

レゴ向けの状態遷移図設計支援システム

今井康平[†] 紫合 治[‡]

東京電機大学院情報環境学研究科[†] 東京電機大学情報環境学部[‡]

1. はじめに

従来、状態遷移図は状態間の遷移に対してアクション等の記述を行うことができる。だが、状態や遷移が多くなるにつれて状態遷移が複雑になり、各々のドメインの状態が理解しづらくなってしまいう。そこで状態遷移図の状態に可能な限りのドメインの情報を図等で表示することで、各ドメインの状態を把握しやすくするような状態遷移図の設計支援ツールについての提案を行う。

2. 状態指向の状態遷移図

本論文では従来の状態遷移図を遷移指向の状態遷移図とする。そして新たに提案する状態遷移図を状態指向の状態遷移図とする[1]。状態指向の状態遷移図では、遷移には何も書かず状態に対してそのとき各ドメインがどうなっているかをドメインの状態や条件等で記述する。ドメインがその条件を満たさなくなったときに遷移が起こる。

3. ROBOLAB によるレゴプログラミング

ROBOLAB はフローチャート形式でレゴに対するプログラムを入力することができるソフトである[2]。各センサやモータのアイコンがあり、それらを羅列することでプログラムを完成させレゴを動かすことができる。図1に ROBOLAB によるプログラムの例を示す。

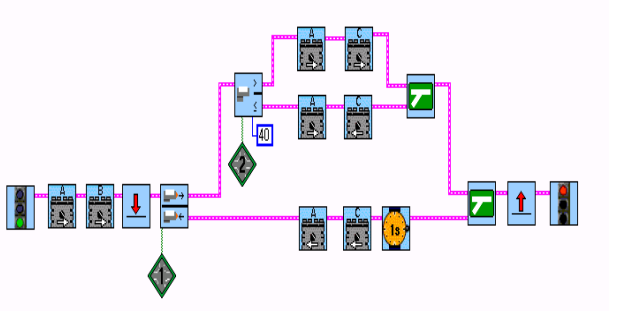


図1 ROBOLAB によるプログラム例

4. 状態指向状態遷移図によるレゴプログラミング

4.1 レゴ用状態指向状態遷移図の提案

以下に状態指向の状態遷移図をレゴのプログラムに応用する方式を示す。図2にレゴ用状態指向状態遷移図の詳細を示す。



図2 状態の説明

1つの状態は、大別すると状態名、条件部、出力部の3つからなる。条件部にはセンサの種類、センサの値、タイマー等の条件を記述する。

表1 遷移の条件

センサ	遷移条件
光センサ	任意の値より明るくなる／暗くなる
タッチセンサ	ON／OFF
音センサ	任意の値より大きい／小さい音を受け取る
超音波センサ	任意の値より離れる／近づく
角度センサ	モータが任意の角度回転する
タイマー	任意の時間が経つ

その記述された条件を満たしている限りその状態を維持し、条件にそぐわなかった場合に状態遷移が起こる。遷移先はそのときの条件を満たしている状態に遷移し、その状態に見合ったモータの動作状態にする。出力部には、その状態の時の各モータの動作状態(前進、後退、停止の状態や回転速度等)を記述する。

A State Transition Diagram Design Support System for LEGO.

[†] Kohei Imai
Graduate School of information environment, Tokyodenki University.

[‡] Osamu Shigo
Tokyodenki University information environment department.

4.2 提案システムのプログラム例

本項で扱うプログラムを以降で説明する。直進、旋回、後退の3種類の状態を用意し、タッチセンサ、光センサの2種類のセンサを用いてLEGOの動作を制御するプログラムである。タッチセンサが押された場合は1秒間後退、押されていない間は光センサが40を超える値の時は直進、40以下の値の時は旋回する。図3にその例を表す。図1は同プログラムをROBOLABによって表現したものである。

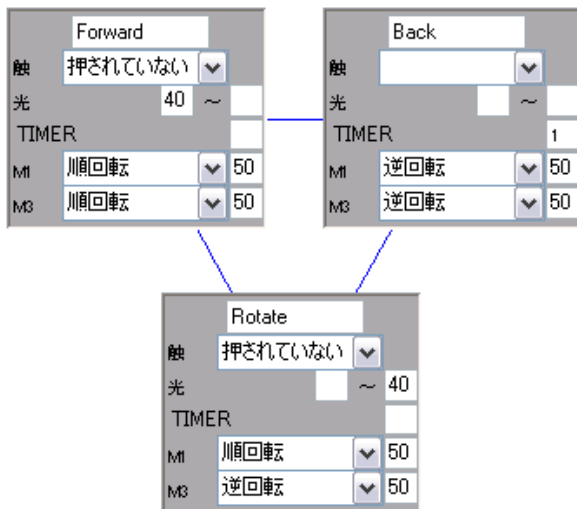


図3 本提案システムによるプログラム例

まず始めにレゴに接続するセンサやモータを右のコントロールボックスで規定する。これによって、状態遷移図の状態に接続されたセンサやモータの状態を設定するためのコンボボックスが表示される。この時ボタンの左からポート1, 2, 3, 4となっている。センサの場合、そのコンボボックスで条件を選択し、必要であれば横にあるテキストボックスで値を設定することもできる。モータの場合、順回転や逆回転などの状態を選択し、テキストボックスで出力の強さを設定することができる。条件を満たさなくなった場合に状態遷移が起こるが、その遷移先はその状態から派生している遷移のいずれかになり、且つ各センサの条件を満たしている状態になる。条件を満たしている状態が複数ある場合、条件部に書かれている条件が多い状態ほど優先度が高く、条件が少ないほど優先度が低くなる。また、もし遷移先に条件に合う状態がない場合はエラーとなる。なお、このエラ

ーは実行前に静的チェックで見える。

4.3 Cコードの自動生成

前述のシステムで作成した状態遷移図からCコードを自動生成し、それをレゴのNXTにアップロードする[3]。図4に生成されたCの一部を示す。

```
if(ecrobot_get_touch_sensor(NXT_PORT_S1) == 0 &&
    ecrobot_get_light_sensor(NXT_PORT_S3) > 40)
{
    //前進
    while(ecrobot_get_touch_sensor(NXT_PORT_S1) == 0
        && ecrobot_get_light_sensor(NXT_PORT_S3) <= 40)
    {
        nxt_motor_set_speed(NXT_PORT_A, 50, 1);
        nxt_motor_set_speed(NXT_PORT_C, 50, 1);
        systick_wait_ms(4);
    }
} else if(ecrobot_get_touch_sensor(NXT_PORT_S1) == 0 &&
    ecrobot_get_light_sensor(NXT_PORT_S3) <= 40)
{
    //旋回
    while(ecrobot_get_touch_sensor(NXT_PORT_S1) == 0 &&
        ecrobot_get_light_sensor(NXT_PORT_S3) <= 40)
    {
        nxt_motor_set_speed(NXT_PORT_A, 50, 1);
        nxt_motor_set_speed(NXT_PORT_C, -50, 1);
        systick_wait_ms(4);
    }
} else
{
    //後退
    while(timer < 250)
    {
        timer++;
        nxt_motor_set_speed(NXT_PORT_A, -50, 1);
        nxt_motor_set_speed(NXT_PORT_C, -50, 1);
        systick_wait_ms(4);
    }
}
```

図4 生成されるCコードの例

5. おわりに

本研究では、状態指向の状態遷移図の概念をレゴのプログラミングに応用した方式を提案した。これにより、初心者にもわかり易く、且つ簡単にプログラミングすることができると思われる。

今後の予定として、プログラミングの初心者による本システムの評価と、状態指向の状態遷移図の概念の有効性と問題点を評価していきたい。

6. 参考文献

- [1] 紫合治, “状態指向のステートチャート”, FOSE2005
- [2] ROBOLAB 2.9 ガイドブック, 株式会社アフレル, 2006
- [3] nxtOSEK/JSP ホームページ, <http://lejos-osek.sourceforge.net/jp/index.htm>