WebGL を用いたプログラムの動的振舞いの可視化システムの構築

Design of a visualization system for dynamic behavior of programs using WebGL

末友 貴大[*1]，香川 考司[*2]
Takahiro Suetomo[*1], Koji Kagawa[*2]
[*1] 香川大学大学院工学研究科
[*1]Graduate School of Engineering, Kagawa University
[*2] 香川大学工学部
[*2]Faculty of Engineering, Kagawa University
Email: s13g466@stmail.eng.kagawa-u.ac.jp

あらまし：プログラミング学習において，学習者がプログラムの動的振舞いを理解することは重要であるが，初心学習者にとって，プログラムの動的振舞いの完全な理解は困難である．本研究では，プログラムの動的振舞いを静止 3D グラフィックスとして可視化するシステムを構築する．WebGL を用いることで，Web ベースシステムとしての実装が可能になり，環境に依存しない環境の提供が可能となる．
キーワード：プログラミング学習， 動的振舞い可視化， Web ベース， WebGL

1. はじめに

プログラムの動的振舞いを理解することで，学習者は自分のプログラムが正しく，効率的に動作しているかどうか知ることができる．しかし，初心学習者が，プログラムのソースコードから，その動的振舞いを完全に理解することは困難である．外部のデバッガを利用することで，調べることは可能であるが，通常のプログラミングの学習に加え，デバッガのインストールおよび操作方法を習得することは学習者にとって大きな負担となる．この問題解決のため，プログラムの動的振舞いを容易な手段で視覚化するシステムが必要とされていると考えられる．

プログラムの動的振舞いをアニメーションによって視覚化する既存のシステムとして Jeliot3 [(1)] が存在する．Jeliot3 は，Java プログラムに対応しており，if 文や再帰など，プログラミングの基本的な文法についての可視化を行い，学習の支援を行う．踊りによるプログラム処理の表現方法 [(2)] では，プログラムの動作を踊りによるアニメーションとして表現している．プログラムの関数，変数をキャラクタとして表現し，キャラクタの動きによって可視化を行い，学習の支援をする．しかし，これらはアニメーションとしての可視化であるため，複数のプログラムの対比が難しいという問題点が考えられる．

そこで本研究では，プログラムの動的振舞いを静止 3D グラフィックスとして可視化を行う Web ベースシステムを提案する．学習者は，模範例と自分の作成したプログラムの比較を行うことで，それらの違いを理解することができる．Web ブラウザ上の動作が可能なため，学習者にソフトウェアのインストール等の負担を強いることがない．また，3D グラフィックスによる表現であるため，学習者に印象づけやすく，プログラミング学習について，より理解が深まることが期待できる．

2. WebGL の使用

本システムは，WebGL を利用して実装する．WebGL とは，Web ブラウザ上で 3D グラフィックスを表示するための JavaScript の API である．OpenGL2.0 をサポートするプラットフォームであれば，特別なプラグインなしで 3D グラフィックスを表示することが可能である．現時点で対応している Web ブラウザは, Mozilla Firefox, Google Chrome, Safari，Opera であり，幅広い環境での利用が可能である．利用者の環境に依存しにくい学習環境の提供が可能になる．

3. システム構成

本システムは, プログラムのソースコードを元に，その動的振舞いに対応した静止 3D グラフィックスを生成し，Web ブラウザ上に表示をすることで，学習者のプログラミング学習を支援するものである．システムは，図 1 に示す構成になっており，大きく分けて，入力されたソースコードの解析を行うバックエンド，ユーザインタフェースの提供を行うフロントエンドからなる．バックエンドはサーバ，フロントエンドはクライアントで動作する．

3.1 バックエンド

バックエンドでは，受け取ったソースコードのトレース，および WebGL のソースコードの生成を行う. Web サーバサイドプログラムとして実装し, Web ブラウザから送信されたソースコードを元にサーバ上のトレーサを実行する．トレーサは，プログラムが実行している行番号および，条件分岐処理などの特別な処理をしていないかどうかの情報を取り出す．そのトレース情報に対応した WebGL のソースコードを生成し，クライアントへのレスポンスとして，これを送信する．バックエンドのトレーサ部を変更することで，さまざまなプログラミング言語の可視

化に対応することが可能である．

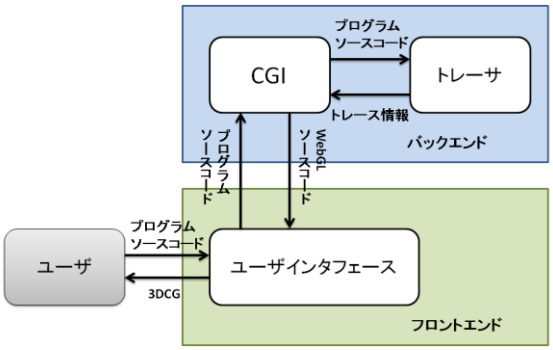

図 1 システム構成図

3.2 フロントエンド

フロントエンドでは，3D グラフィックスの表示，およびユーザインタフェースの提供を行う．ユーザによって入力されたプログラムのソースコード，および図形表示の設定情報をサーバに送信し，サーバから WebGL ソースコードを受け取り，3D グラフィックスとして表示する機能を持つ．表示された 3D グラフィックスは，マウス操作により回転，移動，および拡大縮小を行うことが可能である．学習者の環境により，複雑な図形の表示を行うと，マシンの動作が重くなる場合がある．そこで，本システムのユーザインタフェースでは，図形表示の簡略化を選択できる項目を作成した．

4. プログラムの動的振舞いの可視化

本システムでは，プログラムの動作を 1 本の線と見なした 3D グラフィックスの生成を行う．青色の球がプログラムのスタート，赤色の球がプログラムのゴールである．通常の処理を行っている場合は，画面の上から下へと実行している処理の大きさに対応した長さの線分が引かれる．for 文などによりループ処理を行っている場合は，そのループの大きさに応じて逆戻りをする．また，条件分岐処理を行っている場合は，線上に黄色の球が描かれる．これらの可視化図形によって，学習者は自分のプログラムが正しい動作をしているか，不必要な処理を行なっていないかを知ることができる．

4.1 可視化例

実際にプログラミング教育現場において発生したミスを例にプログラムの可視化を行う．問題は，「正五角形の全ての頂点を結ぶプログラムを作成せよ」である．正 n 角形のすべての頂点を結ぶ線分の数は，n(n-1)/2 本である．そのため，n(n-1)/2 回の線分を引く処理，この問題であれば 10 回の処理を行うことで問題の要件を満たすことができる．しかし，一度引いた線分を何度も引き直し，n^2 回，この場合 25 回の処理を行うプログラムを作成するという学生が数名存在した．この場合であっても，問題の要件は満たしているが，不必要な処理を多く行なっており、非常に非効率的である．これらについて視覚化したものが，図 2 および図 3 である．図 2 が効率的なプログラムの可視化図形，図 3 が非効率的なプログラムの可視化図形である．

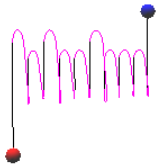

図 2 効率的なプログラム

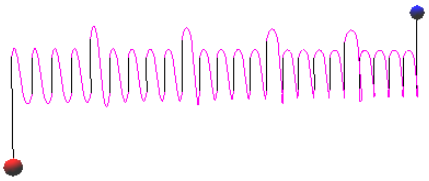

図 3 非効率的なプログラム

図 2 と比べると図 3 は，プログラムの動作を表した線が逆戻りをしている回数が非常に多くなっており，ループ処理を多く行なっていることが分かる．システムによって生成された 3D グラフィックスにより，学習者は，模範例と自分の書いたプログラムの違いを一目で理解することが可能になり，システムの利用によって効率的なプログラムを作成する支援ができると考えられる．

5. おわりに

プログラミング初学者のプログラミング学習支援を目的に，Web ベースの学習支援環境を開発した．ソフトウェアのインストール等，導入の作業を行う必要がなく，学習者の環境に依存しないため，学習支援環境を広く提供できる．また，難しい操作の必要がないため，学習者にかかる負担は小さいと考えられる．

本システムの目的は，学習者にプログラムの動的振舞いの理解の促進をさせることである．学習者は，システムによって表示された 3D グラフィックスにより，自分のプログラムの動作と模範例との違いを理解することができ，それによって，効率的なプログラム作成の支援ができると考えられる．

今後の課題として，初級プログラミング講義において試用し，システムの評価実験を行うことを考えている．

参考文献

(1) Moreno, A. 他，“Visualizing Programs with Jeliot3 ”, AVI 2004, pp.373-376, 2004.
(2) 山石忠弘 他，“ 踊りによるプログラム処理の表現方法”，情報学 CE 研報 2010-CE-107(12), 2010.