学習者自身が物理現象をモデル化するシミュレータ SimSym の提案

木内 康介　増原 英彦　叢 悠悠

対話的なデジタル教材が多く開発され、中学校・高等学校などで活用されている。中でも物理シミュレータは、実験器具等の準備が不要であり、学習者が自由にパラメータを変更しながら実行できるといった点で有用である。物理の学習では物理現象を観測し、その現象を数式として理解するが、既存の物理シミュレータには数式による理解を支援する機能はほとんどない。本研究は、単純な物理現象は時刻と物体の位置の関係式で決定される点に注目し、学習者自身が数式を用いて物理現象を計算機上でモデル化する学習手法と、そのためのソフトウェア SimSym を提案する。SimSym 上で学習者は、教育者が用意したモデルをシミュレートすること、それを観察すること、またモデルを作成しシミュレートすることができる。現在は、簡単な力学現象を表す関数を入力すると、次元に関する検査を行い、二次元平面上のグラフによる可視化を行うシステムが Jupyter Notebook 上のライブラリとして実現されている。

1　はじめに

物理学は、実験と理論の両輪で成立している。現実世界の物理現象は理論によって解釈され、その理論の正当性は実験によって検証される。これは、物理学の学習・教育においても同様である。生徒は実験を通してどのような現象が起こるかを確認できる [2]。また、物理現象を表した数式の意味を学ぶことで、学習者の物理概念の理解を高めることができる [4]。

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 [9]。理由として、実験準備のわずらわしさや予算不足などがある [8]。

それらの課題点を解消できるのが、物理シミュレータである。物理シミュレータは、現実世界における物理現象を計算機上で再現するソフトウェアである。物理シミュレータは、実験と比較して以下の 3 つの特徴がある。

- 実験器具が不要であるため、準備に手間がかからない。
- 学習者が物理現象を特徴付けるパラメータを自由に変更して実行できる。
- 物理シミュレータを用いる授業では、実際に実験を行う授業と同程度の知識が得られる [1]。

このように、物理シミュレータを活用することで、物理学の実験的側面を効果的に学習できる。

その一方で、既存の物理シミュレータは理論的側面に関しては扱うことができない。計算機の数式処理能力を活用することで、理論的側面に関しても物理シミュレータ上で扱えるはずだが、既存の物理シミュレータは物理実験を再現することに注力しているためであると考える。

そこで我々は、物理現象と数式との間の関係性をよりはっきりさせるように物理シミュレータを拡張することを提案する。これによって、学習者が物理現象を数式によって理解することを促進できると期待する。具体的には、学習者自身が数式を用いて物理現象を計算機上でモデル化する学習手法と、そのためのソフトウェア SimSym を提案する。単純な物理現象は時刻と物体の位置の関係式で決定される点に注目し SimSym を設計した。その結果、学習者は自国と物体の位置の関係式を入力するだけでモデル化できる。

本稿では、2 節で既存の物理シミュレータとそれを

Proposal of the SimSym Simulator: Allowing Learners to Model Physical Phenomena Themselves

木内 康介　増原 英彦　叢 悠悠, 東京工業大学, Tokyo Institute of Technology.

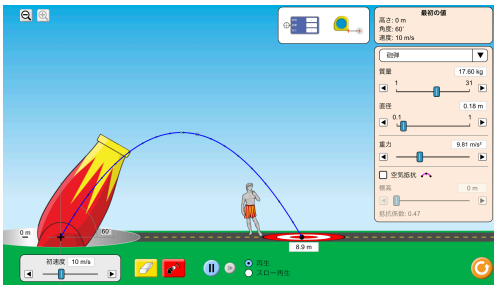

図 1　PhET のスクリーンショット

用いた学習手法紹介する。3 節で学習者自身が物理現象をモデル化する学習手法を提案し、4 節でその手法を実現するソフトウェア SimSym の UI 設計と使用方法を説明する。5 節で SimSym の実現方式について述べる。最後に 6 節でまとめと今後の課題を述べる。

2　既存の物理シミュレータ

既存の物理シミュレータは多数存在するが、その中でも PhET [7] は特に利用が広がっており、実際に授業で利用した調査報告も多数存在する。また、筆者が調査した範囲では、物理現象の数式による理解を促進する機能を持つシミュレータは PhET を含めても見当たらなかった。そのため、今回は代表として PhET を紹介する。

PhET は、物理学の教育に活用できるシミュレーションを作成するためのフレームワークである。いくつもの物理現象に対応するシミュレーションが用意されており、ウェブサイトで実行できる †1。利用者は、設定されたシナリオ内でパラメータを変更しながらシミュレーションが実行できる。

図 1 は PhET 上で斜方投射のシミュレーションを行う様子である。大砲から砲弾が射出されるという物理現象が表現されている。学習者は、初速度や砲弾の質量、仰角などを数値で指定でき、軌道や着弾点を測定できる。

PhET を用いる学習手法

PhET を用いた授業では、おおまかに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。具体例とともに紹介する。

1. シナリオを提示する。
   例:「物体を斜め上に投げたとき、物体はどのような動きをするだろうか。」
2. 学習者が各自で実験結果を予測する。
   例:「物体は放物線を描く。」
3. 教育者がシミュレーションを用いて実験を行う。
4. 学習者は結果を記録し、予測とどう違うかを記録する。
5. 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。
   例:「水平方向の速度は一定であり、垂直方向の速度は $v = v_0 - gt$ という形の式で表される。」

PhET を用いる学習手法では、学習者は与えられたシナリオで起きる物理現象を予測し、教育者が行うシミュレーションを観測する。この際、実行されるシミュレーションはシミュレータの作成者があらかじめ作成したモデルに基づく。

3　学習者が物理現象をモデル化する学習手法

本節では、PhET を用いる学習手法で実現できていない課題をあげ、本研究で提案する学習者自身が物理現象をモデル化する学習手法とその利点を述べる。

3.1　PhET を用いる学習手法における未実現の課題

PhET を用いた学習手法では、物理学の実験的側面は学習できるが、理論的側面の学習に関してのサポートは存在しない。例えば、推論結果として $v = v_0 - gt$ という数式が得られたとしても、その数式がどのような振る舞いをするかは確認できない。したがって、誤った数式を推論したとしても、その振る舞いから誤りを見つけることはできない。

†1　https://phet.colorado.edu/ja/simulations

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

3.2　本研究で提案する学習手法

本研究では、実験的側面と理論的側面を結びつけることのできる手法として、学習者自身が数式を用いてモデル化を行うという学習手法を提案する。学習者は、4 節で紹介する物理シミュレータ SimSym を利用し、以下のような手順で学習する。

1. シナリオを提示する。
2. 学習者が SimSym にあらかじめ用意されたモデルのシミュレーションを行う。
3. 学習者は物体の運動を表す数式を予想する。予想した数式 SimSym に入力し、モデルを作成する。
4. 学習者が作成したモデルをシミュレートし、用意されたモデルと比較する。
5. 学習者は用意されたモデルと一致するようにモデルを修正し、再度シミュレーションを行う。

また、SimSym 上であらかじめ用意されたシミュレーションと学習者が作成したシミュレーションの位置や速度を可視化したり、明らかに不正な数式が入力できないような補助をすることで、学習者がモデルを作成する際の支援を行う。

3.3　提案手法のメリット

提案する手法では、学習者はあらかじめ用意されたモデルをシミュレートするだけでなく、自身の予測や推論に基づいたモデルを作成し、シミュレートすることができる。また、作成したモデルと用意されたモデルを比較しながらモデルを修正することで、物理現象がどのような数式で表されるのかを実験できる。これにより、実験的側面と同時に、物理現象の数式による理解を促進できると考えられる。

4　提案する物理シミュレータ SimSym

SimSym は、学習者自身が数式を用いて物理現象のモデル化を行うという学習手法を実行できる物理シミュレータである。学習者は SimSym 上で、教育者が用意したモデルをシミュレートし観測すること、また数式を用いてモデルを作成しシミュレートすることができる。

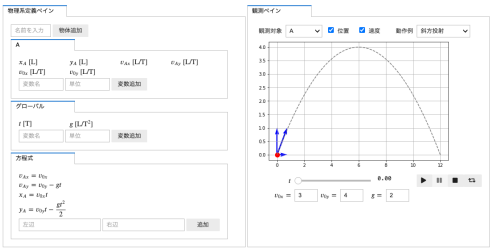

図 2　SimSym 上で斜方投射を表現した例

4.1　SimSym の構成

SimSym の画面は図 2 のようになっており、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数式の定義を行い、観測ペインでシミュレーションを確認する。

4.2　用意されたモデルのシミュレート

観測ペインの「動作例」から、あらかじめ用意されたモデルを選択し読み込むと、シミュレーションを実行できる。学習者は、初期値を変更することで様々な挙動を確認できる。また、この動作例を後に説明するモデル作成の際に参考にできる。

4.3　学習者によるモデルの作成

学習者は、以下の手順に従うことで SimSym 上で物理現象を表すモデルを作成できる。

1. 学習者が物体名を入力すると、SimSym はその物体と物体に紐づいた次元付き変数 (m_A [M], x_A [L] など[†3]) を生成する。
2. 学習者は初速などのモデル定義に必要な変数を追加する。動作例を読み込んでいる場合は、必要な変数は自動で生成される。なお、時刻を表す変数 t [T] や重力加速度 g [L/T^2] はあらかじめ用意されている。
3. 学習者は 1. と 2. で生成した変数を用いて関係式を立式・入力する。
4. 学習者は可視化したい物体とその物理量 (位置・速度・加速度) を選択する。可視化したい値を計

†3　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。

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ため、そこに適当な数値を入力する。

5. シミュレーションの再生ボタンを押すと、時間経過に応じて変数 t の値が変化する。それに伴い、物体の位置・速度などが再計算され描画される。またこの際、動作例のモデルと比較することで、学習者が定義したモデルが現実世界の運動を正しく表現しているか確認できる。

4.4 具体例: 斜方投射

以下では具体例として斜方投射を用い、SimSym の利用方法を説明する。

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. 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} が自動で生成される。
3. 1. と 2. で定義した変数を用いて、以下の関係式を立式・入力する。
   - $v_{Ax} = v_{0x}$
   - $v_{Ay} = v_{0y} - gt$
   - $x_A = v_{0x}t$
   - $y_A = v_{0y}t - \frac{1}{2}gt^2$
4. 可視化したい物体として A を選択すると、v_{Ax}, v_{Ay}, x, y を計算するために必要な g, v_{0x}, v_{0y} が観測ペイン下部に表示されるので、適当な値を入力する。
5. 再生ボタンを押すと、位置が点として、速度が矢印として描画される。この際、動作例の軌道が破線で表示され、比較することができる。

4.5 モデルの作成時に SimSym が行う補助

SimSym では、先述した動作例や変数の自動生成以外にも、学習者によるモデルの作成を補助する機能が存在する。

未定義変数の検査

先ほどの例において、$v = 0$ という数式を定義しようとしたとする。この際、v は未定義の変数であるため、図 3 のようなアラートが表示される。

図 3　未定義変数を検知した際のアラート

図 4　次元の不一致を検知した際のアラート

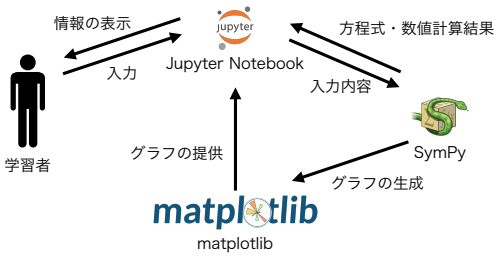

図 5　SimSym のアーキテクチャ

次元の検査

$t = g$ という関係式を定義しようとしたとする。この際、左辺の次元は [T] 、右辺の次元は $[\mathrm{L/T^2}]$ であり、次元が一致しておらず、この関係式は定義できない。その代わり、図 4 のようなアラートが表示される。

5 実現方式

現在、SimSym は Jupyter Notebook [5] 上のライブラリとして構成してあり、Web ブラウザを利用して実行できる。また、シミュレーションとして表示するグラフの生成に matplotlib [3] を、数式の処理や数値計算に SymPy [6] を用いている。

5.1 SimSym のアーキテクチャ

Jupyter Notebook

Jupyter Notebook は Web ブラウザ上で Python のコードを対話的に実行できる環境である。また ipywidgets ライブラリを利用し、GUI を作成できる。これを用い、数式の入力機能やシミュレーションの再生機能の GUI を実現している。

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 を用いることで、グラフを Jupyter Notebook に表示できる。matplotlib で、学習者が入力した数式をもとに物理現象を表現するグラフを作成している。

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定義し、Eq オブジェクトとして等式を定義できる。なお、SimSym では SymPy に次元の検査を追加した拡張ライブラリを作成し利用している。

5.2 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説明する。

1. 学習者が Jupyter 上の入力ウィジェットに物体名を入力し生成ボタンを押すと、SimSym はその物体に紐づいた変数を次元付きの Symbol として定義する。定義された変数は Jupyter 上に表示される。
2. 学習者が新たな変数名と次元を入力すると、SimSym はその変数を次元付きの Symbol として定義する。
3. 学習者が生成された変数を利用した数式を入力すると、SimSym はそれを次元付きの Eq として解釈する。またこの際、不正な次元の数式になっている場合はエラーが生じる。
4. 学習者が可視化したい物体とその物理量を選択すると、SimSym はそれらを計算するために必要な変数の値を入力するためのウィジェットを Jupyter 上に生成する。
5. 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計算され、matplotlib を用いて描画される。

6 まとめと今後の課題

本稿では、学習者自身が数式を用いて物理現象を計算機上でモデル化する学習手法と、そのためのソフトウェア SimSym を提案した。学習者が物理現象を表す数式を SimSym に入力すると、SimSym はその数式に基づく物体の運動を可視化する。教育者が用意したモデルと比較することで、学習者が定義したモデルが現実世界の運動を正しく表現しているか確認できる。また、SimSym が未定義変数や次元を検査することで、学習者のモデル作成を補助している。これらにより、実験的側面だけでなく、理論的側面に関しても物理シミュレータ上で扱えるようになり、従来の物理シミュレータでは行えていなかった現実の物理現象を数式として理解する支援ができる。

今後は以下の課題に取り組む予定である。

評価

SimSym の実際の教育効果はまだ評価できていない。SimSym を用いることで、学習者が物理現象を数式で表現できるようになるか、また数式から物理現象をイメージできるようになるかを実験したい。中学生・高校生などを SimSym を用いるグループ、従来の物理シミュレータを用いるグループ、一般的な物理実験を行うグループに分けて授業を行い、それぞれのグループでの理解度をテストなどで測定し、比較することで学習者の理解度の違いを評価したい。

基本的な公式の提供

現在、学習者は数式を全てキーボードで入力する必要がある。運動方程式や力学的エネルギー保存則などの基本的な公式を SimSym が提供し、学習者が変数を簡単に指定できるようにすることで、学習者が数式を入力する手間を減らし、より効率的に物理現象をモデル化できるようにしたい。

三次元空間への対応

現在 SimSym では二次元のグラフで描画を行なっているが、三次元空間での描画にも対応したい。これらを簡単に切り替えられるようにし、学習者がより多くの物理現象をモデル化できるよう対応したい。

参 考 文 献

[1] Ajredini, F., Izairi, N., and Zajkov, O.: Real Experiments versus Phet Simulations for Better High-School Students' Understanding of Electrostatic Charging, *European Journal Of Physics Education*, Vol. 5, No. 1(2014), pp. 59.

[2] Holubova, R.: The Impact of Experiments in Physics Lessons – "Why, When, How Often?", *DIDFYZ 2019: Formation of the Natural Science Image of the World in the 21st Century*, Terchova, Slovakia, 2019, pp. 030007.

[3] Hunter, J. D.: Matplotlib: A 2D Graphics Environment, *Computing in Science & Engineering*, Vol. 9, No. 03(2007), pp. 90–95.

[4] Kim, M., Cheong, Y., and Song, J.: The Meanings of Physics Equations and Physics Education, *Journal of the Korean Physical Society*, Vol. 73(2018), pp. 145–151.

[5] Kluyver, T., Ragan-Kelley, B., Pérez, F., Bussonnier, M., Frederic, J., Hamrick, J., Grout, J., Corlay, S., Ivanov, P., Abdalla, S., and Willing, C.: Jupyter Notebooks—a Publishing Format for Reproducible Computational Workflows.

[6] Meurer, A., Smith, C. P., Paprocki, M., Čertík, O., Kirpichev, S. B., Rocklin, M., Kumar, Am., Ivanov, S., Moore, J. K., Singh, S., Rathnayake, T., Vig, S., Granger, B. E., Muller, R. P., Bonazzi, F., Gupta, H., Vats, S., Johansson, F., Pedregosa, F., Curry, M. J., Terrel, A. R., Roučka, Š., Saboo, A., Fernando, I., Kulal, S., Cimrman, R., and Scopatz, A.: SymPy: Symbolic Computing in Python, *PeerJ Computer Science*, Vol. 3(2017), pp. e103.

[7] Perkins, K., Adams, W., Dubson, M., Finkelstein, N., Reid, S., Wieman, C., and LeMaster, R.: PhET: Interactive Simulations for Teaching and Learning Physics, *The Physics Teacher*, Vol. 44, No. 1(2006), pp. 18–23.

[8] 鬼塚史朗: 理科教育における実験の意義, 物理教育, Vol. 46, No. 5(1998), pp. 255.

[9] 壮一林, 康文川村, 聡村上: 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(2015), pp. 191–196.