

卒業論文

特殊相対性理論における光のドップラー効果

大阪工業大学
情報科学部 情報システム科
学生番号 B02-095

江本 賢泰

2007年2月15日

概要

ニュートン力学は、速度の遅い物理現象を扱う場合には十分に役に立つのであるが、素粒子のように光速に近い速度で動く現象では矛盾が生じてしまう。この光速に近い運動は特殊相対性理論によって表される。本研究は、光のドップラー効果に着目して、光速に近い速度で移動する観測者が観測する星の光の波長の変化を視覚化した。特にニュートン力学と特殊相対性理論の違いを比べた結果、が分かった。

目次

1	序論	3
2	特殊相対性理論	4
2.1	ニュートン力学における時間と空間	4
2.2	光速不変の原理	5
2.3	ローレンツ変換	6
3	ドップラー効果	7
3.1	ニュートンのドップラー効果	7
3.2	特殊相対性理論のドップラー効果	8
4	横ドップラー効果	10
5	ドップラー効果の可視化	12
5.1	ドップラー効果の可視化	12
5.2	横ドップラー効果の可視化	15
6	結論	17
A	プログラムコード	18

1 序論

ドップラー効果とは、音波や電波などの波の周波数が発生源や観測者との相対的な速度によって、周波数が異なって観測される現象のことである。発生源が近づく場合には波の波長が詰められて周波数が高くなり、逆に遠ざかる場合は波長が伸ばされて低くなる。同様に観測者が動く時、相対的に近づく場合には伝播速度が上がるために周波数が高くなり、遠ざかる場合は低くなる。普段私達を感じる事ができるこの効果は、光の場合でも同様に起こる。遠ざかる光源からの光は赤方偏移し、近づく光源からの光は青方偏移する。しかし、光の伝播は特殊相対性理論に従うため、通常の波のドップラー効果とは違った現象を見せる。この光の世界で起こるドップラー効果を発光体、観測者、それぞれの速度変化によりどう見えるのかシミュレーションを行う。このシミュレーションにあたって、ニュートン力学と特殊相対性理論を用いるのだが、ニュートン力学とは、*S.I. Newton* によって完成された理論で、解析力学などとあわせて古典力学と呼ばれる。質点の運動の理論として、特殊相対性理論は速度が光速よりも十分遅いときニュートン力学で近似でき、量子力学は運動量が十分に大きい場合にニュートン力学で近似できる。またニュートンの万有引力理論は、重力が弱い場合の一般相対性理論の近似である。

相対性理論とは時間と空間の物理学で、この時間と空間を時空と呼ぶ。相対性理論は *A. Einstein* 一人によって完成されたものである。

A. Einstein は *J.C. Maxwell* によって完成された電磁気学とニュートン力学の間に生じた矛盾を解決しようと研究を始めた。そして、ニュートン力学的な時間や空間の概念が誤っていることを見つけた。相対性理論の成立は、1905年に完成した特殊相対性理論、1915年に完成した一般相対性理論による。本研究で用いる特殊相対性理論は光の速さに近い運動を正しく記述するものである。私たちの日常扱うスケールでの有効な理論であるニュートン力学と、特殊相対性理論を、ドップラー効果で比較する。

2 特殊相対性理論

アインシュタインは、電磁気学とニュートン力学の間に生じた矛盾を解決しようとして研究をはじめた。その結果私たちの日常経験から体得しているニュートン力学的な時間や空間の概念が、光速に近い運動状態や強い重力場のもとでは間違っていることを見つけ、相対性理論を作り上げた。この成立にあたって、まず、1905年に特殊相対性理論、次に1916年一般相対性理論が完成した。ここでは特殊相対性理論について述べる。

2.1 ニュートン力学における時間と空間

ニュートンは1687年に出版された『プリキンピア（自然哲学の数学的諸原理）』において絶対時間、空間ということの規定している。これは時空というものは絶対で物質の存在や運動によって変化するものではないと規定したものである。その上で、ニュートン力学の第一法則、第二法則が成り立っている。

第一法則は慣性の法則として知られ、外力が作用していないとき、物体は静止しているか、等速運動をするという法則である。

第二法則は運動の法則であり物体の加速度は外力に比例し、質量に反比例する。

静止、または等速運動というのは、何に対して静止しているのか、等速運動しているのかを定めなければならない。要するに、物体の位置を示す座標としてどのような座標系をとったのかを示さなければならない。私たちの周りの、日常の運動を表すとき、地面に固定した座標系を採用している。この座標系で近似的に慣性の法則が成り立つ。厳密には地球の自転運動や、公転運動により成立しているわけではない。さらに太陽も銀河系の中で回転運動をしているので当然誤差も増える。しかしニュートン力学では地球にたいしてどのような運動をしている座標系が分からないが、結果として慣性の法則が成立している座標系を慣性系と定義する。また、慣性系に対して等速運動している座標系はすべて慣性系であり、慣性系は無限に存在する。

第二法則を式で表したものが次の運動方程式である。

$$f = m \frac{d^2 x}{dt^2} \quad (1)$$

第三法則は作用反作用の法則である。つまり二つの質点間に働く力（作用）を働かせると、他方に同じ大きさで逆向きの力（反作用）を同一作用線上で働き返すというものである。

第二法則での座標系を x 系と呼ぶ。ここで m は質点の質量、 x は質点の座標、 t は時間である。 f はその質点に働く力である。これを別の慣性座標系 x' 系で表しても同じ方程式となり、

$$f = m \frac{d^2 x'}{dt^2} \quad (2)$$

となる。証明はガリレイ変換により行う。ここで x' 系は x 系に対して速度 v で x 軸方向に運動している座標系とすると、その座標変換は図のように次のようになる。

$$\begin{aligned} t' &= t \\ x' &= x - vt \\ y' &= y \\ z' &= z \end{aligned} \quad (3)$$

ガリレイ変換（座標変換図添付）により、時間は座標系の属性ではなく普遍的に流れるもので、両座標系での時間の原点さえ一致させれば、数値としても同じになるという事である。

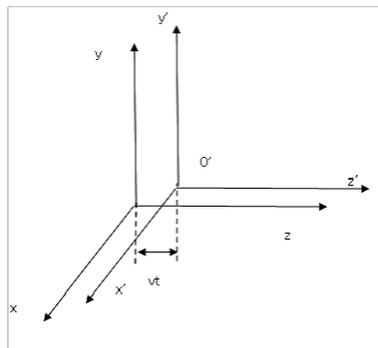


図 1: ガリレイ変換

このように同格であるどのような座標系で記述しても同じでなければならないとする考えを相対性原理という。ガリレイ変換に対して物理法則が不変であることをガリレイの相対性原理という。ニュートンの運動方程式はガリレイの相対性原理を満足する。

2.2 光速不変の原理

1861年マックスウェルは電磁場の基礎方程式を作り上げた。

$$\text{rot } E = -\frac{\partial B}{\partial t} \quad (4)$$

$$\text{rot } H = \frac{\partial D}{\partial t} + j \quad (5)$$

$$\text{div } D = \rho \quad (6)$$

$$\text{div } B = 0 \quad (7)$$

$$D = \epsilon_0 E \quad (8)$$

$$B = \mu_0 H \quad (9)$$

ここで E は電場ベクトル、 H は磁場ベクトル、 D は電束密度、 B は磁束密度、 j は電流密度、 ρ は電荷密度、 ϵ_0, μ_0 はそれぞれ真空の誘電率と透磁率である。これらの式を解くことにより、電磁波の存在を予言した。(4) 式の rot をとり、それに (9),(5),(8) を代入することにより、電磁場の方程式が得られる。

$$\left(\frac{1}{c^2} \frac{\partial^2}{\partial t^2} - \Delta \right) E = 0 \quad (10)$$

$$c^2 = \frac{1}{\mu_0 \epsilon_0} \quad (11)$$

ここで、 Δ はラプラス演算子、 $\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$ である。波の伝播速度である c は (11) 式で定義される量であるが、この値が光の速さと一致することから、光も電磁波の一つであることが明らかになった。

ここで、この速度は一体何に対する速度なのかという疑問が生じた。座標系を決める必要があるが、その解釈に二つの立場が考えられた。第一は、無限に存在する慣性系の中に一つ、絶対的な慣性系が存

在し、マックスウェル方程式はその座標でのみ成立するという考えである。要するに、無限に存在する座標系の中に一つだけ絶対的な慣性系が存在し、マックスウェルの方程式(4)~(9)はその座標系で成立し、光速 c はこの座標系での速度だとする立場である。電磁波はこの絶対慣性系上で静止した仮想的な媒質「エーテル (ether)」を伝播する波であると考えた。異なった座標系での法則や速度が知りたければガリレイ変換によって求めればよいのである。この立場は当時、最も常識的な考えであった。

第二は、全ての慣性系で相対性原理は絶対的真理であり、マックスウェル方程式は成立していると考えられる立場である。この立場では、あらゆる慣性系で電磁波は光速で伝播することになるが、これはガリレイの相対性原理に反する。

第一の立場で絶対慣性系上で静止した仮想媒質、エーテルの存在を確かめるために、1887年、マイケルソンとモーレイは、地球が絶対慣性系に対してどのような速度で運動しているかを求める実験をした。これにより、光はどの慣性系でも速さは同じだと明確にした。つまり、ガリレイ変換は日常生活の範囲では近似とできるが光速に近づくと矛盾することになった。

2.3 ローレンツ変換

慣性座標の間を結ぶ座標変換は、ガリレイ変換では光速不変の原理に矛盾する。しかし、2つの座標系の相対速度 v が光速よりはるかに小さい場合は近似的に成り立つはずで、アインシュタインは次の3つの条件を指導原理としてガリレイ変換を拡張し、正しい座標変換を求めた。

1. 相対性原理 (座標系 (慣性系) の相対性)
2. 光速不変の原理
3. $\frac{c}{v} \ll 1$ の極限でガリレイ変換に一致

まず 1. の座標系の相対性より、座標変換は次式のように線形の変換でなければならない。

$$\begin{pmatrix} ct' \\ x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{30} & a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} ct \\ x \\ y \\ z \end{pmatrix} \quad (12)$$

なぜかという、もし t や x, y, z に関して 2 次や 3 次の項を含むなら、 x 系で等速運動している質点は x' 系で力を受けないにも関わらず加速度運動していることになる。 x 系での等速運動が x' 系でも等速運動であるためには、またその逆も成立するためには、(12) 式のような線形変換でなければならない。

ここで x' 系の時間 t' は x 系の時間 t とは異なるとして変換を考えていることに注意しなければならない。アインシュタインは、絶対時間の概念 (座標系とは無関係に時間は流れているとする概念) を放棄し、時間も座標系ごとに異なるとしたのである。絶対時間の概念を放棄しない限り、先の条件 1~3 をみたくはできないのである。(12) 式の変換では、時間そのものを用いず、それに光速をかけた ct をもちいた。これは、座標と同じ長さの次元をもった量として変換すると、変換の行列が無次元となり、きれいな形式で表現されるからである。

2つの慣性系に関して x' 系が x 系に対して x 軸方向に速度 v で運動しているとする (図) 同様に $t = t' = 0$ で両座標系の時間の原点は一致していたとし、また y 軸と y' 軸、 z 軸と z' 軸は完全に重なっているものとする。つまり、 $y' = y$ 、 $z' = z$ である。

3 ドップラー効果

パトカーが自分の方に近づいてくると、サイレンの音は高く聞こえ、遠ざかると低く聞こえるということを経験することがあるが、この効果を音のドップラー効果と呼ぶ。

3.1 ニュートンのドップラー効果

音源 S と観測者 R が x 軸の正の方向にそれぞれ速度 u 、 v で運動しているとする。音源は周波数 f_0 で周期 τ の音を出しているとする。音の波長 λ は $\lambda = a\tau = a/f_0$ で表される。ここで a は音速である。周期 τ ごとにパルス状に信号を出しているとする。

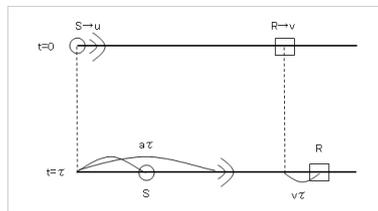


図 2: ニュートン力学のドップラー効果

図に示すように $t = \tau$ で第二のパルスを出すとする。 $t = \tau$ では S は $x = u\tau$ 、先に出たパルスは $x = a\tau$ に進んでいる。すると二つのパルスの間隔を λ' と表すと、

$$\lambda' = (a - u)\tau = \frac{a - u}{f_0} \quad (13)$$

である。観測者に対するパルスの速度は $a - v$ であるので、二つのパルスが観測者に届く時間間隔を τ' と書くと、

$$\tau' = \frac{\lambda'}{a - v} = \frac{a - u}{f_0(a - v)} \quad (14)$$

となる。 τ' の逆数 f_N を実質的な周波数とすると、

$$f_N = f_0 \frac{a - v}{a - u} = f_0 \frac{1 - v/a}{1 - u/a} \quad (15)$$

が得られる。これが音のドップラー効果の公式である。これは、ニュートン力学の範囲では光でも成立する。また音源が静止している場合 ($u = 0$) は、

$$f_N = f_0(1 - v/a) \quad (16)$$

となり、観測者が静止している場合は ($v = 0$)、

$$f_N = \frac{f_0}{1 - u/a} \quad (17)$$

となる。式 16 では観測者が音源から遠ざかるので、周波数は小さくなり、音は低く聞こえる。式 17 では音源が観測者に近づくので、周波数は大きくなり音は高く聞こえる。

3.2 特殊相対性理論のドップラー効果

音と同じことを光で考えてみる。音の場合には音を伝える空気という媒質が存在するので、空気に対して静止している座標系で考えた。光では光を伝える媒質、エーテルの存在は否定されたので、光源の座標系で考える。

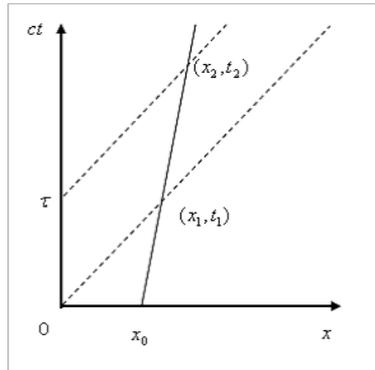


図 3: 光源の座標系から見た観測者の運動

図が示すように、 $t = 0$ で第一のパルスが、その後 $t = \tau$ で第二のパルスが発せられたとする。はじめのパルスを受け取ったときの観測者の座標は (x_1, t_1) で、次にパルスを受け取ったときの位置は (x_2, t_2) とすると、図 3 から、

$$\begin{aligned} x_1 &= ct_1 = x_0 + vt_1 \\ x_2 &= c(t_2 - \tau) = x_0 + vt_2 \end{aligned}$$

となり、よって、

$$t_2 - t_1 = \frac{v\tau}{c - v}, \quad x_2 - x_1 = \frac{v\tau}{c - v}$$

が成り立つ。観測者とともに運動する S' 系では、ローレンツ変換から、

$$\begin{aligned} t'_2 - t'_1 &= \gamma[(t_2 - t_1) - v(x_2 - x_1)/c^2] \\ &= \gamma\left(\frac{c\tau}{c - v} - \frac{v}{c^2} \frac{v\tau}{c - v}\right) \\ &= \frac{\gamma c}{c - v} \left(1 - \frac{v^2}{c^2}\right) \tau \\ &= \frac{1 - \beta}{\gamma(1 - \beta^2)} \tau = \gamma(1 + \beta)\tau \end{aligned}$$

である。ここで、 $\beta = \frac{v}{c}$ と定義した。また、 $\gamma = (1 - \beta^2)^{-1/2}$ である。また、 $t'_2 - t'_1$ は観測者の受け取るみかけの周期 τ' である。

よって、

$$\tau' = \left(\frac{1 + \beta}{1 - \beta} \right)^{1/2} \tau \quad (18)$$

あるいは周波数で表すと、

$$f_{SR} = \left(\frac{1 - \beta}{1 + \beta} \right)^{1/2} f_0 \quad (19)$$

となる。これが光に対するドップラー効果の公式である。音の場合に似ているが、音の場合と違い、光源が動くか観測者が動くかという違いはなく、ただ相対速度のみによる。それはエーテルという媒質が存在しないからである。(19) 式は観測者が光源に対して離れていく場合に相当している。この場合には、周波数が減少する、赤方偏移という。もし、光源と観測者が近づいてくる場合は、(19) 式の β の符号の方を変えればよい。この場合は、周波数が増加し、青方偏移という

。

4 横ドップラー効果

先に導いた公式は、光源が観測者に対して視線上の速度だけを持つ場合であったが、一般化することで横ドップラー効果という特殊相対性理論に特有の性質が現れる。

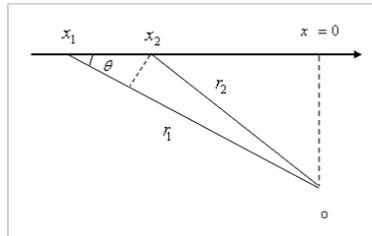


図 4: 波の発射源が、観測者の視線に対して角度を持って運動する場合

図に示すように、光源は図(4)で示すように、光源は直線上を動き、観測者は O にいるとする。光源が $t = t_1, x = x_1$ で第一のパルスを発射し、 $t = t_2, x = x_2$ で第二のパルスを発射したと仮定す。観測者 O からパルス発射地点までの距離はそれぞれ r_1, r_2 である。パルス間隔は先の例と同様に τ とする。先の例では光源に対して静止した座標系で説明したが、横ドップラー効果では観測者に対して静止した座標系である。したがって、パルス間隔 τ は間延びして観測される。

その関係は、

$$t_2 - t_1 = \gamma\tau = \frac{\gamma}{\nu} \quad (20)$$

となる。

パルスが観測点にまで達するには、それぞれ $\frac{r_1}{c}, \frac{r_2}{c}$ の時間がかかる。したがって観測されるパルス間隔を τ' とすれば、

$$\tau' = t_2 + \frac{r_2}{c} - (t_1 + \frac{r_1}{c}) = \gamma\tau - \frac{(r_1 - r_2)}{c} \quad (21)$$

である。

ここで r_1, r_2 を近似して、

$$r_1 - r_2 \approx (x_2 - x_1)\cos\theta = v\gamma\tau\cos\theta \quad (22)$$

が成立する。ここで v は光源の進む速度である。(22) 式を (21) 式に代入し観測される周波数 ν' を求めると、

$$\nu' = \frac{\nu}{\gamma(1 - \beta \cos\theta)} = \nu \frac{(1 - \beta^2)^{\frac{1}{2}}}{1 - \beta \cos\theta} \quad (23)$$

となる。ここで $\theta = \pi$ の場合は、 $\cos\pi = -1$ であるから、(23) 式は (19) 式に一致する。

5 ドップラー効果の可視化

本章では、プログラムを使って実際にドップラー効果を考察してみる。タブレット中の画面左上は左から、光源の元の色、特殊相対性理論での変化とニュートンでの変化を示す。また、グラデーションバーは上が特殊相対性理論での、下がニュートン力学での可視光の範囲を示す。

5.1 ドップラー効果の可視化

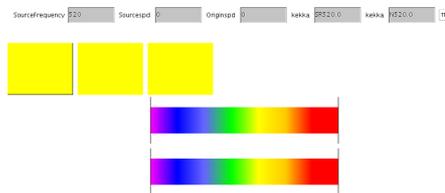


図 5: 速度無い場合

図 5 では速度が無い場合の状態では黄色に見える星の色は変わらないことをあらわしている。

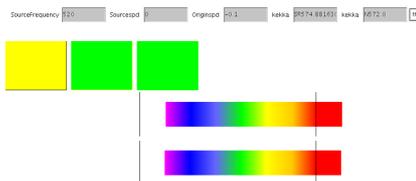


図 6: 光速の 10 % で近づく場合

図 6 では青方偏移し、黄色の星が緑に見えることが分かる。

図 7 では同じく青方偏移するが、特殊相対性理論でのドップラー効果が大きいことが分かる。そして、図 8、図 9 では不可視領域に入る。また、光速に近づくとより特殊相対性理論でのドップラー効果が大きいことが分かる。

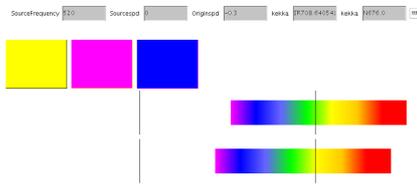


図 7: 光速の 30 % で近づく場合

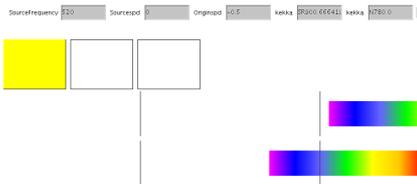


図 8: 光速の 50 % で近づく場合

以上のことより、図 15 が求められる。

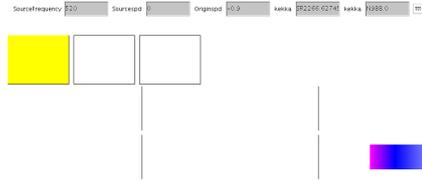


図 9: 光速の 90 % で近づく場合

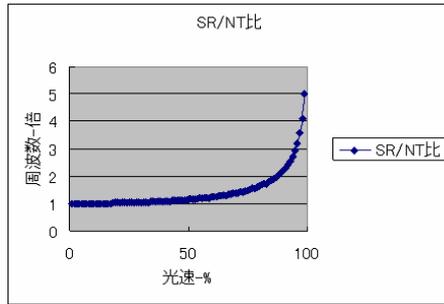


図 10: 周波数変化の比

5.2 横ドップラー効果の可視化

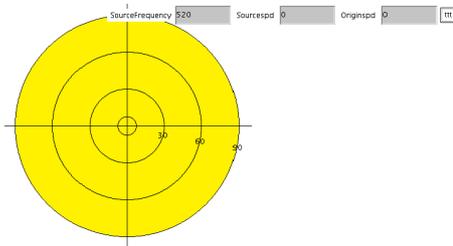


図 11: 速度の無い場合

速度が無い場合は視線の角度に関係なく一様に色の変化は無い。

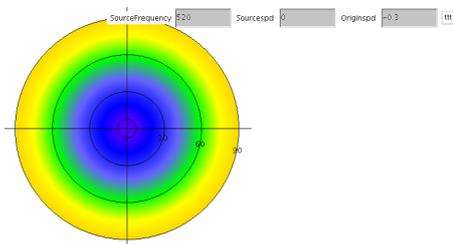


図 12: 光速の 30 %で

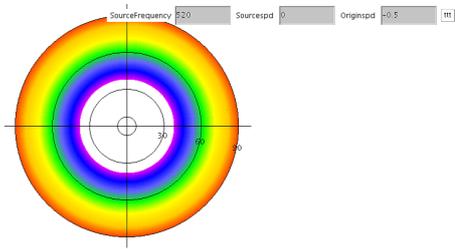


図 13: 光速の 50 %で

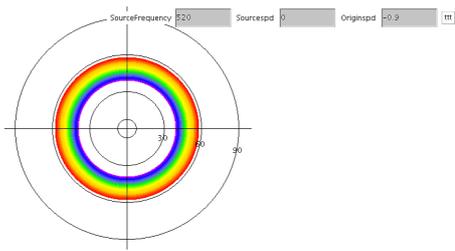


図 14: 光速の 90 %で

6 結論

本研究では、私たちの日常扱うスケールでの有効な理論であるニュートン力学と、光の速さに近い運動を正しく記述する特殊相対性理論を、ドップラー効果で比較した。ここで、ドップラー効果、横ドップラーについて研究による結果を示す。

ドップラー効果の可視化より、ニュートン力学、特殊相対性理論で表される周波数変化を比にとることで、図 15 が求められる。

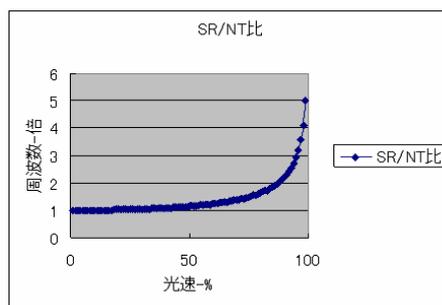


図 15: 周波数変化の比

これより光速の 40 %まではニュートン力学で近似できることがわかる。また光速に近づくにしたがって、特殊相対性理論のドップラー効果が大きく見られ、光速の 50 %では可視光領域は見えなくなる。ここで不可視光領域であった赤外線が可視光に偏移する現象が起きると考えられる。

ドップラー効果の可視化より、中心程青方偏移が起こり、光速の 50 %で可視領域外へ周波数が偏移する。また光速の 50 %付近から $\theta = 90$ 、いわゆる真横に見えるものは赤方偏移が起こり、光速の 60 %付近から赤方偏移により可視領域外へ周波数が偏移する。

また図 14 より見えている景色が虹色に変化し、*StarBow* と呼ばれる現象が起こる。

A プログラムコード

(ニュートン力学と特殊相対性理論でのドップラー効果について。)

```
import java.awt.*;
import java.applet.*;
import java.awt.event.*;
import java.util.*;
import java.applet.Applet;

public class subyukimi extends Applet
    implements ActionListener {

    TextField sf,ss,os,tnm,tnn;
    Button b1 = new Button("ttt");
    Color test1,test2,test3;
    double idou,idoun;

    public void init() {

        add( new Label("SourceFrequency", Label.RIGHT) );
        sf = new TextField( "", 10);    add( sf );
        add( new Label("Sourcespd", Label.CENTER) ) ;
        ss = new TextField( "", 10);    add( ss );
        add( new Label("Originspd", Label.CENTER) ) ;
        os = new TextField( "", 10);    add( os );

        add( new Label("kekka", Label.CENTER) ) ;
        tnm = new TextField( "", 10);    add( tnm );
        os.addActionListener(this);

        add( new Label("kekka", Label.CENTER) ) ;
        tnn = new TextField( "", 10);    add( tnn );
        os.addActionListener(this);

        setBackground(Color.white);
        b1.setBackground(Color.white);
        b1.setForeground(Color.black);
        b1.addActionListener(this);
        add(b1);

    }
}
```

```

public void paint(Graphics g) {
    g.drawRect(75,75,125,100);
    if (test1 !=null){
        g.setColor(test1);
        g.fillRect(75,75,125,100);
    }else{
        g.setColor(Color.black);
        g.drawRect(75,75,125,100);
    }

    g.drawRect(210,75,125,100);
    if (test2 !=null){
        g.setColor(test2);
        g.fillRect(210,75,125,100);
    }else{
        g.setColor(Color.black);
        g.drawRect(210,75,125,100);
    }

    g.drawRect(345,75,125,100);
    if (test3 !=null){
        g.setColor(test3);
        g.fillRect(345,75,125,100);
    }else{
        g.setColor(Color.black);
        g.drawRect(345,75,125,100);
    }

//from coco
    Color    col[] = new Color [312];
    int      c      = 0;

    for(int i=0;i < 52;i++){
        col[i]      = new Color(255-i*5      ,0      ,255);
        col[i+52]   = new Color(i*2      ,i*2      ,255);
        col[i+104]  = new Color(102-i*2      ,100+i*3,255-i*5);
        col[i+156]  = new Color(i*5      ,255      ,0);
        col[i+208]  = new Color(255,255-i      ,0);
        col[i+260]  = new Color(255      ,204-i*4      ,0);
    }

    int cl;

```

```

for(c=0;c<312;c++){
cl=c+15;
if(cl>=312){
cl-=312;
}
}

    int i;
    for(i=0;i<312;i++){
        cl = c + i;

        if(cl >= 312){
            cl -= 312;
        }
        g.setColor(col[cl]);
        g.drawLine(350+i+(int)idou,200 ,350+i+(int)idou ,250);

    }
    g.setColor(Color.red);
    g.fillRect(350+i+(int)idou,200 ,50 ,51);
g.setColor(Color.black);
g.drawLine(350,180,350,270);
g.drawLine(712,180,712,270);

//nuton
Color    coln[] = new Color [312];
int      d      = 0;

    for( int j=0;j < 52;j++){
        coln[j]      = new Color(255-j*5      ,0      ,255);
        coln[j+52]  = new Color(j*2      ,j*2      ,255);
        coln[j+104] = new Color(102-j*2      ,100+j*3,255-j*5);
        coln[j+156] = new Color(j*5      ,255      ,0);
        coln[j+208] = new Color(255,255-j      ,0);
        coln[j+260] = new Color(255      ,204-j*4      ,0);
    }

    cl=0;
for(d=0;d<312;d++){
cl=d+15;
if(cl>=312){
cl-=312;
}
}

```

```

}
}

    int j;
    for(j=0;j<312;j++){
        cl = d + j;

        if(cl >= 312){
            cl -= 312;
        }
        g.setColor(coln[cl]);
        g.drawLine(350+j+(int)idoun,300 ,350+j+(int)idoun ,350);

    }
    g.setColor(Color.red);
    g.fillRect(350+j+(int)idoun,300 ,50 ,51);
    g.setColor(Color.black);
    g.drawLine(350,280,350,370);
    g.drawLine(712,280,712,370);

}

```

```

public void actionPerformed(ActionEvent e) {
    double sf1=Double.parseDouble(sf.getText());
    double ss1=Double.parseDouble(ss.getText());
    double os1=Double.parseDouble(os.getText());
    double v=ss1+os1;
    double he=(1.0-v)/(1.0+v);

    double fn=sf1*((1.0-os1)/(1.0-ss1));
    double ff=sf1*Math.sqrt((1.0-v)/(1.0+v));
}

```

```

        idou=ff-sf1;
        idoun=fn-sf1;
tnm.setText("SR"+ff);
tnn.setText("N"+fn);

if(430.0<=sf1 && sf1<490.0){
    test1=new Color(255,0,0);
}else if(490.0<=sf1 && sf1<510.0){
    test1=new Color(255,200,0);
}else if(510.0<=sf1 && sf1<530.0){
    test1=new Color(255,255,0);
}else if(530.0<=sf1 && sf1<600.0){
    test1=new Color(0,255,0);
}else if(600.0<=sf1 && sf1<670.0){
    test1=new Color(100,100,255);
}else if(670.0<=sf1 && sf1<700.0){
    test1=new Color(0,0,255);
}else if(700.0<=sf1 && sf1<750.0){
    test1=new Color(255,0,255);
} else{
test1 = null;
}

```

```

if( 430.0<=ff && ff<490.0){
    test2=new Color(255,0,0);
}else if(490.0<=ff && ff<510.0){
    test2=new Color(255,200,0);
}else if(510.0<=ff && ff<530.0){
    test2=new Color(255,255,0);
}else if(530.0<=ff && ff<600.0){
    test2=new Color(0,255,0);
}else if(600.0<=ff && ff<670.0){
    test2=new Color(100,100,255);
}else if(670.0<=ff && ff<700.0){
    test2=new Color(0,0,255);
}else if(700.0<=ff && ff<750.0 ){
    test2=new Color(255,0,255);
} else{
test2 = null;
}

```

```

if( 430.0<=fn && fn<490.0){

```

```

        test3=new Color(255,0,0);
    }else if(490.0<=fn && fn<510.0){
        test3=new Color(255,200,0);
    }else if(510.0<=fn && fn<530.0){
        test3=new Color(255,255,0);
    }else if(530.0<=fn && fn<600.0){
        test3=new Color(0,255,0);
    }else if(600.0<=fn && fn<670.0){
        test3=new Color(100,100,255);
    }else if(670.0<=fn && fn<700.0){
        test3=new Color(0,0,255);
    }else if(700.0<=fn && fn<750.0 ){
        test3=new Color(255,0,255);
    } else{
test3 = null;
    }

```

```

repaint();

```

```

}

```

```

}

```

```

\\

```

(横ドブラー効果の可視化)

```

import java.awt.*;
import java.applet.*;
import java.awt.event.*;
import java.util.*;

public class side7kai extends Applet
    implements ActionListener {

    TextField sf,ss,os,ka,tnm;
    Button b1 = new Button("ttt");
    Color test1,test2,test3,test4;

    double ff,f1,f2,f3,f4,sf1,ss1,os1,v;

```

```

public void init() {

    add( new Label("SourceFrequency", Label.RIGHT) );
    sf = new TextField( "", 10);    add( sf );
    sf.addActionListener(this);
    add( new Label("Sourcespd", Label.CENTER) ) ;
    ss = new TextField( "", 10);    add( ss );
    ss.addActionListener(this);
    add( new Label("Originspd", Label.CENTER) ) ;
    os = new TextField( "", 10);    add( os );
    os.addActionListener(this);

setBackground(Color.white);
    b1.setBackground(Color.white);
    b1.setForeground(Color.black);
    b1.addActionListener(this);
    add(b1);

}

public void paint(Graphics g){
    Color    col[] = new Color [312];
    int      c      = 0;

    for(int i=0;i < 52;i++){
        col[i]      = new Color(255-i*5    ,0    ,255);
        col[i+52]  = new Color(i*2    ,i*2    ,255);
        col[i+104] = new Color(102-i*2    ,100+i*3,255-i*5);
        col[i+156] = new Color(i*5    ,255    ,0);
        col[i+208] = new Color(255,255-i    ,0);
        col[i+260] = new Color(255    ,204-i*4    ,0);
    }
}

```

```

    }

int cl;
for(c=0;c<312;c++){
cl=c+15;
if(cl>=312){
cl-=312;
}
}

for(int q=90;q>0;q--){
    double ff=sf1*( Math.sqrt(1.0-(v*v)) /(1.0-(v*(Math.cos((Math.PI/180)*q)))) );
cl=742-(int)ff;

if (0<=cl && cl<311){
    g.setColor(col[cl]);
    g.fillOval(200-(q*2),200-(q*2),(q*2)*2,(q*2)*2);
}else{
    g.setColor(Color.white);
    g.fillOval(200-(q*2),200-(q*2),(q*2)*2,(q*2)*2);
}
}

g.setColor(Color.black);
g.drawLine(200,0,200,400);
g.drawLine(0,200,400,200);
g.drawOval(20,20,360,360);
g.drawOval(80,80,240,240);
g.drawOval(140,140,120,120);
g.drawOval(185,185,30,30);

g.drawString("30",250,220);
g.drawString("60",310,230);
g.drawString("90",370,240);

```

```
}
```

```
public void actionPerformed(ActionEvent e) {  
    sf1=Double.parseDouble(sf.getText());  
    ss1=Double.parseDouble(ss.getText());  
    os1=Double.parseDouble(os.getText());
```

```
    v=ss1-os1;
```

```
    double f1=sf1*( Math.sqrt(1.0-(v*v)) /(1.0-(v*(Math.cos((Math.PI/180)*30))) ));
```

```
    if(430.0<=f1 && f1<490.0){  
        test1=new Color(255,0,0);  
    }else if(490.0<=f1 && f1<510.0){  
        test1=new Color(255,200,0);  
    }else if(510.0<=f1 && f1<530.0){  
        test1=new Color(255,255,0);  
    }else if(530.0<=f1 && f1<600.0){  
        test1=new Color(0,255,0);  
    }else if(600.0<=f1 && f1<670.0){  
        test1=new Color(100,100,255);  
    }else if(670.0<=f1 && f1<700.0){  
        test1=new Color(0,0,255);  
    }else if(700.0<=f1 && f1<750.0){  
        test1=new Color(255,0,255);  
    } else{  
        test1 = null;  
    }  
}
```

```
    repaint();
```

```
    }
```

```
}
```

(周波数比の値決め)

```
class zeek{  
public static void main(String[] args){
```

```

double sf1=520,v=0.5;

for(int q = 1; q < 90;q++){
    double ff=sf1*( Math.sqrt(1.0-(v*v)) /(1.0-(v*(Math.cos((Math.PI/180)*q))) ));
    int cl=742-(int)ff;

//double ff=sf1*( Math.sqrt(1.0-(v*v)) /(1.0-(v*(Math.cos(i)))) );
//System.out.println(i+ " : "+ff);
System.out.println(q+": "+cl+"--ff;"+ff+"---cos;"+Math.cos((Math.PI/180)*q));

}

```

参考文献

- [1] 佐藤勝彦, 相対性理論 (岩波基礎物理シリーズ9), 岩波書店 (1996)
- [2] 松田卓也、二間瀬敏史, なっとくする相対性理論, 講談社 (1996)