空間的なアドレス表現を用いたアルゴリズムアニメーションによるプログラム理解支援ツール

小池 伸弥†　郷 健太郎‡
山梨大学工学部†　山梨大学大学院医学工学総合研究部‡

1. はじめに

情報系学科の多くでは，プログラミング教育においてC言語が主として使用されており（表1），C言語プログラミングは，情報系学科の学生にとって必要不可欠な能力と言ってもよい．しかし，初学者がC言語プログラミングを理解するのには困難を要するのが実情である．

初学者がC言語を学習する時に特に難しいと感じるのはアドレスとポインタの概念である．アドレスとポインタは「名高い難所」[2]と呼ばれ，C言語を習う際の難関と言われている．このアドレスとポインタを初学者がうまくイメージできないために，ポインタを利用したプログラムが理解できていないと考えられる．

そこで本研究では，プログラムの実行過程に応じてデータを平面空間上に可視化することによって初学者のプログラミング理解を支援するC言語デバッガツールを開発する．本手法では目に見えにくいプログラムのデータ構造をグラフィカルな平面空間に表現することで，パンニングとズームを使用した自由なナビゲーションを可能とし，ポインタ等のアドレス表現を直感的に理解することができる．

表1．情報系学科生の学習言語の違い（[1]に基づく）

	授業	授業＋自学	勉強せず
インタプリタ言語	79	19	151
C言語	112	94	54
オブジェクト指向言語	59	41	136
スクリプト系言語	20	5	222
Web系言語	17	9	209

2. 関連研究

プログラム開発を支援するために，プログラムに関する情報を視覚的に表現することをプログラム可視化という[3]．Myersはプログラム可視化を静的コード可視化システム，静的データ可視化システム，動的コード可視化システム，動的データ可視化システムの4つに分類した．これらは対象の可視化表現が静的であるか動的であるか，また，可視化対象がプログラム中のコードであるかデータであるかによって分けられている．また，アニメーションを利用して、動作過程やデータの変化を動的に可視化することをアルゴリズムアニメーションという．

A tool to help beginners learn program with algorithm animation using spatial address representation
†Nobuya Koike, ‡Kentaro Go
†Faculty of Engineering, University of Yamanashi.
‡Interdisciplinary Graduate School of Medicine and Engineering, University of Yamanashi.

3. 提案手法

3.1. 従来手法の問題点

現在，数多くのプログラム可視化ツールが存在するが，プログラミング初学者を対象としているツールは少ない．Code Canvas[4]は可視化空間を利用して膨大なコードファイルや関数の中から目的物を探しやすくしているが，初学者が作るプログラムの規模はそれほど多くないのでプログラミング教育には向いていない．Microsoft Visual C++付属デバッガはポインタを含めた多くの変数を可視化するが，キャラクターベースで可視化され初学者にわかりやすいとはいえない．Jeliot 3[5]のようにグラフィカルベースで変数を表示し，初学者にもわかりやすいようになっている手法もあるが，Jeliot 3はアドレス概念が存在しないJAVAプログラムのデバッガであるため，アドレスの可視化機能はない．

そこで本稿では，アドレスを平面空間上に対応付けたグラフィカルなデバッガを提案する．

表2．各手法の比較

	コード可視化	データ可視化	アドレス可視化
Code Canvas	○	×	×
Jeliot 3	△	○	×
VC++付属デバッガ	△	△	△
提案手法	△	○	○

3.2. 本提案手法の特徴

本手法は，コードに応じて変化するデータを可視化するデバッガの一種であり，動的コード，データ可視化システムに分類される．従来手法との比較を表2に示す．表示領域は大きく分けてコード表示部分とデータ可視化部分に分かれる（図1）．

コード表示部分では，現在実行されているコードがハイライトされ，ステップ実行ボタンを押すことにより次の実行ステップに移行する．その実行でデータの中身に変更があった場合，データ表示部分の平面空間と一覧表示の該当するデータの枠と中身が赤色で強調される．

データ表示部分は変数空間と一覧表示に分かれる．変数空間には現在のプログラム内のデータの状態が平面空間上に表示されている．アドレス空間を平面空間にマッピングして可視化することにより，数字の羅列でしかなかったアドレスを空間上のグラフィカルな任意の点として表すことができる．この変数オブジェクトの色は型によって決定される．一覧表示には現在のデータの一覧が表示されている．この部分にはデータの型，名前，そして内容が表示される．変数空間か一覧表示に表されているデータをクリックすると，そのデータに注目する．注目されたデータは一覧表示ではハイライトされ，変数空

間ではそのデータの内容がポップアップされる．注目したデータがポインタの場合はそのアドレスが指し示すアドレスを破線円で表現する．

変数空間では移動や拡大縮小といったインタラクティブな操作が可能であり，拡大の程度によって表示される内容が異なる．拡大率が小さい場合はデータの内容は表示されない（図 2(a)）が，拡大率を大きくするとデータの内容が常にデータ領域内に表示される（図 2(b)）．そして，さらに拡大率を大きくするとデータがバイト列単位で表示される（図 2(c)）．このように，変数空間にインタラクティブ性を持たせることにより，ユーザに合わせた自由度の高い使用が可能になる．

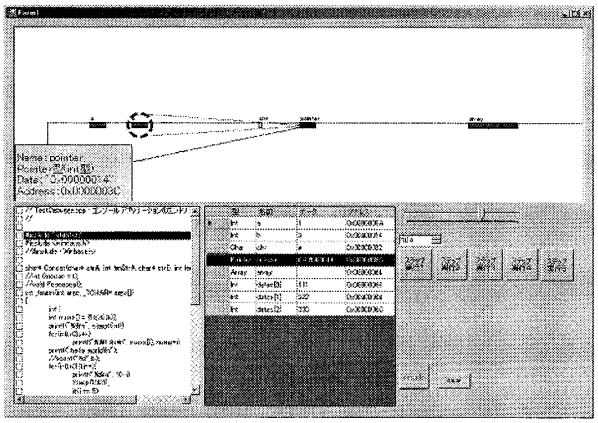
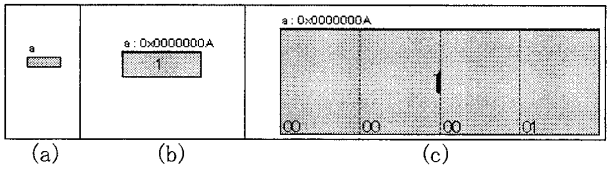


図 1. 提案ツール図



(a) (b) (c)

図 2. 変数空間のズーミング

4. 実験計画

提案手法の有用性を検証するために実験を行った．

4.1. 実験概要

本実験では，変数を空間的なアドレス表現を用いて可視化することが学習者の理解を深めるのかを評価するために，本ツールを用いて被験者のプログラムに対する理解度が深まったかどうかを測定した．プログラミング歴がそれぞれ半年間，2 年間，3 年間の情報系学科 1〜2 年生 3 人を被験者として実施した．

4.2. 実験手順

まず，ツールの使用方法を説明したあとで，指定した 5 つのコードファイルについて以下の手順で行った．

(ア) コードを印刷した紙を見せてどのような動作をしているかを書いてもらう

(イ) 書き終わったら本ツールを用いてそのコードファイルのプログラムを実行してもらい，新しく気付いた点や間違っていた点を書いてもらう

以上の作業が終了した後に理解度とツールの有用性について質問紙調査を行った．なお，コードファイルにはアドレスとポインタ，配列とソート，文字列と領域確保を中心にプログラムされており，質問紙調査では本ツールを用いてプログラムとアドレスの理解が深まったかどうかについて 7 段階のリッカート尺度（1:まったくそう思わない，2:ほとんどそう思わない，3:あまりそう思わない，4:どちらとも言えない，5:ややそう思う，6:かなりそう思う，7:非常にそう思う）を用いた主観評価と本ツールの使用感について記述してもらった．

5. 実験結果と考察

被験者の理解度についての回答結果を表 3に示す．

全体的に「頭の中で実行状態を想定するよりもスムーズに理解できた」という意見が得られた．特に，プログラミング歴が半年の被験者からは「値の入れ替わりが目に見えて脳内で考えるよりも遙かに理解しやすかった」という回答が得られた．プログラミング歴 3 年の被験者の回答では，文字列定数を保存しているアドレスと malloc()などの領域確保で使用されるアドレスが離れた位置にあることに気づくなど，比較的プログラミング歴が長い被験者からも新しい発見があったことがわかった．

被験者の意見からアドレスを平面空間上に対応付けた可視化が有効であることがわかった．さらに，空間に保存されている場所の違いに気づくなど，アーキテクチャの理解にも役立つ可能性が示された．一方で，「アドレス空間は非常に広大なので離れた位置にある変数を見比べる時に拡大縮小をしても移動が大変だ」という意見が挙げられたが，この点については自動で移動するなどの機能を付けることで改善されると考えられる．

表 3. 実験結果

7 段階評価(1:全くそう思わない, ..., 7:非常にそう思う)

プログラミング歴	半年	2 年	3 年
プログラム内容の理解が深まった	6	5	5
アドレスの理解が深まった	5	5	5
教科書主体の学習より理解できた	7	5	6
プログラミング時に役立つと思う	5	5	7

6. おわりに

本研究では，プログラミング理解度向上を目指し，データを平面空間上に可視化するツールを提案した．その結果，初学者にとってプログラムの理解がスムーズに行えたことがわかった．今後は初学者が作ったプログラムを実行できるようにして，Visual C++付属のデバッガと本提案手法の比較実験を行う．

参考文献

[1] 経済産業省委託 IT 教育実態調査プロジェクト，大学における I T教育実態調査報告書－情報系学科卒業生の視点，株式会社テクノフェイス，2004.

[2] 小林健一郎，カーニハン&リッチー「プログラミング言語 C」を読む，講談社，2006.

[3] B. A. Myers. Visual programming, programming by example and program visualization: A taxonomy., Proceedings of the ACM Conference on Human Factors in Computing Systems, pp.59-66, 1986.

[4] Kael Rowan, Code Canvas, http://research.microsoft.com/apps/pubs/default.aspx?id=80076 , 2009.

[5] A. Moreno, N. Myller, E. Sutinen, and M. Ben-Ari. Visualizing programs with Jeliot 3., Proceedings of the International Working Conference on Advanced Visual Interfaces, pp.373-376, 2004.