2005－HI－116（15）
2005／11／18

文字アニメーションの自動合成の試み

水口 充[1]　田中 克己[2]

[1] 情報通信研究機構
[2] 情報通信研究機構／京都大学

文字アニメーションは文字の位置や大きさなどの属性を時間的に変化させることによって文字情報を映像的に表現する手段であり，近年では映画やテレビやゲームなどで多様されている．文字アニメーションは動きによる暗示的な意味表現が可能であるので，テキストベースのコミュニケーションや情報の取捨選択を支援するための情報視覚化手法として利用することが期待できる．本稿では文字アニメーションの表現パタンの分析，および，文章中の単語それぞれに動きを自動的に付与する文字アニメーション合成エンジンの試作について報告する．

An Attempt of an Automatic Animated Text Composer

Mitsuru Minakuchi[1]　Katsumi Tanaka[2]

[1] National Institute of Information and Communications Technology
[2] National Institute of Information and Communications Technology / Kyoto University

Animated text is visual expression that temporally changes display attribute values such as position and size. It is attractive and widely used in films, TV programs, video games, etc. In addition, it may enhance text-based communication and may help users to choose desired information because it can express additional implicit meaning by its motion. We analyzed expression patterns in animated text and developed an automatic animated text composer that attaches proper motion to each word in given texts.

1　はじめに

近年，文字が画面の中を動き回ったり変形したりするようなアニメーション表現を映画やテレビやビデオゲームなどで目にするようになってきた．このような文字アニメーション[1] は一般的になってきたが，ほとんどはデザイナによって試行錯誤的に手作りされている．すなわち，文字アニメーションの歴史は浅く，表現の方法論は十分には確立されていないため，品質はデザイナのスキルに大きく依存している．更に，従来のタイポグラフィやアニメーションなどの表現手法に関する広範な知識が必要なことも文字アニメーションの作成を難しくしている．

このような作成の手間を軽減して，デザインのスキルが無くても簡単に文字アニメーションを利用可能にするために，文字アニメーションを半自動的に生成するツールも提案されている．これらのツールでは，ユーザは入力した文字に対して，予め用意されたアニメーションパタンを選んでパラメータを調整するだけで文字アニメーションを作成することができるが，それでも人手が介在する必要がある．

一方，入力された文章から完全に自動で文字アニメーションを生成することができれば様々なメリットがある．

主なメリットの一つは，ユーザは単に文字を書くだけでよいので，電子メイルやチャットのような文字ベースのコミュニケーションで容易に利用できる．生成された文字アニメーションが洗練されたものでなくても，単に書いた文字が動くだけで楽しいし，時には意外な動きも面白く感じられるかもしれない．更に，手を入れてより良い作品を作るための叩き台として使うこと

[1] kinetic typography[5, 6, 7, 9, 12, 15, 16], temporal typography[18], dynamic text[10], moving type[19] など様々な名称があるが，本稿では「文字アニメーション」と呼ぶことにする．

もできるだろう．

もう一つの大きなメリットは，情報の視覚化ツールとしての可能性である．従来の多くの情報視覚化技法は色や形やアイコンなどの抽象的な表現を使って，データ全体の傾向を理解しやすくすることを目的としていた．しかし，このような表現形態において，個別の情報の詳細を確認するには，例えば見たい情報のアイコンを選択するといったインタラクションが必要であった．これに対し，文字アニメーションは文字そのもの持つ具体的な意味に，動きやタイポグラフィなどの視覚要素で暗示的な意味を付与できる．この特性は，例えば文字アニメーションを注視していなくても情報の取捨選択をある程度行い，興味を持ったら文字を読む，といった日常的な情報の閲覧に利用できる可能性がある．

完全に自動で，品質の良い文字アニメーションを生成することは，自然言語で記述されたシナリオからアニメーションを自動生成するのと同様に，非常にチャレンジングな課題である．しかし，応用分野を限定して程々のものを自動生成することは十分可能であろう．そこで，要素的な表現パタンを「動き部品」として用意し，文章中の各単語の意味に応じて動き部品を割り当てるアプローチで文字アニメーションの自動合成ツールの開発を試みたので報告する．

2　関連研究

文字アニメーションに関する研究は主に，デザイン手法の分析と模索 [9, 15, 18, 19]，作成ツール [6, 7, 10]，表示エンジン [9]，文字アニメーションの応用 [5, 8, 12, 17] に大別される．

文字アニメーションの研究の初期において，Wong は temporal typography を提案し，声の抑揚，リズム，物理的な動きの効果を示した [18]．石崎豪は数多くの作品を通じて kinetic typography の可能性を模索し，同じ文章に対して異なる感情を付与する効果や（図 1），対話中の登場人物，口調，オノマトペの表現可能性を示した．

これらの研究に続いて，デザイン原理の分析を行った研究が幾つかある．Woolman と Bellantoni は空間的な構造，タイポグラフィとしての属性，動きと時間，補助的な図形や音について分析した [19]．Lee らは漫画アニメーションの技法の適用，語り口調，動きの模倣，人物の描写，注意の方向などの表現手法について言及した [9]．

文字アニメーションの作成ツールおよび表示エンジンに関しても幾つかの研究がある．ActiveText [10] は文字に階層構造を適用して動きを設定するツールである．Lee らは信号処理のアプローチを応用して構築された動きのライブラリによる汎用の表示エンジンを開発した [9]．Forlizzi らは Lee らのエンジンを使って文字アニメーションを使ったコミュニケーションのためのオーサリングツール Kinedit を開発した [7]．また Adobe After Effects[2] , Macromedia Flash[2] , Apple Motion[2] などの市販ソフトウェアでは文字のアニメーションの作成をサポートしている．更に，W3C Timed-Text Working Group では時間同期する文字の記述言語を策定中である[3] ．

スクリプト，自然言語シナリオ，ビヘイビアルールなどからアニメーションを自動生成する研究は古くから数多くなされてきた．その一方で，文字アニメーションに特化した自動生成の試みはあまり多くない．Uekita らは日本語のオノマトペの動きの設定に，感性処理した結果に応じたパラメータを利用した [16]．Ishizaki はマルチエージェントモデルを用いてアニメーションするニュース表示システムや電子メイルリーダを提案した [8]．Wang は電気皮膚反応から推定された感情の興奮度に応じて文字の大きさや表示の速度などを自動的に設定するチャットアプリケーションを示した [17]．

3　表現パタンの分析

自然言語からのアニメーション生成は自由度が高いと困難であるので，表現パタンの原則に従った制約を設ける必要がある．既存研究の幾つかは文字アニメーションにおける表現パタンを分析している．しかし，我々は自動合成のためには既存の分析は不十分であると考え，より網羅的な分析・整理を試みた．なお，補助図形や音などの文字以外の要素は文字アニメーションの表現を効果的で魅力的なものにするが，複雑すぎるためここでは扱わない．

3.1　文字アニメーションの定義

まず，文字アニメーションとは何か，について考える．SMIL animation[2] ではアニメーションを，対象となる要素の時間的な操作と定義している．例えば，位置を時間的に更新すれば平行移動となり，大きさを更新すればズームになる．また，SMIL ではある要素に対する時間の進行速度や変化率の調整といった時間操作をサポートしている．

[2] それぞれ Adobe 社，Macromedia 社，Apple 社の登録商標．
[3] http://w3.org/AudioVideo/TT/

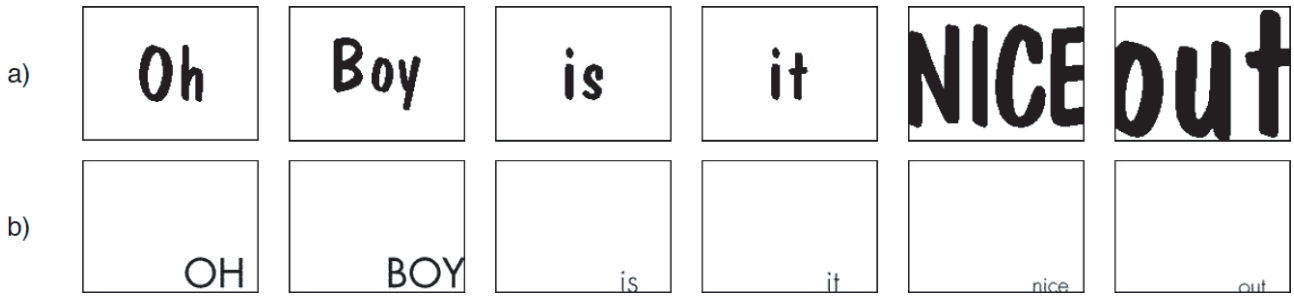

図 1: 同じ文章で異なる感情を表現する例（文献 [9] より引用）.

この定義に基づき，アニメーションを**変化の対象となる表現属性**と**時間的な変化の仕方**との 2 つの要素に分けて考える．文字アニメーションにおいては，前者の表現属性は更に空間の属性と文字自体の属性とに分けられる．空間の属性は位置，大きさ，方向，文字間の間隔などである．文字の属性は色やフォントなどである．時間的な変化は不変，離散，1 次，2 次，などに分類できる．文字アニメーションにおける大半の動きはこれらの要素の組み合わせで表現可能である．

3.2 文字アニメーションの特性

文字アニメーションの表現特性には 3 つの側面が挙げられる．

1 つ目は，明確なメッセージを持つという**文字の特性**である．文字アニメーションにおいてはこの特性は最も支配的である．通常は文字自体の意味を動きの持つ意味が置き換えることはできない [9].

我々は，文字と動きの関係を同意，反意，衝突の 3 つに分類した．同意は動きの意味が文字の意味と類似している関係で，強調や補足に使われる（図 1-a）．反意は相互に反する意味を持つ関係で，対照による強調や反語的表現（図 1-b）に使われる．衝突は動きと文字の意味が無関係な場合である．衝突は難解になりがちなので使いにくいが，効果的に使うと俳句のような一見無関係な意味をぶつけ合うことによる印象的な表現ができる．

2 つ目は**タイポグラフィの特性**である．従来の静的なタイポグラフィは単なる装飾だけでなく，文章の構造の視覚化，読解の流れの誘導，雰囲気の暗示的な表出などの意味表現を行っている．文字アニメーションにおいても同様の意味表現が可能である．

3 つ目は**動画としての特性**である．動き自体は魅力的である上，注意を惹いたり視線を誘導する効果がある．同じ視点に単語を切り替えて表示していく Rapid Serial Visual Presentation[13] のような手法による読解性の支援効果もある．

3.3 表現のパタン

文字アニメーションが暗示的な意味を付与できるということは，動き自体が意味を持ちうることを示している．この，動きのみによる意味表現のパタンを，**静的な表現の時間的変化**，**実世界の動きの模倣**，**記号論的な動き**，の 3 つに分けてみる．表 1 は本節の分類をまとめたものである．

表 1: 表現のパタンの分類.

静的な表現の時間的変化	
実世界の動きの模倣	物理的な動き
	生理現象
	身体言語
記号論的な動き	音の視覚化
	誇張
	象徴

静的な表現は従来のタイポグラフィに基づいている．例えば Times や明朝体などのセリフ系のフォントは Helvetica やゴシック体などのサンセリフ系よりも真面目でフォーマルな印象を与えやすい，太字やイタリック体は重要であることを示す，文字間の間隔が広いと軽く拡張性のある印象を与える，など．このような静的な属性の変化は対照による強調表現となる．例えば，文字が大きくなれば閲覧者の注意を惹き，その部分が強調されていることを伝えられる（図 2）．あるいは，状態の変化を表現することもできる．例えば，文字間の間隔を広げて拡張や成長を表現できる（図 3）．

図 2: 文字の拡大の例.

図 3: 文字間の拡張の例.

実世界の動きの模倣は，巧くデザインされていれば比較的意味を理解しやすい．ここでは，物理的な動き，生理現象，身体言語の 3 つの下位分類に分けてみる．

物理的な動きは，落下や跳ね返りなどの自然現象による動きの模倣表現である．この表現自体の意味は暗示的であるが，文字の持つ意味と組み合わせると効果的になりうる．例えば，snow という文字がゆっくり下方に動くと穏やかな雪の印象を与えるし（図 4-a），速く斜めに移動すると激しい吹雪の印象を与える（図 4-b）．この表現パタンは主語（文字）と動作（動き）で構成されるので，自然言語処理で比較的容易に自動生成できるであろう．

生理現象は，震える，涙を流す，赤くなる，といった人間や動物の反応を表現するパタンである．例えば「お化け」という文字を青くして震えさせるというように，この表現パタンは話者の感情を表現する場合などに使われる．感情の状態を抽出して対応する生理現象の動きパタンを自動的に付与することは可能であろう．

身体言語は実際に見える動きであるのでここに分類したが，記号論的な動きに近く，文化的背景に大きく依存している．身体言語は文字自体で表現するには複雑な動きであるものが多いが，意図的な特殊な動きであり明確なメッセージ性を持つので自動生成には使いやすい．

記号論的な動きは記号や標識などの同じような，共通の認識・理解に立脚している意味表現であり，仮想的な動きを視覚化したものである．ここでも，音の視覚化，誇張，象徴の 3 つの下位分類に分ける．

音の視覚化は比較的容易である．例えば音の大きさは文字の大きさ，ピッチは鉛直方向の位置，テンポは動きの周期，といった具合の，サンプリング波形の模倣が主な表現手法である．音の視覚化は会話文に適用されることが多く，口調を効果的に表現できる [9]．もう一つの主な表現はオノマトペの視覚化である．オノマトペ自体が記号的であるが，同意による相乗効果を利用して印象的な動きを表現できる（図 5）．

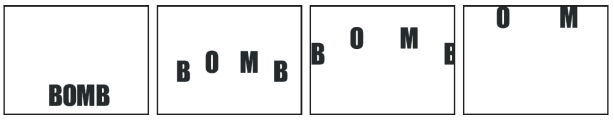

図 5: オノマトペの表現例.

誇張は従来より漫画アニメーションで使われている技法である [14]．これについては Lee らが分析しているので [9] 省略する．

象徴はコンテキストや背景知識に依存する，抽象度の高い表現パタンである．例えば，雪は寒い，寂しいといった感覚を象徴する一方で，スキーやクリスマスのコンテキストでは楽しい印象も与えうる．あるいは，雪を見たことの無い人には意味が理解しにくい．

誇張や象徴の表現パタンの成否は文化的背景に大きく依存するので，これらの表現を自動生成で利用する際には適用範囲を限定する必要があるだろう．

3.4 時間と空間

文字アニメーションは時間と空間を利用した表現であり，対話，視線の誘導，情報構造の表現，疑似 3 次元などの効果を生み出す．

対話は，話者を区別するために空間を使う手法である．視線の誘導は，動くものは見ている人の注意を惹く効果を利用したものである．これらの効果は Lee らの分析 [9] があるので省略する．自動生成に関しては，話者が特定できれば前者は容易であるが，後者は高度な誘導は往々にして意図的であるので難しい．

情報構造の表現は，静的なタイポグラフィが文字の大きさや配置などの空間的属性で文章の重要度や段落間の関係などを表現していたのに加え，文字アニメーションでは時間的表現を利用できる．時間的な間（ま）は情報の仕切りとして機能するし，会話文では話すテンポを表現できる．この効果は比較的容易に自動生成できる．

疑似 3 次元は運動視差を利用した効果で，投影法や物体間の重なりと併用することで奥行き感を表現できる．3 次元的な動きはコンピュータで容易に計算できるが，意味的な効果については更なる分析を要する．

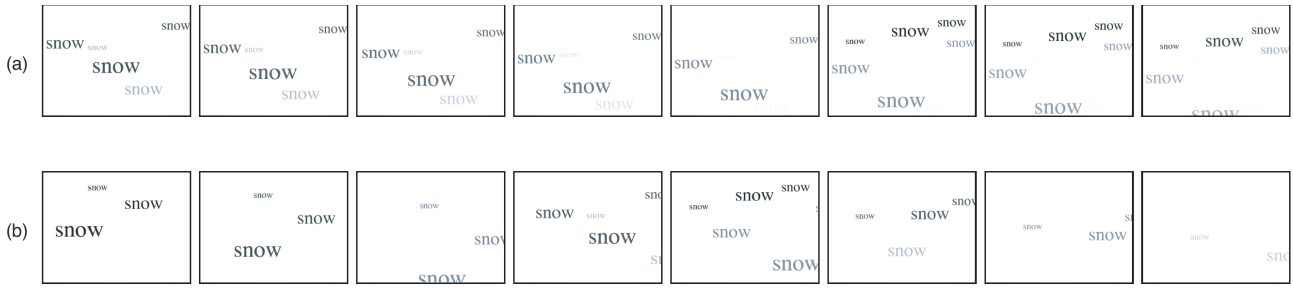

図 4: 物理的な動きの例.

4 実装

4.1 アーキテクチャ

高品質で芸術性の高い文字アニメーションの自動生成は非常に難しい課題であるが，用途に応じて表現のパタンを限定すれば付加的な意味を表現する程度のものの自動生成は可能であると考え，用途に応じてカスタマイズ可能な構造の文字アニメーション自動合成エンジンを設計した.

基本構造を図 6 に示す．大きく，テキスト解析モジュール，動き部品合成モジュール，データ変換モジュールの 3 つから構成される.

テキスト解析モジュールは，自然言語文章から意味や構造を抽出する．このモジュールはアプリケーションに応じた処理で構成される．現在の実装は最も単純な構成の例である．まず，入力文章を形態素解析し単語に分解する．次にそれぞれの単語[4] 毎に，その単語をキーワードとして持つ動き部品が動き部品 DB に登録されているかチェックする．ここで動き部品とは，キーワードで表される意味を表現している動きのパタンを後述の記述言語で記述したものである．登録されていなければ，シソーラスを使って類義語や上位概念語などの関連語を得て，それらの関連語を持つ動き部品の有無をチェックする．この時，動き部品ごとに設定された，前章で説明したような表現のパタンの分類を参照して，できるだけ同じ表現パタンの動き部品を優先的に選ぶ.

次に，動き部品合成モジュールは，各単語ごとに得られた動き部品を単語に割当て[5] ，動きの継続時間や変化の強さなどのパラメータを調整して，すべての単語の動きとしてまとめあげ，後述の記述言語による中間アニメーションデータを出力する．動き部品が見つからなかった単語に対してはデフォルトの動き（通常，静止）を割り当てる.

データ変換モジュールは中間アニメーションデータを最終のフォーマットに変換して出力する．動画，SVG[4], Macromedia Flash などのデータ形式や再生エンジン [9, 10] などの多様な表現手段に広く対応しうるようにプラグイン構造を採っている.

4.2 文字アニメーション記述言語

我々のアプローチでは，動き部品，すなわち，実際に表示される文字とは独立に意味に対応付けられた動きパタンを扱う．また，多様なビューアで閲覧できるように，中間的なデータ形式を出力している．これらの目的のために，文字アニメーションの記述に特化した XML ベースの記述言語 Kinetic Typography Description Language (KTDL) を設計した．KTDL は文書構造，テキスト，フォント，基本的な動き，動きの定義，時間の記述のための要素を提供する.

`ktdl` 要素は KTDL 文書のルート要素であり，描画領域に関する属性（表示の幅，高さ，アスペクト比，背景色など）を持つ.

すべての文字は `text` 要素でマークアップされる．フォントは `font` 要素で指定される．`font` 要素は CSS[1] で定義されているフォントに関する属性の幾つかを利用できる．また，表示する際に使用できるフォントはシステム依存であるので，論理フォント名と物理フォント名をマッピングするために `fontmapping` 要素を用意した．`font` 要素は `fontmapping` 要素で定義される論理フォント名を指定することができる.

基本的な動きとしては，等速運動 `translate`，等加速度移動 `accelerate`，回転 `rotate`，拡大／縮小 `zoom` に加え，色と透明度の変更 `changeColor`，`changeOpacity` 要素を定義した．通常の 2 次元的な動きはこれらの組み合わせで表現できる．これらの要

[4] 日本語の場合は助詞，接頭詞，接尾詞を除く.
[5] 日本語の場合は文節ごとに割り当てる.

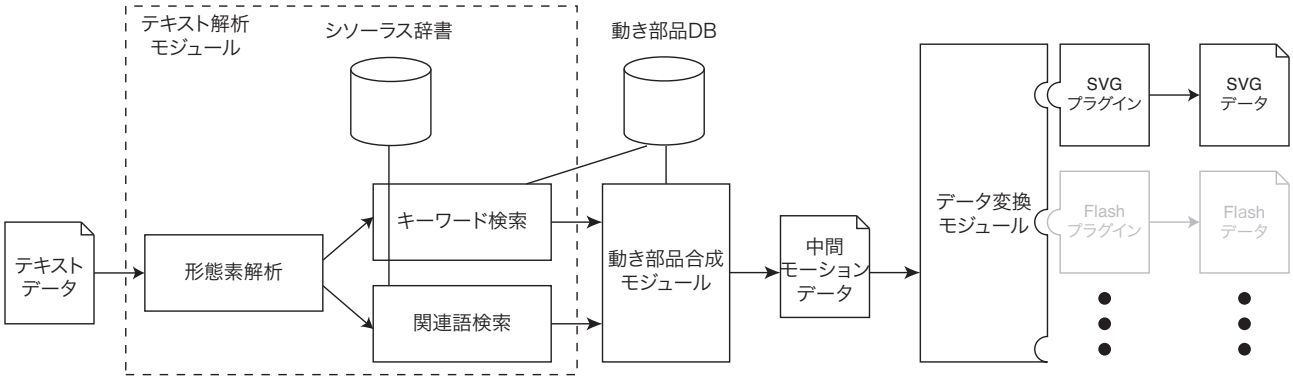

図 6: 自動合成エンジンの構造（グレーの部分は未実装）.

素は text 要素の子要素として記述され，複数の動きは重畳される．これらの要素は，開始位置，終了位置，速度などの要素ごとの動きに関するパラメータの属性と，開始時刻と継続時間の属性を持つ．

defAnimation 要素は動き部品を定義するために使われる．子要素は基本的な動きの要素群である．また，最終的な出力フォーマットに依存する記述も埋め込めるようにした．defAnimation 要素で定義された動き部品は useAnimation 要素で参照される．useAnimation 要素も text 要素の子要素として記述される．

defAnimation 要素は，文字に適切な動き部品を割り当てるために参照するための属性を持つ．keyword 属性は動き部品が表現しようとしている意味を表し，一つないし複数の単語で記述される．category 属性は 3.3 節で述べたような表現のパタンであり，static，physical，gesture（生理現象と身体言語を統合），phonetics，semiotic（誇張と象徴を統合），misc のいずれかの値をとる．

時間の記述は time 要素で行う．time 要素は，開始時刻を表す begin 属性と継続時間を表す dur 属性を持ち，共に絶対時間あるいは相対時間の値を持つことができる．文字アニメーションの場合は動きは逐次的であることが多いので，現在の仕様では time 要素での簡潔な記述にとどめていが，将来的には SMIL 2.0 timing and synchronization module[3] の取り込みも考えられる．

4.3 動作例

図 7 は出力された文字アニメーションの例である．

まず，元の文章 “World regrets North Korea’s quitting nuke talks” を，それぞれの単語に分割し[6]，各単語あるいは関連語を keyword 属性に持つ動き部品を検索した．この例では，それぞれ “world” “unhappy” “nation” “move” “war” “speak” の keyword 属性を持つ動き部品となった．

次に，動き部品合成モジュールは検索された動き部品を各単語に割り当てて中間モーションデータとなる KTDL 文書を生成した．具体的には，各単語を text 要素でマークアップし，動き部品を参照する refAnimation 要素を子要素に追加した．更に，全体の時間を調整して time 要素を記述し，中間モーションデータである KTDL 文書を出力した（図 8）.

最後にデータ変換モジュールは中間 KTDL 文書を解釈してターゲットの形式のデータを出力した（図 9）.

4.4 議論

現在の実装では，文章から SVG 形式の文字アニメーションデータをとりあえず自動生成できている．しかし，解決するべき大きな課題が 3 つある．

一つは，生成される文字アニメーションの品質がテキスト解析モジュールと動き部品 DB に大きく依存する点である．現在の実装では単語の意味を基本として動き部品を割り当てており，各単語ごとの関連性や文全体での意味などは考慮していない．このため，単語がばらばらに動いているようになりがちである．フレーズレベルでの処理や文意の反映などの，メタレベルの意味解析を導入する必要がある．また，動き部品 DB については手作業で充実させていくしかないが，一度作成した文字アニメーションを部品化することは可能であるので，再利用の方法について検討したい．

二つ目は，出力形式依存の表現の制約である．例えば，ランダムに振動するような動きや単語を構成する文字が増えていくような動きは文字アニメーションで

[6] North Korea’s は 1 語として処理した.

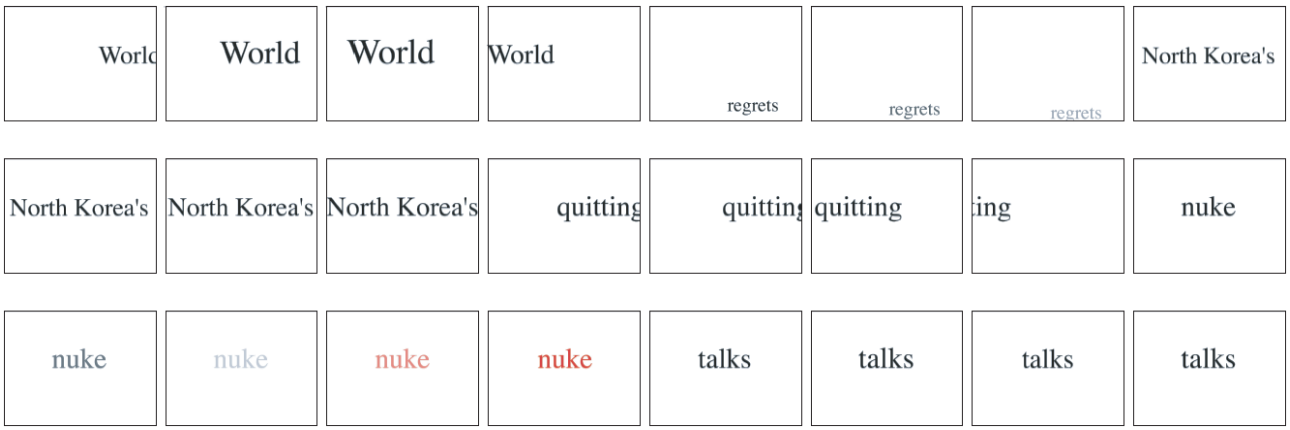

図 7: 自動生成された文字アニメーションの例.

```
<?xml version="1.0"?>
<ktdl width="800" height="600" color="#000" background-color="#fff">

<font font-family="politics"/>

<time dur="2s">
  <text text-anchor="middle">
    World
    <useAnimation ref="world"/>
  </text>
</time>
<time dur="1s">
  <text text-anchor="middle">
    regrets
    <useAnimation ref="unhappy"/>
  </text>
</tiem>
  .....
</ktdl>
```

```
<fontmapping name="politics">Times</fontmapping>
```

```
<defAnimation id="world" keyword="world, universe">
  <translate from="100% 50%" to="50% 50%" dur="1s"/>
  <zoom from="60" to="100" dur="1s"/>
  <translate from="50% 50%" to="0% 50%" begin="1s" dur="1s"/>
  <zoom from="100" to="60" begin="1s" dur="1s"/>
</defAnimation>
```

```
<defAnimation id="unhappy" keyword="unhappy">
  <translate from="80% 95%" to="80% 95%" dur="1s"/>
  <zoom from="40" to="20" dur="1s"/>
  <changeOpacity from="1.0" to="0.2" dur="1s"/>
</defAnimation>
```

図 8: 中間データの例．□内は文書内の別の部分に記述されていて矢印の始点の要素から参照されている．

```
<?xml version="1.0"?>
<svg fill="#000">
  <rect x="0" y="0" width="800" height="600" fill="#fff"/>
  <text font-family="Times" text-anchor="middle" fill-opacity="0" transform="translate(0,0)">
    World
    <set attributeName="fill-opacity" to="1.0" begin="0s" dur="2s"/>
    <animateTransform attributeName="transform" type="translate" from="800 300" to="400 300" begin="0s"
     dur="1s" calcMode="linear" additive="sum"/>
    <animate attributeType="CSS" attributeName="font-size" from="60.0" to="100.0" begin="0s" dur="1s"/>
    <animateTransform attributeName="transform" type="translate" from="400 300" to="0 300" begin="1s"
     dur="1s" calcMode="linear" additive="sum"/>
    <animate attributeType="CSS" attributeName="font-size" from="100.0" to="60.0" begin="1s" dur="1s"/>
  </text>
  .....
</svg>
```

図 9: SVG 形式による最終出力データの例.

よく使われる表現であるが，現在の KTDL では固有の要素としてはサポートしておらず，複雑な記述で擬似的に表現するしかない．また，これらの表現を SVG で記述する際も，スクリプトを利用して要素や属性の値を変更する必要がある．汎用的に使われる動きの要素の KTDL への追加と，多様なデータ形式への変換方法を検討する必要がある．

三つ目は，時間同期の制御である．現在の実装では各単語の動きは基本的には直列的につなぎ合わされる．動き部品の開始時刻に相対的な負の値を設定すればクロスオーバーする動きが表現できるが，前後の動きが妥当となる保証は難しい．また，会話文の表現のよう

な並列して動く表現を生成するには，文単位の処理に対応できる動き部品の定義方法が必要になる．このために，テンプレート的に文字を割り当てるような記述方法を検討したい．

5 まとめ

本稿では，文章から文字アニメーションを自動的に生成することを目標として，表現技法の分析を行い，文字アニメーションの記述言語および合成エンジンのプロトタイプについて紹介した．現在の実装はまだ動き始めたばかりの段階で不十分な点が多い．今後，4.4 節で述べた内容に関して改良していく予定である．

また，情報視覚化としての効果についても調査していきたい．我々の予備実験では，周辺ディスプレイに文字を表示した際に，動く文字は可読性は高くないものの，注意を惹きやすく面白みも感じられる可能性を示した．この特性を利用して，ユビキタス環境における掲示板のような情報提示システムへの応用を進めている [11].

参考文献

[1] Cascading Style Sheets, level 2 CSS2 specification, http://www.w3.org/TR/REC-CSS2/

[2] SMIL Animation, 2001. http://www.w3.org/TR/2001/REC-smil-animation-20010904/

[3] Synchronized Multimedia Integration Language (SMIL 2.0) - [second edition], 2005. http://www.w3.org/TR/2005/REC-SMIL2-20050107/

[4] Scalable Vector Graphics (SVG) 1.1 Specification, 2003. http://www.w3.org/TR/SVG11/

[5] Bodine, K. and Pignol, M. Kinetic typography-based instant messaging. in *CHI '03 extended abstracts*, pp. 914–915, 2003.

[6] Chao, C. and Maeda, J. Concrete programming paradigm for kinetic typography. in *Proceedings of IEEE VL'97*, pp. 446–447, 1997.

[7] Forlizzi, J., Lee, J., and Hudson, S. The kinedit system: affective messages using dynamic texts. in *Proceedings of CHI '03*, pp. 377–384, 2003.

[8] Ishizaki, S. Multiagent model of dynamic design: visualization as an emergent behavior of active design agents. in *Proceedings of CHI '96*, pp. 347–354, 1996.

[9] Lee, J., Forlizzi, J., and Hudson, S. The kinetic typography engine: an extensible system for animating expressive text. in *Proceedings of UIST '02*, pp. 81–90, 2002.

[10] Lewis, J. and Weyers, A. ActiveText: a method for creating dynamic and interactive texts. in *Proceedings of UIST '99*, pp. 131–140, 1999.

[11] Minakuchi, M., Nakamura, S., and Tanaka, K. AmbientBrowser: Web Browser for Everyday Enrichment. in *Proceedings of INTETAIN2005*, pp. 92–101, 2005. (to appear)

[12] Möhler, G., Osen, M., and Harrikari, H. A user interface framework for kinetic typography-enabled messaging applications. in *CHI '04 extended abstracts*, pp. 1505–1508, 2004.

[13] Potter, M. Rapid serial visual presentation (RSVP): a method for studying language processing. *New Methods in Reading Comprehension Research*, pp. 91–118, 1984.

[14] Thomas, F. and Johnston, O. *The Illusion of Life: Disney Animation*, Abbeville Press, 1981.

[15] Uekita, Y., Harada, Y., and Furukata, M. The possibility of kinetic typography expression in the internet art museum. in *Proceedings of IEEE SMC'99*. pp. 230–235, 1999.

[16] Uekita, Y., Sakamoto, J., and Furukata, M. Composing motion grammar of kinetic typography. in *Proceedings of IEEE VL'00*, pp. 91–92, 2000.

[17] Wang, H., Prendinger, H., and Igarashi, T. Communicating emotions in online chat using physiological sensors and animated text. in *CHI '04 extended abstracts*, pp. 1171–1174, 2004.

[18] Wong, Y. Temporal typography: a proposal to enrich written expression. in *CHI '96 Conference companion*, pp. 408–409, 1996.

[19] Woolman, M. and Bellantoni, J. *Moving Type*, F+W Publications, 2001.