센서 통합 능력을 갖는 다중 로봇 Controller의 설계 기술

서일홍, 여희주, 엄광식

한양대학교 전자공학과 지능제어 및 로보틱스 연구실

1. 서 론

최근 생산성을 향상을 통한 국제 경쟁력을 강화를 위해 산업용 로봇을 응용한 다중 로봇 제어 시스템[1-3]의 개발에 점점 더 관심이 집중되고 있다. 이러한 시스템의 잠재적인 응용이 보다 정교한 조립 작업의 자동화를 포함하여 전 공장의 무인화까지를 목표로 하고 있으며 나가아서는 심해 작업이나 우주 공간에서 작업의 로봇화를 요구하고 있는 실정을 감안할 때 이들 요구에 부응하기 위해서 가장 핵심적인 연구인 복수개 로봇 및 다중센서를 제어할 수 있는 다중 로봇 시스템 제어장치의 개발은 반드시 필요하게 될 것이다. 더군다나 움직이는 장애물과 같은 환경의 변화를 고려할 때, 센서 정보에 의해 환경을 묘사하는 방법[4-6]이 필요하다. 시각, 거리, 근접, 접촉, 및 힘 센서 등과 같은 유용한 센서들에 의해서 다양한 정보를 얻을 수 있다. 따라서, 다중 로봇 제어 시스템을 운영하는 데 있어서 운동계획, 협조제어 능력과 함께 특히 필요한 기능은 센서 정보의 융합과 통합으로 생각할 수 있다. 이 문제를 해결하기 위해서 제어 시스템은 Multi-Robot이 협조제어를 할 수 있도록 해야하고 다중센서를 융합할 수 있어야 한다. 그러나 ,현재 상품화된 거의 모든 로봇 제이 시스템은 기본적으로 사용자가 한 대의 로봇만을 사용할 수 있게 되어 있고, 또한 적절한 제어 알고리즘과 계산 방법의 미흡으로 인해 효과적으로 다중센서를 융합할 수 없다. 따라서, 이러한 다중로봇 및 다중센서를 처리할 수 있는 로봇 시스템 제어기의 개발은 현재까지 연구되어온 한개의 로봇 제어에 관한 연구결과를 그대로 확장, 응용할 수 있는 단계를 넘어서서 새로운 방법을 개발하여야 한다.

따라서, 본 연구에서는 다중 프로세서에 근거해서 다중센서 통합 능력을 가지는 다중 로봇 협조제어 시스템을 구현하는 방법을 설명하고자 한다. 특히, 개발한 전체 제어 시스템은 12축을 동시에 제어할 수 있도록 하였으며, 또한 로봇이 변화하는 외부환경에 능동적으로 대처하며 좀더 다양한 종류의 작업에 응용할 수 있도록 Sensor-Based Robot Motion Control을 강화하였다.

이를 위하여 본 연구에서는 신뢰성과 확장성이 뛰어난 CPU30 (32Bit CPU 보드), VME BUS와 실시간 운영체계 (Real-Time O.S)인 VxWorks를 토대로 하여 동시동작 및 분산처리 구조를 갖도록 하였으며, 이러한 시스템에 Ethernet을 통해 SUN Workstation과 연결하여 확장성 및 유연성이 높은 전체 시스템을 구성하여 실험하였다.

2. 다중 로봇 제어 시스템의 구성

2.1 전체 시스템의 구성

본 연구에서는 다음과 같은 여러 개의 프로세서로 나누어 분산처리 구조를 갖도록 하여 확장성 및 유연성이 높은 시스템이 되도록 구성하였다.

먼저 시스템 전체를 관리하며 언어 및 로봇 동작의 교시 그리고 자기 진단 등의 기능을 하는 Supervisory Processor 물체의 위치와 자세 및 형태를 인식하는 Vision Processor, Man-Machine Interface를 위한 Display Processor, 그리고 로봇의 제어를 담당하는 로봇 제어 Processor 등으로 구성되어 있고, 이들 각각의 통신은 Bus Arbitration에 의한 VME Global Bus를 통해 공유 메모리(Common Memory)를 이용하여 수행한다.

특히 Supervisory System은 Real Time O.S인 VxWorks를 사용하여 구현하였고, Servo 시스템은 Inter-

rupt방식을 사용하여 구현하였다.

2.2 제어 시스템의 H/W 구성

본 연구에서는 여러개의 프로세서로 나누어 분산처리 구조를 갖도록 하여 확장성 및 유연성이 높은 시스템을 구성하였다. 이에 대한 Hardware 구성은 그림 1과 같다.

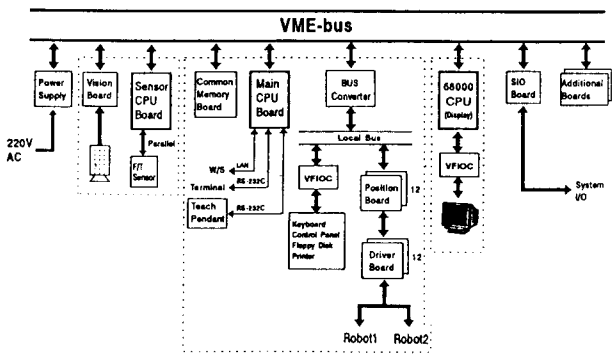


그림 1. 다중로봇 협조제어 시스템의 H/W 구조.

2.3 제어 시스템의 S/W 구성

개발된 Software는 C언어에 기반을 두고 모든 프로그램을 세분화하여 계층적 제어구조를 이루게 하고 수정 및 편집을 단위 프로그램 모듈별로 할 수 있게 하는 새로운 형태의 매니퓰레이터 레벨 로봇 제어언어를 설계하였다. 이에 대한 제어 시스템의 S/W 구조는 그림2와 같다.

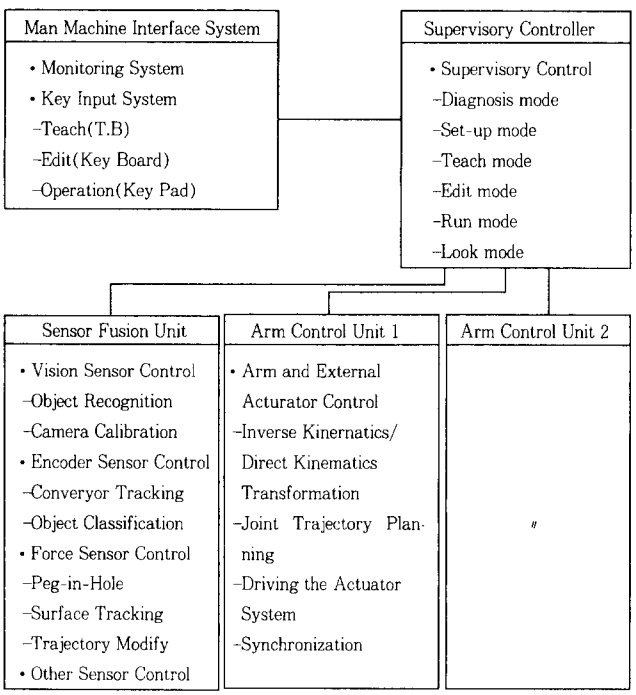


그림 2. 다중 로봇 협조제어 시스템의 S/W 구조.

3. 다중 로봇 제어 시스템의 기능별 구성

3.1 Supervisory System

Supervisory 시스템을 설계하는데 있어서, Supervisory는 각 Slave에 작업을 효과적으로 할당하여 수행시켜야 하고, 또한 조작자에게는 로봇 언어와 같은 사용자에게 친숙한 Man-Machine 인터페이스를 준비하여 시스템 Set-Up뿐만 아니라 운영상의 실수를 줄일 수 있도록 고려되어야 한다. 또한, 전체 시스템의 원활한 유지를 위해 자기진단 기능이 필요하다. 이들 요건을 충족시키기 위해서, 본 시스템에서는 다음과 같이 6가지 모드를 설정하였다.

(1) Diagnosis Mode, (2) Set-Up Mode,
(3) Teach Mode, (4) Edit Mode, (5) Look Mode,
(6) Run Mode

(1) Diagnosis Mode

진단 모드에서는 우선 Supervisory System을 메인 하드웨어 시스템(CPU Board, Common Memory Board, Motor Servo Modules) 주변 하드웨어 시스템(Floppy Disk Driver, Printer, Teaching Box)등의 상태를 확인하고, 그것들의 현 상태를 표시한다.

(2) Set-Up Mode

Set-Up 모드에서는 응용시에 제어 시스템의 융통성을 부여하기 위해서 여러가지 파라미터를 Set-Up하거나 수정할 수 있다. 본 시스템에서는 각 로봇 파라미터(로봇 타입, 기구학 변수, Motor 정격), 원점 좌표 파라미터(홈 위치, 속도, 오프셋)와 시스템 파라미터(Base 좌표계, Tool 좌표계, 위치레벨, 전용 I/O, 외부장치)를 설정한다. 그래서 이들 입력 파라미터를 조정하여 어떤 타입의 로봇(X-Y-Z 직교형, 수평 다관절형, 수직 다관절형)도 제어할 수 있다.

(3) Teach Mode

로봇의 엔드-이펙트를 정의된 여러 좌표계를 기준으로 이동시켜, 로봇의 "위치"와 "방향"을 메모리에 기억시키고 각 점의 이름을 할당하여 운동 계획에 사용한다. 본 연구에서는 여러 좌표계(Robot-Base Coord. User-Defined Coord. Tool Coord)를 준비해서 Teaching을 용이하게 하는 한편, CAD 데이타를 직접 입력하는 MDI(Manual Data Input)와 모터의 전원을 끈 채로 로봇을 직접 이동시켜 Teaching하는 Free Teach 모드가 있다.

Teaching Box는 두 대의 로봇 및 외부 축을 조작하고, 작업에 대한 궤적을 지시, 수정하며 실제로 로봇의 작업을 편집할 때 로봇 언어중 MOVE 관련 명령어(MOVJ, MOVL, MOVA, MOVC...)들을 편집, 등록, 수정할 수 있다. 아울러, 로봇의 상태 표시 기능으로 I/O Device의 상태

를 표시 제어할 수 있다.

(4) Edit Mode

본 시스템에서 사용하는 로봇 언어가 High Level Language에 가까우므로 본 에디터는 PC 의 풀키(Full Key)를 사용하여 프로그램을 작성할 수 있고, 또한 많이 쓰이는 명령어의 편리를 위하여 로봇 언어 MACRO 기능을 갖추고 있다. 본 에디터는 Full Screen Editor로 Doubly Linked List Structure로 편집중인 라인을 가르키는 포인트는 Text를 위한 버퍼와 시작과 끝을 가르키는 포인트를 가지고 있다. 그리고, 본 에디터가 갖고 있는 주요 기능으로 Search & Replace, Block Copy, Move ,Delete, Block Print, File Print, File Read/Write 등이 있다.

(5) Run Mode

Supervisory System은 프로그램된대로 자기 자신의 작업을 수행할 뿐만 아니라, 상태를 모니터링하면서 각 서보에 필요한 작업을 적절히 할당한다. 프로그램의 정확성을 확인하기 위해서 세가지 형태로 프로그램을 수행할 수 있도록 하였다. 즉 단계별 동작 확인을 위한 단계별 수행(Step Run) 처음 단계에서 마지막 단계까지 동작 확인을 위한 사이클 수행(Cycle Run) 과 동작 확인 후 반복 동작을 위한 자동 수행(Auto Run)이다. 충돌 회피와 두 대 팔의 협조 작업을 효과적으로 조작하기 위해 Supervisory는 궤적을 계획하고 계산하여 이를 Common Memory를 통해 각 Servo System과 통신한다. 또, 연속 궤적 제어의 부드로운 동작을 보장하기 위해 각 관절동작이 계획되어질 때 자동적인 가감속이 고려되어야 한다. 이를 위해 2차 이산 Shaping Filter가 직각 공간에서 궤적을 보상하기 위해 적용된 후 관절 궤적으로 변환된다.

(6) Look Mode

Look Mode에서는 크게 선처리와 후처리로 나눌 수 있다. 선처리는 카메라 선택(4대중 한대를 선택) Image Grab, Snap, Threshold를 이용한 Binary Processing등의 기능이 있고, 후처리에서는 윈도우처리와 물체 학습, 저장 기능이 있다. 물체 학습은 각 물체의 특성에 따라 Chain Code, Projection, Matching등 여러가지 방법을 선택할 수 있다. 학습된 물체의 특징량은 그 이름과 함께 Common Memory에 저장된다. 이 데이타는 Supervisory CPU가 물체를 인식하여 동작할 때 쓰인다. 이들 기능 모두를 효과적으로 수행하기 위해 Supervisory 컨트롤러는 Real Time O.S 인 VxWorks를 사용하여 구현하였다.

3.2 Vision 시스템

비젼 시스템은 VME 사양에 맞게 설계된 SVS900-DT 보드를 사용하였는데 이는 해상도가 512×512이고 명암도가 256단계이다. 본 비젼시스템은 물체인식, 위치검출의 역할을 담당하고 있으며, 또한 최대 4대의 카메라를 사용할 수 있도록 함으로써 필요 부분에 설치하여 효율을 높일 수 있도록 하였다. 이를 위한 제어 명령어가 표 1에 정의되어 있다.

3.3 로봇 제어 시스템(Arm Control System)

이 시스템은 Supervisory 시스템에서 받은 로봇 명령에 대해 Inverse Kinematics을 풀고 이를 관절 운동으로 변환하여 Actuator 시스템을 구동시켜 실제로 로봇 Arm을 제어한다. 제어장치는 Position Board와 Power Amp로 구성되어 있으며 최대 12 축까지 제어할 수 있다. 이 장치는 매 Solution 시간마다 다음의 4가지 일을 수행한다.

1) 원하는 궤적 정보를 얻고 로봇의 현 상태를 알기 위해 Supervisory통신
2) 관절 궤적을 구하기 위해서 Inverse Kinematics 변환과 현재 위치 표시를 위해 Direct Kinematics 변환
3) 관절 궤적 계획(Pulse Generation and Exponential Filtering)
4) 매 5msec당 펄스 명령을 분배

4. 주변기기 및 센서 인터페이스 시스템의 기능별 구성

4.1 Man-Machine Interface System(VFIOC)

VFIOC(Video/Floppy/IO Card)는 VME 사양에 맞게 디자인된 Man-Machine Interface Board이다. 이는 PC BUS Converter Interface, Floppy Interface, Parallel Interface, Keyboard Interface, Keypad Interface의 모듈을 내장하고 있으며 이에 대한 블럭도는 그림 3과 같다.

(1) PC Bus Converter Interface

PC Bus Converter Interface는 Motorola I/O Channel Bus의 Signal을 IBM PC/XT Bus의 Signal로 전환시키는데 이때 Address Expansion Register를 사용하여 Address Windows를 확장시켜 준다.

(2) Floppy Interface

Floppy Interface는 3.5″ Floppy Drive와 IBM PC 데이타 포멧을 사용한다. 이의 구현은 VxWorks Kernel의 Read, Write 및 Format 기능을 이용하였다.

(3) Keyboard Interface

Keyboard Interface는 IBM PC/AT Keyboard로 부터 데이타를 받아 Host로 보내는데 이때 VxWorks Kernel을

이용하여 Read/Write 및 Status Report 기능을 제공한다.

(4) Keypad Interface

Keyboard Interface는 16개의 Push Button을 인식하여 간이 Keyboard 기능을 제공하는데, 이때 여러개의 Key가 눌려지는 경우, 최초의 Key 데이타를 Latch한다.

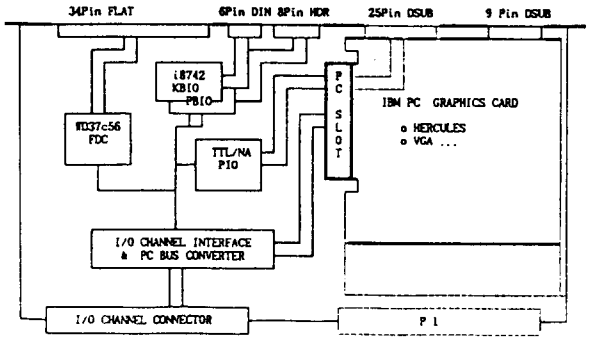


그림 3. VFIOC Board의 전체 구성.

4.2 다중 로봇용 Multi-Functional Teach Box

본 연구에서 개발한 Teach Box는 두대의 로봇 및 외부축을 조작하고, 작업에 대한 궤적을 지시. 수정하며 실제로 로봇의 작업을 편집할때 로봇 언어 중 70%~80%를 사용하게 되는 MOVE관련 명령어(MOVJ, MOVL, MOVA, MOVC,…)들을 편집, 등록, 수정할 수 있다. 아울러 로봇의 상태표시 기능으로 I/O Device의 상태를 표시, 제어할 수 있으며 또 로봇의 위치 및 자세를 사용자가 선택한 좌표계를 기준으로 나타낼 수 있는 등 기존의 단순한 Teaching Box 기능보다 크게 향상시켰으며, Teaching Box의 디스플레이를 위해 LCD를 이용하여 Man-Machine 인터페이스 부분을 향상시켜 사용자로 하여금 보다 편리하게 명령어의 편집, 등록 및 수정 등을 할 수 있게 하였다.

본 연구에서는 개발한 Teaching Box의 기능을 다음과 같이 크게 5가지로 대별된다.

1) 교시 기능 및 축 조작 기능
2) 궤적의 확인, 실행 가능
3) 명령의 등록 및 편집, 수정 가능
4) 로봇의 상태 표시기능
5) 긴급사태시 조작 및 안전기능

4.3 다중센서 인터페이스 시스템

로봇의 작업 내용이 복잡해지고, Motion 중심이 아닌 Task 중심의 프로그램을 하게 되면 로봇은 항상 외부 변화를 감시하고 이에 대응하는 동작을 수행해야 한다. 따라서 환경 변화를 감시하기 위해 센서가 필요하고 센서 신호에 의존하여 Trajectory Planning을 하는 Motion이나 Motion을 수정하는 기능이 필요하다.

본 시스템에서는 센서처리 보드를 준비하여 16개의 Digital Input Signal을 Check하는 Din(Digital Input), Digital Signal을 출력하는 Dout(Digital Output), Conveyor Tracking을 위해 Conveyor Encoder 신호, Vision 신호등을 인식한다. Digital Signal 입력 장치로는 MACRO6744 Board, 출력장치로는 MACRO6745 Board를 이용하였다.

4.4 Conveyor Tracking

Conveyor 동기운전 기능이란 Conveyor 이동량을 고려한 궤적 계획에 의해서 Conveyor가 정지한 상태로 교시한 궤적을 동작 중인 Conveyor위로 재현하는 기능이며, 궤적 추종을 위한 Conveyor의 이동량의 검출은 엔코드의 Feedback Pulse 누적치로 계산된다. 보정기능의 하나로 속도 변동에 따른 추종오차를 줄이기 위하여 평균 속도를 일정 주기마다 (Ts=40msec) 샘플링하여 Extrapolation을 통해 다음 주기에 가야 할 양을 미리 계산하여 추종한다. 또한 동기동작의 개시 순간 추종 지연에 따른 오차를 줄이기 위하여, 동기 개시시 추종지연시간 후의 오차량을 계산해 Conveyor 방향으로 추종 오차량을 보정한다. 전체 Conveyor 시스템 구성은 그림 4와 같다. 사용된 Sensor는 위와 같이 평균속도 측정용 동기 개시용 Limit Sensor가 있어 Conveyor상의 Object가 평균속도 측정 리미트 센서에 닿았을 경우 기준이 되는 Conveyor의 속도를 구하고, 동기개시 리미트 센서에 닿았을 때 동기 동작이 시작된다.

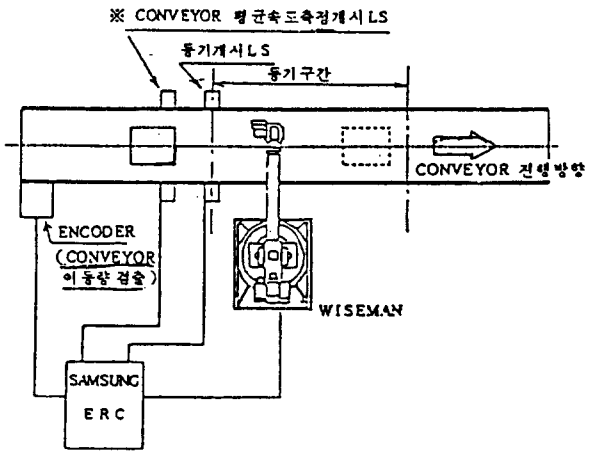


그림 4. Conveyor 동기 운전의 시스템 구성.

5. 다중 로봇 시스템을 위한 로봇 제어 언어

5.1 기본 동작 명령어(Basic Motion Commands)

본 연구에서 개발한 로봇 제어 언어는 한 대의 로봇을 위한 상용화된 로봇 언어를 포함하여 전체 87가지의 명령어을 제공하고 있다(표 1). 우선, 관절운동(PTP)을 위한

표 1. 다중로봇 제어 언어 일람표.

Group	Instruction	Structure(Syntax)
Basic Motion	MOVJ MOVL MOVC MOVA MOVI MOV_EXT	movj/movl ⟨robot⟩ to ⟨locl⟩ with sp=⟨speed⟩ pl=⟨position level⟩ movc/mova ⟨robot⟩ via ⟨loc1⟩ ⟨loc2⟩ with sp=⟨speed⟩ pl=⟨position level⟩ movi ⟨robot⟩ by ⟨offset⟩ with sp=⟨speed⟩ mov_ext ⟨device⟩ to ⟨loc1⟩ with sp=⟨speed⟩
Master & Slave Motion	MS_MOVL MS_MOVI MS_MOVC MS_MOVA	ms_movl ⟨robot⟩ to ⟨loc1⟩ with sp=⟨speed⟩ ms_movi ⟨robot⟩ to ⟨offset⟩ with sp=⟨speed⟩ ms_movc/ms_movc ⟨robot⟩ via ⟨loc1⟩ ⟨loc2⟩ with sp=⟨speed⟩
Collision Avoidance	COLL_AVOID	do coll_avoid
Conditional Instructions	ON condition DO action	on ⟨statement⟩ do ⟨statement⟩
Conveyor Tracking	CV_SPCHK CV_SYNC MOVS CVEND	cvspchk cv #⟨num⟩ cvsync ⟨robot⟩ with uf #⟨num⟩ t=⟨num⟩ movs ⟨robot⟩ to ⟨loc1⟩ with sp=⟨speed⟩ cvend
Concurrent Motion	COBEGIN COEND	cobegin motion instruction 1 motion instruction 2 coend
Special Motion	SET_PALLET SET_SEG SFTON SFTOFF	set_pallet ⟨robot⟩ for ⟨loc1⟩ ⟨loc2⟩ set_seg x=⟨num⟩ y=⟨num⟩ z=⟨num⟩ sfton ⟨robot⟩ sftoff ⟨robot⟩
Coordinate Translation	TRANS_COORD REF_COORD RECOVER_COORD AFFIX UNFIX MAKEPOSITION	trans_coord ⟨loc1⟩ for ⟨coord⟩ ref_coord ⟨coord⟩ recover_coord ⟨robot⟩ affix ⟨object⟩ to ⟨robot⟩ unfix ⟨robot⟩ for ⟨robot⟩ makepostion(TRANS, TOOL1...LOC)
Gripper Control	GRASP/RELEASE GRIP_UP GRIP_DOWN GRIP_ROTATE TOOL	grasp/release ⟨robot⟩ until force=⟨num⟩ with width=⟨width⟩ grip_up/grip_down ⟨robot⟩ with sp=⟨speed⟩ disp=⟨displacement⟩ grip_rotate ⟨robot⟩ with sp=⟨speed⟩ disp=⟨degree⟩ tool⟨tool#⟩
Boolean Logical Operators	AND/OR NOT/XOR	⟨variable⟩ and ⟨variable⟩

Group	Instruction	Structure(Syntax)
Relational Operator	〈,〈=,=,〉,〉= ==,!=	〈variable〉==〈variable〉
Algebric Operators	+,−,*,/	〈variable〉+〈variable〉

Control Instructions	IF, THEN
	ELSE, ENDIF
	FOR, STEP
	NEXT
	WHILE, WEND
	FCALL, FRET
	JCALL
	NOP
	PAUSE
		if (expression) then [else]... endif
		
		for 〈i var〉=〈num〉 to 〈num〉 step=〈num〉...next i〈two_num〉
		while (experssion)...wend
		fcall 〈function name〉 [with 〈num〉]... fret
		jcall 〈job filename〉
		
		pause if in#10=1

Vision	SV_INIT SV_GRAB SV CLS SV_THSET SV SNAP SV RECOG USE CAMERA SET SIZE SET GRAY PRINT	 sv_thset with 〈num〉 use camera 〈num〉 set size 〈num〉 set gray 〈num〉 print 〈string〉 on 〈x position〉, 〈y position〉
Input/Output Control	PULSE DIN/DOUT IN/OUT WAIT	pulse #〈port〉, 〈dutation〉 din/dout #〈bit〉 〈state〉 in/out #〈byte〉 〈state〉 wait until time 〈relop〉 〈num〉 wait until din #〈port〉 〈state〉
Sensor Related Instruction	TIM, DIST FORCE LIMIT	 limit #〈number〉
ETC	DEFINE END SPEED	define 〈robot〉 master/slave_ speed=〈speed〉

MOVJ. 연속 궤도 운동(CP)을 위한 MOVL. 주어진 세점을 통과하는 원호 운동을 위한 MOVC. 현재 위치에서 주어진 오프셋 거리만큼 증분치 운동을 위한 MOVI와 같은 기본 운동 명령어를 제공한다. 그러나, 이들 기본 명령어만을 이용하여 두대의 로봇이 함께 일하는 것이 실제로는 어려우므로 본 시스템에서는 Master & Slave 운동 명령어를 준비하였다. 한 대 로봇을 Master로 다른 한 대를 Slave로 정의하여, Slave 운동이 Master의 운동을 추종하도록 한다. 특히 MOVJ을 제외한 모든 기본 운동 명령어는 Table1에서 보는 바와 같이 Master/Slave운동으로 사용할 수 있다. Master/Slave 운동은 두 로봇간의 Kinematic Relation을 유지하도록 Slave가 추종하는 방법과 Slave Robot에 장착한 F/T Sensor의 Force Control을 하는 방법으로 추종할 수 있다.

본 시스템은 Positioner 혹은 Conveyor 시스템과 같은 외부 장치를 포함하여 동시에 12축까지 제어할 수 있기 때문

에 두대 로봇 사이의 충돌 회피와 외부 장치와의 동기를 고려하여야 한다. 이를 위해 조건 운동 명령어를 ON condition DO action의 문법으로 제공하였다. 그러나 조건적 명령어(Conditional Motion)는 위에 기술한 모든 운동 명령어에 사용할 수 있다. 예를 들어 두 대의 로봇 사이의 가능한 충돌 회피를 하기 위해, 거리 조건과 함께 COLL-AVOID 명령어를 다음과 같이 사용할 수 있다.

```
cobegin
  movl robot1 to loc1 with sp = 60;
  movl robot2 to loc2 ON dist < 10 DO coll_
  avoid;
coend
```

특히, 일반적인 3-차원 충돌회피[12,13]는 실시간에 구현하기가 어렵기 때문에 본 시스템에서는 단지 두대의 수평 다관절형 로봇에만 적용되는 충돌회피 기능을 구현하였다. 사용된 알고리즘은 다음과 같다. 만약 두대 로봇 사이의 거리가 주어진 조건보다 작으면(충돌 가능성이 있으면) i) 두대 로봇이 다른 방향으로 움직이는 경우에 Master 로봇은 목표를 향해 움직이고 Slave 로봇은 충돌이 회피될 때까지 안전한 지점까지 회피하고, ii) 두대의 로봇이 같은 방향으로 움직이는 경우에는 우선 Master 로봇이 움직이고, 뒤 따라오는 로봇은 잠시 동안 기다린다(Wait)

이러한 동작 명령어는 16점 디지탈 입력 신호를 확인하는 DIN조건, 동작 시간을 확인하는 TIME 조건, 입력 센서신호에 근거로 시스템이 궤적을 다시 계획하는 MOVS동작, 시스템을 멈추는 STOP동작, 16점 디지탈 신호를 출력하는 DOUT과 같이 이용할 수 있다.

전술한 동작 명령어와 조건 문장을 구현하기 위해서 Real Time O.S VxWorks을 채택했다. 특히, 모든 동작 명령어는 프로세스로써 정의한다. COBEGIN과 COEND 문장인 경우 모든 프로세스는 COBEGIN과 COEND사이에서 Create 되고 Spawn 된다. Conditional Motion인 ON conditional DO action 문장의 경우에서는 Motion을 수행하다가 Condition을 만족하면 Motion을 정지시키고 Action을 수행한다.

전술한 동작 명령어 외에 Teach Point근처를 연속적으로 경유하는 Position Level과 좌표계를 변환하여 작업하게 하는 Trans-Coord와 기준 좌표계를 새로 설정하는 Ref-Coord등을 이용한 Job Shift 기능, Palletizing 기능 등이 있다. 이외에도 Tool 제어, 논리 연산, Loop 제어, 산술 연산 명령어 등이 구현되어 있다.(표 1 참조)

5.2 다중로봇 제어를 위한 Task 단위의 Concurrent Motion

로봇이 수행해야 할 작업이 복잡해지고 다양해짐에 따라 여러대의 로봇 또는 한대의 로봇과 외부 디바이스가 협조작업을 하는 것이 필요하다. 이를 위하여, 로봇 제어기는 독립적인 제어 및 동기동작(Synchronous Motion)이 가능해야 한다. 그러나 기존 산업용 로봇은 한대의 Controller가 한대의 로봇을 제어하기 때문에 제어기간의 통신을 이용해야 하는 어려움이 있으며, 최근 몇몇 제어기에서만 다중 로봇의 제어가 가능하다.

Task 단위의 동시 동작을 하기 위해서 각 로봇이 수행할 일련의 작업을 각각의 Task로 정의하고 이렇게 정의된 Task의 작업을 순차적으로 실행시켜 주는 독립된 프로세서를 생성하게 된다. 따라서 Cobegin~Coend 내에서 여러 개의 Task를 동작시킬 경우 생성된 각 프로세서를 Multi-Tasking O.S인 VxWorks[10]를 이용하여 동시 실행시킴으로써 Task 단위의 Concurrent Motion을 구현할 수 있었다. 또한 Task간의 동기제어 및 협조 작업을 위한 명령어인 "sync#"는 VxWorks O.S에서 제공되는 Semaphore 기능을 이용하여 구현하였으며, 각 Task의 같은 번호의 Sync 명령간의 동기가 맞추어지도록 정의하였다. Task 단위의 동시 동작의 문법 체계는 다음과 같다.

```
cobegin
  call task1;
  call task2;
coend
function task1
{
      movj robot1 to loc1 with sp = 10;
      movj robot1 to loc2 with sp = 10;
      sync1;
      grasp robot1;
      movj robot1 to loc3 with sp = 10;
      movj robot1 to loc3;
      sync2;
      release robot1;
}
function task2
{
      movj robot2 to loc1 with sp = 10;
      sync1;
      grasp robot2;
      movl robot2 to loc5;
      sync2;
      release robot2;
}
```

5.3 센서 기반형 로봇 동작 명령어(Sensor-Based Robot Motion Commands)

일본 YASKAWA사의 ERC와 스웨덴 ABR사의 IRB2000 산업용 로봇 제어언어는 한대의 로봇을 위한 기본 명령어만을 제공하고 있으므로 User가 로봇의 기본동작 제어 명령만으로 작업을 일일이 프로그램해야 한다는 어려움이 있을 뿐만 아니라, 응용할 수 있는 작업이 제약되어 있다. 따라서, 보다 강력한 로봇 언어를 개발하기 위해서는, 기존 로봇 언어에서 로봇이 변화하는 외부 환경에 능동적으로 대처하며 좀더 다양한 종류의 작업에 응용할 수 있도록 Sensor-Based Robot Motion Control 기능의 강화가 필수적이다. 따라서 본 연구에서는 F/T 센서 및 Vision 센서를 이용하는 다음과 같은 Sensor-Based Robot Control 기능을 개발하였다.

(1) On-Line Path Modification.

기존의 산업용 로봇에 사용되는 센서로는 로봇의 위치나 속도제어를 위한 레졸버, 엔코더, 타코메터와 홈 위치 및 리미트 위치 감지를 위한 센서등이 대부분 이었다. 그러나, 정밀 조립작업이나 용접작업과 같은 작업을 수행해야 할 경우 정밀한 Motion과 경로보정을 위해서는 Force(Tactile) 센서, Vision 센서등이 필요하다. 예를 들면, Peg-In-Hole과 같은 조립 작업의 경우 Teach한 위치와의 오차가 생기게 되면 센서를 이용하지 않는 로봇의 경우는 에러를 발생하게 되지만, 센서를 사용한 로봇의 경우 이러한 오차에 대해 능동적으로 대처할 수 있다. 목표 위치가 고정되어 있을 경우, 주어진 경로를 따르면서 매 샘플링 타임마다 외부센서 데이터에 의해 로봇 매니퓰레이터의 경로를 보정하기 위해 기존의 movl 명령어에 위치 및 자세의 보정량 포함하도록 확장하였다. 이때 보정량인 Correction은 새로운 옵션에 의해 내부적으로 정의하였다. Movl에 대한 문법체계는 다음과 같고 Path Modification 구조 및 센서 데이타의 전달구조는 각각 그림 5, 그림 6과 같다.

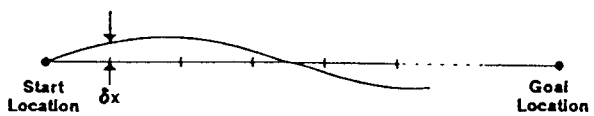


그림 5. On-Line Path Modification의 구조.

define inport10 path_mod with file=filename;
movl robot1 to loc1 with correction=path_mod, sp=50;
filename : reference file for sensory motion

〈Example〉 Force Control

① Desired Force : F_d
② Environmental Stiffness : K_e
③ Selection Matrix : S

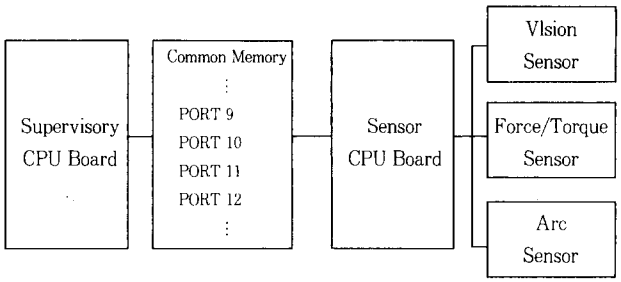


그림 6. Common Memory Port를 이용한 센서 데이타의 전달구조.

그와 같은 Movl-with-correction 명령어는 용접 작업이나 그라인딩 작업에서 경로를 보정하는 작업 유용하게 적용될 수 있다.

(2) Tracking of Moving Target

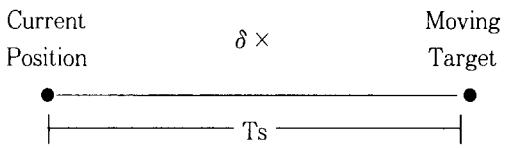


그림 7. Tracking of Moving Target의 구조.

몇 개의 주어진 조건이 만족할 때까지 외부 센서 데이타에 의해 그림 7에서와 같이 Moving Target을 추적하기 위해 move 명령어를 구현하였다. 여기에서 로봇이나 외부 디바이스가 움직이는 위치는 Position Equation에 의해 정의된다. Move 명령어의 문법체계는 다음과 같다.

move 〈robot〉 [with 〈position eq.〉] [until 〈condition〉]
〈position eq.〉=[〈position eq.〉 +] 〈4×4 H.T.M〉
〈Example〉
move robot1 with slave_pos until time<30;
slave_pos=master_pos+base1−diff1;

여기에서 base1과 diff1은 Teaching Pendant에 의해 저장된 4×4 동차변환행렬이다. 그리고 master_pos와 slave_pos는 각각 Master 로봇 혹은 Slave 로봇의 현재위치이다. 또한 '+','−'은 각각 매트릭스의 Direct Multiplication과 Inverse Matrix의 Multiplication을 쉽게 표현하기 위하여 사용된다. 이때 사용되는 Condition은 다음과 같다.

〈Example〉

Condition

Time>δ, FORCE>δ
DIST>δ, LIMIT#1=ON
Boolean Expression

이와 같은 move 명령어는 Master & Slave Motion이나 Visual Servoing, Conveyor Tracking 작업 등에 응용될 수 있다. 따라서 이렇게 제안·정의된 Sensor-Based Robot Language와 기존의 로봇 언어를 함께 이용한다면 표현할 수 있는 작업의 종류와 기능이 더욱 많아져 보다 다양하고 효율적으로 작업을 구현할 수 있다.

(3) Experiment

Exp. 1 : Surface Tracking

앞에서 설명한 On-Line Path Modification 명령어인 movl-with-correction을 이용하여 로봇이 임의의 곡면을 따라 이동하는 Surface Tracking 작업을 다음과 같이 프로그램하였고 그림 8은 실험사진을 나타낸다.

```
define inport1 force1 with file=force_file;
movi robot1 to loc50 with sp=10;
movi robot1 to loc51 with correction=force1;
movi robot1 to loc52 with sp=10;
movi robot1 to loc46 with sp=10;
end
```

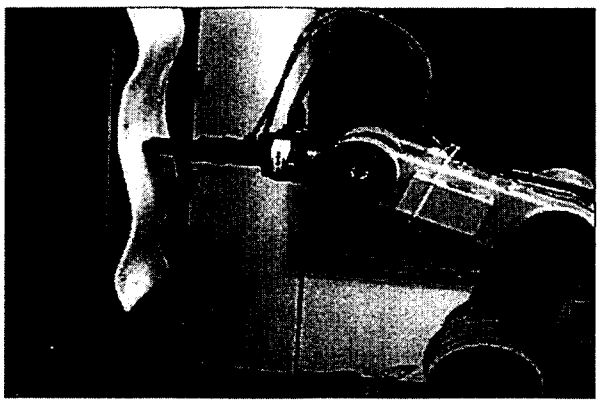


그림 8. Surface Tracking 실험사진.

Exp. 2 : Master & Slave Motion

Force Feedback 정보를 이용하여 Slave 로봇이 Master 로봇을 추적하는 Master & Slave Motion을 앞에서 설명한 Moving Target을 추적하는 기능을 이용하여 다음과 같이 프로그램하였으며 그림 9은 실험 사진을 보여준다.

```
define inport 1 force with file=force_file;
grasp robot2;
movj robot1 to loc11 with sp=10;
movj robot1 to loc11 with sp=10;
cobegin
movl robot2 to loc7 with sp=5;
move robot1 with position_eq until time<22;
position_eq=robot1+force;
coend
end
```

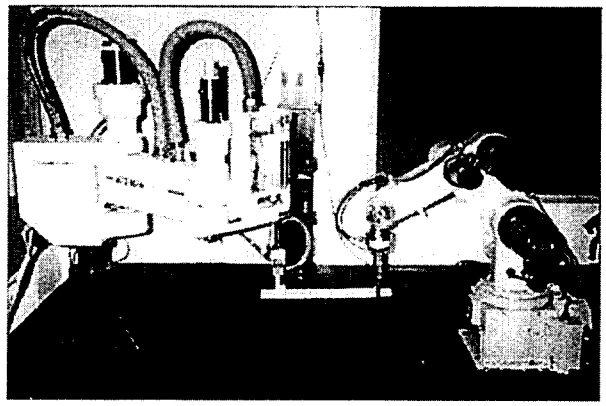


그림 9. Master & Slave Motion의 실험사진.

Exp. 3 : Visual Servoing

시각 Feedback 정보를 이용하여 SCARA 로봇이 2차원 상에서 움직이는 물체를 추적하는 작업을 앞에서 설명한 이동물체 추적 기능을 이용하여 다음과 같이 프로그램하였으며 그림 10는 실험 사진을 보여준다.

```
define inport 2 vision with file=vision_file;
movj robot2 to loc3 with sp=10;
move robot1 with position_eq until time<45;
position_eq=robot1+vision;
end
```

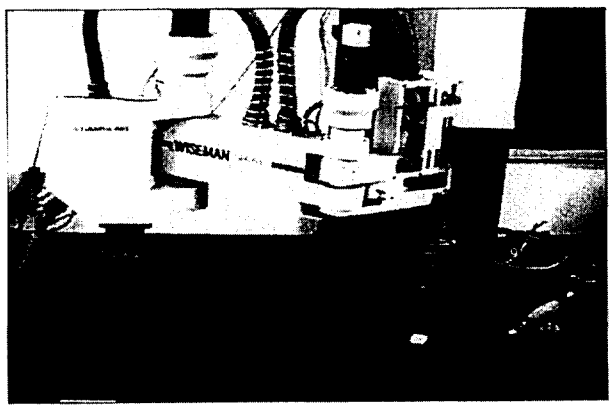


그림 10. Visual Servoing의 실험사진.

6. 맺음말

지금까지 Multi-Tasking Real Time O.S인 VxWorks를 기본으로 하여 다중센서 융합(Multi-Sensor Fusion) 능력을 갖는 다중 로봇 협조제어 시스템의 구현에 대하여 살펴보았다. 본 제어 시스템은 두대 로봇의 제어에 필요한 장애물 회피, 조건 동작(Conditional Motion) 혹은 동시동작(Concurrent Motion)과 외부 디바이스와의 동기 Motion (Conveyor Tracking)을 수행할 수 있게 구현하였고, 몇몇 작업을 통해 우수성을 입증하였다.

앞으로 본 연구와 관련한 추후 과제로는 1) 자유도가 6 관절형인 수직다관절 매니플레이터를 위한 충돌회피 알고리즘의 개발, 2) Two Arm Robot의 상대 위치를 위한 Auto-Calibration 시스템의 개발, 3) CAD Based Trajectory 생성 등이 있다.

참 고 문 헌

[1] J. W. Roach and M. N. Boaz, "Coordinating the Motions of Robot Arms in a Common Workspace," *IEEE Journal of Robotics and Automation*, Vol. RA-3, No. 5, October, pp. 437-444, 1987.

[2] R. Guptill and S.Paul, "Multiple Robotic Devices : Position Specification and Coordination," *IEEE International Conf. on Robotics and Automation*, pp. 1655-1659, 1987.

[3] A. A. Mangaser, Y. Wang, and E. S. Butner. "Concurrent Programming Support for a Multi-Manipulator Experiment on RIPS," *IEEE Int. Conf. on Robotics and Autuomation*, pp. 853-859, 1989.

[4] M. A. Truk, "A Vision System for Autonomous Land Vehicle Navigation," *IEEE Trans. Patt. Anal. Mach. Intell.*, Vol. 10, No. 3, pp 342-361, May 1988.

[5] S. Venkatesan and C. Archibald, "Real Time Tracking in Five Degrees of Freedom using Two Wrist-mounted Laser Range Finders," *Proc. of IEEE Int. Conf. on Robotics and Automation*, pp. 2004-2010, May 1990.

[6] A. J. Koivo and N. Houshang, "Real-Time Vision Feedback for Serving Robotics Manipulator with Self-Turning Controller," *IEEE Trans. on Sys. Man. Cybern.*, Vol. 21, No. 1, pp. 134-141, January/February 1991.

[7] M. A. Abid, R.O Eason and R. C. Gonzalez, "Autonomous Robotics Inspection and Manipulation using Multisensor Feedback," *IEEE Computer*, Vol. 24, No. 4, pp.17-31, April 1991.

[8] D. M. LYONS and M. A. ARBIB, "A Formal Model of Computation for Sensor-Based Robotics," *IEEE Trans. on Robotics and Automaton*, Vol. 2. No. 3. pp.280-293. June 1989.

[9] Z.Bein, S.R.Oh, I.H.Suh, J.O.Kim, and Y.S.Oh, "Automatic Assembly for Microelectronic Components," *IEEE Control Systems Magazine*. Vol.9, No. 4, June 1989.

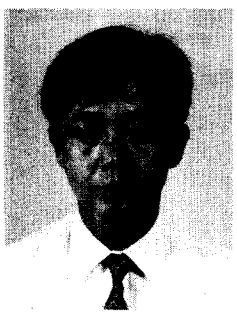
[10] Motorola Series in Solidstate Electronics, VMEbus Specification Manual, 2nd Printing Revision C.1 October 1985.

[11] S.Mujtaba and R.Goldman : AL Users Manual, Stanford Artificial Intelligent Laboratory Memo, AIM-323, 1979.

[12] R.A.Basta, R.Mehrotra, and M. R. Varanasi, "Detecting and Avoiding Collisions between Two Robot Arms in a Common Workspace," *Robot Control Theory and Application*, pp. 185-192. 1988.

[13] R.A. Basta, R. Mehrotra, and M. R. Varanasi, "Collision Detection for Planning Collision Free Motion of Two Robot Arms," *Proc. of the IEEE International Conf. on Robotics and Automation*, 1988.

저 자 소 개

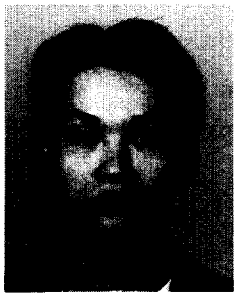


서 일 홍

1977년 서울대 공대 졸업. 1982년 한국과학기술원 졸업(공박)
1982년~1985년 3월 대우중공업 기술연구소 근무.
1987년~1988년 미국 미시간대 객원 연구원.
현재 한양대 전자공학과 교수/안산캠퍼스 교무부처장
(425-791) 경기도 안산시 사1동 1271
TEL. 0345) 400-5172 / FAX. 0345) 408-5803

여 희 주

1965년 5월 2일생.
1988년 한양대학교 전자공학과 졸업.
1990년 동 대학원 전자공학과 졸업(석사)
현재 한양대 대학원 전자공학과 박사과정
(425-791) 경기도 안산시 사1동 1271
TEL. (0345)408-5802 / FAX. 0345) 408-5803



엄 광 식

1970년 2월 5일생
1993년 한양대 전자공학과 졸업
1994년 동 대학원 기전공학과 졸업(석사)
현재 한양대 대학원 전자공학과 박사과정
(425-791) 경기도 안산시 사1동 1271
TEL. 0345) 408-5802 / FAX. 0345) 408-5803