

A Calibration Method of Human Arm or Leg for Dynamic Animation and Haptic Rendering

Takeshi Ohtsuki, Ritsu Shikata and Hiroshi Noborio

Graduate School of Engineering Osaka Electro-Communication University

Hatsu-Cho 18-8, Neyagawa, Osaka 572-8530, Japan

nobori@noblab.osakac.ac.jp., <http://www.noblab.jp/top/>

Abstract

The purpose of this research is to calibrate sets of kinematics and dynamic parameters of human arm from its captured points via many behaviors. If both sets are successively identified, we can easily generate dynamic animation of human such as handling, pitching, walking, running and so on. In addition, we can straightforwardly touch a virtual arm or leg while feeling reactive force/moment by a haptic rendering (tactile feedback display). In robotics, many types of algorithms have been proposed for calibrating kinematics and dynamic parameters of robotic manipulator. They are meaningless for calibrating a human arm as follows: (1) we do not know any control scheme and feedback gain. (2) We do not know any torque of each actuator. (3) We do not know any set of good initial parameters of kinematics and dynamics. To overcome these, we propose a probabilistic algorithm to calibrate sets of kinematics and dynamic parameters of a human arm from its successive captured points. The calibration algorithm based on GA (Genetic Algorithm) does not require any set of good initial set of kinematics and dynamic parameters.

1 Introduction

Digital human is a new research field to understand movable mechanism of human body, to control a mechanical humanoid in a real world, and to operate an animated humanoid in a virtual world. In this paper, we propose a probabilistic algorithm to calibrate sets of kinematics and dynamic parameters of human arm or leg successively.

First of all, a robot needs two kinds of parameters. One is geometric parameters, which define homogeneous transformations between successive links. The kinematics parameters are estimated by some method as [1]. Another is dynamic parameters including the maximum of static friction, dynamic friction and viscous damping friction of each joint, and the mass, cen-

ter of mass, inertia tensor of each link. The dynamic parameters are calibrated from their initial values including errors by a few special motions or a lot of general motions. Each motion consists of joint angle, velocity, acceleration, and torque [1]~[7].

On the other hand, a human has bones and muscles. Their functions and roles of have been deeply analyzed in the medical field. After calibrating models of bones and muscles completely, we can perfectly generate human behaviors with many kinds of forces and moments [8],[9]. However, the modeling is difficult and time consuming. To avoid this, we capture many kinds of human behaviors and then construct kinematics and dynamic model of human arm and leg. In this paper, we calibrate kinematics and dynamics of human arm without knowing each role of their bones and muscles. After installing them in PC, we can enjoy dynamic animation of human body via visual and tactile information.

The differences between human arm and robot arm are as follows: (1) We do not know any control scheme and feedback gain. (2) We do not know any torque of each actuator. (3) We do not know any set of good initial parameters of kinematics and dynamics. To overcome these drawbacks, we propose a probabilistic algorithm for calibrating kinematics and dynamic parameters of human arm or leg without knowing control scheme and feedback gain, joint torques, and initial parameters. The algorithm calibrates them by a set of sequences of landmark positions around a human arm. Furthermore, our algorithm is based on GA. In general, GA needs not any initial set of calibrated parameters.

In this paper, section 2 describes human and robot arms and their motion capture device. Section 3 explains robot kinematics by a modified Denavit-Hartenberg method. Then, in section 4, we explain the genetic algorithm for calibrating kinematics and dynamics. Section 5 gives robot dynamics by Newton

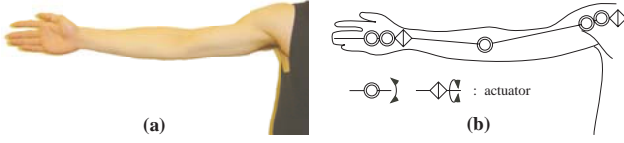


Figure 1 : (a) Human arm. (b) Its link structure.

Euler equations. Section 6 illustrates our calibration method. Section 7 describes how to make dynamic animation by Runge-Kutta and Euler methods based on the calibrated dynamics. Section 8 evaluates experimental results. Finally in section 9, we will give a few conclusions.

2 Human and Robot Arms and Their Sensing Devices

In this section, we explain a human arm, a robot arm, and motion capture device.

2.1 Human and Robot Arms

In this paper, we model a human arm or leg without knowing each role of bones and muscles (Fig.1). In our approach, we calibrate kinematics and dynamic parameters of human arm or leg successively from many captured data around the body. Especially in kinematics calibration, we do not exactly know how to connect many joints at shoulder, elbow and wrist of a human arm. For this reason, we calibrate many parameters in a modified Denavit-Hartenberg method [1],[6],[10]. In this research, we calibrate kinematics and dynamic parameters of a direct-drive manipulator with 2 D.O.F without seeing any control scheme and feedback gain, any torque of each actuator, any initial values (Fig.2).

The kinematics parameters of DD arm are provided by Shinmeiwa Co. In addition, a better dynamic parameters are completely calibrated by our smart approach mixing two classic calibration algorithms [7]. Unfortunately, since the approach completely needs precise kinematics, joint angles measured in motor encoders, and motor torques. In this paper, we propose a probabilistic approach to calibrate kinematics and dynamic parameters from only captured sequences of landmark positions around the DD arm.

2.2 Sensing Device for Motion Capture

In order to obtain an arbitrary position around a robotic manipulator in the real time, we use an acceleration scales (PCB Piezotronics co.) located on the position (Fig.3). First of all, the acceleration scales capture X, Y, and Z accelerations at an arbitrary position in the sampling time 32kHz. Secondly, we integrate a sequence of accelerations two times to obtain

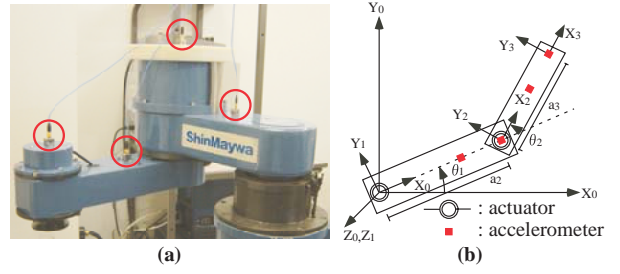


Figure 2 : (a) DD arm and its acceleration scales. (b) The link structure.

a sequence of positions. By processing four successive accelerations $\ddot{x}^t, \dots, \ddot{x}^{t+3}$ through some low-pass filter (the moving average whose number is four), we can remove some noises. Secondly, by integrating an acceleration along X-axis by $\dot{x}^{i+1} = \dot{x}^i + \ddot{x}^i \Delta t$, we calculate a velocity along X-axis, and then by integrating the velocity by $x^{i+1} = x^i + \dot{x}^i \Delta t$, we get a position along X-axis. By the same way, we calculate another position along Y-axis. Finally, by using two scales around each link, we measure joint angle of the link.



Figure 3 : A photo of an acceleration scales to measure three degrees-of-freedom accelerations along X, Y and Z axes.

3 Kinematics by Modified D-H Method

In order to represent kinematics of robot arm, we use Denavit-Hartenberg expression (D-H method). This defines coordinate of each link by four types of kinematics parameters (Fig.4).

θ_i : An angle from x_{i-1} axis to x_i axis around z_i axis in the clockwise direction.

d_i : A distance from x_{i-1} axis to x_i axis, which is measured in the positive direction along z_i axis.

α_i : An angle from z_{i-1} axis to z_i axis around x_{i-1} axis in the clockwise direction.

a_i : A distance from z_{i-1} axis to z_i axis, which is measured in the positive direction along x_{i-1} axis.

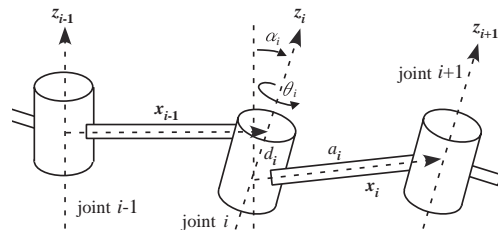


Figure 4 : The parameters in relative coordinate system of each link.

The kinematics representation used in this work as the basis for calibration is the widely used representation that is defined by Denavit and Hartenberg [10],

$$\begin{aligned}
{}^{i-1}T_i &= \text{Rot}(y_{i-1}, \beta_i) \text{Rot}(x_{i-1}, \alpha_i) \text{Trans}(0, 0, d_i) \text{Rot}(z_{i-1}, \theta_i) \text{Trans}(a_i, 0, 0) \\
&= \begin{bmatrix} C\beta_i & 0 & S\beta_i & 0 \\ 0 & 1 & 0 & 0 \\ -S\beta_i & 0 & C\beta_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C\alpha_i & -S\alpha_i & 0 \\ 0 & S\alpha_i & C\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} C\theta_i & -S\theta_i & 0 & 0 \\ S\theta_i & C\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} C\beta_i C\theta_i + S\alpha_i S\beta_i S\theta_i & C\theta_i S\alpha_i S\beta_i - C\beta_i S\theta_i & C\alpha_i S\beta_i & d_i C\alpha_i S\beta_i + a_i(C\beta_i C\theta_i + S\alpha_i S\beta_i S\theta_i) \\ C\alpha_i S\theta_i & C\alpha_i C\theta_i & -S\alpha_i & -d_i S\alpha_i + a_i C\alpha_i S\theta_i \\ -C\theta_i S\beta_i + C\beta_i S\alpha_i S\theta_i & C\beta_i C\theta_i S\alpha_i + S\beta_i S\theta_i & C\alpha_i C\beta_i & d_i C\alpha_i C\beta_i + a_i(-C\theta_i S\beta_i + C\beta_i S\alpha_i S\theta_i) \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)
\end{aligned}$$

in which transformations between successive link coordinate frames are expressed in terms of 3 fixed parameters and one joint variable. For all-revolute robots, the fixed parameters are a_i , d_i and α_i for link i .

The defective point of the D-H representation is that the parameters are discontinuous with respect to perturbations of nearly parallel joint axes. This discontinuity problem is solved with the substitution of an additional rotation parameter (β_i) for nearly parallel axes transformations and suppression of the parameter d_i to zero [1],[6],[10].

β_i : An angle from z_{i-1} axis to z_i axis around y_{i-1} axis in the clockwise direction.

This is called as a modified D-H method. We calibrate five parameters between neighbor joints.

First of all, homogeneous 4×4 transformation matrix ${}^{i-1}T_i$ of each link is calculated as the equation (1). Then, we substitute kinematics parameters of each link such as $a_i, \theta_i, d_i, \alpha_i$ and β_i into the equation (1), we obtain 4×4 transformation matrix of each link. Furthermore, using these matrices successively, we obtain 4×4 transformation matrix 0T_n between coordinate systems of the tip link and the base link (n : the number of links). The local coordinate system of the base link is to be the absolute coordinate system.

$${}^0T_n = {}^0T_1 {}^1T_2 \dots {}^{n-1}T_n \quad (2)$$

By this equation (2), we represent position and orientation r of manipulator tip by a set of parameters of the D-H method. Here, α is a vector whose elements are α_i in all link coordinate systems ($0 \leq i \leq n$). That is, $\alpha = [\alpha_1, \alpha_2, \dots, \alpha_n]^T$ is defined. In the same way, a, θ, d, β are denoted as vectors whose elements are $a_i, \theta_i, d_i, \beta_i$ in all link coordinate systems ($0 \leq i \leq n$). In conclusion, we obtain the following equation (ϕ : a set of all kinematics parameters):

$$r = f(\alpha, a, \theta, d, \beta) = f(\phi) \quad (3)$$

4 Dynamics by Newton-Euler Method

The dynamics of multi-link structure can be expressed as Newton-Euler equations as follows:

1. After giving joint parameters $\theta_i, \dot{\theta}_i, \ddot{\theta}_i$, we calculate revolution angle ω_i , angular velocity $\dot{\omega}_i$, angular acceleration $\ddot{\omega}_i$, and linear acceleration \dot{v}_i at the center of mass of each link i are successively calculated from the base to the tip.

First of all, we give a set of initial parameters, kinematics and dynamic parameters.

${}^0\omega_0 = {}^0\dot{\omega}_0 = 0, {}^0\dot{v}_0 = -\tilde{g}$ (g : the acceleration vector of gravity)

m_i : mass of each link i ($i = 1, 2, \dots, n$), ${}^i\hat{s}_i$: vector from the origin of coordinate system of each link i to its center of mass, iI_i : inertia tensor of each link i , ${}^{i-1}\hat{p}_i$: translation part of homogeneous 4×4 transformation matrix ${}^{i-1}T_i$, ${}^{i-1}R_i$: rotation part of homogeneous 4×4 transformation matrix ${}^{i-1}T_i$, ρ_i : 0 if i is evolutionary joint, 1 if i is translation joint, ${}^{n+1}f_{n+1}$: outer force acting to the link n , ${}^{n+1}n_{n+1}$: outer moment acting to the link n

Direct calculation part ($i = 1, 2, \dots, n$):

$${}^i\omega_i = {}^iR_{i-1} {}^{i-1}\omega_{i-1} + \rho_i z_0 \dot{\theta}_i$$

$${}^i\dot{\omega}_i = {}^iR_{i-1} {}^{i-1}\dot{\omega}_{i-1} + \rho_i \{z_0 \ddot{\theta}_i + {}^iR_{i-1} {}^{i-1}\omega_{i-1} \times (z_0 \dot{\theta}_i)\}$$

$${}^i\dot{v}_i = {}^iR_{i-1} \{ {}^{i-1}\dot{v}_{i-1} + {}^{i-1}\dot{\omega}_{i-1} \times {}^{i-1}\hat{p}_i + {}^{i-1}\omega_{i-1} \times ({}^{i-1}\omega_{i-1} \times {}^{i-1}\hat{p}_i) \} + (1-\rho) \{ z_0 \ddot{\theta}_i + 2({}^iR_{i-1} {}^{i-1}\dot{\omega}_{i-1}) \times (z_0 \dot{\theta}_i) \}$$

2. we successively calculate force ${}^i f_i$ and moment ${}^i n_i$ around each joint $i+1$ from force and moment acting at the mass center of each link i from the tip to the base. Then, we calculate torques τ_i acting to each joint i .

Inverse calculation part ($i = n, \dots, 2, 1$):

$${}^i f_i = m_i {}^i\dot{v}_i + {}^i\dot{\omega}_i \times m_i {}^i\hat{s}_i + {}^i\omega_i \times ({}^i\omega_i \times m_i {}^i\hat{s}_i) + {}^iR_{i+1} {}^{i+1}f_{i+1}$$

$${}^i n_i = {}^iI_i {}^i\dot{\omega}_i + {}^i\omega_i \times ({}^iI_i {}^i\omega_i) + m_i {}^i\hat{s}_i \times {}^i\dot{v}_i + {}^iR_{i+1} ({}^{i+1}\hat{p}_{i+1} \times {}^{i+1}f_{i+1} + {}^{i+1}n_{i+1})$$

$$\tau_i = \begin{cases} z_0 {}^T i n_i + D_i \dot{\theta}_i + f r_i \text{sgn}(\dot{\theta}_i) & \text{(IF angular joint)} \\ z_0 {}^T i f_i + D_i \dot{\theta}_i + f r_i \text{sgn}(\dot{\theta}_i) & \text{(IF linear joint)} \end{cases}$$

By using the above relations, the dynamics of the degrees-of-freedom manipulator is generally expressed by

$$\tau = M(\theta)\ddot{\theta} + h(\dot{\theta}, \theta) + g(\theta) + D\dot{\theta} + E(\dot{\theta}, \theta) \quad (4)$$

For a DD arm, $\theta = (\theta_1, \theta_2)^T$ is the vector of joints angles, $\dot{\theta}$ is the vector of angular velocities, $\ddot{\theta}$ is the vector of angular accelerations and $\tau = (\tau_1, \tau_2)^T$ is the vector of joint torques. $M(\theta)$ is regarded as the inertial force, $h(\dot{\theta}, \theta)$ is considered as the centrifugal force and $D\dot{\theta}$ and $E(\dot{\theta}, \theta)$ are the viscous and dynamic frictions. This is a horizontal manipulator, and consequently the gravity item is initially neglected ($g(\theta)$ is the gravity).

Moreover, the following equation $h(\dot{\theta}, \theta) = d/dt\{M(\theta)\dot{\theta}\} - 1/2(\partial/\partial\theta)\{\dot{\theta}^T M(\theta)\dot{\theta}\}$ can be calculated by a set of parameters of $M(\theta)$, $g(\theta)$, D and $E(\dot{\theta}, \theta)$.

$$M(\theta) = \begin{bmatrix} M_{11} & , & M_{12} \\ M_{21} & , & M_{22} \end{bmatrix}, \quad D = \begin{bmatrix} D_1 & , & 0 \\ 0 & , & D_2 \end{bmatrix}$$

$$E_i = f_{ci} \text{sgn} \dot{\theta}_i, \quad (\dot{\theta}_i \neq 0) \quad \text{or} \quad E_i = F_i, \quad (\dot{\theta}_i = 0)$$

$$M_{11} = I_{1zz} + m_2 L_1^2 + I_{2zz} + 2m_2 L_1 (\hat{s}_{2x} C_2 - \hat{s}_{2y} S_2)$$

$$M_{12}, M_{21} = I_{2zz} + m_2 L_1 (\hat{s}_{2x} C_2 - \hat{s}_{2y} S_2), \quad M_{22} = I_{2zz}.$$

Therefore, robot calibration is to identify dynamic parameters $F_1, F_2, D_1, D_2, f_{c1}, f_{c2}, I_{1zz} + m_2 L_1^2, I_{2zz}, m_2 \hat{s}_{2x}$ and $m_2 \hat{s}_{2y}$. Here, F_i, f_{ci} and D_i are the maximum of static friction, dynamic friction and viscous damping friction of i -th joint, respectively. m_i, I_i and L_i are the mass, inertia tensor and length of i -th link. \hat{s}_i is the center of mass m_i .

5 Genetic Algorithms

GA is an algorithm for simulating biological evolution in PC. GA is a kind of probabilistic search. GA consists of three operations, i.e., selection, crossover mutation. A gain is gradually evaluated after it is successively managed by three operations. The evolution is continued until a given evaluation value is maximized or minimized. A concept of calibration based on GA is shown in Fig.5.

- Step 1. Design chromosome as combination of kinematics parameters for calibration.
- Step 2. Generate the first generation with N individuals. Each individual has kinematics parameters randomly.
- Step 3. Calculate evaluation value of each individual. The higher the evaluation value is, the smaller an error $\Delta r = r^* - r$ is. r^* is a set of measured parameters from sensors, and r is a set of calculated parameters from each individual.
- Step 4. Remove several individuals whose evaluation values are lower, whose number is determined in advance.

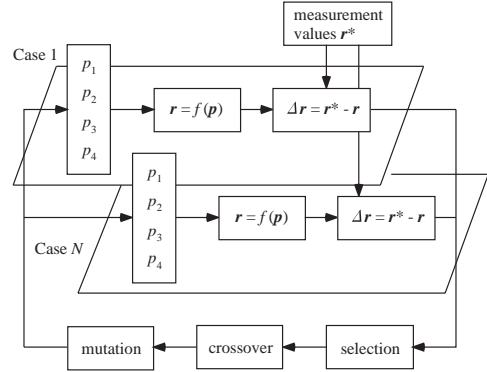


Figure 5 : The flow-chart of calibration algorithm based on GA

- Step 5. Mutation occurs for a few of individuals whose number is bounded as small number in advance.
- Step 6. The other individuals are crossover, and next generation appears as N new individuals.

We iterate the above procedure until 500 generations appear for kinematics calibration and 1000 generations appear for dynamic calibration. Finally we finish GA with the optimal solution.

The reason to use GA is as follows: Concerning to convergence stability and calibrated parameter precision, we compare three types of calibration algorithms based on GA (Genetic Algorithm), NR (Newton-Raphson) and NM (Nelder-Mead) in our problem. NR (Newton-Raphson) method requires a good set of initial values for calibrating parameters, requests the steepest descendent function (differentiate r in our case). However in general, it is difficult for us to acquire the good initial set for a human arm or leg. In such a case, even though NR spends much calculation time, it cannot obtain any accurate set and consequently its convergence becomes unstable. As shown in Fig.6, convergence of a near-optimal set by NR is time consuming against that by GA. NM (Nelder-Mead) method is to minimize a non-linear function based on the simplex method. It does not need any steepest descendent function but any also needs a set of initial values in calibrating parameters. Because of the same reason, even though NM spends much calculation time, it cannot obtain any accurate set and consequently its convergence becomes unstable and slow as illustrated in Fig.6(b).

As contrasted with these, GA does not need both, i.e., a good set of initial calibrating parameters and the steepest descendent function. It requires only the target function r . GA is probabilistic flexible approach and its convergence is relatively stable. As a result, GA quickly selects the final calibrated set which leads high precision of the tip position among three methods

Table 1 : Simulation result: Even though an error [mm] between measured and calibrated tip positions increases and synchronously errors [deg] between measured and calibrated joint angles increase, differences [mm] between true and calibrated kinematics parameters are small enough. The true parameters are provided by the company. The calibration algorithm is based on GA.

Tip error [mm]	θ error [deg]	First joint				Second joint			
		α [deg]	a [m]	d [m]	β [deg]	α [deg]	a [m]	d [m]	β [deg]
True parameters provided by the company		0.00000	0.25000	0.00000	0.00000	0.00000	0.30000	0.00000	0.00000
0	0	0.00000	0.24592	0.00000	0.00000	-0.00510	0.30049	0.00000	0.02872
1	1	0.00000	0.24924	0.00000	0.00000	0.02518	0.31351	0.00000	0.01104
10		0.00000	0.25193	0.00000	0.00000	-0.04391	0.31180	0.00000	0.05441
30		0.00000	0.24331	0.00000	0.00000	0.01191	0.30355	0.00000	0.00232
1	3	0.00000	0.25356	0.00000	0.00000	-0.01301	0.29297	0.00000	0.00189
10		0.00000	0.25224	0.00000	0.00000	0.02729	0.30047	0.00000	-0.00941
30		0.00000	0.24603	0.00000	0.00000	-0.01425	0.30110	0.00000	-0.03788
1	10	0.00000	0.25975	0.00000	0.00000	0.02465	0.28541	0.00000	0.03759
10		0.00000	0.25736	0.00000	0.00000	-0.01828	0.28669	0.00000	-0.07649
30		0.00000	0.25821	0.00000	0.00000	0.00549	0.28019	0.00000	-0.01607
1	15	0.00000	0.26472	0.00000	0.00000	-0.00161	0.27257	0.00000	-0.02935
10		0.00000	0.25618	0.00000	0.00000	0.01282	0.27069	0.00000	0.01622
30		0.00000	0.24974	0.00000	0.00000	0.01176	0.27923	0.00000	-0.03088

as shown in Fig.6(a).

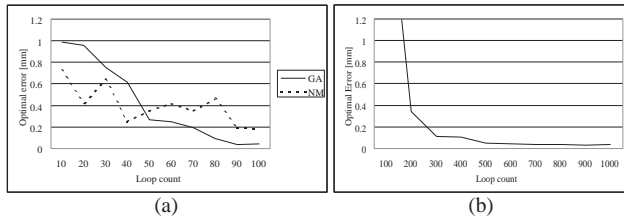


Figure 6 : Convergence properties of tip positions under (a) GA and NM calibration algorithms (b) NR calibration algorithm.

6 Calibration Algorithms

In this section, we explain how to calibrate a set of kinematics parameters and then another set of dynamic parameters in this order.

6.1 Kinematics Calibration

The topic of this section is how to calibrate all parameters of robot kinematics. In a modified D-H method explained previously, we describe relative relationship of position and orientation between neighbor links. Furthermore, even though we do not know any arm structure including many links, we can calibrate the structure itself by using the modified D-H method, i.e., $r = f(\alpha, a, \theta, d, \beta) = f(\phi)$. In the kinematics calibration, the difference $\Delta r = r^* - r$ decreases in GA. Here, true tip position r^* is experimentally measured by the acceleration scale, and virtual tip position $r = f(\alpha, a, \theta, d, \beta) = f(\phi)$ based on measured θ is calculated. By converging $\Delta r = r^* - r$ to zero, we successively change ϕ and then calibrate a set of kinematics parameters. Overall, the calibration of kinematics requires a set of tip position and joint angles captured from four acceleration scales.

For our DD arm, we calibrate a set of kinematics parameters from measured tip position and joint angles, and they are captured from four scale positions.

Table 2 : Experimental result: Even though an error [mm] between measured and calibrated tip positions implicitly exists and also errors [deg] between measured and calibrated joint angles implicitly exist, differences [mm] between true and calibrated kinematics parameters are small enough. The true parameters are provided by the company. The calibration algorithm is based on GA.

joint	true parameters		calibrated parameters	
	first	second	first	second
α [deg]	0.000	0.000	0.000000	-0.000024
a [m]	0.250	0.300	0.245785	0.303113
d [m]	0.000	0.000	0.000000	0.000000
β [deg]	0.000	0.000	0.000000	-0.003617

Each of four positions usually includes the Gaussian white noise, and consequently measured tip position and joint angles have the same noises. For this reason, we investigate how much Gaussian white noises are acceptable in tip position and joint angles for convergence of kinematics parameters. When the error is included in the standard distribution by 67 percentages, simulation results are shown in Table 1. As illustrated in Table 1, we understand our calibration algorithm is robust against errors of captured tip position and joint angles, e.g., 30 [mm] and 15 [deg], respectively. As a result, such Gaussian errors are negligible in our kinematics calibration. In fact, we can calibrate a set of kinematics parameters from measured tip position and joint angles experimentally (Table 2).

In the kinematics calibration by GA, even though noises of tip position and joint angles increase, errors of calibrated kinematics parameters sometimes decrease (sometimes increase). They are not connected each other. The reason is that GA is a probabilistic approach without depending on a set of initial values of kinematics parameters. In addition, since α [deg] and d [m] of the first joint, and d [m] of the second joint are zero in our DD arm, they are fixed as zero

in our GA calibration. These can be given by some restricted conditions in GA.

6.2 Dynamics Calibration

The topic of this section is how to calibrate all parameters of robot dynamics. In order to calibrate a set of dynamic parameters in the entire classic calibration algorithm, we successively need joint angles θ , angular velocities $\dot{\theta}$, angular accelerations $\ddot{\theta}$, and then joint torques τ . However, if we want to calibrate a human arm or leg, we cannot unfortunately measure any torque τ directly. To overcome this drawback, we artificially generate τ by many kinds of PD control laws and feedback gains as follows:

$$\tau_{pd} = k_p(\theta^* - \theta) + k_v\dot{\theta} \quad (5)$$

$$\tau_{pd} = k_p(\theta^* - \theta) + k_v\dot{\theta} + k_a\ddot{\theta} \quad (6)$$

$$\tau_{pd} = k_p(\theta^* - \theta) \quad (7)$$

k_p, k_v, k_a : Gain θ^* : target point θ : present point

Then, using $\theta, \dot{\theta}, \ddot{\theta}$ as measured data, we calibrate a set of dynamic parameters by GA. First of all, for $\tau_{pd}, \theta, \dot{\theta}, \ddot{\theta}$, we calculate

$$\tau_{ca} = M(\theta)\ddot{\theta} + h(\dot{\theta}, \theta) + g(\theta) + D\dot{\theta} + E(\dot{\theta}, \theta). \quad (8)$$

τ_{ca} : Calculated τ

In the dynamics calibration, we decrease

$$\Delta r = (\tau_{pd} - \tau_{ca})^2 \quad (9)$$

by GA. By adjusting a set of dynamic parameters, GA minimizes the difference between true and virtual positions of arm tip. The simulation results existing Gaussian white noises in angles θ are shown in Table 3. In this case, terms $I_1 + m_2L_1^2$, $m_2\hat{s}_{2x}$, $m_2\hat{s}_{2y}$, I_2 concerning to inertia and centrifugal forces tend to increase. However, terms D_1, D_2, f_{c1}, f_{c2} concerning to friction forces synchronously increase in our dynamics calibration based on GA. As a result, as long as errors of the angles θ are less than 10 percentages, we can obtain a better dynamics.

7 Motion Sequence Based on Dynamic Equation and Runge-Kutta Method

In this section, we explain a classic approach to make a sequence of joint angles of a virtual manipulator in a 3-D graphics world. The approach consists of Runge-Kutta and Euler methods based on calibrated kinematics and dynamic equations.

1. We initialize $t = 0$ and $\theta_i^t = \dot{\theta}_i^t = 0$. Then, in order to move a virtual manipulator, we give an arbitrary torque τ_i^t .
2. We calculate $\ddot{\theta}_i^t$ by a calibrated dynamic equation $\ddot{\theta}_i^t = M(\theta_i^t)^{-1}(\tau_i^t - h(\theta_i^t, \dot{\theta}_i^t) - g(\theta) - D\dot{\theta}_i^t - E(\dot{\theta}, \theta))$.

3. By Runge-Kutta method, we calculate a next angular velocity $\dot{\theta}_i^{t+1}$ from a present angular acceleration $\ddot{\theta}_i^t$ as follows: $f(t, \theta_i^t) = \dot{\theta}_i^t$, $k_1 = hf(t, \dot{\theta}_i^t)$, $k_2 = hf(t + h/2, \dot{\theta}_i^t + k_1/2)$, $k_3 = hf(t + h/2, \dot{\theta}_i^t + k_2/2)$, $k_4 = hf(t + h, \dot{\theta}_i^t + k_3)$, $k = (k_1 + 2k_2 + 2k_3 + k_4)/6$, $\dot{\theta}_i^{t+1} = \dot{\theta}_i^t + k$. In succession, by Euler method, we calculate θ_i^{t+1} from $\dot{\theta}_i^t$ as follows: $\theta_i^{t+1} = \theta_i^t + \dot{\theta}_i^{t+1}h$.
4. If $t \geq t_{end}$ is kept, the algorithm ends. Otherwise, we set $t=t+h$ and return to step 2.

8 Parameter Calibration and Dynamic Animation

First of all, we capture two kinds of motion patterns, i.e., learning patterns and evaluating patterns by controlling a DD arm based on a PD feedback schema and its arbitrary gains. The control law is based on the equation (5), and both feedback gains are selected by 9.5 and -9.5 for maintaining the stable control of DD arm. However, we cannot see any law and gains for a human arm or leg because we do not know how it moves. Therefore for the DD arm, we cannot use them for calibrating its set of dynamic parameters. 5 learning patterns are used for calibrating dynamic parameters in GA (In part A of Fig.7). On the other hand, 17 evaluating patterns are used for evaluating artificial animation based on calibrated parameters (In part B of Fig.7). Each motion pattern consists of joint angle, velocity and acceleration calculated from many captured sequences of landmark positions around each link of the arm. The sampling time of captured positions is 40 [μs], and for example, five learning patterns consists of 1054688 data. Note that the motion pattern never includes any joint torque.

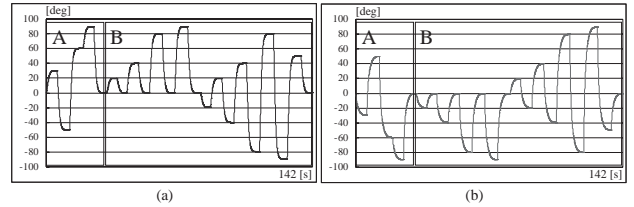


Figure 7 : Experimental results: (a) Learning data of 5 motion patterns for the first joint are described in the part A, and evaluating data of 17 motion patterns for the first joint are described in the part B. (b) Learning data of 5 motion patterns for the second joint are described in the part A, and evaluating data of 17 motion patterns for the second joint are described in the part B.

8.1 Generalization of Calibrated Dynamics

In this paragraph, we investigate whether the proposed dynamics calibration is generally used or not. For this purpose, we check whether many sets of dynamic parameters calibrated in different control laws, different feedback gains, and different numbers of learning data are similar or not.

Table 3 : Simulation result: Even though errors [deg] between measured and calibrated joint angles increase, differences between true and calibrated dynamic parameters are small enough. The calibration algorithm is based on GA. Note that the true parameters were completely calibrated by our proposed smart algorithm based on successive joint angles, angular velocities, angular accelerations, and torques [7].

error [%]	$I_1 + m_2 L_1^2$	$m_2 \hat{s}_{2x}$	$m_2 \hat{s}_{2y}$	I_2	D_1	D_2	f_{c1}	f_{c2}
true	1.986076	1.718632	0.181763	0.471209	-0.002409	0.000311	0.010918	-0.002271
0	1.986364	1.695451	0.218829	0.473631	0.023210	-0.000317	-0.005511	0.004903
3	1.986366	1.695452	0.218919	0.473770	0.025214	-0.000339	-0.005713	0.005126
5	1.955109	1.720987	0.198811	0.463530	0.026118	-0.000362	0.011338	-0.002305
10	2.033218	1.802012	0.207333	0.510248	-0.002630	-0.000361	0.011241	-0.002554

First of all, as illustrated in Tables 4 and 5, even though we misunderstand a true control law and its feedback gains, our dynamics calibration based on misunderstood law and gains gets a set of good dynamic parameters. Especially in Table 4, qualities of viscous and Coulomb friction terms, i.e., D_1, D_2, f_{c1}, f_{c2} are not so good. However, generated dynamic animation is not always affected by them. Moreover, qualities of inertia terms, i.e., $I_1 + m_2 L_1^2, m_2 \hat{s}_{2x}, m_2 \hat{s}_{2y}, I_2$ are not good, but dynamic animation is relatively good because of the balance between two calibrated sets.

Secondly, as shown in Table 6, we investigate whether qualities of dynamic parameters increase or not, which are calibrated by changing the number of learning data. In this calibration, we use the same control equation (5) and feedback gains $k_p=9.5$ and $k_v=-9.5$, which we control a real DD arm stably. From these results, the larger the number of learning data is, the more the precision of all the dynamic parameters is. As a result, calibration based GA can be applied for calibrating human arm or leg whose control scheme, feedback gain, joint torques are completely uncertain.

Finally in all tables, a true set of dynamic parameters was calibrated by our smart algorithm [7]. The algorithm consists of two procedures: the former finds an approximated set of initial dynamic parameters by a few special motions, and the latter selects a precise set of dynamic parameters by the least square method under an enormous number of general motions. Each of both procedures uses exact sequences of joint angles, angular velocities, angular accelerations and torques. In addition, the former three are precisely measured by high-resolution joint encoders.

8.2 Real and Virtual Dynamic Animations

When a real DD arm is supervised by the PD control based on equation (5) with feedback gains $k_p=9.5$ and $k_v=-9.5$, we generate a real dynamic animation including a sequence of its joint angles. On the other hand, when a virtual DD arm is supervised by the same scheme, we generate a virtual dynamic animation including a sequence of its joint angles. We note that the virtual DD arm is built by calibrated sets of kinematics parameters and dynamics parameters. In Fig.7, we describe a real dynamic animation whose se-

quence of joint angles are directly captured from four acceleration scales. As contrasted with this, we describe a virtual dynamic animation whose sequence of joint angles are artificially generated in PC (Fig.8). Comparing with these animations, even though we cannot measure any joint torque, we can get all dynamic parameters precisely and consequently generate exact dynamic animation. In the learning patterns (the part A), average and maximum differences between real and virtual joint angles are 0.73 and 3.51 degrees, respectively. Also in the evaluating patterns (the part B), average and maximum differences between real and virtual joint angles are 1.91 and 12.7 degrees, respectively.

Furthermore, if the number of data is changed from 1000 to 500000 in the same control law and feedback gain (i.e., equation (5) and $k_p = 9.5, k_v = -9.5$), which corresponds to the trial number 2 in Table 6, average and maximum differences between real and virtual joint angles are 0.94 and 3.70 degrees, respectively in the learning patterns (in the part A), average and maximum differences between real and virtual joint angles are 1.6 and 15.3 degrees, respectively in the evaluating patterns (in the part B). Moreover, if the gains are changed from $k_p = 9.5, k_v = -9.5$ to $k_p = 11.5, k_v = -11.5$ in the equation (5) with 300000 data, which corresponds to the trial number 7 in Table 4, average and maximum differences between real and virtual joint angles are 2.51 and 4.06 degrees, respectively in the learning patterns (in the part A), average and maximum differences between real and virtual joint angles are 3.93 and 18.2 degrees, respectively in the evaluating patterns (in the part B). Finally, if the control law is changed from the equation (5) to (7) with the various gain and 300000 data, which corresponds to the trial number 3 in Table 5, average and maximum differences between real and virtual joint angles are 2.99 and 5.73 degrees, respectively in the learning patterns (in the part A), average and maximum differences between real and virtual joint angles are 5.4 and 102.7 degrees, respectively in the evaluating patterns (in the part B). The reason of the maximum difference (102.7 degree) is time delay between real and virtual animations, which is accumulated in the Runge-Kutta's integration.

Table 4 : A set of dynamic parameters calibrated in GA while changing a set of two gains.

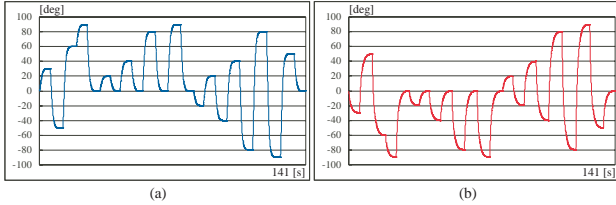
trial number	gain		dynamic parameters							
	k_p	k_v	$I_1 + m_2 L_1^2$	$m_2 \hat{s}_{2x}$	$m_2 \hat{s}_{2y}$	I_2	D_1	D_2	f_{c1}	f_{c2}
0	true		1.986076	1.718632	0.181763	0.471209	-0.002409	0.000311	0.010918	-0.002271
1	9.5	-9.5	1.985170	1.717460	0.175769	0.470695	0.021347	0.001663	-0.007767	0.000015
2	1.5	-1.5	1.869282	1.690460	0.085222	0.449882	0.212757	-0.102037	0.095659	-0.247547
3	5.5	-5.5	1.968730	1.722643	0.124559	0.468498	0.339681	0.000748	-0.181659	0.000046
4	7.5	-7.5	1.982657	1.687480	0.156176	0.472709	0.207507	-0.061082	-0.091325	0.050492
5	8.5	-8.5	1.868730	1.689831	0.124895	0.469352	0.249989	0.060441	-0.144549	-0.026383
6	10.5	-10.5	1.874896	1.722365	0.079423	0.445090	-0.468620	0.017624	-0.001938	-0.458488
7	11.5	-11.5	2.109619	1.759582	0.287457	0.682475	0.164714	-0.044144	-0.095361	0.135760

Table 5 : A set of dynamic parameters calibrated in GA while changing control laws.

trial number	gain			dynamic parameters							
	k_p	k_v	k_a	$I_1 + m_2 L_1^2$	$m_2 \hat{s}_{2x}$	$m_2 \hat{s}_{2y}$	I_2	D_1	D_2	f_{c1}	f_{c2}
0	true			1.986076	1.718632	0.181763	0.471209	-0.002409	0.000311	0.010918	-0.002271
1	9.5	-9.5	-1.5	2.042711	1.771038	0.290418	0.511296	-0.011266	-0.001823	-0.000011	-0.004312
2	9.5	-9.5	-3.5	2.147862	1.978971	0.590418	0.838907	-0.028721	-0.011873	-0.002113	-0.008002
3	5.5	-5.5	-5.5	1.913293	1.749721	0.100278	0.505012	0.248843	-0.152310	0.129829	-0.278322
4	1.5	-1.5	-1.5	1.992348	1.700519	0.091100	0.320199	0.288740	0.001221	0.147633	0.012918
5	5.5	—	—	2.021246	1.802931	0.278861	0.521320	-0.003163	0.000006	0.000157	-0.001989
6	1.5	—	—	1.792465	1.688373	0.200023	0.469322	-0.001506	0.000405	0.021877	-0.003125

Table 6 : A set of dynamic parameters calibrated in GA while changing the number of data.

trial number	number of data	$I_1 + m_2 L_1^2$	$m_2 \hat{s}_{2x}$	$m_2 \hat{s}_{2y}$	I_2	D_1	D_2	f_{c1}	f_{c2}
0	true	1.986076	1.718632	0.181763	0.471209	-0.002409	0.000311	0.010918	-0.002271
1	1000	2.179670	1.874980	0.184588	0.518914	0.121080	0.019547	-0.046891	-0.015610
2	100000	2.062490	1.781230	0.156268	0.490593	-0.062486	0.082048	0.000992	-0.093980
3	300000	1.985170	1.717460	0.175769	0.470695	0.021347	0.001663	-0.007767	0.000015
4	500000	2.065510	1.687480	0.193362	0.484825	0.000021	0.061021	0.052644	0.000007

**Figure 8** : Virtual results: (a) Learning data of 5 motion patterns for the first joint are described in the part A, and evaluating data of 17 motion patterns for the first joint are described in the part B. (b) Learning data of 5 motion patterns for the second joint are described in the part A, and evaluating data of 17 motion patterns for the second joint are described in the part B.

As a result, we see that the difference between real and virtual animations based on the same scheme is too small. Furthermore, we understand that the difference between real and virtual animations based on different schemes is small enough.

9 Conclusions

In this research, we proposed a smart approach for calibrating kinematics and dynamic parameters for an arm (consists of connected multi-links) without knowing their good initial values and torques running in the arm. The approach is based on GA, which needs not a good set of initial values of calibrating parameters. The calibration is successively processing from arbitrary tip and endpoint positions captured around all links by the acceleration scales. Finally, the superiority of our algorithm is checked in several experimental results.

Acknowledgments

This research is supported in part by 2002 Grants-in-aid for Scientific Research from the Ministry of Education, Science and Culture, Japan (No.14550247).

References

- [1] S.A.Hayati, "Robot arm geometric link parameter estimation," *Proc. of the IEEE Conf. Decision and Control Symp. Adapt. Processes*, Vol.22, pp.1477-1483, 1983.
- [2] H.Mayeda, K.Osuka and A.Kangawa, "A new identification method for serial manipulator arms," *Proc. of the IFAC World Congress*, pp.74-79, 1984.
- [3] A.Mukerjee and D.H.Ballard, "Self-calibration in robot manipulators," *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pp.1050-1057, 1985.
- [4] H.B.Olsen and G.A.Bekey, "Identification of parameters in models of robots with rotary joints," *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pp.1045-1049, 1985.
- [5] H.B.Olsen and G.A.Bekey, "Identification of robot dynamics," *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pp.1004-1010, 1986.
- [6] H.Kawasaki and K.Nishimura, "Terminal-link parameter estimation of robotic manipulators," *IEEE Journal of Robotics and Automation*, pp.485-490, 1988.
- [7] T.Otsuki, T.Iguchi, Y.Murata and H.Noborio, "On the Identification of Robot Parameters by the Classic Calibration Algorithms and Error Absorbing Trees," *Proc. of the IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, pp.1916-1923, 2002.
- [8] W.Maurel, Y.Wu, N.M.Thalmann, D.Thalmann, "Biomechanical models for soft tissue simulation," *Springer Basic Research Series*, Springer-Verlag Berlin/Heidelberg 1998.
- [9] W.Maurel, "3D modeling of the human upper limb including the biomechanics of Joints, Muscles and Soft Tissues," *Ph.D. Thesis 1906*, Laboratoire d'Infographie - Ecole Polytechnique Federale de Lausanne, 1998.
- [10] J.Denavit and R.S.Hartenberg, "A kinematic notation for lower-pair mechanisms based on matrices," *ASME Journal of Applied Mechanics*, pp.215-221, 1955.