High Precision Path Generation of an LCD Glass-Handling Robot

Phil-Joo, Cho*, Hyo-Gyu, Kim, and Dong-Il, Kim

Mechatronics & Manufacturing Technology Center, Samsung Electronics, Suwon, Korea
(* Tel : +82-31-203-2773; E-mail: pj.cho@samsung.com)

Abstract: Progress in the LCD industries has been very rapid. Therefore, their manufacturing lines require larger LCD glass-handling robots and more precise path control of the robots. In this paper, we present two practical advanced algorithms for high-precision path generation of an LCD glass-handling robot. One is high-precision path interpolation for continuous motion, which connects several single motions and is a reliable solution for a short robot cycle time. We demonstrate that the proposed algorithm can reduce path error by approximately 91% compared with existing algorithms without increasing cycle time. The second is real-time static deflection compensation, which can optimally compensate the static deflection of the handling robot without any additional sensors, measurement instruments or mechanical axes. This reduces vertical path error to approximately 60% of the existing system error. All of these algorithms have been commercialized and applied to a seventh-generation LCD glass-handling robot.

Keywords: High-precision path interpolation, dual acceleration/deceleration filter, real-time static deflection compensation

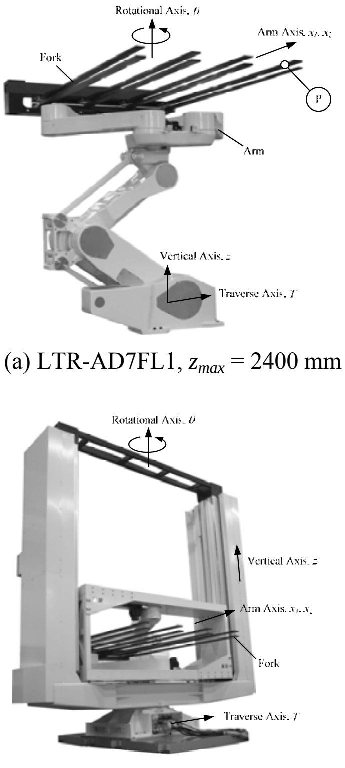
1. INTRODUCTION

The current rapidly increasing demand for liquid crystal display (LCD) devices requires larger panels and continuous improvement in productivity to enable the device manufacturers to be competitive in the market. Increasing the size of the mother glass, which is the basic material for panel production, is critical for satisfying both requirements.

In the production of LCD devices, the robot is a key unit in that it handles (i.e. loads/unloads) the mother glasses between process equipment, such as chemical vapor deposition (CVD), etching, and lithography. The larger the size of the mother glass, the larger the handling robots and process equipment required. In a seventh-generation LCD manufacturing line, the size of mother glass is up to 1870 × 2200 mm so the size of the handling robot must be sufficiently large. Although the handling robot is designed to meet this constraint, the manufacturing line requires a minimum footprint and cycle time for the handling robot. The cycle time is very restricted by the structure of the handling robot. Fig. 1 shows typical configurations of LCD glass-handling robots used in a seventh-generation LCD manufacturing line. There are two types of robots according to z axis structure; one is linkage type (LTR-AD7FL1) and the other is dual-telescopic type (LTR-AD7FT0). Both consist of two arms x_1, x_2, a vertical axis z, a rotational axis θ and a traverse axis T. Because these handling robots have a flexible fork mounted on the end of the arm, which tends to vibrate, the reduction of cycle time by simply increasing robot speed is limited. Continuous motion, which connects several single motions, may be a reliable solution for reducing cycle time.

Vertical path accuracy is very important in the handling robot when considering its operating conditions. In a seventh-generation LCD manufacturing line, the robot arm weighs approximately 190 kg including the fork and glass. Large static deflection of the fork is not avoidable by this weight. In spite of this deflection, the handling robot should transfer the glass on the fork to the glass buffer within 2.5 s, which is located in 4 m away and has the entrance height of only 25 mm for glass-handling. This buffer has maximum 30 slots for glass-loading and stores 30 mother glasses, which may be located in four directions around the robot; i.e., $\theta = 0°$, 90°, 180°, and 270°. According to experiments, however, the static deflection varies according to vertical position z and rotational position θ as well as arm position x.

Therefore, maximization of vertical path accuracy regardless of the robot position—that is, minimizing static deflection—is indispensable for the handling robot.



(a) LTR-AD7FL1, $z_{max} = 2400$ mm

(b) LTR-AD7FT0, $z_{max} = 3600$ mm

Fig. 1 Typical configurations of seventh-generation LCD handling robots.

To satisfy the above requirements for the handling robot, we propose two high-precision path generation algorithms: one is high-precision path interpolation for continuous motion, and the other is real-time static deflection compensation. First, when continuous motion with different motion modes is planned for a short robot cycle time, we show that the proposed path interpolation has a much smaller path error than the existing method, without increasing the cycle time. We also show that optimal continuous motion with arbitrary overlap length is possible using this algorithm. In addition, real-time static deflection compensation is proposed, which guarantees optimal compensation values for the entire robot work range. This can also enhance the vertical path accuracy of arm loading or unloading. All of these have been commercialized and applied to seventh-generation LCD handling robots of Samsung Electronics.

2. HIGH PRECISION PATH INTERPOLATION FOR CONTINUOUS MOTION

In the robot controller, the path interpolation means the calculation of each motor position at every control stage based on the motion command using the following process [1]:

- the acceleration/deceleration process that prevent robot shock and vibration;
- the inverse kinematics process that converts the Cartesian coordinates into joint coordinates; and
- the user interrupts handling process that deals with velocity overrides, pause, resume, etc.

Therefore, the robot's performance, including path accuracy, is mainly decided by the path interpolation. In implementation, the acceleration/deceleration process is principally realized by either of two well-known methods: (i) convolution by an acceleration/deceleration filter [2-5] and (ii) direct calculation by user-defined functions [6-8]. The first method is widely used because of its simple mathematical structure and handling of user interrupts. However, convolution by an acceleration/deceleration filter causes considerable path error when the robot moves continuously. The novel interpolation algorithm proposed can minimize the path error by adopting a dual acceleration/deceleration filter.

2.1 Conventional Path Interpolation

Fig. 2 shows the basic principle of the acceleration/ deceleration process using a single filter. If *v*[*n*] is the input to the system and impulse response *h*[*n*] is the output, then *w*[*n*] is the discrete time convolution of *h*[*n*] and *v*[*n*] [3].

$$w[n] = v[n] * h[n] = \sum_{k=1}^{n} v[k]h[n-k] \tag{1}$$

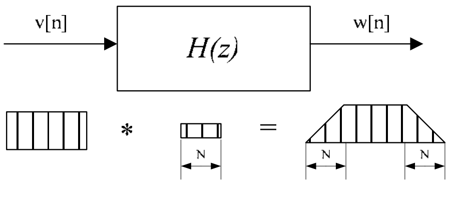


Fig. 2 The acceleration/deceleration process using a single filter

The robot's motion is generally classified into three modes: point-to-point (PTP), linear path (LP), and circular path (CP). In the PTP mode, the path along which the robot moves in Cartesian coordinates need not be considered, and the joint velocity $\Delta\theta$ is given directly as the input value to the acceleration/deceleration filter.

However, the LP or CP modes are restricted by position and posture in Cartesian coordinates. The order in which the acceleration/deceleration and the inverse kinematics are applied greatly affects the path accuracy. From the velocity kinematics, Δx in Cartesian coordinates and $\Delta\theta$ in joint coordinates have the following relationship:

$$\Delta x = J(\theta)\Delta\theta \tag{2}$$

where $J(\theta)$ is the Jacobian matrix and is a function of joint position.

If the acceleration/deceleration process is executed after the inverse kinematics process, the final joint position does not satisfy Eq. (2) because of the general nonlinear relationship between Δx and $\Delta\theta$. This causes a path error in Cartesian coordinates.

If the motion in the LP mode having the first-order acceleration/deceleration is interpolated before the acceleration/deceleration process in Cartesian coordinates, as shown in Fig. 3, the first joint velocity $\Delta\theta_1$ is obtained by subtracting the initial joint position θ_0 from the first interpolation result θ_1. Because the *Z*-transformation of the first-order acceleration/deceleration filter is given by Eq. (3) with an acceleration/deceleration period *m*, $\Delta\theta_1'$ after the acceleration/deceleration of $\Delta\theta_1$ becomes $\Delta\theta_1/m$. Then θ_1', corresponding to θ_1 after acceleration/deceleration, is calculated using Eq. (4).

$$H_l(\theta) = \frac{1}{m}\frac{1-z^{-m}}{1-z^{-1}} \tag{3}$$

$$\theta_1' = \theta_0 + \Delta\theta_1' = \frac{1\cdot\theta_1 + (m-1)\cdot\theta_0}{m} \tag{4}$$

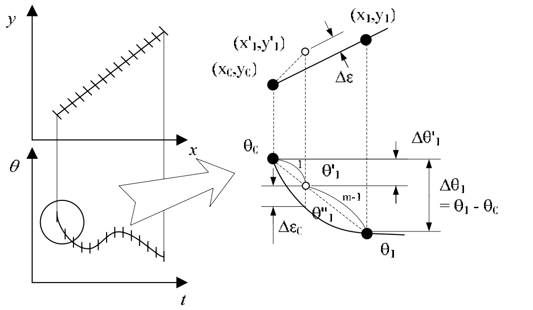


Fig. 3 A path error generated by the acceleration/deceleration process

Equation (4) shows that θ_1' is a point on the line connecting θ_0 and θ_1. If the relationship in Eq. (2) is linear, the position in joint coordinates is θ_1'. However, the nonlinearity results in a path error $\Delta\varepsilon_\theta$ between θ_1', the actual position that the fork of the robot reaches, and θ_1'', the desired position, as shown in Fig. 3. The longer the acceleration/deceleration time or the larger the velocity, the larger the path error.

Therefore, to remove the path error in the LP or CP mode, where the inverse kinematics process is necessary in the path interpolation, the acceleration/deceleration process should be executed before the inverse kinematics process. That is, the linear velocity Δx should be the input value of the acceleration/deceleration filter in the LP mode, and the circumferential velocity $\Delta\psi$ should be the input value in the CP mode for error-free path. This method has no problem in a single motion without superposition between motions.

The productivity of a manufacturing line can be improved considerably by reducing the cycle time of the handling robots. However, the handling robot has a flexible structure so that there is a limit to the reduction of cycle time by simply increasing robot speed. Because of this, the programmer of the robot motion does not put a delay between robot motions and connect several single motions into one continuous motion. The continuous motion can be composed of different motion modes or it may contain some repeated modes.

However, when planning continuous motion with different motion modes, as shown in Fig. 4, the above path interpolation algorithm has a serious problem. Because there is only a single acceleration/deceleration filter, the inverse kinematics process must be performed before the acceleration/deceleration process so that the dimensions of the filter input values for each motion are identical. Therefore, as described above, the previous interpolation has an inherent path error during the acceleration/deceleration process because it is performed after the inverse kinematics process. Fig. 5 shows the existing continuous motion interpolation scheme using a single acceleration/ deceleration filter.

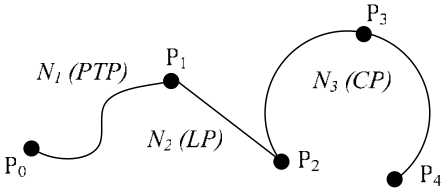


Fig. 4 A continuous motion with different motion modes

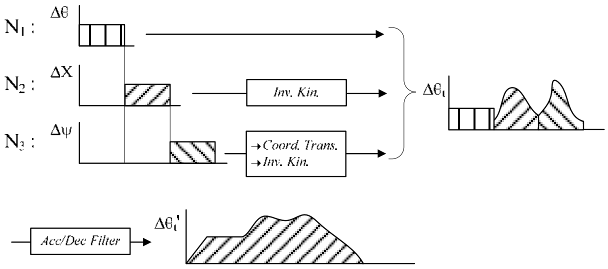


Fig. 5 The existing continuous motion interpolation scheme using a single acceleration/deceleration filter

2.2 The Proposed Path Interpolation

This paper proposes a new path interpolation algorithm to overcome this limitation. The key to the algorithm is that a dual acceleration/deceleration filter is assigned for the acceleration/deceleration process considering the dimensions of acceleration/deceleration filter input values to be different for each motion. Using this algorithm, each motion segment in the continuous motion can have its own acceleration/ deceleration filter so that it can calculate the acceleration/ deceleration independently. Fig. 6 shows the proposed continuous motion interpolation scheme using a dual acceleration/deceleration filter.

The proposed algorithm has a very simple structure and is simple to implement. The interpolation procedure is as follows. First, the acceleration/deceleration process is performed independently for each motion. Next, the joint velocities are calculated using inverse kinematics and these are superimposed during the acceleration/deceleration time; this then becomes the final joint velocity. In the case of continuous motion with two or more motion segments, using the dual acceleration/deceleration filter is possible because the filter can be reused for the new motion segment by reinitializing. The superimposed final joint velocity guarantees an error-free path, except for the motion overlap region, regardless of motion modes or velocity and acceleration/deceleration time because independent path interpolation is possible for each motion segment.

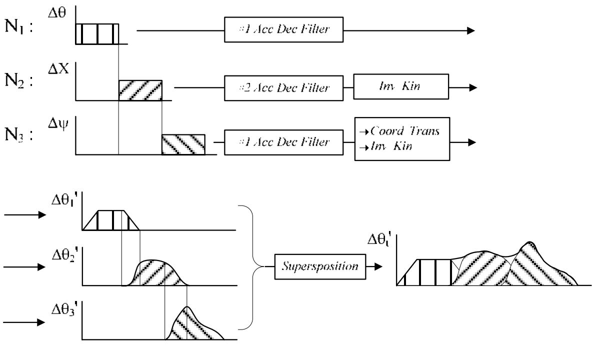


Fig. 6 The proposed continuous motion interpolation scheme using a dual acceleration/deceleration filter

Furthermore, if this algorithm is applied, optimal continuous motion with arbitrary overlap length is possible by adjusting the overlap region of the dual acceleration /deceleration filter. In the existing general robot controller, acceleration time and deceleration time must be the same as the overlap time in continuous motion, and the user cannot set these independently. Therefore, the practical use of this type of continuous motion is limited. However, if the proposed algorithm is adopted, acceleration, deceleration and overlap time can be set by the user independently, and robot cycle time can be reduced significantly without deterioration of path accuracy. This is because the proposed algorithm interpolates each motion independently. Fig. 7 shows a flow chart for the continuous motion interpolation using a dual acceleration/ deceleration filter.

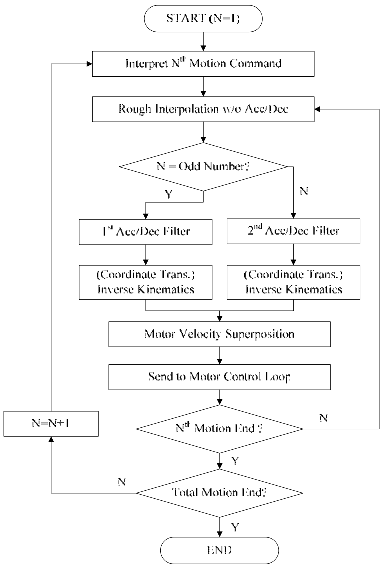


Fig. 7 A flow chart for the continuous motion interpolation using a dual acceleration/deceleration filter

3. REAL-TIME STATIC DEFLECTION COMPENSATION

When handling mother glasses, there are many cases where the robot arm is extended in various vertical *z* positions in the range of 0-3600 mm. At that time, the robot arm has very large static deflection due to self-load, joint flexibility, glass load, etc. Although the optimal design analysis for the robot arm is carried out to minimize this, it is not sufficient for static deflection demand. As mentioned before, in the case of a seventh-generation LCD manufacturing line, the robot arm weighs approximately 190 kg including the fork and glass, but the glass buffer for glass-loading is located in 4 m away and has the entrance height of only 25 mm for the glass loading/unloading. Therefore, minimization of this deflection is the most critical factor for the handling robot. In this paper, a real-time static deflection compensation algorithm is proposed.

Because this algorithm does not require any additional sensors, measurement instruments or mechanical axes, it can be generally applied to a manufacturing line. In real-time static deflection compensation, definition of the compensation formula is most important because it is the basis of all compensation procedures. In this Section, a static deflection compensation formula and forward/inverse kinematics considering compensation are presented.

3.1 Definition of the Static Deflection Compensation Formula

In order to provide the optimal compensation value at any robot position, the static deflection compensation formula is defined as the function of vertical position *z*, rotational position θ as well as arm position *x*. Based on static deflection experimental data, the compensation value *dz* is defined as a second-order polynomial in *x* and *z*, sixth-order in θ. To make joint velocity have no discontinuity because of user interrupts, such as pause and velocity override, it is defined as a function that can be differentiated in any arbitrary position.

The definition of *dz* is shown in Eq. (5), where $p(\theta)$, $q(\theta)$ and $r(\theta)$ are all defined as sixth-order polynomials about θ, and x_m denotes the largest stroke of *x*.

$$dz(z,x,\theta)=\left(\frac{x}{x_m}\right)^2\left(p(\theta)z^2+q(\theta)z+r(\theta)\right) \tag{5}$$

In Eq. (5), reference data for the compensation value δ_{ij} (*i, j* = 1 to 4) are required to determine $p(\theta)$, $q(\theta)$ and $r(\theta)$, where δ_{ij} denotes the compensation value at $z = z_i$, $\theta = \theta_j$; z_i takes the values of 0, $z_m/2$, z_m; θ_j takes the values of $-\pi/2$, 0, $\pi/2$, π; and z_m takes the value of the maximum *z* stroke.

$$z(0,x_m,\theta)=\left(\frac{x}{x_m}\right)^2 f_1(\theta)=\left(\frac{x}{x_m}\right)^2\sum_{k=1}^{7}a_k\theta^{7-k} \tag{6}$$

$$f_1(\theta)=\begin{cases}\delta_{11} & \theta=-\pi,\pi\\ \delta_{12} & \theta=-\pi/2\\ \delta_{13} & \theta=0\\ \delta_{14} & \theta=\ \pi/2\end{cases} \tag{7}$$

where

$$\begin{aligned}
a_1 &= 4\left(-7\delta_{11}+8\delta_{12}-9\delta_{13}+8\delta_{14}\right)/9\pi^6\\
a_2 &= 16\left(\delta_{13}-\delta_{11}\right)/9\pi^5\\
a_3 &= \left(47\delta_{14}-64\delta_{13}+81\delta_{12}-64\delta_{11}\right)/9\pi^4\\
a_4 &= 32\left(\delta_{11}-\delta_{13}\right)/9\pi^3\\
a_5 &= 2\left(-5\delta_{14}+16\delta_{13}-27\delta_{12}+16\delta_{11}\right)/9\pi^2\\
a_6 &= 16\left(\delta_{13}-\delta_{11}\right)/9\pi\\
a_7 &= \delta_{12}
\end{aligned}$$

The functions $(x/x_m)^2 f_2(\theta)$ at $z = z_2 = z_m/2$ and $(x/x_m)^2 f_3(\theta)$ at $z = z_3 = z_m$ are also defined as δ_{2j} and δ_{3j}, respectively, similarly to Eq. (6). The compensation curve along the *z* axis is defined as a second-order curve passing through the three compensation points at $\theta = \theta_s$, $z = 0$ and $z_m/2$, z_m. Fig. 8 shows this curve. Therefore, $p(\theta)$, $q(\theta)$ and $r(\theta)$ in Eq. (5) are calculated as follows.

$$\begin{aligned}
p(\theta) &= \left(2f_1(\theta)-4f_2(\theta)+2f_3(\theta)\right)/z_m^2\\
q(\theta) &= \left(-3f_1(\theta)+4f_2(\theta)-f_3(\theta)\right)/z_m\\
r(\theta) &= f_1(\theta)
\end{aligned} \tag{8}$$

3.2 Forward/Inverse Kinematics Analysis considering Static Deflection Compensation

To apply compensation effectively, it is best that the user does not know the *z* axis rise and fall caused by the real-time compensation. If the *z* value commanded by the user is varied during motion by real-time compensation, it becomes very difficult for the user to perform robot teaching or motion programming. Therefore, a *z* value that the user recognizes should not be changed during motion although the actual *z* value of the robot changes. This constraint can be resolved by forward kinematics considering the static deflection compensation.

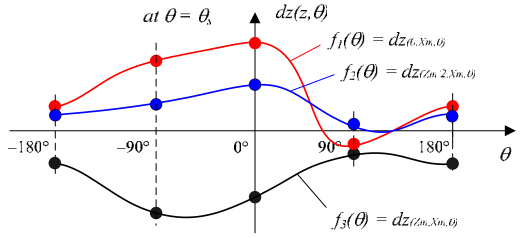


Fig. 8 The compensation curve along the *z* axis at $\theta = \theta_s$.

In static deflection compensation, forward kinematics means the calculation of the *z* position before compensation from the actual *z* position. First, the coordinate's value (z',θ, x) is calculated without compensation; where z' is the actual *z* position and differs from the target position z_r. Then z_r and z' have the following relationship.

$$\begin{aligned}
z' &= z_r+dz(z_r,x,\theta)\\
&= z_r+\left(\frac{x}{x_m}\right)^2\left(p(\theta)z_r^{\ 2}+q(\theta)z_r+r(\theta)\right)\\
&\triangleq z_r+(ax^2)\left(p(\theta)z_r^{\ 2}+q(\theta)z_r+r(\theta)\right)
\end{aligned} \tag{9}$$

If Eq. (9) is arranged to z_r, it becomes a second-order formula as shown in Eq. (10), and the existence of a real solution in forward kinematics can be decided by the determinant Eq. (11).

$$\left[(ax^2)p(\theta)\right]{z_r}^2+\left[1+(ax^2)q(\theta)\right]z_r+\left[(ax^2)r(\theta)-z'\right] \qquad (10)$$
$$\triangleq A{z_r}^2+Bz_r+C=0$$

$$\begin{aligned} Det &\triangleq B^2-4AC \\ &=(ax^2)^2\left[q(\theta)^2-4p(\theta)r(\theta)\right] \\ &\quad+(ax^2)\left[2q(\theta)+4p(\theta)z'\right]+1 \\ &\triangleq Q^2\alpha+Q\beta+1 \end{aligned} \qquad (11)$$

The determinant is always close to unity—that is, a positive value—because both α and β are much smaller than unity in Eq. (11) considering that δ_{ij} is much smaller than z_m. Therefore, Eq. (10) always guarantees two real solutions.

$$\alpha=\frac{1}{{z_m}^2}\left[9{f_1}^2+16{f_2}^2+{f_3}^2-8f_1f_2-8f_2f_3-2f_1f_3\right]\ll 1 \qquad (12)$$
$$\beta\approx\frac{1}{z_m}\left[2f_1-8f_2+f_3\right]\ll 1$$

$$\begin{aligned} Det &= Q^2\alpha+Q\beta+1 \\ &\approx 1\pm\varepsilon>0 \qquad (0<\varepsilon\ll 1,\ \because 0\le Q\le 1) \\ \therefore Det &> 0\ (\forall z_r,\theta) \end{aligned} \qquad (13)$$

Consider the feasibility of these solutions. Considering *p(θ)*, *q(θ)* « 1, it is clear that a solution with a negative determinant is not a valid solution. Therefore, if appropriate constraints are placed on δ_{ij} and z_m, Eq. (13) always provides only a single real solution.

$$\begin{aligned} z_r &= \frac{-B\pm\sqrt{Det}}{2A}=\frac{-1-Qq(\theta)\pm\sqrt{Det}}{2Qp(\theta)} \\ &\approx\frac{-1-Qq(\theta)\pm 1}{2Qp(\theta)} \end{aligned} \qquad (14)$$

$$\begin{aligned} z_{r1} &= \frac{-B+\sqrt{Det}}{2A}=\frac{-Qq(\theta)}{2Qp(\theta)}=-\frac{q(\theta)}{2p(\theta)} \\ z_{r2} &= \frac{-B-\sqrt{Det}}{2A}=\frac{-2-Qq(\theta)}{2Qp(\theta)} \\ \therefore z_r &= z_{r1}\quad (\because |z_{r2}|\gg 1) \end{aligned} \qquad (15)$$

These calculations, however, are based on the assumption that the denominator *A* in Eq. (14) is not zero. If *A* is close to zero in Eq. (10), the solution can be calculated simply as shown in Eq. (16). Physically, the case when the determinant is close to zero means that the compensation curve along the *z* axis is approximated by a first-order polynomial.

$$z_r=-\frac{C}{B}=\frac{z'-Qr(\theta)}{1+Qq(\theta)}\quad (if\ A\approx 0) \qquad (16)$$

Therefore, the general solution of the forward kinematics considering compensation is arranged as follows.

$$z_r=\begin{cases}\dfrac{z'-Qr(\theta)}{1+Qq(\theta)} & (A\approx 0)\\[2ex] \dfrac{-B+\sqrt{Det}}{2A} & (A\neq 0)\end{cases} \qquad (17)$$

where

$$\begin{aligned} &A=(ax^2)p(\theta),\qquad B=1+(ax^2)q(\theta) \\ &C=(ax^2)r(\theta)-z',\ Det=B^2-4AC \\ &Q=ax^2,\ a=1/x_m^2 \\ &p(\theta)=\left(2f_1(\theta)-4f_2(\theta)+2f_3(\theta)\right)/z_m^2 \\ &q(\theta)=\left(-3f_1(\theta)+4f_2(\theta)-f_3(\theta)\right)/z_m \\ &r(\theta)=f_1(\theta) \\ &f_i(\theta)=\sum_{k=1}^{7}a_{ki}\theta^{7-k}\quad (i=1,2,3,4) \\ &a_{1i}=4\left(-7\delta_{i1}+8\delta_{i2}-9\delta_{i3}+8\delta_{i4}\right)/9\pi^6 \\ &a_{2i}=16\left(\delta_{i3}-\delta_{i1}\right)/9\pi^5 \\ &a_{3i}=\left(47\delta_{i4}-64\delta_{i3}+81\delta_{i2}-64\delta_{i1}\right)/9\pi^4 \\ &a_{4i}=32\left(\delta_{i1}-\delta_{i3}\right)/9\pi^3 \\ &a_{5i}=2\left(-5\delta_{i4}+16\delta_{i3}-27\delta_{i2}+16\delta_{i1}\right)/9\pi^2 \\ &a_{6i}=16\left(\delta_{i3}-\delta_{i1}\right)/9\pi \\ &a_{7i}=\delta_{i2} \end{aligned}$$

Because inverse kinematics considering compensation is the actual *z* position z' calculation process from the target position z_r, it is very simple. That is, if z_r, *x* and θ are given, z' can be easily calculated by Eq. (9).

4. EXPERIMENTAL RESULTS

4.1 Experimental Results for High-Precision Continuous Motion Interpolation

To verify the effectiveness of the proposed interpolation algorithm, continuous-motion experiments using both the existing and proposed algorithms were performed on an LCD glass-handling robot with a SCARA arm as shown in Fig. 9 (LTR-AS7CL1). The detail specifications are summarized in Table 1. The *z* axis of the robot is composed of two equal linkages. Because the arm takes a SCARA form, it has an efficient handling motion within the limited robot work range. In addition, it has a traverse axis at the base for rapid positioning between processing equipment and glass buffers.

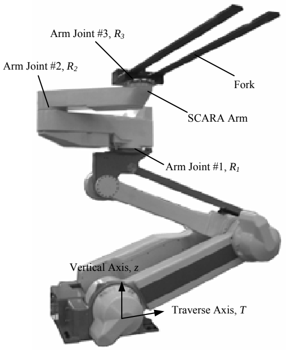


Fig. 9 An LCD handling robot with a SCARA arm. (LTR-AS7CL1, z_{max} = 2200 mm)

Fig. 10 shows the test path for this experiment. The path, which starts at p_s and finishes at p_f, is one that could be applied to glass transfer between two glass buffers. It is composed of several linear and circular paths and is executed successively. (LP1-CP1-LP2-CP2-LP3) The velocity of linear and circular paths is 1000 mm/s, and acceleration, deceleration and overlap time are 0.8 s. Although optimal continuous motion with arbitrary overlap length is possible using the proposed method, acceleration, deceleration and overlap time were set equal for comparison with the existing algorithm.

Table 1 Specifications of LTR-AS7CL1.

Item	Specification
degree of freedom	4 DOF / 5 Axis
max. glass size	591 × 1041 mm
work range (max. velocity)	z: 2200 mm (1500 mm/s) R_1: 345° (210 °/s) R_2: 300° (210 °/s) R_3: 330° (250 °/s) T : 5000 mm (1300 mm/s)
repeatability	± 0.3 mm
payload	25 kgf/arm
weight	700 kgf
cleanliness[a]	Class 4

[a]Cleanliness is defined according to ISO 14644. From this standard, Class N means that 10^N of particles with the size of 0.1 μm exist in 1 m^3.

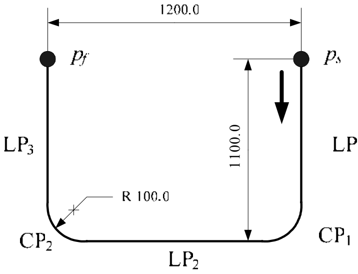
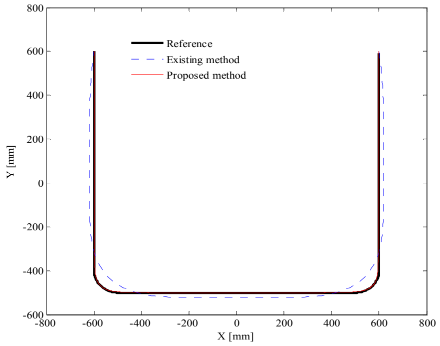


Fig. 10 Test path for continuous motion experiment.

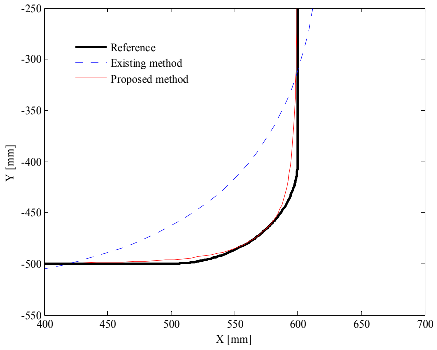
The experimental results are shown in Fig. 11. In this figure, (a) shows interpolation results in the whole test path, (b) shows magnification results of first corner part (CP1), and (c) shows the path error as a function of time.

Although the cycle time (4.34 s) of the proposed method is equal to that of the previous method, the path errors are obviously different. That is, in the existing method, a large path error is generated as soon as motion starts, and it increases to 20 mm in the linear paths (LP1, LP2, LP3) and to -50 mm in the circular paths (CP1, CP2).

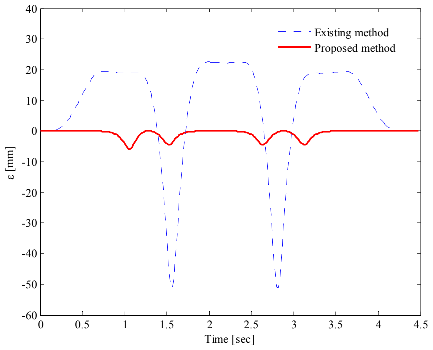
However, the proposed method has no path error in regions other than the overlap region where the path error has a maximum value of 6 mm. Therefore, the average path error of the proposed method is reduced by 91% without increasing the cycle time. Table 2 shows the maximum, p-p, rms value and reduction rate in each interpolation method.



(a) Interpolation results in the whole test path



(b) Magnification results of first corner part (CP1)



(c) Path error as a function of time

Fig. 11 Experimental results for continuous motion interpolation.

As described in 2.2, because acceleration and deceleration time must be the same as overlap time, the existing continuous motion method is seldom used in practice. Since the robot work range is limited in an actual manufacturing line, motion overlap region is also limited.

Table 2 Comparison of interpolation results.

	Existing method	Proposed method	Reduction rate
$\lvert\varepsilon_{max}\rvert$	51.050 mm	5.974 mm	88.3%
$\varepsilon_{p\text{-}p}$	73.566 mm	5.975 mm	91.9%
ε_{rms}	19.034 mm	1.638 mm	91.4%

In order to obtain the maximum reduction of the robot cycle time, therefore, the user must be able to set acceleration, deceleration and overlap times independently, and overlap lengths, rather than overlap times, should be used as the overlap variables. Furthermore, a specific motion must be able to overlap other motions by different overlap lengths. When the proposed interpolation method is applied using a dual acceleration/deceleration filter, all of these are possible.

An optimal continuous motion experiment using the proposed interpolation method was performed to verify the cycle time reduction effect compared with single motion. The test path for this experiment was a standard motion of the SCARA handling robot (glass acquisition→transfer→injection→return to the start position). Considering the robot's working environment, the continuous function was not applied to the entire motion, and the overlap length was adjusted optimally at the possible points.

Fig. 12 shows velocities of the first joint of the SCARA arm as a function of time for both optimal continuous motion and single motion. This figure shows that total cycle time (16.375 s) of the optimal continuous motion considering the robot's environment is 3.032 s less than that of the single motion (19.407 s); i.e., a 15.6% reduction of the robot's cycle time. This optimal continuous motion was programmed on site, in consultation with the robot's motion programmer, and is currently used in the seventh-generation LCD manufacturing line. Therefore, this result has directly contributed to an improvement in productivity.

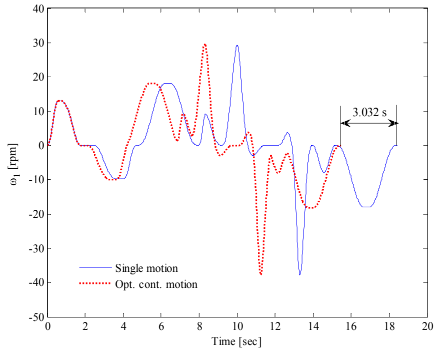


Fig. 12 Velocities of the first joint of the SCARA arm as a function of time

4.2 Experimental Results for Real-Time Static Deflection Compensation

The experiment for the real-time static deflection compensation algorithm was performed on the robot with linkage type *z* axis in Fig. 1 (a) (LTR-AD7FL1). Two arms for glass-handling were prepared, and the detail specifications are summarized in Table 3.

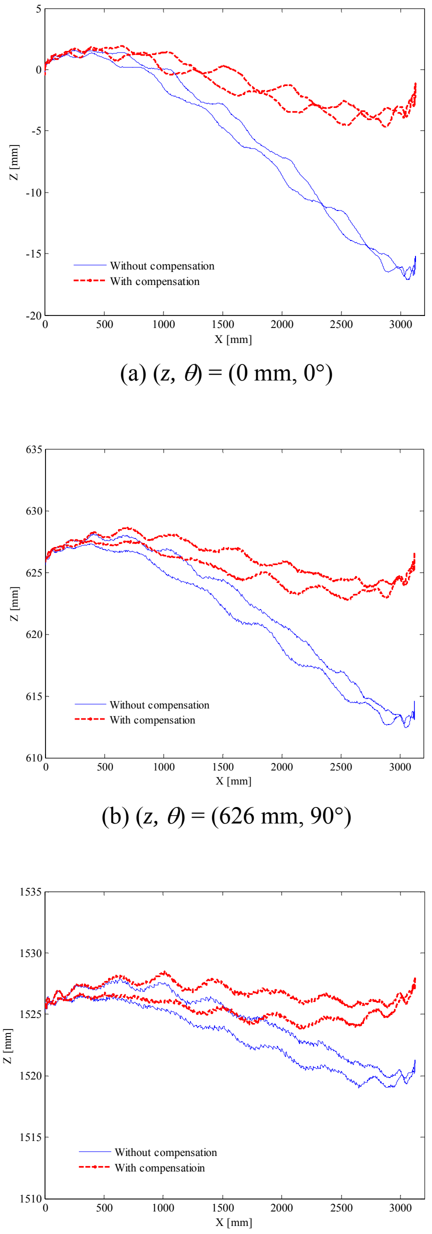
To verify optimal compensation effects regardless of robot position, experiments were performed at three different positions; (z,θ) = (0 mm, 0°), (626 mm, 90°) and (1526 mm, -90°), respectively. In addition, the robot arm moved back and forth with a maximum velocity of 3.2 m/s over a distance of 3.12 m. Then, *P* position in Fig. 1 was measured in real time during motion using a real-time 3D position measuring instrument (Laser Tracker, FARO).

Table 3 Specifications of LTR-AD7FL1.

Item	Specification
degree of freedom	4 DOF / 5 Axis
max. glass size	1870 × 2200 mm
work range	z : 2400 mm (1100 mm/s)
(max. velocity)	θ: 340° (180 °/s)
	x_1: 3697 mm (3200 mm/s)
	x_2 : 3697 mm (3200 mm/s)
	T : 7000 mm (1500 mm/s)
repeatability	± 0.3 mm
payload	80 kgf/arm
weight	1450 kgf
cleanliness	Class 4

Fig. 13 shows the experimental results for each position. Because these are not the results in stationary positions, these include arm vibration as well as static deflection. In these figures, the real-time static deflection compensation method shows an average of 60% reduction in vertical path error compared with the non-compensation method, regardless of *z*, θ position. Naturally, as described in 3.2, the *z* position that the user recognizes does not change during motion because forward kinematics considering compensation is applied. Therefore, the user can program motion and robot teaching on site without considering the compensation effect. Table 4 shows vertical path accuracies measured at each position.

The proposed algorithms were also applied to the handling robot with dual-telescopic type *z* axis in Fig. 1 (b) (LTR-AD7FT0). While the experimental results showed the better performances because of the advantage in mechanical structure, the results are not included in this paper due to the limited space.



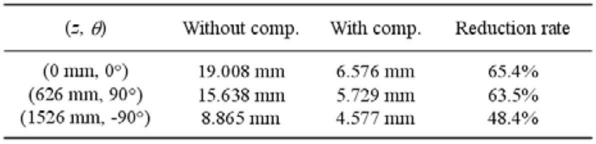
(a) (z, θ) = (0 mm, 0°)

(b) (z, θ) = (626 mm, 90°)

(c) (z, θ) = (1526 mm, -90°)

Fig. 13 Experimental results for real-time static deflection compensation.

Table 4 Comparison of vertical path accuracies



(z, θ)	Without comp.	With comp.	Reduction rate
(0 mm, 0°)	19.008 mm	6.576 mm	65.4%
(626 mm, 90°)	15.638 mm	5.729 mm	63.5%
(1526 mm, -90°)	8.865 mm	4.577 mm	48.4%

5. CONCLUSION

This paper proposes two advanced and practical algorithms for high-precision path generation of an LCD glass-handling robot.

First, a high-precision interpolation algorithm for continuous motion using a dual acceleration/deceleration filter is proposed. When the proposed algorithm is used with the robot performing continuous motion with different motion modes, such as PTP, LP and CP, an error-free path except overlap regions is possible regardless of acceleration, deceleration time, moving distance, velocity, etc. This is because each motion can carry out independent acceleration/ deceleration and inverse kinematics. In the case of an arbitrary number of continuous motions, only two acceleration /deceleration filters are sufficient, as a used filter may be reused. It is also possible to program optimal continuous motion with arbitrary overlap lengths, and each motion segment in the continuous motion can have different acceleration, deceleration and overlap times. Experimental results showed that when acceleration and deceleration time were set equal to the overlap time, the path error of the proposed algorithm was reduced, on average, by 91% of that of the existing method although both had the same cycle time. In addition, for optimal continuous motion composed of different overlap lengths and different acceleration and deceleration times for each motion segment, while considering the robot's working environment, a 15.6% reduction of cycle time compared with single motion was achieved without deterioration of path accuracy. This result has directly contributed to an improvement in productivity.

Second, real-time static deflection compensation was proposed for static deflection, which commonly occurs in LCD glass-handling robots. The compensation formula was defined as a function of vertical position z, rotational position θ and arm position x to provide an optimal compensation value at any robot position. It is defined as a function that can be differentiated at an arbitrary position to make joint velocity continuous even with user interrupts, such as pause and velocity override, etc. In addition, the z position that the user recognizes does not change during motion. Therefore, the user can program the robot's motion and teach it without considering the compensation because the forward kinematics considering compensation is included in the motion. This algorithm is very practical because it does not require additional sensors, measuring instruments or mechanical axes. Experimental results showed that the proposed method has an average of 60% reduction in vertical path error compared with the non-compensation method, regardless of robot position.

The above two proposed algorithms have been commercialized and are currently applied to seventh -generation handling robots which are being used in the seventh-generation LCD manufacturing line of Samsung Electronics.

REFERENCES

[1] T. C. Chang, R. A. Wysk, and H. P. Wang, *Computer-Aided Manufacturing*, Prentice Hall, New Jersery, 1991.

[2] O. Masory and Y. Koren, "Reference-word circular interpolators for CNC systems," *Trans. ASME, J. Eng. Ind.*, Vol. 104, pp. 400–405, Nov, 1982.

[3] D. S. Khalsa, "High performance motion control trajectory commands based on the convolution integral and digital Filtering," *Proc. Intelligent Motion Conf.*, pp. 54–61, 1990.

[4] D. I. Kim, J. W. Jeon, and S. Kim, "Software acceleration/deceleration methods for industrial robots and CNC machine tools," *Mechatronics*, Vol. 4, No. 1, pp. 37–53, 1994.

[5] S. K. Cheng and C. K. Tsai, "Method and system for path planning in Cartesian space," *U.S. Patent* 5 434 489, Jul. 18, 1995.

[6] K. S. Fu, R. C. Gonzalez, and C. S. G. Lee, *Robotics: control, sensing, vision and intelligence*, McGraw-Hill, Singapole, 1987.

[7] Y. Nakamura, *Advanced robotics: redundancy and optimization*, Addison-Wesley, New York, 1991.

[8] X. Yang, "Efficient circular arc interpolation based on active tolerance control," *Computer-Aided Design*, Vol. 34, pp. 1037-1046, 2002.

[9] W. L. Xu, S. K. Tso, and X. S. Wang, "Sensor-based deflection modeling and compensation control of flexible robotic manipulator," *Mechanism and Machine Theory*, Vol. 33, pp. 909-924, 1998.

[10] [10] A. Rouvinen and H. Handroos, "Deflection compensation of a flexible hydraulic manipulator utilizing neural networks," *Mechatronics*, Vol. 7, pp. 355-368, 1997.