컴퓨터에 의해 제어되는 Robot Arm의 제어구조와 제어방법

鄭 明 振
韓國科學技術院 電氣및 電子工學科(工博)

Ⅰ. 서 론

Robot arm은 여러 개로 연결된 rigld bodies의 집합체이며 일반적으로 첫번째 link는 기본 좌표계가 있는 받침대에 연결되어 있고 마지막 link는 assembly task를 수행할 수 있는 기계손(end-effector 혹은 hand)이 붙어 있는 links와 joints의 집합체이다(그림1).

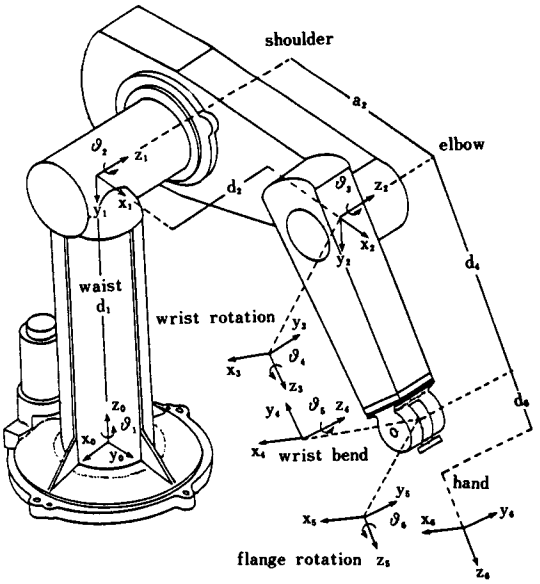

그림 1. Robot arm(PUMA)

이 글에서는 이러한 robot arm에 대한 제반 사항과 제어 구조 그리고 제어기의 설계에 문제되는 점과 지금까지의 제어 방법에 대한 연구 결과를 소개한다.

Ⅱ. Robot Arm의 제어구조와 제어방식

대부분의 자동 생산기기는 단지 생산과정에서 특정한 지정된 기능만을 수행하는 특수 목적기기였다. 이러한 기기는 다른 생산과정에서는 이용될 수 없는 등 적응 능력이 제한되었기 때문에 간단한 reprogram으로 여러가지 제조생산 과정에 이용될 수 있는 로보트가 최근 각광을 받고 있다. 최근 중·소형의 컴퓨터의 기술 발전(계산속도, 기억용량)과 가격 절감으로 더욱 로보트의 잠재 능력을 개발해 갈 수 있으며 더욱 인간과 같은 판단 능력이 있고 주위 환경에 적응할 수 있는 로보트의 출현이 예상되고 있다. 로보트의 이용에 따른 여러가지 중요한 이점은 생산가의 절감, 품질 향상에 따른 신뢰도를 높일 수 있고 인간이 할 수 없는 작업조건(채탄, 채굴, 방사선 물질 취급, 우주나 심해 자원 개발)이나 너무나 단조로운 작업조건(품목 분류) 하에 충분한 기능을 발휘할 수 있다는 것 등이다.

최초의 robot arm은 1947년 아르곤국립연구소(Argonne National Lab.)에서 인체에 해로운 방사선 물질 취급을 위해 개발되었다. 최초의 robot arm은 master-slave형이였으며 기계적으로 연결되어 동작되었기 때문에 많은 공간을 차지하였으며 힘에 제한을 받았고 humar operatom의 도움을 요하였다. 이후에 개발된 형은 joy stick이나 toggle switches에 의해 원격 조정되었으나 이 역시 동작시 인간의 도움이 요구되었다. 1961년 디지탈 컴퓨터의 발달과 함께 MIT Lab[1]에서 컴퓨터에 의한 robot arm이 개발되었으며, 주로 AI(artificial intelligence)를 위한, 이 후 Stanford AI Lab., MIT AI Lab., Propevs Lab., JPL (Jet Propulsin Lab.)등 여러 학교 기관 연구소와 일본 등지에서 연구가 진행되었다. 최근 들어 국제 경쟁에 따른 생산가 절감이나 품질 개선을 위해 industrial robot arm에 대한 연구가 급속히 진행되고 있다. 이러한 robot arm의 발전과 함께 제어 방식을 크게 3단계로 나눌 수 있다.

첫째는 1947년 이후 1960년까지 수동식 방식(manual

control)으로 모든 제어 신호가 human operator에 의해 아날로그 방식으로 입력되는 것이며 이 방식은 현재도 주로 원자력 분야에서 널리 이용되고 있다. 둘째는 1961년 이후 1970년대 중반까지의 프로그램 방식(program control)으로 제어 신호가 인간에 의해 manual teaching이나 recording에 의해 입력된 후 입력된 동작을 반복하는 것으로 많은 반복된 동작을 요하는 assembly task(자동차 산업)에 많이 사용되고 있다. 셋째는 지금까지의 동일한 동작만을 하는 경지를 벗어난 모든 것이 컴퓨터에 의해 움직이는 로보트 시스템의 잠재 능력 개발을 위한 컴퓨터 제어 방식(computer-control)에 대한 연구가 활발히 진행되고 있다.

이러한 로보트 시스템은 다음과 같은 3개의 서보시스템으로 나눌 수 있다(그림 2).

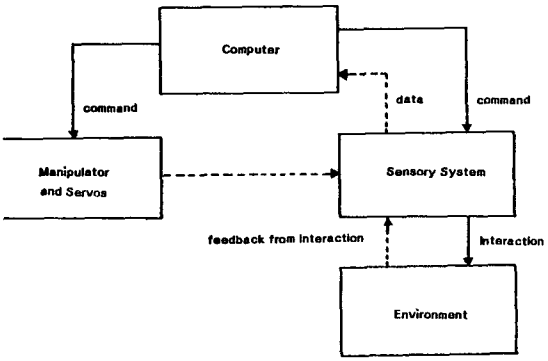

그림 2. 로보트 시스템

첫째가 robot arm(manipulator)와 서어보 시스템,
둘째가 컴퓨터 시스템,
셋째가 감각 시스템(sensory system)이다.

Robot arm은 컴퓨터로 제어될 수 있는 multi degree-of-freedom manipulator(보통 6 degree-of-freedom)이며 서어보 시스템은 robot arm의 각 joint를 움직이기 위한 기계적인 것 혹은 전기 기계적인 actuator들로 이루어져 있다. 컴퓨터 시스템은 host 컴퓨터와 특수 목적 프로세서 그리고 여러가지 임무 수행을 위한 소프트웨어로 이루어져 있으며 감각 시스템은 위치와 속도 측정을 위한 센서는 물론 주위 환경에 대한 정보를 얻기 위한 압력 센서(pressure 혹은 force), 접촉 센서(tactile) 그리고 물체의 거리, 이동 속도 그리고 방향 크기등을 알기 위한 visual sensor 등으로 이루어져 있다.

그리고 이에 근거한 로보트 시스템의 hierachical 제어 구조는 다음과 같다(그림 3).

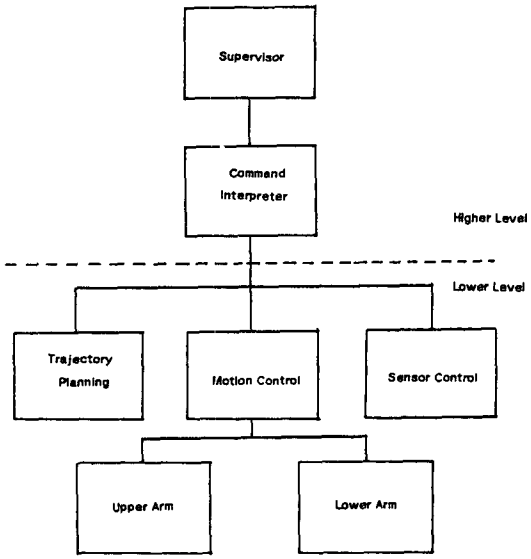

그림 3. 로보트 시스템의 제어 구조

제어 구조는 일반적으로 2개의 level로 이루어져 있으며 윗 level은 인간의 두뇌와 중추신경에 해당되는데 인간과 로보트 시스템과의 커뮤니케이션을 위한 interface program과 자체내의 모든 움직임을 통괄하는 executive program 그리고 이러한 명령을 받아 해석하고 아랫 level의 서브 시스템에 적절한 정보를 제공하고 시작시키는 command interpreter로 이루어져 있다. 한편 아랫 level은 운동신경 내지 감각기관에 해당하며 이러한 서브 시스템을 위한 좀 더 구체적인 계획이 이루어지는 곳이며 궤도 계획(trajectory planning) 시스템, 운동 제어(motion control) 시스템 그리고 감각 제어 시스템(sensory control)로 이루어져 있다.

궤도 계획 시스템은 감각 센서를 통하여 윗 level로부터 기계손(hand)의 위치와 물체의 위치에 대한 정보를 받아 robot arm이 움직이는 안전한 궤도 계획(충돌 방지)를 세운다. 그리고 운동 제어 시스템은 궤도 계획 시스템으로 부터의 궤도를 정확히 따르게끔 위치, 속도, 혹은 가속도 등을 제어 방법에 따라 입력을 계산하여 서어보 시스템을 작동시킨다. 그리고 감각 제어 시스템은 주위 환경과 상호 작용하게끔 제어하며 동시에 기계손(hand)의 위치와 방향(혹은 joint의 위치와 속도), 물체의 무게, 크기, 방향, 이동 속

도 등에 대한 정보를 얻는다. 여기에서 로보트 제어 시스템의 구조중 아랫 level 특히 운동 제어 시스템에 대해 집중적으로 소개코자 한다.

Ⅲ. Robot Arm의 기구학적인 면

제어 방법을 논하기 전 제어기 설계에 필요한 robot arm의 수학적인 모델을 얻기 위해 robot arm의 kinematics와 dynamics를 간단히 소개한다.

Robot arm은 joint와 link가 하나의 degree-of-freedom을 이루며 임의 방향에서 물체에 접근하기 위해선 6 degree-of-freedom을 요한다. 이러한 joint 와 link는 상호간 연결되어 서로 상대적으로 움직이기 때문에 인접한 link사이(좌표축)의 상호 관계는 매우 복잡하며 이러한 문제는 robot arm의 수학적 모델을 복잡하게 하는 요인이 된다. 이러한 link간의 상호 관계는 4×4 Denaoit-Hartenberg 매트릭스[2, 3]에 의해 체계적으로 표현될 수 있다(그림 4).

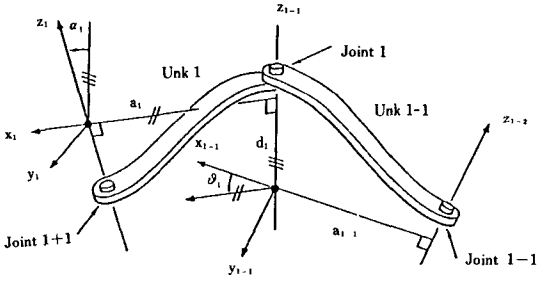

그림 4. Joint-link의 기하학적인 변수

그림에서 볼 수 있듯이 각 joint 축은 두 link의 연결 부분에 형성되며 각 joint는 두개의 수선(normal line)을 갖는다.

i 번째 link의 좌표계를 (x_i, y_i, z_i)라 하고 $(i-1)$번째 link의 좌표계를 $(x_{i-1}, y_{i-1}, z_{i-1})$라 할 때 i번째 좌표계에 의해 표시되는 위치는 2번의 translation 과 2번의 rotation에 의해 $(i-1)$번째 좌표계에 의해 다음의 매트릭스에 의해 상대적으로 표현될 수 있다.

$$T_{i-1}^{i}=\begin{bmatrix} \cos\theta_i & -\cos\alpha_i \sin\theta_i & \sin\alpha_i \sin\theta_i & a_i \cos\theta_i \\ \sin\theta_i & \cos\alpha_i \cos\theta_i & -\sin\alpha_i \cos\theta_i & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad (1)$$

여기에서 파라미터 a_i는 두 joint를 연결하는 공통 수선(common normal)의 거리이며 α_i는 a_i에 수직되는 평면과 z_i 축과의 사이각이다(오른손 법칙). θ_i는 joint축을 연결하는 두 수선사이의 각도이며(오른손 법칙) d_i는 joint축 상에 나타나는 두 공통 수선의 거리이다. 여기에서 a_i와 α_i는 robot arm의 설계시 결정되어지며 joint형에 따라 회전 joint인 경우 θ_i가 변수 d_i가 정수가 되며 직선 joint인 경우 d_i가 변수 θ_i가 정수가 된다.

식 (1)을 이용하여 i 번째 좌표계에 의해 표현된 위치 벡터 $r_i=(x_i, y_i, z_i, 1)^T$를 $(i-1)$번째 좌표계의 위치 벡터 $r_{i-1}=(x_{i-1}, y_{i-1}, z_{i-1}, 1)^T$으로 나타내기 위해선 아래와 같은 식을 이용한다.

$$r_{i-1}=T_{i-1}^{i}\ r_i \quad (2)$$

각 joint 상호간의 위치와 방향도 중요하지만 제일 중요한 것은 마지막 link(hand)의 위치와 방향이다. 이를 알기 위해선 식 (1)에 나타난 4×4 매트릭스를 연속적으로 곱하여 나아가면 된다.

$$T_0^n=T_0^1\ T_1^2\ \cdots\ T_{n-1}^n \quad (3)$$

여기서 n은 joint 수이며 0은 기준 좌표계이다.

위의 식들은 robot arm의 dynamics의 결정에 주요한 식들이며 이 식들은 주어진 변수$(\alpha_i, d_i, a_i, \theta_i)$로부터 joints들의 위치와 방향을 구하는데 이용된다.

한편 주어진 joint의 위치와 방향으로부터 회전 joint인 경우 θ_i를 구하기 위해선 robot arm의 기하학적인 면을 충분히 이해하여야 한다. 전자를 kinematic problem이라 부르고 후자를 inverse kinematic problem이라 부르는데 후자는 주로 궤도계획에 주로 사용된다.

Ⅳ. 로보트의 수학적인 모델

Robot arm의 수학적인 모델은 앞에 소개한 kinematics와 뉴톤의 원리나 Lagrangian 원리에 의해 구하여 질 수 있다. 지금까지 여러가지 모델을 유도하는 방법들이 발표되고 있지만 가장 대표적인 방법은 Lagrange-Euler(L-E) 방법[4]과 Newton-Euler(N-E)[5] 방법이 있다. 이 방법들은 수학적으로 동일하나 수식의 형태때문 서로 장단점을 갖는다. 즉 L-E 방법은 주어진 joint에 대한 값(위치, 속도, 가속도) 로부터 joint torque를 구하는데 많은 계산 시간을 요하는 반면 robot arm에 대한 explicit한 형태를 볼 수 있어 제어 설계에 큰 잇점을 주고 있다. 한편 N-E방법은 recursive한 특징때문에 주어진 joint에 대한 값으로부터 joint torque를 구하는데 요하는 계산시간을 상당히 단축할 수 있으나 유도된 수식으로 부터 제어기를 설계하는데는 부적당하다. 앞에 소개한 방법에 의해 최종적으로 얻어지는 수학적인 모델은 다음과 같

은 2次 벡터 미분방정식으로 나타낼 수 있다.

$$D(\theta)\ddot{\vartheta}+H(\vartheta,\dot{\vartheta})+G(\theta)=\tau \qquad (4)$$

여기서 $D(\theta)$는 n×n symmetric inertia 매트릭스이며 $H(\theta,\dot{\vartheta})$는 n×1 Coriolis와 원심력 벡터이며 $G(\theta)$는 n×1 중력 벡터이다.

τ는 최종 joint 토오크이며 n×1 벡터로 나타낼 수 있다.

위의 두 방법에 의한 토오크 계산에 필요한 operation은 다음 표와 같다(표 1, 2).

표 1. L-E 방법에서의 계산

Lagrange-Euler Equations of Motion	Multiplications	Additions
T_j^i	$32n(n-1)$	$24n(n-1)$
$-m_j g\frac{\partial T_j^i}{\partial \vartheta_i}\bar{r}_j$	$4n(9n-7)$	$\frac{1}{2}n(51n-45)$
$\sum_{j=i}^{n} m_j g\frac{\partial T_j^i}{\partial \vartheta_i}\bar{r}_j$	0	$\frac{1}{2}n(n-1)$
$\mathrm{Trace}\left\{\frac{\partial T_0^k}{\partial \vartheta_j}J_k\left[\frac{\partial T_0^k}{\partial \vartheta_i}\right]^T\right\}$	$\frac{128}{3}n(n+1)(n+2)$	$\frac{65}{2}n(n+1)(n+2)$
$\sum_{k=\max(i,j)}^{n}\mathrm{Trace}\left\{\frac{\partial T_0^k}{\partial \vartheta_j}J_k\left[\frac{\partial T_0^k}{\partial \vartheta_i}\right]^T\right\}$	0	$\frac{1}{6}n(n-1)(n+1)$
$\mathrm{Trace}\left\{\frac{\partial^2 T_0^m}{\partial \vartheta_j \partial \vartheta_k}J_m\left[\frac{\partial T_0^m}{\partial \vartheta_i}\right]^T\right\}$	$\frac{128}{3}n^2(n+1)(n+2)$	$\frac{65}{2}n^2(n+1)(n+2)$
$\sum_{m=\max(i,j,k)}^{n}\mathrm{Trace}\left\{\frac{\partial^2 T_0^m}{\partial \vartheta_j \partial \vartheta_k}J_m\left[\frac{\partial T_0^m}{\partial \vartheta_i}\right]^T\right\}$	0	$\frac{1}{6}n^2(n-1)(n+1)$
$\tau = D(\vartheta)\ddot{\vartheta} + H(\vartheta,\dot{\vartheta}) + G(\vartheta)$	n^2	$n(n+1)$
Total	$\frac{128}{3}n^4+\frac{512}{3}n^3$	$\frac{98}{3}n^4+\frac{781}{6}n^3$
Mathematical Operations	$+\frac{841}{3}n^2+\frac{82}{3}n$	$+\frac{628}{3}n^2+\frac{137}{6}n$

표 2. N-E 방법에 의한 계산

Newton-Euler Equations of Motion	Multiplications	Additions
ω_i	9 n	7 n
α_i	9 n	9 n
a_i	27 n	22n
$\bar{a}_i$	15 n	14n
F_i	3 n	0
f_i	9 n - 9	9 n - 6
N_i	24 n	18n
n_i	21n - 15	24n -15
Total Mathematical Operations	117n -24	103n -21

Where n is the number of degrees-of-freedom of a manipulator

앞의 표에서 알 수 있듯이 N-E 방법은 joint 수에 선형으로 증가하나 L-E 방법은 4승으로 증가하므로 joint 수가 증가할수록 엄청난 계산이 필요함을 알 수 있다. 이에 따라 많은 경우 Coriolis항 또는 inertia 매트릭스의 off-diagonal terms을 무시함으로서 계산을 줄이나 이 방법은 느린 속도로 동작할 때는 가능하나 robot arm의 속도가 빨라지며 많은 오차를 내게 한다.

V. Robot Arm의 제어 방법

Robot arm의 제어 문제는 robot arm의 수학적인 모델을 바탕으로 하여 바라는 시스템 성능(주로 robot arm의 joint나 hand의 위치)를 얻기 위해 제어 방법을 설계하는 것이다. Robot arm의 제어는 바라는 joint 제도(결과적으로 hand의 제도)를 따르게끔 각 joint에 적절한 토오크나 힘을 가하는 것이나 앞에서 설명하였듯이 가장 어려운 문제는 상호 관계에 따른 복잡하고 결합된 비선형 dynamic equation 때문이다. 이러한 어려운 문제를 극복하기 위한 많은 제어 방법이 발표되었는데 이 중 주목할만한 제어 방법에 대해 설명하고자 한다.

초기의 제어 방법중 하나는 RMRC[6] (resolved motion rate control)이다. 이 방식은 kinematics를 이용한 것으로 robot hand를 바라는 위치로 움직이기 위해 적절한 joint 각 속도를 inverse jacobian 매트릭스를 이용하여 구하는 것이다. 이 방법은 open-loop 형태이며 작은 범위에서 동작될 때는 좋으나 매 sampling마다 inverse 매트릭스를 구하여야 하며 경우에 따라 생기는 singulavity 문제가 이 방법의 결점으로 지적되고 있다.

CMAC(cerebellar model articulate control)[7]은 table look-up에 의한 제어 방법이며 인간의 신체에서 볼 수 있는 반응과 비슷한 형태이다. 즉 reference 입력(경험)과 feedback(감각기관에 의한 보상)을 합하여 입력 벡타를 구한 후 이로부터 적절한 출력 값이 저장되어 있는 기억소자의 극소(두뇌)를 찾아 반응하는 것이다.

그러나 robot arm이 무한개의 궤도를 움직임을 생각할 때 기억 장치의 크기나 interpolation에 따른 정확도등이 문제점이다. 또 다른 제어 방법으로 computed torque 방법[8]이 있으며 이 방법은 원하는 궤도를 따르기 위해 필요한 입력 토오크를 robot arm의 수학적인 모델로 부터 직접 계산하는 것이다. 이때 필요한 데이타는 joint의 원하는 위치, 속도(혹은 가

속도)이며 모델의 부정확 또한 변수의 계속적인 변화 때문에 corrected 토오크의 계산을 위해 주로 classic한 feedback 제어 방법이 이용되고 있다. 계산 시간을 단축하기 위해 간소화한 모델(주로 결합부분을 무시함)이 주로 사용되며 이의 문제점은 robot arm이 고속으로 움직일 때의 modeling의 심한 오차이다. 이 점을 극복하기 위해 N-E방법으로의 모델을 사용하는 방법[9]이 제안되었다.

Near-minimum 제어 방법[10]은 robot arm의 비선형 모델을 선형 모델로 변환시킨 다음 최적 제어에 의한 값과 근사한 제어 입력을 구하는 것이다. 이 방법은 4개 이상의 joint를 가진 robot arm에 대해선 너무 복잡하며 외부 물체의 무게에 의한 영향등을 무시하였다.

비선형 피이드백 제어방법[11,12]은 복잡한 비선형 부분을 피이드 백으로 상쇄시킨 다음 나머지 선형 모델에 대해 PD제어를 적용시키는 것이다. 상쇄시키는 방법으로 비선형 부분의 정확한 계산에 의한 것과 MRAC(model reference adaptive control)에 의한 방법등이 있다.

또한 robot arm의 속도와 외부 물체에 대한 영향을 보완한 적응 제어방법이 응용되고 있다. 이 방법[13]중 하나는 각 joint-pair를 2次 선형 모델로 가정하고 이에 맞는 ARMA(auto regressive moving average) 모델의 parameter를 매 sampling때마다 입력과 출력의 값을 이용하여 update함으로서 다음 sampling 동안의 출력을 계산하는 방법이며 이 방법은 처음 parameter identification의 단계에서 큰 tracking error를 볼 수 있다.

또 다른 방법[14]은 N-E방법을 이용하여 nominal 토오크 값을 구한 다음 나머지 작은 perturbation을 parameter identification에 의한 방법으로 보완하는 것이다. Feedforward compensator와 feedback compensator를 독립적으로 계산이 가능하며 computed 토오크 제어 방법과 적응 제어 방법을 동시에 이용한 방법이다.

Ⅵ. 결 론

지금까지 로보트 시스템의 제어 구조와 이에 따른 제어 방법등을 소개하였다.

앞에서 언급하였듯이 로보트 시스템의 제어 방식은 적응 제어 방식이 될 것이며 더 나아가 AI(artificial intelligence)와 sensory 시스템을 함께 효과적으로 이용해 주위 환경에 적응함으로서 완전한 자동화를 이룰 수 있다.

參 考 文 獻

[1] Ernst, H.A., *MH-1, A Computer-Operated Mechnical Hand.* 1962 Spring Joint Computer Conference AFIPS Proceedings, pp. 39-51, San Francisco, May 1-3, 1962.

[2] Denavit, J., Hartenberg R.S., "A kinematic notation for lower-pair mechanisms based on matrices," *Journal of Applied Mechanics,* pp. 215-221, June 1955.

[3] Lee, C. S. G., "Robot arm kinematics, dynamics, and control," *IEEE Computer,* vol. 15, no. 2, pp. 62-80, Dec. 1982.

[4] Lewis, R.A., *Autonomous Manipulation on a Robot: Summary of Manipulator Software Functions.* Technical Memorandum 33-679, Jet Propulsion Laboratory, March 1974.

[5] Luh, J. Y. S., Walker, M. W., Paul, R., "On-line computational schemes for mechanical manipulators," *Transactions of the ASME, Journal of Dynamic Systems, Measurement and Control,* vol. 102, pp. 67-76, June 1980.

[6] Whitney, D. E., "Resolved motion rate control of manipulators and human prostheses," *IEEE Transactions on Man-Machine System,* vol. MMS-10, no. 2, pp. 47-53, June 1969.

[7] Albus, J. S., "A new approach to manipulator control: the cerebellar model articulation controller (CMAC)," *Transactions of the ASME, Journal of Dynamic Systems, Measurement and Control,* vol. 97, pp. 220-227, September 1975.

[8] Markiewicz. B. R., *Analysis of the Computer Torque Drive Method and Comparison with Conventional Position Servo for a Computer-Controlled Manipulator.* Technical Memorandum 33-601, Jet Propulsion Laboratory, March 1973.

[9] Lee, C. S. G., Chung, M. J., Turney, J. L., Mudge, T. N., *On the Control of Mechnical Manipulators.* Proceedings of the Sixth IFAC Conference in Estimation and Parameter Identification, Washington D.C. pp. 1454-1459, June 1982.

[10] Kahn, M.E., Roth B., "The near-mini-

mum time control of open-loop articulated kinematic chains," *Transactions of the ASME, Journal of Dynamic Systems, Measurement of Control,* vol. 93, pp. 164-172, September 1971.

[11] Hemani, H., Camana, P. C., "Nonlinear feedback in simple locomotion system," *IEEE Transactions on Automatic Control,* vol. AC-19, pp. 855-860, 1976.

[12] Horowitz, R., Tomizuka, M., *An Adaptive Control Scheme for Mechanical Manipulator-Compensation of Nonlinearity and Decoupling Control.* The Dynamic System and Control Division of the ASME Winter Annual Meeting at Chicago, 16-21 November 1980.

[13] Koivo, A. J., Guo, T. H., "Adaptive linear controller for Robotic manipulators," *IEEE Transactions on Automatic Control,* vol. AC-28, no. 2, pp. 162-171, February, 1983.

[14] Lee, C.S.G., Chung, M.J., *An Adaptive Control Strategy for Computer-Based Manipulators.* Proceedings of the 21st IEEE Conference on Decision and Control, Orlando, Florida, pp. 95-100, Dec. 8-10, 1982. *

♣ 用 語 解 說 ♣

Sequential File의 長點

- 하나의 record를 access한 다음에는 저절로 그 다음에 위치해 있는 record를 access 할 수 있도록 된다.

그러나 短點은 :

- 旣存의 file에 새로운 record를 첨가하려면 매우 시간의 낭비가 있게 된다. 그림에서 file A에 record 2 다음에 record 3을 첨가하려면 record 4부터 file A의 마지막 record까지 하나씩 뒤로 밀려나야 된다. 이것은 file A의 부피가 크면 클수록 더욱 대단한 시간의 소모를 가져온다.
- 기존의 file에 있는 어떤 record를 삭제하려면 매우 시간의 낭비가 있게 된다. 그림에서 file 의 record 2를 삭제하려면 record 4 이후에 나오는 record들 모두를 하나씩 앞으로 당겨야 하므로 역시 file의 부피가 클 때는 대단한 시간의 소모가 야기된다.
- Sequential file에서는 record를 access할 때에도 순차적으로 해야 한다. 그리하여 file 內에서 record 시작 주소로부터 가까이 저장되어 있는 record를 access할 때는 access time이 많이 걸리지 않으나 예를 들어 file A의 마지막 record인 record n을 access 하려면 record 1을 access하는 것보다 n배의 시간이 소모된다. 여기서도 file A에 속하는 record의 수효 n에 따라서 record 시작 주소로부터 멀리 떨어져서 저장되어 있는 record는 access time이 길어지게 된다.

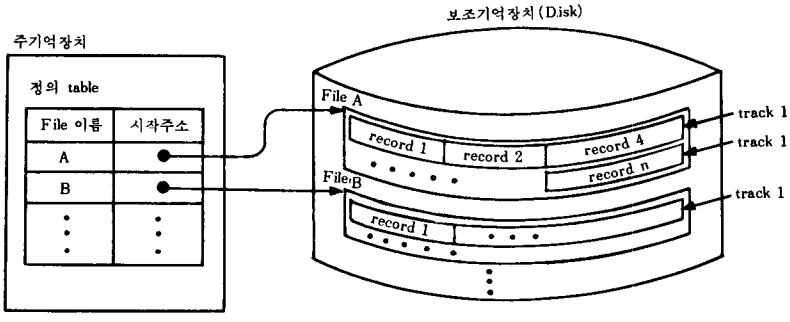

그림. Sequential file