Dynamic Analysis for Realistic Motion Simulation in Virtual World

Daisuke Tsubouchi*, Tetsuro Ogi** and Hirohisa Noguchi***

*Department of Mechanical Engineering, Keio University
3-14-1 Hiyoshi, Kouhoku-ku, Yokohama, 223-8522, Japan
tsubo@noguchi.sd.keio.ac.jp
**MVL Research Center, Telecommunications Advancement Organization of Japan / IML, Univirsity of Tokyo
***Department of System Design Engineering, Keio University

Abstract

In this study, in order to realize real-time motion simulation in virtual environment, a new efficient time integration scheme for finite element method was proposed. The proposed method is called 'iterative Newmark' method. In this method, it is not necessary to calculate the inverse of coefficient matrix like the explicit time integration scheme and has a stability criterion of the conventional Newmark method. This method was applied to several examples of the motion simulation, and the real-time and realistic motion was realized.

Keywords : Finite Element Method, Dynamics, Real-time Simulation, Virtual Reality

1. Introduction

In order to construct realistic virtual worlds, it is important to simulate the realistic movement of the objects. For instance, the virtual objects should be moved according to the law of motion and be deformed by applied force.

By using the finite element method, the realistic movement and deformation can be simulated. As for the static analysis using the finite element method in virtual world, several studies have been presented, especially in the field of haptic rendering[1]. However it is impossible to simulate the dynamic motions (deformation, translation, rotation etc.) of virtual objects by only using the static analysis. In order to realize these motions, dynamic analysis using the time integration scheme is indispensable. Although several time integration schemes have been introduced for dynamic analysis, these existing schemes are not suitable for virtual reality. Because the virtual reality applications require efficiency and stability for the interactive and real-time simulation, while the scientific analysis gives priority to the accuracy in the calculation.

In this study, a new efficient and stable time integration scheme was proposed to overcome these problems.

2. Methods

2.1. Overview of time integration schemes

In general, two kinds of time integration schemes are utilized for finite element method. One is an explicit time integration scheme such as central difference method, and the other is an implicit time integration scheme such as Newmark-β method. Although the explicit time integration scheme spends low computation cost as it does not need to calculate the inverse matrix, its stability is conditionally guaranteed with a small time increment. On the other hand, the implicit time integration has opposite features.

In this study, a new time integration method based on the Newmark-β method with the advantage of explicit methods was proposed.

2.2. Newmark-β method

The discrete equation of motion is formulated by the following equation[2].

$$\mathbf{M}_t\ddot{\mathbf{u}}_t + \mathbf{C}_t\dot{\mathbf{u}}_t + \mathbf{K}_t\mathbf{u}_t = \mathbf{F}_t \tag{1}$$

where $\mathbf{M}_t$ is the mass matrix, $\mathbf{C}_t$ is the viscous damping matrix, $\mathbf{K}_t$ is the stiffness matrix, $\mathbf{F}_t$ is the applied force vector and $\ddot{\mathbf{u}}_t$, $\dot{\mathbf{u}}_t$ and $\mathbf{u}_t$ are the acceleration, velocity and displacement vectors, respectively. When the elapsed time is $t+\Delta t$, Eq. (1) is represented by Eq. (2).

$$\mathbf{M}_{t+\Delta t}\ddot{\mathbf{u}}_{t+\Delta t} + \mathbf{C}_{t+\Delta t}\dot{\mathbf{u}}_{t+\Delta t} + \mathbf{K}_{t+\Delta t}\mathbf{u}_{t+\Delta t} = \mathbf{F}_{t+\Delta t} \tag{2}$$

When the Newmark-β method is used, $\mathbf{u}_t$ and $\dot{\mathbf{u}}_t$ are approximated as follows:

$$\dot{\mathbf{u}}_{t+\Delta t} = \dot{\mathbf{u}}_t + \frac{\Delta t}{2}\left(\ddot{\mathbf{u}}_t + \ddot{\mathbf{u}}_{t+\Delta t}\right) \tag{3}$$

$$\mathbf{u}_{t+\Delta t} = \mathbf{u}_t + \Delta t\dot{\mathbf{u}}_t + \frac{\Delta t^2}{2}\left\{(1-2\beta)\ddot{\mathbf{u}}_t + 2\beta\ddot{\mathbf{u}}_{t+\Delta t}\right\} \tag{4}$$

Eq. (3) and Eq. (4) are finite difference formulas. The parameter β determines the characteristics of stability and accuracy of this algorithm.

By substituting Eq. (3) and Eq. (4) for $\mathbf{u}_t$ and $\dot{\mathbf{u}}_t$ in Eq. (2), the following equation is obtained

$$\begin{aligned}&\left(\mathbf{M}_{t+\Delta t} + \frac{\Delta t}{2}\mathbf{C}_{t+\Delta t} + \beta\Delta t^2\mathbf{K}_{t+\Delta t}\right)\ddot{\mathbf{u}}_{t+\Delta t}\\ &= -\mathbf{C}_{t+\Delta t}\left(\dot{\mathbf{u}}_t + \frac{\Delta t}{2}\ddot{\mathbf{u}}_t\right)\\ &\quad -\mathbf{K}_{t+\Delta t}\left\{\mathbf{u}_t + \Delta t\dot{\mathbf{u}}_t + \frac{\Delta t}{2}(1-2\beta)\ddot{\mathbf{u}}_t\right\} + \mathbf{F}_{t+\Delta t}\end{aligned} \tag{5}$$

Assuming that $\mathbf{u}_t$, $\dot{\mathbf{u}}_t$ and $\ddot{\mathbf{u}}_t$ are known from the previous step of the calculations, $\ddot{\mathbf{u}}_{t+\Delta t}$ is determined by solving Eq. (5). And the $\dot{\mathbf{u}}_{t+\Delta t}$ and $\mathbf{u}_{t+\Delta t}$ are determined from Eqs. (3)-(4).

However, this method can hardly be applied to virtual reality applications, because it costs much computational time to calculate the inverse of coefficient matrix in Eq. (5).

On the other hand, the Newmark-β method has an advantage of unconditionally stability under the condition of $\beta \geq \frac{1}{4}$. In this study, we propose iterative Newmark method that has a stability criterion equivalent to the conventional Newmark method and does not need to calculate the inverse of coefficient matrix like the explicit time integration scheme.

2.3 Iterative Newmark method

2.3.1 Overview of iterative Newmark method

Moving the term of a stiffness matrix in the left hand side of Eq. (5) to the right hand side, Eq. (5) is rewritten as Eq. (6).

$$\begin{aligned}&\left(\mathbf{M}_{t+\Delta t} + \frac{\Delta t}{2}\mathbf{C}_{t+\Delta t}\right)\ddot{\mathbf{u}}^{(2)}_{t+\Delta t}\\ &= -\mathbf{C}_{t+\Delta t}\left(\dot{\mathbf{u}}_t + \frac{\Delta t}{2}\ddot{\mathbf{u}}_t\right) - \beta\Delta t^2\mathbf{K}_{t+\Delta t}\ddot{\mathbf{u}}^{(1)}_{t+\Delta t}\\ &\quad -\mathbf{K}_{t+\Delta t}\left\{\mathbf{u}_t + \Delta t\dot{\mathbf{u}}_t + \frac{\Delta t}{2}(1-2\beta)\ddot{\mathbf{u}}_t\right\} + \mathbf{F}_{t+\Delta t}\end{aligned} \tag{6}$$

In this equation, the acceleration term in the right hand side is represented by $\ddot{\mathbf{u}}^{(1)}_{t+\Delta t}$, and the acceleration term in the left hand side is represented by $\ddot{\mathbf{u}}^{(2)}_{t+\Delta t}$. $\ddot{\mathbf{u}}^{(1)}_{t+\Delta t}$ means the first predictor, and $\ddot{\mathbf{u}}^{(2)}_{t+\Delta t}$ the second. Then the n th predictor (n th iteration) can be represented using the (n-1) th after predictor as follows:

$$\begin{aligned}&\left(\mathbf{M}_{t+\Delta t} + \frac{\Delta t}{2}\mathbf{C}_{t+\Delta t}\right)\ddot{\mathbf{u}}^{(n)}_{t+\Delta t}\\ &= -\mathbf{C}_{t+\Delta t}\left(\dot{\mathbf{u}}_t + \frac{\Delta t}{2}\ddot{\mathbf{u}}_t\right) - \beta\Delta t^2\mathbf{K}_{t+\Delta t}\ddot{\mathbf{u}}^{(n-1)}_{t+\Delta t}\\ &\quad -\mathbf{K}_{t+\Delta t}\left\{\mathbf{u}_t + \Delta t\dot{\mathbf{u}}_t + \frac{\Delta t}{2}(1-2\beta)\ddot{\mathbf{u}}_t\right\} + \mathbf{F}_{t+\Delta t}\end{aligned} \tag{7}$$

In this equation, since $\mathbf{M}_{t+\Delta t}$ and $\mathbf{C}_{t+\Delta t}$ can be diagonalized, $\ddot{\mathbf{u}}^{(n)}_{t+\Delta t}$ can be obtained without calculating the inverse matrix. By iterating this calculation until convergence, $\ddot{\mathbf{u}}_{t+\Delta t}$ is finally obtained.

2.3.2 Prediction of the acceleration term

In this method, it is important to predict appropriate initial acceleration term $\ddot{\mathbf{u}}^{(1)}_{t+\Delta t}$, If an inappropriate initial predictor were given, $\ddot{\mathbf{u}}_{t+\Delta t}$ could not be converged.

However, if we can choose a valid prediction, the solution may be converged in smaller number of iterations. In order to realize a real-time simulation, calculation performance more than 40 Hz is required.

When the time step Δt is enough small, we can assume approximately that the acceleration changes linearly. Therefore, in this method, the first prediction was given as follow:

$$\ddot{\mathbf{u}}^{(1)}_{t+\Delta t} = 2\ddot{\mathbf{u}}_{t} - \ddot{\mathbf{u}}_{t-\Delta t} \tag{8}$$

This assumption would be appropriate on the condition that the acceleration changes slightly.

2.3.2 Convergence of iteration

In this section, we discuss the convergence condition of $\ddot{\mathbf{u}}^{(n)}_{t+\Delta t}$. By subtracting Eq. (7) from the equation for (n+1), Eq (9) is given.

$$\begin{aligned}&\ddot{\mathbf{u}}^{(n+1)}_{t+\Delta t} - \ddot{\mathbf{u}}^{(n)}_{t+\Delta t}\\ &= -\beta\Delta t^2\left(\mathbf{M}_{t+\Delta t} + \frac{\Delta t}{2}\mathbf{C}_{t+\Delta t}\right)^{-1}\mathbf{K}_{t+\Delta t}\left(\ddot{\mathbf{u}}^{(n)}_{t+\Delta t} - \ddot{\mathbf{u}}^{(n-1)}_{t+\Delta t}\right)\end{aligned} \tag{9}$$

The convergence condition is given as follows:

$$\lim_{n\to\infty}\left(\ddot{\mathbf{u}}^{(n)}_{t+\Delta t} - \ddot{\mathbf{u}}^{(n-1)}_{t+\Delta t}\right) = \mathbf{0} \tag{10}$$

$$\lim_{n\to\infty}\left(\ddot{\mathbf{u}}^{(n)}_{t+\Delta t}\right) = \ddot{\mathbf{u}}_{t+\Delta t} \tag{11}$$

Therefore, as for the convergence condition in this method, the following equation is finally obtained.

$$max\left|\lambda_i\right| \le 1 \tag{12}$$

where λ_i is the i th eigenvalue of the coefficient matrix in the right hand side of Eq. (9).

3. Experiment

3.1 Hardware

In order to evaluate the effectiveness of this proposed iterative Newmark method, we implemented this algorithm in several kinds of motion simulations of the object in the virtual environment. We used a workstation (SGI Octain R12000 300MHz×2, IRIX 6.5).

3.2 Judgement of convergence

Based on Eq. (10), we regarded that $\ddot{\mathbf{u}}^{(n+1)}_{t+\Delta t}$ is converged to $\ddot{\mathbf{u}}_{t+\Delta t}$ on the following condition.

$$\ddot{\mathbf{u}}^{(n+1)}_{t+\Delta t} - \ddot{\mathbf{u}}^{(n)}_{t+\Delta t} \le 10^{-5} \tag{13}$$

If $\ddot{\mathbf{u}}_{t+\Delta t}$ is not converged after more than 100 iterations, the time step Δt is reduced in half in order to avoid divergence.

3.3 Analysis model

As for the analysis model, simple spring model was used. Fig. 1 shows the example of the analysis model. This model consists of springs, dampers and masses, and the springs are intersected partially in order to represented a share stiffness. Though the model is not completely accurate for the purpose of the strict scientific analysis, it may be used to simulate the deformation of the object in the virtual world.

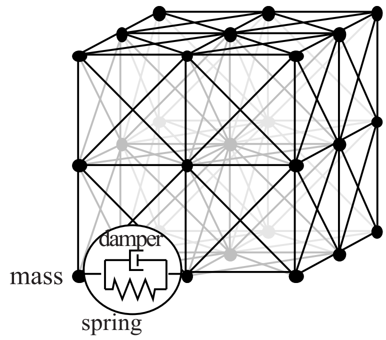

Fig. 1 Analysis model

3.4 Boundary condition

In this experiment, the proposed method described in the previous section was applied to three types of movements, such as deformation, translation and rotation. Fig. 2 shows the condition for the deformation test. Fig. 3 shows the condition for the movement test that includes the translation and the deformation. Fig. 4 shows the condition for the movement that includes deformation, translation and rotation. We adjusted the virtual world and the real world by sleeping computation, because the calculation time is much faster than the time in the real world in this example.

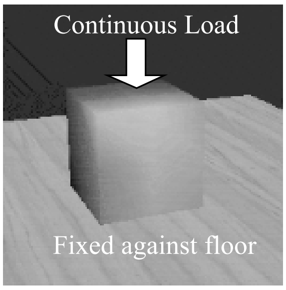

Fig. 2: Deformation test

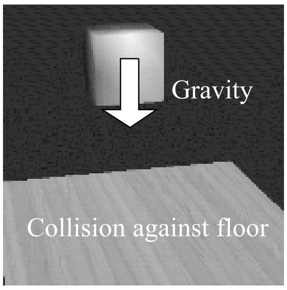

Fig. 3: Translation and deformation test

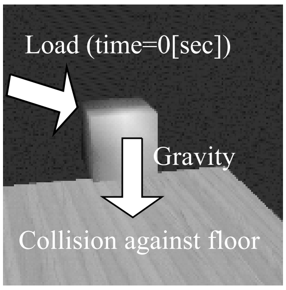

Fig. 4: Rotation, translation and deformation test

3.5 Motion simulation

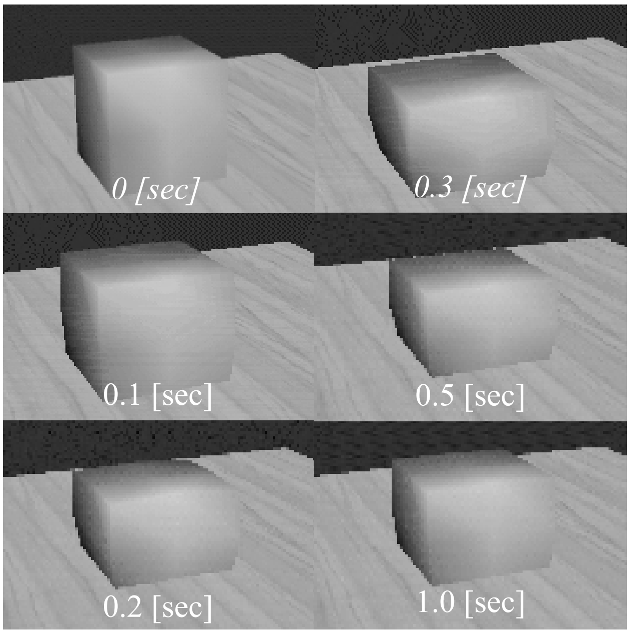

Fig. 5 Movement of deformation

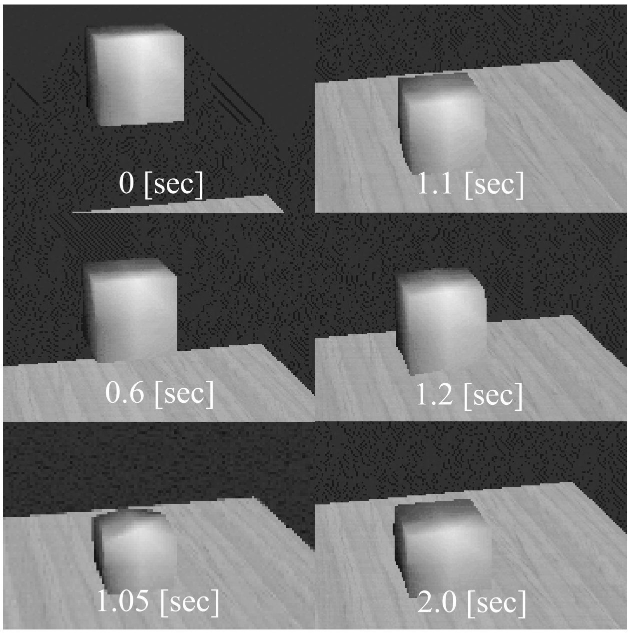

Fig. 6 Movement of deformation and translation

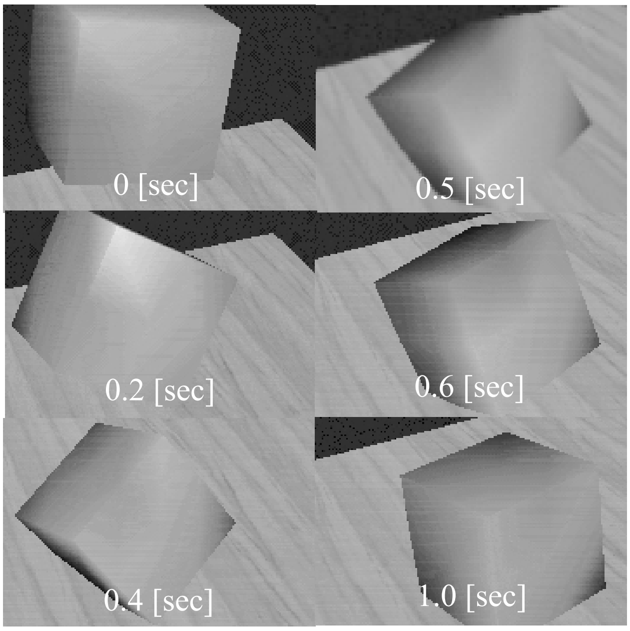

Fig. 7 Movement of deformation, translation, and rotation

In the deformation test, the load was applied to the object from the upper side continuously in time. The movement of the object simulated by the proposed method was shown in Fig. 5. In Fig. 5, after the vibrations, the equilibrium state was obtained between applied forces and the reaction forces.

In this experiment, since the computation time for one step was shorter than Δt (=0.001[sec]), a real-time simulation was realized. This spring model was simply assumed to be the linear, however, the simulated movement of the object seemed to be natural.

In the translation and deformation case, the object was dropped according to the gravity. In Fig. 6, the simulated movement of the object is shown. In this experiment, the simulated movement seemed to be natural, though the strict physical model, such as the friction between the object and the floor, was not implemented.

In the deformation, transition and rotation case, the object was dropped with the applied force from the side direction. In Fig.7, the simulated movements of the virtual object was shown. In this experiment, the rotation on the floor is somewhat unnatural. For example, the objects rotated to the wrong direction in this model, because we didn't take the friction against the floor and the balance of the angular moment into account. In order to achieve more realistic motion including rotation, the strict physical model should be required. However, in this experiment, the real-time calculation was achieved by using the proposed method.

3.6 Comparison of results by Newmark-β method and iterative Newmark method

We compared the average computation costs for solving $\ddot{\mathbf{u}}_{t+\Delta t}$ the both in the Newmark-β method and the iterative Newmark method. Table 1 shows the result. The numerical experiments were conducted under the same condition of deformation simulation in Fig. 2. The analysis model shown in Fig. 1 was used, and the number of nodes was varied from 8 to 64. In the both cases, the computation costs were increased according to the increase of the number of nodes. And, in any case, the computation cost in the iterative Newmark method was smaller than the Newmark-β method.

Table 1: Computation cost for the solution of $\ddot{\mathbf{u}}_{t+\Delta t}$

Node	Newmark-β method [ms]	iterative Newmark method [ms]
8	1.732	0.051
27	2.093	0.219
64	10.065	0.597

3.7 Iteration number and computation time

We simulated several movements by changing the time step Δt and the iteration number to the convergence was counted. In this test, the analysis model shown in Fig.1 was used and the number of nodes was 27. Fig. 8, Fig. 9 and Fig. 10 show the iteration numbers of the calculation for each test. In these figures, the iteration numbers for Δt =0.01[s] and Δt =0.001[s] were compared among these movement tests.

From these results, the number of the iteration was obviously decreased when the fraction size of the time step was small. In addition, it was noted that the iteration number became large, when the acceleration was changed suddenly at the collision point against the floor as shown in Fig. 9.

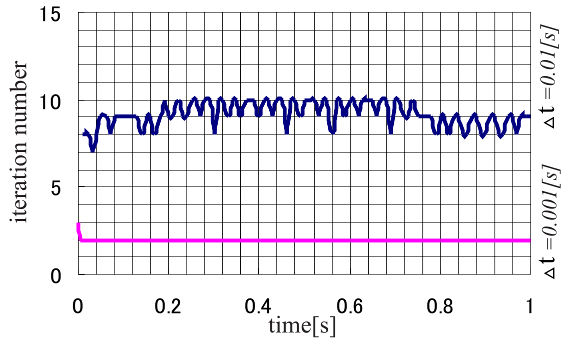

Fig. 8: Iteration numbers of the case in Fig. 2

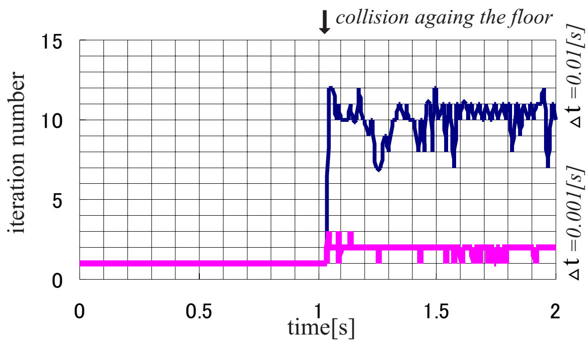

Fig. 9: Iteration numbers of the case in Fig. 3

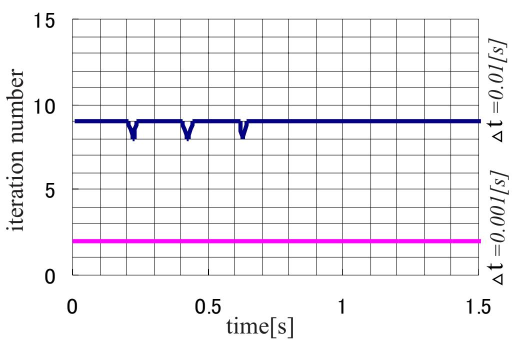

Fig. 10: Iteration numbers of the case in Fig. 4

Furthermore, we examined the computation cost and the iteration number in detail. The analysis model of Fig. 1 was used, and the number of nodes was changed from 8 to 64.

In Fig. 11, we can see that, in the case of Δt=0.01[s], the computation cost was increased according to the increase of the node number. Then we compared the computation cost for each iteration (Fig. 12) and the average iteration number (Fig. 13).

In Fig. 12, the computation cost for each iteration was almost the same between in the case of Δt=0.01[s] and Δt=0.001[s]. The computation cost increased linearly, when the analysis model were bigger.

However, we found from Fig. 13 that the iteration number became larger when Δt=0.01[s], and the total computation time depended on the iteration number.

In order to realize a real-time simulation, the total computation time (iteration number times each computation time) must be shorter than the time step. Therefore, we must carefully examine the relation between Δt and the iteration number to realize the real-time simulation. In addition, we must also examine the more appropriate prediction method of the acceleration term.

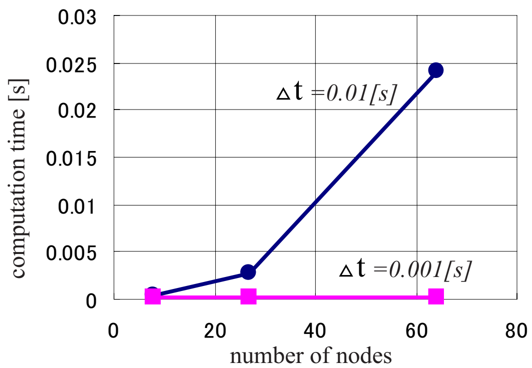

Fig. 11 Average Computation time to reach the solution of the next time step $\ddot{\mathbf{u}}_{t+\Delta t}$

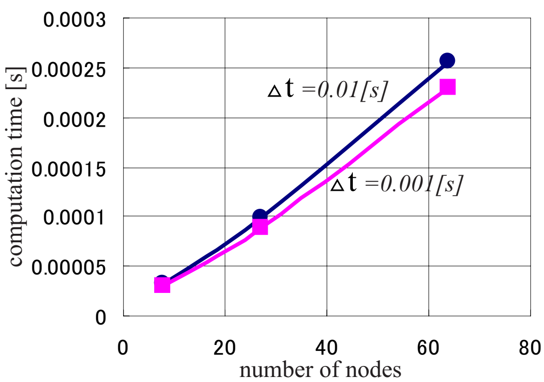

Fig. 12 Computation time to reach the solution of the next iteration $\ddot{\mathbf{u}}^{(n)}_{t+\Delta t}$

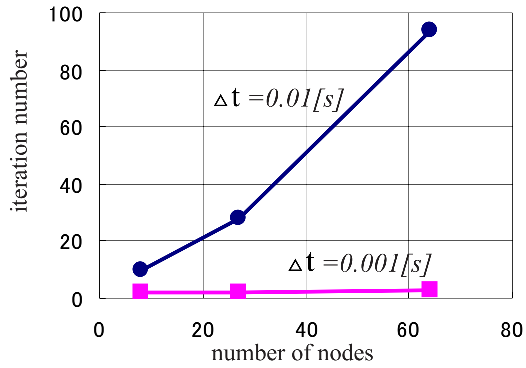

Fig. 13 Iteration number to reach the solution of the next time step $\ddot{\mathbf{u}}_{t+\Delta t}$

4. Conclusions

In this study, in order to realize real-time deformation analysis in virtual reality, a new efficient time integration scheme for finite element method was proposed.

This method named 'iterative Newmark method' is suitable for virtual reality applications, because this method has the result by computational efficiency and stability, compared with the Newmark-β method.

This method was applied to several motion simulations such as deformation, movement and rotation of the object, and the real-time calculation was achieved by using the proposed method.

References

1. Koichi, HIROTA., Toyohisa, KANEKO.: Representation of Soft Objects in Virtual Environment, ICAT 98, pp.59-62(1998).
2. Belytschko,T. and Thomas,T.J.R.: COMPUTATIONAL METHODS FOR TRANSIENT ANALYSIS,NORTH-HOLLAND,Chap.2 (1983)