'90 韓國自動制御學術會議論文集 1990. 10.26~27

直接驅動型로봇의 加減速時間 短縮에 관한 硏究

林桂榮 李光男 高光一

金星産電硏究所

Analysis to Reduce the Acceleration Time and Deceleration Time of Direct Drive Robot

Kye Young Lim, Gwang Nam Lee, Kwang Il Koh

Goldstar Industrial Systems Co., Ltd

ABSTRACT

This paper represents a control method of improving the performance of direct drive robot. The direct transfer of torque and rotational speed of direct drive motor to the robot body without reduction gear makes the robot speed fast. However, the variation of inertia matrix and low friction cause the control difficult, and one more effort must be in the reducing the acceleration and deceleration time to reduce the cycle time.

To fasten the cycle time and to improve the robustness of robot, one control method is developed, and implemented in the Goldstar DD robot. This method does not need to change the conventional PI type control structure, but one additional compensational control law is required. The control law can be obtained via inverse dynamic model of robot, and inverse model of existing control loop. The effects of this control law are shown in this paper.

1. 序論

現在 産業用 로봇 가운데 組立用으로 널리 쓰이는 로봇은 4축 水平多關節 로봇으로서, 本硏究所에서는 直流모터(DC MOTOR)를 구동원(Actuator)으로 한 4축 水平多關節로봇을 이미 개발한바 있다[5-6]. 組立用으로 쓰이는 로봇이 더 빠르게 組立作業을 행할 수 있다면 이는 조립용 로봇으로서 향상된 로봇이라 할 것이다. 이러한 必要에 의해 개발된 것이 直接驅動型로봇(Direct Drive Robot)이며, 本硏究所에서도 1989年 直接驅動型로봇(Direct Drive Robot)을 개발하였다[7]. 旣存의 組立用 로봇은 모터축(Motor Shaft)과 매니퓰레이터축(Manipulator Shaft)사이에 감속기(Reduction Gear)를 부착시켜 回轉數의 傳達(Transfer of Rotational)을 줄이고 토크의 傳達(Transfer of Torque)을 올리게 한 로봇이다. 이에 반해 直接驅動型로봇(Direct Drive Robot)은 감속기 없이 모터 축을 매니퓰레이터 축에 直接 連結시켜 傳達回轉數와 傳達토크를 增減없이 그대로 傳達시킨 로봇이다[3,8]. 이러한 理由로 直接驅動型로봇은 作業時間이 短縮되는 등의 長點을 지니게 되나, 驅動時 慣性(Inertia)의 變動 폭이 커지는 短點을 안게 됨을 把握하였다.

本硏究所에서 개발한 Goldstar Direct Drive Robot을 最大動作範圍(A축, B축 : 0° →280°)로 驅動시킬 때 1.09초 所要되고, 旣存의 감속기가 부착된 로봇으로 驅動시킬 경우 1.5초 所要되는 바, 作業時間 短縮의 側面에서 볼때 直接驅動型로봇이 有利하다. 한 가지 看過할 수 없는 점은 Goldstar Direct Drive Robot을 最大動作範圍로 驅動시킬때 所要되는 1.09초 중 0.8초 (73%)가 加減速時間으로 所要되며, 一定範圍 (A축 : 0° →144° , B축 : 0° →280°)以下로 驅動시킬 때 所要되는 0.8초가 모두 加減速時間으로 所要된다는 점이다. 이에 Goldstar Direct Drive Robot의 作業時間을 좌우하는 加減速時間을 短縮할 必要에 이르렀다. 또한 直接驅動型로봇의 短點으로 把握된 慣性의 큰 變動을 克服해야 할 必要에 이르렀다.

이에 本論文에서는 直接驅動型로봇의 短點으로 把握된 慣性의 큰 變動을 克服하고 加減速時間을 短縮하기 위하여, 종래의 PI 제어기에 動的補償構造(Structure of Dynamic Compensation)를 添加한 制御構造를 제시한다. 이 制御構造의 特徵으로는 從來의 PI 제어기 전류 Loop를 수정하지 아니하고도, 慣性 等의 變動과 加減速時間 短縮에 對應할 수있는 動的補償(Dynamic Compensation)을 이룬다는 점을 들 수 있다. 本論文에서 제시한 動的補償構造는 逆動的모델(Inverse Dynamic Model)과 從來의 PI 제어기 전류 Loop의 逆모델(Inverse Model)에 基礎한 制御構造이다.

本論文에서는 直接驅動型로봇의 短點으로 把握된 慣性變動에 대해 從來의 PI 제어기와 本動的補償構造로 各各 시뮬레이션을 實行하고, 各各의 位置誤差(Position Error)를 比較하기로 한다. 같은 方法으로 直接驅動型로봇의 加速時間을 短縮(0.4초→0.1초)한 경우에 대해서도 各各의 位置誤差를 比較하기로 하고, 加速時間을 短縮했을 경우 Cycle Time 側面에서의 效果를 밝히기로 한다. 또한 本論文에서는 逆動的모델에 필요한 速度를 $\dot{\theta}_d$ (軌道生成에서 구한 速度)와 ω_d(位置 PI 제어기의 出力)로 하여 各各 시뮬레이션을 實行하고, 逆動的모델(Inverse Dynamic Model)에 必要한 速度는 ω_d로 하여야 함을 밝히기로 한다. 한편 動的補償을 위한 計算을 PI 제어기 Speed Loop의 Sampling Time (Goldstar Drive Robot의 경우, 1.6ms)마다 實行한다는 것은 제어기에 計算量의 負擔을 주는 바, 動的補償(Dynamics Compensation)의 效果를 維持하면서 動的補償構造의 Sampling Time을 느리게할 수 있다면 그 實益이 있다 하겠다. 시뮬레이션을 통해 本動的補償構造의 Sampling Time 上限値를 推定하기로 한다.

2. Goldstar Direct Drive Robot의 記述 및 問題提起

本硏究所에서 개발한 直接驅動型로봇(Direct Drive Robot)에 대해 簡略하게 記述하고, 加減速時間 短縮과 慣性變動에 따른 問題를 提起한다.

2.1 Goldstar Direct Drive Robot

2.1.1 機構學(Kinematics)

Goldstar Direct Drive Robot의 外觀은 그림 2-1과 같다. 正機構學(Forward Kinematics)[1-2, 9-10]의 側面에서 Goldstar Direct Drive Robot의 擧動을 記述하면 [5-7],

$$ {}^{o}_{4}T = {}^{o}_{1}T\ {}^{1}_{2}T\ {}^{2}_{3}T\ {}^{3}_{4}T = \left[\begin{array}{ccc|c} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \quad (2.1) $$

과 같다. 여기서,

$n_x = -\sin(\theta_1 + \theta_2 + \theta_4)$
$n_y = \cos(\theta_1 + \theta_2 + \theta_4)$
$n_z = 0$
$o_x = -\cos(\theta_1 + \theta_2 + \theta_4)$
$o_y = -\sin(\theta_1 + \theta_2 + \theta_4)$
$o_z = 0$
$p_x = a_1\cos\theta_1 + a_2\cos(\theta_1 + \theta_2)$
$p_y = a_1\sin\theta_1 + a_2\sin(\theta_1 + \theta_2)$
$p_z = -d_2 - d_3$

와 같다. ${}^{i}_{j}T$는 링크(Link) i에서 링크 j로의 變換을 나타내는 變換行列(Transformation Matrix)이며, a_1, a_2, d_2, d_4, θ_1, θ_2, θ_4는 그림 2-1에 나타내었다. 逆機構學(Inverse Kinematics)[1-2, 9-10]의 側面에서 Goldstar Direct Drive Robot의 擧動을 記述하면[5-7],

$$ \theta_2 = \tan^{-1}\left[\ \{\ \pm\ (1-b)^{1/2}\ \}\ /\ b\ \right] \quad (2.2) $$

[단, $b = (p_x^2 + p_y^2 - a_2^2 - a_1^2)/(2a_1a_2)$]

$$ \theta_1 = \tan^{-1}\{p_x/p_y\} + \tan^{-1}\{a_1\sin\theta_2/(a_2 + a_1\cos\theta_2)\} - \theta_2 \quad (2.3) $$

$$ d_3 = -d_2 - p_z \quad (2.4) $$

$$ \theta_4 = \tan^{-1}(n_x/o_x) - \theta_1 - \theta_2 \quad (2.5) $$

와 같다. 逆機構學(Inverse Kinematics)식(2.2-2.5)의 計算은 Goldstar Direct Drive Robot의 Main Controller에서 이루어진다.

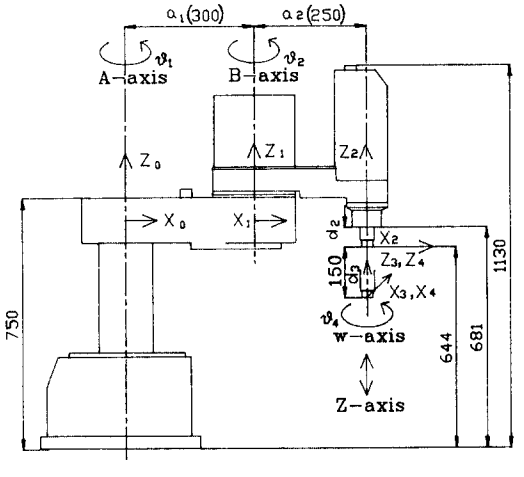

그림2-1. Goldstar Direct Drive Robot의 外觀

2.1.2 動力學(Dynamics)

Lagrange-Euler Equation[1-2, 9-10]으로 Goldstar Direct Drive Robot의 動的擧動을 記述하면[4-7, 11],

$$ \tau = D(q)\cdot\ddot{q} + H(q,\dot{q}) + G(q) \quad (2.6) $$

여기서,

τ : Joint에 가해진 Torque (또는 Force)를 나타내는 4 x 1 벡터, $(\tau_1,\tau_2,\tau_3,\tau_4)^t$.

q : Joint Position을 나타내는 4 x 1 벡터, $(q_1,q_2,q_3,q_4)^t$. 回轉하는 조인트는 θ로 표시하였고, 直線으로 움직이는 조인트는 d로 표시하였다.

$\dot{q}$: Joint velocity를 나타내는 4 x 1 벡터, $(\dot{q}_1,\dot{q}_2,\dot{q}_3,\dot{q}_4)^t$.

$\ddot{q}$: Joint acceleration을 나타내는 4 x 1 벡터, $(\ddot{q}_1,\ddot{q}_2,\ddot{q}_3,\ddot{q}_4)^t$.

$D(q)$: 慣性(Inertia)을 나타내는 4 x 4 行列, $[D_{ik}]$ (i,k=1, · · ·,4).

$H(q,\dot{q})$: 코리올리와 遠心力을 나타내는 4 x 1 벡터, $(H_1,H_2,H_3,H_4)^t$

$G(q)$: 重力을 나타내는 4 x 1 벡터, $(G_1,G_2,G_3,G_4)^t$.

式記述의 簡便을 위해 $\cos\theta_1$,$\cos\theta_2$,$\sin\theta_1$,$\sin\theta_2$를 各各 c_1,c_2,s_1,s_2로 記述하였다. 식(2.6)의 $D(q)$, $H(q,\dot{q})$, $G(q)$를 각각의 Element로 기술하면,

$D_{11} = m_1 * (K_{1zz}^2 + 2a_1x_1 + a_1^2) + m_2 * (K_{2zz}^2 + 2a_2\bar{x}_2 + 2a_1\bar{x}_2c_2 + 2a_1a_2c_2 + a_1^2 + a_2^2) + m_3 * (K_{3zz}^2 - 2a_2\bar{y}_3 - 2a_1\bar{y}_3c_2 + 2a_1a_2c_2 + a_1^2 + a_2^2) + m_4 * (K_{4zz}^2 + 2a_1a_2c_2 + a_1^2 + a_2^2)$

$D_{22} = + m_2 * (K_{2zz}^2 + 2a_2\bar{x}_2 + a_2^2) + m_3 * (K_{3zz}^2 - 2a_2\bar{y}_3 + a_2^2) + m_4*(K_{4zz}^2 + a_2^2)$

$D_{33} = m_3 + m_4$

$D_{44} = m_4K_{4zz}^2$

$D_{12} = D_{21} = m_2 * (K_{2zz}^2 + 2a_2\bar{x}_2 + a_1\bar{x}_2c_2 + a_1a_2c_2 + a_2^2) + m_3 * (K_{3zz}^2 - 2\bar{a}_2y_2 - a\bar{y}_3c_2 + a_1a_2c_2 + a_2^2) + m_4*(K_{4zz}^2 + c_2a_1a_2 + a_2^2)$

$D_{13} = D_{31} = 0$

$D_{14} = D_{41} = m_4\ K_{4zz}^2$

$D_{23} = D_{32} = 0$

$D_{24} = D_{42} = m_4K_{4zz}^2$

$D_{34} = D_{43} = 0$

$H_1 = 2 * \{ m_2 * (-a_1\bar{x}_2s_2 - a_1a_2s_2) + m_3 * (a_1y_3s_2 - a_1a_2s_2) + m_4 * (-a_1a_2s_2) \} * \theta_1\theta_2 + \{-m_2a_1\bar{x}_2s_2 - m_2a_1a_2s_2 + m_3a_1\bar{y}_3s_2 - (m_3 + m_4) * a_1a_2s_2 \} * \theta_2^2$

$H_2 = (m_2a_1\bar{x}_2s_2 + m_2a_1a_2s_2 - m_3a_1\bar{y}_3s_2 + m_3a_1a_2s_3 + m_4a_1a_2s_2) * \theta_1^2$

$H_3 = 0$

$H_4 = 0$

$G_1 = 0$

$G_2 = 0$

$G_3 = -g * (m_3 + m_4)$, g : 重力 加速度

$G_4 = 0$

와 같다. 여기서,

K_{jxx},K_{jyy},K_{jzz} : 링크 j의 座標系에서 바라본 Gyration 半徑.

m_j : 링크 j의 質量(Mass).

$\bar{r}_j = (\bar{x}_j,\bar{y}_j,\bar{z}_j,1)^t$: 링크 j의 座標系의 原點으로부터 Link j의 質量中心(Mass Center)까지의 位置 Vector.

2.1.3 軌道 生成(Trajectory Generation)

Goldstar Direct Drive Robot의 구동방식은 4축 동시구동으로, 동시구동의 기준이 되는 축의 軌道生成(Trajectory Generation)[1-2]을 記述하면[5-7],

$0 \le t \le 2$ tacc에서는

$h = t / (2tacc)$

$dum = \{\theta(t_f) - \theta(t_i)\} / T_1 * h$

$\theta = dum * tacc * (2 - h) * h^2$ (2.7)

$\dot{\theta} = dum * 2 * (1.5 - h) * h$ (2.8)

$\ddot{\theta} = dum * (1 - h) * 3 / tacc$ (2.9)

$2tacc \le t \le T_1$에서는

$h = (t - tacc) / T_1$

$\theta = \{\theta(t_f) - \theta(t_i)\} * h$ (2.10)

$\dot{\theta} = \{\theta(t_f) - \theta(t_i)\} / T_1$ (2.11)

$\ddot{\theta} = 0$ (2.12)

$T_1 \le t \le T_1 + 2tacc$에서는

$h = (t - T_1) / (2 * tacc)$

$dum = \{\theta(t_f) - \theta(t_i)\} / T_1 * h^2$

$\theta = -\{\theta(t_f) - \theta(t_i)\} / T_1 * tacc * [\{(2-h) * h^2 - 2\} * h + 1] + \{\theta(t_f) - \theta(t_i)\}$ (2.13)

$\dot{\theta} = -\{\theta(t_f) - \theta(t_i)\} / T_1 * \{(1.5-h) * 2h^2 - 1\}$ (2.14)

$\ddot{\theta} = -\{\theta(t_f) - \theta(t_i)\} / T_1 * (1 - h) * 3h / tacc$ (2.15)

가 된다. 단,

2tacc : 加速時間(Acceleration Time)

$\theta(t_i)$: 初期位置

$\theta(t_f)$: 最終位置

T : 等速이 끝나고 減速이 始作되는 순간의 時間.

軌道生成(Trajectory Generation)은 Goldstar Direct Drive Robot의 Main Controller에서 이루어진다.

2.1.4 制御構造(Structure of Control)

그림 2-2와 같이 Goldstar Direct Drive Robot의 制御構造는 Main Controller, Axis Controller, 그리고 Servo Driver로 이루어져 있다. Main Controller에서는 直線, 원호, 조인트 보간등의 Trajectory Plan을 수행하며, Teaching 기능을 수행하며, User Program을 해석하며, 로봇 動作을 制御하며, Robot Controller의 全體狀態를 監視하고, 缺陷을 感知하여 誤動作을 防止하는 等의 기능을 갖는다. Axis Controller에서는 4축 동시위치 제어를 하며, 로봇 각 관절의 원점을 조정하는등의 기능을 갖는다.Servo Controller에서는 속도제어와 전류제어를 하는 등의 기능을 갖는다.

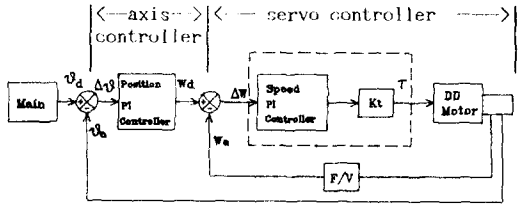

그림 2-2. Goldstar Direct Drive Robot의 制御構造

2.2 問題 提起

直接驅動型로봇(Direct Drive Robot)의 長點 중 하나로 作業時間의 短縮을 든다[3,7-8]. Goldstar Direct Drive Robot을 最大動作範圍(A-axis, B-axis : 0° → 280°)로 驅動시킬때 所要되는 1.09초 중 0.8초(73%)가 可減速時間으로 所要되고, 一定範圍(A축 : 0° → 144° , B축 : 0° → 280°)以下로 驅動시킬때 所要되는 0.8초가 모두 加減速時間으로 所要됨을 把握하였다. 이에 Goldstar Direct Drive Robot의 作業時間을 좌우하는 加減速時間을 短縮할 必要에 이르렀다. 또한 直接驅動型로봇의 短點으로 把握된 慣性(Inertia)의 큰 變動을 克服해야 할 必要에 이르렀다.

이에 本論文에서는 從來의 PI 제어기 전류 Loop를 수정하지 아니하고, 加減速時間 短縮과 慣性의 變動에 對應할 수 있는 動的補償構造(Structure of Dynamic Compensation)를 提示한다.

3. 動的補償構造(Structure of Dynamic Compensation)

3.1 本補償構造의 必要性

Goldstar Direct Drive Robot의 制御構造(Structure of Control)는 그림 2-2와 같이 PI 제어기로 이루어져 있다. 本論文의 2.2 問題提起에서 提示한 것처럼, 加減速時間(acceleration and deceleration time) 短縮과 慣性(Inertia)의 變動에 對應할 수 있는 動的補償構造(Structure of Dynamic Compensation)가 必要하게 되었고, 그림 3-1에 이 補償構造를 提示한다.

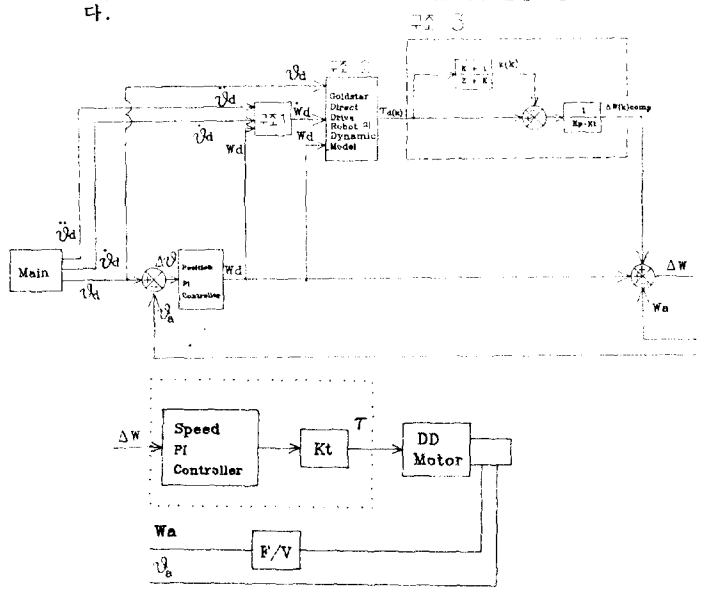

그림 3-1. 動的補償構造(Structure of Dynamic Compensation)

3.2 補償構造(Structure of Compensation)

本論文의 補償構造는 逆動力學(Inverse Dynamics)의 槪念을 利用하여 目標速度(Desired Speed)의 보상값을 구하기 위한 것으로 그림 3-1과 같이 크게 세부분(構造 1,2,3)으로 이루어져 있다.

構造1은,軌道生成(Trajectory Generation)의 式(2.8, 2.9, 2.11, 2.12, 2.14, 2.15)에서 구한 $\dot{\theta}_d$(目標速度)와 $\ddot{\theta}_d$(目標加速度), 그리고 Wa(그림 2-2의 위치 제어기의 出力)를Input으로 하여 Wd를 구한다.

構造 2는 逆動力學을 이용한 것으로서, Goldstar Direct Drive Robot의 모델링 (Modeling) 式(2.6)을 根據로하여 動的補償(Dynamic Compensation)의 基準이 되는 Torque (τ_d)를 구한다. Input은, θ_d(軌道生成의 式 2.7, 2.10, 2.13 에서 구한 目標位置)와 Wa(그림 2-2의 위치 제어기의 출력), 그리고

W_d(構造 1에서 구한 값)로 이루어져 있다.

構造 3은 그림 2-2에서 빗금친 構造의 逆모델(Inverse Model)로서, 기준 토크(τ_d)를 Input으로 하여 속도 보상값(ΔW_{comp})을 구한다. 그림 2-2에서 빗금친 構造의 傳達函數(Transfer Function)를 구하면,

$$\frac{\tau_d(z)}{\Delta W_{comp}(z)} = \left(K_p + \frac{K_i}{Z - 1} \right) (K_t) \quad (3.1)$$

가 된다 逆(Inverse)을 취하면,

$$\frac{\Delta W_{comp}(z)}{\tau_d(z)} = \frac{1}{(K_t \cdot K_p)} \left(1 - \frac{K + 1}{Z + K} \right) \quad (3.2)$$

이 되며, $K = \frac{K_i}{K_p} - 1$ (3.3)

이다. 단,

τ_d : 構造 2에서 구한 토크로서, 補償의 基準이 되는 토크.

ΔW_{comp} : 속도 보상값

K_p : 속도 제어기의 Proportional Gain

K_i : 속도 제어기의 Integral Gain

K_t : 토크 상수

(식 3.2)를 根據로 한 構造 3은 그림 (3-2)과 같다.

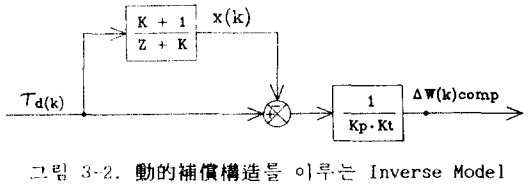

그림 3-2. 動的補償構造를 이루는 Inverse Model

構造 3을 Discrete Time Model로 記述하면,

$$X_{(k+1)} = -K \cdot X_{(k)} + (k + 1) \cdot \tau_{d(k)} \quad (3.4)$$

$$\Delta W_{(k)comp} = \frac{1}{K_p \cdot K_t} \cdot \left[-X_{(k)} + \tau_{d(k)} \right] \quad (3.5)$$

Stability 側面에서 식 (3.4)는 $|K| \leq 1$ 을 滿足해야 한다. 만일, $|K| > 1$인 경우에는 逆모델 $\left(\frac{K+1}{Z+K}\right)$ 대신에 近似逆모델(Approximated Inverse Model)을 쓰면 된다. 여기서 近似逆모델로는 DC gain이 1인 1차 필터 $\left(\frac{a+1}{z+a}, |a| \ll 1\right)$ 를 採擇하기로 한다.

4. 시뮬레이션(Simulation)

本論文에서는 Goldstar Direct Drive Robot을 플랜트(Plant)로 하여 시뮬레이션을 實行하였다. [5-7]. 本論文에서 提示한 動的補償構造의 逆動的모델(Inverse Dynamic Model)은 플랜트에 대해 모델오차(慣性: 30%, 質量: 10%, 質量中心: 10%)를 갖는 것으로 하였다. 直接驅動型 로봇의 短點으로 把握된 慣性變動에 대해 從來의 PI 제어기와 本論文에서 提示한 動的補償構造로 各各 시뮬레이션을 實行(A축 : 0° → 180° , B축 : 0° → 280°)하였으며, 각각의 位置誤差(Position Error)를 그림 4-1에 나타내었다. 慣性變動에 심한 條件을 주기 위해 가반중량(Payload)을 10Kg(Goldstar Direct Drive Robot의 최대가반중량: 5Kg)으로 하였고, 加速時間을 0.4초로 하였다. 從來의 PI 제어기로 시뮬레이션을 實行했을때의 最終位置誤差(Final Position Error)가 3.58 x 10^{-1}rad인 반면, 本動的補償構造로의 最終位置誤差는 2.63 x 10^{-4} rad임을 確認하였다.

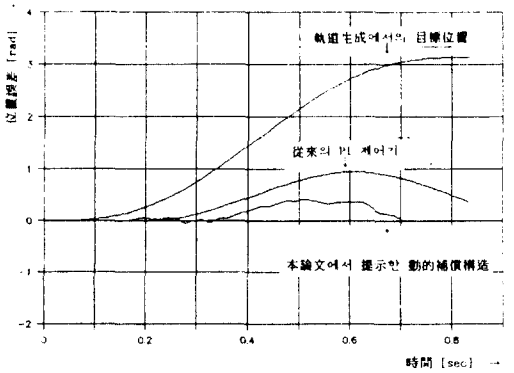

그림 4-1. 慣性變動에 대한 位置誤差

또한 直接驅動型로봇의 加速時間을 短縮(0.4초 → 0.1초)한 경우에 대해 시뮬레이션을 實行(A축 : 0° →180° ,B축 :0° → 280°)한 結果, Cycle Time을 35.8% 短縮(0.848초 → 0.544초)할 수 있었다. 加速時間을 0.1초로 하여 從來의 PI 제어기와 本動的補償構造로 各各 시뮬레션을 實行하였으며, 各各의 位置誤差를 그림 4-2에 나타내었다. 從來의 PI 제어기로 시뮬레이션을 實行했을때의 最終位置誤差가 1.18 x 10^{0}rad인 반면, 本動的補償構造로의 最終位置誤差는 3.41 x 10^{-4}rad임을 確認하였다.

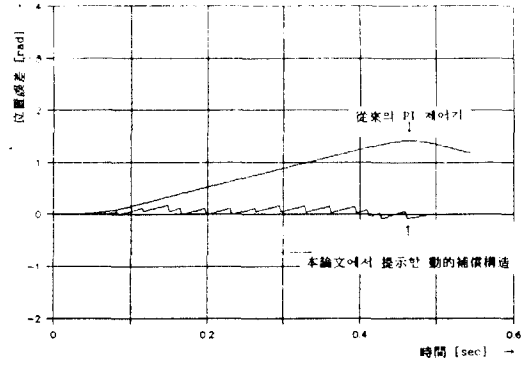

그림 4-2. 加速時間을 短縮한 경우에 대한 位置誤差

위의 加速時間을 短縮한 경우와 同一한 條件으로 시뮬레이션을 하되, 또한 本動的補償構造의 逆動的모델에 必要한 速度를 $\dot{\theta}_d$(軌道生成에서의 速度)와 W_d(位置 PI 제어기의 出力)로 하여 各各 시뮬레이션을 實行하였으며, 各各의 位置誤差를 그림 4-3에 나타내었다. 逆動的모델에 必要한 速度를 W_d(위치 제어기의 出力)로 하여야 함을 確認하였다.

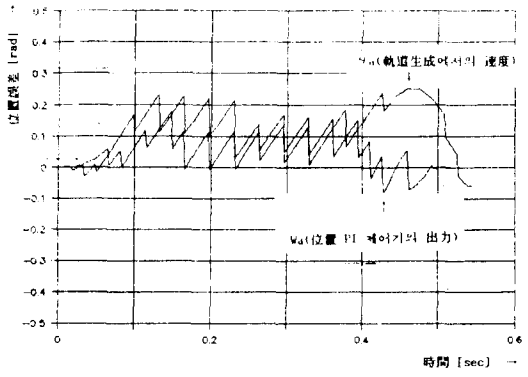

그림 4-3. 逆動的모델에 必要한 速度를 $\dot{\theta}_d$(軌道生成에서의 速度)와 Wa(位置 PI 제어기의 出力)로 하였을 때의 位置誤差.

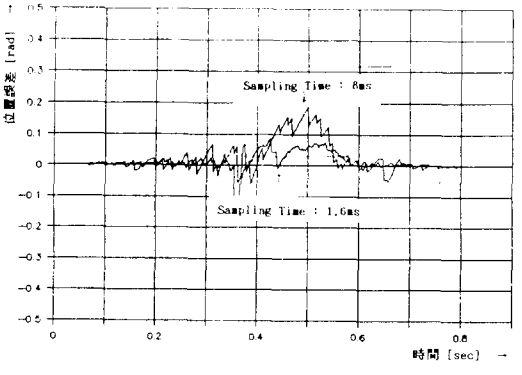

그림 4-4. 動的補償構造 Sampling Time을 8ms으로 하였을 때의 位置誤差.

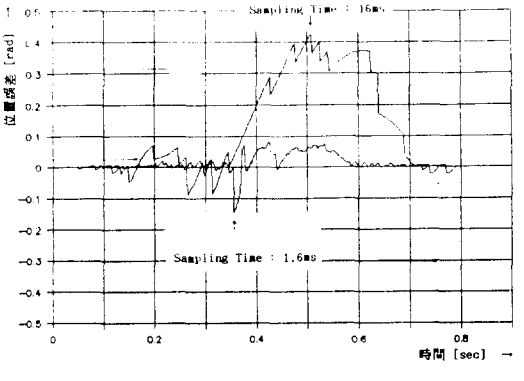

그림 4-5. 動的補償構造 Sampling Time을 16ms으로 하였을 때의 位置誤差.

한편 動的補償을 위한 計算을 PI 제어기 速度 Loop의 Sampling Time(Goldstar Direct Drive Robot의 경우, 1.6ms)마다 實行한다는 것은 제어기에 計算量의 負擔을 주는바, 動的補償의 效果를 維持하면서 動的補償構造의 Sampling Time을 느리게 할 수 있다면, 그 實益이 있다하겠다. 이에 시뮬레이션을 통하여 動的補償構造의 Sampling Time 上限値를 16ms으로 推定하였고, 이는 PI 제어기 速度 Loop Sampling Time의 10倍에 相當함을 把握하였다. 動的補償構造의 Sampling Time을 각각 1.6ms, 8ms, 16ms, 32ms로 하여 시뮬레이션을 實行하였고, 各各의 位置誤差를 그림 4-4, 4-5, 4-6에 나타내었다.

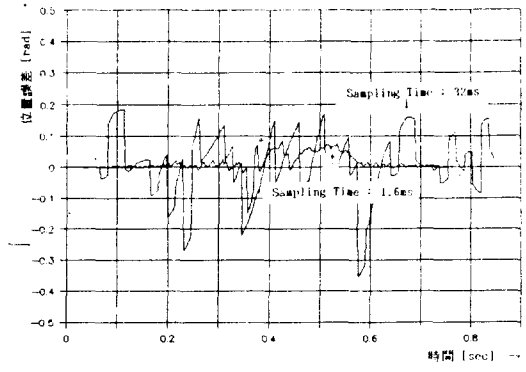

그림 4-6. 動的補償構造 Sampling Time을 32ms으로 하였을 때의 位置誤差.

5. 結論

本論文에서는 直接驅動型로봇(Direct Drive Robot)의 短點으로 把握된 慣性變動(Variation of Inertia)의 큰 폭을 克服하고 加減速時間을 短縮하기 위하여, 從來의 PI 제어기에 動的補償構造(Structure of Dynamic Compensation)를 添加한 制御 構造를 提示하였다. 이렇게 함으로써, 종래의 PI 제어기 전류 Loop를 수정하지 않고도 慣性 等의 變動과 加減速時間 短縮에 따른 動的補償(Dynamic Compensation)을 이룰수 있었다. 本論文에서 提示한 動的補償構造는 逆動的모델(Inverse Dynamic Model)과 從來의 PI 제어기 속도 Loop의 逆모델(Inverse Model)에 基礎한 制御構造이다.

直接驅動型로봇의 短點으로 把握된 慣性變動에 대해 從來의 PI제어기로 시뮬레이션을 實行(A軸 : 0° → 180°, B軸 : 0° → 280°)했을 때의 最終位置誤差(Final Position Error)는 3.58 x 10^{-1}rad으로 목표위치(3.14rad)의 11.4% 정도인 반면, 本動的補償構造로 시뮬레이션을 實行했을 때의 最終位置誤差는 2.63 x 10^{-4}rad으로 目標位置(3.14rad)의 0.00838% 정도임을 確認하였다.

直接驅動型로봇의 加速時間을 短縮(0.4초→0.1초)한 경우에 대해 시뮬레이션을 實行(A軸 : 0° → 180°, B軸 : 0° → 280°)한 結果, Cycle Time을 35.8%(0.848초→0.544초) 短縮할 수 있었다. 이 경우에 대해 從來의 PI제어기로 시뮬레이션을 實行(A軸 : 0° → 180°, B軸 : 0° → 280°)했을 때의 最終位置誤差(Final Position Error)는 1.18rad으로 目標位置(3.14rad)의 37.6% 정도인 반면, 本動的補償構造로 시뮬레이션을 實行했을 때의 最終位置誤差는 3.41 x 10^{-4}rad으로 目標位置(3.14rad)의 0.0109% 정도임을 確認하였다.

逆動的모델(Inverse Dynamic Model)에 必要한 速度를 $\dot{\theta}_d$ (軌道生成에서의 速度)와 Wa(위치 PI 제어기의 出力)로 하여 各各 시뮬레이션을 實行하였고, 逆動的