列車模型レイアウトを用いた初学者向け計算機学習環境の実現

Development of Computer Learning Environment for Beginners Using Model Railway Layout

佐々本博和 [1), 2)]，野間春生 [1)]，須佐見憲史 [1)]，伊藤雄一 [2)]
北村喜文 [2)]，岸野文郎 [2)]，鉄谷信二 [1)]
Hirokazu SASAMOTO, Haruo, NOMA, Kenji SUSAMI, Yuichi ITOH
Yoshifumi KITAMURA, Fumio KISHINO, Nobuji TETSUTANI

1) 株式会社 国際電気通信基礎技術研究所 メディア情報科学研究所
ATR Media Information Science Laboratories, Japan
{sasamoto, noma, susami, tetsutani}@atr.jp
2) 大阪大学大学院情報科学研究科
Graduate School of Information Science and Technology, Osaka University, Japan
{sasamoto, itoh, kitamura, kishino}@ist.osaka-u.ac.jp

Abstract

This research is aimed at developing a new methodology that supports technologies for beginners to learn about computation and programming through hands-on play experience with toys. It allows beginners to acquire the concept and knowledge of computation and programming by playing with a model railway, and to understand programming and strategic thinking using a railway layout. In this research, we proposed such a hands-on learning environment of computer and programming study for beginners using a model railway and built the 1st trial system.

1. はじめに

初学者が抽象的な概念を学習する際, その概念をいくつかの具体的な実例に関連付け, それらを比較しながら学ぶことが重要である. 例えば, 幼児は絵本によって絵と文字の関連を理解し, やがては自然に高度な読み書きの技術を会得する. これと同様に, コンピュータの初期習得過程においても, それらが持つ意味そのものを具体化させた玩具で遊ぶことによって実感し, 将来高度な計算機技術の習得が容易となることが期待できる. 我々は, その具象化方法として列車レイアウト遊びを用いた新しい幼児向け計算機学習システムを提案してきた[1][2].

本稿では, この計算機学習環境を実現するための技術課題を検討し, それらを踏まえて設計を進めた第一試作について報告する.

2. 関連研究

計算機教育用に列車レイアウトのアナロジを導入した例がいくつか報告されている[3][4][5]. これらの先行研究では, 本提案と同様に, 計算機と列車運行の間の類似性に着目しているが, いずれも固定レイアウトを計算機の動作制約の表現, あるいは計算機そのものの状態表示装置として使う段階に留まっている. また対象は, 「計算機を使って何をするか」を学ぶことを前提とした学生であり, 初学者を対象としたものではない.

3. 列車模型レイアウトを用いた初学者向け計算機学習環境と利点

一般的に, 計算機上で動くプログラムはノイマンスタイルであり, コンピュータは事前に保存されたプログラムに沿って動作し, その構成要素は条件分岐と演算から構成されている. 一方, 列車は敷設されたレールの上を貨車や客車を牽引し, 逐次運行する. 我々はこの類似性に着目し, 表 1 のようにプログラムの基本要素を列車レイアウトに対応させ, 上記のようなコンピュータの概念を列車のレイアウト遊びによって学べる環境を提案する. この対応付けを行った列車レイアウトの一例を図 1 に示す. 出発地点のレールへ列車を置き, ボタンを押して発車させると, 列車は初めの Load ユニットで牽引する貨車に貨物を 1 つ積む. 次に列車が

表 1　コンピュータから列車への写像

Computer	Operation of a train
program	Rail layout
Condition jump	Point
Program counter	Position of train
Variable	Dump car
Variable value	Ball in dump car
Operation of variable	Load / Unload a ball

Point ユニットまで進むと, 貨車に貨物が積まれているかをチェックし, 貨物の有無や色, 数に従ってポイントを切り替える. 列車内に貨物がある場合, 列車は Unload unit で貨物を降ろす. 貨物がない場合は, 列車はゴールへ到達する. これは単純なループアルゴリズムのレイアウト例であるが, 図 2 に示すようなボールを積み下ろしするユニットと, ボールの色や有無によってポイントの切り替える単純な機能を持つ 3 種類のユニットだけで構成されている.

提案する学習システムは, 学習者にとって, 課題に合わせてレールを自由に組み替えたりボールを列車に積んだりなど, 純粋な遊びであり, 学習者が好んで積極的に取り組むことができる. また, 列車が意に反する動作をした時には, 手でレイアウト内の列車を止めたり, 貨車のボールを移動させて動作を変えることができる. これは, すなわち, ブレークポイントの設定や, 変数の操作のデバッグに相当するが, それらプログラムに不可欠な操作を手を使って直感的に操作できる.

さらに, グループ学習の教材としても効果が期待できる. グループで 1 つレイアウトを作る際には, それぞれのアイデアをレイアウトとして具象化できるため, 容易に互いのアイデアであるレイアウトを融合させることができる.

4. 技術課題

前述した初学者に対する列車レイアウトを用いた学習環境を構築するために, 次に示す技術課題が挙げられる.

- **コンピュータの基本要素の列車ユニットへの投影**
- **自由で柔軟なレイアウトの組み替え**
- **レイアウトの自動制御**

まず, 学習の目的である計算機の仕組みを列車レイアウトで実現するために, 計算機が有する演算や関数を, どの程度までレイアウトのユニットに投影するべきかを検討せねばならない. 図 1 では, 積み下ろしとポイントの切り替えの最小限の機能だけでレイアウトを構築しているが, さらにシフト演算や論理演算など, CPU が持つ機能を列車ユニットに対応させユニットの種類を増やせば, アプリケーションとして OS の仕組みやコンピュータシミュレーションなども構築できる.

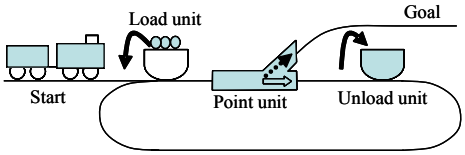


図 1　ループのレイアウト例

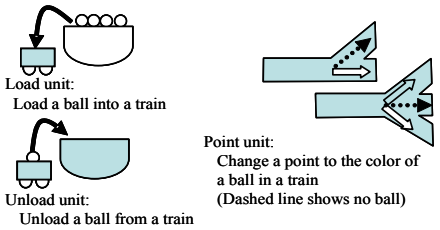


図 2　列車レイアウトユニットの例

次に, 学習者がこのシステムを使って, 自由にプログラムのアイデアを具現化するためには, システムがレイアウトの構成を自動認識し, レイアウト構築の際に, 簡単かつ柔軟に基本ユニットを組み替えできなければならない. 反面, レールやユニット部品の機械的・電気的な接続が容易でありながら確実でなければならない. また, ユニットの動作が直感的なものでなければならない.

さらに, 学習者は列車とユニットの動作を見ることで, コンピュータの概念を直感的に学習するので, レイアウト上で列車を動かす際に, レイアウトを構成するユニットと列車が確実に協調して動作しなければならない. そのため, 表面的には単純に動いているように見えながら, その背後では貨車の積載状況や列車の運行状況をモニタし, レイアウトを自動制御しなければならない.

5. レイアウトの構築例

前述のように本システムの特徴は, 基本ユニットを用いて多様なレイアウトを構築し, 教材として与えられる課題を実現することにある. そこで, 3 章で例示したループ以外にも, 積み下ろしとポイント切り替えの最小限のユニットを用いて, 次に示すような様々なレイアウトが構成できる.

- **ボールの振り分け**

  図 3 は,Load unit (A)にある 2 種類の色のボールを, それぞれ Unload unit (A),Unload unit (B)に振り分けるという動作を行うレイアウトである. これは, 「ボールの振り分け」 を行う課題を構成している.

- 足し算

  図 4 は,Load unit (A)のボールの数と Load unit (C)内のボールの数を足し合わせた分を Unload unit (B)でとりだすレイアウトである. よってこれは「足し算」の概念を学ぶことができる.
- 引き算

  図 5 では,Load unit (A)のボールの数だけ列車を周回させ Load unit (D)から Unload unit (C)にボールを移すことで, Load unit (D)から Load unit (A)のボールの数分だけ減らす,「引き算」の概念をレイアウトとして実現できる.

さらに, 上記したレイアウトを組み合わせて,「コピー」,「比較」などのレイアウト, あるいは「掛け算」,「割り算」などの計算が行えるレイアウトや, 他の演算処理を実現するユニットを追加することで, さらに複雑なレイアウトへ発展できる[2].

6. 実装

6.1. ハードウェア構成

4 章の技術課題で述べたように, 自由かつ柔軟なレイアウトの構築とレイアウト上での列車運行イベントを実現するためには, 列車とレイアウト部品となるレールユニットとの間で通信を行い, それぞれのレールユニット上で起こる振る舞いを制御しなければならない. 本稿では,提案手法の可能性を探ることを目的とし,最小限のユニットを自由に組み替え,様々なレイアウトを作成できるシステムを構築することを優先した第一試作を行った.それぞれのユニットと列車は, トミー社製のプラレールに, Microchip 社製の PIC とそれに接続するセンサとアクチュエータを埋め込んだ. プラレールは幼児期の子供向けの鉄道模型玩具であり, レールを自由に柔軟に組み合わせることが可能である. システムは図 6 のように, 全体のシステムを統括する Base unit を中心に,各レールユニットを RS-232C によって接続するという構成にした. また,Base unit と列車間の通信は R.F.Solutions 社の無線シリアル通信モジュールを用いた.

扱えるボールの色は 2 種類, 列車に乗せることができるボールは 1 つという制限を持たせた. ボールの積み降ろしや, ポイントの切り替えに用いるアクチュエータとしてソレノイドを, それぞれのユニットへの列車到着を確認するためのセンサとしてフォトインタラプタを用いた.

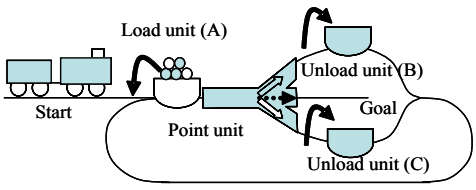


図 3　振り分けのレイアウト例

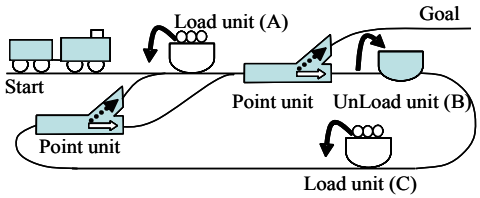


図 4　足し算のレイアウト例

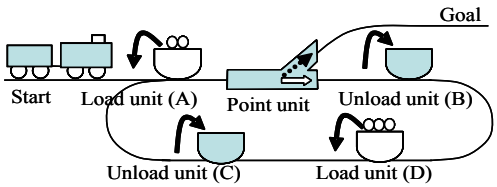


図 5　引き算のレイアウト例

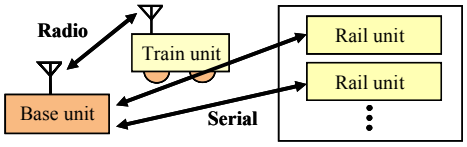


図 6　システム構成図

6.2. システムの動作

システムの動作を図 7 を用いて説明する. Base unit には, 複数のレールユニットを接続するためのポートを用意する. レールユニットはポートの位置に依存せずに接続が可能である. また, それぞれのレールユニットには, あらかじめユニークな ID 番号を割り当てる. さらに, 例えば, ID 番号 1 の Load unit で積んだボールのイベント情報は ID 番号 2 の Point unit で受け取る, といったようにユニット間を 1 対 1 で関連付け, ID の組み合わせを固定してある.

次に, ユニット間のデータの流れについて説明する. 例として, Load unit でボールを乗せ, そのボールの色によって Point unit のポイントが切り替わる場合のユニット間のイベントの流れを示す.

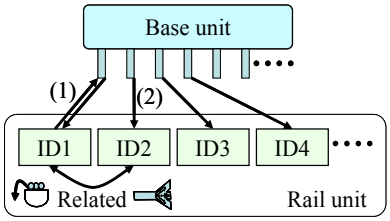


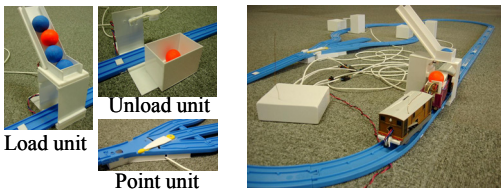
図 7　ユニット間のデータの流れ

あらかじめ, Load unit には ID 番号 1 を Point unit には ID 番号 2 を割り当て, お互い関連付けておく. まず, 列車が Load unit に到着した際, Load unit から列車ユニットを介して Base unit へボール挿入のイベントを送る(1). この際 Load unit の ID 番号を付加して送る. 次に Base unit から全ユニットに, ID 番号を付加したボール挿入のイベントを送る(2). このとき, あらかじめ関連付けていた ID 番号 2 の Point unit のみイベントを受け, ボールの色に従ってポイントを切り替える. したがって,例えば Point unit が 2 つ以上ある場合でも,独立してポイントを切り替えることが可能となる.

6.3. 実装結果

実装したレールユニットを図 8 (a)に示す. 実装したレイアウト部品を用いて, ボールの振り分けとボールのコピーのレイアウトを構築した. 図 8 (b)に今回実装を行ったレイアウト部品を用いた「振り分け」のレイアウトを示す. さらに, これらの基本ユニット部品の組み替えを行うことで, 5 章で例示した各レイアウトに再構成できることを確認した.

一方で, 今回の実装では, レールと列車間をベースユニットを介して通信するスター型の通信方式を採用したため, ユニット間のイベントのやり取りが容易となり, 個々のユニットの構成と, それらを結ぶ配線も単純というメリットがある. しかし, レイアウトユニットを増やせばそのユニットの数だけ配線が必要となり, レイアウトの変更作業には若干手間どう. この問題を解決するために, 我々が開発した ActiveCube[6]のブロック間の通信方式を用いることで, ユニット間の通信を行う予定である. ActiveCube ではブロックを組み合わせるだけでブロック全体が通信を行い, その全体形状をリアルタイムに認識できる. この方式によりレールレイアウトの形状を外部配線なしにリアルタイムで認識でき, より自由かつ柔軟にレイアウトを組み替えることができる. また, レールユニットを独立して動作させるために, Load unit, Point unit を 1 対 1 に関連付けたので, レイアウトを構築する際に, ユニット間の関連を考慮に入れながら構築しなければならないといった制約ができる. この問題に関しても, 今後解決していく必要がある.



(a) Rail unit　　(b) Rail layout (sorting)

図 8　実装図

7. おわりに

列車レイアウトシステムを用いた教育環境が初学者へのコンピュータ教育に適していることを述べ, その環境を実現するための技術課題を述べ, 試作環境を実装した. そして, 実装環境でいくつかのレイアウトの組み替えを行い, 動作を確認した. 今後は, 個々のレイアウト部品やシステム構成を改良し, より自由かつ柔軟にレイアウトを組み替えることができるシステムを構築し, 提案システムの有効性の評価・検証を行っていく予定である.

8. 参考文献

[1] Noma, Sasamoto, Itoh, Kitamura, Kishino, Tetsutani, “Computer learning system for pre-school-age children based on a haptized model railway”, Creating, Connecting and Collaborating Through Computing, No. 1, pp. 118-119, 2003

[2] 佐々本, 野間, 伊藤, 北村, 岸野, 鉄谷, “列車模型を用いた幼児向け計算機学習システムの構築”, 日本バーチャルリアリティ学会, No. 8, pp. 159-163 , 2003

[3] J. McCormick, “Using a model railroad to teach digital process control”, 19th SIGCSE Technical Symposium on Computer Science Education, Vol. 20, No. 1, pp. 304-308, 1988.

[4] R. Dowsing, P. Saward, “Computer-controlled model railway for use in teaching real-time programming”, Micro-process & Microsystem, Vol. 6, No. 10 pp. 529-533, 1982.

[5] R. Tosten, “Using a model railroad system in an artificial intelligence and operating system course”, ACM SIGCSE Bulletin, Vol. 25, No. 1, pp. 30-32, 1993.

[6] 伊藤，河合，北村，岸野，“リアルタイム 3 次元形状モデリングとインタラクションのための双方向ユーザインタフェース ActiveCube”，情報処理学会論文誌, Vol. 42，No. 6, pp. 1338-1347, 2001.