N-018

プログラミングにおけるポインタ操作学習のための支援システム
A Supporting System for Learning Pointer Operations in Programming

三浦 義之† Yoshiyuki Miura
鈴木 岳大† Takehiro Suzuki
金子 敬一† Keiichi Kaneko

1. はじめに

言語 C におけるポインタは，多くの初学者にとって理解が困難な概念である．ポインタを学び始めの頃には，抽象的にこういう処理をしたいと完全に認識していても，具体的に実装することができない学習者が，しばしば見られる．本研究では，そのような学習者を支援するために開発した VIE（Visual Interactive Educational）システムについて述べる．

VIE システムは，プログラム中のデータを視覚化する．学習者は，ステップ実行によって，視覚化されたデータが変化する様子を観察することができる．さらに，学習者は，データに対して，その値を変更するような操作を加えることができる．その際，VIE システムは，その操作に対応するコードを生成して学習者に提示する．もちろん，学習者の意図を完全に把握しなければ，正しいコードを生成することはできないし，学習者の意図を完全に把握することは，不可能である．そこで VIE システムでは，考えられる候補をいくつか提示する，という手法を採った．

2. 先行研究

使用者によって視覚化領域のデータに加えられた操作をコードに反映するシステムは，既にいくつか提案されている[1, 2, 3]．しかしながら，これらのシステムは，開発者向けに作られていたり，操作に対してコードが一意に決まるような単純な場合だけを扱っていたりする．そのため，ポインタを学び始めた学習者の支援という，本研究の目的には合わない．

このような背景から，学習者が視覚化領域に操作を加えることができ，その操作に相当するコードの候補を複数提示して学習者のポインタ操作の理解を支援することができる VIE システムを開発した．以下本稿では，VIE システムの概要とその評価について述べる．

3. VIE システムの概要

VIE システムでは，プログラムの編集，コンパイル，ステップ実行，実行の取消しといった従来のプログラミング学習環境で一般的なことに加えて，ス

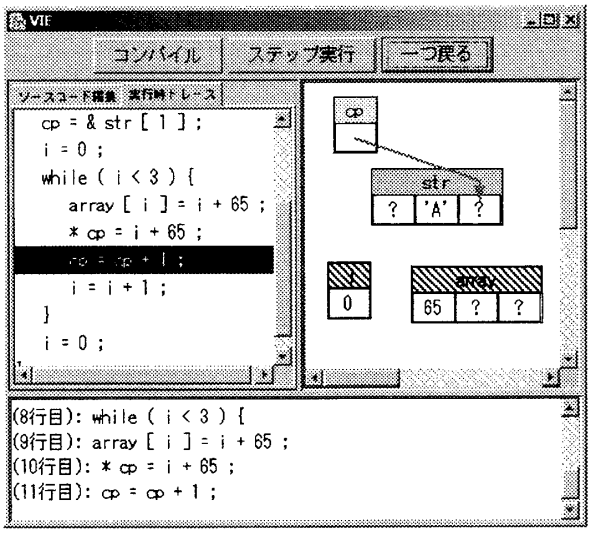

図 1　VIE の実行画面

ップ実行中のデータの視覚化および視覚化領域でのデータの変更を行うことができる．さらに，VIE システムは，学習者が行った操作に対応する複数のコード候補を生成し，提示する．図 1 に VIE システムの実行画面を示す．

図 1 は，プログラムの編集，コンパイルが終わり，ステップ実行を行っている状態を示している．ステップ実行で，プログラムを 1 文ずつ実行していく度に，図 1 右側の視覚化領域で表されているデータにも変化が反映される．この視覚化領域のデータには，マウス操作とキーボード入力により，数値を代入したり，ポインタを表す矢印を付け替えたりといった変更を加えることができる．そして，その変更操作に対応するコードを，図 1 下部のメッセージエリアに表示する．

しかしながら，学習者の操作に相当するコードは，一意に定まらない．例えば，配列 ar の 0 番目の要素を指しているポインタ p を，1 番目の要素を指すように変更するだけでも，p = p + 1;，p = &ar[1]; などと複数の可能性が考えられる．このような，複数の候補を生成するため，生成規則をいくつか用意した．

一方で，複雑すぎる候補が生成された場合，そのような候補は学習者に提示しないことが望ましい場合もある．そのため，評価関数というものを VIE システムに実装した．評価関数は，生成された候補の要素を字句に分解し，字句の種類ごとに設定した重

† 東京農工大学工学部

みを足し合わせることによって，候補の複雑度を測る．そして，最終的には，設定された閾値を越える複雑度を持つ候補は，学習者に提示されるリストから除外される．

言語 C はポインタ操作を含む一般的な言語であるため，VIE システムでは言語 C を対象としている．しかしながら，ポインタ操作を学ぶのに言語 C の全てを理解する必要はない．そのため，我々は言語 C のサブセットである SmartC という言語を定義した．VIE システムでは，この SmartC を使ってプログラミングを行う．図 2 に，SmartC の言語構成を示す．

int，char 型変数
int，char 型ポインタ変数
int，char 型一次元配列
代入文
if 文
while 文
四則演算，大小比較などの演算子

図 2　SmartC の言語構成

4. 評価

VIE システムの有効性を検証するため，簡単な実験を行った．ここでは，VIE システムが適切なコード候補を生成できるか否かを評価基準とした．具体的には，いくつかのプログラムを用意し，ポインタ変数への代入文の直前まで実行し，次に実行される代入文に相当する操作を視覚化領域で行う．そして，プログラムに書いてある代入文が，VIE システムにより生成されるかどうかを調べた．図 3 に，今回の実験で使った SmartC のプログラムのうち，特徴的な結果を示したプログラムを 1 つ示す．

```
int main(void){
  char str[6]; char *p; int c;
  //文字列"Hello"を配列 str に格納
  str[0] = 72; str[1] = 101;
  str[2] = 108; str[3] = 108;
  str[4] = 111; str[5] = 0;
  c = 0;  p = &str[0];
  while (*p != 0) {
    c = c + 1;
    p = p + 1; //(*)
  }
}
```

図 3　実験で使ったプログラム

図 3 のプログラムを使った実験では，(*)行の直前までステップ実行してから，視覚化領域でポインタを 1 つ右にずらす操作を行う．そして，その際に実際にプログラムで書かれている p = p + 1; が候補に含まれているかどうかを調べた．前述の操作をした際に得られた候補のリストを図 4 に示す．

```
p = &str[1];
p = p+1;
p = &str[str[5]+1];
p = &str[c];
```

図 4　提示された候補のリスト

図 4 より，図 3 に書かれている p = p + 1; が候補として生成されていることが分かる．さらに，この実験結果は，p = p + 1; だけではなく，p = &str[c]; と書くこともできること，つまりプログラミングの多様性を示すこともできるという，興味深いものになった．

また，今回の実験では，図 3 のプログラムを含めて全部で 5 つのプログラムを使った．そのうち，図 3 のプログラムを含めて 3 つのプログラムにおいて，実際にプログラムに書かれているのと同じポインタ変数への代入文が，候補として生成されていることが確認できた．一方，2 つのプログラムについては，プログラムに書かれている代入文が生成できなかった．いずれも別のポインタ変数からアドレス値を計算するという形（つまり p2 = p1 + 1; の形）であり，これは生成規則が不足していることが原因である．

5. まとめ

本稿では，言語 C におけるポインタ学習を支援するために開発した VIE システムについて述べた．このシステムでは，データを視覚化し，視覚化されたデータに対し行われた変更操作に対応するコードの候補を生成する，という手法を採った．実験を行った結果，簡単なプログラムについては，十分な候補が生成されることが確認できた．

今後の課題として，実験で明らかになった生成規則の不足を補うこと，今の枠組みを構造体でも使えるように拡張すること，実際に学習者に使ってもらい，システムの有効性を検証することが挙げられる．

参考文献

[1]Borland Software Corporation, "JBuilder", http://www.borland.com/jbuilder/.
[2]Michael Kolling, Bruce Quig, Andrew Patterson and John Rosenberg, "The BlueJ system and its pedagogy", Journal of Computer Science Education, Special Issue on Learning and Teaching Object Technology, Vol. 13, No. 4, pp. 249 - 268. Dec. 2003.
[3]株式会社　永和システムマネジメント, "JUDE", http://jude.esm.jp/.