SCARA형 로봇트를 위한 프로그래밍 시스템

°김 성 국*, 신 유 식**, 임 준 홍***, 변 증 남**

* 한국전기통신공사, ** 한국과학기술원, *** 한국항공대학

A Programming System for SCARA-Type Robots

°S.K.Kim*, Y.S.Shin**, J.Lim***, Z.Bien**

* KTA, ** KIAST EE, *** Hankuk Aviation College

ABSTRACT

In this paper, a programming system for SCARA-type robots is designed, consisting of robot language, computational facilities and programming tools for handling interconnection environments. In designing the robot language, CLRC(C Library for Robot Control) is introduced, using the general-purpose language 'C' as base programming language. Also the motion primitives for Continuous Path control as well as Point-To-Point motion are included. By means of frame and homogeneous transformations the system is capable of applying the SCARA-type robot efficiently and easily for any given task.

1. 서 론

산업용 로봇트는 주어진 작업을 수행하기 위해 여러 가지의 프로그램된 운동을 통해 부품, 도구등을 조작하거나 운반하기 위해 설계된 재프로그램할 수 있는 기계장치(reprogrammable device) 이다[1]. 기존의 자동화에 비해 산업용 로봇트를 사용한 자동화의 주요 잇점은 reprogrammability 에 있다[2]. 오늘날 산업용 로봇트 시스템은 제2세대를 맞이하고 있다. 이것은 하나 또는 여러 대의 로봇트 팔과 sensor의 결합으로 특징지워진다. 로봇트에 고도의 융통성(flexibility)을 제공함으로써 적응성(adaptivity)과 다양성(versatility)을 갖게한다[3]. 따라서 로봇트가 주어진 본래의 역할을 효과적으로 수행하도록 하기 위해서는 사용자와 로봇트 시스템을 연결시켜주도록 컴퓨터에 의한 프로그램 능력이 필요하다. 본 논문에서는 이러한 관점에서 로봇트 프로그래밍 시스템을 설계한다.

복잡한 작업에 적용될 있고 효율적인 프로그램 환경을 제공하기 위해서는 frame과 좌표 변환(coordinate transformation)의 자유로운 사용이 뒷받침되어야 한다. 이를 위해서는 로봇트 언어는 Structured programming level의 언어[4]가 되어야한다. 처음부터 새로운 base language를 설계하는 overhead를 줄이고 로봇트의 운동 제어에 주력할 수 있도록 현존하는 범용언어(general purpose language)를 기본으로 필요한 로봇트 semantics를 부가함으로써, Structured Programming Level의 로봇트 언어를 설계한다. 기본 언어(base language)로써 범용 언어인 C 언어를 사용한다. C 언어는 시스템 프로그래밍에 적합하고 pointer의 기능과 주소연산(address arithmetic)을 수행하도록 하는 능력을 가지므로 base language로써 적합하다고 할 수 있다[5]. 이와같은 compiled-languages based system은 두 가지 단점이 있다. 하나는 컴파일 시간이 프로그램의 편집과 테스트 시간을 증가시키는 점이고, 다른 하나는 프로그램이 잘못된 경우 전체작업을 멈춰야 한다는 점이다[6]. 그러나 이러한 단점들은 실제로 이와같은 시스템이 갖는 융통성(flexibility)과 일반성(generality)에 의해 충분히 보상된다.

로봇트 프로그래밍 시스템을 설계하는 본 논문에서는 다음의 사항들에 중점을 둔다. 첫째, SCARA형 로봇트를 대상으로 하여 운동제어(motion control) 인 PTP(Point-to-Point)와 CP(Continuous Path)를 구현한다. CP에는 직선, 원 그리고 포물선 운동이 포함된다. 그리하여 선택적 compliance를 갖는 SCARA형 로봇트로 하여금 융통성 있게 운동을 수행하도록 한다.

둘째, 이러한 토대위에서 C 언어를 이용한 로봇트 언어 CLRC(C Library for Robot Control)를 설계한다. 이것은 high-level 언어인 C 언어에 의하여 제공되는 모든 특성을 이용한다. User 프로그램은 C 프로그램과 같으나 로봇트 manipulation의 관점에서 전용화된 기존의 로봇트 언어와 같은 의미로 생각될 수 있다.

2. 로봇트의 운동제어

본 장에서는 적용대상인 SCARA형 로봇트에 대해 설명한다. 그리고 이들을 토대로 하여 주목적인 PTP와 CP의 로봇트 운동을 구현한다.

2.1 SCARA형 로봇트

SCARA는 Selective Compliance Assembly Robot Arm의 약어이다. 즉, 선택적 Compliance를 갖는 조립용 로봇트이다. 본 논문에서 사용된 로봇트의 각 Link에 대한 data는 다음과 같다.(그림 1 참조) 여기서 1축, 2축 그리고 3축은 revolute joint 즉, 회전운동을 한다. 그리고 3축은 prismatic joint 즉, 상하 운동을 한다. 1축과 2축은 DC모터, 3축은 공압(hydrauric) 그리고 4축은 step motor에 의해 제어되도록 되어 있다.

link	length	range
1	370[mm]	-95 ~ +95
2	230[mm]	-135 ~ +135
3	x	75[mm]
4	x	4320 [12회전]

Table 1. Link parameters

2.2 PTP(Point-to-Point)

PTP는 로봇트의 가장 단순한 운동으로써 한 점에서 다른 점으로 로봇트의 손을 이동시키는 것이다. 이때 로봇트는 중간에 지나야 할 점들이 지정되지 않는다. 이것을 구현하는 제일 쉬운 방법은 로봇트를 원하는 위치로 이동

시켜 그 때의 Joint 값들을 기억시킨 후 다시 반복 운동하도록 하는 것이다. 이런 방법은 아직도 산업계에서는 가장 널리 사용되는 방법이다. 본 논문에서는 이러한 방법 대신에 다음과 같은 방법을 사용한다. 로봇트가 운동해야 할 초기값과 최종값이 'frame'으로 주어진다. 초기값은 현재 로봇트의 위치를 나타내므로 동시에 configuration이 주어진다. 그러나 최종값은 로봇트가 가야 할 최종 location이므로 confi- guration을 정해주어야 한다. inverse kinematics를 풀어 두 configuration 중에서 Joint1이 더 작게 운동하는 경우를 택하도록 한다.([7] 참조)

2.3 CP(Continuous Path)

control hardware system은 Position pack과 Servo pack으로 구성된다[7]. 로봇트의 운동은 컴퓨터로부터 오는 pulse 열(train)에 의하여 제어된다. SCARA의 CP motion은 pulse 비(rate)와 pulse의 수에 의하여 결정된다. 또한 motor의 속도는 Servo pack에 전달되는 Voltage 입력에의해 정해진다. 즉, pulse 열은 DAC에 의해 Voltage로 바뀌게 된다. 이들 속도는 servo process에서 5ms마다 수행되는 interrupt service routine에서 의해 새로이 더해지는 pulse수라고 생각할 수 있다. 이때 새로이 더해지는 수가 양수이면 Joint는 시계 방향, 음수이면 반시계 방향이 된다. 따라서 하나의 중간점(knot point)에서 다음 중간점으로 운동하도록 한다면 이때 주는 pulse는 다음 중간점의 pulse수와 이것에 도달할 때까지 5ms마다 주는 pulse수가 있다. 수행되는 CP motion은 직선, 원 그리고 포물선 운동으로 구성된다.

(1) straight-line motion

직선의 경우 로봇트의 손이 통과하는 점들의 sequence들로써 운동을 정의한다. 이것을 구현하는데는 2가지 방법이 있다.

첫번째 방법[8]은 중간점들이 운동 중에 Cartesian space에서 직선을 따라 일정한 간격으로 보간된다. 그 다음에 kinematic equation을 풀어 이에 대응하는 중간 Joint parameter 값들을 얻게 된다. 두 번째 방법은 Taylor의 알고리즘[9]이다. 이것은 Cartesian space 상의 직선으로부터 규정된 오차(deviation) 범위를 로봇트의 손이 벗어나지 않도록 하면서 로봇트가 Joint 값의 직선보간에 의해 제어될 수 있도록 적당하게 중간점들을 결정한다.

로봇트의 작업공간 내에서 일정한 두 점 사이의 간격에 대해서도 그 위치에 따라 직선에 대한 오차는 각각 다르다. 따라서 첫번째 경우 적당한 간격을 선택하는 것이 어렵게 된다. 만약, 간격을 상당히 작게 한다면 로봇트는 그 점들을 지나야 하므로 그만큼 overhead가 따른다. 간격을 크게하는 경우 직선에 대한 오차가 커진다.따라서 후자의 방법을 택하기로 한다.

(2) circular motion

공간상에 세 점이 주어지면 그것들이 이루는 원을 따라 운동을 하도록 한다. 이때 운동방향은 주어진 점들의 순서에 따라 방향이 정해진다. 이것은 작업대상이 로봇트의 손이 원운동을 하도록 요할 때 그 필요성이 있다.

(3) parabolic motion

Lagrange interpolation에 의해 세 점을 정확히 지나면서 나머지 중간점들을 구한다. 여기서 운동방향은 주어진 점들의 순서에 의해 정해진다. 이것은 네 점의 경우 그리고 더 많은 경우에 대해서도 확장이 가능하다.

2.4 Speed

로봇트의 운동시 최대 속도는 모터의 특성과 로봇트의 손이 어떠한 위치에 있는지에 따라 결정된다. 즉, 이 속도는 로봇트 손(hand)의 속도를 말한다. 따라서 속도를 생각할 때는 로봇트의 configuration이 고려된다.

SCARA 로봇트에서 joint1과 joint2의 모터가 최대 rpm을 가질 때 다음과 같이 최대 속도와 최소 속도를 얻을 수 있다.

최소속도 = 320 mm (in worst Configuration)

최대속도 = 1800 mm (in best Configuration)

여기서 로봇트의 위치에 따라 실제 로봇트 손의 속도는 달라짐을 알 수 있다. 위의 식을 Servo Computer에서 sampling time인 5ms에 대한 속도로 바꾸면 다음과 같다.

in worst Configuration : 1.6 mm/(5ms)

in best Configuration : 9 mm/(5ms)

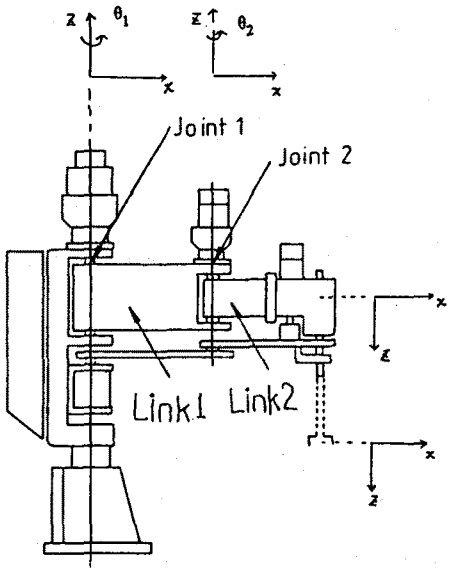

그림 1. SCARA-Type Robot의 외형

3. 로봇트 언어 CLRC

로봇트에 주어지는 작업은 공간상에서 로봇트의 손(hand or end-effector)이 위치해야 할 점들의 sequence들로 주어진다. 따라서 로봇트의 각 Joint들을 효과적으로 제어하여 주어진 작업을 수행하도록 하여야 한다. 결국 작업을 지시하는 User와 그것을 수행하는 로봇트간에는 통신수단이 필요하게 된다. 이러한 통신수단의 역할을 하는 것이 바로 로봇트 언어이다. 로봇트 프로그래밍 시스템에서 이것은 가장 중요한 요소가 된다. 성능이 좋고 제어가 잘된다 해도 사용하기 어렵고 효율적인 작업을 할 수 없다면 인간의 작업을 대체하고자 하는 로봇트의 의미는 어느정도 상실될 것이다. 본 연구에서는 SCARA형 로봇트를 대상으로 한다. 이것은 주로 조립작업에 사용하기 위하여 만들어진 로봇트이다. 이러한 SCARA형 로봇트의 목적에 맞게 효율적인 작업환경을 제공하기 위해 앞장에서 설명한 로봇트의 운동제어를 기본으로 하여 다음과 같은 사항에 중점을 두어 로봇트 언어를 설계한다.

첫째, 로봇트 언어의 level은 Structured Programming Level로 한다. 그리하여 frame과 homogeneous transformation을 자유롭게 사용할 수 있도록 한다.

둘째, 주 적용대상은 SCARA형 로봇트 이지만 기존의 전용의 로봇트 언어가 적용된 로봇트 시스템에만 사용할 수 있는 단점을 보완하기 위해 좀 더 일반적으로 적용될 수 있도록 한다.

셋째, 로봇트 언어가 확장성(extandibility)을 갖도록 한다. 즉, User가 기존에 만들어진 primitive들을 이용하여 원하는 작업을 수행하기 위해 추가의 primitive들을 만들 수 있도록 한다. 그리하여 어느정도 Task-oriented 언어를 구현할 수 있게 한다.

위의 목적에 맞도록 처음부터 새로운 로봇트 언어를 구현하려면 상당한 overhead를 안게 된다. 따라서 본 연구에서는 high-level 언어인 C 언어를 base language로 하여 이러한 overhead를 줄이도록 한다. 그리하여 로봇트의 운동제어에 중점을 둘 수 있도록 한다.

3.1 CLRC 의 구성및 특성

CLRC는 C Library for Robot Control의 약어로써 로봇트의 제어를 위해 설계된 C 언어가 확장된 형태의 s/w로써 규정될 수 있다. 따라서 로봇트 응용 프로그램은 일종의 C 프로그램이 된다. 그러나 로봇트 manipulation의 관점에서 볼때 이것은 기존에 설계된 전용의 로봇트 언어와 같은 역할을 한다. 이러한 의미에서 CLRC를 본 연구에서 적용된 SCARA형 로봇트에 사용되는 로봇트 언어로 명명한다. CLRC는 다음과 같은 특성을 갖는다.

첫째, C 언어와 같은 제어문, 구조(structure) 그리고 문법을 갖는다.

둘째, 새로운 data type 'frame'이 추가된다.

셋째, compiler-type 언어이다.

넷째, 외부 sensor와의 결합이 쉽게 이루어진다.

다섯째, 융통성(flexibility)와 일반성(generality)을 갖는다. 즉, 외부 sensor로부터 data를 받아 변화하는 상황에 쉽게 적응할 수 있고 기존의 전용의 로봇트 언어가 적용된 로봇트 시스템에만 전용으로 사용되는 단점을 보완하여 다른 로봇트 시스템에도 좀 더 일반적으로 적용될 수 있다.

3.2 동작 Mode

(1) Home mode

로봇트 시스템에 처음 power를 인가하면 로봇트는 자신의 위치를 모르게 된다. 그리하여 로봇트의 각 Joint에 기계적으로 연결된 home switch(photo coupler)까지 각 link를 느리게 이동시킨다. 이것이 곧 로봇트의 물리적 원점(physical home)이 된다. 그 다음 로봇트의 link1과 link2를 일직선으로 하여 중심에 오도록 한다. 이때 로봇트는 logical home의 위치에 있게 된다. 이 위치를 기준으로 하여 로봇트는 원하는 위치에 있도록 제어될 수 있다. Home mode는 로봇트의 위치를 초기화시킨다.

(2) Teach mode

작업공간 내의 한 물체에 대해 위치와 방향을 로봇트의 base를 기준으로 하여 정확히 알아내기는 어렵다. 그리하여 로봇트의 손(hand or end-effector)을 원하는 곳으로 이동시켜 그 물체에 대한 작업을 수행하도록 하기 위해서는 로봇트의 손의 위치와 방향을 메모리에 기억시키는 기능이 필요하다. 이렇게 기억된 data는 로봇트의 운동을 계획(planning)하는데 사용된다. 이러한 기능은 주로 따로 연결된 teach box를 통해 이루어지는 것이 보통이다. 본 논문에서는 teach box대신 terminal을 사용하도록 한다. 따라서 terminal의 특정한 key들을 이용하여 이것을 수행할 수 있도록 한다. teach되는 data의 type은 'frame'이다. 이것은 homogeneous transformation matrix를 나타내 주는 추가된 data type인 'frame'을 사용한다.

(3) Execution mode

이 mode에서는 맨 처음 Home mode가 끝난 후 로봇트로 하여금 주어진 작업을 수행하기 위해 로봇트의 운동을 제어한다. 이러한 운동을 하도록 하기 위해서는 로봇트의 손이 위치해야 할 data가 필요하게 된다. 이러한 data는 외부 sensor에서 받거나 또는 teach mode에서 직접 로봇트를 이동시킨 후 얻을 수 있다.

3.3 Language Primitives

CLRC의 language primitive들은 base language 인 C 언어와 추가된 로봇트 primitive들로 구성된다. 여기서 로봇트 primitive는 로봇트와 관련된 Kinematics, 로봇트의 운동 그리고 시스템에 관련된 s/w를 포함하게 된다. 그리고 추가의 data type으로 'frame'이 정의된다. 다음에서는 operation mode의 종류에 따른 language primitive들이 설명된다.

(1) Home mode

home( ) :

로봇트의 location(position and orientation)을 초기화한다.

(2) Teach mode

teach(loc) :

terminal상에서 manual operation으로 로봇트를 이동하여 'frame'으로 선언된 변수 'loc'의 값을 memory에 기억시킨다.

(3) Execution mode

move(loc1,loc2,speed) :

변수 'loc1'과 'loc2'는 'frame'으로 선언된다. 변수 'speed'는 integer로 선언된다. 이것의 범위는 최저 속도 1 에서 최고 속도 100 까지 주어진다. 최고 속도는 worst case의 경우에 Arm configuration에서의 최고 속도로 주어진다. 최저 속도는 이것의 1/100이 된다. 이것은 이후로 로봇트의 운동에 관련된 primitive들에 모두 적용된다. 'loc1'에서 'loc2'로 Cartesian space상에서 주어진 속도 'speed'를 가지고 PTP로 움직이도록 한다.

straight(loc1,loc2,speed) :

'loc1'에서 'loc2'로 향하는 직선을 따라 로봇트의 손이 주어진 속도로 움직인다. 여기서 User는 주어진 직선에 대해 위치와 방향에 대한 maximum deviation을 결정할 수 있다. 즉 주어진 두점 사이의 중간점(knot point)의 갯수를 결정해 줄 수 있다.

circle(loc1,loc2,loc3,speed) :

세 점으로 구성된 원 운동을 한다. 이때 방향은 'loc1'에서 시작하여 'loc2'와 'loc3'를 거쳐 'loc1'에서 멈추게 된다.

arc(loc1,loc2,loc3,speed) :

세 점으로 구성된 원의 일부분의 운동을 하게 된다. 즉, 'loc1'에서 시작하여 'loc2'를 거쳐 'loc3'에서 멈추게 된다.

parabola(loc1,loc2.loc3,speed) :

세 점으로 구성되는 포물선 운동을 하게 된다.

rectangle(loc1,loc2,loc3,loc4,speed) :

네 점으로 구성되는 사각형 운동을 한다.

(4) Others

make_coord(obj)

coord(obj)

tool(obj)

3.4 응용 프로그램

다음 보기는 frame과 좌표변환의 사용을 보여준다. 사각형의 물체에 대해 작업을 수행한다고 가정하자. 물체의 초기위치를 obj1, 나중위치를 obj2라고 하면 처음에 한 번만 teach하면(obj1의 경우) 다음부터는 다시 그러한 teach과정을 거칠 필요가 없게 된다. 이때 obj2의 값만 알려주면 된다. 여기서 obj1과 obj2는 'frame'으로 선언된다.

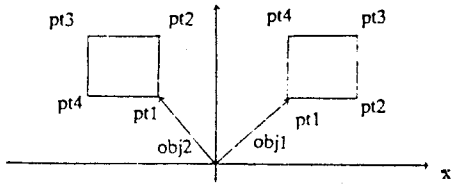

그림 2. 예제 작업

```
#include "robot.h"
main()
{
    frame pt1, pt2, pt3, pt4;
```

```
    frame obj1, obj2;

    make_coord(obj1); /* with reference to base
coordinate */
    coord(obj1);          /* frame */
    teach(pt1);
    teach(pt2);
    teach(pt3);
    teach(pt4);
    rectangle(pt1,pt2,pt3,pt4,speed);
    make_coord(obj2);
    coord(obj2);
    rectangle(pt1,pt2,tp3,pt4,speed);
} /* End of main */
```

4. 결 론

본 논문에서는 SCARA형 로봇트를 대상으로 한 프로그래밍 시스템을 설계하였다. 여기서 가장 중점을 두는 것은 로봇트 시스템과 사용자간의 통신역할을 하는 로봇트 언어이다. 즉, 로봇트 언어에서는 로봇트 제어를 위해 프로그램을 통해서 User와 로봇트 시스템간에 통신을 위한 전반적인 환경을 조성해 준다.

로봇트 언어가 설계되려면 먼저 운동 제어가 전제되어야 한다. 따라서 로봇트의 운동 제어에서 SCARA형 로봇트를 위한 CP와 PTP를 구현하였다. CP에서는 직선, 원 그리고 포물선 운동이 포함된다. 특히 직선 의 경우 Taylor의 알고리즘을 사용하였다. 이것은 운동을 수행하기 전에 모든 중간점(knot point)들을 미리 계산해야 한다. 따라서 처음에 약간의 계산시간을 요하는 단점이 있으나 미리 정해진 간격으로 중간점들을 결정하는 것보다 원하는 로봇트가 직선운동을 하기에는 더 효율적이다. 한편, Teach box의 기능은 Terminal로 대체된다. 따라서 Teach mode에서 user는 Terminal을 통해 로봇트를 이동시켜 그 점의 위치와 방향을 기억시킨다.

위의 사항들을 근간으로 하여 로봇트 언어를 설계하였다. C 언어를 base language로 하여 새로운 Robot semantics를 부가함으로써 처음부터 새로이 로봇트 언어를 설계하는 overhead를 줄였고 따라서, 로봇트의 운동 제어에 주력할 수 있었다. 설계된 프로그래밍 시스템은 다음과 같은 특성을 갖는다.

첫째, 로봇트가 주어진 작업을 하나의 물체에 대해 수행할 때 teach과정을 한번만 거치면 물체가 다른 곳으로 이동되더라도 다시 처음부터 teach할 필요가 없이 같은 작업을 옮겨진 물체에 대해 수행할 수 있다.

둘째, tool이 바뀌더라도 그 tool에 대한 data만 바꾸어 주면 쉽게 적응할 수 있어 여러개의 tool을 바꾸어 가며 추가의 부담이 없이 일련의 작업수행이 가능하다.

세째, 융통성과 일반성을 갖는다.

네째, sensor integration과 multiple robot 제어가 가능하다.

참 고 문 헌

1. Michel Parent and Claude Laurgeau, "Robot Technology v.5 Logic and Programming" Prentice-Hall, 1983
2. Tomas Lozano-Perez, "Robot Programming", Proceedings of IEEE, v.71, No.7, July 1983
3. Vincent Hayward and Richard P. Paul, "Robot Manipulator Control Using the "C" Language under UNIX", IEEE, 1983
4. Susan Bonner and Kang G. Shin, "A Comparative Study of Robot Languages", IEEE Computer, 1982
5. Brian W. Kernighan and DEnnis M. Richie, " The C Programming Language", Prentice-Hall, 1978
6. Vincent Hayward and Richard P.Paul, " Introduction to RCCL: A Robot Control "C" Library", IEEE, 1984
7. 김 성 국, "SCARA형 로봇트를 위한 프로그래밍 시스템의 설계", 한국과학기술원, 1987
8. Richard Paul, "Manipulator Cartesian Path Control", IEEE Trans on SMC, Nov 1979
9. Russell H. Taylor, "Planning and Execution of Straight Line Manipulator Trajectories", IBM Journal of Research and Design, v23, No4, July 1979