2M-4

# Parallel Animation Processing with 3-Variable Transfer Functions Using Object-Oriented Language

Naoya Sato, Kazuyoshi Mori
School of Computer Science and Engineering
The University of Aizu, Aizu-Wakamatsu 965-8580, JAPAN
Voice: [+81](242)37-2615; Fax: [+81](242)37-2747
Email: {m5081203,k-mori}@u-aizu.ac.jp

**Keywords:** Multidimensional Signal Processing, Animation Processing, Transfer Function, Object-Oriented Programming, Parallel Computation.

## 1 Introduction

Recently, the computer graphics load is increasing gradually. To make animation processing effecitive, we have developed a high performance parallel computation environment for the animation processing by using 3-variable transfer functions, PAPTOOL (Parallel Animation processing Package with Transfer function using Object-Oriented Language). The PAPTOOL has been written by an object-oriented language, JAVA. It is efficient in parallel computing environment[2].

The purpose of this study is to evaluate the parallel computation environment of the PAPTOOL for animation processing using transfer functions[1].

## 2 PAPTOOL

The PAPTOOL is a package consisting of (i) A Client, (ii) An Allocator, and (iii) Single-Input-Single-Output(SISO) simulators. They are softwares and are assigned to computers as shown in Figure 1. This provides a parallel computing environment with the client-server model.

The Client provides a user interface. The Server shown in the right side of Figure 1 composed of the Allocator and SISO simulators.

The Allocator receives an animation vector and a system description from the Client, which have been provided by user. Then the Allocator calculates the transfer matrix. Then the Allocator allocates the following to the SISO simulators: (i) Entries of the transfer matrix (i.e. transfer functions), and (ii) Elements of the animation vector received from the Client (i.e. animation signal).

Each SISO simulator calculates the output animation signal from the input animation signal and the transfer function given by the Allocator. The Allocator will collect the output animation signals of the SISO simulators, reconstructs an animation vector and sends it back to the Client.

## 3 Evaluations

The computers we used for this evaluation are Sun Java Workstation W1100z.

We evaluate following aspects: (i) Turnaround time, (ii) Amount of memory that the process spent, and (iii) Total number of CPU-seconds that the process spent.

The input animation vector $\mathbf{f} = [f_1\ f_2\ f_3]^t$ is given as $f_i = \rho\tau_i \sum_{k=1}^{5} \sigma_k \quad (i = 1, 2, 3)$, where

$$\rho = (1 + d_1 + d_2)^5,$$

$$\sigma_1 = (1 - 0.99 d_1 d_3)^{-1},\ \sigma_2 = (1 - 0.99 d_2 d_3)^{-1},$$

$$\sigma_3 = (1 - 0.99 d_1 d_2 d_3)^{-1}, \sigma_4 = (1 - 0.99 d_1^2 d_2 d_3)^{-1},$$

$$\sigma_5 = (1 - 0.99 d_1 d_2^2 d_3)^{-1},$$

$$\tau_1 = d_1^{10} d_2^{100},\ \tau_2 = d_1^{50} d_2^{50},\ \tau_3 = d_1^{100} d_2^{10}.$$

The effects on animations are shown in Figure 2.

The transfer matrix $T = (t_{ij})$ is given as follows:

$$t_{1,1} = \frac{1 + d_3^{50} + d_3^{100}}{1 - 0.9 d_3}, \quad t_{1,2} = \frac{d_3^{50} + d_3^{100}}{1 - 0.9 d_3},$$

$$t_{1,3} = \frac{d_3^{100}}{1 - 0.9 d_3}, \quad t_{2,1} = \frac{d_3^{100}}{1 - 0.9 d_3},$$

$$t_{2,2} = \frac{1 + d_3^{50} + d_3^{100}}{1 - 0.9 d_3}, \quad t_{2,3} = \frac{d_3^{50} + d_3^{100}}{1 - 0.9 d_3},$$

$$t_{3,1} = \frac{d_3^{50} + d_3^{100}}{1 - 0.9 d_3}, \quad t_{3,2} = \frac{d_3^{100}}{1 - 0.9 d_3},$$

$$t_{3,3} = \frac{1 + d_3^{50} + d_3^{100}}{1 - 0.9 d_3}.$$

To evaluate the effects of distributed parallel computing, the results from parallel computation was compared with that from non-parallel computation. We also change the number of computers that can use in parallel. In addition, performances were compared by changing resolutions of animations. To simplify this experiment, we assume that horizontal and vertical pixel sizes are same, and that all animations are fixed to only 200 frames. In the graphs shown hereafter, horizontal axis denotes the number of horizontal or vertical pixels, where horizontal and vertical pixel sizes are equal.

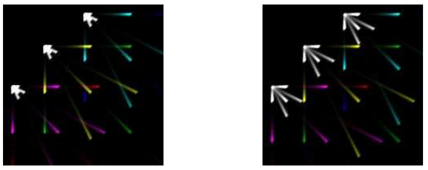


Figure 2: Images of result animation in evaluations

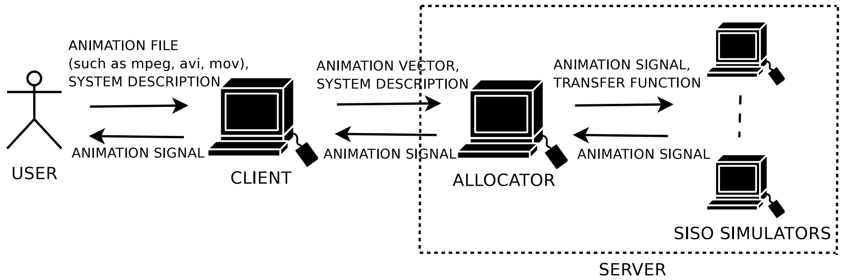



Figure 1: PAPTOOL environment

### Results and Discussion

**Resolution** Figure 3(a) shows (i) the turnaround time in parallel computation environment with multiple computers and (ii) the turnaround time in non-parallel computation environment with a computer. The turnaround time in the parallel computation environment is shorter than that of the non-parallel computation environment. The non-parallel computation was given up over 400 pixels because the swap began so that the turnaround time increased extremely.

**Amount of Memory** Figure 3(b) shows the amounts of memories under the parallel computation environment and under the non-parallel computation. In the non-parallel computation environment the amount of memory goes over 700MB at the 300 pixels. The computer used in the evaluation has only 1GB memory. Thus, in the non-parallel computation environment, the evaluation at the 400 pixels met the swap so that the execution time increased extremely.

**The Number of Computers** The turnaround time with respect to the number of computers is also investigated. The horizontal and vertical pixel sizes are 200 pixels. These results are shown in Figure 3(c). In the figure, the number of computers shows the number of SISO servers. We have evaluated ten times for each number of computers. Transfer functions are assigned to SISO servers randomly. The top and the bottom of vertical bars stands for the maximal and the minimal turnaround time, respectively. The middle cross shows the average turnaround time. The turnaround time decreases as the number of computers increases.

## 4 Conclusions

In this paper, the effects of the animation processing on animation signals have been evaluated with the PAPTOOL. Since animations are considered as transfer functions, effective parallel computation environment can be obtained easily. By using Java language as object-oriented language, the environment has high portability, extensibility, and security.

When the animations is given as 3-variable transfer functions, the load of computers can be distributed by using factorization. From the distribution, we can provide good performance environment.

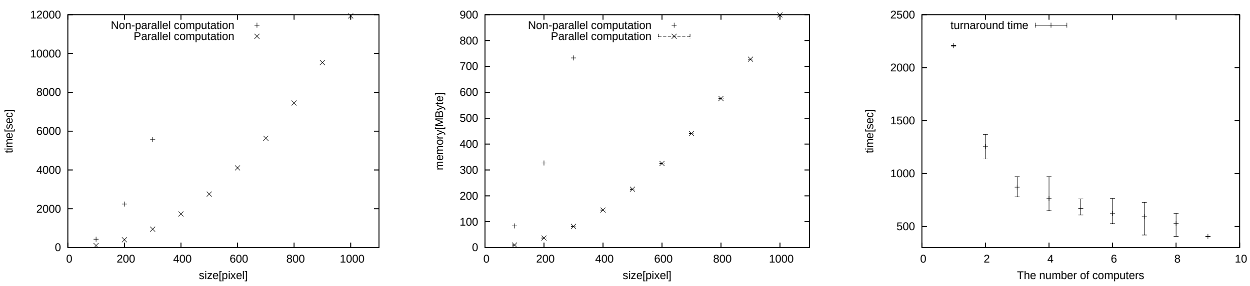
## References

[1] T. Higuchi, *Introduction to signal processing*, Japan, Shokodo, 1986.

[2] S. Ragsdale, *PARALLEL PROGRAMMING*, MacGraw-hill, (1993).

[3] Hartmut Schröder, Holger Blume, *One-And-Multidimensional Signal Processing: Algorithms and Applications in Image Processinses for One and Multi-Dimensional Signal Processing*, JOHN WILEY & SONS, LTD, (2000).

[4] N. Sato and K. Mori: Parallel Animation Processing with Transfer Functions Using Object-Oriented Language, *Proceedings of the Eighth International Conference on Human and Computers*, pp. 61/66, Aug. (2005).




(a)Turnaround time w.r.t. size

(b)Amount of memory of SISO simulators

(c)Turnaround time w.r.t. the number of computers

Figure 3: Evaluation