令和4年度 学士論文

物理系を定義させるシミュレータ SimSymの提案

東京工業大学 情報理工学院 数理・計算科学系
学籍番号 18B04657
木内 康介

指導教員
増原 英彦 教授

令和5年2月28日

概要

物理教育のために、物理実験のシミュレータが利用されている。高等学校での物理学の授業では実験が重要であるが、生徒全員が実験を経験しているわけではない。理由としては、実験用の装置の準備や測定が難しいことや、実験を行うのに時間を要することが考えられる。シミュレータを用いることで、実験と同様の学習効果を得ることができる。

しかし既存のシミュレータでは、学習者は可視化されている運動とその背景に存在する数式の間の関係を理解することは容易ではない。既存のシミュレータでは、教育者が物理法則から現象を数式で記述し、シミュレーションに変換する。学習者はシミュレーションを見ることで現実の物理現象を確認することができるが、それと物理法則との関係性は授業などで学ぶのみである。

本研究は、学習者に物理系を定義させるシミュレータ Simulation with Symbols (SimSym) を提案する。学習者は SimSym で系内の物体をそのパラメータとともに定義し、その物体の運動を表す方程式を立式する。シミュレーションを実行すると、定義した物理系に基づいて数値計算がなされ、物体の運動が可視化される。これにより、現実の物理現象と物理法則の間の対応を学習者自らの経験を通して理解することができると考えられる。この際学習者は、SimSym に実装された動作例を参考にすることで、定義した物理系と現実の運動を比較することができたり、次元の異なる物理量の和が存在する不正な方程式を立式すると警告されるなど、正しい物理系を作成するための補助を受ける。

謝辞

本研究を進めるにあたり、増原英彦教授、叢悠悠助教にアドバイスやご指導をいただきました。この場を借りて感謝申し上げます。また、増原研究室の皆様に多くのコメントを頂きました。特に、ご自身の研究でお忙しい中で何度も協力してくださった田辺さん、私と同じく教育に関する研究をしており相談に乗ってくれた角田さん、学生室で雑談相手になってくれた津山くんに深く感謝しております。

目 次

第1章　はじめに

高等学校での物理学の授業では、実験が重要である。Holubova [1] は「実験室での作業は理論的な概念を検証する最も重要な方法であり、生徒は実験を通してどのような現象が起きるかを確認することができる」と述べている。

しかし実際は、生徒全員が実験を経験しているわけではない。林らは、2014 年に大学生を対象に物理実験の経験を調査した [2]。これによると、力学分野で最も基本的な「運動の法則」に関する実験経験は 60%であった。また、斜方投射の基本的な問題である「モンキーハンティング (図 1.1)」に関する実験は 10%に満たない。理由としては、実験用の装置の準備や測定が難しいことや、実験を行うのに時間を要することが考えられる。

小球を、位置 $(0, 0)$ から初速 v_0、 仰角 θ で発射したところ、位置 (l, h) から自由落下してくる物体に衝突した。 $\tan\theta$ の満たすべき条件を求めよ。ただし、重力加速度の大きさを g とする。

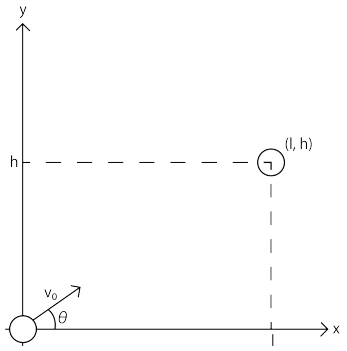


図 1.1: モンキーハンティング

そこで実験の代替として近年利用されているのが、物理実験のシミュレータである。シミュレータを用いることで、実験と同様の学習効果を得ることができる。Ajredini [3] は、既存の物理実験シミュレータである PhET [4] を用いる授業と実際に実験を行う授業を実施し、テストを行った。その結果、シミュレーションによって得られる知識と実際の実験によって得られる知識の間には有意な差はないと結論づけた。

しかし既存のシミュレータでは、学習者は可視化されている運動とその背景に存在する数式の間の関係を理解することは容易ではない。学習者が理論を学習するとき、物理法則やそれを前提とした物体の運動を方程式を通して学習する。図 1.2 は実際に生徒が扱うシチュエーションの例である。学習者は、まず教科書等で物理法則を方程式として学習する。現実の運動を表す際は、学習した物理法則を基に方程式で表現する。一方既存のシミュレータでは、教育者が物理法則から現象を数式で記述し、シミュレーションに変換しており、学習者は可視化されている運動とその背景に存在する数式の間の関係を理解することは容

易ではない。既存のシミュレータである PhET では、図 1.3 のように速度や質量、位置のような物理量を数値でしか確認できず、描画されている物理系がどのような方程式によって表現されているのかわからない。そのため、学習者はシミュレーションを見ることで現実の物理現象を確認することができるが、それと物理法則との関係性は授業などで学ぶのみである。

ボールを x 軸方向の初速 v_{0x}、y 軸方向の初速 v_{0y} で投げる。このとき、ボールがどのような軌道を描くか考える。ただし、重力加速度の大きさを g とする。投げる時刻を $t = 0$ とする。等加速度運動の公式より、時刻 t における位置 x, y は次のように表される:

$$x = v_{0x}t \tag{1.1}$$

$$y = v_{0y}t - \frac{1}{2}gt^2 \tag{1.2}$$

(1.1) より、$t = \dfrac{x}{v_{0x}}$ 。(1.2) に代入して、$y = \dfrac{v_{0y}}{v_{0x}}x - \dfrac{1}{2}\dfrac{g}{v_{0x}^2}x^2 = -\left(x - \dfrac{v_{0y}}{2v_{0x}}\right)^2 + \dfrac{v_{0y}^2}{4v_{0x}^2}$

よって、頂点を $\left(\dfrac{v_{0y}}{2v_{0x}}, \dfrac{v_{0y}^2}{4v_{0x}^2}\right)$ とする放物線を描く。

図 1.2: 学習者が方程式で物理現象を表現する例

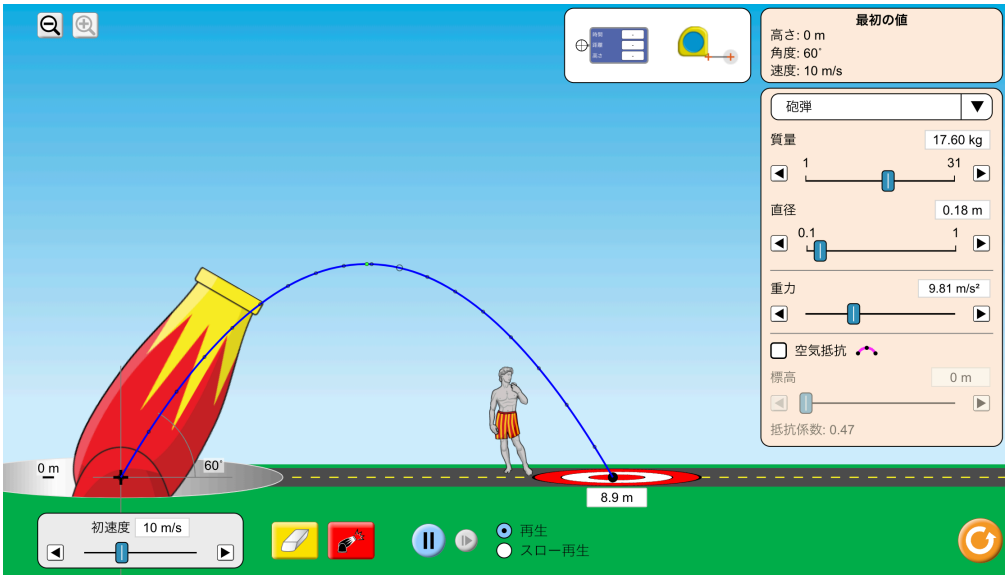


図 1.3: PhET のシミュレーション例

現実の物理現象と物理法則との関係性は、学習者自身が物理法則から現象を数式で記述し、その振る舞いを観測することでよりよく理解できると思われる。誤った定義をすると想定外の動作をし、そこから物理法則や方程式の細部に対する考察を学習者自身が行うことも期待できる。

そこで本研究では、学習者に物理系を定義させるシミュレータ Simulation with Sym-

bols (SimSym) を提案する。学習者は SimSym で系内の物体をそのパラメータとともに定義し、その物体の運動を表す方程式を立式する。シミュレーションを実行すると、定義した物理系に基づいて数値計算がなされ、物体の運動が可視化される。これにより、現実の物理現象と物理法則の間の対応を学習者自らの経験を通して理解することができると考えられる。この際学習者は、SimSym に実装された動作例を参考にすることで、定義した物理系と現実の運動を比較することができたり、次元の異なる物理量の和が存在する不正な方程式を立式すると警告されるなど、正しい物理系を作成するための補助を受ける。

本論文の構成は以下の通りである。第 2 章で、斜方投射を通して既存のシミュレータでの表現について説明する。第 3 章で、SimSym の構成を説明する。第 4 章で、SimSym の実現方法について説明する。第 5 章で、既存のシミュレータを用いた実例について紹介する。第 6 章で、まとめと今後の展望について述べる。

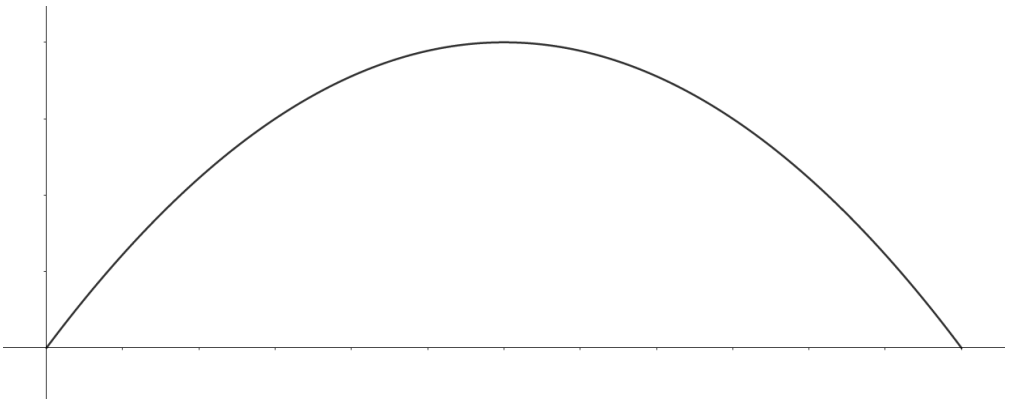
第2章　既存のシミュレータによる表現

本章では、斜方投射の例を通して既存のシミュレータである PhET [4] の表現方法ついて説明する。

2.1　斜方投射

斜方投射は、次のような現象である。

> ある物体を原点から時刻 $t = 0$ に、x 軸方向の初速 v_{0x} 、y 軸方向の初速 v_{0y} で投げる。このとき、重力加速度の大きさを g とすると、物体は次のような軌道を描く。
>
> $$\begin{cases} x = v_{0x}t \\ y = v_{0y}t - \dfrac{1}{2}gt^2 \end{cases} \tag{2.1}$$



2.2　PhET による斜方投射の表現

PhET (Physics Education Technology) は、コロラド大学ボルダー校によるプロジェクトで、物理学の教育に活用できるシミュレーションの作成を目標としている。2023 年 2 月現在、ウェブサイト[1]上では 50 以上のシミュレーションが公開されている。また、物理学のみならず化学・数学・生物学・地球科学などのシミュレーションも公開されている。PhET を用いることで、実際の実験を行うのと同様な教育効果が得られる [3]。

PhET を用いた授業では、以下のような順番で授業を構成することが推奨されている[2]。

1. シナリオを提示する。

[1] https://phet.colorado.edu

[2] https://phet.colorado.edu/files/guides/UG_Phys_Guide-Lecture-Overview_en.pdf

2. 学習者が各自で法則について予想する。

3. 学習者間でディスカッションを行い、予測を修正する。

4. 教育者が学習者の予測や推論を引き出す。

5. 教育者がシミュレーションを用いて実験を行う。

6. 学習者は結果を記録し、予測とどう違うかを記録する。

7. 学習者全体でディスカッションを行う。推論に重点を置く。

図 1.3 は、PhET 上で斜方投射のシミュレーションを行った様子である。大砲から砲弾が射出されるという設定で表現されている。学習者は、初速度や砲弾の質量、仰角などを数値で指定することができ、軌道や着弾点を測定することができる。

このシミュレーションを通して、学習者は斜方投射の運動を観察することができる。一方で、シミュレーションを作成するのは教育者である。そのため、この軌道や着弾点の位置が具体的にどのような方程式に基づいて計算されたものなのかはわからず、式 (2.1) が現実の現象と一致しているという実感は得にくいと考えられる。

第3章　SimSym

本章では、学習者に物理系を定義させるシミュレータである SimSym を紹介する。

3.1　アイデア

SimSym の基本的なアイデアは、現実の物理現象と物理法則の間の対応を学習者自らの経験を通して理解することができるという点である。既存のシミュレータでは、教育者が物理法則から現象を数式で記述し、シミュレーションに変換する。学習者はシミュレーションを見ることで現実の物理現象を確認することができるが、それと物理法則との関係性は授業などで学ぶのみである (図 3.1)。SimSym では、物理法則から現象を数式で記述し、シミュレーションに変換するまでの工程も学習者が行う。これにより、従来は授業などで教わる必要があった現実の物理現象と物理法則の間の対応を学習者自らの経験を通して理解することができると考えられる (図 3.2)。

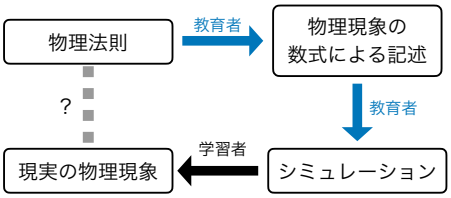


図 3.1: 既存のシミュレータによる理解

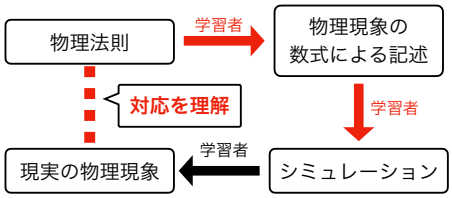


図 3.2: SimSym による理解

3.2　画面構成

図 3.3 は、SimSym 上で斜方投射のシミュレーションを作成した例である。SimSym の画面は、図 3.3 のように左半分の物理系定義ペインと右半分の観測ペインに分かれている。物理系定義ペインで物体の作成や方程式の定義を行い、観測ペインでシミュレーションの結果を確認する。

3.3　使用の流れ

学習者は、以下に示す動作を順番に行うことで、物理系を定義しシミュレーションを作成することができる。

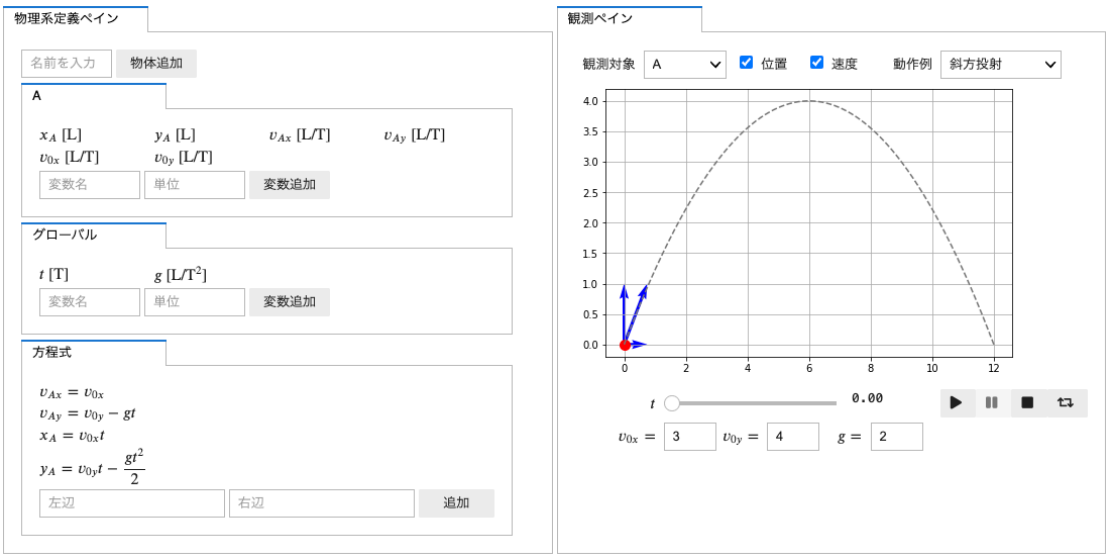


図 3.3: SimSym 上で斜方投射を表現した例

1. **物体名の入力**
   学習者が物体名を入力すると、SimSym がその物体を生成する。同時に、その物体に紐付いた物理量に対応する次元付き変数 (m_A[M], x_A[L] など[1]) を生成する。

2. **動作例の選択 (オプション)**
   学習者は、作成した物体を選択した上で、現実の運動を正しく表現した動作例を選択することができる。動作例を選択した場合、その動作に必要な初期値を表す変数が物体に紐づく形で生成される。

3. **方程式の立式**
   学習者は、1. と 2. で生成された変数を使って方程式を立式する。この際、物体に紐付く次元付き変数と、どの物体にも紐付かないグローバルな次元付き変数を自由に追加することができる。なお、時刻を表す変数 t[T] や重力加速度 g[L/T^2] などの物理定数はグローバルな変数として用意されている。

4. **観測対象の指定**
   学習者は観測したい物体とその物理量 (位置・速度・加速度) を指定する。物体の選択はドロップダウンリストで、物理量の選択はチェックボックスで行う。

5. **初期値の入力**
   観測対象を指定した際に、その値を計算するために必要な変数が観測ペイン下部に表示される。描画の際には観測対象の数値的な情報が必要なので、これらに値を入力する必要がある。また、動作例の選択によって生成された変数に初期値を入力すると、その初期値に基づく動作例が破線で表示される。

6. **シミュレーションの再生**
   シミュレーションを再生すると、時刻 t が変化しながら観測対象の物理量が描画さ

[1] M: Mass(質量), L: Length(長さ), T: Time(時間) などを次元と呼ぶ。例えば速度の次元は [L/T] と表される。

れる。物体の位置は座標平面上の位置として、速度と加速度はベクトルとして描画される。またこの際、動作例と学習者が定義した物理系の運動を比較することで、現実と同じ運動を表現しているか確認することができる。

3.4　具体例: 斜方投射

以下では具体的な例として斜方投射を用い、 SimSym の動作を見ていく。図 3.3 は、SimSym 上で x 軸方向の初速が v_{0x}, y 軸方向の初速が v_{0y}, 重力加速度の大きさが g であるような斜方投射を表現した例である。

1. 物体 A を作成すると、物体 A の位置を表す変数 x_A, y_A と、速度を表す変数 v_{Ax}, v_{Ay} が生成される。
2. 観測対象として A を選択し、動作例として斜方投射を選択すると、初速を表す変数 v_{0x}, v_{0y} が物体 A に紐付く変数として追加される。
3. 1. と 2. で用意した変数を用いて以下の方程式を立式する。
   - $v_{Ax} = v_{0x}$
   - $v_{Ay} = v_{0y} - gt$
   - $x_A = v_{0x}t$
   - $y_A = v_{0y}t - \dfrac{1}{2}gt^2$
4. v_{Ax}, v_{Ay}, x_A, y_A を計算するのに必要な g, v_{0x}, v_{0y} に値を入力する。
5. シミュレーションを再生すると、位置と速度が描画される。この際、破線で表示された正しい軌道と比較することができる。

3.4.1　誤りに気づくことができる例

以下では、SimSym を用いることで物理系を誤って定義してしまったことに気づくことができる例を紹介する。

方程式のミス

重力加速度の向きを間違え、

- $v_{Ax} = v_{0x}$
- $v_{Ay} = v_{0y} + gt$
- $x_A = v_{0x}t$
- $y_A = v_{0y}t + \dfrac{1}{2}gt^2$

のように定義すると、図 3.4 の実線のような軌道を描いて運動するが、これは動作例の破線と大幅に異なる。そのため、これは現実の運動を正しく表せていないことがわかる。

次元の不一致

一般に、次元の異なる値同士の和や差を求めることはできない。そのため、方程式の立式の際に $v_{0x} + t$ ([L/T] + [T]) のように次元の一致していない式を定義しようとすると、図 3.5 のように警告される。

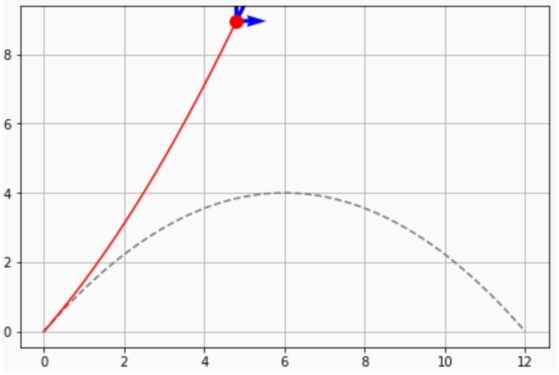


図 3.4: 誤った運動の定義例

$$v_{0x} + t$$

次元が不正なため定義できません:
[L/T] + [T]

図 3.5: 誤った次元の定義例

第4章　実現

SimSym は Python を用いて実現されている。入力と表示に Jupyter Notebook [5] 、グラフの描画に matplotlib [6] 、方程式の処理や数値計算に SymPy [7] を用いている。

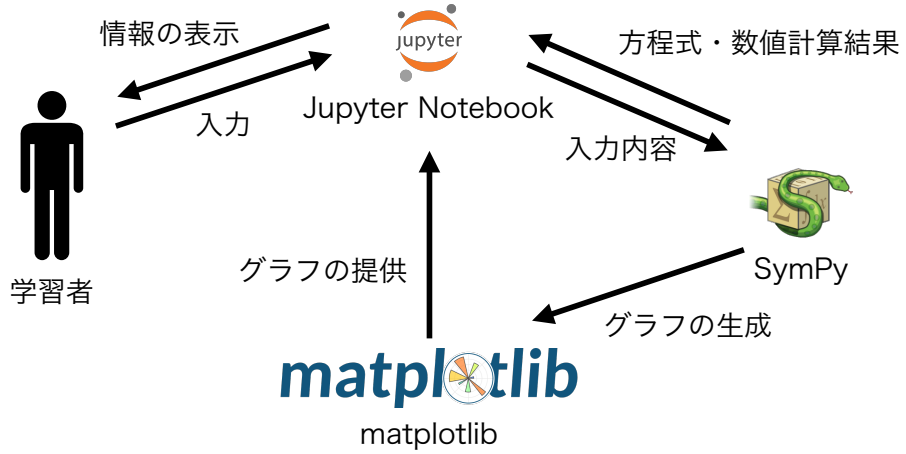


図 4.1: 簡単な全体像

4.1　使用している道具

4.1.1　Jupyter Notebook

Jupyter Notebook は IPython [8] から分離したプロジェクトで、Jupyter Notebook から IPython カーネルを呼び出すことで、Python のコードを対話的に実行することができる。また、ipywidgets ライブラリを利用すると、Jupyter Notebook 上に GUI を作成することができる。図 4.2 は、Jupyter Notebook 上にクリックすることのできる Button と、文字を入力できる Text の Widget を生成する例である。

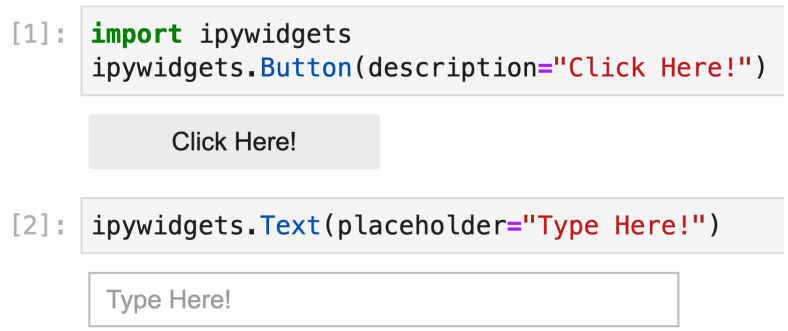


図 4.2: Jupyter Notebook 上に Button と Text の Widget を生成する例

4.1.2　matplotlib

matplotlib は、グラフを描画するためのライブラリである。Jupyter Notebook 上で matplotlib を用いると、グラフを Jupyter Notebook に表示することができる。また、出力先として ipywidgets の Output を用いることで、ipywidgets で作成した GUI にグラフを組み込むことができる。図 4.3 は matplotlib を用いて簡単なグラフを描画する例である。

[1]:
```
import matplotlib.pyplot as plt
x = [i / 100 for i in range(400)]
y = [t**2 for t in x]
plt.plot(x, y)
plt.show()
```

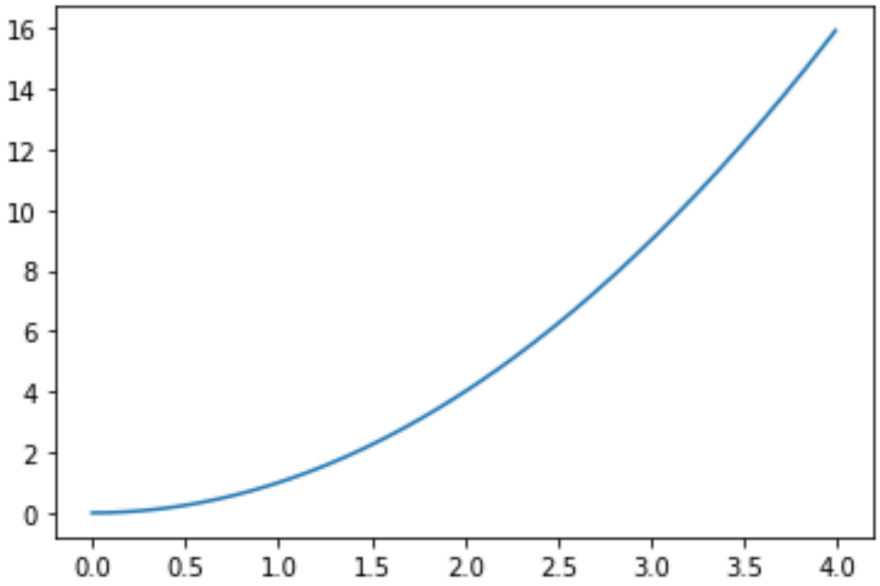


図 4.3: matplotlib でグラフを描画する例

4.1.3　SymPy

SymPy は、記号計算のための Python ライブラリである。Symbol オブジェクトとして変数を作成し、Eq オブジェクトとして方程式を定義できる。subs 関数を使うことで、作成した方程式に数値を代入することができる。図 4.4 は、SymPy で Equation を定義し、数値を代入する例である。

```
[1]: from sympy import Eq, symbols, solve
     v, v0, a, t = symbols('v v_0 a t')
     eq = Eq(v, v0 + a * t)
     eq
```

[1]: $v = at + v_0$

```
[2]: eq.subs({v0:1, a: 2, t: 3})
```

[2]: $v = 7$

図 4.4: SymPy で Equation を定義し、数値を代入する例

4.2　処理の流れ

以下では、学習者の操作によって SimSym がどのように物理系を定義しているかを簡単に説明する。

1. 学習者が Jupyter 上の Widget に物体名を入力すると、SimSym はその物体に紐づく変数を SymPy の Symbol として生成する。生成された変数は Jupyter 上に表示される。
2. 学習者が物体と動作例を選択すると、SimSym は動作例に応じた変数を Symbol として生成し、動作例の運動を表す Sympy の Equation を生成する。
3. 生成された変数を利用した方程式を学習者が Jupyter 上の Widget に入力すると、SimSym はそれを Equation として解釈する。またこの際、不正な次元の方程式になっていないかを確認する。
4. SimSym は、選択された物体の座標・速度などを数値計算するために数値を与える必要のある変数を推論し、それらを入力するための Widget を生成する。
5. 学習者が数値を入力すると、Equation に数値が代入される。同時に動作例を表す Equation にも数値が代入され、matplotlib を用いて破線で描画される。
6. 学習者が再生ボタンを押すと、時刻を表す変数 t が時間変化し、物体の位置や座標が再計算され、matplotlib を用いて描画される。

4.2.1　実装できていない部分

先述した SimSym の機能のうち、

- 学習者の入力を Equation として解釈する (3.)
- 不正な次元の方程式になっていないかを確認する (3.)
- 数値計算するために数値を与える必要のある変数を推論する (4.)

という部分が未実装である。それぞれについて方針を述べる。

学習者の入力を Equation として解釈する

入力の解釈は、sympy.parsing.sympy_parser の parse_expr 関数を使うことで可能である。この関数では、文字列を SymPy の Symbol, Sum, Mul や Eq などに変換できる。実際に行う例を図 4.5 に示す。一方、単にこれを用いるだけでは未定義の変数を使うことができてしまう。そのため、解釈された変数が定義されたものなのかを確認する必要がある。また、学習者がアンダースコアなどを入力する必要があったり、意図しない解釈をされる (v_Ax と v_{Ax} など) 可能性がある。Jupyter に表示されている変数をクリックすることでその変数を入力することができれば、このような誤りは防止でき、学習者の入力にかかる手間も減らすことができる。

[1]:
```
from sympy.parsing.sympy_parser import parse_expr
eq = parse_expr('v=v_0 + at', transformations='all')
eq
```

[1]: $v = at + v_0$

[2]:
```
eq.subs({'v_0':1, 'a': 2, 't': 3})
```

[2]: $v = 7$

図 4.5: parse_expr 関数で文字列をパースする例

不正な次元の方程式になっていないかを確認する

現在の実装では、 $g + t$ ($[\mathrm{L/T^2}] + [\mathrm{T}]$) のように次元が不正な方程式も定義できてしまう。SymPy の変数に次元の情報を付加し、異なる次元間の加減などを検出する必要がある。自動で追加される変数は生成する際に次元の情報を付加すればよいが、学習者が新たに追加する変数については次元を手動で指定する必要がある。L, M, T などの組み合わせを文字列で入力したものを解釈する、各次元の指数を数値で入力したものを解釈するなどの方法が考えられる。

数値計算するために数値を与える必要のある変数を推論する

基本的には、学習者が定義した方程式に含まれる変数のうち物体に紐づいている位置・速度以外の変数の値を入力させれば良い。しかし、次のような場合も存在する。

$$x_A = X$$
$$X = v_0 t$$

ここで、x_A は物体の x 座標を表す変数で、X と v_0 は学習者が定義した変数である。先述した方針では、X と v_0 に値を入力する必要がある。しかし実際は X か v_0 の一方にのみ値を入力すればよい。このような場合に対応する方法を考える必要がある。

第5章　関連研究

この章では、既存のシミュレータを用いた教育の実例について紹介する。

5.1　PhETの教育効果

Ajrediniの 実験

Ajredini [3] は、マケドニアの高校生に PhET を用いて電荷について教える実験を行った。実際の実験を行うグループ、PhET のシミュレーションを利用するグループ、実験を行わないグループの 3 つに分け、電荷に関する授業を行った。その後、いくつかのシチュエーションを指定し、スケッチを描かせた。各シチュエーションと正答率は表 5.1 の通りであり、実際の実験と PhET との間の有意差は見られなかった。

---

2 つの軽い中性の金属球を糸でぶら下げる。次のような場合についてスケッチを描け。

a). 両方の球体に毛布で擦ったプラスチック棒を接触させ、帯電させる

b). (a) の状態で、球体をより遠くに置く

c). 球体 A は毛布で擦ったプラスチック棒で帯電させ、球体 B は皮で擦ったガラス棒で帯電させる。

d). 両方の球体をプラスチック棒で帯電させるが、球体 A は球体 B より多く帯電させる。

e). 球体 A はプラスチック棒で帯電させ、球体 B は中性のままにする

f). 球体 A はガラス棒で帯電させ、球体 B は中性のままにする

グループ	実験	PhET	座学
a)	75	66	43
b)	35	34	14
c)	67	70	31
d)	23	26	12
e) and f)	11	7	3

表 5.1: Ajredini による実験の結果

Prima の実験

Prima [9] は、インドネシアの中学生に PhET を用いて太陽系について教える実験を行った。PhET を用いるグループと用いないグループに分け授業を行い、その結果を Normalized Gain を用いて評価している。満点を 100 とする pre-test と post-test の平均点をそれぞれ ⟨pre-test⟩, ⟨post-test⟩ とすると、 Normalized Gain $\langle g \rangle$ は以下のように求められる:

$$\langle g \rangle = \frac{\langle \text{post-test} \rangle - \langle \text{pre-test} \rangle}{100 - \langle \text{pre-test} \rangle}$$

また、テストは Bloom's Taxonomy に基づき Remembering, Understanding, Applying, Analyzing の 4 領域で行われた。結果は表 5.2 の通りであり、全ての分野において PhET を用いた方が Normalized Gain が大きい。

Cognitive level	Control Group			Experiment Group		
	pretest	posttest	N-Gain	pretest	posttest	N-Gain
Remembering (C1)	33.33	76.19	0.64	44.44	84.12	0.71
Understanding (C2)	38.62	57.67	0.31	51.85	75.66	0.49
Applying (C3)	54.76	80.95	0.57	69.04	92.85	0.76
Analyzing (C4)	56.34	75.39	0.43	57.14	80.15	0.53

表 5.2: Prima による実験の結果

Rehman の実験

Rehman [10] は、パキスタンの高校生に PhET を用いて重さと質量の概念について教える実験を行った。また、この授業は週 5 回・1ヶ月間行われた。結果は表 5.3 の通りであり、有意差が認められる。

	Control Group	Experiment Group
pretest	11.82	12.68
posttest	16.84	29.46

表 5.3: Rehman による実験の結果

5.2　Excel を用いる例

Uddin ら [11] は、Microsoft Excel を用いて物理系を定義・可視化する手法を提案した。この手法では、自由に方程式を定義することが可能であるが、単位の異なる足し算などの不正な計算をしてしまっても気づくことができない。また、Excel 上で表現された数式は実際の数式と視覚的に遠い。例えば、風速の分布を描画する例でワイブル分布 $f(t) = \frac{m}{\eta}\left(\frac{t}{\eta}\right)^{m-1} \exp\left\{-\left(\frac{t}{\eta}\right)^{m}\right\}$ を用いているが、Excel 上でのこの関数の表現は ((H$1/H$2) * ((C3/H$2)^(H$1-1)) * EXP(-((C3/H$2)^H$1))) となっている。そのため、これを見てもワイブル分布であるとはわかりにくい。

5.3　Easy Java Simulations

Easy Java Simulations [12] は、ユーザが入力した数式をもとにシミュレーションを生成してくれる Java のソフトウェアである。シミュレーションの科学的な側面にほとんどの時間を集中させ、その他の必要なタスクをコンピュータに自動的に行わせるように設計されている。Easy Java Simulations は、ユーザ自身がシミュレーションを作成できるという点で SimSym と類似している。 SimSym には動作例が存在し、次元のチェックを行うため、学習者が物理系を作成した際により誤りに気づきやすいと考えられる。

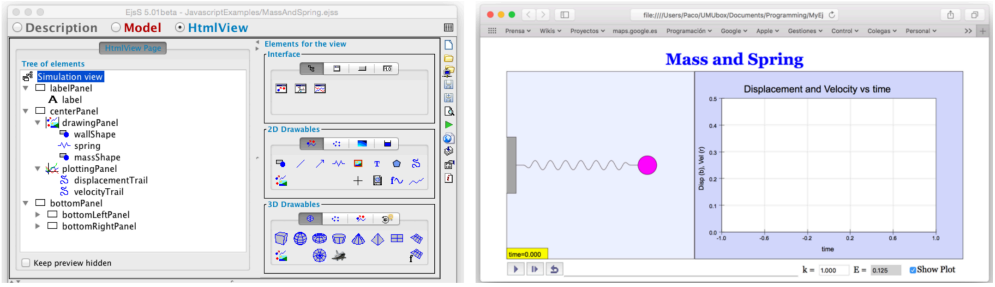


図 5.1: Easy Java Simulations で作成したシミュレーションの例 [13]

第6章　まとめと課題

6.1　まとめ

本研究では、学習者に物理系を定義させるシミュレータである SimSym (Simulation with Symbols) を提案した。学習者が物理現象を表す方程式を SimSym に入力すると、SimSym はその方程式に基づく物体の運動を可視化する。学習者は SimSym に実装された動作例を参考にすることで、定義した物理系と現実の運動を比較することができたり、次元の異なる物理量の和が存在する不正な方程式を立式すると警告されるなど、正しい物理系を作成するための補助を受ける。SimSym では、物理法則から現象を数式で記述し、シミュレーションに変換するまでの工程も学習者が行うため、従来は授業などで教わる必要があった現実の物理現象と物理法則の間の対応を学習者自らの経験を通して理解することができると考えられる。

6.2　今後の課題

6.2.1　評価

SimSym の教育効果を評価する必要がある。高校生を、SimSym を用いるグループ・PhET を用いるグループ・座学のグループに分け、pretest と posttest を行う。その結果を Hake [14] が導入した Normalized Gain を用いて比較することで、SimSym の教育効果を測定できると考える。

また、SimSym の利点は、現実の運動を確認できるだけでなく、現実の物理現象と物理法則の間の対応を学習者自らの経験を通して理解することができるという点である。そのため、各グループでディスカッションを行い、その内容からどれだけ現実の物理現象を正しく理解できているかを確認することも有効であると考える。

6.2.2　方程式計算の支援

現在、SimSym 上に方程式を入力する際、その方程式の導出は学習者が行う必要がある。しかし、移項した際の符号の変え忘れや係数の間違いなど、SimSym では検出できないミスが存在する。この作業を以下のような機能を加えてサポートしたい。

求解の自動化

$mgh = \frac{1}{2}mv_{Ax}^2$ などの入力から自動で $v_{Ax} = \sqrt{2gh}$ と式変形を行うことができれば、計算ミスを防ぐことができる。これは、SymPy の solve 関数などを利用して実装することができる。ただし、自動の式変形では対応できない内容もある。例えば、$mgh = \frac{1}{2}mv_{Ax}^2$

を v_{Ax} に関して解いた解は $v_{Ax} = \pm\sqrt{2gh}$ であり、正の場合と負の場合が存在する。このようなときに、学習者に選択させる必要がある。また、式変形を全て自動化してしまうと、学習者自身の式変形に関する能力が成長しない。そのため、自動で求解するかどうかを切り替えられるようにする。

基本的な公式 (運動方程式、力学的エネルギー保存則等) の提供

基本的な公式を提供し、各値に変数を割り当てるだけ方程式を定義することができるようにする。これにより、学習者の負担を大きく軽減することができる。先述した自動求解を組み合わせることで、適切な公式を選び適切に変数を割り当てるだけで物理系が作成できる。また、公式を解説付きで一覧にしたり、公式の検索機能をつけることで、公式を覚えきれてない初学者に対してもサポートができる。

6.2.3　場への対応

重力場・電場・磁場などのベクトル場や、ポテンシャルなどのスカラー場の可視化に対応することを考えている。場を可視化することができれば、万有引力の法則やクーロンの法則などに従う物体の運動をよりよく表現することができる。図 6.1 は、matplotlib を用いてベクトル場を可視化する例である。学習者にベクトル場を表す関数を定義させることも可能だが、学習者が生成した物体から学習者が指定した場を自動的に生成し描画することで、より手軽に利用することができる。

[1]:
```
import numpy as np
import matplotlib.pyplot as plt

X, Y = np.meshgrid(np.arange(-2.2, 2.2, 0.4), np.arange(-2.2, 2.2, 0.4))
U = X/(X**2 + Y**2)
V = Y/(X**2 + Y**2)

plt.quiver(X, Y, U, V, color='red', angles='xy', scale_units='xy', scale=5.0)
plt.xlim([-2,2])
plt.ylim([-2,2])

plt.plot(0, 0, 'o')
plt.draw()
```

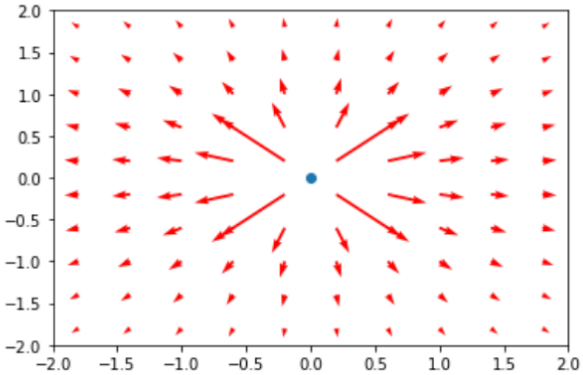


図 6.1: ベクトル場の可視化例

6.2.4　三次元空間への対応

現在は二次元のグラフで描画を行なっているが、電磁気的な力を受ける場合などは三次元的な運動をする事が多いため、三次元空間を扱う必要がある。方程式の定義や数値計算は単純に拡張すれば良いが、現在の matplotlib を用いた可視化では三次元空間に対応することは容易ではないため、可視化手法を検討する。

参考文献

[1] Renata Holubova. The impact of experiments in physics lessons – “why, when, how often?”. *AIP Conference Proceedings*, Vol. 2152, No. 1, p. 030007, 2019.

[2] 林 壮一, 川村 康文, 村上 聡. 大学生に対する高校物理実験および放射線学習の現状調査. 物理教育, Vol. 63, No. 3, pp. 191–196, 2015.

[3] Fadil Ajredini, Neset Izairi, and Oliver Zajkov. Real Experiments versus Phet Simulations for Better High-School Students’ Understanding of Electrostatic Charging. *European Journal Of Physics Education*, Vol. 5, No. 1, p. 59, February 2014.

[4] Katherine Perkins, Wendy Adams, Michael Dubson, Noah Finkelstein, Sam Reid, Carl Wieman, and Ron LeMaster. PhET: Interactive Simulations for Teaching and Learning Physics. *The Physics Teacher*, Vol. 44, No. 1, pp. 18–23, January 2006.

[5] Thomas Kluyver, Benjamin Ragan-Kelley, Fernando Pérez, Matthias Bussonnier, Jonathan Frederic, Jessica Hamrick, Jason Grout, Sylvain Corlay, Paul Ivanov, Safia Abdalla, and Carol Willing. Jupyter Notebooks—a publishing format for reproducible computational workflows.

[6] John D. Hunter. Matplotlib: A 2D Graphics Environment. *Computing in Science & Engineering*, Vol. 9, No. 03, pp. 90–95, May 2007. Publisher: IEEE Computer Society.

[7] Aaron Meurer, Christopher P. Smith, Mateusz Paprocki, Ondřej Čertík, Sergey B. Kirpichev, Matthew Rocklin, AMiT Kumar, Sergiu Ivanov, Jason K. Moore, Sartaj Singh, Thilina Rathnayake, Sean Vig, Brian E. Granger, Richard P. Muller, Francesco Bonazzi, Harsh Gupta, Shivam Vats, Fredrik Johansson, Fabian Pedregosa, Matthew J. Curry, Andy R. Terrel, Štěpán Roučka, Ashutosh Saboo, Isuru Fernando, Sumith Kulal, Robert Cimrman, and Anthony Scopatz. SymPy: symbolic computing in Python. *PeerJ Computer Science*, Vol. 3, p. e103, January 2017. Publisher: PeerJ Inc.

[8] Fernando Perez and Brian E. Granger. IPython: A System for Interactive Scientific Computing. *Computing in Science & Engineering*, Vol. 9, No. 3, pp. 21–29, May 2007. Conference Name: Computing in Science & Engineering.

[9] Eka Cahya Prima, Aldia Ridwani Putri, and Nuryani Rustaman. Learning solar system using PhET simulation to improve students’ understanding and motivation. *Journal of Science Learning*, Vol. 1, No. 2, p. 60, March 2018.

[10] Nadia Rehman, Wanlan Zhang, Amir Mahmood, and Faiz Alam. Teaching physics with interactive computer simulation at secondary level. *Cadernos de Educação Tecnologia e Sociedade*, Vol. 14, No. 1, p. 127, March 2021.

[11] Zaheer Uddin, Muhammad Ahsanuddin, and Danish Ahmed Khan. Teaching physics using Microsoft Excel. *Physics Education*, Vol. 52, No. 5, p. 053001, July 2017. Publisher: IOP Publishing.

[12] Wolfgang Christian and Francisco Esquembre. Modeling Physics with Easy Java Simulations. *The Physics Teacher*, Vol. 45, No. 8, pp. 475–480, November 2007.

[13] Francisco Esquembre, Félix J. García Clemente, Rafael Chicón, Lawrence Wee, Leong Tze Kwang, and Darren Tan. Easy Java/JavaScript Simulations as a tool for Learning Analytics, October 2019. arXiv:1910.09156 [physics].

[14] Richard Hake. Interactive-engagement versus traditional methods: A six-thousand-student survey of mechanics test data for introductory physics courses. *American Journal of Physics - AMER J PHYS*, Vol. 66, , 01 1998.