四足歩行ロボットの準動的歩行*

滝田 好宏^{*1}, 背戸一登^{*1}, 長松 昭男^{*2}

The Semi-Dynamical Walking of a Quadrupedal Robot

Yoshihiro TAKITA, Kazuto SETO, and Akio NAGAMATSU

The previous report shows a quadrupedal walking robot which has a limitation on its walking speed because of walking static balanced with three legs by movement of the center of gravity. This paper proposes a new walking method, that is, semi-dynamical walking with two-legs-support, such as horses. For a realization of this kind of walking, it is necessary that each ankle of the legs has a control mechanism on the roll angle, because they support the position in walking. In this paper, the walking pattern is studied through analyzing the dynamic stability in walking, and developed walking control programs on a software servo control system which is broken up in order to control the micro-processors. The experiment for a quadrupedal walking robot shows that this kind of walking is 4 times faster than the previous kind of walking.

Key Words: Application of Control, Quadrupedal Walking Robot, Semi-dynamical Walking, Software Servo Controller, Micro-Processor

1. 緒 言

移動機能を持った知能ロボットは、人間に代わる労働力として工場のみならず農業、サービス業などにおいてその要望が高まってきていることから、近年移動ロボットの開発が盛んになってきている。これは、マイクロエレクトロニクス技術の発達により、小形で高性能なマイクロプロセッサを機械に組み込むことができるようになったためである。

地上で用いる産業用の移動ロボットは比較的課題が少ない車輪式のものが実用化されているが、その使用範囲がほぼ平面に限られているために、凹凸がある場合には大きな問題になっている。一方、脚式では表面状態を選ばず平滑化して移動できるが、この種のロボットのほとんどが研究段階である。脚式移動ロボットの研究は1脚から8脚^{(1)~(7)}までさまざまであるが、この中で4脚式は安定性、積載性、作業性から他の脚数より実用性が高いと思われる。

そこで、すでに前報⁽⁸⁾では牛馬形四足歩行ロボットについて、歩行パターンを与えることで静的歩行を実

現しているが、歩行速度は0.011 m/sと低速であった。これは、3脚では体重を支えるために、接地する足が描く三角形内に重心を移動する動作を4回繰返すことにより1歩進むことができるが、このような重心の移動を繰返すために歩行の高速化は望めないのが、高速化ができる機構と歩行方法の開発が必要になる。

そこで本研究では、前報の四足歩行ロボットの高速化のために、重心移動の少ない歩行として、牛馬などの歩行に見られるように、対角の2脚で重心を支える方法を提案する。まず、本歩行方法による本ロボットの動特性解析を行い、ループゲインの設定だけで安定化されることを示す。そして、本歩行を実現するために、ロボットの足首に姿勢保持のための制御機構を追加し、これを有効に利用した歩行パターンを開発している。このパターンをソフトウェアサーボシステムのプログラムに組み込み歩行実験を行い、その結果、前報と比較して4倍の歩行速度を実現することができたので報告する。

2. 四足歩行ロボットの動特性

2.1 四足歩行ロボットの構造 本研究では前報四足歩行ロボットに足首部にロール方向の制御を行うための構造変更と制御系を付加し、その結果、対角の

* 昭和62年10月6日 第945回講演会において講演、原稿受付 昭和62年4月9日。

^{*1} 正員、防衛大学校 (〒239 横浜賀市走水1-10-20)。

^{*2} 正員、東京工業大学 (〒152 東京都目黒区大岡町2-12-1)。

2脚で体重を支えることによって生じるモーメントを打ち消すことが可能になった。その本歩行ロボットの構造概要を図1に示す。このロボットは1脚につき4関節を有し、三つの関節は歯付きベルトを介してDCサーボモータによって駆動され、また α 軸はDCサーボモータの軸と直結し駆動される。本ロボットの胴体にとり載されたソフトウェアサーボコントローラは、各関節の軸に直結されたポテンションメータにより角度を検出し、ホストコンピュータから指令された角度になるようにDCサーボモータを制御している。

2.2 四足歩行ロボットのモデル化 前報の静的歩行では接地する足が描く三角形内に重心がくるようにして歩行を可能にしているが、この歩行には重心の移動が伴うために歩行速度には限界がある。

そこで本ロボットをより高速で歩行させるためには、重心の移動を最小限とする歩容を考案しなければならない。そこで、対角の脚同士が交互に体重を支えながら歩行する歩容を仮定し、このとき生じるロール方向の変位は接地する足首の α 軸を制御することによって押さえられるものとする、接地する脚の足首は仮想的な接地面とすることができ、図2に示すモデルが作成できる。ここで、遊脚の足首部は α 軸が一体になっているとする。この場合の形状諸元を表1に示す。ここで、 l_i は軸間距離、 a_i は質量 m_i の関節からの距離を表す。また、対角同士の脚の歩行パターンは同一とし、胴体質量は前後の脚に均等に分けられるものとする、前後の脚は同一の特性をもつと仮定できる。

図2において、相対角 θ と絶対角 q との関係は次のように表される。

$$\theta = Fq \quad \text{.....(1)}$$

図1 四足歩行ロボットの概要

ここに、

$$F = \begin{bmatrix} 1 & & & & \\ -1 & 1 & & & 0 \\ & -1 & 1 & & \\ & & 1 & -1 & \\ 0 & & & 1 & -1 \\ & & & & 1 & -1 \end{bmatrix}$$

$$\theta = [\theta_1 \theta_2 \dots \theta_6]^T$$

$$q = [q_1 q_2 \dots q_6]^T$$

関節 i を駆動するDCサーボモータの特性を

$$n_i T_i + J_i \dot{\omega}_i + \gamma_i \omega_i = \mu_i I_{ai} \quad \text{.....(2)}$$

とする。ここで、 ω_i はDCサーボモータの回転速度、 J_i はモータ電機子の慣性モーメント、 γ_i はモータの内部抵抗、 μ_i はモータのトルク定数、 I_{ai} は駆動電流、 n_i はプーリの減速比、 T_i は駆動トルクを表す。なお、モータの回転速度 ω_i と関節角 θ_i の関係は $\dot{\theta}_i = n_i \omega_i$ となる。

2.3 四足歩行ロボットの運動方程式 図2に示すモデルの θ_i 軸を原点にとり各質量の座標 (x, y) から、運動エネルギー T 、位置エネルギー U 、損失エネルギー D は

$$T = \frac{1}{2} \sum_{i=1}^6 \{I_i \dot{\theta}_i^2 + m_i (\dot{x}_i^2 + \dot{y}_i^2)\} \quad \text{.....(3)}$$

$$U = \sum_{i=1}^6 m_i g y_i \quad \text{.....(4)}$$

$$D = \frac{1}{2} \sum_{i=1}^6 D_i \dot{\theta}_i^2 \quad \text{.....(5)}$$

ここで、 I_i は i リンクの慣性モーメント、 g は重力の

表1 四足歩行ロボットの諸元

i	1, 5, 7, 11	2, 4, 8, 10	3, 9	6, 12
l_i m	0.15	0.15	0.65	—
a_i m	0.22	0.51	0	0.04
m_i kg	0.366	0.433	4.07	0.43
I_i kgm ²	1.77×10^{-4}	1.13×10^{-3}	0.32	6.88×10^{-4}
n_i	0.5	0.41	0.41	—

$$J_i = 0.018 \text{ kgm}^2, \mu_i = 2.86 \text{ Nm/A}, \gamma_i = 1.07 \text{ Nms}$$

図2 2脚支持モデル

加速度, D_i は減衰係数を表す. 式(3)から式(5)をラグランジュの運動方程式⁽¹⁰⁾に適用すると, 運動方程式は次のような非線形常微分方程式で表される.

$$\{J_1(q) + F^T J_0 F\} \ddot{q} + R(q) \dot{q}^2 + F^T (D + B_0) F \dot{q} + G(q) = Eu \dots\dots\dots (6)$$

ここに, $J_1(q) = \{p_{ij} \cos(q_i - q_j)\}$

$$J_0 = \{\text{diag}(J_{ii})\}, J_{ii} = J_i/n_i^2$$

$$R(q) = \{p_{ij} \sin(q_i - q_j)\}$$

$$D = \{\text{diag}(p_{ii})\}, D_{ii} = D_i$$

$$B_0 = \{\text{diag}(\gamma_{ii})\}, \gamma_{ii} = r_i/n_i^2$$

$$G(p) = -(h_1 \sin p_1, \dots, h_6 \sin p_6)^T$$

$$u = (Ia_1, \dots, Ia_6)^T$$

$$E = F^T M$$

$$M = \{\text{diag}(M_{ii})\}, M_{ii} = \mu_i/n_i^2$$

ここで, p, h は付録に詳細が記述されている.

式(6)の線形化⁽¹⁰⁾のために平衡点を $\bar{q}$, $\bar{u}$, 偏差を Δq , Δu とすると, $q = \bar{q} + \Delta q$, $u = \bar{u} + \Delta u$ となり, これを式(6)に代入して一次関数で近似すると

$$J \Delta \ddot{q} + S \Delta \dot{q} + G \Delta q = E \Delta u \dots\dots\dots (7)$$

ここで, $J = J_1(\bar{q}) + F^T J_0 F$

$$S = F^T (D + B_0) F$$

$$G = -(h_1 \cos \bar{q}_1, \dots, h_6 \cos \bar{q}_6)$$

$$J_1(\bar{q}) = \{p_{ij} \cos(\bar{q}_i - \bar{q}_j)\}$$

となる.

これを, $\Delta q = x_1$, $\Delta \dot{q} = x_2$ とおいて, 状態方程式を用いて表すと

$$\dot{x}(t) = Ax(t) + B \Delta u(t) \dots\dots\dots (8)$$

ここで, $x(t) = [x^T x^T]^T$

$$A = \begin{bmatrix} O & I \\ -J^{-1}G & -J^{-1}S \end{bmatrix}$$

$$B = \begin{bmatrix} O \\ J^{-1}E \end{bmatrix}$$

となる. なお, I は 6×6 の単位行列である.

3. 四足歩行ロボットの歩行シミュレーション

3.1 歩容モデル 本研究で採用した2脚支持による歩行モデルを図3に示す. 図3において, ○は遊脚, ●は立脚を表す. 図3(a)と図3(c)のように, 対

図3 準動的歩行の方法

角の脚が接地している間に他の脚が足を踏み出している状態を踏出し区間と定義する. 図3(b)のように踏出し区間が終わりすべての脚で体重を支持している状態を支持区間と定義する. ここで, 対角脚の歩容は同じとする. この歩行を実現する歩容はロボットが平面上を等速直線運動しているとする, 図4のようなステイック線図で表せる. ここで, S_t は歩幅, Δx は時分割した刻み幅, x_d はパターン開始の偏差, x_s は遊脚の区間で s_t の中間にとる. H_p は遊脚時の θ_1 軸の通過高さ, θ_1, θ_3 軸の x 方向の距離を x_p, y 方向の距離を H とする. 図4における各関節の関係は

$$\begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix} = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} A_1 \\ A_2 \\ A_3 \end{bmatrix} + \begin{bmatrix} 3\pi/2 \\ \pi \\ 0 \end{bmatrix} \dots\dots\dots (9)$$

$$\begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ -1 & 0 & 0 \\ 0 & -1 & -1 \end{bmatrix} \begin{bmatrix} A_1 \\ A_2 \\ A_3 \end{bmatrix} + \begin{bmatrix} -\pi \\ 3\pi/2 \\ \pi \end{bmatrix} \dots\dots\dots (10)$$

ここに,

$$A_1 = \cos^{-1}\{(l_1^2 + l_2^2 - H^2 - x_p^2)/2l_1l_2\}$$

$$A_2 = \cos^{-1}\{(H^2 + x_p^2 + l_1^2 - l_2^2)/2l_1\sqrt{H^2 + x^2}\}$$

$$A_3 = \tan^{-1} H/x_p$$

となる.

図3のA, C脚のパターンを図4のようにすると, B, D脚は半歩分位相を遅らした位置から開始するように設定する.

3.2 歩行パターン決定基準 歩行パターンを決定する評価基準には, モータのトルクを最小にする場

図4 歩行パターンの幾何学的関係

合と歩行速度を最大にする場合が考えられる。トルクを最小にするには、静止時のトルクが最小の場合であり式(7)において $q_1=q_2=0$, すなわち歩幅が零で $H=l_1+l_2$ となる。一方、歩行速度を最大にするには歩幅を最大にすることになり、そのとき、

$$\sqrt{H^2+x_p^2} < (l_1+l_2) \quad \cdots \cdots (11)$$

の関係が成り立たなければならない。歩幅を大きくすれば H は小さくなり大きなトルクを必要とする。このように、評価基準によって相反する結果になる。ここでは、歩幅を定めて、式(11)の関係が成り立つ最大の H を採用する。

踏出し区間と支持区間の間隔は s_t と x_s によって決定される。 $(s_t/2-x_s)=0$ のとき支持区間はなく、瞬間的に支持脚が入れ替わることになり、 $(s_t/2-x_s)>0$ のときは支持区間があり前者より支持脚の交換が滑らかに行われる。ここでは後者を採用し遊脚区間を $6\Delta x$ に設定している。

パターン開始の偏差 x_d によって重心位置を前後に

図5 固有値による安定判別

表2 歩行パターンの諸元

Δx	0.01 m
s_t	0.16 m
x_s	0.06 m
H_p	0.02 m
$H_{\max}$	0.29 m
x_d	-0.01 m

図6 ステップ応答の解析結果

変化させることができる。これを負にとれば重心が前方に移動し、正にとれば後方に移ることになる。歩行時には前方に、減速時には後方に重心がくるようにすれば歩行がなめらかになると考えられる。ここでは、 $x_d=-0.01$ m にとり、前者を採用している。

3.3 歩行シミュレーション ここで採用した歩行パターンを本ロボットに与えた場合に、安定な歩行が可能であるかを、すでに求めた状態方程式(8)を基にして解析してみる。まず、ソフトウェアサーボ系で用いるために式(8)を離散化すると

$$x(k+1) = Px(k) + Qu(k) \quad \cdots \cdots (12)$$

ここに、

$$P = e^{A\Delta t}, \quad Q = \int_0^{\Delta t} e^{A\tau} d\tau B$$

ここで、 Δt はサンプリング間隔を表す。式(12)において、 $u(k) = -Kx(k)$ としてフィードバックを構成すると

$$x(k+1) = (P - QK)x(k) = P_c x(k) \quad \cdots \cdots (13)$$

と表せる。ここで、 K はゲイン行列でループゲインだけの場合には

$$K = \begin{bmatrix} k_1 & 0 & \cdots & 0 \\ & \ddots & & \\ 0 & k_6 & & \end{bmatrix} \begin{bmatrix} F & 0 \\ 0 & 0 \end{bmatrix} \quad \cdots \cdots (14)$$

になる。図5には $k_i (i=1\sim6)$ が0と安定で大きなゲイン600の場合の P_c の固有値を示す。このように、ループゲインの設定だけでも固有値は単位円内にあり安定化されている。図6には $k_i=600$ にとり開始点 ϕ_s から $2\Delta x$ 間のステップ入力による応答を示している。ここで、歩行パターンのパラメータを表2にし、サンプリング間隔を $880 \mu s$ とした。図6から単なる位置フィードバックの設定だけでも、歩行の制御には充分であることがわかる。

図7 四足歩行ロボットの制御方式概要

4. 四足歩行ロボット歩行実験

4.1 四足歩行ロボットの制御方式 本ロボットに用いた制御方式の概要を図7に示す。これは、前報のソフトウェアサーボコントローラに α 軸を制御する回路を追加して、4関節を1枚のボードで制御する方式を採用している。各コントローラはホストコンピュータに接続されている。ホストコンピュータでは、図4の歩行パターンからすべての関節の目標値を計算しコントローラに出力し、またコントローラ側から現在の関節角を入力しロボットの状態を把握することができる。このような制御システムは分散制御方式⁽⁹⁾と呼ばれ、制御関節が多くなり複雑化するほど本システムの構成が有効になる。

本ロボットの制御プログラムはシステムの特徴を生かして、ホスト側には複雑な数値計算ができるPASCALが、コントローラ側には高速処理ができるアセンブリ言語が用いられている。

4.2 ソフトウェアサーボコントローラ 本ロボットのコントローラの構成を図8に示す。これは、CPU、EP-ROM、RAM、A-D変換器、マルチプレクサ、入出力ポート、電流変換形D-A変換方式DCモータドライバから構成されている。EP-ROMには4関節をラウンドロビン方式により制御するプログラムが書き込まれている。このプログラムではホスト側から

図8 ソフトウェアサーボコントローラの構成

図9 ソフトウェアサーボのブロック図

目標値を入力し、A-D変換器から入力した角度から制御量を計算してドライバーに出力している、そのブロック図を図9に示す。ここで、 K_p はループゲインで、この値は起動直後にホスト側から入力される。また、 K_v は速度フィードバックゲインで零とする。 G_f は外力の前置補償でホスト側で目標値に加えられている。

4.3 実験装置 図10に本歩行ロボットの外観を示す。胴体にはソフトウェアサーボコントローラ、DCモータドライバおよびホストコンピュータとのインタフェースボードがとう載されていて、総重量は14.5 kgである。ロボットには電源とホストとの通信ケーブルが接続されている。ロボットのロール軸を除く関節には写真によって軌跡を調べるために発光ダイオードが取り付けられている。また、足の平の四隅には接地力によって開閉するセンサによって足の接地状況が検出できるようになっている。

4.4 歩行実験 図11には図6のシミュレーション

図10 本ロボットの外観

図11 ステップ応答の実験結果

図 12 準動的歩行の各関節の目標値

図 13 歩行時の各関節の応答

図 14 歩行時の各関節の軌跡

ンと同じ条件で開始位置 ϕ_s から 2 ステップ先の目標値を与えた場合のステップ応答の実験結果を示す。このとき、コントローラのループゲイン K_p は安定に動作する最大の 5 に設定されている。また、 α 軸は足の平と脚が常に 90° になるよう制御されている。このように、実験と解析結果はほぼ一致しており、解析結果の妥当性が示された。この結果から、ホストコンピュータがコントローラ側に目標値を出力する間隔は 0.2 s 程度に設定する。

図 12 には静止状態から 2 ステップさせる場合にホスト側からコントローラ側に与えられる各関節の目標値を示す。図 13 には歩行時の応答を示す。起動時には応答の遅れが見られるが、1 ステップけば定常的な応答を示

すようになる。このとき 1 ステップに要する時間は 3.6 s で歩幅は 0.16 m、すなわち歩行速度は 0.044 m/s である。これは、静的歩行の 4 倍の速度であり、高速化が実現された。

図 14 には歩行時の関節の動きを開放写真によって示す。ここでは、シャッター開放後ストロボを 1 回発光させている。図 14 から各関節の軌跡は理論的に与えられた歩行パターンとは胴体の弾性変形のために多少違っているが、胴体はほぼ水平を保ちながら歩行していることがわかる。

5. 結 論

四足歩行ロボットの高速歩行のための歩行方法を提案し、理論解析と実験を行い次の知見を得た。

(1) 四足歩行ロボットの足首にロール方向のトルクを打ち消す制御を与えれば、対角の 2 脚支持による歩行が可能である。

(2) 姿勢保持のための重心移動がない歩行パターンにすることによって高速歩行が可能で、静的歩行より約 4 倍の速度を実現した。

(3) ロボットの制御システムとして分散制御方式のソフトウェアサーボは多関節の制御に有効であり、拡張が容易に行える。

なお、ロボットを実用的にするためには方向の変更、不整地の歩行、自立歩行を実現しなければならない。

そのためには知識制御システムの開発が今後の課題である。

最後に、本ロボットの開発に多大な協力いただいた防衛大学校 山之内技官に謝意を表す。

付 録

式(6)における p, h の詳細を下記に示す。

$$\begin{aligned} p_{11} &= \{I_1 + m_1 a_1^2 \\ &\quad + (m_2 + m_3 + m_4 + m_5 + m_6) l_1^2\} \\ p_{22} &= \{I_2 + m_2 a_2^2 \\ &\quad + (m_3 + m_4 + m_5 + m_6) l_2^2\} \\ p_{33} &= \{I_3 + m_3 a_3^2\} \\ p_{44} &= \{I_4 + m_4 (l_4 - a_4)^2 + (m_5 + m_6) l_4^2\} \\ p_{55} &= \{I_5 + m_5 (l_5 - a_5)^2 + m_6 l_5^2\} \\ p_{66} &= \{I_6 + m_6 a_6^2\} \\ p_{12} &= m_2 a_2^2 \\ &\quad + (m_3 + m_4 + m_5 + m_6) l_1 l_2 \\ p_{13} &= m_3 l_1 a_3 \\ p_{14} &= -\{m_4 l_1 (l_4 - a_4) + (m_5 + m_6) l_1 l_4\} \\ p_{15} &= -\{m_5 l_1 (l_5 - a_5) + m_6 l_1 l_5\} \\ p_{16} &= -m_6 l_1 a_6 \\ p_{23} &= m_3 l_2 a_3 \\ p_{24} &= -\{m_4 l_2 (l_4 - a_4) \\ &\quad + (m_5 + m_6) l_2 l_4\} \\ p_{25} &= -\{m_5 l_2 (l_5 - a_5) + m_6 l_2 l_5\} \end{aligned}$$

$$\begin{aligned} p_{26} &= -m_6 l_2 a_6 \\ p_{45} &= \{m_5 l_4 (l_5 - a_5) + m_6 l_4 l_5\} \\ p_{46} &= m_6 l_4 a_6 \\ p_{56} &= m_6 l_5 a_6 \\ p_{ik} &= q_{ki} (i=1 \cdots 6, k=1 \cdots 6) \\ h_1 &= -g \{m_1 a_1 \\ &\quad + (m_2 + m_3 + m_4 + m_5 + m_6) l_1\} \\ h_2 &= -g \{m_2 a_2 + (m_3 + m_4 + m_5 + m_6) l_2\} \\ h_3 &= -g m_3 a_3 \\ h_4 &= g \{m_4 (l_4 - a_4) + (m_5 + m_6) l_4\} \\ h_5 &= g \{m_5 (l_5 - a_5) + m_6 l_5\} \\ h_6 &= g m_6 a_6 \end{aligned}$$

文 献

- (1) 中野, ロボット工学入門, (昭58), 107, オーム社.
- (2) Raibert, M. H. and Brown, H. B., Jr., *Trans. ASME, J. Dyn. Syst. Meas. Control*, **106** (1979), 75.
- (3) Furusho, J. and Masubuchi, M., *Trans. ASME, J. Dyn. Syst. Meas. Control*, **108** (1986), 111.
- (4) 美多, コンピュータローラ, **2** (昭58), 76, オーム社.
- (5) Seireg, A. and Deterson, R. M., *Proc. 4th World Cong. Theory Mach. Mech.*, (1979), 359.
- (6) 広瀬・梅谷, 計測自動制御学会論文集, **16-5** (昭55), 133.
- (7) McGhee, R. B. and Iswandhi, G. I., *IEEE, SMC-9-4* (1979), 176.
- (8) 滝田・背戸, 機論, **52-484, C** (昭61), 3249.
- (9) 背戸・滝田・富成, 機論, **50-457, C** (昭59), 1751.
- (10) 古田・ほか3名, メカニカルシステム制御, (昭59), 145, オーム社.

討 論

〔質問〕 広瀬 茂 男 (東京工業大学工学部)

(1) 「安定な歩行が可能であることを状態方程式(8)を基に解析した」との記述があるが、この解析は、脚を平衡点近傍で微小に振った場合のサーボ系の安定性を解析したのであり、重心の安定支持という歩行本来の安定性解析とは別ものではないか。

(2) 「対角の2脚で重心を支える方法を提案する」との記述があるが、これはいわゆるトロット歩容で機械モデルでも実現されているものではないか(付1)。

(3) 「足首の α 駆動軸を追加し、それを有効に利用した」との記述があるが、トロット歩容においては胴体の傾斜角の軸と α 軸は一致せず、姿勢制御には利用できないと思われる。実験でも 90° に固定しているとなっているが、 α 軸はどのように利用されたのか。

(4) 足裏の幅に対する歩幅 S_i から考えると重心が対角2脚の足裏多角形の内部にある静的歩行とも思える。

〔回答〕 (1) 二脚支持による、四足歩行ロボットの動特性解析のために、前後の2脚ずつに分けて、各々同じ運動をするものと仮定した。そのため、状態方程式(8)は6関節にサーボ系を含んだ一脚支持モデルを表している。したがって、制御系に歩行パターンが与えられた場合に、系が安定であれば、同時に運動の安定性が補償されていることになり、安定した歩行が可能であると考えられる。しかし、歩行の高速化を図るためには、常に制御系を安定にしておく必要はなく、安定、不安定を切替える方法も考えられる。

(2) この歩行はご指摘のようにトロット歩容である。ここで提案した歩行方法は、静止状態でも2脚で支持することができ、本歩行パターンの再生のみで歩行することができる。したがって、機械モデルで行っているような静動遷移歩行は考える必要はない。

(付1) 広瀬, ほか2名, 26回SICE学術講演会予稿集, (昭62), 509-510.

(3) 本歩行パターンは、胴体の高さを一定にし、等速直線運動しているとして定めている。 α 軸が自由に回転する場合には、遊脚と胴体の移動のために、支持脚を支点にする不釣り合いが生じて倒れ込み歩行が不可能になり、また、 α 軸の回転を固定すると足の平の大きさの調整によって不釣り合いは抑えられるが、水平面の移動に限られてしまう。そこで、表面を選ばず不釣り合いを補償する方法として、 α 軸を一定角度に制御する方法を提案した。本研究において α 軸の目標値は、水平面の移動を前提としたために脚と接地面が常に直角になるように設定されている。制御系は、不釣り合いによって生じた外乱トルクと α 軸の回転角に比例した制御トルクによって釣り合い、倒れ込みが押えられる。このとき、定常偏差が残るがループゲインを大きくすることによって、これを小さくすることができる。

(4) ここに提案した歩行方式は、静的安定を維持できることからすれば静的歩行であるが、三脚支持による静的歩行と全く異なっている。本歩行方法は、今後の研究によって動的歩行を実現しうる歩行方式と考えられることから、準動的歩行と名付けている。

〔質問〕 毛利 彰〔九州大学工学部〕

α 軸の制御は具体的にはどのようにしているのか。

〔回答〕 広瀬茂男氏への回答の(3)を参照願いたい。

〔質問〕 松原 武徳

〔三菱重工業(株)高砂研究所〕

(1) ペイロードについて、準動的歩行は他方式歩行に比べて有利か。

(2) 移動速度についてはどうか。その下方限界は

どこか。

(3) 不整地歩行の場合でも、準動的歩行は実現可能か。

(4) 分散制御方式のソフトウェアサーボは多関節の制御に有効であると考えているようだが、具体的に何軸程度からか。

〔回答〕 (1) 本歩行ロボットでは、二脚支持と三脚支持によるペイロードの違いはほとんどないと考えられる。それは、三脚支持において、均等に重量が配分されるわけではなく、対角の2脚で主な重量を支持し、残りの支持脚では遊脚との重量の不釣り合い分のわずかな重量が作用するためである。

(2) この歩行方式は、二脚支持による静止状態から歩行を開始することが可能になっている。

(3) 不整地での歩行は今後の課題になっているが、この歩行方法で可能と考えられる。この不整地歩行を実現するためには、ロボットに姿勢角を測定するセンサを取付け、これを利用して、胴体を一定の姿勢を維持するように各関節の目標値を変化させる制御機能を付加することが必要になる。本研究で用いた分散制御のソフトウェアサーボシステムは、このような機能の拡張が容易なシステムと言える。

(4) 本研究でのソフトウェアサーボのサンプリング間隔は4関節の制御で880 μ sであり、歩行ロボットの制御には十分な制御間隔と考えられる。一般的には、1台のCPUで一自由度を制御した場合のサンプリング周波数を制御対象の構造周波数の10倍(制御系に必要とされる最低のサンプリング周波数)で割ることによって、1台のCPUで制御可能な自由度が決定される。