平成 12 年度　卒業論文

四脚ロボットによる
サッカー行動生成のための
局所環境認識

Recognition of Local Environment for Soccer by Quadruped Robots

指導教官　　湯浅 秀男　助教授

工学部 精密機械工学科

学生証番号　　90316

上田 隆一

目 次

第1章　序論

1.1 研究の背景

1.1.1 RoboCup

毎年夏，ロボットサッカーの世界大会である“ RoboCup ”が開催されている．RoboCup 開催の意義は，複数の自律ロボットがサッカーの試合を実現するというロボティクスと人工知能の研究における新たな標準問題となっていることである [浅田 2000].

かつて，人工知能の標準問題としてコンピュータチェスがあった．コンピュータチェスでは,「人間のグランドマスターを打ち負かすコンピュータプログラム」が最終目標であり，1997 年にその目標は達成された．そして，コンピュータチェスのプログラムを作成する過程で開発されたアルゴリズムやアーキテクチャは幅広い分野で現在利用されており，この事実は，目標が達成されたことよりも重要と言える．

RoboCup にも同様の側面がある．すなわち，最終目標を,「FIFA ルールに従い，2050 年までに自律型のヒューマノイドロボット 11 台で人間のワールドカップチャンピオンを打破する」ことと定め，その解決に向けて研究を集中させ[1)]，その過程で様々な技術革新を起こそうというのである．

1.1.2 標準問題としてのサッカー

RoboCup におけるサッカーは，チェスと比較して Table1.1 のような特徴がある．Table1.1 を見て分かるように，RoboCup では，コンピュータシミュレーション上の人工知能研究では扱われない問題や，扱いにくかった問題が扱われる．また，人工知能以外に RoboCup において扱われる問題には，ロボットの移動機構の製作や，センサの製作などのハードウェア開発がある．

題材としてサッカーが選ばれているのは，

- 現状の技術で実現可能であるが，最適解を見つけることは非常に困難である．

[1)] 現在，世界 35ヶ国，3000 人以上もの研究者，学生が参画している．

- 解の優劣には獲得点数という極めて明快な評価基準がある．

という側面があるからである [浅田 1997]．現在の RoboCup のように，リーグによってはいまだに「団子サッカー」から抜け出ていないレベルから，将来の RoboCup のように人と対戦できるレベルに達しても，サッカーであれば大きなルール変更をする必要がない．このことは，標準問題としては重要な性質と言える．

Table 1.1 チェスと RoboCup の比較（[浅田 2000] より）

	チェス	**RoboCup**
環境	静的	動的
状態変化	順番交代	実時間
情報	完全	不完全・不確実
センサデータ	記号的	非記号的
制御	中央集権	分散的

1.1.3 リーグ構成

RoboCup は，いくつかのリーグに分かれており，研究で扱われる課題がリーグごとに少しずつ異なっている．それぞれについて説明する．

Small-size League 直径 18[cm] 以内の大きさのロボット５台までで戦う．リモートブレイン方式[2)]で，天井カメラの画像を視覚情報として利用する．これは，知的交通システム（ITS）などで，複数のカメラを用いて乗り物を制御する課題に対応している．

Middle-size League 直径 50cm 以内のロボット（５台以内）を，完全自律分散で動作させる．実時間画像処理，行動学習，協調行動の獲得などが課題となっている．

Legged Robot League 後述．

[2)] CPU がロボットの外部にあること．

Simulation League　計算機上に仮想のサッカーフィールドを用意し，フィールドと 11 のプレーヤー制御プログラムを，ネットワークを介したクライアントサーバシステムで接続する．実機のリーグでは扱いにくいチームワーク・実時間推論・学習などの課題を扱う．

これらの他に，2002 年から，Humanoid League が導入される予定である．

1.1.4　Legged Robot League

筆者は，RoboCup の一部門“ Legged Robot League（脚式ロボットリーグ）”に参加することを前提に研究をしている．このリーグでは，SONY の四脚ロボット“ ERS-1100 ”(Fig.1.1) を共通のプラットフォームとしている．このため，次のような特徴を有している[3)]．

Fig.　1.1　ERS-1100

共通のセンシング能力　ロボットのセンサを取り替えることも，新たに付け加えることも許されていないので，同じセンサからより多くの情報を得るアルゴリズムが要求される．

限られた計算能力　CPU のクロック周波数は 100MHz，メモリは 16MB と，数年前の PC 程度の計算能力しかない．したがって，なるべく計算量の少ないアルゴリズムが要求される．

[3)] Legged Robot League に関する詳細は，付録 A を参照のこと．

多自由度　ERS-1100 には 18 の自由度がある．これは，多彩な動きを可能にする反面，胴体が揺れてセンシングに悪影響を与えるなどの原因にもなる．

これらの特徴から，Legged Robot League では純粋にソフトウェアの性能を競うことになり，共通のセンサデータから少ない処理でより多くの情報を引き出し，正確・迅速なサッカー行動を可能にするアルゴリズムが評価される．

1.2 従来の研究

1.2.1 認識と行動

ロボットサッカーのアルゴリズムは一般に,「センサを利用して外界・内界の情報を獲得し，行動を決定する」ものである．その過程のうち，行動決定以前の段階，すなわちセンサ情報から行動を決定するための情報を抽出する過程を，本研究では「認識」と定義する．その認識過程の設計方法には大きく分けて二通りが存在する（Fig.1.2）．ひとつは，ボール・ゴール・自機・他機などの「位置情報」をロボット座標系や絶対座標系などの位置情報に変換して（1），その後，位置情報に基づいて行動を決定するアルゴリズムを設計する（2）という方法である．もうひとつは，センサ情報を位置情報に変換せず，そのまま行動と直結させる（3）という方法である．後者の例として，次の研究がある．

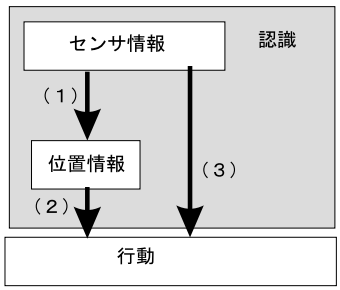

Fig. 1.2 認識，行動設計の 2 つのアプローチ

[Hugel 2000] では，CCD カメラ画像から反射的に行動を決定するアルゴリズムを設計し，シンプルなルールの組み合わせでボールに対する周り込みなどの比較的複雑な行動を実現した．その設計方法の有効性は，1999 年，2000 年の RoboCup で優秀な成績を収めたことにより実証されている．

光永ら [Mitsunaga 2000] は，教示を用い，CCD カメラ画像とそのときにとるべき行動・次に CCD カメラを向けるべき方向を対応づける方法をとった．この方法では，情報理論を利用することによって，効率の良い観測行動と，教示データを反映した行動の獲得を実現した．

これらの方法は，センサ情報の処理時間がほとんどかからないため，簡単な行

動の設計に適している．しかし，今後 Legged Robot League のレベルが上がって，より複雑な行動が要求されたとき，プログラミング量や教示量が増加するために適用が困難になると考えられる．

そこで，当チームでは前者の設計指針を採用している．つまり，画像から位置情報を抽出し，位置情報を基礎として行動設計を行っている．サッカー行動を可能にするための位置情報を Table1.2 に挙げる．

Table 1.2 位置情報

位置情報	可能となるサッカー行動	リーグ内での達成状況
ボールの方向	ボールに近づく	◎
ゴールの方向	ゴール方向にボールを蹴る	◎
他者の相対位置	衝突回避	×
自己の絶対位置	ポジション取り	○
他者の絶対位置	連携プレー	×
他者の進行方向	スルーパス	×

自己の絶対位置に関しては，ボールを蹴る方向の判断などは行える状況であるが，ポジション取りを行えるほどの精度は得られていない．また，他者の位置情報の獲得は，衝突を回避するために必要であるが，リーグ内では未解決である．他者の絶対位置や進行方向は自己の絶対位置と他者の相対位置を精度良く行えるようになった後の問題であるので，現在のリーグ内では自己位置同定と他者の相対位置を求める研究が重要と言える．そこで，次節では自己位置同定について，次々節では衝突回避についてのリーグ内での研究例を述べる．

1.2.2 自己位置同定

Legged Robot League のフィールド（Fig.1.3）には，ERS-1100 が脚ロボットでデッドレコニングの精度が悪いことを補うためにランドマーク（Fig.1.4）が立っており，これを利用した自己位置同定の研究結果が多く発表されている．その中でも多く見られるのは，フィールドをグリッド状に切り，どのグリッドにロボットが存在するかを推定する方法である．

[Wendler 2000] では，フィールドを 18 × 28 区画に区切って，区画ごとにカメラに映るランドマーク・ゴールの相対位置・大きさを記録しておき，カメラ画像と

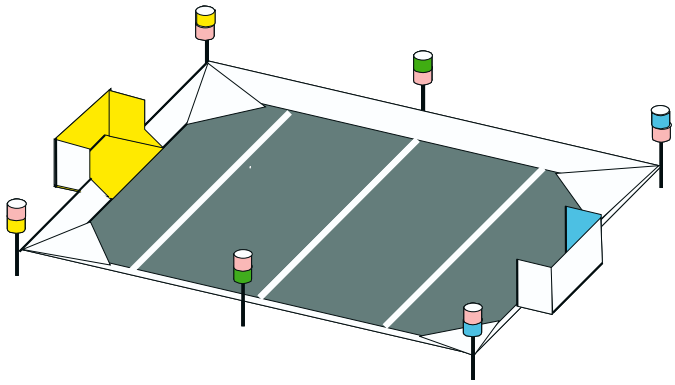

Fig. 1.3 フィールド

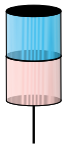

Fig. 1.4 ランドマーク

比較してどの区画にロボットが存在するかを求めるアルゴリズムを実装した.

[Veloso 1999] では，フィールドを 20 × 30 区画，ロボットの向きを 8 区画に区切り，ランドマーク 2 本以上の距離から，ベイズの公式を使用して自己位置同定する方法を実装した.

[Lenser 2000] では，上記の区画に区切る方法と異なり，点の集合によって位置を表現する方法“ Monte Carlo Localization[Fox 1999] ”を改良した“ Sensor Resetting Localization（SRL）”を用いた．これにより，デッドレコニング情報を用い，観測・計算時間を短縮が可能になった．この方法は筆者の所属するチームでも採用されている [横井 2001]．SRL については第 4 章で詳しく述べる.

自己位置同定の研究例をいくつか紹介したが，これらの手法はいずれも，デッドレコニング・ランドマーク・ゴールの情報を用いている．それ以外の情報として，壁・ラインの位置を利用することは可能であるが，そのような例は見られない．[Buschka 2000] でラインを計測することが試みられているが，実現していない.

1.2.3 衝突回避

同リーグのフィールド（Fig.1.3）では，回避すべき障害物と呼べるものが３つ存在する．

ロボット 自機の他にフィールド上に５台存在し，かなりの頻度で衝突する．

壁 傾斜がついており，乗り上げると転倒の原因となる．

自陣ペナルティーライン フォワードは自陣ペナルティーラインを越えると反則なので[4]，フォワードにとって自陣ペナルティーラインは障害物と言える．

しかし，これらを回避するアルゴリズムを実装しても，位置を精度よく計測できなければ誤差を考慮して障害物を遠い地点から回避しなければならなくなる．このことは，Legged Robot League では不利になる．具体例を挙げると次のようになる．

i. 不要な遠回りの原因になる．

ii. 壁際は敵が攻め上がる事が多く，壁に近づかないと敵の突破を許しがちになる．

iii. 守備のとき，フォワード[5]がペナルティーラインよりかなり手前で守備をやめる．

したがって，障害物回避を実現するためには，対象となる障害物の位置を精度よく計測できることが必要である．

物体の位置を計測するアルゴリズムは，ランドマークや，ゴール・ボールについて，多くのチームでアルゴリズムが開発されている．これらの計測では，Fig.1.5 で見られるように対象物の画像上での面積や幅・高さを利用することである程度の精度が得られる．

ところが，他のロボットや壁の位置を計測する場合はこの方法が利用できない．それは，ロボットの場合は Fig.1.6 のように同じ距離でも画像に映る面積や幅・高

[4] A.1.2 項参照．
[5] キーパー以外は全てフォワードなので，フォワードは守備もする．

さが一定ではなく，壁の場合はその上端を抽出することが困難であるために画像中での高さから距離を求めることができず（Fig.1.7），ラインの場合は細すぎて幅から距離を計算できないからである．

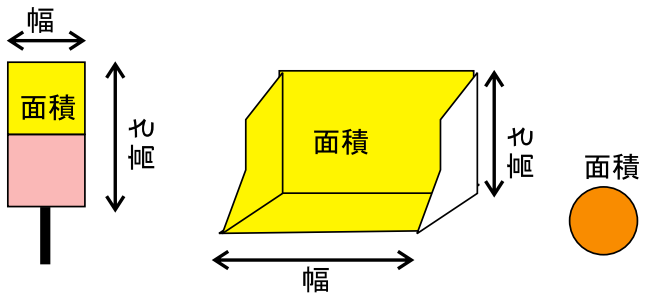

Fig. 1.5 ランドマーク・ゴール・ボールの距離を計測する手掛かり

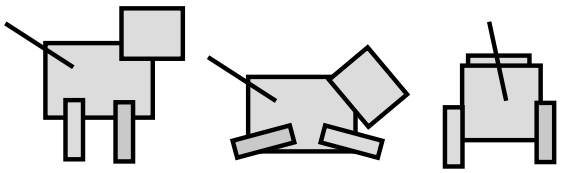

Fig. 1.6 同じ距離にあるロボットの見え方

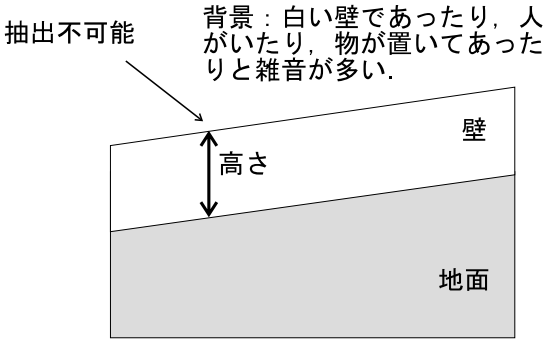

Fig. 1.7 壁の見え方

Fig.1.5 のような情報の得られない物体の位置を単眼視で計測する方法としては，[久保田 1989] の中などで利用されている逆透視変換が有力である．これは，画像平面と計測対象の点が存在している平面の位置関係を求め，画像上での点の位置

からもとの点の位置を計算する方法である．しかし，この方法を ERS-1100 に適用するには，CCD カメラの位置とフィールドの位置関係を求める必要があるが，CCD カメラが頭部・脚部の計 15 自由度の影響を受けて位置・姿勢を変化させるため計算が困難になる．そのため，リーグ内でのこの手法の実用例はない．

以上のような状況のため，Legged Robot League では，現在のところロボット同士の衝突回避の概念がほとんどない．また，ペナルティーライン回避についても iii のような現象が起こるよりは，反則を何度犯してもペナルティーライン寸前で守備をする方が有利な状況である．

1.3 研究の目的

従来研究から，Legged Robot League では以下の問題が解決されていないことが分かる．

- **画像中の面積・幅の情報が利用できない物体の位置を計測する．**

また，画像中の面積・幅の情報が利用できない物体の位置を計測することができないため，次のことが達成されていない．

i. 障害物（ロボット・壁・ライン）の位置を計測し，必要なときに回避する．

ii. 壁・ラインを観測し，自己位置同定に利用する．

i では障害物が近距離に存在するときに高精度で計測できれば，障害物にぎりぎりまで接近できる．また ii でも，ランドマークやゴールの位置計測にはカメラにそれらの大部分が収まるだけの距離が必要となるので[6)]，壁・ラインを至近距離で精度良く位置を計測できるならば，ランドマークやゴールの位置計測では得られない自己位置同定の精度が得られると期待できる．

そこで，研究の目的を，

四脚ロボットによるサッカー行動生成のための局所環境認識

と定める．「局所環境認識」とは，ロボット・壁・ラインを近距離でカメラ画像から抽出し，位置計測することを指す．そして，ランドマークのように対象物と距離をおいて画像に対象物全体を入れ，その大きさから距離をもとめて位置計測を行うもの（大域的）とは異なった手法を使用することを表している．また，この「局所的な計測手法」の精度は，数 cm 程度を目標とする．これは，大域的な位置計測では得られない精度である（Fig.1.8）．

本研究では，「局所的な計測手法」（つまり画像中の面積・幅の情報が利用できない物体の位置を計測する手法）を，逆透視変換を用いて設計・実装し，計測時間・計測誤差について評価する．そして，その応用例として i，ii のためのアルゴリズムを製作し，実装した計測手法の有用性を示す．

[6)] ランドマークの場合，約 500mm の距離が必要である．ただし，遠距離の観測が可能で，3 m までは計測できる．

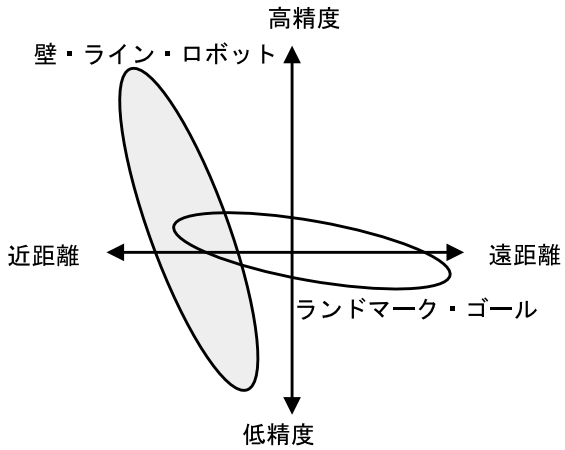

Fig. 1.8 計測対象物の距離と精度の関係

1.4 本論文の構成

第 1 章では，研究の背景，従来研究を踏まえて，本研究の目的を明らかにした．

第 2 章では，画像上での観測対象の位置を求める過程について述べる．

第 3 章では，画像上の観測対象の位置をロボット座標系に変換する過程について述べる．

第 4 章では，壁・ラインの計測結果を自己位置同定に利用する方法について述べる．

第 5 章では，第 2，3，4 章のアルゴリズムについて，誤差や計算時間等の性能を評価する．また，自己位置同定に壁・ラインの計測結果を利用した際の効果を評価する．

第 6 章では，実装したアルゴリズムで実現できる行動の例を示す．

第 7 章では，結論および今後の展望を述べる．

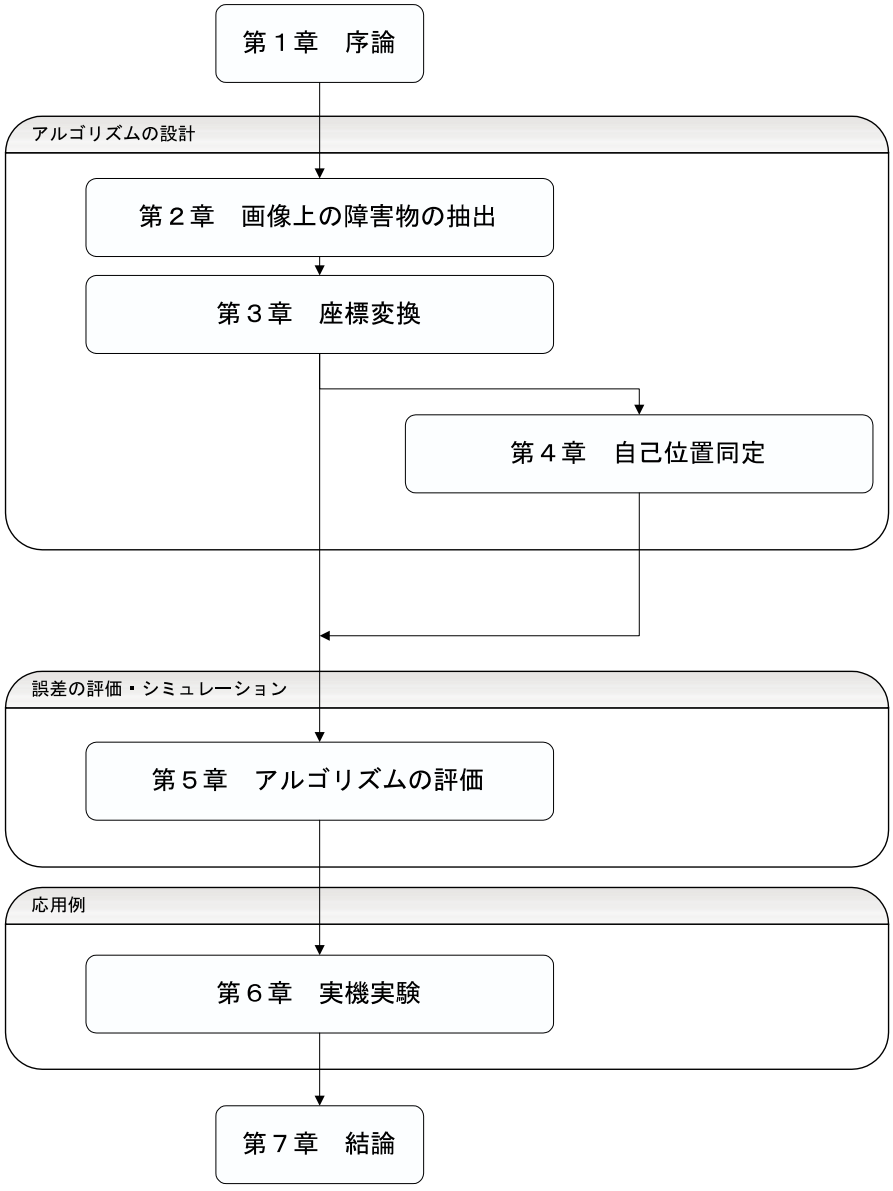

Fig. 1.9 本論文の構成

第2章　画像上の観測対象物の抽出

2.1 はじめに

本章では，観測対象物を画像上から抽出する手法について述べる．

2.1.1 構成

2.2 節において，CCD カメラ画像について座標系を設定する．

2.3 節において，画像上での壁・ラインの位置を求める過程を示す．

2.4 節において，画像上でのロボットの位置を求める過程を示す．

2.5 節において，本章を総括する．

2.1.2 解決すべき問題点

本章で設計するアルゴリズムで，問題となる点を列挙すると，次のようになる．

低画素数 CCD カメラから得られるカラー画像（Fig.2.1）は横 88 画素，縦 60 画素である．これは，移動ロボットに CCD カメラを搭載した例 [久保田 1989]（256 × 256 画素），[Hanek 2000]（384 × 288 画素）と比べるとかなりの低画素数である．

フィールド外のノイズ ロボットの視野には，フィールド外に存在する人や物などの雑音が入る．

フィールド内のノイズ 試合では，自機の他に５台のロボットがフィールド上で動き回る．これらは，影などのノイズを作り出したり，視界を遮ったりする．これは，壁・ラインの位置を求める際にノイズとなる．

CPU の処理速度 画像処理は一般に計算量が多く，これを 100MHzCPU で行わなければならない．本章のアルゴリズムを障害物回避に使用する場合を考えると，許される計算時間はせいぜい１秒程度である．

2.2 座標系の設定

Fig.2.1 のように，88×60 の CCD カメラ画像について，左上端の画素の中心を原点，右上端の画素の中心を $(87, 0)$，左下端の画素の中心を $(0, 59)$ として画像座標系 Σ_{image} を設定する．以後，$x_{\mathrm{m}} = 87, y_{\mathrm{m}} = 59$ とする．このとき，Fig.2.2 のように，画像と光軸の交点は $(x_{\mathrm{m}}/2, y_{\mathrm{m}}/2) = (43.5, 29.5)$ となる．

Fig. 2.1 画像座標系

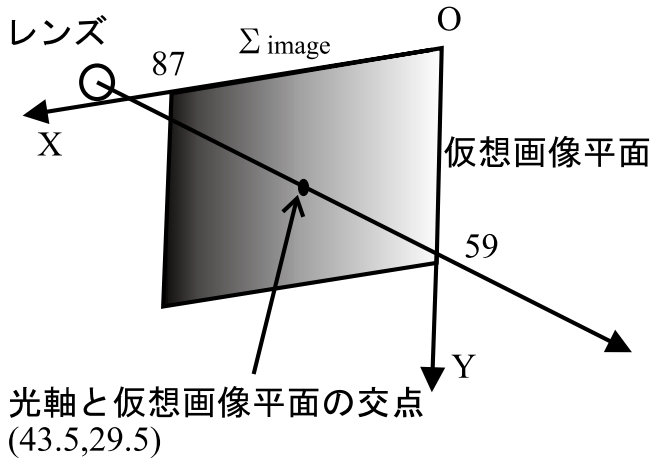

Fig. 2.2 光軸との関係

2.3　壁・ラインとフィールドの境界線抽出

本節では，壁・ラインとフィールドの境界線（エッジ）をカメラ画像から抽出する過程について説明する．本過程で設計するアルゴリズムの処理の流れを Fig.2.3 に示す．それぞれの過程について順に説明する．

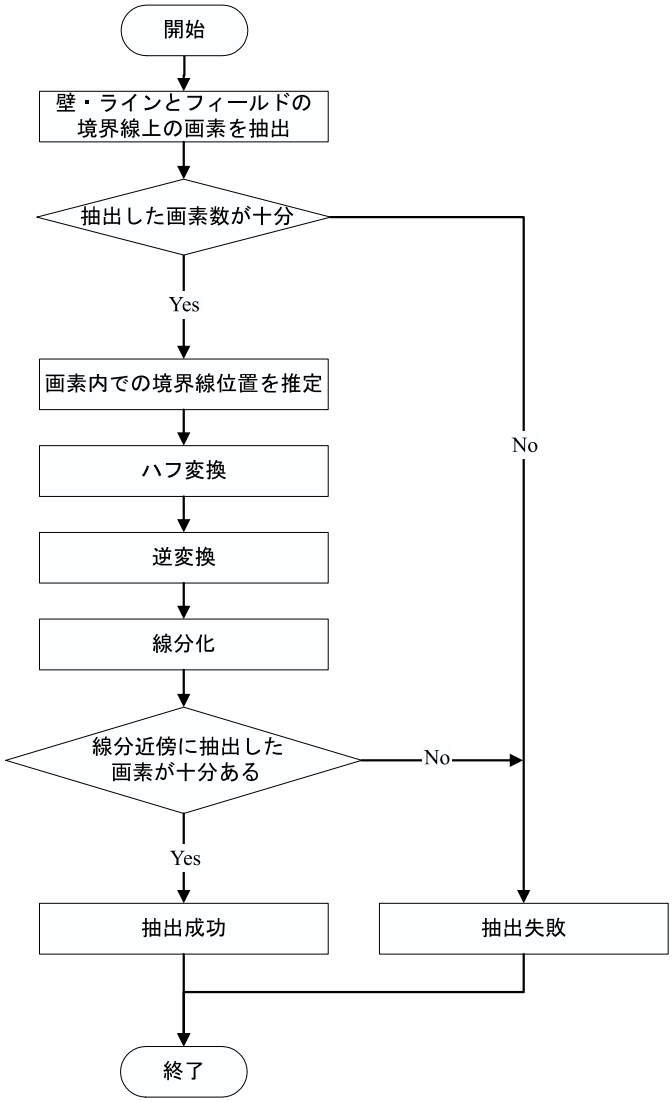

Fig.　2.3　壁・ラインの抽出の流れ

2.3.1 境界画素の抽出

ラインと，フィールドを画像から区別する手掛かりには，次の２つが考えられる．

色情報 A.2節で説明するC画像（Fig.A.8）を利用する．C画像の“ Dark Green ”はフィールドの色に合わせてある（Fig.2.4）．

輝度 ラインは白色で明るいのに対し，フィールドは暗い緑色で，両者は輝度が大きく異なる（Fig.2.5）．

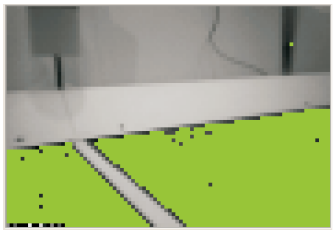

Fig. 2.4 “ Dark Green ”の情報

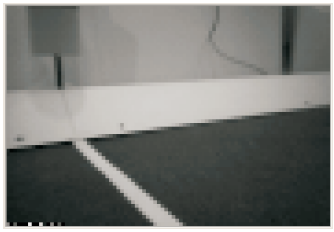

Fig. 2.5 輝度

そこで，まずC画像でフィールドの縁の画素を抽出し，抽出した画素が，前後左右いずれかの画素と輝度が大きく異なる場合にエッジ上の点として抽出するという方法をとった．抽出の結果をFig.2.6に示す．

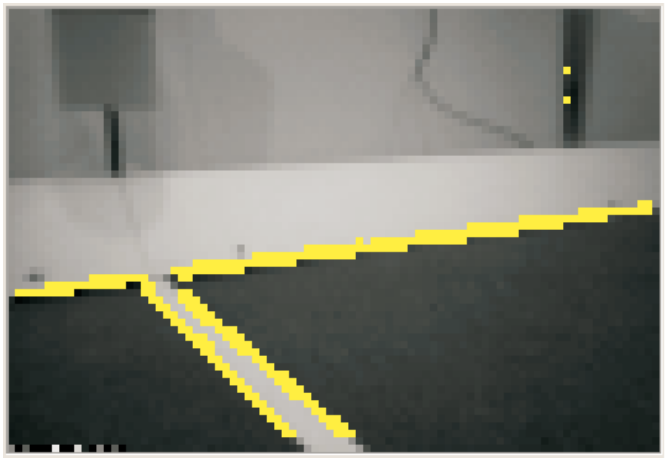

Fig. 2.6 抽出された画素

2.3.2 画素内でのエッジ位置の推定

2.3.1 項で抽出した画素は，Fig.2.6 のように非常に粗く，このままではこれらの点から直線を導く際に大きな誤差が生じる．そこで，エッジが通っている画素内でのより細かいエッジの位置を推定する．

推定は，

$$\begin{aligned}(\text{画素の輝度}) = \{&(\text{画素内の壁・ラインの面積}) \times (\text{周囲の壁・ラインの輝度}) \\ &+(\text{画素内のフィールドの面積}) \times (\text{周囲のフィールドの輝度})\} \\ &/(\text{画素の面積}) \quad (2.3.1)\end{aligned}$$

となっていると仮定して行う．エッジの通っている画素内に Fig.2.7 のように座標を設定し，求める位置を (x_e, y_e)，上下左右の輝度をそれぞれ p_t, p_b, p_l, p_r，エッジの映っている画素の輝度を p_c とすると，

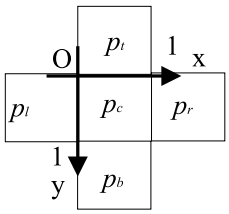

Fig. 2.7 推定方法

$$x_e = \frac{p_c - p_l}{p_r - p_l} \quad (2.3.2)$$

$$y_e = \frac{p_c - p_t}{p_b - p_t} \quad (2.3.3)$$

となる．この方法で，画素内のエッジの位置を求めると，Fig.2.8 のように，より詳細にエッジの位置を表すことができる．

2.3.3 ハフ変換

抽出した点を以下に説明する手順でハフ変換し，エッジを抽出する（Fig.2.9）.

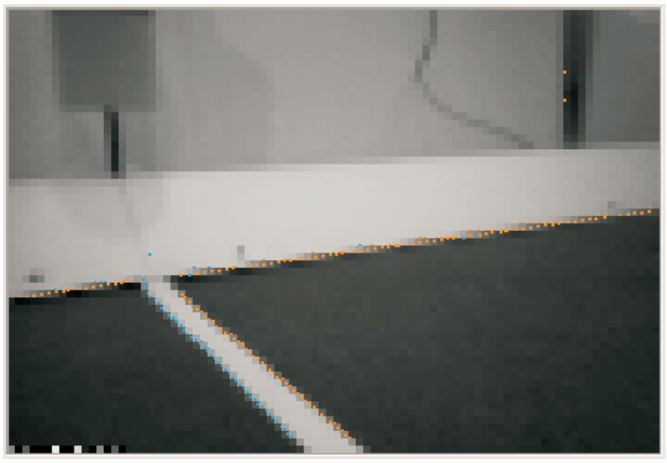

Fig. 2.8 推定

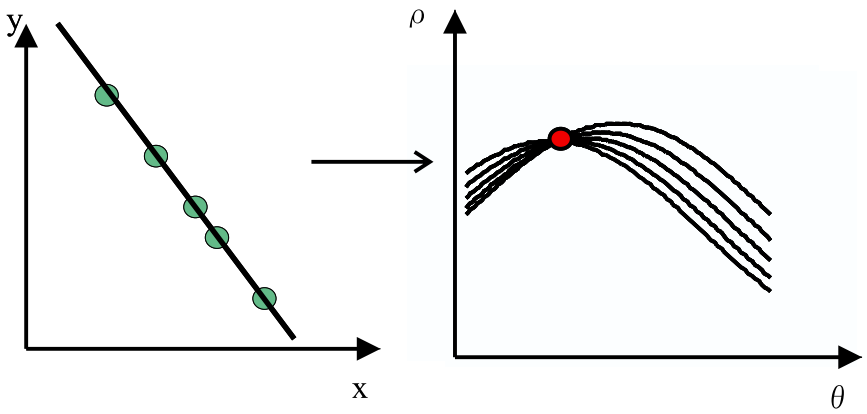

Fig. 2.9 ハフ変換

理論上の手順

i. $\Sigma_{\text{image}}(x, y)$ とハフ変換平面 $h(\theta, \rho)$ の関係を次式のように定める．

$$\rho = x\cos\theta + y\sin\theta \tag{2.3.4}$$

ii. 抽出した点 $\mathrm{P}_i(x_i, y_i) \quad (i = 1, 2, 3, \ldots, n)$ を，$h(\theta, \rho)$ 上の曲線 $\rho = x_i\cos\theta + y_i\sin\theta$ に変換する．

iii. $h(\theta, \rho)$ 上で，曲線が一番多く交わっている点 $\mathrm{Q}_{\max}(\theta_0, \rho_0)$ を抽出する．

iv. $\mathrm{Q}_{\max}$ を逆変換し，Σ_{image} 上の直線 $\rho_0 = x\cos\theta_0 + y\sin\theta_0$ を得る．

実際の設計

上記の手順を，次のようにプログラミングした．

(1) 抽出した点について左右の画素の輝度を比較し，右の明るいもの（Fig.2.8 で青い点），左の明るいもの（Fig.2.8 で赤い点）に分類する．
これは，Fig.2.8 で見られるようなラインの両端など，エッジが非常に近い位置に 2 本以上ある場合に，ハフ変換で直線が精度良く抽出できないときがあるからである．

(2) ハフ変換平面 $h(\theta, \rho)$ の分解能を θ 軸方向に 2deg，ρ 軸方向に 1 画素とし，2 次元配列とする．この 2 次元配列を，メモリ上に 2 つ確保する．

(3) 分類した点の集合ごとに，ハフ変換する．
この過程では，抽出した点 $\mathrm{P}_i(x_i, y_i)$ に対して次のような演算をする．

   (a) $\rho = x_i \cos\theta + y_i \sin\theta$ に $\theta = \pi m/90 (m = 0, 1, 2, \ldots, 89)$ を代入．

   (b) 得られた ρ を整数 ρ_m に丸める．

   (c) ハフ変換平面上の点 $\mathrm{Q}_m(\theta_m, \rho_m)$ に相当する配列の要素をインクリメントする．

   ここで，n 個の点に対して上記のような演算をすると，三角関数を使用する回数は，$n \times 90 \times 2$ 回となる．この際，C 言語標準ライブラリ中の $\sin(), \cos()$ をこの回数だけ呼び出すと，計算が非常に遅くなる．そこで，あらかじめ $\sin, \cos$ の $0, 2, 4, \ldots, 178$[deg] での値を配列で持たせておく方法をとった．

(4) 各々のハフ変換平面の配列から，値が最大の要素を取得．これらの要素に対応する点 $\mathrm{Q}_{\mathrm{max1}}(\theta_1, \rho_1), \mathrm{Q}_{\mathrm{max2}}(\theta_2, \rho_2)$ を得る．

(5) $\mathrm{Q}_{\mathrm{max1}}, \mathrm{Q}_{\mathrm{max2}}$ から Σ_{image} 上の直線 $l_1 : \rho_1 = x\cos\theta_1 + y\sin\theta_1, l_2 : \rho_2 = x\cos\theta_2 + y\sin\theta_2$ を求める．

以上の手法で得た直線を Fig.2.11 に示す．

2.3.4 直線から線分への変換

前項で得られた直線の両端を切り落とし，線分とする．直線 l_1, l_2 を以下の手順で処理する．

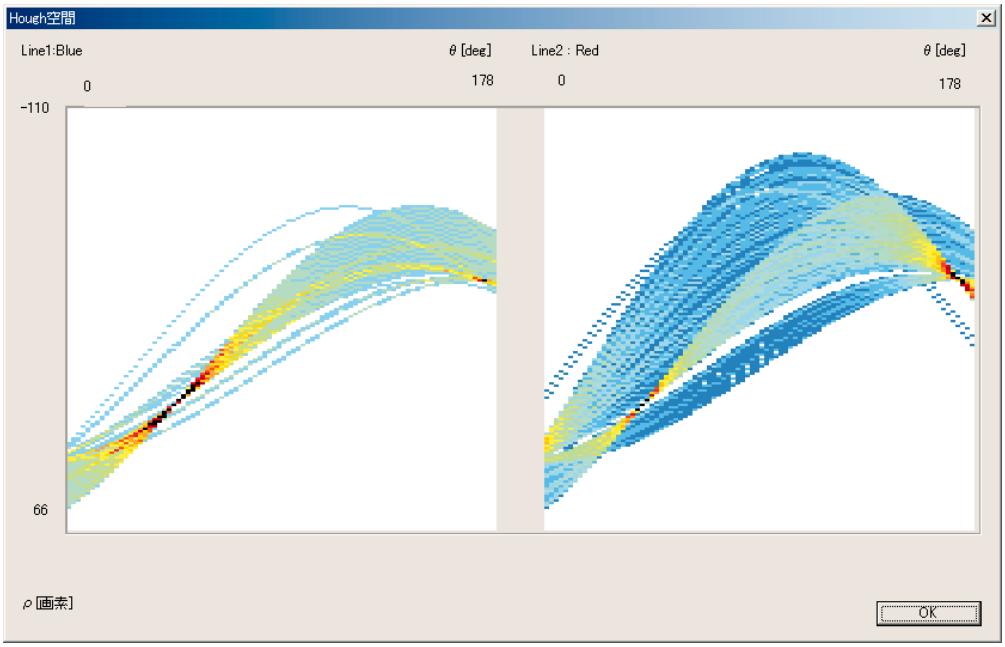

Fig. 2.10 ハフ空間

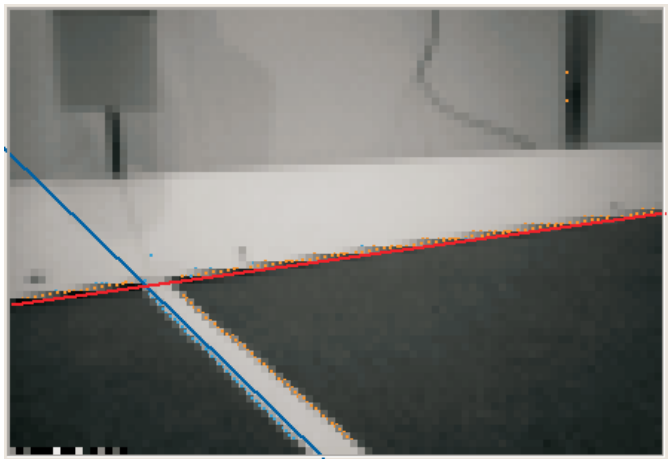

Fig. 2.11 直線の抽出

(1) 直線の近傍の点を集める.

(2) 集めた点の中で，両端の点を選ぶ.

(3) 両端の点を結ぶ.

本来なら，(1) で集めた点に対して最小二乗法を行い，エッジの位置を求めるのが望ましいが，計算時間の都合上，行わない．

2.3.5 情報の付加

得られた線分について，次の情報を付加する．

信頼度 2.3.4 項の (1) の際に集まった点の数を直線の信頼度とする．

壁かラインかの判断 C 画像でエッジの上下（あるいは左右）のどちらかに 30 画素以上ダークグリーンがなければ壁，両方に 1000 画素以上ある場合をライン，どちらでもない場合は“ unknown ”とする[1)]．

PC 上で壁・ラインを抽出した結果の例を Fig.2.12 に示す．

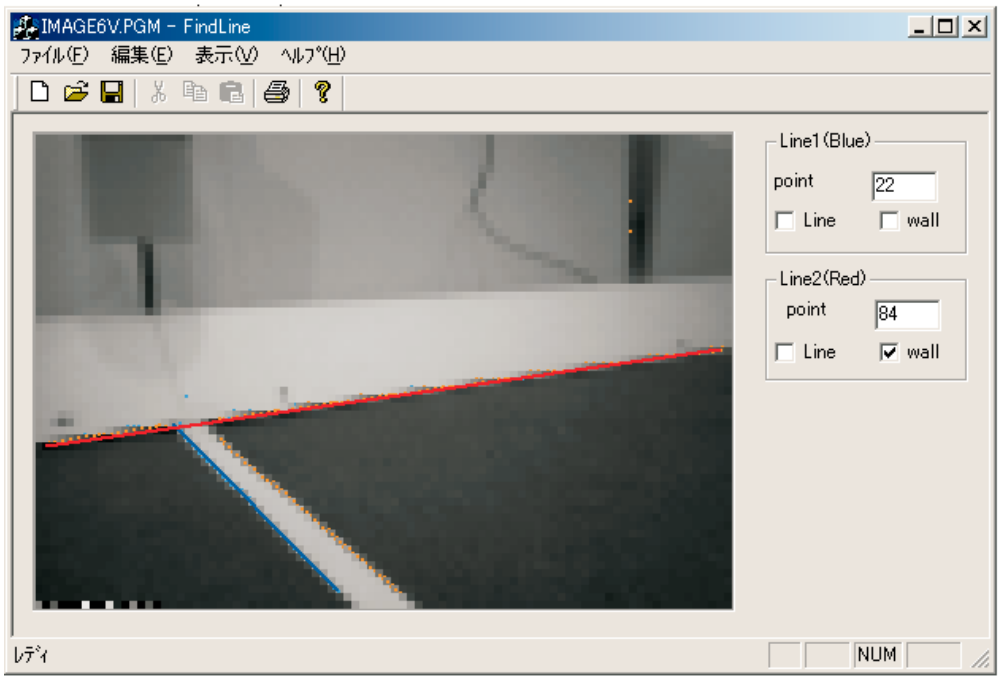

Fig. 2.12 壁・ラインの抽出結果の例

[1)]“ 30 ”や“ 1000 ”という画素数は，誤認識が絶対にないように定めた数字であるが，壁のすぐ後ろに深緑色の服を着た人がいると破綻する．

2.4 ロボットの抽出

2.4.1 抽出する点

1.2 節で述べたように，ロボットの距離を求める際は，画像上でのロボットの大きさは利用できない．また第 3 章で詳しく述べるが，画像からロボット座標系への座標変換は変換する点がフィールドと同じ高さにあることを仮定して行っている．

そこで，ロボットの接地部分（Fig.2.13）の映っている画素１つを抽出し，その点をロボット座標系に変換することでロボットの位置を求めるという方法をとる．ここでは，画素１つを抽出するまでのアルゴリズム（Fig.2.14）を説明する．

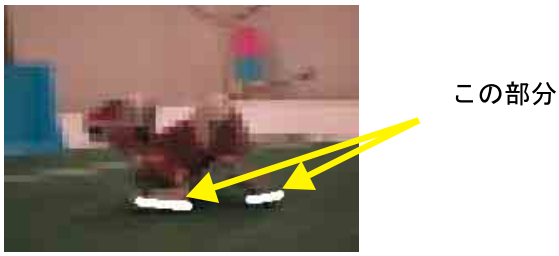

Fig. 2.13 ロボットの接地部分

2.4.2 メディアン・フィルタ

ロボットには，A.2.5 項で定められているように青あるいは赤のステッカーが貼られており，C 画像の“ Dark Blue ”,“ Red ”からロボットが認識できる．また，接地部分の検出は C 画像でフィールドを表す“ Dark Green ”を利用する．

ただし，ロボットのステッカーは面積が小さい分，周囲の雑音の影響を受けやすい．また，フィールドが映っていても“ Dark Green ”と判断されない画素がたまにあるが，これが原因となってかなり手前の画素が接地部分と検出される恐れがある．そこで，C 画像の“ Dark Blue ”,“ Red ”,“ Dark Green ”についてメディアン・フィルタ [八木 1992] をかける．

メディアン・フィルタは，フィルタにかける画素と，その周囲の８つの画素の合計９画素のパラメータ（輝度など）を比較して，パラメータの中央値を採用す

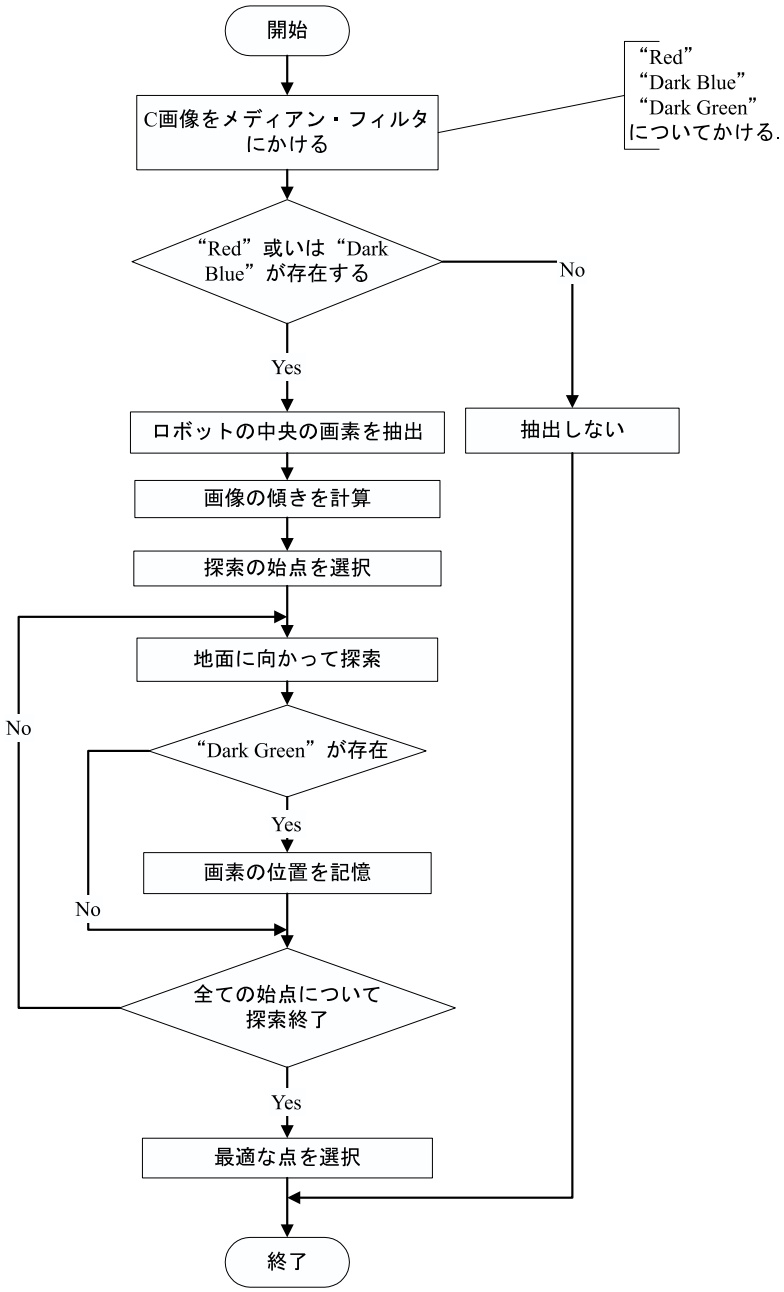

Fig. 2.14 ロボット抽出アルゴリズムの流れ

るというものである．C 画像は一色について 2 値化されているので，9 画素のうちのうちの 5 つが true であれば true，5 つが false であれば false を代入する．

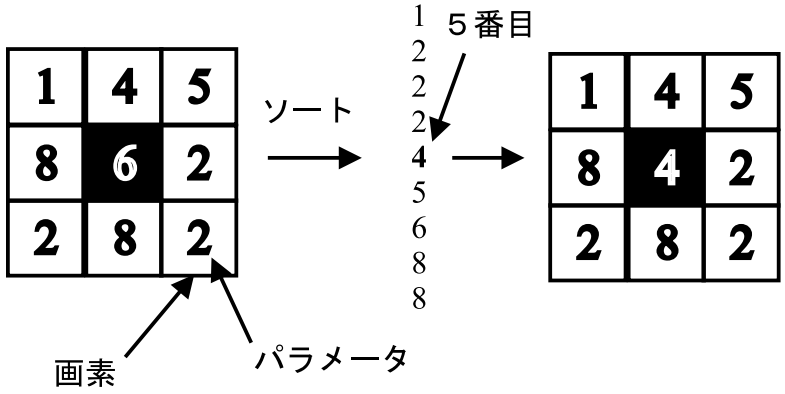

Fig. 2.15 メディアン・フィルタの処理例

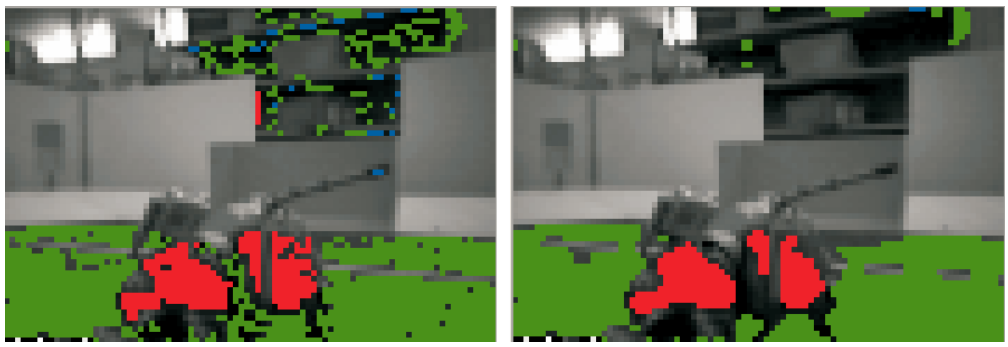

Fig. 2.16 メディアン・フィルタの処理例（左：処理前，右：処理後）

2.4.3 探索の手順

ロボットの中心位置の検出 接地部分を探すための基点として，画像でのロボットの中心を求める．中心は，C 画像上の“ Red ”（あるいは“ Dark Blue ”）の画素数を左右で同じ数にする直線と，上下で同じ数にする直線の交点とする．

画像の傾きの計算 画像が水平な状態（Σ_{image} の y 軸が下向きで，地面と垂直に交わる状態）から，光軸を中心にどれだけ回転しているかを求める．これは，第 3 章で計算するカメラ座標系からロボット座標系への変換行列 T を使用する．しかしまだ T の求め方を説明していないので，本節では，画像が左回りを正として φ 傾いているとして話を進める．実際の計算方法は 3.6 節で述べる．

探索の始点の決定　画像が φ 傾いているとき，中心 $(c_{\mathrm{x}}, c_{\mathrm{y}})$ を通る地面に水平な直線上にある点の座標は，次のように表現できる．

$$(u\cos\varphi + c_{\mathrm{x}}, u\sin\varphi + c_{\mathrm{y}}) \qquad (u : \text{任意の実数}) \tag{2.4.1}$$

この座標に $u = -20, -19, \ldots, 19, 20$ を代入した 41 の座標を始点とする（Fig.2.17）．

Fig.　2.17　始点

フィールド方向への探索　各始点について，ベクトル $(-\sin\varphi, \cos\varphi)$ の方向に C 画像を探索し，フィールドの色である“ Dark Green ”があったらその画素の座標と始点からの距離を記録する．画像の端まで探索して“ Dark Green ”がなければ探索失敗として終了する（Fig.2.18）．

Fig.　2.18　探索後

出力（探索が１回以上成功した場合）　探索した“ Dark Green ”の画素には，接地部分ではなく胴体の下端を抽出したものがあるので，探索した画素のうちで接地部分である可能性が最も高い画素を出力しなくてはならない．そこで，探索が成功した画素について，始点を構成する直線からの距離を比較して最も距離の長いものを選択し，出力する（Fig.2.19）．

出力（探索が１回以上も成功しなかった場合） 探索した画像の端のなかで，始点から最も遠いものを選び，端の画素を出力する．この画素を第 3 章の座標変換でロボット座標系にすると「この地点より手前にロボットが存在する」という情報として利用できる．ただし探索が成功した場合の出力と区別ができるようにする．

Fig. 2.19 出力

2.5 おわりに

本章では，観測対象物を画像上から抽出する手法について述べた．

2.2 節において，CCD カメラ画像について座標系を設定した．

2.3 節において，画像上での壁・ラインの位置を抽出する過程を示した．この過程では，低画素数による量子化誤差を画素内のエッジの位置を推定することで防ぎ，抽出にハフ変換を用いることでノイズに対してロバストなエッジ抽出を実現した．

2.4 節において，画像上でのロボットの接地部分を探索する過程を示した．この過程では，画像の傾きと色情報（C 画像）を利用して画像全体を探索する必要をなくし，処理を軽くした．また，ノイズに起因する C 画像の不完全な色情報をメディアンフィルタで補った．

第3章　座標変換

3.1 はじめに

本章では，画像上の点を逆透視変換を用いて画像上からロボット座標系に変換する過程を扱う．また，2.4.3 項で必要とするカメラ画像の傾きの計算式を求める．

3.1.1 構成

3.2 節において，カメラ座標系，ロボット座標系，胴体座標系について設定する．

3.3 節において，画像座標系からカメラ座標系への変換式を求める．

3.4 節において，カメラ座標系からロボット座標系への変換式を求める．

3.5 節において，座標変換アルゴリズムを説明する．

3.6 節において，カメラ画像の傾きの計算式を求める．

3.7 節において，本章を総括する．

3.1.2 解決すべき問題点

本章で設計するアルゴリズムで，問題となる点は，CCD カメラの姿勢が頭部 3 自由度，脚部 12 自由度の合計 15 自由度の影響を受けることである．また，歩き地面に接地する部分はつま先だけでなく，肘や手首など何ヶ所かあることも，問題を複雑にしている．

3.2 座標系の設定

カメラ座標系

カメラ座標系 Σ_{cam} を Fig.3.1 のように定義する.

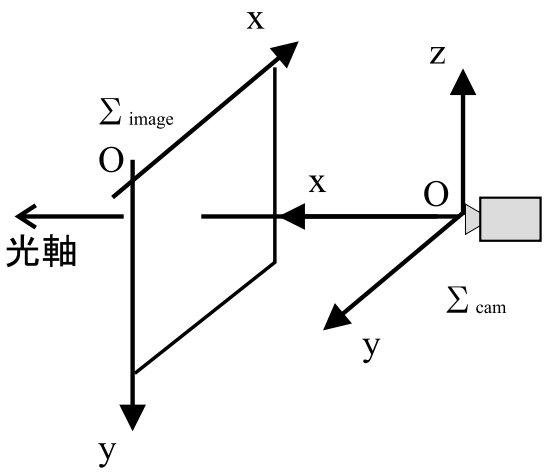

Fig. 3.1 カメラ座標系

- 原点：レンズの焦点.
- x 軸：光軸と平行，方向はカメラの前方.
- z 軸：Σ_{image} の y 軸と平行で逆向き.
- y 軸：Σ_{image} の x 軸と平行で逆向き.

ロボット座標系と胴体座標系

ロボット座標系 Σ_{robot} を Fig.3.2 のように定義する．また，計算の補助のため，胴体座標系 Σ_{body} を定義する．

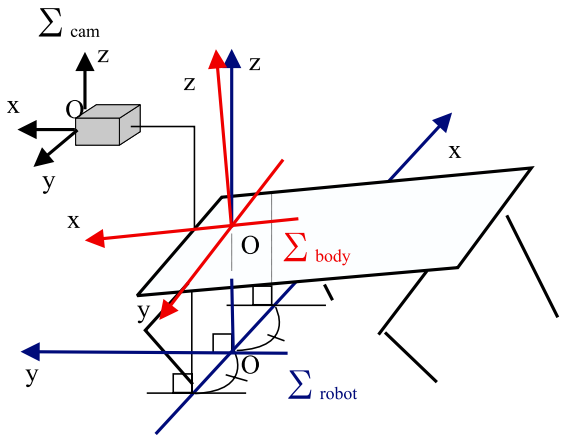

Fig. 3.2 ロボット座標系

ロボット座標系

- 原点：両前脚と胴体の各接合部分からフィールドに降ろした 2 本の垂線の足の中点.
- x 軸：両前脚の胴体の接合部分からフィールドに降ろした 2 本の垂線の足を通り，ロボットの右手方向を正とする.
- y 軸：フィールドに平行で x 軸と垂直に交わり，ロボットの前方を正とする.
- z 軸：フィールドに対し鉛直方向で，上方を正とする.

胴体座標系

- 原点：両前脚と胴体の接合部分の中点.
- x 軸：両前脚と胴体の接合部分を結んだ線分に垂直で，胴体に平行．ロボットの前方を正とする.
- y 軸：両前脚と胴体の接合部分を結んだ線分を通る.
- z 軸：胴体に対して垂直で，上方を正とする.

3.3 画像からカメラ座標系への変換式

Σ_{image} と Σ_{cam} の対応づけの方法を説明する[1)]．Fig.3.3 において，観測対象となる点を Σ_{cam} で $\mathrm{P_c}(x_{\mathrm{c}}, y_{\mathrm{c}}, z_{\mathrm{c}})$，この点を仮想画像平面に透視投影した点を $\mathrm{P_i}$ とする．

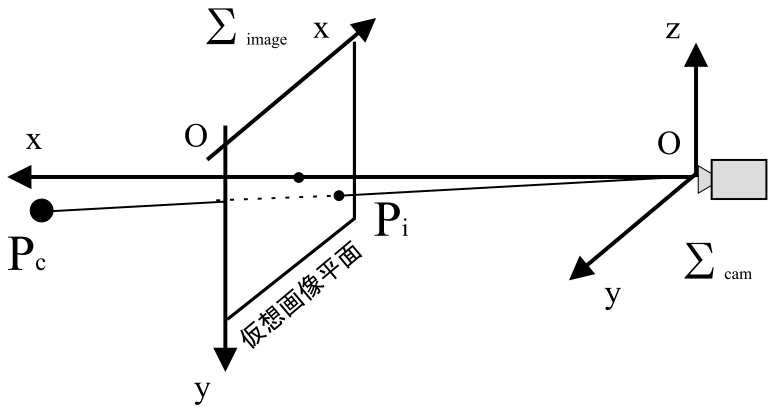

Fig. 3.3 画像座標系とカメラ座標系の関係

Fig.3.3 を Σ_{cam} の xy 平面，xz 平面と平行な方向から見ると Fig.3.4 のようになる．h_{m} は CCD カメラ画像の水平方向の画角で 53[deg]，v_{m} は CCD カメラ画像の垂直方向の画角で 41[deg] である．

ここで，$\mathrm{P_i}$ の Σ_{image} での座標を $(x_{\mathrm{i}}, y_{\mathrm{i}})$ とすると，画像の水平方向について以下のような式が成り立つ．

$$\tan h = \frac{-y_{\mathrm{c}}}{x_{\mathrm{c}}} \tag{3.3.1}$$

$$\frac{x_{\mathrm{m}}}{\tan h_{\mathrm{m}}} = \frac{(x_{\mathrm{i}} - x_{\mathrm{m}}/2)}{\tan h} \tag{3.3.2}$$

式 (3.3.1),(3.3.2) から次の式が成り立つ．

$$y_{\mathrm{c}} = x_{\mathrm{c}} \frac{x_{\mathrm{m}}/2 - x_{\mathrm{i}}}{x_{\mathrm{m}}} \tan h_{\mathrm{m}} \tag{3.3.3}$$

また，同様にして画像の垂直方向について以下の式が成り立つ．

$$z_{\mathrm{c}} = x_{\mathrm{c}} \frac{y_{\mathrm{m}}/2 - y_{\mathrm{i}}}{y_{\mathrm{m}}} \tan v_{\mathrm{m}} \tag{3.3.4}$$

[1)] ERS-1100 の CCD カメラ画像では画像の歪みは確認されないので，[久米 1997] で述べられているようなレンズ歪み補正のためのキャリブレーションは行わない．

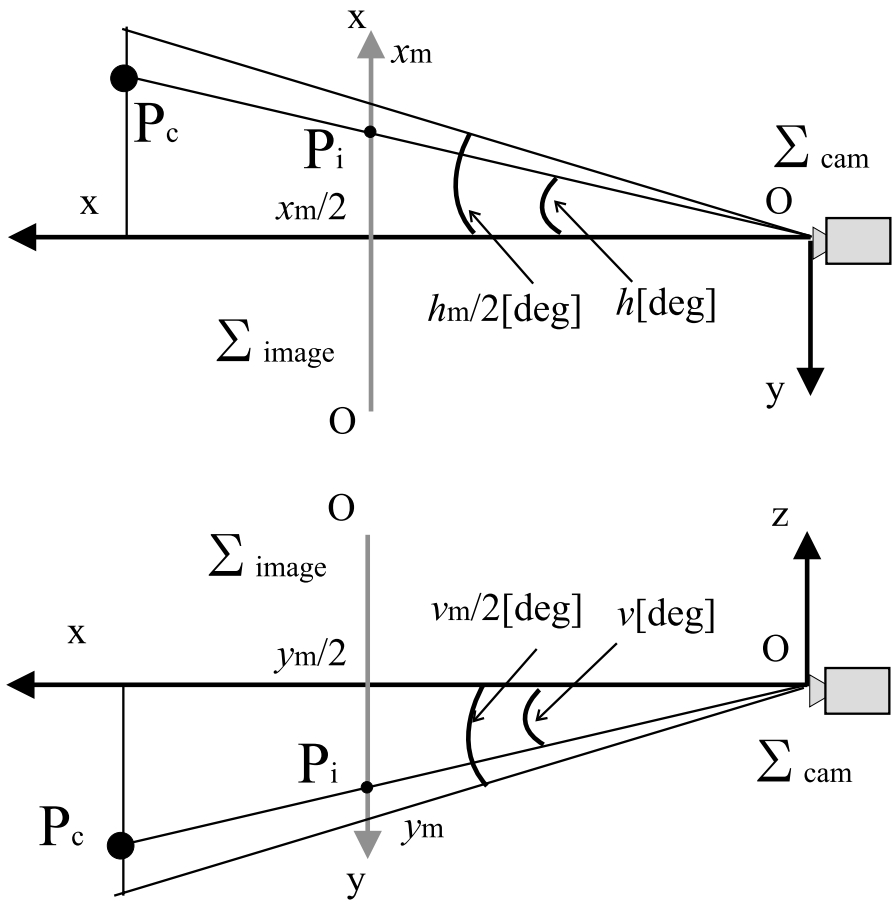

Fig. 3.4 画像座標系とカメラ座標系の関係

3.4 カメラ座標系からロボット座標系への変換式

Σ_{cam} から Σ_{robot} への変換は同次変換 [吉川 1988] を用いる.

3.4.1 カメラ座標系から胴体座標系への同次変換行列

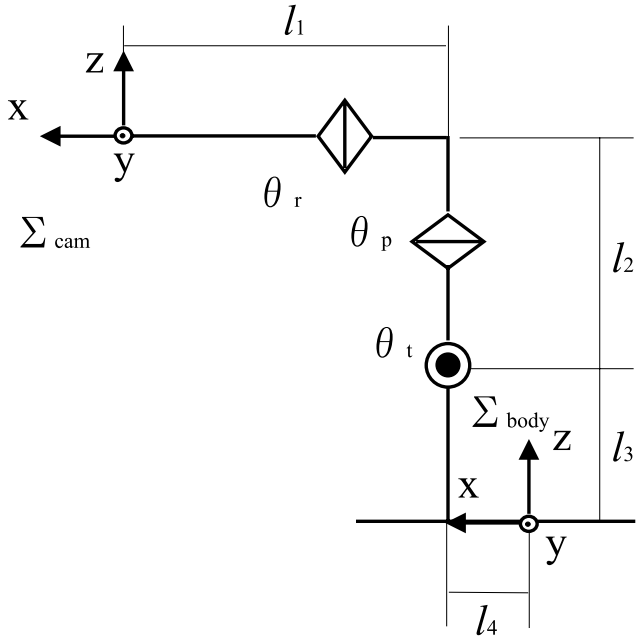

Fig. 3.5 頭部関節角

頭部の関節角の定義

- チルト角 θ_{t}：Fig.3.5 のように関節の前後の軸が平行なときを 0[deg] とし，Fig.3.5 で右に回転するときを正とする.
- パン角 θ_{p}：$\theta_{\mathrm{t}} = 0$[deg] のとき，Σ_{cam} と Σ_{body} の x 軸が平行となるときに 0[deg] とし，Fig.3.5 で Σ_{cam} が手前に回転するときを正とする.
- ロール角 θ_{r}：$\theta_{\mathrm{t}} = 0$[deg]，$\theta_{\mathrm{p}} = 0$[deg] のとき，Σ_{cam} と Σ_{body} が平行になるときを $\theta_{\mathrm{r}} = 0$[deg] とし，Fig.3.5 で Σ_{cam} の z 軸が向こう側に倒れる向きに回転するときを正とする.

同次変換行列

以上のように角度を定義すると，$\Sigma_{\rm cam}$ から $\Sigma_{\rm body}$ への同次変換行列 ${}^{\rm body}T_{\rm cam}$ は次のようになる．

$$
{}^{\rm body}T_{\rm cam} = \begin{pmatrix} \cos\theta_{\rm t} & 0 & -\sin\theta_{\rm t} & l_4 \\ 0 & 1 & 0 & 0 \\ \sin\theta_{\rm t} & 0 & \cos\theta_{\rm t} & l_3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\theta_{\rm p} & -\sin\theta_{\rm p} & 0 & 0 \\ \sin\theta_{\rm p} & \cos\theta_{\rm p} & 0 & 0 \\ 0 & 0 & 1 & l_2 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & l_1 \\ 0 & \cos\theta_{\rm r} & -\sin\theta_{\rm r} & 0 \\ 0 & \sin\theta_{\rm r} & \cos\theta_{\rm r} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.4.1}
$$

3.4.2 胴体座標系からロボット座標系への同次変換行列

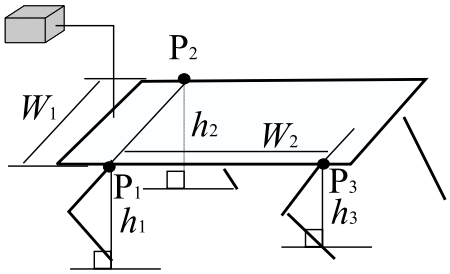

Fig. 3.6 脚の高さ，幅

脚と胴体の接合部を点 $\rm P_1$（左前脚），$\rm P_2$（右前脚），$\rm P_3$（左後脚）としてフィールドからの高さをそれぞれ h_1, h_2, h_3 とし，$|{\rm P_1P_2}| = W_1, |{\rm P_1P_3}| = W_2$[2]とする（Fig.3.6）．このとき，$\Sigma_{\rm robot}$ において，${\rm P_1}(-\sqrt{W_1^2-(h_1-h_2)^2}/2, 0, h_1)$，${\rm P_2}(\sqrt{W_1^2-(h_1-h_2)^2}/2, 0, h_2)$ となる．

ここで ${\rm P_3}(x_3, y_3, h_3)$ とすると，ERS-1100 の形状から ${\rm P_1P_3} \perp {\rm P_1P_2}$ が成り立つ

[2] ERS-1100 では，$W_1 = 85.0$[mm]，$W_2 = 130.2$[mm]

ので，$\overrightarrow{\mathrm{P_1P_2}} \cdot \overrightarrow{\mathrm{P_1P_3}} = 0$ より，

$$\begin{aligned} x_3 &= -\frac{\sqrt{W_1^2 - (h_1 - h_2)^2}}{2} - \frac{(h_1 - h_2)(h_1 - h_3)}{\sqrt{W_1^2 - (h_1 - h_2)^2}} \\ &= -\frac{\sqrt{W_1^2 - (h_1 - h_2)^2}}{2} - \frac{W_2}{\sqrt{1 - \left(\dfrac{h_1 - h_2}{W_1}\right)^2}} \frac{h_1 - h_2}{W_1} \frac{h_1 - h_3}{W_2} \end{aligned} \tag{3.4.2}$$

となる．また，$|\mathrm{P_1P_3}| = W_2$ なので，

$$\sqrt{\left(x_3 + \frac{\sqrt{W_1^2 - (h_1 - h_2)^2}}{2}\right)^2 + y_3^2 + (h_1 - h_3)^2} = W_2 \tag{3.4.3}$$

となり，これに式 (3.4.2) を代入して解くと，

$$y_3 = -\frac{W_2}{\sqrt{1 - \left(\dfrac{h_1 - h_2}{W_1}\right)^2}} \sqrt{1 - \left(\frac{h_1 - h_2}{W_1}\right)^2 - \left(\frac{h_1 - h_3}{W_2}\right)^2} \tag{3.4.4}$$

を得る．ここで，

$$\alpha = (h_1 - h_2)/W_1 \tag{3.4.5}$$

$$\beta = (h_1 - h_3)/W_2 \tag{3.4.6}$$

とおくと，式 (3.4.2),(3.4.4) はそれぞれ，

$$x_3 = -W_1 \frac{\sqrt{1 - \alpha^2}}{2} - \frac{W_2 \alpha \beta}{\sqrt{1 - \alpha^2}} \tag{3.4.7}$$

$$y_3 = -W_2 \frac{\sqrt{1 - \alpha^2 - \beta^2}}{\sqrt{1 - \alpha^2}} \tag{3.4.8}$$

とできる．3.2 項での胴体座標系の定義から，Σ_{body} の x,y,z 軸に平行な単位ベクト

ル $\boldsymbol{e}_{\mathrm{bx}}, \boldsymbol{e}_{\mathrm{by}}, \boldsymbol{e}_{\mathrm{bz}}$ を Σ_{robot} で表すと次のようになる．

$$\begin{aligned}\boldsymbol{e}_{\mathrm{bx}} &= \overrightarrow{\mathrm{P_1P_3}}/|\mathrm{P_1P_3}| \\ &= \left(-\frac{\sqrt{W_1^2-(h_2-h_1)^2}}{2} - x_3, -y_3, h_1-h_3\right)/W_2 \\ &= \left(\frac{\alpha\beta}{\sqrt{1-\alpha^2}}, \frac{\sqrt{1-\alpha^2-\beta^2}}{\sqrt{1-\alpha^2}}, \beta\right)\end{aligned} \tag{3.4.9}$$

$$\begin{aligned}\boldsymbol{e}_{\mathrm{by}} &= \overrightarrow{\mathrm{P_1P_2}}/|\mathrm{P_1P_2}| \\ &= \left(-\sqrt{W_1^2-(h_1-h_2)^2}, 0, h_1-h_2\right)/W_1 \\ &= (-\sqrt{1-\alpha^2}, 0, \alpha)\end{aligned} \tag{3.4.10}$$

$$\begin{aligned}\boldsymbol{e}_{\mathrm{bz}} &= \boldsymbol{e}_{\mathrm{bx}} \times \boldsymbol{e}_{\mathrm{by}} \\ &= \left(\frac{\alpha\sqrt{1-\alpha^2-\beta^2}}{\sqrt{1-\alpha^2}}, \frac{-\beta}{\sqrt{1-\alpha^2}}, \sqrt{1-\alpha^2-\beta^2}\right)\end{aligned} \tag{3.4.11}$$

また，Σ_{body} の原点の座標は，Σ_{robot} で，$(0, 0, (h_1+h_2)/2)$ と表される．従って，Σ_{body} から Σ_{robot} への同次変換行列 ${}^{\mathrm{robot}}T_{\mathrm{body}}$ は，次のようになる．

$$\begin{aligned}{}^{\mathrm{robot}}T_{\mathrm{body}} &= \begin{pmatrix} & & & 0 \\ \boldsymbol{e}_{\mathrm{bx}}^t & \boldsymbol{e}_{\mathrm{by}}^t & \boldsymbol{e}_{\mathrm{bz}}^t & 0 \\ & & & \frac{h_1+h_2}{2} \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ &= \begin{pmatrix} \frac{\alpha\beta}{\sqrt{1-\alpha^2}} & -\sqrt{1-\alpha^2} & \frac{\alpha\sqrt{1-\alpha^2-\beta^2}}{\sqrt{1-\alpha^2}} & 0 \\ \frac{\sqrt{1-\alpha^2-\beta^2}}{\sqrt{1-\alpha^2}} & 0 & \frac{-\beta}{\sqrt{1-\alpha^2}} & 0 \\ \beta & \alpha & \sqrt{1-\alpha^2-\beta^2} & \frac{h_1+h_2}{2} \\ 0 & 0 & 0 & 1 \end{pmatrix}\end{aligned} \tag{3.4.12}$$

3.5 フィールド上の点の座標変換

画像に映ったフィールド上の点を Σ_{image} で $(x_{\mathrm{i}}, y_{\mathrm{i}})$，$\Sigma_{\mathrm{cam}}$ で $(x_{\mathrm{c}}, y_{\mathrm{c}}, z_{\mathrm{c}})$，$\Sigma_{\mathrm{robot}}$ で $(x_{\mathrm{r}}, y_{\mathrm{r}}, 0)$ と表し，この点を $(x_{\mathrm{i}}, y_{\mathrm{i}})$ から $(x_{\mathrm{r}}, y_{\mathrm{r}}, 0)$ に変換する過程を説明する (Fig.3.7)．

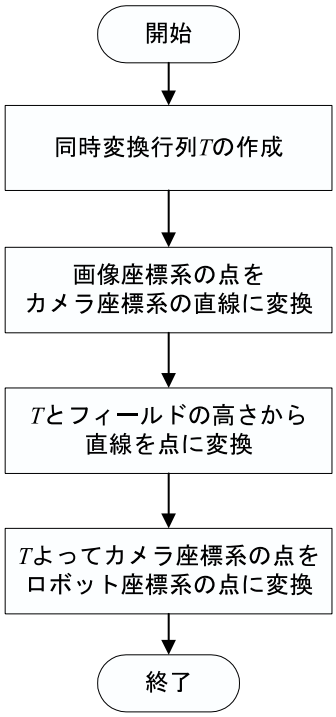

Fig. 3.7 変換過程

3.5.1 同次変換行列の作成

${}^{\mathrm{r}}T_{\mathrm{c}} = {}^{\mathrm{robot}}T_{\mathrm{body}}\ {}^{\mathrm{body}}T_{\mathrm{cam}}$ とすると，${}^{\mathrm{r}}T_{\mathrm{c}}$ を決定する変数は，$\theta_{\mathrm{t}}, \theta_{\mathrm{p}}, \theta_{\mathrm{r}}, h_1, h_2, h_3$ である．このうち $\theta_{\mathrm{t}}, \theta_{\mathrm{p}}, \theta_{\mathrm{r}}$ は頭部のエンコーダから精度の良い値が得られる．しかし h_1, h_2, h_3 については，地面に接地する脚の部分が膝や肘，手首などさまざまで，脚部のエンコーダーの値のみからは一意に決定できない．

ERS-1100 には静止しているときの姿勢が数種類あり，一歩歩くごとにそのうちのいずれかの姿勢をとる．そこで，その姿勢のときの h_1, h_2, h_3 を計測して Table3.1 のような表をプログラム中に配列で持っておき，現在の姿勢に応じて値を入れ替

えるというアルゴリズムにした.

Table 3.1 脚の高さ（誤差 ±0.5mm）

姿勢	h_1[mm]	h_2[mm]	h_3[mm]
前進１	75	80	125
前進２	80	75	125
後退１	80	83	110
後退２	83	80	112
回転１	80	80	113
⋮			

3.5.2 画像座標系からカメラ座標系への変換

3.3 節で求めた変換式 (3.3.3), (3.3.4) を用いる．この過程では，x_c は求まらず，(x_c, y_c, z_c) は式 (3.3.3), (3.3.4) で，x_c を媒介変数とする直線として表される (Fig.3.8)．この直線とフィールド面（Σ_{robot} 座標系における $z_r = 0$ の面）との交点を求めることにより，直線を点に変換する．

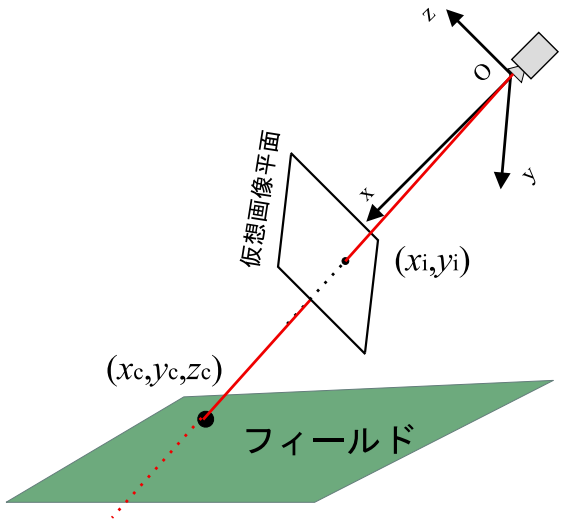

Fig. 3.8 式 (3.3.3), (3.3.4) の表す直線

3.5.3 直線の点への変換

$(x_\mathrm{c}, y_\mathrm{c}, z_\mathrm{c})$ と $(x_\mathrm{r}, y_\mathrm{r}, 0)$ の関係は次のようになる．

$$\begin{pmatrix} x_\mathrm{r} \\ y_\mathrm{r} \\ 0 \\ 1 \end{pmatrix} = {}^\mathrm{r}T_\mathrm{c} \begin{pmatrix} x_\mathrm{c} \\ y_\mathrm{c} \\ z_\mathrm{c} \\ 1 \end{pmatrix} \tag{3.5.1}$$

となる．ここで ${}^\mathrm{r}T_\mathrm{c}$ の i 行 j 列の元を t_{ij} とすると，式 (3.5.1) の第 3 行は，

$$0 = t_{31}x_\mathrm{c} + t_{32}y_\mathrm{c} + t_{33}z_\mathrm{c} + t_{34} \tag{3.5.2}$$

と表せる．一方，式 (3.3.3)，(3.3.4) より，

$$0 = t_{31}x_\mathrm{c} + t_{32}x_\mathrm{c}\frac{x_\mathrm{m}/2 - x_\mathrm{i}}{y_\mathrm{m}}\tan h_\mathrm{m} + t_{33}x_\mathrm{c}\frac{y_\mathrm{m}/2 - y_\mathrm{i}}{y_\mathrm{m}}\tan v_\mathrm{m} + t_{34} \tag{3.5.3}$$

を得る．したがって $t_{31} + t_{32}\dfrac{x_\mathrm{m}/2 - x_\mathrm{i}}{y_\mathrm{m}}\tan h_\mathrm{m} + t_{33}\dfrac{y_\mathrm{m}/2 - y_\mathrm{i}}{y_\mathrm{m}}\tan v_\mathrm{m} \neq 0$ ならば，次式が成り立つ．

$$x_\mathrm{c} = \frac{-t_{34}}{t_{31} + t_{32}\dfrac{x_\mathrm{m}/2 - x_\mathrm{i}}{y_\mathrm{m}}\tan h_\mathrm{m} + t_{33}\dfrac{y_\mathrm{m}/2 - y_\mathrm{i}}{y_\mathrm{m}}\tan v_\mathrm{m}} \tag{3.5.4}$$

x_c が求まったので，式 (3.3.3)，(3.3.4) から点 $(x_\mathrm{c}, y_\mathrm{c}, z_\mathrm{c})$ が求まる．

3.5.4 カメラ座標系からロボット座標系への変換

式 (3.3.3)，(3.3.4)，(3.5.1) から，

$$\begin{aligned} x_\mathrm{r} &= t_{11}x_\mathrm{c} + t_{12}y_\mathrm{c} + t_{13}z_\mathrm{c} + t_{14} \\ &= \left(t_{11} + t_{12}\frac{x_\mathrm{m}/2 - x_\mathrm{i}}{y_\mathrm{m}}\tan h_\mathrm{m} + t_{13}\frac{y_\mathrm{m}/2 - y_\mathrm{i}}{y_\mathrm{m}}\tan v_\mathrm{m}\right)x_\mathrm{c} + t_{14} \\ &= t_{14} - \frac{t_{11} + t_{12}\dfrac{x_\mathrm{m}/2 - x_\mathrm{i}}{y_\mathrm{m}}\tan h_\mathrm{m} + t_{13}\dfrac{y_\mathrm{m}/2 - y_\mathrm{i}}{y_\mathrm{m}}\tan v_\mathrm{m}}{t_{31} + t_{32}\dfrac{x_\mathrm{m}/2 - x_\mathrm{i}}{y_\mathrm{m}}\tan h_\mathrm{m} + t_{33}\dfrac{y_\mathrm{m}/2 - y_\mathrm{i}}{y_\mathrm{m}}\tan v_\mathrm{m}}t_{34} \end{aligned} \tag{3.5.5}$$

を得る．同様に，

$$\begin{aligned} y_\mathrm{r} &= t_{21}x_\mathrm{c} + t_{22}y_\mathrm{c} + t_{23}z_\mathrm{c} + t_{24} \\ &= t_{24} - \frac{t_{21} + t_{22}\dfrac{x_\mathrm{m}/2 - x_\mathrm{i}}{y_\mathrm{m}}\tan h_\mathrm{m} + t_{23}\dfrac{y_\mathrm{m}/2 - y_\mathrm{i}}{y_\mathrm{m}}\tan v_\mathrm{m}}{t_{31} + t_{32}\dfrac{x_\mathrm{m}/2 - x_\mathrm{i}}{y_\mathrm{m}}\tan h_\mathrm{m} + t_{33}\dfrac{y_\mathrm{m}/2 - y_\mathrm{i}}{y_\mathrm{m}}\tan v_\mathrm{m}}t_{34} \end{aligned} \tag{3.5.6}$$

となる．これで，画像上の点をロボット座標系で表せた．

3.6 画像の傾きの計算

本節では，2.4.3 項で定義した画像の傾き φ を，3.5.1 項で定義した同次変換行列 ${}^{\mathrm{r}}T_{\mathrm{c}}$ を使用して求める方法を述べる．以下，座標・ベクトルはすべて Σ_{robot} で表す．

Σ_{cam} の x,y 軸の方向ベクトルをそれぞれ $\boldsymbol{e}_{\mathrm{x}}, \boldsymbol{e}_{\mathrm{y}}$ とすると，$(\boldsymbol{e}_{\mathrm{x}}{}^t|0)^t = {}^{\mathrm{r}}T_{\mathrm{c}}(1,0,0,0)^t$，$(\boldsymbol{e}_{\mathrm{y}}{}^t|0)^t = {}^{\mathrm{r}}T_{\mathrm{c}}(0,1,0,0)^t$ となり，$\boldsymbol{e}_{\mathrm{x}} = (t_{11}, t_{21}, t_{31})^t, \boldsymbol{e}_{\mathrm{x}} = (t_{12}, t_{22}, t_{32})^t$ となる．ここで，Σ_{cam} の yz 平面上にあり，フィールドに平行な単位ベクトル $\boldsymbol{e}_{\mathrm{h}} = (a_{\mathrm{x}}, a_{\mathrm{y}}, 0)^t (a_{\mathrm{x}} \leq 0$ とする) を考えると（Fig.3.9），$\boldsymbol{e}_{\mathrm{x}} \cdot \boldsymbol{e}_{\mathrm{h}} = 0, |\boldsymbol{e}_{\mathrm{h}}| = 1$ なので，

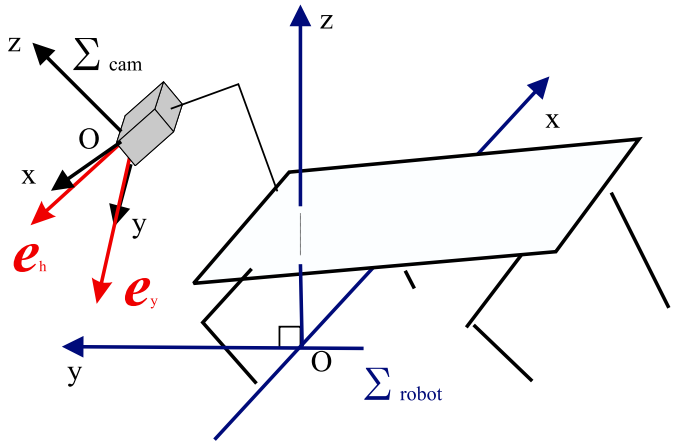

Fig. 3.9 $\boldsymbol{e}_{\mathrm{h}}, \boldsymbol{e}_{\mathrm{y}}$

$$t_{11}a_{\mathrm{x}} + t_{21}a_{\mathrm{y}} = 0 \tag{3.6.1}$$

$$a_{\mathrm{x}}^2 + a_{\mathrm{y}}^2 = 1 \tag{3.6.2}$$

となり，これらを解くと，

$$(a_{\mathrm{x}}, a_{\mathrm{y}}) = \pm\frac{1}{\sqrt{t_{11}^2 + t_{21}^2}}(-t_{21}, t_{11}) \tag{3.6.3}$$

となる．ただし，CCD カメラを Σ_{robot} の y 軸に対して逆方向に向けないときは $t_{21} \geq 0$ となるので，$a_{\mathrm{x}} \leq 0$ より，

$$\boldsymbol{e}_{\mathrm{h}} = \frac{1}{\sqrt{t_{11}^2 + t_{21}^2}}(-t_{21}, t_{11}, 0) \tag{3.6.4}$$

とできる．φ は $\boldsymbol{e}_{\mathrm{y}}$ と $\boldsymbol{e}_{\mathrm{h}}$ のなす角なので，

$$\begin{aligned}\cos\varphi &= \frac{\boldsymbol{e}_{\mathrm{y}}\cdot\boldsymbol{e}_{\mathrm{h}}}{|\boldsymbol{e}_{\mathrm{y}}||\boldsymbol{e}_{\mathrm{h}}|}\\ &= \frac{t_{11}t_{22}-t_{21}t_{12}}{\sqrt{t_{11}^2+t_{21}^2}}\end{aligned} \tag{3.6.5}$$

ゆえに，

$$\varphi = \pm\arccos\frac{t_{11}t_{22}-t_{21}t_{12}}{\sqrt{t_{11}^2+t_{21}^2}} \tag{3.6.6}$$

となる．符号は$\boldsymbol{e}_{\mathrm{y}}$がフィールドに対して上向きのとき正，下向きのとき負なので，φ の式は，

$$\varphi = \begin{cases} -\arccos\dfrac{t_{11}t_{22}-t_{21}t_{12}}{\sqrt{t_{11}^2+t_{21}^2}} & (t_{32}\geq 0)\\ \arccos\dfrac{t_{11}t_{22}-t_{21}t_{12}}{\sqrt{t_{11}^2+t_{21}^2}} & (t_{32}<0)\end{cases} \tag{3.6.7}$$

で表される．

3.7 おわりに

本章では，逆透視変換を用い，画像上の点（だだし，第 2 章で求めたようなフィールドと同じ高さの点）を画像上からロボット座標系に変換する手法について述べた．また，本章で求めた同次変換行列によって，2.4.3 項で必要とするカメラ画像の傾きの計算式を求めた．

3.2 節において，カメラ座標系，ロボット座標系について設定した．

3.3 節において，画像座標系からカメラ座標系への式を求めた．

3.4 節において，カメラ座標系からロボット座標系への同時変換行列を求めた．脚部の関節角から姿勢が判断できないという問題は，あらかじめ歩様ごとに脚と胴体の接合部分の高さを計測しておき，その値を用いることで解決した．

3.5 節において，3.3，3.4 節の変換式を用いた座標変換アルゴリズムを説明した．

3.6 節において，同次変換行列を利用して，カメラ画像の傾きの計算式を求めた．

第4章　自己位置同定

4.1 はじめに

本章では，第 2，3 章で設計した壁・ライン位置計測アルゴリズムを自己位置同定に利用する手法を述べる．

4.2 節において，筆者の所属するチームで採用されている自己位置同定手法について説明する．

4.3 節において，自己位置同定のために必要な，壁・ライン位置計測の結果からロボットの存在確率分布を求める計算方法について述べる．

4.4 節において，本章を総括する．

4.2 Sensor Resetting Localization

本節では，筆者の所属するチームの，“ Sensor Resetting Localization ”（SRL）[Lenser 2000] を用いた自己位置同定手法について説明する．

4.2.1 アルゴリズム

SRL は，存在確率分布を多数の点（サンプル）で表現する方法である．サンプルは，フィールド座標系でのロボットの位置・姿勢 $\nu_i = (x_i, y_i, \theta_i)$ と，重み（確率）p_i をパラメータに持つ．サンプルが n 個あり，重みの合計が 1 であるとして次の処理を行う（Fig.4.1）．

デッドレコニング　ロボットが歩いた歩数によってサンプルの (x_i, y_i, θ_i) を変化させる．歩行したときの距離・角度のばらつきのデータをもとに，サンプル一つ一つに乱数を使って雑音を乗せる．

センサ入力　センサから情報（情報 s とする）が得られたとき，p_i を次のように変化させる．

$$p_i \leftarrow p_i P(s|\nu_i) \tag{4.2.1}$$

$$p_i \leftarrow p_i / \sum_{k=1}^{n} p_k \quad \text{（正規化）} \tag{4.2.2}$$

ただし，$P(s|\nu)$ は，位置・姿勢が ν のときに情報 s を得る確率（連続値の場合は確率密度）である．

置きなおし　重みがほとんど 0 になってしまったサンプルは消去し，なくなった分だけ重みの大きいサンプルを分割するなどして新たなサンプルを作る．

リセット　すべてのサンプルの存在確率が 0 になった場合は，新たなセンサ情報にしたがって，全てのサンプルを置きなおす．

4.2.2 利点

SRL は，[Wendler 2000],[Veloso 1999],[Buschka 2000] などの区画を区切る方法に比べて以下のような利点がある．

i. 精度の限界がない．

ii. ロボットが存在しそうな ν の範囲で集中的に計算するので計算量が少ない．

iii. ロボットが存在しそうな ν の範囲外でメモリを浪費しない．

iv. デッドレコニングが扱いやすい．

また，カルマンフィルターのように $P(s|\nu)$ がガウス分布に従う必要はなく，さまざまな分布が扱える．

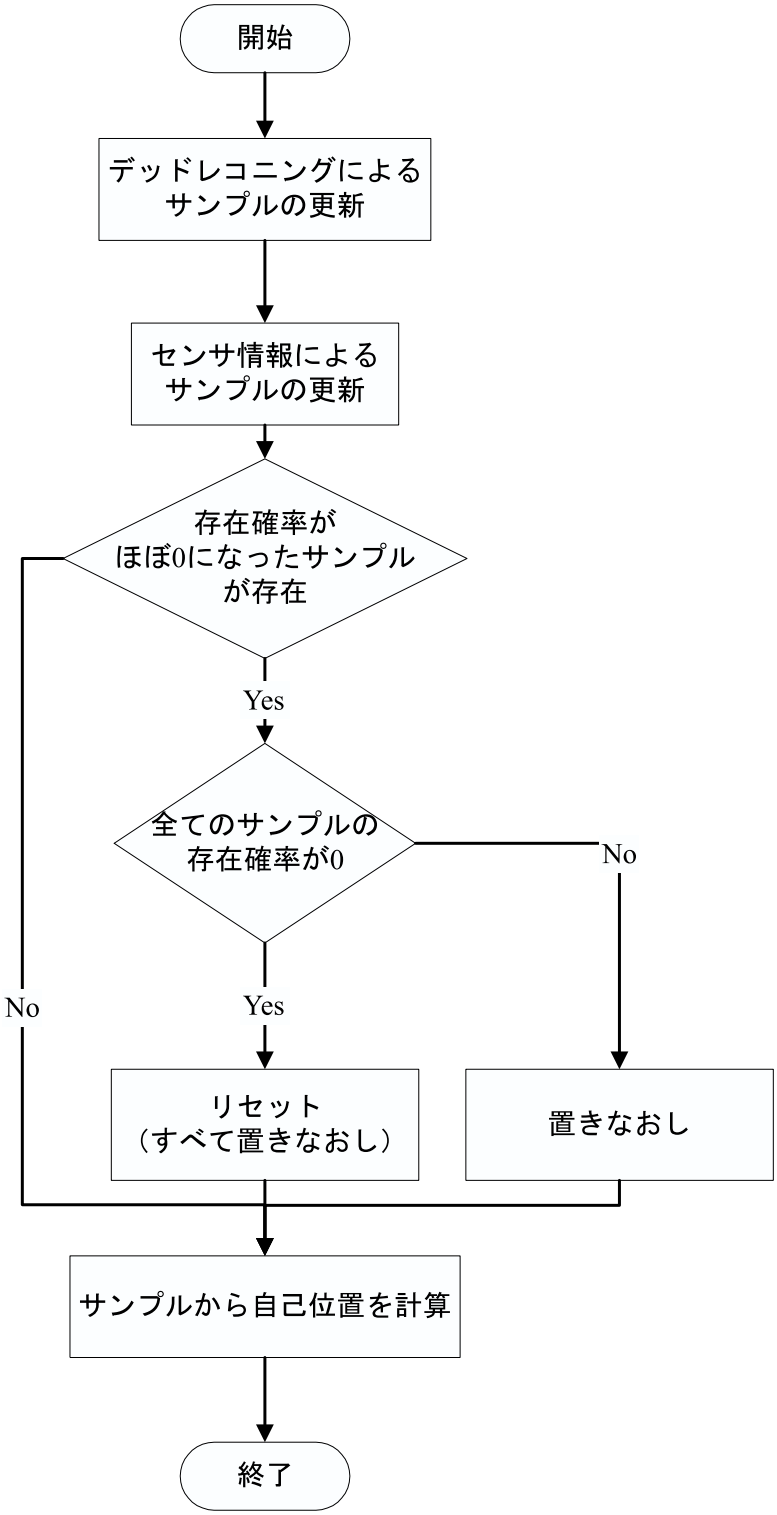

Fig. 4.1 Sensor Resetting Localization

4.3 壁・ライン計測後の存在確率分布

第 2，3 章で設計した壁・ライン位置計測アルゴリズムを SRL に組み込むには，$P(s|\nu)$ を計算するアルゴリズムが必要になる．そこで，壁・ライン位置計測アルゴリズムの出力を $P(s|\nu)$ に変換する方法を説明する．

4.3.1 諸定義

壁・ラインの距離，角度　観測したラインについて Fig.4.2 のように距離を r，角度を ζ とする．ただし，$\lambda = 0$ のときは $r > 0$ とみなして ζ を決定する．

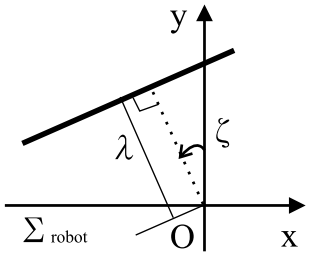

Fig. 4.2　d と ζ の定義

フィールド座標系の定義　フィールド座標系 Σ_{field} を Fig.4.3 のように定義する（原点はフィールドの中心）．また，ロボットの Σ_{field} 上での姿勢 θ は，図のように Σ_{field} の x 軸と Σ_{robot} の y 軸のなす角とし，定義域を-180[deg] から 180[deg] とする．

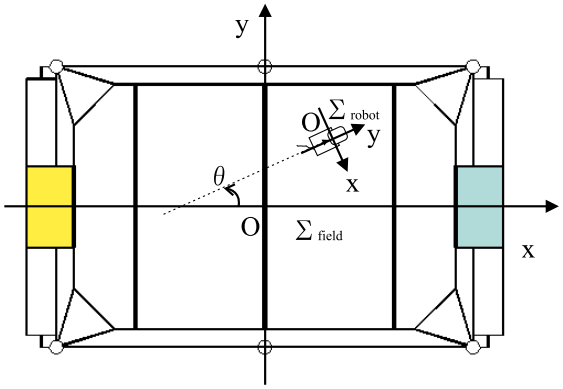

Fig. 4.3　フィールド座標系の定義

4.3.2 ロボットからのラインの見え方

Σ_{field} 上の直線 $\ell_j : y\cos\kappa_j - x\sin\kappa_j = \rho_j \quad (j = 1, 2, \ldots, 11)$[1)]を考え（Fig.4.4），この直線がサンプル $\nu_i = (x_i, y_i, \theta_i)$（前節参照）と同じ位置・姿勢であると仮定したロボットからどのように観測されるか計算する．

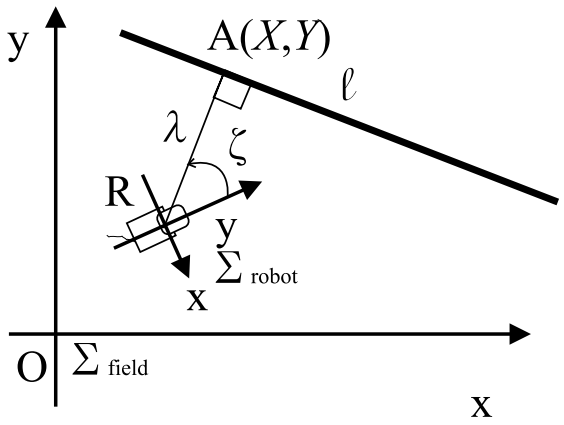

Fig. 4.4 直線 ℓ_j

ℓ_j にロボットの位置（点 $\mathrm{R}(x_i, y_i)$ とする）から下ろした垂線の足を点 $\mathrm{A}(X, Y)$ とする．このとき $\overrightarrow{\mathrm{AR}} \perp \ell_j$，$\ell_j$ 上に A が存在することから，

$$(X - x_i, Y - y_i) \cdot (\cos\kappa_j, \sin\kappa_j) = 0 \tag{4.3.1}$$

$$Y\cos\kappa_j - X\sin\kappa_j = \rho_j \tag{4.3.2}$$

となる．これらを解くと，

$$X = -\rho_j \sin\kappa_j + (x_i\cos\kappa_j + y_i\sin\kappa_j)\cos\kappa_j \tag{4.3.3}$$

$$Y = \rho_j \cos\kappa_j + (x_i\cos\kappa_j + y_i\sin\kappa_j)\sin\kappa_j \tag{4.3.4}$$

となる．Σ_{field} から Σ_{robot} への同次変換行列 ${}^{\text{robot}}T_{\text{field}}$ は（x,y 座標のみを考えると），

$${}^{\text{robot}}T_{\text{field}} = \begin{pmatrix} \sin\theta_i & -\cos\theta_i & -x_i\sin\theta_i + y_i\cos\theta_i \\ \cos\theta_i & \sin\theta_i & -x_i\cos\theta_i - y_i\sin\theta_i \\ 0 & 0 & 1 \end{pmatrix} \tag{4.3.5}$$

[1)] Table4.1,4.2 参照

と表せる．したがって，A を Σ_{robot} で表した点を $(X_{\mathrm{r}}, Y_{\mathrm{r}})$ とすると，

$$\begin{pmatrix} X_{\mathrm{r}} \\ Y_{\mathrm{r}} \\ 1 \end{pmatrix} = \begin{pmatrix} \sin\theta_i & -\cos\theta_i & -x_i\sin\theta_i + y_i\cos\theta_i \\ \cos\theta_i & \sin\theta_i & -x_i\cos\theta_i - y_i\sin\theta_i \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} \tag{4.3.6}$$

式 (4.3.6) に式 (4.3.3),(4.3.4) を代入すると，

$$X_{\mathrm{r}} = -\rho_j\cos(\theta_i - \kappa_j) + (x_i\cos\kappa_j + y_i\sin\kappa_j)\sin(\theta_i - \kappa_j) - x_i\sin\theta_i + y_i\cos\theta_i \tag{4.3.7}$$

$$Y_{\mathrm{r}} = \rho_j\sin(\theta_i - \kappa_j) + (x_i\cos\kappa_j + y_i\sin\kappa_j)\cos(\theta_i - \kappa_j) - x_i\cos\theta_i - y_i\sin\theta_i \tag{4.3.8}$$

を得る．この $X_{\mathrm{r}}, Y_{\mathrm{r}}$ と，ロボットからの直線 ℓ_j の距離 λ_j，角度 ζ_j の関係は次のようになる[2)]．

$$\lambda_j = \sqrt{X_{\mathrm{r}}^2 + Y_{\mathrm{r}}^2} \tag{4.3.9}$$

$$\zeta_j = \mathrm{atan2}(-X_{\mathrm{r}}, Y_{\mathrm{r}}) \tag{4.3.10}$$

ただし，$\mathrm{atan2}(y, x)$ は C 標準ライブラリの関数で，次のように定義されている．

$$\mathrm{atan2}(y,x) = \begin{cases} \arctan(y/x) & (x > 0) \\ \arctan(y/x) + \pi & (x < 0, y \geq 0) \\ \arctan(y/x) - \pi & (x < 0, y < 0) \\ \pi/2 & (x = 0, y > 0) \\ -\pi/2 & (x = 0, y < 0) \\ \text{定義域エラー} & (x = 0, y = 0) \end{cases} \tag{4.3.11}$$

4.3.3 存在確率密度分布の計算

前項で導いた λ_j，ζ_j は理想値であり，実際の観測結果には誤差が乗っている．ロボットから距離 λ_j，角度 ζ_j の位置にある壁・ライン ℓ_j を観測したときに，計測結果が距離 $\acute{\lambda_j}$，角度 $\acute{\zeta_j}$ となる確率密度を $p_j(\acute{\lambda_j}, \acute{\zeta_j}|\lambda_j, \zeta_j)$ とする．$p_j(\acute{\lambda_j}, \acute{\zeta_j}|\lambda_j, \zeta_j)$ は，5.3.1 項の測定結果より次のように正規分布で表現した．

[2)]式 (4.3.10) は，実際には κ と θ_i の関数なので，$X_{\mathrm{r}} = Y_{\mathrm{r}} = 0$ のときは x_i, y_i に適当な値を入れて計算しなおせばよい．

$$\sigma_\lambda = 0.0002\ \lambda_j^2 - 0.0869\ \lambda_j + 19.695 \tag{4.3.12}$$

$$\sigma_\zeta = 0.0000007\ \lambda_j^2 + 0.0007\ \lambda_j + 3.3993 \tag{4.3.13}$$

$$p_j(\acute{\lambda_j}, \acute{\zeta_j}|\lambda_j, \zeta_j) = \frac{1}{2\pi\sigma_\lambda\sigma_\zeta} \exp\left\{ -\frac{(\acute{\lambda} - \lambda_j)^2}{2\sigma_\lambda^2} - \frac{(\acute{\zeta} - \zeta_j)^2}{2\sigma_\zeta^2} \right\} \tag{4.3.14}$$

壁・ラインがフィールド上に m 本存在すると，ロボットの位置・姿勢が ν_i のとき，観測結果が $\acute{\lambda}$，$\acute{\zeta}$ となる確率密度 $P(\acute{\lambda}, \acute{\zeta}|\nu_i)$ は次のように定まる．

$$P(\acute{\lambda}, \acute{\zeta}|\nu_i) = \frac{1}{m} \sum_{k=1}^{m} p_k(\acute{\lambda}, \acute{\zeta}|\lambda_k, \zeta_k) \tag{4.3.15}$$

フィールド上の壁・ラインは，Fig.4.5 のよう 11 本存在し，それぞれの ρ_j, κ_j は Table4.1，4.2 のようになっている．

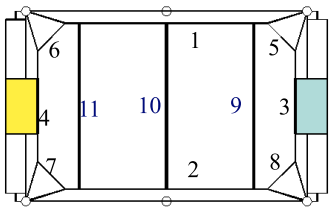

Fig. 4.5 壁・ラインの位置

Table 4.1 壁の位置

j	名称	ρ_j[mm]	κ_j[deg]
1	タッチライン 1	900	0
2	タッチライン 2	-900	0
3	ゴールライン 1	1400	90
4	ゴールライン 2	-1400	90
5	コーナー 1	$1000\sqrt{2}$	-45
6	コーナー 2	$1000\sqrt{2}$	45
7	コーナー 3	$-1000\sqrt{2}$	-45
8	コーナー 4	$-1000\sqrt{2}$	45

Table 4.2 ラインの位置

j	名称	ρ_j[mm]	κ_j[deg]
9	ペナルティーライン 1	950	90
10	ペナルティーライン 2	-950	90
11	ハーフウェーライン	0	90

フィールド上での $P(s|\nu)$ の計算は，2.3.5 項で付加した壁・ラインの判断を使用して次のように行う．

壁のとき

$$P(\acute{\lambda}, \acute{\zeta}|\nu_i) = \frac{1}{8}\sum_{k=1}^{8} p_k(\acute{\lambda}, \acute{\zeta}|\lambda_k, \zeta_k) \tag{4.3.16}$$

ラインのとき

$$P(\acute{\lambda}, \acute{\zeta}|\nu_i) = \frac{1}{3}\sum_{k=9}^{11} p_k(\acute{\lambda}, \acute{\zeta}|\lambda_k, \zeta_k) \tag{4.3.17}$$

“ unknown ”のとき

$$P(\acute{\lambda}, \acute{\zeta}|\nu_i) = \frac{1}{11}\sum_{k=1}^{11} p_k(\acute{\lambda}, \acute{\zeta}|\lambda_k, \zeta_k) \tag{4.3.18}$$

4.4 おわりに

本章では，第 2，3 章で設計した壁・ライン位置計測アルゴリズムを自己位置同定に利用する手法を述べた．

4.2 節において，筆者の所属するチームで採用されている自己位置同定手法“Sensor Resetting Localization ”について，他の自己位置同定手法と比較し，説明した．

4.3 節において，自己位置同定のために必要な，壁・ライン位置計測の結果からロボットの存在確率分布を求める計算方法について述べた．存在確率分布は，正規分布の重ね合わせで表現した．

第5章　アルゴリズムの評価

5.1 はじめに

本章では，第 2，3，4 章で設計したアルゴリズムについて評価を行う．

5.2 節において，第 2，3 章で設計した「障害物位置計測アルゴリズム」の理論上の誤差について述べる．

5.3 節において，障害物位置計測アルゴリズムを実機で実行した場合の誤差，計算時間，雑音の影響について述べる．

5.4 節において，壁・ライン位置計測アルゴリズムを自己位置同定に利用した際の有効性についてシミュレーションを用いて検証する．

5.5 節において，本章を総括する．

5.2 障害物位置計測アルゴリズムの理論上の誤差

本節では，第 2，3 章で設計した障害物位置計測アルゴリズムの理論上の誤差について述べる．障害物位置計測アルゴリズムで考えられる誤差の原因を列挙すると次のようになる．

i. 画像上の画素の抽出位置のずれ.

ii. 画像の量子化誤差（ロボット位置計測の場合）.

iii. 計算で使用する脚の高さ h_1, h_2, h_3 と実際の脚の高さのずれ.

iv. CCD カメラのレンズの歪み.

v. 頭部関節角 t, p, r の計測誤差.

レンズの歪みはほとんどなく，頭部関節のエンコーダは精度が良いので i，ii，iii について扱うこととする．また，本研究では「近距離のものを精度良く位置計測する」ことを目標としているので，必要な精度は 500[mm] 以内では計測対象の距離の 10% 程度，1m 以上の距離での精度は遠いと分かる程度（40% 程度）とする．

5.2.1 画像抽出の際の誤差

ここでは i，ii による誤差について扱う．抽出した画素が実際の位置から縦に m 画素，横に n 画素ずれた場合のロボット座標系での誤差を $(\Delta x_{\mathrm{r1}}, \Delta y_{\mathrm{r1}}, 0)$ とすると，式 (3.5.5)，(3.5.6) から,

$$\Delta x_{\mathrm{r1}} = t_{34}\left(-\frac{t_{11}+t_{12}\dfrac{x_{\mathrm{m}}/2-x_{\mathrm{i}}-m}{x_{\mathrm{m}}}\tan h_{\mathrm{m}}+t_{13}\dfrac{y_{\mathrm{m}}/2-y_{\mathrm{i}}-n}{y_{\mathrm{m}}}\tan v_{\mathrm{m}}}{t_{31}+t_{32}\dfrac{x_{\mathrm{m}}/2-x_{\mathrm{i}}-m}{x_{\mathrm{m}}}\tan h_{\mathrm{m}}+t_{33}\dfrac{y_{\mathrm{m}}/2-y_{\mathrm{i}}-n}{y_{\mathrm{m}}}\tan v_{\mathrm{m}}}\right.$$
$$\left.+\frac{t_{11}+t_{12}\dfrac{x_{\mathrm{m}}/2-x_{\mathrm{i}}}{x_{\mathrm{m}}}\tan h_{\mathrm{m}}+t_{13}\dfrac{y_{\mathrm{m}}/2-y_{\mathrm{i}}}{y_{\mathrm{m}}}\tan v_{\mathrm{m}}}{t_{31}+t_{32}\dfrac{x_{\mathrm{m}}/2-x_{\mathrm{i}}}{x_{\mathrm{m}}}\tan h_{\mathrm{m}}+t_{33}\dfrac{y_{\mathrm{m}}/2-y_{\mathrm{i}}}{y_{\mathrm{m}}}\tan v_{\mathrm{m}}}\right) \tag{5.2.1}$$

$$\Delta y_{\mathrm{r1}} = t_{34}\left(-\frac{t_{21}+t_{22}\dfrac{x_{\mathrm{m}}/2-x_{\mathrm{i}}-m}{x_{\mathrm{m}}}\tan h_{\mathrm{m}}+t_{23}\dfrac{y_{\mathrm{m}}/2-y_{\mathrm{i}}-n}{y_{\mathrm{m}}}\tan v_{\mathrm{m}}}{t_{31}+t_{32}\dfrac{x_{\mathrm{m}}/2-x_{\mathrm{i}}-m}{x_{\mathrm{m}}}\tan h_{\mathrm{m}}+t_{33}\dfrac{y_{\mathrm{m}}/2-y_{\mathrm{i}}-n}{y_{\mathrm{m}}}\tan v_{\mathrm{m}}}\right.$$
$$\left.+\frac{t_{21}+t_{22}\dfrac{x_{\mathrm{m}}/2-x_{\mathrm{i}}}{x_{\mathrm{m}}}\tan h_{\mathrm{m}}+t_{23}\dfrac{y_{\mathrm{m}}/2-y_{\mathrm{i}}}{y_{\mathrm{m}}}\tan v_{\mathrm{m}}}{t_{31}+t_{32}\dfrac{x_{\mathrm{m}}/2-x_{\mathrm{i}}}{x_{\mathrm{m}}}\tan h_{\mathrm{m}}+t_{33}\dfrac{y_{\mathrm{m}}/2-y_{\mathrm{i}}}{y_{\mathrm{m}}}\tan v_{\mathrm{m}}}\right) \tag{5.2.2}$$

となる．特に，画像の中心に映った点を，縦に m 画素，横に n 画素間違えて抽出したときの誤差は，

$$\Delta x_{\mathrm{r1}} = t_{34}\left(-\frac{t_{11}-t_{12}\dfrac{m}{x_{\mathrm{m}}}\tan h_{\mathrm{m}}-t_{13}\dfrac{n}{y_{\mathrm{m}}}\tan v_{\mathrm{m}}}{t_{31}-t_{32}\dfrac{m}{x_{\mathrm{m}}}\tan h_{\mathrm{m}}-t_{33}\dfrac{n}{y_{\mathrm{m}}}\tan v_{\mathrm{m}}}+\frac{t_{11}}{t_{31}}\right) \tag{5.2.3}$$

$$\Delta y_{\mathrm{r1}} = t_{34}\left(-\frac{t_{21}-t_{22}\dfrac{m}{x_{\mathrm{m}}}\tan h_{\mathrm{m}}-t_{23}\dfrac{n}{y_{\mathrm{m}}}\tan v_{\mathrm{m}}}{t_{31}-t_{32}\dfrac{m}{x_{\mathrm{m}}}\tan h_{\mathrm{m}}-t_{33}\dfrac{n}{y_{\mathrm{m}}}\tan v_{\mathrm{m}}}+\frac{t_{21}}{t_{31}}\right) \tag{5.2.4}$$

となる．

$m=0, n=1$ を式 (5.2.2) に代入し，同次変換行列 ${}^{\mathrm{r}}T_{\mathrm{c}}$ に，$p=0$[deg]，$r=0$[deg][1)]と直進時の 1 姿勢 $(h_1,h_2,h_3)=(80,75,125)$[mm] を代入して t を変化させたときの $|\Delta y_{\mathrm{r}}|$ は，Fig.5.1 のようになる．Fig.5.1 では，およそ 1500[mm] まで誤差が距離の 10% 以内に収まっていることが分かる．これは，目標とする値を十分に満たしている．

5.2.2 胴体の傾きによる誤差

ここではiiiによる誤差について扱う．同次変換行列作成の際は，脚の高さ h_1,h_2,h_3 を Table3.1 を用いてあらかじめ計測した値を使用するので，歩行中，静止中に関わらず実際の姿勢がその値より数 mm ずれることがある．このときの誤差を $(\Delta x_{\mathrm{r2}},\Delta y_{\mathrm{r2}},0)$，Table3.1 を使用して計算する ${}^{\mathrm{r}}T_{\mathrm{c}}$ を $\bar{T}$，実際のカメラ座標系とロ

[1)] CCD カメラが前を向いている状態.

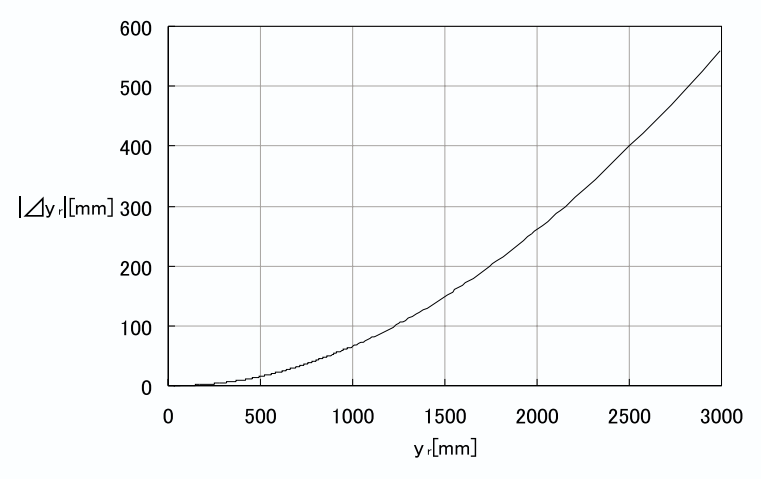

Fig. 5.1 1画素のずれによる誤差

ボット座標系の関係を表す ${}^{\mathrm{r}}T_{\mathrm{c}}$ を $\hat{T}$，$\boldsymbol{a} = \left(1, \dfrac{x_{\mathrm{m}}/2 - x_{\mathrm{i}}}{x_{\mathrm{m}}} \tan h_{\mathrm{m}}, \dfrac{y_{\mathrm{m}}/2 - y_{\mathrm{i}}}{y_{\mathrm{m}}} \tan v_{\mathrm{m}}\right)^t$ とすると，

$$\Delta x_{\mathrm{r2}} = \left(\bar{t}_{14} - \frac{(\bar{t}_{11}, \bar{t}_{12}, \bar{t}_{13})\boldsymbol{a}}{(\bar{t}_{31}, \bar{t}_{32}, \bar{t}_{33})\boldsymbol{a}} \bar{t}_{34}\right) - \left(\grave{t}_{14} - \frac{(\grave{t}_{11}, \grave{t}_{12}, \grave{t}_{13})\boldsymbol{a}}{(\grave{t}_{31}, \grave{t}_{32}, \grave{t}_{33})\boldsymbol{a}} \grave{t}_{34}\right) \tag{5.2.5}$$

$$\Delta y_{\mathrm{r2}} = \left(\bar{t}_{24} - \frac{(\bar{t}_{21}, \bar{t}_{22}, \bar{t}_{23})\boldsymbol{a}}{(\bar{t}_{31}, \bar{t}_{32}, \bar{t}_{33})\boldsymbol{a}} \bar{t}_{34}\right) - \left(\grave{t}_{24} - \frac{(\grave{t}_{21}, \grave{t}_{22}, \grave{t}_{23})\boldsymbol{a}}{(\grave{t}_{31}, \grave{t}_{32}, \grave{t}_{33})\boldsymbol{a}} \grave{t}_{34}\right) \tag{5.2.6}$$

となる．特に，画像の中心に映ったものについては，$\boldsymbol{a} = (1, 0, 0)^t$ なので，

$$\Delta x_{\mathrm{r2}} = \left(\bar{t}_{14} - \frac{\bar{t}_{11}}{\bar{t}_{31}} \bar{t}_{34}\right) - \left(\grave{t}_{14} - \frac{\grave{t}_{11}}{\grave{t}_{31}} \grave{t}_{34}\right) \tag{5.2.7}$$

$$\Delta y_{\mathrm{r2}} = \left(\bar{t}_{24} - \frac{\bar{t}_{21}}{\bar{t}_{31}} \bar{t}_{34}\right) - \left(\grave{t}_{24} - \frac{\grave{t}_{21}}{\grave{t}_{31}} \grave{t}_{34}\right) \tag{5.2.8}$$

となる．当チームの歩行は再現性が高く，測定した結果では (h_1, h_2, h_3) の誤差は各脚 2,3mm 程度である．$(h_1, h_2, h_3) = (80, 75, 125)[\mathrm{mm}]$ から，右前脚が $\Delta h[\mathrm{mm}]$ 下がり，左後脚が $\Delta h[\mathrm{mm}]$ 上がった状態[2)]で t を変化させたときの画像の中心に映った物体の $|\Delta y_{\mathrm{r}}|$ は，Fig.5.2 のようになる．

[2)] ロボットは４本ある脚のうち３本で自重を支えることが多いが，そのうちの１本が，首の向きが変わって重心が変化したときに入れ替わることがある．このときにこのような高さのずれが起こる．

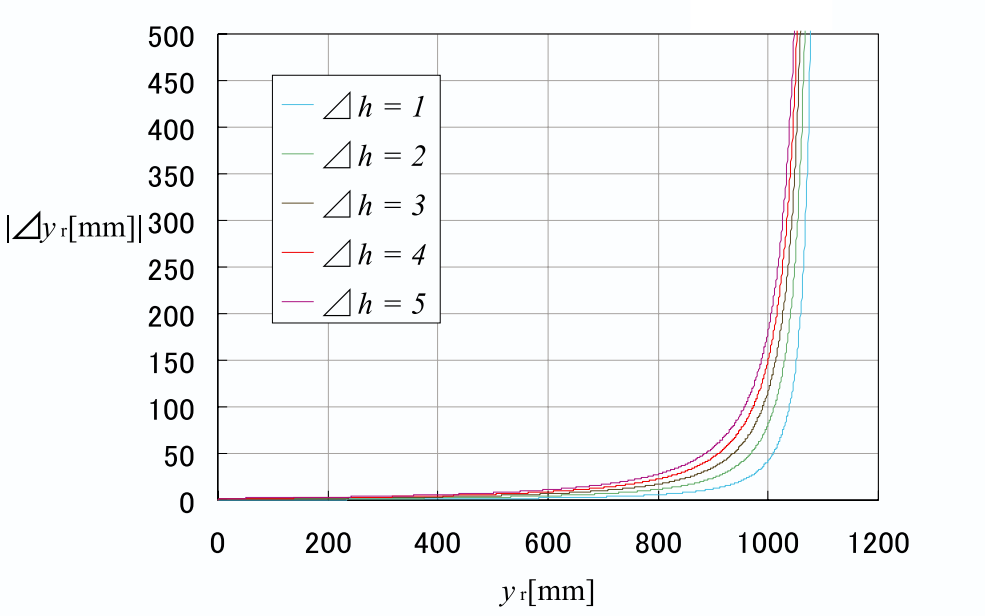

Fig. 5.2 右前下がりにずれたときの誤差

また，前両脚が Δh[mm] 下がったとき[3)]の $|\Delta y_r|$ は，Fig.5.3 のようになる．これらのグラフから，胴体の傾きによる誤差は近距離ではほとんど無視できるが，1[m] を越えると急激に大きくなることが分かる．

Fig.5.1，5.2 から，理論上，500[mm] 程度の近距離では目標よりも高い精度が期待できると言える．ただし，目標の精度が得られるのは 1 [m] 以内であると分かった．

[3)] これは，後から押された時や，首の向きが変わったときに起こる．

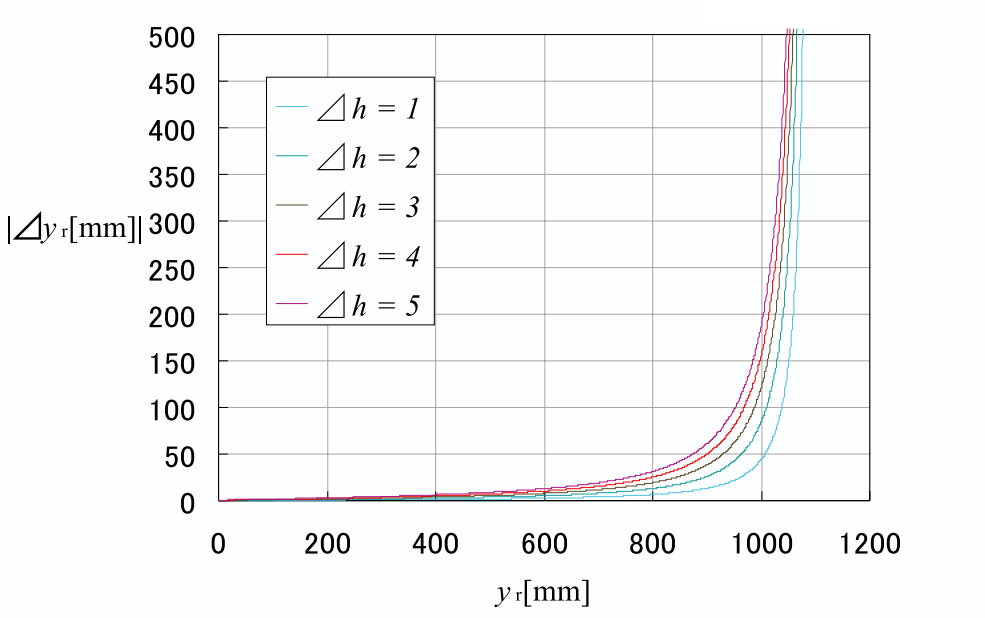

Fig. 5.3 胴体が前下がりにずれたときの誤差

5.3 障害物位置計測の実機実験

本節では，実機における障害物位置計測アルゴリズムの実機による計測のばらつき・誤差，計算処理時間，雑音の影響について述べる．

5.3.1 壁・ライン位置計測アルゴリズムの誤差

壁・ライン位置計測アルゴリズムを実装し，ロボットで壁の距離，角度を計測した．

実験１ ばらつきの分布を調べるため，$\lambda = 400\text{mm}$，$\zeta = -30[\text{deg}]$（4.2 節参照）で，t を 5[deg] 刻み，p を 5[deg] 刻みで変化させ，ペナルティーラインを観測した．結果を度数分布にしたものが Fig.5.4，5.5 である．また，計測結果の平均値・標準偏差は次のようになった．

- 距離：平均値が 366[mm]，標準偏差が 14.0[mm].
- 角度：平均値が-27[deg]，標準偏差が 3.5[deg].

距離・角度ともに正規分布に近いばらつき方であった．

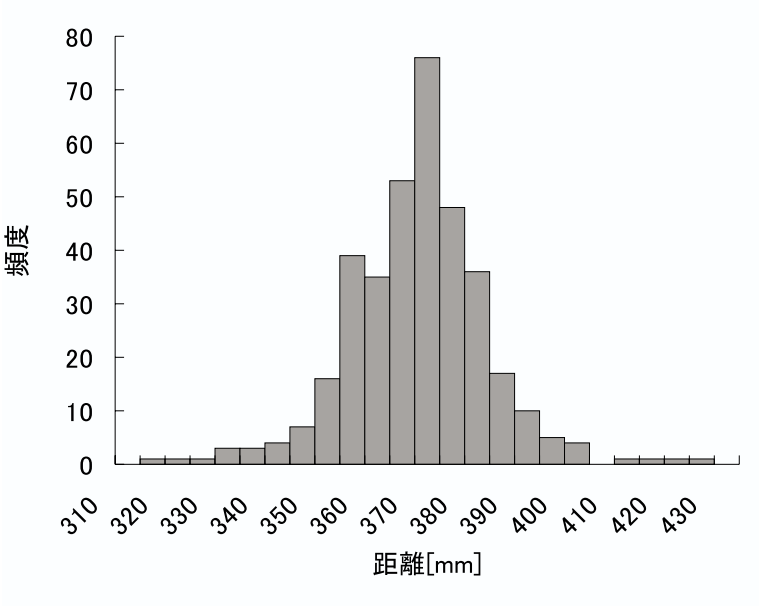

Fig. 5.4 測定結果の度数分布（距離）

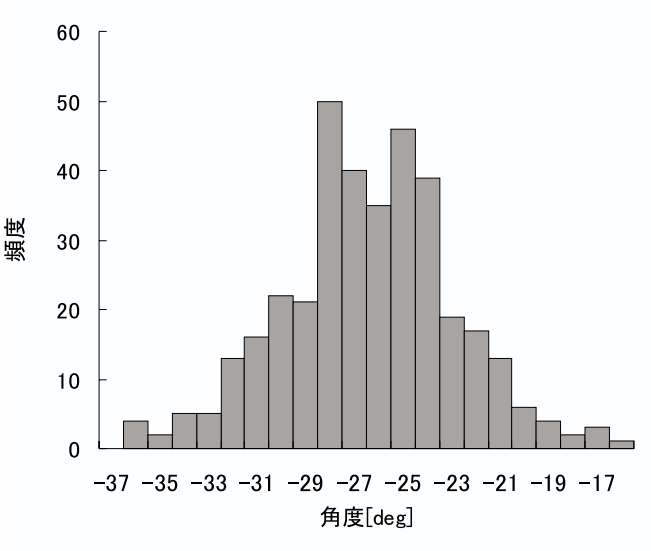

Fig. 5.5 測定結果の度数分布（角度）

実験 2　距離と計測結果のばらつきの関係を調べるため，以下の条件でタッチライン（壁）を計測した．

- ERS-1100 をタッチラインに垂直（$\zeta = 0$）に置く．
- t を 10[deg] 刻み，p を 10[deg] 刻みで変化させる．
- 他のラインを抽出することを防ぐため，他のラインが視界に入らないようにする[4]．

距離を変えてラインを計測した結果の距離・角度の平均値・標準偏差を Fig.5.6，5.7 に示す[5]．

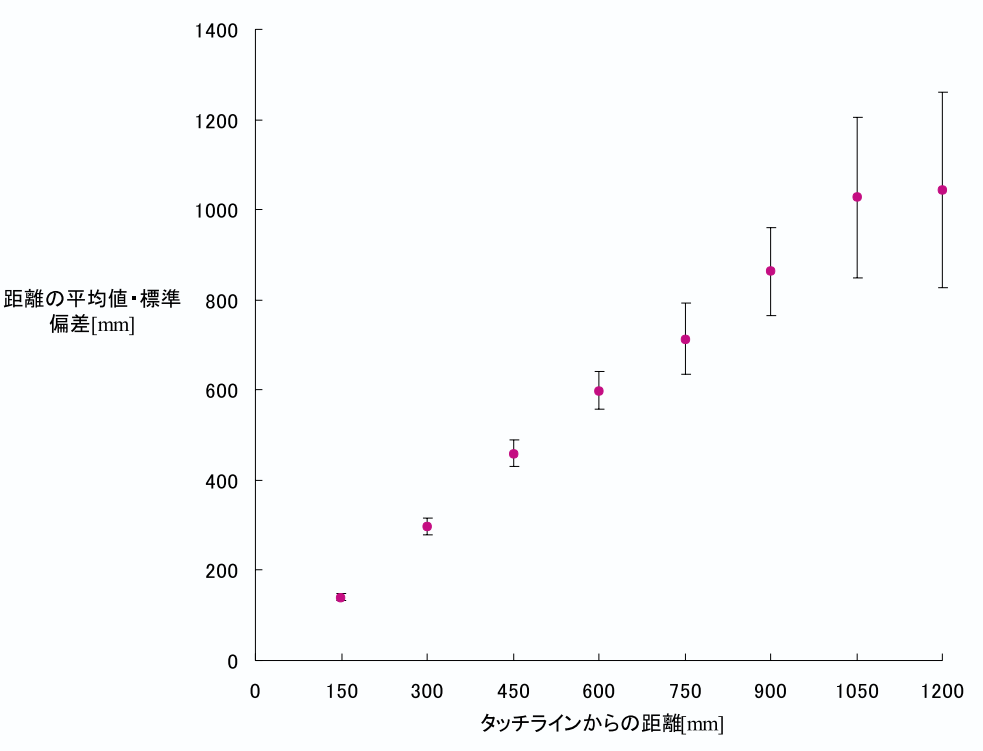

Fig.　5.6　タッチラインの計測距離の平均値と標準偏差

[4]ペナルティーライン・ハーフウェイラインをフィールドから剥がした．
[5]式 (4.3.12)，(4.3.13) は，Fig.5.6，5.7 の標準偏差から求めた回帰曲線の式である．

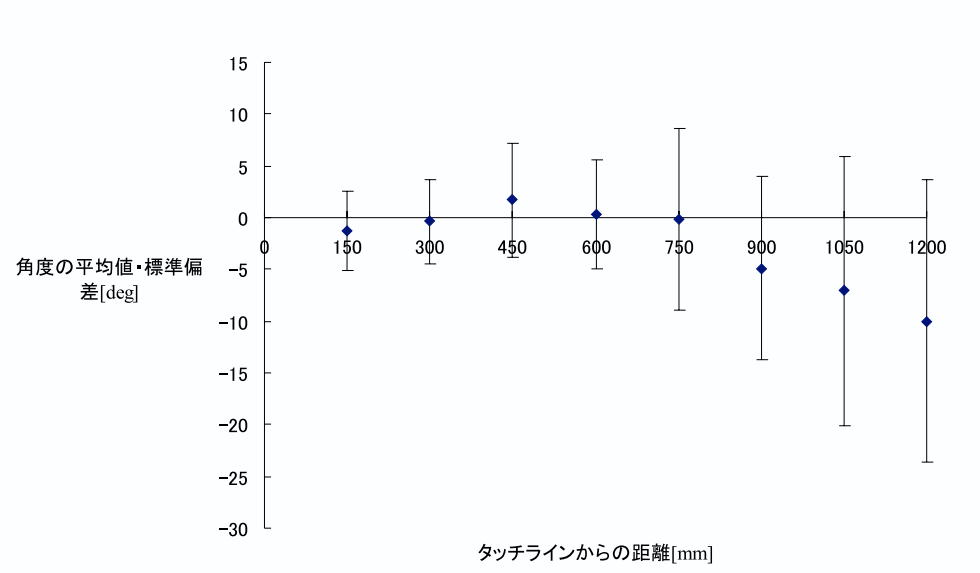

Fig. 5.7 タッチラインとの計測角度の平均値と標準偏差

これらの図からは理論値と同様に，１ [m] 以内では，真の値からの平均値のずれ，標準偏差はともに大きくなく，目標をほぼ満たしている[6]．また，胴体の傾きの影響で１ [m] 辺りで計測の平均値が真の値から大きく外れる様子が見られる．

5.3.2 ロボット位置計測アルゴリズムの誤差

ロボット位置計測アルゴリズムを実装し，ロボットの位置を計測した．Fig.5.8 のようにロボットを置き（姿勢は Fig.1.1 と同じ.），t を 10[deg] 刻み，p を 10[deg] 刻み，距離 d を 100[mm] 刻みで変化させて計測した．計測の平均値と標準偏差を Fig.5.9 に示す．

Fig.5.9 では，500[mm] の時が最も精度良く計測でき，それより近くても遠くても真の値からの平均値のずれ，標準偏差が大きくなる．この理由は，以下のように考えられる．

- 500[mm] より遠い距離：画素の量子化誤差や胴体のずれによる影響が大きく

[6]距離というのは壁とロボットの最短距離であり，実際に CCD カメラ画像に入る地点はさらに遠い地点であることを考えると，理論値の計算で使用した「１画素のずれ」は，壁・ライン位置計測アルゴリズムでは実際にはあまり起こらないと思われる．

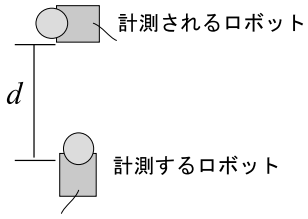

Fig. 5.8 計測するロボットの位置

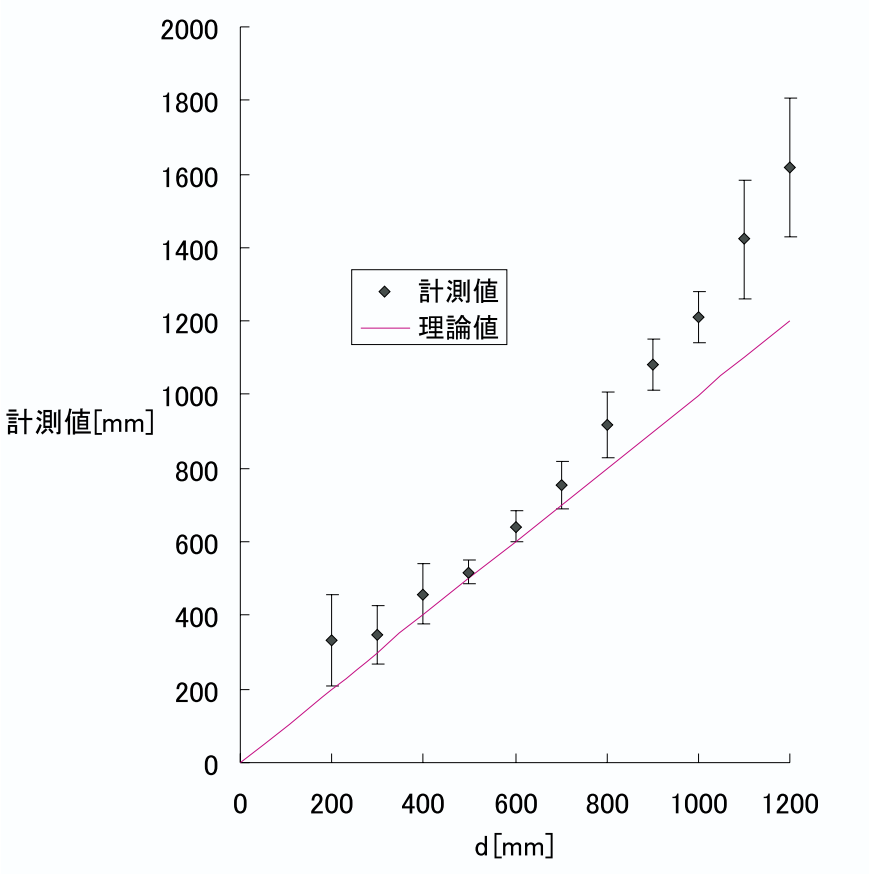

Fig. 5.9 ロボットの距離計測結果

なる.

- 500[mm] より近い距離：ロボットが画像に大きく映り，接地部分を抽出せずに胴体の下端を抽出する頻度が多くなる.

5.3.3 計算時間

障害物位置計測アルゴリズムは，障害物回避に使用するためには１歩につき最低１回実行しなければならない．現在の当チームの直進は１秒間におよそ２歩であるので，最低でも壁・ライン位置計測とロボット位置計測の両方を0.5秒以内で行う必要がある．

壁・ライン位置計測アルゴリズム全体にかかる時間を計ったところ，最大0.3[s]であった．そのうちt, p, r, h_1, h_2, h_3を入力して変換行列Tを作成するという処理の時間は1[ms]であったので，壁・ライン位置計測アルゴリズムでは，計算のほとんど全てを画像処理にかけていることになる．また，2.3.3項で，ハフ変換の際にC標準ライブラリの$\sin(), \cos()$は使用しないと述べたが，$\sin(), \cos()$を使用するアルゴリズムにして計算時間を計ったところ，アルゴリズム全体で平均約1[s]かかった．つまり，この工夫だけで３倍以上計算が速くなったことになる．また，ロボット位置計測アルゴリズム全体でかかる時間は，ロボットを認識しなかったときに0.15[s]，１台認識したときは0.21[s]，２台認識したときは0.24[s]であった．

したがって，壁・ライン位置計測とロボット位置計測のアルゴリズムを両方実行すると，0.5秒前後の時間がかかる．サッカーのためには他のアルゴリズムを実行する時間も必要であるので，現状では１歩につき壁・ライン位置計測かロボット位置計測のどちらか一方しか行えないことになる．しかし，どちらか一方であれば１歩ごとに実行しても問題はないと言える．

5.3.4 雑音の影響

視界に他機やボールが入ることは，壁・ライン位置計測に悪影響を与える．そこで，影響の度合いを調べるため以下のような実験を行った．

- フィールド座標系で$(x, y, \theta) = (-550, 0, 180)$の位置に自機を置く．
- 400[mm]先のペナルティーラインとの間に２台の他機を置く．２台の間には50[mm]の隙間を空け，隙間が自機の真正面に来るようにする．他機とペナルティーラインの間隔をd[mm]とする（Fig.5.10）．
- pを-60[deg]から60[deg]まで15[deg]刻み，tを-35[deg]から15[deg]まで10[deg]刻みで動かして観測（54回観測を行い，一回で２本分の結果が得られるので

108 本分のデータが得られる）する．

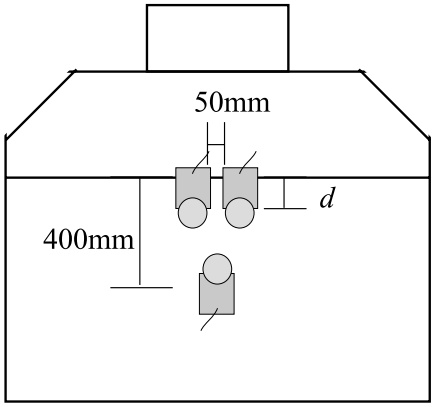

Fig. 5.10 ロボットの位置

Fig.5.11 から Fig.5.17[7)]は，観測結果が得られたデータ[8)]（線分）をロボット座標系に全て描いたグラフである．これらのデータについて表にまとめると Table5.1 のようになる．Table5.1 では，他のロボットがフィールドに存在してもラインの計測が可能であること，ペナルティーラインが計測できた場合，計測結果には障害物の影響がそれほど表れないことが分かる．また，視界が狭くてもいずれかの壁・ラインを抽出可能なことが多いことも，壁・ライン位置計測アルゴリズムが雑音に対して強いことを示している．

Table 5.1 ペナルティーラインの計測結果

d[mm]	計測回数	他のラインの計測回数	λ[mm]		ζ[deg]	
			平均	標準偏差	平均	標準偏差
他機無し	45	10	385	29	-3.0	3.1
0	36	16	379	22	-1.3	4.1
50	42	16	377	30	-1.1	5.3
100	35	17	382	32	-2.4	7.3
150	33	18	399	54	-2.8	8.6
200	32	17	376	49	-1.4	6.8
250	15	8	378	28	-3.3	4.0

[7)]図中の赤い線分はライン，青い線分は壁と判断されたものである．
[8)]2.3.5 項で付加した「信頼度」が 10 以上のものである．

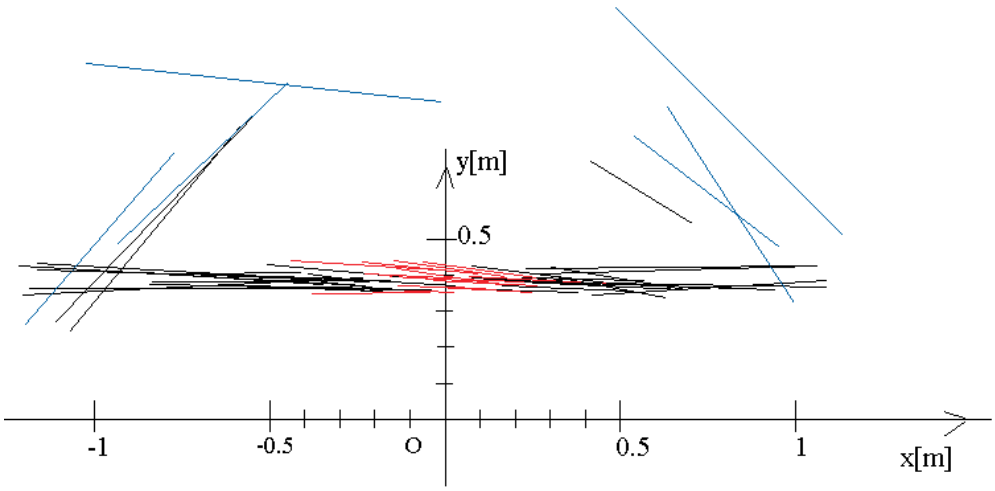

Fig. 5.11 他機なし

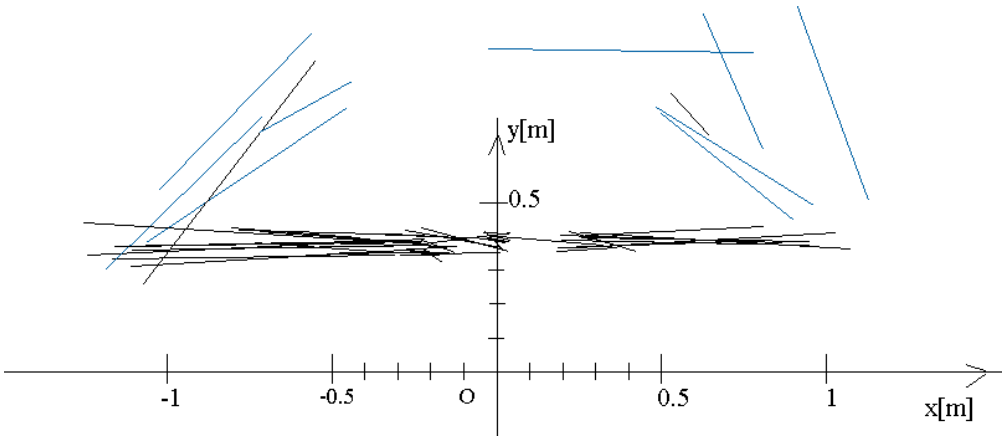

Fig. 5.12 $d = 0$

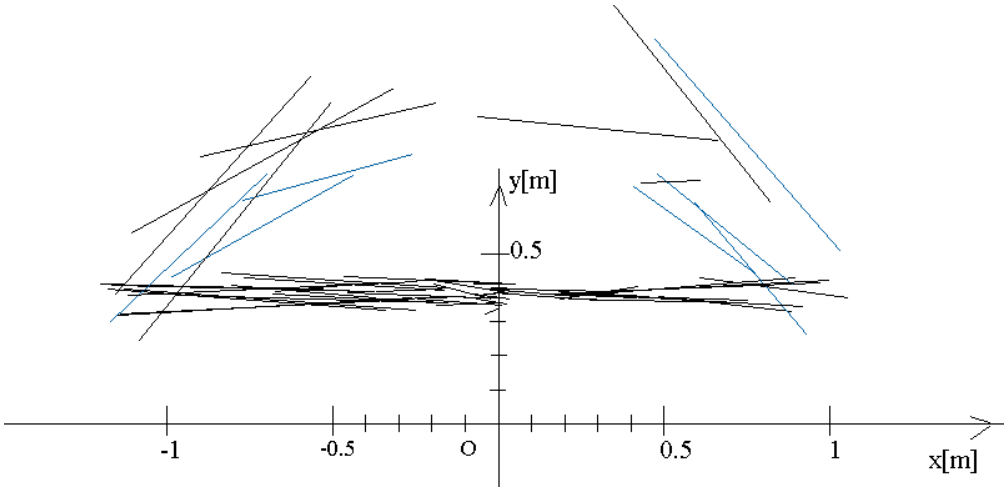

Fig. 5.13 $d = 50$

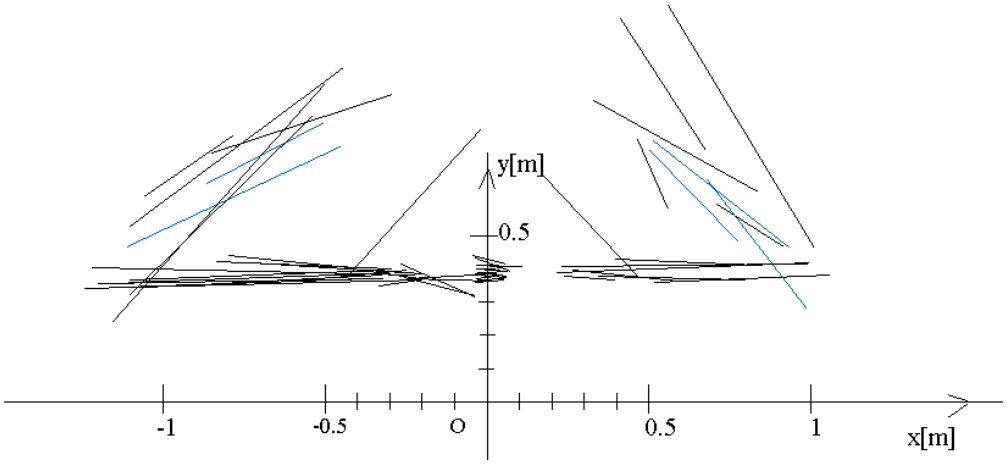

Fig. 5.14 $d = 100$

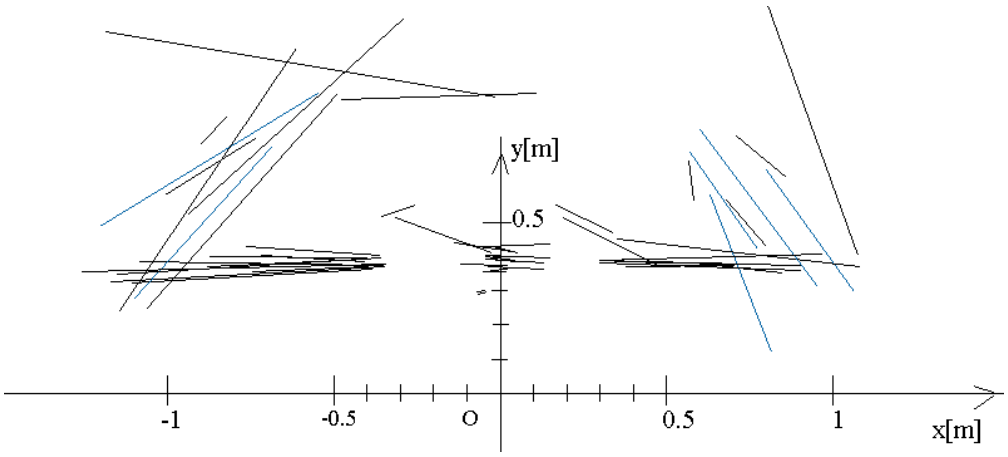

Fig. 5.15 $d = 150$

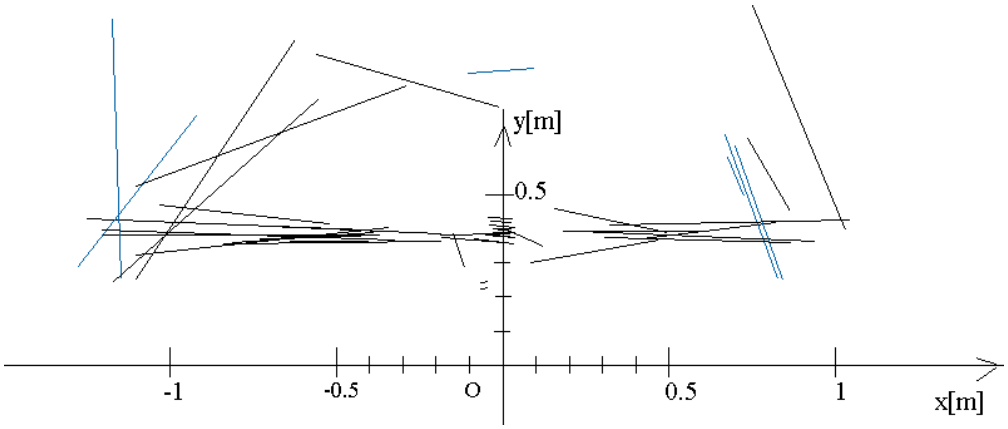

Fig. 5.16 $d = 200$

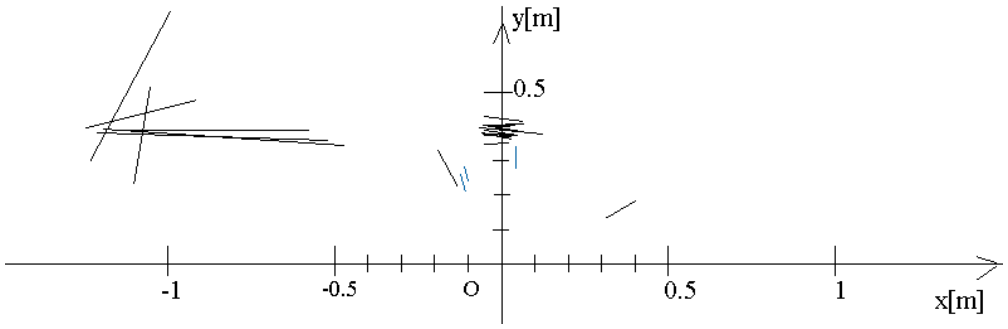

Fig. 5.17 $d = 250$

5.4 自己位置同定

本節では，シミュレーションによって，自己位置同定における壁・ライン位置計測アルゴリズムの有効性を評価する．

5.4.1 シミュレーション環境

本節では [横井 2001] で紹介されている SRL のシミュレーション環境に壁・ライン位置計測アルゴリズムを組み込むなど拡張したものを使用する．このシミュレーション環境を Fig.5.18 に示し，その主要な部分について説明する．

上の画面 サンプル（4.2.1 参照）の現在の分布状況を表示.

下の画面 現在のロボットの位置を表示.

YP,GP,SP,PY,PG,PS ランドマークの位置.

Correct Position ロボットの現在位置（左から Σ_{field} での x 座標，y 座標，角度）.

result Calculation of POSITION ロボットの推定位置（サンプルの重みつき平均）.

Standard Deviation サンプルの重みつき標準偏差.

ライン観測 新たに拡張した部分で，次の要素からなる．

- ライン選択ボタン：ロボットに観測させる壁・ラインを選択.
- Observe ボタン：ロボットの実際の位置と選択した壁・ラインの位置から距離，角度を求める．この際，式 (4.3.12)，(4.3.13) と乱数によって観測誤差を乗せる.
- Line，Wall チェックボックス：観測したものが壁かラインか“ unknown ”かを操作する.

- 適用：Line，Wall チェックボックスの状況から，式 (4.3.16)，(4.3.17)，(4.3.18) を使い分けて $P(\acute{\lambda}, \acute{\zeta}|\nu_i)$ を計算，サンプルに計算結果を反映する．

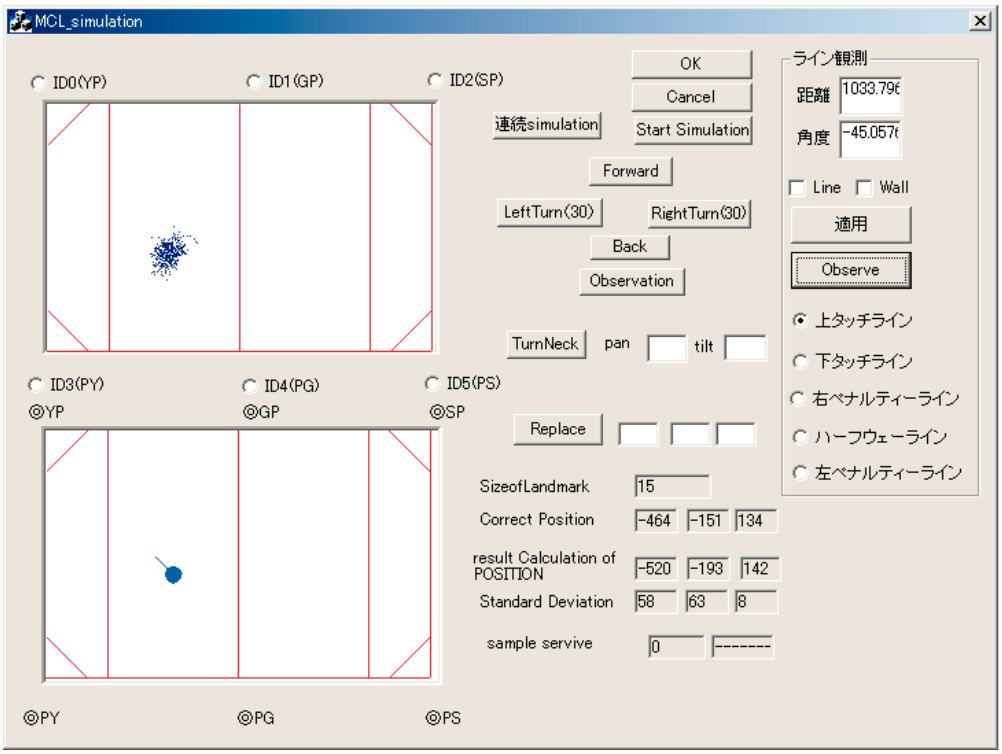

Fig. 5.18 SRL シミュレーション

5.4.2 シミュレーション

上記のシミュレーション環境を使用して，４種類の自己位置同定の条件で「敵陣から自陣ペナルティーラインまで戻る」というタスク[9]について比較を行った．自己位置同定の条件は，

A. デッドレコニングのみ．

B. 一歩ごとに一つのランドマークを観測する．

C. 一歩ごとにペナルティーライン，ハーフウェーラインのうちの一本を観測する（ただし壁かラインかは分からないとする）．

[9] このタスクは，「Two Defenders（A.1.2 項参照）を意識した守備」を想定して設定した．

D. 一歩ごとに（ii）,（iii）を両方行う.

また，サンプルの数は 400 とした．具体的なアルゴリズムを Fig.5.19 に示す.

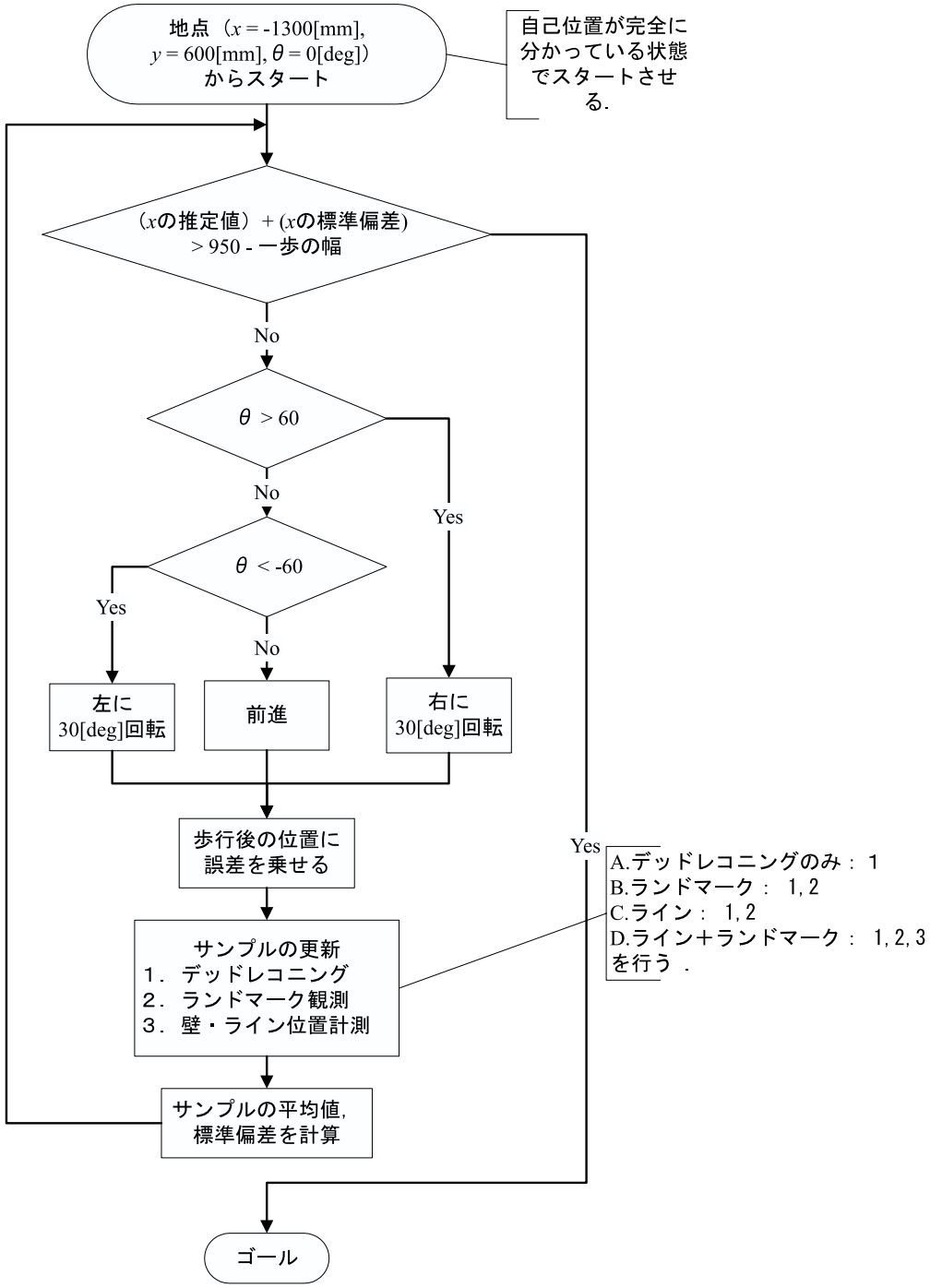

Fig. 5.19 シミュレーションのアルゴリズム

各条件ごとに最終的なロボットの位置（フィールドの x 座標）を度数分布表にすると Fig.5.20 のようになった．壁・ライン位置計測の場合，目標となる 950[mm] の前後 50[mm] で停止する頻度が，ランドマーク位置計測の場合の約 2 倍となっている．これは，壁やラインの付近に到達するためには，壁・ラインの位置から自己位置同定を行うことが有効であることを示している．また，壁・ラインによる自己位置同定では，壁・ラインが近くに存在すれば，ランドマークを使用するよりも高精度の自己位置同定が見込めることも示している．

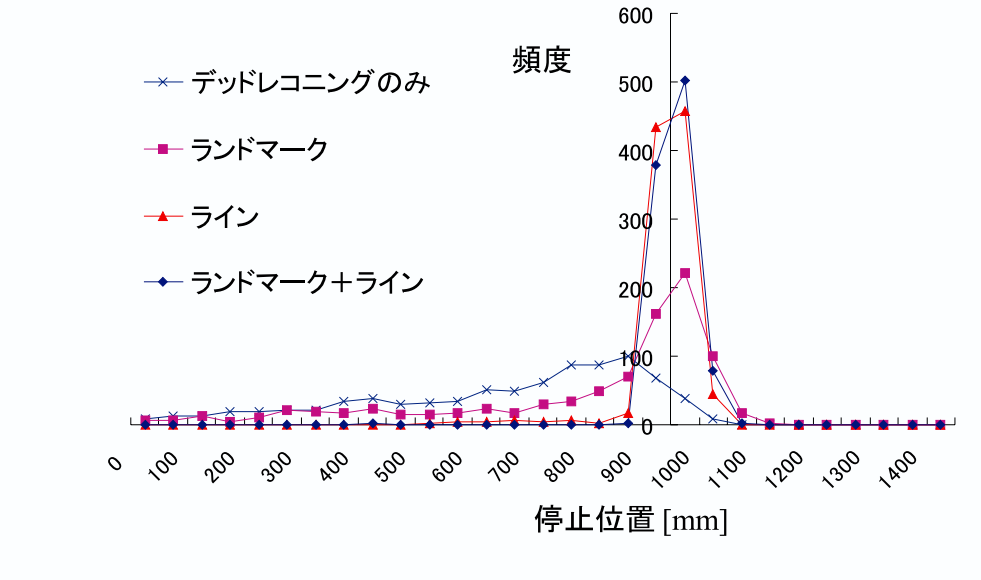

Fig. 5.20 シミュレーション結果

5.5 おわりに

本章では，第 2，3，4 章で設計したアルゴリズムについて評価を行った.

5.2 節において，第 2，3 章で設計した「障害物位置計測アルゴリズム」の理論上の誤差について，１ m 以下では画素のずれが誤差に影響し，１ m 以上では胴体の傾きが大きな誤差の原因となることを述べた.

5.3 節において，障害物位置計測アルゴリズムを実機で実行した場合，誤差については実用に耐えうることと，一歩ごとに壁・ライン位置計測とロボット位置計測を両方行うことは困難であること，壁・ライン位置計測アルゴリズムが雑音に対して強いことを述べた.

5.4 節において，壁・ライン位置計測アルゴリズムを自己位置同定に利用した際の有効性を，シミュレーションを用いて示した.

第6章　実機実験

6.1 はじめに

本章では，第 2 章，第 3 章で設計した障害物位置計測アルゴリズムと，第 4 章で設計した自己位置同定用のアルゴリズムの具体的な使用例を示し，設計したアルゴリズムの有効性について述べる．

6.2 節において，障害物位置計測アルゴリズムを使用した障害物回避の例を示す．

6.3 節において，壁・ライン位置計測アルゴリズムを自己位置同定に使用した例を示す．

6.4 節において，本章を総括する．

6.2 障害物回避実験

6.2.1 壁・ライン回避実験

壁・ライン位置計測アルゴリズムを使用して，壁・ラインを回避する行動を実装した．アルゴリズムは Fig.6.1 のようにした．

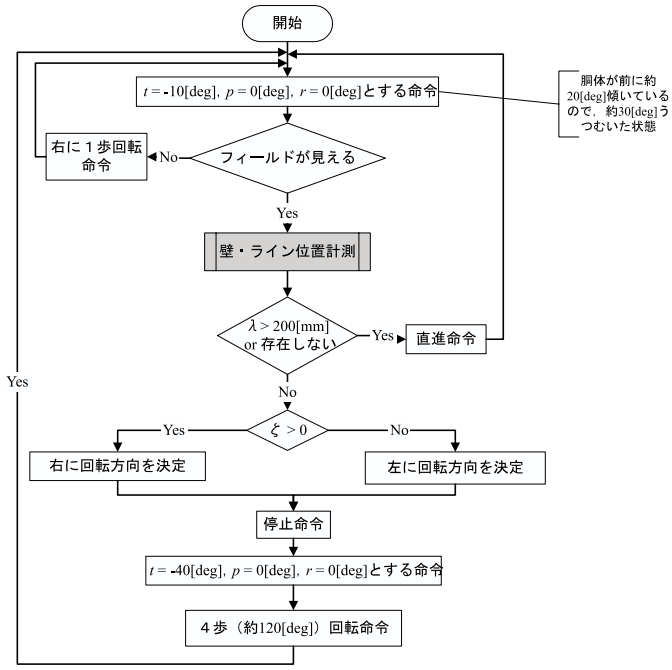

Fig. 6.1 壁・ライン回避アルゴリズム

Fig.6.1 のアルゴリズムの性能を評価するため，ERS-1100 にペナルティーラインとハーフウェイラインの間を 30 分間往復させ[1)]，壁への衝突回数やラインを脚が２本以上越える[2)]回数などを調べた．実験の条件は次のようになっている．

i. 歩行の速度は 0.13[m/s] であり，１歩で約 80[mm] 進む[3)]．

ii. 停止命令を出すのは壁・ラインを 200[mm] 以内に発見したとき．

[1)] ERS-1100 のバッテリーは常に歩行していると約 15 分で切れるので，３回に分けて実験を行った．

[2)] フォワードがこれを自陣ペナルティーラインで行うと“ Two Defenders ”の反則（A.1.2 節参照）になる．

[3)] 当チームの最高速歩行である．

200[mm] は，１歩 80[mm] であると，あと２歩で壁に衝突する距離である（Fig.6.2）．歩行と Fig.6.1 のアルゴリズムは非同期で実行されるので，停止命令から実際に止まるまでには１歩進んでしまうことから ii の設定にした．

Fig. 6.2 壁まで 200[mm] の位置

実験結果をまとめると，Table6.1 のようになった．この表からは壁・ライン位置計測アルゴリズムによって壁・ライン回避が 84% の確率で成功したと言える．

Table 6.1 実験結果

事象	回数
ラインに遭遇	109
壁に遭遇	62
脚が２本ラインから出る	9
壁に衝突後，歩行	17
壁に衝突後，３歩以上歩行	0
200[mm] 以上手前で回避	2
ラインから出て帰らない	0

実験中に見られた回避行動の例を Fig.6.3 に示す．この図では，ロボットが壁やラインに十分近い位置で回避を行っている．また，計算時間のかかりすぎによって歩行が遅くなるという現象は見られなかった．

以上の結果から，壁・ライン位置計測アルゴリズムを使用することで，壁やラインから十分近い位置への接近と，壁・ラインの回避が両立できることを確認した．

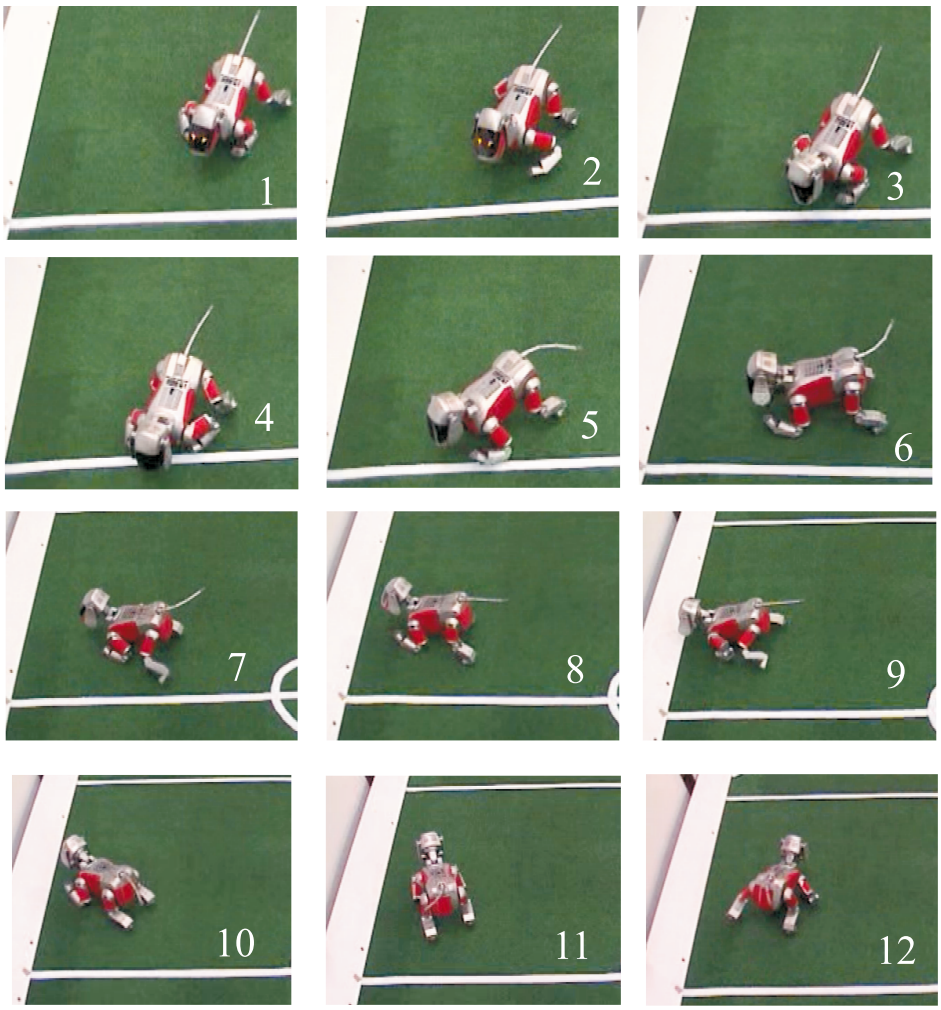

Fig. 6.3 壁・ライン回避行動

6.2.2 ロボット回避実験

ロボットの回避については２通りの方法で実験を行う．実験１では，典型的な衝突回避行動の例を示し，実験２では衝突回避率を調べる．

実験１

味方同士の回避行動を実装した．フォワードのうち一台を「味方ロボットが前方 L_{f}[mm] 以内，左右 L_{s}[mm] 以内に存在したら横歩きでかわす（横歩き型）」というアルゴリズム（Fig.6.4）にして，もう一台を「味方ロボットが前方 L_{f}[mm] 以内，左右 L_{s}[mm] 以内に存在したら立ち止まる（停止型）」というアルゴリズム（Fig.6.5）

にした．$L_{\mathrm{f}} = 400, L_{\mathrm{s}} = 300$ として得られた回避行動の例を Fig.6.6，6.7 に示す．

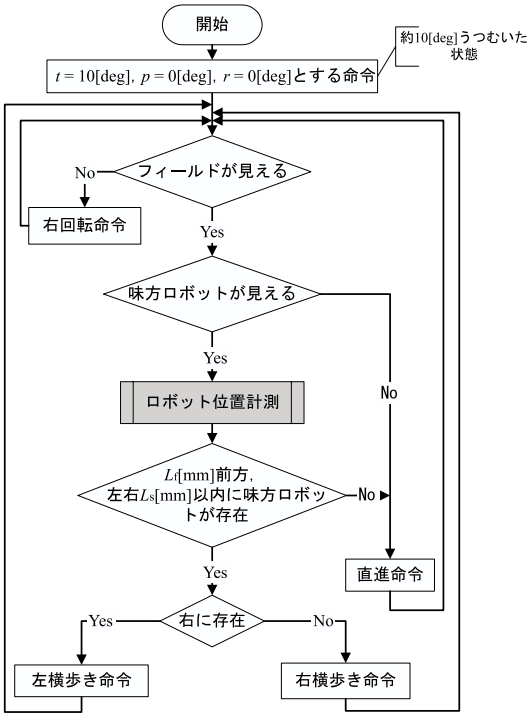

Fig. 6.4 横歩き型

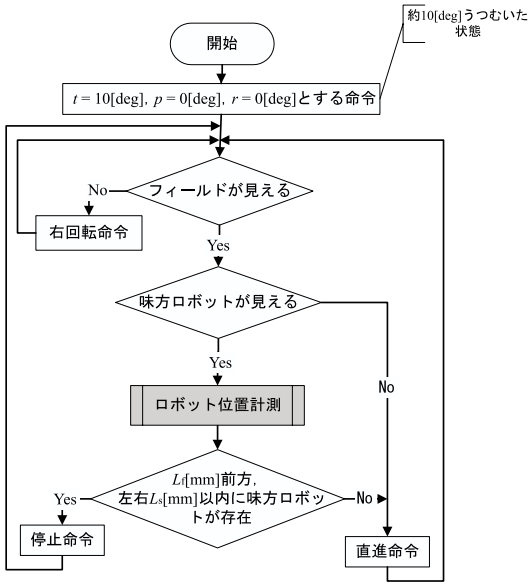

Fig. 6.5 停止型

Fig.6.6 は，正面衝突回避の例である．１の左側が停止型，右側が横歩き型である．停止型は３で横歩き型を発見し，停止．７で横歩き型が視界から消えたので前進を再開している．横歩き型は，５，６で回避のための横歩きを行い，７で前進を再開している．

Fig.6.7 は，側面からの衝突を回避する例である．１の左側が横歩き方，右側が停止型のロボットである．停止型のロボットは３でロボットを発見し，１歩オーバーランした後，４，５，６で横歩き型のロボットが通過するのを待ち，７で前進を再開している．

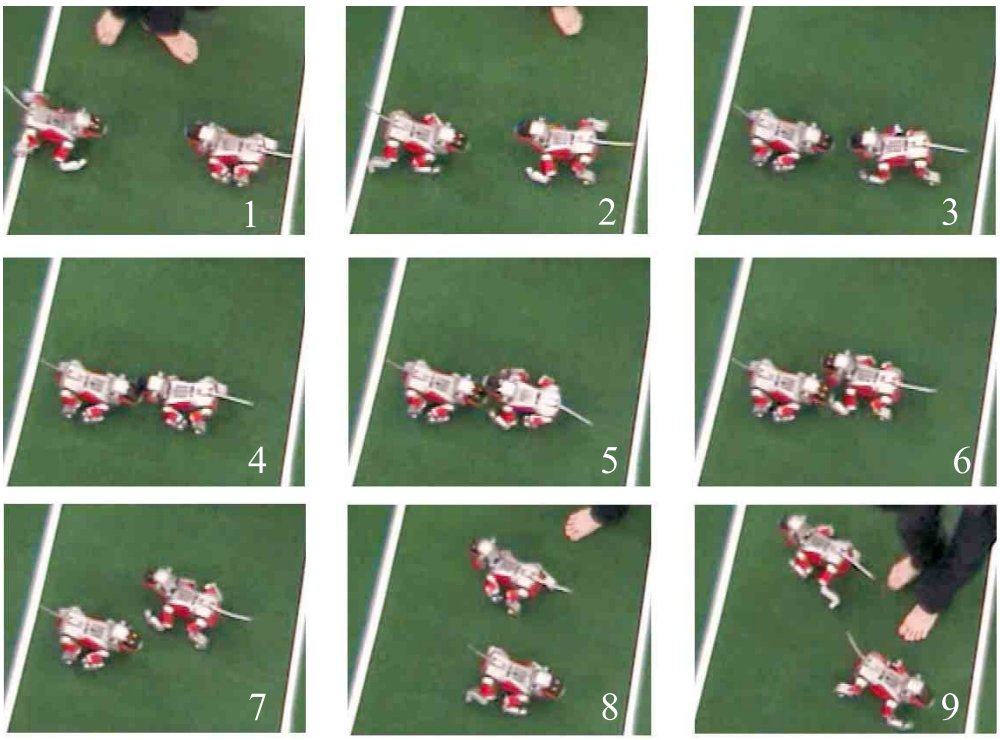

Fig. 6.6 正面衝突回避の例

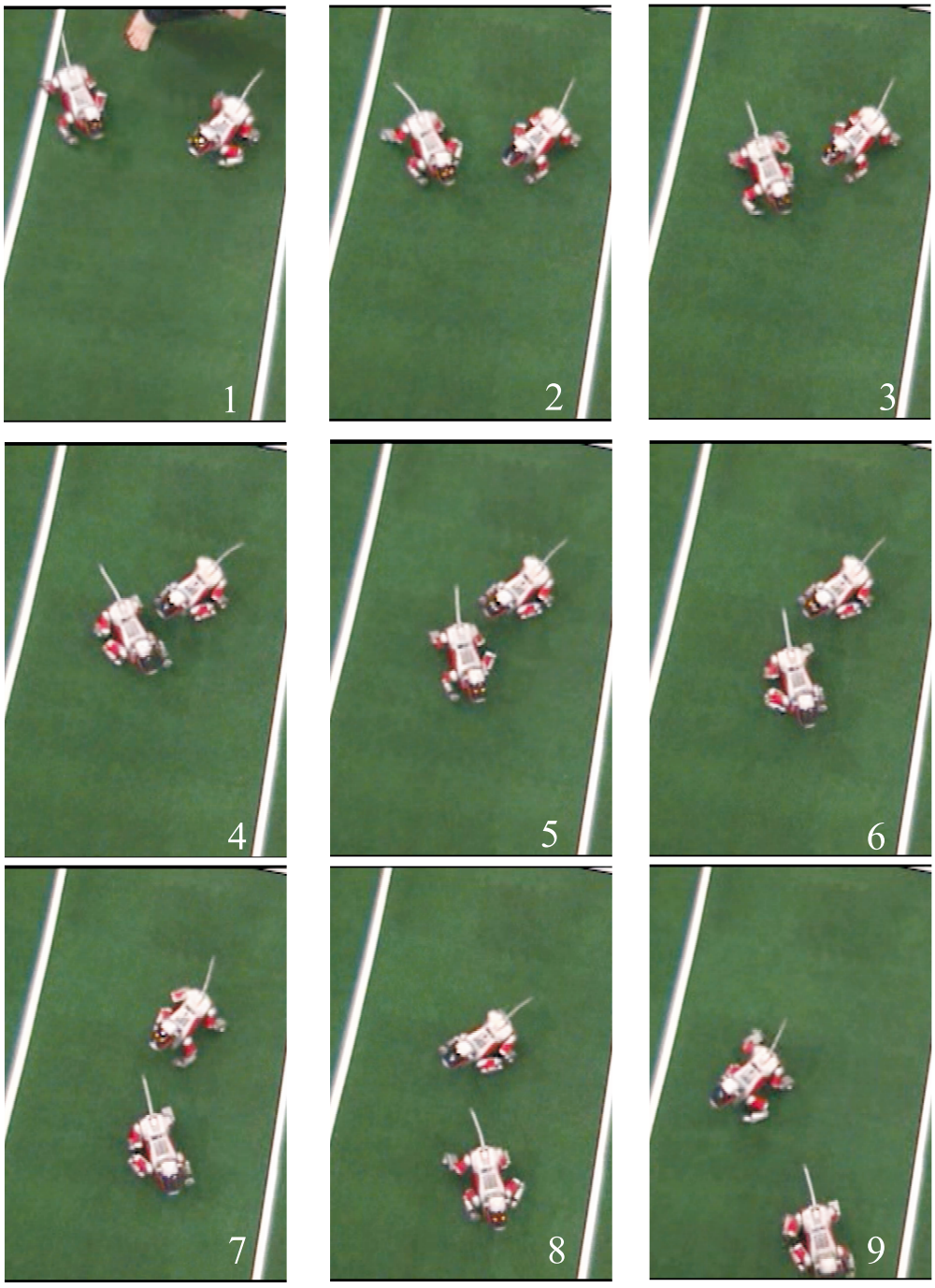

Fig. 6.7 側面衝突回避の例

実験２

実戦を想定して次のように実験１のアルゴリズムを変更し，衝突回避の成功率を調べる実験を行った．

- $L_{\mathrm{f}} = 500[\mathrm{mm}]$，$L_{\mathrm{s}} = 300[\mathrm{mm}]$ とする．
- 横歩き型のロボットの，フィールドが見えないときの回転方向を右回転から左回転に変更（ロボット同士の遭遇頻度を上げるため）．
- ロボットの首の向きを $(t, p, r) = (10, 0, 0), (10, -45, 0), (10, 0, 0), (10, 45, 0)$ の順に，0.8[s] ごとに変化させる．ただし，ロボットを近距離に発見したら回避するまで首の向きを変化させない．

また，衝突頻度を上げるために，ペナルティーラインとハーフウェイラインの中間あたりに Fig.6.8 のような壁を置いた．

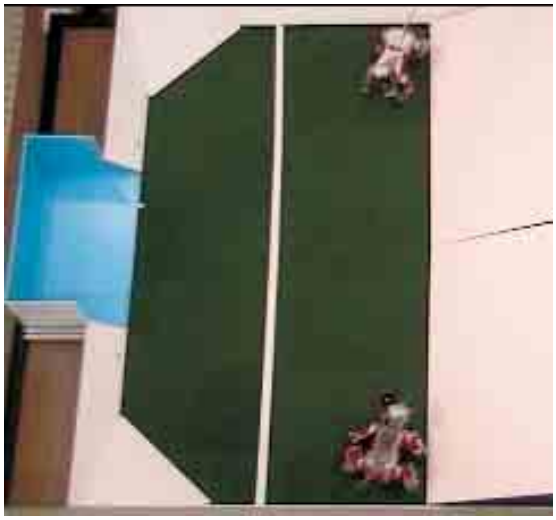

Fig. 6.8 実験環境

約 20 分間２台のロボットを歩行させ，Table6.2 のような結果を得た．結果的に回避した場合を入れると，回避成功の確率は 71% であった．

実験中に見られた衝突の多くは Fig.6.9 のいずれかのパターンに属していた．Fig.6.9 の左側は，横歩き型のロボットは，歩行中の停止型ロボットが右に存在す

[4] フィールドが見えなくて回転した（Fig.6.4 参照）．

Table 6.2 実験結果

事象	回数
回避成功	24
結果的に回避[4)]	1
衝突して進路が少し変化	2
脚が絡む	8
計	35

ると判断して右に回避しようとしているが，停止型ロボットが同じ方向に進んでいるために避けることができず衝突するという例である．この衝突は，情報として歩行中のロボットの進行方向を使用していない場合避けられない．したがって，回避されるロボットの進行方向や速度が推定できることが望ましいが，ERS-1100 の CCD カメラの性能では，かなり困難であると思われる．

Fig.6.9 の右側は，壁が障害となって横歩き型が停止型を避けられないうちに衝突するという例である．これは，壁回避や自己位置同定を利用することで避けることができる．

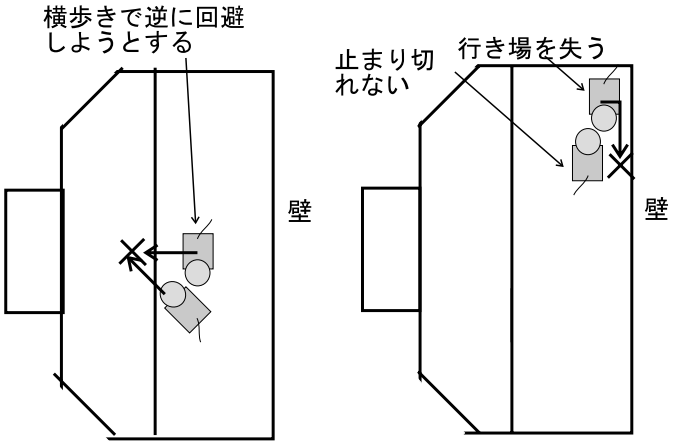

Fig. 6.9 衝突例

以上の結果から，ロボット位置計測アルゴリズムを使用すると，71% はロボット同士の衝突回避が可能になり，壁回避や自己位置同定を利用することで成功確率はさらに大きくなることが見込めることを確認した．しかし，衝突の全てを防ぐには他機の進行方向などを利用するなどの改善が必要であることも分かった．

6.3 自己位置同定

自己位置同定に壁・ライン位置計測アルゴリズムを利用することの効果を示すために，5.4.2 項においてシミュレーションを行った「敵陣から自陣ペナルティーラインまで戻る」というタスクを実装し，停止した位置，停止位置までにかかった時間を調べた．シミュレーションのアルゴリズム（Fig.5.18）内の条件のうち，変更した点を以下に示す．

- スタートのロボットの位置・姿勢を $(x_{\mathrm{s}}, y_{\mathrm{s}}, \theta_{\mathrm{s}}) = (-950, 0, 90)$ とする．また，フィールドに位置・姿勢が $(x, y, \theta) = (0, 0, 90)$ の静止したロボットを置き，これを回避させる (Fig.6.10)．

- スタート時のサンプル (x_i, y_i, θ_i) を乱数を使用してばらつかせる．正規分布にしたがい，平均値 $(x_{\mathrm{s}}, y_{\mathrm{s}}, \theta_{\mathrm{s}})$，$x, y$ についてそれぞれ標準偏差 100[mm]，θ について標準偏差 10[deg] で無相関にばらつかせる．

- ランドマーク，壁・ライン位置計測は，x_i の平均値が 0 以上になったら開始する．これは，ランドマーク位置計測を行う首の角度ではロボットの回避ができないので，回避が終わったころからランドマーク位置計測を開始するためである．

- 壁・ラインの位置計測は，一歩につき一回．ランドマークの位置計測はランドマークが視界に入り次第行う．また，頭部関節角を次のように変化させる[5)]．

  i. ランドマーク位置計測時：$t = 25$[deg],$r = 0$[deg]．p は，0,30,60,30, -30,-60,-30[deg] の順に 0.8 秒ごとに変化させる．

  ii. i 以外の時：$t = -10$[deg] とする．p, r は i と同じ．

- y_i の平均値が 700[mm] 以上，あるいは-700[mm] 以下になったら θ_i の平均値にしたがって壁を回避する．

- ランドマーク位置計測の場合，停止条件「（x_i の平均値）＋（x_i の標準偏差）> 870[mm]」の後もサンプルが更新されたら動いてよい．これは，ランドマーク位置計測の場合，サンプルのリセット（4.2.1 項参照）が起こり，一時的に x_i の標準偏差が増大することがあるからである．

[5)] 頭部の動きは歩行と非同期である．

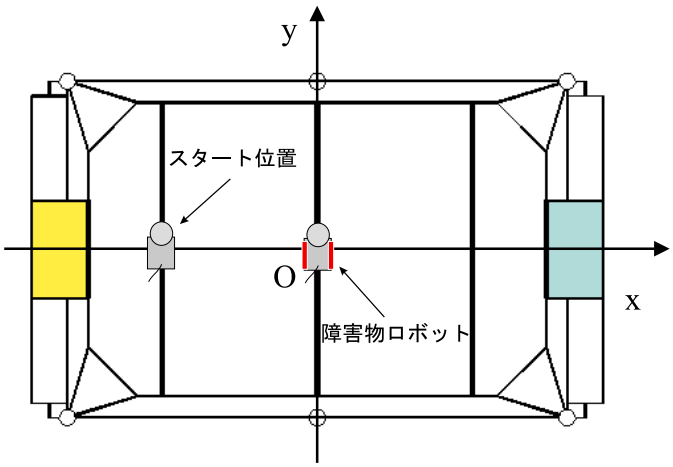

Fig. 6.10 ロボットの初期位置

デッドレコニングのみ・デッドレコニングとライン位置計測・デッドレコニングとランドマーク位置計測の場合をそれぞれ 10 回ずつ行い，得られた到達位置と理想値（950[mm]）との差 e_X，停止までの時間 t_G について Table6.3 のようにまとめた.

Table 6.3 実験結果

使用する情報	デッドレコニングのみ		デッドレコニング＋壁・ライン		デッドレコニング＋ランドマーク	
	e_X[mm]	t_G[sec]	e_X[mm]	t_G[sec]	e_X[mm]	t_G[sec]
	-240	24	-80	37	110	37
	-260	33	-1040	71	-170	26
	-1050	36	-80	39	-270	43
	-180	39	-110	98	80	40
	-430	27	-310	33	140	32
	-1080	36	-150	40	-460	50
	-610	25	210	51	-560	26
	-450	27	-50	57	-50	27
	-380	25	-940	314	-420	26
	-30	26	-30	98	-50	60
平均	-471	30	-258	84	-165	37

Table6.3 からは以下の傾向が読み取れる．

i. デッドレコニングのみでは，到達までの時間が早いが，大きく位置を間違えることがある．

ii. デッドレコニング＋壁・ラインでは，到達までに時間がかかる．これは，壁・ライン位置計測して自己位置同定するまでに時間がかかり，歩行が一瞬止まるためである．また，大きく位置を間違えることがある一方で，目標のライン付近まで近づけば，精度良く止まることができる．

iii. デッドレコニング＋ランドマークでは，デッドレコニングより時間がかかるが，大きく位置を間違えることは他の場合に比べて少ない．

ii で見られる大きな位置の間違えは，以下の条件が重なったとき壁・ライン位置計測を行うと起こると考えられる．

1. サンプルが大きくばらついている．
2. 観測するラインとロボットの位置・姿勢の相対的な位置関係と，観測したものとは異なるラインと一つのサンプル ν_{e} の相対位置・姿勢がほぼ一致する．
3. ロボットと近い位置・姿勢のサンプルが存在するが，ν_{e} ほどはラインとの相対位置・姿勢が一致していない．

このとき，サンプル ν_{e} の重みが大きくなり，ロボットの位置・姿勢に近いサンプルの重みを大きく上回る（Fig.6.11）．

この間違えが起こると，ロボットの推定位置が大きく異なるにもかかわらずサンプルのばらつき（重みつき標準偏差）が小さくなるという，致命的な状況になる．同様の例がシミュレーションで確認できたので，Fig.6.12 に示す．

この問題の解決策としては，次のことが考えられる．

a. サンプル数を増加させる．
b. ランドマークの位置計測などで大まかな位置・姿勢が分かった後に壁・ライン位置を自己位置同定に反映させる．

a の方法は，計算量の問題から困難である．また，b の方法は，壁・ラインが近い場合にはランドマーク計測よりも正確な自己位置同定ができるので，正確な自己位置が必要な場合には有効である．

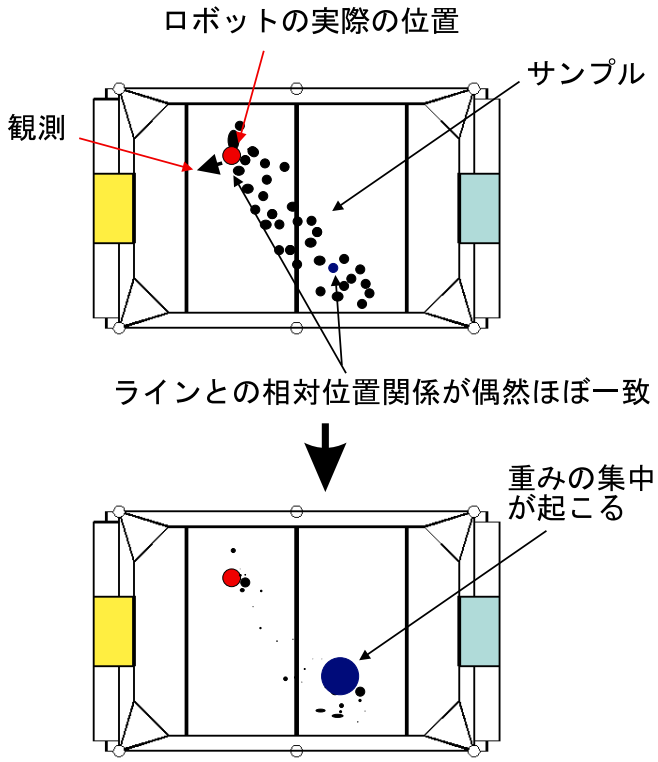

Fig. 6.11 ラインの取り違え

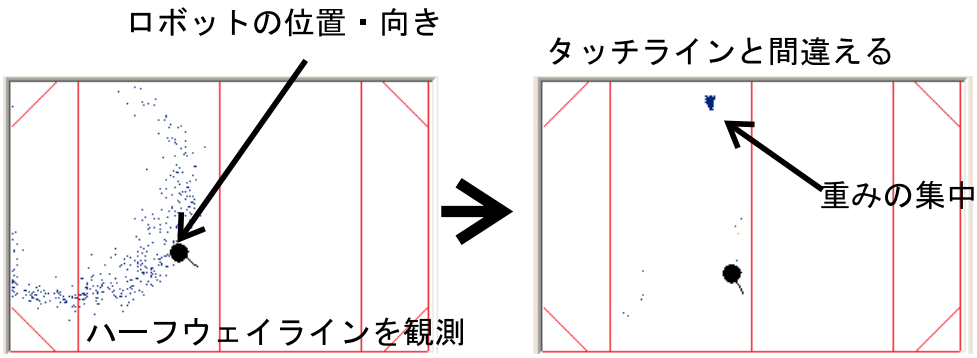

Fig. 6.12 ラインの取り違え（シミュレーション）

以上の結果・考察から，壁・ライン位置計測アルゴリズムが自己位置同定に有効であるのは，サンプルのばらつきの少ない状況で，さらに正確な位置情報が必要なときであると言える．

6.4 おわりに

本章では，第 2 章，第 3 章，第 4 章で設計したアルゴリズムの具体的な使用例と，設計したアルゴリズムの有効性について述べた．

6.2 節において，位置計測アルゴリズムを使用した障害物回避の例を示した．壁・ラインの回避では，84% の成功率．ロボット同士の回避では，71% の成功率が得られた．

6.3 節において，壁・ライン位置計測アルゴリズムを自己位置同定に使用した例を示し，壁・ラインの位置情報を使用すべき状況を明らかにした．

第7章　結論

7.1 結論

本論文では，Legged Robot League で確立されていなかった「画像中の面積・幅の情報が利用できない物体の位置を計測する手法（障害物位置計測アルゴリズム）」を提案・実装・評価し，この手法を用いて障害物回避行動の獲得・自己位置同定の精度を向上することを目指した．

まず，障害物位置計測アルゴリズムを，画像上から観測対象物を抽出する過程と，観測対象物の画像上での位置をロボット座標系へ変換する過程の，２つの過程に分けて設計した．画像上の観測対象物の抽出過程・座標変換過程で挙げた解決すべき問題点について，解決法と，その解決法について評価・考察したことをまとめると以下のようになる．

画像上の観測対象物の抽出過程

低画素数　これは，壁・ラインとフィールドの境界線抽出の際，ハフ変換に悪影響を与える原因となった．そこで，抽出した画素から，画素内での境界線の位置を推定することで解決し，結果としてハフ変換の信頼性，精度に向上させることができた．

フィールド外のノイズ　ハフ変換，メディアン・フィルタを利用することで解決した．結果として，誤認識の比較的少ない抽出が可能となった．

フィールド内のノイズ　ハフ変換を用いることで，壁・ラインの位置計測の際に他機が視界を遮る問題を解決し，その結果 5.3.4 項で見られるように，フィールド内の他機によるノイズに対してロバストな壁・ライン位置計測が可能となった．

CPU の処理速度　画像上の観測対象物の抽出過程で最も計算量が多かったのはハフ変換の部分であったので，ハフ変換の計算量を減らした．方法として，C 標準ライブラリの $\sin(), \cos()$ を使用せずに，$\sin(), \cos()$ の計算値の配列をあらかじめ作成しておく方法をとった．この方法により，壁・ライン位置計測アルゴリズム全体の計算時間が 1/3 に減少した．

座標変換過程

この過程では，CCD カメラが 15 自由度の影響を受けるために，カメラ座標系からロボット座標系への同次変換行列の計算が困難であることが問題であった．この問題では，Table3.1 のようにあらかじめ脚の高さを姿勢ごとに計測した表をプログラム中に配列で持っておき，現在の姿勢に応じて表の値から同次変換行列を作成するというアルゴリズムを用いることで解決した．この方法では，精度に悪影響を及ぼす恐れがあったが，Fig.5.2，Fig.5.3 のように，１ [m] までは高い精度が保証されることが分かった．

これらの問題点の解決により，精度，計算時間ともに障害物回避，自己位置同定に使用するために十分な障害物位置計測アルゴリズムを実現した．それによって，本研究で第一の目的とした「画像中の面積・幅の情報が利用できない物体の位置を計測する」は達成された．

次に，障害物位置計測アルゴリズムを利用した障害物回避行動を実装した．回避行動は,「一定の距離・範囲内に障害物が入ったら避ける」という単純なものであったが，それでも壁・ラインについて 84% ，ロボットについて 71% の成功確率を達成し，障害物位置計測アルゴリズムの有効性を示した．これにより「障害物（ロボット・壁・ライン）の位置を計測し，必要なときに回避すること」が達成された．

また，壁・ライン位置計測アルゴリズムを“ Sensor Resetting Localization ”に組み込み，自己位置同定のシミュレーションと実機実験を行った．シミュレーションでは，壁・ライン位置計測アルゴリズムによって壁・ライン付近の自己位置同定を高精度で行えることを示した．また，実機実験では「壁・ラインを観測し，自己位置同定に利用すること」が達成された．そして，壁・ライン位置計測アルゴリズムは Sensor Resetting Localization のサンプルのばらつきが小さい状況で，さらに正確な位置情報が必要なときに自己位置同定に使用することが有効であることを明らかにした．

7.2 今後の展望

本研究の次の段階として考えられる作業・研究を挙げる．

能動的に観測対象物を視界に入れる 本研究では，障害物回避・自己位置同定の実機実験で，首の関節角の条件を天下り的に与えた．しかし，最適な首の関節角を調べて頭部を制御するアルゴリズムを実装すれは，位置計測の精度の向上，観測時間の短縮が期待できる．そこで，SRL のサンプルを使用して，確率的に最も視界に壁やラインが入りやすい関節角を計算する手法を提案する予定である．

画像処理の高速化 現状では壁・ライン・ロボットの位置を全て計測しているとロボットが１歩進む時間以内に処理が終わらない場合があるので，障害物位置計測アルゴリズムの高速化を目指す．現在最も計算時間がかかるのはハフ変換の部分である．本研究では従来からのハフ変換を用いたが，高速にハフ変換を行う手法がいくつか提案されているのでそれらの実装を検討する必要がある．

存在確率密度の計算の高速化 自己位置同定の実機実験では，壁・ライン位置計測アルゴリズムの出力から SRL のサンプルの重みを変える際に処理時間がかかり，歩行に影響が出た．これは，壁・ライン位置計測アルゴリズムを自己位置同定に使用する効果を小さくしているように思われる．現在，存在確率密度分布は正規分布で表現しているが，これを近似する分布で代用し，高速化をはかる必要がある．

協調作業 本研究では，味方同士の衝突回避など，協調の初歩的な行動が実現できた．今後，自己位置同定なども利用すると，パスなどのより高度な協調作業も現実可能であると思われる．

謝辞

本論文は，東京大学工学部精密機械工学科　新井・湯浅・太田研究室において書かれたものです．執筆に当たり以下の多くの方にお世話になりました。深くお礼申し上げます.

指導教官である湯浅助教授には，研究に対する基本的な姿勢や態度などについて直接ご指導をいただき大変感謝しています．また，その工学的のみならず，多様で豊富な知識を心から尊敬しています.

新井教授には鋭い洞察力・膨大な知識に基づく適切なご助言を折に触れていただきました．常に視点を高く保ち，示唆に富んだご指摘は，非常に参考になりました．心から感謝しております．また，新井先生の生き様も魅力的で，多く学ぶところがありました.

太田助教授には，本論文を完成させるために有効な文献をたくさん紹介していただきました．また，何度か研究について直接指導を受けることができ，感激しております．本当に有難うございました.

助手の前田さんには，14 号館に行った際，14 号館の PC のことについての質問に答えていただきました．大変助かりました.

博士 3 年の井上さんは，研究室全体のことにも常に目を向けていらっしゃるという印象をうけました．研究に対する良い意味でのこだわりと情熱を尊敬しています.

博士 3 年の中村さんは，自らの研究の傍らで後輩の指導にも余念がない方であると思います.

博士 3 年の原さんには，大変研究熱心な方で，いつも良い刺激を受けることができました.

博士 3 年の山下さんは，４月に内線の応答の仕方を教えていただいたのが印象に残っています.

直接ご指導いただいた博士２年の小林さんには，この一年間ずっとお世話になりっぱなしでした．特に，なにか文章を書くたびに何度もしつこく校正をお願いしたにもかかわらず，いやな顔ひとつせず引き受けて下さり，とても感謝しています．また，研究に対する姿勢などが大変参考になりました．

博士２年の松本さんには，研究室でともに過ごす時間が長かったため，よく雑談させていただきました．どんな話題でも，楽しく話ができました．

博士１年の稲垣さんには，おなじく身近な相談相手として，多くの分からない事柄に答えていただきました．また，ロボットコンテストのとき，何とかロボットを完成させようという気迫には感銘を受けました．

博士１年の杉さんには，何度か食事をご一緒し，話を聞かせていただきました．

博士１年のタオさんは，いつも明るく，和やかな気持ちにさせて下さる方でした．

博士課程の熊谷さんは，社会人として仕事と大学研究の両立をし，幅広い知識を得ようとする姿に感銘を受けました．私も見習いたいと思います．

同グループの修士２年横井さんとは，研究の上で共同で進める作業も多く，二人三脚，励まし励まされの関係で頑張りました．たくさんのアドバイスをいただき，大変感謝しております．また，２，３年前の研究室の様子の話は大変面白かったです．

修士２年の岩田さんは，さわやかさがとても印象に残っています．

修士２年の竹内さんには，時折温かい言葉をかけていただき，とても励まされました．

修士２年の千葉さんには，研究に対する真摯な態度を学ばせていただきました．

修士 2 年の中山さんには，楽しい話を聞かせていただきました．

修士 2 年の山本さんには，14 号館に行ったとき，気さくに話しかけていただきました．

同グループの修士 1 年深瀬さんには，多くの事柄で相談に乗っていただき，研究の基礎的な部分で支えていただきました．RoboCup2001 ではともにアメリカの大空に羽ばたきましょう．来年度もよろしくお願いします．

修士 1 年の香月さんには，たまにしか 14 号館に行かないにもかかわらず，よく話しかけていただきました．

修士 1 年の市川さんは，研究室の OA 係りとして，普段の研究室生活を支えてくださいました．

修士 1 年の大倉さんには，ミニ論や公開研など要所でお世話になることが多く，発表における基本姿勢や心構えなどを教わりお世話になりました．

修士 1 年の中村さんの何事にも積極的に挑戦するバイタリティを僕も見習いたいと常々思っています．

修士 1 年の金子さんには，アメリカ滞在時の数々の武勇伝を聞かせていただき，今年当地に赴く身として武者震いせずにはいられませんでした．

修士 1 年の藤本さんは，WorkStation 係りとして研究室の計算機環境をサポートしてくださいました．

修士 1 年の菊地さんは，PC 係りとして研究室の計算機環境をサポートしてくださいました．

修士 1 年のチェンさんの楽しい笑顔に，研究漬けの生活ですさんだ心を幾度も慰められました．

秘書の井口さん，米岡さん，山本さんには陰ながら研究生活を支えていただき，感謝しています．

鈴木君，鎌田君，水田君，原君，松本君，お疲れ様でした．この一年間同じ研究室で一緒に頑張ることができ，とても嬉しく思います．特に同じ 2 号館で研究生活を共にして来た鈴木君には，書類の苦手な私を何度も窮地から救ってもらい，感謝しています．

参考文献

[Buschka 2000] P. Buschka, A. Saffiotti and Z. Wasik. “ Fuzzy Landmark-Based Localization for a Legged Robot, ”*Proc. of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000),* pp. 1205-1210, 2000.

[Fox 1999] Dieter Fox, Wolfram Burgard, Frank Dellaert and Sebastian Thrun. “ Monte Carlo Localization: Efficient Position Estimation for Mobile Robots, ” *In Proc. of the Sixteenth National Conference on Artificial Intelligence (AAAI-99),* Orlando, Florida, pp. 343-349, 1999.

[Hanek 2000] Robert Hanek and Thorsten Schmitt. “ Vision-Based Localization and Data Fusion in a System of Cooperating Mobile Robots, ”*Proc. of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000),* pp. 1199-1204, 2000.

[Hugel 2000] Vincent Hugel, Patrick Bonnin and Pierre Blazevic. “ Reactive and Adaptive Control Architecture Designed for the Sony Legged Robots League in RoboCup 1999, ”*Proc. of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000),* pp. 1032-1037, 2000.

[Lenser 2000] Scott Lenser and Manuela Veloso. “ Sensor Resetting Localization for Poorly Modelled Robots, ”*Proc. of IEEE International Conference on Robotics and Automation (ICRA-2000),* pp. 1225-1232, 2000.

[Mitsunaga 2000] Noriaki Mitsunaga and Minoru Asada. “ Observation strategy for decision making based on information criterion, ”*Proc. of the fourth RoboCup Workshop,* Melbourne, Australia, 2000.

[Veloso 1998] Manuela Veloso, William Uther, Masahiro Fujita, Minoru Asada and Hiroaki Kitano. “ Playing Soccer with Legged Robots, ”*Proc. of the*

1998 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '98), pp. 437-442, 1998.

[Veloso 1999] Manuela Veloso and William Uther. “ The CMTrio-98 Sony Legged Robot Team, ”*Minoru Asada and Hiroaki Kitano, editors, RoboCup-98: Robot Soccer World Cup II,* pp.491-497, Springer Verlag, Berlin, 1999.

[Wendler 2000] Jan Wendler, Steffen Brüggert, Hans-Dieter Burkhard, and Helmut Myritz. “ Fault-tolerant self localization by case-based reasoning, ”*Proc. of the fourth RoboCup Workshop,* Melbourne, Australia, 2000.

[浅田 1997] 浅田 稔, 國吉 康夫, 野田 五十樹, 北野 宏明 :“ 研究活動とロボットコンテスト（RoboCup）, ”日本ロボット学会誌, Vol. 15, No. 1, pp. 1001-1004, 1997.

[浅田 2000] 浅田 稔, 北野 宏明 :“ ロボカップ戦略：研究プロジェクトとしての意義と価値, ”日本ロボット学会誌, Vol. 18, No. 8, pp. 1081-1084, 2000.

[大西 1998] 大西 博之, 鈴木 寿, 有本 卓 :“ ハフおよびフーリエ変換を用いた拡大・回転・平行移動検出法の部品位置決めへの応用, ”日本ロボット学会誌, Vol. 16, No. 2, pp. 232-240, 1998.

[久保田 1989] 久保田 孝, 橋本 秀紀, 原島 文雄 :“ ローカルプランニングによる移動ロボットの経路探索, ”日本ロボット学会誌, Vol. 7, No. 4, pp. 267-274, 1989.

[久米 1997] 久米 正夫 :“ カメラキャリブレーションとその評価法について, ”日本ロボット学会誌, Vol. 15, No. 2, pp. 183-186, 1997.

[谷内田 1990] 谷内田 正彦, 石黒 浩 :“ ロボットの視覚, ”人工知能学会誌, Vol. 5, No. 6, pp. 720-728, 1990.

[横井 2001] 横井 真浩 :“ 四脚ロボットにおける観測コストを考慮したナビゲーション, ”平成 13 年度修士論文, 東京大学, 2001.

[Horn 93] Berthold K. P. Horn(1986), NTT ヒューマンインタフェース研究所　プロジェクト RVT 訳 :“ ロボットビジョン, ”朝倉書店, 1993.

[八木 1992] 八木 伸行, 井上 誠喜, 林 正樹, 中須 英輔, 三谷 公二, 奥井 誠人, 鈴木 正一, 金次 保明 :“ C 言語で学ぶ実践画像処理, ”オーム社, 1992.

[吉川 1988] 吉川 恒夫 :“ ロボット制御基礎論, ”コロナ社, 1988.

付 録A　Legged Robot League

A.1　フィールド

Legged Robot League で使用されるフィールドは Fig.A.1 のようなものである．また，寸法は Fig.A.2 のようになっている．本研究に関係する特徴について説明する．

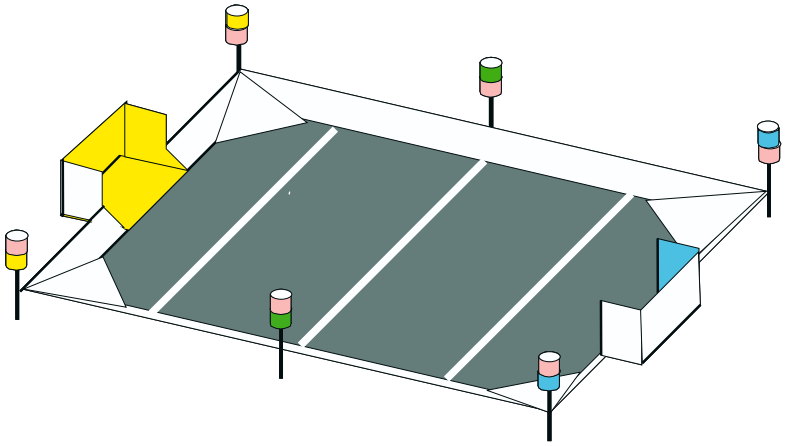

Fig.　A.1　フィールド

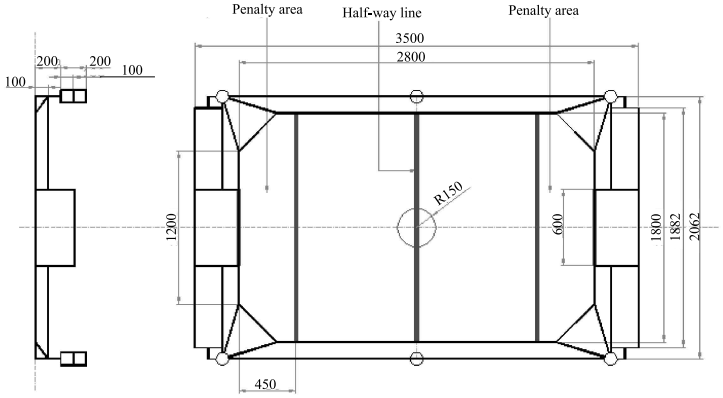

Fig.　A.2　寸法

A.1.1　壁

壁は，ロボットやボールがフィールド外に出ることを防ぐ目的で設置されている．壁には傾斜がつけられており，ここにロボットが乗り上げると，転倒の原因

となる．また，デッドレコニングに悪影響を及ぼす．

A.1.2 ライン

ハーフウェイライン・ペナルティーライン２本の計３本が描かれている．線の幅は 25[mm]．ラインに関するルールとして“ Two Defenders ”がある．これは，自陣ペナルティーエリア内に２台以上のロボットが入ってはならないというもので，後から入ろうとするロボットは，ペナルティーラインを２本の脚が越えた時点でハーフウェーラインの一端に置き直される．

A.1.3 ランドマーク

円筒形で，フィールドの周囲に６本立っている．お互いに識別できるようにスカイブルー・グリーン・イエロー・ピンクの４色で塗り分けられている．

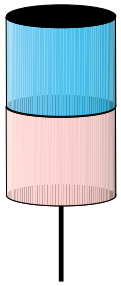

Fig. A.3 ランドマーク

A.1.4 センターサークル

Fig.A.2 では，センターサークルが描かれることになっているが，実験用のフィールドには，問題の単純化のため，描かない．

A.1.5 外部のノイズ

公式のフィールドの外側には白い柵があり，ロボットの視界に観客が入らないようになっているが，ロボットが顔を上げると人の顔が視界に入る．また，実験用のフィールドは，ノイズに対してはほとんど配慮がされていない．

A.2 ERS-1100

ERS-1100 は Fig.A.4 のような四脚ロボットである．このロボットの主な特徴について説明する．

Fig. A.4 ERS-1100

A.2.1 CPU，メインメモリ

CPU は 100MHz，64 ビット RISC プロセッサ（MIPS R4300）．メインメモリは 16MB.

A.2.2 CCD カメラ

固定焦点，18 万画素．しかし，処理速度の都合上，カラー画像として出力されるのは水平方向 88 画素，垂直方向 60 画素の計 5280 画素である．各画素のパラ

メータは 256 階調の YUV 値で表される．また，画角は水平方向 53[deg]，垂直方向 41[deg].

CCD カメラの設置されている位置は，ERS-1100 の頭部の鼻先である（Fig.A.5）．頭部には３つの関節があり，CCD カメラの位置・姿勢は，この３つの関節によって変化する（Fig.A.6）．

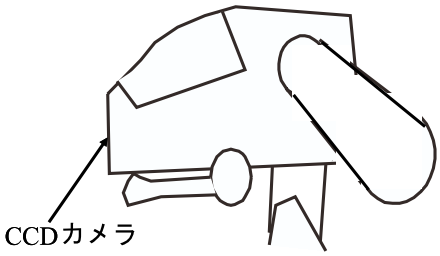

Fig. A.5 CCD カメラの位置

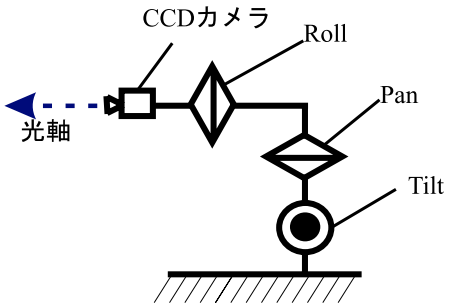

Fig. A.6 頭部の関節と CCD カメラの関係

A.2.3 CDT

CDT は各画素の YUV 値を閾値処理して色を検出するハードウェアである．出力は C 画像と呼ばれ (Fig.A.7)，各画素ごとに 1 BYTE のデータを持つ．1 BYTE の各ビットは，下の桁からオレンジ・グリーン・ダークグリーン・スカイブルー・イエロー・ピンク・ダークブルー・レッドに割り当てられており，画素の YUV 値が閾値に入っていると 1，入っていないと 0 となる．C 画像の例を Fig.A.8 に示す.

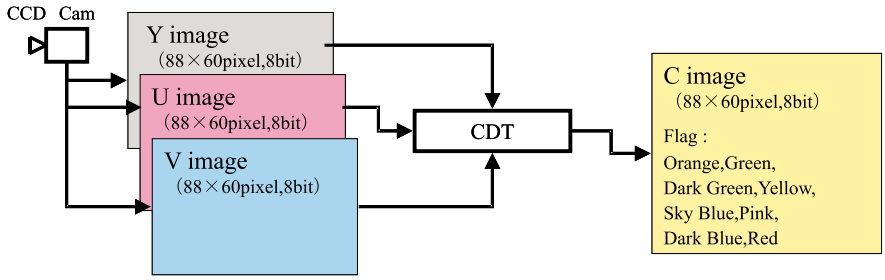

Fig. A.7 CDT

Fig. A.8 C 画像

A.2.4 自由度

18 自由度．本研究に関係するのは，CCD カメラの説明で述べた頭部 3 自由度と各脚 3 自由度の計 15 自由度である．

A.2.5 ステッカー

試合では，チームを識別するためにレッドとダークブルーのステッカーを貼る．