制約に基づくアニメーション作成環境 Grifonにおけるデータ構造の階層化

中村 好一† 石井 大輔† 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

現在、アニメーション作成システムにはさまざまなものがある。たとえば、Flashなどの高度なアニメーションを作るための作成環境などにおいては、ユーザはそれを扱うためのテクニックを学んでいく必要がある。逆に、たとえば、PowerPointなどの簡単な操作でアニメーションが作成できる環境では、単純なアニメーションしか作成することができない。また煩雑な繰り返し作業などが必要になってくるというデメリットがある。

そこで本研究では、一般ユーザが簡単に、かつ直感的にいろいろな種類のアニメーションを作成できるようなアニメーション作成システムGrifonの開発を行なった。

1.1 Grifonの特徴

Grifonは、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ためのアニメーション作成環境である。

Grifonでは、制約に基づきアニメーションを直感的に表現する。特にHybrid並行制約に基づき離散的変化と連続変化からなるアニメーションを表現できる。また幾何制約により図形の幾何学的規則を表現できる。図1は、Grifonのスクリーンショットである。

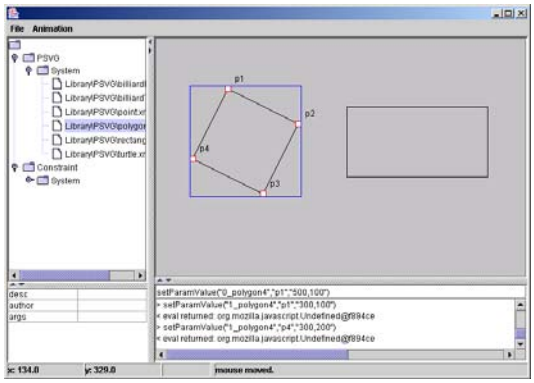

図 1: Grifonのスクリーンショット

2 制約の階層的・局所的な処理

制約に優先順位をつけて階層的に充足する手法が提案されている。また、図形全体の制約を一括して充足するのではなく、局所的に処理できると望ましい。

本研究では、図形 (PSVG) を階層化された構造とし、その中で階層的・局所的に制約充足を行う手法を提案した。

3 データ構造の階層化

3.1 データ構造

PSVG (Parametric Scalable Vector Graphics) ベクター画像情報にパラメータが付随したものである。パラメータのとりうる値を決めることによって画像属性の操作を行なうことができる。

幾何制約 図形の相互関係を崩さず、幾何学的性質を維持する。

Hybrid 並行制約 図形同士の時間的関係を維持する。

Hierarchical data structure in Constraint-Based Animation Tool Grifon

Koichi NAKAMURA†, Daisuke ISHII†, Taro OHNO‡, Soichiro WAKATSUKI‡, Kazunori UEDA‡

†Graduate School of Science and Engineering, Waseda University

‡School of Science and Engineering, Waseda University

{koichi,dai, ohno, wakatsuki, ueda}@ueda.info.waseda.ac.jp

データ構造の階層化について 2 つの平行な (平行という制約がかかった) 直線と正方形が描かれた例をもちいて説明する。

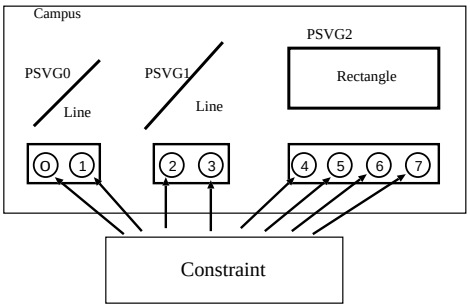

図 2: データ構造の階層化前

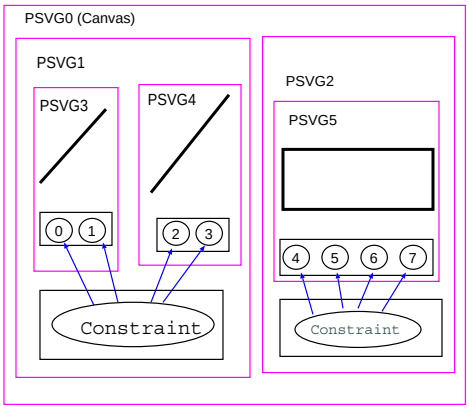

図 3: データ構造の階層化後

図 2 は、ユーザがキャンパス上に 3 つの PSVG を描いた例である。PSVG はインスタンス化されると「PSVG0」などの ID を持つ。PSVG のパラメタは円形表されている。

3.2 現状の課題

現状では、PSVG と制約が一元的な集合として管理されているので、ある図形を動かしたときに、その動かした図形と関係のない制約の計算も (最初から制約の計算をやり直す) 行なっている。

例えば、図 2 において、PSVG0 が動かされたとする。この制約が計算される段階で、理想的には図形が移動した PSVG0 と PSVG1 にかかる制約の計算だけをすれば良い。しかし、現在は PSVG0 と PSVG1 にかかる制約の計算と PSVG2 にかかる制約の計算を行なっている。

3.3 データ構造階層化の設計

上で述べた現状の課題を解決するために、データ構造の階層化を行なった。各 PSVG に制約を直接持たせるように設計を行った。Constraint は、幾何制約と Hybrid 並行制約を指す。

図 3 の例では、相互関係のある PSVG3 と PSVG4 を一つの PSVG として PSVG1 と考えることにした。つまり、制約のかかっている図形同士をグループ化した。PSVG にまたがる制約は、さらに親の PSVG で制約をかければ良い。

3.4 考察

データ構造の階層化を行なうことにより、相互関係のある図形同士だけに対して制約の計算をするだけで良いのでシステムの処理効率の向上なると考えられる。例えば、図 1 において、PSVG0 が動かされたとする。この時、全体にかかる制約の計算を解き直すのではなく、図形が移動した PSVG0 と PSVG1 にかかる制約の計算だけをすると考えられる。

4 まとめと今後の課題

現状の課題である処理効率向上のために、既存のデータ構造を見直し、データ構造の再設計を行った。データ構造の階層化によって、全体にかかる制約の計算を解き直すのではなく、制約の計算の局所化が可能になると考えられる。今回は、設計に時間をかかってしまい実装は思うようにできなかった。今後の課題は、実装し、階層化によってどれくらいのメリットがあるのかを評価することである。

参考文献

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

[2] 石井大輔:制約に基づいたアニメーション作成システム Grifon の設計と実装, 早稲田大学大学院理工学研究科情報科学専攻修士論文,2003.

[3] JHotDraw as Open-Source Project,2003. `http://www.jhotdraw.org`