メディアアートのための制約プログラミング環境

細部, 博史 / HOSOBE, Hiroshi

(雑誌名 / Journal or Publication Title)
科学研究費助成事業 研究成果報告書
(開始ページ / Start Page)
1
(終了ページ / End Page)
4
(発行年 / Year)
2017-06-09

様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９　（共通）

科学研究費助成事業　　研究成果報告書

平成　２９　年　　６　月　　９　日現在

機関番号：　３２６７５
研究種目：　挑戦的萌芽研究
研究期間：　2013～2016
課題番号：　２５５４００２９

研究課題名（和文）メディアアートのための制約プログラミング環境

研究課題名（英文）A Constraint Programming Environment for Media Arts

研究代表者
細部　博史（HOSOBE, Hiroshi）

法政大学・情報科学部・教授

研究者番号：６０３２１５７７

交付決定額（研究期間全体）：（直接経費）　　2,900,000円

研究成果の概要（和文）：コンピュータグラフィックスを用いた視覚的アート（メディアアート）の制作を容易にすることを目的として，プログラミング環境に関する研究を行った．そのためのアプローチとして，制約プログラミングを導入した．制約プログラミングは高水準な数式や論理式などを扱うことができ，従来の命令型プログラミングよりも直観的なプログラミングを可能にする．本研究ではメディアアート制作のための制約プログラミング言語を構築した．

研究成果の概要（英文）：To enable the easy creation of visual arts by using computer graphics (i.e., media arts), we conducted research on a programming environment. For this purpose, we adopted constraint programming. Constraint programming allows the treatment of high-level arithmetic equations and logical formulas, and therefore enables more intuitive programming than conventional imperative programming. We constructed a constraint programming language for the creation of media arts.

研究分野：　情報学

キーワード：　プログラミング環境　メディアアート　制約

１．研究開始当初の背景

コンピュータグラフィックスを用いた視覚的アート(以下，メディアアート）が身近になっている．特にウェブでメディアアート的表現が普及し，Flash を用いた美しくインタラクティブなコンテンツが多く見られるようになった．メディアアート制作者にとってプログラミングは重要なスキルとなりつつあり，Processing（引用文献①)のようにメディアアートを主な対象としたプログラミング環境も登場している．しかし，Flash や Processing はグラフィックスやインタラクションの扱いには優れているものの，命令型プログラミングに基づいており，図形要素の座標などの計算手順の記述を必要とするため，メディアアート制作者にとって敷居が高い．

視覚的アプリケーションを対象とした非プログラマあるいは初心者向けのプログラミング環境は多数開発されている．例えばタートルグラフィックスで有名な LOGO，グラフィカルユーザインタフェース(GUI)とハイパーテキストを対象とした HyperCard，3 次元グラフィックスを扱う Alice（引用文献②)，図形書換ルールによってアニメーションを記述する Viscuit（引用文献③)などがある．しかし，これらの多くは命令型プログラミングを基本としており，計算手順の記述を必要とする．例外的に Viscuit はルールベースのプログラミング方式と柔軟な実行機構を採用し，子供にも利用可能な環境となっているが，反面，正確な計算の記述には適していない．

このような命令型プログラミングの問題に対処するために，本研究代表者は制約プログラミングの研究を行ってきた(引用文献④⑤)．制約プログラミングは高水準な数式や論理式などを制約として扱うことができ，より直観的なプログラム作成を可能にし，特に視覚的アプリケーションの構築に適している(引用文献⑥)．

２．研究の目的

本研究はメディアアートコンテンツを容易に制作するためのプログラミング環境の構築を目的とする．そのためのアプローチとして，制約プログラミングを導入する．これによって，メディアアートにおける図形要素の座標などを求めるときに，計算手順を記述するのではなく，高水準な制約を記述することができるため，メディアアート制作者にとってより直観的にプログラムを作成できるようになる．

３．研究の方法

本研究では，メディアアート制作のための制約プログラミング技術を開発し，プログラミング環境として提供する．より具体的には，メディアアートに適したプログラム処理技術と，その要素技術としての制約処理技術を開発し，さらにこれらを実装したメディアアートプログラミング環境を構築する．

４．研究成果

(1)制約命令型プログラミング言語の提案

新しいプログラミング言語 P5CP を提案した．P5CP は，JavaScript 言語と p5.js ライブラリの組合せに対して，制約プログラミングの機能を加えたものである．JavaScript 自体は命令型プログラミング言語であるため，制約プログラミングの機能をさらに備えた本言語は，制約命令型プログラミング言語と見なすことができる．

本言語は，Kaleidoscope'90 などの制約命令型プログラミング言語とは異なり，制約プログラミングと命令型プログラミングの密接な統合を行わない．本言語では 1 つのプログラムの中に制約プログラム(constraint program; CP) と命令型プログラム(imperative program; IP)が混在するが，それぞれが明確に区別できる形で存在する．プログラムの実行も，CP を処理する CP フェーズと，IP を処理する IP フェーズが交互に動作することで進められる．フェーズ内で計算を進める基本機構が CP フェーズではガードであり，IP フェーズではイベントである．

プログラム実行中の CP，IP の状態はそれぞれ CP 変数，IP 変数によって保持される．CP，IP のいずれも CP 変数，IP 変数の両方の値を読み取ることができるが，基本的に CP フェーズ，IP フェーズの実行で変更されるのはそれぞれ CP 変数，IP 変数の値である．CP 変数，IP 変数の区別は変数名によって行われ，$x や，$foo のようにドル記号で始まるものを CP 変数とし，それ以外の変数を IP 変数とする．

JavaScript では本来，var 文で宣言されたもののみを変数と呼ぶべきであるが，本言語では JavaScript における変数，オブジェクトのプロパティ，関数の仮パラメータを区別せず，単に変数と呼ぶ．ただし，本言語で CP 変数にできるのは，JavaScript におけるオブジェクト(グローバルオブジェクトを含む)のプロパティに相当するもののみである．例えば単に$x のように書かれた場合，これはグローバルオブジェクトのプロパティを指し，bar.$foo のように書かれた場合，これはオブジェクト bar のプロパティ$foo を指す．

本言語のプログラムでは IP が主体であり，その中に CP の断片が埋め込まれる．IP フェーズの実行中に CP の断片が現れると，CP 変数の生成・初期化，または(ガード付き)制約の生成が行われる．

CP は CP 変数宣言文と always 文からなる．CP 変数宣言文は CP 変数を生成して初期化する．構文的には，変数名がドル記号で始まることを除き，通常の JavaScript における等号記号を用いたプロパティの宣言文と同様である．具体的には，グローバル環境で，新しい変数名$x に対して“$x = 1;”のように

記述するか，オブジェクトのコンストラクタで，新しい変数名$fooに対して “this.$foo=100;”のように記述することで，CP変数を宣言できる．

always文はCP変数の初期化後に成立すべき制約を宣言する．always文にはガード付きの制約を記述でき，この場合，ガードが成立するときにのみ制約が採用される．ガード付きの制約は，JavaScriptにおけるif文と同じ構文を用い，条件部にガードを，then節に制約(ガード付きでもよい)を記述する．さらにif文にelse節を付けることもできる．ガードとして記述できるのは，JavaScriptにおける条件式のうち，副作用のないものである．一方，制約として記述できるのは，単方向制約と常微分方程式制約である．

IPの中で，CP変数を記述することでその値を読み取ることができる．一方，CP変数に代入することはできない．他の部分でIPの記述はJavaScriptとほぼ同様である．

(2)計算モデルの提案

制約命令型プログラミング言語P5CPの計算モデルを提案した．P5CPでは，CPを処理するCPフェーズと，IPを処理するIPフェーズが交互に動作する．基本的にCPフェーズは従来のハイブリッド並行制約プログラミング(hybrid concurrent constraint programming; HCCP)に相当し，IPフェーズは従来のイベント駆動プログラミング(event-driven programming; EDP)に相当しており，P5CPの計算モデルはこれらの処理を分割した上で，交互に動作するようにしたものである．

通常のHCCPでは，初期化の後，時間が進行しない状態変化である離散変化(複数の離散変化が続けて生じる場合も含む)と，時間が進行する状態変化である連続変化が交互に繰り返し生じる．連続変化は特に常微分方程式を解く(例えばボールが落下する)ことで生じ，離散変化は連続変化の終了直後(例えばボールが床で弾む)に生じる．このように離散変化と連続変化の切り替えが生じるのは，状態の変化に応じてガードの真偽が変わり，そのために採用される制約も変わるためである．

p5.jsにおけるEDPでは，最初にsetup関数が呼び出され，その後，一定時間ごとにdraw関数が呼び出されることで処理が進行する．さらに，これらの関数呼び出しの合間にも，他のイベント(例えばキーの入力やマウスの移動)の発生に応じて，対応する関数が呼び出される．ここでsetup，drawの呼び出しも特殊なイベントによるものであると見なせば，全ての処理はイベントに応じた関数呼び出しによって行われると言える．

P5CPの計算モデルでは，EDPにおけるdrawの各呼び出しの直前に区切りを入れ，それぞれのまとまりをHCCPにおける離散変化の途中に差し込むことで，CPフェーズとIPフェーズが交互に動作する全体の処理を構成する．ここで，時間の進行は連続変化のみで生じるものとする．

本計算モデルでは，時間の扱いに関して以下の2つの近似を行っている．1つ目は，setup，draw以外のイベント処理が直前のsetup，drawと同じ時刻に行われるとしている点である．これらのイベント処理はキーの入力やマウスの移動に応じたものであり，本来はこれらのイベントの発生までに時間が進行しているはずであるが，本計算モデルでは時間の進行を考えない．2つ目は，連続変化で次のdrawの呼び出しの直前まで時間が進行するとしている点である．p5.jsと同様，drawの呼び出しは一定時間ごとに行われるため，連続変化ではその分だけ時間が進行する．通常のHCCPで，連続変化の終了は，採用されていた制約のガードが偽になることで生じるため，連続変化の長さが一定になるとは限らないが，本計算モデルでは連続変化も一定時間ごとに区切り，若干の時間のずれは許容する．

これらの時間の扱いに関する近似を行っていても，CPフェーズにおけるガードに応じた処理と，IPフェーズにおけるイベントに応じた処理が可能である．また，これらのフェーズは他方の変数の値を読み取ることで情報を交換できるため，協調動作することが可能である．

(3)処理系の実装

提案言語P5CPの処理系の実装方法を示した．本処理系はP5CPのプログラムからJavaScriptのプログラムへのトランスレータであり，このためにScala言語とパーサ生成系ANTLRを用いている．CPの実行で必要となる制約解消系は新たにJavaScriptで開発したものであり，常微分方程式の解消には改良Euler法を採用している．また，IPの実行のためにp5.jsを用いている．

＜引用文献＞

①C. Reas and B. Fry, Processing: Programming for the Media Arts, AI Soc., 20(4), 526-538, 2006.
②M. Conway, S. Audia, T. Burnette, D. Cosgrove, and K. Christiansen, Alice: Lessons Learned from Building a 3D System for Novices, Proc. ACM CHI, 486-493, 2000.
③原田康徳, 子供向けビジュアル言語Viscuitとそのインタフェース, 情報処理学会研究報告：ヒューマンインタフェース(HI), 116, 41-48, 2005.
④H. Hosobe, A Modular Geometric Constraint Solver for User Interface Applications, Proc. ACM UIST, 91-100, 2001.
⑤H. Hosobe, A Simplex-Based Scalable Linear Constraint Solver for User

Interface Applications, Proc. IEEE ICTAI, 793-798, 2011.
⑥M. Christie, H. Hosobe, and K. Marriott, Trends and Issues in Using Constraint Programming for Graphical Applications, Trends in Constraint Programming, ISTE, 371-381, 2007.

５．主な発表論文等
（研究代表者、研究分担者及び連携研究者には下線）

〔雑誌論文〕（計７件）
①Satoru Imura and Hiroshi Hosobe, Biometric Authentication Using the Motion of a Hand, Proceedings of the 4th ACM Symposium on Spatial User Interaction (SUI2016), 査読有, 221-221, 2016.
DOI:10.1145/2983310.2989210
②Hiroshi Hosobe, Toward a New Constraint Imperative Programming Language for Interactive Graphics, Companion Proceedings of the 15th International Conference on Modularity (Modularity2016), 査読有, 34-35, 2016.
DOI:10.1145/2892664.2892668
③Yusaku Yokouchi and Hiroshi Hosobe, A Mouse-Like Hands-Free Gesture Technique for Two-Dimensional Pointing, Communications in Computer and Information Science (HCI International 2015-Posters' Extended Abstracts, Part I), 査読有, 528, 558-563, 2015.
DOI:10.1007/978-3-319-21380-4_95
④Hiroshi Hosobe, A Hierarchical Method for Solving Soft Nonlinear Constraints, Procedia Computer Science (Proc. SCSE2015), 査読有, 62, 378-384, 2015.
DOI:10.1016/j.procs.2015.08.422
⑤Azusa Kurihara, Akira Sasaki, Ken Wakita, and Hiroshi Hosobe, A Programming Environment for Visual Block-Based Domain-Specific Languages, Procedia Computer Science (Proc. SCSE2015), 査読有, 62, 287-296, 2015.
DOI:10.1016/j.procs.2015.08.452

〔学会発表〕（計４件）
①細部博史, 対話型視覚的アプリケーションのための制約命令型プログラミング言語, 日本ソフトウェア科学会第32回大会, 2015.9.8-2015.9.11, 早稲田大学西早稲田キャンパス(東京都・新宿区).
②Hiroshi Hosobe, A Numerical Optimization-Based Method for Visualizing Graphs, Computer Visualization-Concepts and Challenges, 2014.3.10-2014.3.13, 湘南国際村センター(神奈川県・葉山町).

６．研究組織
(1)研究代表者
細部　博史（HOSOBE, Hiroshi）
法政大学・情報科学部・教授
研究者番号：60321577