マルチメディアプログラミングによるシミュレーションソフト作成例

岡村好庸*, 中島貴志**, 春山和男*

An example of simulation software by multi-media programing

Yoshinobu OKAMURA, Takashi NAKASHIMA and Kazuo HARUYAMA

Abstract

A simulation software for special relativity is developed by using Visual Basic which we call multi-media programing language, which can handle pictures and sounds effectively in addition to the computation. Three events of the special relativity, *i.e.*, Lorentz contraction, slowing of clocks and direction shift of light observed from moving rocket can be simulated by the present software.

1. はじめに

プログラミング言語は近年OSの進展に伴い,急激に変化し始めている. 特にWindowsの出現により, 新しく開発されたプログラミング言語Visual Basic[1]やVisual C++では従来のBASICやCにおけるプログラム開発ではやや困難とされていた画像や音声を容易に取り扱う事が出来る. 我々は従来の数値計算や表計算などのコンピューティングを主体としたプログラミング言語に対して, このようなプログラミング言語をマルチメディアプログラミング言語と呼ぶことにする. この言語は一般にOSのグラフィカルユーザーインターフェース(GUI)を利用してアプリケーションプログラムを作成するツールであり, プログラミングというよりはある程度出来上がった部品を組み立てていくというイメージで捉えることができる. したがってプログラミングにあたり, 部品にあたるものの機能を理解し使いこなせるという学習は付加的に必要となる. しかしプログラム開発者が必要とする支援機能が十分に用意されているため, 結果的に画像や音声を取り扱うプログラム開発においては効率の飛躍的な向上が期待できる.

マルチメディアプログラミング言語を用いたアプリケーション開発として, ゲームソフト, 教材ソフト, 実務または研究用シミュレーションソフト, 計測処理ソフトこれらがオーバーラップしたものなどが考えられる. 社会のニーズとともにマルチメディアプログラミング言語は今後高専の情報教育のなかで幅広く取り入れられていくと思われるので, その重要性を認識して今後の情報教育のため

*宇部工業高等専門学校 電気工学科

**宇部工業高等専門学校 電気工学科, 現在 長岡技術科学大学

の一環として, 今回Visual Basicを用いて, 特殊相対性理論に関するシミュレーションソフトを作成した. 特殊相対性理論を選んだ理由は学生の多くが興味をもっているテーマであること, 結果を画像(ビジュアル)で表示することにより他の分野より理解を深めるのに適していること, また言語との相性も考慮した. このようなシミュレーション教材としては, 広瀬, 喜多村著「パソコンで学ぶ相対性理論」[2]があるがBasicによるプログラミングを用いているため出力部のプログラムに多大の労力をとられる難点がある. これに対してVisual Basicではオブジェクトと呼ばれる様々な部品があらかじめ用意されており, それをフォームと呼ばれるウィンドウ上に配置してイベントプロシージャを記述することでプログラムが作成される. フォーム上に配置したオブジェクト群がそのまま実行画面となるため, フォームを見ながらどのような操作を想定したら使いやすいのかを考えながらのプログラムの作成が可能である[3].

2. 特殊相対性理論に関するシミュレーション

本シミュレーションソフト作成の目的は, 特殊相対性理論の入り口ともいうべき部分の, 「光速に近い速さで移動するときに周囲の様々な現象はどのように見えるか」という疑問を解決するようなシミュレータを開発することにより, 特殊相対性理論の世界での現象を視覚的にとらえることにある. 現在様々な特殊相対性理論に関する資料は存在するが, その中でローレンツ変換についての記述は単なる数式の羅列で終っていることが多く, 定量的に具体的に表現したものは非常に少ないので, この点も開発にあたり考慮した.

特殊相対性理論において表れる顕著な現象は「物体のローレンツ収縮」と「動いている時計の遅れ」であるが[4], さらに今回は高速で移動するロケットから観察される星の様子を示す「光の集中現象」もシミュレートできるようにした.

2.1 基本画面

作成したシミュレーションソフトを起動すると, 図1が表示される. ユーザは「物体の収縮」, 「時計の遅れ」, 「光の集中現象」の3つのコマンドボタンから実行したいテーマを1つ選びクリックすることでそのテーマをシミュレートできる. また中止ボタンをクリックすればシミュレーションソフトを終了できる.

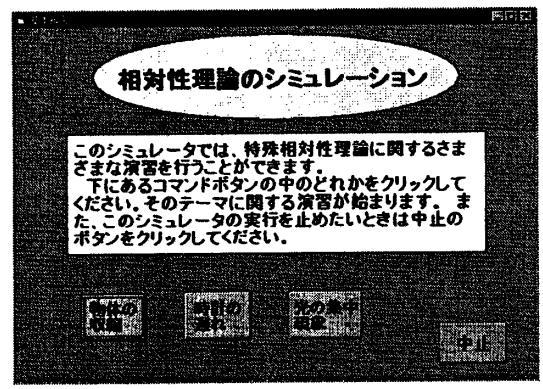

図1

2.2 物体の収縮

等速運動している棒の長さLはローレンツ収縮の式により表される[4].

$$L = L_0\sqrt{1-(u/c)^2} \qquad (2.2.1)$$

ここでL_0は静止しているときの棒の長さ, uは棒の移動速度, cは光速である. 具体的にはL_0を$1m$として次にuを設定する. 棒の移動速度が設定値になるまで増加させ収縮の様子をアニメーションする.

「基本画面」において「物体の収縮」ボタンをクリックすると図2が表示される.

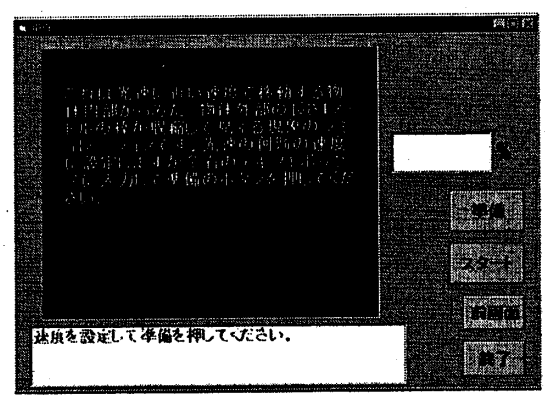

図2

画面右上のテキストボックスに棒の速さを光速に対する割合で入力して, 「準備」ボタンをクリック

し次に「スタート」ボタンをクリックするとアニメーションが始まり，結果が表示される．図３は結果の表示画面である．

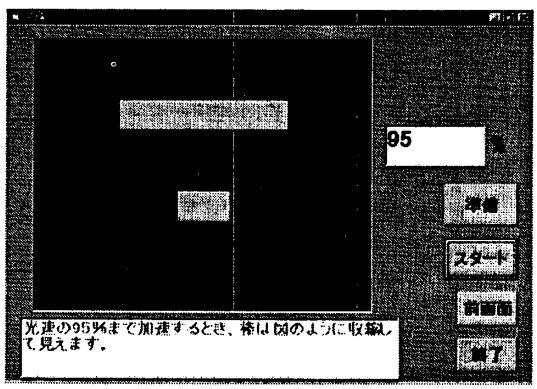

図３

「終了」ボタンをクリックするといつでも実行を中断できる．また「前画面」ボタンをクリックすると「基本画面」に戻るがその際いままで入力した数値などはすべてクリアされる．

2.3 時計の遅れ

動いている時計の時刻t'は，静止している時計の時刻tを用いて

$$t' = t\sqrt{1-(u/c)^2} \qquad (2.3.1)$$

と表される[4]．すなわち動いている時計は遅く進むように見える．この現象を，静止している時計が１年を経過したとき，それに対して設定された速さuで動いている時計の経過時間をシミュレートする．

「基本画面」において「時計の遅れ」ボタンをクリックすると図４が表示される．

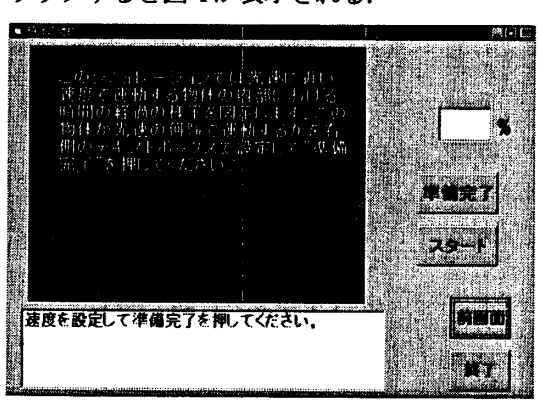

図４

画面右上のテキストボックスに移動している時計の速さを光速に対する割合で入力して，「準備」ボタンをクリックし次に「スタート」ボタンをクリックするとアニメーションが始まり，結果が表示される．図５は結果の表示画面である．

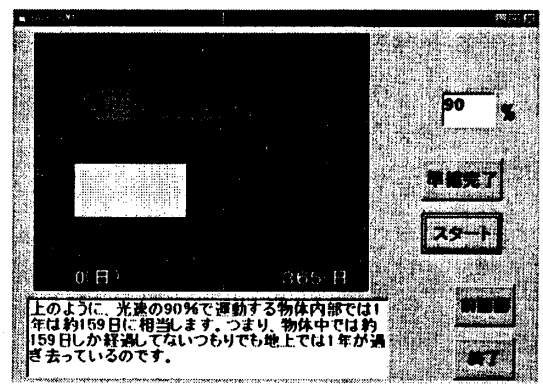

図５

「終了」「前画面」ボタンは前と同様である．

2.4 光の集中現象

ロケットが高速に近い速度で移動すると，視界前方の星の光は進行方向に集中する．もしロケットが光速で移動すると光は進行方向の１点に集まる．これはuで動いているロケットの進行方向と星の光がくる方向との角度をθ'とすると

$$\tan\theta' = \frac{c\sin\theta\sqrt{1-(u/c)^2}}{c\cos\theta + u} \qquad (2.4.1)$$

と表される．ここでθはロケットが静止しているときのθ'の値である．星までの距離と方向を乱数で与え画面に表示した後にuを設定する．ロケットの移動速度が設定値になるまで増加させ星の光の集中する様子をアニメーションする．(2.4.1)式はローレンツ変換の場合に用いられ，ガリレイ変換には(2.4.1)式において$\sqrt{1-(u/c)^2}$を１とおいた式が使われる．

「基本画面」において「光の集中現象」ボタンをクリックすると図６が表示される．

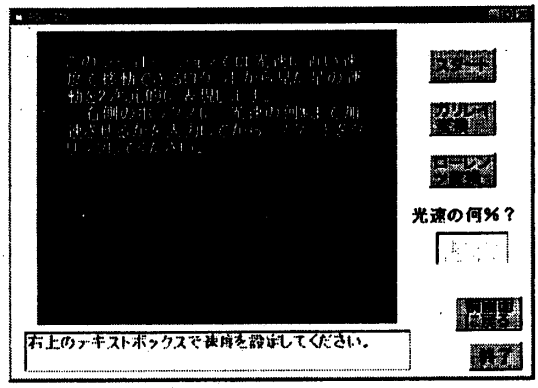

図6

画面右のテキストボックスに移動しているロケットの速さを光速に対する割合で入力して，「スタート」ボタンをクリックすると静止状態における星の様子が図7のように画面に表示される．

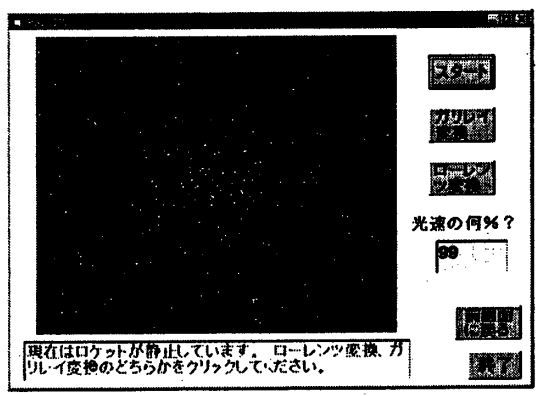

図7

この状態でガリレイ変換またはローレンツ変換のどちらかを選択する．図8は「ガリレイ変換」ボタンをクリックした場合，図9は「ローレンツ変換」ボタンをクリックした場合の画面を示す．

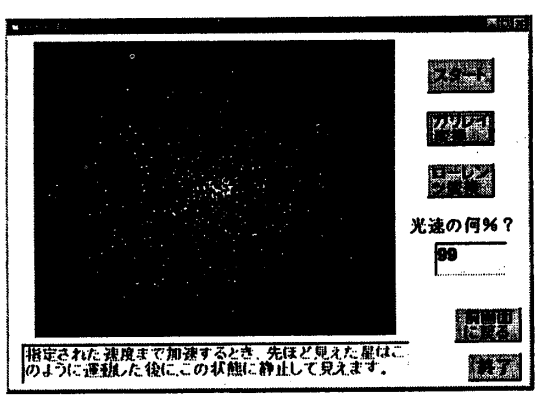

図8

図9

ローレンツ変換の場合にはガリレイ変換と比べて光がより集中していることが分かる．「終了」「前画面」ボタンは前と同様である．

3. まとめ

本ソフトウエアの作成にあたり，「視覚的」という点に重点をおき，そのためにマルチメディアプログラミング言語である Visual Basic を用いた．Windows 環境下においてのソフトウエア開発において，この言語は出力をイメージしながらプログラムできること，およびユーザーがオブジェクトにどのような実行をするかを個別に記述することによりプログラムの構造がより明確であることなどの長所により今後広く使われていくと思われる．当初の目標はある程度達成できたが，さらに音声の支援機能やヘルプファイルの活用を通してマルチメディアプログラミング言語にふさわしいマルチメディア対応のソフトウエア開発も可能であるのでこれらについては今後の課題としておく．

付録 (2.4.1) 式の導出

座標系 $O-xyz$(S 系) に対して x 方向に一定の相対速度 u で運動している座標系 $O'-x'y'z'$(S' 系) とは次のローレンツ変換で結び付けられる[4]．

$$
\begin{aligned}
x' &= \frac{x-ut}{\sqrt{1-(u/c)^2}}, \\
y' &= y, \\
z' &= z, \\
t' &= \frac{t-(ux/c^2)}{\sqrt{1-(u/c)^2}}
\end{aligned}
$$

今, S系で速度$\mathbf{v} = (v_x, v_y, v_z)$で運動している質点をS′系で観測するとその速度$\mathbf{v}' = (v'_x, v'_y, v'_z)$は

$$\begin{aligned} v'_x &= \frac{dx'}{dt'} = \frac{v_x - u}{1-(uv_x/c^2)}, \\ v'_y &= \frac{dy'}{dt'} = \frac{v_y\sqrt{1-(u/c)^2}}{1-(uv_x/c^2)}, \\ v'_z &= \frac{dz'}{dt'} = \frac{v_z\sqrt{1-(u/c)^2}}{1-(uv_x/c^2)} \end{aligned}$$

質点をxy面内でx軸からθの角度をなす方向からくる光であると考えると, その速度はS系において

$$\mathbf{v} = (-c\cos\theta, -c\sin\theta, 0)$$

となる. したがってS′系においては

$$\mathbf{v}' = \left(\frac{-c\cos\theta - u}{1+(u/c)\cos\theta}, \frac{-c\sin\theta\sqrt{1-(u/c)^2}}{1+(u/c)\cos\theta}, 0\right)$$

すなわち, x'軸となす角度θ'は

$$\tan\theta' = \frac{v'_y}{v'_x} = \frac{c\sin\theta\sqrt{1-(u/c)^2}}{c\cos\theta + u}$$

参考文献

1) Microsoft VisualBasic Version 4.0 ランゲージリファレンス (マイクロソフト,1995).

2) 広瀬立成, 喜多村章一 : パソコンで学ぶ相対性理論 (コロナ社,1990).

3) 野田晃 : Visual Basic 4.0 ではじめる Window95 プログラミング (ナツメ社,1996).

4) 原康夫: 物理学 (学術図書,1991).

(平成9年9月24日受理)