制約に基づくアニメーション作成環境 Grifon

石井 大輔† 中村 好一† 大野 太郎‡ 若槻 聡一郎‡ 上田 和紀‡
†早稲田大学大学院 理工学研究科 ‡早稲田大学 理工学部

1 はじめに

本研究では、おもに論理的な概念や関係を表すようなアニメーションを柔軟に表現することを目指し、アニメーション作成システム Grifon の設計・実装を行った。

Grifon では、アニメーションに適した高級なデータ表現を実現するために、制約に基づきアニメーションを表現した。制約には、専門性の高い内容をシンプルかつ直感的に表現できる、モジュール化された表現が容易である、といった特徴がある。

Grifon ではアニメーション中の時間的変化を Hybrid 並行制約によって、図形オブジェクトの空間的な関係を幾何制約によって表現した。Hybrid 並行制約は時間的変化を表現するための枠組みで、とくに時間軸上の連続的・離散的な処理の表現に優れている。幾何制約は図形の座標やサイズなどの実数属性値を変数とし、変数間の幾何学的な関係を表現する。Grifon は Hybrid 並行制約と幾何制約を同時に扱い、直感的に制約を充たすための制約充足系を備えている。

Grifon はアニメーションライブラリを備え、階層的なアニメーション部品に基づいた作成作業を行うことができる。アニメーションは、図形部品を表すパラメタ化 SVG と制約定義の 2 種類の部品から構成される。

2 制約に基づいたアニメーション

2.1 Hybrid 並行制約プログラミング

Hybrid 並行制約プログラミング (Hybrid Concurrent Constraint Programming) [1] は、並行制約プログラミングを時間軸に沿った表現が可能なように拡張した枠組みである。表 1 のプログラム例は、ボールが床に跳ね返る動作を表したものである。ボールが床まで落下した時点での挙動を宣言的に記述しておくと、解釈時に挙動が起こる時点が計算される。

表 1: Hybrid 並行制約プログラムの例

```
y = 10, y' = 0,              // 初期状態
hence {
  cont(y),                   // 連続的な変数の指定
  if y > 0 then y'' = -10,   // 落下（加速度の設定）
  if y = 0 then              // 床に達した
  if (prev(y') > -0.000001) then always y' = 0
      else y' = -0.5 * prev(y')
                             // 跳ね返る
}
```

2.2 幾何制約

幾何制約は図形中の点や長さなどの属性値をドメインとした制約である。たとえば「ある点をある座標に固定する」や「2 つの点の x 座標を等しくする」といった制約である。

2.3 パラメタ化 SVG

パラメタ化 SVG (Parameterized Scalable Vector Graphics、以下「PSVG」とする) は一般的なベクター画像データに、位置座標やサイズ、色、生成個数などを操作するパラメタが付随したデータである。

2.4 制約システム

Grifon では Hyblid 並行制約と幾何制約を、PSVG のパラメタ値を制約のドメインとし、用いる。制約充足が行われる過程にアニメーションが実現される。

3 Grifon の実装

Grifon は Java (JDK 1.4.2) によって実装した。

Grifon はおもに、GUI、アニメーションライブラリ、制約充足系から構成される。

3.1 GUI

GUI はユーザが、アニメーション部品群の読み込みや PSVG パラメタと制約変数のマッチングなどの操作を行ったり、アニメーションの表示を行うための環境である。おもにライブラリビューとキャンバスから構成される (図 1)。

Design and Implementation of the Grifon Constraint-based Animation Authoring System

Daisuke ISHII†, Koichi NAKAMURA†, Taro OHNO‡, Soichiro WAKATSUKI‡, Kazunori UEDA‡
†Graduate School of Science and Engineering, Waseda University
‡School of Science and Engineering, Waseda University
{dai, koichi, ohno, wakatsuki, ueda}@ueda.info.waseda.ac.jp

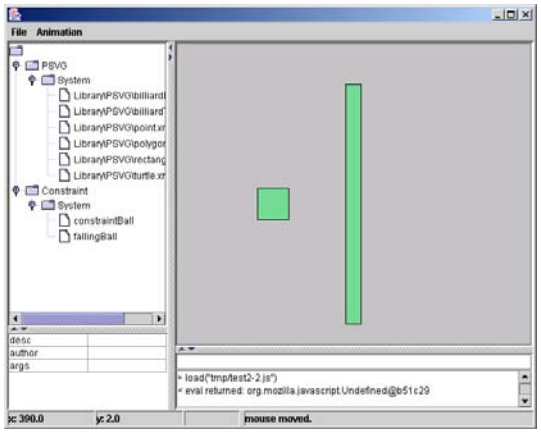

図 1: GUI のスクリーンショット

3.2 アニメーションライブラリ

アニメーションライブラリは PSVG と制約の 2 種類のデータからなる。データ形式は XML に基づく。

内部において PSVG のパラメタは、`GrifonTerm` インタフェースを各型ごとに実装したオブジェクトとして表される。Hybrid 並行制約と幾何制約は、`GrifonConstraint` インタフェースで抽象されたオブジェクトとして表される。

3.3 制約充足系

3.3.1 構成

Hybrid 並行制約の処理には Gupta によるインタプリタを、幾何制約の処理には Cassowary [2] を利用した。`GrifonSolver` クラスが一括して Hybrid 並行制約と幾何制約の充足処理を行う。

3.3.2 処理アルゴリズム

(1) `GrifonSolver` は Hybrid 並行制約を処理し、アニメーションの各フレームにおけるパラメタ値のサンプリングデータを生成する。まずキャンバスからの情報を元に Hybrid 並行制約のスクリプトを生成し、インタプリタへ渡す。インタプリタは数値計算により離散変化が起こる時点を求める。離散変化時点における処理と、次の時点までの時間の処理を交互に繰り返し、サンプリングデータを生成する。

(2) 次に `GrifonSolver` はサンプリングデータに基づき、アニメーションを試行する。

(2') 途中で幾何制約が充たされない状態が生じた場合、Hybrid 並行制約インタプリタへ知らせる。インタプリタは知らされた時点から処理をやり直す。

(3) アニメーションに必要な、幾何制約を充たしたサンプリングデータが全て生成されたならば終了する。

4 Grifon によるアニメーションの例

アニメーションの例として、Hybrid 並行制約を用いたビリヤード台の例 (図 2) や、Hybrid 並行制約と幾何制約を用いた矩形同士の衝突の例 (図 3) を作成した。後者の例では Hybrid 並行制約と幾何制約間の階層を指定することにより、動作を変えることができる。

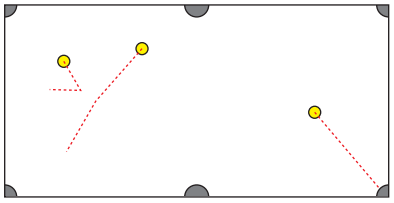

図 2: アニメーションの例 (1)

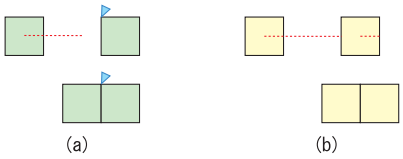

図 3: アニメーションの例 (2)

5 まとめと今後の課題

本論文では、制約によるアニメーションの表現手法と、それに基づくシステムについて述べた。Hybrid 並行制約により高度な動作を簡潔に表現でき、幾何制約と組み合わせることで、柔軟な作成作業が行える。

今後、ライブラリを充実させるとともに、階層的に部品を構成できるような仕組みを実装したい。また、たとえば motion signals 形式などの抽象的なデータを出力し、他のシステムと連携させ、より高度なアニメーションを実現することを考えている。

本研究の一部は、IPA 平成 14 年度未踏ソフトウェア創造事業「未踏ユース」の補助を得て行った。

参考文献

[1] Gupta, V., Jagadeesan, R., Saraswat, V. A., Bobrow, D. G.: Programming in Hybrid Constraint Languages. Hybrid Systems II, LNCS 999, Springer Verlag, 1995.

[2] Borning, A., Marriott, K., Stuckey, P. and Xiao, Y.: Solving Linear Arithmetic Constraints for User Interface Applications, In *Proc. of the 1997 ACM Symposium on UIST*, 1997, pp. 87–96.