0).y = sPosY
'MainScreen.ImageBufferMain.Put TargetArray, imRGB24Planar, imAllBands, 0, 0, 640, 480
Do
    'If SourceArray(sPosX, sPosY, 0) = 255 Or SourceArray(sPosX, sPosY, 1) = 255 Then
    If DirX = 1 And DirY = 0 Then
        If SourceArray(sPosX + 1, sPosY, band) = 255 Then '4
```

```
        DirX = 1
        DirY = 0
    ElseIf SourceArray(sPosX + 1, sPosY - 1, band) = 255 Then '3
        DirX = 1
        DirY = -1
    ElseIf SourceArray(sPosX, sPosY - 1, band) = 255 Then '2
        DirX = 0
        DirY = -1
    ElseIf SourceArray(sPosX - 1, sPosY - 1, band) = 255 Then '1
        DirX = -1
        DirY = -1
    ElseIf SourceArray(sPosX + 1, sPosY + 1, band) = 255 Then '5
        DirX = 1
        DirY = 1
    ElseIf SourceArray(sPosX, sPosY + 1, band) = 255 Then '6
        DirX = 0
        DirY = 1
    ElseIf SourceArray(sPosX - 1, sPosY + 1, band) = 255 Then '7
        DirX = -1
        DirY = 1
    Else
        DirX = 0
        DirY = 0
    End If

ElseIf DirX = 1 And DirY = -1 Then
    If SourceArray(sPosX + 1, sPosY - 1, band) = 255 Then '3
        DirX = 1
        DirY = -1
    ElseIf SourceArray(sPosX, sPosY - 1, band) = 255 Then '2
        DirX = 0
        DirY = -1
    ElseIf SourceArray(sPosX - 1, sPosY - 1, band) = 255 Then '1
        DirX = -1
        DirY = -1
    ElseIf SourceArray(sPosX - 1, sPosY, band) = 255 Then '8
        DirX = -1
        DirY = 0
    ElseIf SourceArray(sPosX + 1, sPosY, band) = 255 Then '4
        DirX = 1
        DirY = 0
    ElseIf SourceArray(sPosX + 1, sPosY + 1, band) = 255 Then '5
        DirX = 1
        DirY = 1
    ElseIf SourceArray(sPosX, sPosY + 1, band) = 255 Then '6
        DirX = 0
        DirY = 1
    Else
        DirX = 0
        DirY = 0
    End If
ElseIf DirX = 1 And DirY = 1 Then
    If SourceArray(sPosX + 1, sPosY + 1, band) = 255 Then '5
        DirX = 1
        DirY = 1
    ElseIf SourceArray(sPosX + 1, sPosY, band) = 255 Then '4
        DirX = 1
        DirY = 0
    ElseIf SourceArray(sPosX + 1, sPosY - 1, band) = 255 Then '3
```

```
        DirX = 1
        DirY = -1
    ElseIf SourceArray(sPosX, sPosY - 1, band) = 255 Then '2
        DirX = 0
        DirY = -1
    ElseIf SourceArray(sPosX, sPosY + 1, band) = 255 Then '6
        DirX = 0
        DirY = 1
    ElseIf SourceArray(sPosX - 1, sPosY + 1, band) = 255 Then '7
        DirX = -1
        DirY = 1
    ElseIf SourceArray(sPosX - 1, sPosY, band) = 255 Then '8
        DirX = -1
        DirY = 0
    Else
        DirX = 0
        DirY = 0
    End If
ElseIf DirX = 0 And DirY = -1 Then
    If SourceArray(sPosX, sPosY - 1, band) = 255 Then '2
        DirX = 0
        DirY = -1
    ElseIf SourceArray(sPosX - 1, sPosY - 1, band) = 255 Then '1
        DirX = -1
        DirY = -1
    ElseIf SourceArray(sPosX - 1, sPosY, band) = 255 Then '8
        DirX = -1
        DirY = 0
    ElseIf SourceArray(sPosX - 1, sPosY + 1, band) = 255 Then '7
        DirX = -1
        DirY = 1
    ElseIf SourceArray(sPosX + 1, sPosY - 1, band) = 255 Then '3
        DirX = 1
        DirY = -1
    ElseIf SourceArray(sPosX + 1, sPosY, band) = 255 Then '4
        DirX = 1
        DirY = 0
    ElseIf SourceArray(sPosX + 1, sPosY + 1, band) = 255 Then '5
        DirX = 1
        DirY = 1
    Else
        DirX = 0
        DirY = 0
    End If
ElseIf DirX = 0 And DirY = 1 Then
    If SourceArray(sPosX, sPosY + 1, band) = 255 Then '6
        DirX = 0
        DirY = 1
    ElseIf SourceArray(sPosX + 1, sPosY + 1, band) = 255 Then '5
        DirX = 1
        DirY = 1
    ElseIf SourceArray(sPosX + 1, sPosY, band) = 255 Then '4
        DirX = 1
        DirY = 0

    ElseIf SourceArray(sPosX + 1, sPosY - 1, band) = 255 Then '3
        DirX = 1
        DirY = -1
    ElseIf SourceArray(sPosX - 1, sPosY + 1, band) = 255 Then '7
```

```
        DirX = -1
        DirY = 1
    ElseIf SourceArray(sPosX - 1, sPosY, band) = 255 Then '8
        DirX = -1
        DirY = 0
    ElseIf SourceArray(sPosX - 1, sPosY - 1, band) = 255 Then '1
        DirX = -1
        DirY = -1
    Else
        DirX = 0
        DirY = 0
    End If
ElseIf DirX = -1 And DirY = -1 Then
    If SourceArray(sPosX - 1, sPosY - 1, band) = 255 Then '1
        DirX = -1
        DirY = -1
    ElseIf SourceArray(sPosX - 1, sPosY, band) = 255 Then '8
        DirX = -1
        DirY = 0
    ElseIf SourceArray(sPosX - 1, sPosY + 1, band) = 255 Then '7
        DirX = -1
        DirY = 1
    ElseIf SourceArray(sPosX, sPosY + 1, band) = 255 Then '6
        DirX = 0
        DirY = 1
    ElseIf SourceArray(sPosX, sPosY - 1, band) = 255 Then '2
        DirX = 0
        DirY = -1
    ElseIf SourceArray(sPosX + 1, sPosY - 1, band) = 255 Then '3
        DirX = 1
        DirY = -1
    ElseIf SourceArray(sPosX + 1, sPosY, band) = 255 Then '4
        DirX = 1
        DirY = 0
    ElseIf SourceArray(sPosX + 1, sPosY + 1, band) = 255 Then '5
        DirX = 1
        DirY = 1
    Else
        DirX = 0
        DirY = 0
    End If
ElseIf DirX = -1 And DirY = 1 Then
    If SourceArray(sPosX - 1, sPosY + 1, band) = 255 Then '7
        DirX = -1
        DirY = 1
    ElseIf SourceArray(sPosX, sPosY + 1, band) = 255 Then '6
        DirX = 0
        DirY = 1
    ElseIf SourceArray(sPosX - 1, sPosY, band) = 255 Then '8
        DirX = -1
        DirY = 0
    ElseIf SourceArray(sPosX + 1, sPosY + 1, band) = 255 Then '5
        DirX = 1
        DirY = 1
    ElseIf SourceArray(sPosX - 1, sPosY - 1, band) = 255 Then '1
        DirX = -1
        DirY = -1
    ElseIf SourceArray(sPosX + 1, sPosY, band) = 255 Then '4
        DirX = 1
```

```
        DirY = 0
    ElseIf SourceArray(sPosX, sPosY - 1, band) = 255 Then '2
        DirX = 0
        DirY = -1
    Else
        DirX = 0
        DirY = 0
    End If
ElseIf DirX = -1 And DirY = 0 Then
    If SourceArray(sPosX - 1, sPosY, band) = 255 Then '8
        DirX = -1
        DirY = 0
    ElseIf SourceArray(sPosX - 1, sPosY + 1, band) = 255 Then '7
        DirX = -1
        DirY = 1
    ElseIf SourceArray(sPosX, sPosY + 1, band) = 255 Then '6
        DirX = 0
        DirY = 1
    ElseIf SourceArray(sPosX + 1, sPosY + 1, band) = 255 Then '5
        DirX = 1
        DirY = 1
    ElseIf SourceArray(sPosX - 1, sPosY - 1, band) = 255 Then '1
        DirX = -1
        DirY = -1
    ElseIf SourceArray(sPosX, sPosY - 1, band) = 255 Then '2
        DirX = 0
        DirY = -1
    ElseIf SourceArray(sPosX + 1, sPosY - 1, band) = 255 Then '3
        DirX = 1
        DirY = -1
    Else
        DirX = 0
        DirY = 0
    End If
End If
If DirX = 0 And DirY = 0 Then
    indx = False
    For k = 2 To 100
        For j = -k To k
            For i = -k To k
                If SourceArray(sPosX + i, sPosY + j, band) = 255 And _
                    SourceArray(sPosX + i, sPosY + j, 1) = 0 And (i <> 0 And j <> 0) Then
                    DirX = i
                    DirY = j
                    indx = True
                    Exit For
                End If
            Next i
            If indx = True Then Exit For
        Next j
        If indx = True Then
            Exit For
        End If
    Next k
End If
'End If
cnt = cnt + 1
sPosX = sPosX + DirX
sPosY = sPosY + DirY
```

```
        If DirX > 0 Then
            DirX = 1
        ElseIf DirX = 0 Then
            DirX = 0
        Else
            DirX = -1
        End If
        If DirY > 0 Then
            DirY = 1
        ElseIf DirY = 0 Then
            DirY = 0
        Else
            DirY = -1
        End If
        ObjectPoint(cnt).x = sPosX
        ObjectPoint(cnt).y = sPosY
        TargetArray(sPosX, sPosY, 0) = 0
        TargetArray(sPosX, sPosY, 1) = 255
        TargetArray(sPosX, sPosY, 2) = 0
        tmpArray(sPosX, sPosY, 0) = 0
        tmpArray(sPosX, sPosY, 1) = 255
        tmpArray(sPosX, sPosY, 2) = 0
        'TargetArray(sPosX, sPosY, 1) = 255
        'MainScreen.ImageBufferMain.Put TargetArray, imRGB24Planar, imAllBands, 0, 0, 640, 480
        DoEvents
        If cnt > 10000 Or sPosX = 638 Or sPosY = 478 Or sPosX = 1 Or sPosY = 1 Then Exit Do
        MainScreen.txtMessage = Str(sPosOrgX) + ", " + Str(sPosOrgY) + ", " + Str(sPosX) + ", " + Str(sPosY)
        If (sPosX + 1 = sPosOrgX Or sPosX - 1 = sPosOrgX Or sPosX = sPosOrgX) And _
            (sPosY + 1 = sPosOrgY Or sPosY - 1 = sPosOrgY Or sPosY = sPosOrgY) And cnt > 5 Then Exit Do
    Loop
    DoEvents
End Sub


Public Sub ColorToGray(Sx As Long, Sy As Long, Ex As Long, Ey As Long, SourceArray() As Byte, TargetArray() As Byte)
'주어진 sx,sy,ex,ey영역의 칼라 영상을 그레이 스케일로  변경시킨다.
    Dim x As Long
    Dim y As Long
    Dim tmpdata As Byte
    For y = Sy To Ey - 1 '상에서 하로
        For x = Sx To Ex - 1 '좌에서 우로
            tmpdata = CByte((CLng(SourceArray(x, y, 0)) + CLng(SourceArray(x, y, 1)) + CLng(SourceArray(x, y, 2))) \ 3)
            ' R,G,B값의 평균을 tmpdata에 저장
            '여기서 CByte(숫자)함수는 ()내의 숫자를 8비트 형식(byte)으로 변환하는함수
            '다음으로 CLng(숫자)함수는 ()내의 숫자를 Long형식으로 변환하는함수
            'ImageArrayColor(x, y, 0) = tmpdata
            'ImageArrayColor(x, y, 1) = tmpdata
            'ImageArrayColor(x, y, 2) = tmpdata
            '각 이미지배열칼라에 tmpdata를 저장, 주 화면에 회색조 영상을 볼 수 있게 한다.
            '화면에 회색조를 표시하지 않게 하려면 위의 세 줄을 사용하지 않으면 된다.
            TargetArray(x, y, 0) = tmpdata
            TargetArray(x, y, 1) = tmpdata
            TargetArray(x, y, 2) = tmpdata
            '흑백 영상 처리를 하기 위하여 별도의 이미지배열흑백에 tmpdata를 저장한다.(필수)
            '절화 영상처리 알고리즘 대부분이 이 회색조를 이용하기 때문
        Next x
    Next y
End Sub
```

```
Public Sub Threshold(Sx As Long, Sy As Long, Ex As Long, Ey As Long, band As Channel, cutvalue As cutvalue, SourceArray() A
Byte, TargetArray() As Byte)
'이치화 루틴, 주어지 영역에 대하여, 주어진 밴드(R,G,B 밴드 중 하나)에 대하여 주어진 cutvalue를 기준으로 이치화
'특별하게 어떤 밴드만 이치화 할 필요가 있을 때 사용할 수 있다.
    Dim x As Long
    Dim y As Long
    For y = Sy To Ey - 1
        For x = Sx To Ex - 1
            If band = DefAllChannel Then
                If SourceArray(x, y, 0) > cutvalue.RedCutValue Then
                    TargetArray(x, y, 0) = 255
                Else
                    TargetArray(x, y, 0) = 0
                End If
                If SourceArray(x, y, 1) > cutvalue.GreenCutValue Then
                    TargetArray(x, y, 1) = 255
                Else
                    TargetArray(x, y, 1) = 0
                End If
                If SourceArray(x, y, 2) > cutvalue.BlueCutValue Then
                    TargetArray(x, y, 2) = 255
                Else
                    TargetArray(x, y, 2) = 0
                End If
            ElseIf band = AllChannel Then
                If SourceArray(x, y, 0) > cutvalue.AllCutValue Then
                    TargetArray(x, y, 0) = 255
                Else
                    TargetArray(x, y, 0) = 0
                End If
                If SourceArray(x, y, 1) > cutvalue.AllCutValue Then
                    TargetArray(x, y, 1) = 255
                Else
                    TargetArray(x, y, 1) = 0
                End If
                If SourceArray(x, y, 2) > cutvalue.AllCutValue Then
                    TargetArray(x, y, 2) = 255
                Else
                    TargetArray(x, y, 2) = 0
                End If
            Else
                If band = RedChannel Then
                    If SourceArray(x, y, band) > cutvalue.RedCutValue Then
                        TargetArray(x, y, band) = 255
                    Else
                        TargetArray(x, y, band) = 0
                    End If
                ElseIf band = greenChannel Then
                    If SourceArray(x, y, band) > cutvalue.GreenCutValue Then
                        TargetArray(x, y, band) = 255
                    Else
                        TargetArray(x, y, band) = 0
                    End If
                Else
                    If SourceArray(x, y, band) > cutvalue.BlueCutValue Then
                        TargetArray(x, y, band) = 255
                    Else
                        TargetArray(x, y, band) = 0
                    End If
```

```
            End If
        End If
    Next x
Next y
End Sub

Public Sub ThresholdBlock(Sx As Long, Sy As Long, Ex As Long, Ey As Long, band As Channel, Cell As Byte, SourceArray() As Byte,
TargetArray() As Byte)
'이치화 루틴, 주어지 영역에 대하여, 주어진 밴드(R,G,B 밴드 중 하나)에 대하여 주어진 cutvalue를 기준으로 이치화
'특별하게 어떤 밴드만 이치화 할 필요가 있을 때 사용할 수 있다.
    Dim x As Long
    Dim y As Long
    Dim i As Long
    Dim j As Long
    Dim SumR As Double
    Dim SumG As Double
    Dim SumB As Double
    Dim Sum As Double
    Dim cutvalueR As Double
    Dim cutvalueG As Double
    Dim cutvalueB As Double
    Dim cutvalue As Double
    Sum = 0: SumR = 0: SumG = 0: SumB = 0
    For y = Sy To Ey - 1 Step Cell
        For x = Sx To Ex - 1 Step Cell
            If (x + Cell > Ex - 1) Or (y + Cell > Ey - 1) Then Exit For
            For j = 0 To Cell - 1
                For i = 0 To Cell - 1
                    If band = AllChannel Then
                        SumR = SumR + CDbl(SourceArray(x + i, y + j, 0))
                        SumG = SumG + CDbl(SourceArray(x + i, y + j, 1))
                        SumB = SumB + CDbl(SourceArray(x + i, y + j, 2))
                    Else
                        Sum = Sum + CDbl(SourceArray(x + i, y + j, band))
                    End If
                Next i
            Next j
            If band = AllChannel Then
                cutvalueR = CDbl(SumR / CDbl(CDbl(Cell) * CDbl(Cell)))
                cutvalueG = CDbl(SumG / CDbl(CDbl(Cell) * CDbl(Cell)))
                cutvalueB = CDbl(SumB / CDbl(CDbl(Cell) * CDbl(Cell)))
            Else
                cutvalue = CDbl(Sum / CDbl(CDbl(Cell) * CDbl(Cell)))
            End If
            For j = 0 To Cell - 1
                For i = 0 To Cell - 1
                    If band = AllChannel Then
                        If SourceArray(x + i, y + j, 0) >= cutvalueR Then
                            TargetArray(x + i, y + j, 0) = 255
                        Else
                            TargetArray(x + i, y + j, 0) = 0
                        End If
                        If SourceArray(x + i, y + j, 1) >= cutvalueG Then
                            TargetArray(x + i, y + j, 1) = 255
                        Else
                            TargetArray(x + i, y + j, 1) = 0
                        End If
                        If SourceArray(x + i, y + j, 2) >= cutvalueB Then
                            TargetArray(x + i, y + j, 2) = 255
```

```
                    Else
                        TargetArray(x + i, y + j, 2) = 0
                    End If
                Else
                    If SourceArray(x + i, y + j, band) >= cutvalue Then
                        TargetArray(x + i, y + j, band) = 255
                    Else
                        TargetArray(x + i, y + j, band) = 0
                    End If
                End If
            Next i
        Next j
        Sum = 0: SumR = 0: SumG = 0: SumB = 0
    Next x
Next y
End Sub

Public Sub CopyArray(Sx As Long, Sy As Long, Ex As Long, Ey As Long, SourceArray() As Byte, TargetArray() As Byte)
'흑백 배열에 대한 이치화, cutvalue를 기준으로 이치화
'회색조로 변환된 영상 데이터에 대한 이치화, 이 때cutvalue는 문턱값이다.
    Dim x As Long
    Dim y As Long
    For y = Sy To Ey
        For x = Sx To Ex
            TargetArray(x, y, 0) = SourceArray(x, y, 0)
            TargetArray(x, y, 1) = SourceArray(x, y, 1)
            TargetArray(x, y, 2) = SourceArray(x, y, 2)
        Next x
    Next y
End Sub


Public Sub ErosionForBinary(Sx As Long, Sy As Long, Ex As Long, Ey As Long, band As Channel, SourceArray() As Byte
TargetArray() As Byte)
    Dim x As Long
    Dim y As Long
    Dim SumR As Byte
    Dim SumG As Byte
    Dim SumB As Byte
    Dim Sum As Byte
    Dim row As Byte
    Dim column As Byte
    Dim temp(2, 2) As Byte
    Dim mask(2, 2) As Byte
    Dim TmpBuffer(639, 479, 2) As Byte

    Sum = 0: SumR = 0: SumG = 0: SumB = 0
    mask(0, 0) = 255: mask(0, 1) = 255: mask(0, 2) = 255
    mask(1, 0) = 255: mask(1, 1) = 255: mask(1, 2) = 255
    mask(2, 0) = 255: mask(2, 1) = 255: mask(2, 2) = 255
    For y = Sy + 1 To Ey - 2
        For x = Sx + 1 To Ex - 2
            For row = 0 To 2
                For column = 0 To 2
                    If band = AllChannel Then
                        If mask(row, column) = SourceArray(row + x, column + y, 0) Then
                            SumR = SumR + 1
                        End If
                        If mask(row, column) = SourceArray(row + x, column + y, 1) Then
```

```
                    SumG = SumG + 1
                End If
                If mask(row, column) = SourceArray(row + x, column + y, 2) Then
                    SumB = SumB + 1
                End If
            ElseIf band <> DefAllChannel Then
                If mask(row, column) = SourceArray(row + x, column + y, band) Then
                    Sum = Sum + 1
                End If
            End If
        Next column
    Next row
    If band = AllChannel Then
        If SumR = 9 Then
            TmpBuffer(1 + x, 1 + y, 0) = 255
        Else
            TmpBuffer(1 + x, 1 + y, 0) = 0
        End If
        SumR = 0
        If SumG = 9 Then
            TmpBuffer(1 + x, 1 + y, 1) = 255
        Else
            TmpBuffer(1 + x, 1 + y, 1) = 0
        End If
        SumG = 0
        If SumB = 9 Then
            TmpBuffer(1 + x, 1 + y, 2) = 255
        Else
            TmpBuffer(1 + x, 1 + y, 2) = 0
        End If
        SumB = 0
    ElseIf band <> DefAllChannel Then
        If Sum = 9 Then
            TmpBuffer(1 + x, 1 + y, band) = 255
        Else
            TmpBuffer(1 + x, 1 + y, band) = 0
        End If
        Sum = 0
    End If
  Next x
Next y
For y = Sy + 1 To Ey - 2
  For x = Sx + 1 To Ex - 2
    If band = AllChannel Then
      TargetArray(x, y, 0) = TmpBuffer(x, y, 0)
      TargetArray(x, y, 1) = TmpBuffer(x, y, 1)
      TargetArray(x, y, 2) = TmpBuffer(x, y, 2)
    ElseIf band <> DefAllChannel Then
      TargetArray(x, y, band) = TmpBuffer(x, y, band)
    End If
  Next x
Next y
End Sub

Public Sub DilationForBinary(Sx As Long, Sy As Long, Ex As Long, Ey As Long, band As Channel, SourceArray() As Byte,
TargetArray() As Byte)
  Dim x As Long
  Dim y As Long
  Dim Sum As Byte
```

```
Dim row As Byte
Dim column As Byte
Dim temp(2, 2) As Byte
Dim mask(2, 2) As Byte
Dim TmpBuffer(639, 479, 2) As Byte

Sum = 0: SumR = 0: SumG = 0: SumB = 0
mask(0, 0) = 0: mask(0, 1) = 0: mask(0, 2) = 0
mask(1, 0) = 0: mask(1, 1) = 0: mask(1, 2) = 0
mask(2, 0) = 0: mask(2, 1) = 0: mask(2, 2) = 0
For y = Sy + 1 To Ey - 2
    For x = Sx + 1 To Ex - 2
        For row = 0 To 2
            For column = 0 To 2
                If band = AllChannel Then
                    If mask(row, column) = SourceArray(row + x, column + y, 0) Then
                        SumR = SumR + 1
                    End If
                    If mask(row, column) = SourceArray(row + x, column + y, 1) Then
                        SumG = SumG + 1
                    End If
                    If mask(row, column) = SourceArray(row + x, column + y, 2) Then
                        SumB = SumB + 1
                    End If
                ElseIf band <> DefAllChannel Then
                    If mask(row, column) = SourceArray(row + x, column + y, band) Then
                        Sum = Sum + 1
                    End If
                End If
            Next column
        Next row
        If band = AllChannel Then
            If SumR = 9 Then
                TmpBuffer(1 + x, 1 + y, 0) = 0
            Else
                TmpBuffer(1 + x, 1 + y, 0) = 255
            End If
            SumR = 0
            If SumG = 9 Then
                TmpBuffer(1 + x, 1 + y, 1) = 0
            Else
                TmpBuffer(1 + x, 1 + y, 1) = 255
            End If
            SumG = 0
            If SumB = 9 Then
                TmpBuffer(1 + x, 1 + y, 2) = 0
            Else
                TmpBuffer(1 + x, 1 + y, 2) = 255
            End If
            SumB = 0
        ElseIf band <> DefAllChannel Then
            If Sum = 9 Then
                TmpBuffer(1 + x, 1 + y, band) = 0
            Else
                TmpBuffer(1 + x, 1 + y, band) = 255
            End If
            Sum = 0
        End If
    Next x
```

```
    Next y
    For y = Sy + 1 To Ey - 2
        For x = Sx + 1 To Ex - 2
            If band = AllChannel Then
                TargetArray(x, y, 0) = TmpBuffer(x, y, 0)
                TargetArray(x, y, 1) = TmpBuffer(x, y, 1)
                TargetArray(x, y, 2) = TmpBuffer(x, y, 2)
            ElseIf band <> DefAllChannel Then
                TargetArray(x, y, band) = TmpBuffer(x, y, band)
            End If
        Next x
    Next y
End Sub

Public Sub ErosionForGray(Sx As Long, Sy As Long, Ex As Long, Ey As Long, band As Channel, SourceArray() As Byte, TargetArray(
As Byte)
    Dim x As Long
    Dim y As Long
    Dim i, j As Long
    Dim MinR As Byte
    Dim MinG As Byte
    Dim MinB As Byte
    Dim Min As Byte
    Dim SumR As Long
    Dim SumG As Long
    Dim SumB As Long
    Dim Sum As Long

    Dim row As Byte
    Dim column As Byte
    Dim temp(2, 2) As Byte
    Dim mask(2, 2) As Byte
    Dim TmpBuffer(639, 479, 2) As Byte

    Sum = 0: SumR = 0: SumG = 0: SumB = 0
    Min = 255: MinR = 255: MinG = 255: MinB = 255
    mask(0, 0) = 0: mask(0, 1) = 0: mask(0, 2) = 0
    mask(1, 0) = 0: mask(1, 1) = 0: mask(1, 2) = 0
    mask(2, 0) = 0: mask(2, 1) = 0: mask(2, 2) = 0

    For y = Sy + 1 To Ey - 2
        For x = Sx + 1 To Ex - 2
            For row = 0 To 2
                For column = 0 To 2
                    If band = AllChannel Then
                        SumR = mask(row, column) + SourceArray(row + x, column + y, 0)
                        SumG = mask(row, column) + SourceArray(row + x, column + y, 1)
                        SumB = mask(row, column) + SourceArray(row + x, column + y, 2)
                        If MinR > SumR Then MinR = SumR
                        If MinG > SumG Then MinG = SumG
                        If MinB > SumB Then MinB = SumB
                    ElseIf band <> DefAllChannel Then
                        Sum = mask(row, column) + SourceArray(row + x, column + y, band)
                        If Min > Sum Then Min = Sum
                    End If
                Next column
            Next row
            If band = AllChannel Then
                TmpBuffer(1 + x, 1 + y, 0) = MinR
```

```
                TmpBuffer(1 + x, 1 + y, 1) = MinG
                TmpBuffer(1 + x, 1 + y, 2) = MinB
                SumR = 0: SumG = 0: SumB = 0
                MinR = 255: MinG = 255: MinB = 255
            ElseIf band <> DefAllChannel Then
                TmpBuffer(1 + x, 1 + y, band) = Min
                Sum = 0
                Min = 255
            End If
        Next x
    Next y
    For y = Sy + 1 To Ey - 2
        For x = Sx + 1 To Ex - 2
            If band = AllChannel Then
                TargetArray(x, y, 0) = TmpBuffer(x, y, 0)
                TargetArray(x, y, 1) = TmpBuffer(x, y, 1)
                TargetArray(x, y, 2) = TmpBuffer(x, y, 2)
            ElseIf band <> DefAllChannel Then
                TargetArray(x, y, band) = TmpBuffer(x, y, band)
            End If
        Next x
    Next y
End Sub

Public Sub DilationForGray(Sx As Long, Sy As Long, Ex As Long, Ey As Long, band As Channel, SourceArray() As Byte, TargetArray(
As Byte)
    Dim x As Long
    Dim y As Long
    Dim i, j As Long
    Dim MaxR As Byte
    Dim MaxG As Byte
    Dim MaxB As Byte
    Dim Max As Byte
    Dim SumR As Long
    Dim SumG As Long
    Dim SumB As Long
    Dim Sum As Long

    Dim row As Byte
    Dim column As Byte
    Dim temp(2, 2) As Byte
    Dim mask(2, 2) As Byte
    Dim TmpBuffer(639, 479, 2) As Byte

    Sum = 0: SumR = 0: SumG = 0: SumB = 0
    Max = 0: MaxR = 0: MaxG = 0: MaxB = 0
    mask(0, 0) = 0: mask(0, 1) = 0: mask(0, 2) = 0
    mask(1, 0) = 0: mask(1, 1) = 0: mask(1, 2) = 0
    mask(2, 0) = 0: mask(2, 1) = 0: mask(2, 2) = 0

    For y = Sy + 1 To Ey - 2
        For x = Sx + 1 To Ex - 2
            For row = 0 To 2
                For column = 0 To 2
                    If band = AllChannel Then
                        SumR = mask(row, column) + SourceArray(row + x, column + y, 0)
                        If MaxR < SumR Then MaxR = SumR
                        SumG = mask(row, column) + SourceArray(row + x, column + y, 1)
                        If MaxG < SumG Then MaxG = SumG
```

```
                    SumB = mask(row, column) + SourceArray(row + x, column + y, 2)
                    If MaxB < SumB Then MaxB = SumB
                ElseIf band <> DefAllChannel Then
                    Sum = mask(row, column) + SourceArray(row + x, column + y, band)
                    If Max < Sum Then Max = Sum
                End If
            Next column
        Next row
        If band = AllChannel Then
            TmpBuffer(1 + x, 1 + y, 0) = MaxR
            TmpBuffer(1 + x, 1 + y, 1) = MaxG
            TmpBuffer(1 + x, 1 + y, 2) = MaxB
            SumR = 0: SumG = 0: SumB = 0
            MaxR = 0: MaxG = 0: MaxB = 0
        ElseIf band <> DefAllChannel Then
            TmpBuffer(1 + x, 1 + y, band) = Max
            Sum = 0
            Max = 0
        End If
    Next x
Next y
For y = Sy + 1 To Ey - 2
    For x = Sx + 1 To Ex - 2
        If band = AllChannel Then
            TargetArray(x, y, 0) = TmpBuffer(x, y, 0)
            TargetArray(x, y, 1) = TmpBuffer(x, y, 1)
            TargetArray(x, y, 2) = TmpBuffer(x, y, 2)
        ElseIf band <> DefAllChannel Then
            TargetArray(x, y, band) = TmpBuffer(x, y, band)
        End If
    Next x
Next y
End Sub

Public Sub DisplayZoom(ZoomX As Byte, ZoomY As Byte, band As Channel, SourceArray() As Byte, TargetArray() As Byte)
  Dim x As Long
  Dim y As Long
  Dim i As Long
  Dim j As Long
  Dim xx As Long
  Dim yy As Long
  Dim ImageTmpRed As Long
  Dim ImageTmpGreen As Long
  Dim ImageTmpBlue As Long
  Dim ImageTmp As Long
  Dim TmpRedA As Long
  Dim TmpRedB As Long
  Dim TmpRedC As Long
  Dim TmpGreenA As Long
  Dim TmpGreenB As Long
  Dim TmpGreenC As Long
  Dim TmpBlueA As Long
  Dim TmpBlueB As Long
  Dim TmpBlueC As Long
  Dim TmpA As Long
  Dim TmpB As Long
  Dim TmpC As Long

    For y = 0 To 479
```

```
    For x = 0 To 639
        If band = AllChannel Then
            TargetArray(x, y, 0) = 0
            TargetArray(x, y, 1) = 0
            TargetArray(x, y, 2) = 0
        ElseIf band <> DefAllChannel Then
            TargetArray(x, y, band) = 0
        End If
    Next x
Next y
xx = ProcSx
yy = ProcSy
For y = 0 To 239 Step ZoomY
    For x = 0 To 319 Step ZoomX
        If band = AllChannel Then
            TargetArray(x, y, 0) = SourceArray(xx, yy, 0)
            TargetArray(x, y, 1) = SourceArray(xx, yy, 1)
            TargetArray(x, y, 2) = SourceArray(xx, yy, 2)
        ElseIf band <> DefAllChannel Then
            If band = RedChannel Then
                TargetArray(x, y, 0) = SourceArray(xx, yy, 0)
                TargetArray(x, y, 1) = SourceArray(xx, yy, 0)
                TargetArray(x, y, 2) = SourceArray(xx, yy, 0)
            ElseIf band = greenChannel Then
                TargetArray(x, y, 0) = SourceArray(xx, yy, 1)
                TargetArray(x, y, 1) = SourceArray(xx, yy, 1)
                TargetArray(x, y, 2) = SourceArray(xx, yy, 1)
            Else
                TargetArray(x, y, 0) = SourceArray(xx, yy, 2)
                TargetArray(x, y, 1) = SourceArray(xx, yy, 2)
                TargetArray(x, y, 2) = SourceArray(xx, yy, 2)
            End If
        End If
        xx = xx + 1
    Next x
    xx = ProcSx
    yy = yy + 1
Next y

For y = 0 To 239 Step ZoomY
    For x = 0 To 319 Step ZoomX
        If band = AllChannel Then
            TmpRedA = (CLng(TargetArray(x + ZoomX, y, 0)) - CLng(TargetArray(x, y, 0))) / ZoomX
            TmpRedB = (CLng(TargetArray(x, y + ZoomY, 0)) - CLng(TargetArray(x, y, 0))) / ZoomY
            TmpRedC = TargetArray(x, y, 0)
            TmpGreenA = (CLng(TargetArray(x + ZoomX, y, 1)) - CLng(TargetArray(x, y, 1))) / ZoomX
            TmpGreenB = (CLng(TargetArray(x, y + ZoomY, 1)) - CLng(TargetArray(x, y, 1))) / ZoomY
            TmpGreenC = TargetArray(x, y, 1)
            TmpBlueA = (CLng(TargetArray(x + ZoomX, y, 2)) - CLng(TargetArray(x, y, 2))) / ZoomX
            TmpBlueB = (CLng(TargetArray(x, y + ZoomY, 2)) - CLng(TargetArray(x, y, 2))) / ZoomY
            TmpBlueC = TargetArray(x, y, 2)
        ElseIf band <> DefAllChannel Then
            TmpA = (CLng(TargetArray(x + ZoomX, y, band)) - CLng(TargetArray(x, y, band))) / ZoomX
            TmpB = (CLng(TargetArray(x, y + ZoomY, band)) - CLng(TargetArray(x, y, band))) / ZoomY
            TmpC = TargetArray(x, y, band)
        End If
        For j = 0 To ZoomY
            For i = 0 To ZoomX - j
                If Not ((i = 0 And j = 0) Or (i = ZoomX And j = 0) Or (i = 0 And j = ZoomY)) Then
```

```
                    If band = AllChannel Then
                        ImageTmpRed = (TmpRedA * i) + (TmpRedB * j) + TmpRedC
                        ImageTmpGreen = (TmpGreenA * i) + (TmpGreenB * j) + TmpGreenC
                        ImageTmpBlue = (TmpBlueA * i) + (TmpBlueB * j) + TmpBlueC
                        If ImageTmpRed < 0 Then ImageTmpRed = 0
                        If ImageTmpGreen < 0 Then ImageTmpGreen = 0
                        If ImageTmpBlue < 0 Then ImageTmpBlue = 0
                        If ImageTmpRed > 255 Then ImageTmpRed = 255
                        If ImageTmpGreen > 255 Then ImageTmpGreen = 255
                        If ImageTmpBlue > 255 Then ImageTmpBlue = 255
                        TargetArray(x + i, y + j, 0) = ImageTmpRed
                        TargetArray(x + i, y + j, 1) = ImageTmpGreen
                        TargetArray(x + i, y + j, 2) = ImageTmpBlue
                    ElseIf band <> DefAllChannel Then
                        ImageTmp = (TmpA * i) + (TmpB * j) + TmpC
                        If ImageTmp < 0 Then ImageTmp = 0
                        If ImageTmp > 255 Then ImageTmp = 255
                        TargetArray(x + i, y + j, band) = ImageTmp
                        'MainScreen.ImageBuffer1.Put TargetArray, imRGB24Planar, imAllBands, x + i, y + j, 1, 1
                    End If
                End If
            Next i
        Next j
    Next x
    DoEvents
Next y
DoEvents
For y = 0 To 239 Step ZoomY
    For x = 0 To 319 Step ZoomX
        If band = AllChannel Then
            TmpRedA = (CLng(TargetArray(x + ZoomX, y + ZoomY, 0)) - CLng(TargetArray(x, y + ZoomY, 0))) / ZoomX
            TmpRedB = (CLng(TargetArray(x + ZoomX, y + ZoomY, 0)) - CLng(TargetArray(x + ZoomX, y, 0))) / ZoomY
            TmpRedC = CLng(TargetArray(x + ZoomX, y, 0)) + CLng(TargetArray(x, y + ZoomY, 0)) - CLng(TargetArray(x
ZoomX, y + ZoomY, 0))
            TmpGreenA = (CLng(TargetArray(x + ZoomX, y + ZoomY, 1)) - CLng(TargetArray(x, y + ZoomY, 1))) / ZoomX
            TmpGreenB = (CLng(TargetArray(x + ZoomX, y + ZoomY, 1)) - CLng(TargetArray(x + ZoomX, y, 1))) / ZoomY
            TmpGreenC = CLng(TargetArray(x + ZoomX, y, 1)) + CLng(TargetArray(x, y + ZoomY, 1)) - CLng(TargetArray(x
ZoomX, y + ZoomY, 1))
            TmpBlueA = (CLng(TargetArray(x + ZoomX, y + ZoomY, 2)) - CLng(TargetArray(x, y + ZoomY, 2))) / ZoomX
            TmpBlueB = (CLng(TargetArray(x + ZoomX, y + ZoomY, 2)) - CLng(TargetArray(x + ZoomX, y, 2))) / ZoomY
            TmpBlueC = CLng(TargetArray(x + ZoomX, y, 2)) + CLng(TargetArray(x, y + ZoomY, 2)) - CLng(TargetArray(x
ZoomX, y + ZoomY, 2))
        ElseIf band <> DefAllChannel Then
            TmpA = (CLng(TargetArray(x + ZoomX, y + ZoomY, band)) - CLng(TargetArray(x, y + ZoomY, band))) / ZoomX
            TmpB = (CLng(TargetArray(x + ZoomX, y + ZoomY, band)) - CLng(TargetArray(x + ZoomX, y, band))) / ZoomY
            TmpC = CLng(TargetArray(x + ZoomX, y, band)) + CLng(TargetArray(x, y + ZoomY, band)) - CLng(TargetArray(x
ZoomX, y + ZoomY, band))
        End If
        For j = 0 To ZoomY
            For i = ZoomX To ZoomX - j Step -1
                If Not ((i = ZoomX And j = ZoomY) Or (i = ZoomX And j = 0) Or (i = 0 And j = ZoomY)) Then
                    If band = AllChannel Then
                        ImageTmpRed = (TmpRedA * i) + (TmpRedB * j) + TmpRedC
                        ImageTmpGreen = (TmpGreenA * i) + (TmpGreenB * j) + TmpGreenC
                        ImageTmpBlue = (TmpBlueA * i) + (TmpBlueB * j) + TmpBlueC
                        If ImageTmpRed < 0 Then ImageTmpRed = 0
                        If ImageTmpGreen < 0 Then ImageTmpGreen = 0
                        If ImageTmpBlue < 0 Then ImageTmpBlue = 0
                        If ImageTmpRed > 255 Then ImageTmpRed = 255
```

```
                    If ImageTmpGreen > 255 Then ImageTmpGreen = 255
                    If ImageTmpBlue > 255 Then ImageTmpBlue = 255
                    TargetArray(x + i, y + j, 0) = ImageTmpRed
                    TargetArray(x + i, y + j, 1) = ImageTmpGreen
                    TargetArray(x + i, y + j, 2) = ImageTmpBlue
                ElseIf band <> DefAllChannel Then
                    ImageTmp = (TmpA * i) + (TmpB * j) + TmpC
                    If ImageTmp < 0 Then ImageTmp = 0
                    If ImageTmp > 255 Then ImageTmp = 255
                    TargetArray(x + i, y + j, band) = ImageTmp
                End If
            End If
        Next i
    Next j
  Next x
  DoEvents
Next y
End Sub

Public Sub AddImage(Sx As Long, Sy As Long, Ex As Long, Ey As Long, band As Channel, SourceArray1() As Byte, SourceArray2()
As Byte, TargetArray() As Byte)
  For y = Sy To Ey
    For x = Sx To Ex
      If band = AllChannel Then
        If CLng(SourceArray1(x, y, 0)) + CLng(SourceArray2(x, y, 0)) > 255 Then
          TargetArray(x, y, 0) = 255
        Else
          TargetArray(x, y, 0) = SourceArray1(x, y, 0) + SourceArray2(x, y, 0)
        End If
        If CLng(SourceArray1(x, y, 1)) + CLng(SourceArray2(x, y, 1)) > 255 Then
          TargetArray(x, y, 1) = 255
        Else
          TargetArray(x, y, 1) = SourceArray1(x, y, 1) + SourceArray2(x, y, 1)
        End If
        If CLng(SourceArray1(x, y, 2)) + CLng(SourceArray2(x, y, 2)) > 255 Then
          TargetArray(x, y, 2) = 255
        Else
          TargetArray(x, y, 2) = SourceArray1(x, y, 2) + SourceArray2(x, y, 2)
        End If
      ElseIf band <> DefAllChannel Then
        If CLng(SourceArray1(x, y, band)) + CLng(SourceArray2(x, y, band)) > 255 Then
          TargetArray(x, y, band) = 255
        Else
          TargetArray(x, y, band) = SourceArray1(x, y, band) + SourceArray2(x, y, band)
        End If
      End If
    Next x
  Next y
End Sub

Public Sub HistoEqual(Sx As Long, Sy As Long, Ex As Long, Ey As Long, band As Channel, SourceArray() As Byte, TargetArray() A
Byte)
  Dim k, kR, kG, kB As Long
  Dim Sum, SumR, SumG, SumB As Long
  Dim TotalPixels As Long
  Dim HistoR(255) As Long
  Dim HistoG(255) As Long
  Dim HistoB(255) As Long
  Dim Histo(255) As Long
```

```
Dim SumOfHistoR(255) As Long
Dim SumOfHistoG(255) As Long
Dim SumOfHistoB(255) As Long
Dim SumOfHisto(255) As Long
Dim z As Double
Dim i, j As Long

If band = AllChannel Then
    kR = 0: SumR = 0
    kG = 0: SumG = 0
    kB = 0: SumB = 0
    TotalPixels = 0
ElseIf band <> DefAllChannel Then
    k = 0: Sum = 0: TotalPixels = 0
End If

For i = 0 To 255
    If band = AllChannel Then
        HistoR(i) = 0
        HistoG(i) = 0
        HistoB(i) = 0
        SumOfHistoR(i) = 0
        SumOfHistoG(i) = 0
        SumOfHistoB(i) = 0
    ElseIf band <> DefAllChannel Then
        Histo(i) = 0
        SumOfHisto(i) = 0
    End If
Next i
For j = Sy To Ey - 1
    For i = Sx To Ex - 1
        If band = AllChannel Then
            kR = SourceArray(i, j, 0)
            kG = SourceArray(i, j, 1)
            kB = SourceArray(i, j, 2)
            HistoR(kR) = HistoR(kR) + 1
            HistoG(kG) = HistoG(kG) + 1
            HistoB(kB) = HistoB(kB) + 1
        ElseIf band <> DefAllChannel Then
            k = SourceArray(i, j, band)
            Histo(k) = Histo(k) + 1
        End If
    Next i
Next j
For i = 0 To 255
    If band = AllChannel Then
        SumR = SumR + HistoR(i)
        SumG = SumG + HistoG(i)
        SumB = SumB + HistoB(i)
        SumOfHistoR(i) = SumR
        SumOfHistoG(i) = SumG
        SumOfHistoB(i) = SumB
    ElseIf band <> DefAllChannel Then
        Sum = Sum + Histo(i)
        SumOfHisto(i) = Sum
    End If
Next i
TotalPixels = (Ex - Sx) * (Ey - Sy)
z = 255# / CDbl(TotalPixels)
```

```
    For j = Sy To Ey - 1
        For i = Sx To Ex - 1
            If band = AllChannel Then
                kR = SourceArray(i, j, 0)
                kG = SourceArray(i, j, 1)
                kB = SourceArray(i, j, 2)
                TargetArray(i, j, 0) = CByte(CDbl(SumOfHistoR(kR)) * z)
                TargetArray(i, j, 1) = CByte(CDbl(SumOfHistoG(kG)) * z)
                TargetArray(i, j, 2) = CByte(CDbl(SumOfHistoB(kB)) * z)
            ElseIf band <> DefAllChannel Then
                k = SourceArray(i, j, band)
                TargetArray(i, j, band) = CByte(CDbl(SumOfHisto(k)) * z)
            End If
        Next i
    Next j
End Sub

Public Sub HistoStretch(Sx As Long, Sy As Long, Ex As Long, Ey As Long, band As Channel, SourceArray() As Byte, TargetArray()
As Byte)
    Dim i, j As Long
    Dim HistoR(255) As Long
    Dim HistoG(255) As Long
    Dim HistoB(255) As Long
    Dim Histo(255) As Long
    Dim LUTR(255) As Long
    Dim LUTG(255) As Long
    Dim LUTB(255) As Long
    Dim LUT(255) As Long
    Dim LowThreshR, HighThreshR As Long
    Dim LowThreshG, HighThreshG As Long
    Dim LowThreshB, HighThreshB As Long
    Dim LowThresh, HighThresh As Long
    Dim ScaleFactorR As Double
    Dim ScaleFactorG As Double
    Dim ScaleFactorB As Double
    Dim ScaleFactor As Double
    If band = AllChannel Then
        LowThreshR = 0
        LowThreshG = 0
        LowThreshB = 0
        HighThreshR = 255
        HighThreshG = 255
        HighThreshB = 255
    ElseIf band <> DefAllChannel Then
        LowThresh = 0
        HighThresh = 255
    End If

    For i = 0 To 255
        If band = AllChannel Then
            HistoR(i) = 0
            HistoG(i) = 0
            HistoB(i) = 0
        ElseIf band <> DefAllChannel Then
            Histo(i) = 0
        End If
    Next i
    For i = Sx To Ex - 1
        For j = Sy To Ey - 1
```

```
        If band = AllChannel Then
            HistoR(SourceArray(i, j, 0)) = HistoR(SourceArray(i, j, 0)) + 1
            HistoG(SourceArray(i, j, 1)) = HistoG(SourceArray(i, j, 1)) + 1
            HistoB(SourceArray(i, j, 2)) = HistoB(SourceArray(i, j, 2)) + 1
        ElseIf band <> DefAllChannel Then
            Histo(SourceArray(i, j, band)) = Histo(SourceArray(i, j, band)) + 1
        End If
    Next j
Next i
If band = AllChannel Then
    For i = 0 To 255
        If HistoR(i) Then
            LowThreshR = i
            Exit For
        End If
    Next i
    For i = 0 To 255
        If HistoG(i) Then
            LowThreshG = i
            Exit For
        End If
    Next i
    For i = 0 To 255
        If HistoB(i) Then
            LowThreshB = i
            Exit For
        End If
    Next i

    For i = 255 To 0 Step -1
        If HistoR(i) Then
            HighThreshR = i
            Exit For
        End If
    Next i
    For i = 255 To 0 Step -1
        If HistoG(i) Then
            HighThreshG = i
            Exit For
        End If
    Next i
    For i = 255 To 0 Step -1
        If HistoB(i) Then
            HighThreshB = i
            Exit For
        End If
    Next i

    For i = 0 To LowThreshR - 1
        LUTR(i) = 0
    Next i
    For i = 0 To LowThreshG - 1
        LUTG(i) = 0
    Next i
    For i = 0 To LowThreshB - 1
        LUTB(i) = 0
    Next i

    For i = 255 To HighThreshR + 1 Step -1
```

```
        LUTR(i) = 255
    Next i
    For i = 255 To HighThreshG + 1 Step -1
        LUTG(i) = 255
    Next i
    For i = 255 To HighThreshB + 1 Step -1
        LUTB(i) = 255
    Next i

    ScaleFactorR = 255# / CDbl(HighThreshR - LowThreshR)
    For i = LowThreshR To HighThreshR
        LUTR(i) = CByte(CDbl(i - LowThreshR) * ScaleFactorR)
    Next i

    ScaleFactorG = 255# / CDbl(HighThreshG - LowThreshG)
    For i = LowThreshG To HighThreshG
        LUTG(i) = CByte(CDbl(i - LowThreshG) * ScaleFactorG)
    Next i

    ScaleFactorB = 255# / CDbl(HighThreshB - LowThreshB)
    For i = LowThreshB To HighThreshB
        LUTB(i) = CByte(CDbl(i - LowThreshB) * ScaleFactorB)
    Next i
ElseIf band <> DefAllChannel Then
    For i = 0 To 255
        If Histo(i) Then
            LowThresh = i
            Exit For
        End If
    Next i
    For i = 255 To 0 Step -1
        If Histo(i) Then
            HighThresh = i
            Exit For
        End If
    Next i
    For i = 0 To LowThresh - 1
        LUT(i) = 0
    Next i
    For i = 255 To HighThresh + 1 Step -1
        LUT(i) = 255
    Next i
    Scale_Factor = 255# / CDbl(HighThresh - LowThresh)
    For i = LowThresh To HighThresh
        LUT(i) = CByte(CDbl(i - LowThresh) * Scale_Factor)
    Next i
End If
For i = Sx To Ex
    For j = Sy To Ey
        If band = AllChannel Then
            TargetArray(i, j, 0) = LUTR(SourceArray(i, j, 0))
            TargetArray(i, j, 1) = LUTG(SourceArray(i, j, 1))
            TargetArray(i, j, 2) = LUTB(SourceArray(i, j, 2))
        ElseIf band <> DefAllChannel Then
            TargetArray(i, j, band) = LUT(SourceArray(i, j, band))
        End If
    Next j
Next i
```

```
End Sub

Public Sub LogarithmOP(Sx As Long, Sy As Long, Ex As Long, Ey As Long, band As Channel, SourceArray() As Byte, TargetArray()
As Byte)
    Dim i, j As Long
    Dim ValueR As Double
    Dim ValueG As Double
    Dim ValueB As Double
    Dim Value As Double
    Dim c As Double
    If band = AllChannel Then
        ValueR = 0#
        ValueG = 0#
        ValueB = 0#
        c = 255# / Log(1# + 255#)
    ElseIf band <> DefAllChannel Then
        Value = 0#
        c = 255# / Log(1# + 255#)
    End If
    For i = Sx To Ex
        For j = Sy To Ey
            If band = AllChannel Then
                ValueR = c * Log(1# + CDbl(SourceArray(i, j, 0)))
                ValueG = c * Log(1# + CDbl(SourceArray(i, j, 1)))
                ValueB = c * Log(1# + CDbl(SourceArray(i, j, 2)))
            ElseIf band <> DefAllChannel Then
                Value = c * Log(1# + CDbl(SourceArray(i, j, band)))
            End If

            If band = AllChannel Then
                If ValueR > 255# Then
                    ValueR = 255#
                End If
                If ValueG > 255# Then
                    ValueG = 255#
                End If
                If ValueB > 255# Then
                    ValueB = 255#
                End If
                TargetArray(i, j, 0) = CInt(ValueR)
                TargetArray(i, j, 1) = CInt(ValueG)
                TargetArray(i, j, 2) = CInt(ValueB)
            ElseIf band <> DefAllChannel Then
                If Value > 255# Then
                    Value = 255#
                End If
                TargetArray(i, j, band) = CInt(Value)
            End If
        Next j
    Next i
End Sub

Public Sub Sobel(Sx As Long, Sy As Long, Ex As Long, Ey As Long, band As Channel, SourceArray() As Byte, TargetArray() As Byte)
    Dim x As Long
    Dim y As Long
    Dim i As Long
    Dim j As Long
    Dim CenterValueRed1 As Long
    Dim CenterValueGreen1 As Long
```

```
Dim CenterValueBlue1 As Long
Dim CenterValue1 As Long
Dim CenterValueRed2 As Long
Dim CenterValueGreen2 As Long
Dim CenterValueBlue2 As Long
Dim CenterValue2 As Long
Dim SumRed As Long
Dim SumGreen As Long
Dim SumBlue As Long
Dim Sum As Long
Dim mask1(2, 2) As Long
Dim mask2(2, 2) As Long
Dim tmpArray(639, 479, 2) As Byte

mask1(0, 0) = -1: mask1(0, 1) = 0:  mask1(0, 2) = 1
mask1(1, 0) = -2: mask1(1, 1) = 0:  mask1(1, 2) = 2
mask1(2, 0) = -1: mask1(2, 1) = 0:  mask1(2, 2) = 1

mask2(0, 0) = 1:  mask2(0, 1) = 2:  mask2(0, 2) = 1
mask2(1, 0) = 0:  mask2(1, 1) = 0:  mask2(1, 2) = 0
mask2(2, 0) = -1: mask2(2, 1) = -2: mask2(2, 2) = -1

Sum = 0
SumRed = 0
SumGreen = 0
SumBlue = 0
CenterValue1 = 0
CenterValueRed1 = 0
CenterValueGreen1 = 0
CenterValueBlue1 = 0
CenterValue2 = 0
CenterValueRed2 = 0
CenterValueGreen2 = 0
CenterValueBlue2 = 0
For x = Sx + 1 To Ex - 3
    For y = Sy + 1 To Ey - 3
        For i = 0 To 2
            For j = 0 To 2
                If band = AllChannel Then
                    CenterValueRed1 = CenterValueRed1 + CLng(SourceArray(i + x, j + y, 0)) * mask1(i, j)
                    CenterValueGreen1 = CenterValueGreen1 + CLng(SourceArray(i + x, j + y, 1)) * mask1(i, j)
                    CenterValueBlue1 = CenterValueBlue1 + CLng(SourceArray(i + x, j + y, 2)) * mask1(i, j)
                    CenterValueRed2 = CenterValueRed2 + CLng(SourceArray(i + x, j + y, 0)) * mask2(i, j)
                    CenterValueGreen2 = CenterValueGreen2 + CLng(SourceArray(i + x, j + y, 1)) * mask2(i, j)
                    CenterValueBlue2 = CenterValueBlue2 + CLng(SourceArray(i + x, j + y, 2)) * mask2(i, j)
                ElseIf band <> DefAllChannel Then
                    CenterValue1 = CenterValue1 + CLng(SourceArray(i + x, j + y, band)) * mask1(i, j)
                    CenterValue2 = CenterValue2 + CLng(SourceArray(i + x, j + y, band)) * mask2(i, j)
                End If
            Next j
        Next i
        If band = AllChannel Then
            SumRed = Abs(CenterValueRed1) + Abs(CenterValueRed2)
            SumGreen = Abs(CenterValueGreen1) + Abs(CenterValueGreen2)
            SumBlue = Abs(CenterValueBlue1) + Abs(CenterValueBlue2)
            If SumRed > 255 Then SumRed = 255
            If SumGreen > 255 Then SumGreen = 255
            If SumBlue > 255 Then SumBlue = 255
            tmpArray(x + 1, y + 1, 0) = CByte(SumRed)
```

```
                tmpArray(x + 1, y + 1, 1) = CByte(SumGreen)
                tmpArray(x + 1, y + 1, 2) = CByte(SumBlue)
                CenterValueRed1 = 0
                CenterValueGreen1 = 0
                CenterValueBlue1 = 0
                CenterValueRed2 = 0
                CenterValueGreen2 = 0
                CenterValueBlue2 = 0
                SumRed = 0
                SumGreen = 0
                SumBlue = 0
            ElseIf band <> DefAllChannel Then
                Sum = Abs(CenterValue1) + Abs(CenterValue2)
                If Sum > 255 Then Sum = 255
                tmpArray(x + 1, y + 1, band) = CByte(Sum)
                CenterValue = 0
                Sum = 0
            End If
        Next y
    Next x
    For x = Sx + 1 To Ex - 1
        For y = Sy + 1 To Ey - 1
            If band = AllChannel Then
                TargetArray(x, y, 0) = tmpArray(x, y, 0)
                TargetArray(x, y, 1) = tmpArray(x, y, 1)
                TargetArray(x, y, 2) = tmpArray(x, y, 2)
            ElseIf band <> DefAllChannel Then
                TargetArray(x, y, band) = tmpArray(x, y, band)
            End If
        Next y
    Next x
End Sub

Public Sub Prewitt(Sx As Long, Sy As Long, Ex As Long, Ey As Long, band As Channel, SourceArray() As Byte, TargetArray() A
Byte)
    Dim x As Long
    Dim y As Long
    Dim i As Long
    Dim j As Long
    Dim CenterValueRed1 As Long
    Dim CenterValueGreen1 As Long
    Dim CenterValueBlue1 As Long
    Dim CenterValue1 As Long
    Dim CenterValueRed2 As Long
    Dim CenterValueGreen2 As Long
    Dim CenterValueBlue2 As Long
    Dim CenterValue2 As Long
    Dim SumRed As Long
    Dim SumGreen As Long
    Dim SumBlue As Long
    Dim Sum As Long
    Dim mask1(2, 2) As Long
    Dim mask2(2, 2) As Long
    Dim tmpArray(639, 479, 2) As Byte

    mask1(0, 0) = -1: mask1(0, 1) = 0:  mask1(0, 2) = 1
    mask1(1, 0) = -1: mask1(1, 1) = 0:  mask1(1, 2) = 1
    mask1(2, 0) = -1: mask1(2, 1) = 0:  mask1(2, 2) = 1
```

```
mask2(0, 0) = 1:  mask2(0, 1) = 1:  mask2(0, 2) = 1
mask2(1, 0) = 0:  mask2(1, 1) = 0:  mask2(1, 2) = 0
mask2(2, 0) = -1: mask2(2, 1) = -1: mask2(2, 2) = -1

Sum = 0
SumRed = 0
SumGreen = 0
SumBlue = 0
CenterValue1 = 0
CenterValueRed1 = 0
CenterValueGreen1 = 0
CenterValueBlue1 = 0
CenterValue2 = 0
CenterValueRed2 = 0
CenterValueGreen2 = 0
CenterValueBlue2 = 0
For x = Sx + 1 To Ex - 3
    For y = Sy + 1 To Ey - 3
        For i = 0 To 2
            For j = 0 To 2
                If band = AllChannel Then
                    CenterValueRed1 = CenterValueRed1 + CLng(SourceArray(i + x, j + y, 0)) * mask1(i, j)
                    CenterValueGreen1 = CenterValueGreen1 + CLng(SourceArray(i + x, j + y, 1)) * mask1(i, j)
                    CenterValueBlue1 = CenterValueBlue1 + CLng(SourceArray(i + x, j + y, 2)) * mask1(i, j)
                    CenterValueRed2 = CenterValueRed2 + CLng(SourceArray(i + x, j + y, 0)) * mask2(i, j)
                    CenterValueGreen2 = CenterValueGreen2 + CLng(SourceArray(i + x, j + y, 1)) * mask2(i, j)
                    CenterValueBlue2 = CenterValueBlue2 + CLng(SourceArray(i + x, j + y, 2)) * mask2(i, j)
                ElseIf band <> DefAllChannel Then
                    CenterValue1 = CenterValue1 + CLng(SourceArray(i + x, j + y, band)) * mask1(i, j)
                    CenterValue2 = CenterValue2 + CLng(SourceArray(i + x, j + y, band)) * mask2(i, j)
                End If
            Next j
        Next i
        If band = AllChannel Then
            SumRed = Abs(CenterValueRed1) + Abs(CenterValueRed2)
            SumGreen = Abs(CenterValueGreen1) + Abs(CenterValueGreen2)
            SumBlue = Abs(CenterValueBlue1) + Abs(CenterValueBlue2)
            If SumRed > 255 Then SumRed = 255
            If SumGreen > 255 Then SumGreen = 255
            If SumBlue > 255 Then SumBlue = 255
            tmpArray(x + 1, y + 1, 0) = CByte(SumRed)
            tmpArray(x + 1, y + 1, 1) = CByte(SumGreen)
            tmpArray(x + 1, y + 1, 2) = CByte(SumBlue)
            CenterValueRed1 = 0
            CenterValueGreen1 = 0
            CenterValueBlue1 = 0
            CenterValueRed2 = 0
            CenterValueGreen2 = 0
            CenterValueBlue2 = 0
            SumRed = 0
            SumGreen = 0
            SumBlue = 0
        ElseIf band <> DefAllChannel Then
            Sum = Abs(CenterValue1) + Abs(CenterValue2)
            If Sum > 255 Then Sum = 255
            tmpArray(x + 1, y + 1, band) = CByte(Sum)
            CenterValue = 0
            Sum = 0
        End If
```

```
        Next y
    Next x
    For x = Sx + 1 To Ex - 1
        For y = Sy + 1 To Ey - 1
            If band = AllChannel Then
                TargetArray(x, y, 0) = tmpArray(x, y, 0)
                TargetArray(x, y, 1) = tmpArray(x, y, 1)
                TargetArray(x, y, 2) = tmpArray(x, y, 2)
            ElseIf band <> DefAllChannel Then
                TargetArray(x, y, band) = tmpArray(x, y, band)
            End If
        Next y
    Next x
End Sub

Public Sub Robert(Sx As Long, Sy As Long, Ex As Long, Ey As Long, band As Channel, SourceArray() As Byte, TargetArray() A
Byte)
    Dim x As Long
    Dim y As Long
    Dim i As Long
    Dim j As Long
    Dim CenterValueRed1 As Long
    Dim CenterValueGreen1 As Long
    Dim CenterValueBlue1 As Long
    Dim CenterValue1 As Long
    Dim CenterValueRed2 As Long
    Dim CenterValueGreen2 As Long
    Dim CenterValueBlue2 As Long
    Dim CenterValue2 As Long
    Dim SumRed As Long
    Dim SumGreen As Long
    Dim SumBlue As Long
    Dim Sum As Long
    Dim mask1(2, 2) As Long
    Dim mask2(2, 2) As Long
    Dim tmpArray(639, 479, 2) As Byte
    Sum = 0
    mask1(0, 0) = 0:  mask1(0, 1) = 0: mask1(0, 2) = -1
    mask1(1, 0) = 0:  mask1(1, 1) = 1: mask1(1, 2) = 0
    mask1(2, 0) = 0:  mask1(2, 1) = 0: mask1(2, 2) = 0

    mask2(0, 0) = -1: mask2(0, 1) = 0: mask2(0, 2) = 0
    mask2(1, 0) = 0:  mask2(1, 1) = 1: mask2(1, 2) = 0
    mask2(2, 0) = 0:  mask2(2, 1) = 0: mask2(2, 2) = 0
    Sum = 0
    SumRed = 0
    SumGreen = 0
    SumBlue = 0
    CenterValue1 = 0
    CenterValueRed1 = 0
    CenterValueGreen1 = 0
    CenterValueBlue1 = 0
    CenterValue2 = 0
    CenterValueRed2 = 0
    CenterValueGreen2 = 0
    CenterValueBlue2 = 0
    For x = Sx + 1 To Ex - 3
        For y = Sy + 1 To Ey - 3
            For i = 0 To 2
```

```
            For j = 0 To 2
                If band = AllChannel Then
                    CenterValueRed1 = CenterValueRed1 + CLng(SourceArray(i + x, j + y, 0)) * mask1(i, j)
                    CenterValueGreen1 = CenterValueGreen1 + CLng(SourceArray(i + x, j + y, 1)) * mask1(i, j)
                    CenterValueBlue1 = CenterValueBlue1 + CLng(SourceArray(i + x, j + y, 2)) * mask1(i, j)
                    CenterValueRed2 = CenterValueRed2 + CLng(SourceArray(i + x, j + y, 0)) * mask2(i, j)
                    CenterValueGreen2 = CenterValueGreen2 + CLng(SourceArray(i + x, j + y, 1)) * mask2(i, j)
                    CenterValueBlue2 = CenterValueBlue2 + CLng(SourceArray(i + x, j + y, 2)) * mask2(i, j)
                ElseIf band <> DefAllChannel Then
                    CenterValue1 = CenterValue1 + CLng(SourceArray(i + x, j + y, band)) * mask1(i, j)
                    CenterValue2 = CenterValue2 + CLng(SourceArray(i + x, j + y, band)) * mask2(i, j)
                End If
            Next j
        Next i
        If band = AllChannel Then
            SumRed = Abs(CenterValueRed1) + Abs(CenterValueRed2)
            SumGreen = Abs(CenterValueGreen1) + Abs(CenterValueGreen2)
            SumBlue = Abs(CenterValueBlue1) + Abs(CenterValueBlue2)
            If SumRed > 255 Then SumRed = 255
            If SumGreen > 255 Then SumGreen = 255
            If SumBlue > 255 Then SumBlue = 255
            tmpArray(x + 1, y + 1, 0) = CByte(SumRed)
            tmpArray(x + 1, y + 1, 1) = CByte(SumGreen)
            tmpArray(x + 1, y + 1, 2) = CByte(SumBlue)
            CenterValueRed1 = 0
            CenterValueGreen1 = 0
            CenterValueBlue1 = 0
            CenterValueRed2 = 0
            CenterValueGreen2 = 0
            CenterValueBlue2 = 0
            SumRed = 0
            SumGreen = 0
            SumBlue = 0
        ElseIf band <> DefAllChannel Then
            Sum = Abs(CenterValue1) + Abs(CenterValue2)
            If Sum > 255 Then Sum = 255
            tmpArray(x + 1, y + 1, band) = CByte(Sum)
            CenterValue = 0
            Sum = 0
        End If
    Next y
Next x
For x = Sx + 1 To Ex - 1
    For y = Sy + 1 To Ey - 1
        If band = AllChannel Then
            TargetArray(x, y, 0) = tmpArray(x, y, 0)
            TargetArray(x, y, 1) = tmpArray(x, y, 1)
            TargetArray(x, y, 2) = tmpArray(x, y, 2)
        ElseIf band <> DefAllChannel Then
            TargetArray(x, y, band) = tmpArray(x, y, band)
        End If
    Next y
Next x
End Sub

Public Sub Lapliacian(Sx As Long, Sy As Long, Ex As Long, Ey As Long, band As Channel, SourceArray() As Byte, TargetArray() A
Byte)
    Dim x As Long
    Dim y As Long
```

```
Dim i As Long
Dim j As Long
Dim CenterValueRed As Long
Dim CenterValueGreen As Long
Dim CenterValueBlue As Long
Dim CenterValue As Long
Dim SumRed As Long
Dim SumGreen As Long
Dim SumBlue As Long
Dim Sum As Long
Dim mask(2, 2) As Long
Dim tmpArray(639, 479, 2) As Byte

mask(0, 0) = -1: mask(0, 1) = -1: mask(0, 2) = -1
mask(1, 0) = -1: mask(1, 1) = 8:  mask(1, 2) = -1
mask(2, 0) = -1: mask(2, 1) = -1: mask(2, 2) = -1

Sum = 0
CenterValue = 0
CenterValueRed = 0
CenterValueGreen = 0
CenterValueBlue = 0

For x = Sx + 1 To Ex - 3
    For y = Sy + 1 To Ey - 3
        For i = 0 To 2
            For j = 0 To 2
                If band = AllChannel Then
                    CenterValueRed = CenterValueRed + CLng(SourceArray(i + x, j + y, 0)) * mask(i, j)
                    CenterValueGreen = CenterValueGreen + CLng(SourceArray(i + x, j + y, 1)) * mask(
                    CenterValueBlue = CenterValueBlue + CLng(SourceArray(i + x, j + y, 2)) * mask(i, j
                ElseIf band <> DefAllChannel Then
                    CenterValue = CenterValue + CLng(SourceArray(i + x, j + y, band)) * mask(i, j)
                End If
            Next j
        Next i
        If band = AllChannel Then
            SumRed = Abs(CenterValueRed)
            SumGreen = Abs(CenterValueGreen)
            SumBlue = Abs(CenterValueBlue)
            If SumRed > 255 Then SumRed = 255
            If SumGreen > 255 Then SumGreen = 255
            If SumBlue > 255 Then SumBlue = 255
            tmpArray(x, y, 0) = CByte(SumRed)
            tmpArray(x, y, 1) = CByte(SumGreen)
            tmpArray(x, y, 2) = CByte(SumBlue)
            CenterValueRed = 0
            CenterValueGreen = 0
            CenterValueBlue = 0
        ElseIf band <> DefAllChannel Then
            Sum = Abs(CenterValue)
            If Sum > 255 Then Sum = 255
            tmpArray(x, y, band) = CByte(Sum)
            CenterValue = 0
        End If
    Next y
Next x
For x = Sx + 1 To Ex - 1
    For y = Sy + 1 To Ey - 1
```

```
            If band = AllChannel Then
                TargetArray(x, y, 0) = tmpArray(x, y, 0)
                TargetArray(x, y, 1) = tmpArray(x, y, 1)
                TargetArray(x, y, 2) = tmpArray(x, y, 2)
            ElseIf band <> DefAllChannel Then
                TargetArray(x, y, band) = tmpArray(x, y, band)
            End If
        Next y
    Next x
End Sub

Public Sub GaussianConv(Sx As Long, Sy As Long, Ex As Long, Ey As Long, band As Channel, SourceArray() As Byte, TargetArray()
As Byte)
    Dim x As Long
    Dim y As Long
    Dim i As Long
    Dim j As Long
    Dim CenterValueRed As Long
    Dim CenterValueGreen As Long
    Dim CenterValueBlue As Long
    Dim CenterValue As Long
    Dim SumRed As Double
    Dim SumGreen As Double
    Dim SumBlue As Double
    Dim Sum As Double
    Dim mask(4, 4) As Long
    Dim tmpArray(639, 479, 2) As Byte

    mask(0, 0) = 2: mask(0, 1) = 4:  mask(0, 2) = 5:  mask(0, 3) = 4:  mask(0, 4) = 2
    mask(1, 0) = 4: mask(1, 1) = 9:  mask(1, 2) = 12: mask(1, 3) = 9:  mask(1, 4) = 4
    mask(2, 0) = 5: mask(2, 1) = 12: mask(2, 2) = 15: mask(2, 3) = 12: mask(2, 4) = 5
    mask(3, 0) = 4: mask(3, 1) = 9:  mask(3, 2) = 12: mask(3, 3) = 9:  mask(3, 4) = 4
    mask(4, 0) = 2: mask(4, 1) = 4:  mask(4, 2) = 5:  mask(4, 3) = 4:  mask(4, 4) = 2

    Sum = 0
    CenterValue = 0
    CenterValueRed = 0
    CenterValueGreen = 0
    CenterValueBlue = 0

    For x = Sx + 1 To Ex - 5
        For y = Sy + 1 To Ey - 5
            For i = 0 To 4
                For j = 0 To 4
                    If band = AllChannel Then
                        CenterValueRed = CenterValueRed + CLng(SourceArray(i + x, j + y, 0)) * mask(i, j)
                        CenterValueGreen = CenterValueGreen + CLng(SourceArray(i + x, j + y, 1)) * mask(i, j)
                        CenterValueBlue = CenterValueBlue + CLng(SourceArray(i + x, j + y, 2)) * mask(i, j)
                    ElseIf band <> DefAllChannel Then
                        CenterValue = CenterValue + CLng(SourceArray(i + x, j + y, band)) * mask(i, j)
                    End If
                Next j
            Next i
            If band = AllChannel Then
                SumRed = CDbl(Abs(CenterValueRed) / 115)
                SumGreen = CDbl(Abs(CenterValueGreen) / 115)
                SumBlue = CDbl(Abs(CenterValueBlue) / 115)
                If SumRed > 255 Then SumRed = 255
                If SumGreen > 255 Then SumGreen = 255
```

```
                If SumBlue > 255 Then SumBlue = 255
                tmpArray(x + 2, y + 2, 0) = CByte(SumRed)
                tmpArray(x + 2, y + 2, 1) = CByte(SumGreen)
                tmpArray(x + 2, y + 2, 2) = CByte(SumBlue)
                CenterValueRed = 0
                CenterValueGreen = 0
                CenterValueBlue = 0
            ElseIf band <> DefAllChannel Then
                Sum = CDbl(Abs(CenterValue) / 115)
                If Sum > 255 Then Sum = 255
                tmpArray(x + 2, y + 2, band) = CByte(Sum)
                CenterValue = 0
            End If
        Next y
    Next x
    For x = Sx + 1 To Ex - 1
        For y = Sy + 1 To Ey - 1
            If band = AllChannel Then
                TargetArray(x, y, 0) = tmpArray(x, y, 0)
                TargetArray(x, y, 1) = tmpArray(x, y, 1)
                TargetArray(x, y, 2) = tmpArray(x, y, 2)
            ElseIf band <> DefAllChannel Then
                TargetArray(x, y, band) = tmpArray(x, y, band)
            End If
        Next y
    Next x
End Sub

Public Sub LowPassFilter(Sx As Long, Sy As Long, Ex As Long, Ey As Long, band As Channel, SourceArray() As Byte, TargetArray(
As Byte)
    Dim x As Long
    Dim y As Long
    Dim i As Long
    Dim j As Long
    Dim CenterValueRed As Long
    Dim CenterValueGreen As Long
    Dim CenterValueBlue As Long
    Dim CenterValue As Long
    Dim SumRed As Double
    Dim SumGreen As Double
    Dim SumBlue As Double
    Dim Sum As Double
    Dim mask(4, 4) As Long
    Dim tmpArray(639, 479, 2) As Byte

    mask(0, 0) = 1: mask(0, 1) = 1: mask(0, 2) = 1
    mask(1, 0) = 1: mask(1, 1) = 1: mask(1, 2) = 1
    mask(2, 0) = 1: mask(2, 1) = 1: mask(2, 2) = 1

    Sum = 0
    CenterValue = 0
    CenterValueRed = 0
    CenterValueGreen = 0
    CenterValueBlue = 0

    For x = Sx + 1 To Ex - 3
        For y = Sy + 1 To Ey - 3
            For i = 0 To 2
                For j = 0 To 2
```

```
                If band = AllChannel Then
                    CenterValueRed = CenterValueRed + CLng(SourceArray(i + x, j + y, 0)) * mask(i, j)
                    CenterValueGreen = CenterValueGreen + CLng(SourceArray(i + x, j + y, 1)) * mask(i, j)
                    CenterValueBlue = CenterValueBlue + CLng(SourceArray(i + x, j + y, 2)) * mask(i, j)
                ElseIf band <> DefAllChannel Then
                    CenterValue = CenterValue + CLng(SourceArray(i + x, j + y, band)) * mask(i, j)
                End If
            Next j
        Next i
        If band = AllChannel Then
            SumRed = CDbl(Abs(CenterValueRed) / 9)
            SumGreen = CDbl(Abs(CenterValueGreen) / 9)
            SumBlue = CDbl(Abs(CenterValueBlue) / 9)
            If SumRed > 255 Then SumRed = 255
            If SumGreen > 255 Then SumGreen = 255
            If SumBlue > 255 Then SumBlue = 255
            tmpArray(x + 1, y + 1, 0) = CByte(SumRed)
            tmpArray(x + 1, y + 1, 1) = CByte(SumGreen)
            tmpArray(x + 1, y + 1, 2) = CByte(SumBlue)
            CenterValueRed = 0
            CenterValueGreen = 0
            CenterValueBlue = 0
        ElseIf band <> DefAllChannel Then
            Sum = CDbl(Abs(CenterValue) / 9)
            If Sum > 255 Then Sum = 255
            tmpArray(x + 1, y + 1, band) = CByte(Sum)
            CenterValue = 0
        End If
    Next y
Next x
For x = Sx + 1 To Ex - 1
    For y = Sy + 1 To Ey - 1
        If band = AllChannel Then
            TargetArray(x, y, 0) = tmpArray(x, y, 0)
            TargetArray(x, y, 1) = tmpArray(x, y, 1)
            TargetArray(x, y, 2) = tmpArray(x, y, 2)
        ElseIf band <> DefAllChannel Then
            TargetArray(x, y, band) = tmpArray(x, y, band)
        End If
    Next y
Next x
End Sub

Public Sub HighPassFilter(Sx As Long, Sy As Long, Ex As Long, Ey As Long, band As Channel, SourceArray() As Byte, TargetArray()
As Byte)
    Dim x As Long
    Dim y As Long
    Dim i As Long
    Dim j As Long
    Dim CenterValueRed As Long
    Dim CenterValueGreen As Long
    Dim CenterValueBlue As Long
    Dim CenterValue As Long
    Dim SumRed As Long
    Dim SumGreen As Long
    Dim SumBlue As Long
    Dim Sum As Double
    Dim mask(4, 4) As Long
    Dim tmpArray(639, 479, 2) As Byte
```

```
    mask(0, 0) = 0: mask(0, 1) = -1: mask(0, 2) = 0
    mask(1, 0) = -1: mask(1, 1) = 5: mask(1, 2) = -1
    mask(2, 0) = 0: mask(2, 1) = -1: mask(2, 2) = 0

    Sum = 0
    CenterValue = 0
    CenterValueRed = 0
    CenterValueGreen = 0
    CenterValueBlue = 0

    For x = Sx + 1 To Ex - 3
        For y = Sy + 1 To Ey - 3
            For i = 0 To 2
                For j = 0 To 2
                    If band = AllChannel Then
                        CenterValueRed = CenterValueRed + CLng(SourceArray(i + x, j + y, 0)) * mask(i, j)
                        CenterValueGreen = CenterValueGreen + CLng(SourceArray(i + x, j + y, 1)) * mask(i, j)
                        CenterValueBlue = CenterValueBlue + CLng(SourceArray(i + x, j + y, 2)) * mask(i, j)
                    ElseIf band <> DefAllChannel Then
                        CenterValue = CenterValue + CLng(SourceArray(i + x, j + y, band)) * mask(i, j)
                    End If
                Next j
            Next i
            If band = AllChannel Then
                SumRed = Abs(CenterValueRed)
                SumGreen = Abs(CenterValueGreen)
                SumBlue = Abs(CenterValueBlue)
                If SumRed > 255 Then SumRed = 255
                If SumGreen > 255 Then SumGreen = 255
                If SumBlue > 255 Then SumBlue = 255
                tmpArray(x + 1, y + 1, 0) = CByte(SumRed)
                tmpArray(x + 1, y + 1, 1) = CByte(SumGreen)
                tmpArray(x + 1, y + 1, 2) = CByte(SumBlue)
                CenterValueRed = 0
                CenterValueGreen = 0
                CenterValueBlue = 0
            ElseIf band <> DefAllChannel Then
                Sum = Abs(CenterValue) / 9
                If Sum > 255 Then Sum = 255
                tmpArray(x + 1, y + 1, band) = CByte(Sum)
                CenterValue = 0
            End If
        Next y
    Next x
    For x = Sx + 1 To Ex - 1
        For y = Sy + 1 To Ey - 1
            If band = AllChannel Then
                TargetArray(x, y, 0) = tmpArray(x, y, 0)
                TargetArray(x, y, 1) = tmpArray(x, y, 1)
                TargetArray(x, y, 2) = tmpArray(x, y, 2)
            ElseIf band <> DefAllChannel Then
                TargetArray(x, y, band) = tmpArray(x, y, band)
            End If
        Next y
    Next x
End Sub

Public Sub Thinning(Sx As Long, Sy As Long, Ex As Long, Ey As Long, band As Channel, SourceArray() As Byte, TargetArray() A
```

```
Byte)
    Dim ca As Long
    Dim cb As Long
    Dim cc As Long
    Dim cd As Long
    Dim spl As Long
    Dim npl As Long
    Dim x As Long
    Dim y As Long
    Dim flag As Boolean
    Dim hv As Long
    Dim chk As Long
    Dim cnt As Long
    Dim tot As Boolean

    Dim tmpArray1(639, 479, 2) As Byte
    Dim tmpArray2(639, 479, 2) As Byte

    For x = Sx To Ex
        For y = Sy To Ey
            For i = 0 To 2
                tmpArray1(x, y, i) = SourceArray(x, y, i)
            Next i
        Next y
    Next x

    Do
        For y = Sy + 1 To Ey - 1
            For x = Sx + 1 To Ex - 1
                tmpArray2(x, y, 0) = 0
            Next x
        Next y
        flag = False
        chk = cnt Mod 2
        cnt = cnt + 1
        For y = Sy + 1 To Ey - 2
            For x = Sx + 1 To Ex - 2
                If tmpArray1(x, y, 0) = 255 Then
                    ca = 0: cb = 0: cc = 0: cd = 0: spl = 0
                    npl = CLng(tmpArray1(x - 1, y - 1, 0)) + CLng(tmpArray1(x, y - 1, 0)) + CLng(tmpArray1(x + 1, y - 1, 0)
                        + CLng(tmpArray1(x - 1, y, 0)) + CLng(tmpArray1(x + 1, y, 0)) _
                        + CLng(tmpArray1(x - 1, y + 1, 0)) + CLng(tmpArray1(x, y + 1, 0)) + CLng(tmpArray1(x + 1, y + 1, 0)
                    If npl >= 2 * 255 And npl <= 6 * 255 Then
                        ca = 0
                    Else
                        ca = 1
                    End If
                    If tmpArray1(x, y - 1, 0) = 0 And tmpArray1(x + 1, y - 1, 0) = 255 Then spl = spl + 1
                    If tmpArray1(x + 1, y - 1, 0) = 0 And tmpArray1(x + 1, y, 0) = 255 Then spl = spl + 1
                    If tmpArray1(x + 1, y, 0) = 0 And tmpArray1(x + 1, y + 1, 0) = 255 Then spl = spl + 1
                    If tmpArray1(x + 1, y + 1, 0) = 0 And tmpArray1(x, y + 1, 0) = 255 Then spl = spl + 1
                    If tmpArray1(x, y + 1, 0) = 0 And tmpArray1(x - 1, y + 1, 0) = 255 Then spl = spl + 1
                    If tmpArray1(x - 1, y + 1, 0) = 0 And tmpArray1(x - 1, y, 0) = 255 Then spl = spl + 1
                    If tmpArray1(x - 1, y, 0) = 0 And tmpArray1(x - 1, y - 1, 0) = 255 Then spl = spl + 1
                    If tmpArray1(x - 1, y - 1, 0) = 0 And tmpArray1(x, y - 1, 0) = 255 Then spl = spl + 1
                    If spl = 1 Then
                        cb = 0
                    Else
                        cb = 1
```

```
                End If

                If chk = 0 Then
                    cc = CLng(tmpArray1(x, y - 1, 0)) * CLng(tmpArray1(x + 1, y, 0)) * CLng(tmpArray1(x, y + 1, 0))
                    cd = CLng(tmpArray1(x + 1, y, 0)) * CLng(tmpArray1(x, y + 1, 0)) * CLng(tmpArray1(x - 1, y, 0))
                Else
                    cc = CLng(tmpArray1(x, y - 1, 0)) * CLng(tmpArray1(x + 1, y, 0)) * CLng(tmpArray1(x - 1, y, 0))
                    cd = CLng(tmpArray1(x, y - 1, 0)) * CLng(tmpArray1(x, y + 1, 0)) * CLng(tmpArray1(x - 1, y, 0))
                End If
                tot = ca Or cb Or cc Or cd
                If tot = 0 Then
                    tmpArray2(x, y, 0) = 255
                    flag = True
                End If
            End If
        Next x
    Next y
    For y = Sy + 1 To Ey - 1
        For x = Sx + 1 To Ex - 1
            If tmpArray2(x, y, 0) = 255 Then
                tmpArray1(x, y, 0) = 0
                TargetArray(x, y, 0) = 0
            ' Else
            '     TmpArray1(x, y, 0) = TmpArray2(x, y, 0)
            End If
            'TargetArray(x, y, 0) = TmpArray2(x, y, 0)
            'MainScreen.ImageBufferMain.Put TargetArray, imRGB24Planar, imAllBands, 0, 0, 640, 480
        Next x
    Next y
    MainScreen.ImageBufferMain.Put TargetArray, imRGB24Planar, imAllBands, 0, 0, 640, 480
    DoEvents
Loop While flag
For y = Sy + 1 To Ey - 1
    For x = Sx + 1 To Ex - 1
        hv = 0
        If tmpArray1(x, y, 0) = 255 Then
            If tmpArray1(x, y - 1, 0) = 255 And tmpArray1(x + 1, y, 0) = 255 Then hv = hv + 1
            If tmpArray1(x + 1, y, 0) = 255 And tmpArray1(x, y + 1, 0) = 255 Then hv = hv + 1
            If tmpArray1(x, y + 1, 0) = 255 And tmpArray1(x - 1, y, 0) = 255 Then hv = hv + 1
            If tmpArray1(x - 1, y, 0) = 255 And tmpArray1(x, y - 1, 0) = 255 Then hv = hv + 1
            If hv = 1 Then tmpArray1(x, y, 0) = 0
        End If
    Next x
Next y
For y = Sy + 1 To Ey - 1
    For x = Sx + 1 To Ex - 1
        TargetArray(x, y, 0) = tmpArray1(x, y, 0)
    Next x
Next y
End Sub

Public Sub Negative(Sx As Long, Sy As Long, Ex As Long, Ey As Long, band As Channel, SourceArray() As Byte, TargetArray() A
Byte)
    Dim x As Long
    Dim y As Long
    For y = Sy + 1 To Ey - 1
        For x = Sx + 1 To Ex - 1
            If band = AllChannel Then
                TargetArray(x, y, 0) = 255 - SourceArray(x, y, 0)
```

```
                TargetArray(x, y, 1) = 255 - SourceArray(x, y, 1)
                TargetArray(x, y, 2) = 255 - SourceArray(x, y, 2)
            ElseIf band <> DefAllChannel Then
                TargetArray(x, y, band) = 255 - SourceArray(x, y, band)
            End If
        Next x
    Next y
End Sub

Public Function FindProcessArea(Sx As Long, Sy As Long, Ex As Long, Ey As Long, band As Channel, cutvalue As cutvalue
SourceArray() As Byte, TargetArray() As Byte) As AreaData
    Dim x As Long
    Dim y As Long
    Dim indx As Boolean
    indx = False
    For y = Sy + 1 To Ey - 1
        For x = Sx + 1 To Ex - 1
            If band = AllChannel Then
                If SourceArray(x, y, 0) > RedCutValue Or _
                   SourceArray(x, y, 1) > GreenCutValue Or _
                   SourceArray(x, y, 2) > BlueCutValue Then
                    FindProcessArea.Sy = y
                    indx = True
                End If
            ElseIf band <> DefAllChannel Then
                If SourceArray(x, y, band) > AllCutValue Then
                    FindProcessArea.Sy = y
                    indx = True
                    Exit For
                End If
            End If
        Next x
        If indx = True Then
            Exit For
        End If
    Next y
    indx = False
    For y = Ey - 1 To Sy + 1 Step -1
        For x = Sx + 1 To Ex - 1
            If band = AllChannel Then
                If SourceArray(x, y, 0) > RedCutValue Or _
                   SourceArray(x, y, 1) > GreenCutValue Or _
                   SourceArray(x, y, 2) > BlueCutValue Then
                    FindProcessArea.Ey = y + 1
                    indx = True
                End If
            ElseIf band <> DefAllChannel Then
                If SourceArray(x, y, band) > AllCutValue Then
                    FindProcessArea.Ey = y + 1
                    indx = True
                    Exit For
                End If
            End If
        Next x
        If indx = True Then
            Exit For
        End If
    Next y
    indx = False
```

```
    For x = Sx + 1 To Ex - 1
        For y = Sy + 1 To Ey - 1
            If band = AllChannel Then
                If SourceArray(x, y, 0) > RedCutValue Or _
                   SourceArray(x, y, 1) > GreenCutValue Or _
                   SourceArray(x, y, 2) > BlueCutValue Then
                    FindProcessArea.Sx = x
                    indx = True
                End If
            ElseIf band <> DefAllChannel Then
                If SourceArray(x, y, band) > AllCutValue Then
                    FindProcessArea.Sx = x
                    indx = True
                    Exit For
                End If
            End If
        Next y
        If indx = True Then
            Exit For
        End If
    Next x
    indx = False
    For x = Ex - 1 To Sx + 1 Step -1
        For y = Sy + 1 To Ey - 1
            If band = AllChannel Then
                If SourceArray(x, y, 0) > RedCutValue Or _
                   SourceArray(x, y, 1) > GreenCutValue Or _
                   SourceArray(x, y, 2) > BlueCutValue Then
                    FindProcessArea.Ex = x + 1
                    indx = True
                End If
            ElseIf band <> DefAllChannel Then
                If SourceArray(x, y, band) > AllCutValue Then
                    FindProcessArea.Ex = x + 1
                    indx = True
                    Exit For
                End If
            End If
        Next y
        If indx = True Then
            Exit For
        End If
    Next x
End Function

Public Function FindCenter(pArray() As PointData) As PointData
    Dim i As Long
    Dim SumX As Double
    Dim SumY As Double
    Dim Pcount As Double
    For i = 0 To 10000
        If pArray(i).x <> 0 And pArray(i).y <> 0 Then
            SumX = SumX + pArray(i).x
            SumY = SumY + pArray(i).y
            Pcount = Pcount + 1
        End If
    Next i
    FindCenter.x = CLng(SumX / Pcount)
    FindCenter.y = CLng(SumY / Pcount)
```

End Function

3. 전체 조립도 설계도면