卒業論文

Maya上での物理現象の視覚化に利用する粒子法スクリプト

関西学院大学理工学部
情報科学科 5691 川口拓哉

2010年 3月

指導教員　　西谷滋人　教授

概 要

初学者の物理学習において，物理現象を視覚化することで直感的な理解に繋がる．その様々な物理現象の理解は重要であり，そのサポートをするため，本研究では初学者の視覚的な理解を助けるシミュレーションを作成することを目的とした．

その視覚化のために，西谷研究室では数値計算ソフト Maple が多く用いられているがシミュレーション表現に限界がある．CG アニメーションソフト Maya を使用することにより，シミュレーションのクオリティーを高め，表現の幅を広げることが可能である．

視覚化の際には，Ruby と Mel の 2 つのスクリプト言語を使用した．Maya では全ての動作を Mel で記述することが出来るが，一般の言語に比べ，ループや線形代数の記述が複雑になるため，シンプルなスクリプト言語である Ruby を使用することでプログラムの作成を容易にした．

本研究では，物理現象，特に粒子の温度変化，結晶成長，連成振動，分散関係についてのシミュレーションを作成した．粒子の温度変化，結晶成長では分子動力学法を用いて粒子の軌跡を算出し，視覚化を行った．連成振動では，質点間に加わる力を算出し，速度，加速度を求めることで位置を割り出し軌道を求め，視覚化を行った．分散関係では，弦と鎖は波数と角振動数を求めることで，ある一点での弦と鎖それぞれの波を視覚化することが出来る．

これらのシミュレーションより，粒子の変化，結晶の様子，ばねによる質点の動き，弦の波と鎖の波の違いをシミュレーションを通して直感的に理解することが出来る．

目 次

第1章　序論

1.1　研究の目的と背景

今まで物理を学習していなかった学生に とって, 大学での物理の授業や物理の計算を 理解するのは非常に困難である. 一方, 的確 に視覚化すれば, 物理現象を即席に理解する ことが可能である. 本研究では運動する粒子 の温度変化の関係，分散関係では一周期あたりの波の変化を共に色の変化で表現することで見やすく分かりやすいシミュレー ションの作成を目的とした.

1.2　結晶成長

結晶成長とは，融点以下に冷やされた過冷却融液の中に結晶の核となるものがあれば，融液中の原子は核に付着し結晶が大きくなっていくことである．これは，融液中の原子が結晶表面に入射した際，結晶表面と衝突している極めて短い時間内に，原子の運動エネルギーの大部分を結晶表面での熱の発生という形で奪われ，入射した原子は結晶表面を構成する原子との結合を振り切ることは出来ず，表面拡散しながら徐々に結晶にとりこまれていくからである．本研究では，この融液と結晶の界面での原子の変化のシミュレーションを作成した [5].

1.3　連成振動

連成振動とは，複数の質点が互いに力を及ぼし合って運動しているときのことをという．(図 1.1) は簡単な連成振動の図である．

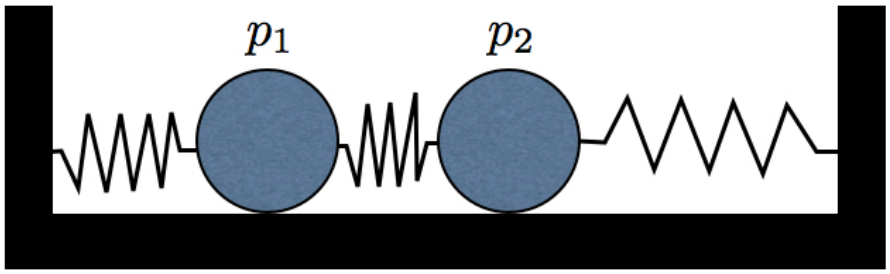

図 1.1: 連成振動

この図では，p1 もしくは p2 の質点が動くことで他の質点にばねにより力がかかる．なお，本研究では摩擦力などの力は無視できるものとしている．

1.4 分散関係

分散関係とは，角振動数ωと波数 k との間に成り立っている関係である．(図 1.2) の中の n1 は弦の分散関係で，n2 は鎖の分散関係である．n1 とは違い，n2 の様に角振動数ωが波 k に比例していない場合を分散があると言う．分散というのは，違う波数の波がばらばらに分かれるということで，光をプリズムで分光出来るのは分散があるからである．波数 k の伝播速度はω (k)/k で求めることが出来る．これより，弦 n1 の場合は全ての波数において，波が同速度 v で伝わるので図のように崩れることはない．一方，鎖 n2 の場合は，波数の大きな細かな波の伝播速度が遅いので図では形が崩れている．

図中の n1，n2 の角振動数ωと波数 k の関係式はそれぞれ,

$$\omega = vk \tag{1.1}$$

$$\omega = \left(\frac{2v}{a}\right) sin\left(\frac{ka}{2}\right) \tag{1.2}$$

となる．ここで，v は速度，a は質点間の距離である．速度は,

$$v = \sqrt{\frac{k}{m}}a \tag{1.3}$$

で，表すことが出来る [1].

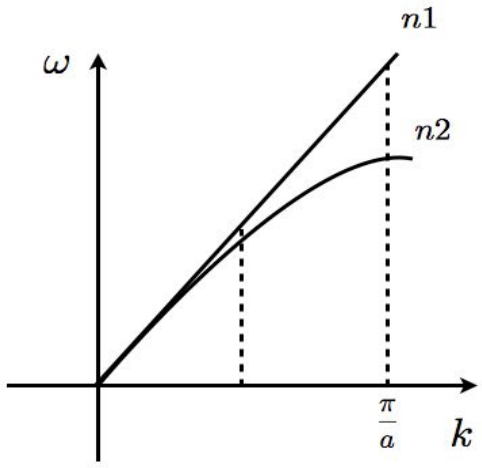

図 1.2: 分散関係

1.5 弦と鎖

弦を短い区間に切って考えると，短い区間で質点が弾性のあるばねで繋がっている図の様な鎖に近似することが出来る．図中の質点間距離 a を 0 に近づけ，質点の数 n を増やしていくことで，より弦に近似することが出来る [1].

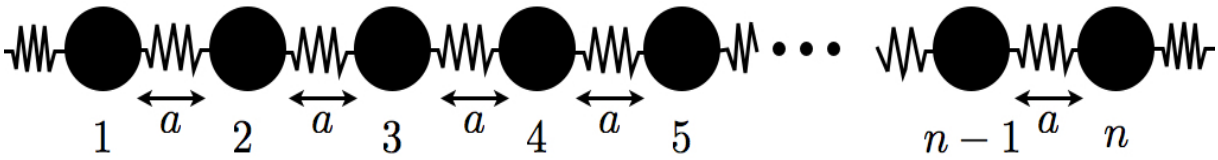

図 1.3: 弦への近似

1.6 シミュレーション作成

本研究では以下の項目について，シミュレーションを作成した.

- 粒子の温度変化

速度 0，加速度 0 の粒子の塊を安定した最密構造で配置し，そこに初速度を持った一つの粒子を衝突させる．それぞれの粒子間の力を Lenard-Jones ポテンシャルの式を用い算出し，速度と加速度を求める．その値をもとに Verlet のアルゴリズムにより時間毎の粒子の位置を求め，粒子を移動させる．時間毎の速度を求めることで温度の値を求めることが出来る．その値によって色を変化させ温度変化の様子を視覚化した．

- 結晶成長の様子

結晶と仮定した粒子の塊を下部に配置し，初速度を与えた多数の粒子を散乱させる．それぞれの粒子間の力は Lenard-Jones ポテンシャルの式を用い算出し，速度と加速度を求める．その値をもとに Verlet のアルゴリズムにより時間毎の粒子の位置を求め，移動させる．結晶成長の過程では，速度を持つ粒子は結晶本体に運動エネルギーを熱エネルギーとして吸収される．そのため，プログラム内では結晶本体に衝突した粒子は速度の値を下げることで運動エネルギーの減少を表した．その際に，どのように粒子が結晶に取り込まれていくか温度が変化していくか視覚化した．

- 連成振動

連成振動ではフックの法則でそれぞれの質点にかかる力を求め，時間毎の速度と加速度を求める．その値をもとに Verlet のアルゴリズムを用いて，時間毎の位置の値を算出する．その結果を用いて，連成振動では質点がどのように移動するかを視覚化した．

- 波の分散関係

波の変位の式から時間毎に質点の位置の値を求め，移動させていく．(図 1.1) から弦と鎖では，波数 k により角振動数 ω の値が変わる．大きく変わった際に，弦と鎖では波にどのような差異があるか視覚化した．

第2章　手法

2.1　Maya

シミュレーション手法として，本研究ではMayaを使用した．Mayaは，オープンアーキテクチャを基板とした強力な統合型の3Dモデリング，アニメーション，レンダリングソリューションである．多くのフィルムやビデオアーティスト，ゲーム開発者，マルチメディアデザイナー，3DCGに関わるSOHOデザイナーなどが使用しているプロ仕様のハイエンドソフトである．グラフィックス性に優れ，非常に高度な機能を有しているため，自由度も高い．また，MayaはそのGUIの全てをMELというスクリプト言語で実行可能となっている．MELとはMayaで使用できるスクリプト言語であり，MayaのGUIの機能の全てをまかなうことが可能となっている．MELのみでCGの作成，カスタマイズを行うことができるが，Maya専用のスクリプトエディタ上でしか実行できない．また，線形計算などの数値計算に向いておらず，構文も複雑な構成となっている．[3]

2.2　Ruby

Rubyとはまつもとゆきひろ氏により開発されたオブジェクト指向スクリプト言語である．一般的に，Rubyは数値計算には向いていないと言われる．実際に，RubyはCやFortanなどの他の言語と比べ，実行速度などの面で遅い．しかし，Rubyの言語は，単純な構文で書かれており，可読性に優れている．さらにコンパイルを必要としないインプリンタ方式を採用しているため，実行の手間が少ない．加えて，高機能なスクリーンエディタとして有名なEmacsと併用すれば，構文に応じてスクリプトが色分けされるので．開発環境が非常に良いものとなる．本研究では，以上のことからインターフェースとしてRubyを選定した．[4]

2.3　分子動力学法

分子動力学法は, 系を構成する粒子の運動方程式を時間について離散化 し, それらの方程式を連立して解いて粒子の運動を追跡していく方法であ る. ニュートンの運動方程式はエネルギー保存則を満足する. したがって, 熱力学的平衡状態を対象としたシミュレーションの場合には, 小正準集団 に対してのみ適用できる.

2.3.1 基礎方程式

系のエネルギーが保存される小正準集団や非平衡状態に対するシミュレーションに際しては，ニュートンの運動方程式が用いられる．粒子 i の位置ベクトルを r_i，粒子 i に作用する力を f_i とすれば，ニュートン運動方程式は次の様に書くことが出来る．

$$m\frac{d^2r_i}{dt^2} = f_i, i = 1, 2, ..., N \tag{2.1}$$

速度ベクトル v_i は位置の微分から，

$$v_i = \frac{dr_i}{dt} \tag{2.2}$$

もし，外力が作用しなければ，系の運動エネルギーや運動量および角運動量が保存される．しかし，シミュレーションでは一般に有限のシミュレーション領域を設定し，境界の影響を少なくするために周期境界条件を用いているので，必ずしもこれらの量が保存されるとは限らない．

2.3.2 Verlet のアルゴリズム

Verlet のアルゴリズムは初期状態の速度以外は全く用いず，粒子間に働く力をもとに粒子を逐次的に移動させる方法である．

Verlet のアルゴリズムでは（2.1）から直接粒子の位置の時間発展を求める差分方程式を作る．時刻 t+h と時刻 t-h における粒子の位置 r_i(t+h) と r_i(t-h) をテーラー展開し，(2.1）と $\mathrm{dr}_i/\mathrm{dt}=\dot{\mathrm{r}}_i$ を用いると，

$$r_i(t+h) = r_i(t) + h\dot{r}_i(t) + \frac{h^2}{2}\frac{f(t)}{m} \tag{2.3}$$

$$r_i(t-h) = r_i(t) - h\dot{r}_i(t) + \frac{h^2}{2}\frac{f(t)}{m} \tag{2.4}$$

を得ることが出来る．これらの和と差を求めると，

$$r_i(t+h) + r_i(t-h) = 2r_i(t) + h^2\frac{f_i(t)}{m} \tag{2.5}$$

$$r_i(t+h) - r_i(t-h) = 2hr_i(t) \tag{2.6}$$

となり，ここから時刻（t+h）における位置と時刻 t における速度は，

$$r_i(t+h) = 2r_i(t) - r_i(t-h) + h^2\frac{f_i(t)}{m} \tag{2.7}$$

$$\dot{r_i}(t) = \frac{1}{2h}\Big(r_i(t+h) - r_i(t-h)\Big) \tag{2.8}$$

となる．これがVerletの差分式である．時刻t+hにおける位置を求めるには2つの時刻tと(t+h)での位置が必要である．初期条件と位置を速度で与えると，t=hにおける位置r_i(h)は（2.7）から求めることが出来る．これとr_i(0)からr_i(2h)を計算し，(2.8）より速度$\dot{r_i}$を得ることが出来る [2].

2.4 Lenard-Jonesポテンシャル

Lenard-Jonesポテンシャルとは，分子間ポテンシャルを計算するために2つの粒子間の相互作用ポテンシャルエネルギーを表す経験的なモデルの1つで，ポテンシャル曲線を表す式が簡単で扱いやすいとされている．Lenard-Jonesの一般形の式は，

$$U(r) = 4\epsilon\left[\left(\frac{\sigma}{r}\right)^p - \left(\frac{\sigma}{r}\right)^q\right] \tag{2.9}$$

と表される．Uはポテンシャルエネルギー，rは粒子間の距離である．次数p,qはそれぞれ引力項，斥力項である．pを12，qを6とする式はLenard-Jonesポテンシャルの代表例である．

$$U(r) = 4\epsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right] \tag{2.10}$$

Lenard-Jonesポテンシャルは，長さσとエネルギーεという2つのパラメータを持っている．r=σでU(r)=0となる粒子間の距離で，εはU(r)の極小となる点である．Lenard-Jonesポテンシャルは，この2つの値によって一意に決まる．粒子間に働く力はポテンシャルエネルギーU(r)を粒子間距離rで微分することでもとめることが出来る．U(r)を微分した式は，

$$F(r) = -\frac{d}{dr}U(r) = 4\epsilon\left(12\frac{\sigma^{12}}{r^{13}} - 6\frac{\sigma^6}{r^7}\right) \tag{2.11}$$

となる [2].

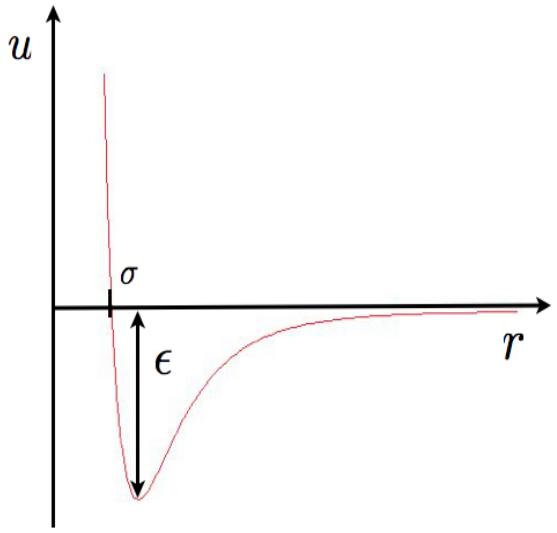

図 2.1: Lenard-Jones ポテンシャル

2.5 温度

温度は粒子の熱運動と関係し，熱速度，すなわち，粒子の速度から平均流速を引いた速度で定義される．今回の系が静止しているものと仮定しているので，粒子の速度そのものが熱速度となる．温度 T が与えられた系の場合，粒子の速度はマクスウェル分布に従った分布となる．運動エネルギー K(p) の集団平均を求めると

$$\langle K(p) \rangle = \langle \frac{1}{2m} \sum_{i=1}^{N} p_i^2 \rangle = 3N \frac{kT}{2} \tag{2.12}$$

のようになる．運動エネルギーは，

$$E = \frac{1}{2} m v^2 \tag{2.13}$$

と表すことが出来るので，式（2.13）より運動エネルギー K は

$$K_i = \frac{3}{2} NkT = \frac{1}{2} m v^2 \tag{2.14}$$

で表すことが出来る. そこから，

$$T_i = \frac{1}{3} \frac{1}{kN} m v^2 \tag{2.15}$$

と表すことが出来る．

2.6　フックの法則

フックの法則とは，ばねの伸びまたは縮みとばねの弾性力の大きさは正比例するという法則である．そこから (図 2.2) の様な場合，ばね定数を k とすると，弾性力 F とばねの伸びまたは縮み x の関係式は（2.16）のようになる．

図 2.2: ばね

$$F = kx \tag{2.16}$$

弦や鎖は物質をばねで繋げた物と仮定することが出来るので，図 1.2 の様に考えることが出来る．図中の a,b,c,d の物質の変位をそれぞれ x1,x2,x3,x4 とした時，物質 b の場合力 F との関係式は（2.17）のようになる．ただし，図中のばねのばね定数は全て k とする．

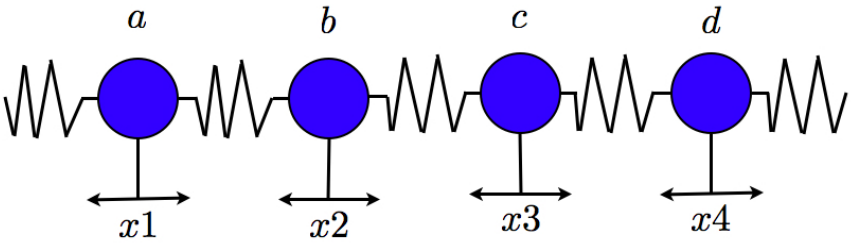

図 2.3: ばね

$$F = -k(x2 - x1) + k(x3 - x2) \tag{2.17}$$

2.7　波の変位

波を表す (図 2.4) では，ある位置 x においての時間毎の変位を表している．ここで，A は振幅,v は速度,t は時間,f は振動数, λ は波長とする．

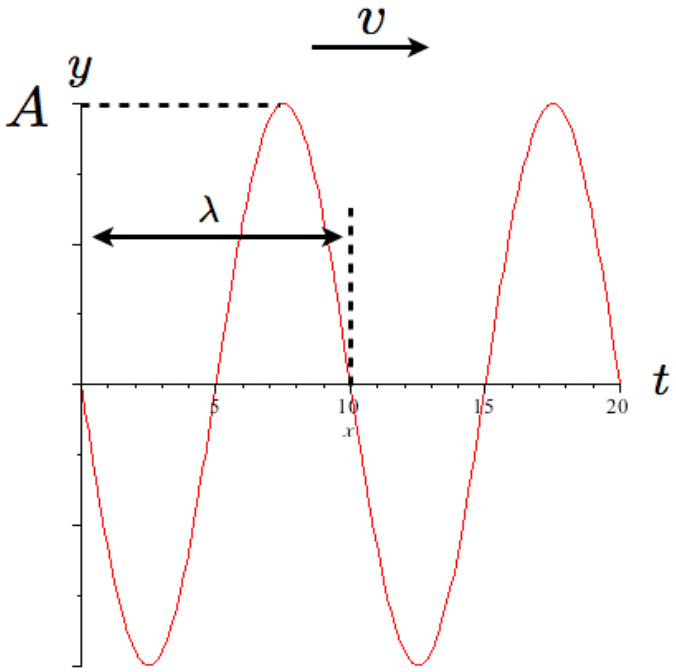

図 2.4: 波の変位

一般的に，変位 y の値は，

$$y = Asin2\pi(ft - \frac{x}{\lambda}) \tag{2.18}$$

等速円運動において，ωは，

$$\omega = 2\pi f \tag{2.19}$$

と表すことが出来る．

波数とは単位長（2 π）あたりの含まれる波の数なので，波の波数 k は，

$$k = \frac{2\pi}{\lambda} \tag{2.20}$$

と表すことが出来る．

式（2.18）と（2.19）を式（2.20）に代入することで，以下の変位の式に変換することが出来る．

$$y = Asin(\omega t - kx) \tag{2.21}$$

上記の式に分散曲線において算出されたωの値（1.1）（1.2）と波数 k を代入することで，図（1.2）の弦 n1 と鎖 n2 の波数 k の値による波の変位を求めることが出来る．

2.8 カラー化

シミュレーションを作成する際に，粒子の温度変化や進行する波の色を変えた．粒子の温度変化の場合は式 (2.15) から，値が高ければ赤く，低ければ青い色に表示することで，温度変化が見て分かる．分散関係の波の場合では，一周期分の波の色を変化させることで波の進行が見て分かる．物理現象を色の変化で表すことで，シミュレーションの見やすさが向上し，直感的な理解が可能となる．

2.9 Ruby によるプログラムの作成

2.9.1 初期条件の設定 (粒子の温度変化)

7 つの粒子を作成し，(図 2.1) の様に安定した最密構造に配置する．

```
$init[label] = [move,speed,accel,label of around atoms]
```

粒子を作成する際は，上記のプログラムで$init に格納している配列の要素，move，speed，accel，label of around atoms は粒子の位置，速度，加速度，周りの他の粒子のパラメータである．

```
$init[0] = [Vector[0,0,0],Vector[0,0,0],Vector[0,0,0],[]]
$init[1] = [Vector[0,1,0],Vector[0,0,0],Vector[0,0,0],[]]
$init[2] = [Vector[0,-1,0],Vector[0,0,0],Vector[0,0,0],[]]
$init[3] = [Vector[sqrt(3),0.5,0],Vector[0,0,0],Vector[0,0,0],[]]
$init[4] = [Vector[sqrt(3),-0.5,0],Vector[0,0,0],Vector[0,0,0],[]]
$init[5] = [Vector[-sqrt(3),0.5,0],Vector[0,0,0],Vector[0,0,0],[]]
$init[6] = [Vector[-sqrt(3),-0.5,0],Vector[0,0,0],Vector[0,0,0],[]]
```

衝突させる粒子に初速度を与える

```
$init[7] = [Vector[3,3,0],Vector[-2,-2,0],Vector[0,0,0],[]]
```

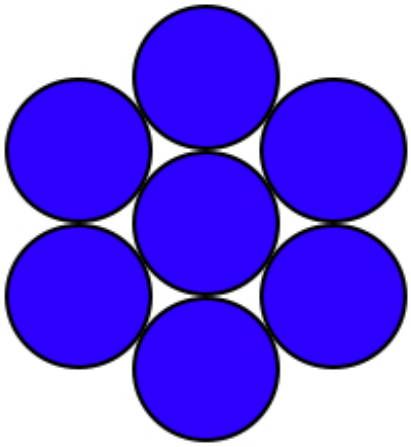

図 2.5: 最密構造

2.9.2 初期条件の設定 (結晶成長の様子)

粒子の塊を結晶として下部に固定し，他の粒子には適当な初速度を与え散乱させる.

固定させる粒子

```
for i  in 0..11 then


$init[i] = [Vector[-6+(1.1)*i,-6,0],Vector[0,0,0],Vector[0,0,0],[]]


end
```

適当な初速度を与えた散乱させる粒子

```
$init[0] = [Vector[-4,4,0],Vector[-2,-2,0],Vector[0,0,0],[]]
$init[1] = [Vector[0,-3,0],Vector[1,2,0],Vector[0,0,0],[]]
$init[2] = [Vector[1,0,0],Vector[-1,-2,0],Vector[0,0,0],[]]
$init[3] = [Vector[-2.5,5,0],Vector[1,2,0],Vector[0,0,0],[]]
$init[4] = [Vector[-3,1.5,0],Vector[1,-2,0],Vector[0,0,0],[]]
$init[5] = [Vector[4,5.2,0],Vector[-1,2,0],Vector[0,0,0],[]]
$init[6] = [Vector[0,4.2,0],Vector[1,-2,0],Vector[0,0,0],[]]
```

2.9.3 Lenard-Jones ポテンシャル

Lenard-Jones ポテンシャルは粒子間の距離を用いて，粒子間のポテンシャルを求めるものなので，初めに，全ての粒子において，周りの粒子との距離を計算する．

```
for i in 0..8 do
for j in 0..$init[i][3].size-1 do
label  = $init[i][3][j] #周りの他の粒子を順に格納する
tmp = $init[i][0] - $init[label][0] #周りの粒子との距離を求める
end
end
```

上記のプラグラムにより，粒子間の距離を求めたので,（1.11）の Lenard-Jones ポテンシャルの式に代入していく．

```
r = (tmp.r)
kk = 5*24*(2*(1/r)**13)-(1/r)**7) #Lnerd-Jones ポテンシャルの公式に数値を代入
f = tmp*kk
```

これにより，粒子間のポテンシャルを求めることが出来る．この結果を Verlet のアルゴリズムで使用する．

2.9.4 Verlet のアルゴリズム

Lenard-Jones ポテンシャルにより求めた粒子間の力をもとに，粒子を逐次的に移動させる．

そのポテンシャルとニュートンの運動方程式（1.1）から，それぞれの粒子の加速度 a を求めることが出来る．

$$a = \frac{F}{m} \tag{2.22}$$

本研究ではシミュレーション中の粒子の質量を 1 としているので加速度は F となり，加速度の変化は以下のプログラムとなる．

```
$init[i][2] = $init[i][2]+f
```

上記のプログラムから，速度の値を求めるプログラムを作成する．

```
$init[i][1] = $init[i][1]+$init[i][2]*dt
```

ここで求めた値を，以下の Verlet のアルゴリズムのプログラムに代入し，粒子の位置を算出する．

```
$init[i][0] = $init[i][0] + $init[i][1]*dt + ($init[i][2]*(dt**2))
```

これにより，時間毎に粒子を移動させることが出来る．

2.9.5 温度の表示

温度 T は（1.14）の式を用いて表すことが出来る．速度 v 以外は定数なので，温度は速度 v により変化することが分かる．プログラム内では速度の変化だけを考え，速度を温度と仮定しシミュレーションを作成した
まず，速度の値を tmp に格納する..

```
tmp = temperature($init[i][1])
```

以上のプログラムにより求めた値により，色を変化させるための場合分けをしていく．

```
if(tmp<50)then
file1 = currentTime_color(file1,t,color[i+2],[0,0,1])
elsif(tmp>=50 && tmp<70)then
file1 = currentTime_color(file1,t,color[i+2],[0.1,0,1])
elsif(tmp>=70 && tmp<90)then
file1 = currentTime_color(file1,t,color[i+2],[0.2,0,1])
elsif(tmp>=90 && tmp<110)then
```

```
file1 = currentTime_color(file1,t,color[i+2],[0.3,0,1])
elsif(tmp>=110 && tmp<130)then
file1 = currentTime_color(file1,t,color[i+2],[0.4,0,1])
.
.
.
elsif(tmp>=370 && tmp<390)then
file1 = currentTime_color(file1,t,color[i+2],[1,0,0.2])
elsif(tmp>=390 && tmp<410)then
file1 = currentTime_color(file1,t,color[i+2],[1,0,0.1])
else
file1 = currentTime_color(file1,t,color[i+2],[1,0,0])
end
```

以上により，温度が低い場合は粒子を青く表示し，高くなれば赤に近づくシミュレーションを作成した.

2.9.6 フックの法則

フックの法則の公式 (1.18) を用いて，それぞれの質点に加わる力を求める.

```
k = 100 #ばね定数
for a in 1..$move-1 do

$init[a][2] = $init[a][2]+

(($init[a][0]-Vector[n+n*a,0,0])-($init[a-1][0]-Vector[n*a,0,0]))*(k)+
(($init[a][0]-Vector[n+n*(a+1),0,0])-($init[a][0]-Vector[n+n*a,0,0]))*(k)

#フックの法則

end
```

これにより，全ての質点にかかる力を求めることが出来る.

2.9.7 分散曲線での波の変位

分散曲線上の波を，変位の式（1.22）を用いて質点の位置を算出し，動かしていく．弦と鎖ではωの値が違うので，それぞれのωを式（1.16）（1.17）に基づいて値を求め代入していく.

弦の周波数

```
omega = v*k
```

鎖の周波数

```
omega = (2*v/$dis)*sin(k*$dis/2)
```

変位

```
a*sin(omega*dt-k*i*$dis)
```

以上のプログラムにより，弦，鎖それぞれの変位を求めることが出来る．

2.9.8 一周期毎の色変化

波のシミュレーションを作成した際に，どのように波がすすんでいるか見やすいように，決まった一周期分の波に色をつけ移動させた．初めに，決まった一周期分の波を指定し色を変える．

```
if i in 0..60 do
if i>=0&&i<=41 then
file1,color[i] = color_blinn(file1,[1,1,0])
else
file1,color[i] = color_blinn(file1,[1,1,1])
end
end
```

その一周期分の色を波の速度に合わせ移動させていく．以下のプログラムでは 21 秒毎に変化していくようになっている．

```
if t%21==0&&t!=o then
file1 = currentTime_color(file1,t-20,color[0],[1,1,0])
file1 = currentTime_color(file1,t-20,color[0],[1,1,1])


file1 = currentTime_color(file1,t-20,color[41],[1,1,1])
file1 = currentTime_color(file1,t-20,color[41],[1,1,0])
end
```

以上のプログラムにより，波の変化に色を加えることが出来る．

第3章　結果

3.1　温度変化のシミュレーション

- 安定した最密構造の原子に適当な初速度を与えた粒子を衝突させ，粒子の動きの変化と温度の変化を色で視覚化した (図 3.1).

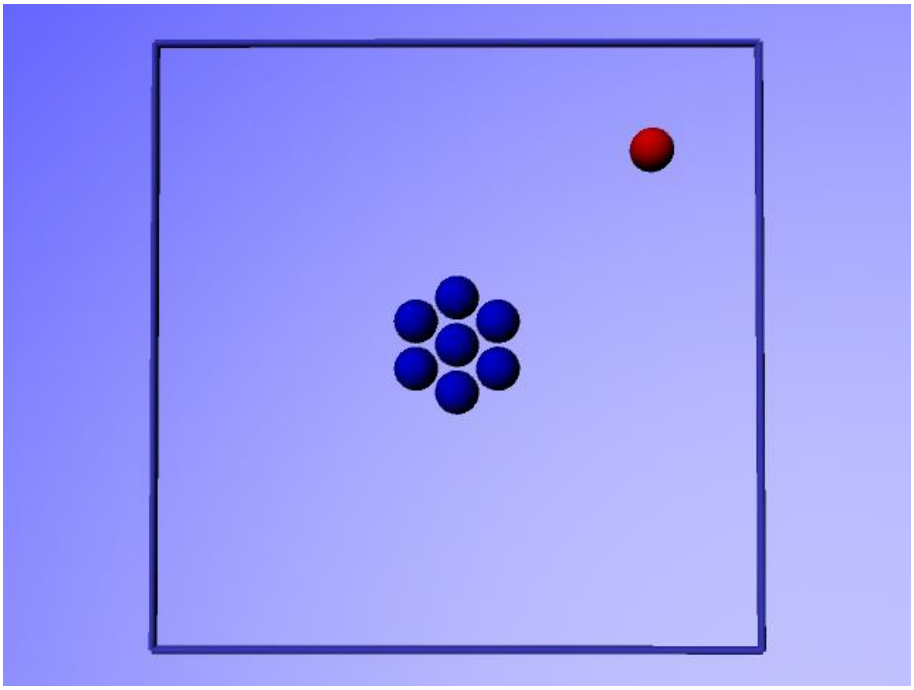

図 3.1: 温度変化（初期状態）

- (図 3.2) より，衝突後は安定していた時よりも多くの粒子が赤に近い色になり，形が崩れている．熱を持った粒子が衝突することで，安定していた粒子も熱を持ち不安定になっている様子が分かる．これは，水に熱を加え，沸騰している様子としても捉えられる．

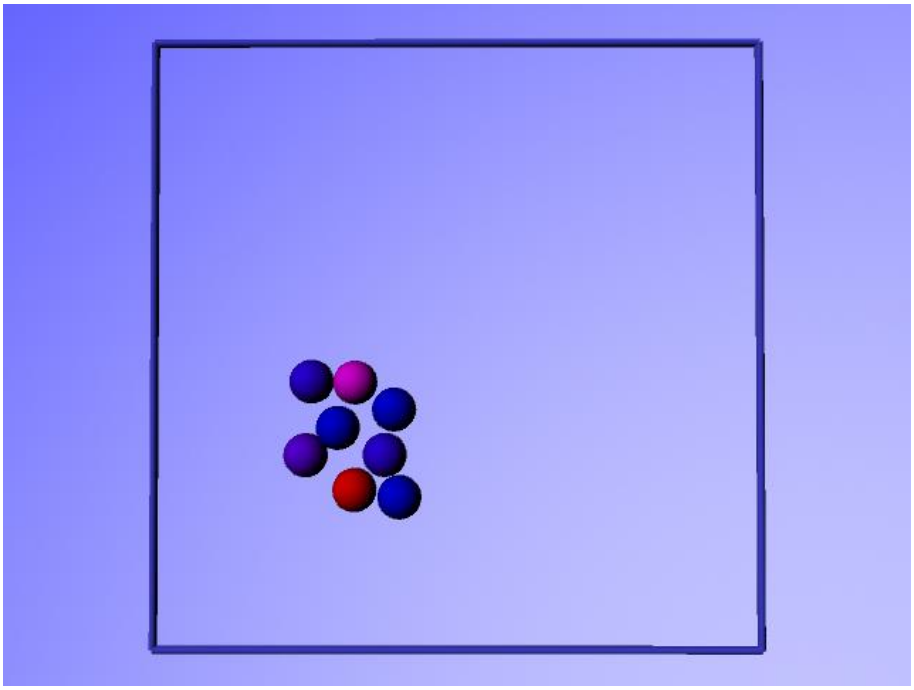

図 3.2: 衝突後

3.2　結晶成長のシミュレーション

- 粒子の塊を結晶として下部に固定し，適当な初速度を与えた粒子を散乱させ動きの変化と温度の変化を視覚化した (図 3.3).

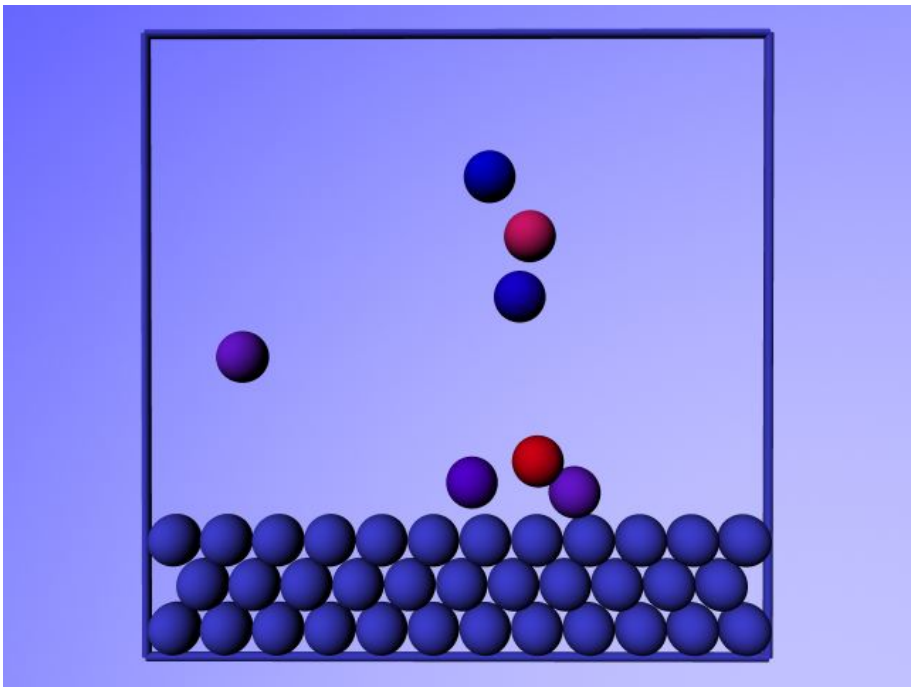

図 3.3: 結晶の核と散乱する粒子

- 散乱した粒子は分子間力により結晶に近づき，衝突した際に，運動エネルギーを熱エネルギーとして結晶に奪われる．運動エネルギーの奪われた粒子は表面拡散しながら，最終的に (図 3.4) のように安定した位置に定着し，結晶の核に取り込まれる．

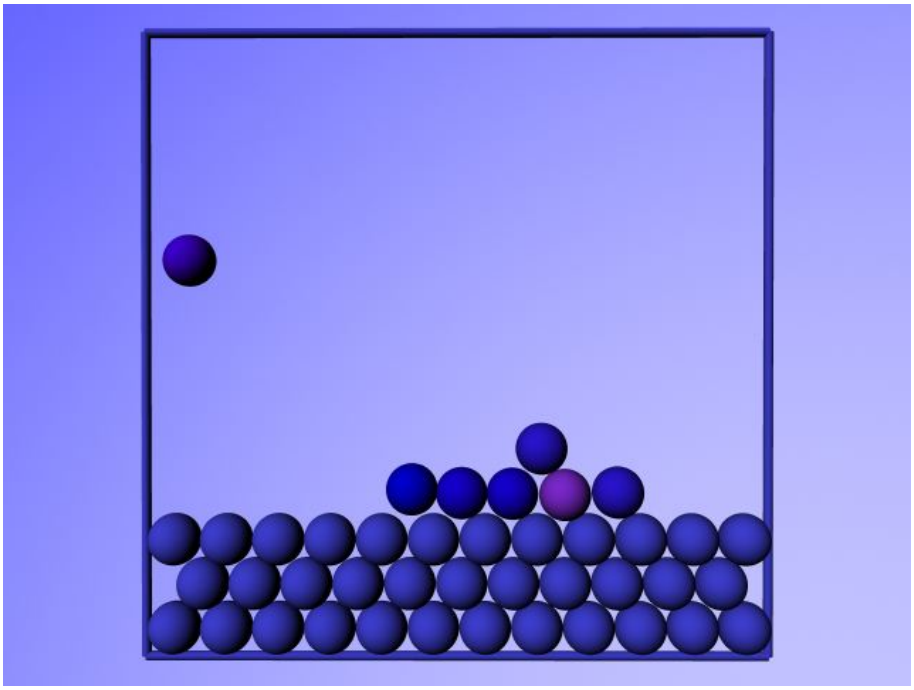

図 3.4: 結晶に取り込まれていく粒子

3.3　連成振動のシミュレーション

- 図の様に一次元で粒子を並べ，式（1.20）を用いて，粒子間で及ぼし合う力を算出し，速度，加速度，位置を求める．

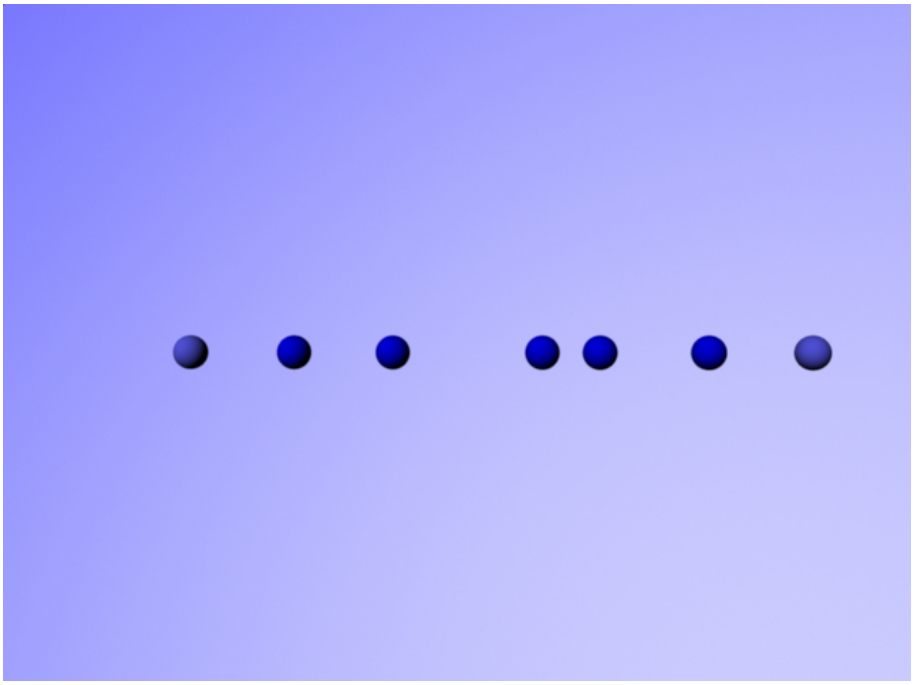

図 3.5: 連成振動

- 連成振動では軌跡をたどることで，動いている質点が規則性のある波を描いていることが分かる．

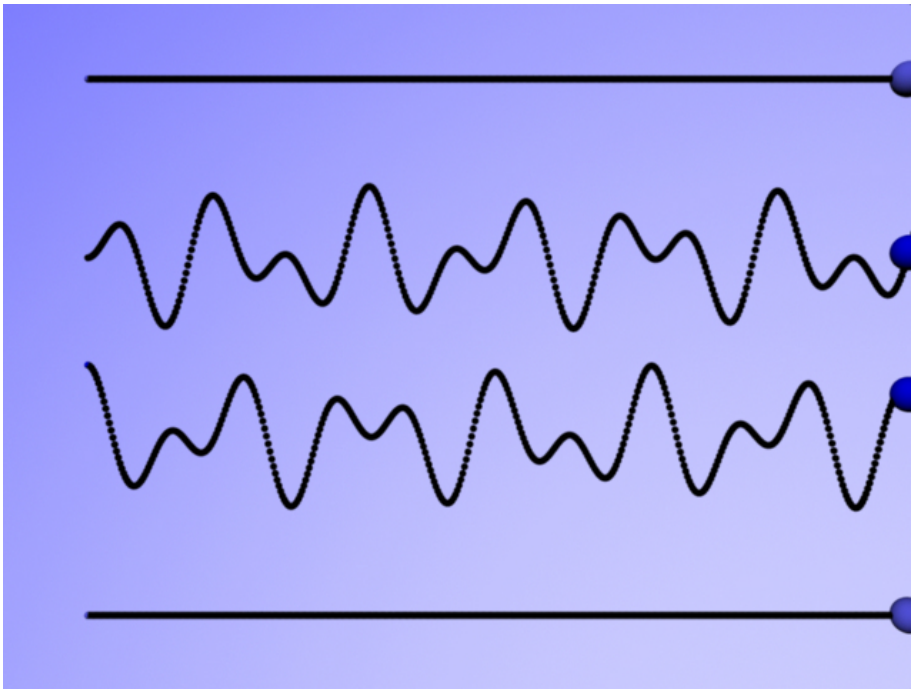

図 3.6: 連成振動による波

3.4　分散関係のシミュレーション

分散曲線では，図の様に波数 k が小さければ弦と鎖の周波数ωがほとんど変わらない．一方，波数 k が大きくなればなるほどωの値に差が出てくる．

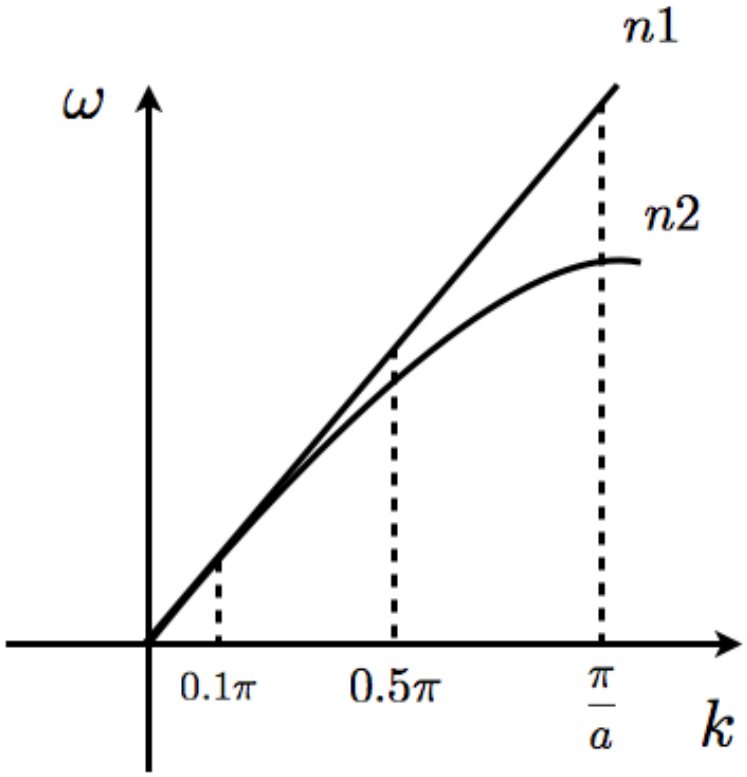

図 3.7: 分散曲線

以下に Maya のより分散曲線上の波を視覚化したシミュレーションを参照する．

3.4.1　0.1 π の時の弦

図は波数 k が 0.1 π の時の弦である．波数が低いので緩やかな波になり，式 (1.16) から周波数 ω も低い値なので，波の速度もゆっくりである．

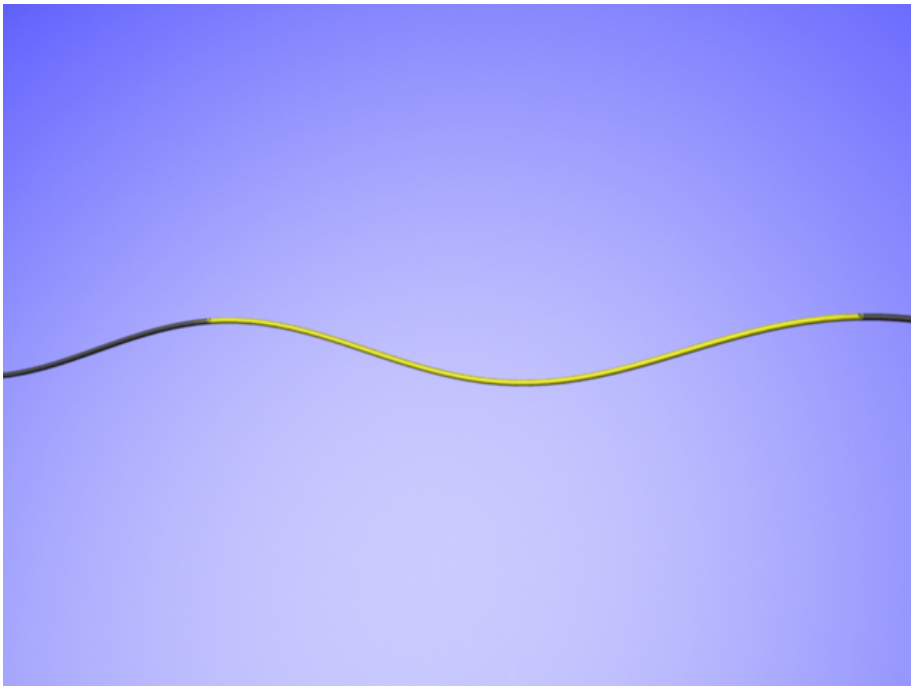

図 3.8: 弦の波（0.1 π）

3.4.2　0.1 π の時の鎖

図は波数 k が 0.1 π の時の鎖である．(図 3.8) と同様に，緩やかな波で速度も遅いため，弦との変化はほとんど見当たらない．

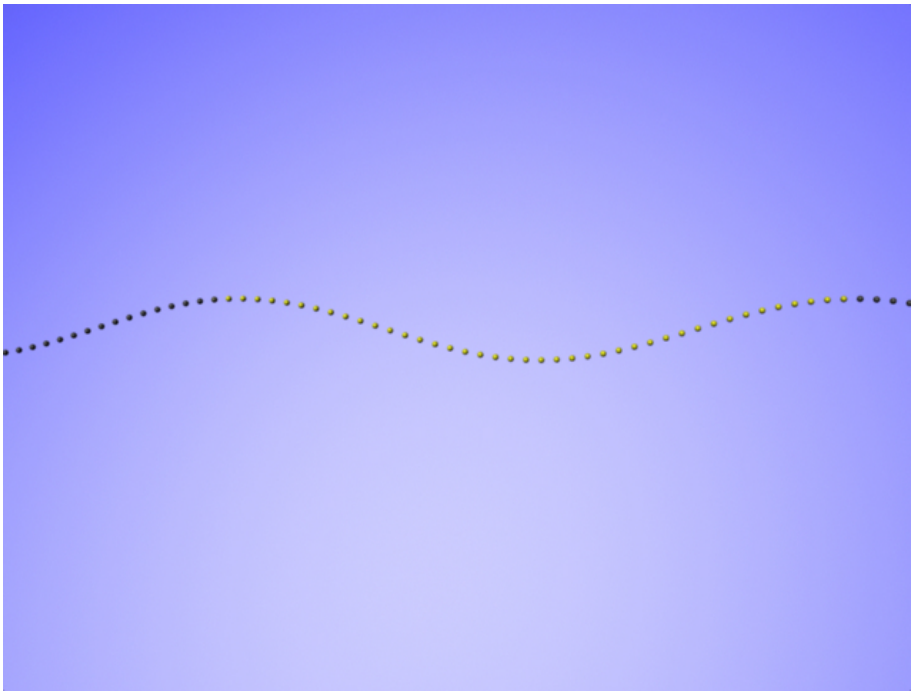

図 3.9: 鎖の波（0.1 π）

3.4.3　0.5 π の時の弦

図は波数 k が 0.5 π の時の弦である．(図 3.8) に比べ，波数が大きくなったため，波の数が多くなっている．波一周期の進む速度も 0.1 π の時に比べ，一定の時間内に多くの波を形成している．

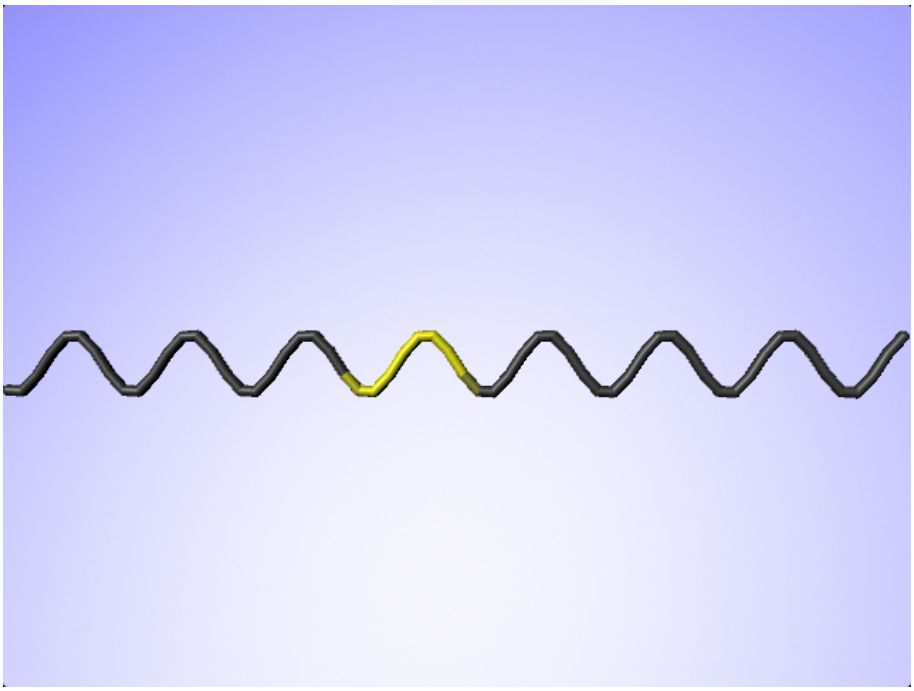

図 3.10: 弦の波（0.5 π）

3.4.4　0.5 π の時の鎖

図は波数 k が 0.5 π の時の鎖である．(図 3.10) と同様に，波数が大きくなったため，波の数が多くなっているが，(図 3.11) から明らかに (図 3.10) よりも波の速度が落ちている．これは，(図 3.7) からも分かるように，波数 k が大きくなる毎に周波数 ω の値が変わってくることから成っている．

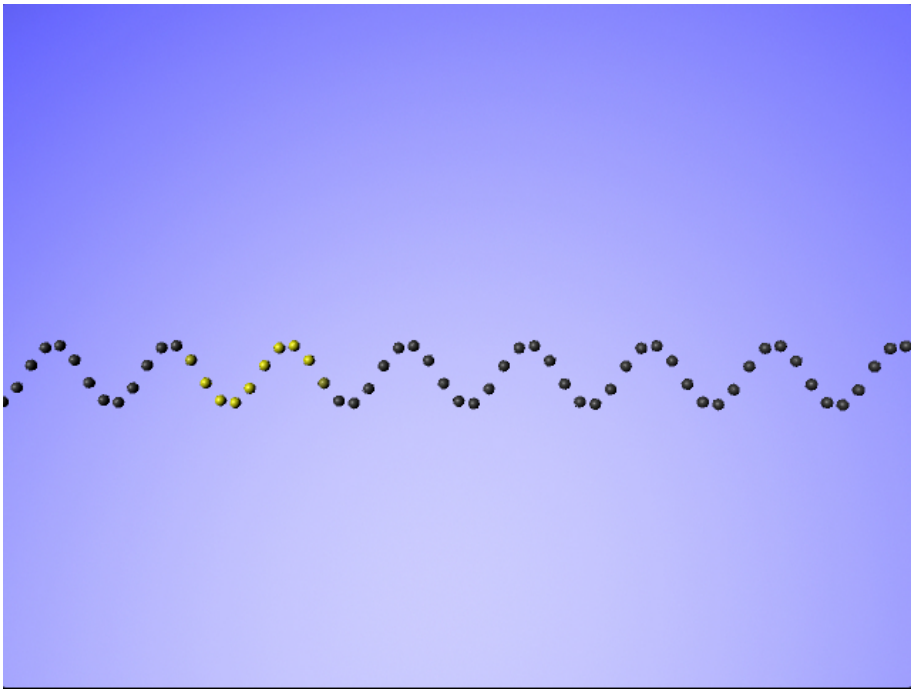

図 3.11: 鎖の波（0.5 π）

第4章　総括

本研究の結果を以下に示す.

- Maya は大変グラフィック性に優れているが，全ての動作をループや線形計算の記述が複雑なスクリプト言語である MEL で行うため物理のシミュレーションには向いていなかった．しかし，シンプルなスクリプト言語である Ruby を用いることで解決し，簡易化することで幅広いシミュレーションを作成することが可能となった.

- 温度変化のシミュレーションでは，実際にどのように粒子が動き，どのように温度が変化しているかを視覚的に表現した.

- 結晶成長のシミュレーションでは，衝突する粒子の運動エネルギーの変化，核の結晶に取り込まれるまでの表面拡散などの様子を的確に表現することが出来た.

- 連成振動のシミュレーションでは，ただ単にばねの力により動いている質点をシミュレートするのではなく，その軌跡を残すことで連成振動と波の関係性を視覚的に結びつけることが出来た.

- 分散関係のシミュレーションでは，波の一周期分の色を変えることで波の動きが的確に分かる．そこから，分散関係についてシミュレーションを通して観察することが出来るようになった.

- Maya はグラフィック性に優れている．シミュレーションでは細部にわたる変化を視覚化することが出来るので，Maple などのソフトに比べて，より印象深いシミュレーションとなった.

参考文献

[1] 吉岡大二郎　著,「振動と波動」（東京大学出版会，２００５）

[2] 谷沖由香　著,「運動する粒子の温度の視覚化」（関西学院，卒業論文，２００８）

[3] 阿部知弘　著「MEL 教科書-Maya プログミン グ入門」(ボーンデジタル）

[4] オブジェクト指向スクリプト言語 Ruby 公式サイト，http://www.ruby-lang.org/ja.

[5] 黒田登志雄　著,「結晶は生きている」（サイエンス社）

謝辞

本研究を遂行するにあたり，終始多大なる有益なご指導，及び丁寧な助言を頂いた西谷滋人教授に深い感謝の意を表します．また，本研究の進行に伴い，西谷研究員の皆様にも様々な知識の提供，ご協力を頂き，本研究を大成することができました．最後になりましたが，この場を借りて心から深く御礼申し上げます．