資　料

会話形グラフィック連続系シミュレータ*

臼井支朗** 浅見秀雄*** 池谷和夫**

Abstract

A digital programming language, based on simulating an analog computer rather than directly solving arithmetic and differential equations, has more ability for digital simulation of continuous systems. By eliminating the details associated with computer simulation (program development, card punching and debugging) one would be free to concentrate on the conceptual aspects of the simulation problem.

In this paper, much attention has been paid to improving man-machine interaction by using a block oriented language based upon functional blocks and extensive graphic capabilities.

The structure of a model to be simulated can be drawn in diagrammatic form with a light pen. The system automatically analyses the structure and sorts the order in which the functional blocks of the model have to be computed. The desired response appears on the same graphic screen during the computation and afterwards. Therefore, the system permits the user not familiar with programming technique to communicate with the computer through graphical representation during the study.

1. まえがき

近年，境界領域を含め広範な分野においてシミュレーション技法を用いた研究が急速に展開されてきている．その中でも連続系シミュレータは，アナログむき入力言語を用いた問題むきソフトウェアの一つとして開発が進められて，すでに 20 有余年の歴史とともにその変遷，性能比較を含め，各方面から解説されている[1),2)]．しかしながらこれらの言語のほとんどは，簡単とはいえシステム記述に際し各ブロックの入出力関係を表現するための言語の学習を必要としている．すなわち従来の連続系シミュレータは，そのほとんどが Fortran タイプの入力言語を用いた文形式のシステム記述などディジタル計算機固有のプログラミング技術を要求しているとともに，カード・リーダ，ライン・プリンタなどを入出力装置としたバッチ処理を前提としたものであり，その使用において多くのわずらわしさがあった．一方，図形入出力装置の充実・普及に伴い，グラフィックスを介した会話的処理が各方面で実用化されるに至り，シミュレーション言語も次第にその方向へ移行しつつある[3)]．

本論文では，言語としてアナログ計算機の演算ブロックに似た図形および接続線を用い，システム・ブロック・ダイアグラムをグラフィック上に記述し，シミュレーションの実行，表示・出力などすべての操作を会話的に行える連続系ディジタル・シミュレータについて述べる．これに類似したシミュレータは，CSMP を改良したミニコン・ベースのものが米国[4)]にあり，著者の一人がユーザとして使用したその経験をもとに実用性・使い易さを主眼にここに新たに設計・開発[5)]したものである．我が国においても類似したシミュレータの報告[6),7)]があるが，いずれも予備実験的なものであり，大型汎用計算機システムによりman-oriented

* Interactive Graphic Simulator for Continuous Systems by Shiro USUI, Kasuo IKEGAYA (Faculty of Engineering, Nagoya University), and Hideo ASAMI (Yokosuka Electrical Communication Laboratory, N.T.T.)
** 名古屋大学工学部電気学科
*** 日本電信電話公社横須賀電気通信研究所

に設計された本シミュレータは，その使い易さ，実用性から一つの新しい形態として今後充分評価されるものであると考えられる．すなわち，本シミュレータの特徴は，Fortran 言語はもちろん，その他のプログラミングに関する知識を全く必要とせず，すべてグラフィックスとの対話，すなわちブロック・ダイアグラムの視覚的提示を介し，その構造・パラメータをインタラクティブに修正・変更しながら問題解決に至るという，非常に効率よいシミュレーション研究を可能にする点にある．さらにその結果は，必要であればオンライン結合されているプロッタにより，鮮明な図形出力として直接得られるため，いわゆるバッチ処理による煩わしい作図プログラミングから解放される．本文ではその基本的構造，ブロック・ダイアグラム接続表作成アルゴリズムおよび全体の会話的機能を中心に述べる．

2. システム概要

開発したシミュレータは，アナログ計算機に似た演算ブロック図形を入力言語としている．ソフトウェアは会話的処理に適したインタプリティブ方式であり，シミュレーション・システムのブロック・ダイアグラム構造および各演算ブロックのパラメータを指定することにより通常の数値計算に相当する種々の問題を解くことができる．

2.1 基本ブロック

使用できる基本ブロックは，信号発生ブロック，演算ブロック，表示用スコープからなる 30 種類が定義されており，それぞれ最大 3 つの入力変数 (x, y, z)，1 つの出力変数 (u) および各ブロックを定義するに必要な最大 4 つのパラメータ (P_1, P_2, P_3, P_4) をもつ．なおパラメータは整数型，実数型および文字型のいずれかである．

信号発生ブロックには定数，周期関数（正弦波，矩形波，三角波および鋸歯状波），パルス，雑音（一様乱数，正規乱数），指数関数，ランプ関数がある．

演算ブロックには，一入力，二入力，三入力のものがあり一入力線形要素として係数器と符号変換器，非線形要素として sgn 関数，遅延，ヒステリシス，不感帯，飽和特性，クリッパー，オフセット，量子化器および関数演算（| |, $\sqrt{\ }$, sin, cos, $\tan^{-1}$, exp, $\log_e$, $\log_{10}$, tanh）がある．その他データ補間による関数発生器も用意されている．二入力要素は乗算器，除算器，零次ホールド，冪乗演算，ストッパーおよび表示用二現象スコープからなる．三入力要素には加算器，重みつき加算器，リレー，積分器およびユーザ・ブロックがある．ユーザ・ブロックは入出力変数 (x, y, z, u)，パラメータ (P_i, i=1～4) および時刻 (t) を引数とする Fortran 表現のサブルーチンにより定義できる演算ブロックで，Fortran 組込み関数を含めユーザ仕様の機能をもたせることができるとともにモデル内に何度も現われるサブ・システムをマクロ化することも可能である．

2.2 ダイアグラム・フェーズ

システム・ブロック・ダイアグラムは画面下部に表示されたこれらの基本ブロック・メニューをライトペン操作により接続線を用いて記述・構成する．この段階をダイアグラム・フェーズと呼ぶ (**Fig. 1**)．

〈ダイアグラム・フェーズの機能〉

1) ダイアグラムの作成・変更
2) ダイアグラムの消去 (CLEAR)
3) 標題の入力 (TITLE)
4) ブロック，接続線の消去指定 (ERASE)
5) 計算条件（計算時間間隔，表示時間間隔，最終時刻）の指定
6) 出力表示スコープのスケール指定
7) ブロックのパラメータ指定
8) ダイアグラム・リスト出力 (LIST)
9) ダイアグラムをディスクへ貯蔵 (FILF) およびその検索 (RET) (10 ページ)
10) ディスク上のダイアグラム・ファイルを MT へ書き込み (SAVE) およびその呼び出し (KEEP)
11) ダイアグラムの一時退避 (SHELT, RECALL)
12) ダイアグラムのプロット (COPY)

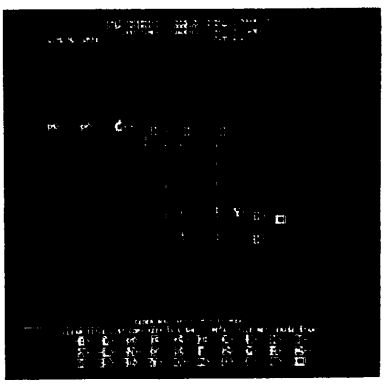

Fig. 1 Diagram phase: showing CEDER BOG LAKE ECOSYSTEM MODEL which is used as an example in the text.

13) 実行開始指令 (START)
14) ジョブ終了 (CLOSE)

2.3 アナリシス・フェーズ

実行指令 (START) をピック・アップすることによりブロック・ダイアグラム接続表の作成，計算順序付け，初期設定を自動的に行ったのち画面はスコープ画面に変わり出力が計算ステップ経過とともに表示される．これをアナリシス・フェーズと呼ぶ．

〈アナリシス・フェーズの機能〉

1) スコープのつながっている出力端の値を計算経過とともにグラフ表示
2) 出力値のライン・プリンタ出力指令
3) 実行停止 (PAUSE)，再開 (RESTART) 指令
4) パラメータ変更指令（あらかじめ指定されたきざみでパラメータを変更し時刻0から計算をくり返す．結果は重ねて表示される）(MODIFY)
5) 出力グラフの点線／実線切換え
6) グリッドの表示／消去指定
7) 出力グラフのプロット (COPY)
8) ダイアグラム・フェーズへ復帰 (RETURN)
9) ジョブ終了 (CLOSE)

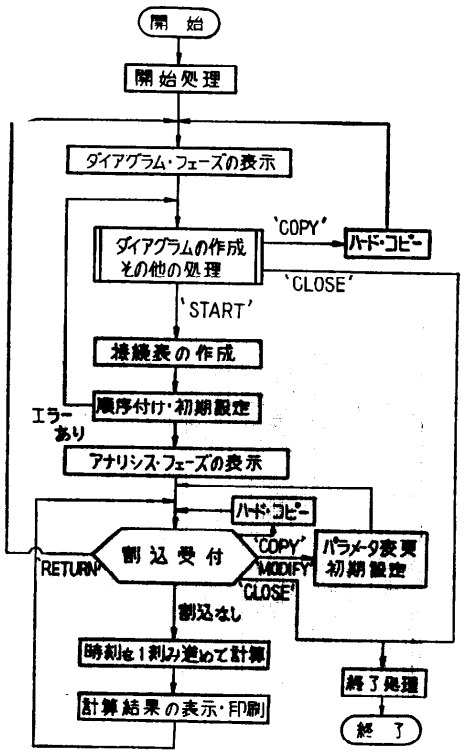

Fig. 2 Functional flow chart of our simulator.

この二つのフェーズを会話的に反復，くり返し実行することにより，与えられたモデル・シミュレーションや数値計算の解がグラフ出力される．**Fig. 2** に本システム全体の概略フロー・チャートを示す．

2.4 使用機器構成

使用した計算機システムは主記憶384 kBのFACOM 230-48，グラフィック・ディスプレイはリフレッシュ型F 6233 A，ディスク・ファイル (200 MB)，XYプロッタおよび MT 装置からなりこれらを GD コンソールのライトペン，キーボードからオンライン操作している．なおディスプレイ・バッファのメモリ・サイズは 32 kB である．オペレーティング・システムは，OS II/VS である．使用言語は Fortran および GSP (Graphic Subroutine Package) でほぼ 4000 ステップを要している．

3. ダイアグラム記述

3.1 ブロックの登録とブロック番号

ブロック・ダイアグラムの記述作成にはグラフィック・ディスプレイ装置のトラッキング機能を用いる．これは，十字形のトラッキング・シンボルがライトペンを追随することにより画面上任意の位置を指定できる機能である．座標（整変数）は画面中心を原点とし，半径 2047 ラスタ・ユニットで与えられる．ブロックの表示・登録は，画面下部に表示されている 30 種類のブロック・メニュー端子をライトペンでピック・アップすることによって行う．そのときトラッキング・シンボルの位置にピック・アップされた端子の先がくるようにピック・アップしたブロックと同じ種類のブロックが表示・登録され，順次ブロック番号が割り当てられる．ただし，同じ位置に2個以上のブロックが重ねて表示されることはない．ブロックは 80 個まで表示・登録できる．

3.2 ブロックの接続と線番号

線をひくにはトラッキング・シンボルを引っぱって止める（ライトペン・スイッチを切る）か，あるいは端子またはすでにひかれている線をピック・アップする．このときトラッキング・シンボルの初めの位置から止まった位置まで，あるいは端子または線の先端（ピック・アップしたところから近い方の先端）まで線がひかれ，順次線番号が割り当てられる．線は 160 本まで使用できる．なお線を他の線や端子に「接続する」には必ずその線または端子をピック・アップしなければならない．消去指定 (ERASE) により線・ブロ

Table 1 Block table for the system shown in Fig. 3

ブロック番号 (subscript)	種類 (integer)	出力端座標 (integer)		入力先接続表 (integer)			パラメータ (real)				出力値 (real)	補助用 (real)
		x	y	1	2	3	1	2	3	4		
1	9	0	0	0	1	2	0.0	−0.2	−1.0	*	0.0	*
2	9	250	−35	1	0	0	1.0	0.0	0.0	*	1.0	*
3	30	525	−10	2	*	0	*	*	*	*	*	*
4	0	*	*	*	*	*	*	*	*	*	*	*
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
80	0	*	*	*	*	*	*	*	*	*	*	*

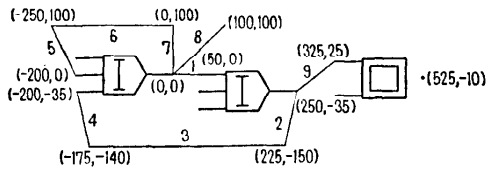

Fig. 3 Second order system block diagram along with its position cordinates and line numbers.

ックはピック・アップにより自動的に消去される.

3.3 パラメータ指定

各ブロックのパラメータを指定するには，まずそのブロック本体をピック・アップする．この操作により画面上部の計算条件とスケール指定の表示が消え，その位置に，そのブロックに必要なパラメータ名と初期値が表示される．ここで変更したパラメータをピック・アップし，キーボードから新しい値・変数を入力する．これらのダイアグラム情報は LIST をピック・アップすることによりリスト出力される.

4. ダイアグラムの内部表現

4.1 ブロック表と接続線表

各ブロックに関する情報はメモリ内部ではブロック番号を添字とする配列に記憶される．これをブロック表と呼び，ブロックの種類番号，出力端座標，入力先接続表，パラメータ，出力値および補助配列から構成される．たとえば **Fig. 3** の2次系に対するブロック表は **Table 1** で与えられる．ここで入力先接続表はダイアグラムの接続に関する情報であり，各入力端が何番のブロックの出力端に接続されているかを示し，その出力値を読み出すことにより入力値を求めることができる．この入力先接続表は各ブロックの出力端座標と接続線先端座標に基づいて以下のアルゴリズムにより自動的に作成される．なお使われていないブロック番号に対応するブロック種類番号は0である.

接続線情報は線番号を添字とする配列にその両端の座標 (x_1, y_1), (x_2, y_2) を記憶する．これを接続線表と呼び Fig. 3 の例では **Table 2** のようになる．なお使われていない線番号に対応する x_1 は 32,767 である.

Table 2 Line table for the system shown in Fig. 3

線番号 (subscript)	先端座標 (integer)			
	x_1	x_2	y_1	y_2
1	0	50	0	0
2	250	225	−35	−150
3	225	−175	−150	−140
4	−175	−200	−140	−35
5	−200	−250	0	100
6	−250	0	100	100
7	0	0	100	0
8	0	100	0	100
9	250	325	−35	25
10	32,767	*	*	*
⋮	⋮	⋮	⋮	⋮
160	32,767	*	*	*

4.2 接続表作成アルゴリズム

入力先接続表の作成アルゴリズムは，入力端から線をたどり，前段ブロックの出力端を見出すという基本的な方法を用いた．いま着目している入力端に直接または線を介して接続されているすべての線の集合をその入力端を根元とする木とみなす．木には高々1個の出力端が接続されているものとする.

いま $p(0 \leq p \leq 160)$ をポインタ，S_p を線番号スタックとすればアルゴリズムは，つぎのように与えられる.

Step 1　$p \leftarrow 0$, $(x, y) \leftarrow$ 入力端座標

Step 2　(x, y) に一致する出力端座標をもつブロックはあるか．あればそのブロック番号を入力先と決定し完了.

Step 3　(x, y) に一致する座標をもつ線はあるか．あれば Step 4 へ，なければ Step 5 へ.

Step 4　その線番号を N, $p \leftarrow p+1$ とし，一致した座標が (x_1, y_1) ならば $S_p \leftarrow N$ かつ $(x, y) \leftarrow (x_2, y_2)$; (x_2, y_2) ならば $S_p \leftarrow -N$ かつ $(x, y) \leftarrow (x_1, y_1)$ として Step 2 へ.

Step 5　$p=0$ ならば入力先なしと決定し完了.

Step 6　$p \leftarrow p-1$ かつその線の反対側の先端座標を (x, y) としまだたどっていない線で (x, y) に一致する座標をさがす．もしあれば Step 4 へ.

Step 7　$p=0$ ならば入力先なしと決定し完了.

Step 8　$N \leftarrow |S_p|$ とし線番号 N について

$S_p<0$ ならば $(x, y) \leftarrow (x_1, y_1)$

$S_p>0$ ならば $(x, y) \leftarrow (x_2, y_2)$

として Step 6 へ.

すなわちまず入力端座標 (x, y) を出力端座標から計算する*. つぎに出力端座標で (x, y) に一致するものを使われているすべてのブロック番号について調べる. もしなければ線の先端座標についても調べる. 線の座標で一致するものがあれば, その反対側の先端座標を改めて (x, y) とする. これで木の第1分枝が終了する. このとき根元からの段数を整変数として記憶し, その整変数を添字とする整配列にたどった線の番号を記憶する(線番号スタック). さらに線をたどるたびに線番号を添字とする論理配列に論理値 TRUE を記憶する. こうしてたどった順にその線の番号および方向が記憶されつつ進み, 最先端に何もつながっていなければもどってくることになる. さらに別の枝分れがあればその方向へ探索を進める.

このアルゴリズムによればループ状の接続があってもそれにはまりこむことはない.

接続表作成に要する時間はブロックの入力端子数や接続の仕方などにより異なる. ここでは座標はラスタ・ユニットそのものを整変数として扱っており, IF 文実行時間の短縮を計っている. 使用可能な最大ブロック数 80 個 (入力端子数 240), 最大接続線数 160 本を用いた試験的ダイアグラムで約6秒, 実用的にはさほど複雑でなければ1秒以下で作成が終了する. なおパラメータ修正などを行ってもダイアグラム構造に変更のない限り接続表は改めて作成されない.

本アルゴリズムの問題点は出力端どうしの接続があっても検出できないことにある. しかし全体のブロック・ダイアグラムが画面上に表示されており, 視覚的チェックが容易であるため使用上ほとんど問題はない.

5. 計算順序付けと初期設定

数値積分には固定刻み幅の4次 Runge-Kutta-Gill 法を用いた. この計算に際し, 以下のような自動順序付けを行う. 説明の単純化のため, ここではオイラー法による数値積分を考えるが, 何ら一般性は失われない.

さて時刻 $t_k = k\Delta t\ (k = 0, 1, 2, \cdots)$ における状態 (各ブロックの出力値) を計算するとき, 積分器の入力値として読み出される値は, t_{k-1} における状態でなければならない. 従って, 計算の順序はまず積分器, つぎにそれ以外のブロックという順になる. ただし積分器がカスケードに接続されている場合, はじめに前段について t_k における出力を計算してしまうと後段の入力としては, t_k における状態が読み出されることになる. そこで, 積分器については, 計算がすべて終了した時点で出力配列に移すという, 一括積分操作を行う.

ブロック・ダイアグラム全体の順序付けは, まず, 実行開始指令により, すべてのブロックについて初期設定を行う. そのとき積分器および入力端のないブロックはただちに初期出力が決定される. 入力端子をもつ積分器以外のブロックについては, そのすべての入力端の入力先の初期値が決定されていることを確認してから初期出力を決定する. 入力先の初期値が未決定ならばつぎのブロックへ進む. こうしてすべてのブロックについて初期値が決定されるまでこの手順をくり返す. なお初期設定の終ったブロックは, そのブロック番号を添字とする論理配列に論理値 TRUE を記憶する. さらに計算順序を添字とする整配列に初期設定の完了した順にそのブロック番号を記憶し, 以後, その順序でブロックの出力を計算する. なお演算はすべて単精度である.

6. エラー・チェック機能

本シミュレータの設計・開発に際しては, まえがきに述べたように極力, 実用性と使い易さを追求した. とくに, プログラミングに関する知識を全く要求しないことは, 対象とするユーザを著しく広範なものにしている. さらに実用に際し起りうる諸問題のほとんどすべてに対しエラー・チェック機能をもち, そのメッセージ表示および図形点滅などの確認機構によりすべて会話的に対応処置できるよう配慮されている. すなわち, 本システムには 60 近くのメッセージが用意されており各種の指令が正常に行われたことを示すもの, 各ステップでエラー・チェックにかかったことを示すエラー・メッセージなど, 適宜表示される. これらは操作の確認, パラメータ・条件の不備, 演算エラーの発生などを知らせるとともに必要な対処法を提供し, その処理を即時に行えるようになっている. とくに初期設定, 実行途中に発生したエラーに対しては, ダイアグラム・フェーズへもどしたときエラー発生ブロックが点滅し, その位置を明示するとともにパラメータが表示される. これらの機能は, 本シミュレータ稼動中に生じうると考えられる OS エラー, Fortran

* たとえば出力端座標を (x, y) とすれば3つの入力端座標はそれぞれ $(x-200, y+35)$, $(x-200, y)$, $(x-200, y-35)$ で与えられる.

エラーなどをすべて先回りチェックすることにより実現されており，過去1年間にわたる実用テスト期間中，異常動作は発生していない．

7. シミュレーション例

本章では，簡単なモデルを例にとり，本シミュレータの機能を概観する．とりあげる例は米国ミネソタ州の CEDER BOG LAKE ECOSYSTEM MODEL[8] である．ここで1年を 2π とし，2年間にわたるシミュレーションを行う．状態変数として太陽から供給されるエネルギーを x_s，植物の量を x_p，草食動物の数を x_h，肉食動物の数を x_c，死んだ動植物が腐触・沈澱して蓄積される有機物質の量を x_o，動植物によるエネルギー消費総量を x_e とする．これら変数を関連づける微分方程式はつぎのように記述される．

$$x_s=95.5(1+0.635\sin t)$$
$$\dot{x}_p=x_s-4.03x_p$$
$$\dot{x}_h=0.48x_p-17.87x_h$$
$$\dot{x}_c=4.85x_h-4.65x_c$$
$$\dot{x}_o=2.55x_p+6.12x_h+1.95x_c$$
$$\dot{x}_e=1.00x_p+6.90x_h+2.70x_c$$

ここで各変数の初期値は $x_p(0)=0.83$, $x_h(0)=0.003$, $x_c(0)=0.0001$, $x_o(0)=x_e(0)=0$ である．

Fig. 1 は，このモデルを記述したダイアグラム・フェーズを示したものであり，モデルの構造が一見して理解できる．**Fig. 4** は有機物質の蓄積量 x_o のシミュレーション結果を示すアナリシス・フェーズで，ここではとくに，太陽エネルギー（x_s）が減少したときの影響を調べるためゲイン・ブロックをパラメータ化し（Fig. 1 のブロックGの上に＊で示されている），その値を 95.5 から −10 ステップで変化させたときのシミュレーション結果をプロット出力したものである．

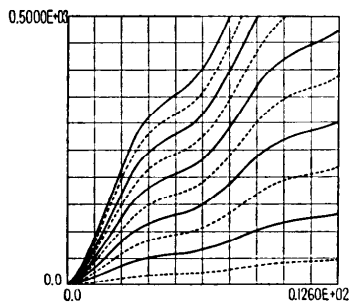

Fig. 4 Simulation results for CEDER BOG LAKE ECOSYSTEM MODEL: parametric changes for x_o with decreasing solar energy.

（パラメータ変更ごとに実線／点線を切換えた）．他の変数もスコープをつなぎかえるだけで同様に出力でき，またそれらを重ね書きプロットすることも可能である．

8. むすび

従来開発されてきた数多くの連続系シミュレータを実用的観点から再考し，使い易さを主眼に入力言語としてブロック図形を用い，システム記述，実行，結果の表示・プロットを含め，すべての操作をグラフィックスとの会話形式で実行できるシステムを実現した．これにより，従来 Fortran などの言語的手続きにより解かれてきた多くの数値計算問題は，いわゆるプログラミングをしなくても解け，プロットされた解が得られる．本シミュレータは今後さらに改良されるべき点も多々あるが，数年前には夢に近かった電子計算機のかような使用法を，一つの実用的形態として実現したものである．今後こうした問題向き言語は各種端末のインテリジェント化と相まって，マルチプロセッサ・システム[9]により一層使い易い形態へと発展していくであろう．

終りに本システム開発にあたり名古屋大学工学部 福村晃夫教授，鳥脇純一郎助教授，同大型計算機センター吉田雄二助教授はじめ多くの方々のご指導，ご協力を賜った．ここに謝意を表す次第である．

なお本シミュレータは現在 NUSS II として名古屋大学大型計算機センターに実行形式で登録・公開されている[10]．

参考文献

1) 三巻達夫：ダイナミック・システムのディジタル・シミュレーション，計測と制御，Vol. 7, No. 4, pp. 34～44 (1968).
2) 寺尾 満：ダイナミカル・システムのシミュレーション〔II〕—ディジタルシミュレーション—，計測と制御，Vol. 12, No. 4, pp. 34～45 (1973).
3) P. P. J. van den Bosch & H. P. R. S. Chouten: SIM—An Interactive Simulation Program for Both Continuous and Discrete Systems, In L. Dekker, ed.: Simulation of Systems, North-Holland, Amsterdam, pp. 1067～1070 (1976).
4) "LOOK, PROFESSOR, NO PROGRAMMING," FOREFRONT 1968/1970, Research in the College of Engineering, Univ. of California, Berkeley, pp. 35～38 (1970).
5) 臼井支朗，浅見秀雄，池谷和夫：会話型連続システムシミュレータ，情報処理学会マン・マシン

システム研究会資料, Vol. 26, No. 1, pp. 1～9 (1976).
6) 小倉俊一, 天水 昇, 佐藤清実, 斉藤宗昭: ブロック線図の図形入力による連続系シミュレータの試作, 昭和50年度電子通信学会全国大会, 1317, p. 1337 (1975).
7) 大須賀節雄, 山内平行: OLBA—実験的ブロック解析システム, 情報処理, Vol. 12, No. 2, pp. 103～113 (1971).
8) A. A. B. Pritsker: The GASP IV Simulation Language, Wiley-Interscience, New York, p. 324 (1974).
9) E. Yura, R. Yoshikawa, Y. Nara, T. Kimura, & H. Aiso: An Approach to Parallel Processing for Continuous Dynamic System Simulator with Micro Processors. Proc. of 2nd USA-JAPAN Computer Conference, pp. 172～177 (1975).
10) 臼井支朗, 浅見秀雄, 中野 洋, 熊谷 毅, 池谷和夫: 会話型グラフィック連続系シミュレータ NUSS II, 名古屋大学大型計算機センターニュース, Vol. 9, No. 3, pp. 197～212 (1978).

(昭和53年2月3日受付)
(昭和53年3月14日再受付)