기술 문서

Kinematic Calculations

Kinematic Calculations (Inverse-kinematics)

운동학 계산 (역운동학)

July 31, 2007

운동역학적 계산 (Kinematic Calculation)

Coordinate-System Kinematic Calculations

Turbo PMAC 은 복잡한 운동학 계산들의 수행과 적용을 매우 쉽게 해 줄 수 있는 구조를 제공하고 있습니다. 운동학 계산(Kinematic Calculation)은 실제 엑츄에이터(관절)와 툴 팁(tool-tip)의 좌표간에 비선형적인 계산관계가 성립될 때에 필요하게 됩니다 .주로 직교좌표계가 아닐 때에 사용되게 됩니다. 이러한 계산은 주로 로봇을 움직이는 어플리케이션에 사용되며, 로봇 관절에 해당하지 않는 다른 엑츄에이터와 함께 사용될 수 있습니다. 예를 들면, 하나 또는 두 개의 회전 축을 가지고 있는 4 또는 5 축의 다관절 공작 툴에 대해서, 사용자는 툴의 끝점의 경로만 프로그램 하게 되고, 나머지는 콘트롤러에서 필요한 모터의 위치를 계산하게 하는 것입니다.

이러한 기능으로 인해서 기계가 어떠한 기하학적 형상을 가지더라도, 기계의 움직임에 대한 프로그램이 툴팁 (주로 직교좌표계)을 기준으로 해서 작성될 수 있습니다. 운동학 계산 루틴은 콘트롤러에 내장되어서, 경로를 작성하거나 기계를 동작시키는 사람에게는 보이지 않게 동작하게 됩니다. 이러한 루틴은 해당 기계에 대해서는 변하지 않으며, 간혹 파라메터에 또는 로직에 의해서 툴의 종류나 길이가 달라지는데 대한 일반적인 변경 사항을 적용할 수 있습니다.

터보 PMAC 의 전문용어로 풀이하면, 툴팁 좌표는 축들에 대한 것입니다. 이러한 축들은 사용자가 정의한 공학 단위를 가지며, 특정 문자(A,B,C,U,V,W,X,Y,Z)로 구분됩니다.

Note :

PMAC 의 표준 축 정의는 관절 모터와 툴 팁 축들간의 선형적 계산에 근거해서 동작하게 됩니다. 여기서 다루는 내용들은 비선형적 관계를 가지는 좀 더 복잡한 경우에 대해서 설명하고 있습니다.

전방향 운동학 (forward-kinematics)의 계산은 관절의 위치를 입력으로 사용하고, 이를 툴 팁의 좌표로 변환하게 됩니다. 이러한 계산은 툴 팁의 좌표로 프로그램되어 있는 이송 동작을 시작할 때에 처음 시작 위치에서의 좌표 값을 알아 내기 위해서 필요하게 됩니다. 각 관절에 있는 위치 센서의 값들을 변환해서 실제 툴 팁에 있는 엑츄에이터의 좌표 값을 알아낼 때에도 이와 똑 같은 계산을 사용하게 됩니다. (Turbo PMAC 의 전방향 운동학 프로그램 버퍼에서는 이러한 위치 계산 기능을 지원하지 않습니다. 그러나, 똑 같은 계산을 PLC 프로그램에 넣어서 위치 계산 목적으로 사용할 수 있습니다.)

역(방향) 운동학 (inverse-kinematics)의 계산은 툴 팁의 위치를 입력으로 사용하고, 이를 관절의 좌표로 변환하게 됩니다. 이러한 계산은 툴 팁의 좌표로 프로그램되어 있는 모든 이송의 끝점들에 대해서 필요로 하게 됩니다. 또한, 이 끝점으로 이송하는 경로가 중요할 때에는 이송하는 도중에도 일정한 간격으로 반드시 계산이 이뤄져야 합니다.

Note :

의례적인 로봇 분석에서는 관절의 위치와 그 관절 위치계산에 필요한 구동기의 위치를 구분하게 됩니다. 일반적으로, 두 위치는 같다고 할 수 있지만, 두 개의 모터가 관절을 다르게 움직이는 경우가 있을 수 있으며, 그러한 경우에는 매우 중요한 차이점을 보이게 됩니다. 만약 당신의 시스템이 관절과 구동기의 위치를 구별할 때에는, 전방향 운동학 계산은 반드시 이러한 차이점을 포함하고 있어야 하며, 관절의 위치를 중간 단계로 사용하면서 항상 구동기 위치와 툴 팁 위치 사이를 움직여야 합니다. 이 문서에서는 관절의 위치에 대해서만 언급할 것입니다. 하지만, 특정 어플리케이션에서는 구동기의 위치를 기술적으로 언급할 수도 있습니다.

운동학 프로그램 버퍼 생성하기 (Creating the Kinematic Program Buffers)

Turbo PMAC 은 운동학적 계산의 실행을 이행하기 위해서 특별한 전방향 운동학과 역(방향) 운동학 프로그램 버퍼를 사용합니다. 각각의 좌표계는 이와 같은 프로그램 버퍼를 하나씩 가질 수 있으며, 이 버퍼 내부에 있는 알고리즘은 필요한 때에 모션 프로그램의 서브루틴으로 자동 실행되게 됩니다.

전방향 운동학 프로그램 생성하기 (Creating the Forward-Kinematic Program)

"**OPEN FORWARD**"라는 온라인 명령으로 지정된 좌표계의 전방향 운동학 프로그램 버퍼를 열게 됩니다. 온라인 명령인 "**CLEAR**"는 이 버퍼 내부에 존재하는 모든 내용들을 지우게 됩니다. 뒤이어서, Turbo PMAC 에 보내지는 어떠한 프로그램 명령이라도 모두 열려 있는 버퍼에 들어가게 됩니다. (**ADDRESS**, **CMDx**, **SENDx** 를 제외한 PLC 명령도 포함합니다.) 온라인 명령인 "**CLOSE**"로 버퍼를 닫게 됩니다.

모든 전방향 운동학 프로그램의 실행 전에는 Turbo PMAC 이 자동적으로 해당 좌표계에 있는 각 모터 **xx** 에 대한 현재의 지령 위치를 전역변수인 **Pxx** 에 복사하게 됩니다. 이 값들은 실수로 정의되어 있으며, 카운트 단위를 사용하고 있습니다. 전방향 운동학 프로그램은 이 값들을 입력으로 사용하게 됩니다.

모든 전방향 운동학 프로그램의 실행 후에는 Turbo PMAC 이 해당 좌표계에 대해서 사용자의 공학 단위로 되어 있는 **Q1 - Q9** 의 값들을 취하게 되고, 이 값을 다시 해당 좌표계의 9 개 축에 대한 타겟 위치 레지스터에 복사하게 됩니다. 그러고 나서, 그 값들이 처음 프로그램 이송에 대한 시작 위치로 사용되게 됩니다. 아래 테이블은 각 변수들이 영향을 주는 축들의 위치와 suggested M 변수를 사용했을 때의 각 레지스터에 대해서 보여주고 있습니다. (디버깅 목적으로 열거되었습니다.)

Axis-Position Q-Variable	Axis Letter	Target Register Suggested M-Variable	Axis-Position Q-Variable	Axis Letter	Target Register Suggested M-Variable	Axis-Position Q-Variable	Axis Letter	Target Register Suggested M-Variable
Q1	A	Msx41	Q4	U	Msx44	Q7	X	Msx47
Q2	B	Msx42	Q5	V	Msx45	Q8	Y	Msx48
Q3	C	Msx43	Q6	W	Msx46	Q9	Z	Msx49

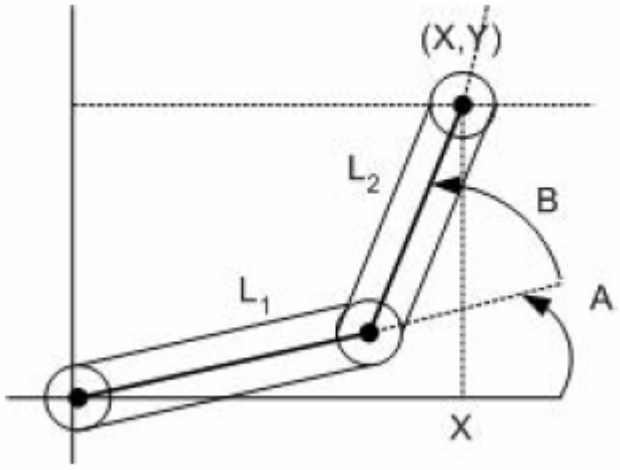
전방향 운동학 프로그램의 기본적인 목적은 좌표계에서 사용되는 각 모터에 대해서 **P1** 부터 **P32** 까지 얻을 수 있는 관절 좌표를 이용해서 팁의 좌표 값을 계산한 후에, 이 값들을 **Q1 – Q9** 범위의 변수들에 대입하는 것입니다.

여러분이 작성한 전방향 운동학 프로그램 내에서, 원점 검색이나 절대위치 읽기를 통해서 위치 참조(일반적으로 원점검색)가 적당하게 이뤄졌는지를 확인하는 것이 좋을 것입니다. 이는 각 모터의 원점 검색 종료 상태를 나타내는 비트를 보고 확인할 수 있습니다. 위치 참조가 이뤄지지 않은 상태에서는 "**run-time error**" 비트를 설정할 수도 있습니다. (원점검색이 수행되지 않았을 때)

예약된 변수들 (Reserved Variables)

만약에 시스템에서 운동학적 계산을 사용하고 있다면, 전역 변수인 **P1** - **P32** 와 좌표계 변수인 **Q1** – **Q10** 은 절대로 다른 목적으로 사용되어서는 안됩니다. 왜냐하면, Turbo PMAC 에서는 운동학 계산 루틴을 실행할 때에 이들 변수에 자동으로 사용하기 때문입니다. (**Q10** 은 아래 설명된 것과 같이, 역방향 운동학 계산에서 속도 계산을 포함하느냐 하지 않느냐를 구분하기 위해서 사용됩니다.) 만약 PVT 모드 이송을 포함하는 역방향 운동학 (inverse-kinematics) 계산이 사용될 경우에는, 추가적으로 전역 변수인 **P101** – **P132** 와 좌표계 변수인 **Q11** – **Q19** 를 다른 목적으로 사용해서는 안됩니다. 왜냐하면, Turbo PMAC 이 역운동학 계산 루틴의 속도 부분을 계산할 때에 자동적으로 이 변수들을 사용하기 때문입니다.

Example



상박(L1)의 길이가 400mm 이고 하박(L2)의 길이가 300mm 인, 2 축으로 구성된 어깨-팔꿈치 로봇을 예로 들어 보겠습니다. 어깨 관절(A)과 팔꿈치 관절(B)는 모두 각도당 1000 count 에 해당되는 분해능을 가지고 있습니다. 두 관절이 모두 0 도의 위치에 있을 때에는, 두 개의 연결부가 X 축을 따라서 쭉 펼쳐지게 됩니다. 정방향 운동학적 식은 다음과 같습니다.

$$X = L_1 \cos(A) + L_2 \cos(A+B)$$
$$Y = L_1 \sin(A) + L_2 \sin(A+B)$$

위의 식들을 적용하기 위해서, Turbo PMAC 의 1 번 좌표계에 있는 정방향 운동학적 프로그램을 사용하게 됩니다. Motor 1 번에 연결된 어깨 각도와 Motor 2 번에 연결된 팔꿈치 각도를 X 와 Y 로 된 팁의 좌표로 mm 단위로 변환하게 됩니다. 아래의 설정과 프로그램을 참고하시기 바랍니다.

```
; Setup for program
I15 = 0                          ; 수학계산에서 각도를 사용하게 합니다.
&1                               ; 좌표계 1 번을 지정합니다.
M145->Y:$0000C0,10,1             ; 모터 1 번 원점 검색 종료 비트
M245->Y:$000140,10,1             ; 모터 2 번 원점 검색 종료 비트
M5182->Y:$00203F,22,1            ; 좌표계 1 번 run-time 에러 비트
Q91 = 400                        ; L1 (mm 단위)
Q92 = 300                        ; L2 (mm 단위)
Q93 = 1000                       ; A 와 B 에 대한 각도당 카운트 값( count/deg )
; Forward-kinematic program buffer for repeated execution
&1 OPEN FORWARD                  ; 좌표계 1 번에 대한 전방향 운동학 버퍼
CLEAR                            ; 현재 들어있는 내용을 모두 지웁니다.
IF (M145=1 AND M245=1)           ; 정상적으로 위치 참조가 되었으면?
   Q7=Q91*COS(P1/Q93)+Q92*COS((P1+P2)/Q93)     ; X 축 위치
   Q8=Q91*SIN(P1/Q93)+Q92*SIN((P1+P2)/Q93)     ; Y 축 위치
ELSE                             ; 비정상적인 경우, 프로그램 종료
   M5182=1                       ; run-time 에러 비트를 설정합니다.
ENDIF
CLOSE
```

정방향 운동학 프로그램은 좌표계에 있는 모든 축들에 대해서, 역방향 운동학 프로그램에서 모터 위치들이 계산되든 되지 않든지 간에, 축의 위치를 반드시 계산해 줘야 합니다. 예를 들어, 이

팔이 팁(끝)에 수직축의 정의(C.S.1 에 #3->100Z, 밀리미터 당 100 카운트-모터와 축간의 선형적인 관계) 를 가지는 모터를 가지고 있는 경우, 위의 프로그램은 이 모터와 축에 대해서 **Q9=P3/100** 과 같은 계산을 정방향 운동학 프로그램에서 수행하도록 포함할 필요가 있습니다.

Note :

만약 정방향 운동학 알고리즘이 옳지 않고, 수학적으로 정확한 역방향 운동학 알고리즘의 역해를 갖지 않는다면, 정방향 운동학 알고리즘이 수행된 이후에 첫 번째 이송의 시작 부분에서 갑작스럽게 또는 잠재적으로 위험하게 모터가 튈 수 있습니다. 개발 초기단계에서 **fatal following error limits** 값인 **Ixx11** 을 빡빡하게 설정해서 큰 에러가 발생했을 때에 위험한 동작이 발생하지 않도록 해야 합니다.

반복문에 대한 해법 (Iterative Solutions)

몇몇 시스템, 특히 스튜어트 플랫폼(hexapods)과 같은 병렬연결 구조는 정방향 운동학 식에 대한 적당한 폐쇄형 해 (closed-form solutions)를 갖고 있지 않으며 반복 연산의 수행을 필요로 하게 됩니다. 일반적으로, 이런 경우에는 정방향 운동학 프로그램 내에서 **WHILE ... ENDWHILE** 루프에 의해 처리됩니다. 이때 절대로 무한루프가 되도록 해서는 안됩니다. 만약 일정 사이클 동안 해가 구해지지 않으면, 프로그램이 멈추도록 해야 합니다. (아래에 있는 역방향 운동학적 식에 대한 예제에서 어떻게 프로그램을 정지하는지 보세요)

이러한 경우에, 계산이 필요한 시간만큼 수행될 수 있도록 프로그램 계산 지연 변수인 **I11** 값을 디폴트 값인 0 으로 설정하는 것이 최선입니다. 만약 **I11** 값이 0 보드 크고 정방향 운동학 계산과 첫 번째 이송의 계산이 **I11** msec 이내에 끝나지 않게 되면, Turbo PMAC 은 **run-time** 에러와 함께 종료되게 됩니다. 어떠한 경우라도, 정방향 운동학 계산이 대략 25 msec 를 넘게 되면, watchdog 에러를 발생시킬 수 있습니다.

위치 보고를 위한 정방향 운동역학 (Position-Reporting Forward Kinematics)

정방향 운동학 계산의 또 다른 용도는 위치를 보고하기 위한 기능으로 사용하는 것이며, 실제 관절 위치를 계속해서 읽어서 팁(tip)의 위치로 변환해서 보고하도록 합니다. Turbo PMAC 에 있는 정방향 운동학 프로그램 버퍼는 이러한 기능을 지원하지 않습니다. (시작 위치 계산과 위치 보고를 위해서 프로그램을 사용하는 것은 잠재적으로 중복 사용과 레지스터 충돌을 발생할 수 있습니다.)

만약 Turbo PMAC 으로 작성된 어플리케이션에서 위치 보고와 초기 팁 위치를 계산하는 목적으로 정방향 운동학적 계산이 동시에 수행되길 바란다면, 위치 보고를 위한 계산은 반드시 PLC 프로그램에 들어가야 합니다. 아래의 PLC 프로그램은 예제에 나와 있는 "어깨-팔꿈치" 로봇의 위치 보고용 기능으로 사용될 수 있습니다.

```
; M-Variable definitions for actual position registers
M162->D:$8B                    ; 모터 1 의 실제 위치 레지스터
M262->D:$10B                   ; 모터 2 의 실제 위치 레지스터
; Forward-kinematic PLC program buffer for position reporting
OPEN PLC 10                    ; 좌표계 1 번에 대한 전방향 운동학 계산 PLC 버퍼
CLEAR                          ; 현재 들어있는 내용을 모두 지웁니다.
P51=M162/(I108*32*Q93)         ; 실제 A 위치 ( degree )
P52=M262/(I208*32*Q93)         ; 실제 B 위치 ( degree )
Q27=Q91*COS(P51)+Q92*COS(P51+P52)      ; 실제 X 위치 ( mm )
Q28=Q91*SIN(P51)+Q92*SIN(P51+P52)      ; 실제 Y 위치 ( mm )
CLOSE
```

역방향 운동학 프로그램 생성하기 (Creating the Inverse-Kinematic Program)

온라인 명령인 "**OPEN INVERSE**" 로 지정된 좌표계의 역방향 운동학 버퍼를 열게 됩니다. 온라인 명령인 "**CLEAR**"로 버퍼에 들어 있는 모든 내용을 지웁니다. 뒤이어서, Turbo PMAC 에 보내지는 어떠한 프로그램 명령이라도 모두 열려 있는 버퍼에 들어가게 됩니다. (**ADDRESS**, **CMDx**, **SENDx** 를 제외한 PLC 명령도 포함합니다.) 온라인 명령인 "CLOSE" 로 버퍼로 버퍼를 닫게 됩니다.

모든 역방향 운동학 프로그램의 실행 전에는, Turbo PMAC 이 해당 좌표계의 각 축에 대한 현재의 타겟 위치들을 좌표계 변수인 Q1 – Q9 범위의 변수에 복사하게 됩니다. 이 값들은 실수로 정의되어 있으며, 공학 단위를 사용하고 있습니다. 그런 다음 프로그램에서는 이 변수들을 계산을 위한 "입력"으로 사용할 수 있게 됩니다. 아래 테이블은 각 축에 대한 변수를 보여주고 있습니다.

Axis-Position Q-Variable	Axis Letter	Axis-Position Q-Variable	Axis Letter	Axis-Position Q-Variable	Axis Letter
Q1	A	Q4	U	Q7	X
Q2	B	Q5	V	Q8	Y
Q3	C	Q6	W	Q9	Z

모든 역방향 운동학 프로그램의 실행 후에는, Turbo PMAC 은 해당 좌표계에 **#xx->I** 로 정의되어 있는 각 모터에 대응하는 변수인 **Pxx** (**P1** – **P32**)에 있는 값을 읽게 됩니다. 이 값들은 실수로 되어 있으며, 모터의 기본 단위인 "count"로 읽혀지게 됩니다. Turbo PMAC 은 자동으로 이 값들을 해당 모터의 타겟 위치 레지스터에 복사하게 되고, 이 값들이 각 모터의 섬세한 보간에 사용되게 됩니다.

가끔은 역방향 운동학 해석 축에 포함되지 않는 모터들이 같은 좌표계에 있을 수 있습니다. 이 모터들은 자기 위치를 축 정의로부터 바로 얻게 되고 역방향 운동학 프로그램에 의해서는 영향을 받지 않게 됩니다.

역방향 운동학 프로그램의 기본적인 목적은 좌표계에서 사용되는 각 축에 대해서 Q1 부터 Q9 에 해당하는 팁(tip)의 위치를 이용해서 각 관절의 좌표값을 계산하고, 이 값들을 P1 – P32 범위의 변수들에 대입하는 것입니다.

Example

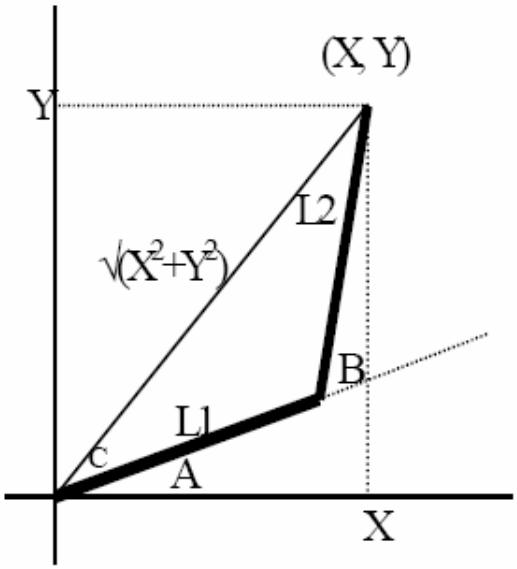
두 축으로 구성된 어깨-팔꿈치 로봇의 예제에 이어서, 좀 간편히 해석하기 위해서 B 의 값을 양수로 제한하였습니다. (오른팔 구조) 이때 역방향 운동학 계산식은 아래와 같습니다.

$$B = +\cos^{-1}\left(\frac{X^2+Y^2-L_1^2-L_2^2}{2L_1L_2}\right)$$

$$A+C = a\tan 2(Y,X)$$

$$C = +\cos^{-1}\left(\frac{X^2+Y^2+L_1^2-L_2^2}{2L_1\sqrt{X^2+Y^2}}\right)$$

$$A = (A+C)-C$$



좌표계 1 번에 대한 Turbo PMAC 의 역방향 운동학적 프로그램에 위의 식들을 적용하기 위해서는 아래의 프로그램이 사용되어야 합니다. (X 와 Y 의 밀리미터 단위의 팁(tip)좌표를 모터 1 번의 어깨 각과 모터 2 번의 팔꿈치 각으로 변환하게 됩니다.) 이 시스템의 상수인 Q91(L1), Q92(L2), Q93(cnt/deg)는 위의 정방향 운동학적 프로그램에서 사용한 것과 동일합니다.

```
; Setup for program
&1                          ; 좌표계 1 번을 지정합니다.
#1->I                       ; 모터 1 번을 C.S. 1 의 역방향 운동학 축에 할당합니다.
#2->I                       ; 모터 2 번을 C.S. 1 의 역방향 운동학 축에 할당합니다.
M5182->Y:$00203F,22,1       ; 좌표계 1 번 run-time 에러 비트
; 추가적인 시스템 상수를 미리 계산해 둡니다.
Q94 = Q91*Q91+Q92*Q92       ; L1^2 + L2^2
Q95 = 2*Q91*Q92             ; 2*L1*L2
Q96 = Q91*Q91-Q92*Q92       ; L1^2 - L2^2
; Inverse-kinematic algorithm to be executed repeatedly
&1 OPEN INVERSE             ; 좌표계 1 번에 대한 역방향 운동학 버퍼
CLEAR                       ; 현재 들어있는 내용을 모두 지웁니다.
Q20=Q7*Q7+Q8*Q8             ; X^2 + Y^2
Q21=(Q20-Q94)/Q95           ; cos(B)
```

```
IF (ABS(Q21)<0.9998)                    ; 각도 B 에 대해서 1 도의 마진(1~179 도)을 두었는가?
   Q22=ACOS(Q21)                        ; B (deg)
   Q0=Q7                                ; X into cosine argument for ATAN2
   Q23=ATAN2(Q8)                        ; A+C = ATAN2(Y,X)
   Q24=ACOS((Q20+Q96)/(2*Q91*SQRT(Q20)))        ; C (deg)
   Q25=Q23-Q24                          ; A (deg)
   P1=Q25*Q93                           ; Motor 1 = 1000A
   P2=Q22*Q93                           ; Motor 2 = 1000B
ELSE                                    ; 비정상인 경우, 프로그램 종료
   M5182=1                              ; run-time 에러 비트를 설정합니다.
ENDIF
CLOSE
```

위 예제에 대한 주석

- 양수 값의 아크코사인(arc-cosine)해를 취함으로써, 자동적으로 오른팔의 경우에 대해서 적용하게 되었습니다. 보다 일반적인 경우의 해에서는, 어떤 기준에 따라서 양수 또는 음수 값이 사용되도록 선택을 해야 합니다.
- 위에서는 좀 더 명백한 논리를 보여주기 위해서 식들을 분리해 두었지만, 계산 효율성을 높이기 위해서 여러 식들을 하나로 합치는 것이 가능합니다.
- 이 예제에서는 PEWIN 에서 변수에 대해 의미 있는 이름으로 대체해서 사용할 수 있는 매크로 기능을 사용하지 않았지만, 보다 복잡한 어플리케이션에서는 변수들을 매크로로 대체하는 것을 강력히 권장합니다.
- 이 예제에서는 역방향 운동학 해석이 불가능한 경우에 대해서, 해당 좌표계의 "run-time error" 상태 비트를 설정하여 프로그램을 정지하게 하였습니다. 이로 인해 Turbo PMAC 이 모션 프로그램을 실행을 중지하고 Abort 명령을 수행하도록 하였습니다. 이 문제에 대처하기 위해서 다른 방법도 사용될 수 있습니다.

만약에 이 로봇이 팁(tip)에 수직 축을 가지고 있는 경우에는, 모터와 축간의 정의는 일반적인 선형적 축 정의(예, **#3->100Z**, 100 count/mm 인 경우)를 사용할 수 있습니다. 이 경우에 모터의 위치는 특별한 역방향 운동학 프로그램을 사용하지 않고 계산될 수 있습니다. 또는 경우에 따라, 모터를 역방향 운동학 축(**#3->I**) 로 정의하고 모터의 위치가 역방향 운동학 프로그램 내에서 계산되게 할 수도 있습니다. (예, **P3=Q9*100**, 100 count/mm 스케일의 Z 축을 사용해서 3 번 모터의 위치를 계산합니다.)

회전 축의 Rollover(회전 뛰기)

만약에 회전하는 역방향 운동학 축이 시스템에 있고, 이 축이 "**Roll over**" 기능을 갖고 있다면, 역방향 운동학 프로그램에서는 명시적으로 **rollover** 계산을 처리할 수 있도록 해야 합니다. **Ixx27** 이 설정되었을 때에 자동적으로 **rollover** 기능을 갖고 있는 **A, B, C** 축은 역방향 운동학 축으로 사용될 수 없습니다. **Rollover** 를 제대로 처리하기 위한 방법은 현재 값과 이전 값을 차이를 취하는 것이며, 이 값의 차이는 반드시 **±180°** 이내에 있도록 해야 합니다. 이러한 기능은 Turbo PMAC 의 나머지 연산자인 '**%**' 로 구현될 수 있습니다. 그러고 나서 이 차이 값이 이전 값에 더해지게 됩니다. 수학적으로 식은 아래와 같습니다.

$$\Delta\theta = \left(\theta_{new-temp} - \theta_{old}\right)\%(-180)$$

$$\theta_{new} = \theta_{old} + \Delta\theta$$

Turbo PMAC 에서 음의 연산수인 '***-n***' (예를 들면 -180 같이)을 이용해서 나머지 연산을 수행하면, 그 결과는 항상 ***±n*** 범위에 들어오게 됩니다.

예를 들어보면, 위의 예에 있는 **A** 축이 **rollover** 기능을 갖고 있을 때에, Q25=Q23-Q24 식은 아래와 같이 대체 될 수 있습니다.

Q25=P1/Q93+(Q23-Q24-P1/Q93)%-180 ; rollover 를 처리하는 경우.

(P1/Q93) 이라는 값이 θ_{old} 이고, 이 값은 역방향 운동학 루틴의 이전 사이클에서 계산된 값이거나 처음에 정방향 운동학 루틴에서 계산된 것입니다. 또한 (Q23-Q24) 는 $\theta_{new-temp}$ 입니다. 두 값은 모두 각도 (degree) 단위입니다.

속도 계산 플래그(Flag)

Turbo PMAC 에서는 PVT 모드를 제외한 모든 이송 모드에서 역방향 운동학 프로그램에서 속도 값을 계산하지 않는다는 의미로, 자동으로 해당 좌표계의 변수인 **Q10** 의 값을 0 으로 설정합니다. 만약에 PVT 와 다른 이송 모드 모두를 사용하려 한다면, 역방향 운동학 프로그램 내에서 **Q10** 을 사용해서 처리하도록 명시해야 합니다. (아래를 참고하세요.)

반복문에 대한 해법 (Iterative Solutions)

몇몇 로봇 기구에서는 역방향 운동학 식에 대한 적당한 폐쇄형 해(closed-form solutions)를 갖고 있지 않아 반복 연산의 수행을 필요로 하게 됩니다. 일반적으로, 이런 경우에는 역방향 운동학 프로그램 내에서 **WHILE ... ENDWHILE** 루프에 의해 처리됩니다. 역방향 운동학 프로그램 내에서 **WHILE** 루프를 여러 번 실행한다고 해서 일반적은 모션 프로그램에서처럼 Blending 이송을 못하게 하지는 않습니다. 하지만, 과도한 반복 수행은 계산 수행이 필요한 시간 내에 끝낼 수 없도록 할 수 있습니다. 이러한 경우에 프로그램은 자동으로 종료되면서 **run-time error** 가 발생하게 됩니다.

PVT 모드에 대한 역방향 운동학 프로그램(Inverse-Kinematic Program for PVT Mode)

Turbo PMAC 은 PVT 모드에서 운동학 계산을 사용해서 역방향 운동학 프로그램 내에서 팁(tip) 영역에서 관절 영역으로의 속도 변환을 할 수 있습니다. PVT 모드의 이송에서, 위치 계산은 다른 이송 모드와 마찬가지로 처리됩니다. 이때 추가로 속도 변환의 계산이 반드시 수행되어야 합니다.

운동학 계산을 활성화(**Isx50=1**)시켜서 PVT 모드 이송을 수행하면, Turbo PMAC 은 역방향 운동학 프로그램을 수행하기 전에 해당 좌표계의 변수인 **Q11 – Q19** 에 PVT 명령에서 사용된 각 축의 지령 속도 값을 자동으로 복사하게 됩니다. 이 값들은 부호화 실수 변수이며, 공학 속도 단위를 가집니다. 이 속도 단위는 길이/각도 단위와 좌표계의 Isx90 으로 정의된 시간 단위에 의해서 정의됩니다. 아래 테이블은 각 축에 대한 변수 정의를 보여줍니다.

Axis-Velocity Q-Variable	Axis Letter	Axis-Velocity Q-Variable	Axis Letter	Axis-Velocity Q-Variable	Axis Letter
Q11	A	Q14	U	Q17	X
Q12	B	Q15	V	Q18	Y
Q13	C	Q16	W	Q19	Z

Turbo PMAC 은 이때에 역방향 운동학 프로그램에서 각 축(tip)의 속도를 사용해서 모터(관절)의 속도 값을 계산할 수 있도록, **Q10** 변수를 **1** 로 설정해야 합니다.

이 모드에서는 역방향 운동학 프로그램의 실행 후에, Turbo PMAC 은 해당 좌표계에서 역방향 운동학 축(**#xx->I**)으로 정의되어 있는 각각의 모터에 대해서 사용하도록 **P1xx** (**P101 – P132**) 변수 값들을 읽어오게 됩니다. 이 값들은 실수 값이며, **Isx90** 밀리세크 당 카운트 단위로 계산되어 있어야 Turbo PMAC 에서 사용할 수 있습니다. Turbo PMAC 은 해당 모터에 대해서 PVT 이송을 수행하기 위한 위치 값인 **Pxx** 와 더불어 이 값을 모터(관절)의 속도 값으로 사용하게 됩니다.

그러므로 PVT 이송에 대해서는, 역방향 운동학 프로그램이 축(tip)의 위치인 **Q1 – Q9** 값을 모터(관절)의 위치인 **P1 – P32** 로 변환해야 할 뿐 아니라, 축(tip)의 속도인 **Q11 – Q19** 값을 모터(관절)의 속도인 **P101 – P132** 로 변환해야 합니다. 일반적으로, 이 속도 변환은 기구에 대한 "inverse Jacobian matrix"의 해를 갖고 있습니다.

Example

위의 예제인 "어깨-팔꿈치" 로봇을 계속해서 예로 들면, 관절(모터)의 속도를 팁(축)의 속도의 함수로 표현한 식은 아래와 같습니다.

$$\dot{A} = \frac{L_2 \cos(A+B)\dot{X} + L_2 \sin(A+B)\dot{Y}}{L_1 L_2 \sin B}$$

$$\dot{B} = \frac{\left[-L_1 \cos A - L_2 \cos(A+B)\right]\dot{X} + \left[-L_1 \sin A - L_2 \sin(A+B)\right]\dot{Y}}{L_1 L_2 \sin B} = \frac{-X\dot{X} - Y\dot{Y}}{L_1 L_2 \sin B}$$

각도 **A** 와 **B** 는 역방향 운동학 프로그램에서 위치 부분으로 계산이 되었습니다. 여기서 주의할 것은 각도 **B** 가 **0** 또는 **180** 도에 이르게 되면 속도 값이 무한대로 되는 것입니다. 현재 예제에서는 **B** 에 대한 값을 양수로 한정하였기 때문에, 각도 **B** 가 1 도 보다 작거나 179 도 보다 큰 경우 (**sin B<0.0175**)에 대해서 에러로 처리하면 될 것입니다.

```
&1
OPEN INVERSE
CLEAR
{위치계산은 위의 예제에서 수행하였으므로 생략합니다.}

IF (Q10=1)                                  ; PVT 모드이면?
  Q26 = SIN(Q22)                            ; sin(B)
  IF (Q26>0.0175)                           ; 특이점 부근(속도무한대)이 아니면?
      Q27=Q91*Q92*Q26                       ; L1*L2*sinB
      Q28=COS(Q25+Q22)                      ; cos(A+B)
      Q29=SIN(Q25+Q22)                      ; sin(A+B)
      Q30=(Q92*Q28*Q17+Q92*Q29*Q18)/Q27     ; dA/dt
      Q31=(-Q7*Q17-Q8*Q18)/Q27              ; dB/dt
      P101=Q30*Q93                          ; #1 speed in cts/(Isx90 msec)
      P102=Q31*Q93                          ; #2 speed in cts/(Isx90 msec)
  ELSE                                      ; 특이점 부근(속도무한대)이면?
      M5182=1                               ; run-time error bit 를 설정합니다.
  ENDIF
ENDIF
```

CLOSE

이러한 경우에, 이미 역방향 운동학 알고리즘의 위치 계산 부분에서 확인을 했기 때문에, 따로 각도 **B** 가 **0** 도 또는 **180** 도 부근에 있는지 확인하는 것은 불필요 합니다. 여기에서 체크하는 부분을 보여주는 것은 방법적인 원칙을 설명하기 위한 것입니다. 이 예제에서는 특이점(singularity) 근방에서만 run-time 에러를 발생시키고, 나머지에 대해서는 모든 동작이 가능합니다.

운동학을 통한 좌표계 변환 (Coordinate System Transformation with Kinematics)

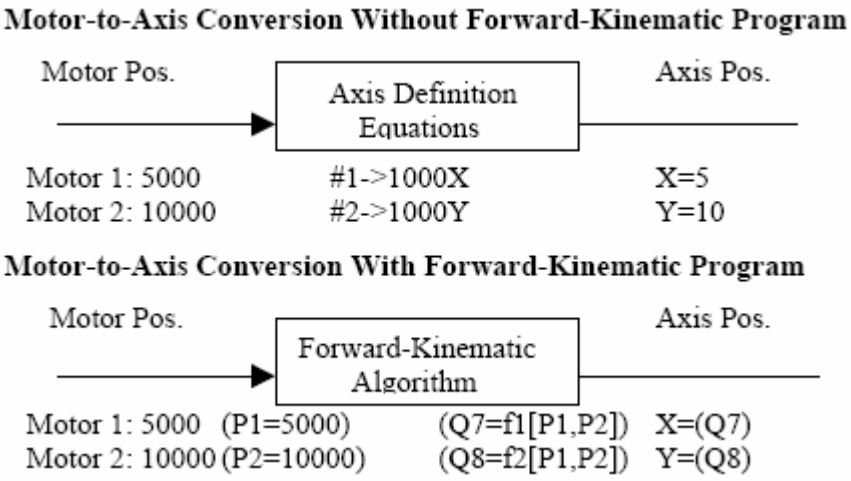
운동학 알고리즘을 사용하는 좌표계에 대해서도 일반적인 축 정의 선언을 사용하는 좌표계와 동일하게, 온라인 명령인 **{axis}=** 과 버퍼 명령인 **PSET** 명령을 사용해서 프로그램의 원점을 이동시킬 수 있습니다. 이들 중 하나의 명령이 실행되면, Turbo PMAC 은 해당 좌표계에 있는 모터들에 대해서 새로운 위치 바이어스 레지스터(suggested M-variable **Mxx64**)를 계산하기 위해서 역방향 운동학 알고리즘을 수행하게 됩니다. 이 부분은 사용자들에게는 보이지 않게 수행되며, 결과적으로 각 축에 대한 프로그램 원점에 오프셋을 적용하게 만듭니다. **X**,**Y**,**Z** 축에 대한 축 변환에도 운동학 알고리즘이 사용될 수 있습니다.

운동학 프로그램 실행하기 (Executing the Kinematic Programs)

좌표계에 대해서 정방향과 역방향 운동학 프로그램 버퍼가 생성되고 난 후에는, Turbo PMAC 은 운동학적 계산수행을 활성화하는 좌표계 시스템 변수인 **Isx50** 이 1 로 설정되면 자동적으로 적당한 시점에 이 프로그램들을 실행하게 됩니다. 적당한 시점에 운동학 프로그램이 실행되게 하기 위해서 일반적인 모션 프로그램을 변경할 필요는 전혀 없습니다.

정방향 운동학 프로그램은 **Isx50** 이 1 로 설정되어 있을 때에, R(run) 또는 S(step) 명령이 주어질 때 마다 자동적으로 실행되게 됩니다. 이러한 동작은 모터(관절)가 프로그램에서의 마지막 이송 후에 조그(Jog) 명령 등으로 이동한 후에도 처음 이송 위치 계산에 대해서 시작 축(tip)의 위치가 정확한지를 확인하기 위해서 수행됩니다. 정방향 운동학 프로그램은 또한 **Isx50** 이 1 로 설정되어 있을 때에 **PMATCH** 명령이 주어질 때 마다 매번 자동으로 수행되게 됩니다.

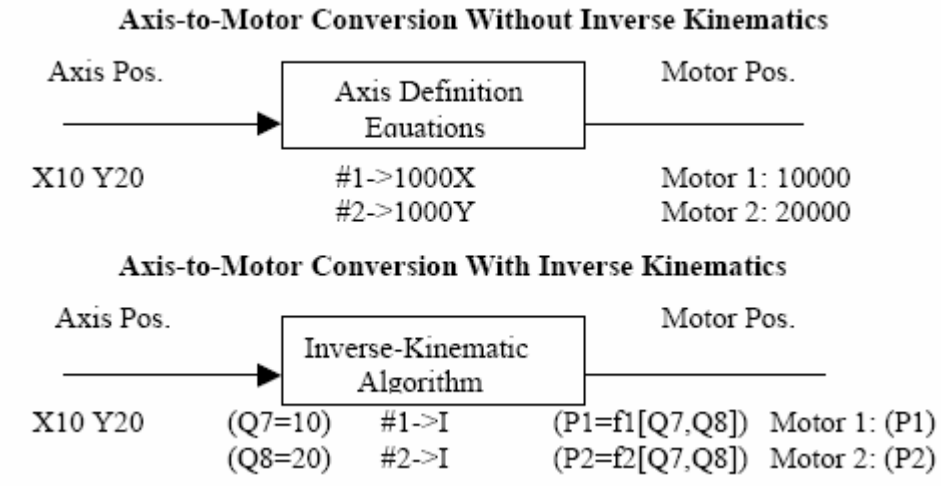
(Isx50=0 이고 일반적인 축 정의 선언을 사용했을 때에, Turbo PMAC 은 현재의 모터 지령 위치로부터 축의 시작 위치를 구하기 위해서 축 정의 선언 식을 수학적으로 역 변환 하는 동일한 기능을 수행합니다. 축 정의 선언은 기술적으로는 역방향 운동학 식과 동일합니다. 따라서, 이 정의의 수학적 역 변환은 전방향 운동학 식이 됩니다. 표준 축 정의 선언은 수학적으로 선형 식에만 한정되어 있기 때문에, 그 식의 역 변환은 일반적으로 자동으로 구해지게 됩니다.)



역방향 운동학 프로그램은 Turbo PMAC 이 모션 프로그램 수행 중에 새로운 축의 위치를 계산할 때마다 매번 수행되게 됩니다. 이 동작은 **RAPID** 모드와 같은 분할 되지 않은 이송에 대해서는 각 프로그램 이송 블록의 끝에서 수행됩니다. **Isx13>0** 로 설정되었을 때의 LINEAR 와 CIRCLE 이송 모드와 같은 분할된 이송에 대해서는 각각의 중간 분할 위치에서 매 **Isx13** 밀리세크 마다 수행됩니다.

Note:

일반적인 축 정의 선언에서는 축 위치로부터 모터의 위치를 구하기 위해서 축 정의 선언에 사용된 식을 동일하게 사용하게 됩니다.



RAPID 모드와 같이 프로그램 된 끝점에서만 역방향 운동학 프로그램이 수행될 때에는, 모든 보간은 관절(모터) 영역에서 이뤄지게 됩니다. 이러한 경우, 프로그램 된 끝점이 매우 멀리 있다면 위치에서 위치로 이동하는 팁(tip)의 경로가 좋을 수 없습니다. 다시 말해, 일반적으로 곧은 직선이 아닙니다.

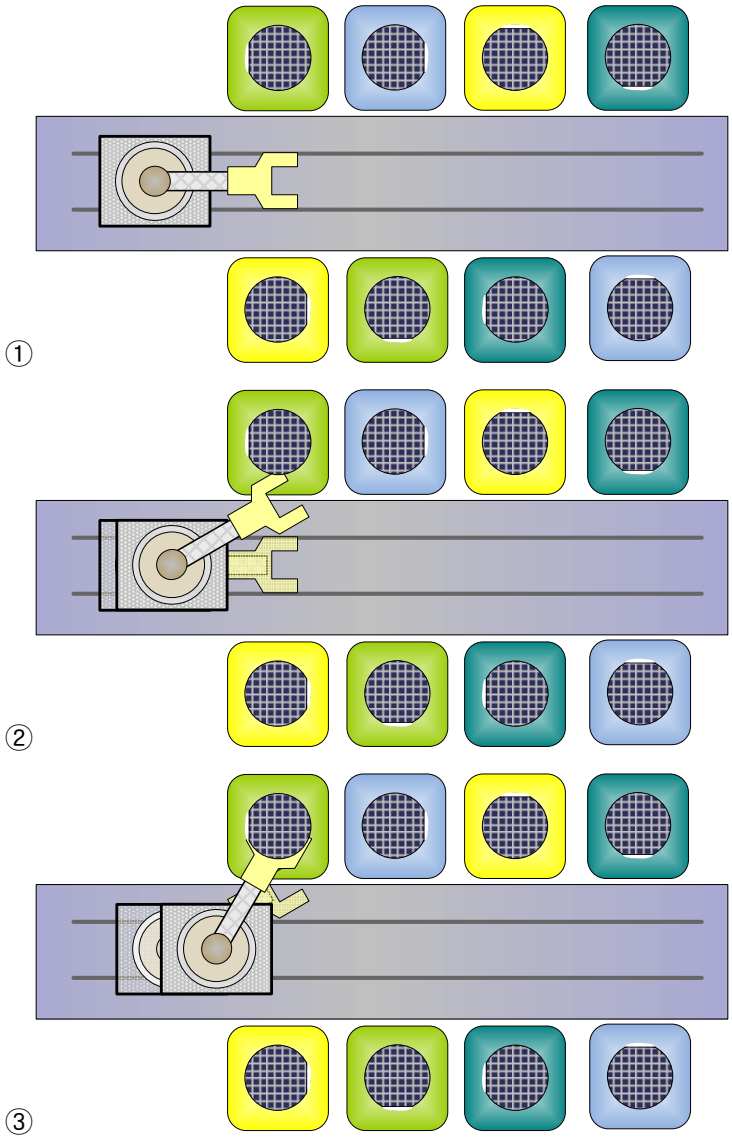
매 중간 분할 범위마다 역방향 운동학 프로그램이 수행될 때에는, **coarse interpolation** (대충의 보간)이 팁(축) 영역에서 수행되게 되어 경로가 좋아지게 됩니다. 분할된 좌표를 관절(모터)의 위치로 변환한 이후에는, 관절(모터) 영역에서 **cubic spline**(3 차곡선함수) 으로 각 분할 영역간의 **fine interpolation** (정밀한 보간)이 이뤄지게 됩니다. 하지만, 분할 조각들이 가까이 있기 때문에 (전형적으로 **5 ~ 20 msec** 마다) 이상적인 팁의 경로에서 벗어나는 양은 무시할 수 있을 정도 입니다.

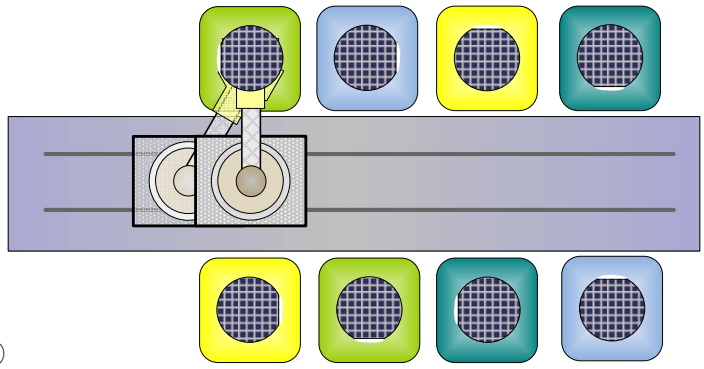
만약에 해당 좌표계에 대해서 특별한 **lookahead** 버퍼가 활성화 된 경우(해당 좌표계에 Lookahead Buffer 가 정의되었을 때의 LINEAR 또는 CIRCLE 모드의 이송, **Isx13>0** and **Isx20>0**), 관절(모터)에 대해서 계산된 내부적인 스플라인 조각들은 자동적으로 **lookahead buffer** 에 들어가게 됩니다. 여기서 이 값들은 각 모터의 위치, 속도, 가속도 제한 값에 대해서 연속적으로 체크됩니다. 이러한 기능으로 인해 Turbo PMAC 은 특이점(singularity) 근처에서 발생할 수 있는 예외적인 동작에 대해서 자동으로 체크하고 교정할 수 있게 됩니다. 따라서, 사용자가 그러한 계산을 할 필요는 없습니다.

간단한 운동학 프로그램 작성하기

아래에서는 지금까지 살펴본 내용을 바탕으로 간단한 운동학 프로그램을 작성해 보도록 하겠습니다. 예제에서 사용하게 될 시스템은 간단한 웨이퍼 반입/반출 장비입니다. 아래 그림에서 보시는 바와 같이 가운데 레일을 통해서 로봇이 좌우로 움직이게 되며, 로봇 팔은 몸체의 중심을 기준으로 회전하게 됩니다. 웨이퍼를 반입/반출하는 과정에서 창고의 폭이 좁으므로 로봇 팔은 수직으로 상하 운동을 해야 합니다. 이 경우에 일반적인 선형적 축 정의로는 로봇 팔을 수직으로 움직일 수가 없습니다. 따라서, 이 로봇에 대한 운동학 프로그램을 작성해서 쉽게 구동이 될 수 있도록 해 보겠습니다.

아래 그림들은 로봇이 웨이퍼를 반출하러 가는 과정을 순서대로 도식화 한 것입니다.

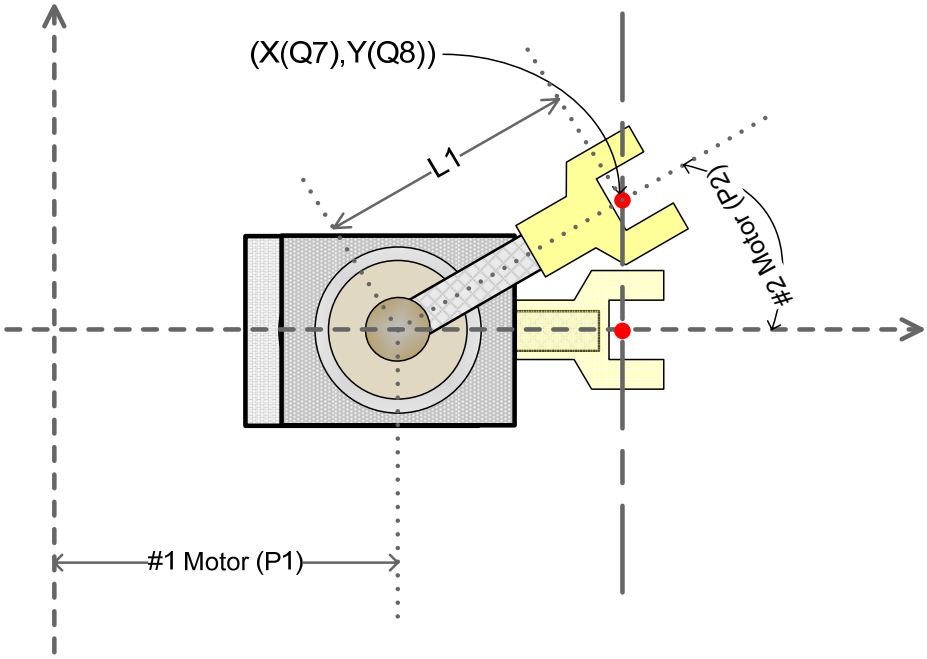




④

로봇은 창고에 수직으로 팔을 진입하기 위해서 그림에서와 같이 ①②③④ 의 과정을 거치게 됩니다. 실제 운동학 프로그램을 사용하면 **X,Y** 방향에 대해서 선형적인 이송을 할 수 있도록 구현할 수 있습니다.

아래 그림은 로봇의 이동 방향을 **X,Y** 로 두고 로봇 팔의 Tip 좌표를 이용해서 운동학 프로그램 식을 위한 그림입니다.



#1 Scale : 1000 cts/mm (Q91)

#2 Scale : 5.556 cts/deg (Q92)

L1 : 300 mm (Q93)

X : Q7

Y : Q8

#1 : P1

#2 : P2

위의 그림에 의해서 운동학 프로그램을 각각 구하면 아래와 같습니다.

Forward Kinematics

$$Q7 = \frac{P1}{Q91} + Q93*\cos\left(\frac{P2}{Q92}\right)$$

$$Q8 = Q93*\sin\left(\frac{P2}{Q92}\right)$$

Inverse Kinematics

$$P1 = Q91 * (Q7 - \sqrt{Q93*Q93 - Q8*Q8})$$

$$P2 = \sin^{-1}\left(\frac{Q8}{Q93}\right) * Q92$$

{운동학 프로그램 작성}

```
; Setup for program
I15 = 0
&1
#1->I
#2->I
M162->D:$8B
M262->D:$10B
M145->Y:$C0,10,1
M245->Y:$140,10,1
M5182->Y:$00203F,22,1
Q91 = 1000
Q92 = 5.556
Q93 = 300
; Kinematic Programs
&1 OPEN FORWARD
        CLEAR
        IF (M145=1 AND M245=1)                  ;정상적으로 원점검색 후...
            Q7 = P1/Q91 + Q93*COS(P2/Q92)       ;X 축의 좌표
            Q8 = Q93*SIN(P2/Q92)                ;Y 축의 좌표
        ELSE
```

```
            M5182 = 1                                   ; C.S.1 run-time error
        ENDIF
CLOSE
&1 OPEN INVERSE
        CLEAR
        IF ( ABS(Q8)!>Q93)                  ; Y 좌표 값이 L1 보다 작은 경우에만 유효
            P1 = Q91*(Q7-SQRT(Q93*Q93-Q8*Q8))   ; #1 모터의 위치
            P2 = ASIN(Q8/Q93)*Q92               ; #2 모터의 위치
        ELSE
            M5182 = 1                           ;C.S.1 run-time error
        ENDIF
CLOSE
; Kinematic Calculation Enable & Set Move Segmentation Time
I5150 = 1                   ; kinematic calculation enable
I5113 = 5                   ; move segmentation time (in msec)
; Position Reporting PLC
OPEN PLC 2
        CLEAR
        P150 = M162/(I108*32)/Q91 + Q93*COS(M262/(I208*32)/Q92)
        P151 = Q93*SIN(M261/(I208*32)/Q92)
CLOSE
```