B 305 動力学を考慮したシミュレータによるナビゲーションに関する研究

The research of Navigation algorithm using ODE simulator

○ 安東 大地 (日本文理大学), 正 武村 泰範 (日本文理大学)

Daichi Ando, Nippon Bunri University, 1727, Ichigi Oita-city, Oita
Yasunori Takemura, Nippon Bunri University, 1727 Ichigi Oita-city, Oita

Abstract: Recently, in the oceanic survey area, the underwater robot is drawn attention as good tools. However, Autonomous Underwater Vehicle (AUV) cannot use the wireless communication and GPS and so on. Therefore, AUV needs the high-level autonomy navigation system.

Actually, if we develop the real robot or navigation system, we needs a big amount of the developing money. And, If we made the robot, it is very difficult that the robot modify again. In this time, the simulation is attracting attention around the navigation fields. Open Dynamics Engine (ODE) is simulator, which has been used since 2001.

In this paper, we aim to develop the navigation system in the underwater. At first, we made the omni-directional mobile robot in the simulator. And, we simulate the 2 dimensional environment navigation system.

Key Words: Open Dynamics Engine, Navigation system, simulation

1. 序論

近年海洋調査の分野では、水中ロボットの利用が注目されている。特に、遠隔操作・電源供給用のケーブルによる拘束を受けない自律型水中ロボット(AUV)は、広範囲の観測を行う事が出来るため有用である。AUVは、主に環境調査などを主の目的としており、海底の調査を行う為や水質等の環境を調査する為に様々な機器を搭載している[1]。しかし、AUVは水中で活躍する上で、地上で活躍するロボットとは異なり、無線での通信やGPSなどの観測装置を搭載する事が出来ず、高い自律性を必要とする。そこで、水中ロボットにおける大きな課題の1つとして、高度なナビゲーションシステムを搭載する必要がある。

実際にナビゲーションシステムやロボットの開発には決して少なくない開発コストが必要であり、時間もかかる。そして、問題点を修正するためにロボットを何度も製作し直すというのは難しい。そこで、シミュレーションによる開発が注目されている。コンピュータ上でシミュレーションを行えば、実際のロボットを製作し動かさなくてもロボットの動作を確認することができるため開発コストを抑えることができる。

Open Dynamics Engine : ODEは、ラッセル・スミスらが中心となって2001年から開発が進められている物理計算エンジンである。コンピュータゲーム、3Dの製作ソフトシミュレータの物理エンジンとして広く使われている。最大の特徴は、オープンソースでありながら商用レベルの品質を備えていることである。ライセンスはGNUのLGPL(Lesser General Public License)とBSDライセンスの両ライセンスが適用されている。そのため、ODEのライブラリを使用したソースコードは公開する必要がない。ソフトの使用・改変が自由であり、無保障・著作権等を明記すれば再配布もできるなどの特徴がある[2,3]。

ODE自身は、正確性よりも高速性と安定性を重視しており、精度をあまり必要としない用途には最適である。動力学演算部分と衝突検出演算部分から構成されており、簡易描画ライブラリとしてOpenGLをベースに作られたdrawstuffが付属している。これは実行結果を簡易的に描画するのが目的でありODE自体の一部ではないので、drawstuffでの描画を行わずに演算結果の数値のみを取り出す事も可能である。

そこで、本研究では水中ロボットのナビゲーションシステ

ム構築を目標とし、オープンソースの物理計算エンジン(Open Dynamics Engine : ODE)によるロボットのシミュレータ開発を行い、2次元空間を移動する全方位移動ロボットによる障害物回避を含むナビゲートのシミュレーションを行ったので報告する。

2. 全方位移動ロボットの制御理論

2-1. 全方位移動ロボットの開発

全方位移動ロボットの本体(Fig.1)は、実際にあるロボットを参考に全長0.48[m]、重量20[kg]と設定する[4]。車輪は、半径0.1mのオムニホイールを使用した。オムニホイールを前後左右に配置することで、全方位移動を可能にした。シミュレーション上でオムニホイールをそのまま再現すると計算に時間がかかるため、車輪の摩擦係数設定を、回転方向は大きく、軸方向は小さくすることで再現する。

2-2. 全方位移動ロボットの運動学

全方位移動ロボットは前後左右に90度ずつずらしたオムニホイールの4車輪によって移動する。それぞれの車輪は角速度を与えることで回転する。角速度は、現在地から目標地までの距離の偏差に比例して操作量を変化させるP制御によって与えている。このとき、ロボットの最大速度を1.5[m/s]、最大トルクを20.0[N]とする。

Fig.1 Autonomous Mobile Robot

3-1. 障害物回避の制御

障害物回避の方法として、代表的なポテンシャル法を利用した。ポテンシャル法とは、ロボットの進行方向に障害物がある場合に、障害物からは斥力が、目的地からは引力が発生していると仮定し、その合成ベクトルに進むことで障害物を避ける手法である(Fig.2)。

位置 r にいる物体のポテンシャルエネルギー $U(r)$ は物体からの引力と斥力の足し算となり、次式で表せる。

$$U(r) = U_{\text{引力}}(r) + U_{\text{斥力}}(r) \quad <1>$$

$U_{\text{引力}}(r)$, $U_{\text{斥力}}(r)$ にはレナードジョーンズ関数型を使用する。

$$U(r) = -\frac{A}{r^n} + \frac{B}{r^m} \quad <2>$$

A , B は引力・斥力の大きさを表し, n , m は力の強まり方を決め, いずれも正の定数である。

これらは, ロボットと障害物の大きさ, 慣性の影響などに合わせて調整するパラメータである。

2-4. デッドロックの回避

障害物の回避にポテンシャル法を利用するため, 斥力と引力が釣り合った時に同じ動作を繰り返してしまうデッドロックに陥る場合がある。

進行方向を2つの障害物の斥力に囲まれ, 斥力と引力が釣り合った場合のデッドロック回避方法として, 斥力の合成を行った。2つの障害物を範囲に収める新たな斥力を追加することでデッドロックを回避する(Fig.3)。

Fig.4 は障害物を2個設置したフィールド上での斥力の大きさを表したものである。Fig.5 において Fig.4 のように左からロボットが進行してきた場合, 赤色の斥力のくぼみにロボットが入ると斥力と引力が釣り合い, デッドロックが発生する。

Fig.5 は Fig.4 の障害物の斥力を合成した図である。斥力のくぼみが無くなり, デッドロックが発生しにくくなっている。

Fig.2 Obstacle avoidance by potential

Fig.3 Avoiding Deadlock

Fig.4 Repulsive force field

Fig.5 Repulsive force field after the synthesis

3. 全方位置動ロボットのシミュレーション

3-1. 実験内容

・軌道実験

目標地点を四角, 円, 8 の字, 渦巻き模様になるように設置し, 全方位置動ロボットが目標地点に到達すると, 目標となる軌道に対して次の目標地点を設定して, ナビゲートを行わせる。本校では, 実験結果として, 円と8の字をナビゲートしたときに, 使用した式は以下の式となる。

$$\begin{aligned} x &= \cos \theta \\ y &= \sin \theta \end{aligned} \quad <3>$$

$$\begin{aligned} x &= 2 \cos \theta \\ y &= 4 \sin \theta \end{aligned} \quad <4>$$

<3>式は, 円の軌道を示しており, <4>式は, 8 の時のと

きに使用した軌道の式を示している。このとき、目標地点は、 θ を設定し目標地点を設定した。実験結果では、この目標地点を θ の与え方を変更する事でどれくらいの誤差が出るかを調べ、そのときの到達時間を比較検討した。

・障害物回避実験

出発地点から目的地までの間に障害物を設置し、障害物回避を行わせながら目的地までのナビゲートを行う。デッドロック回避のための斥力合成をする場合としない場合の検証も行う。

3-2. 障害物回避実験結果

・障害物1個の場合

(1)初期位置と目的地の間に障害物がある場合を Fig.6 に示す。目的地は $x=0.0, y=10.0$ と設定し、障害物の位置は $x=6, y=0$ とした。障害物を上手く回避している事がわかる。

(2)障害物の位置をマイナス y 方向に移動させた場合の結果を Fig.7 に示す。目的地は $x=20.0, y=0.0$ と設定し、障害物の位置は、 $x=5.5, y=-0.5$ とした。この場合も、上手く回避行動を取っており、(1)の結果に比べ、小さな軌道で回避していることがわかる。

(3)目的地のそばに障害物を設置した場合の結果を Fig.8 に示す。目的地は $x=10.0, y=0.0$ と設定し、障害物の位置は、 $x=9.0, y=0.0$ とした。目標地点に近い為、障害物回避行動が大きく、ゴールへ到達するまでに大きな軌道を描いている。

・障害物2個の場合

(1)進行方向を阻むように障害物を配置した場合を Fig.9 に示す。目的地は $x=10.0, y=10.0$ 、障害物1の位置は $x=5.0, y=6.0$ 、障害物2の位置は $x=6.0, y=5.0$ と設定した。ロボットは、目標地点に向かってまっすぐ進み、障害物に挟まれて、障害物の手前でデッドロックが発生した。

(2)(1)において斥力の合成を行った場合の結果を Fig.10 に示す。斥力が働き、障害物に対して大きく回避行動を取っている事がわかる。

3-3. 軌道誤差実験結果

・円軌道の場合の軌道に対する誤差の割合を Fig.11 に示し軌道を1周するまでの経過時間を Fig.12 に示す。やはり、軌道を与える際の θ が大きくなればなるほど誤差が大きくなり、到達するまでの時間が大きくなる事がわかる。同様に、8の字においての結果をそれぞれ、Fig.13, 14 に示す。こちらの軌道でも同様の結果を示している事がわかる。

Fig.6 One obstacle avoidance 1

Fig.7 One obstacle avoidance 2

Fig.8 One obstacle avoidance 3

Fig.9 Two obstacle avoidance 1

Fig.10 Two obstacle avoidance 2

Fig.11 The error of trajectory obsequence experiment (circle)

Fig.12 The elapsed time of trajectory obsequence experiment (circle)

Fig. 13 The error of trajectory obsequence experiment (a figure of eight)

Fig. 14 The elapsed time of trajectory obsequence experiment (a figure of eight)

4. 考察

軌道実験では、すべての通過点を通過できていたが、綺麗な軌道を描いていたわけではなく、通過点以外では軌道を外れていることが多い。ロボットは通過点まで最短距離で移動しようとするため、曲線の軌道では与えた軌道の精度に依存する。また、ロボットが加速や減速する際に、動力学的計算上、進む方向によっては4つの車輪にそれぞれ違う大きさの負荷がかかり、場合によっては車輪のスリップや空転を招く。そのため、進行方向に影響し誤差がでるのではないかと考えられる。

障害物回避実験では、すべての場合で目的地に到着し、斥力の合成を行うことで近い距離にある複数の障害物も回避できていた。しかし、時間がかかる場合があった。原因としては、障害物に対して真っ直ぐ進行した場合に斥力と引力の方向が重なるため、その方向がずれるまでは短時間の間デッドロック状態になってしまうと考えられる。

軌道誤差比較実験では、どちらの結果もたどった経路は違うものの、通過点の間隔が誤差と到着時間に与える影響には同じ傾向がみられる。目標地点を移動させることで軌道を描かせた場合は、通過点を通過させる場合に比べ、かなり誤差が少ないことがわかる。だが、目標地点に紐で引っ張られているような動きとなるため、速度が速い場合の進行方向の変更や、進行方向の急な変化、目標地点との距離が離れている場合などで誤差が大きくなる。また、目標地点との距離を縮めれば、誤差は少ないものの、時間がかかる。以上の結果より、通過点の間隔や距離の設定が誤差と到着時間のバランスをとるうえで重要であることがわかる。

5. 結論

2次元空間での全方位移動ロボットにおけるポテンシャル法による障害物回避を含むナビゲートの有用性を、シミュレーションによって確認した。

実験の結果から今後の課題として、障害物回避時の時間短

縮、障害物を回避できず目的地に到着できない場合への対処、速度や重量が変化した場合の動作への影響などが挙げられる。また、今後の展望としては、全方位移動ロボットにおける様々な設定の最適な値の模索、目的地により早く到着するための最適な移動方法の模索、障害物が移動していた場合への対応、ロボットの重心が変化した場合の考慮などが考えられる。

以上が本研究の結論であるが、シミュレーション自体の有用性を判断するためには実際のロボットでの動作と比較しなければならない。その上でシミュレータを精査することで本来あるべき現実により近いシミュレーションが可能になるのではないかと考えられる。

参考文献

- [1] 日本出村 公成: “簡単! 実践! ロボットシミュレーション” 森北出版株式会社(2007)
- [2] 出村公成: “Open Dynamics Engine を用いた教育用シミュレータと教科書の開発”, ロボティクス・メカトロニクス講演会’ 07, 1A1-G09 (2007)
- [3] 出村公成: “Open Dynamics Engine によるロボットプログラミング教育”, 第24回日本ロボット学会学術講演会, 1L34 (2006)
- [4] Y.Takemura, Amir A.F. Nassiraei, “A System Design Concept Based on Omni-Directional Mobility, Safety and Modularity for an Autonomous Mobile Soccer Robot”, Journal of Bionic Engineering Suppl., pp.121-129, 2008