College Analysis による物理シミュレーション３

ー幾何シミュレーションー

福井正康

福山平成大学経営学部経営学科

概要

我々は統合プログラム College Analysis のグラフィック機能を利用して、高等学校から大学教養課程程度の授業に利用できる、数学的な記述を利用した幾何アニメーションと物理シミュレーションのプログラムを作成してきた。この論文では幾何アニメーションと物理シミュレーションの融合について議論する。対象とする現象は、質点系の運動、重力による惑星運動、導体の静電誘導、点電荷と導体電荷による電気力線、電流による磁力線、電場と磁場による点電荷の運動である。

キーワード

College Analysis，物理，シミュレーション，アニメーション

URL:　http://www.heisei-u.ac.jp/ba/fukui/

１．はじめに

我々は、社会システム分析ソフトウェア College Analysis の中で、数学的な記述による 2 次元及び 3 次元幾何アニメーションプログラムを作成した[1]。これは関数とグラフ、数学と運動の関係などを理解するために作られたプログラムであるが、その根底には物理シミュレーションの際の運動するボールや惑星、その周囲の床や壁、バネや紐、惑星の表面画像や空間物質等、シミュレーションを取り巻く環境についてアニメーションを用いて描画したいという目的があった。また、これまでの我々が作成したシミュレーションは、計算を終えた後、結果を表示するもので、運動の軌道が表示できるなどの良い点もあるが、実行時間に制限があり、長時間動かし続けることは不可能であった。我々はこの問題についても、計算と描画を繰り返すことによって、リアルタイムにシミュレーションを観察できるようにしたいと思う。この論文ではアニメーションとシミュレーションの融合の問題について議論する。

我々が最初に組み込んだ物理シミュレーションは、質点系の運動、重力による惑星運動、静電誘導、電気力線、電流による磁力線、電磁気による運動である。殆どの問題はリアルタイムに変化や運動を計算可能であるが、例えば静電誘導のように、計算時間がかかるものもある。その場合には動的なシミュレーションの形式は取らず、その部分は静的な動きのないシミュレーションとする。例えば、静電誘導の起きた導体の近くを微小な電気を帯びた粒子が運動するような場合、最初に、導体の静電誘導の計算を静的な問題として実行した後、動的な問題として粒子の運動を計算する。静的な処理と動的な処理を、計算機の能力の限界を見て思い切りよく使い分け、できるだけリアルタイムに利用者がシミュレーションを楽しめるようにすることが重要であると考えている。

最近、物理エンジンを組み込んだ美しい剛体や流体のシミュレーションツールやゲームが作られているが、我々のプログラムには残念ながらこのような機能はない。しかし、シミュレーションの範囲を電磁気学まで拡げ、さらに他の分野も視野に入れて今後の拡張を考えている。1 つのプログラム環境の中に、どれだけのシミュレーションを含めることができるか、またそれらを目の前でリアルタイムに表現するにはどのような方法が考えられるか、この論文は今後の可能性を示唆するものである。

２．データ構造とプログラム

ここでは幾何アニメーションのプログラム構造について簡単に説明しておく。参考文献 1 の 3 次元幾何アニメーションの利用法を再掲する。

3 次元幾何アニメーションのコマンド

描画範囲変更

rangex xmin, xmax　（rangey ymin, ymax / rangez zmin,zmax）
range min, max　'すべての軸の変更

時間変数（time=0,1,2,…）

time
定義
define 変数名 = 定義式
ループ（動く変数名は lp=0,1,2,…）
loop 回数
図形描画
endloop
面図形表示モード
paint mode　　mode 1:面と縁, 2:面のみ, 3:縁のみ　デフォルトは面と縁
描画開始・終了時間（秒）
starttime 秒
stoptime 秒
描画条件
disp 式　　式≧0 のときそれ以降を描画
背景黒色
black
描画繰り返し
playback 時間（秒）
座標軸描画
axis
図形カットモード
cutmode
初期角度指定
angle 角度(度)
小さな球体　tr:不透明度
ball (x, y, z), r, color [, tr]
大きな球体　theta:傾き(度) phi:向き(度) rot:回転(度/秒) tdiv:theta 方向分割数 pdiv:phi 方向分割数
sphere (x, y, z), r, color [, theta, phi, rot, tdiv, pdiv, tr]　　spher も可
三角形
tri (x1, y1, z1)-(x2, y2, z2,)-(x3,y3,z3), color, tr
正 n 角形 n:正 n 角形 rdiv:r 方向分割数
poly (x, y, z), r, n, color [, theta, phi, rot, rdiv, tr]
小さな直方体（描画に多少の乱れが生じる）
box (x, y, z), wx, wy, wz, color [, theta, phi, rot, tr]
連結 mode: 1:細線 2:太線 3:矢印 4:太矢印 5:バネ
connect (x1, y1, z1)-(x2, y2, z2), color [, mode, バネ半径/矢印長さ]
関数　func x=f(y, z), func y=f(z, x) も可
func z=f(x, y), color [, xmin, ymin, xmax, ymax, div, tr]
func f(x, y), color [, xmin, ymin, xmax, ymax, div, tr]
パラメータ曲線［0<=u<=1］　mode 1:細線 2:太線 3:矢印 4:太矢印, r:矢印長さ
param1 (x(u), y(u), z(u)), color [, mode, div, r]
パラメータ曲面［0<=u,v<=1］
param2 (x(u,v), y(u,v), z(u,v)), color [, div1, div2, tr]
文字列　文字列の中に<変数名>も可　color=image1-10
string (x,y,z), "文字列", color [, size, "format1", "format2", …]
または print (x,y,z), "文字列", color [, size, "format1", "format2", …]

幾何アニメーションの各描画要素は構造体として表される。例えば、3 次元のボールは以下の構造体で与えられる。

```
Structure a3_balldata
    Dim sx() As String          'ボールの中心座標
    Dim r As String             '半径
    Dim c As String             '色
    Dim tr As String            '不透明度（0～1,1 のとき不透明）
    Dim starttime As Double     '描画開始時刻
    Dim stoptime As Double      '描画終了時刻
    Dim dispmode As String      'disp 式 のように使い、式≧0 のとき表示
End Structure
```

また、0≦u,v≦1 のパラメータで表現される 3 次元パラメータ関数は以下の構造体で与えられる。

```
Structure a3_paramdata2
    Dim sx() As String          'u,v のパラメータで与えられる座標
    Dim c As String             '色
    Dim div1, div2 As Integer   '関数の分割数（u,v の分割数）
    Dim tr As String            '不透明度
    Dim paint As Integer        '表示法（0:線と面，1:線のみ，2:面のみ）
    Dim imode As Integer        '0:指定された色，1～:テキスチャー画像の番号
    Dim starttime As Double     '描画開始時刻
    Dim stoptime As Double      '描画終了時刻
    Dim dispmode As String      ' disp 式 のように使い、式≧0 のとき表示
End Structure
```

描画データへの変換はこれらの構造体のデータを元に行われる。プログラムの流れは、図 2.1 のようになっている。

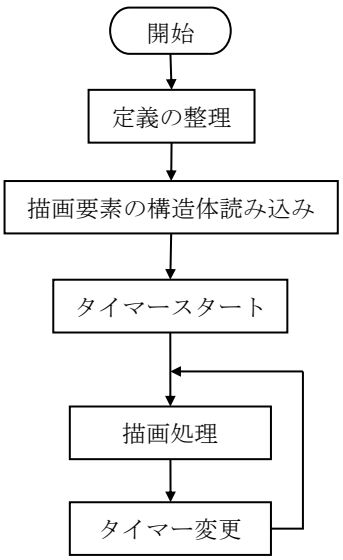

図 2.1　アニメーションプログラムの流れ

「定義の整理」とは、set 命令による数式を計算して変数の値とし、define 命令による定義から、定義の階層構造を整理し、単独で値が求められる数式に変える処理である。その際、定義式に書かれた数式は逆ポーランド形式に変換された上で処理される。その後、図形の描画命令は各図形の構造体に読み込まれる。描画命令に含まれる数式［例えば (lp+1)*sin(time/2) ］は逆ポーランド形式［ LP 1 + TIME 2 / SIN * ］に変換され、上で述べた定義がその数式に代入される。複数の図形をまとめて定義するループ処理の変数「lp」もこのとき値が代入される。

これらの処理が終わったらタイマーがスタートし、描画処理が開始される。その際、時間を表す time 変数にその時の経過時間の値が代入され、数式が逆ポーランド形式から数値に変換される。このように描画の前に、数式を一度逆ポーランド形式に変換し、必要な代入を済ませておくと、描画処理の計算時間が短縮される。

我々はアニメーション処理の中にシミュレーションによる計算を組み込むことになるが、現在の段階では、シミュレーションの種類により、組み込む位置は 2 か所に分かれる。図 2.2 のように質点の運動や重力運動等、時間に応じた物体の動きを計算する処理は描画処理の前で、静電誘導など計算時間がかかり、動きを要求されない場合は、プログラムの構造体入力とタイマースタートの間に入れる。

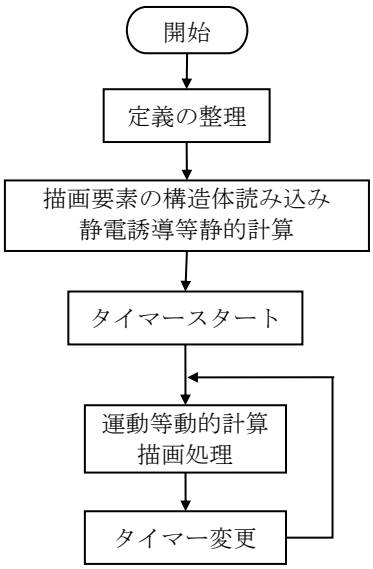

図 2.2　シミュレーションプログラムの流れ

静的計算の部分は、計算機の処理能力が上がると動的計算の部分に移る可能性もある。以後それぞれの例に基づきデータ構造や利用法を説明する。

質点の運動や電磁気現象などを扱うための、シミュレーション用のコマンドをまとめて示しておく。

シミュレーション分割

simdiv 整数

1 回の描画の間に何回の計算を行うかを指定する。後に述べるメニューの「シミュレーション分割」を変更するときに利用する。デフォルトは 10 回である。

質点

particle name, mass, charge, x,y,z, vx,vy,vz

planet も同じ

質点を定義する。以下の運動方程式等と合わせて使い、描画には ball や spher コマンドを利用する。x,y,z は初期位置、vx,vy,vz は初速度、name は質点の名前である。これは描画の際に、xname、vxname、massname 等の書式で、質点の x 座標、x 方向の速さ、質量等として参照可能である。(例：名前が p1 の場合、x 座標は xp1、x 方向の速さは vxp1、質量は massp1)

name$, xname$, vxname$ 等で loop の中での名前や x 座標、x 方向の速さ等を指定できる。(loop 変数 lp に対応して、$=0,1,2,…と変化する)

質量は ball や spher の半径の中でも利用可能である。

運動方程式

axname=式

m*a=f の m に相当する質量部分は入れないものとする。例えば、azp1=-9.8*zp1 のように記述する。通常の運動方程式と異なるので注意を要する。方程式内で x 方向の加速度、速さ、位置は axp1, vxp1, xp1 のように指定する。

ポテンシャル

potential=式

全体で 1 つだけ設定可能である。運動方程式と共存可能である。これにより摩擦などを表現することができる。

束縛条件(複数可能)

bind 条件式=0

複数個の条件式を並べて定義することが可能である。運動方程式と共存可能である。

重力計算

gravity

これがあると particle (planet) の定義のみで運動方程式は不要になる。メニュー上の「衝突重力」は太陽と地球との現在の重力加速度の倍数で設定する。これを上回る重力加速度が生じた場合、2 つの particle (planet) は結合（質量、運動量は保存する）する。これにより惑星形成などのシミュレーションができる。

描画の中央指定

center 名前

重力シミュレーションで中心に描く星を名前で指定する。

電磁力学計算

emmode

これにより、荷電粒子間の相互作用、荷電粒子への静止帯電導体の作用、荷電粒子への電流の作る磁場の作用を組み込んだ荷電粒子の運動のシミュレーションを行う。帯電導体や電流は荷電粒子からの影響を受けないものとする。運動方程式は不要である。

帯電導体（静電誘導）

conductor (x(u,v), y(u,v), z(u,v)), name, charge, [colormode, div1, div2, comb1,comb2,…]

3 次元パラメータ関数の要領で形状を指定する。(0<=u,v<=1) charge は導体全体の電荷である。

colormode は電荷分布の表示形式で、0：導体ごとの標準化電荷密度、1：全導体の標準化電荷密度、2：電荷密度 0 を黒とした赤青濃淡表示、3：電荷密度正と負の赤青表示、として指定する。デフォルトは 0 である。comb1, comb2, … は導線接続を行う導体の name を指定する。
静電誘導の問題は最初の計算に時間がかかるので、静的な処理となる。そのため、影響を及ぼす荷電粒子の位置を変えたり、導体の形状を変える時間ごとのシミュレーションはできない。

電気力線

eline (x,y,z), color, [mode, div]

(x,y,z) は電気力線を描くための起点、mode は 0：細線、1：太線である。
荷電粒子や導体の間に働く電場の電気力線を描く。導体がある場合、時間的に変化するシミュレーションはできない。

電流 [0≦u≦1]

ecurrent (x(u), y(u), z(u)), current, color [mode, div]
電流の方向は u=0→u-1, curret(A), mode 0:細線 1:太線

磁力線

mline (x,y,z), color, [mode, div]

(x,y,z) は磁力線を描くための起点、mode は 0：細線、1：太線である。

メニュー［分析－教育・科学他－物理シミュレーション－幾何シミュレーション］を選択すると図 2.3 のような実行メニューが表示される。

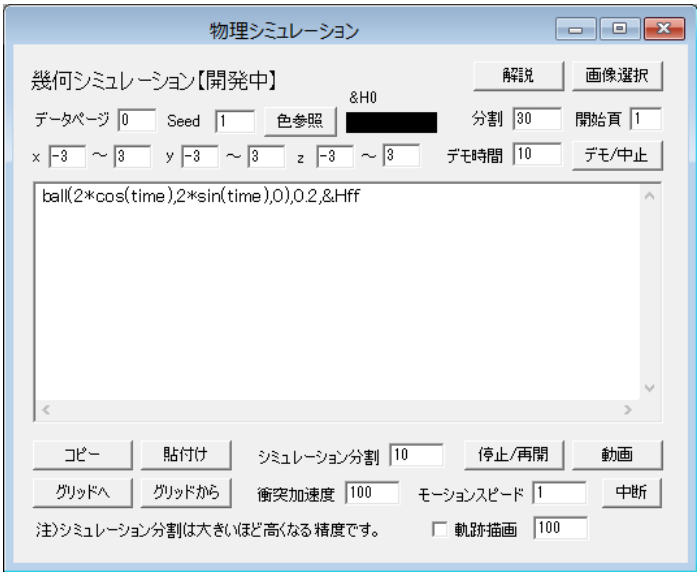

図 2.3　幾何シミュレーション実行メニュー

幾何シミュレーションは今後も開発が続くので【開発中】となっているが、論文では変更された機能や追加された機能をまとめて段階的に紹介する。このプログラムの利用法は、幾何アニメーションと類似しているが、追加機能も設けている。例えば「シミュレーション分割」テキストボックスでは 1 回の描画の間に何回の質点移動の計算を実行するかを与え、「衝突加速度」テキストボックスでは、

重力シミュレーションや電荷運動シミュレーションの際の惑星や粒子間で結合が起きる加速度を指定する。この衝突加速度は太陽重力による地球の加速度の倍数で指定する。但し、荷電粒子の運動では異なる電荷間でのみ結合が起きる。この結合の際には質量と運動量に加えて電荷も保存する。

我々のプログラムは、シミュレーションを実行し、その過程をアニメーションとして描画するものであるが、これだと粒子の運動の軌道は実行中にしかわからない、しかし、実効メニューの「軌道描画」チェックボックスにチェックを入れ、残したい粒子の個数（描画全体での個数）を右のテキストボックスに入力しておくと、粒子の軌道が表示される。以後、これらのコマンドを使った例を示し、利用法を説明するが、動きが分かりにくい場合は軌道描画を使って結果を表示する場合もある。

３．シミュレーションの例

我々は幾何アニメーションと質点系の運動、惑星運動、静電誘導、電荷と電気力線、電流と磁力線、電磁場の中の荷電粒子の運動などのシミュレーションを組み合わせた。これらは静電誘導を除いてこれまで物理シミュレーションとして単独で作成してきたものである[2][3]。そのため理論はすでに説明済みであるので省略し、ここではその利用法だけ説明する。但し、静電誘導については初めて作成されたものであるので、理論も含めて解説する。

3.1 質点系の運動

質点系の運動は、運動方程式、ポテンシャルエネルギーの形式で記述することができる。また、運動における束縛条件も付けることが可能である。この節ではこれらを使った簡単な例を示す。

例１　シミュレーションとアニメーション（２つのバネ）

```
1.  angle 0
2.  axis
3.  define l1=((x1+1.5)^2+y1^2+(3-z1)^2)^0.5
4.  potential =9.8*z1+4*(l1-2)^2
5.  particle 1,1,0 ,0,0,1, 0,1,0
6.  connect (-1.5,0,3)-(x1,y1,z1+0.2),&H800000,5
7.  ball (x1,y1,z1),0.4,&Hff
8.  define z2=2*sin(2*time)-1
9.  connect (1.5,0,3)-(1.5,0,z2+0.2),&H800000,5
10. ball (1.5,0,z2),0.4,&Hff00
11. print (0,-3.5,-3.5),"シミュレーションだからこんなことも",&Hff0000,20
```

このプログラムは、左側がシミュレーション、右側が単振動を数式で表現したアニメーションである。シミュレーションは縦方向の振動だけでなく、すべての方向の運動が可能である。1 行目は、最初に図が表示されたときの視点の水平から測った角度である。この場合、0 度であり、水平から見て

いることになる。2 行目は座標軸を描くように指定している。3 行目ではバネの長さを定義している。4 行目は重力とバネのポテンシャルエネルギーで、質量は後に示すように 1 である。ポテンシャルを記述した場合、自動的に運動方程式にポテンシャルに基づく力が追加される。この場合は、他の力が働かないので、運動方程式は省略できる。5 行目はバネの端に吊るされた質点の名前(1)、質量(1)、電荷(0)、3 つの初期座標（0,0,1)、3 つの初速度（0,1,0）を表す。ここまでがシミュレーションの記述である。6 行目は左側のバネの記述で、mode=5 の細いバネを表している。ボールの中心の座標は(x1,y1,z1）で、x に質点の名前 1 を付けて、x1 としている。7 行目はボールの描画である。8 行目は、右側のアニメーションのボールの運動の定義式である。9 行目は右側のバネ、10 行目は右側の質点の描画である。11 行目はプログラムについてのコメントの表示である。これらはすべて幾何アニメーションの命令である。実行結果を図 3.1 に示す。

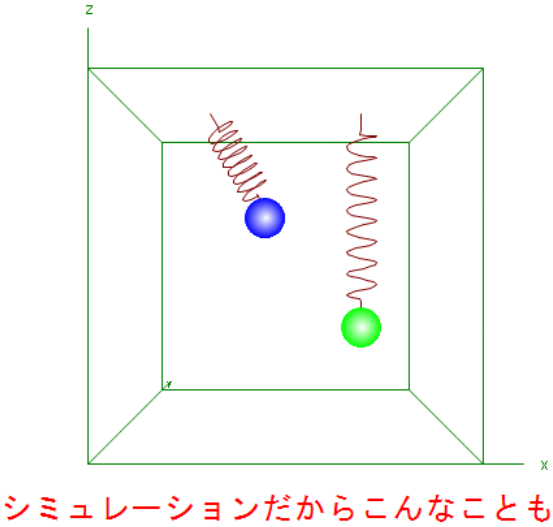

図 3.1　シミュレーションとアニメーション

例２　ポテンシャルと運動方程式の共存

```
1.   angle 0
2.   axis
3.   define l1=(x1^2+y1^2+(3-z1)^2)^0.5
4.   potential =9.8*z1+4*(l1-2)^2
5.   particle 1,1,0 ,1.5,0,1,  0,1,0
6.   ax1=-0.3*vx1
7.   ay1=-0.3*vy1
```

```
8.   az1=-0.3*vz1
9.   connect (0,0,3)-(x1,y1,z1+0.2),&H800000,5
10.  ball (x1,y1,z1),0.4,&Hff
11.  print (0,-3.5,-3.5),"抵抗による運動の減衰",&Hff0000,20
12.  print (0,-3.5,-4),"ポテンシャルと運動方程式の共存",&Hff0000,15
```

これはポテンシャルと運動方程式が共存する摩擦抵抗による運動の減衰の例である。この場合、質点に働く力はポテンシャルだけで表せないので、運動方程式の直接の記述が必要である。4 目行がポテンシャル、6-8 行目が運動方程式であるが、運動方程式の中にはポテンシャルによる力が自動的に追加されるので、書き込まない。また質量についても記述しない。11-12 行目はコメントの表示である。実行結果は図 3.2 に示されるが、一瞬を切り取ると通常のバネ振り子である。

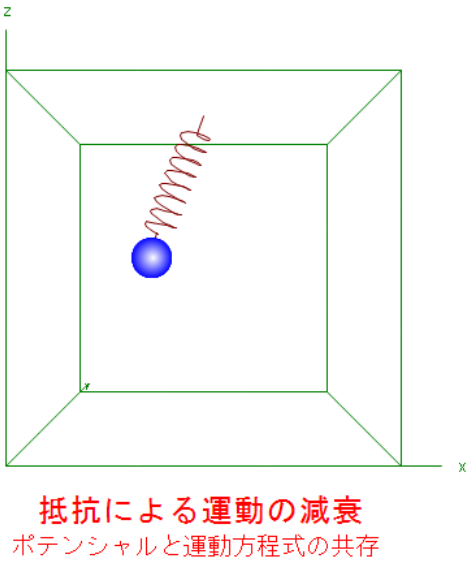

図 3.2　ポテンシャルと運動方程式の共存

例３　束縛問題１（振り子）

```
1.   angle 0
2.   axis
3.   potential=9.8*za
4.   bind xa^2+ya^2+za^2-9=0
5.   particle a,1,0, 3,0,0, 0,0,0
6.   ball (xa,ya,za),0.3,&Hff
7.   connect (0,0,0)-(0.95*xa,0.95*y,0.95*za),&H800000,2
8.   print (0,-3.5,-3.5),"未定定数法（束縛問題）",&Hff0000,20
```

これは糸の代わりに固い棒を使った振り子のシミュレーションである。初期条件の与え方によって通常の振り子のように運動したり、棒の一端を支点にして回転したりする。これを表現するために運動方程式に束縛条件を与え、未定常数法によって解を求める。4 行目は束縛条件を表す。即ち、棒の端から質点までの長さは 3 となる。しかし、7 行目のように棒の表示は、ボールの半径もあることから、5%短めになっている。また棒は mode=2 の太い実線で表される。束縛条件を有する問題は、束縛条件を満たすように初期条件を与えなければならない。実行結果を図 3.3 に示す。

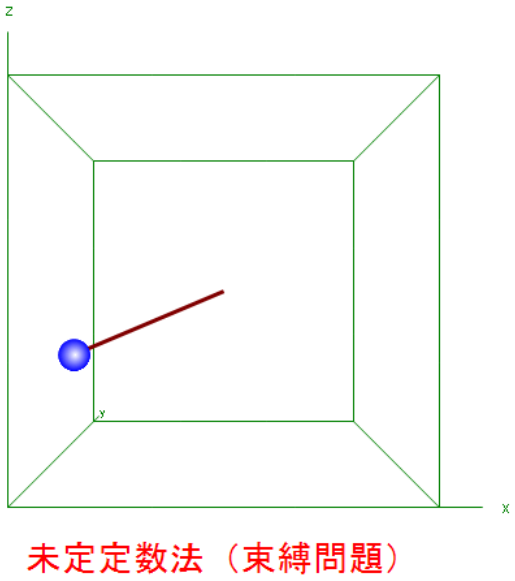

図 3.3　束縛問題 1

例 4　束縛問題 2（双曲形の床）

```
1.   range -2,2
2.   bind z1+2/(x1^2+y1^2)^0.5=0
3.   func z=-2/(x^2+y^2)^0.5-0.2,&Hffff,,,,,,0.2
4.   potential=9.8*z1
5.   particle 1,1,0, 2.2,0,-2/(2.2^2)^0.5, 0,2.3,0
6.   ball (x1,y1,z1),0.2,&H8000
7.   print (0,-2,-2.5),"束縛運動（床）",&Hff0000,30
```

これは束縛条件が双曲形の平面上となっている運動の例である。2 行目が束縛条件、3 行目はその床を表しているが、ボールの大きさを考えて、束縛条件の位置より 0.2 下に下げて表示している。4

行目はポテンシャルを表している。5 行目はボールの名前、質量、初期値を与える。実行結果を図 3.4 に与える。これは軌道を表示するモードで表したものである。

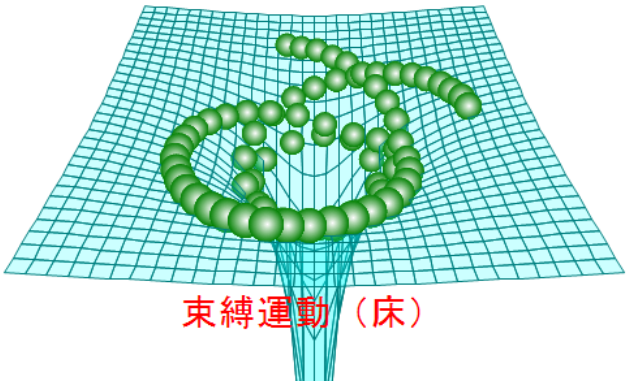

図 3.4　束縛問題２

3.2 重力による運動

重力によるシミュレーションでは、惑星程度以上の質量の物体を対象とし、プログラムでは「gravity」コマンドを利用する。これを用いると運動方程式は不要になり、惑星の質量、初期位置、初期速度だけを指定することでシミュレーションが実行できる。重力の強さが弱いため、地球の質量=1、地球の軌道半径=1、重力定数×太陽質量=1 の単位系を用いる。そのため、1 年は 2π秒になる。また、他の惑星による 1 つの惑星の加速度の大きさが、メニューで指定された「衝突加速度」を超えると 2 つの惑星は衝突し、結合するように設定されている。実際は、結合や飛散、様々であろうが、惑星形成過程などのモデルを作るために簡単化している。衝突加速度は太陽からの重力で地球が受ける加速度の倍数で指定されており、デフォルトでは地球の軌道半径が現在の 1/10 になるところの加速度に設定されている。この距離は太陽の半径に相当する。衝突した場合の処理は、2 つの星の 1 つを重心の位置に、質量と運動量を保存する形で残し、他方を十分離れた距離に質量を 0 にして置く。以後この質量 0 の星は計算に加えない。計算には標準的なルンゲークッタ法を用いているが、現在のパソコンでは 200 個程度の惑星の運動はリアルタイムに表現できる。

例５　５連星

```
1.  black
2.  gravity
3.  range -1,1
4.  axis
5.  paint 2
6.  planet p1,10000,1, 0.5,-0.5,0, 0,0.5,0
7.  planet p2,10000,1, 0.5,0.5,0, -0.5,0,0
```

```
8.   planet p3,10000,1, -0.5,0.5,0, 0,-0.5,0
9.   planet p4,10000,1, -0.5,-0.5,0, 0.5,0,0
10.  planet p5,100000,1, 0.01,0,0.5, 0,0,0
11.  ball (xp1,yp1,zp1),0.05,&Hff
12.  ball (xp2,yp2,zp2),0.05,&Hff00
13.  ball (xp3,yp3,zp3),0.05,&Hffff
14.  ball (xp4,yp4,zp4),0.05,&Hffff00
15.  sphere (xp5,yp5,zp5),0.1,image1,,,30*time
16.  define ti=time/(6*pi)
17.  print (0,-1.2,0),"<ti>年",&Hffffff,20,"0.0"
18.  print (0,-1.2,-1), "重力シミュレーション",&Hffff00,20
```

この問題は、きれいに同期の取れた同質量の 4 連星の中央に質量の大きな星が近づく場合のシミュレーションである。1 行目では宇宙の雰囲気を作るために背景を黒にしている。2 行目はこれが重力シミュレーションであることを示している。3 行目は描画の大まかな範囲を 3 つの軸とも-1 から 1 まで（太陽までの距離の 2 倍）としている。4 行目では奥行きを感じやすくするため、座標軸を描く。この座標軸の位置が上の描画の範囲である。6-10 行目でそれぞれの星の初期条件を与える。4 つの連星は太陽質量の約 1/3、中央の星は太陽質量の約 3 倍である。11-15 行目でそれらを描画するが、p5 の星については、sphere コマンドで、実行メニューの「画像選択」で読み込んだ画像、image1 を貼り付ける。その際、分割線を消すために 5 行目に「paint 2」命令を入れている。17 行目は、16 行目で計算した実年数を形式 "0.0" により小数点以下 1 桁で表示している。実行結果を図 3.5 に示す。

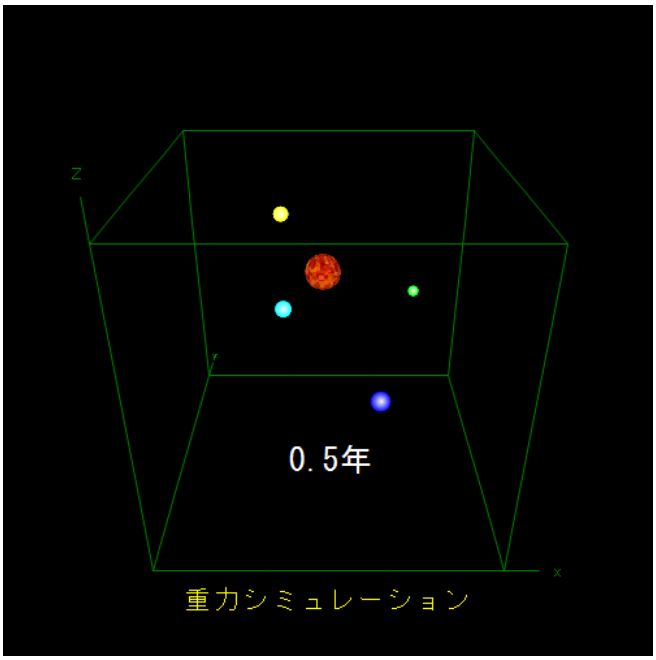

図 3.5　5 連星

例６　恒星の誕生

```
1.   black
2.   gravity
3.   axis
4.   range -5,5
5.   set t1=2*pi*rnd
6.   set r1=7*rnd
7.   define x=r1*cos(t1)
8.   define y=r1*sin(t1)
9.   define z=exp(-x^2-y^2)
10.  define vx=-y/10
11.  define vy=x/10
12.  define vz=0
13.  set col=rainbow(rnd)
14.  loop 50
15.  particle p$,33300,0, x,y,z,vx,vy,vz
16.  define r=(massp$/100000)^0.333*0.2
17.  ball (xp$,yp$,zp$),r,col
18.  endloop
19.  print (0,-6,-5),"太陽距離×10, 太陽質量÷10",&Hffffff,20
```

これは円盤状にゆっくり回転している 80 個の星の集団が重力で引き合って巨大な星ができて行く過程をシミュレーションするものである。時間を短くするために、すべての星の質量を太陽質量にしている。4 行目より、座標軸の範囲は地球から太陽までの 10 倍に設定している。5 行目は極座標におけるそれぞれの星の初期角度、6 行目は動径方向の距離を乱数で与えている。7,8 行目はそれらの初期極座標を直交座標に置き換えている。また、9 行目で z 座標を定義している。10-12 行目は初期位置に応じた星の速度を与えている。これは回転円盤を想定している。13 行目でそれぞれの星の色を設定している。14 行目で星の個数を loop で設定し、15 行目での名前、質量、初期座標、初期速度を設定している。星の名前は、p$としているが、$は loop 変数 lp に応じて数値を代入して行くもので、星の名前は、p0, p1, p2, … ,p49 となる。星が結合して質量を増すにつれて半径を大きく表示するために、15 行目で質量による星の半径を定義している。星の質量は massp$として、mass+ p$の意味で特定する。同様に、17 行目で星を描画するが、その位置は xp$として、x+p$の意味である。図 3.6 にシミュレーションを始めたばかりの状態と時間が経過した状態を並べて示す。星が結合して大きくなっている様子が分かる。

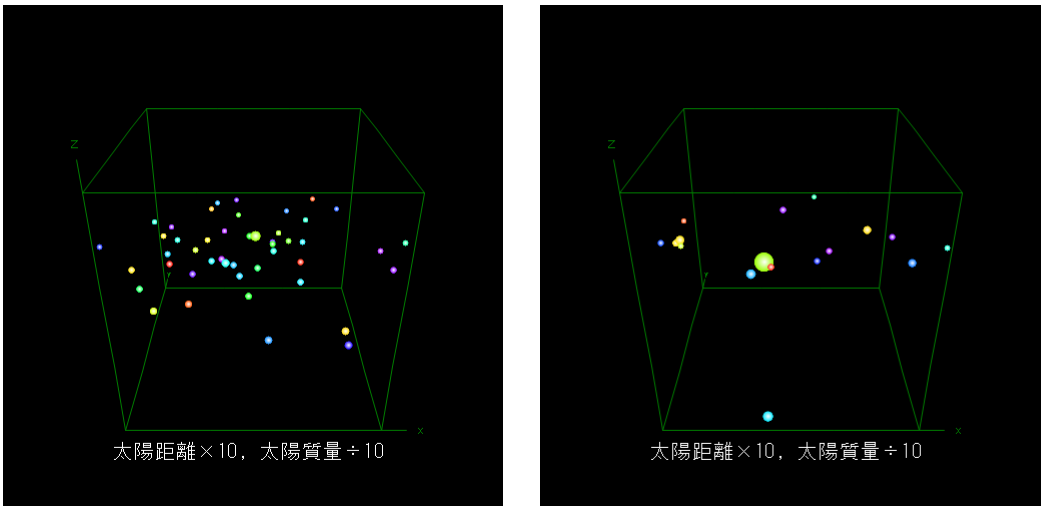

図 3.6　恒星の誕生過程

例７　惑星形成過程

```
1.  black
2.  gravity
3.  axis
4.  center s
5.  define rs=(masss/60000)^0.333*0.1
6.  planet s,3330000,0, 0,0,0, 0,0,0
7.  ball (xs,ys,zs),rs,&Hff0000
8.  set r1=4*abs(nrnd)+1
9.  set a=2*pi*rnd
10. set c=rainbow(rnd)
11. define x=r1*cos(a)
12. define y=r1*sin(a)
13. define z=0
14. define vx=-(10/r1)^0.5*sin(a)
15. define vy=(10/r1)^0.5*cos(a)
16. define vz=0
17. loop 200
18. planet p$,10000,0, x,y,z, vx,vy,vz
19. define rp=(massp$/60000)^0.333*0.1
20. ball (xp$,yp$,zp$), rp, c
21. endloop
```

これは、中心に太陽の 10 倍の質量の星があり、その周りを太陽の約 1/30 の質量の 200 個の星が回っているシミュレーションである。時間の関係からこの程度の質量でシミュレーションをしている。4 行目はどの星を中心に描くかを与えている。ここでは s という名前の星が中心である。5 行目ではこの星の半径を与え、6 行目では質量、初期位置、初期速度を指定している。7 行目でこの星を描い

ている。8 行目と 9 行目は周りの星の極座標の動径距離と角度、10 行目は色を与えている。11-13 行目でこれを直交座標に変換し、14-16 で初期速度を定義している。17 行目で星を 200 個使うことを指定し、18 行目で、質量、初期位置、初期速度を指定している。19 行目で質量に対する半径を指定し、20 行目でこれらのことを使って描画している。図 3.7 にシミュレーションを始めたばかりの状態と時間が経過した状態を並べて示す。

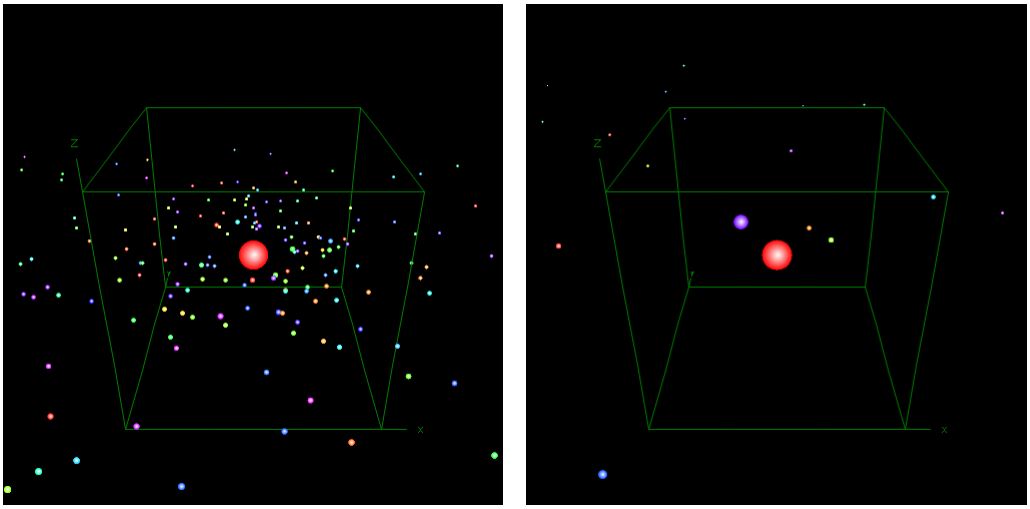

図 3.7　惑星形成過程

3.3 静電誘導

静電誘導については、理論的背景を説明しておく必要がある。我々は 3 次元パラメータ曲面を導体の表面とみなし、電荷を与えた場合や荷電粒子を近づけた場合の表面電荷密度の分布を求めてみる。

点電荷 m 個と導体 n 個が存在する系を考える。導体 a は表面を細かく n_a 個のセルに分割し、そのセルをさらに細かく分割するモデルを考える。今、点電荷 p の位置を $\boldsymbol{r}_p$、電荷を q_p、導体 a 上の i 番目の分割のセルの代表点の位置を $\boldsymbol{r}_{ai}$、セルの面積を ds_{ai}、1 つのセル内の電荷密度を一定と仮定して σ_{ai}、セルの電荷を $q_{ai} = \sigma_{ai} ds_{ai}$ とする。またその 1 つのセルをさらに細かく n_{ai} 個に分割して、その代表点の位置を $\boldsymbol{r}_{aik}$、その面積を ds_{aik} とする。但し、セルを細かく分割した代表点とセルの代表点は重ならないものとする。導体 a 上のセルの代表点の位置 $\boldsymbol{r}_{ai}$ での電位 V_{ai} は近似的に以下で与えられる。

$$V_{ai} = \frac{1}{4\pi\varepsilon_0} \sum_{b=1}^{n} \sum_{j=1}^{n_b} \sum_{k=1}^{n_{bj}} \frac{\sigma_{bj} ds_{bjk}}{\left| \boldsymbol{r}_{ai} - \boldsymbol{r}_{bjk} \right|} + \frac{1}{4\pi\varepsilon_0} \sum_{p=1}^{m} \frac{q_p}{\left| \boldsymbol{r}_{ai} - \boldsymbol{r}_p \right|}$$

ここで、

$$A_{aibj} \equiv \frac{1}{4\pi\varepsilon_0 ds_{bj}} \sum_{k=1}^{n_{bj}} \frac{ds_{bjk}}{\left|\boldsymbol{r}_{ai} - \boldsymbol{r}_{bjk}\right|}, \quad b_{ai} \equiv \frac{1}{4\pi\varepsilon_0} \sum_{p=1}^{m} \frac{q_p}{\left|\boldsymbol{r}_{ai} - \boldsymbol{r}_p\right|} \tag{1}$$

と定義すると以下となる。

$$V_{ai} = \sum_{b=1}^{n}\sum_{j=1}^{n_b} A_{aibj}\sigma_{bj} ds_{bj} + b_{ai} = \sum_{b=1}^{n}\sum_{j=1}^{n_b} A_{aibj} q_{bj} + b_{ai}$$

ここで導体 a 上の電位はすべて同じであることから、$V_{ai} = V_a$ と書くことができ、以下の関係式を得る。

$$\sum_{b=1}^{n}\sum_{j=1}^{n_b} A_{aibj} q_{bj} = V_a - b_{ai}$$

この中で既知の量は、A_{aibj}，ds_{bj}，b_{ai} である。ここで、ai と bj をそれぞれ１つの添え字のように扱うと上式は、q_{bj} についての連立１次方程式となり、その解は A_{aibj} の逆行列を A_{aibj}^{-1} とすると以下のようになる。

$$q_{ai} = \sum_{b=1}^{n}\sum_{j=1}^{n_b} A_{aibj}^{-1}\left(V_b - b_{bj}\right) = \sum_{b=1}^{n}\sum_{j=1}^{n_b} A_{aibj}^{-1} V_b - \sum_{b=1}^{n}\sum_{j=1}^{n_b} A_{aibj}^{-1} b_{bj} \tag{2}$$

さて、ここで導体 a の全電荷を Q_a とし、さらに導体上での電位はすべて同じ V_a であるので以下の式を得る。

$$\begin{aligned} Q_a &= \sum_{i=1}^{n_a} q_{ai} = \sum_{i=1}^{n_a}\sum_{b=1}^{n}\sum_{j=1}^{n_b} A_{aibj}^{-1} V_b - \sum_{i=1}^{n_a}\sum_{b=1}^{n}\sum_{j=1}^{n_b} A_{aibj}^{-1} b_{bj} \\ &= \sum_{b=1}^{n}\sum_{i=1}^{n_a}\sum_{j=1}^{n_b} A_{aibj}^{-1} V_b - \sum_{b=1}^{n}\sum_{i=1}^{n_a}\sum_{j=1}^{n_b} A_{aibj}^{-1} b_{bj} = \sum_{b=1}^{n} C_{ab} V_b - D_a \end{aligned}$$

ここに、

$$C_{ab} \equiv \sum_{i=1}^{n_a}\sum_{j=1}^{n_b} A_{aibj}^{-1}, \quad D_a \equiv \sum_{i=1}^{n_a}\sum_{b=1}^{n}\sum_{j=1}^{n_b} A_{aibj}^{-1} b_{bj}$$

即ち

$$\sum_{b=1}^{n} C_{ab} V_b = Q_a + D_a \tag{3}$$

この中でまだ計算できていない量は導体上の電位 V_b だけであり、これはこの連立１次方程式を解くことにより、C_{ab} の逆行列 C_{ab}^{-1} を用いて以下のように求めることができる。

$$\hat{V}_a = \sum_{b=1}^{n} C_{ab}^{-1}\left(Q_b + D_b\right)$$

この電位の表式を (2) 式に代入して、各セルの電荷 $\hat{q}_{ai}$ を求めることができる。

$$\hat{q}_{ai} = \sum_{b=1}^{n}\sum_{j=1}^{n_b} A_{aibj}^{-1}\hat{V}_b - \sum_{b=1}^{n}\sum_{j=1}^{n_b} A_{aibj}^{-1} b_{bj} \tag{4}$$

また電荷密度も $d\hat{\sigma}_{ai} = \hat{q}_{ai} / ds_{ai}$ のように求めることができる。

さて、一連の計算の中で時間がかかる部分を検討してみよう。1 つは (2) 式を求める際の逆行列の計算で、もう 1 つは (1) 式の A_{aibj} についての計算である。ここでは後者について以下の高速化を考える。即ち、ある距離 r_0 を考え、以下のような計算をする。

$$A_{aibj} = \frac{1}{4\pi\varepsilon_0 ds_{bj}} \sum_{k=1}^{n_{bj}} \frac{ds_{bjk}}{\left|\boldsymbol{r}_{ai} - \boldsymbol{r}_{bjk}\right|} \sim \begin{cases} \text{そのまま利用} & for\ \left|\boldsymbol{r}_{ai} - \boldsymbol{r}_{bj}\right| \le r_0 \\ \dfrac{1}{4\pi\varepsilon_0 \left|\boldsymbol{r}_{ai} - \boldsymbol{r}_{bj}\right|} & for\ \left|\boldsymbol{r}_{ai} - \boldsymbol{r}_{bj}\right| > r_0 \end{cases}$$

ここで n 個の導体中 r 番目の以降の導体が導線で繋がっている場合を考える。r 番目以降の導体の電位はすべて等しく V_R とし、電荷の総量を Q_R とすると、(3)式は以下のように書き換えられる。

$$\sum_{b=1}^{r-1} C_{ab}V_b + \sum_{b=r}^{n} C_{ab}V_R = Q_a + D_a \qquad a < r$$

$$\sum_{a=r}^{n}\sum_{b=1}^{r-1} C_{ab}V_b + \sum_{a=r}^{n}\sum_{b=r}^{n} C_{ab}V_R = Q_R + \sum_{a=r}^{n} D_a \qquad a = r$$

ここで、

$${}^tQ'_a = \begin{pmatrix} Q_1 & \cdots & Q_{r-1} & Q_R \end{pmatrix}, \quad {}^tD'_a = \begin{pmatrix} D_1 & \cdots & D_{r-1} & \sum_{k=r}^{n} D_k \end{pmatrix}$$

を用いると、以下の方程式となる。

$$\sum_{b=1}^{r} C'_{ab}V'_b = Q'_a + D'_a$$

この解を用いて、

$$\begin{pmatrix} \hat{V}_1 & \cdots & \hat{V}_{r-1} & \hat{V}_r & \cdots & \hat{V}_n \end{pmatrix} = \begin{pmatrix} \hat{V}'_1 & \cdots & \hat{V}'_{r-1} & \hat{V}'_R & \cdots & \hat{V}'_R \end{pmatrix}$$

として、(4) 式に代入し各セルの電荷を求める。

ここでは単一の導体を帯電させる場合、単一の導体の近くに電荷を近づけた場合、複数の導体を繋ぎ電荷を近づけた場合の静電誘導、静電誘導された導体から影響を受けた荷電粒子の運動の例について説明する。静電誘導は計算時間がかかるため、このプログラムではタイマーの開始前に計算が実行される。そのため、時間と共に導体の帯電状態が変化するような動的なシミュレーションは現時点では不可能である。しかし、計算が終わった後、微小な荷電粒子がその電場の中で運動するシミュレー

ションは可能である。

例８　帯電による静電誘導

```
1.   define x=2*sin(pi*u)*cos(2*pi*v)
2.   define y=2*sin(pi*u)*sin(2*pi*v)
3.   define z=5*cos(pi*u)
4.   conductor(x,y,z),s1,1,0
```

1-3 行目は導体の形状（楕円体）を指定するパラメータ関数を表す。4 行目は導体の定義で、名前を s1、全電荷を 1 クーロン、電荷密度の表示モードは導体ごとの標準化電荷密度である。これは各ブロックごとの電荷密度を標準化し（平均 0、分散 1)、値が 2 以上を赤、-2 以下を青、中間は虹色に表示するモードである。実行結果を図 3.8 に示す。

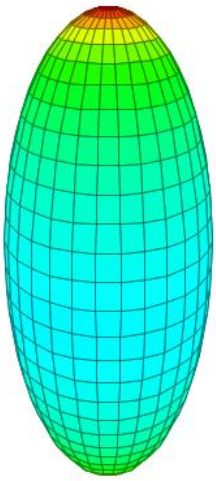

図 3.8　帯電による静電誘導

例９　電荷による静電誘導

```
1.   define x=2*cos(2*pi*u)
2.   define y=2*sin(2*pi*u)
3.   define z=6*v-3
4.   particle p,1,1,  0,-4,0,  0,0,0
5.   ball (xp,yp,zp),0.2,&Hff0000
6.   conductor(x,y,z),s1,0,3
```

これは導体の近くに荷電粒子がある場合の静電誘導のシミュレーションである。1-3 行目は導体の形状（円筒）を表すパラメータ関数である。4 行目は、名前 p、質量 1、電荷 1、位置 (0,-4,0)、速度

(0,0,0) の荷電粒子を表す。5 行目では荷電粒子を描画している。電荷が正であるので赤色で表している。6 行目は導体の定義で、名前は s1、全電荷 0、描画モードは電荷密度を正負で赤青 2 色で塗り分けるモードである。シミュレーション結果を図 3.9 に示す。

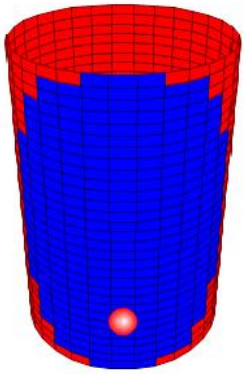

図 3.9　電荷による静電誘導

例 10　複数の導体と電荷による静電誘導

```
1.   define x1=2.5*sin(pi*u)*cos(2*pi*v)+3
2.   define y1=2.5*sin(pi*u)*sin(2*pi*v)
3.   define z1=2.5*cos(pi*u)
4.   define x2=1.5*sin(pi*u)*cos(2*pi*v)-3
5.   define y2=1.5*sin(pi*u)*sin(2*pi*v)
6.   define z2=1.5*cos(pi*u)+1
7.   define x3=1*sin(pi*u)*cos(2*pi*v)-2
8.   define y3=1*sin(pi*u)*sin(2*pi*v)
9.   define z3=1*cos(pi*u)-2
10.  conductor(x1,y1,z1),s1,0,1,10,20,s2
11.  conductor(x2,y2,z2),s2,0,1,10,20,s1
12.  conductor(x3,y3,z3),s3,1,1,10,20
13.  particle a,1,1,  0,0,2,0,0,0
14.  ball (xa,ya,za),0.15, &Hff0000
15.  define x=6*u-3
16.  define y=0
17.  define z=1-u
18.  paraml(x,y,z),&Hff00,2
```

これは 3 つの導体と 1 つの電荷の静電誘導の例である。但し 3 つの導体のうち 2 つは導線で繋いでいる。導体の形状はすべて球にしている。また、計算に時間がかかるため、分割数は少なめにしている。1-9 行目は 3 つの導体の形状を与えるパラメータ関数である。10-12 行目は導体の定義で、名前は s1,s2,s3、全電荷は 0,0,1 で、s1 と s2 の導体は導線で結合されている。導線で結合されているこ

とは 2 つの導体の電位が同じであることを意味する。13 行目は荷電粒子の定義、14 行目はその描画である。15-17 行目は繋いだ導線を表すパラメータ曲線を表し、18 行目はその描画である。実行結果を図 3.10 に示す。

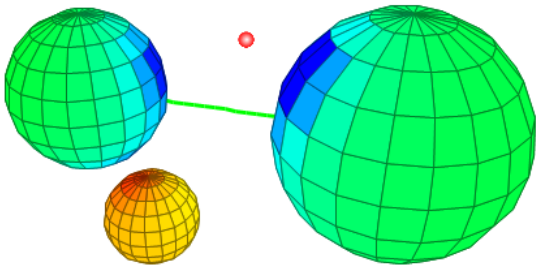

図 3.10　複数の導体と電荷の静電誘導

例 11　帯電導体と荷電粒子

```
1.   emmode
2.   define x=1.5*sin(pi*u)*cos(2*pi*v)
3.   define y=1.5*sin(pi*u)*sin(2*pi*v)
4.   define z=4*cos(pi*u)
5.   conductor(x,y,z),s1,1,2
6.   particle a,0.001,-10^(-11), 4,0,0, 0,3,2
7.   ball (xa,ya,za), 0.2, &Hff
```

これは帯電した導体の周りを微小な電荷を帯びた粒子が運動するシミュレーションである。静電誘導は動的な処理が計算時間の都合で困難であるため、荷電粒子が帯電導体に影響を与えることはないと仮定する。1 行目は、荷電粒子の運動のシミュレーションを行うことの宣言である。2-5 行目は例８でも述べた帯電した導体の静電誘導を計算させるプログラムである。ここで帯電導体の電気量は全体で-10^{-11}coulomb、静電誘導の表示モードは、電荷密度 0 を黒とした赤青濃淡表示である。6 行目は荷電粒子の定義で、質量は 0.001kg(1g)、電荷は -10^{-11} coulomb、初期座標は(-4,0,0)、初期速度は(0,3,2)である。7 行目は荷電粒子を青色で表示するコマンドである。荷電粒子は帯電導体の周りを運動する。図 3.11 に実行結果を示す。これは粒子の「軌道描画」のモードで描画したものである。

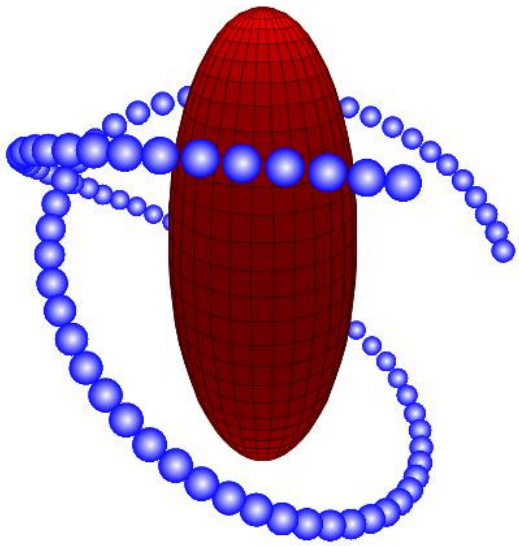

図 3.11　帯電導体と荷電粒子

3.4 電気力線

電気力線は空間の 1 点を指定し、そこから 2 つの方向に電気力線を繋いで行く方法で描画される。電荷の位置が時間的に変化する場合、電気力線は指定された点を起点として変化して行く。この起点自体も時間的に変化させることができる。ここでは、荷電粒子による電気力線と荷電粒子と導体の静電誘導による電気力線のプログラムを示す。

例 12　運動する電荷による電気力線

```
define x1=1*sin(3*time)+2
define x2=-1*sin(2*time)-2
define y3=2*cos(1.5*time)
define z3=2*sin(1.5*time)
particle p1,1,-0.5, x1,0,0, 0,0,0
particle p2,1,-0.5, x2,0,0, 0,0,0
particle p3,1,1, 0,y3,z3, 0,0,0
ball (x1,0,0),0.15,&Hff
ball(x2,0,0),0.15,&Hff
ball(0,y3,z3),0.2,&Hff0000
set x=6*rnd-3
set y=6*rnd-3
set z=6*rnd-3
loop 30
eline (x,y,z),&Hff00ff,1
```

```
16.  endloop
```

これは 3 つの運動する電荷によって電気力線が変化する様子を表したシミュレーションである。正の電荷には赤色、負の電荷には青色を用いている。5-7 行目で荷電粒子を定義している。粒子 p1, p2 は負、p3 は正の電荷を持っている。また、1-4 行目で定義する運動は p1, p2 が x 軸方向の単振動、p3 が y-z 平面内の円運動である。8-10 行目は粒子の描画である。11-13 行目で電気力線の起点を 14 行目のループにより 30 点与え、15 行目でそれを描画している。電気力線の起点の定義はループの中に書いてもよい。実行結果を図 3.12 に示す。

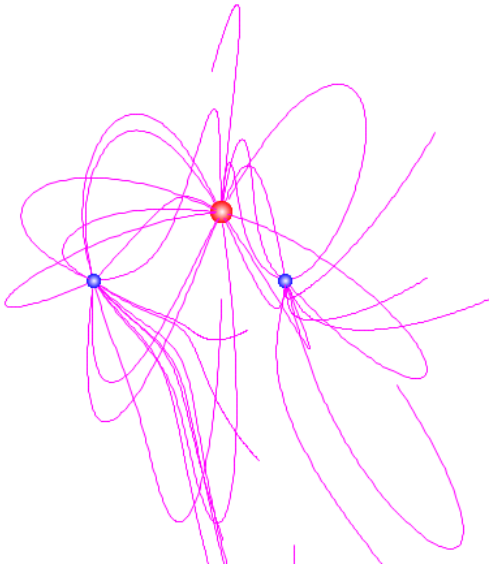

図 3.12　運動する電荷による電気力線

例 13 電荷と導体の作る電気力線

```
1.   angle 0
2.   define x=10*u-5
3.   define y=0
4.   define z=10*v-5
5.   particle p1,1,1,  0,-4,0,  0,0,0
6.   ball (xp1,yp1,zp1),0.2,&Hff0000
7.   conductor(x,y,z),s1,0,1
8.   set xe=6*rnd-3
9.   set ze=6*rnd-3
10.  loop 25
11.  eline (xe,-2,ze),&Hff00ff
12.  endloop
```

これは荷電粒子と導体の板の間の電気力線を描くプログラムである。荷電粒子よって導体に静電誘導が起きており、これらが電場を作っている。1 行目は視点を z=0 の位置に設定するコマンドである。2-4 行目は y 軸に垂直な平面を表すパラメータ関数である。5 行目は荷電粒子の定義で、6 行目はその描画である。7 行目は 2-4 行目のパラメータ関数の形状の導体で、全電荷は 0、全導体の標準化電荷密度を色で設定するモードになっている。この場合、導体が 1 つしかないので、単独の導体の標準化電荷密度も同じである。8,9 行目では 10 行目からの loop の中に利用する電気力線の起点を与える乱数を設定している。実行結果を図 3.13 に示す。

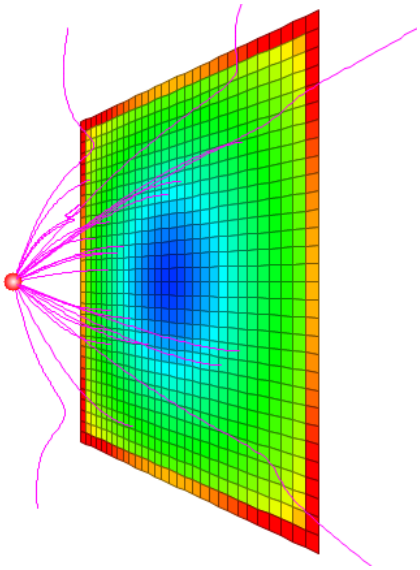

図 3.13　電荷と導体の作る電気力線

3.5 磁力線と荷電粒子の運動

ここでは電流が作る磁場の磁力線表示と電磁場中の荷電粒子の運動を求めるプログラムの例を示す。電流の形状はパラメータ曲線の形式で指定し、磁力線は電気力線と同じように起点を決めて描画する。荷電粒子の運動は、他の電荷との相互作用、静電誘導された導体の電場、電流が作る磁場からの影響で決まる。

例 14　磁力線と荷電粒子の運動

```
1.    playback 15
2.    emmode
3.    define x1=2*cos(2*pi*u+pi/2)
4.    define y1=2*sin(2*pi*u+pi/2)
```

```
5.   define z1=0.3*lp+0.5
6.   loop 3
7.   ecurrent (x1,y1,z1),10^6,&Hff,,30
8.   endloop
9.   define x2=2*cos(2*pi*u+pi/2)
10.  define y2=2*sin(2*pi*u+pi/2)
11.  define z2=-0.3*lp-0.5
12.  loop 3
13.  ecurrent (x2,y2,z2),10^6,&Hff,,30
14.  endloop
15.  define x3=1*cos(2*pi*lp/10)
16.  define y3=1*sin(2*pi*lp/10)
17.  define z3=0
18.  loop 10
19.  mline (x3,y3,z3),&H8000
20.  endloop
21.  particle a,0.001,0.0001, 1,0,0, 0,-2,0
22.  ball (xa,ya,za),0.2,&Hff0000
```

並行で一様な磁場中ではその磁場と垂直の平面内で荷電粒子は円運動する。これはコイルの中に近似的に一様な磁場を作り、そこで荷電粒子を運動させるプログラムである。もちろん正確には一様性が成り立つわけではないので、時間が立つと荷電粒子はコイルの外に出てしまうが、それまでの間ほぼ円運動をする。1 行目はシミュレーションの実行時間を 15 秒とし、それを繰り返す。2 行目は電磁力学を計算するモードである。3-5 行目は円形を表すパラメータ関数である。特に 5 行目の lp は繰り返し loop の中で使われると、0,1,2 のように値が代入される。6-8 行目は 3 回の繰り返しにより、3 つの上側の円形電流を表している。電流の強さは、10^{10} アンペアである。9-11 行目の円形を表すパラメータ関数で、12-14 行目は下側の 3 つの円形電流を表す。15-17 行目は磁力線の起点を与える。起点の数は 18 行目により 10 本であることが分かる。19 行目で磁力線を描き、21 行目で荷電粒子の定義、22 行目でその描画である。このモデルでは重力は考えていない。実行結果を図 3.14 に示す。

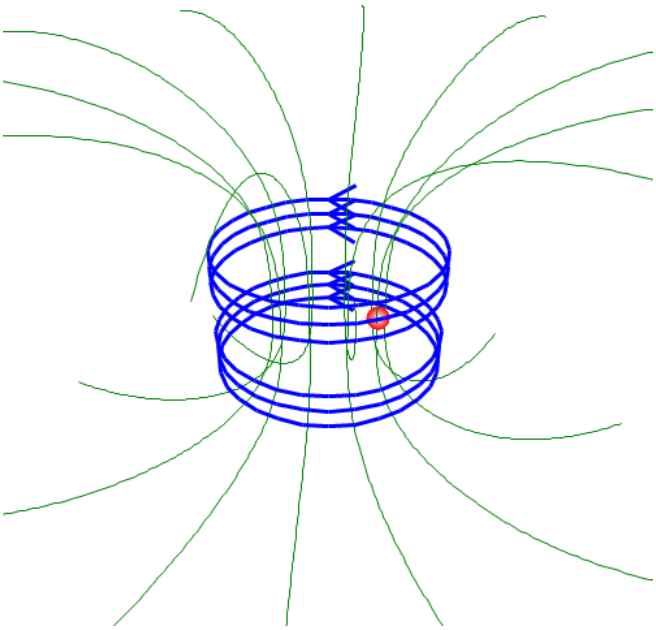

図 3.14　磁力線と荷電粒子の運動

例１５　磁場による荷電粒子の散乱

```
playback 2
black
emmode
define x=1.5*cos(2*pi*u+pi/2)
define y=1.5*sin(2*pi*u+pi/2)
define z=0.5*lp-0.5
loop 3
ecurrent (x,y,z),10^10,&Hff,,30
endloop
define x1=1*cos(2*pi*lp/10)
define y1=1*sin(2*pi*lp/10)
define z1=0
loop 10
mline (x1,y1,z1),&Hff00ff
endloop
set x2=4+4*rnd
set y2=4*rnd-2
set z2=4*rnd-2+3
set a=2*int(2*rnd)
sphere (0,0,0),2,image1,,,30*time
loop 10
particle a$,0.00005,(a-1)*10^(-9), x2,y2,z2, -5,0,-2
ball (xa$, ya$, za$), 0.1, rainbow(a/3)
```

24.　endloop

地球が作る磁場は放射線を散乱させ、地球への侵入を妨げている。これをモデルとして示したものがこのプログラムである。但し、モデルのスケール、磁場の大きさ、粒子の質量、速度等、実際とは異なり、見やすいように調整している。地球磁場の雰囲気を出すために、コイルを半透明な球殻で覆っている。プログラムの大部分が例 14 と同じであるが、21-24 行目に見るように、荷電粒子は 10 個入射する。その電荷は 10^{-9} か -10^{-9} の乱数で与えられている。19 行目の a は 0≦rnd＜1 より、0 か 2 の乱数である。それを 22 行で(a-1)*10^(-9) のようにして電荷に使っている。また 23 行目でもその値を rainbow(a/3) として色に使っている。rainbow() 関数は 0 から 1 の間の値を赤から紫の色番号に変換する関数で、ここでは、rainbow(0) と rainbow(2/3) の色を利用している。図 3.15 にその結果を示す。これは粒子の「軌道描画」のモードを用い、「モーションスピード」を 0.5 として、半分にし、通常の 2 倍の密度で描いたものである。

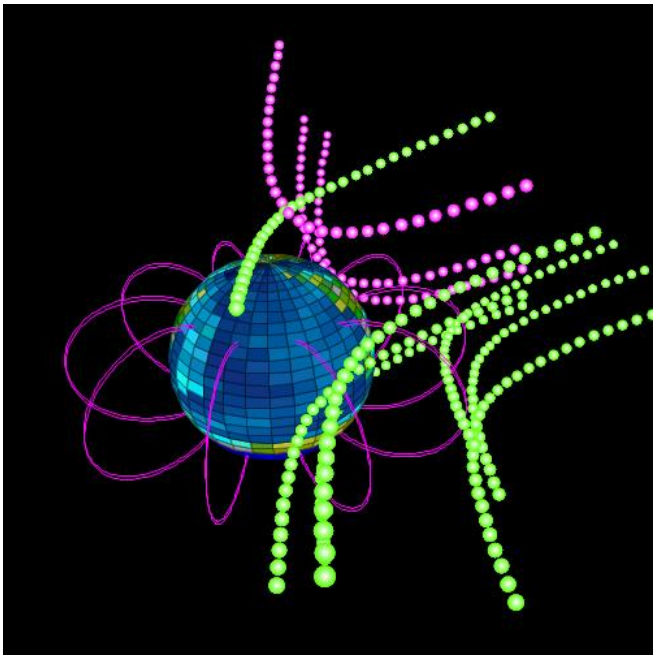

図 3.15　磁場による荷電粒子の散乱

４．おわりに

我々は、物理シミュレーションと幾何アニメーションを融合させ、幾何シミュレーションというプログラムを作成した。これはコマンドにより、シミュレーションを実行するだけでなく、その実行環境を含めて結果を表示する形式のシミュレーションプログラムである。最初の課題は、幾何アニメーションのプログラムの中に統一的に力学と電磁気学のシミュレーションの要素を取り込み、結果をリ

アルタイムに表示させることであった。我々のプログラムのデータ形式とアルゴリズムはこれらを解決し、シミュレーションを可能にした。但し、静電誘導のような元々計算時間のかかる現象では、動的なシミュレーションは困難である。シミュレーションを始める前に電場を計算して、それを一定として後の計算を続けることは可能である。

これらのシミュレーションのスピードは、パソコンの処理能力に依存する。そのため、現在は動きが滑らかでないものでも、将来は遅滞なく動く可能性を持っている。我々が開発を進める時間経過に従って、シミュレーションが可能になる現象が増えて行くことにもなる。

これからもっと多くの物理現象を対象に考えて行くが、波動・熱、流体、量子現象は現在考えている大きな課題である。また、剛体系の問題も重要であるが、これは物理エンジンで深く考えられているため、もはや追いつける問題ではないだろう。プログラムの仕様用途を考えて、現象を選んでシミュレーションに加えて行くことは重要である。

参考文献

[1] 社会システム分析のための統合化プログラム２２ ― 幾何アニメーション ―, 福井正康, 福山平成大学経営研究, 第 10 号, (2014) 79-104.

[2] College Analysis による物理シミュレーション１ ―質点系の運動・惑星シミュレーション―, 福井正康, 福山平成大学経営研究, 第 9 号, (2013) 53-79.

[3] College Analysis による物理シミュレーション２ ― 電荷と電場・電流と磁場 ―, 福井正康, 福山平成大学経営研究, 第 9 号, (2013) 81-105.

Physics Simulation in College Analysis 3

- Geometric Simulation -

Masayasu FUKUI

Department of Business Administration, Faculty of Business Administration,
Fukuyama Heisei University

Abstract

We have created a program of physical simulation and geometric animation using graphic capabilities of the integration program College Analysis. We will discuss the fusion of these two programs in this paper. Phenomena that we handle are movement of mass system, planetary motion due to gravity, electrostatic induction of the conductor, lines of electric force due to point charge and conductor charge, lines of magnetic force and motion of a point charge due to electric and magnetic fields.

Keywords

College Analysis, physics, simulation, animation

URL: http://www.heisei-u.ac.jp/ba/fukui/