録画画像とコード片の対応関係可視化によるゲーム開発支援

深堀 孔明　　加藤 淳　　坂本大介　　五十嵐 健夫*

概要. 昨今ではすぐれたゲーム開発環境が多数世の中で公開されており，個人でのゲーム開発がより一般的なものになりつつある．ところが現在のコーディングエディタはソースコードをファイル単位で区切って表示しているために，離れた位置にある複数のコードを同時に編集する際，非常に煩わしい思いをしがちである．そこで本研究では，プログラムの実行画面の録画画像とプログラムのコード片を同一のビュー内に表示することにより，ユーザが大量のソースコードの中から目的のコードに容易に辿りつくことを可能にする開発環境を提案する．さらに本システムはゲーム開発用に最適化されており，プログラミングに詳しくないユーザでも容易にゲーム制作を行うことが可能になると期待される．

1 はじめに

本研究では，新しいゲーム制作用開発環境を提案する．一般にコーディングを行う際，複数の異なる位置にあるコードを同時に編集することが多い．従来のコードエディタではプログラムコードはひとつのビュー内にファイル単位で区切られたコードが表示される．したがって，目的のコード位置に移動するために，逐一対応するファイルを開き，画面をスクロールして目的のコード位置を探す必要がある．

そこで我々は，一般的なゲームの実装に用いられるアーキテクチャに最適化されたゲーム制作用開発環境を提案する．本環境では，ゲームプログラムの実行中は実行画面が常に録画される．同時に，画面内の各ピクセルがコード中のどのコード片によって描画されたのかという情報が記録される．ユーザインタフェースには，図 2 のように，複数の関数ごとに区切られたコード片，録画された実行画面および録画内容の時系列を表すタイムラインが表示される．これにより，過去の実行画面中の任意のピクセルを選択すると，そのピクセルを描画したクラスの各メンバ関数のコード片を表示することが可能となる．

以上により，ゲーム制作において大量のソースコードを扱う中で，目的のコード位置を直感的に探すことができるようになると期待される．

2 提案手法

我々の提案する開発環境では，一般的なゲームの実装に用いられるアーキテクチャを前提としている．すなわち，ゲームは複数のシーン（タイトルシーン，ステージ選択シーン等）からなり，その間を遷移しながらゲームが進行する．また各シーンは複数のオブジェクトからなっており，プレイヤーの入力に応じて毎フレーム状態が更新される．

本インタフェースはシーンの遷移を編集するビュー，およびシーン内の各オブジェクトを編集するビューの 2 種類のビューから構成される．また，本システムはオブジェクト指向言語の開発環境を想定しており，各シーン，オブジェクトはクラスのインスタンスとして実装される．以下にその詳細を示す．

2.1 シーン遷移ビュー

シーン遷移ビューは，ゲームの各シーンの録画画像およびタイムライン，シーン間の遷移を表す矢印，および遷移条件を表すコード片からなる（図 1）．各コード片は真偽値型を返す関数であり，遷移元のシーンに属すメンバ関数として扱われる．画面下部のタイムラインにはプログラム実行開始からの毎フレームのスクリーンショットが表示される．

ユーザはマウス操作でシーンおよびシーン間の遷移（矢印）を追加し，遷移条件を書くことでシーン遷移を編集する．これによりシーン毎の関係性を記述する．また，シーンにおいてはフレーム毎にシーン遷移の判定を行い，遷移の必要がある場合には，記述された関係性をもとに遷移を行う．

ユーザはインタフェース上で各シーンないしタイムラインの画像を選択することで，以下で示すシーン編集ビューに移動することができる．

2.2 シーン編集ビュー

シーン編集ビューは，シーンの録画画像，タイムライン，および各オブジェクトのコード片が表示されている（図 2）．

タイムラインを選択すると，録画画像として対応する時点のスナップショットが表示される．

シーンの録画画像上のあるピクセルを選択すると，そのピクセルを描画したオブジェクトの画像および，そのオブジェクトが属すクラスの各メンバ関数がコー

* Koumei Fukahori, 東京大学理学部情報科学科, Jun Kato, Daisuke Sakamoto and Takeo Igarashi, 東京大学大学院情報理工系研究科

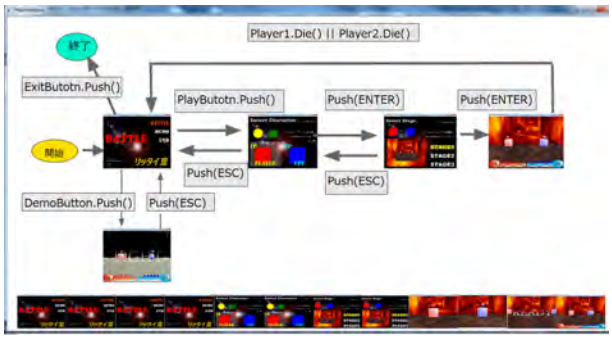

図 1. シーン遷移ビュー

図 2. シーン編集ビュー

ド片として表示される．オブジェクトをビュー上で追加すると，オブジェクトが描画する画像および，クラスのメンバ関数がコード片として表示される．

ユーザはマウス操作で型を指定したオブジェクトを追加できる．ビュー上で例えば右クリックを行うとオブジェクトを新規作成するためのフォームが出てきて，作りたいオブジェクトの型を既存の型から選択できる．また，新しいクラスのオブジェクトを作成することもでき，そのときは黒画面と空のメンバ関数（初期化，更新，描画関数）のコード片が追加される．ユーザがこれらの関数を実装しプログラムを実行すると，シーン画像とタイムラインが更新されるだけでなく，今実装した描画関数内で描画された映像が黒画面だった領域に表示される．

3 関連研究

ひとつのビューに複数のコード片を表示するシステムには”Code Bubbles”[1] がある．このシステムでは実行画面を表示することができず，描画された画像に対応するコード位置へ移動することが難しいという課題がある．また録画された実行画面から対応するプログラムコードに移動するシステムとして”Whyline”[2] がある．指定したピクセルが描画されたコード位置に移動することができるが汎用のプログラム用に作られているため，対応するクラスを特定することはできない．

本稿で提案するシステムはインタフェース上に描画された画像を用いて，対応するコード位置へ移動することができ，かつ対応するクラスに用意にアクセスことができるため，これらの研究とは異なる．また，このためこれまで以上にゲーム制作において効率的に作業が行うことができる開発環境であると考えている．

4 まとめ

本研究では，録画画面とコード片を同一のビュー内に表示するインタフェースおよびゲーム開発に特化した開発環境のデザインを提案した．このシステムを利用することになり，ユーザは大量のソースコードの中から目的のコードに容易にアクセスできるようになり，複数の異なる位置にあるコードの編集をストレスを感じることなく行うことができるようなると期待される．

参考文献

[1] Bragdon, A., Zeleznik, R.C., Reiss, S.P., Karumuri, S., Cheung, W., Kaplan, J., Coleman, C., Adeputra, F., and Jr., J.J.L. Code bubbles: a working set-based interface for code understanding and maintenance. In *Proceedings of the 28th International Conference on Human Factors in Computing Systems*, pp. 2503–2512, 2010.

[2] Ko, A.J. and Myers, B.A. Debugging Reinvented: Asking and Answering Why and Why Not Questions about Program Behavior. In *International Conference on Software Engineering*, pp. 301-310, 2008