

卒 業 論 文

同期マルチメディア言語を用いた 研究キーワード紹介システム

氏 名：末富 俊樹

学籍番号：2210980107-4

指導教員：山崎 勝弘 教授

提出日：2002年2月18日

立命館大学 理工学部 情報学科

内容梗概

近年急速にブロードバンドといった高速ネットワークを安価に利用できる時代になってきた。このことは、ユーザの需要に応えた自然な流れだといえる。その中で、リアルタイム性が要求され、ストリーミング技術の必要性は更に高まっている。ストリーミングにより動画や音声の配信が容易になり、大容量のコンテンツ配信が増えてきた。しかし、文字情報や静止画といった比較的軽いものならともかく、動画や音声をスムーズにストリーミング配信するには、その劣化が避けられない。

本研究では、複数のメディアを使って情報を補うことによって回避しようとする考え方のもとに、同期マルチメディアという技術を使ってのコンテンツ配信を利用した。SMIL という同期マルチメディア言語を使用し、複数のメディアを使って如何に軽いデータで、どこまで視聴者に伝わりやすいコンテンツが作成できるかを考えた。

本論文では、この技術を使って複数メディアを同期させたコンテンツのストリーミング配信システムを構築した。アニメーションや音声のデータをできるだけ軽くして、視聴者が見るのに快適な構成や各メディアの配置を配慮することもテーマに置いた。今回は Web サーバでの擬似ストリーミングで実験したが、SMIL による同期パターンを様々試した結果次のような事がわかった。CG や Flash によるアニメーションは、それぞれ個々では情報量が多く、また低容量でストリーミングには向いているが、SMIL で同期させた場合に動作の不具合が見られたり、指定時間に同期させた他メディアの再生が間に合わなかったりなど、ストリーミングに対する向き不向きが、各メディアの種類、数、そしてその配置などに左右されることが実験を通してわかった。

このように、複数のメディアをどこまで同期させ、どのようなパターンで同期をとればどれ程軽く、そして視聴者に快適に並列処理の基礎を伝えることができるかをテーマにコンテンツを作成し、考察した。

目次：

1	はじめに.....	5
2	ストリーミングと同期マルチメディアの技術.....	7
2.1	ブロードバンド時代のストリーミング技術.....	7
2.2	同期マルチメディアの技術.....	9
2.3	同期マルチメディア言語：SMIL.....	9
3	同期マルチメディアを用いた研究キーワード紹介システム.....	12
3.1	システムの構成.....	12
3.2	各キーワードの内容.....	13
3.3	各メディアの同期方法.....	15
4	並列処理キーワードの作成.....	16
4.1	使用するマルチメディア.....	16
4.1.1	CGの作成.....	16
4.1.2	アニメーションの作成.....	18
4.1.3	テキストの作成.....	19
4.1.4	音声の作成.....	20
4.1.5	静止画の作成.....	21
4.2	各キーワードの内容と同期方法.....	22
5	システムの考察.....	36
6	おわりに.....	37
	謝辞.....	38
	参考文献.....	39
	付録 SMILの同期部分HTMLファイル.....	40

図目次：

図1	：ストリーミングとダウンロードの待ち時間の違い.....	7
図2	：ストリーミングの手順.....	8
図3	：SMILの仕組み.....	9
図4	：メタファイル.....	10
図5	：画面のインライン表示.....	11
図6	：コントロールバーのインライン表示.....	11
図7	：同期表現のパターン.....	12
図8	：RealTextによる時間指定.....	15
図9	：SMILによる同期表現.....	15

図 1 0 : モデラー画面.....	16
図 1 1 : レイアウト画面.....	17
図 1 2 : Flash の画面.....	18
図 1 3 : 負荷均衡の RealText コード.....	19
図 1 4 : RealText の表示.....	19
図 1 5 : サウンドレコーダー.....	20
図 1 6 : RealProducer.....	20
図 1 7 : 並列処理とは (1) の 5 秒後.....	22
図 1 8 : 並列処理とは (1) の 10 秒後.....	23
図 1 9 : 並列処理とは (1) の 16 秒後.....	23
図 2 0 : スケーラビリティのある部分.....	24
図 2 1 : スケーラビリティのない部分.....	24
図 2 2 : 並列処理とは (2) の 9 秒後.....	25
図 2 3 : 並列処理とは (2) の 23 秒後.....	25
図 2 4 : 負荷均衡の初期画面.....	26
図 2 5 : 始めの仕事の振り分け.....	27
図 2 6 : 始めの振り分け後.....	27
図 2 7 : 右端のスレーブの仕事量が最大.....	27
図 2 8 : 最終状態.....	27
図 2 9 : 同期とは.....	28
図 3 0 : 並列処理とは (3)	28
図 3 1 : 分割統治法.....	29
図 3 2 : 分割統治法の初期状態.....	30
図 3 3 : 1 つ下位に分割.....	30
図 3 4 : 最下位に分割.....	30
図 3 5 : 再帰的な結合.....	30
図 3 6 : プロセッサファーム.....	31
図 3 7 : 初期画像を分割.....	32
図 3 8 : 処理中.....	32
図 3 9 : 結果画像の回収.....	32
図 4 0 : エッジ抽出最終状態.....	32
図 4 1 : プロセスネットワーク.....	33
図 4 2 : 左のロボットが作業.....	33
図 4 3 : 中央のロボットが作業.....	33
図 4 4 : 右のロボットが作業.....	33
図 4 5 : 繰り返し変換とは.....	34

図 4 6 : 3 体下降中.....	35
図 4 7 : 1 体完成.....	35
図 4 8 : 残り 2 体下降中.....	35
図 4 9 : 2 体目完成.....	35

表目次 :

表 1 : ストリーム方式の長所と短所.....	8
表 2 : SMIL で使用可能なファイル形式.....	10
表 3 : 各コンテンツ.....	12
表 4 : RealPix.....	21

1 はじめに

ブロードバンドといった高速ネットワークを安価に利用できる時代になってきた。つい先頃に普及していた ISDN から、今では ADSL にとって替わり、更には CATV や光ファイバーケーブルなどの高速回線の時代がやってくると言われる。このことの示す意味は大きく、インターネットそのものが高速化することによって大容量の映像を配信することも可能になった。先の米国大統領選挙で Bush 氏と Goa 氏が接戦を繰り広げる中、多くの国民が TV に代わって Web サイトを見ていたという[7]。これは、リアルタイム性という面で、視聴者が TV よりもインターネットを選んだと言える。

リアルタイム性という点において、ユーザのニーズに応えるべくストリーミングという技術がある。ストリーミングとは、インターネット上で音声や映像といったマルチメディアデータを流れるようにユーザに配信できる技術で、いわばテレビやラジオのように放送のようなものである。

しかし、高速なネットワークを利用しても静止画やテキストなどの比較的軽いデータ配信は問題にならないが、動画や音声などの大容量のコンテンツ配信がサーバに大きな負担をかけるのは間違いない。ストリーミング技術では、配信するデータに圧縮をかけて、少しでもストリームデータを小さなものにしようとする。こうすることで発生する問題がコンテンツの情報劣化である。映像を配信しても劣化したものではユーザに正確な情報が伝わらない。このような劣化した情報を他のメディアで補おうというのが、シンクロナイズド・マルチメディアである。この技術は、劣化した映像に音声や文字などで情報を加え、ユーザによりはっきりした情報を提供するものである。

本研究では、シンクロナイズド・マルチメディアを利用して、本研究室の研究キーワードを紹介するシステムを作成した。このシンクロナイズド・マルチメディアを利用することにより動画の情報劣化を補い、また複数のメディアを同期させることによって研究キーワードを視覚と聴覚に訴えることによってより理解しやすいものを目指した。

まず、本研究室の研究テーマである並列処理の基本的な考え方を 3 つのコンテンツに分けて作成した。これには文字情報とアニメーション、必要に応じて静止画と音声を導入した。この 3 つの並列処理の説明文中に、理解しづらいキーワードが出現する。この場合はそこにリンクを貼り、別途作成したコンテンツで説明した。

本研究では SMIL という同期マルチメディア言語を利用して、アニメーション、CG、文字、静止画、そして音声を同期させてコンテンツを作成した。これにより、RealSystem のストリーミングを使用し、様々なファイルを統合した。文字では RealText、アニメーションでは RealFlash、CG では RealVideo、静止画では ImageFile、そして音声では RealAudio を SMIL で使用するファイル形式として扱った。

本論文では、2 章でストリーミング技術と同期マルチメディア言語である SMIL につ

いて述べる。3章では、作成したシステムの構成、各キーワードの内容、そして各キーワードの同期部分の説明をする。4章では、4.1で各ファイルの作成を、4.2ではその同期方法を述べる。5章で作成したシステムの考察を述べ、6章で本論文をまとめて考察をする。

2 ストリーミングと同期マルチメディアの技術

2.1 ブロードバンド時代のストリーミング技術

ブロードバンドの普及によってビジネス構造の選択が広がりつつある。動画を含む配信サービスが展開し、ストリーミング配信は重要な技術である。

ストリーミング技術では動画や音声といったマルチメディアデータを全てダウンロードする前に、少しずつバッファにためながら再生できる。こうすることで文字通り流れるように映像や音楽を楽しむことができる。これはダウンロード再生と比べると待ち時間が大きく違うことがわかる（図1）。

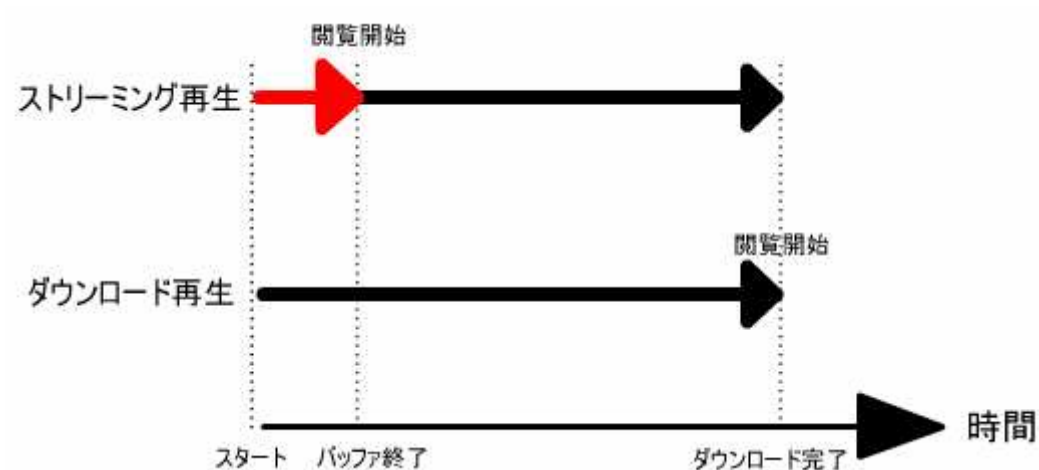


図1：ストリーミングとダウンロードの待ち時間の違い

ストリーミングでは、いったんアナログのソースをデジタル形式にする。このままではデータが大きいためエンコード、つまり圧縮をかけてネットワークに配信する。受信側はこれをデコードし、先ほどと逆の手順で視聴する（図2）。



図2：ストリーミングの手順

ストリーミング技術には大きく分けて2つの方式がある。オンデマンド方式とライブ方式である。前者は、サーバに蓄積された映像をユーザが見たい時に選択して視聴できるものであり、一般のTVとは違って時間を選ばずにいられるシステムである。この方式では、ユーザに時間とコンテンツの選択の自由が与えられるメリットもあるが、同じサーバのコンテンツを複数のユーザが視聴するので負荷がかかってしまうというデメリットもある。後者は、いわばTVのようなもので、その時に配信されている映像しか見られない。しかしオンデマンドと違ってサーバの負荷が少量で済むというメリットもある。これは中継している映像が大きくても、ストリーミングに必要な記憶容量はそのうちのほんの少量すむからだ。また、ライブ方式の1つに、ライブ+オンデマンド方式というものもある。これは、サーバに保存された既存のデータをあたかもライブ放送のように流すもので、オンデマンド方式のように見たい映像を選択はできるが、ライブ方式のように既にデータが流れている可能性がある。つまり、オンデマンド方式のメリットである選択の自由と、ライブ方式のメリットであるサーバの軽い負荷が可能なシステムであるといえる。(表1)

表1：ストリーム方式の長所と短所

	長所	短所
ライブ方式	<ul style="list-style-type: none"> ・速報性あり ・サーバ負荷少量 	<ul style="list-style-type: none"> ・ユーザによる時間選択不可 ・ユーザによるコンテンツ選択不可
オンデマンド方式	<ul style="list-style-type: none"> ・ユーザによる時間選択可 ・ユーザによるコンテンツ選択可 	<ul style="list-style-type: none"> ・サーバ負荷大 ・速報性なし
ライブ+オンデマンド方式	<ul style="list-style-type: none"> ・サーバ負荷少量 ・ユーザによるコンテンツ選択可 	<ul style="list-style-type: none"> ・速報性なし ・ユーザによる時間選択不可

2.2 同期マルチメディアの技術

2.1で述べたストリーミング技術には、圧縮による情報の劣化が避けられない。そこで同期マルチメディアの技術を使う。音声や映像それぞれが単体であれば、圧縮時の劣化により視聴者にその内容が伝わり難いが、例えば映像を音声で説明しつつ静止画や文字テキストを付加して同時に流すことができれば、それぞれの内容を補ったものになる。本研究ではSMIL(synchronized Multimedia Integration Language)という様々なメディアを統合的にあつかう言語を利用した。

2.3 同期マルチメディア言語：SMIL

SMIL は1998年6月に「SMIL1.0」として策定されたばかりの言語で、RealNetworks社によるストリーミングシステム(RealSystem)がコンテンツ制作における標準になっている[3]。SMILでは、文字、音声、映像、画像、そしてアニメーションといった複数のメディアを統合したコンテンツ制作だけでなく、時間制御することによって画像や文字の表示のタイミングを計ったり、途中の文字にリンクを貼って別のページを表示することを可能とするマルチメディアプレゼンテーションを作成できる(図3)。

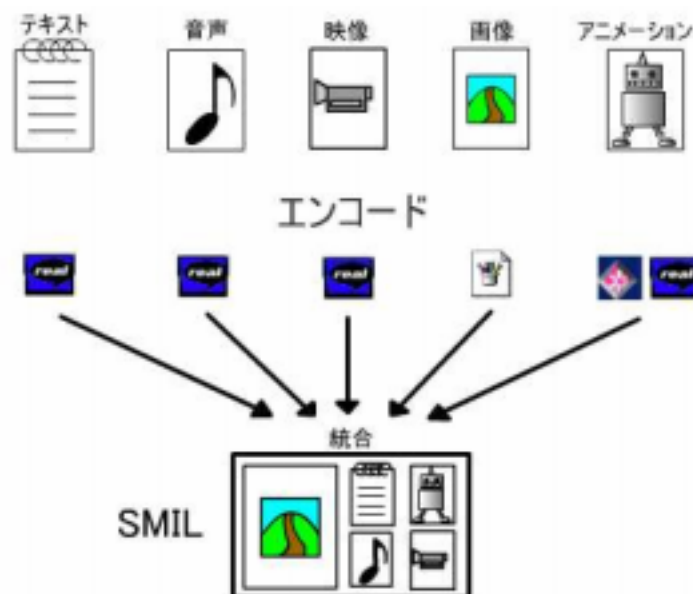


図3：SMILの仕組み

また、SMIL で扱うことのできるファイル形式は、RealSystem では RelasystemG2 以降、多くの形式をサポートしている（表 2）。

表 2：SMIL で使用可能なファイル形式

形式	拡張子	ファイルの内容
RealAudio	.ra .rm	音声
RealVideo	.rm .rv	ビデオ映像
RealText	.rt	テキスト
RealPix	.rp	gif jpegファイルの制御
RealFlash	.swf	Flashアニメーション
TextFile	.txt	通常のテキスト
ImageFile	.gif .jpg	画像

これらのファイルを統合してマルチメディアプレゼンテーションを作成した。統合する時は、Windows なら「メモ帳」、Mac なら「SimpleText」等のテキストエディタを使うことができる。統合したファイルを「.smi」もしくは「.smil」といった形式で保存する。作成したファイルは RealPlayer で見ることができる。しかし、この作成された SMIL ファイルは、アップロードして直接リンクを貼っても再生されないので、対象となるファイルの URL を貼ったメタファイルを作成する必要がある。実際にアップロードする際には、このメタファイルにリンクを貼って再生することになる（図 4）。

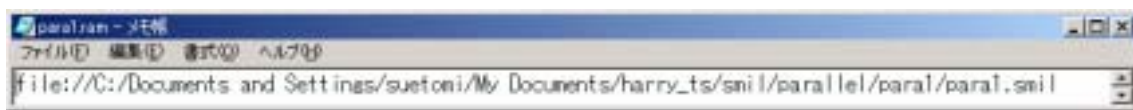


図 4：メタファイル

メタファイルには 2 種類あり、「.ram」形式で保存するポップアップ表示と、「.rpm」形式で保存するインライン表示がある。共に絶対パスでなければならない。ポップアップ表示の場合はリンクをクリックすることで直接 RealPlayer が作動するが、インライン表示では、ブラウザにコンテンツが埋め込まれた形で、RealPlayer に見られるようなコントロールバーを付加した形で動作する。インライン表示は HTML 内に画面の表示（図 5）と、コントロールバーの表示のためのコードを追加しなければならない（図 6）。

```

<p align="center">
<object id=video1
classid="clsid:CFCDAA03-8BE4-11cf-B84B-0020AFB8C9FA"
height="570" width="800">
  <param name="controls" value="ImageWindow">
  <param name="console" value="Clip1">
  <param name="autostart" value="false">
  <param name="src" value="para1.rpm">
  <embed src="para1.rpm" type="audio/x-pn-realaudio-plugin" console="Clip1" controls="ImageWindow"
height="570" width="800" autostart=false>
  </embed>
</object>
</p>

```

5 : 画面のインライン表示

```

<!-- コントロールキー -->
<p align="center">
<object id=video1
classid="clsid:CFCDAA03-8BE4-11cf-B84B-0020AFB8C9FA"
height="30" width="275">
  <param name="controls" value="ControlPanel">
  <param name="console" value="Clip1">
  <embed type="audio/x-pn-realaudio-plugin" console="Clip1" controls="ControlPanel"
height="30" width="275" autostart=false>
  </embed>
</object>
</p>

```

図 6 : コントロールバーのインライン表示

主に画面の高さ、横幅を指定し、para1.rpm というメタファイルを表示することを表します。Controls は表示するコントロールの種類、console はコンソール名で、図 5 も図 6 も同じものを用いることにより別のオブジェクトタグでも対応して動作することが可能である。autostart は false ならば Play ボタンを押すまで再生されず、true にすれば自動再生を可能にする。

3 同期マルチメディアを用いた研究キーワード紹介システム

3.1 システムの構成

本研究では、並列処理の基礎概念を、キーワードを説明しながらプレゼンテーション方式で紹介するシステムを構築した。並列処理の基礎を3つに分けて説明し、適宜研究キーワードが出てくるたびにリンクを貼って別途説明するコンテンツを作成した。

全てで10のコンテンツに分けてあり、以下の様になる(表3)。

表3：各コンテンツ

・ (1) 並列処理とは	(A)
・ スケーラビリティとは	(B)
・ (2) 並列処理とは	(C)
・ 負荷均衡とは	(D)
・ 同期とは	(E)
・ (3) 並列処理とは	(F)
・ 分割統治法とは	(G)
・ プロセッサファームとは	(H)
・ プロセスネットワークとは	(I)
・ 繰り返し変換とは	(J)

10個のコンテンツ全てをSMILで作成した。コンテンツによって音声、文字、アニメーションなどの組み合わせを考えて、いくつかの同期パターンで表現した(図7)。

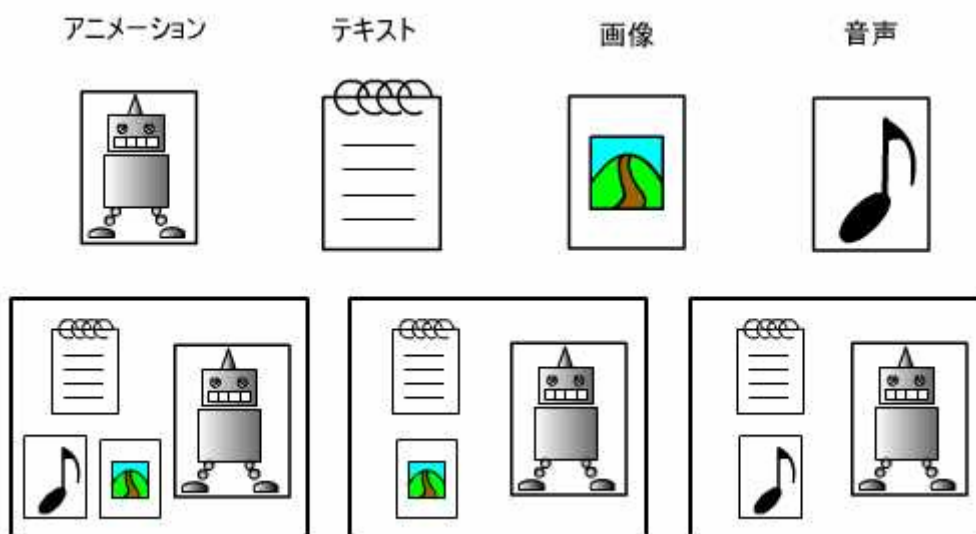


図7：同期表現のパターン

3.2 各キーワードの内容

表3における各キーワードを説明する。

(A) 並列処理とは(1)

ここでは並列処理のメリットを述べた。一台のプロセッサで問題を処理するよりも、複数台で処理できれば、より速く問題を解決できる。このことを、普段の生活に当てはめて例を挙げて説明した。もし課された宿題を解く時間が足りず、締め切りに間に合いそうもない時に、周りの友人に手伝ってもらうことで解決を目指す。このことは並列処理の基礎概念であり、そこでスケーラビリティ (scalability) が問題になってくる。

(B) スケーラビリティとは

プロセッサの台数を増やただけで単純に速度向上が得られるとは限らない。台数を増やした時にそれに見合う性能向上が得られる可能性をスケーラビリティという。ここでは、プロセッサ数とスピードアップを軸としたグラフを使うことで説明した。

(C) 並列処理とは(2)

並列処理において重要なテーマの高速性の追求について説明した。処理にかかる時間は各プロセッサの演算時間と通信時間である。複数のプロセッサのうち、1台だけに負荷をかけると結果として時間をとってしまう。これを解決するために負荷均衡という考え方があり、(D)で説明する。また、1つのプロセッサの処理結果が、他のプロセッサの処理に影響を及ぼしてしまい、結果として時間をとってしまう。これを解決するために同期という考え方があり、(E)で説明する。これらの説明を(A)のように課された宿題を例にとって説明した。

(D) 負荷均衡とは

一台のプロセッサに負荷がかかると、その処理が終わる時間が結果として全体にかかる時間となってしまう、高速性の追求に反する。そこで各プロセッサにかかる負荷をできるだけ均等にすることを負荷均衡という。ここでは、様々な大きさのボールを様々な大きさの処理内容に見立てて、各プロセッサに均等になるように配当するアニメーションで説明した。

(E) 同期とは

各プロセッサの時間的な統一を図ることを同期という。特にバリア同期とは、最も遅いタスクを処理するプロセッサを待ってから処理を始めることを言う。

(F) 並列処理とは(3)

並列マシンの2つのメモリモデルである共有メモリ型と分散メモリ型のマシンについて述べた。前者は、単一のメモリ空間が相互結合網を介して複数のプロセッサと結合されている。このためにプログラミングが容易だが、拡張性に乏しいというデメリットもある。後者は、メモリをそろえたプロセッサが相互結合網を介して接続されているので、プロセッサの追加が容易な反面、プロセッサ間のデータ転送を明示しなければならず、プログラミングが難しいというデメリットもある。

また、並列に問題を解くための並列アルゴリズムをリンクを貼って紹介した。

(G) 分割統治法とは

一般的に大規模で再帰的な問題を解くためのアルゴリズムで、ここでは積分計算を例にとって説明した。問題を下位の部分問題に分割し、更にそれ自身を分割していくことにより、再帰的に結合して解く。これを積分計算を台形則で解くことに当てはめて説明した。

(H) プロセッサファームとは

この方法では、分割したタスクを各々独立に計算でき、その結果を統合して求める。ここでは、画像処理におけるエッジ抽出を例にとって説明した。マスタにある画像を4つに分割し、各スレーブに渡す。各スレーブはそれぞれがエッジ抽出した結果の画像をマスタに返すというものにした。

(I) プロセスネットワークとは

この方法では、計算を複数のステージに分けて、流れ作業のように解く。ここでは、自動車工場のパイプライン処理を例にとって説明した。作業する人をプロセッサ、自動車の部品をタスクに見立てて表現した。

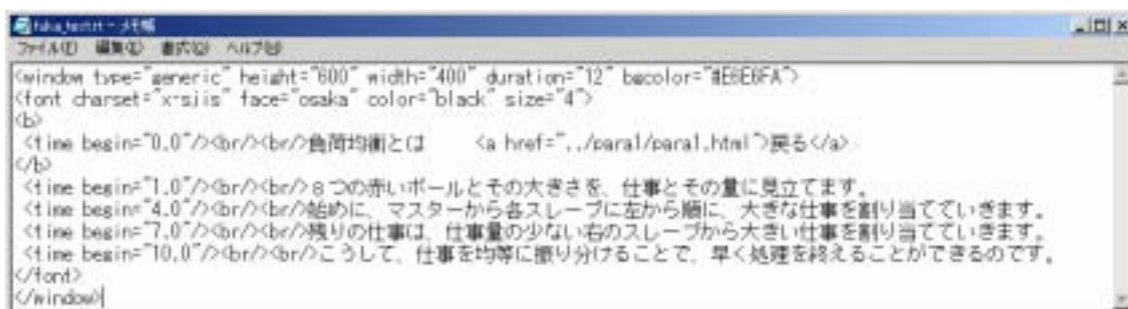
(J) 繰り返し変換とは

ある定まった手続きを繰り返し適用することによって、求める解に近づけていく。反復法で解ける問題に適している。

3.3 各メディアの同期方法

全てのコンテンツは SMIL を利用して同期をとった。主に RealText、CG や Flash アニメーション、静止画、そして音声によって成り立っている。

例えば、RealText では文字表示を時間指定できる（図 8）。

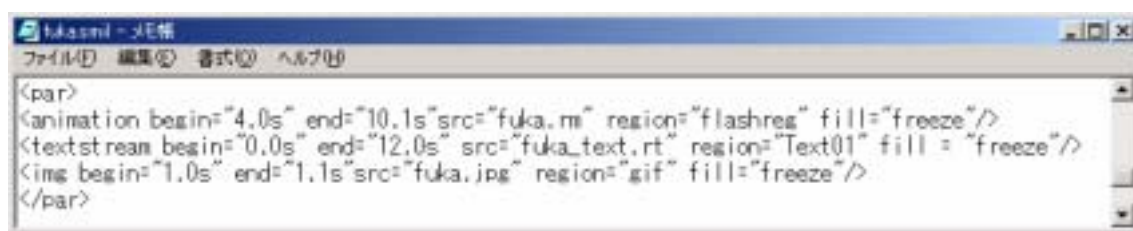


```
<window type="generic" height="600" width="400" duration="12" bgcolor="#E6E6FA">
<font charset="x-sjis" face="osaka" color="black" size="4">
<b>
<time begin="0.0"/><br/><br/>負荷均衡とは <a href=".,./paral/paral.html">異なる</a>
</b>
<time begin="1.0"/><br/><br/>8つの赤いボールとその大きさを、仕事とその量に見立てます。
<time begin="4.0"/><br/><br/>始めに、マスターから各スレーブに左から順に、大きな仕事を割り当てていきます。
<time begin="7.0"/><br/><br/>残りの仕事は、仕事量の少ない右のスレーブから大きい仕事を割り当てていきます。
<time begin="10.0"/><br/><br/>こうして、仕事を均等に振り分けることで、早く処理を終えることができます。
</font>
</window>
```

図 8 : RealText による時間指定

図 8 は負荷均衡の RealText 部分で、開始時間が 0.0 秒、次の文字表示が 1.0 秒、4.0 秒、7.0 秒、10.0 秒となり、一番上の行の duration="12"が全体の時間を表す。つまり、このファイルは 12 秒間のコンテンツであることになる。

更にこのファイルは同様に SMIL によって、他の CG や音声などとの時間指定ができる（図 9）。



```
<par>
<animation begin="4.0s" end="10.1s" src="fuka.m" region="flashreg" fill="freeze"/>
<textstream begin="0.0s" end="12.0s" src="fuka_text.rt" region="Text01" fill="freeze"/>

</par>
```

図 9 : SMIL による同期表現

図 9 の animation は CG が、4.0 秒に始まり、10.1 秒に終わることを表している。その他に textstream は RealText の表示を表し、img は静止画を表す。fill="freeze"は表示後もそのままの表示された状態にする意味を表す。

つまり、SMIL のコードの中で時間を制御し、RealText の文字情報も別途制御していることがわかる。

4 並列処理キーワードの作成

4.1 使用するマルチメディア

4.1.1 CGの作成

CGを作成するソフトとしてLightWave 3Dを使用した。LightWave3DによるCG作成は大きく分けて2つの作業に分けられる。1つはモデリングで、オブジェクトと呼ばれる物体を加工し配置していく。この作業はモデラーと呼ばれる画面で行われる(図10)。

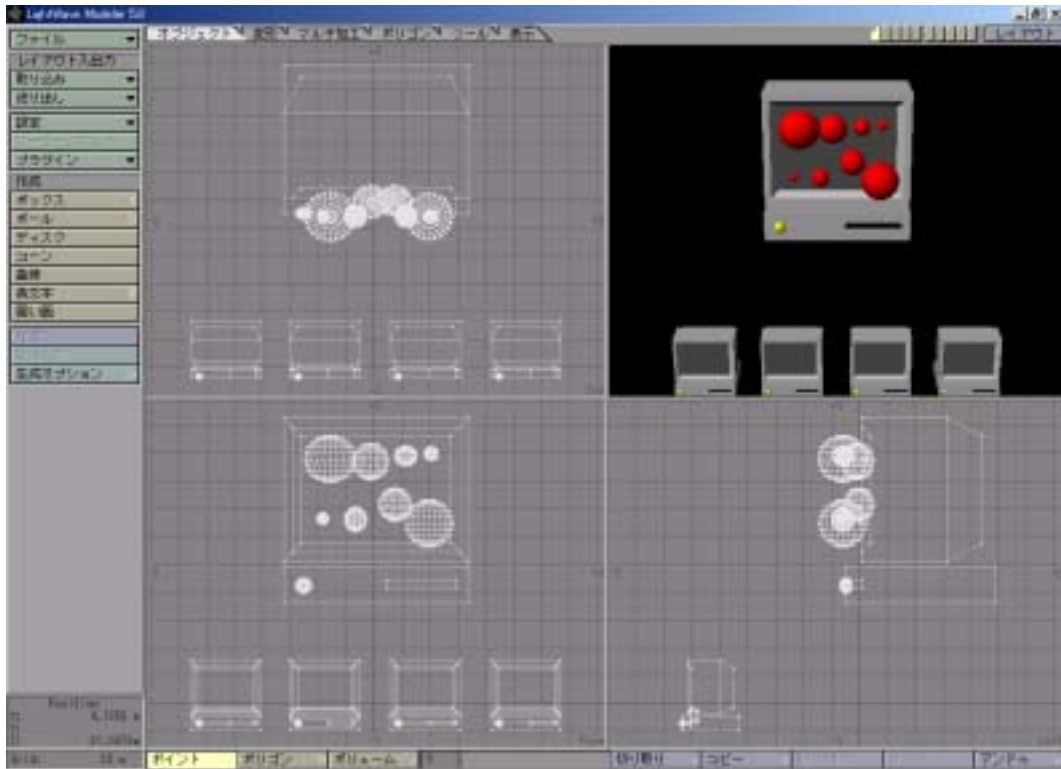


図 10 : モデラー画面

図10の画面右上「レイアウト」文字の左にある10個のボタンがある。LightWave3Dではこの「レイヤーボタン」が特徴で、これはTVアニメーションに用いられる透明なシートと思えばいい。このそれぞれのレイヤーにオブジェクトを配置し、それぞれを独立に動かすことができる。この考え方は4.1.2のFlashにも用いられている。

もう1つの画面はレイアウトという画面で、図11に表す。

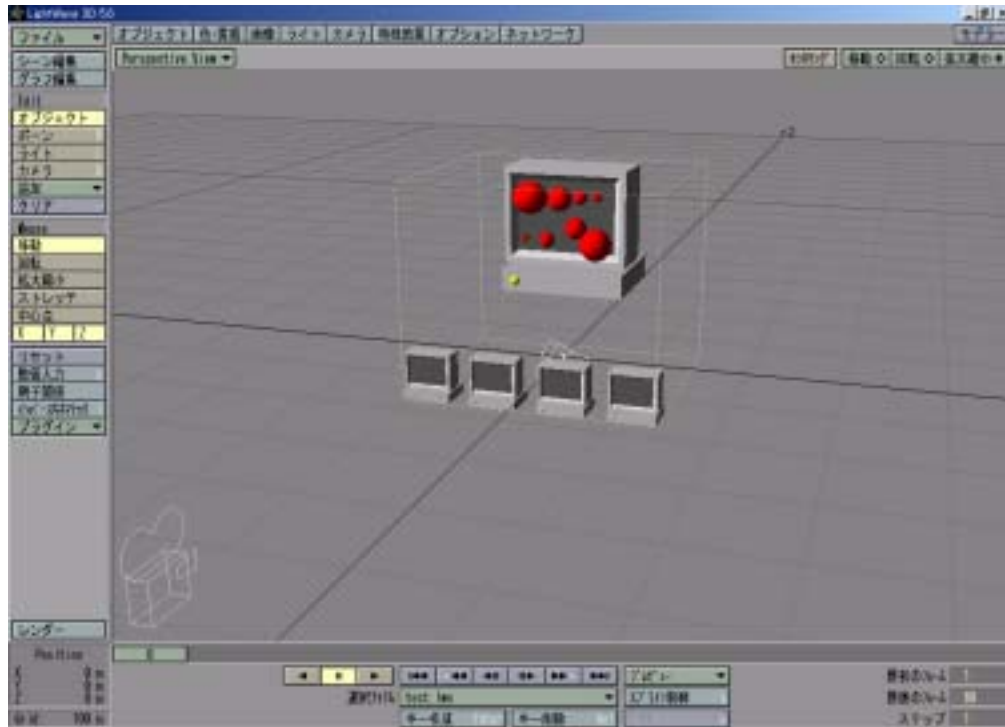


図 11：レイアウト画面

図 11 のレイアウト画面ではオブジェクトの操作はもちろん、カメラワークやライトの設定をする。画面下のスクロールバーを使って、オブジェクトを動かしながらキーフレームを作成していく。各フレームごとにオブジェクトを動かすことによって、動いているように見せる。左下にある「レンダ」ボタンによってレンダリングをして、CGとして表示する。

4.1.2 アニメーションの作成

アニメーションを作成するソフトに Flash 5 を使用した。Flash では主にグラフィック技能、アニメーション機能、パブリッシュ機能がある。グラフィック機能とはペイント機能のことであり、LightWave3D のようにレイヤーを使って描く。各レイヤーにアニメーション機能を使って動きをつけて、パブリッシュ、つまり.swf 形式や.jpg 形式等にかきだすといった手順を踏む。

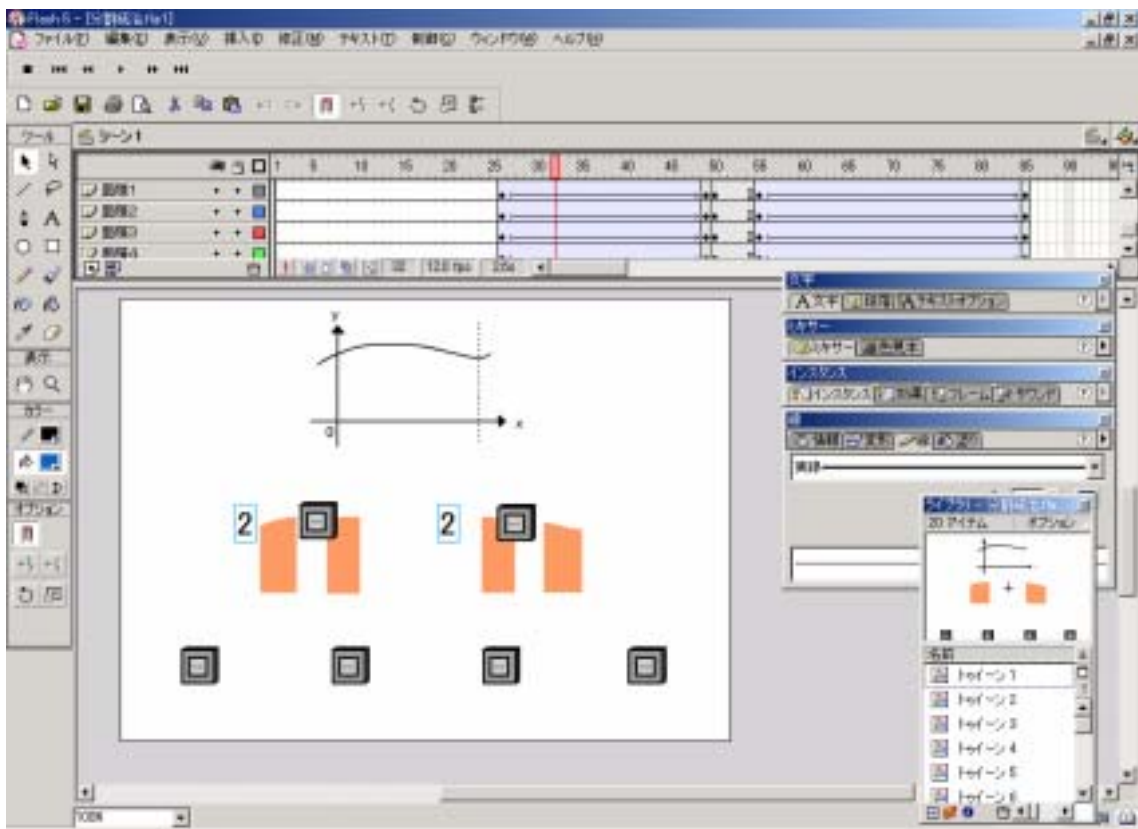


図 12 : Flash の画面

図 12 では分割統治法の作成画面を例にとって示す。画面左上あたりにある面積 1 , 2 ... と書かれたものがレイヤーで、ここでそれぞれのオブジェクトを作成する。作成したものに動きを付けてアニメーションにする。最後にパブリッシュして.swf 形式の RealFlash ファイルを生成する。

4.1.3 テキストの作成

キーワード紹介のためのテキストには、RealSystem の RealText を使用した。RealText の場合、メモ帳を使って作成した。時間のタイミングを図ったり、2D や 3D のものを作成できる。ここでは負荷均衡のテキスト部分を例にとって説明する。

```
<time begin="1.0"/><br/><br/> 8つの赤いボールとその大きさを、仕事とその量に見立てます。  
<time begin="4.0"/><br/><br/> 始めに、マスターから各スレーブに左から順に、大きな仕事を割り当てていきます。  
<time begin="7.0"/><br/><br/> 残りの仕事は、仕事量の少ない右のスレーブから大きい仕事を割り当てていきます。  
<time begin="10.0"/><br/><br/> こうして、仕事を均等に振り分けることで、早く処理を終えることができます。
```

図 13：負荷均衡の RealText コード

図 13 は各文章の表示開始秒数を表している。1 秒、4 秒、7 秒、10 秒ごとに文章が表示される。最終的には図 14 のような画面が生成される。リンクを貼ることもでき、図 14 の画面中央部上にある「戻る」にはそれを適用している。拡張子は.rt として保存する。

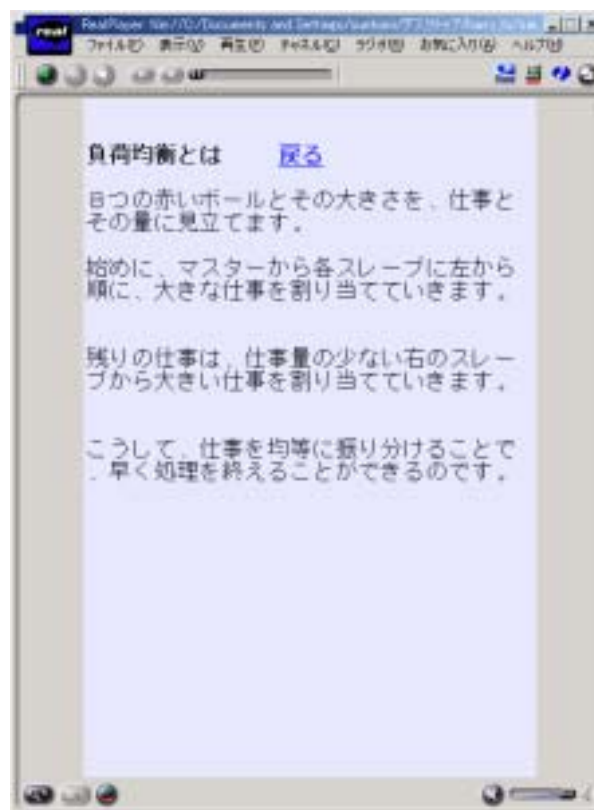


図 14：RealText の表示

4.1.4 音声の作成

音声を付加するために RealSystem の RealAudio 形式のサウンドを使用した。RealAudio の作成には、Windows に標準で付属しているサウンドレコーダーを使用した(図 15)。



図 15 : サウンドレコーダー

これを使用して音声を録音して、いったん *.wav 形式に保存する。これをストリーミングできるフォーマットに変換しなければならない。つまりエンコードしなければならないので RealProducer を使って変換した(図 16)。

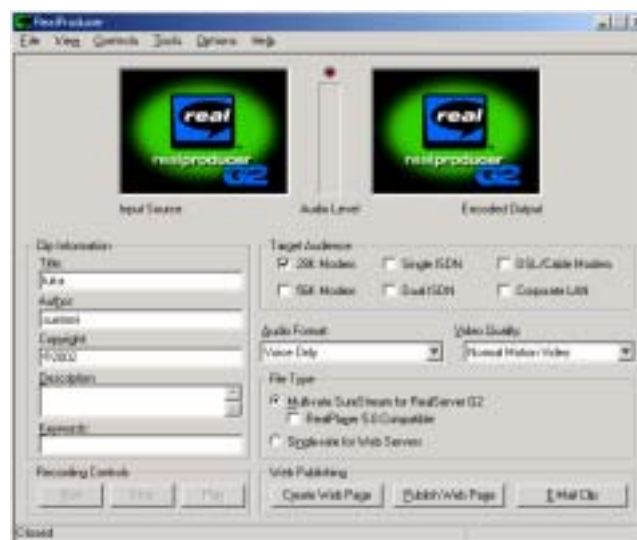


図 16 : RealProducer

これを使ってエンコードして *.rm 形式のファイルに変換する。こうすることでストリーミングで使用できるファイル形式となる。*.wav 等の音声ファイル以外にも、*.avi 形式のムービーファイルも同じ要領で変換できる。4.1.1 で作成した 3 DCG も RealProducer を使って変換したものを利用した。

4.1.5 静止画の作成

静止画は主に jpg 画像を使用した、RealSystem の RealPix も使用した（表 4）。

表 4 : RealPix

```
<image handle="1" name="para3_1.jpg"/>
<image handle="2" name="para3_2.jpg"/>
<image handle="3" name="para3_3.jpg"/>
<image handle="4" name="para3_4.jpg"/>
<image handle="5" name="para3_5.jpg"/>
<image handle="6" name="para3_6.jpg"/>
<image handle="7" name="para3_7.jpg"/>
<image handle="8" name="para3_8.jpg"/>

<fill start="0" color="white"/>
<fadein start="3" duration="2" target="1"/>
<crossfade start="5" duration="1" target="2"/>
<crossfade start="7" duration="4" target="3"/>
<fadeout start="12" duration="1" color="white"/>
<fadein start="13" duration="1" target="4"/>
<crossfade start="14" duration="2" target="5"/>
<crossfade start="17" duration="1" target="6"/>
<crossfade start="19" duration="1" target="7"/>
<crossfade start="21" duration="1" target="8"/>
<fadeout start="23" duration="1" color="white"/>
```

表 4 では、並列処理では (3) で用いた RealPix の静止画同期部分のコードを記述したものである。上 8 行では、8 つの jpg 画像に handle という番号を付けて、下の 11 行の中で target として同じ番号を使う。これにより画像を番号で制御する。Fadein,crossfade,fadeout はそれぞれ画像をフェードイン、画像を重ねてフェード、フェードアウトすることを示す。開始時間と画像表示時間は、start,duration で表す。

4.2 各キーワードの内容と同期方法

各キーワードの紹介のために作成したコンテンツを、その内容と同期方法を交えて説明する。

(A) 並列処理とは(1)

ここでは、並列処理がいかに便利であるかを日常の例をとって基礎概念を説明した。図 17 は、開始 5 秒後の映像で左画面の文章と同期して右上のアニメーションが動いている様子を表す。このアニメーションでは、ロボットが 1 人で膨大な宿題を抱えて困っている。次に図 18 に示す文章とアニメーションが同期して開始から 10 秒後に現れる。これは、他の 3 台のロボットに手伝ってもらうことにより宿題の負担を軽減する様子を表している。この後、12 秒後、14 秒後に文章が続き、その間は右下のアニメーションは静止している。最後に図 19 のように、16 秒後に文章とアニメーションが出現し、ロボットをコンピュータに置き換えて考える事により、コンピュータの世界でも一台のプロセッサで解くよりも複数のプロセッサで解く方が良い場合がある事を表した。尚、最後に出現するスケーラビリティという単語はわかりづらいので、リンクを貼って(B)で別途説明コンテンツを作成した。

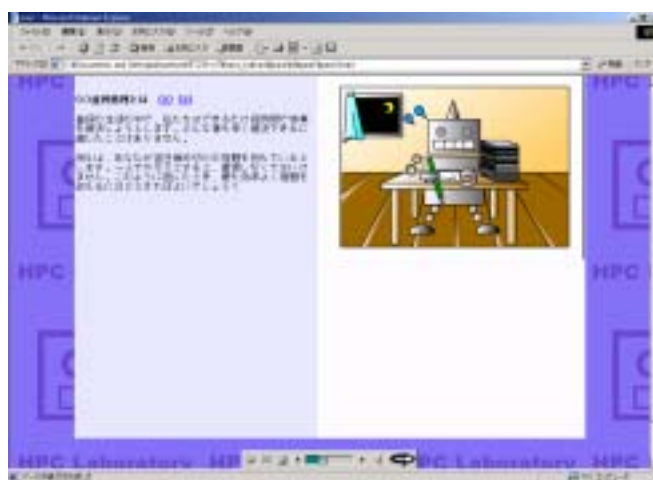


図 17 : 並列処理とは(1)の5秒後



図 18：並列処理とは（1）の 10 秒後



図 19：並列処理とは（1）の 16 秒後

(B) スケーラビリティとは

開始と同時に左に文章、右上にアニメーション、右下に静止画が、そして音声による説明が同期する。右上のアニメーションでは矢印がグラフに沿って移動していく。図 20 では、赤色の矢印が横軸の 4 台まで移動していることを示し、プロセッサ数 4 台までのスピードアップがあることを表す。赤色の矢印は 4 台まで移動すると図 21 のように緑色の矢印に変化して移動を続ける。

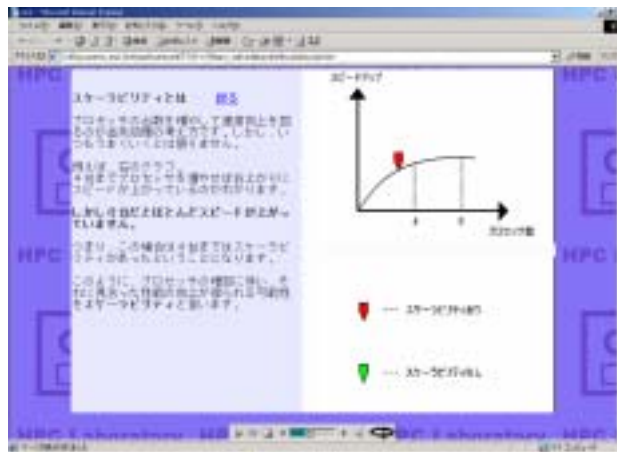


図 20 : スケーラビリティのある部分

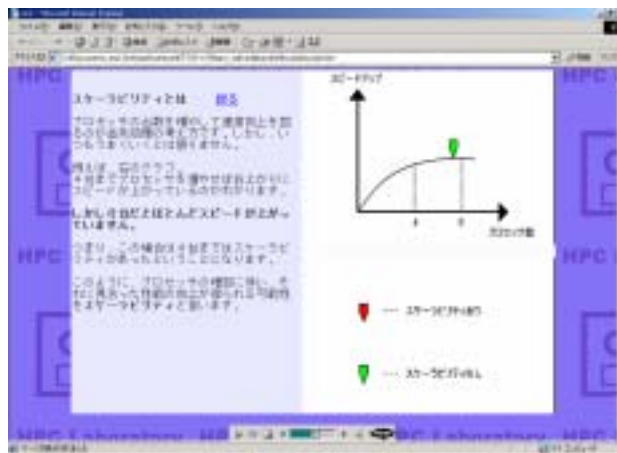


図 21 : スケーラビリティのない部分

(C) 並列処理とは(2)

ここでは、並列処理の基本目的である「高速性の追求」を説明し、そのために必要な負荷均衡と同期の考え方を説明した。

図 22 では、文章、右上のアニメーション、そして音声による説明が同期して始まる。アニメーションは 9 秒後動作する。この時同時に音声もスタートする。アニメーションは、一番左のロボットの宿題が多すぎて他のロボットよりも処理に時間がかかってしまい、結局全体の時間が多くなってしまう。これで、均等に処理を分けることの重要性と表す。こういった概念を負荷均衡といい、このキーワードはリンクを貼って別途説明した。

図 23 では、23 秒後に右下に音声説明と同時にアニメーションが出現する状態を表す。これは、ロボット A が宿題を解くにはロボット B の結果を使わなければならないので、ロボット A がその結果を待っている状態。つまり、同期を計らなければならない状態を表す。これも音声で説明する。尚、同期にもリンクを貼っている。



図 22 : 並列処理とは(2)の9秒後



図 23 : 並列処理とは(2)の23秒後

(D) 負荷均衡とは

図 24 では、開始と同時に文章、静止画が同時に始まる初期状態を表す。文章はこの後、1、4、7、10 秒の順に表示され、4 秒後にアニメーションと音声説明が同期して開始する。

アニメーション部分を、図 25 から 28 で説明する。図 25 では、マスタから各スレーブに、様々な量の仕事が渡っている。図 26 で各スレーブのモニターが赤く染まっているのは、マスタから与えられた仕事量を表している。赤く染まっているスレーブほど多くの仕事を受け持っている。図 26 では左のスレーブほど多くの仕事を受け持ち、右に行くほど軽い仕事量を受け持っている。負荷均衡を計るため、マスタに残った残りの仕事のうち一番大きい仕事を、現時点で受け持っている仕事の一番少ない右のスレーブに渡す様子を表している。そうすると、今度は図 27 のように一番右のスレーブの仕事量が多くなる。その後はマスタに残った仕事量の大きい順に、残りのスレーブに右から順に当てはめることにより、各スレーブの負荷を等しくすると図 28 のようになり、すべてのスレーブに同量の仕事が振り分けられた形になり、負荷均衡を計ったことになる。



図 24：負荷均衡の初期画面

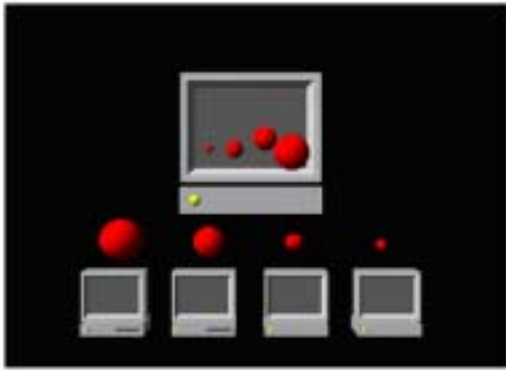


図 25 : 始めの仕事の振り分け

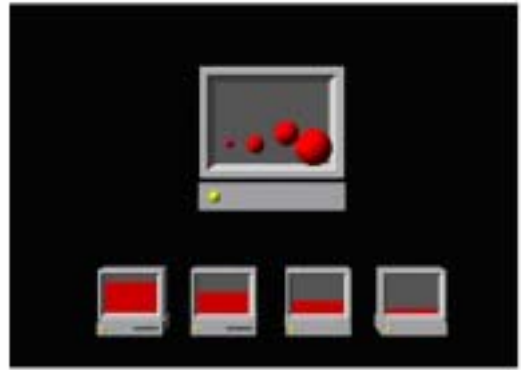


図 26 : 始めの振り分け後

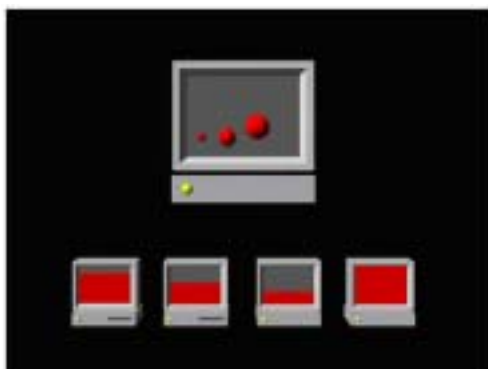


図 27 : 右端のスレーブの仕事量が最大

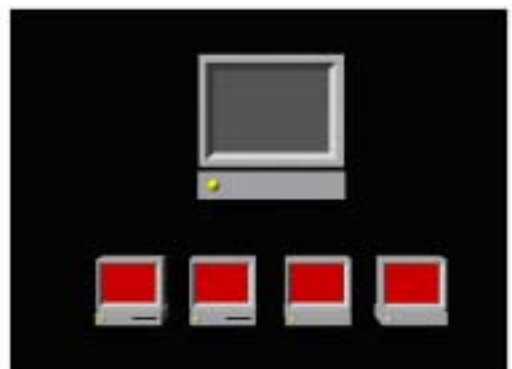


図 28 : 最終状態

(E) 同期とは

図 29 では、全てのメディアが同時に開始する。アニメーションはバリア同期を示しており、一番処理の遅いロボット A の処理を他のロボットが終わるまで待っている。



図 29：同期とは

(F) 並列処理とは (3)

図 30 では共有メモリ型、分散メモリ型の並列マシンを説明し、4つの並列アルゴリズムを紹介してリンクを貼った。左に文章が0、1、4、...、22秒と小刻みに現れ、それに同期して右上に静止画が現れる。RealPixを使っているので、フェードイン、フェードアウトを繰り返しながら現れる。



図 30：並列処理とは (3)

(G) 分割統治法とは

図 31 では、開始と同時に文章、アニメーション、静止画、それに同期して音声で説明が流れる。分割統治法による再帰的なアルゴリズムを、面積計算を例にとって説明する。

図 32 では、面積 4 の計算をする初期状態を表す。図 33 では面積を下位のプロセッサに分割している。図 34 で更に下位のプロセッサに分割し、そこで分割された面積をそれぞれのプロセッサが計算している。図 35 は求めた個々の解が上位に移動して行き、再帰的に結合して解を求める状態を表す。



図 31 : 分割統治法

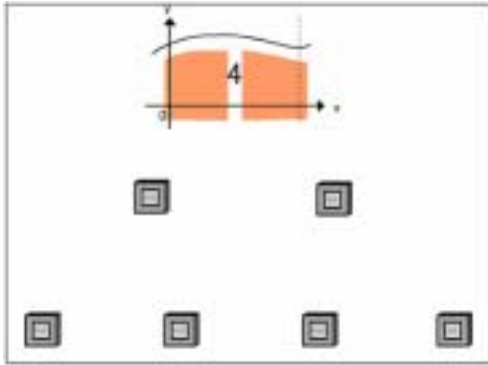


図 32 : 分割統治法の初期状態

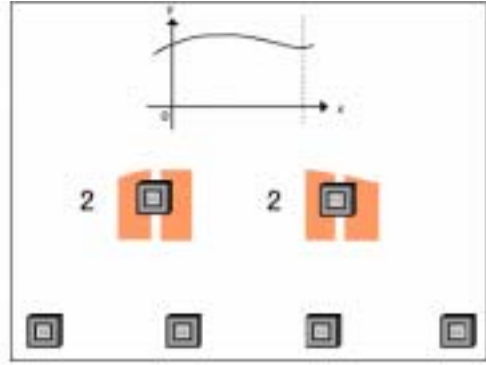


図 33 : 1つ下位に分割

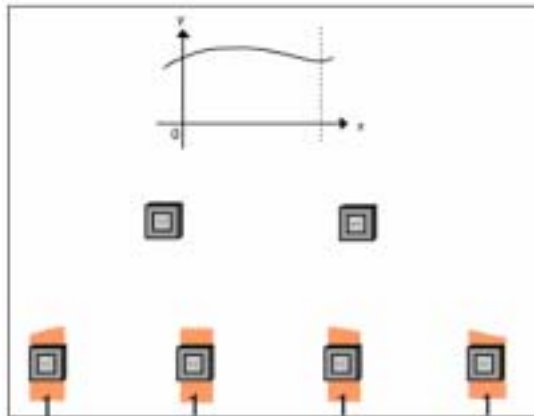


図 34 : 最下位に分割

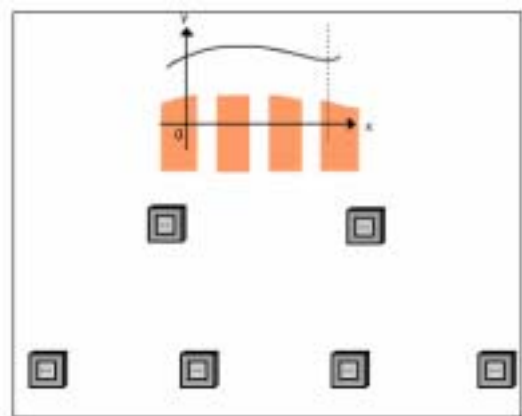


図 35 : 再帰的な結合

(H) プロセッサファームとは

図 36 では、文章、アニメーション、静止画、そして音声同期している状態を表す。エッジ抽出を例にとって説明した。

図 37 で、赤く塗りつぶした円の初期画像を 4 分割して各スレーブに渡す。図 38 では各スレーブがエッジ抽出している状態で、モニターを黄色く点滅させることで処理中であることを表す。図 39 で、各スレーブから処理後のエッジ抽出結果の黒ぶちの円がマスタに返っていく状態を示し、図 40 のように最終状態になる。



図 36 : プロセッサファーム



図 37 : 初期画像を分割



図 38 : 処理中



図 39 : 結果画像の回収



図 40 : エッジ抽出最終状態

(I) プロセスネットワークとは

図 41 のように、文章、アニメーション、静止画、そして音声で説明。アニメーションはタスクを自動車の部品に、ロボットをプロセッサに見立ててパイプライン処理を表現した。左のロボットが車体を（図 42）、真中のロボットがタイヤを（図 43）、右端のロボットが窓を（図 44）製作する流れ作業でプロセスネットワークを表現した。



図 41：プロセスネットワーク

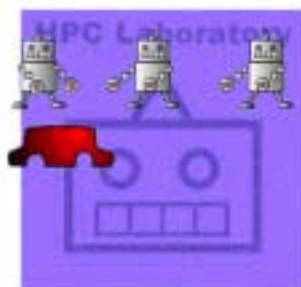


図 42：左が作業

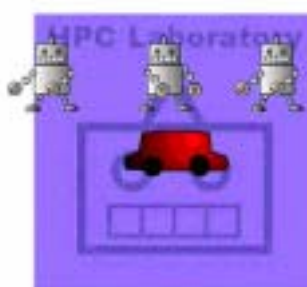


図 43：中央が作業

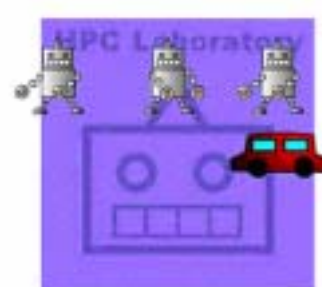


図 44：右が作業

(J) 繰り返し変換とは

図 45 は、文章、アニメーション、静止画、そして音声を同期させた最終状態を表す。アニメーションでは様々な初期値を様々な状態の未完成なロボットに例えて各オブジェクトを繰り返し変換していくように表した。

初期状態では、左のロボットは腕だけがなく、真中のロボットは顔と胴体しかなく、そして右のロボットは顔しかない。

始めの処理で上の様々な 3 体のロボットが画面下に降りて行って (図 46) 各スレーブで処理され、左のロボットに腕が付き完成する (図 47)。処理中はモニター画面を黄色く点滅させて表現した。真中のロボットは足が付き、右のロボットは胴体が付くが、完成品に満たない残りの 2 体のロボットは画面上のほうに戻って行く。同じように、残りの 2 体の未完成のロボットが画面下に降りて行き (図 48) 今度は真中のロボットに腕が付き完成する (図 49)。そして最後に右のロボットがもう一度繰り返し処理されて、全てのロボットが完成する。



図 45 : 繰り返し変換とは



图 46 : 3 体下降中

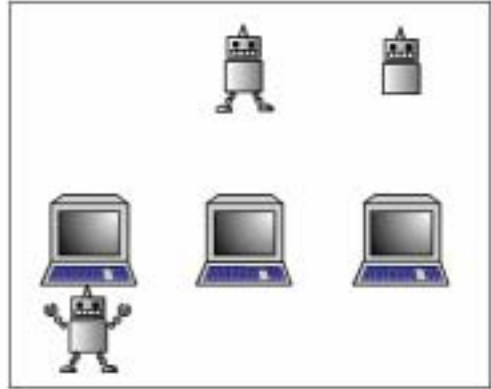


图 47 : 1 体完成

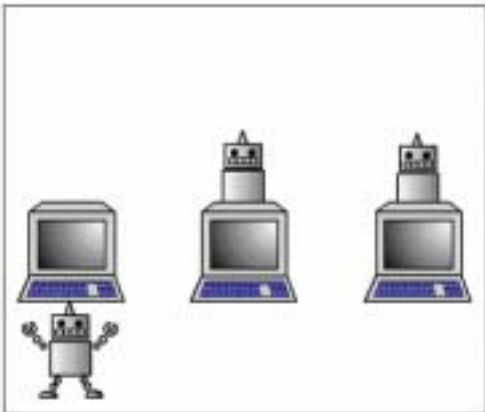


图 48 : 残 2 体下降中

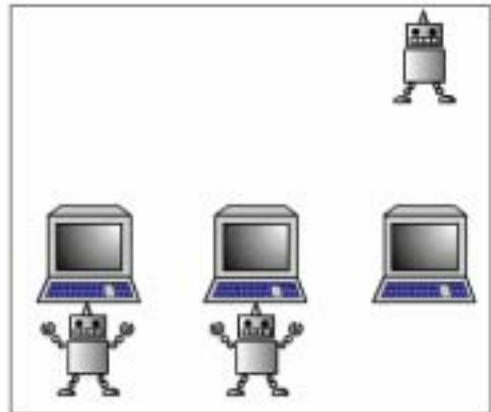


图 49 : 2 体目完成

5 システムの考察

作成した並列処理の研究キーワードを紹介するシステムの考察を行う。

4章で作成した10個のコンテンツには、メディアの同期のさせ方を数パターンがある。Flash アニメーション+テキスト+静止画のパターン、それに音声を付加したパターン、静止画をフェードイン、フェードアウトを繰り返しながらあたかも動画のように見せたものを使ったり、CGの動きに同期してテキストを表示させたものなど様々なものを作成した。

その結果、まず、SMILでは複数のアニメーションを同時に使うことで、ストリーミング時のバッファが追いつかず動作が悪くなるがあった。これが生じた事で考えられることは2つある。1つ目は、コンテンツを制作する際に必要最小限のメディアを使用すべきであることだ。ストリーミング配信の際に劣化する情報を補うことを1つのテーマとしたのに対し、過度なアニメーションを多用することを考えてしまったためこのような事が起こった。2つ目は視聴者側の目を無視していたことが挙げられる。複雑なアニメーションを使ったり、多くのメディアを同期することが視聴者にとっての快適な閲覧と理解につながらないとわかった。

また、SMILではスムーズに同期させる上での各メディアの相性が見られた。例えば、アニメーションにテキストを一行ずつ同期させてアニメーションを説明するよりも、テキストを一度に表示させた上で音声を同期させる方が、よりスムーズになった。また、簡単なアニメーションであれば、静止画を数秒ごとに制御させる RealPix を使用して動きのあるように見せることで、より軽いデータで済んだ。

6 おわりに

本研究では、研究キーワードを SMIL という同期マルチメディア言語を用いて、如何にわかりやすく、スムーズに視聴者に伝えるかに重きを置いた。ストリーミングによって配信することにより、より軽いデータでどこまで多くの情報を伝えることができるか、多くのメディアがある中、どのメディアを使い、どういった配置が最も良いか等がテーマだった。

実際作成したシステムは、アニメーション、CG、音声、テキスト、そして静止画を SMIL という同期マルチメディア言語を用いた。並列処理の基礎概念を紹介するこのシステムでは、並列処理という分野に少しでも興味を持ってもらいたく作成した。そのことは同時に、知識のない視聴者に対して情報を提供するということであり、各メディアをどのように使うかが問題であった。システム試作において感じたことは、上に挙げた各メディアを多用したり、アニメーションや、テキストを小刻みに時間制御したりすることで、それが果たして視聴者に伝わりやすいものになっているか、といった懸念だった。Flash によるアニメーションは、それ自体わかりやすいが複数使用すると動作の悪化につながり、テキストベースの RealText は、時間制御できたりリンクを貼ったりできるが、他のメディアと同期の相性が悪かったりして、視聴者にとってわかりやすいコンテンツと、より多くのメディア同期が排反の関係に近いことが感じられた。

今後の課題としては、SMIL での各メディアの相性や、コンテンツの構成、配置を改善すること、そして、今回は初心者対象のコンテンツになったが、様々なユーザに対して個別のものを配信したりすることが考えられる。快適なストリーミングに堪える新しいメディアの使用も検討できる。

謝辞

本研究の機会を与えて下さり、数々の助言を頂きました山崎勝弘教授に心より感謝致します。また、本研究にあたり、いろいろな面で貴重なご意見、ご指導を頂きました本研究室の皆様に、心より感謝申し上げます。

参考文献

- [1] 小畑健二：シンクロナイズド・マルチメディアを用いた研究室紹介システム、立命館大学大学院理工学研究科修士論文、2001.
- [2] 川口智也：マルチメディアを同期させた研究キーワードの作成、立命館大学工学部情報学科卒業論文、2001.
- [3] エーアムック 274：ストリーミング・Web で動画を見せよう！、2001.
- [4] 湯浅太一・安村道晃・中田登志之 編：はじめての並列プログラミング、共立出版株式会社、1999.
- [5] 山崎勝弘：コンピュータは進化する、1999.
- [6] 大澤光 編著：インターネットストリーミング、共立出版株式会社、2000.
- [7] ビデオ 12月号別冊：ブロードバンドスタイル、写真工業出版社、2001.
- [8] 境祐司：速習 Web デザイン FLASH5、技術評論社 2001.
- [9] モーリー・ロバートソン他：インターネットストリーミングブック、翔泳社、2000.
- [10] NewTek：LightWave 3 D ver5.5 ユーザーガイド、1997.
- [11] 由水桂：LightWave3D スーパーテクニック、ソフトバンク、1999.
- [12] リアルネットワークス社：<http://www.jp.real.com/>.
- [13] Real Video de SMIL：<http://homepage1.nifty.com/junkosan/real/>.
- [14] SMIL でいこう：<http://www.smi.co.jp/web-cm/smil/index.html>.

付録 SMIL の同期部分 HTML ファイル

(1) 並列処理とは (2) の HTML ファイル (para2.html)

```
<HTML>
<HEAD>
<TITLE>smil</TITLE>
</HEAD>

<STYLE TYPE="text/css">
  BODY {background-image:url(../../image/back_rob.jpg);}
</STYLE>

<p align="center">
<object id=video1

  classid="clsid:CFCDA03-8BE4-11cf-B84B-0020AFBCCFA"

  height="570" width="800">
    <param name="controls" value="ImageWindow">
    <param name="console" value="Clip1">
    <param name="autostart" value="false">
    <param name="src" value="para2.rpm">
    <embed src="para2.rpm" type="audio/x-pn-realaudio-plugin"
console="Clip1" controls="ImageWindow"

  height="570" width="800" autostart=false>
    </embed>
    </object>
</p>

<!-- コントロールキー -->
<p align="center">
<object id=video1
```

```
classid="clsid:CFCDA03-8BE4-11cf-B84B-0020AFBCCFA"
```

```
height="30" width="275">
```

```
  <param name="controls" value="ControlPanel">
```

```
  <param name="console" value="Clip1">
```

```
  <embed type="audio/x-pn-realaudio-plugin" console="Clip1"
```

```
controls="ControlPanel"
```

```
height="30" width="275" autostart=false>
```

```
  </embed>
```

```
  </object>
```

```
</p>
```

```
<HR>
```

```
<BODY>
```

```
ブラウザにより動作しない場合は
```

```
<A HREF="para2.ram">ここ</A>
```

```
をクリック
```

```
</BODY>
```

```
</HTML>
```

(2) 並列処理とは (2) の同期部分 (para2.smil)

```
<smil>
```

```
  <head>
```

```
    <!-- Presentation attributes. -->
```

```
    <meta name="title" content="para2" />
```

```
    <meta name="author" content="suetomi" />
```

```
    <meta name="copyright" content="(c) 2002 HPC Laboratory" />
```

```
  <layout>
```

```
    <root-layout width="800" height="570" background-color="white" />
```

```
    <region id="flashreg" left="400" top="0" width="399" height="289"
```

```
z-index="1" fit="meet"/>
```

```
    <region id="flashreg2" left="400" top="290" width="399"
```

```

height="289" z-index="1" fit="meet"/>
  <region id="Text01" left="0" top="0" width="399" height="570"
background-color="white" fit="meet"/>
  </layout>
</head>

  <body>
  <par>
    <animation begin="9.0s" end="19.0s"src="para2_fuka.swf"
region="flashreg" fill="freeze"/>
    <animation begin="23.0s" end="32.0s"src="para2_douki.swf"
region="flashreg2" fill="freeze"/>
    <textstream begin="0.0s" end="10.0s" src="para2_text.rt"
region="Text01" fill = "freeze"/>
    <audio src="para2.rm"/>
  </par>
</body>
</smil>

```