区分多項式型の軌道生成アルゴリズムを用いた 実時間障害物回避

正員 松 永 信 智 (熊本大)

正員 川 路 茂 保 (熊本大)

Real-Time Obstacle Avoidance with Reference Generation Algorithm using Piecewise Polynomials

Nobutomo Matsunaga, Member, Shigeyasu Kawaji, Member (Kumamoto University)

This paper presents a real time obstacle avoidance algorithm with the reference generator using piecewise polynomials. The algorithm consists of three steps. At the first step, by using the future information of path, the interaction point of the pilot point and obstacles are checked. At the second step, the intermittent point data for the trajectory generator are generated not to collide with the obstacles, where the collision point is projected on an arbitrary point on the circumference at a distance from the obstacles. At the third step, using the new data a collision free path is generated by the reference generator using piecewise polynomials. Some results of computer simulation are shown to evaluate the effectiveness of the proposed method.

キーワード：障害物回避，区分多項式，目標軌道生成，パイロットポイント

1. ま え が き

作業空間に存在する様々な障害物を回避する技術は，作業の安全を確保し能率を向上させるといった意味で重要である。障害物回避動作をもつ制御系を構成する方法として現在までに様々な観点から研究が行われており，これらは大きく二つの方法に大別できる。一つの方法は，コンフィグレーション空間法⁽¹⁾に代表される作業計画からのアプローチである。このアルゴリズムの特徴は安全で効率の良い軌道を幾何学的に解くことであるが，高速の回避動作が実現できない，実時間処理に向いていないといった欠点がある。もう一つは，障害物の近辺に設定した人工ポテンシャル⁽²⁾に基づいて回避する方法であり，その特徴は突発的な状況に対処できる点にある。しかしながら，障害物と制御対象の距離関係で得られる反力などの局所的な情報を用いることにより高速な回避は実現できるが，回避動作を明確に把握できない，設定した最終位置に到達できない場合があるという欠点がある。

ロボットや移動機械に作業を指示する場合，オペレータは作業のキーポイントになる間欠的な目標位置，時刻データを与える。ところが，モーションコントロールの観点から考えると，高速高精度な制御を実現するには高階微分情報を含む軌道を生成する必要がある。このような問題に対して，著者らは区分多項式型の軌道生成手法を提案し，モーションコントロールに応用し有効性を明らかにした⁽³⁾。このアルゴリズムは，システムの初期値を入換えるだけで任意の多項式軌道を生成できるという特徴をもち，逐次軌道を変化させる問題に適している。

本論文では，区分多項式型の目標軌道生成器を用いた障害物回避アルゴリズムを提案する。そのために，まず区分多項式型目標軌道生成器の構成および特徴を明らかにする。このアルゴリズムはオペレータから与えられた離散的な点列データを基に目標軌道を生成するものであり，現在時刻前後の3点でのデータを用いるだけで区分的に連続な多項式軌道を実時間生成することが可能である。

また、軌道の生成ステップが初期値計算と初期値に基づく実時間軌道生成の部分に分割されているという特徴をもっている。次に、このような目標軌道生成器の特徴を生かした障害物回避アルゴリズムを提案する。本アルゴリズムは、目標軌道生成器の初期値ベクトルの計算に用いる離散点データを作業空間の障害物を考慮して生成するものである。例えば、指定する通過点が障害物内にある場合はその離散データを修正し、指定する通過点が障害物内でない場合は障害物を効果的に避けるような離散データを自律生成する。具体的には、オペレータの与える離散データに基づき発生する目標軌道と作業領域の障害物の位置情報から未来の衝突地点を求め、その点を障害物の近辺に設定した円領域に写像した点に軌道を修正することによって回避軌道を実現する。最後に提案する障害物回避アルゴリズムの有効性を、シミュレーションによって検証する。

2. 実時間軌道生成アルゴリズム

作業計画を行う際に、作業に即した作業点あるいは経過点をオペレータは何らかの形で教示しなければならない。いまオペレータの与える間欠的なデータ列を、次に示すような不等間隔の時刻 $\hat{t}_i$ と作業座標データ $\hat{P}_i$ の N_u 個の組とする。

$$(\hat{t}_i, \hat{P}_i) \quad (i=1, 2, \dots, N_u)$$

ここで、与える時刻データの領域 $(\hat{t}_i, \hat{t}_{i+1})$ をユーザーモードのセグメント、 $(\hat{t}_i, \hat{P}_i)$ をユーザーモードの節点データ (U データ) と呼ぶ。

このような間欠データ列から作業に必要な連続的な軌道を生成する最も直感的な手法として、オフラインで計算しておいた軌道データを制御実行時にピックアップする手法が考えられる。しかしながら、この場合コントローラとしての計算機の機能の大部分を膨大な軌道データで占有するという欠点がある。このような理由から、現在では何らかの形で実時間で離散点を補間するアルゴリズムが用いられている。離散点を実時間で補間するアルゴリズムに対して、

- (1) 滑らかな軌道が生成できること
- (2) 実時間計算に向いていること
- (3) 軌道情報が常に取出し可能なこと

の三つの条件は少なくとも必要である。ここで、(1)の滑らかさはオペレータの要求によって速度や加速度のレベルで時間に関して連続となる軌道を生成することである。(2)は計算アルゴリズムが少ないステップでできており、逐次計算に向いていることである。(3)は軌道の位置だけでなく動的制御に必要な速度、加速

度などの情報が常に利用できることである。以上の(1)～(3)の条件を満足したうえで更に高速高精度な制御を実現するために、目標軌道生成器は現代制御理論で扱いやすいように状態方程式表現されているほうが望ましい。

<2・1> 連続時間軌道生成システムの構成⁽³⁾

いま、高階微分情報を含む目標軌道の状態を

$$\xi = \begin{bmatrix} P(t) \\ \dot{P}(t) \\ \vdots \\ P(t)^{(\mu+1)} \end{bmatrix} \in R^{n(\mu+2)} \quad \dots\dots\dots (1)$$

ここで、 ξ : 線形自由系とする。

$$\left. \begin{aligned} \dot{\xi}(t) &= A_r \xi(t) \\ P(t) &= C_r \xi(t) \end{aligned} \right\} \quad \dots\dots\dots (2)$$

$$: \xi(t_i) = \xi_i$$

の状態ベクトルである。ただし、 $A_r \in R^{n(\mu+2) \times n(\mu+2)}$, $C_r \in R^{n \times n(\mu+2)}$ である。(2)式を時刻 t_i に初期状態 ξ_i で自由応答させた結果、時間領域で得られるデータは

$$P(t) = C_r e^{A_r(t-t_i)} \xi_i, \quad t \in (t_i, t_{i+1}) \quad \dots\dots\dots (3)$$

となる。このデータの生成に引続いて次の時刻 t_{i+1} に初期状態 ξ_{i+1} を与えると、それ以後は前のセグメントとは独立なデータが得られる。ここで、差換える時刻と初期状態のデータ列

$$(t_i, \xi_i) \quad (i=1, 2, \dots, N_m)$$

をマシンモードの節点データ (M データ) と呼ぶ。

これらの初期状態 ξ_i の間に

$$\begin{bmatrix} C_r \phi_i \\ C_r A_r \phi_i \\ \vdots \\ C_r A_r^\mu \phi_i \end{bmatrix} \xi_i = \begin{bmatrix} C_r \\ C_r A_r \\ \vdots \\ C_r A_r^\mu \end{bmatrix} \xi_{i+1} \quad \dots\dots\dots (4)$$

$$\phi \equiv e^{A_r(t_{i+1}-t_i)}$$

なる関係があれば、 $(\hat{t}_i, \hat{t}_{N_u})$ の領域で生成されるデータは滑らかさ指数 μ の連続データとなる。ここで、滑らかさは軌道の μ 階微分情報が時間に関して連続であることを意味しており、 $\mu=1$ ならば速度連続、 $\mu=2$ ならば加速度連続である。ここで A_r を

$$A_r = \begin{bmatrix} 0 & I_n & 0 & \cdots & 0 \\ 0 & 0 & I_n & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & I_n \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix} \quad \dots\dots\dots (5)$$

と選べば、生成されるデータは時間に関して区分的な多項式として表すことができる。得られる軌道は階段関数関数を数回積分して得られる簡単な関数ではあるが、各セグメントが独立であることから、組合せにより複雑な軌道を表現することができる。例えば、

$$P(t) = a_0 + a_1 t + a_2 t^2 \dots \dots \dots (6)$$

の軌道は

$$\xi(0) = \begin{bmatrix} P(0) \\ \dot{P}(0) \\ \ddot{P}(0) \end{bmatrix} = \begin{bmatrix} a_0 \\ a_1 \\ 2a_2 \end{bmatrix} \dots \dots \dots (7)$$

とおいたときに $\xi(t)$ の (1, 1) 成分となっている。なお、 A_r の選択により円軌道を生成することも可能である⁽³⁾。

〈2・2〉 初期ベクトルの選定 ロボットや移動機械などメカニカルシステムに最もよく用いられる目標軌道は速度連続な軌道である。ここでは、速度連続な軌道を生成する Taylor の手法⁽⁴⁾ を前節の目標軌道生成器を用いて実現する問題を考える。Taylor のアルゴリズムは、

- (i) 与えられた U データを速度一定の直線で結び、折れ線軌道を生成する。
- (ii) 節点の前後に加減速時間（コーナセグメント）を設定し、残ったセグメント（リニアセグメント）との境界で適当な連続性が満たされるように多項式で補間する。

という 2 段階のステップで生成される。このアルゴリズムの特徴は、 U データの前後に一定の加減速時間 t_{acc} を設定することによりアルゴリズムの簡単化を行っている点にある。

速度連続な目標値を線形自由システムで生成する場合は、まず U データから M データを生成する必要がある。図 1 に横軸に時間、縦軸に位置をとった場合の U データと M データの関係を示す。 M データは $N_m = 2N_u$ 個の冗長な情報を含んだ節点データであり、図 1 において時刻 $\hat{t}_i$ の前後 t_{acc} で初期値をそれぞれ次のように入換えることにより速度連続な軌道を生成することができる。

$$(t_j, \xi_j) = \begin{cases} (t_i - t_{acc}, \xi_i^-) & (j=2i+1) \\ (t_i + t_{acc}, \xi_i^+) & (j=2i) \end{cases} \dots (8)$$

ただし、

$$\begin{cases} \xi_i^- = \xi_i(t_i - t_{acc}) \\ \xi_i^+ = \xi_i(t_i + t_{acc}) \end{cases} \dots \dots \dots (9)$$

であり、それぞれコーナセグメントの始点、リニアセグメントの始点を示す M データである。 M データの

図 1 U データと M データの関係

Fig. 1. The relationship between U -data and M -data.

間を速度連続 ($\mu=1$) な軌道で補間するには、(4) 式の拘束条件を用いると次のように求まる⁽³⁾。

$$\xi_i^- = \begin{bmatrix} \hat{P}_i - \frac{t_{acc}}{T_i} \Delta P_i \\ \frac{\Delta P_i}{T_i} \\ \frac{1}{2t_{acc}} \left(\frac{\Delta P_{i+1}}{T_{i+1}} - \frac{\Delta P_i}{T_i} \right) \end{bmatrix} \dots \dots \dots (10)$$

$$\xi_i^+ = \begin{bmatrix} \hat{P}_i + \frac{t_{acc}}{T_{i+1}} \Delta P_{i+1} \\ \frac{\Delta P_{i+1}}{T_{i+1}} \\ 0 \end{bmatrix} \dots \dots \dots (11)$$

ただし、

$$\begin{cases} \Delta P_i = \hat{P}_i - \hat{P}_{i-1} \\ T_i = \hat{t}_i - \hat{t}_{i-1} \end{cases} \dots \dots \dots (12)$$

である。

ここで、軌道生成に必要な M データが、たかだか 3 時刻分のデータであることに注意する。つまり、 i 番目の軌道生成に必要なデータは $i-1, i, i+1$ 時刻での U データさえわかればよく、図 2 に示す手順により目標軌道が生成できる。同図において、まずオペレータの与えた点列データから 3 時刻分の U データを取出し、(10)~(12) 式に従って M データを生成する。次に得られた M データと (2) 式からオンラインで目標軌道を生成し、モーションコントローラに目標値として与える。このように、初期値計算と実時間での目標軌道生成のブロックが自然な形で分割されていることは、障害物回避のような軌道変更をたびたび行うシステムにとっては有効である。

なお、図 1 において指定した U データ上を経由しないのは CP (Continuous Path) を生成しているた

図 2 オンライン軌道生成手順
Fig. 2. The procedure of on-line reference generation.

めで、マニピュレータや移動ビークルなどの経路制御に用いられる。なお本手法は、 U データとして $(\hat{t}_k, \hat{P}_k)$, $(\hat{t}_{k+1}, \hat{P}_{k+1})$ のように異なる時刻で同じ位置を指定することによって、指定した点を通過する PTP (Point To Point) 動作も実現可能である。

3. M データ変更による回避アルゴリズム

区分多項式型の目標軌道生成器を用いて障害物回避を行う問題を考える。突発的に障害物が現れたときの最も一般的な対処法は、その場で停止し指示があるまで待機する方法である。しかし、生産性向上という意味では、障害物を回避する軌道を自律生成し、作業を継続する機構が必要である。このような機能を実現するためには、障害物回避動作中の目標値は例外処理的なものではなく、回避動作中も制御中と同じ滑らかさをもつ目標値を発生する必要がある。ここでは障害物回避動作に対して、第 2 章で述べた区分多項式型の目標軌道を生成するための、 M データの計算手法を明らかにする。

〈3・1〉 リニアセグメントでの回避 リニアセグメント区間は等速度で加速度は零である。従って軌道を変更するためには、 M データ間の自由応答を終了させ、新しい M データとの間で軌道生成すればよい。この際、速度連続の条件から M 節点の前後では滑らかさ $\mu=1$ で連続でなければならない。この条件を満たすために、軌道変更を開始させる点の t_{acc} 後の軌道上に U データの仮想点 $U_v(\hat{t}_v, \hat{P}_v)$ を考える。ここで、 $\hat{t}_v, \hat{P}_v$ はそれぞれ

$$\left. \begin{aligned} \hat{t}_v &= \hat{t}_{i-1} + t_0 + t_{acc} + t_{cal} \\ \hat{P}_v &= \hat{P}_{i-1} + (\hat{t}_v - \hat{t}_{i-1}) \frac{\Delta P_i}{T_i} \end{aligned} \right\} \dots\dots\dots (13)$$

で表され、 t_0 は現在時刻、 t_{cal} は新しい軌道の M データを計算するのに要する時間である。

滑らかさ $\mu=1$ の条件を満たしつつ、変更点 $U_i(\hat{t}_i, \hat{P}_i)$ へ到達するための M データは

図 3 リニアセグメントでの変更
Fig. 3. The modification in linear segment.

$$\xi_v^- = \begin{bmatrix} \hat{P}_v - \frac{t_{acc}}{T_i} \Delta P_i \\ \frac{\Delta P_i}{T_i} \\ \frac{1}{2t_{acc}} \left(\frac{\Delta P_i'}{T_i'} - \frac{\Delta P_i}{T_i} \right) \end{bmatrix} \dots\dots\dots (14)$$

$$\xi_v^+ = \begin{bmatrix} \hat{P}_v + \frac{t_{acc}}{T_i'} \Delta P_i' \\ \frac{\Delta P_i'}{T_i'} \\ 0 \end{bmatrix} \dots\dots\dots (15)$$

となる。ただし、

$$\left. \begin{aligned} \Delta P_i' &= \hat{P}_i' - \hat{P}_v \\ T_i' &= \hat{t}_i - \hat{t}_v \end{aligned} \right\} \dots\dots\dots (16)$$

図 3 にリニアセグメントでの変更手法を示す。同図において、 $\hat{t}_{i-1} + t_0$ 時刻で障害物が検出されるものとする (C 点)。まず、 $\hat{P}_{i-1}$ から $\hat{P}_i$ の線分上の $t_{acc} + t_{cal}$ 時刻後に仮想点 U_v を設定する。次に、仮想点と T_i 後の回避点 P_i' の間に加減速時間を設定し回避動作を行う。この場合、オペレータが設定した初期軌道から逸脱するのは $[\hat{t}_{i-1} + t_0 + t_{cal}, \hat{t}_{i+1}]$ の区間である。なお、変更する U データの時刻は軌道生成の計算時間遅れを考慮して、仮想点 U_v の $2t_{acc}$ 以後でなければならない。

〈3・2〉 コーナセグメントでの回避 コーナリングセグメントでは軌道上の各点における速度が異なるために、リニアセグメントと同様に滑らかさ $\mu=1$ の軌道変更を即座に行うのは困難である。従って、コーナリングセグメント終了後に軌道変更させることにする。そこで、コーナリング終了時から t_{acc} 後に仮想点 $U_v(\hat{t}_v, \hat{P}_v)$ を考える。ただし、

図 4 コーナセグメントでの変更
Fig. 4. The modification in corner segment.

$$\left. \begin{aligned} \hat{t}_v &= \hat{t}_{i-1} + 2t_{acc} \\ \hat{P}_i &= \hat{P}_{i-1} + \frac{2t_{acc}}{T_i} \Delta P_i \end{aligned} \right\} \dots\dots\dots (17)$$

変更点 $U'(\hat{t}_v, \hat{P}_i)$ に到達するための M データは

$$\xi_v^- = \begin{bmatrix} \hat{P}_v - \frac{t_{acc}}{T_i} \Delta P_i \\ \frac{\Delta P_i}{T_i} \\ \frac{1}{2t_{acc}} \left(\frac{\Delta P_i'}{T_i'} - \frac{\Delta P_i}{T_i} \right) \end{bmatrix} \dots\dots\dots (18)$$

$$\xi_v^+ = \begin{bmatrix} \hat{P}_v + \frac{t_{acc}}{T_i'} \Delta P_i' \\ \frac{\Delta P_i'}{T_i'} \\ 0 \end{bmatrix} \dots\dots\dots (19)$$

となる。ただし、

$$\left. \begin{aligned} \Delta P_i' &= \hat{P}_i - \hat{P}_v \\ T_i' &= \hat{t}_i - \hat{t}_v \end{aligned} \right\} \dots\dots\dots (20)$$

である。コーナリングセグメントでの変更手法を図 4 に示す。同図において、加減速区間で障害物を検出したとする (C 点)。まず、 $\hat{P}_{i-1}$ 前後の加減速動作が終了するまで待ち、 $\hat{t}_{i-1} + t_{acc}$ 時刻から即軌道変更に入る。このときの仮想点 U_v は、 $\hat{P}_{i-1}$ から $\hat{P}_i$ の線分上の $2t_{acc}$ 時刻後に設定される。この結果は変更時刻のみが異なるのみでエリアセグメントでの変更手法と一致する。

4. 実時間障害物回避アルゴリズム

〈4・1〉 障害物の記述 障害物回避問題を考える場合、形状、大きさ、位置、方向など障害物に関する情報をどのように記述し扱うかが問題になる。障害物の扱いとしては、なるべく物体を忠実に扱うために多角形で近似したソリッドモデルを扱う手法があるが、

図 5 障害物の近似表現
Fig. 5. An approximation of obstacle.

演算時間が膨大になる、凹多面体で近似された場合のゴールへの到達解を求める問題など多くの問題がありオンライン処理には不向きである。オンラインでの障害物回避を考える場合は、障害物はなるべく簡単に扱えるほうが好ましく、円形近似⁽²⁾あるいは楕円体近似⁽⁶⁾が用いられてきた。ここではアルゴリズムに重点を置き、近似法は図 5 に示す簡単な円近似を採用する。この円近似は、近似精度が粗くなる反面

- (i) 形状および大きさが半径のみで記述できる
 - (ii) 方向は関係なく中心位置を扱うだけでよい
- といった利点をもっている。

〈4・2〉 パイロットポイントを用いた U データの自律発生 オンラインで障害物を回避する場合、障害物の近辺に構成した人工ポテンシャル場からの反力をサーボ系の外乱として処理することにより、回避軌道を生成する手法が多く用いられている。このような局所的な回避は、不慮の事態に対応するという意味では重要であるが、制御対象の状態によってどのように回避するかわからない。従って、局所的な回避だけでなく、あらかじめわかっている静的な障害物に対しても有効なアルゴリズムは回避経路のプランニングという点でも重要である。

このような問題に対する一つの解決法としては、第 3 章で述べた各セグメントでの M データ変更アルゴリズムを用いて障害物回避を行う手法が考えられる。つまり衝突状態を検出後、軌道上に仮想点 $U_v(\hat{t}_v, \hat{P}_v)$ を即座に作成し、新しい U データに対して軌道生成する方法である。この方法は、少なくとも $2t_{acc}$ 以後に計画した回避動作に入れるという点では優れているが、高速動作が要求される状況では回避動作が間に合わない場合が生じる。従って、軌道が障害物にどこでどのように衝突するかという情報をあらかじめ検出し、回避動作を予見的に行う必要がある。そこで障害物の障害物との衝突情報を予見的に得る方法として、図 6 に示すように、現在の位置 (E 点) と別に、

図 6 E 点と P 点の時間関係
Fig. 6. The time relation of point E and P.

常に t 秒先の軌道を走るパイロットポイント (P 点) を導入する。一般的に、U データの時間間隔 $\hat{t}_i = \hat{t}_i - \hat{t}_{i-1}$ は不均一なので、パイロットポイントは E に対して少なくと $t = \max(\hat{t}_i, 1 \leq t \leq N_u)$ の先の情報を検出する必要がある。これは、障害物と衝突する少なくとも一つ前のノードから回避運動を始める必要があるからである。例えば、同図で $\hat{P}_i$ と $\hat{P}_{i+1}$ のセグメントで障害物と衝突する場合には、図中の D 点以前で回避軌道に移る必要がある。また特殊な場合として、時間間隔が均一な場合は $t = \hat{t}_i - \hat{t}_{i-1}$ となり、E 点が M 節点を通過するとき、P 点は二つ先の M 節点を通過する。また、E 点がリニアセグメントを通過するとき、P 点もリニアセグメントを通過することになる。従って、障害物と衝突する位置と時刻の情報を基に障害物を回避する新しい U データを求め、E 点の次のコーナリングセグメントから障害物回避動作を行えばよい。なお、この場合軌道上に仮想 U データを作成する必要はなく、(13)式において $\hat{P}_v = \hat{P}_i$, $\hat{t}_v = \hat{t}_i$ として (14), (15) 式を適用すればよい。

ここで、具体的なアルゴリズムを明らかにする前に、もう一度回避軌道生成の目的について考えてみると、

[M₁] 障害物に衝突しない目標軌道を発生することから、障害物を回避する U データを発生すればよい。しかしながら、作業計画の性質上

[M₂] 設定軌道からなるべく離れない

といった制限が加わる。このような制御目的をふまえたうえで、図 7 に示すような仮想点つまり新しい U データを発生する手法を提案する。同図において、パイロットポイントの衝突点は障害物の中心から任意の半径 R の円周上に写像され新しい U データとなる。この結果、設定値はただか円周の半径だけしかずれず、軌道生成問題とは独立に、回避軌道の評価を行うことができる。また、障害物を回避する際の U データの発生に関しては、次の規則に従うものとする。

R 1. 円内に U データがない場合には、 t 秒後のデータとして新しい U データを発生する。

図 7 P 点写像による軌道生成
Fig. 7. Path generation using P-point projection.

図 8 写像法 1
Fig. 8. The concept of projection method 1.

R 2. 円内に U データがある場合は、その U データはキャンセルし新しい U データを発生する。

以下では、障害物と P 点の衝突座標 (x_p, y_p) を用いた具体的な写像の例として、衝突点を写像する二つの方法を示す。なお障害物に関する情報として、障害物の中心座標 (x_0, y_0) 、半径 r_0 、データ設定エリアの半径 R が既知であるとする。

〔写像法 1〕 障害物回避を行う際に、衝突点と障害物の中心を通る直線上に回避点を作成する手法であり、新しい U データは

$$x = x_0 + (x_0 - x_p) \frac{R}{r_0} \dots\dots\dots (21)$$

$$y = \begin{cases} y_0 + \sqrt{R^2 - (x_0 - x)^2} & (y_p \leq y_0) \\ y_0 - \sqrt{R^2 - (x_0 - x)^2} & (y_p \geq y_0) \end{cases} \dots\dots (22)$$

で変更される。概念を図 8 に示す。

この方法はアルゴリズムとしては簡単であるが、目標軌道が障害物と交わる領域が大きければ大きいほど回避のための時間がかかるという欠点がある。これは、図 8 において OP_g の線分と $\hat{P}_{i-1}\hat{P}_i$ の線分の成す角 (図中 α) が小さいほど $P_g\hat{P}_i$ の線分が再び障害物と交わる回数が多くなるからである。最悪の場合、線

分 $\hat{P}_i \hat{P}_{i+1}$ が障害物の中心を通るとき $\alpha=0$ となり、回避することができない。

〔写像法 2〕 衝突点と障害物の関係と、現在地点と回避後の U データを結ぶ直線と障害物の関係を用いて、写像する手法を示す。本手法は、多くの情報を用いて高速な回避を実現するものであり、〔写像法 1〕の欠点を克服するものである。まず、衝突時と障害物の位置関係を示す次の記号を定義する。

衝突点が障害物の中心に対して上: $PO_y=1$

衝突点が障害物の中心に対して下: $PO_y=-1$

衝突点が障害物の中心に対して右: $PO_x=1$

衝突点が障害物の中心に対して左: $PO_x=-1$

また、現在地点と U データを結ぶ直線と障害物との関係として、

結んだ直線が障害物の上: $LO_y=1$

結んだ直線が障害物の下: $LO_y=-1$

を定義する。情報の組合せから、新しい U データを次のように設定する。

(1) $PO_y=1$ and $LO_y=1$ のとき、 P 点を y 軸に平行に上側に写像する。

$$\left. \begin{aligned} x &= x_p \\ y &= y_0 + \sqrt{R^2 - (x_0 - x_p)^2} \end{aligned} \right\} \dots\dots\dots (23)$$

(2) $PO_y=-1$ and $LO_y=-1$ のとき、 P 点を y 軸に平行に下側に写像する。

$$\left. \begin{aligned} x &= x_p \\ y &= y_0 - \sqrt{R^2 - (x_0 - x_p)^2} \end{aligned} \right\} \dots\dots\dots (24)$$

(3) $PO_y \cdot LO_y = -1$ and $PO_x=1$ のとき、 P 点を x 軸に平行に右側に写像する。

(1) 上部写像

(2) 下部写像

(3) 左写像

(4) 右写像

図 9 写像法 2

Fig. 9. The concept of projection method 2.

$$\left. \begin{aligned} x &= x_0 + \sqrt{R^2 - (x_0 - x_p)^2} \\ y &= y_p \end{aligned} \right\} \dots\dots\dots (25)$$

これらの写像を図示すると図 9 となる。

(4) $PO_y \cdot LO_y = -1$ and $PO_x = -1$ のとき、 P 点を y 軸に平行に左側に写像する。

$$\left. \begin{aligned} x &= x_0 - \sqrt{R^2 - (x_0 - x_p)^2} \\ y &= y_p \end{aligned} \right\} \dots\dots\dots (26)$$

障害物回避軌道を生成するユーザデータ変更方法として、〔写像法 1〕と〔写像法 2〕の二つの方法を示したが、「障害物に衝突しない目標軌道を発生する」といった障害物回避の初期目的を満足する回避点決定は無数に存在する。従って、写像アルゴリズムや写像半径 R の決定の選択にあたっては、オペレータが何らかの基準に従って適切に定める必要がある。

5. シミュレーション結果

図 10 に示すように作業エリアを縦 0.5 m, 横 1.0 m の長方形のスペースとし、 X - Y 平面を移動する物体の障害物回避問題を考える。まず、オペレータが表 1 に示すようなデータを初期設定するものとする。障害物回避のアルゴリズムが用いられなければ、その目標軌道は図 10 のようになり、障害物に衝突する。

ここで、障害物の大きさおよび位置が ITV などの

図 10 オペレータの与える目標軌道

Fig.10. Initial path using the data on Table 1.

表 1 初期設定する U データ
Table 1. Initial point specified by operator.

Time[s]	X [m]	Y [m]
0.0	0.0	0.0
1.0	0.0	0.0
2.0	0.1	0.2
3.0	0.6	0.3
4.0	1.0	0.5
5.0	1.0	0.5

表 2 障害物データ
Table 2. Obstacle information.

No	X[m]	Y[m]	r_0 [m]
1	0.25	0.20	0.07
2	0.50	0.35	0.05
3	0.80	0.35	0.05

図 11 オンライン変更された目標軌道
(写像法 1)

Fig. 11. The collision-free path using projection method 1.

図 12 オンライン変更された目標軌道
(写像法 2)

Fig. 12. The collision free path using projection method 2.

外界センサを用いて表 2 のように得られたものとする。写像半径 R を $R=2r_0$ とし、本論文で提案した〔写像法 1〕の回避アルゴリズムを用いたときの回避軌道を図 11 に示す。また、〔写像法 2〕による回避軌道は図 12 のようになる。更に比較として文献 (8) の仮想モデルを用いた局所的な方法を適用した場合の回避軌道を図 13 に示す。

〔写像法 1〕(図 11) と〔写像法 2〕(図 12) を比べると、〔写像法 1〕がゴールに達成するまでに 9 秒かかっているのに対し、〔写像法 2〕は 8 秒で到達可能である。この差は、与えられた軌道の障害物への衝突具合によ

図 13 局所的な障害物回避 (仮想モデル)

Fig. 13. The local collision free path using VIM ($M=0.2$ kg, $K=245$ kg/s², $D=14$ kg/s).

て顕著になる。このことは、回避点の決定の際に多くの情報をどのように利用するかが障害物回避の鍵になることを示している。

また、図 13 の局所的な回避軌道と図 11、図 12 の結果を比較すると、局所的な回避が障害物からの反発力によって明確な回避計画が困難であるのに対し、本手法は Taylor の速度連続軌道の拡張として障害物を回避する経路が自律的に生成されているのがわかる。

6. む す び

本論文では、区分多項式型の目標軌道生成器に与える初期値データを差換えることにより障害物回避を行う手法を示した。軌道生成アルゴリズムは、モーションコントローラに与える滑らかな軌道を生成する部分と、オペレータの与えた離散点を滑らかにつなぐ初期値発生部から構成されており、区分的に連続な多項式軌道を発生することができる。従って、軌道生成と別に、障害物回避を実現する初期値発生問題を考えることができ、階層的に問題解決を行うことができる。

提案した障害物回避アルゴリズムの特徴は、軌道中に仮想点を発生し障害物回避を実現する新しい U データの間で、即時に任意の滑らかさを有する軌道を生成できることである。この障害物回避手法は、制御対象と障害物の間の相対関係しかわからない場合に有効である。障害物や制御対象の絶対位置が既知の場合は、パイロットポイントを発生させ未来情報を得ることにより予見的に対処でき、軌道修正アルゴリズムはさらに簡単になるという特徴がある。

回避アルゴリズムの鍵となるのが U データの自律発生機構であり、ここでは〔写像法 1〕と〔写像法 2〕の二つの方法を示したが、「障害物に衝突しない目標軌道を発生する」といった障害物回避の初期目的を満足す

る回避点決定には無数に存在する。従って、写像アルゴリズムや写像半径 R の選択にあたっては、障害物回避の制御目的が定量的に与えられない限り、オペレータの何らかの基準に従って適切に定める必要がある。このことは、回避軌道計画においては提案した幾何学的な写像法のみならず、直感や経験などの知識ベースを利活用することを示唆する。提案した手法が線形制御理論に基づく動的な軌道生成と静的な回避点の生成の二つのブロックに分かれていることを考えると、本質的にファジィやニューラルネットなど知識利用の手法と競合するものではなく、むしろうまく利活用する方向にあるものと考えられる。

(平成4年1月10日受付, 同4年5月6日再受付)

文 献

- (1) 尾崎:「マニピュレータの障害物回避」, ロボット学誌, 2, No. 6, 76 (昭59-12)
- (2) O. Khartib: "Real Time Obstacle Avoidance for Manipulators and Mobile Robots", *Int. J. Robotics Research*, 5, No. 1, 90 (1986)
- (3) 川路・汐月・松永:「区分多項式を用いたマニピュレータの制御」, 電学論D, 107, 5 (昭62-1)
- (4) R. H. Taylor: "Planning and Execution of Straight Line Manipulator Trajectories", *IBM J. Res. & Development*, 27-4, 152 (1979)
- (5) 奥富・森:「ポテンシャル場を用いたロボットの動作決定」, ロボット学誌, 1-3, 66 (昭60-10)
- (6) T. Lozano-Perez: "Automatic Planning of Manipulator Transfer Movement", *IEEE Trans. SMC*, SMC-11, No. 10, 681 (1981)

- (7) 松永・川路:「区分多項式を用いた障害物回避アルゴリズム」, 第10回計測自動制御学会九州支部学術講演会予稿集, p. 287 (平3)
- (8) 小菅・古田・横山:「ロボットの仮想内部モデル追従制御系」, 計測自動制御学論集, 24, 55 (昭63-1)

松 永 信 智 (正員)

昭和37年10月2日生。62年3月熊本大学大学院工学研究科電子工学専攻修了。同年4月オムロン(株)入社。平成2年4月熊本大学大学院自然科学研究科博士課程入学, 現在に至る。ファジィ制御など知的制御系設計に関する研究に従事。計測自動制御学会会員。

川 路 茂 保 (正員)

昭和19年1月18日生。44年3月熊本大学大学院工学研究科電気工学専攻修了。同年4月同工学部助手, 同助教授を経て, 現在電気情報工学科教授。工学博士。多変数制御理論, 知識制御系などの設計法およびロボットへの応用に関する研究に従事。IEEE 会員。