干渉ヤコビ行列を用いた ロボットアームの実時間障害物回避

兼 子 一* 新 井 健 生* 井 上 健 司* 前 泰 志*

Real-time Obstacle Avoidance with Collision Jacobian for Robot Arm

Hajime Kaneko*, Tatsuo Arai*, Kenji Inoue* and Yasushi Mae*

This paper proposes a realtime trajectory planning and control method of a robot arm for obstacle avoidance. If the distance between a link and a object are small, suggesting that a collision is about to occur, then an alternative trajectory is calculated to prevent the collision while minimizing the error in the desired (commanded) trajectory. The trajectory control algorithm that results in the form of a constrained minimization problem. Key to the trajectory modification algorithm is derivation of the Jacobian matrix relating changes in endeffector velocity to interference verocity. Simulation results demonstrate the performance of this algorithm and experimental results verify it's effectiveness.

Key Words: Obstacle Avoidance, Trajectory Planning, Parallel Mechanism

1. は じ め に

ロボットアームは周囲の障害物との接触およびリンク同士の干渉を避けつつ動作しなければならない。ロボットの障害物回避法は、大別してオフライン軌道計画と実時間での障害物回避に分類できる。

オフライン軌道計画は、周囲の障害物すべてについて位置・形状・大きさが既知であるとの仮定の下でロボットが障害物に衝突せずに初期位置より目標位置へ到達する軌道を事前に求めるものである。軌道計画の手順は大きく二つのステップに分けられる。まずは環境を表現する手法の問題、次にロボットが干渉を起こさない最適軌道を導出する手法の問題である。

環境を表現するために、リンクや障害物は様々な形状で近似される。重松らはシンプレックス法を用いて多面体の積集合部分を抽出し干渉を検出する方法を示した [1]。水垣らは円柱で物体を近似する手法を用いた [2]。小沢らは様々な形状で近似された物体同士の干渉チェックをそれらの幾何学的情報を元に高速で行う手法を示した [3]。他の環境表現手法としては、オクトツリーを用いて空間を再帰的に分割する方法がある [4]。また、ロボットアームの軌道設計には状態コンフィギュレーション空間における表現も有用である [5]。

Udupa は多面体によって表現された障害物を円筒で表現されたリンクの半径分だけ拡大することによって、リンクを線分として扱い軌道を求めた [6]。しかし、この手法を多くのリン

クを持つアームに対して適用することは困難である。Loef らはリンクに障害物から離れる動きを与えるポテンシャルを導入した [7]。このポテンシャル法を目的地の記述に拡張すると、ロボットの軌道は運動学的解法によって得ることができる [8]。状態コンフィギュレーション空間による表現はこの手法に適しているが、障害物を状態空間上にいかに表現するかについては課題が残されている。尾崎らは多面体で表されたアームの占有空間を考慮しつつ、オペレータより与えられた大局的な軌道を元に軌道設計を行う手法を提案した [9]。

これらの手法を実時間の回避制御に応用するためにはいくつかの問題がある。周囲の環境が未知あるいは動的である場合、ロボットは障害物との接触の危険性が生じたときには即座に回避動作を決定しなければならない。また、オペレータからの指令値が前もって与えられていない場合についても、実時間での制御が必要である。ロボット・障害物間の距離の導出法は実時間制御の動作決定において重要な要素である。

移動ロボットの実時間障害物回避については様々な研究がすでにためされている [10] [11]。ロボットアームに関する実時間障害物回避の一例として、Petrov の研究が挙げられる [12]。彼は、リンクが接している障害物のみの限られた情報の下で手探りでの迷路抜け問題に解を与えた。

本研究では、障害物・リンク間の距離を取り扱う手法として干渉ヤコビ行列を考案し、これを用いたロボットアームの実時間障害物回避法を提案する。干渉ヤコビ行列は、リンクの障害物への接近速度と、エンドエフェクタ速度の関係を示すものであり、その詳細は第 2 章で述べる。提案する障害物回避アルゴリズムは第 3 章にて述べる。この方法では、いずれかのリンク

原稿受付 1999 年 2 月 22 日

*大阪大学大学院基礎工学研究科

*Graduate School of Engineering Science, Osaka University

が障害物に接近している場合、接触を回避しつつ与えられた目標速度との誤差を最小化するようにエンドエフェクタ速度を修正する。修正された速度を求める計算には繰り返しを含まないので、提案する手法は実時間制御に適している。第4章では、この手法を平行アームにおけるリンク同士の干渉回避に適用した実験例を示す。

2. 干渉ヤコビ行列

2.1 干渉ヤコビ行ベクトル

本研究では、冗長でないロボットアームを取り扱い、その自由度を n とする。各関節変数を q_i ($i = 1, \dots, n$) とし、それらを並べた関節変数ベクトルを $\mathbf{q} = [q_1, \dots, q_n]^T$ と表す。また、エンドエフェクタの並進速度 $\mathbf{v}_p$ 、角速度 ω を並べたベクトルを $\mathbf{v} = [\mathbf{v}_p^T, \omega^T]^T$ で表し、単に“エンドエフェクタ速度”と呼ぶことにする。

はじめに、リンク・障害物の組のうち一つについて考える。通常用いるヤコビ行列 J は、エンドエフェクタ速度 $\mathbf{v}$ と関節速度 $\dot{\mathbf{q}}$ の関係を次式のように与える。

$$\mathbf{v} = J\dot{\mathbf{q}} \quad (1)$$

ただし、ヤコビ行列 J は正則とする。

J と同様に、エンドエフェクタ速度と、あるリンクとある障害物の間の接近速度との関係を与える行ベクトルを定義する。リンク・障害物間の距離を d とすると、この組の接近速度 v_a とエンドエフェクタ速度 $\mathbf{v}$ の関係は行ベクトル $\mathbf{j}$ を用いて次式で与えられる。

$$v_a = -\dot{d} = \mathbf{j}\mathbf{v} \quad (2)$$

このリンク・障害物の組に対応した行ベクトル $\mathbf{j}$ を“干渉ヤコビ行ベクトル”と呼ぶことにする (Fig. 1)。

一例として、Fig. 2 に示すアームのリンク2と障害物に関する干渉ヤコビ行ベクトルの導出法を述べる。このアームは3個の回転関節によって構成された平面アームであり、関節変数ベクトルを $\mathbf{q} = [q_1, q_2, q_3]^T$ 、エンドエフェクタ速度を $\mathbf{v} = [\dot{x}, \dot{y}, \dot{\theta}]^T$ と表す。ただし、 (x, y) はエンドエフェクタの位置、 θ は姿勢である。Fig. 2 の状態において、リンク2と障害物間の距離は次式のように得られる。

$$d = |\mathbf{p}_d| = |\mathbf{p}_l - \mathbf{p}_o| \quad (3)$$

ただし、 $\mathbf{p}_l, \mathbf{p}_o$ はそれぞれリンク2、障害物上の最短距離を与える点を示す位置ベクトルであり、アームの姿勢 $\mathbf{q}$ に依存する。なお、この場合で影響を与えるのは (q_1, q_2) のみである。式 (1) を $\dot{\mathbf{q}} = J^{-1}\mathbf{v}$ と変形して利用すると、この組の接近速度 v_a とエンドエフェクタ速度 $\mathbf{v}$ との関係は以下のように求められる。

$$\begin{aligned} v_a = -\dot{d} &= -\frac{1}{d}\mathbf{p}_d^T \dot{\mathbf{p}}_d \\ &= -\frac{\mathbf{p}_d^T}{d} \begin{bmatrix} \frac{\partial \mathbf{p}_d}{\partial q_1} & \frac{\partial \mathbf{p}_d}{\partial q_2} & \frac{\partial \mathbf{p}_d}{\partial q_3} \end{bmatrix} \dot{\mathbf{q}} \\ &= -\frac{\mathbf{p}_d^T}{d} \begin{bmatrix} \frac{\partial \mathbf{p}_d}{\partial q_1} & \frac{\partial \mathbf{p}_d}{\partial q_2} & 0 \end{bmatrix} J^{-1}\mathbf{v} \end{aligned} \quad (4)$$

Fig. 1 Collision Jacobian column vector

Fig. 2 Example for collision Jacobian column vector

Fig. 3 Collision Jacobian

ゆえに、この組に関する干渉ヤコビ行ベクトル $\mathbf{j}$ は次式のように得られる。

$$\mathbf{j} = -\frac{\mathbf{p}_d^T}{d} \begin{bmatrix} \frac{\partial \mathbf{p}_d}{\partial q_1} & \frac{\partial \mathbf{p}_d}{\partial q_2} & 0 \end{bmatrix} J^{-1} \quad (5)$$

上記の例より明らかなように、リンク・障害物の組に関する干渉ヤコビ行ベクトルを求めるためには、障害物への距離が最小となる点 ($\mathbf{p}_l, \mathbf{p}_o$) を知ることが必要である。

2.2 干渉ヤコビ行列

Fig. 3 のように m 組のリンク・障害物の組について接触の危険性があり、接近速度を修正する場合を考える。 $v_{a,k}, \mathbf{j}_k$ ($k = 1, \dots, m$) はそれぞれ k 番目の組に関しての接近

速度と干渉ヤコビ行ベクトルを示す。接近速度ベクトルを $\mathbf{v}_a = [v_{a,1}, \dots, v_{a,m}]^T$ と定義すると、式 (2) を各組について適用したものをまとめて、

$$\mathbf{v}_a = J_c \mathbf{v} \quad (6)$$

と表せる。ただし、

$$J_c \equiv \begin{bmatrix} j_1 \\ j_2 \\ \vdots \\ j_m \end{bmatrix} \quad (7)$$

干渉ヤコビ行ベクトルによって構成された $m \times n$ 行列 J_c は m 組の干渉の危険性のあるリンク・障害物の組すべてについての接近速度ベクトル $\mathbf{v}_a$ とエンドエフェクタ速度 $\mathbf{v}$ の関係を示すものである。この J_c を“干渉ヤコビ行列”と名付ける。

3. 干渉ヤコビ行列による障害物回避

3.1 アルゴリズムの概要

第2章で述べたようにアームに冗長性はなく、エンドエフェクタ速度の目標値はオペレータから与えられると仮定する。

アルゴリズムの概要は以下の通りである。

- (1) 所定の安全距離よりも接近しているリンク・障害物の組をすべてのリンク・障害物の組み合わせの中から選び出す (Fig. 4 参照)。
- (2) それらの組の中から、エンドエフェクタが目標値に従って動いた場合に安全接近速度より大きな接近速度を生じる組を選び出す (Fig. 4 参照)。
- (3) 選び出された、距離と接近速度がともに危険な状態にあるリンク・障害物の組について接近速度を安全接近速度に抑えつつ、目標値との誤差が最小になる修正されたエンドエフェクタ速度を求める。

3.2 物体間距離の算出

第2.1節で述べたように、リンク・障害物の組に対応する干渉ヤコビ行ベクトルを計算するためには最短距離を与えるリン

ク・障害物上の2点を知らなければならない。障害物の大きさ、形状、位置が既知であれば、それらの点は障害物のモデルから得ることができる。また、リンクに装備された接近センサなどを用いて検出しても良い。センサを利用した方法は、環境が動的あるいは未知である場合に有効である。本稿で提案する障害物回避法は近接2点の導出法に依存しないため、これらの点はすでに得られていると仮定する。

リンク・障害物間の距離 d があらかじめ設定した安全な距離 $d_{\min}$ 以下である組は、干渉の危険があるとみなす。第1段階では、干渉の危険がある組をすべて選び出す。

3.3 安全接近速度

駆動モータの出力には限界があり、急な加速や減速は動作精度を落とすだけでなくモータや関節に損傷を与える場合もある。ここで、各リンクに関して最大加速度 a を設けると、リンクの障害物に対する接近速度の上限 $v_{\max}$ は次式で与えられる。

$$\begin{cases} v_{\max} - at = 0 \\ d = \int_0^t (v_{\max} - a\tau) d\tau \end{cases} \quad (8)$$

リンク・障害物間の距離が d のとき、その組の接近速度 v_a が $v_{\max}$ 以下であれば、衝突やモータなどへの過負荷を起こすことなく停止することが可能である。この $v_{\max}$ を“安全接近速度”と名付ける。

エンドエフェクタ速度が目標値 $\mathbf{v}_d$ であると仮定しリンク・障害物の組について接近速度 v_a を求めたとき、接近速度が上記の安全接近速度 $v_{\max}$ 以上である場合その組において干渉の発生が予想される。ゆえに第2段階では、第1段階で選んだ組の中からさらに接近速度が安全接近速度を超える組を選び出す。

3.4 エンドエフェクタ速度の修正

第2段階で選び出したリンクと障害物の組が m 組あるとし、添字 k ($k = 1, \dots, m$) を用いてそれらを区別する。ここで j_k , $v_{a,k}$ および $v_{\max,k}$ は、それぞれ第 k 組の干渉ヤコビ行ベクトル、接近速度、安全接近速度である。干渉ヤコビ行列 J_c は式 (7) に従って得られる。

すべてのリンク・障害物の組に関して干渉を回避するためには、それぞれの接近速度 $v_{a,k}$ を安全接近速度 $v_{\max,k}$ に抑えなければならない。つまり、

$$v_{a,k} = v_{\max,k} \quad (k = 1, \dots, m) \quad (9)$$

ここで、修正後のエンドエフェクタ速度 $\hat{\mathbf{v}}$ は式 (6), (9) より導かれる条件を満たせばよい。

$$\mathbf{v}_{\max} = J_c \hat{\mathbf{v}} \quad (10)$$

ただし、 $\mathbf{v}_{\max} = [v_{\max,1}, \dots, v_{\max,m}]^T$ は安全接近速度を並べたベクトルである。この条件を満たしつつ、目標エンドエフェクタ速度 $\mathbf{v}_d$ との誤差が最小となる速度 $\hat{\mathbf{v}}$ を求める。最適化条件は次式の通り。

$$\min(\mathbf{v}_d - \hat{\mathbf{v}})^T W (\mathbf{v}_d - \hat{\mathbf{v}}) \quad (11)$$

ただし、 W はエンドエフェクタ速度に関する重みづけを表す正定行列である。

Fig. 4 Flow chart for judging whether a link-obstacle pair will collide

式(10)(11)で表される最適化問題をラグランジュの未定乗数法を使って解くと、求める修正指令値 $\hat{\mathbf{v}}$ は次式によって得られる。

$$\hat{\mathbf{v}} = [I_n - W^{-1}J_c^T(J_c W^{-1}J_c^T)^{-1}(I_m - W_c)J_c]\mathbf{v}_d \quad (12)$$

ただし、 $W_c = \text{diag} \left[\frac{v_{\max,1}}{J_1 \mathbf{v}_d}, \dots, \frac{v_{\max,m}}{J_m \mathbf{v}_d} \right]$ であり、 I_n, I_m はそれぞれ $n \times n, m \times m$ の単位行列である。 $\hat{\mathbf{v}}$ を求める計算に繰り返しを含まないため、この手法は実時間制御に適している。

(a) Overview of the arm

(b) structure of the links

(c) Arrangement of the links

Fig. 5 Parallel arm used in experiment

4. パラレルアームのリンク干渉防止への適用例

4.1 実験に用いたパラレルアーム

パラレルアームは、シリアルアームと比べ一般的に高速、高精度、高剛性であるが、構造上リンク同士の干渉が起きやすい欠点がある。ここでリンク自身を障害物とみなせば、本稿で提案する障害物回避法をリンク干渉の防止に適用することができる。

Fig. 5 に本実験で用いたアームを示す。このアームはスチュワートプラットホームの一種で、エンドエフェクタとベースは 6 本のリンクで繋がれている。Fig. 5 (b) に示すように、各リンクはベースよりユニバーサルジョイント、直動関節、回転関節、ユニバーサルジョイントの順でエンドエフェクタに繋がれており、このうち直動関節以外は解放されている。各リンクの直動関節を駆動することによってリンク長を変化させ、エンドエフェクタの位置・姿勢を 6 自由度で制御することができる構造である。動作の等方性を考慮してリンク配置を決定したため、このアームのリンクは Fig. 5 (c) に示す配置となっている。非常にリンク同士の干渉が起りやすい配置のため、このアームを制御する際には常にリンク干渉の防止を講じなければならない。

この実験では、各リンクは円筒で近似した。各リンクの組について、距離と干渉ヤコビ行列は以下のように求めることができる。Fig. 6 において、 Σ はベースに固定された基準座標系、 Σ_h はエンドエフェクタに固定された手先の座標系である。 B_i, C_i はそれぞれ、リンク i がベース、エンドエフェクタに連結されている地点を示す。まず、 Σ から見た Σ_h の位置ベクトル $\mathbf{p}$ と回転行列 R が与えられたとき、リンク i の長さ l_i と、向きを示す単位ベクトル $\mathbf{z}_i$ は次のように得られる。

$$\mathbf{w}_i = \mathbf{p} + R\mathbf{c}_i^h - \mathbf{b}_i \quad (13)$$

$$l_i = |\mathbf{w}_i| \quad (14)$$

$$\mathbf{z}_i = \mathbf{w}_i / l_i \quad (15)$$

ただし、 $\mathbf{c}_i^h$ は Σ_h 上における C_i の位置ベクトル、 $\mathbf{b}_i$ は Σ 上における B_i の位置ベクトルである。ここからは、リンク i と j の組について考える。 M_i はその組の共通垂線とリンク i との交点であり、 M_j はリンク j との交点である。それらの位置ベクトル $\mathbf{m}_i, \mathbf{m}_j$ は次式によって得られる。

$$\mathbf{m}_i = \mathbf{b}_i + s_{ij}\mathbf{z}_i \quad (16)$$

Fig. 6 Variables of experimental parallel arm

$$\mathbf{m}_j = \mathbf{b}_j + s_{ji} \mathbf{z}_j \quad (17)$$

$$\begin{aligned} s_{ij} &= |\mathbf{m}_i - \mathbf{b}_i| \\ &= \frac{(\mathbf{b}_i - \mathbf{b}_j) \cdot \mathbf{z}_j \cos \alpha_{ij} + (\mathbf{b}_j - \mathbf{b}_i) \cdot \mathbf{z}_i}{\cos^2 \alpha_{ij} - 1} \end{aligned} \quad (18)$$

$$\begin{aligned} s_{ji} &= |\mathbf{m}_j - \mathbf{b}_j| \\ &= \frac{(\mathbf{b}_j - \mathbf{b}_i) \cdot \mathbf{z}_i \cos \alpha_{ij} + (\mathbf{b}_i - \mathbf{b}_j) \cdot \mathbf{z}_j}{\cos^2 \alpha_{ij} - 1} \end{aligned} \quad (19)$$

ただし, $\cos^2 \alpha_{ij} = \mathbf{z}_i^T \mathbf{z}_j$. ゆえに, リンク i, j 間の距離 d_{ij} は $\mathbf{m}_{ij} = \mathbf{m}_j - \mathbf{m}_i$ とおくと,

$$d_{ij} = |\mathbf{m}_{ij}| \quad (20)$$

となる. 次に, エンドエフェクタ速度 $\mathbf{v} = [\mathbf{v}_p^T, \boldsymbol{\omega}^T]^T$ が与えられたときの Σ 上における C_i の速度は,

$$\dot{\mathbf{c}}_i = \mathbf{v}_p + \boldsymbol{\omega} \times R \mathbf{c}_i^h \quad (21)$$

であり, Fig. 6 に示すように Σ 上における M_i の速度は,

$$\dot{\mathbf{m}}_i = \frac{s_{ij}}{l_i} \dot{\mathbf{c}}_i \quad (22)$$

となる. ただし, $\mathbf{c}_i$ は Σ 上における C_i の位置ベクトルである. これより, リンク i, j 間の接近速度 $v_{a,ij}$ は,

$$\begin{aligned} v_{a,ij} &= -\dot{d}_{ij} = \frac{\mathbf{m}_{ij}}{d_{ij}} \cdot \dot{\mathbf{m}}_{ij} \\ &= \frac{1}{d_{ij}} \left(\left(\frac{s_{ij}}{l_i} - \frac{s_{ji}}{l_j} \right) \mathbf{m}_{ij} \cdot \mathbf{v}_p \right. \\ &\quad \left. + \left(\left(\frac{s_{ij}}{l_i} \mathbf{c}_i - \frac{s_{ji}}{l_j} \mathbf{c}_j \right) \times \mathbf{m}_{ij} \right) \cdot \boldsymbol{\omega} \right) \end{aligned} \quad (23)$$

ゆえに, この組に対応する干渉コピ行ベクトル $\mathbf{j}_{ij}$ は次式のように表せる.

$$\mathbf{j}_{ij} = \frac{1}{d_{ij}} \left[\begin{array}{c} \left(\frac{k_{ij}}{l_i} - \frac{k_{ji}}{l_j} \right) \mathbf{m}_{ij} \\ \left(\frac{k_{ij}}{l_i} \mathbf{c}_i - \frac{k_{ji}}{l_j} \mathbf{c}_j \right) \times \mathbf{m}_{ij} \end{array} \right]^T \quad (24)$$

ここでは共通垂線が有効リンク長の範囲内にある場合を示した. 共通垂線の足が有効リンク長の外にある場合には, 最短距離を与える点はリンクの上下端か, 上下端から相手のリンクに対しておろされる垂線の足のうちのいずれかとなるが, ここではその決定法の詳細は省略する.

4.2 シミュレーション結果

このパラレルアームを基本姿勢から z 軸回りに約 13 [deg] 回転させると, リンク 1 とリンク 4, リンク 2 とリンク 5, リンク 3 とリンク 6 がそれぞれ干渉を起こす. ただし, どれだけの回転を z 軸回りに与えるとリンク間干渉が発生するかはエンドエフェクタの位置や姿勢に依存するため, 単純に z 軸回りの回転角のみを監視して 13 [deg] 回転した時点で z 軸回りの回転成分を抑えるだけでは正確にリンク干渉を防止することはできない. そこで提案する手法に基づき, 上述の限界を超えて z 軸回りに回転しようとする目標エンドエフェクタ速度を与えた場合

Fig. 7 Simulation result: distance between two links

についてこれら 3 組のリンク干渉回避シミュレーションを行った. この場合, 各組のリンク間距離の変化はアームが対称な構造であるためにほぼ等しい.

シミュレーション結果を Fig. 7 に示す. (a) は z 軸回りのエンドエフェクタ軌道, (b) はリンク間距離の変化であり, エンドエフェクタ速度の目標値に修正を行わず動かした場合を破線で, 修正後の変化を実線で示した. このように, 提案する手法によってパラレルアームのリンク間干渉防止を行えることを確認した.

4.3 実験結果

この実験では, オペレータの人間がジョイスティックを用いて z 軸回りに前述の限界を超える回転を行う指令値を含むエンドエフェクタ速度を与えた. 今回与えた動作では, まず基本姿勢から z 軸回りに正の回転を行い, リンク間干渉が起きる状態になった後もしばらく同じ方向に指令を与え続けてリンク間干渉が行われていることを確認した. その後いったん z 軸回りに負の回転を行ってリンク間の距離を離れた後, 今度はリンク干渉の防止が干渉に直接影響を与えない z 成分の動きに制限が与えられていないことを確認するためにリンク干渉を起こす z 軸回りの正の回転を与えつつ z 軸方向の並進移動も与えた.

実験結果として (a) z 軸方向の変位, (b) z 軸回りの回転角, (c) リンク間距離の変化を Fig. 8 に示す. (a) に示した z 軸方向の変位は, 指令値に修正を加えず動かした場合と同様の変化をしている. また, (c) よりリンク間の干渉が回避できていることが分かる. 提案する手法で 3 組のリンク間で干渉回避を行った場合, 計算機 Gateway Optiplex GXi (CPU: MMX Pentium 200MHz) を用いて 1 回の計算を行うのに平均 0.27 [msec] を要した.

Fig. 8 Experimental result: distance between two links

5. おわりに

本稿では、リンク・障害物間の接近速度とエンドエフェクタ速度の関係を表す干渉ヤコビ行列を考案し、これを用いたロ

ボットアームの実時間障害物回避法を提案した。本手法は、計算手順に繰り返しが含まれず所要時間が短いため、実時間障害物回避に有用である。パラレルアームとシリアルアームの双方への適用が可能であるが、干渉ヤコビ行列の導出に逆運動学を用いるため、第4節で述べたように逆運動学が容易なパラレルアームに適した手法である。

最後に、御議論いただいた機械技研の小関義彦氏、テキサス大学の Jeffrey C. Hudgens 氏、御支援いただいた(財)製造科学技術センター(旧 IROFA)に感謝の意を表します。

参考文献

- [1] 重松他: "3-D 形状モデル間干渉問題の一解法", 精密機械, vol.49, no.11, pp.89-94, 1983.
- [2] 水垣他: "幾何モデルによるロボットの簡易干渉チェック", 精密機械, vol.50, no.6, pp.30-36, 1984.
- [3] 小沢他: "オフラインロボット教示における高速干渉チェックの方式", 日本ロボット学会誌, vol.4, no.2, pp.79-87, 1986.
- [4] 登尾, 有本: "オクトツリーを利用したロボットと障害物の近接2点決定アルゴリズム", 日本ロボット学会誌, vol.7, no.3, pp.151-160, 1989.
- [5] 奥富, 森: "ポテンシャル場を用いたロボットの動作決定", 日本ロボット学会誌, vol.1, no.3, pp.226-232, 1983.
- [6] S.M. Udupa: "Collision Detection and Avoidance in Computer Guidance of a Manipulator," 5th Int. Joint Conf. on Artificial Intelligence, pp.737-748, 1977.
- [7] L.A. Loef and A.H. Soni: "An Algorithm for Computer Guidance of a Manipulator in Between Obstacles," Trans. of ASME, Journal of Engineering for Industry, vol.97-3, pp.836-842, 1975.
- [8] O. Khatib and J.F. Le Maitre: "Dynamic Control of Manipulators Operating in a Complex Environment," On Theory and Practice of Robots and Manipulators, 3rd CISM/IFToMM Symp., pp.287-292, 1978.
- [9] 尾崎: "マニピュレータの占有空間を考慮した障害物回避動作の決定法", 計測自動制御学会論文集, vol.18, no.9, pp.72-79, 1982.
- [10] K. Fujimura and H. Samet: "Path Planning among Moving Obstacles Using Spatial Indexing," Proc. IEEE Int. Conf. on Robotics and Automation, pp.1662-1667, 1988.
- [11] 永田, 土谷: "移動ロボットの障害物回避に関する知的制御", 日本ロボット学会誌, vol.11, no.8, pp.1203-1211, 1993.
- [12] A.A. Petrov and I.M. Sirota: "Control of a Robot-Manipulator with Obstacle Avoidance under Little Information about the Environment," Proc. of 8th IFAC Symp, vol.4, pp.1903-1908, 1981.

兼子 一 (Hajime Kaneko)

1974年生。1998年大阪大学基礎工学部システム工学科卒業。同年、同大学大学院基礎工学研究科システム人間系専攻システム科学分野修士課程に進学、現在に至る。パラレルメカニズムの研究に従事。(日本ロボット学会学生会員)

井上健司 (Kenji Inoue)

1963年生。1988年東京大学大学院工学系研究科修士課程修了。同年、東京大学工学部助手。1993年大阪大学工学部助手。1995年同学部講師。1998年より大阪大学大学院基礎工学研究科講師となり現在に至る。超冗長マニピュレータ、ロボット協調システム、人間型ロボットの研究に従事。博士(工学)。精密工学会会員。(日本ロボット学会正会員)

新井健生 (Tatsuo Arai)

1952年生。1977年東京大学大学院工学系研究科修士課程修了。工業技術院機械技術研究所を経て、現在大阪大学大学院基礎工学研究科教授。パラレルメカニズム、マイクロマニピュレーション、作業移動型ロボット等の研究に従事。1986年~1987年MIT客員研究員。日本機械学会、計測自動制御学会、IEEE等の会員。(日本ロボット学会正会員)

前 泰志 (Yasushi Mae)

1971年生。1993年大阪大学工学部電子制御機械工学科卒業。1998年同大学大学院工学研究科電子制御機械工学専攻博士課程修了。博士(工学)。同年、同大学大学院基礎工学研究科助手。コンピュータビジョン、知能ロボットの研究に従事。情報処理学会の会員。(日本ロボット学会正会員)