数学・理科におけるコンピュータ利用

── コンピュータの画像等を利用し生徒たちに視覚的に訴える ──

土 田 宏 治（函館ラ・サール中学・高等学校教諭）

第１章　はじめに

我が国は、石油・石炭・鉄鉱石など、地下資源が乏しい。しかしその反面、しっかりと教育されて優秀な人材資源には豊富に恵まれている。そこで、これまで「科学技術立国」あるいは「頭脳立国」を国是として、世界の人々の物質的生活を豊かにし、かつ科学の進歩に絶大な貢献を果たしてきた。このことに関して、私学教育が担ってきた役割は極めて大きいものがある。ただ、最近、気になることがある。高校生の「理数科目嫌い」、さらには「理工系進学志望者数の激減」ということである。かつて、高校３年生のクラスでは、大多数の生徒たちが意欲的に理工系分野を志望し、大学の競争率も激しく、どこも難関を極めていた。当然、優秀な技術者・科学者が次々に育てられ、様々な分野に大量に送り込まれ、めざましい活躍を遂げてきた。ところが、ここ数年、理工系進学希望者は、クラスには、ほんの数名、数えるほどしかいなくなってしまった。このままこの状態が続くならば、この国に将来はありえない。どうしてこのようなことになってしまったのか。追究してみると、確かにいろいろな原因がある。その中でも特に重大なものとして、生徒たちの意識に「理数系の科目はむずかしい。分かりにくい。苦手だ。」という感覚が根強く存在するようになってしまったということがあげられる。さらには、その理由として「とにかく数式や計算がむずかしく、うんざりしてしまう。こんなものなければいい。」ということである。理数科目において、数式は必要不可欠なものである。数式は科学を語る言語である。数式なくして科学は存在しえない。折角、興味・関心を持って取り組み、学習を始めたものの、この複雑多様で難解な数式に圧倒され、混乱してしまい、挫折してしまう生徒があまりにも多い。

そこで、「理数科目への興味・関心・意欲」を生徒たちに再び取り戻させるためにコンピュータを活用することを考えた。数式を式そのものとしてみるだけでは、無味乾燥で、とっつきにくい。しかし、それを図形として表現し、認識するならば、視覚的に実感を伴ってとらえることができる。数式の中には時間的な動きや確率的な変化を伴うものもある。複雑で緻密な画像を黒板とチョークだけで正確に表現することは、なかなか困難である。特に動きのあるものを表現することは、ほとんど不可能に近い。たとえ、できたとしても、その準備には膨大な時間と労力を必要とする。ところが、これらは、コンピュータを利用することによって、比較的容易に実現可能となる。コンピュータで作成した図形画像を彼らの眼前に大きく提示することによって、彼らに驚き・感動を与えることができる。そして、このことが、より現象の理解を深め、学習への興味・関心・意欲を高めさせることになる。

また、コンピュータでは計算を瞬時に行うことができるので、パラメータの変化による違いをすばやく認識することができる。さらには、コンピュータでは乱数を使って確率的な現象を容易に実現することもできるなど、これらの機能を有効に活用した。

数学の分野では、微分・積分、確率・統計、平行移動、対称移動、一次変換、関数、２次曲線（楕円・双曲線・放物線、離心率)、三角関数、漸化式、パラメータ表示（サイクロイド・カージオイド・レムニスケート･リサージュ･･･）等において、物理学の分野では、放物運動、円運動、衝突、単振動、波動、電界、磁界等において、化学の分野では、反応速度、化学平衡、原子の構造、結晶、滴定曲線（酸塩基・酸化還元）等において、生物学の分野では遺伝等において、様々な活用ができる。これらの場面におけるより有効なプログラムの作成、システムの構築を研究した。

第２章　使用するコンピュータ言語および基本画面

第１節　使用コンピュータ言語および使用機器

「Microsoft Visual Basic.NET」を使用した。この言語は、多少の厳密さ・きめの細かさには欠けるものの、柔軟性があって使い易く小回りが利き、今回の研究題目についてのシステムを構築する言語として必要十分なものである。また、「Excel」や「Word」などに搭載されている言語もこの「Visual Basic」をベースとした言語（VBA）であり、この言語は一般に広く使われ、身近なものとなっている。

また、提示手段として、プロジェクターを使用した。

第２節　基本画面

まず、グラフや図形を表示するための座標が必要である。以下にそのプログラムを示す。

```
1.      Dim grph As System.Drawing.Graphics = Me.CreateGraphics
2.      Dim drawpen As New Pen(Color.Black, 2)
3.      Dim fillbrush As New SolidBrush(Color.Black)
4.      grph.DrawLine(drawpen, 50, 375, 1200, 375)
5.      grph.DrawLine(drawpen, 625, 50, 625, 700)
6.      grph.DrawLine(drawpen, 1200, 375, 1185, 367)
7.      grph.DrawLine(drawpen, 1200, 375, 1185, 383)
8.      grph.DrawLine(drawpen, 625, 50, 633, 65)
9.      grph.DrawLine(drawpen, 625, 50, 617, 65)
10.     grph.DrawString("x", New Font("MS UI Gothic", 18, Font.Style.Bold), fillbrush0, 1200, 361)
11.     grph.DrawString("y", New Font("MS UI Gothic", 18, Font.Style.Bold), fillbrush0, 615, 24)
12.     grph.DrawString("0", New Font("MS UI Gothic", 16, Font.Style.Bold), fillbrush0, 605, 375)
13.     grph.DrawString("タイトル名", New Font("MS UI Gothic", 20, Font.Style.Bold), fillbrush0, 200, 50)
```

「1.」 フォーム（Me）にグラフィクスを作成するための変数grphを宣言する。

「2.」 ペンdrawpen（色＝黒、太さ＝２ピクセル）を作成する。

「3.」 ブラシfillbrush（色＝黒）を作成する。

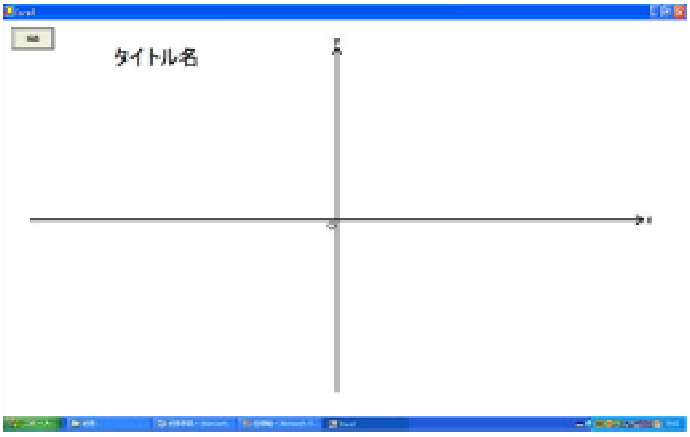
「4.－9.」 drawpenを使って、矢印のついたｘ軸・ｙ軸を作成する。

「10.－12.」 指定されたフォントで、fillbrushを使って、x、y、原点Oを記入する。

原点の座標は（625，375）である。

「13.」 画面にタイトル名を記入する。

ただし、この座標は標準のものであり、扱うグラフ・図形によって、座標軸を移動することや、場合によっては、消去するなどの変形を行う。



第３章　活用例

第１節　　リサージュ曲線（ｘ＝ａ sin ｍθ、ｙ＝ａ sin ｎθ）

リサージュ曲線を描画する。以下にそのプログラム（一部）を示す。

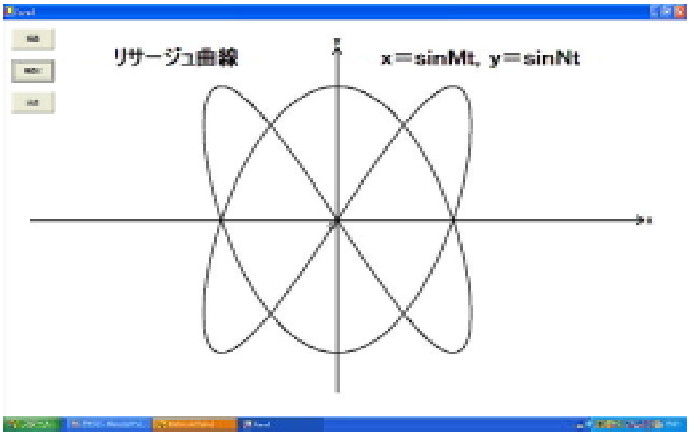
```
1.      a = 250
2.      m = 2
3.      n = 3
4.      For i = 0 To 359 Step 2
5.          ang = i / 180 * Math.PI
6.          x1 = a * Math.Sin(m * ang)
7.          y1 = a * Math.Sin(n * ang)
8.          ang = (i + 2) / 180 * Math.PI
9.          x2 = a * Math.Sin(m * ang)
10.         y2 = a * Math.Sin(n * ang)
11.         grph.DrawLine(drawpen1, 625 + x1, 375 - y1, 625 + x2, 375 - y2)
12.     Next
```

「1.」－「3.」ａ、ｍ、ｎの値を指定する。

「4.」－「12.」座標（ｘ，ｙ）を計算し、グラフを表示する。

これは、一気にグラフを表示するプログラムである。タイマーコントロール（Timer）を使うことに

よって、描画の手順や動きを把握することができる。



第２節　円周率πを求める

乱数を利用して円周率を求める。以下にそのプログラム（一部）を示す。

```
1.      x = Int(Rnd() * 500) + 1
2.      y = Int(Rnd() * 500) + 1
3.      grph.DrawEllipse(drawpen, 500 + x, 600 - y, 2, 2)
4.      If ((x - 500) ^ 2 + (y - 500) ^ 2) < 500 ^ 2 Then naka = naka + 1
5.      sousuu =  sousuu + 1
6.      ensyuuritu = naka / sousuu * 4.0
7.      Label1.Text = Format(sousuu, "0")
8.      Label2.Text = Format(ensyuuritu, "0.000")
```

「1.－2.」　１以上500以下の乱数を発生させ、ｘ座標、ｙ座標に代入する。

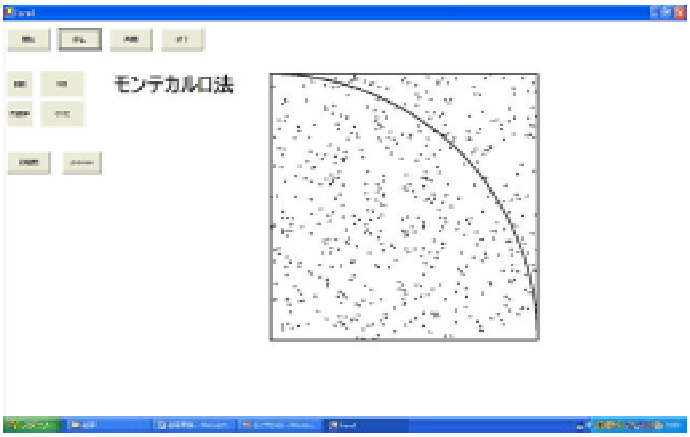
「3.」　点（ｘ，ｙ）をグラフ上に表示する。

「4.」　点（ｘ，ｙ）うち四分円の中にあるもの（naka）を数える。

「5.」　点（ｘ，ｙ）の総数（sousuu）を数える。

「6.」　円周率＝全体のうち四分円の中にあるものの割合×4.0として円周率（ensyuuritu）を求める。

「7.－8.」　点（ｘ，ｙ）の総数と円周率の値を表示する。



第３節　二項分布

乱数を利用して二項分布のグラフを求める。以下にそのプログラム（一部）を示す。

```
1.      Dim p(100) As Integer
2.      n = 100
3.      m = 100000
4.      For i = 1 To m
5.          count = 0
6.          For j = 1 To n
7.              If Int(Rnd() * 2) + 1 = 1 Then count = count + 1
8.          Next
9.          p(count) = p(count) + 1
10.     Next
11.     x = 0
12.     For i = 0 To n
13.         y = p(i) / m * 5000
14.         grph.FillEllipse(fillbrush, 297 + x, 597 - y, 7, 7)
15.         x = x + 8
16.     Next
```

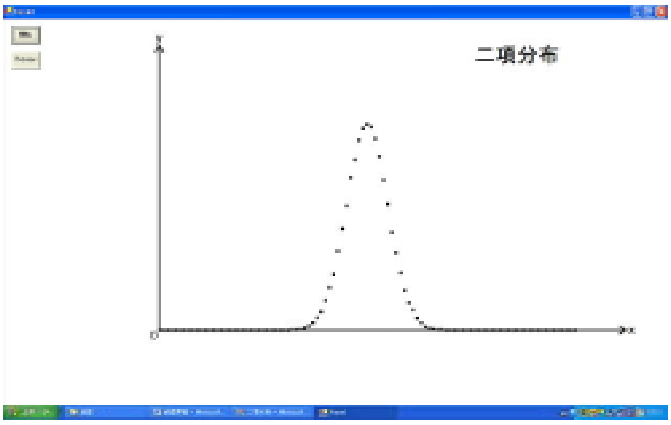
「1.」 配列変数p(100)を宣言する。

「2.」 独立試行をｎ（＝100）回行う。

「3.」 ｎ回の独立試行をｍ（＝100000）回行う。

「4.」ー「10」 独立試行の実行。１回の試行で事象が起こる確率は１／２である。

「11.」ー「16」 求められたグラフを表示する。



第４節　半減期

乱数を利用して半減期のグラフを求める。以下にそのプログラム（一部）を示す。

```
1.      Dim nokori(50) As Integer
2.      nokori(0) = 100000
3.      For i = 1 To 50
4.          count = 0
```

```
5.          For j = 1 To nokori(i - 1)
6.              If Int(Rnd() * 12) + 1 = 1 Then count = count + 1
7.          Next
8.          nokori(i) = nokori(i - 1) - count
9.      Next
10.     x = 2
11.     For i = 0 To 50
12.         y = 10
13.         For j = 1 To nokori(i) / 2000
14.             grph.DrawEllipse(drawpen, 300 + x, 600 - y, 7, 7)
15.             y = y + 9
16.         Next
17.         x = x + 12
18.     Next
```

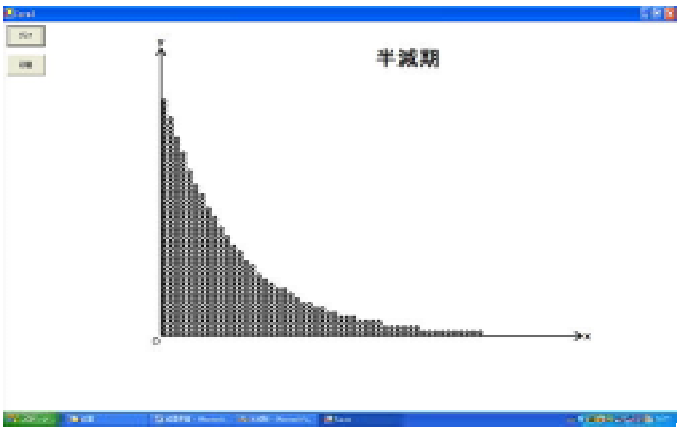
「1.」 配列変数nokori(50)を宣言する。

「2.」 最初の原子核数をnokori(0)（＝100000）個とする。

「3.」－「9.」 乱数を用いて、ひとつの時間間隔ごとの壊変数（count）を求める。

壊変確率は１／12である。

「10.」－「18.」 求められたグラフを表示する。



第５節　化学平衡

乱数を利用して化学平衡のグラフを求める。以下にそのプログラム（一部）を示す。

```
1.      Dim nokori1(90), nokori2(90) As Integer
2.      nokori1(0) = 100000
3.      nokori2(0) = 0
4.      For i = 1 To 90
5.          count1 = 0
6.          For j = 1 To nokori1(i - 1)
7.              If Int(Rnd() * 25) + 1 = 1 Then count1 = count1 + 1
8.          Next
9.          count2 = 0
10.         For j = 1 To nokori2(i - 1)
```

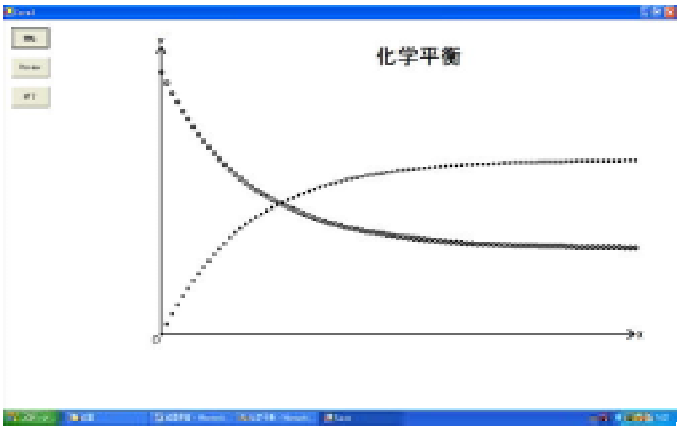
```
11.             If Int(Rnd() * 50) + 1 = 1 Then count2 = count2 + 1
12.         Next
13.         nokori1(i) = nokori1(i - 1) - count1 + count2
14.         nokori2(i) = nokori2(i - 1) + count1 - count2
15.     Next
16.     x = 0
17.     For i = 0 To 90
18.         y1 = nokori1(i) / 200
19.         y2 = nokori2(i) / 200
20.         grph.DrawEllipse(drawpen, 297 + x, 597 - y1, 7, 7)
21.         grph.FillEllipse(fillbrush, 297 + x, 597 - y2, 7, 7)
22.         x = x + 10
23.     Next
```

「1.」配列変数nokori1(90)、nokori2(90)を宣言する。

「2.」－「3.」最初の反応物１の個数を100000個、生成物２の個数を０個とする。

「4.」－「15.」反応物１が生成物２に変化する確率を１／25、生成物２が反応物１にもどる確率を１／50とする。

「16.」－「23.」求められたグラフを表示する。



第６節　ドップラー効果

動画を使うとドップラー効果の理解が容易になる。以下にそのプログラム（一部）を示す。

```
1.      v0 = 60
2.      v = 40
3.      x0 = k * 5 * v / v0
4.      grph.FillEllipse(fillbrush0, 620 + x0, 370, 10, 10)
5.      For j = 0 To 20
6.          If k >= j * 12 Then
7.              r = k * 5 - j * v0
8.              a = j * v
9.              If r <= 300 Then
10.                 For i = 1 To 360
11.                     ang = i / 180 * Math.PI
```

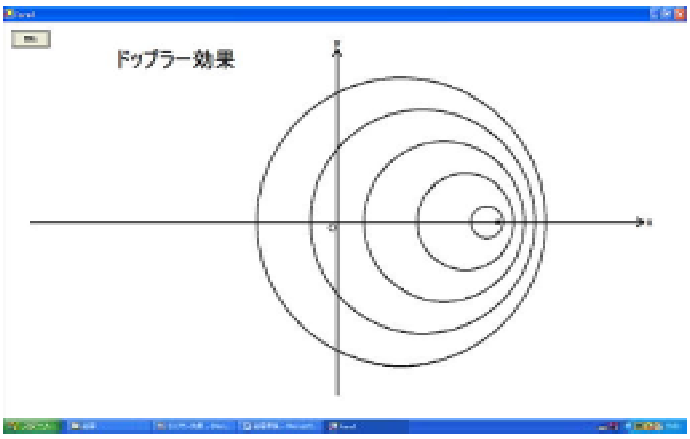
```
12.                     x1 = r * Math.Cos(ang) + a
13.                     y1 = r * Math.Sin(ang)
14.                     ang = (i + 1) / 180 * Math.PI
15.                     x2 = r * Math.Cos(ang) + a
16.                     y2 = r * Math.Sin(ang)
17.                     grph.DrawLine(drawpen1, 625 + x1, 375 - y1, 625 + x2, 375 - y2)
18.                 Next
19.             End If
20.         End If
21.     Next
22.     k = k + 1
```

「1.」－「2.」音波が伝わる速さv0、音源が移動する速さvを設定する。

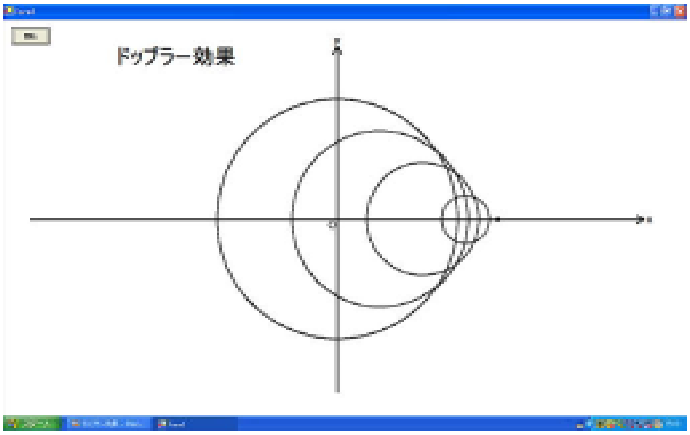
「3.」－「4.」時々刻々の音源の動きを表示する。

「5.」－「22.」時々刻々の波（円形波）の動きを表示する。

音源の速さv ＜ 音速v0のとき



音源の速さv ＞ v0音速のとき（超音速）



第４章　考察および今後の課題

コンピュータを活用する有益性として、主に次の４点があげられる。

① 正確なグラフや図形を短時間かつ容易に表現することができる。
② 確率的な現象を扱うことができる。
③ 動きや手順を表現できる。
④ 生徒の意識に、より大きな驚きや衝撃を与えることができる。

①について、チョークを使って黒板に書こうとすると、正確さや複雑さによっては長時間を必要とする。特に立体図形を分かりやすく描くことは難しい。OHP を利用したとしても、その準備には多くの労力と時間がかかる。その割には、臨機応変にパラメータ変化による違いを表現することができない。

②について、コンピュータでは　Rnd（　）とういう関数を使うことによって、容易に確率的な現象を扱うことができる。実際にサイコロを振らせたり、ジャンケンをさせたり、カードを引かせたりするなどによって、生徒たちに実験させる方法もあるが、試行回数は、せいぜい数百回が限界で、数万回・数十万回の試行を繰り返すことは事実上、不可能である。

③について、コンピュータではタイマーコントロール（Timer）を使うことによって、容易に動画を作成することができる。特に、物理の波動分野では、進行波と定常波、縦波と横波、ドップラー効果などにおいて、動きを認識せずに真の理解はありえない。

④について、大画面で幾何学的に美しい図形を目の前に見せられると、感動的である。コンピュータそのものや、その知識が広く普及している今日、「家に帰って、自分でもやってみよう。」という意欲的な生徒も多数出てきている。

この紀要で、活用例をいくつか紹介したが、字数の関係で紹介しきれなかったものがまだまだたくさんある。また、授業中に使用するのみでなく、生徒の自学自習の手助けになるものとしてなど、コンピュータを利用することによって効果的な場面が無限といっていいほどに存在する。今後、さらに発展させ、深めていかなければならない研究課題である。