物理学の学習のためのプログラマブルな シミュレータと環境の提案

東京工業大学 情報理工学院 数理・計算科学系
18B04657 木内康介
指導教員 増原英彦教授

1　はじめに

高等学校における物理学の授業において、実験は重要である。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。力学分野における最も基本的な「運動の法則」に関する実験の経験は、2014 年の調査時点で 60% 程度にしか満たない [2]。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。

そこで実験の代替として近年利用されているのが、シミュレーションである。シミュレーションを用いることで、実験と同様な学習効果を得ることができる [3]。

しかし、既存のシミュレータで確認できる情報と、実際に生徒が解く問題の回答との間のギャップは大きい。例えば PhET [4] は、図 1 のようにそれぞれの値や計算結果を数値として表現している。一方、生徒が解くことになる物理の問題は図 2 のように文字を利用した方程式の計算が主流である。そのため、既存のシミュレータでは生徒が求めた方程式がどのような運動を表現しているのか直接確認することができない。

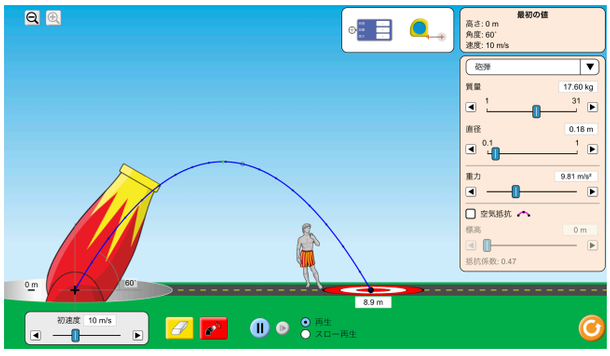

図 1　PhET のシミュレーション例

そこで本研究では、物体の運動を方程式で定義できるシミュレータである SimSym (Simulation with Symbols) を提案する。SimSym では、定義した方程式に従って数値計算がなされ、シミュレーションが実行され、物体の運動が可視化される。これにより、導出した方程式がどのような運動と対応するか簡単に確認することができる。

$$\begin{cases} ma = F & (1) \\ v = v_0 + at & (2) \end{cases}$$

$$(1) \text{ より}, \quad a = \frac{F}{m}$$

$$(2) \text{ に代入して}, \quad v = v_0 + \frac{F}{m}t$$

図 2　方程式の計算例

2　SimSym

SimSym は、ブラウザ上で実行できるシミュレータである。ユーザは、SimSym 上で物体と方程式を作り、それらを紐付ける。シミュレーションを実行すると、時間経過に応じて物体の位置が計算され、ブラウザ上に描画される。

まず、物体の名前を入力し物体作成ボタンを押すと、その物体と最低限必要なフィールド (x 座標・y 座標) が生成される。次に、使いたい定数・変数を追加し、方程式を定義する。この際、定数及び変数にはシミュレーションの計算に用いる初期値を設定する必要がある。なお、時刻を表す変数 t は最初から用意されている。その後、定義した定数・変数を使って方程式を定義する。定義した方程式を物体のフィールドにドラッグすると、そのフィールドに方程式を紐付けることができる。その状態で再生すると、時刻 t が変化しながら各方程式が計算され、方程式に基づいて物体が運動する。

以下では具体例を見ていく。図 3 は、SimSym 上で斜方投射を表現した例である。これを実行すると、t が変化し、各方程式が定数の初期値に従って計算される。この例では y 座標に紐付けている方程式が $v_{0y}t - \frac{gt^2}{2}$ となっていて、図 4 のような軌道を描く。これは、大まかに現実世界で物体を投げたときの運動と一致する。一方、例えば y 座標に紐付ける方程式を $v_{0y}t + \frac{gt^2}{2}$ とすると、図 5 のような軌道を描き、現実世界の運動、即ち想定していた運動とは異なる

ことがわかる。このように、方程式が表している運動をシミュレーションで確認することが可能である。

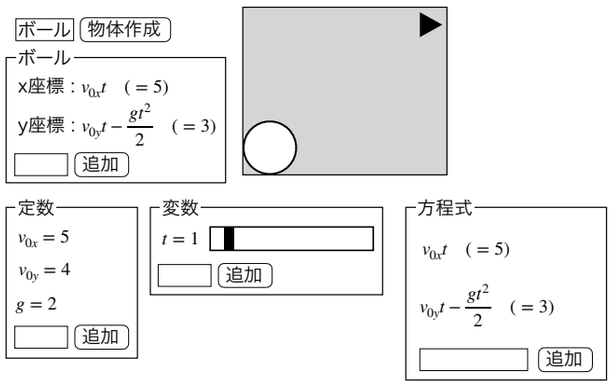

図 3　SimSym 上で斜方投射を表現した例

図 4　斜方投射の
”妥当な”軌道

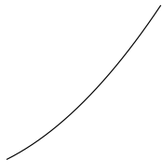

図 5　斜方投射の
”誤った”軌道

3　実装

実装は、フロントエンドに lively.next[*1] を、文字式の計算に SymPy [5] を用いた。SymPy は Python のライブラリであるが、WebAssembly で実装された CPython 処理系の Pyodide[*2] を用いることでブラウザ上で完結させた。

lively.next

lively.next は、JavaScript で記述されたコンポーネントを組み合わせることで GUI アプリケーションを作成することができる Web プログラミング環境である。ユーザが物体や定数、方程式を追加するインターフェースとシミュレーションを表示する画面は lively.next で作成した。

SymPy

SymPy は、方程式の定義や数値代入のための Python ライブラリである。lively.next に入力された内容をもとに、SymPy で方程式を定義し、数値計算を行っている。Python ライブラリであるが、 Pyodide を用いることでブラウザ上で実行させた。

3.1　Pyodide

Pyodide は、WebAssembly で実装された CPython 処理系である。ブラウザ上で Python 並びにいくつかのパッケージを実行することができる。これを用いて、lively.next から SymPy を呼び出して計算することをブラウザ上だけで完結させることができた。

[*1] https://lively-next.org
[*2] https://pyodide.org/en/stable/

4　まとめと課題

本研究では文字式を基にしたシミュレーションが可能であるシミュレータ SimSym を提案した。SimSym を用いることで、生徒は自身が導出した文字式がどのような動きと対応するか確認することができる。結果として、物理学に対するより直感的・本質的な理解が期待できる。

今後の課題として、SimSym の教育効果の評価がある。評価手法として、Hake [6] が導入した normalized gain を用いた実験を検討する。

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 壮一ほか. 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, et al. 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 K. Perkins, et al. PhET: Interactive Simulations for Teaching and Learning Physics. *The Physics Teacher*, Vol. 44, No. 1, pp. 18–23, January 2006.

[5] Aaron Meurer, et al. SymPy: symbolic computing in Python. *PeerJ Computer Science*, Vol. 3, p. e103, January 2017. Publisher: PeerJ Inc.

[6] 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.