Modeling and Simulation of Continuous Time-Invariant Systems in Simulation Based Learning Environments

Gang Chen[1], Yi Li[2], and Jinyang Shi[1]

1 Educational Technology Department, School of Journalism and Communication
Yangzhou University, Yangzhou, 225002, China
chengangyz@gmail. com
2 Educational Technology Department, School of Educational Science
Nanjing Normal University, Nanjing, 210097, China

Abstract: Simulation based learning environments are widely used in elementary and secondary science education. However, a large number of these learning environments are domain-dependent so that they are only useful in some special domains. In this paper, we put forward a new method of simulating continuous time-invariant systems by using frame based knowledge representation and interpretive structural modeling. This method is helpful to realize the separation of knowledge representation and simulation program, and promotes the reusability of a simulation based learning environment.

Keywords: simulation based learning environment, continuous time-invariant system, interpretive structural modeling, frame based knowledge representation

1 Introduction

Science education has two primary goals: one is to teach learners about scientific knowledge about the natural world; and the other is to help them grasp scientific skills, methods and procedures [1]. Therefore, science learning is no longer seen as the more or less directed transfer of knowledge from an authority (e.g., a book or a teacher) to a learner, but as a process of knowledge construction in which learners play an active role [2]. In other words, learners should learn science by doing science just like scientists. This type of learning pattern is called “scientific inquiry learning”.

To support this new learning pattern, many computer simulation based learning environments have come out, in which learners can learn science by discovering the knowledge behind the simulations of the natural world systems or building their own models according to their understanding of the natural world systems [3,4].

However, a large number of these simulation based learning environments are domain-dependent. That is, designers have embedded relative domain knowledge into the code of software in the phase of design, so that these learning environments are only useful in some special domains. For example, a learning environment for Newtonian’s motion laws can not be used for gas state equation $P*V=n*r*T$.

In our opinion, domain-independence is a better idea. In other words, domain knowledge should be separated from the code of software, so that learning environments can model and simulate more natural world systems. Therefore, according to knowledge engineering methods, we have built a knowledge representation frame and simulation engine for natural world systems.

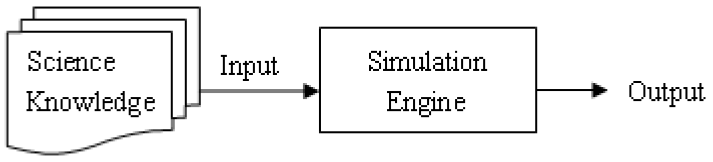


Fig. 1. Separating knowledge and simulation engine

In this paper, we firstly analyze the characteristics of continuous time-invariant system to which the most of scientific knowledge in elementary and secondary education belongs; in section 3, we introduce the frame based knowledge representation of continuous time-invariant systems; in section 4, we discuss the key algorithms in the simulation engine.

2 Continuous Time-Invariant System

In elementary and secondary science education, learners often need to learn the relationships among several factors in the natural world. For example, when an iron block is under some liquid, the buoyancy has relationship with the density of the liquid and the volume of the block; if the latter two factors increase, the buoyancy will increase synchronously. The iron block and the liquid compose a system, which has the following characteristics:

(1) The factors of the system change continuously with respect to time;

(2) The equations describing the relationship of the factors don’t change with respect to time.

All the systems that have the two characteristics are called continuous time-invariant system. The most of elementary and secondary science knowledge is about this type of system.

For the convenience of the following discussion, we give another detailed example of continuous time-invariant system. See figure 2. There is 20*mol* ideal gas in a closed container whose mass, volume, pressure, temperature, and specific heat capacity is 20*kg*, 30m^3, 120*Pa*, 21.65*K*, and 900$J.kg^{-1}.K$ respectively. People can control the temperature of the container by burning the stove whose power is 100*J*.

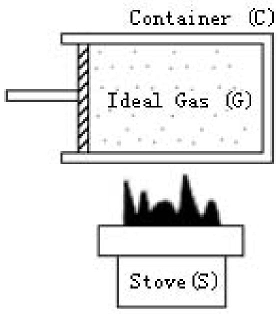


Fig. 2. Container-gas system

Table 1. Variables in the container-gas system

ID	Meaning	ID	Meaning
P_C	pressure of container	P_G	pressure of ideal gas
T_C	temperature of container	T_G	temperature of ideal gas
V_C	volume of container	V_G	volume of ideal gas
M_C	mass of container	n	the amount of ideal gas
SH_C	specific heat capacity of container	R	constant of ideal gas
PW_S	power of stove		

3 Knowledge Representation

3.1 Structure of Continuous Time-Invariant System

To describe the structure of continuous time-invariant systems, we depart a system into five parts: entities, variables, parameters, inputs and relations.

Entities are the most basic elements in a system, and can be seen as distinguishable objects in the natural world. Examples of entities in the container-gas system are the closed container, ideal gas, stove and so on. An entity is often denoted by a symbolic constant. We can use C to stand for the container, and use G to stand for the ideal gas.

A variable corresponds to a property of an entity, e.g. the volume, temperature and pressure of the ideal gas. Similar with an entity, a variable can also be represented by a symbolic constant. See table1. One can use T_C and T_G to stand for the temperature of container and ideal gas respectively. A variable is described by four attributes:

(1) the value, which is often numerical in continuous time-invariant system;

(2) the value range, which is the set of all possible values;

(3) the meaning, which the variable stands for, e.g. V_C stands for the volume of the container;

(4) the unit, e.g. the unit of V_C is m^3.

Parameters represent external factors, which influence a system and are not changed by factors inside a system. So, the difference between a parameters and a variable is that the former is a constant. Typical examples include the ideal gas constant whose value is 8.31 $Pa.m^3.mol^{-1}.K^{-1}$, the acceleration of gravity whose value is 9.8$m.s^{-2}$ on the earth, and so on. Much like a variable, a parameter is also described by three attributes: value, meaning and unit.

An input corresponds to a simulation condition of a system, and is also called experimental framework by some researchers [5]. A continuous time-invariant system may have several different inputs, in which the system may have different behaviors. An input is usually defined as how some variables are controlled by a learner during a simulation. For example, in the container-gas system, the input is described as burning the stove whose power is 100J and keeping the volume constant. So, an input can be expressed by a function group, e.g.

$$\begin{cases} PW_{S(t)} = 100 \\ V_{C(t)} = 30 \end{cases} \tag{1}$$

(The symbol of t stands for the variable of time which counts from 0.)

Relations are math models of a system. They explain how variables are associated with each other, and can be used to predict how a system will change under a certain input. Relations are expressed in the form of a function group. In the container-gas system, the function group under the condition of heating the container and keeping volume constant is listed below:

$$\begin{cases} H_{(t)} = PW_S * t \\ T_{C(t)} = T_{C(0)} + \dfrac{H_{(t)}}{SH_C * M_C} \\ T_{G(t)} = T_{C(t)} \\ P_{G(t)} = \dfrac{n * R * T_{G(t)}}{V_{G(t)}} \\ P_{C(t)} = P_{G(t)} \\ V_{G(t)} = V_{C(t)} \end{cases} \tag{2}$$

($H_{(t)}$ stands for the quantity of heat abstracted by the container from time 0 to time t.)

What must be stressed is that every input corresponds to a special function group. For example, the function group (2) is not useful under the condition of changing the volume and keeping the temperature constant.

3.2 Frame Based Knowledge Representation

Frame based knowledge representation is a good approach to describe a system like the container-gas system. The fundamental idea of a frame is rather simple: a frame can be seen as a generic data structure about an object; and it is essentially a collection of slots and slot-values [6]. When a frame is being used, the slot-values can be altered to make the frame corresponding to the particular situation at hand.

According to the structure of a continuous time-invariant system, we build a frame to describe such systems. See table2 and table3.

Table 2. Frame structure of continuous time-invariant systems

```
System:
  Entities:
    Entity:
      Name:
      Variable:<Name, Value, Value Range, Unit, Meaning >
      …
    Entity:
      …
  Parameters:
    Parameter: <Name, Value, Unit, Meaning >
      …
  Input:
  Relations:
```

Table 3. Frame of container-gas system

System: container-gas system	
Entities:	Parameters:
Entity:	Parameter:
Name: *container*	Name: *R*
Variable:	Value: *8.31*
Name: P_C	Unit: $Pa.\ m^3.mol^{-1}.K^{-1}$
Value:*120*	Meaning: *constant of ideal gas*
Range: $(0,+\infty)$	Input:
Meaning: *pressure of the container*	$PW_{S(t)}=100$;
Unit: *Pa*	$V_{C(t)}=30$;
Variable:	Relations:
Name: T_C	$H_{(t)}=PW_S*t$;
…	$T_{C(t)}=T_{C(0)}+H_{(t)}/(SH_C*M_C)$;
Entity:	$T_{G(t)}=T_{C(t)}$;
Name: *ideal gas*	$P_{G(t)}=n*R*T_{G(t)}/V_{G(t)}$;
Variable:	$P_{C(t)}=P_{G(t)}$;
…	$V_{G(t)}=V_{C(t)}$;

4. Simulation Engine

4.1 Basic Simulation Procedure

We use a simulation engine to simulate different systems that are defined by frames. There is a simulation clock (SC) in the simulation engine. The basic simulation procedure is that the simulation engine calculates the functions in the relations part of the frame every other interval. See table4.

Table 4. Basic simulation procedure in simulation engine

```
Simulate ()
{
  While (SC.CurrentTime <= SC.EndTime)             // limit the time length of simulation
  {
     Calculate_Function_group();                   // calculate all the functions in relations
     If (any variable's value out of its value range)  // make sure all variables do
        break;                                     // not beyond their value range
     SC.CurrentTime+=Time_Step;                    // add an interval to the current
  }                                                // time of the simulation clock
}
```

4.2 Deciding the Calculation Order of Functions

However, functions in a function group can be arranged in many different orders. It means the simulation engine needs to decide in which order these functions should be calculated. In fact, a function like $y=f(x_1, x_2, \ldots,x_n)$ implies there is a dependence relationship between y and $(x_1, x_2, \ldots,x_n)$. That is to say, one should calculate the value of $(x_1, x_2, \ldots,x_n)$ firstly, and then y. If there is another function $x_n=g(z_1,z_2,\ldots z_n)$, the calculation order should be $(z_1,z_2,\ldots z_n)$, $(x_1, x_2, \ldots,x_n)$, and then y. So, we can say a function group corresponds to a dependence digraph. What a simulation engine should do is to draw the dependence digraph and decide the calculation order according to it.

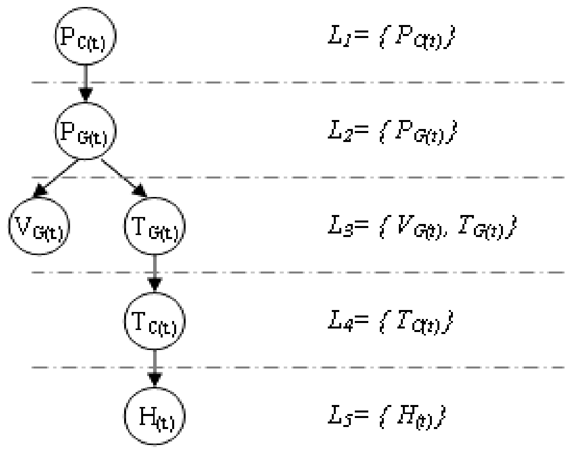
Interpretive structural modeling(ISM) method is helpful to solve this problem. ISM was proposed by David W. Malone in 1975 [7]. It is a method which can be applied to a system--such as a network or a society--to better understand both direct and indirect relationships among the system's components. The algorithm using ISM to decide the calculation order can be divided into five steps:

$$\begin{array}{c|cccccc} & H_{(t)} & T_{C(t)} & T_{G(t)} & P_{C(t)} & P_{G(t)} & V_{G(t)} \\ \hline H_{(t)} & 1 & 0 & 0 & 0 & 0 & 0 \\ T_{C(t)} & 1 & 1 & 0 & 0 & 0 & 0 \\ T_{G(t)} & 0 & 1 & 1 & 0 & 0 & 0 \\ P_{C(t)} & 0 & 0 & 0 & 1 & 1 & 0 \\ P_{G(t)} & 0 & 0 & 1 & 0 & 1 & 1 \\ V_{G(t)} & 0 & 0 & 0 & 0 & 0 & 1 \end{array}$$

(a) adjacent matrix

$$\begin{array}{c|cccccc} & H_{(t)} & T_{C(t)} & T_{G(t)} & P_{C(t)} & P_{G(t)} & V_{G(t)} \\ \hline H_{(t)} & 1 & 0 & 0 & 0 & 0 & 0 \\ T_{C(t)} & 1 & 1 & 0 & 0 & 0 & 0 \\ T_{G(t)} & 1 & 1 & 1 & 0 & 0 & 0 \\ P_{C(t)} & 1 & 1 & 1 & 1 & 1 & 1 \\ P_{G(t)} & 1 & 1 & 1 & 0 & 1 & 1 \\ V_{G(t)} & 0 & 0 & 0 & 0 & 0 & 1 \end{array}$$

(b) reachability matrix



(c) dependence relationship

Fig. 3. Using ISM to get calculation order of functions in container-gas system

(1) Building a matrix for all variables in the left side of functions.

We assume all variables in the left side of functions compose a set $V= \{v_1, v_2,\ldots,v_n\}$ (v_i is a variable). Build a matrix A *of size* $n*n$, and all elements of A is set to 0. The *ith* row and column correspond to the variable v_i. For example, to function group (2) ,
$V= \{H_{(t)}, T_{C(t)} ,T_{G(t)} , P_{G(t)} , P_{C(t)} ,V_{G(t)}\}$.

(2) Building an adjacent matrix according to dependence relationships in the function group.

In every function, we assume the variable on the left side is v_i. If there is a $v_j \in V$ and v_j appears in the right side of the function, then we set the element a_{ij} in matrix A to be 1, which means v_i depends on v_j. The new matrix A is called an adjacent matrix, whose operations are all Boolean.

(3) Building a reachability matrix.

Calculate $S=(A+I)^{n-1}$, where I is an identity matrix of size n. S is called a reachability matrix. If $s_{ij}=1$, then it means v_i depends on v_j directly or indirectly; otherwise, it means v_i doesn't depend on v_j.

(4) Getting the dependence digraph.

The reachability matrix S can be used to get the dependence digraph. To do this, we firstly define an antecedent set $A(v_i)$ for each variable $v_i \in V$, as all of those variables on which v_i depends. $A(v_i)$ is stated mathematically as: $A(v_i) = \{ v_j \in V \mid s_{ij} = 1\}$.

Secondly, we define a reachability set $R(v_i)$ for each variable $v_i \in V$, as all of those variables which depend on v_i. Stated mathematically: $R(v_i) = \{ v_j \in V \mid s_{ji} = 1 \}$.

Then, we define ($L_1, L_2, \ldots, L_k$) as levels of the dependence relationship among all variables from top to bottom, and

$L_k = \{ v_i \in (V - L_0 - L_1 - \ldots - L_{k-1}) \mid R_{k-1}(v_i) = R_{k-1}(v_i) \cap A_{k-1}(v_i) \}$.

Where L_0 is defined as an empty set ϕ for convenience of calculation, $R_{k-1}(v_i)$ and $A_{k-1}(v_i)$ stand for the reachability set and antecedent set of v_i in the set of $(V - L_0 - L_1 - \ldots - L_{k-1})$ respectively.

(5) Deciding the calculation order of functions.

Therefore, the calculation order of functions is that variables in L_k should be calculated firstly, then L_{k-1}, $L_{k-2}\ldots$ until L_1. To every variable in the same L_j, there is no special order.

5. Implementation

We have developed a simulation based learning environment for elementary and secondary science education based on the method introduced in this paper. Technically, the frame structure is defined as a XML schema, and simulation engine is programmed using C#. Because of the separation of simulation engine and knowledge representation, teachers can add new science knowledge into the learning environment which will make a simulation dynamically. Now, many frames of continuous time-invariant system--such as pendulum, buoyancy, spring and so on--have been built, and the learning environment runs very well.

6. Conclusion and Future Work

In this paper, we discuss how to simulate continuous time-invariant systems using frame representation and interpretive structural modeling. This new method is helpful to improve reusability of simulation based learning environments. In fact, the frame representation is based on the object-oriented opinion. It implies that a frame is reusable. For example, several frames can be integrated into a more complex frame. So, the reuse of frames is our future work.

Acknowledgement

This work was supported by the project “Thinking with Graphs” funded by social science research foundation of Yangzhou University.

References

1. Li, J. & Klahr, D. The Psychology of Scientific Thinking: Implications for Science Teaching and Learning. In J. Rhoton & P. Shane (Eds.) Teaching Science in the 21st Century. National Science Teachers Association and National Science Education Leadership Association: NSTA Press (2006)
2. Joolingen, W.R. van, & de Jong, T. Design and implementation of simulation-based discovery environments: the SMISLE solution. Journal of Artificial Intelligence and Education, 7, 253-277 (1996)
3. De Jong, T, & van Joolingen, W. R. Scientific discovery learning with computer simulations of conceptual domains. Review of Educational Research, 68, 179-201 (1998)
4. Forbus, K., Carney, K., Sherin, B. and Ureel, L. Qualitative modeling for middle-school students. Proceedings of the 18th International Qualitative Reasoning Workshop, Evanston, Illinois, USA, August(2004)
5. Weihong Wang. Modeling and simulation. Beijing: science publishing house(2001)
6. O. Lassila and D. McGuiness., The Role of Frame-Based Representation on the Semantic Web, Linkoping Electronic Articles in Computer and Information Science, vol.6 (2001)
7. Malone, D.W., "An introduction to the application of interpretive structural modeling", Proceedings of IEEE, Vol. 63 pp.397-404 (1975)