e－puckにおける走行ルート自動訂正方式の検討

酒井隆行†　寺島悠貴†　大谷真†
湘南工科大学情報工学科†

1. はじめに

e-puck とはスイス連邦工科大学ローザンヌ校が開発した自律型小型移動ロボットである[1]。走行用のステッピングモータ、赤外線近接センサー、LEDなどを搭載しており、これらのデバイスをソフトウェアで制御する。しかしこれらのハードウェア機能を使うプログラムは現時点では極めて少ない。本研究では与えられた走行ルートにそって走行し、障害物が存在した場合それを回避し自律的に走行ルートを改訂する e-puck 制御ソフトウェアを開発した。本論文では、走行ルートの表現方法、各走行命令の制御方式、障害物回避方式、および走行ルート改訂方式について述べる。なお本研究では e-puck 搭載デバイスのうち左右車輪回転用ステッピングモータ、前方と左右の赤外線近接センサー、および、表示用の LED を使用した。

2. 走行ルートの表現

走行ルートは固定長(14バイト)の走行命令の配列で表現した。各走行命令は表1のとおりとした。

表1 走行命令とパラメータ

命令	パラメータ			
type	Speed_sec	distance	angle	time
straight	p [mm/s]	q [mm]		
rotate	p [° /s]		q [°]	
stop				r [ms]

3. 走行命令の制御

(1) 直進(straight)

左右のステッピングモータを同一速度x(ステップ/秒)で同一ステップy(ステップ)回転させる。

Practice on Automatic Route Correction for e-puck
† Takayuki Sakai, Yuuki Terashima, Makoto Oya - Department of Information Science, Shonan Institute of Technology

xとyはstraightのパラメータpとqから次の式で求める。

$$x=\alpha p,\quad y=\alpha q$$

ただし$\alpha = 7.76 = 1000 \div 41\pi$。(ここで1000は車輪1回転のステップ数、41は車輪の直径(mm))。

(2) 自転(rotate)

e-puck の上部から見て時計回りの場合について以下説明する。左右のステッピングモータをそれぞれ逆に回転させる。左車輪は正方向、右車輪は逆方向に同一ステップ数回転させる。これによりe-puckの中心点を中心に指定された度数、回転することになる。詳しくは左車輪のステッピングモータは速度x(ステップ/秒)でyステップ回転させ、右車輪は速度－x(ステップ/秒)でyステップ回転させる。xとyはrotateのパラメータpとqから次の式で求める。

$$x=\beta p,\quad y=\beta q$$

ただし$\beta = 3.59 = 53\pi \times (1000 \div 41\pi) \div 360$(ここで53は車輪間の距離(mm)、1000は車輪1回転のステップ数、41は車輪の直径(mm))。

e-puck を上部から見た図

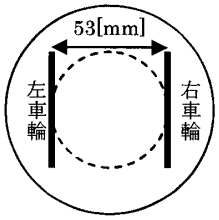

図1 e-puck の車輪構造

4. 障害物回避

(1) 障害物の回避

与えられた走行ルート(O ルートという)を走行中に障害物を検地したときには、図2に示すように右回避または左回避で障害物を迂回する。回避を行った場合は、実際走行したルート(D ルート)を記録しておく。なお、回避方法を単純化するために、障害

物の配置には以下の制約を与えた。

- 障害物は上部からみて長方形である。
- 直進走行ルート上にのみ障害物がある。
- 障害物は走行ルートに対して直角に配置されている。
- 障害物は複数あってもよいが、e-puck が通り抜けられる距離以上離れて配置されている。

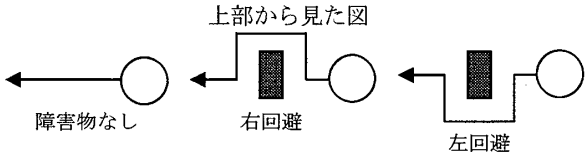

図 2 障害物の避け方

(2) 障害物回避方法

右回避の場合について制御方式を図 3 に示した。図中では、直進を S、壁探知を WD、壁沿いから抜けることを WO、右に 90° 自転を R、左に 90° 自転を L で示している。

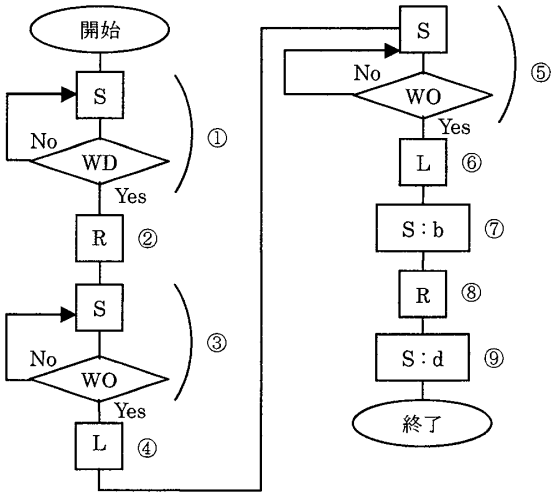

図 3 回避の制御方式

①正面センサーが、壁を検知するまで直進する。走行距離を a とする。②右に 90° 自転。③左のセンサーを使い、壁沿いから抜けるまで直進する。走行距離を b とする。④左に 90° 自転。⑤左センサーが、壁沿いから抜けるまで直進する。走行距離を c とする。⑥左に 90° 自転。⑦距離 b 直進する。⑧右に 90° 自転。⑨距離 d 直進する。ただし d =e−(a + c)。以上のように回避動作を行った際に、図 4 のように記録される。

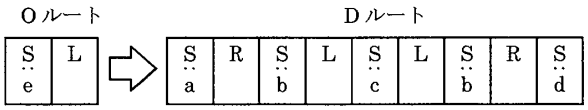

図 4 D ルートの記録

5. 最短回避走行ルートの決定

O ルート全体に対し右回避と左回避を実行したのち、記録されたそれぞれの D ルートをもとに最短回避走行ルート(Min ルート)を決定する。以下、図 5 を例として、その方式説明する。

D ルート全体は部分 O ルートと部分 D ルートを交互につないだものになる。図 5 では、O_1, O_2, O_3 は部分 O ルート、D_1, D_2 は右回避の部分 D ルート、E_1, E_2 は左回避の部分 D ルートを表している。各部分 D ルートに対して、回避距離$\delta(D) = b+c+b = 2b+c$（図 4 の記号を使用）と定義しておく。Min ルートは次のように決定する:(1)部分 O ルートはそのままコピーする。(2)部分 D ルートは回避距離が短い方をコピーする。例えば、$\delta(D_1) < \delta(E_1)$, $\delta(D_2) > \delta(E_2)$ なら、Min ルートは図 5 のように決定される。

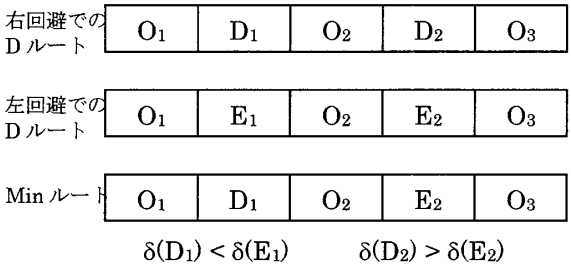

図 5 最短回避走行ルートの生成方式

6. まとめ

走行ルートを表現し、命令の制御をして障害物回避を行い、最短距離走行ルートに訂正することができた。しかし現段階では回避は複雑なもの(斜めに走行し、迂回など)ができない。今後はその部分や e-puck の動作や障害物などの条件を減らし改良する必要がある。

参考文献

1. e-puck.org, http://www.e-puck.org/