可視化・疑似体験教材作成を支援するソフトウェア

理学研究科 凝縮系科学専攻
修士 1 年 桜井雅史

概 要

物理・数学等の講義の中で、コンピューターを使って手軽に物理現象をアニメーションさせたり、インタラクティブに実演する教材を作成し、実際に授業に応用した。また、そのような教材を手軽に作成できるソフトウエア・システムを作成した。

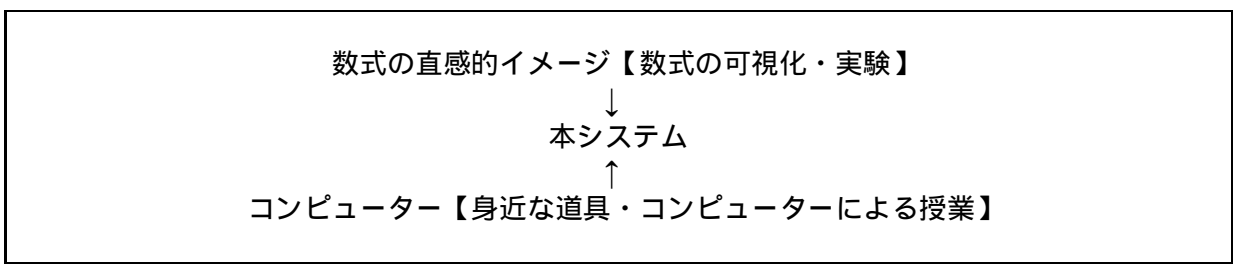
1 目的

物理や数学などにおいて、数式の持つ意味を直感的にイメージすることは、重要で不可欠な能力である。特に物理では、「あるモデルの仕組みを表す方程式を立てる」、「数式からモデルの様子をイメージする」などの場面は日常茶飯事であり、物理を学ぶものにとって数式の扱い以上に必要とされる能力である。

従来の板書及び教科書による授業スタイルでは、受講者にイメージをつかませるのに苦労が必要である。例えば、時間変化するモデルの様子や、係数を変化させた時の応答をインタラクティブに体験するということは、静的な黒板や 2 次元の教科書では実現が難しく、敢えて実行したとしても、ちょっとした説明のために実験や動画作成の準備に多大な労力と時間が必要である。

一方、近年はコンピューターの能力の発達によって手軽にコンピューターを利用できるようになった。さらに教育方面でも、コンピューターを使うことによって数式を即座に計算したり、動画や 3 次元的な図形を効果的に表示することができるため、単なるコンピューター利用の授業にとどまらず、普通の授業等に教材として活用される場面も多くなってきた。しかし、この場合も教育者、利用者共に十分な知識や準備を必要としたり、インストールやソフトウエア購入の手間があるため、手軽に利用するということは難しい。

そこで、コンピューターを使って手軽に物理現象をアニメーションさせたり、インタラクティブに実演するような教材を手軽に作成できるシステムあれば大変便利にちがいない（あわよくば、いまの理系離れを何とかできるかもしれない。）と思い、作成を試みた。



1.1 教育とマルチメディア技術

ここで、簡単に現在の教育とマルチメディアの活用について簡単に触れておく。

マルチメディア教育について

近年、呪文のようにマルチメディアという言葉が氾濫してきているが、教育現場における「マルチメディア」とは、従来の板書、教科書以外の教育媒体のことをさすことが多い。例えば、ビデオやオーディオ等、また最近ではコンピューターやインターネットなども含まれるようである。

しかし、筆者のこれまでの経験や、周りの意見を総合すると、これだけ情報化社会に発展した現在でもなお、ごく一部を除いて学校での教育方法は全く変化が見られず、未だ有効なマルチメディア教育というものが広く行われていないように思われる。

子どもを巡る環境の変化

2,30 年前と比べて、子どもの環境は大きく変化している。特に娯楽方面では、外で遊んだり、読書をするということは稀で、子ども達が主に接しているものはテレビ、漫画、雑誌などの映像中心の書物であったり、ゲームやコンピューターなどの双方向メディアである。

これが良いかどうかの議論は非常に興味深い問題ではあるが、おそらく答えの出すことが難しい領域であり、ここではそのことは問題にしない。重要な点は、子ども達の置かれている環境はまさにマルチメディアであるということである。

教育の変化

一般に、子どもたちは勉強が嫌いであるが、遊ぶことは好きである。そのため、教育も子ども達の遊びを取り入れたものが効果的であるが、現在広く行われている板書、教科書による授業では、活字から離れてしまった子ども達の興味を引くことは難しい。そこで、現状に即した教育方法として、音や映像、双方向性を取り入れたマルチメディアシステムを採用して、全く新しいやり方で教育を行う必要があるのではないかと考える。

マルチメディア教育

新しい教育方法であるマルチメディア教育は、単に黒板、教科書を電子的に置き換えた、いわゆる「電子テキスト」的なもので終わりでははない。一方向的な授業ではなく、双方向的なインタラクティブなものであり、机に座って一人静かに勉強する姿よりは、多人数で相談したり、一緒にゲームを楽しむ姿をイメージするべきである[1]。

このような全く新しいやり方は、従来の方法に慣れ親しんだ教師の方々にとって未知のものであり、時にははばかばかしく思えるかもしれない。「大人」はこのような方法で学んでこなかったので認める事すら時間がかかることが予想される。しかし、現に学力低下が物語っているように、従来の方法は改善を迫られている。

[1] 無論、全ての授業をこの方法でやるべきだというわけではない。また、最近の風潮としてある甘やかしや幼稚化も良くない。

どうするべきか？

新しい試みは既にいろいろな方面で行われているが、現在の教科書に代わる絶対的なものは確立されていないようである。以下に主な取り組みの例を記してみた。

- 電子テキスト的なもの

  教科書を電子的にして、映像、音、双方向性を追加したもの。既存の方法の拡張であり、簡単に導入でき、失敗や導入コストが少ない。

- 授業の情報掲示板

  授業についての情報を Web などで補助的に発信したり、また電子掲示板やアンケートなどを使って一方向になりがちな授業に双方向性を持たせたもの。これも従来の授業の拡張といえる。

- 自己完結的なゲーム形式のもの

  コンピューターが先生となり、問題を出題して生徒が答えを入力するといったものが多い。他にも、ゲームの中でクイズや問題、あるいは疑似体験等を通して意識せずに学習しようというものもある。

- 通信手段として利用

  コンピューターネットワークを利用し、離れた地点で講義を受けたり、また、離れた地点の他の学校と共同で何かのプロジェクトを行うなど。

- データベースとして利用

  インターネットや百科事典、天気、経済、技術などのデータベースを利用したもので、必要な情報を素早く引き出し、コンピューターを駆使してデータを加工して何らかの結果を得る、といったもの。

- 表現の道具として利用

  絵や、音楽、また動画やプレゼンテーション資料の作成をコンピューターを使って行い、Web などを利用して発表も行うもの。コンピューターの利用により、従来では出来なかった表現の幅を得ることができる。

ここに分類した以外にも数多くの試みがあり、それぞれ成果をあげているようである[2]。

また、それぞれの試みは教師の方々の技術力に大きく拠っているところが大きく、全ての生徒が同等の教育を受ける状況には程遠い。これがいわゆるデジタルデバイド（デジタル格差）であり、これからマルチメディア教育を推し進めていくに当たり、多くの教師の方がコンピューターに精通することが重要になってくる。

[2] しかし、効果をあげているのは新しい方法もさることながら、教師の方々の教育に対する前向きな姿勢も大きいのではないかと思う。

本研究との関連

本研究では、ここに挙げたもの「電子テキスト的なもの」に当てはまる。従来の方法を継承しつつ、教科書のみの講義にあった問題を解決し、またコンピューターに関する深い知識を必要とせずに誰でも教育現場に応用できるようなシステムの作成が今回の目標である。

2　現存のシステムの考察

ここでは既に存在する取り組みについて簡単に触れる。

2.1　従来の板書・教科書の問題点

冒頭の目的にも述べたとおり、動いたり、立体的なものが表現しにくい。また、情報がほぼ一方向的であるため、実際に実感したり、興味を引くことが難しい。

欠点

- 時間変化する (動く) ものが表現しにくい
- 複雑なものや立体的な表現が難しい
- 式の係数による系の変化を実感させにくい (一方向性)

なお、教卓の上で実験をして見せたり、ビデオ映像を作成して見せることによりこれらの欠点を補うような方法もあるが、いずれも準備に時間がかかったり、手軽に実現できる実験の幅が狭いなどの問題は依然残る。

2.2　数式処理システム

Mathematica に代表されるようなコンピュータによって数式・数値を処理するようなソフトウエアを使い、方程式や数式を可視化して講義を支援するものである。

最近の数式処理システムは、高校生程度の数学を一瞬で解き、また第一線の研究者が日常的に数式を扱う道具として使用できるほどの性能を持っている。無論、本システムのように、数式をグラフやアニメーションで可視化することもできる。しかしそのような高性能さゆえ、市販されているソフトウエアは高価で、しかもインストールや操作方法も大変であり、実際に講義で使用したり、全ての受講者に操作させることは難しい。

利点

- 高性能で万能

欠点

- 高価
- インストール、操作が大変

2.3 他の Java Applet 教材

Java はコンピューターのプログラム言語の一つである。ホームページ等の Web 上でソフトウエアを実行できることから、動画や、インタラクティブなページ作成に利用されているほか、OS を問わずに動作し、また生産性も高いことから企業のサーバーシステムや、携帯電話等の組込み用途にも広く使用され始めている。

この Java を使い、Web 上に教材を作成した例は数多くあり、日本でもこのような Web サイトや、教材作成のノウハウを集めた「Java 物理教材メーリングリスト[3]」がある。しかし、完成品のソフトのみが公開されており、本格的な教材作成支援ライブラリやドキュメント等があまり無く、多数の方々が独立して作られているため、見た目や操作に統一感がないものが多い。

利点

- Java の動く全てのコンピューター上で利用可能
- 教材等の数も多く、実際に講義で利用出来るものがある

欠点

- 見た目、操作に統一性がない
- カスタマイズや再利用が難しい

3 本システムの特徴

以上の点を考察し、「数式の可視化」「手軽な利用」「利用者によるカスタマイズ、再利用」を実現することが本システムの目標である。

ここで、本システムの技術的な特徴を簡単に説明する。なお、さらに具体的な詳しい説明を末尾の付録に示した。

3.1 高い再利用性

まず、今回のシステムの根幹を成す「再利用性」と「オブジェクト指向」について簡単に説明する。

再利用性

ソフトウエアに限らず一般的に、何かを作り出す時には、新しく無から作り出すよりも既存の資産を利用した場合が、コストが低く期間も短縮できる。そのため、長年ソフトウエア作成の現場においては、「再利用」は大きなテーマとなっている。

しかし、全ての資産が再利用可能であるわけではなく、再利用を前提としたそれなりの工夫と努力が無ければ再利用は難しい。例えば、どんなに優れた機能を持ったソフトウエ

[3] http://www.nep.chubu.ac.jp/ nepjava/

ア資源であっても、使い方を記したドキュメントが無ければそれを利用することは難しく、また汎用性が低く適用範囲が限定されるようであれば、その利用価値は低い。

本システムでは、次に記すオブジェクト指向技術を使い、一般の講義担当者が簡単に目的となるソフトを作成できるようにした。また、末尾の付録に示すように利用に関するドキュメントを用意し、また必要であればシステム全体のソースファイルも参照することが出来るため、利用者はほとんどの必要な情報を得ることが出来る。

オブジェクト指向

プログラミング言語にはいろいろな分類が存在するが、大きなグループとして「手続き指向」と「オブジェクト指向」がある。それぞれ一言で表わすと、手続き指向は、

> 何かやりたいこと (欲しい答え) があって、その目的達成のための作業の手順を順番に並べていくプログラムの作り方

また、オブジェクト指向は、

> 何か扱いたいデータの集団があって、それらの性質や機能、またデータ間の関係を考えていき、実際にそれらが協調しあって目標を達成するように、データ間の関係を作り上げていくプログラムの作り方

であると言える。

前者の手続き指向によるプログラムの作成方法は次のようである。

プログラムには明確な最終目標があり、その目標を処理を順番に並べていくイメージでプログラムを作っていく。途中、目標があいまいであったり、複雑な内容を含んでいる場合は、小さな処理に分解していく。(図 1 参照)

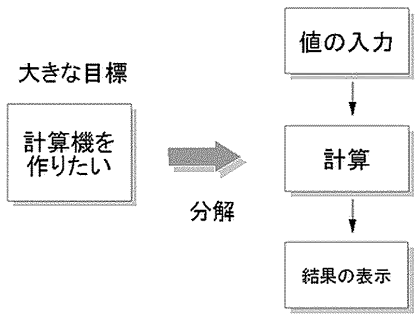


図 1: 小さな処理単位に分解していく

後者のオブジェクト指向では、プログラムを考える方法が全く異なる。

まず、扱いたい問題領域 (目標) を考え、そこに存在する動作の主体やデータ (これらをオブジェクトと呼ぶ) を洗い出していく【分析】。そして、それらのオブジェクトの性質や関係を見つけ出し、目標を達成できるように組立てていく (図 2 参照)【設計】。最後に、できた設計図を基にプログラムを組み立てる【実装】。

前述のように、オブジェクト指向は多少プログラムの作り方が複雑ではあるが、手続き指向に比べて、主に以下のような利点がある。

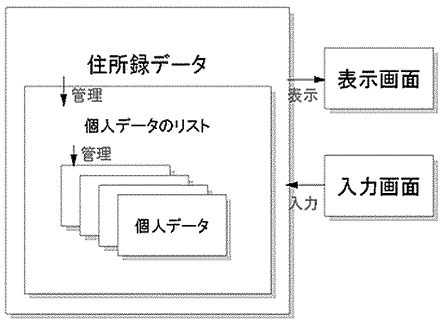


図 2: 住所録アプリケーションの例

- ソフトウエア資源の再利用が容易
- 大規模なソフトウエア開発の管理が容易
- ソフトウエア開発時間の短縮
- GUI (グラフィックを使った操作) との親和性の高さ

ここに挙げた他にもオブジェクト指向技術の利点は枚挙に暇が無く、また、適用についても様々な方法論が存在する。

3.2 利用のしやすさ

本システムは、利用形態として三種類の場面を想定している。

1. 作成された教材を使う
2. 簡易作成ソフトで教材を利用者自ら作る
3. 利用者自ら直接 Java のソフトウエアライブラリを使用して教材を作成する

まず、最初の１．については、他の Java を使用した教材と使用の形態は変わらないが、本システムに含まれる高度なグラフ表示部品、三次元表示部品等を利用しているため、出力は美しく、また操作方法もほぼ統一されたものとなっている。

次の２．については、既存の教材には無いものを自ら作成できるものである。この仕組みでは利用者はプログラムについての特別な知識は必要無く、作成したい教材のパターンを選び、質問に答えるだけで簡単な教材を作成できる。しかも、全てが Web を使用した構造になっているため、利用者のコンピューターに特別なインストール作業は必要が無い[4]。

３．については、Java のオブジェクト指向技術を駆使したライブラリを使用することが出来るので、Java のプログラムがある程度作成できる人であれば、簡単に高度な教材を作成することが出来る。

以上のように、特に講義担当者が必要な教材を作成できるところに重点を置いた。

[4]ただし、InternetExplorer4.xx などの JDK1.1.8 以上の Java 実行環境が必要。最近の Windows や Macintosh では標準で含まれている。

4　スケジュール

1999 年度の研究・開発のスケジュールは以下のようであった。本業の研究等もあり非常に忙しく、なかなか十分な時間をかけることが出来なかったが[5]、やりたいことはほぼ出来たように思う。

1999 年	6 月	'99 C&C プロジェクト参加
	7 月	試作教材を実際に講義で使用
	9 月	講義での感想等を元に、実際に求められる教材を考察
	10 月	汎用化へ向けて整備
	12 月	教材作成システムの作成
2000 年	3 月	試作品の完成

5　試作教材「基礎科学のための数学的手法」

教材が実際どのように使用され機能するか、また講義担当者及び受講者の意見感想を求めるために、具体的な教材を作成した。これらの教材群は Web ページ[6]で公開されているほか、講義を担当された九大院理：小田垣孝教授の書かれた同名のテキスト[7]で、補助テキストとして参照されている。

5.1　教材イメージ

以下に教材の一部を挙げる。

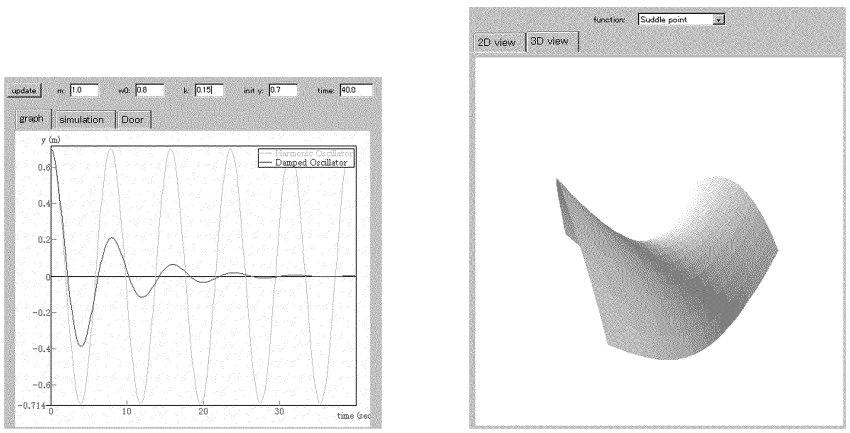


図 3: 方程式の解、グラフの表示など

図 3 のように単純に式を表示したり、微分方程式の解をグラフによって表示することが出来る。また、2 次元スカラー関数のグラフも立体的に表示し、またマウスの操作によって自由に回転したり拡大できる。

[5]特に、3 月は体を壊したり、学会等の発表の準備もあり、生きた心地がしなかった
[6]http://www.cmt.phys.kyushu-u.ac.jp/~M.Sakurai/phys/physmath/
[7]小田垣孝著「基礎科学のための数学的手法」裳華房 ISBN4-7853-2091-5

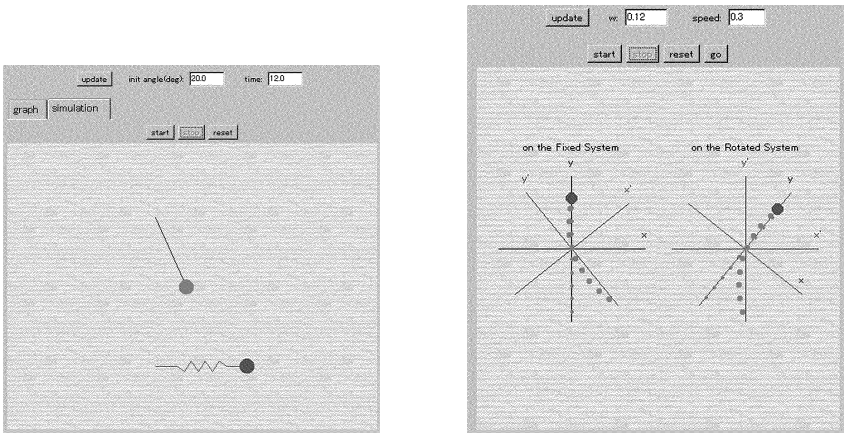


図 4: アニメーションの表示

運動方程式は初めて学ぶ生徒にとって、単にグラフに表示するだけでは実際の動きをイメージすることは難しい。図 4 のように動きを見せることによって、その仕組みを実感することができ、またパラメーターを変えるなどして自ら試すことでさらに理解が深まることが期待される。

5.2 講義と考察

教材を使用した講義は、大学 1 年生の為の力学を中心にした「物理数学 (小田垣孝教授)」で、物理を学ぶ大学新 1 年生を対象に新しく開講された講義である。

講義は以下のように行われた。

1. まず、本日学習する内容についてのアニメーションを見せ、目的を明確にする。
2. 次に、従来どおりの板書を使った講義を行い、方程式を説明する。
3. 最後に、説明した方程式を教材使って、解をグラフに表示したり、またパラメーターを変化させた様子を説明してまとめる。

以下、講義担当者と受講者の感想である。

講義担当者より

「今まで、運動の時間変化や係数の変化を分かりやすく説明することが難しかったが、少なくともこの教材を使うことによってやりたかったことが実現できて良かった。少なくとも、教えるほうには非常に便利だった。」

受講者のアンケートより

「HP(この教材) を使ったのは、やや自己満足っぽく思えた。目的が見えない。あれを見てもより理解が深まったとは思えないが、とっつきやすくはなっていたと思う」

授業に対して求められたアンケートだったので、教材について意見を述べたものはこれしか得られなかった。やや否定的であり、多少は役には立ったようであるが、この教材を

使ったやり方だけでなく、まだまだ新 1 年生を対象にした講義自体に工夫が必要であるようだった。

しかしそのほかの回答は、教材には触れていないが、かなり講義自体に満足したと言う声が多数を占めていた。講義についてのグラフを図 5 に示す。

以上より、教材だけでなく、やはりマルチメディアを使った講義についてのノウハウも研究する必要があると実感した。

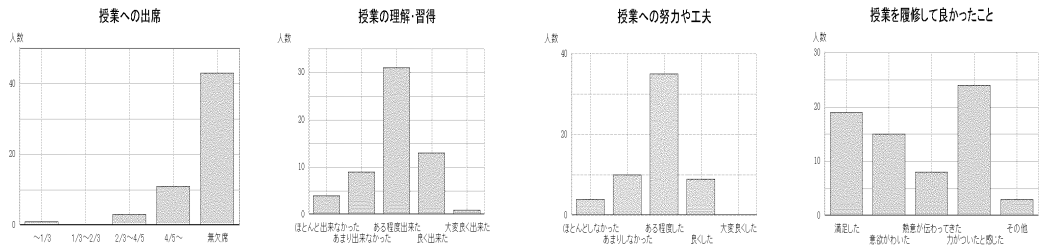


図 5: アンケートの結果の一部

6　試作システム「バーチャルラボラトリー」

次に、各教材の部品を汎用化し、簡単な操作で教材を作成できるシステム[8]を試作した。これにより、簡単なものに限られるとはいえ、講義担当者自身が、講義の内容に必要なものを自分で作成することが出きる。

6.1　紹介

今回は、技術的なデモンストレーションとして「グラフの表示 (図 6)」「1 階微分方程式の解の表示」「2 階微分方程式の解の表示 (図 7)」を試作した。

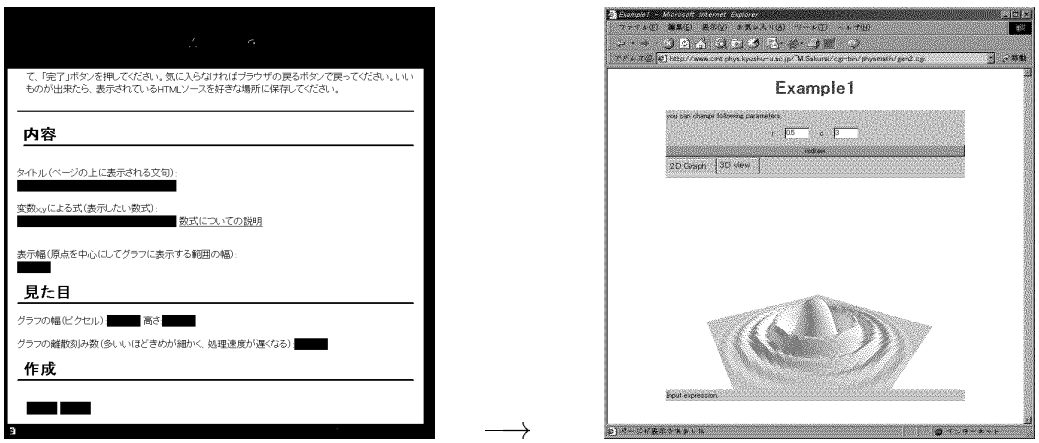


図 6: 2 次元スカラー関数表示デモの作成の様子

[8] http://www.cmt.phys.kyushu-u.ac.jp/~M.Sakurai/phys/gen/

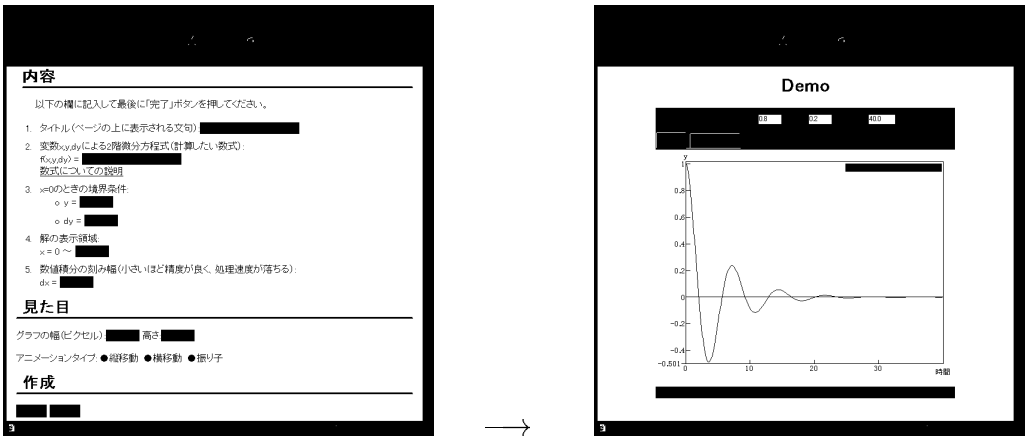


図 7: 微分方程式デモの作成の様子。グラフでの表示のほか、簡易アニメーションもつけることができる。

普段なら必要なソフトウエアのインストール作業もほとんど必要無く[9]、Web に必要な事項を記入するだけで作成が完了する。出来たページは単なる HTML ファイルとして保存しておけば、後にネットワーク接続可能な場所でもう一度表示させることができるので、講義前にさっと作って持っていき、講義で使った後に Web ページに載せておくなどの使い方が期待できる。

7 今後の展開

以上、可視化・疑似体験教材作成を支援するソフトウェアとシステムを作成した。今回試作した教材作成システムは、あくまで「数式の可視化」「手軽な利用」「利用者によるカスタマイズ」の実証的なデモンストレーションであり、現在は全ての人が満足に利用できるほど完成度は高くない。しかし、ここに示すように既に技術は確立されたので時間と気力さえあればいくらでも完成度を上げ、あらゆるニーズに対応することは可能である。

現在、システムはちょっとした機能の追加や、調整など以外の開発は諸他の都合のため停止している。作成したライブラリを含めたシステムは、もう少しまとめて一般に公開し、同様な試みを行っている方々と一緒に成長させていきたいと考えている。

また、教材の開発だけでなく、このようなマルチメディア教材を使用した授業についても、効果的な学習のための研究が必要である。

8 最後に

今回、教育とマルチメディアについて考える機会を得て、現在の教育を取り巻く問題の一端を垣間見たような気がした。学校では、文部省の計画の下、マルチメディアや情報教育のためのインフラ整備が急速に進んでいるようだが、そのハードウエアの上に立つべき、目的や方法を含めたソフトウエアの整備についての取り組みがどう進められようとしてい

[9] ただし、最新 (JDK1.1.8 以上) の Java 実行環境が必要。InternetExplorer4.xx, NetscapeNavigator4.06 以上であればほとんど問題無いが、一部実行環境側の不具合により実行できない場合がある。

るのか分かりにくいように思われた。技術立国日本の一員として、他国に負けないもっと積極的な理系教育を望みたいと思う。

9 謝辞

このような研究の機会と御助言をいただきました、杉岡洋一総長とC&Cの坂口教授をはじめ関係者の皆様には大変お世話になりました。この場を借りて感謝を申し上げます。また、C&Cの研究のために講座の方にいろいろとご迷惑をおかけしてしまいながらも、ご理解くださった凝縮系基礎論講座の小田垣教授、吉森助教授に感謝申し上げます。

10　付録：システムの解説

この章では、システムの技術的な解説を述べる。対象はJavaでプログラムがある程度作成でき、Javaのクラスライブラリを使用できるレベルを想定している。

10.1　概要

システムの基本的用件は、「数式の可視化」「手軽な利用」「高再利用性」である。よって、数値計算とグラフィック表示を汎用的に作成し、その上で教材作成ライブラリをシンプルにまとめた。(図8を参照)

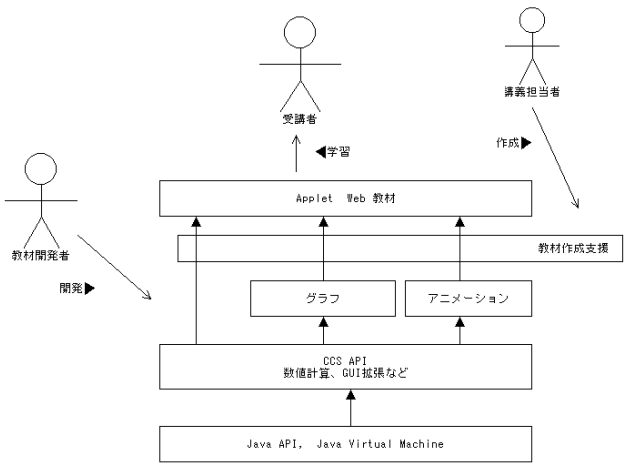


図 8: システム構成

10.2　数値計算・AWT拡張ライブラリ「CSS」

まず、数値演算やGUI(Graphical User Interface)のAWT(Abstract Window Toolkit)拡張を行い、グラフなどを表示できる拙作のクラスライブラリ「CCS(Class Collection Sakurai)」について簡単に紹介する。

数値計算部

数値計算はCCSのメイン機能である。Fortranのように速度や精度を至上とするものではないが、「Function:関数」というオブジェクトを通して、数値積分、数値微分、補完、ファイル入出力、グラフの出力、さらに微分方程式の数値積分を使いやすく実装した。また、行列・ベクトル演算、簡単な線形計算も簡単ながら実装し、一通りの数値計算を簡単に実現できるようになっている。

1 変数関数インターフェイス「sakurai.math.AFunction」

ある値を入力すると、それに応じた値を返すという関数の機能をそのまま定義したもの。定義は以下のようで、実装クラスは倍精度実数を返す f(x) だけを実装すれば良い。

```
//関数インターフェイス定義
public interface AFunction {
   public double f(double x);
}
```

使い方の一例：Test.java

```
import sakurai.math.*;

public class Test {
    public static void main(String [] args) {
        //1 変数関数の生成 sin(x)*exp(x)
        AFunction a = new AFunction() {
          public double f(double x) {
            return Math.sin(x)*Math.exp(-x);
          }
        }

        //後述のグラフ機能を利用して関数を表示
        sakurai.comp.graph.Graph.show(a);
    }
}
```

数値積分「sakurai.math.Integral」

関数オブジェクトを引数にして、指定された領域を積分する。

このような関数を引数にするようなオブジェクトを「演算子オブジェクト」で統一し、そのインターフェイスを基底インターフェイスAOperatorで定義する。AIntegralのシンプソン積分法によるASimpsonIntegralを実装した。

使用例：

```
import sakurai.math.*;
import sakurai.math.exp.*;
//クラス宣言、メソッド宣言は省略

//sin(x) を用意（後述の数式解析を使用。引数の関数を得る）
AExpression ex = new AExpression("sin(x)");
AFunction a = ex.getFunction();

//シンプソン積分オブジェクトを用意
AIntegral sim = new ASimpsonIntegral();

//区間 [0:π ] を積分して表示
System.out.println( sim.range(0,Math.PI).operate(a));
```

数値微分「sakurai.math.Differential」

積分と同様に、関数を引数にして数値微分を行う。
使用例：

```
//sin(x) を用意
AExpression ex = new AExpression("sin(x)");
AFunction a = ex.getFunction();

//1 階微分する演算子
ADifferemtial dif = new AFirstDifferential();
//関数 a を位置 0.25 において微分した値を表示します
System.out.println( dif.point( 0.25 ).operate( a ) );
```

この応用で関数から導関数を生成するADifferentialFunctionも用意した。

配列と関数「sakurai.math.AArrayalFunction」

数値を配列変数に格納して使う時に、この配列を実数でアクセスして関数のごとく扱いたいときがよくある。また、配列の間の点を補完して滑らかにするというケースもよくある。そこで、関数オブジェクトと配列をつなぐクラスAArrayalFunctionを用意した。
使用例：

```
import sakurai.math.*;

//main
double [] x = {0,1,2, 3, 4};
double [] y = {0,1,0.5,0.25,0.125};//配列から

//関数を作る
AArrayalFunction af = new AArayalFunction( x, y );
//スプラインで補完して（デフォルトは線形補完）
af.setInterpolater(new SprineInerpolater());

//後は普通の関数と同じように使える
AIntergral it = new ASimpsonIntegral();
System.out.println( it.range(0,4).operate( af ) );
```

ベクトルと行列「sakurai.math.Vector, Matrix」

多成分の数値を扱う際に必要なベクトルと行列を表わすクラスである。基本的な代数演算を定義し、数値計算から 3 次元グラフィックスまで応用できる。
使用例：

```
import sakurai.math.*;

//main
//成分を用意
double [][] ar = new double [2][2];
ar[0][0]=2;ar[1][0]=1;
ar[0][1]=5;ar[1][1]=4;

//行列オブジェクトの作成
Matrix mt1 = new Matrix(ar);
System.out.println(mt1);

//逆行列
Matrix mt2 = mt1.inverse();
System.out.println(mt2);

//行列の計算
Matrix e = mt1.mult(mt2);
System.out.println(e);

//ベクトルの作成・計算
MathVector v1 = new MathVector(1,2);
MathVector v2 = mt1.mult(v1);
System.out.println(v2);
```

スカラー関数、ベクトル関数「sakurai.math.ScalarFunction, VectorFunction」

前述のベクトルを使用して多成分のスカラー関数 ScalarFunction、ベクトル関数 VectorFunction を表わすクラスを作成し、またベクトルの微分演算子も定義した。このため、以下の使用例のように簡単に相互の変換ができるので、例の結果の図 9 のように、容易にベクトルやスカラーの確認ができる。

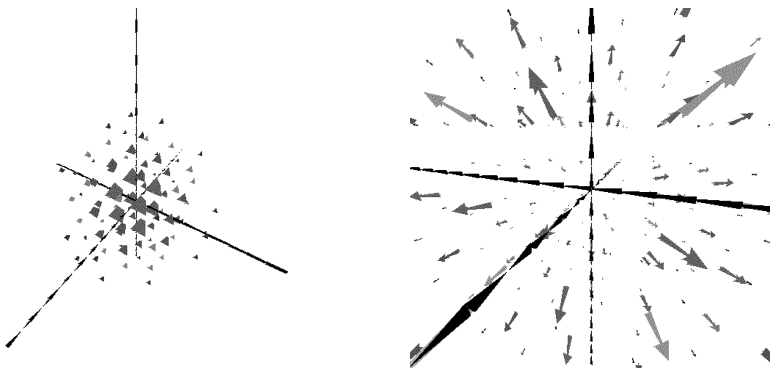


図 9: 3 次元でスカラー関数、ベクトル関数の表示

使用例：

```
import sakurai.math.*;
import sakurai.comp.graph.Graph;

//分布電荷ポテンシャルクラス(スカラー関数の応用)
class Potential implements ScalarFunction {
    double q=1;//電荷
    public double f(MathVector r) {
        // = q*exp(-r^2)
        return q*Math.exp(-r.getSquare());
    }
    public int getDimension() {return 3;}
}

//main

// ポテンシャルの作成 φ
ScalarFunction potential = new Potential();
//ベクトル微分(E =∇φ )
VectorFunction E = new Gradient(
        FunctionUtil.multiple(potential,-1));
//ベクトル微分(ρ =∇・E)
ScalarFunction divE = new Divergence(E);
//ベクトル微分(0 =∇× E)
VectorFunction rotE = new Rotation(E);
//それぞれ表示
Graph.show(potential);
Graph.show(E);
Graph.show(divE);
```

微分方程式「sakurai.math.difeq.DiffEqSolver, DiffEqSolver2など」

微分方程式を数値積分によって求めるクラス群である。このフレームワークを使用すると、解きたい微分方程式を定義して代入するだけで適当な解を得ることができるので、特に力学などの方面で威力を発揮する。様々な積分アルゴリズムも使用することや、簡単な連立積分方程式も解くことも可能である。

使用例：調和振動子($\frac{d^2y}{dx^2} = -y$)

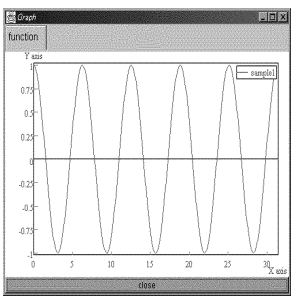


図 10: 調和振動子を解いて表示

```
import sakurai.math.*;
import sakurai.math.exp.*;
import sakurai.math.difeq.*;
import sakurai.comp.graph.Graph;

//main
//方程式の作成 (f = -y)
Expression ex = new Expression("-y");
ScalarFunction equation = ex.getFunction();
//積分エンジンの構築
DiffEqSolver2 eng = new DiffEqSolver2();
//初期条件の設定 (x0, y0, dy0)
MathVector init = new MathVector(3);
init.set(0,1,0);
//解を得る ( x=0 ~ π の範囲 )
AArrayalFunction [] s = eng.solve(equation,init,Math.PI*10);
Graph.show(s[0]);
```

数式解析部「sakurai.math.exp.AExpression, Expression」

普通、数式はあらかじめソースファイルに書いておくものであるが、実行時に動的に式を構築したい場合がよくある。例えば、式を入力してそのグラフを表示するような場合、文字列として入力された式を評価して値を得なければならない。また、今までの「使用例」にも使ってきたとおり、簡単なプログラムの作成時にも非常に有効である。今回、Sun Microsystems が配布している JavaCC (Version 0.8pre2) を使用して、構文解析を行うクラスを作成した。Fortran 的な文法で数式をかける上、基本的な関数、また未定義な文字を後で値を代入して使う変数として扱うことが出来るなど、今回の教材作成の中で大変苦労した部分であり、大きな役割を担っている。

使用例は今までの例に出てきているので割愛する。

その他

線形連立方程式の解や、体積積分、FFT、代数方程式の解など、よく使われる機能を実験的ながら実装している。詳しくは CCS API ドキュメントやソースを参照。

10.3 グラフ表示部「sakurai.comp.graph. ~ 」

あらゆるグラフを表示する重要な部品であり、今回の教材の可視化の要である。多くのクラスから成り立っており、データの形式や表示の方法などを柔軟に対応できるように設計した。簡単に使用するには Graph.show(関数) が便利である。詳しくは CCS API ドキュメントやソースの使用例を参照。

特徴

- 背景に画像などを指定可能

- 2 次元平面
  - 自動的に領域、目盛の表示
  - グラフの分割や重ね合わせ
  - ベクトルや等高線の表示
  - ヒストグラム、円グラフなどの任意のグラフの形状に拡張可能
- 3 次元空間
  - 平面の表示
  - ベクトルなどの場の表示
  - 3 次元空間のスカラー量などの表示
  - 赤青メガネによる擬似 3 次元表示など

構成図

図 11 と図 12 に UML によるクラス間の静的な構成を示す。

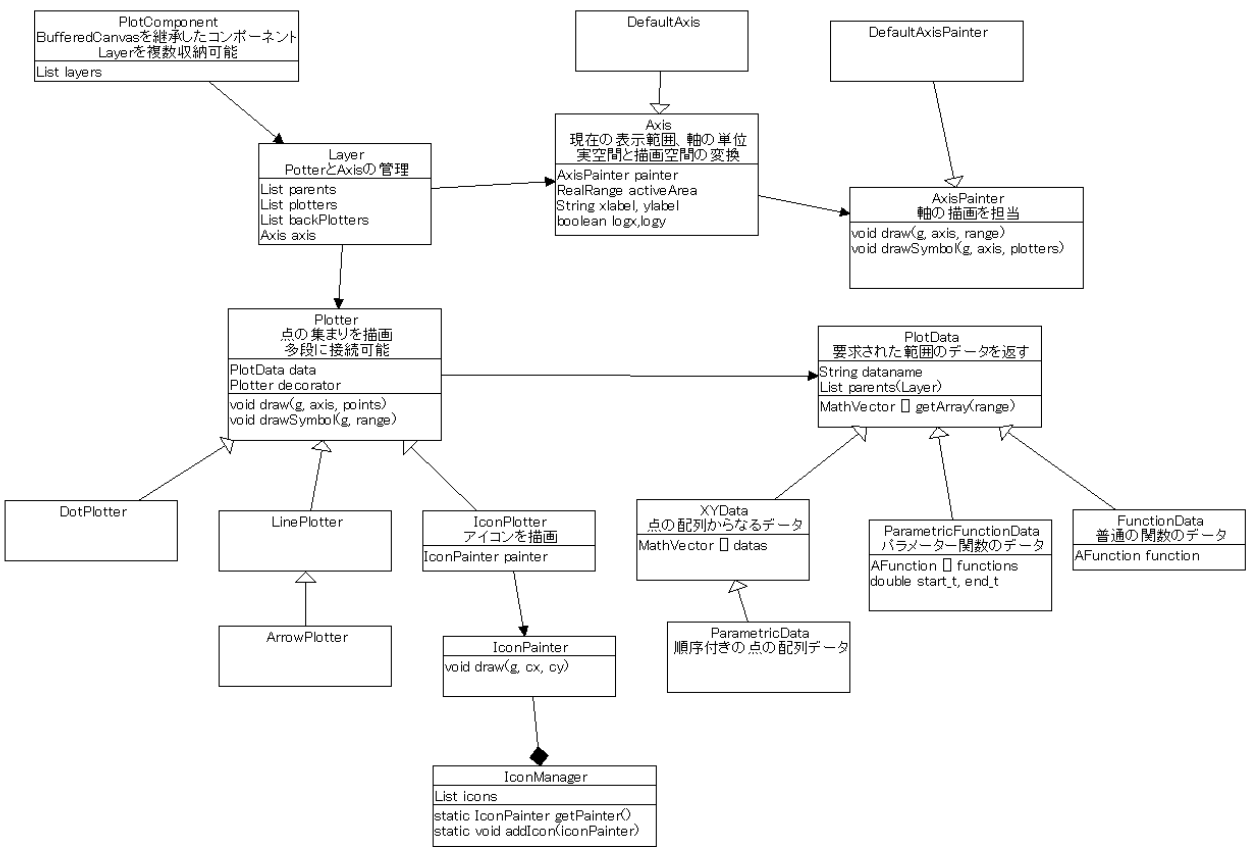


図 11: sakurai.comp.graph パッケージ

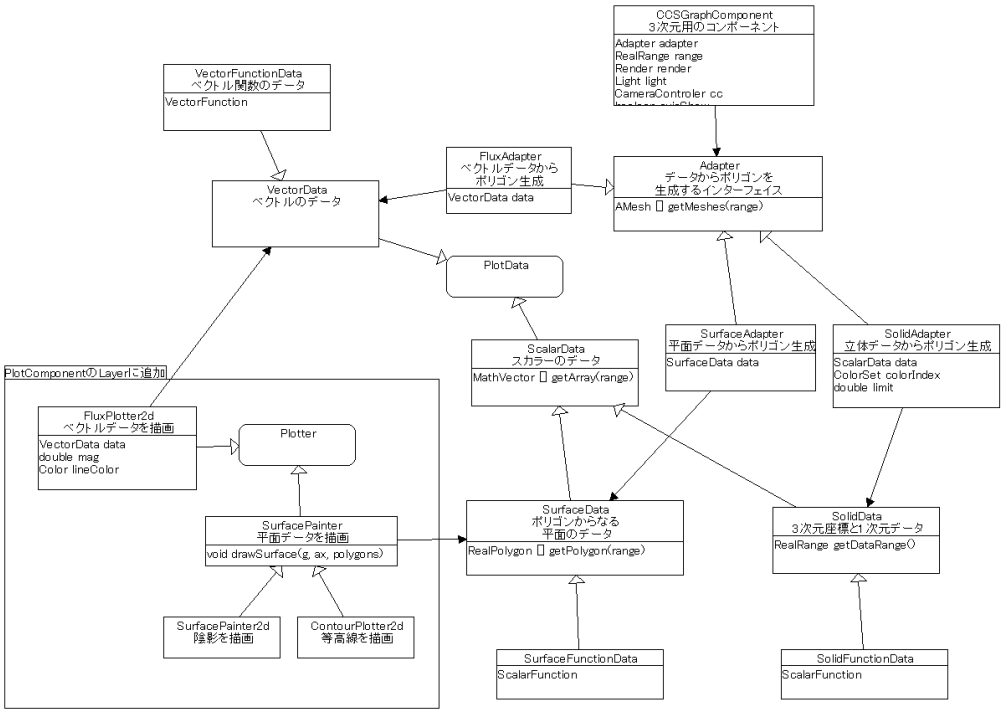


図 12: sakurai.comp.graph.three パッケージ

10.4 アニメーション表示系

アニメーションの表示系は、グラフ表示と並んで本システムのもう一つの目玉である。大まかな構成は図 13 のようになっており、物体PhysObject を何かしらの時間の関数で動かし、それに付随して物体間の接続を表わすConnection や各時間のイベントを定義するPhysEvent が集まって一つのシーンPhysScene を構成している。

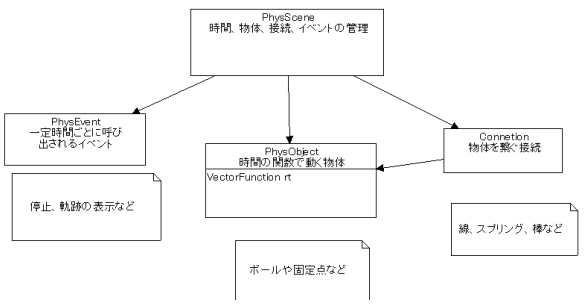


図 13: phys.anm パッケージの概要

実際の使い方は、パラメーターなどを使用した運動方程式を、微分方程式の積分エンジンなどで解き、物体にセットしてPhysScene に登録する。後はPhysScene.start() で勝手に時間を進めて動かしてくれる。必要に応じて物体を増やしたり、接続線でつないだりと、簡単な構成ながらかなりの表現を実現することができる。詳しくはphys.applets 以下のソースを参照。