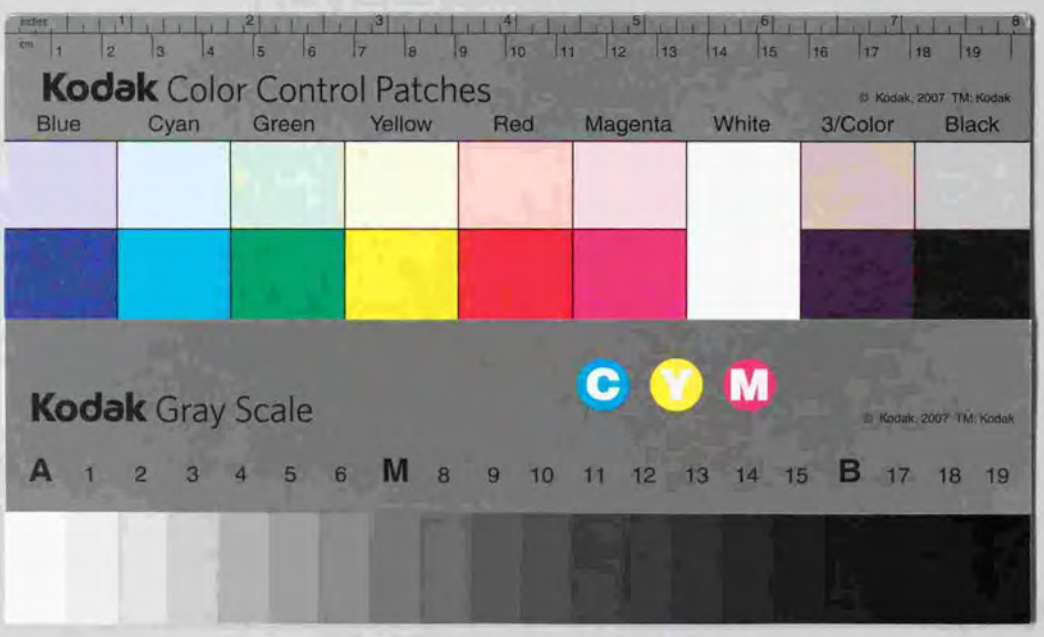
ビューベーストアプローチによる移動ロボットの経路誘導に関する研究

宋 勇豪



ビューベーストアプローチによる移動ロボットの経路誘導に関する研究

平成 10 年 3 月

松本 吉央

目次

1 序論 1

1.1 研究の背景と目的 3

1.2 本論文の構成 3

2 ビューベーストアプローチによる移動ロボットの経路誘導 7

2.1 視覚移動ロボットの経路誘導 9

2.1.1 移動ロボットの視覚 9

2.1.2 視覚移動ロボットのための地図 13

2.2 ビューベーストアプローチによる認識 19

2.2.1 ビューベーストアプローチ 19

2.2.2 ビューベーストアプローチの特徴 20

2.2.3 ビューベーストアプローチによる物体認識 21

2.2.4 ビューベーストアプローチによるシーンの認識 22

2.3 ビューシーケンスによる経路誘導 25

2.3.1 ビューシーケンスの概要 25

2.3.2 ビューシーケンスのマッチング 28

2.3.3 画像空間上でのビューシーケンス 29

2.3.4 マッチングエラーの推移 31

2.3.5 ビューに必要な性質 31

2.3.6 ビューシーケンスの特徴 34

3 視覚移動ロボットのシステム開発 37

3.1 フィールド画像混合を利用したステレオ視覚システム 39

3.1.1 画像の多重化 39

3.1.2 フィールド単位での画像の多重化 40

3.1.3 フィールド多重化回路の実装 46

3.1.4 評価実験 48

3.2 PC ベースの汎用ロボットカーネル 54

3.2.1 ロボットのプラットフォーム 54

3.2.2 汎用ロボットカーネル：ハイパーマシン 54

3.2.3 PC ベースのハイパーマシン 55

3.2.4 ロボット用インターフェースボード 58

3.2.5 ハイパーマシンを用いたロボットの構築例 61
3.3 実験システム 62
3.3.1 屋内用実験システム 62
3.3.2 屋外用実験システム 66

4 ビューシーケンスに基づく経路誘導 69
4.1 ビューシーケンス 71
4.1.1 ビューの生成 71
4.1.2 ビューのマッチング 72
4.1.3 ビューシーケンスの生成 77
4.1.4 ビューシーケンスによる経路表現 79
4.2 ビューシーケンスに基づく認識 81
4.2.1 自己位置の認識 81
4.2.2 障害物の検出 83
4.3 ビューシーケンスに基づく行動 85
4.3.1 直進 86
4.3.2 右左折 87
4.4 経路誘導実験 88
4.4.1 経路教示 88
4.4.2 経路誘導 89
4.4.3 特徴が少ない環境での走行 93
4.5 考察 95
4.5.1 ビューの解像度 95
4.5.2 平滑化の効果 98
4.5.3 走行精度 101

5 全方位ビューシーケンスに基づく経路誘導 103
5.1 全方位ビューシーケンス 105
5.1.1 全方位ビューの生成 106
5.1.2 全方位ビューのマッチング 112
5.1.3 全方位ビューシーケンスの生成 114
5.2 全方位ビューシーケンスに基づく認識 117

5.2.1 全方位ビューシーケンス内の位置推定 117
5.2.2 経路に垂直な方向のずれの検出 118
5.3 ビューシーケンスに基づく行動 120
5.3.1 前進 120
5.3.2 右左折 121
5.4 経路誘導実験 122
5.4.1 教示走行 122
5.4.2 自律走行 122
5.5 全方位ビューシーケンスによる改善点 124
5.5.1 位置誤差を与えた場合の走行軌跡 124
5.5.2 マッチングの安定性 124
5.5.3 双方向の経路誘導 127

6 自律的な地図表現の獲得 129
6.1 地図表現 131
6.1.1 経路表現と地図表現 131
6.1.2 地図表現の獲得の手順 132
6.2 ロボット前方の空間の検出 133
6.2.1 テンプレートマッチングによるステレオ視 133
6.2.2 ステレオマッチング 134
6.2.3 複数解像度のマッチングの統合 135
6.2.4 空間の検出 139
6.2.5 走行実験 142
6.3 ロボット側方の空間の検出 144
6.3.1 全方位画像上の運動視差 144
6.3.2 交差点の検出実験 146
6.4 自律的地図獲得 148
6.4.1 地図作成の手順 148
6.4.2 地図を用いた自律移動 152
6.4.3 地図獲得実験 154

7 仮想環境を用いたビュー・シミュレーション 161
7.1 ビューのシミュレーション . . . 163
7.2 ビュー・シミュレーション・システム . . . 166
7.2.1 ハードウェア構成 . . . 166
7.2.2 ソフトウェア構成 . . . 168
7.3 仮想世界のモデリング . . . 171
7.3.1 環境モデル . . . 171
7.3.2 ロボットの運動モデル . . . 171
7.4 廊下環境のモデリング . . . 171
7.4.1 幾何モデル . . . 171
7.4.2 テクスチャの取り込み . . . 172
7.5 走行シミュレーション . . . 176
7.5.1 システムの評価実験 . . . 176
7.5.2 走行経路の予測 . . . 177
7.5.3 実ロボットでの走行精度の改善 . . . 178
7.5.4 考察 . . . 180
7.6 オンライン・ビュー・シミュレーション・システム . . . 181
7.6.1 オンラインのシミュレーション . . . 181
7.6.2 ハードウェア構成 . . . 183
7.6.3 ソフトウェア構成 . . . 184
7.7 オンライン経路生成 . . . 186
7.7.1 走行手順 . . . 186
7.7.2 幾何モデル . . . 186
7.7.3 テクスチャの生成 . . . 186
7.7.4 テクスチャの更新 . . . 187
7.7.5 仮想教示走行 . . . 188
7.7.6 走行実験 . . . 189

8 屋外環境における経路誘導への展開 191
8.1 屋外でのビュー . . . 193
8.2 エッジ画像による経路誘導 . . . 194
8.2.1 エッジ画像の生成 . . . 194
8.2.2 エッジ画像のマッチング . . . 196

8.2.3 エッジ画像を用いた経路誘導 . . . 202
8.3 色相画像を用いたビュー . . . 204
8.3.1 色相画像の生成 . . . 204
8.3.2 色相画像を用いたマッチング . . . 205
8.3.3 色相画像を用いた経路誘導 . . . 206
8.4 視差画像を用いたビュー . . . 209
8.4.1 視差画像の生成 . . . 209
8.4.2 視差画像のマッチング . . . 210
8.4.3 視差画像を用いた経路誘導 . . . 215

9 結論および考察 217
9.1 結論 . . . 219
9.2 考察 . . . 221

参考文献 229

第 1 章

序論

1.1 研究の背景と目的

これまでのロボットはごく限られた環境でしか実用化されておらず，今後はより高度な認識および行動能力を持つロボットが，その適用範囲をオフィスや家庭など人間の生活する環境にまで広げて行くことが求められている．ロボットの行動のためには，外界の情報をセンサから獲得し，自己位置や周囲の状況を認識する必要がある．外界の認識には，人間の場合と同様に視覚が最も重要な役割を果たしうると考えられ，コンピュータビジョンの分野では，以前から視覚(2次元画像)を用いた3次元情報の認識アルゴリズムの研究が盛んに行われてきた．しかし一般にこのアプローチは不良設定問題を含み，計算量が大きく実時間性を確保することが難しいだけでなくキャリブレーション誤差やノイズに弱い．

このような背景から，近年3次元情報を用いずに2次元画像をそのまま認識に用いる“ビューベースト”と呼ばれるアプローチが注目されている．このアプローチは，あらかじめ認識対象の様々な見え方を記憶し，画像の照合により認識を実現するというものである．線分抽出，領域分割，3次元情報の復元といった画像の解釈は行なわず，従来のアプローチよりも2次元画像をより直接的に利用する．3次元モデルを必要としないため，実環境の複雑な対象の認識にも適用することができると期待される．本研究では，このビューベーストアプローチに基づき，新たな移動ロボットのための環境表現およびそれを用いたナビゲーション手法を提案し，実環境での走行を実現することを目的とする．

1.2 本論文の構成

本論文は全9章から構成される．以下に各章の概要について述べる．

第1章「序論」では，本研究の背景と目的，および本論文の構成について述べる．

第2章「移動ロボットにおける視覚に基づく環境認識」では，まず従来の視覚を持つ移動ロボットの研究の流れについて考察し，これまでの移動ロボットのために提案された地図および経路表現における問題点を整理する．次に近年コンピュータビジョンの分野で注目されている，3次元情報を用いずに2次元画像をそのまま認識に用いる“ビューベーストアプローチ(見え方に基づく方法)”について検討し，移動ロボットのためのビューの記憶に基づく環境認識手法を提案する．ビューの記憶に基づく環境認識という考え方は，本研究を通して一貫して用いられるものであり，ここではこのアプローチで鍵となる「どのようなビューを記憶するか」「どのような照合(マッチング)の手法を用いるか」について検討する．

第3章「視覚を有する移動ロボットの開発」では本研究で用いる屋内実験用と屋外実験用の2台の移動ロボットの概要，およびそれらを構築するにあたり開発した以下の2つの技術について述べる．一つは，ロボットの共通のカーネル部分と個々のロボットに依存する部分に切

り分け，カーネル部分を標準化することで，実験用ロボットおよびソフトウェアの開発およびメンテナンスを効率的に行なうことができるようにすることを目的としたPCベースの知能ロボット汎用カーネルである．もう一つは，コンパクトなステレオ視覚システムを実現するためのフィールド多重化である．これは，2つのビデオ信号をフィールド(1/60s)毎に切替えることで1つのビデオ信号に合成し，画像処理システムに入力する手法である．これにより通常の単一の画像処理システムにステレオ画像を入力し立体視等の処理を行なうすることが可能となり，移動ロボットに搭載できるコンパクトで高度な視覚を実現できる．

第4章「ビューシーケンスに基づく経路誘導」では，移動ロボットのためのビューベーストアプローチに基づく経路表現として“ビューシーケンス”を提案し，ロボットの経路誘導を行なう手法について述べる．これまでの移動ロボットの研究においても，経路の見え方を記憶や学習に利用している例はあったが，本手法ではリアルタイムに経路上のどこにいるのかを認識し，同時に誘導を実現することを可能とした．経路モデルとしては，ビューシーケンス(走行時に見える進行方向の視野全体の画像の列)を用い，画像のマッチングには，ハードウェアによる高速なテンプレートマッチングを利用する．走行時には，記憶したビューシーケンスと現在のビューのマッチングにより環境認識を行ない，1)明るさなど，環境の見え方が時間によって大きく変化しない，2)カメラの視野に，ある程度近くまでの走行環境（床，壁など）がうつっている，という条件を満たす建物内等の環境において，1)経路上の自己位置および障害物の認識と走行制御が実時間で可能，2)一回の記録走行により容易に経路教示が行なえる，という特徴がある．

第5章「全方位ビューシーケンスに基づく経路誘導」では，前章で提案したビューシーケンスの問題点として1)一方向(片道)の経路しか表現できない，2)環境の変化が大きくなるとマッチングに失敗しやすい，3)経路への追従性が低い，という点を挙げ，これらを解決するために，全方位視覚センサを利用して生成する経路表現“全方位ビューシーケンス”を提案する．

全方位視覚センサとしては回転双曲面鏡を用いる．得られた画像は円筒面状のパノラマ画像に変換され，テンプレートマッチングを用いてマッチングが行なわれる．全方位ビューシーケンスはロボットの周囲360度の情報を含んでいるため，一つの経路表現で双方向の移動が可能となる．また，これまでのビューシーケンスよりも視野が広いために含まれている情報が多く，環境の変化に強いマッチングを実現できる．さらに，全方位視覚センサを用いて複数(例えば前方と後方の2つ)のマッチングを行なうことで，複数のカメラを持つのと同じ効果が得られため，定性的な制御方法でもこれまでより正確な誘導を行うことが可能となる．

第6章「自律的な走行環境の獲得」では，ロボットが視覚により自律的に環境を動き回り，建物内の環境表現(地図)を獲得する手法について述べる．前章までで述べた経路誘導の手法では，あらかじめオペレータが何らかの形でロボットを移動させることで経路のビューを教示する必要があった．また，ある地点からある地点までの経路を教示するのは容易であるが，建物全体の地図を作成するのは手間がかかる作業であった．しかし，ロボットが自律的に環境を動き回ることができれば，自動的に地図を生成することが可能となる．

ここでは，まずはじめに両眼立体視や全方位画像上のオプティカルフローにより実現した，走行可能な空間を検出する手法について述べる．この機能を用いてロボットは建物内の自律移動を行ないながら，同時に全方位ビューシーケンスを記憶していくことで，建物内の環境全体の地図を獲得する．地図を獲得した後には，ロボットはゴールを指示されると獲得した地図から経路を生成して，自律走行することが可能となる．

第7章「仮想環境を用いたビューシミュレーション」ではグラフィックスワークステーションと画像処理システムを組み合わせて開発した，見え方を利用する移動ロボット用のシミュレータについて述べる．

通常の移動ロボットのシミュレータでは，環境は2次元平面上に表現され，ロボットの認識はその平面上の幾何情報を得ることでシミュレートされる．しかし本研究ではビューベーストアプローチにより，ロボットは環境の幾何情報を介しないで環境のビューを直接利用してロボットの行動を実現しており，これまでのシミュレータで行動をシミュレートすることができない．

そこで，本章ではグラフィックスワークステーションを用いてビューを生成し，画像処理システムを用いてその画像を処理して仮想的な行動を行ない，ビューを更新する，という処理ループを持つ視覚移動ロボット用のシミュレータ“ビューシミュレーションシステム”の構築と評価実験について述べる．次に仮想環境内で教示走行を行なうことで，実際の教示なしに実ロボットが経路を計画しながら走行を行なうことが可能となることを示し，最後にロボットが見え方を想像しながら行動する“オンラインビューシミュレーション”という概念を提案する．

第8章「屋外環境における経路誘導への展開」では，屋外のように明るさが変化する環境でも経路誘導を実現することを目指し，ビューベーストアプローチにおけるビューの概念を拡張する．前章までの経路表現で用いていた濃淡画像というビューは，明るさの変化に弱いという欠点があった．そこで，濃淡画像よりも明るさの変化に強い画像をビューとして利用することを試みる．

具体的には微分画像，色相画像，両眼の視差画像等をビューとして利用する．これらの画像は線分抽出，領域分割，3次元情報の復元といった画像の解釈を行なわないというビューベーストアプローチの基本的な考え方に沿ったものである．本章では，これらのビューを用いることで屋外の経路誘導を実現し，ビューベーストアプローチの可能性を示す．

第9章「結論および考察」では，これまでの各章で展開した議論を総括し，結論を示す．また本研究の成果と問題点について考察し，ビューベーストアプローチの今後の展開について議論する．

第 2 章

ビューベーストアプローチによる移動ロボットの経路誘導

2.1 視覚移動ロボットの経路誘導

2.1.1 移動ロボットの視覚

これまでのロボットではごく限られた環境における行動しか実用化されていない．今後はより高度な認識および行動能力を持つロボットが，その適用範囲をオフィスや家庭など人間の生活する環境にまで広げて行くことが求められている．ロボットに必要となる最も基本的な行動の一つに，「移動」が挙げられる．ロボットの移動のためには，外界の情報をセンサから獲得し，自己位置や周囲の状況を認識する必要がある．

外界を認識し，ロボットの自己位置を推定するという問題に対し，誘導ケーブルや特殊なマーカー，あるいは灯台のような設備を環境中に設置することで解決するものがあったが [1]，これらの手法は設備の設置や保守の手間が問題となる．また，超音波 [2]，光パターン [3]，偏向光 [4]，レーザー光 [5] などを投影することで3次元認識を行なう手法も古くから研究されているが，これらのアクティブセンサを用いる方法は複数のロボットが存在する環境では信号が干渉し，正しい認識ができないてしまうという問題がある．そのため今後ロボットが社会に広まっていくことを想定した場合には，受動的な視覚センサによる認識能力が不可欠と考えられるため，本研究では受動的な視覚センサ（ビデオカメラ）を搭載したロボットの移動を対象とする．

上に述べた理由のように，ロボットにおける外界の認識には，人間の場合と同様にロボットでも視覚が最も重要な役割を果たしうると考えられており，ビジョンの研究はこれまで盛んに行われてきた．例えば単眼視覚では，それだけを用いて環境の3次元形状を復元することはできないが，対象の持つ様々な物理的拘束条件を用いることで3次元情報を推定することができる．そのようなアルゴリズムとして shape from X （X には contour, texture, shading 等が入る）で総称される研究が，に行われてきた [6]．このように，人間の持つ視覚による画像認識の機能を機械で実現しようという研究アプローチは「コンピュータビジョン」と呼ばれる．Marr が提案した視覚の計算理論はコンピュータビジョンの研究方法に大きな影響を与え，画像認識は様々な視覚モジュールに細分化され，モジュールに閉じた形で研究される傾向が強まった．このアプローチに沿った研究は，幾何学，光学，最適化問題などの手法を駆使し，学問的には大きな成果をあげている．しかし，ロボットビジョン，特に移動ロボットのビジョンでは実環境において実時間で必要な情報を得ることが不可欠であり，画像全体から詳細な対象の再構成を目指すコンピュータビジョンとは，目的がやや方向が異なる．そこで，以降ではビジョンの研究のうち，移動ロボットのためのロボットビジョンの研究例を取り上げる．移動ロボットにおいて環境の情報を視覚を用いて得る手法には，大きく分けて (1) 単眼視覚を用いる方法，(2) ステレオ視覚を用いる方法，(3) 全方位視覚を用いる方法，の3つがある．

単眼視覚

静止画像の利用 静止した1台のカメラ画像からは，物体の3次元位置や大きさは定まらない．しかし，移動ロボットの研究では「物体は床面上に立っている」という仮定を用いれば，物体の位置や大きさを推定することができる．この方法は簡単であるが，物体が床面から浮いていると，推定された位置が実際よりも遠くになるため，衝突の危険が出てくる．

床面上にない物体を検出する，つまりロボットが移動できる自由空間を単眼視で認識するのによく用いられる方法として，2枚の画像を利用する方法がある．カメラにより得られた画像は，まず全て床面であるとして逆透視変換される．この画像を，ロボットの既知の移動後にもう得られた画像と比較する（差分をとる）ことで，床面より高い物体を抽出することが可能となる．このようにすると画像の対応付けが必要ないという利点があるが，逆透視変換の計算量が多い．

動画像処理 走っている車から景色を眺めると，その動きのパターンから車の動きを推定できる．このように，オプティカルフローを用いてロボットの動きや環境の3次元情報を得るための研究は盛んに行なわれている．ロボットが直進運動する場合の解析は比較的容易であるが，回転運動も含めると難しくなる．しかし，もしオプティカルフローから回転成分を分離して，直進成分だけを取り出すことができれば，そこからロボットの動きや環境の距離情報を得ることができる．オプティカルフローから直進成分を得る方法としては，運動視差を用いる方法，消失点や無限遠点から求める方法などが提案されている．しかし，オプティカルフローを用いる方法は，理論的研究はよくなされたが，実際の画像から精度良くオプティカルフローを得ることが難しいため，実際のシステムではあまり用いられていないようである [7].

ステレオ視覚

単眼画像を用いる手法は，いわゆる不良設定問題を解こうとしているわけで，いずれの手法もシーンに対して何らかの仮定を置いており，適応できる範囲が限られている．一方，複数の視点（ステレオ視覚）から撮られた画像を処理すると，対象に関する仮定を持たなくても三角測量の原理に基づき3次元情報を得ることができる．初期のステレオ視覚の研究としては，2眼ステレオ画像上の特徴点を用いる方法がある．しかし2眼ステレオの問題は2枚の画像間の対応付けが難しいことで，疎密法 [8]，動的計画法 [9] などを利用することが考えられてきた．また，対応付けを容易にするために，カメラの台数を増やし，より多くの幾何学的拘束を用いるという3眼ステレオ視 [10, 11] も考案されている．また，従来の研究ではエッジ点を単位として対応付けを行なうことが多かったが，より構造化された方法として線分を単位として対応付けを行なう研究が提案された [12]．線分は方向や長さなどの情報を持つため，点よりも

対応候補が減るので対応付けを行ないやすい．ただし，このためには線分抽出が信頼性良く行なえる必要があり，実環境で利用するには問題が多い．また，画像のパターンそのものを利用した相関による対応付けは，点や線分などの特徴を用いるものと比較して

- エッジ抽出などの前処理を行なう必要がない．
- 線分抽出のような，情報の抽象化が必要ない．
- 細かい模様やグラデーション等，線分以外のテクスチャの情報が有効に利用される．

といった利点がある反面，

- 計算量が多い．
- 探索空間が大きい．
- 適切なウィンドウサイズを決定するのが難しい．

という問題がある．しかし近年のハードウェアの進歩により計算量の問題は克服されつつあり，高性能な実時間ステレオシステムが開発されている [13].

全方位視覚

全方位視覚の獲得法 ロボットが未知環境に置かれたとき，まずその周囲の環境全体の構造を概略でよいから知りたい．そのためには，ロボットを中心とした全方位の画像が得られれば便利である．全方位の画像を得る手段としては，(1) カメラを回転させる方法，(2) 魚眼レンズを用いる方法，(3) 円錐などの回転体の鏡を用いる方法，が提案されている．

カメラを回転させる方法の利点は，全方位の視覚情報だけでなく全方位の距離情報と，カメラ回転台の回転精度に応じた正確な方位情報を得ることが可能であることである．距離情報を得るためには，1台のカメラの焦点を回転軸からずらして設置しておく（単眼全方位ステレオ），あるいは上下2台のカメラを同一の回転軸で回転させる，異なる地点で撮影した2枚の全方位画像を用いる方法がある．

カメラを回転させる方法としては，実時間性という点で問題があるため，実時間で全方位の視覚情報を得るとして，円錐，球，双曲面といった回転体の鏡に映し出された全方位の画像を，通常のカメラで撮影するというものが提案された．円錐と球の主な違いは撮像範囲である．円錐を用いた場合は，カメラを取り付けているロボットやその近傍が映らない．一方，球を用いた場合ロボットの近傍まで映るが，遠方の物体の投影が小さく歪んだものになる．双曲面はロボットの近傍を映しながら，遠方は（漸近する）円錐に近い画像を得ることができるだ

けでなく，その画像の一部を通常のカメラで撮影したのを同じ（透視変換された）画像に変換することが可能であるという特徴がある．これらの鏡を用いた実時間で全方位の視覚情報が得られるシステムの欠点は，画像が歪んでいること，そのために解像度が一定でないこと，カメラを回転させる方法に比べて全体的に解像度が低いこと，である．

全方位視覚の利用法　全方位の視覚情報を持つことは，単にカメラの視野が広がったというだけでなく，他にも幾つかの利点を生む．その一つは，全方位の視覚情報のフィードバックによりロボットを制御することで，未知環境内での正確な直線運動が実現できることである．ロボットは環境内から何らかの方法で2点を選び，その2点が全方位画像上で常に180度の間隔を保って見えるように制御を行なう．これによりロボットは正確に直線運動できる．また，全方位の視覚情報は，その構造が既知の環境内でのロボットの位置決めにも有効である．環境内の特徴点の座標が与えられている場合，ある観測点における特徴点の見え方から，その地点の座標を知ることができる．環境を床面上の2次元平面で表すなら最低3点の特徴点の位置が既知であればよいことになる．ただし，正確にロボットの位置決めを行なうには，より多くの特徴点を用いたり，3点の選び方に注意を払うことが必要である．

全方位の視覚情報は他数の移動体が存在する環境における障害物回避にも有効である．この場合は撮像時間に実時間性が要求されるため，ロボットは鏡を用いた撮像機構を搭載する必要がある．環境中には直線運動する物体と静止物体があり，その中をロボットが直線運動すると仮定すると，ロボットに衝突する可能性がある物体は，全方位画像上では常にロボットの進行方向に一定の方位で観測される．ロボットがそのような物体との衝突を避けるには，移動速度や経路を変更し，物体が一定の方位で観測されるのを避ければよい．また，Zheng[14]らは全方位画像にDPマッチングを適用することで交差点での自己位置の認識を行った．しかし計算コストが高く，実際のロボットの誘導には用いられるには至っていない．山澤ら[15]は，全方位画像を床面に平行な平面への投影に変換し，その画像をテンプレートマッチングに用いることで移動量を推定した．しかしロボットの近傍の床面の情報しか用いていないためにロボットの経路上の位置認識には適していない．ただし，ここで用いられている山澤らの開発した全方位視覚センサは，ロボットの全周囲の画像情報を一度に得ることができるという点で，見え方に基づきリアルタイムで認識を行うロボットに適したセンサであると言える．

2.1.2 視覚移動ロボットのための地図

ロボットを目的地まで経路に沿って誘導する際，周囲の環境に対する知識をあらかじめ記憶していないと試行錯誤的に動き回って目的地を探索するしかなく，非常に効率が悪い．円滑な移動を実現するためには，何らかの形でその走行環境に関する知識を記憶しておくことが必要となる．このような環境表現に関する知識を広い意味での「地図」と呼ぶことにする．

移動ロボットにおける地図の役割は，センサ情報を用いて (1) 行動をプランニングするために自分の経路上の位置を同定すること，(2) ステアリング等の制御量を生成するために経路からのずれを検出すること，の2つに分けられる．以下にこれまでの研究で用いられてきた移動ロボットのための地図の形態を分類した上で，本研究で提案するビューシーケンスの特徴を述べる．

幾何学的モデル

2次元モデル　古くから研究が行なわれている，デッドレコニングや超音波距離センサ，あるいはレンジセンサを用いた移動ロボットでは，一般に幾何学的な2次元の地形図が用いられていた[16]．これらのロボットでは，得られるセンサ情報が2次元平面上の移動軌跡あるいは壁までの距離などの幾何的な情報のみであるため，それらの情報と地図のマッチングをとるのには地形図が都合がよいからである．

視覚を有する移動ロボットでも，距離センサを併用するものでは，同じように2次元的な地図を用た研究が行なわれている．前山ら[17]は，超音波距離センサと視覚を併用して街路樹をランドマークとして検出し，地図とマッチングをとることで，屋外の歩道での走行を行なった．幾何学的な2次元形状に基づく地図情報は，建物や道路の設計図（CADモデル）などから直接作成することも可能であるが，そのようなデータが入手できない場合には人間が実際に計測する必要があり，手間がかかってしまう．

このため，2次元の環境の形状を獲得するための研究も行なわれている．Tsujiらは床面から伸びている垂直エッジの位置を計測することにより局所的な2次元のモデルを作成し，移動可能な空間を検出する研究を行なった[18]．また，レンジデータとカメラ画像データを統合して環境を解釈する[19]，画像中で注視した点を中心に垂直エッジの位置を計測し，地図を作成する[20]といった研究も行なわれている．しかし，これらの研究は移動しながら未知の環境を探索し，環境の構造を理解することを目指しており，移動ロボットを目的地へ誘導するための地図という観点に立つと必ずしも有効な表現形式が得られるとはいえない．つまり，これらの地図から自己位置を算出するためには，地図作成の際に行なった環境計測と同じ処理により周囲の位置情報をまず獲得する必要がある．そして，この位置情報とあらかじめ持っている地

図情報とのマッチングをとり，その場所に該当するかを調べなければならないからである．

道路の2次元モデル CMU の Navlab の FERMI[21] という道路検出システムでは，道路の形状モデルをあらかじめ知識として持っている．画像から左右の道路境界の候補となる線分群を抽出してから3次元空間に変換し，幾何学的に並行なら道路とし，さらに道路の図面と比較する．そして，ロボットの道路に対する相対位置をを求めて道路の中心を通るように走行制御する．

3次元モデル 屋内ならば人工物に限られているため，例えばその建築物の CAD モデルがあればそのまま3次元モデルとなる．しかしこのようなモデルはそのままロボットに利用しやすい情報を持っているとは言い難い．ロボットが直接得ることができる情報は2次元画像であるからである．ただ，視覚から直接得られるのは2次元の情報であるが，

- 環境の構造に仮定を置く．
- カメラを複数台利用する．
- 動きながら連続して環境を観測する．

などの工夫により環境の3次元情報を復元することが可能となる．ロボットが環境の3次元モデルをあらかじめ持っていれば，復元した情報と照合することで，自己位置を認識することができることになる．しかし，一般にこのような3次元情報の抽出は計算量が多く，キャリブレーション誤差やノイズに弱いため，移動ロボットへの視覚処理への応用は少ない．

金出らは，最大6台のカメラを用いたビデオレート・ステレオビジョンシステムを開発した．このシステムでは多数の DSP を用い SSSD と呼ばれる手法により 128[pixel] × 128[pixel] 分解能で 7[bit] の視差を 1/30 毎に求めることができる．しかしこのシステムは規模が大きく，移動ロボットに搭載して環境の3次元情報を得るために用いることはまだ困難なようである．

Morita らは，ビデオで撮影した連続画像を用いて，撮影対象の3次元形状とカメラの動きを同時に復元する手法を開発した．彼らの手法は，画像中の特徴点を連続的にトラッキングしその座標の時系列データを特異値分解することで，対象の3次元形状とカメラの回転運動の2つに分離するというものであるが，得られる3次元形状には大きさが含まれていないので，ロボットの視覚機能としては不十分である．また，カメラの投影法にも透視変換ではなく並行投影という仮定が入っており，対象までの距離が十分に遠い場合にしか有効でないため，環境中を移動するようなロボットの環境認識には適していない．

このように視覚からの3次元的な環境情報の抽出は現実には困難であるため，逆に3次元モデルを2次元画像に投影しその画像とロボットが得る画像の間でマッチングを行う，という手法が3次元モデルを地図とする移動ロボットの経路誘導の研究では主流となっている．

坪内らは環境の特徴的なものの3次元の形状データを持ち，そこから画像を生成し，実際の画像とのマッチングを行なって位置の同定を行なった [22]．この手法では，環境内の有彩色の特徴物を (1) 色，(2) 形状，(3) 3次元位置，で定義した環境地図を作成し，その地図から見えると予想される特徴物の画像を生成する．実際に見えている画像からも特徴物を抽出して，その画像のずれから視点の位置の推定を行なうというものである．

また，Claude らは屋内・屋外環境で3次元幾何モデル情報を利用したモデルベースの視覚移動ロボットのナビゲーションを行った [23]．Kosaka らも同様に屋内環境で3次元幾何モデル情報を利用したモデルベースの視覚移動ロボットのナビゲーションを行った [24]．

これらの研究ではいずれも3次元モデルから想定されるエッジ情報や色情報といった抽象化された，絞り込んだ情報でのマッチングを行なっている．このため，マッチングの前処理として特徴抽出が重要となるが，エッジや色を安定に抽出ことは現実には難しい問題である．しかし近年，モデリング，レンダリング等 CG 技術の進歩により，3次元モデルを用いたリアルな画像が高速に生成できるようになっているため，エッジや色などの限られた情報を用いるのではなく，予想されるビュー全体を生成し，それと現実の画像を比較して自分の位置を推定する，という方向に発展する可能性があると思われる．ただしこの場合，環境の形状だけでなくその色や模様までモデル化しなければならないため，現状では3次元モデルを生成する方法にまだ難点がある．

トポロジカルな地図

廊下に沿った走行を行なうモジュールや交差点の認識を行なうモジュールがあれば，走行状況を監視するモジュールがトポロジカルな地図上でロボットの位置を更新し，次に行なう動作を決定することで目的地までの走行を実現することができる．

Meng ら [25] では，ニューラルネットにより道路に沿った走行を行なうモジュールと同時に，ランドマーク検出モジュールがドア，交差点や廊下のくぼんだ部分などの認識行ない，あらかじめ持っている環境のトポロジカルな地図の上での自分の位置を更新し，行動をプランニングする．このトポロジカルな地図には距離情報は必要なく，含まれていない．そのため地図の内容は定性的な情報だけから成り立っており，幾何学的な地図よりも製作が容易であるという利点がある．

また，Mataric [26] は SSA に基づく移動ロボットに自ら移動しながら部屋内でトポロジカルな地図を生成させるという研究を行なったが，そこで得られる地図は SSA により壁などに

沿って走行した一周の閉じた経路の形状のものに限られており，地図としての応用範囲は狭いものにとどまっている．

処理の手続き的な情報

目的地へ誘導するために必要な情報を，使いやすい形で記述した地図を用いることで，開空間，ドア，壁などの占める位置情報を記述した幾何的な地図を用いるよりも情報の切り出しを容易にし，ナビゲーションに要する処理を軽減できると考えられる．この考えに沿った地図として，ALVINN の開発で用いた Annotated Map がある [27]．この地図には移動経路中のどこで，何の処理を起動するかという手続き的なデータが記述されている．ただし，この研究では開発した各処理要素を，設定した目的地へ移動ロボットを誘導するために統合することが目的であり，このような手続き的な地図を如何にして作成するかには触れられていない．人間が制御プログラムを作成する要領でこのような地図を構築しようとすると，多数の走行経路が存在する場合には地図作成のコストが膨大となり，変更も容易ではない．そこで，小野口らは移動環境において実際に収集した画像を対話的に処理することで，多重情報地図とよばれる地図を構築するシステムを提案している [28]．また，ハイパースクーターの研究 [29] では，人間が搭乗して実際にロボットが得る画像を見ながらコミュニケーションを行なうことができるため，画像のうちのどこをどのように見るか，またそれがどのように見える時にはどのような行動を行なうか，を対話的に指示しながら走行のための知識をオンラインでロボットへ与えている．

見え方情報

見え方の記憶　教示走行時の画像を記憶することで，環境の理解の必要性を少なくすることができるはずである．このような考えに基づいて，画像を記憶して自律走行に用いようとする研究がいくつか行われている．これはビューベーストアプローチに近い考え方であるが，マッチングに特徴抽出されたものが用いられている点が異なる．

黒須ら [30] はカメラと VTR を用いて準備走行で廊下における画像を記録し，その画像中の縦線までの距離をステレオ法により計算した．自律走行時には，得られる廊下の画像中の縦線とマッチングをとることで，ロボットの廊下における位置の推定を行なった．この手法では，マッチング自体はフレームレートで行えるものの，(1)VTR をロボットに搭載することが必要，(2) 縦線だけをマッチングに用いているため，走行環境が縦線の多い廊下に限定され，またマッチングに用いる情報が少ないため安定な結果を得ることが難しい，(3) 障害物などによる環境の変化に弱い，などの問題がある．

三河ら [31] はビジュアルサーボを用い，あらかじめ教示走行において記憶した特徴点 (4

点) の写った目標画像を与えた．自律走行時には，この目標画像と現在の画像の差から，ロボットの位置・姿勢を制御できることを示した．しかし，この手法では，実際の環境下での特徴量の抽出については触れておらず，応用には問題が残る．また計算量が多いため実時間性が求められる用途には用いることができない．

Zheng [14] らは走行経路の進行方向に対してカメラを 90 度横を向け，走行中に横方向に見える画像のうち，ある縦スリット上の画像を連続して取り込んで並べた画像 "パノラミックビュー" を提案した．スリットを 2 本用いることで距離情報が得られ，この画像情報と距離情報を合わせて "経路パノラマ表現" と名付けた．また，経路パノラマ表現を用いてランドマークとして適した特徴的な領域を抽出したり [32]，環境地図獲得に用いたりしている [33]．しかし，ここで得られる距離情報は，走行経路に対するロボットの位置のずれを検出して制御を行なうことができるほど精度が高くない．つまり，経路の進行方向に対する位置は正確に認識することができるが，経路の横方向の誤差には弱い．そのため，パノラミックビューを直接自律走行に用いることはこれまでのところ行なわれていない．また，カメラは横方向を向いているため，パノラミックビューを自律走行を行なう際には別に進行方向を向くカメラが必要になり，システムが大きくなるという欠点がある．

見え方の学習　Pomerleau[34] や Meng[25] が開発したニューラルネットを用いた走行システムは，道路の見え方をニューラルネットへの入力とし，ステアリングの制御量を出力にし，人間の運転を教師として学習することで，屋外での道路へ沿った走行を実現した．

これらの走行手法は，単純で見え方の変化が少なく，通路が遠くまで見渡せる高速道路や直線廊下のような限定された環境ではロバストな誘導が実現されている．また，ニューラルネットで学習されたデータのサイズは常に一定であり，学習させた状況の量とは関係ないという点も，情報圧縮の意味では優れている．しかし，それぞれの学習手法の汎化能力に依存して走行可能な状況は制限され，路面の状態，境界，道路の形状の様々な変化に対応することは困難である．また，これらの手法はその場その場に適した制御量を生成し経路に沿って走行することはできるが，経路に沿った方向の自己位置の認識はできない．つまり，現在ロボットが経路のどのあたりを走行しているのかを認識し，ある位置で止まるなどの機能を実現するためには，他の認識手法を併用する必要がある．

ただし，これらの見え方の学習を用いる方法はいずれも低い解像度の画像しか用いていない．そのため画像の細かい部分を見なくても経路誘導は可能である，あるいは逆に粗いところしか見ないことによって処理速度を上げることができることを示しており，従来の細かく画像を分析するロボットビジョンとは異なるアプローチの可能性を示唆している．

ビューベーストアプローチ Pomerleau[34], Meng[25] のように，ニューラルネットの学習により経路誘導を行う場合，環境の複雑性が学習の汎化能力を越えてしまうことが想定されるため，適用範囲が限られる．また，見え方が似ている状況では似た行動をとる，という仮定がおかれており，文脈（今どこを走っているのか）を考慮した走行はできない．また Crespi ら [35] はメモリベーストアプローチにより見え方と通路中の位置を学習することで，廊下に沿った走行を実現した．しかしここでメモリベースで記憶しているのは廊下の中央からの変位であり，ニューラルネットで学習したのと同じ結果しか得られない．

そこで「見え方と行動の関係」を学習により一つにまとめるのではなく，経路に沿って全て記憶してしまえばよいのではないか，という考えが出て来る．こうすれば汎化の必要がなくなり，経路のどこにいるかという文脈が出て来るため，一つの（似た）見え方に複数の行動がマッピングされることも可能になる．次節では，このようにたくさんの見え方を記憶して認識に用いるアプローチ（ビューベーストアプローチ）について考察し，そのアプローチに基づき経路誘導を行なう手法を提案する．

2.2 ビューベーストアプローチによる認識

2.2.1 ビューベーストアプローチ

人工知能の分野では，“記憶に基づく推論 (Memory-Based Reasoning)” と呼ばれる推論の方法が研究されている [36, 37]．記憶に基づく推論では，いくつかの回答の中から最も適切なものを選択するような問題において，類似した問題同士の回答は同じになると仮定し，たくさんの事例の記憶を用いて適切な回答を導き出す．記憶に基づく推論は，与えられた事例の類似事例を検索したり分類したりする手法である Nearest Neighbor 法（最小近傍法）[38] の応用として位置付けられる．“記憶に基づく方法 (Memory-Based Approach)” とは，このように認識のために他のアプローチと比較してより直接的に記憶を利用する手法の総称であり，膨大な記憶と単純な処理により現実世界の複雑な問題を解こうというアプローチである．このような考え方を画像のパターン認識に適用したものが，“見え方に基づく方法 (View-Based Approach, Appearance-Based Approach，以後ビューベーストアプローチ)”[39] である．

画像を用いた認識は基本的には，あらかじめ蓄えられた物体のモデルと，入力画像から何らかの処理により抽出された特徴との比較・照合により実現される．従来の3次元物体の認識はモデルと入力特徴との照合に幾何学的な特徴を利用する，“幾何学的特徴に基づく方法 (Model-Based Approach)” と呼ばれるものが一般的であり，盛んに研究が行われてきた．このアプローチでは，まず2次元画像からエッジやコーナなどの形状特徴を抽出し，これをもとに3次元的形状を抽出する．この3次元形状をあらかじめ用意してある3次元モデルから作られた特徴との間で照合するのである．このアプローチでは物体の3次元的な回転などの変化に容易に対処できるという点は優れているものの，2次元画像から幾何学的特徴を精度よく抽出することや，そこから3次元形状を復元すること，また3次元形状のモデルを作成することは現実には必ずしも容易ではない．また，形状特徴の3次元的ではなく2次元的な位置を照合に利用する手法や，幾何学的な不変特徴を利用する手法も提案されているが，同様な理由からまだ研究途上である．

これに対して，ビューベーストアプローチでは，2次元の見え方画像をそのまま利用する．3次元物体はその取り得る2次元の見え方をあらかじめすべて記憶され，入力画像と比較することで認識される．このアプローチでは画像を，特徴抽出を行うことなしにより直接的に利用することが特徴であり，また認識対象のモデルは画像を記憶することで作成できるため，従来の3次元的な形状モデルを用いる認識手法と比較して容易である．従ってこのアプローチは，ノイズが多い実画像から複雑な形状をもつ物体を安定して認識することが期待できるが，膨大な画像データの記憶が必要となるため，そのままでは記憶量や計算量の観点からみてあまり現実的ではなかった．

しかし近年の計算機の性能向上は著しく，パーソナルコンピュータのレベルでも実メモリ容量，計算能力，記憶装置の容量が飛躍的に増大している．また画像処理専用プロセッサの開発も進んでいる．これらの計算機環境の変化により，ビューベーストアプローチが現実味を帯びてきており，画像データを安定なままに保ちながらデータ量をある程度削減することができれば有効な認識手法になり得ると考えられ，近年3次元の物体認識や移動ロボットにおけるシーンの認識のためのいくつかの具体的手法が提案されている．

2.2.2 ビューベーストアプローチの特徴

これまでに述べたように，画像認識は幾何学的特徴を照合するモデルベーストアプローチと，2次元の見かけ画像を照合するビューベーストアプローチとに大別できる．ここではそれぞれの特性についてまとめる．

モデルベーストアプローチ

- 視点の変化，物体の3次元的な回転などにより生じる画像の大きな変動を吸収することが容易である．
- 幾何学的特徴は一般に局所的な特徴であるために，物体の一部が他の物体に覆われる隠れの問題や，物体が複雑な背景の中にある切り出しの問題に対処しやすい．
- モデル辞書の容量は一般に小さい．

ビューベーストアプローチ

- 雑音の多い実画像の中から幾何学的特徴を正確に抽出することは一般に容易ではないが，ビューベーストアプローチでは画像そのものを入力とするためにこのような特徴抽出は不要となる．
- モデル辞書の容量は大きくなるものの，物体の例から辞書を作成する学習や教示は容易である．

このように，両者のアプローチは異なる特性を持っているため，それぞれの長所・短所を把握し，実際の認識問題に適用するのが望ましい．

ところで，人間が3次元物体を認識するときに，はたして2次元照合を利用しているのか，それとも3次元照合を利用しているのだろうか？これは心理学の分野でも興味がもたれている内容である．Edelman らはメンタルローテーションの心理実験により，以下の知見を示した．

> 人間が3次元物体を認識する際には複数の方法を併用していると考えられる．ある物体がその個人にとってあまり見慣れていないような場合には3次元構造を考慮しながらモデルとの照合をとるが，よく見慣れた物体については2次元照合を行っている．

これは，日常よく見る頻度の高い物体については，人間も処理の単純な2次元照合を用いていることを示していることになり興味深い．

2.2.3 ビューベーストアプローチによる物体認識

視覚による3次元物体認識は，コンピュータビジョンの分野における主要な研究テーマの一つである．ここでは3次元物体認識にビューベーストアプローチを適用した研究として近年注目されている，ビューベーストアプローチを用いた物体認識手法について述べる．

固有空間法

固有空間（共分散行列の固有ベクトルの作る空間）は，統計処理においては主成分分析とよばれる次元圧縮法として古くから利用されてきた．この考え方をパターン認識に適用することで，多次元の特徴量を低次元化することができる．固有空間を利用したパターン認識はまず 1960 年頃に文字認識に適用された．しかし，当時は計算機の能力があまり高くなかったために次元の低い画像しか扱えなかったことと，手書き文字認識に関しては形状的な特徴を利用した認識手法と比較して高い認識精度が得られなかったことから，固有空間法に基づく2次元照合はその後の文字認識の分野では大きな流れにはならなかった．

近年になって，人間の顔の画像を固有ベクトルで表現し認識する手法が提案された [10]．顔における各部分の配置は個人によらず類似しており画像間に強い相関関係がある．そこで，少ない固有ベクトルの線形結合で効率良く顔画像を表現でき，認識することが可能となる．Turk らはその固有ベクトルのことを「固有顔 (eigen face)」と名付けた．

パラメトリック固有空間法

これまでの固有空間を利用した主な目的はクラス分けであったが，それらに対し，村瀬らは固有空間上の多様体で3次元物体の見え方を表現する手法を提案した [11]．物体の姿勢や光源位置をパラメータとした多数のサンプル画像を用い，固有空間上の多様体（曲面）で物体の見え方を表現するもので，この手法はパラメトリック固有空間法と名付けられた．認識時には，対象となる画像を固有空間上に射影し，その点と曲面の距離が最小となるパラメータを求めることで，物体の姿勢や光源位置が求まることになる．4種類の物体に対して，1800 枚の

画像をサンプルとして学習を行い，それとは異なる 1080 枚のテスト画像を用いて認識実験を行ったところ認識率は 99.8 %，姿勢推定の精度は平均 1.2 度であったという報告があり，認識の精度は物体の見え方の複雑さなどに依存するが，比較的高い精度で認識が可能となっている．認識には 10 次元程度のベクトルデータを用いており，もとの画像データと比べ大幅な記憶量の削減が実現できている．パラメトリック固有空間法は汎用の画像の表現法であり，物体認識だけでなく幅広い応用が可能であり，これまでに動画像処理，照明計画，ビジュアルサーボ等への応用が報告されている．

2.2.4 ビューベーストアプローチによるシーンの認識

ビューベーストアプローチにより，移動ロボットのためのシーンの認識を行なう手法では，まずはじめに画像を記憶するための教示走行を行なったあとで，その経路表現に含まれる画像と現在の画像の比較・対応付け（マッチング）を行なうことで認識を行なう．画像データは情報量が多いため，そのまま大量のシーンを記憶することは難しく，またシーンのマッチングの計算コストも大きい．そこで，(1) どのようなビューを記憶し，(2) どのようなマッチングを行なうか，という点が各手法によって異なる．

DP マッチングを用いたシーンの認識

Zheng[14] らは走行経路の進行方向に対してカメラを 90 度横を向け，走行中に横方向に見える画像をスキャナでスキャンするように走査し，得られる画像をパノラミックビューと名付けた．一度教示のために走行を行い，そのパノラミックビューを用いて以後の走行時に得られるパノラミックビューを DP マッチングにより対応づけ，自己位置を求めるというものである．また，交差点での進行方向の認識のために，全方位画像の DP マッチングを用いている．

ただし DP マッチングの中では見え方を直接用いるのではなく，局所的な画像特徴であるエッジを対応付けに用いているため，この研究をビューベーストアプローチに分類することは厳密には正しくないが，経路に沿った見え方をそのまま記憶し，認識に用いているという点はビューベーストアプローチに近い考え方である．

フーリエ変換を用いたシーンの認識

前田ら [42] は移動ロボットの位置推定の手法として，全方位画像をフーリエ変換したデータをマッチングに用いる手法を提案した．ロボットはあらかじめグリッド状の既知の位置において多数の全方位画像を記憶する．全方位画像はカメラを回転させることによって得ている．この画像をそのまま用いるのではなく，周波数領域にフーリエ変換した上で周波領域のデータを記憶およびマッチングに用いるのが特徴的である．

フーリエ変換することにより，全方位画像の回転による変化（位相のずれ）は影響を及ぼさなくなるため，ロボットの姿勢にかかわらず位置を求めることができる．姿勢は位置が決定した後に求めることができるため，はじめから位置と姿勢の両方を探索範囲とするよりも，少ない計算量で効率よく探索を実現することができるのである．また，用いるデータを低周波数領域に限定することでデータ圧縮をして記憶量や計算量を削減している．また，未知の位置で得られた画像を，相互にマッチングをとって類似度を求めその値に応じて並べることで，相互の位置関係を復元することも試みている．

固有空間法を用いたシーンの認識

前田らは，固有空間法を移動ロボットのシーンの認識に適用した [43]．ロボットは周囲の画像を通常のカメラで観察する．得られた画像は固有空間に投影され，あらかじめ記憶した固有空間上の画像とマッチングがとられる．しかしロボットの視野は狭いため特徴的なシーンを含んでいるとは限らなく，また似た環境が多数存在したり環境に変化がある場合も考えられるので，一度のマッチングでロボットの位置を認識することは期待できない．そこで彼らは一つの位置で複数の方向の観察を行なったり，移動しながら観察を行なったりして，その複数の観察結果を統合しする方法を提案しており，そのための行動を決める戦略についても述べている．この戦略は，あらかじめ記憶した固有空間上の画像を利用し，あいまいさをできるだけ少なくするようにカメラの向きを変えたり，移動したりするものである．

この手法は，アクティブビジョンの枠組の中でビューベーストアプローチによる効率の良いシーンの認識を目指しており，興味深いアプローチであるが，位置認識の精度は，記憶した位置から 50[cm] 離れると 50% 程度であり，現状では決して十分な精度が得られているとはいえない．

テンプレートマッチングを用いたシーンの認識

Horswill[44] は移動ロボットの位置の認識に，見え方をランドマークとして用い，そのマッチングにより建物内の場所を認識することに成功している．彼は MIT の AI lab の彼の研究室のあるフロアを認識し，案内するロボット Polly を開発した．Polly は CCD から直接 64 × 48 の画像を得て，通常は単純な画像処理により床面など建物内の特徴を抽出を行なって障害物回避をしながら走行している．そして環境内の位置の認識のためには，16 × 12 の画像を用いたマッチングを用いている．これは，あらかじめ環境内の認識したい場所（各部屋の前など）で見える画像をあらかじめ画像を記憶しておき，ロボットが自律移動しているときに見える画像が記憶した画像と一致したら，その場所にいることを認識できるというものである．

これまでランドマークには環境の中の局所的な特徴が用いられることが多かったが，この手法は特定のランドマークを用いることなく，代わりにシーン全体の画像を解像度を下げた上で用い，位置の認識を行なっているところが特徴的である．ランドマークを画像から抽出するという処理および人工的なランドマークの敷設の必要性がなくなるという利点がある．

ビューシーケンスによるシーンの認識

本研究では，ビューベーストアプローチによりシーンの認識をする手法として“ビューシーケンス”を提案する．ビューシーケンスとは，経路に沿ってロボットが進むときに見る環境の見え方（ビュー）を，ある間隔で連続的に記憶した画像の列である．ロボットは，まずはじめにオペレータの操縦により教示走行を行い，経路に沿って得られるビューを自動的に記憶することで，ビューシーケンスによる経路表現を作成する．ロボットが自律走行を行うときには，記憶したビューシーケンスの中の画像と走行時に得られる画像のマッチングを実時間で行い，ロボットの現在位置や障害物を認識し，ステアリングを制御する．本手法は，

- 明るさなど，環境の見え方が時間によって大きく変化しない
- カメラの視野に，ある程度近くまでの走行環境（床，壁など）がうつっている

という条件を満たす建物内等の環境において，

- 経路上の自己位置，進行方向および障害物を認識できる
- 実時間で走行制御を行なうことができる
- 一回の記録走行により容易に経路教示が行なえる

という特徴がある．次節では，ビューシーケンスについて詳しく説明する．

2.3 ビューシーケンスによる経路誘導

2.3.1 ビューシーケンスの概要

ここまでに述べたビューベースの物体 / シーンの認識の研究では，それが記憶したものうちのどの部分と一致するか，を認識することが目的であった．本研究では，移動ロボットの経路誘導のためのビューベーストアプローチによるシーンの認識法を提案するが，経路誘導のためのシーンの認識は，単なる物体 / シーンの認識と比べて以下のような違いがある．

1. 実時間性が不可欠である．
2. 自己位置だけでなく，障害物検出や進行方向の制御を行う必要がある．
3. 画像全体が認識対象であり，画像を切り出す必要がない．
4. ロボットの移動は連続的であるので，記憶の全てを探索対象とする必要はない．

これらの特徴を考慮した上で，本研究では移動ロボットのための新たな視覚的な経路表現として“ビューシーケンス”を提案する．ビューシーケンスとは，経路に沿ってロボットが進むときに見る環境の見え方（ビュー）を，ある間隔で連続的に記憶した画像の列である．

ビューシーケンスによる経路誘導のコンセプトイメージを **Figure 2.1** に，また処理の概要を **Figure 2.2** に，それぞれ示す．ロボットは，まずはじめにオペレータの操縦により教示走行を行い，経路に沿って得られるビューを自動的に記憶することで，ビューシーケンスによる経路表現を作成する．ロボットが自律走行を行うときには，記憶したビューシーケンスの中の画像と走行時に得られる画像のマッチングを実時間で行い，ロボットの現在位置や障害物を認識し，ステアリングを制御する．

この手法では，記憶したビューシーケンスと走行時に見えるビューの比較により環境認識を行ない，

- 明るさなど，環境の見え方が時間によって大きく変化しない
- カメラの視野に，ある程度近くまでの走行環境（床，壁など）がうつっている

という条件を満たす建物内等の環境において，

- 経路上の自己位置，進行方向および障害物を認識できる
- 実時間で走行制御を行なうことができる
- 一回の記録走行により容易に経路教示が行なえる

という特徴がある．

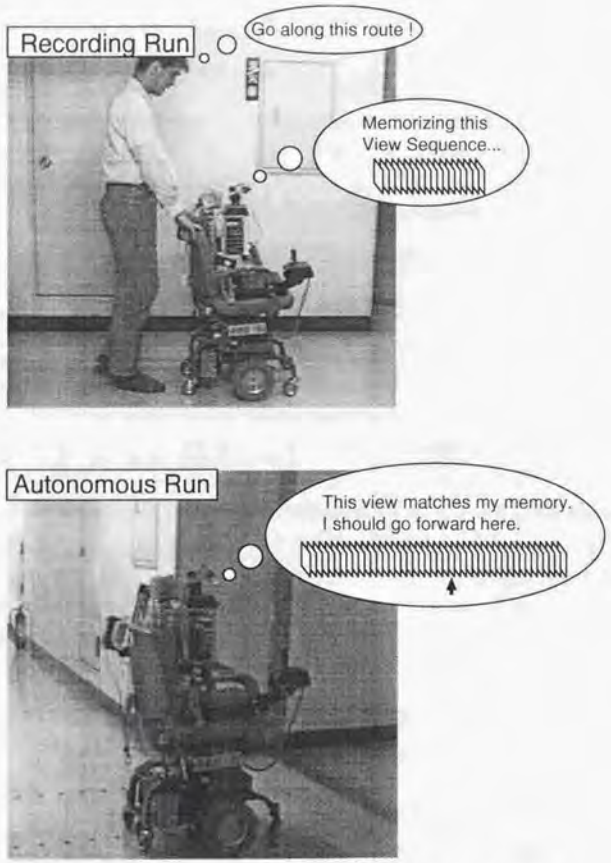


Figure 2.1 Conceptual image of processing View Sequence.

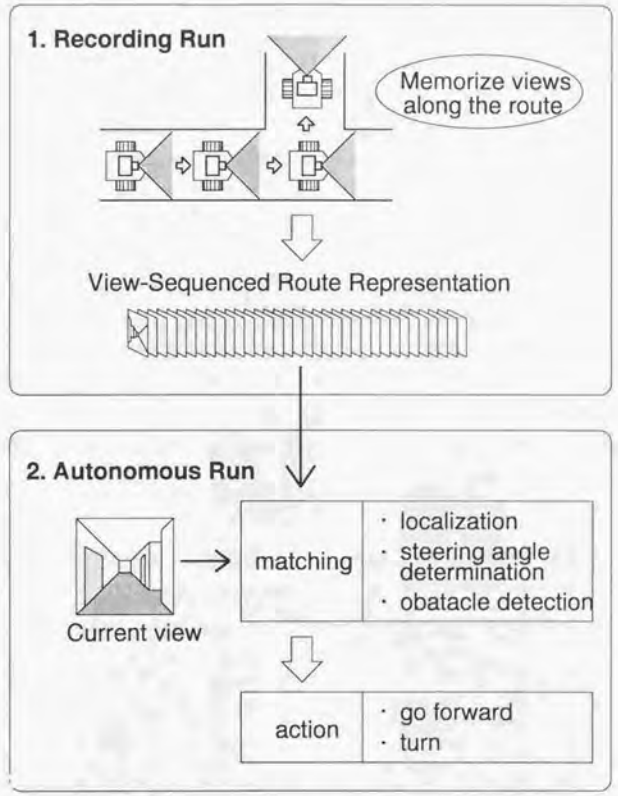


Figure 2.2 Abstract of processing View Sequence.

2.3.2 ビューシーケンスのマッチング

記憶したビューと現在のビューの 2 枚のビューを比較し，それらがどのくらい似ているかを計算する処理を，ビューのマッチングと呼ぶ．これは **Figure 2.3**に示すように，ビューの中心部の縦長の領域をテンプレートとしたブロックマッチングでマッチングエラー e は以下の式で定義される．

$$e(u) = \sum_{x=s}^{xsize-s-1} \sum_{y=0}^{ysize-1} |I_1(x,y) - I_2(x+i,y)| \tag{2.1}$$

$$e = \{\min e(u) | \ -s \leq u \leq (s-1)\} \tag{2.2}$$

ただし $xsize, ysize$ はビューの縦横のサイズ，$I_1(x,y), I_2(x,y)$ は 2 つのビュー I_1, I_2 における (i,j) の位置の画素の輝度値，s は水平方向の探索範囲である．マッチングの結果として，マッチングエラー e（ブロック間誤差，この値が小さいほど相関が高い）と，そのおよびその時の水平方向の変位 u が得られる．テンプレートが縦長であるのは，ロボットの回転や並進によりビューの変位が横方向には大きくなる可能性があるり探索範囲を設定したのに対して，ロボットの移動が平面上であるので縦方向には（理想的には）発生せず，縦方向には探索範囲が必要ないからである．

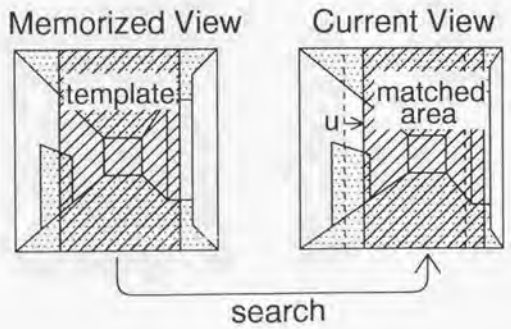


Figure 2.3 Matching of views.

2.3.3 画像空間上でのビューシーケンス

画像を n 次元の画像空間（n は画素数）上のベクトル $X = (x_1, x_2, \cdots, x_n)$ とみなす（**Figure 2.4**）．すると，2 つの画像 X, Y 間の（ユークリッド）距離 d は，

$$d = \sqrt{(x_1 - y_1)^2 + (x_1 - y_1)^2 + \cdots + (x_n - y_n)^2} \tag{2.3}$$

となる．この計算を簡単にするために，画像空間の次元を下げる手段として，Nayar らは固有空間法 [41] を用いている．ここでいう固有空間とは，画像ベクトル集合に対する共分散行列の大きな固有値に対応する固有ベクトルが張る空間である．こうすると，大きな固有値に対応する固有ベクトルだけ (上位 10 個程度) を利用してマッチングによる認識が可能になるため，記憶量やマッチングの計算量が少なくなる．これに対し，本研究では画像の次元は下げずに，画像間の距離として 1- ノルムと呼ばれる

$$d' = |x_1 - y_1| + |x_1 - y_1| + \cdots + |x_n - y_n| \tag{2.4}$$

を近似的に用い，これを 2 枚の画像のマッチングエラーと呼ぶことにする．

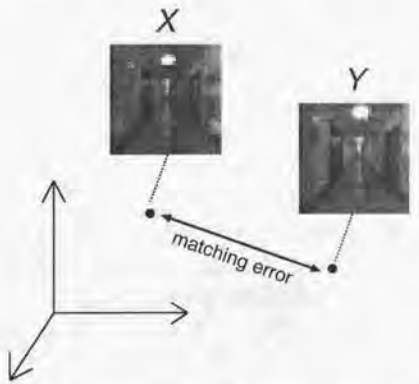


Figure 2.4 Matching Error of Views in Image Space.

このマッチングエラーをある探索範囲で計算しその最小値を求めるものが式 (2.1),(2.2) のテンプレートマッチングであり，この計算は単純でありハードウェアで高速に実行することが可能になっている [45] ことが利点である．この場合，n はテンプレートの画素数である．また，探索範囲を持つことで画像が多少ずれていても正しい画像間の距離を求めることができる

だけでなく，そのずれもマッチングの結果として得られるため，ロボットの制御に用いることができる．さらに，生の画像を記憶しているためにマッチした画像の差分をとることで，容易に変化領域を抽出することも可能となる．ただし画像の大きさ (n) は記憶量や計算量に大きく効いてくるため，慎重に決定する必要がある．

ロボットにビューシーケンスを与えるためには，一度経路に沿ってロボットを移動させる必要がある．この走行は教示走行と呼ばれる．教示走行時のビューの様子を **Figure 2.5**の上に示す．ロボットは i 枚目のビュー V_i を記憶したところとする．ロボットが移動すると，そこで得られるビュー V は徐々に V_i から離れて行く．このマッチングエラーを計算しながらある経路上を移動して，閾値を越える v_{i+1} で新たなビューを記憶する．これにより，画像空間上ではビューは等間隔に並ぶことになり，全体としてある走行経路は画像空間上では **Figure 2.5** 下のようなビューシーケンスとして表現される．

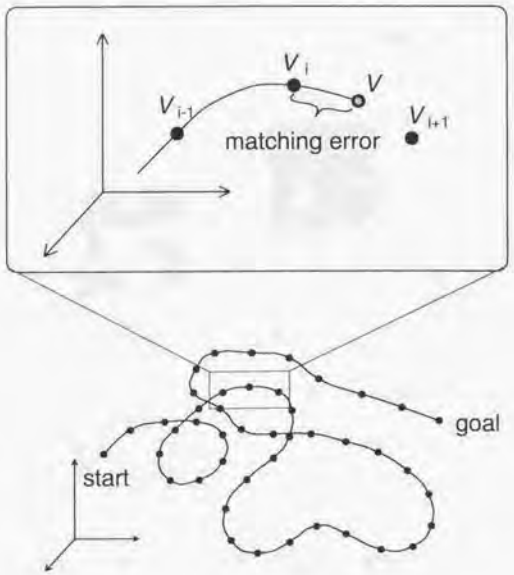


Figure 2.5 View Sequence in Image Space.

2.3.4 マッチングエラーの推移

ロボットの移動距離に比例して，マッチングエラーが直線的に増加すると仮定すると，教示走行中のマッチングエラーの推移は，**Figure 2.6**のようになる．図中の $P1, P2 \cdots$ は経路中のビューを記憶した位置，$e1, e2 \cdots$ は直前に記憶したビューとその位置でのマッチングエラーである．マッチングエラーはロボットが移動するに従って大きくなるが，その変化の仕方（傾き）は環境に依存して異なるため，ロボットの移動距離で見ると等間隔とはならない．

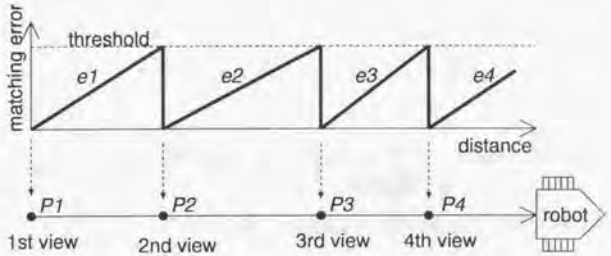


Figure 2.6 Matching Error in Teaching Run.

次に，記憶したビューシーケンスを用いて自律走行する場合を考える．マッチングエラーの増加のしかたが経路の前後方向で同じだと仮定すると，**Figure 2.6** のようにして記憶したビューシーケンスでは **Figure 2.7** のようにマッチングエラーが推移すると期待できる．ここでマッチングエラーとしているのは，全てのビューと各位置でのビューのマッチングエラーのうちの最小のものである．最小のものを探すためには，常に全てのビューをマッチングに用いるわけではない．例えば **Figure 2.7**の $P1$ 付近にいることが分かっている時には，1 枚目のビューと 2 枚目のビューだけを用いて，その小さい方がマッチングエラーだとみなせる．ロボットが移動して行き，$e1'$ よりも $e2'$ の方が小さくなったら次は 2 枚目のビューと 3 枚目のビューだけを用い，$e2'$ と $e3'$ を比較すると行った具合である．これは，マッチングの計算時間をビューシーケンスの長さに関わらず一定にするためだけでなく，**Figure 2.5** のように，画像空間内でビューシーケンスが接近しても，シーンを誤認識しないようにするためでもある．

2.3.5 ビューに必要な性質

Figure 2.6, **Figure 2.7**はマッチングエラーがロボットの移動に比例して直線的に増加

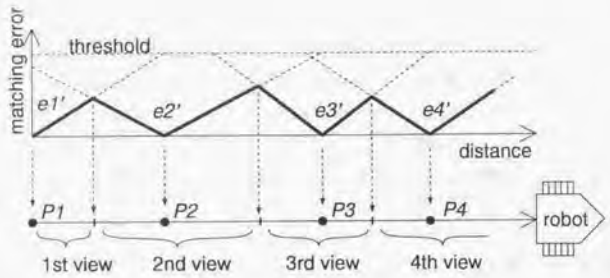


Figure 2.7 Matching Error in Autonomous Run.

すると仮定しているが，実際にはマッチングエラーはどのような増加をすれば，ビューシーケンスのマッチングは成功するかをここで考察してみる．

Figure 2.8 左は，マッチングが成功する場合の典型的なビューシーケンスを示している．自律走行時のビュー V は，記憶したビュー M_{i-1}, M_i, M_{i+1} の近傍を移動している．このとき，e_i, e_{i+1} を比較するだけで，V が M_i, M_{i+1} のどちらに近いかを決定することができる．この場合，ロボットの移動距離（画像を記憶した位置からの距離）とそのビューを用いたマッチングエラーとの関係は，単調増加になる．

次に，**Figure 2.8** 右はマッチングが失敗する場合の典型的なビューシーケンスを示している．自律走行時のビュー V は，全体としては記憶したビュー M_{i-1}, M_i, M_{i+1} の近傍を移動している．しかし，e_i, e_{i+1} を比較するだけで，V が M_i, M_{i+1} のどちらに近いかを決定することができる．この場合，ロボットの位置とそのビューを用いたマッチングエラーとの関係は，全体としては増加傾向であってもノイズが大きく，単調増加にはなっていない．マッチングが単調増加にならないと，2 つのマッチングエラーの比較によってロボットの位置を推定することができない．また教示走行時にも，実際にはロボットがほとんど移動していないのに新たなビューを記憶してしまうといった問題も起きることになる．

このように，ビューシーケンスとして利用可能なビューはマッチングエラーの変化が単調増加するもの，と言うことができる．ただし，マッチングエラーがいつまでも単調増加することはあり得ない．例えば廊下においてある位置から 100[m] 先の画像と 101[m] 先の画像をそれぞれマッチングに用いても，100[m],101[m] という距離とマッチングエラーの大小との相関関係は期待できず，偶然でしか決まらないであろう．このため，ビューシーケンスとして利用可能なビューの条件は，

ビューのマッチングエラーが，ある区間で単調増加すること

と表現できる．この単調増加する区間のうちに，新たなビューを記憶する必要があるため，この区間が長ければ長いほど，長い間隔で画像を記憶してよくなり，効率がよい．

また，上の条件を満たすセンサ情報であれば，画像そのものでなくてもビューシーケンスとして利用することができる．またノイズが多くてそのままではビューシーケンスとして適さないセンサ情報は，フィルタリングなどの前処理により利用可能な情報に変換することが可能な場合もあり得る．

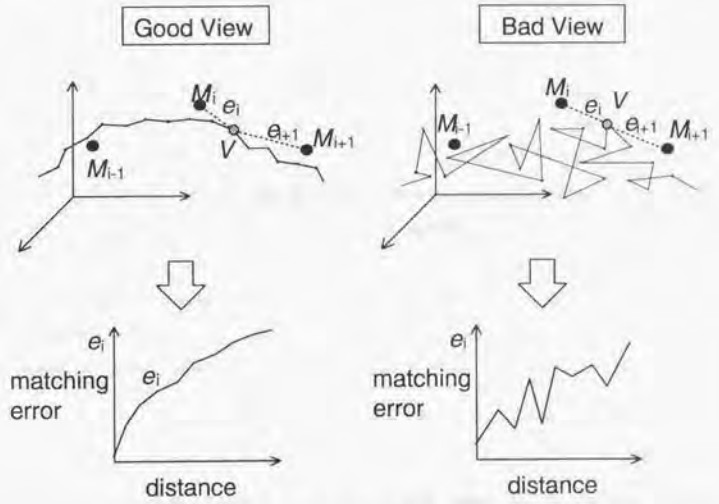


Figure 2.8 Typical Good and Bad Views.

2.3.6 ビューシーケンスの特徴

ビューの局在性

ビューシーケンスでは，経路に沿ってロボットが進むときに見る環境の見え方（ビュー）を，ある間隔で連続的に記憶する．この間隔は，一定距離ごとでも一定時間ごとでもなく，一定の見え方の変化ごとであることが特徴的である．Horswill[44] はビューを用いて「ロボットの位置があらかじめ記憶した何ヶ所かの位置のどれかと一致するか」を判断したが，このビューをいくら増やしていってもビューシーケンスと同じにはなり得ない．廊下のように似たシーンが頻繁に現れる環境では，複数のシーンが1枚の画像にマッチしてしまう可能性がある．そのため離散的に記憶したビューを全探索する Horswill の方法は環境の中で特徴的な小数のシーンの認識にしか有効ではない．ビューシーケンスは，各ビューとの見え方に一定の差が保証された上で連続的に記憶されているため，ロボットが連続的に移動すればマッチするビューは連続して遷移していくことが保証される．これは，柴田が提案した「画像特徴の区間局在性」という概念と類似している．

> ある画像特徴が特徴的であるかどうかはそれ単体では判断できず，その画像特徴が定義される空間におけるある区間において，何らかの演算子を用いて計量した時に周囲と異なるかどうかで決まる

というものである．ビューシーケンスもある区間（経路に沿った物理空間）で必要以上にたくさん記憶しても意味がない．なぜなら，それらのビューはビューシーケンスの中で周囲と比較して特徴的ではなくなってしまい，特徴的でないものをマッチングに用いても正しい認識ができないからである．

マッチングの判定

位置の誤差や画像のノイズ，環境の変化などにより，自律走行時の画像が教示時と完全に一致することはありえない．そこで，自律走行時の画像と教示時の画像のマッチングをとった結果としてマッチングエラーが得られるのであるが，そのエラーの値が閾値よりも小さければ，画像がマッチしている，つまり教示した場所にいると判断することになる．

Horswill の開発したマッチングによるシーンの認識方法では，マッチしているかどうかの判断基準がシーンに依存してしまうという問題点があった．例えば複雑なシーンと単調なシーンでは判断のための閾値が異なり，それを自動的に決定することは困難である．しかし，ビューシーケンスではどのビューが現在マッチングが取れているのか，という判断は記憶した1枚のビューだけで行われるのではなく，連続した複数のビューのうち，最もよくマッチしているものという判断基準だけを用いて実現される．

経路からのずれと障害物の検出

またビューシーケンスのマッチングは探索範囲を持つため，教示走行時と自律走行時のずれを認識することができる．例えば，自律走行時に左に寄っているとき，教示走行時に記憶したビューの中心部の画像を自律走行時の画像の中で探索すると，画像中の左に発見できる，といった具合である．さらに，記憶したビューと走行時のビューの差分をとることにより，変化領域を抽出することも容易であることもビューシーケンスの特徴である．固有空間やフーリエ空間でのマッチングを利用する認識方法では，画像をそのまま記憶してせずに圧縮してしまうために，変化領域の抽出は困難である．

第 3 章

視覚移動ロボットのシステム開発

3.1 フィールド画像混合を利用したステレオ視覚システム

3.1.1 画像の多重化

移動ロボットの視覚において，環境の 3次元的認識は基本的かつ重要な機能である．従来の研究においては，3次元的な認識を行なうためには複数のカメラと，複数の画像処理装置を繋いで用いるという形態のステレオビジョンシステムが多く用いられてきた ([46],[47] など)．しかし，複数の画像をそれぞれ個別のシステムに取り込む場合，(1) マッチング処理のための画像転送や処理結果の統合にシステム間の通信が必要となるために，オーバーヘッドが生じる，(2) カメラの数と同数の画像処理装置が必要となり，システム構成が大きくなる，という欠点がある．

これに対して，一つの画像処理システムのフレームメモリに必要な画像が全て含まれていればステレオ処理の高速化が期待できる上に，システム構成もシンプルにすることができる．近年このような考えに基づき，これまでに複数の画像を 1枚の画像内に収めて処理する研究が行なわれている．

稲葉ら [48] は，ステレオビューアと呼ばれるミラーを 1台のカメラに装着し，画面を 2分割してステレオ視を可能にするシステムを考案したが，視野が半分に狭くなる，2つの視点間の距離が構造的に制限される，という欠点があった．細田ら [49] は MaxVideo200(DataCube) を用いて 2つの入力画像を横方向に圧縮して 1枚の画像にしステレオ処理用の画像処理装置へ入力してるが，この圧縮処理のために時間遅れが生じてしまう，また視差の分解能に必要な横方向の解像度が半分になってしまうという欠点がある．他にも市販の 4画面画像合成装置を利用したステレオシステムもあるが，これも同様に横方向の解像度が半分になる．また溝口ら [50] はロボティクルームのための複数のカメラの画像をピクセル毎に多重化するシステムを試作しているが，ピクセル毎の多重化には高速な画像の切替えが必要となる．

本研究では移動ロボットの視覚処理を対象としているので，視覚処理システムはコンパクトであることが重要である．ここでは，複数のビデオ信号から 1つを選択し，それをビデオ信号のフィールド毎に切替えて画像処理システムに入力する手法 [51] を提案し，それにより実現したコンパクトで柔軟な処理が可能な視覚処理システムについて述べる．またこのシステムを用いたアプリケーションの一つとして，実時間ステレオ視に基づき人間を追従する移動ロボットの実験を行ない，本システムの有効性を示す．

3.1.2 フィールド単位での画像の多重化

NTSC 方式の映像信号は 2:1 インタレース（飛び越し走査）とよばれる走査方式を採用している．これは，2 回の走査により一つの画面を構成するもので，初めのに奇数番目の走査線（奇数フィールド）を走査し，次に偶数番目の走査線（偶数フィールド）を走査する（**Figure 3.1**）．このように 1 フレーム（1/30[s]）内で 2 度走査すると，人間の残像効果により画面のちらつき（フリッカ）を少なくすることができ，同時に前の走査でできた走査線のすき間を次の走査線が埋めるので見かけ上の解像度を上げる効果がある．しかし，この 2 回の走査に時間差が生じるため，動画に対しては映像がずれてぼやけることになるが，人間の目は静止画に比べ動画に対しては解像度が低いので問題ない．このようにインタレース方式は人間の目の特性をうまく利用している．

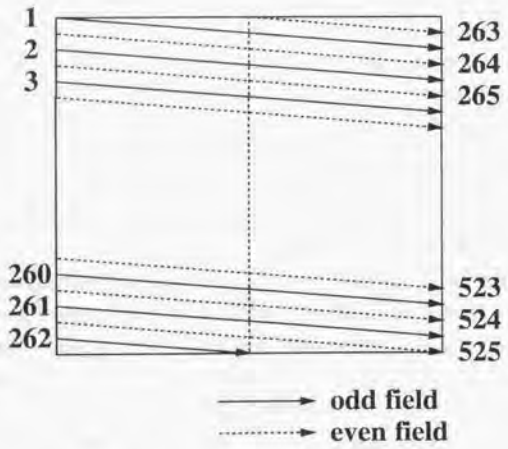


Figure 3.1 Interlace scanning in NTSC.

しかし，ロボットの視覚として NTSC 用カメラを利用する場合，このようなインタレース方式の映像に利点はない．通常（カメラにフレーム蓄積の機能がない限り）奇数フィールドの映像と偶数フィールドの映像の間には 1/60[s] の時間差があるため，それらを同時に 1 枚の映像として処理することは，精度が必要な動画像処理には適していない．つまり，このインターレースモードは，ロボットの視覚処理においてはこれまでは有効に利用されていなかった．

本節で提案する視覚処理システムでは，この 2 つのフィールドの画像にそれぞれ異なる映像を入れる「フィールド多重化」という手法によりビデオ信号を有効利用する．この映像信号の多重化には，アナログ信号の段階で行なうものと，A/D 変換した後のデジタル信号の段階で行なうものがあり，本システムでは両方を組み合わせて用いている．フィールド多重化の欠点として，モニタに出力される画像が 2 つの画像が重なっているために，人間の目には奇妙に映ることが挙げられるが，これは画像処理においては問題にはならない．

アナログフィールド多重化

アナログ多重化は，ビデオ信号のアナログの段階で複数の信号を多重化する方法である．同期のとれた複数のビデオ信号がビデオスイッチ IC に入力され，そのうちの一つが選択，出力される．

アナログ信号の段階で画像の多重化を行なう場合の利点は，画像処理装置に入力する前段階で多重化を行なうことができ，その後の処理には通常の画像処理装置をそのまま利用することができることである．回路の概要を **Figure 3.2**に示す．溝口らのシステム [50] は，アナログ段階での 1 ピクセル単位での多重化を行なっているが，画像の切替えに 12[MHz] 程度の高速性が必要となる．これに比べると，フィールド単位の多重化は画像の切替えの周波数は 60[Hz] とはるかに低速であり，回路の実現が容易になる．

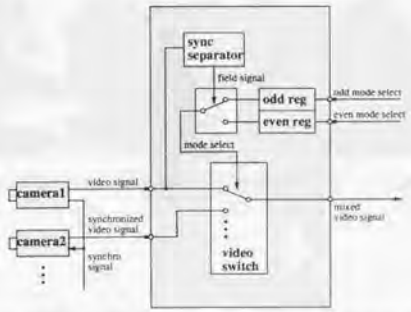


Figure 3.2 Block diagram of analog field mixing device.

本画像処理システムでは，2 台のカメラからの信号を同時に扱えるようにするためにアナログフィールド多重化回路を用いる．ステレオカメラから入力された 2 つの画像を多重化した画像および分離して取り出した画像の例を **Figure 3.3**に示す．

Figure 3.3 Mixed stereo image (1) and separated images from left camera (2) and right camera (3).

デジタルフィールド多重化

デジタル信号の段階で画像の多重化では，複数の信号をそれぞれ同時に A/D 変換した後にフィールド毎に1つを選択して VRAM に格納する．この場合 A/D 回路を入力の数だけ用意する必要があるが，これにより NTSC 形式でない映像信号も多重化することができるようになるという利点が生じる．

デジタルフィールド多重化回路を利用すると，アナログフィールド多重化と同じようにステレオ画像の多重化も実現できるが，ここでは NTSC 形式でない信号も多重化できるという利点を生かし，カラー情報(色差信号)を白黒ビデオ信号と多重化し，これまでの白黒視覚処理システムを拡張する形でカラー視覚処理システムを実現する．

これまでのカラー画像処理システムでは RGB 表色系 でカラーを表現しているものが多かったが，ここでは色差信号を用いる YCrCb 表色系を採用した．その利点としては，これまでの濃淡画像処理ボードに用いている画像 A/D 回路は Y を処理するものであるので，カラー用に追加する画像 A/D 回路が Cb,Cr 用の2つだけですむことが挙げられる．つまり回路的にも処理できる内容的にも，これまでの上位互換にできる．また YCbCr 表色系では CbCr 平面だけで色を表現でき，そのベクトルの大きさが彩度(Saturation)を，偏角が色相(Hue)を表す(**Figure 3.4**)ため，RGB 空間よりも1次元少ない CbCr 空間で色相を取り出すことができる．

デジタルフィールド多重化回路の概要を **Figure 3.5** に示す．画像は odd/even 各フィールド毎に Y(8bit), Cb(8bit), Cr(8bit), CbCr(4bit ずつ合成) の4種類のうち，どの信号を 8bit のフレームメモリに取り込むかを，フレーム毎に選ぶことができる．

また処理目的ごとの設定を Table.3.1 に示す．表のように従来通りの濃淡画像の処理もでき，また濃淡画像とカラー情報を同時に得ることもでる．デジタルフィールド多重化により，カラーと白黒の画像を多重化した場合(**Table 3.1**の Mode 3)の，入力画像と分離して取り出した画像の例を **Figure 3.6** に示す．

Table 3.1 Input mode for digital field mixing device.

Mode	processing	odd	even
1	monochrome	Y	Y
2	color	Cb	Cr
3	monochrome+color	Y	CbCr

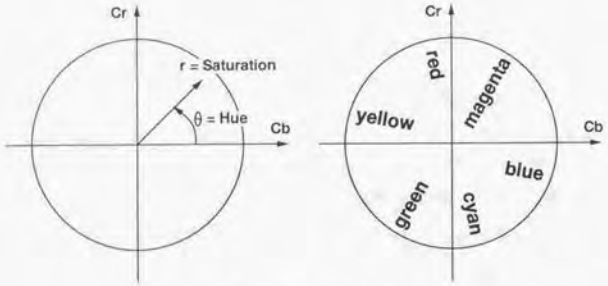


Figure 3.4 Hue in CbCr plane.

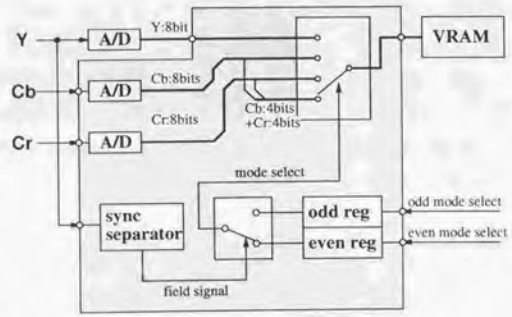


Figure 3.5 Block diagram of digital field mixing device.

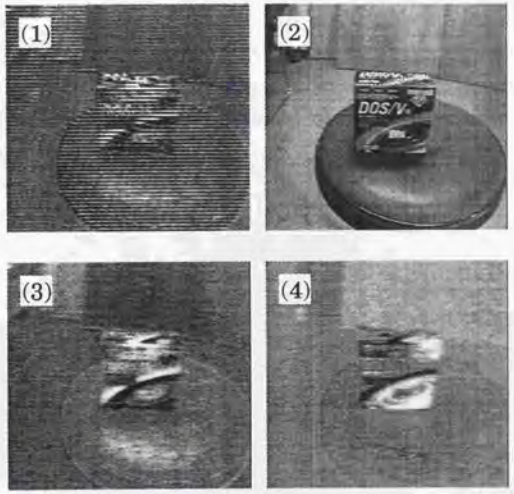


Figure 3.6 Mixed color image (1), and separated images Y (2), Cb (3) and Cr (4).

3.1.3 フィールド多重化回路の実装

フィールド多重化装置として，2つのタイプの回路を実装した．一つはアナログフィールド多重化のみを用いた「汎用多重化回路」，もう一つはアナログ，デジタル両方の多重化を用いた「ステレオ / カラー・トランスミッタ」である．

汎用多重化回路

汎用多重化回路は **Figure 3.7** に示すようにコンパクトな回路である．この回路の機能は，**Figure 3.8** に示すように 2 つの NTSC 信号を入力し，アナログフィールド多重化を行なって 1 つの NTSC 信号を出力する．また，片方のカメラからの画像を分配して，もう一方のカメラの外部同期入力 (VBS) に与えるためのアンプも持つ．この回路の特徴は，出力される信号が通常の NTSC 信号であるため，既存の NTSC 用画像処理装置であればどんなものにでも入力できる点である．

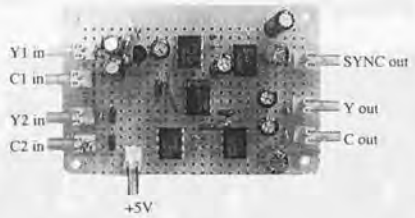


Figure 3.7 Photograph of General-Purpose Mixing Device.

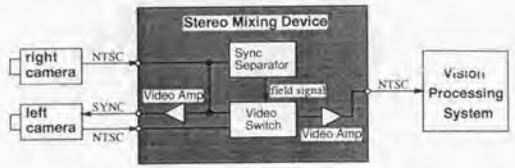


Figure 3.8 Block Diagram of General-Purpose Mixing Device.

ステレオ / カラー・トランスミッタ

ステレオ / カラー・トランスミッタは，**Figure 3.9**に示すように並列画像処理システム [45] のためのトランスミッタユニット（画像 A/D-D/A ボード）上のユニバーサル部に実装した．ビジョンシステム全体の構成を **Figure 3.10**に示す．トランスミッタユニット上の網掛けの部分がフィールド多重化のために拡張された部分であり，各画像処理ボードには全く手を加えず，トランスミッタユニットのだけの拡張となっている．このトランスミッタユニットにステレオカメラからのカラービデオ信号を入力した場合のイメージバスへの出力モードを Table.3.2に示す．ビジョンシステムの各画像処理ボードはトランスピュータ (T805-25[MHz]) と 4[MB] のメインメモリおよび 512[pixel] × 512[pixel] × 8[bit] × 3 枚 (入力，処理，出力用) のフレームメモリ，局所相関演算 LSI(Motion Estimation Processor:MEP) を搭載している．入力されたビデオ信号は，アナログ多重化，A/D 変換，デジタル多重化を行なった後，イメージバスを介して視覚処理ユニットへ送られる．各視覚処理ユニットはこの多重化された画像を同時に受けとり，並列に画像処理を行なう．

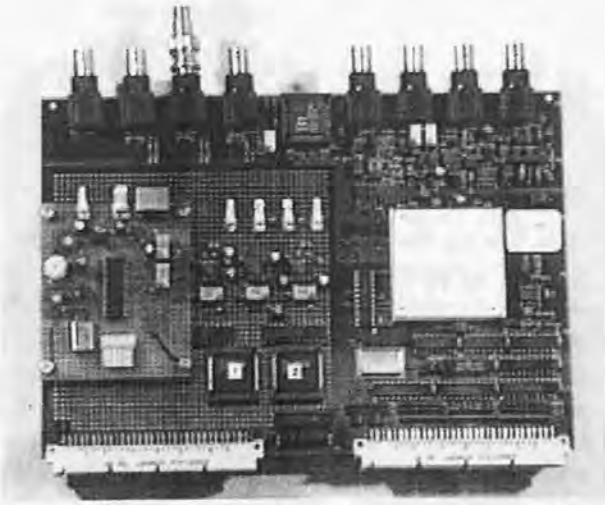


Figure 3.9 Implementation of field mixing device.

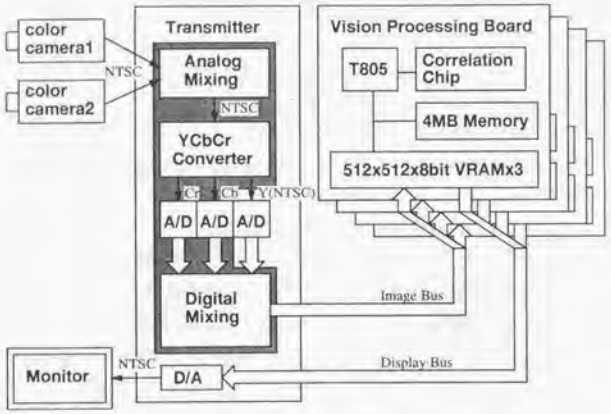


Figure 3.10 Block diagram of total vision system.

3.1.4 評価実験

フィールド多重化画像処理システムの性能を評価するため，通常の画像入力を用いるシステムと本章で開発した多重化を用いるシステムにおいて，同一の画像処理を行ない，処理時間の比較を行なった．

ゼロ視差領域の検出

実験の内容は，ステレオカメラから得られた画像のうち視差がゼロの領域を相関法により抽出するもので，大久保ら [47] が提案したステレオ視による物体追跡の手法を，本システムの視覚処理ユニットに搭載されている局所相関 LSI で実行するようにインプリメントしたものである．

処理の概要を **Figure 3.11**に示す．まず左のカメラから得られる画像を格子状の領域に分割する．その 1 つを参照画像 (Reference Block) とし，右のカメラから得られる画像上で同じ位置にある探索画像 (Search Window) でマッチする場所へのベクトルを相関演算により求める．相関演算のディストーション (マッチングのエラー) 値が小さければ，同じ位置に同じ対象が写っているとみなし視差ゼロの領域とする．ただし画像パターンが一様であったりエッジ状である場合には，ディストーションが小さくても誤マッチングを起こしている可能性が高いので，そのようなパターンはゼロ視差領域から除外する．

Table 3.2 Mode of field mixing transmitter.

Mode	processing	odd	even
1	monochrome	Y1	Y1
2	monochrome	Y2	Y2
3	color	Cb1	Cr1
4	color	Cb2	Cr2
5	monochrome+color	Y1	CbCr1
6	monochrome+color	Y2	CbCr2
7	stereo monochrome	Y1	Y2
8	stereo color	CbCr1	CbCr2

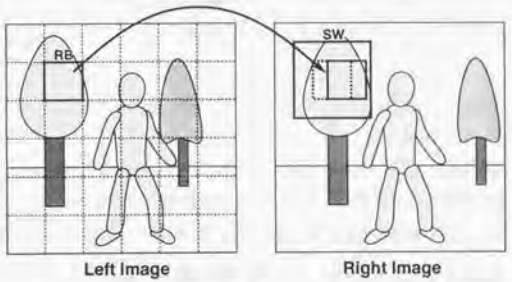


Figure 3.11 Detection of Zero Disparity Area.

システム構成

この実験では 2 枚の視覚処理ユニットを用いてマッチング処理を行ない，ゼロ視差領域を抽出する．比較する 2 種類のシステム構成を **Figure 3.12**に示す．(a) はステレオカメラからの左右の画像をそれぞれ 1 つずつの視覚処理システムに入力し，互いに画像の転送を行ないながらマッチング処理を行なう場合のシステム構成，(b) は左右の画像をフィールド多重化し，その画像を同時に 2 枚の視覚処理ユニットに入力する場合のシステム構成である．2 つの視覚処理ユニットは，全処理領域を上下半分に分割し，IFMMT1 が上半分を，IFMMT2 は下半分をそれぞれ処理することとする．画像の転送はトランスピュータリンク (10[Mbps]) を通して行なわれる．

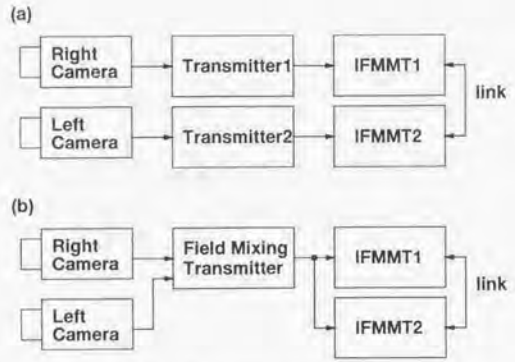


Figure 3.12 Experiment system configuration.

処理手順の比較

(a) のシステムを用いるときの IFMMT1 での処理手順は以下のようになる.

1. 上半分の探索領域の画像を VRAM から取り出し, MEP に送るデータ形式 (SW1') に変換する (VRAM→SW1'),
2. 下半分の参照領域の画像 (RB2) を VRAM から取り出す (VRAM→RB2),
3. RB2 を IFMMT2 に送る (RB2 Send),
4. 上半分の参照領域の画像 RB1 を IFMMT2 から受けとる (RB1 Receive),
5. RB1 を MEP に送るデータ形式に変換する (RB1→RB1'),
6. SW1',RB1' を MEP に送り相関演算を行なう (Corr).
7. 相関値分布によりパターンに特徴が含まれているか判別する (Check).

また (b) のシステムを用いるときの IFMMT1 での処理手順は以下のようになる.

1. 上半分の探索領域の画像を VRAM から取り出し, MEP に送るデータ形式 (SW1') に変換する (VRAM→SW1'),
2. 下半分の参照領域の画像を VRAM から取り出し, MEP に送るデータ形式 (RB1') に変換する (VRAM→RB1'),
3. SW1',RB1' を MEP に送り相関演算を行なう (Corr).
4. 参照領域に特徴が含まれているか判別する (Check).

参照領域に特徴的なパターンが含まれているか, つまり誤マッチングの可能性があるかは, 相関演算の結果として得られる相関値分布の形状が, 先鋭であるかどうかにより判別する [52].

参照領域の画像は 16×16 画素, 探索領域の画像は 32×32 画素, マッチングをとる領域の数は 12行×6列=72 とした. これらの画像は 1 ピクセルごとに間引かれているので, 画面上での処理領域は 1 領域あたり 32×32 画素, 処理領域全体では 384×192 画素である. これを 2 枚の視覚処理ユニットで処理するので 1 枚あたりのマッチングをとる領域の数は 32 となる. この処理を (a),(b) それぞれのシステムでの処理時間を **Figure 3.13** に示す.

(a) のシステムでは RB を交換するための通信と, 通信のバッファから MEP に送るためのデータへの変換の 2 つが必要となり, 実行サイクルは 2 フレームとなる. これに対し, (b) のシステムではフィールド多重化により IFMMT1, IFMMT2 両方で左右のカメラからの画像が得られているために画像を交換する必要がなく, 処理は 1 フレームにおさまっている. また, プログラムの大きさや, 通信バッファを含めたメモリの使用量も小さくてすむ.

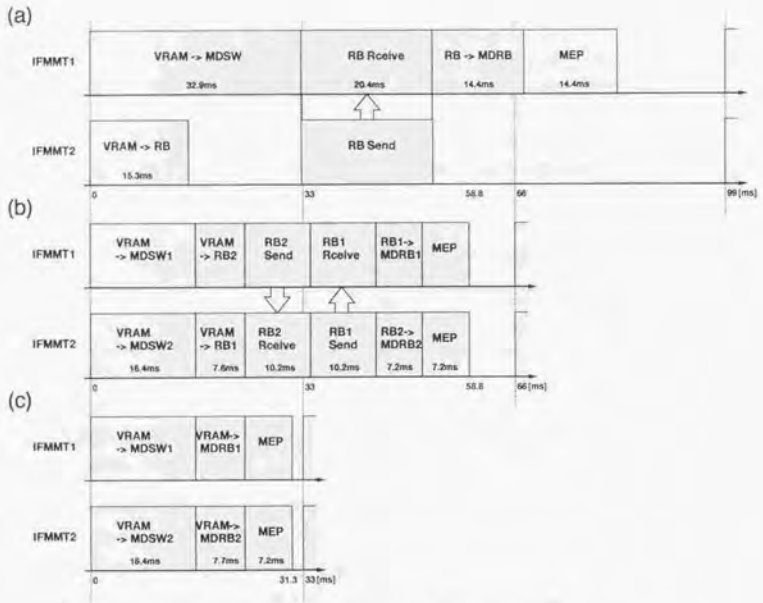


Figure 3.13 Time chart of matching process.

Figure 3.14 Experimental setup.

Figure 3.15 Processing image of zero disparity region based on correlation using mixing device.

Figure 3.16 Processing image of zero disparity region based on correlation using mixing device.

3.2 PC ベースの汎用ロボットカーネル

3.2.1 ロボットのプラットフォーム

現在，知能ロボットは盛んに研究されており，個々の理論や手法が盛んに展開されている．しかし，それらの実験や応用のためのロボットの標準環境(プラットフォーム)は確立されておらず，それぞれの研究者が独自に整備しているのが現状である．そのため，ロボットの開発 / メンテナンスにかかる金銭的 / 時間的コスト，ロボット間でのプログラムの移植性などの面が問題となっており，ロボット学会誌では研究開発プラットフォームの特集号が組まれている [53].

移動ロボットの研究プラットフォームとしては，NOMAD テクノロジ社の NOMAD，デニング社の RMV，ROBOSOFT 社の RBUTER，RWI 社の B21，Kheepera，山彦など様々なロボットが開発・販売されている [54]．また重点領域研究において歩行ロボットの研究プラットフォームとして TITAN-VIII [55] が開発され，多くの研究者に利用されている．また，マニピュレータの制御理論を検証するための標準化アームを作るという構想もある [56]．しかし，ロボットの機構部分ではなく，ロボットのソフトウェアを含み様々なシステムに用いることができるコントローラ [57] については，プラットフォームと呼べるようなシステムはまだ提案されていないようである．

井上 [58] はロボットの実用化を指向したアプローチとして，知能ロボットのカーネル部分を共通化するハイパーマシンというコンセプトを提案した．これは，視覚主導型かつ自立型の知能ロボットのためのコントローラのプラットフォームであるといえる．本研究では，ハイパーマシンの思想に基づき，屋内および屋外での実験のための移動ロボットシステムを開発し，以後の章での実験に利用している．本節では，近年急速に性能が向上している汎用のパーソナルコンピュータを用いて，ロボットカーネルを安価かつコンパクトに構築する手法について述べる．

3.2.2 汎用ロボットカーネル：ハイパーマシン

井上が提案したハイパーマシンとは，ロボットそのものではなく，ロボットのカーネル(中核)部であった．これは自立型の知能ロボット全体を，色々なロボットに共通な一般性のあるカーネル部分と個々のロボットに依存する部分に切り分けることで，それぞれの開発を効率的に行なうことができるようにしたい，という考え方に基づく．ロボットの手足に相当する部分はロボットの種類によって様々であり，共通化は不可能であるので，この部分は必要に応じて個々に開発し，制御，視覚処理などを行なうハイパーマシンと結合して動かす．ハイパーマシンのシステムの概要は Figure 3.17 に示され，以下の3つのサブシステムから構成される．ハイパーマシンでは，これらのサブシステムをできるだけ標準品を用いて構成することで価格を抑え，様々なロボットのカーネルとして広く用られることを目指している．

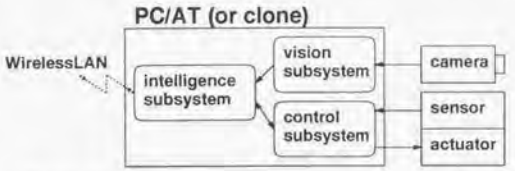


Figure 3.17 System configuration of hypermachine.

視覚サブシステム

知能ロボットの認識処理の中でも，最も重要な位置を占める視覚処理を行なうためのサブシステム．

制御サブシステム

知能ロボットの移動やマニュピレーション等の行動を実現するための機構部分(アクチュエータおよびセンサを含む)の制御を行なうためのサブシステム．制御サブシステムには，アクチュエータやセンサ自身は含まれておらず，それらとのインターフェースを行うだけである．

知能サブシステム

知能ロボットの種々の知能処理の中核を担うためのサブシステム．知能サブシステムには，視覚サブシステムや制御サブシステムをコントロールするためのソフトウェアの他，AI ツール，ユーザーインターフェース構築ツール，プログラミング環境やネットワーク環境を提供する OS なども含まれる．

3.2.3 PC ベースのハイパーマシン

ハイパーマシンの目的はロボットシステム開発における金銭的 / 時間的なコストを押えることであるので，安価で誰もが利用可能な汎用の製品で構成されることが好ましい．しかし数年前までは汎用の PC では十分な性能が得られず，井上らは当初 Transputer を用いて独自に開発した画像処理ボード [59] や制御ボードを用いてハイパーマシンのプロトタイプを開発し，

移動ロボットの制御に用いていた [60],[61] ため，誰もが簡単に利用できるハイパーマシンの構築には至っていなかった．

ここではハイパーマシンのベースとして Transputer に代えて，近年急速に性能が向上している PC/AT 互換機を用いる．ロボットのコントローラとして独自開発のコンピュータやワークステーションを用いるのと比較した場合，PC を用いる利点としては，

- 性能 / 価格比が高い．
- 処理速度や拡張性を用途に応じて選べる．
- 拡張ボード類が豊富に揃っており，機能の付加が容易．
- UNIX(Linux, FreeBSD), Tornade, Windows(95,NT), DOS 等，OS の選択肢が広い．
- 各パーツを買ってきて自作でき，メンテナンスが容易．
- ユーザが多く，メーカの製品開発も盛んであるため，将来性が高い．

などがあげられ，以前のハイパーマシンのプロトタイプの欠点を十分に補っていると考えている．以下に本研究で用いているロボットにおける各サブシステムの構成例を示す．

視覚サブシステム

視覚サブシステムの処理の中枢部分としては，トラッキングビジョン PCI 版 [62](富士通 TRV-CPD6) を用いている．トラッキングビジョンは PCI ロングサイズのカードで，トラッキングやオプティカルフローなどの相関演算に基づく視覚処理が高速に行なえる．なお，本ボードのデバイスドライバは，DOS 版 と Windows95 版のみしか提供されていなかったため，本研究では富士通から提供された DOS 版のライブラリのソースプログラムを Linux に移植し，視覚処理の中核として利用している．

また，汎用画像処理ボード (日立 IP-2000) は ISA バスハーフサイズながら，フィルタリングや色抽出，トラッキング等の多彩な機能を実現しており，コンパクトな視覚処理システムを構築する場合には便利である．また，このボードの上位機種である IP-5000 (PCI バスハーフサイズ) の発売が予定されており，さらに高機能の視覚処理の実現が期待できる．

なお，両眼ステレオ視覚は，これらの画像処理ボードいずれか 1 枚と，外部同期をかけたステレオカメラ，それに 3.1節で述べたフィールド多重化回路の 3 点から成るコンパクトなシステムで実現している．

制御サブシステム

制御サブシステムには，ロボット用インタフェースボード (富士通 RIF-01) を使用する．このボードは自立型の移動ロボットを PC ベースのハイパーマシンで制御することを目的として今回新たに開発したもので，

- PC/AT 互換機の ISA バスハーフサイズの I/O ボード．
- A/D, D/A それぞれ 12bit x 16 チャンネルを持ち，16 自由度までのロボットの制御が可能．
- A/D, D/A の入出力レンジは ± 10[V] で Titech Robot Driver[63] に合わせてある．
- 上記の A/D, D/A 以外に，パルス / ポート入出力 (16bit) と A/D(10bit x 10ch) を持つ IC (日立 Uiversal Pulse Processor) を 2 個搭載．
- A/D,D/A のレンジは固定，A/D に差動入力がない，など機能は限定されているが，コンパクトで使い方も簡単である．

という特徴がある．A/D, D/A のチャンネル数が多く，Titech Robot Driver と組み合わせて使うことで自由度の多い脚型移動ロボットの制御も 1 枚で行なうことが出来る．Titech Robot Driver は東京工業大学 広瀬研究室で開発された DC モータの電流 / 速度 / 位置制御が可能な汎用 DC モータドライバであり，TITAN-VIII に限らず様々なロボットの制御に利用できる．本ボードの仕様ついては，次小節で詳しく述べる．

知能サブシステム

Linux ハイパーマシンでの処理の基本となる OS には Linux を採用した．Linux は POSIX 準処の標準的な UNIX システムであり，UNIX でのプログラム開発に慣れている研究者には使いやすい環境となっている．また 複数プロセスやネットワーク通信機能を用いるプログラムも容易に開発することができる上，通常の端末からロボットにリモートログインすることで，離れた場所からのプログラム開発も可能となる．さらにサーボの制御などのリアルタイム性が強く要求されるような処理を行ないたい場合には RT-Linux というリアルタイムカーネルも用意されている．Linux はフリーウェアとして配布されており，開発コストを抑えるというハイパーマシンの思想にも合致している．

また，Linux は新たなハードウエアを開発した際のデバイスドライバの開発も比較的行ないやすいのも利点の一つである．本研究ではトラッキングビジョン用のドライバを Linux に移植し，視覚処理の中核として利用している．

無線 LAN 移動ロボットでは，通信には無線を用いるの望ましい．そこでハイパーマシンの通信のためのモジュールとして，WaveLAN (NCR 社) もしくは LAN Anywhere (コーラスコンピュータ社) を使用している．どちらも 2.4[GHz] 帯の電波を使用し 2[Mbps] の伝送速度，オープンスペースでは百数十 m のエリアをカバーできる．WaveLAN は ISA のネットワーク

カードで 2[Mbps] の伝送速度を持ち，WavePointII (NCR 社) を用いて従来の有線のイーサネットとの接続が簡単に行なえる．LAN Anywhere はイーサーネットに接続する形態の無線システムで，通常のネットワークカードをそのまま利用できるのが利点であるが，WaveLAN に比べると伝送の遅延がやや大きい．

3.2.4 ロボット用インターフェースボード

本研究で開発したロボット用インターフェースボード (富士通 RIF-01) の外観を **Figure 3.18** に示す．このボードの開発目的は PC ベースのハイパーマシンから様々なロボットを簡単に動かすことができるコンパクトな I/O ボードを作る，ということである．ターゲットとしたロボットシステムのうち，最も自由度が多いものは重点領域研究「知能ロボット」において東工大の広瀬研究室で開発された J.Rob.Leg と Titech Robot Driver を用いた 4 脚歩行ロボット TITAN VIII であったため，その自由度 (12) に足りるように A/D, D/A のチャンネル数を 16 とし，D/A の出力レンジは Titech Robot Driver に合わせ ±10[V] とした．A/D は各関節のポテンショメータを読むことを想定していて，必ずしも D/A と同じ入力レンジである必要はないが D/A に合わせて入力レンジを ±10[V] にした．チャンネル数が 8 以上ありかつ入力レンジが ±10[V] という A/D には，差動入力できる IC が存在せず，またチャンネル数が少ない A/D を使用すると IC の数が増え，ハーフサイズに収めることが難しくなるため，差動入力はできない仕様とした．

また，UPP はその他のセンサの入出力用に搭載した．UPP が持つ 15 種類のパルス処理コマンドは，比較的単純なパルスのカウントやレジスタの比較を行なったり，パルスを発生したりするものであるが，それらを複数組み合わることで，より複雑なパルス入出力処理ができる．具体的な例としては，

- PWM のような任意の周期 / 幅のパルスを出力
- パルスの幅や周期を読む
- 2 相のエンコーダ入力のパルスを読む

などは 1 ～ 2 のコマンドで簡単に実現できる．命令群は一度設定しておけば，PC からは必要に応じてレジスタを読み書きに行くだけでよく，また入出力端子はパルスとしてだけでなく単なるデジタル I/O ポート (ON/OFF) としても使える．

この I/O ボードは A/D,D/A のチャンネル数は 16 と既存 (市販) の PC 用 I/O ボードより多く，また UPP は様々なパルス入出力がでるので，PC + Titech Robot Driver + TITAN VIII というシステムだけでなく，他のロボットの制御にも使える．Titech Robot Driver と共に用いることで，16 自由度までの DC モータとポテンショメータあるいはエンコーダからなるロボットならば制御が可能であり，本研究で用いる車輪型の移動ロボット (カメラ用パンチルタ等の制御も含む) でも，この I/O ボードを利用することにした．

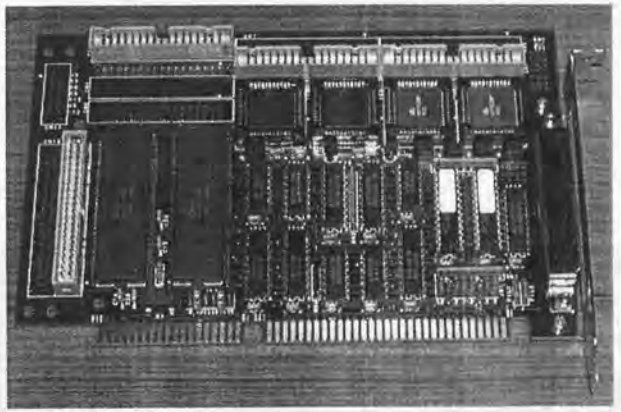


Figure 3.18 Overview of Robot Interface Board RIF-01.

ロボット用インターフェースボード (RIF-01) の仕様	
サイズ	ISA バス ハーフサイズ
アドレス	12bit デコード，8 byte 占有，ジャンパにて設定
割り込み	A/D 変換終了およびパルスの立ち上がり / 下がりで発生可 IRQ9,10,11,12,14,15 からジャンパにて選択

A/D 部の仕様	
使用 LSI	AD7891AP （ANALOG DEVICES 社製） × 2 個
チャンネル数	8ch （シングルエンド入力） × 2 = 16ch
入力レンジ	± 10V 固定
分解能	12bit
変換時間	1.6us （入力チャンネルを切替えた場合 +0.7us）
割り込み	変換終了で発生可

D/A 部の仕様	
使用 LSI	MP7613AP （EXAR 社製） × 2 個
チャンネル数	8ch × 2 = 16ch
出力レンジ	± 10V 固定
分解能	12bit
変換時間	30us

UPP 部の仕様	
使用 LSI	HD63140 （日立製） × 2 個
パルス / ポート入出力チャンネル数	32ch
パルス処理コマンド	15 種類
パルス処理コマンド実行数	最大 16 コマンド
パルス幅分解能	最小 0.25us
パルス処理用レジスタ	16bit x 24 個 x 2 セット
A/D チャンネル数	20ch （シングルエンド入力）
A/D 分解能	10bit
A/D 入力レンジ	+5V 固定
A/D 変換時間	42us
割り込み	パルスの立ち上がり / 立ち下がり，A/D 変換終了で発生可
ウォッチドッグタイマ	0.128ms 〜 131ms の範囲で設定可

3.2.5 ハイパーマシンを用いたロボットの構築例

PC ベースのハイパーマシンを用いて，当研究室ではこれまでに 4 台の移動ロボットシステムを構築した．それらはいずれもこれまで Transputer を用いたシステムだったものを PC ベースのハイパーマシンで置き換えたものである．**Figure 3.19**に，そのうちの 3 つの例を示す．左から，屋外用車輪型ロボット，屋内用車輪型ロボット，脚型移動ロボットである．

屋外用車輪型ロボットはハイパースクータと呼ばれ，座席後部にハイパーマシンが搭載されている．また，ロボットの前部にはステレオカメラ用の 3 自由度パンチルタが搭載されており，ステレオのカラー画像は前述のフィールド多重化回路を用いて合成された後に，トラッキングビジョンに入力されてステレオ処理される．屋内用車輪型ロボットは，デスクトップ PC とハイパースクータと同じステレオカメラシステムを搭載している．脚型移動ロボットは，東工大の広瀬研で開発された 3 自由度 × 4 本の脚を持つ TITAN-VIII にハイパーマシンを搭載した実験システム [64] である．

Figure 3.19 Examples of robot system with hypermachine.

3.3 実験システム

本研究では屋内および屋外での実験のために，2 台の実験システム（視覚移動ロボットシステム）を用いる．どちらも市販の移動ベースを利用し，コンピュータとカメラを搭載して移動ロボットにしたものである．本節ではそれぞれのシステムについての詳細を述べる．

3.3.1 屋内用実験システム

屋内実験用のロボットは，身障者向けの屋内用電動車椅子(Table.)を改造して，コンピュータコントロールできるようにしたものである．左右の車輪を独立に駆動する機構であり，その場で回転することが可能であるため小回りがきき，屋内での実験に適している．本ロボットのシステム構成には，コンピュータシステム (PC98 Note + Transputer, Vividy Note + Transputer, PC ベースのハイパーマシン) とカメラ (単眼，ステレオ) の変遷によりいくつかのバージョンがある．ここではそのうちの最も古い，トランスピュータを用いたバージョンの仕様，外観，システム構成を Table., **Figure 3.20**,**Figure 3.21**に，また最も新しい PC ベースのロボットカーネルを用いたバージョンの仕様，全体と視覚入力部の外観，システム構成を Table., **Figure 3.22**, **Figure 3.23**, **Figure 3.24**に示す．

Table 3.3 Specification of mobile robot base for indoor experiment.

ユニカム 電動車椅子 MOUSE MS-2	
寸法	800L × 600W × 830H
総重量	65kg
操作方式	ジョイスティックレバー
制御方式	左右駆動モータ独立電子制御無段変速
制動方式	電磁ブレーキ
車輪	駆動輪 22.5cm
バッテリー	小型シールバッテリー 12M24
モーター	24V125W × 2 個
最高速度	高速:4.5km/h 中速:3.0km/h
実用登坂力	6 度
最小回転半径	45cm
連続走行時間	2.5h
シャフトエンコーダ	松下電子 ホール IC DN6852

Table 3.4 Specification of mobile robot system (old version) for indoor experiment.

屋内実験用ロボットシステム (旧バージョン) の仕様	
視覚処理システム	jsk-transmitter-02 + jsk-ifm/mt-01 × 4
ビデオカメラ	SONY EVI-310
制御システム	jsk-t-01 + 増設ポート + D/A コンバータ (スピード設定用) + UPP(モータ制御信号用)
バンチルタ	双葉サーボモジュール製 (2 自由度)
ホストコンピュータ	EPSON Vividy Note
トランスピュータ インターフェース	RATOC PCMCIA DIO カード REX5055 + DIO ↔ Imnos Link 変換回路
モニタ	SHARP 液晶ディスプレイ 6E-C10
その他	PC98 用トラックボール (UPP に接続)

Figure 3.20 Photo of HyperMouse (old version).

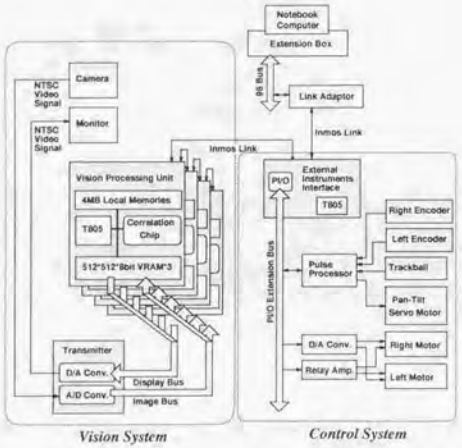


Figure 3.21 Configuration of HyperMouse(old version).

Table 3.5 Specification of mobile robot system (new version) for indoor experiment.

屋内実験用ロボットシステム (新バージョン) の仕様： PC ベースのロボットカーネル	
PC/AT 互換機	マザーボード ADVANTECH PCA-6157 + Pentium 133MHz + 64MB + 2.1GB
視覚処理システム	富士通 カラートラッキングビジョン TRV-CPW5
ビデオカメラ	ステレオ視覚 (SONY EVI-330,370 + 広角レンズ)
	+ 全方位視覚 (SONY EVI-310 + ACCOWLE 双曲面ミラー)
制御システム	富士通 ロボット用インターフェースボード RIF-01
	+ Titech Robot Driver × 5
パンチルタ	ステレオカメラ用パンチルタ (3 自由度)
モニタ	SONY LMD-1041 (NTSC & VGA 表示可能)
その他	PC 用内蔵型無停電電源 AUP-250A

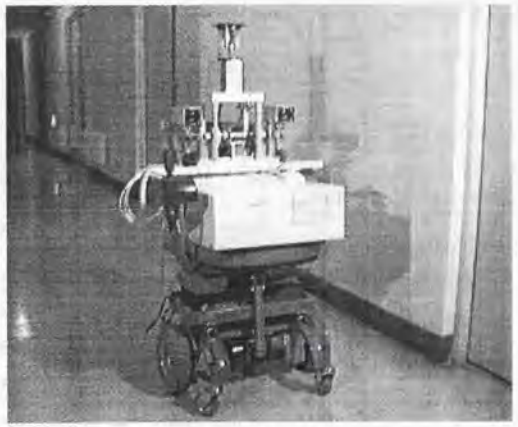


Figure 3.22 Photo of HyperMouse (new version).

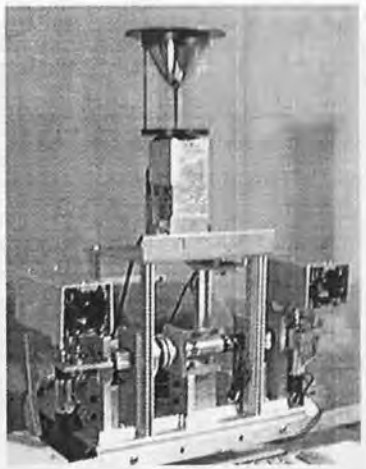


Figure 3.23 Photo of stereo and omni camera system.

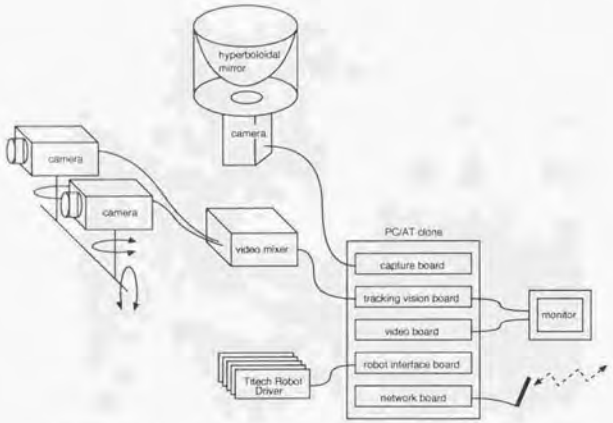


Figure 3.24 Configuration of HyperMouse(new version).

3.3.2 屋外用実験システム

屋外用実験システムは，シニアカーという高齢者向けの電気三輪自動車を改造して，コンピュータコントロールできるようにしたものである．人間が座席に座り，コンピュータの出力画面を見ながらロボットに指示を行ったり，プログラミングを行うことができる．本ロボットのベースとなったシニアカーはディファレンシャル・ギアで連結された後輪を駆動し，前輪によりステアリングを切る機構になっており，直進安定性が高く屋外での実験に適している．このシステムは屋内用実験システムの最新版と同じく，PC ベースのロボットカーネルを用いて構成されている．屋外用実験システムの仕様を Table.3.6に，ベースとなったシニアカーの仕様を **Figure 3.25**に示す．

Table 3.6 Specification of mobile robot base for outdoor experiment.

SANYO シニアカー EWC-35	
寸法	1190L × 680W × 960H
総重量	83kg
駆動方式	後 2 輪駆動 (デフ付き)
制動方式	モーター発電制動、電磁ブレーキ、手動ブレーキ
舵取方式	ハンドルによる前輪操舵
制御方式	アクセルレバーによる無段階制御
バッテリー	12V35Ah × 2 個
モーター	24V370W × 1 個
充電器	自動充電器内蔵 入力 100V 出力 24V
最高速度	前進　高速:6.0km　中速:4.0km　低速:2.0km 後進　2.0km
実用登坂力	10 度
最小回転半径	1100mm
連続走行距離	約 30km

Figure 3.25 Photo of Hyper Scooter.

第 4 章

ビューシーケンスに基づく経路誘導

4.1 ビューシーケンス

4.1.1 ビューの生成

ビュー (view) とは「風景」や「見え方」を意味し，ロボットに搭載されたカメラから得られたそのままの画像全体を指す．ビューベースの認識アプローチの特徴は，画像への特徴抽出処理 (エッジ抽出，領域分割等) をできるだけ行わず，また画像の局所的な特徴を用いずに全体を用いることである．しかしフレームメモリ上の画像は，そのままビューとして用いるにはデータ量が多いため，

- 移動ロボットに搭載できる記憶装置に保持するのは容量的に困難
- 移動ロボットに搭載できる処理装置で処理するのは時間的に困難

である．そこで本研究では解像度を落とした画像をビューとして利用する．Figure 4.1に，カメラから得られたそのままの画像と，解像度を落としてビューとして用いられる画像の例を示す．ビューは記憶と比較に直接用いられるものであり，細かい特徴抽出処理を行なうために用いられるわけではないため，比較的低解像度が低くても構わない．

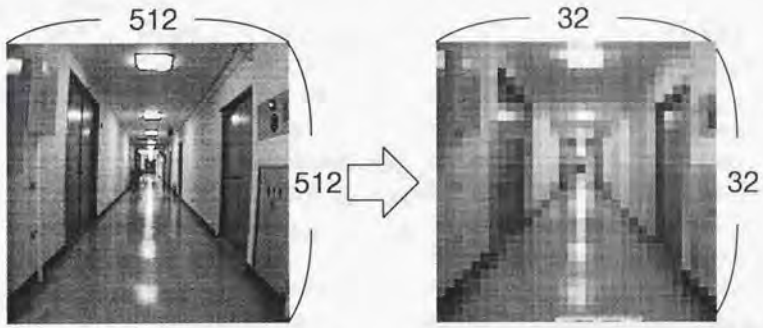


Figure 4.1 Example of a raw image and a generated view.

ビューを生成するために解像度を下げるときには，画像を単に間引くのではなく，平均・縮小処理を行う．これは，ある領域に含まれる複数画素の輝度値の平均をとり，それをその領域を代表する一画素の輝度値とする処理である（Figure 4.2）．

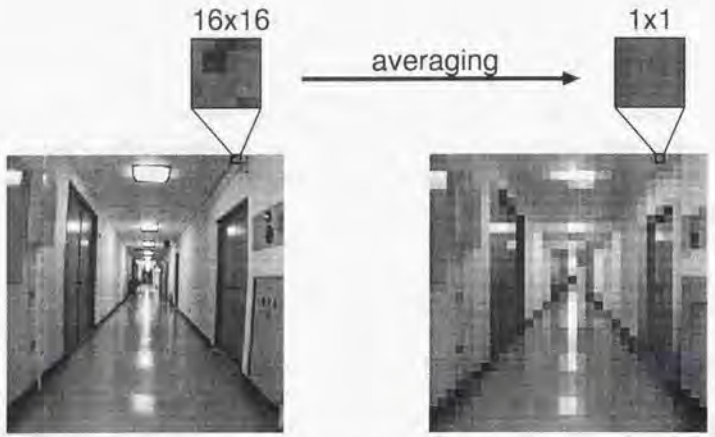


Figure 4.2 Smoothing process in shrinking.

4.1.2 ビューのマッチング

記憶したビューと現在のビューなど，二枚のビューを比較しそれらがどのくらい似ているかを計算する処理を，ビューのマッチングと呼ぶ．これは**Figure 4.3**に示すように，ビューの中心部の縦長の領域をテンプレートとしたブロックマッチングでマッチングエラー e は以下の式で定義される．

$$e(u) = \sum_{x=s}^{xsize-s-1} \sum_{y=0}^{ysize-1} |I_1(x,y) - I_2(x+i,y)| \tag{4.1}$$

$$e = \{\min e(u) | -s \leq u \leq (s-1)\} \tag{4.2}$$

ただし $xsize, ysize$ はビューの縦横のサイズ，$I_1(x,y), I_2(x,y)$ は2つのビュー I_1, I_2 における (i,j) の位置の画素の輝度値，s は水平方向の探索範囲であり，ここでは $s=8$ としたため，$-8 \leq u \leq 7$ の範囲を探索する．また，前述の通り $xsize = 32, ysize = 32$ であるので，テンプレートのサイズは縦 32 画素，横 16 画素となる．画像圧縮用 LSI を搭載した画像処理ボード [45] では この処理を 4.3[ms] で実行することができ，その結果として，マッチングエラー e（ブロック間誤差，この値が小さいほど相関が高い）と，そのおよびその時の水平方向の変位 u が得られる．

テンプレートが縦長であるのは，ロボットの回転や並進によりビューの変位が横方向には大きくなる可能性があり探索範囲を設定したのに対して，ロボットの移動が平面上であるので縦方向には（理想的には）発生せず，縦方向には探索範囲が必要ないからである．

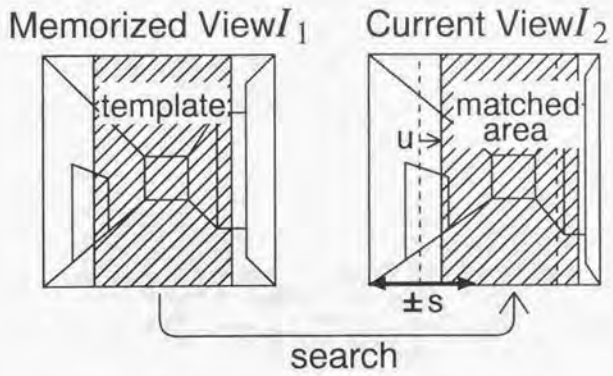


Figure 4.3 Matching method between two view images.

次にロボットが移動した時に位置，姿勢の変化がビューにどのように影響するかを調べた実験の結果を示す．実験環境は廊下（幅 2.5[m]）で，ロボットの初期位置を廊下の中心に置き廊下の長手方向を向たところとする．また，カメラの画角は画角 56[deg] である．

Figure 4.4は，ロボットが回転した時の，u および e の変化の様子を示す．横軸ははじめにビューを記憶した方向からの回転角度である．u の単位はビューの pixel で，この場合 1 pixel は視野角では約 1.5[deg] に対応する．この図から，マッチングによりロボットの回転により発生するビューの水平変位が検出できていることが分かる．15[deg] 以上回転するとテンプレートが視野からはみ出してしまうため，マッチングが失敗し，正しい結果が得られていないが，これはマッチングエラーの増大から判別できる．

また**Figure 4.5**は，ロボットの位置が横方向に移動した時の u および e の変化の様子を示す．横軸は廊下の長手方向とは直角な方向で，はじめにビューを記憶した地点からの移動距離である．この図から，ロボットが原点から離れるにしたがって見え方が変化するために e が増大するが，u は安定して検出できていることが分かる．画像中に非常に遠方の物体しか写っていない場合には，カメラの並進による水平変位の変化はほとんど起こらないが，廊下のように画像中にある程度近くのもの（床や壁）まで写っている場合には，このようにロボットの位

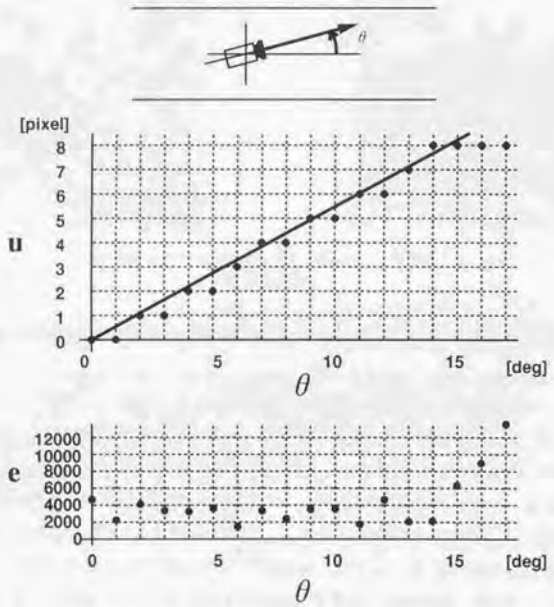


Figure 4.4 Experimental result of matching as a function of rotation of the camera.

置のずれは水平変位を引き起こす．ただしロボットの姿勢（回転）の変化もビューの水平変位を引き起こすため，このマッチングの結果からは，ロボットの位置と姿勢を同時に決定することはできない．

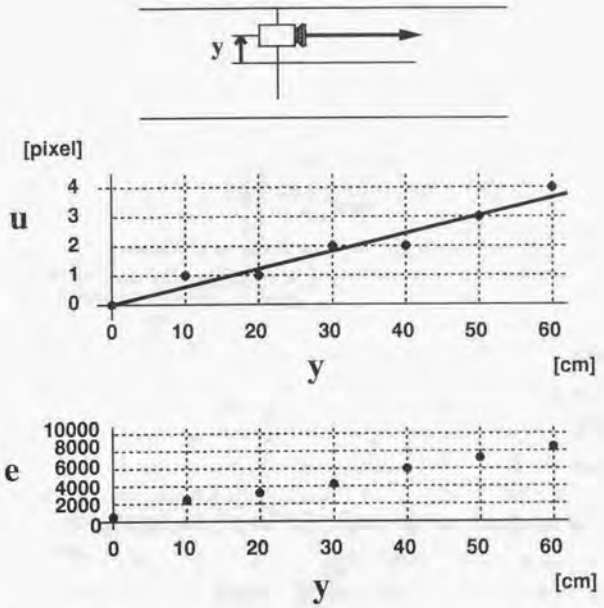


Figure 4.5 Experimental result of matching as a function of lateral translation of the camera.

次に Figure 4.6はロボットが廊下の長手方向に移動した場合のこの変化を示す（u は常に 0 なので省略）．実験では，あらかじめ廊下において 0.9[m] 離れた地点 A,B において 2 枚のビュー M_A, M_B を記憶した．次にその近傍の地点におけるビュー V_x と M_A, M_B のマッチングを行なった．その結果，A 地点付近ではビュー V_x は M_A とのマッチングエラーが小さい（相関が高い）が，地点 B へ近付くに従ってマッチングは徐々に大きく低くなり，逆に V_x と M_B のマッチングが小さくなっていることがわかる．

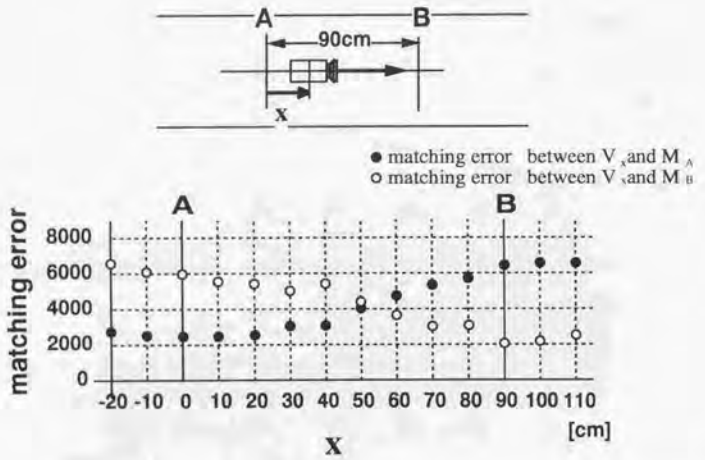


Figure 4.6 Experimental result of matching as a function of longitudinal translation of the camera.

4.1.3 ビューシーケンスの生成

ビューを経路に沿って順番に記憶した画像列をビューシーケンスと呼ぶ．ビューシーケンスはを得るためには，教示走行を行なう必要がある．教示走行では移動経路を直線経路に分割し，各直線経路において以下の手順によりビューシーケンスを生成する．

1. i=0,
2. 現在のビューを記憶し M_i とする,
3. 現在のビュー V_t が，M_i からある程度変化するまでロボットを前進させる,
4. $i = i + 1$,
5. 経路が続くなら 2. に戻る.
6. 終了.

「ビューが変化した」かどうかはビューのマッチングを用いて判断する．つまり，直前に記憶したビューと現在のビューのマッチングを行い，そのエラーが閾値を越えたら新たなビューを記憶するわけである．廊下でのマッチング実験 (**Figure 4.6**) に示したように，マッチングエラーは位置の変化により単調に増加するという性質を持つ．そのため，教示走行中のエラーの推移は **Figure 4.7**に示すようにノコギリ刃状になる．図中の $\varepsilon 3$ は記憶した3枚目のビューと現在のビューのマッチングのエラー値を示す．記憶する間隔は一定距離や一定時間ではなく，一定の「見え方の変化」ごとになる．見え方の変化の閾値は，屋内での実験で 0.5 〜 1[m] 程度の間隔でビューを記憶するように設定した．これは **Figure 4.6**から 1[m] 程度はエラーが単調に増加することが分かっているからである．このような記憶の仕方では一定距離や一定時間ごとに記憶することに比べて，シーンが変化しないところでは記憶を省略することができて効率がよいだけでなく，区別ができる画像しか記憶していないために Zheng [14] や小島ら [65] のように DP マッチング [66] によって連続的なマッチングを用いずに，一度の画像の比較のみで現在位置を認識することができる．こうして記憶したビューシーケンスの例を **Figure 4.8**に示す．

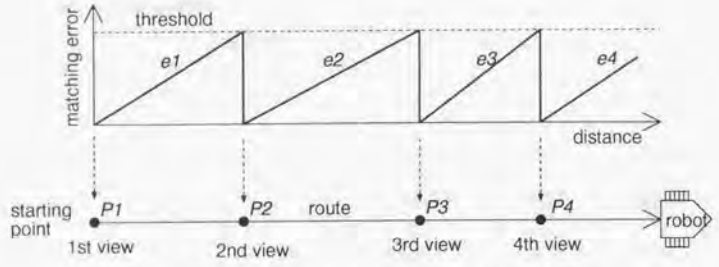


Figure 4.7 Change of matching error in generation of view sequence.

Figure 4.8 An example of generated view sequence.

4.1.4 ビューシーケンスによる経路表現

ビューシーケンスが表現できるのは，**Figure 4.9** 左に示すような直線経路である．完全に直線でない緩やかなカーブを表現することは可能であるが，急なカーブ（例えば交差点を曲がるような走行）を表現することができない．その理由は以下の通りである．

ビューシーケンスは画像のみで構成されており，距離や時間といった概念は含まれていない．画像を記憶するタイミングは画像のマッチングの結果として得られるマッチングエラーのみによって決まり水平変位は関係ない．水平変位は自律走行時の制御にしか利用されない．急なカーブでは水平変位は大きくなるが見え方の変化は小さく，マッチングエラーが増大しないため，教示走行時に正しくビューを記憶することができない．もし水平変位もビューを記憶するタイミングに利用する（つまり水平変位が大きくなったらビューを記憶する）と方向が少しずつ異なるが見え方は近いビューが連続して記憶されることになり，2.1.2 で議論したとおり適当ではない．

さて，ビューシーケンスが **Figure 4.9** 左のような直線経路しか表現できないのでは，ロボットは 1 次元の移動しか出来ないことになり行動の自由度が小さい．そこで **Figure 4.9** 右のような“ある地点から別の地点まで直線経路の繰り返しにより構成された一方向の経路”を表現するために，ビューシーケンスを拡張することを考える．

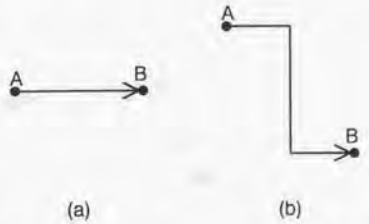


Figure 4.9 Route Representation using View Sequence.

各直線経路のビューシーケンスを並べただけでは，ビュー間の関係が示されていないためどのように行動を行なえば次のビューに進めるのか分からず，経路表現としては不十分である．1 つの直線経路のみの場合では，前進すると次のビューとマッチすることが暗黙のうちに期待できたが，**Figure 4.9** 右のような経路ではそれが期待できない．そこで，ビュー一枚ごとに次のビューとの関係の情報 T_i を明示的に付加して記憶する．たとえば直線経路上のビューには *Forward*，右に曲がる角のビューには *Right*，ゴールのビューには *End* というタグをつける．このタグは，記録時走行に人間がロボットに与える．このように生成した $\{M_i\}$ と $\{T_i\}$

を合わせてビューシーケンスに基づく経路地図の表現と呼ぶ(Figure 4.10).

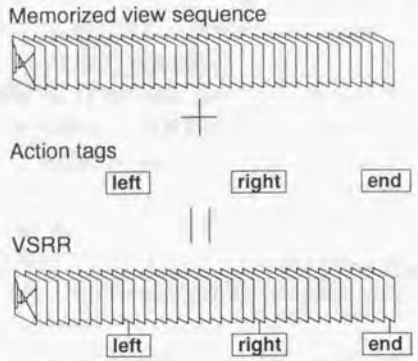


Figure 4.10 View-Sequenced Route Representation.

4.2 ビューシーケンスに基づく認識

4.2.1 自己位置の認識

本研究では移動ロボットの自己位置の認識を

局所的位置の認識 概略の位置が既知の場合に，局所的な探索により正確な位置を認識する

大域的位置の認識 位置が全く不明な場合に，地図全体の大域的な探索により位置を求める

に分け，それぞれをビューのマッチングに基づき実現する [67].

局所的位置の認識

ロボットの経路上の位置は，Figure 4.6に示したビューの局所的な性質，「カメラが地点A,Bの近傍にあることが分かっている場合には，そのどちらにより近いかを判別することができる」により実現される．時刻 $t-\Delta t$ に得られたビュー $V_{t-\Delta t}$ が記憶したビュー M_i とマッチしているとすると，ロボットがゆっくりと前進している場合のビューの変化は連続的であると仮定できるため，ビュー I_t は M_i, M_{i+1} のどちらかとマッチすると期待できる．そこでこの2枚のビューと時刻 t に得られるビューの間でマッチングを行ない(Figure 4.11)，その結果エラー値の低い方のビューを時刻 t でマッチしたビュー，つまり現在のビューシーケンス中での位置とする．

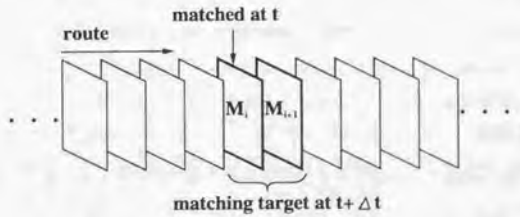


Figure 4.11 Matching target in the view sequence.

Figure 4.12 は，ロボットを直進させた時に得られるビューと，記憶しておいたビューシーケンスとのマッチングの変化を示す．図中の $e3'$ は記憶した3枚目のビューとのマッチングのエラー値であり，$e3'$ はビューを記憶した位置 P3 で極小値をとると期待できる．ここではロボットのビューシーケンス中の位置は3枚目である．ロボットの前進を続けると $e3'$ は徐々に増加して P3 と P4 の中間あたりで $e4'$ を越える．ここでロボットの位置は4枚目に切

り替わる．このような遷移を続けることで，ロボットはビューシーケンス中の自己位置を決定していく．このように探索範囲を近傍だけに制限することで，ビューシーケンス中の全画像とマッチングをとる全探索を行うのに比べて，

1. ビューシーケンスの長さに関係なく処理時間を一定に保つことができる
2. ビューシーケンス中の離れた場所に互いによく似たシーンが複数あるとしても，一度にマッチする可能性があるものは一つしかないため，突然離れた場所のビューにミスマッチしてしまうことを防ぐことができる

という効果がある．

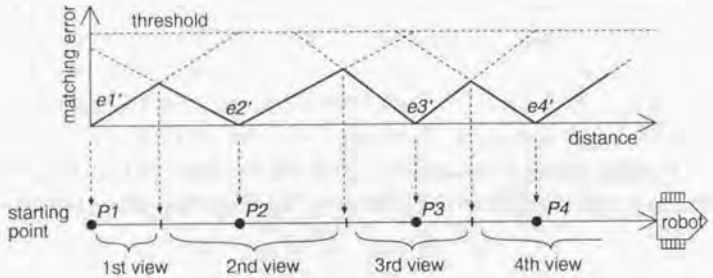


Figure 4.12 Change of matching error in localization.

大域的自己位置の認識

ロボットが移動中に自己の位置を見失った場合や電源投入時等には，過去の履歴なしに自己の位置を知る必要がある．この場合，現在見えているビューと記憶しているビューシーケンス全体の間で **Figure 4.3**のマッチングを行ない，極小値を持つ最もマッチしたビューを現在位置であるとする．廊下のような環境では，似たシーンが複数存在する可能性も考えられる．もし複数のビューがマッチする場合には，一度それら全てを候補としておき，移動しながらマッチングを行なって閾値を越えたらその候補は除くという絞り込みを，候補が1つになるまで続けていけばよい．

Figure 4.13は，廊下である1枚のビューを取り，ビューシーケンス中の全ビュー(85枚)とマッチングを行ったときのエラー値を示す．この場合 e の最小値が示すビュー(17枚目)

は，正しいロボットの位置を示している．実験を行った結果，環境の変化が少ない場合にはこのように最小値を求めるだけで自己位置が決定できることがほとんどであった．

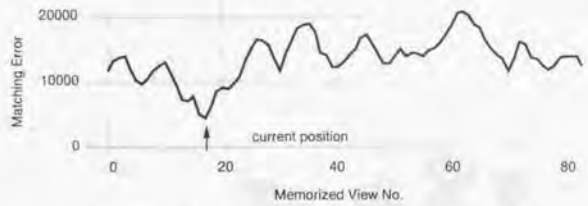


Figure 4.13 Result of global position identification.

4.2.2 障害物の検出

走行路にある障害物の検出は，記憶したビューと現在のビューの差分を利用して行なう．まず現在マッチしているビューを，マッチングの結果のずれの分だけ並行移動する．次にそのビューと現在のビューの差分画像を生成し，変化した領域を抽出する．記録走行時の環境には何も置かれていないということが仮定されていれば，この変化領域は新たに置かれた障害物であるとみなすことができる(**Figure 4.14**)．

このようにして検出された障害物が，危険領域（ロボットの進行方向の走行路）にあるかどうかを調べることで，危険領域に障害物を検出した場合だけ停止し，壁に沿って置かれた消火器等を検出しても走行を続けることができる．このように，一度「標準環境」のビューを記憶しておけば，変化した領域を検出することができ，障害物が出現したとみなすことができる．あるビュー一枚だけから画像処理によって障害物を認識することは非常に難しいが，記憶を用いることで，容易に障害物検出を行うことができるようになる．

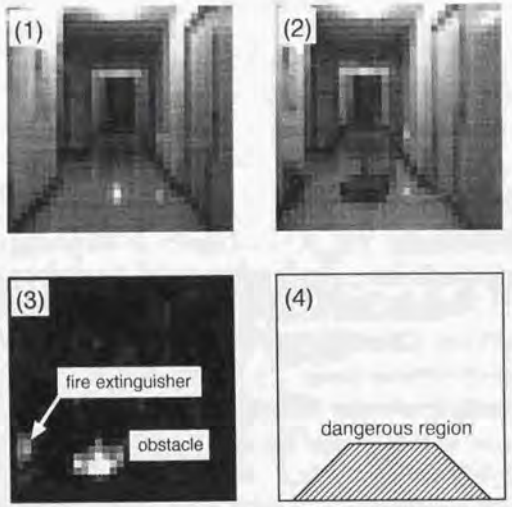


Figure 4.14 Result of obstacle detection. (1)memorized view image, (2)current view image, (3)detected obstacles, and (4)dangerous region.

4.3 ビューシーケンスに基づく行動

ビューシーケンスに基づく走行では，常に自分が記憶したビューシーケンスのうちどこにいるのかをマッチングにより認識した上で，障害物がないことを確認して行動を行なう．走行環境としては屋内の廊下環境を．さらに走行経路としては直線経路を交差点で接続したものを想定している．そこで，走行動作としては

- 直進 (ビューの方向に進む)
- コーナーリング (その場でビューの方向を向く)

の2種類を用意し，それらを繰り返すことで経路に沿った移動を実現する．

ビューシーケンスに基づく経路地図では，ある位置でのビューとそのときに行なうべき動作をセットにして教示走行時に与えてある．**Figure 4.15**では，No.1 から No.10 のビューにマッチしているときは直進し，No.11 のビューにマッチしたところで右折し，No.12 から No.13 のビューにマッチしている間は直進し，No.14 にマッチしたところで停止する，という経路を表現した経路地図の例である．また，このような経路地図を用いて行動行なうアルゴリズムを **Figure 4.16**に示す．ここで `M[N]` は記憶したビューシーケンス，`T[N]` は行動の列，`I` は現在のビューである．

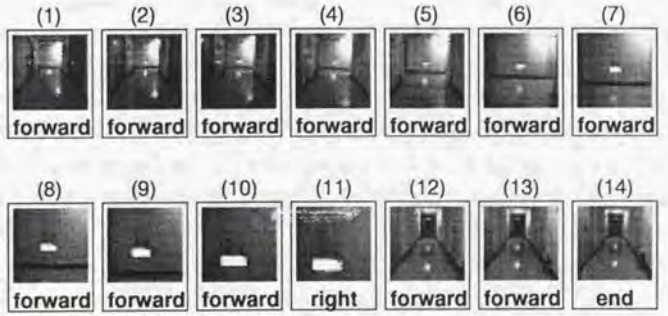


Figure 4.15 Route Representation using View Sequence.

```
 1: typedef struct       /* マッチング結果の構造体 */
 2: {
 3:     int e;           /* マッチングエラー */
 4:     int u;           /* 水平ずれ量 */
 5: } MatchingResult;
 6:
 7: typedef char View[32*32]; /* ビューの型 */
 8: MatchingResult result, result2; /* マッチングの結果 */
 9: View M[N], I;                   /* ビューシーケンスと現在のビュー */
10: int T[N];                       /* 行動の列 */
11: i=0;
12: for(;;){
13:   CreateCurrentView(I);                      /* 現在のビューを生成 */
14:   result  = Matching(I, M[i]);               /* マッチング1 */
15:   result_next = Matching(I, M[i+1]);         /* マッチング2 */
16:   if(result.err > result_next.err){
17:     i++;                                     /* 次のビューへ移動 */
18:     result = result_next;
19:   }
20:   if(T[i] == STOP) break;                    /* 行動が STOP なら終了 */
21:   switch(T[i]){
22:     case RIGHT:turn_right(M[++i]); break; /* 右折 */
23:     case LEFT: turn_left(M[++i]);  break; /* 左折 */
24:     default: go_forward(result);          /* 直進 */
25:   }
26: }
27: stop();
```

Figure 4.16 Processing of Route Representation using View Sequence.

4.3.1 直進

直進時には，局所的な自己位置の認識を行った結果の e が閾値よりも小さければ自己位置の認識が成功していると判断する．ただし，本手法のビューのマッチングの結果からは，カメラの位置と姿勢を同時に決定することはできない．そこで，どちらに進めば記憶した見え方に近付くかという定性的な性質を用いてモータへビジュアルフィードバックをかける．ここで用いる定性的な性質というのは「記憶したビューが現在のビューの右側にマッチしているならば，右にステアリングを切りながら進めば記憶した経路に近付く」というものである．Figure 4.17 にはマッチングの結果，ビューが右にずれていたときの走行路に対するカメラの位置姿勢の例を示す．いずれの場合も，カメラは右前方に進むことでビューのずれが少なくなるなることが分かる．この際の操舵角 θ（正面右向き正）は，$gain$ を定数，u をビューの水平方向のずれ量として，

$$\theta = gain \times u \tag{4.3}$$

という式によって表現される．なお $gain$ の値は実験的に適当なものを求めて用いた．この操舵角は 2 輪独立駆動の移動ロボットでは左右の車輪のスピードの差に変換されて制御が行なわれる．

またマッチングエラーが大きく，マッチングが失敗していると判断した場合は，障害物が出現した等の環境の変化によりマッチするビューが得られなくなったものとして，停止する．障害物が無くなり，マッチングが再び成功するようになった場合には，走行を再開する．

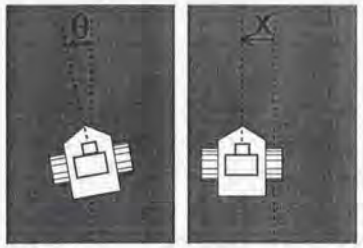


Figure 4.17 Examples of camera position when memorized view matches in right side of current view.

4.3.2 右左折

コーナーリングは，曲がった先に見えた一枚の記憶したビューを用いて行う．まずはじめに，カメラのパンを回転させ，曲がろうとしている方向にカメラを向ける．次に，そのときに得られるビューと記憶したビューのマッチングをとり，記憶したビューが曲がろうとしている方向にあることを確認する．そして，カメラは常にその曲がろうとしている方向を向くようにパンを制御しながら，ロボット本体をその場で回転させる．カメラが本体の正面を向いた時が，コーナーリング動作の終了である (Figure 4.18)．

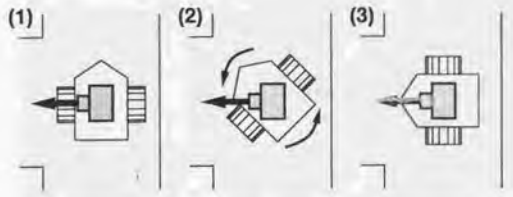


Figure 4.18 flow of cornering.

4.4 経路誘導実験

4.4.1 経路教示

走行経路を教示するには，モニタの画面に表示されているメニューを使って行なう．経路教示時にモニタに表示されている画面の例を **Figure 4.19**に，またメニューの構成を **Figure 4.20**に示す．直線経路を教示する場合には，memorize を選んでロボットをその経路に沿って動かして行くだけで，見え方の変化を調べながら自動的にビューを逐次記憶していく．また角を曲がる時には，曲がる途中のビューは記憶する必要が無いので，left / right turn を押して，ロボットを曲がる方向に向けてからもう一度押すことで，曲がったあとに見えるビューから記憶を続ける．ビューが変化したら記憶していくので，記憶している最中に画面中に人間などの動くものが写ってしまうと，必要のないビューを記憶してしまう．そこで，そのような場合には pause を押して，一時的にビューを記憶しないようにする．記憶したビューシーケンスを画面上で確認するには show を押す．ここで next / previous を押すことでビューシーケンス中のビューを選択する．また，execute で自律走行を開始するが，その前に show で途中まで進めておくことで，その途中のビューの位置から走行を開始できるようになっている．

Figure 4.19 Human interface for teaching run.

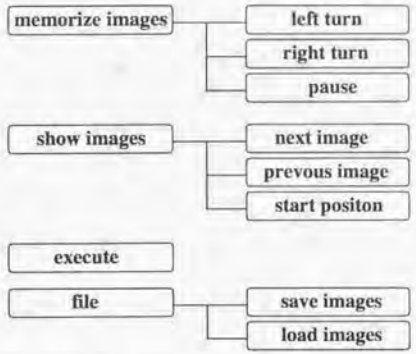


Figure 4.20 Structure of menu.

4.4.2 経路誘導

ビューの生成では 512[pixel] × 512[pixel] のフレームメモリ上の画像を 32[pixel] × 32[pixel] に平滑化して記憶し，その中心部 16[pixel] × 32[pixel] をテンプレートとして用いた．相関演算は，当研究室で開発された画像処理ボードに搭載されている MEP (Motion Estimation Processor) を用いて行っている．

ここでは，工学部9号館にある我々の研究室の前の廊下から出発し，左折，右折，左折を行ない，エレベータの前まで走行する経路（**Figure 4.21**）を教示した．このときに記憶したビューシーケンスを **Figure 4.22**に示す．まず，研究室の前の廊下から出発し (No.1)，角まで進む，角に到着したらカメラを左に向けて曲がった先を確認し (No.2)，カメラは曲がった先を見続けながらロボット本体をその場で回転させる (No.3, No.4)．No.5 ～ No.8 では長い廊下を直進している．このとき，No.7 では自転車（障害物）が左に置かれているが，教示段階から置かれているものであるため，自律走行時のビューのマッチングの障害とはならない．また，この場所では障害物にぶつからないように，あらかじめ廊下の中央よりもやや右を通って経路を教示しておいたので，自律走行時にもその経路を再現している．正面玄関の角に到着したらカメラを右に向けて曲がった先を確認し (No.9)，カメラは曲がった先を見続けながらロボット本体をその場で回転させ (No.10, No.11)，さらに奥へと進む (No.12)．最後に，エレベータの前に到着したら到着したらカメラを左に向けてエレベータを確認し (No.13)，カメラはエレベータを見続けながらロボット本体をその場で回転させ (No.14)，エレベータの方を向いて止まる (No.15, No.16)．

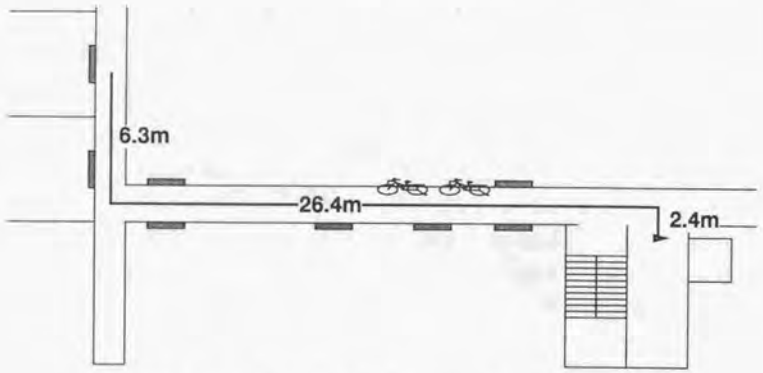


Figure 4.21 Route for Autonomous Navigation.

Figure 4.22 Memorized View Sequence.

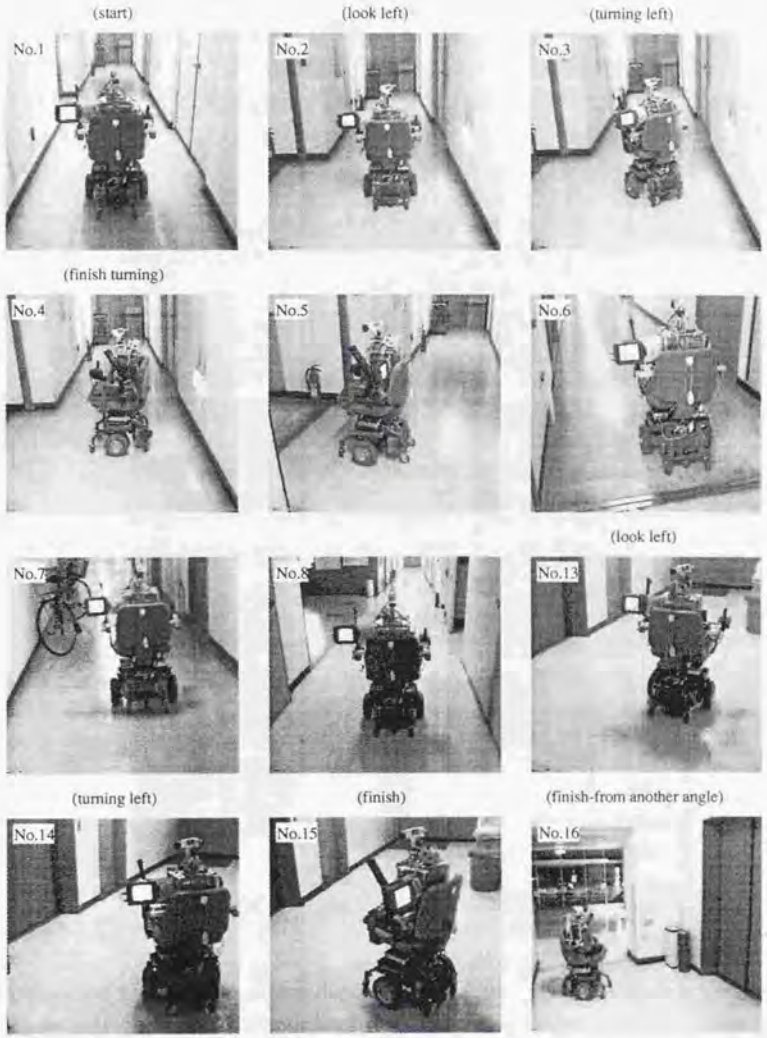


Figure 4.23 Snapshots in Autonomous Navigation.

また，このときのマッチングの様子を Figure 4.24 に示す．マッチングエラーは理想的には Figure 4.12 のようになるはずである．しかし画像のノイズ，あるいはロボットの経路からのずれ等により，マッチングエラーの最小値は 0 にはならず，また細かい変動も見られる．ただし，全体の形状は Figure 4.12 とよく似ており，マッチングの結果の自己位置が成功していることが分かる．

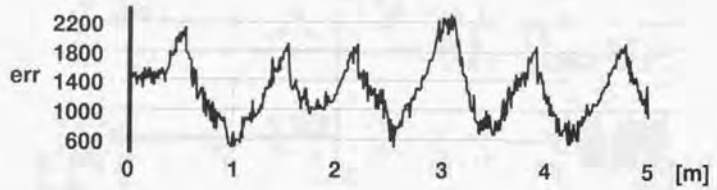


Figure 4.24 Change of matching error in navigation.

自律走行時には，ロボットはビューシーケンスで表される経路のうち現在どこにいるか，常に認識している．環境が変化しない限り突然マッチングが失敗することはないはずである．そこで，マッチングの相関値がある閾値を越えた場合は記憶した時と現在の走行環境の間に変化が生たためにマッチングが失敗しているとして，止まることとした．このときのマッチングの様子を Figure 4.25に示す．障害物のために視野が遮られた場合には，そうでない場合と比べて明らかに最小相関値が高くなるため，その検出は容易に行なえることが分かる．

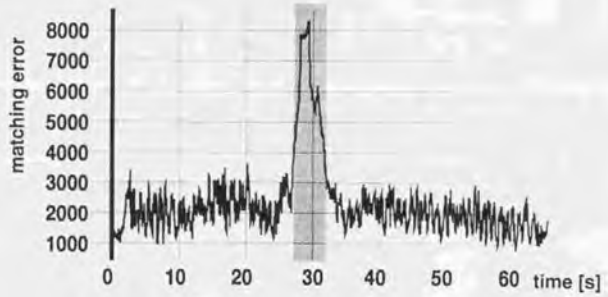


Figure 4.25 Matching error when obstacle appeared.

4.4.3 特徴が少ない環境での走行

視覚により得られる環境の情報が極端に少ない．夜間電灯を消した廊下を走行する実験を行なった．Figure 4.26にこの時に記憶した廊下のおよそ 50[m] の直線経路を表すビューシーケンスを示す．このような特徴の少ない環境では，通常の画像処理による廊下のエッジや床領域を抽出するなどの環境認識を用いていては，走行を実現するのは不可能であるが，ビューシーケンスの記憶を用いることでこのような極端な環境へも適応できることが示された．

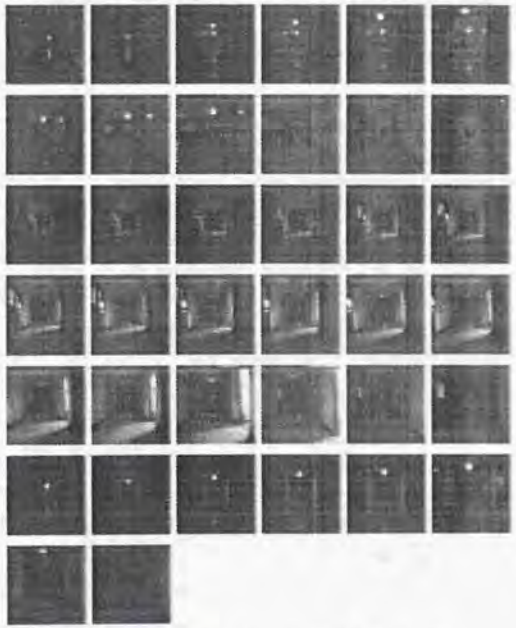


Figure 4.26 View Sequence in dark corridor.

このような環境では，走行中のビューの変化が少ない．教示走行時には，ビューに変化があった時だけ新たなビューを記憶し，変化がない場所では記憶しない．Figure 4.27は，Figure 4.26を記憶した地点を示している，遠くに電灯が見えるだけの場所ではあまりビューを記憶せず，ドアから灯りがもれて明るい場所では多くのビューを記憶している．この図から分かるように，ビューが変化しない，つまり覚える必要のない場所では記憶しておらず，効率

の良い記憶を実現していることが分かる．しかし逆に言うと，ある一枚の画像の場所で止まりたいと思っても，暗くて画像が変化しない場所ではそれだけ位置決めの精度が悪くなってしまうことを意味する．これは，画像だけを用いて位置を認識していることによる限界である．

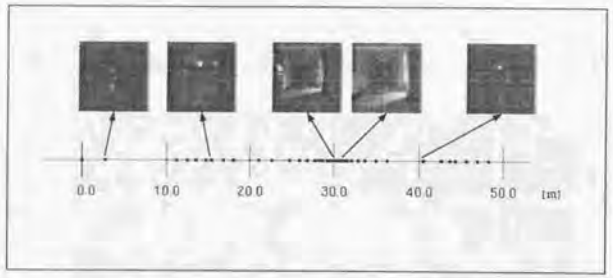


Figure 4.27 Positions where views are memorized in dark corridor.

4.5 考察

4.5.1 ビューの解像度

フレームメモリ上の画像は，そのままビューとして用いるにはデータ量が多い．ビューベースの認識では一般的に，高い画像の解像度の利用が安定した認識につながる，と考えられるが，本研究ではデータ量を削減するため画像の解像度を落とし，記憶およびマッチングの効率を上げる工夫をした．ビューの解像度は，本章の実験では 32×32 としたが，ここでは最適な解像度を決めるために

- マッチングの安定性
- マッチングの計算量
- ビューの記憶量

について検討する．

ビューベースの認識では，画像の連続した変化により，マッチングのエラーも連続して単調増加する必要がある．ビューベースでは連続的に画像列を記憶するものの，それらは画像を表現する空間上で **Figure 4.28**の M_{i-1}, M_i, M_{i+1} のように離散的に分布している．この空間内での距離を測るのがマッチングのエラー関数である．もし画像が V のように M_i, M_{i+1} の間を移動していくならば，マッチングのエラーは V と M_i, M_{i+1} の位置関係を正しく反映するが，V' のように遷移するのであれば，マッチングの結果から正しい認識をすることはできず，またマッチングエラーが大きくなれば，マッチングは失敗したとみなされる．その場合は白丸のような画像も記憶しておけばよいのであるが，記憶する間隔を狭めていくと，記憶量が増えるだけでなくノイズの影響を受けやすくなり，誤認識につながる．そこで，できるだけ長い間隔で記憶できるように，エラーが長い区間で滑らかに単調増加するような種類の画像を用いることが望ましい．エラーの単調増加する区間や滑らかさなどが変わらないのであれば，解像度は低いほうが記憶量および計算量の点で有利である．

ビューの解像度を変えながらマッチング実験を行った結果を **Figure 4.30** に示す．ロボットはある位置で1枚のビューを記憶する．このグラフは，ロボットは徐々に移動しながらその場で得られるビューと記憶したビューのマッチングを行った結果のマッチングエラーをプロットしたものである．元の画像は VRAM 上の 480×480 の領域を利用し，160×160, 96×96, 32×32, 16×16 のビューを全ての点を用いた平滑化により作成する（**Figure 4.29**）．マッチングエラーはビューの画素数で割り，1画素あたりのマッチングエラーとして比較する．上に述べた「マッチングエラーの滑らかさと単調増加する区間の長さ」の観点でこのマッチング結

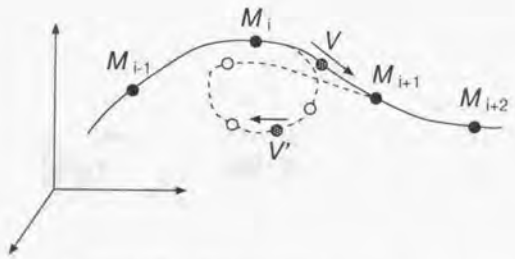


Figure 4.28 View sequence in image space.

果を比べ，どの解像度の画像がビューシーケンスに適しているかを検討する．160×160 および 96×96 は滑らかに単調増加しており，ほとんど差はない．32×32 のグラフはそれに比べるとごく一部でがたつく部分があるが，全体的には滑らかに単調増加している．16×16 の場合はさらにがたつきが大きくなっているので，環境によっては単調増加しなくなることが心配される．そこで，本章で用いるビューの解像度には 32×32 が最適であると判断した．また，このサイズは本研究で用いている相関演算 LSI がサポートするテンプレートマッチングの探索画像のサイズにも一致しており，計算が最も簡単に行なえるという利点がある．

画像の記憶量は画素数に比例するので，解像度を n とすると $O(n^2)$ である．またテンプレートマッチングの計算量は［テンプレートの画素数 × 探索回数］によって決まり，画素数が $O(n^2)$，探索回数は探索範囲を横方向だけなので $O(n)$ で，全体では $O(n^3)$ ということになる．また，現在のビューのマッチング 1 回あたり 4.3[ms] かかっていることから，ビューの解像度を倍の 64 × 64 にすると 8 倍の約 34[ms] かかることになり，ビデオレートでの処理が不可能になるため，これ以上解像度を上げることは適当ではない．

逆に，ビューの解像度をさらに落とせば，より長い距離のデータを表現することが出来るようにはなるが，その反面 (1) ステアリングの制御量の分解能が下がる，(2) 障害物検出のための差分画像の解像度が下がる，という欠点が出てくる．現在のビューは 32[pixel] × 32[pixel] × 8bit であるので 1 枚当たり 1[kB] のデータ量を持つ．1[MB] のメモリ上には約千枚，1[GB] のハードディスクには約百万枚記憶することができる．仮にビューの記憶間隔を 1[m] とすると，このハードディスクはおよそ 1000[km] の経路を表現することができることになる．これは屋内環境における経路表現としては十分な量であり，上記の欠点を考慮するとこれ以上解像度を下げるのは適当ではないと考えられる．

以上の理由から，ビューの解像度の値は 32×32 とした．

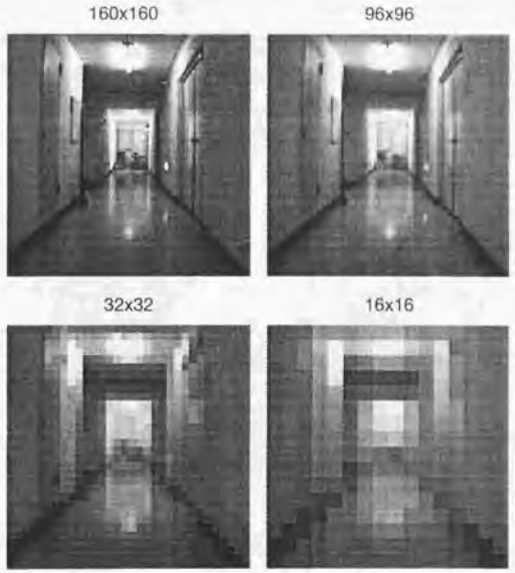


Figure 4.29 Views at various view resolution.

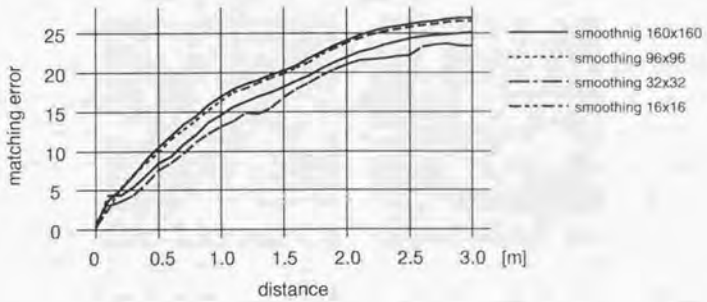


Figure 4.30 Matching error with various view resolution.

また，データ量を削減するには画像を1枚単位で行うだけなく，列として行うことも考えられる．ビューシーケンスではそれぞれのビューは前後のビューと類似しており相関が高いため，この効果は大きい．画像列を圧縮する方法として MPEG 圧縮があるが，それを利用することでデータ量は一桁程度小さくなる．しかし，処理が複雑になり処理時間も増大すること，また画像へのランダムアクセスに制限が生じること，さらに上で述べたように 32×32 にしたことで十分データ量は小さくなっていること，を考慮し現在のところは画像列としての圧縮は行っていない．

4.5.2 平滑化の効果

ビューを生成するために解像度を下げるときには，画像を単に間引くのではなく，平滑化（平均・縮小処理）を行う．これは，ある領域に含まれる複数画素の輝度値の平均をとり，それをその領域を代表する一画素の輝度値とする処理である．平滑化には間引くことと比較して，

1. 画像の高周波成分を取り除く

2. ビデオ信号にのっているノイズを取り除く

という2つの効果がある．平均・縮小処理は，間引きに比べて計算量が多いが，単純な計算の繰り返しであるため専用の LSI が開発されている [68]．このように処理をハードウェア化することで，今後は計算時間は問題にならなくなると期待できる．

まず 1. について説明する．**Figure 4.31**に示すように，間引きでは点，線などの小さな（周波数成分の高い）領域がある場合，同じ画像でも，その位置の少しのずれによって結果の縮小画像は大きく変化する．この現象はサンプリング定理，— サンプリング周波数が f_0 のとき，下の周波数成分は正しくサンプリングできない — によって説明される．この現象は間引くときのサンプリング間隔が大きくなるほど顕著になり，画像上のノイズとして現れて問題となるが，平滑化ではこのような小さな領域も常に縮小画像に一定の影響を与えているので，カメラの微小な動きに対して画像は安定している．実際の廊下画像における，間引きと平滑化の双方による縮小画像の生成結果の比較を **Figure 4.32** に示す．

次に 2. についてであるが，ビデオ信号はアナログであるため，カメラと画像処理ボードの間のケーブルや画像処理ボード上の回路で，多かれ少なかれノイズが含まれることは避けられない．本実験で用いた画像処理システムでは，静止画像を入力していても **Figure 4.33**に示すようにノイズにより±4程度は輝度値が変動していた．このノイズは各フレームの画像の各ピクセルの輝度にランダムにのっているものであると考えられる．平均0のノイズは平滑化（輝

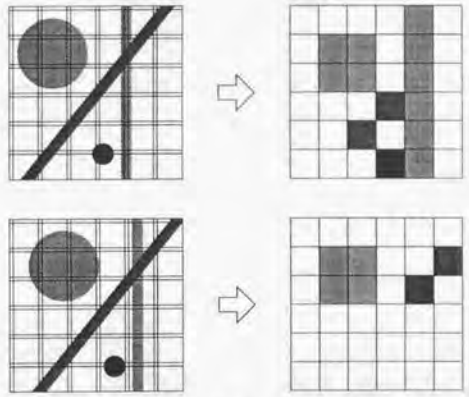


Figure 4.31 Noise in subsampling.

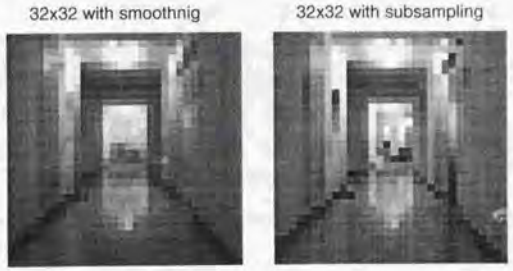


Figure 4.32 Views generated with and without smoothing.

度値の平均をある領域でとる) により減少させることができるはずである. Figure 4.34 は, 同じ静止画像の平滑化後 (16 ピクセルの平均) のある画素の輝度を示す. 輝度値の変動は最大でも ± 1 であり, 平滑化した画像は時間方向に対してかなり安定になっていることが分かる.

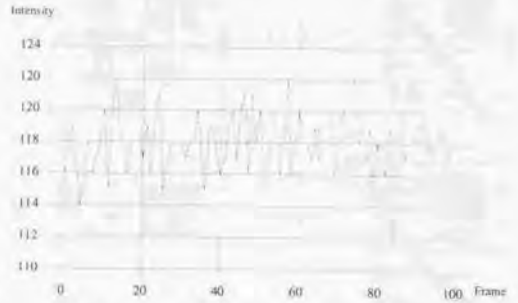


Figure 4.33 Change of intensity of a pixel on a raw image.

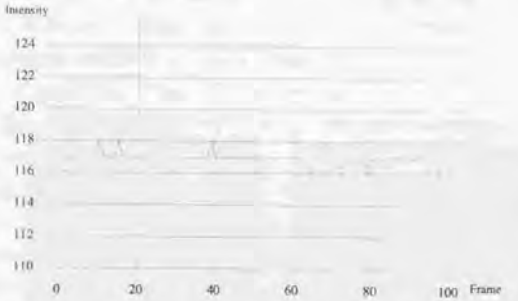


Figure 4.34 Change of intensity of a pixel on a smoothed image.

これらの効果を確認するため, 平滑化を行った場合と行わなかった場合のマッチングエラーの比較を Figure 4.35 に示す. ビューの解像度はどちらも 32×32 であるが, 平滑化を行うことでマッチングエラーの安定性が大きく改善されていることが分かる.

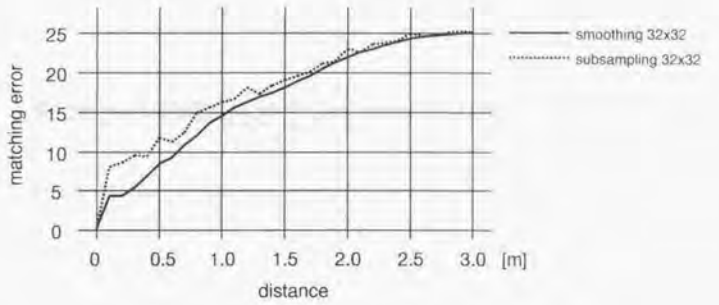


Figure 4.35 Matching error with and without smoothing.

4.5.3 走行精度

本研究で用いたマッチング手法では, ロボットの位置と姿勢を同時に決定することはできないため, 記憶している画像の方向に「向かう」という単純な制御により直進している. ここでは, この制御手法の妥当性を判断するため, (1) 自律走行の正確さ, (2) 必要な教示の正確さ, (3) 初期位置のずれへの対応, の 3 点について考察する.

自律走行の正確さ

自律走行実験での直線部分での走行軌跡を Figure 4.36に示す. $y = 0$ が教示しようとした目標経路である. 同一の教示データを用いて 4 回自律走行を行なった結果得られた軌跡は, 最大でも目標経路から誤差 ±8[cm] であり, 我々が許容範囲と考える ± 10[cm] の誤差に収まっている. また各軌道のばらつきは ±6[cm] 程度である. これらの誤差の原因として, (1) 初期姿勢のずれ, (2) 教示の不正確さ (特にビューを記憶する瞬間の姿勢のずれ), (3) 画像のノイズ, (4) 環境の変化, (5) 制御の時間遅れ, など様々なものが考えられるが, この図からはそれらの合計としての誤差は実用上問題ない範囲であると言える.

必要な教示の正確さ

本実験での経路教示は人間がロボットを押すことで行なうが, 本実験のロボットは 2 輪独立駆動の機構を持つため, クラッチを外した状態では直進性は低く, 正確な教示を行なうのは困難である. この教示の不正確さの影響は, 各軌跡において常に同じように揺れている部分 (Figure 4.36での 5 〜 6[m] の部分など) に現れていると思われるが, その影響は 5[cm] 程度

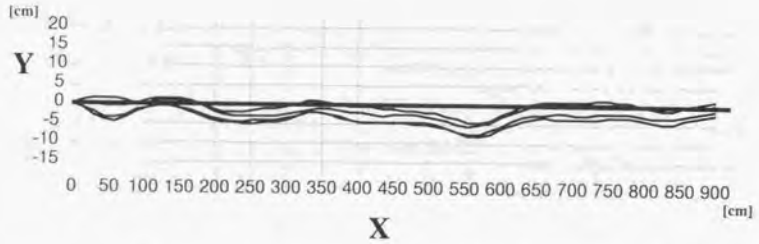


Figure 4.36 Trajectries of autonomous navigation (1).

と見られ特に問題にはならない大きさであった.

初期位置のずれへの対応

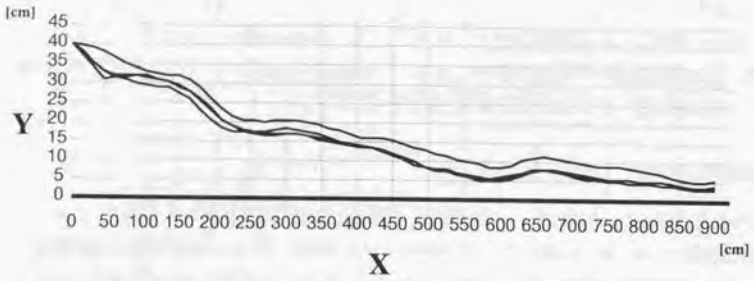


Figure 4.37 Trajectries of autonomous navigation (2).

自律走行時のスタート位置をずらした場合の目標軌道までの追従実験の結果を **Figure 4.37**に示す. スタート位置を 40[cm] 横にずらして 4 回の自律走行を行なった. 本手法の制御は前に述べたように単純なものであるが, 進むに従って目標軌道に近付き, およそ 7[m] 進んだ位置で誤差 10[cm] 以内に入っている. またスタート地点での姿勢のずれを (マッチングが失敗しない範囲で) 与えても自律走行には特に影響しなかった. このように, 初期位置の誤差にも十分対応できているという結果が得られた.

第 5 章

全方位ビューシーケンスに基づく経路誘導

5.1 全方位ビューシーケンス

前章では“ビューシーケンス”と呼ばれる経路表現を用いて，ロボットを誘導する手法を提案した．この手法は実環境において安定した走行を実現したが，単眼カメラから得られる前方のビューだけでは視野が狭い（通常のカメラで30度程度，超広角でも90度程度）ため，利用できる情報が限定される．視野が狭いために生じる問題点として以下のことが挙げられる．

- 同じ位置でも逆方向を向くとビューは異なる．しかし逆方向のビューは教示時に記憶できないため，教示走行と同じ方向にしか自律移動できない．
- 行き止まりやT字路など壁の近くに進む時，前方のビュー内の性とんど全ての領域を壁が占めてしまい，マッチングのための特徴が得られない．
- 直進時にロボットの教示経路からの横方向のずれを，回転のずれと分離して検出できないため，スタート位置が大きくずれていた場合の正しい経路への収束が遅い．
- コーナリングの際，曲がった先の画像が見えないため，一度止まってカメラの向きを変えなければならない．

これらの問題は，アクティブビジョンの考え方に基づいてカメラを動かすことで解決することは可能である．アクティブビジョンは，画像中に行動のための情報が足りないときは，アクティブにカメラを動かすことで解決しようというコンセプトである．しかし，その場合はどこを見るかという戦略が必要になるため処理が複雑になったり，カメラを物理的に動かすことはロボットの行動の実時間性を失わせることになる，といった別の問題を生む．またカメラを複数台搭載することで，複数の視野を同時に処理できるようにしたシステムもあるが，システム構成が複雑になるために移動ロボットに適した解決方法であるとは言えない．

このような問題点を解決するものとして，近年，円錐型の鏡を用いた全方位視覚センサ[15]が提案された．これは一台のカメラで一度に周囲360度の画像を取り込むことができる視覚システムである．本章では，この全方位視覚センサをこれまでのビューシーケンスを用いた環境表現に適用した“全方位ビューシーケンス”を，移動ロボットの経路表現として提案する[69]．全方位ビューは，これまでのビューよりも視野が広く含まれている情報が多いため，より安定なマッチングが実現できる．またビューの歪みを検出することでこれまでのビューシーケンスを用いた手法よりも正確にロボットを誘導することができることを示す．

5.1.1 全方位ビューの生成

全方位ビュー

本章では移動ロボットの経路表現として，これまでのビューシーケンスを拡張した“全方位ビューシーケンス”を提案する．全方位ビューシーケンスは，回転双曲面ミラーを用いた全方位視覚センサによって得られる全方位画像を用いて生成される．本研究で用いる全方位視覚センサの外観を Figure 5.1 に示す．カメラは鉛直上向きに設置されており，ミラーでの反射により全方位の画像が得られる．

カメラから得られる全方位画像は Figure 5.2上に示すように歪んだものである．本研究ではこの画像を Figure 5.2 下に示す円筒投影のパノラマ画像に変換してから用いる．この変換後の画像を“全方位ビュー”と呼ぶ．

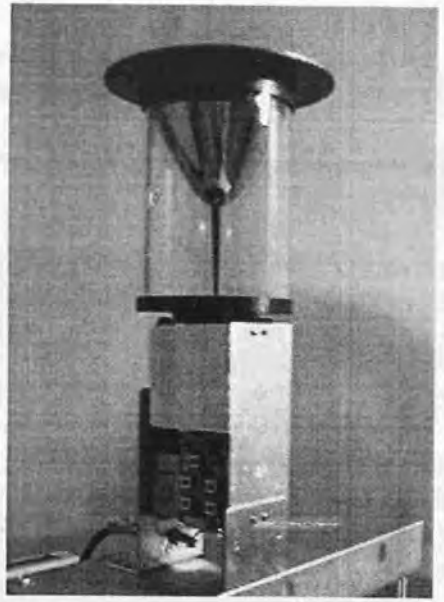


Figure 5.1 Omnidirectional vision sensor.

画像を変換するのには，大きく二つの理由がある．一つは Figure 5.2 上に示すように，

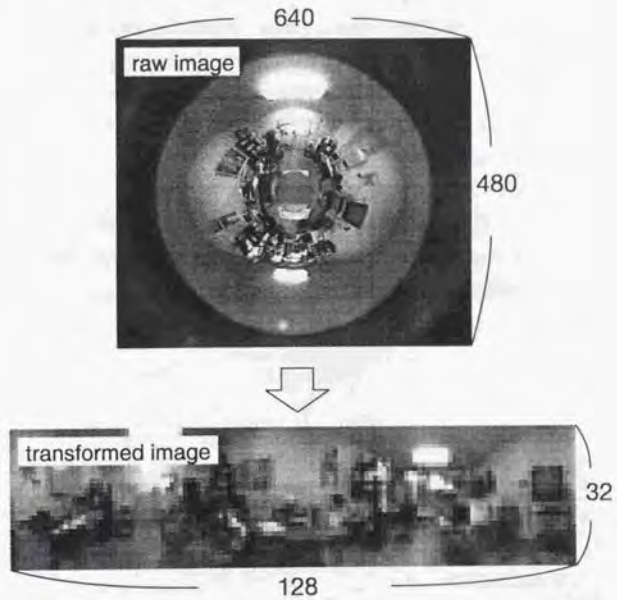


Figure 5.2 A sample of raw omnidirectional image (above), and transformed cylindrical omnidirectional image “Omni-View” (below).

カメラから得られる生の全方位画像は大きく歪んでいるので，そのままではマッチングに適していないことである．ロボットがある地点で回転すると，全方位画像は画面の中央を回転中心に回転する．このような回転量を検出できるマッチングは，単純なテンプレートマッチングと比較して計算量が複雑で，ハードウェアによる相関演算 [45] を利用することができない点で不利であり，またロボットの並進による画像の変形も複雑で，マッチングの結果からロボットの制御を行なうのが難しい．これに対して，変換後の全方位ビューではロボットの回転は画像全体の水平移動として現れ，その検出には通常のテンプレートマッチングが利用できる．またロボットの並進は画像の局所的な拡大縮小および水平移動として現れるので，検出しやすい．

もう一つは，データ量を小さくするためである．ビデオメモリのサイズは 512 × 512 × 8[bit] であるので，データサイズは 256[kB] となり，記憶したりマッチングをとるには大きすぎるが，変換後の画像は 128 × 32 × 8[bit] = 4[kB] となり，メモリやハードディスク上に記憶するのが可能なサイズとなる．

全方位ビューの横のサイズが縦のサイズよりも大きいのは，ロボットが回転すると画像は横方向にずれていくので，自己位置の推定に必要な回転量の検出精度を高めるためには横サイズを大きくする必要があるという理由からである．一方，ロボットは平面上を移動していると仮定できるので垂直方向に画像が移動することはなく，縦の解像度は高い必要は無い．

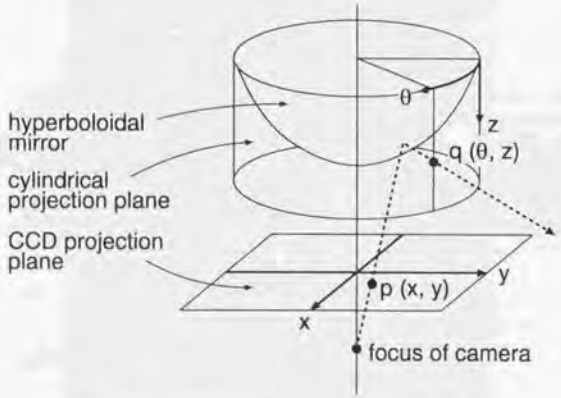


Figure 5.3 Model of omnidirectional vision sensor.

この画像の変換を全方位視覚センサのモデル (**Figure 5.3**) を用いて説明する．$p(x,y)$ はビデオメモリ上の画素，また $q(\theta,z)$ は対応する仮想的な円筒投影面上の画素である．この画素

間の対応表を前もって作成しておき，変換はその表を用いてソフトウェアにより行われる．また，この変換では同時に平滑化を行うことで画像のノイズを減らす．この平滑化は $p(x,y)$ の値を直接 $q(\theta,z)$ とする代わりに $p(x,y)$ の近傍の画素値の平均を取りそれを $q(\theta,z)$ とするものである．

双曲面ミラーの光学特性

画像変換は，円筒射影面上の画素 P と入力画像面上の画素 p の対応表を前もって作成しておき，p 近傍の画素の輝度値を平滑化して P の輝度値とするという方法で行なう．対応表を作成する際には P と p の座標間の関係式が必要となる．この項では関係式の前提となる双曲面ミラーの光学特性について述べ，次項で関係式を示す．

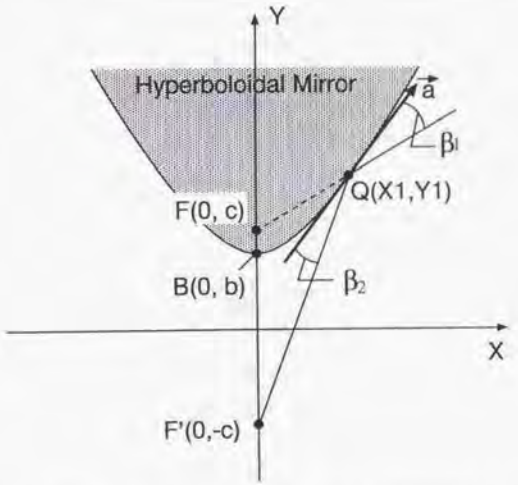


Figure 5.4 Charactoristics of reflection on hyperboloidal mirror.

双曲面ミラーには，焦点 F に向かう光線がミラーに反射すると，反射光はもう一方の焦点 F' を通るという光学特性がある．以下これを示す．

Figure 5.4のように双曲面ミラーの鉛直軸を含む断面を考える．2つの焦点の座標を $F(0,c)$，$F'(0,-c)$，ミラーの底の座標を $B(0,b)$ とすると断面の双曲線の方程式は，

$$b^2X^2 - (c^2 - b^2)Y^2 = -b^2(c^2 - b^2) \tag{5.1}$$

式(5.1)の両辺を X で微分して $\frac{dY}{dX}$ を求めると，

$$\frac{dY}{dX} = \frac{b^2 X}{(c^2 - b^2)Y} \tag{5.2}$$

よってミラー上の点 $Q(X_1, Y_1)$ における双曲線の接線方向のベクトルを $\vec{a}$ とすると，

$$\vec{a} = ((c^2 - b^2)Y_1, b^2 X_1) \tag{5.3}$$

一方，各焦点から点Qへ向かう直線の方向ベクトルは，

$$\overrightarrow{FQ} = (X_1, Y_1 - c), \quad \overrightarrow{F'Q} = (X_1, Y_1 + c) \tag{5.4}$$

$\vec{a}$ と $\overrightarrow{FQ}$，$\overrightarrow{F'Q}$ のなす角をそれぞれ β_1，β_2 とする．

$$\begin{aligned} \cos\beta_1 &= \frac{\vec{a} \cdot \overrightarrow{FQ}}{|\vec{a}| \cdot |\overrightarrow{FQ}|} \\ &= \frac{cX_1}{|\vec{a}|} \cdot \frac{cY_1 - b^2}{\sqrt{X_1^2 + (Y_1 - c)^2}} \end{aligned} \tag{5.5}$$

ここで，点Qは双曲線上にあるので式(5.1)より，

$$X_1^2 = \frac{(c^2 - b^2)(Y_1^2 - b^2)}{b^2} \tag{5.6}$$

式(5.6)を式(5.5)に代入して整理すると

$$\cos\beta_1 = \frac{bcX_1}{|\vec{a}|} \tag{5.7}$$

同様に，

$$\cos\beta_2 = \frac{\vec{a} \cdot \overrightarrow{F'Q}}{|\vec{a}| \cdot |\overrightarrow{F'Q}|} = \frac{bcX_1}{|\vec{a}|} \tag{5.8}$$

式(5.7)と式(5.8)より

$$\beta_1 = \beta_2 \tag{5.9}$$

となる．よって，双曲面の焦点Fに向かう光線がミラーに反射すると，反射光はもう一方の焦点F'を通ることになる．

座標間の関係式

Figure 5.3のように半径 R の円筒投影面上に方位角 θ，鉛直方向の座標 z の画素 $P(\theta, z)$ をとり，Pに対応する入力画像上の画素を $p(x, y)$ とする．ここでは，(θ, z) と (x, y) の関係式を示す．

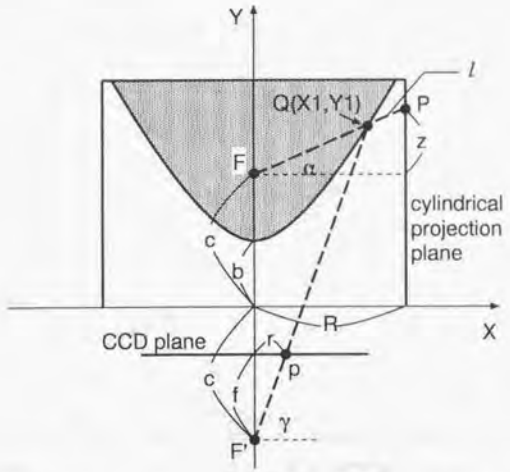


Figure 5.5 Transformation from CCD plane to cylinderical plane.

Figure 5.5のように点Pと鉛直軸を含む断面を考える。焦点FからPを見たときの仰角を α とすると，

$$\alpha = \tan^{-1}\frac{z}{R} \tag{5.10}$$

焦点Fと点Pを通る直線 l の方程式は，

$$Y = (\tan\alpha)X + c \tag{5.11}$$

式(5.11)と双曲線の方程式(式(5.1))を連立させて，直線 l と双曲線の交点Qの座標 (X_1, Y_1) を求める．式(5.11)を式(5.1)に代入して整理すると，

$$\{b^2(1 + \tan^2\alpha) - c^2\tan^2\alpha\}X^2 - 2c(c^2 - b^2)\tan\alpha \cdot X - (c^2 - b^2)^2 = 0$$

両辺に $\cos^2\alpha$ をかけて，

$$(b^2 - c^2\sin^2\alpha)X^2 - 2c(c^2 - b^2)\sin\alpha\cos\alpha \cdot X - (c^2 - b^2)^2\cos^2\alpha = 0$$

$$X_1 = \frac{(c^2 - b^2)\cos\alpha}{b - c\sin\alpha}, \quad Y_1 = \frac{(c^2 - b^2)\sin\alpha}{b - c\sin\alpha} + c \tag{5.12}$$

一方，前項で述べた双曲面ミラーの光学特性により，直線 l に沿ってミラーに入射する光は点 Q で反射して焦点 F' に向かう．焦点 F' から入力画像面までの距離を f，p から鉛直軸までの距離を r，反射光と入力画像面のなす角を γ とすると，

$$\tan\gamma = \frac{f}{r} = \frac{Y_1 + c}{X_1}$$

$$r = \frac{(b^2 - c^2)\cos\alpha}{(b^2 + c^2)\sin\alpha - 2bc} \cdot f \tag{5.13}$$

点 p の座標 (x, y) は，

$$x = r\cos\theta, \quad y = r\sin\theta \tag{5.14}$$

式(5.10)，式(5.13)，式(5.14)より，$\mathrm{P}(\theta, z)$ から点 p の座標 (x, y) を求めることができる．

5.1.2 全方位ビューのマッチング

全方位ビューのマッチングは，用途によりテンプレートのサイズを 360 度，180 度，90 度にして行なう．これらはいずれも相関演算 LSI によるテンプレートマッチング機能を用いて実行される．それぞれのマッチングの結果，水平方向のずれ θ とマッチングエラー e (小さい値ほど画像が似ていることを示す)が得られる．

360 度マッチング

360 度マッチングは，前章で述べた単眼カメラでのビューのマッチングと同様に，自律走行時の自己位置の認識や教示走行時のビューの変化の計測に利用されるものである．360 度マッチングでは，記憶した全方位ビュー全体 360 度分が参照画像(テンプレート)として，現在走行中に得られる全方位ビューが探索画像として利用される．(**Figure 5.6**)．ただし，ロボットの姿勢が記憶時と異なって画像が回転していてもマッチングができるように，ビュー探索画像は全方位画像を横に 2 つ並べた 720 度のものを用いる．この処理には 25.7[ms] かかる．

180 度マッチング

180 度マッチングでは，記憶した全方位ビューの半分 180 度分の領域が参照画像(テンプレート)として，現在走行中に得られる全方位ビュー 540 度分が探索画像として利用される．(**Figure 5.7**)．180 度マッチングは，全方位ビューの一部に人間がうつっている場合など，360 度全部がテンプレートとして用いることができない場合に 360 度マッチングの代わりに用いるものである．この処理には 13.0[ms] かかる．

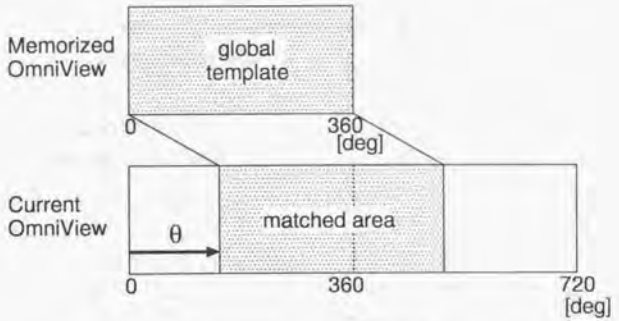


Figure 5.6 360-degree matching of OmniView.

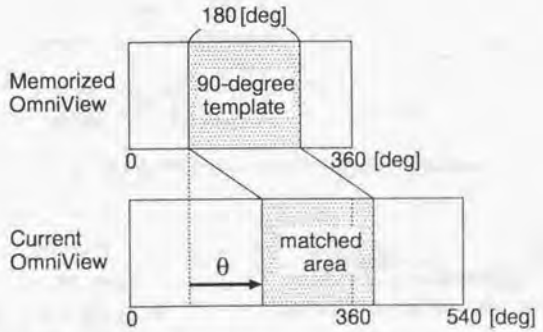


Figure 5.7 180-degree matching of OmniView.

90度マッチング

90度マッチングは，記憶した全方位ビューと走行時の全方位ビューの画像内の歪みを検出するために行われる．記憶したビューの内の90度分の領域がテンプレートとして用いられる(**Figure 5.8**)．このマッチングを複数の90度テンプレートを用いて行い，そのマッチした位置の相対的な変化を見ることで，画像の歪み(広がり / 狭まり)を検出する．一般に，廊下環境ではロボットの全方と後方の画像はテンプレートとして適しているが，側面はテンプレートマッチングに必要な特徴(パターンの変化)が少ないため，マッチングの結果は信頼性が低いものになる．そこで，本論文での実験では，全方位ビューのうち前方部分と後方部分の画像のみを用いる．画像内の歪みを検出するにはテンプレートの幅を小さくする必要があるが，実験の結果テンプレートの幅を90度よりも小さくすると逆にマッチングが不安定になっていくことがに分かったので，テンプレートサイズは90度より小さなものは用意しないこととした．この処理は1回あたり6.7[ms]かかる．

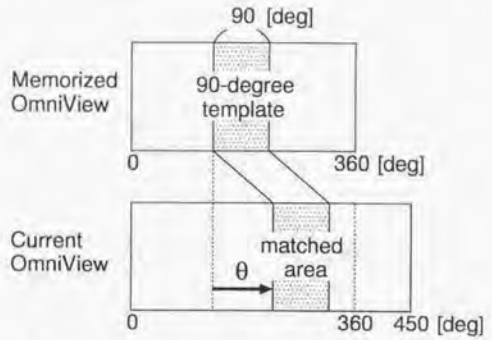


Figure 5.8 90-degree matching of OmniView.

5.1.3 全方位ビューシーケンスの生成

全方位ビュー I を並べて記憶したものを全方位ビューシーケンス $\{M_i \mid 0 \leq i < N\}$ と呼ぶ．全方位ビューシーケンスは，前章でのビューシーケンスと同じように，教示走行で生成される．記録走行は，障害物や歩行中の人間などがない環境で行う．経路全体は直線経路に分割され，それぞれの直線経路をロボットはオペレータの操縦によって移動しながら，以下の手順により全方位ビューシーケンスを記憶する．

1. i=0,
2. 現在の全方位ビューを記憶し M_i とする，
3. 現在の全方位ビュー V_t が，M_i からある程度変化するまでロボットを前進させる，
4. $i = i + 1$,
5. 経路が続くなら2. に戻る．
6. 終了．

全方位ビューの比較は，360度あるいは180度マッチングによって行われる．画像を記憶する間隔は，環境の見え方の変化のしかたによって変動する．記憶するための画像の差の閾値は，廊下環境において0.5[m] ~ 1[m]になるように設定した．

全方位ビューシーケンスにより経路を教示するときの問題として，オペレータがロボットを押して移動させると，そのオペレータ自体が全方位ビューに写ってしまうということが挙げられる．この問題は全方位ビューに死角がないために起きる．

この解決には，ロボットを押して移動さえるのではなく遠隔操縦によって移動させるということがまず考えられる．ロボットは無線LANにより研究室内のネットワークと接続されているため，ロボットの得る画像をオペレータに呈示しオペレータのキー操作によりロボットを移動させるシステムは容易に構築することができる．ところが遠隔操縦を実験した結果，経路に沿ってロボットを直進させることは極めて難しいことが分かった．その理由としては，

- リアルタイムで呈示可能な画像の解像度が，無線LANの通信容量(2[Mbps])により大きくできず，全方位ビュー(128×32)程度が限界である．
- ロボットが左右独立輪で直進性が低いため，キー操作で正確に直進させるのは難しい．

というものが考えられる．

そこでここでは記録走行を“第一教示走行”と“第二教示走行”の2段階に分けることで対処することにした．第一教示走行では**Figure 5.9**のようにオペレータが後方に写っている全方位ビューを構わず記憶する．次に，第二教示走行では，ロボットはオペレータの介入なしに自動的に移動する．この時の位置認識は，360度マッチングの代わりに前方の画像のみを用いた360度マッチングのみにより行われる．オペレータの写っていない“完全な”全方位ビューシーケンスがこの走行中に得られ，その後の自律走行では360度マッチングを用いることができるようになる．

Figure 5.9 Omni-views acquired in the first recoding run.

5.2 全方位ビューシーケンスに基づく認識

5.2.1 全方位ビューシーケンス内の位置推定

Figure 5.10 は A,B 両地点（1[m] 間隔）で記憶された参照用全方位ビュー M_A, M_B と，その周辺で得られる全方位ビュー V との間の 360 度マッチングの実験結果を示す．横軸はロボットの移動距離，縦軸はマッチングエラーである．A 地点の近傍では，V は M_B と比べて M_A とより高い相関を持つ．また B 地点の近傍では，V は M_A よりも M_B とより高い相関を持つ．この実験により，記憶された全方位ビューのうちもっとも近いものは，360 度マッチングにより求めることができることを示している．ただし，このような比較が有効となるのは，V が A,B の近傍で得られたものであることが条件である．このグラフは 360 度マッチングの結果であるが，180 度マッチングでも同様の結果が得られる．

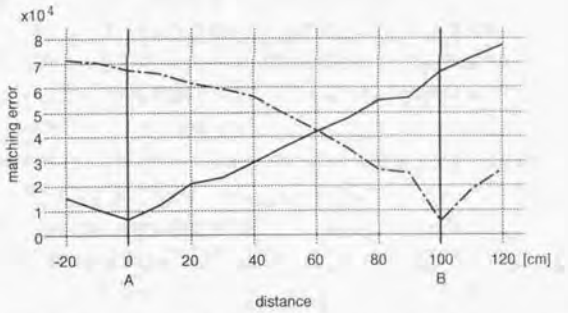


Figure 5.10 Experimental result of 360-degree matching: matching error as a function of longitudinal translation of the camera.

全方位ビューシーケンス内での位置は，このような 360 度あるいは 180 度マッチングの性質を利用して実現される．ロボットが前進している場合，時刻 t での全方位ビュー I_t が記憶した全方位ビューシーケンス中の M_i とマッチしているとする．するとロボットの前進による画像の変化は連続とみなせるので，$I_{t+\Delta t}$ は M_i もしくは M_{i+1} のどちらかとマッチすることが期待できる．そこで，I_t をテンプレートに，M_i および M_{i+1} を探索画像とした 360 度マッチングが実行され，より小さなマッチングエラーを持つ全方位ビューを $t+\Delta t$ でマッチしているものとされる．このような 360 度マッチングを繰り返すことにより，ロボットの位置は更新されていく．

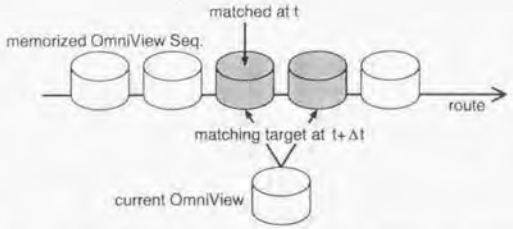


Figure 5.11 Matching target in the Omni-View Sequence for localization.

5.2.2 経路に垂直な方向のずれの検出

ビューシーケンスを用いたこれまでの誘導手法における，単眼のカメラから得られるビューを用いたマッチングの結果からは，ロボットの経路から垂直方向へのずれと姿勢のずれを，それぞれ分離して検出することはできなかった．画像のずれを小さくする方向にビジュアルフィードバックをかけながら走行することにより，これらのずれを徐々に減らして行くことだけしかできないため，初期位置にエラーが大きい場合には経路への追従性はよくなかった．

これに対して，全方位ビューでは局所的マッチングを複数回用いると，複数のカメラを持つのと同じ効果が得られるので，経路に垂直な方向のずれだけを検出することができる．ここでは **Figure 5.12** に示すように，記憶した全方位ビューのうち，前方と後方を90度テンプレートとして用いる．この2つのテンプレートは記憶した画像中では 180[deg] の相対位置をもっているので，ロボットが記憶した経路上にいるときには θ は 180[deg] になるはずである．経路から横にずれている場合には θ の値によってどちらにどのくらいずれているかが分かり，それによってロボットを制御することで正確に記憶した経路に沿うことが可能になる．

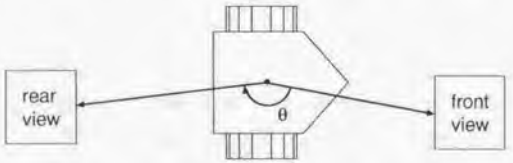


Figure 5.12 Detection of lateral displacement using local matching.

ここでは，ロボットの経路からの横方向のずれが，マッチングの結果にどのように影響するかを調べた実験結果を示す．ロボットは幅3[m]の廊下の中央に廊下の長手方向を向いて置かれている．90度マッチングに用いるテンプレートのサイズは 32[pixel] × 32[pixel] である．全方位ビューのサイズが 128[pixel] × 32[pixel] であるので，90度テンプレートの視野は 90[deg] である．もし画像の中に十分に遠い対象しか写っていなければ，ロボットの横方向のずれは90度テンプレートのマッチング結果のずれ θ には影響を与えないはずである．しかし，実際の画像には床や壁などの比較的近い対象が写っているため，ロボットの横方向のずれと回転のずれが合わさって θ に影響を与える．しかし，θ からはこの2つのずれを分離して認識することはできない．

ところが，相対的な画像のずれ $\theta_{rel} = \theta_{front} - \theta_{rear}$ を用いることで回転のずれをキャンセルし，横方向のずれの影響だけを取り出すことができる．もしロボットが正確に記憶した経路上にいるならば θ_{rel} は 64[pixel] (=180度) になる．**Figure 5.13** は，ロボットに横方向のずれを与えて θ_{rel} を計測した結果である．これにより，ロボットの向きに関係なく 5 ～ 10[cm] の制度で横方向のずれが検出できていることが分かる．

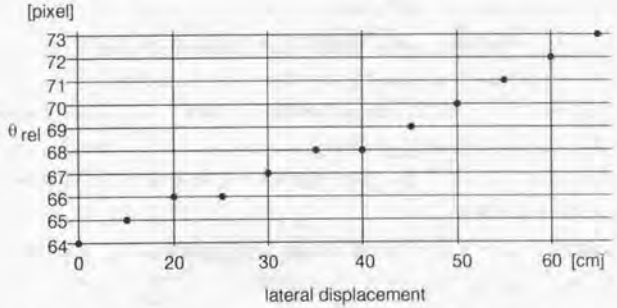


Figure 5.13 Experimental result lateral displacement using 90-degree matching.

5.3 ビューシーケンスに基づく行動

5.3.1 前進

前進中には，まずはじめに大域的マッチングを利用してビューシーケンス中の自己位置の推定を行なう．もしその結果として得られる e が小さければ，位置推定は成功したとみなされ，次に局所的マッチングを用いて横方向のずれが検出される．最後に全方位ビューのうち進行方向の部分に変化(障害物)がないか調べられる．障害物がなければ，ロボットは一定速度で走行を続ける．e の値が大きい，もしくは進行方向に障害物が検出された場合は，e の値が小さくなる，もしくは障害物がなくなるまで停止する．

これまでの我々の提案したナビゲーション手法では，マッチングの結果として回転と横方向のずれを別々に検出することはできなかった．そこでロボットの制御は，「右にステアリングを切りながら前進すれば，画像は左にずれていく」という定性的な性質のみを利用して，記憶した見え方に近付くようにフィードバックをかけていた．

これに対して，全方位ビューシーケンスを用いると，横方向のずれを前方と後方の局所的マッチングを行うことで検出することができる．**Figure 5.14** は自律走行時の局所的マッチングの様子を示している．もし θ_2 が 180[deg] からはずれる場合，ロボットが記録走行時の経路上を正しく進んでいないことになる．したがって，ロボットは θ_2 が 180[deg] に近付くように進行方向を制御する．これは，石黒ら [33] が複数カメラを持つ移動ロボットを経路に正確に追従するために用いた制御方法である．

また，θ_2 が 180[deg] であるときはロボットは正しい経路上にいるが，まだ姿勢がずれている可能性がある．そこで θ_1 が 0 になるように制御される．これはこれまでのビューシーケンスを用いた走行での制御方法であり，経路からのずれが大きい場合の経路への追従性はよくないが，経路からのずれが小さい場合には十分な性能を示すことが分かっている．

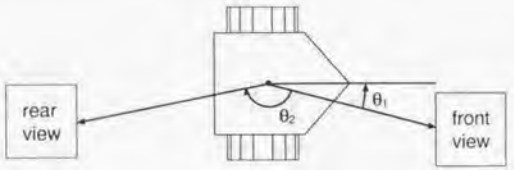


Figure 5.14 Control in going forward action.

5.3.2 右左折

右左折を行う前に，ロボットは記録走行における右左折の位置に正確にいることが望ましい．しかし，ビューシーケンスを用いた我々の以前の誘導手法では，記憶されたビューのうちのある一枚は，0.5 ~ 1.0[m] の範囲の走行時のビューにマッチする．この自己位置の認識できる分解能は，画像を記憶する間隔と等しく，これよりも正確にロボットが位置合わせをすることはできなかった．

これに対して，前方と側方の画像を用いて2度の90度マッチングを行うことで，右左折の時のより正確な位置合わせが可能となる．**Figure 5.15** は，左折する場合の位置合わせの様子を示している．ロボットは前方と左の画像を同時にマッチングに利用しており，θ が，曲がる角度に等しくなるところで停止すれば，記録走行での左折地点に正確に位置を合わせることができる．次にコーナーリングを行うが，その間は360度マッチングだけを行い，画像の向きが記憶したものと一致するまで回転する．

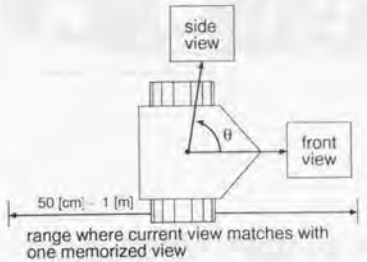


Figure 5.15 Localization before cornering.

5.4 経路誘導実験

5.4.1 教示走行

Figure 5.16 はロボットが走行する経路を示す．実験環境は前章での実験と同じく工学部9号館の屋内である．経路の全長は 31[m] で，ロボットは研究室の前からエレベータの前まで，3回の右左折を行いながら進む．また，**Figure 5.17**には第二教示走行で記憶した全方位ビューシーケンスの中から 8 つのビューを示した．それぞれのビューは経路中の対応する番号の地点で記憶されたものである．

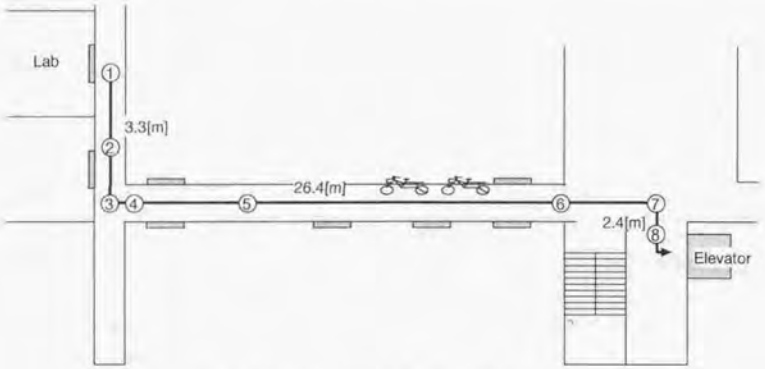


Figure 5.16 Experimental Environment.

5.4.2 自律走行

自律走行の様子を **Figure 5.18** に示す．自律走行中は，スピードは約 0.3[m/s] に保たれている．経路全体を走行するのには，右左折を含めて約 130[s] かかった．全体として，我々のこれまでのビューシーケンスを用いる誘導手法よりも正確に教示経路へ追従して走行することができた．さらに，ビューシーケンス中のビューの順番およびそれぞれのビューの向きを逆にすることで，ロボットは同じ経路を戻って行くことができるようになった．

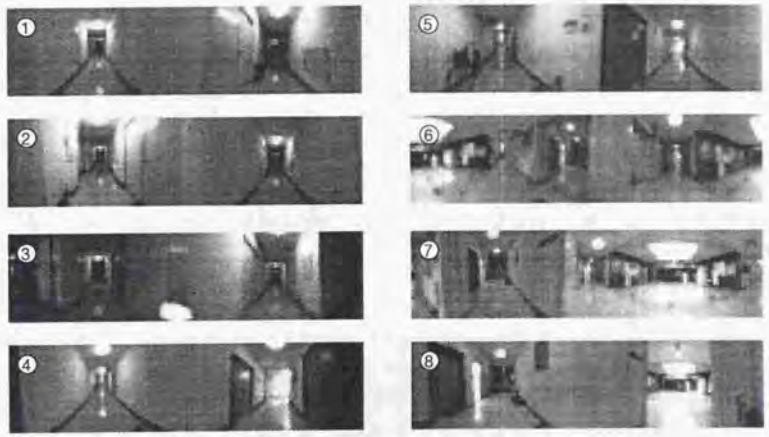


Figure 5.17 Samples of memorized Omni-Views in the route.

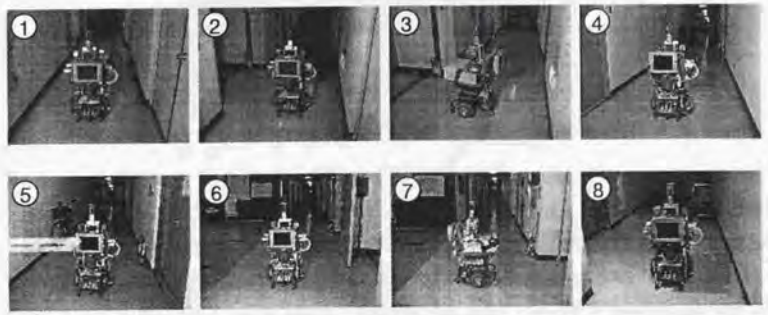


Figure 5.18 Snapshots in autonomous run.

5.5 全方位ビューシーケンスによる改善点

5.5.1 位置誤差を与えた場合の走行軌跡

Figure 5.19 には，ロボットの初期位置に 40[cm] の横方向のずれを与えた時の軌跡を実線で示す．実線は，全方位ビューシーケンスを用い，360 度マッチングによる位置の認識を行ない，かつ前後の画像を用いた 90 度マッチングにより経路のからのずれを認識しながら走行した場合の軌跡である．また，点線は前章で述べた前方の画像だけを用いて走行する手法を用いた場合の軌跡である．本章での制御では，ロボットは横方向のずれを，姿勢のずれから分離して検出することができ，これまでよりも素早く正しい経路に戻ることができていることが確認された．

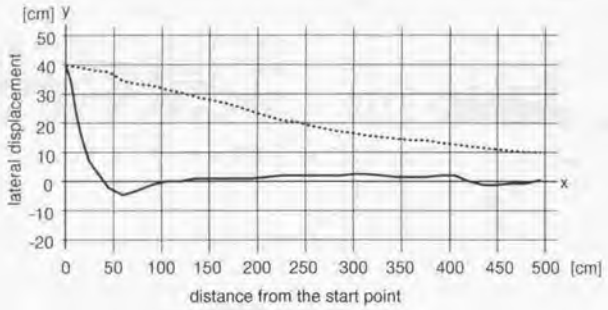


Figure 5.19 Trajectory of autonomous navigation.

5.5.2 マッチングの安定性

全方位ビューの視野の広さにより，前章で述べた通常のカメラを用いたビューシーケンスと比較してどの程度マッチングの性能が改善されているか調べた．実験は **Figure 5.20** に示すエレベータ付近で，ロボットをエレベータのドアに向かって動かして行なった．**Figure 5.21**は，通常のカメラと全方位センサを用いて得られるビューの例を示している．通常のカメラの視野は 50[deg]，また全方位ビューの後部に人間の姿が映ってしまうので，ここでのマッチングは 360 度テンプレートの代わりに，視野を前面だけに限定した 180 度テンプレートを用いる．

まず前方のビューのみを利用してビューシーケンスの生成を試みた結果を **Figure 5.22**に

示す．ビューのサイズは 32[pixel]×32[pixel] で視野は約 50[deg] である．ビューを新たに記憶する際のエラーの閾値は 8500 と定めた．この値は通常の廊下で移動したとき，約 0.5[m] 間隔でビューを記憶する値である．単眼カメラで得られるビューは視野が狭いため，1.5[m] 前方にあるエレベータのドアに画像が占有されてしまう．そのためロボットを移動させてもマッチングエラーは閾値を越えるようには単調増加せず，ビューシーケンスの生成が失敗していることがわかる．

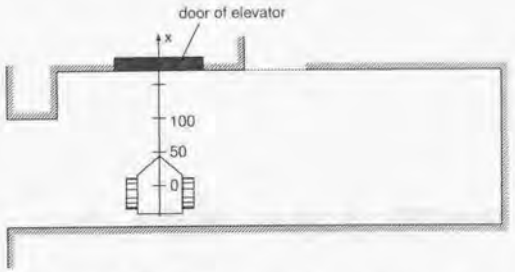


Figure 5.20 Enviromnent in front of elevator.

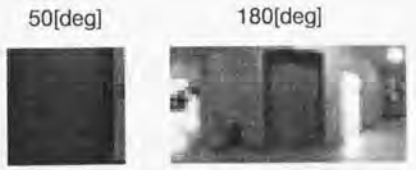


Figure 5.21 Veiws acquired using normal camera (left) and omnidirectional vision sensor (right).

一方，全方位ビューを利用してビューシーケンスの生成と，自律走行時の位置推定を行なった結果を **Figure 5.23**に示す．自律走行時には人間が後ろからロボットを押して，教示走行と同一直線上を正確に移動させ，ロボットにはマッチングと位置認識のみを行なわせた．これは，ロボットの制御のふらつきがマッチングエラーに影響するのを防ぐためである．この実験の結果から，全方位ビューは視野が広くその分多くの画像情報を持つためにマッチングが安定し，ビューシーケンスの生成およびビューシーケンス内での位置推定が正しく行なわれていることが確認できた．

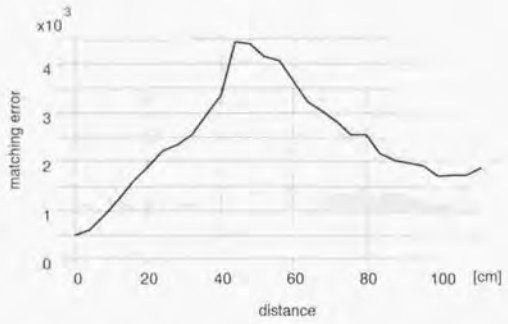


Figure 5.22 Matching error using normal caemra.

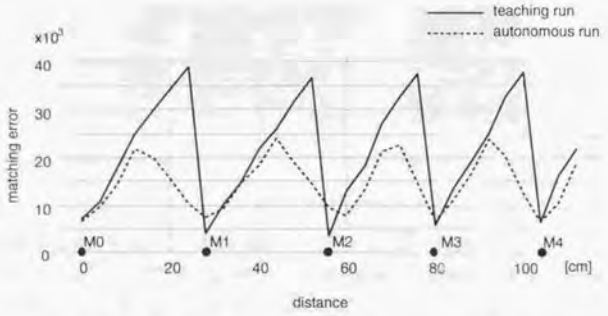


Figure 5.23 Matching error using omnidirectional vision sensor.

5.5.3 双方向の経路誘導

前章で提案したビューシーケンスは前方の画像のみを記憶しており，“ある地点から別の地点まで直線経路の繰り返しにより構成された一方向の経路”を表現していた（Figure 5.24a）が，全方位ビューシーケンスでは，前方のみならず後方の画像も記憶している．（Figure 5.24b）そのためにマッチングに用いるするテンプレート位置を 180 度ずらすことで，教示した経路を逆向きに移動することが可能となる．このため，1 回の経路教示だけで 2 地点間を往復する行動が実現でき，本手法の適用範囲が広がったと言える．

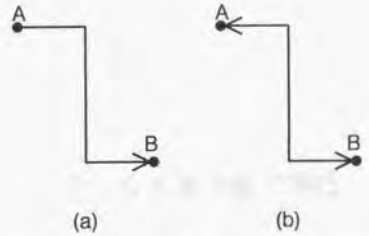


Figure 5.24 Routes represented by former view sequence (a) and omni-view sequence (b).

第 6 章

自律的な地図表現の獲得

6.1 地図表現

6.1.1 経路表現と地図表現

第 3 章で述べたビューシーケンスは **Figure 6.1**(a) のに示すような一方向の経路を，また第 4 章で述べた全方位ビューシーケンスでは **Figure 6.1**(b) に示すような双方向の経路をそれぞれ表現していた．これらは「ある地点から別のある地点まで行くための経路情報」であり，「地図表現」と呼べるような汎用のものにはなっておらず，ロボットの移動能力は限定されている．地図は経路よりも上位のものとして位置付けられ，**Figure 6.1**(c) に示すように複数の地点（ノード）とそれらの間をつなぐビューシーケンス（リンク）から構成される．この情報から任意の地点を結ぶ経路を生成することができるようになり，地図上の任意のスタート地点からゴール地点までの移動が可能になる．

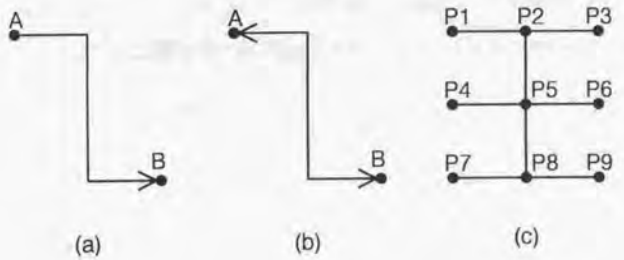


Figure 6.1 (a):route representation for one way, (b):route representation for both ways, (c):map

ビューシーケンスによる地図表現は，複数の地点を定義しそれらの間をつなぐビューシーケンスをつなげていくことで作ることができる．ここで単なるビューシーケンスを用いる場合には，双方向 2 つのビューシーケンスの組により 1 つのリンクを表現する必要があったが，全方位ビューシーケンスを用いれば 1 つで 1 つのリンクを表現することができ都合がよい．

ところで，前章までのビューシーケンスおよび全方位ビューシーケンスでは，ロボットは教示走行おいて経路を獲得していた．しかしこの方法により地図表現を作成するためには，オペレータが建物内のあらゆる位置にロボットを動かしビューシーケンスを記憶させ，かつビューシーケンス間の重複部分を除きながら，接続していくという手間のかかる作業が必要になる．また全方位ビューシーケンスを教示する時は，オペレータがそばにいるとその人が映ったビューを記憶してしまうので，さらに記憶のさせ方に工夫が必要になる．

しかし，このような手間のかかる作業は，ビューシーケンスの本来の特徴であった「教示が容易である」という利点に反しており望ましくない．そこで本章では，ロボットが未知環境において自分で動き回ってビューシーケンスを記憶し，地図表現を構築することを目指す．

6.1.2 地図表現の獲得の手順

地図表現を自律的に獲得するためには，まずはじめにロボットが環境に関する情報を何も持たない状態（未知環境）で，自律的に動き回ることが必要である．通常このような移動ロボットのタスクには超音波センサによる障害物回避を利用することが多いが，本研究では視覚のみを持つ移動ロボットを対象としている．そこで，本章では視覚を用いて移動可能な空間を検出する方法を開発する．ここでは屋内環境を，廊下の空間を認識して直進し，交差点を認識して角を曲がることで動き回ることができるものと仮定し，

a. ステレオ視覚によりロボット前方の開空間を見つけ，その方向に前進する機能

b. 全方位視覚を利用したオプティカルフローで，横方向の開空間を見つける機能

の2つを用意する．

6.2 ロボット前方の空間の検出

6.2.1 テンプレートマッチングによるステレオ視

本節では，ステレオ視覚を利用して，前方にロボットが進める開空間があるかどうかを検出する手法について述べる．左右に並べたステレオカメラで同一対象を見た場合見え方に差（水平視差）が生じるが，この視差は対象までの距離に依存する．すなわち対象までの距離が遠ければ視差は小さく，近ければ大きくなる．この視差を用いると三角測量の原理によりロボットの前方に見えている対象までの距離を求めることができる．そしてロボット前方の一定距離内に何もなければ，進むのに十分な空間があると判断できる．

左右2枚の画像から対応する同一対象の領域を取り出す処理は対応点探索と呼ばれるが，これには大きく分けてエッジ等の特徴ベースのものと，相関ベースのものがある．本研究では後者の方法の単純なテンプレートマッチングを用いる．一般に相関による対応点探索は，線分などの特徴を用いるものと比較して

- エッジ抽出などの前処理を行なう必要がない．
- 線分抽出のような，情報の抽象化が必要ない．
- 細かい模様やグラデーション等，線分以外のテクスチャの情報が有効に利用される．

といった利点がある反面，

- 計算量が多い．
- 探索空間が大きい．
- 適切なウィンドウサイズを決定するのが難しい．

という問題がある．しかし本研究ではそれぞれ，

- 相関演算LSIによる高速なテンプレートマッチングを利用し，またステレオ画像をフィールド多重化技術により1枚のボードに入力することで効率良いマッチング処理を実現する．
- 粗密(Coarse-to-fine)探索を行なうことで探索空間を限定する．
- テンプレートマッチングにおける信頼度を導入することで，粗密探索における複数の解像度のうち最も良いもの決定する．

という工夫により対処する．

6.2.2 ステレオマッチング

対応点探索には相関演算 LSI による高速なテンプレートマッチングを利用する．また 2 台のカメラから得られるステレオ画像は，3.1 で述べたフィールド多重化技術により 1 枚のボードに入力することで効率良くマッチング処理を行なう．

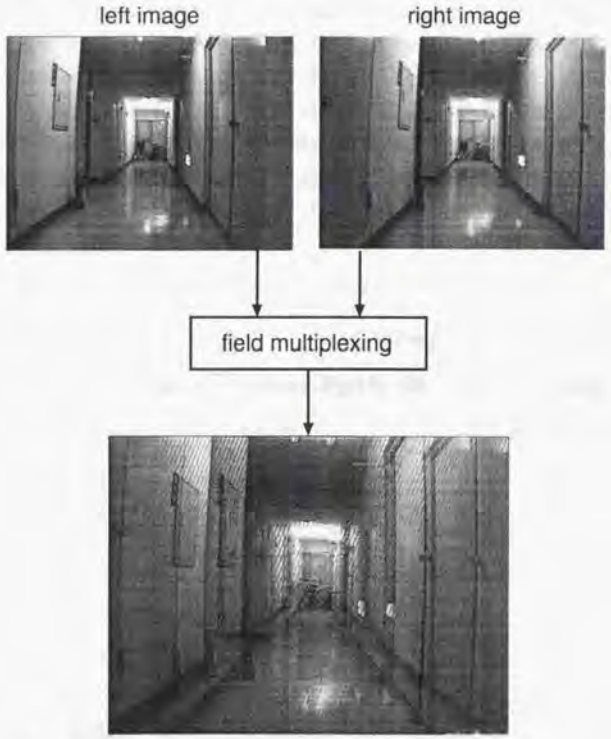


Figure 6.2 Multiplexed stereo image used for stereo matching.

ステレオマッチングにおいては，テンプレート画像として右カメラの画像（VRAM 上の偶数ライン）から 32 × 16（縦方向には 1 ラインおきの画像を用いるので，画面上の見かけの大きさは 32 × 32 の正方形）のテンプレートを水平方向に 32 個とり，各テンプレートに対して

左カメラからの画像（VRAM 上の奇数ライン）を探索画像とする．このような小さいサイズのテンプレートのみを用いて対応点を探索すると，

- 探索範囲が画面上のエピポーラ線である横 1 ラインとなり計算量が多い．
- テンプレートに特徴が少ない場合に，誤マッチングが発生しやすい．

ということが問題になる．そこで，ここではマッチングを大域マッチングと局所マッチングの 2 つに分け，粗密法と組み合わせる．また，マッチングの信頼度を定義することで，粗密法による異なる解像度でのマッチング結果を統合する．

大域マッチング

大域マッチングは，用いる画像の解像度を下げた上で，エピポーラ線上の広い領域を探索範囲として水平方向に広く探索し，対応領域の大まかな位置を見つけるものである（**Figure 6.3**右）．粗密法における画像の間引き間隔を n とすると，テンプレート画像は (32 ×n) × (16 × 2n) 画素，探索領域はテンプレートの位置から右側へ水平に 128 画素，垂直に ± 16n/2 画素とることにした．本研究で使用する画像処理ボードでは，1 回の相関演算の探索範囲は縦横 16n 画素であるので，(128/16n) ×1 回の相関演算を行なうことになる．ここでは，間引き n=4 としたので，画像上での見かけのテンプレートサイズは 128 × 128，相関演算は 2 回となる．

局所マッチング

局所マッチングでは探索範囲を徐々に狭めながら，大域マッチングあるいは一つ前の局所マッチングで見つけた位置の近傍だけを探索してより正確な対応を求める．テンプレート画像のサイズは (32 ×n) × (16 × 2n) で大域マッチングと同じであるが，探索範囲は水平・垂直方向方向に ±16n/2 づつとる．局所マッチングでは相関演算は 1 回だけ行なう．

6.2.3 複数解像度のマッチングの統合

エラー間引きを大きくして大域マッチングを行なえば，テンプレートが大きくなり，広い領域の画像情報が含まれるようになるので，正しいマッチング結果が得られる可能性が高い．しかし，マッチング結果得られる位置の解像度は低い．次に間引きを小さくしながら局所マッチングを行なうことで，探索範囲を限定して計算量を減らした上で，マッチング位置の分解能を（最終的には 1 画素まで）上げていく．この時テンプレートの画像上のサイズは徐々に小さくなるので，テンプレートに含まれる特徴がなくなり誤対応が発生する可能性が高くなる．

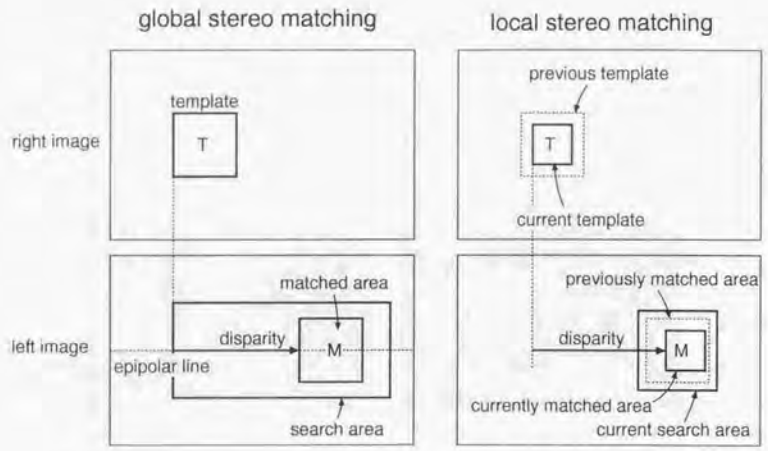


Figure 6.3 Global stereo matching (left) and local stereo matching (right).

そこで，ここではこれら大域的 / 局所的の複数のマッチングの結果を統合するための指標として「マッチングの信頼度」を定義する．**Figure 6.4**は 典型的な 3 種類のテンプレートパターンを示す．上から，マッチングが正しく行なえない一様なパターン (plain) とエッジ状のパターン (edge)，それにマッチングに適した特徴的な画像 (trackable) である．これらの画像について，中心枠内のテンプレート領域 (16×16[pixel]) と，その周囲の 32×32[pixel] を探索画像としてマッチング（自己相関）を行った結果がそれぞれの右のディストーションの分布である．探索範囲は 縦横それぞれ -8 ～ +7 である．

trackable のテンプレートには特徴（模様）が多く含まれているため，位置相関値分布の中心部 (0,0) での値が周辺での値に比べて際だって小さくなっており，位置 (0,0) でマッチしたという信頼度は極めて高い．一方，edge では 水平方向に模様の変化が乏しいために，水平方向のディストーションがほぼ一定となっている．上図では，さらに画像全体に特徴がないために，有意にディストーションが小さくなる位置が存在しない．plain や edge のように特徴の少ない画像を用いたマッチングではディストーションが最小になる位置をテンプレートマッチングにより求めても，周辺位置のディストーションと比べて際だって小さくはならない．以上のことから，マッチした位置でのディストーション（最小ディストーション）とその周辺位置でのディストーションの比を利用したマッチングの信頼度を導入する．**Figure 6.5**のように

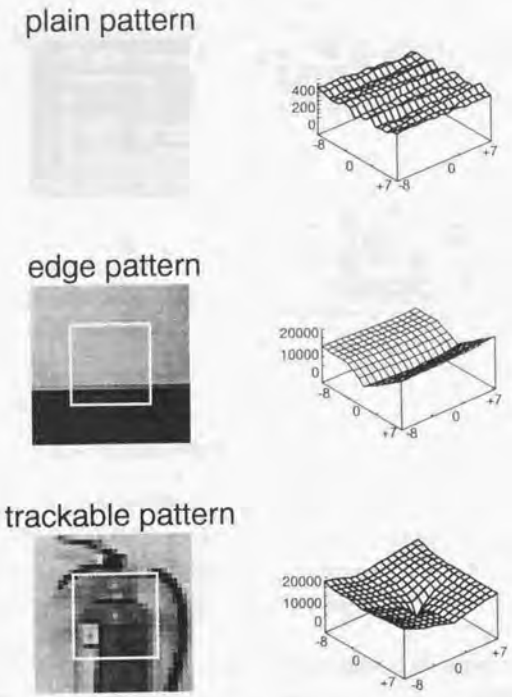


Figure 6.4 Template and distortion array in auto-correlation.

位置 (vx_0, vy_0) で最小ディストーション (d_{min}) をとってマッチしたとする．(vx_0, vy_0) を囲む 16 点 (色をつけた部分) でのディストーションのうち最小のものを d_n とし，次式によりマッチングの信頼度 $reliabiilty$ を定義する．$reliabiilty$ は 0 ～ 100 の値をとり，大きい値はマッチングの信頼度が高いことを意味する．

$$reliabiilty = \frac{d_n - d_{min}}{d_n} \times 100 \tag{6.1}$$

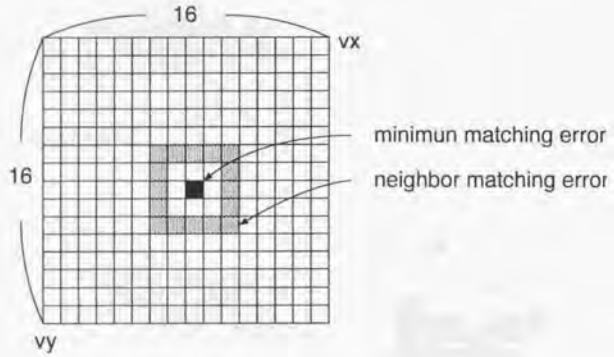


Figure 6.5 Minimum and neighbor matching error.

複数の大域的 / 局所的マッチングの結果を統合するためには，各マッチングにおいてディストーションとともにこの信頼性を求める．間引きが大きい場合は，テンプレートの画面上の見かけのサイズが大きいため，trackable なパターンである可能性が高い．間引きが小さくなってもテンプレートが trackable な場合だけ，より分解能が高いそのマッチング結果を利用することになる．

信頼度を用いた粗密法の効果を確認するため，実際にマッチングで視差を求めてみた (**Figure 6.6**)．図の画像には，中央に右目の画像だけ見えている部分がある．この部分から 32 個のテンプレートをとり，対応領域を探索し視差を求める．結果を各テンプレートの中心位置に ○と×で表示している．○印はある閾値以上の信頼度でマッチングが行なえた位置であり，その径が視差の大きさを表している．×印はマッチングの信頼度が低く，求めた視差を無視していることを表している．上図は一度だけのマッチング (間引きなしの大域マッチングに相当) で対応領域を求めた結果である．下図はまず間引き 4 の大域マッチングを行ない，つぎに局所マッチングを行なった結果である．上図では特徴の少ない部分 (廊下の壁，扉など) ではマッチ

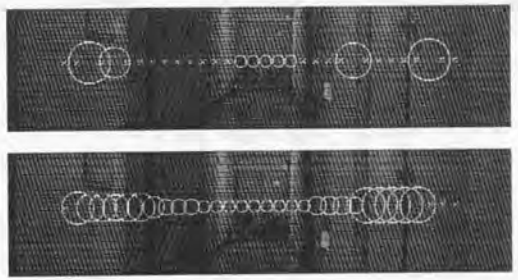


Figure 6.6 Experimental result of stereo matching with and without coarse-to-fine strategy.

ングがうまく行なえていないため視差が求まっていないが，下図では大まかな視差が求まっている．またマッチングに要した時間 (32 個の合計) を計測すると，上図では約 120[ms]，下図では約 75[ms] であった．信頼度を用いた粗密を用いることで相関演算の回数が減少したことが分かる．

6.2.4 空間の検出

Figure 6.7はステレオ視覚系を上から見た図である．左右のカメラは同一水平面内 (xy 平面内) にあり，各カメラの光軸は x 軸と垂直に固定してあると仮定する．また，左右のカメラの焦点をそれぞれ $F_R(D,0)$,$F_L(-D,0)$，焦点距離を f とする．いまある点 P が右カメラでは p_R，左カメラでは p_L に結像したとする．各カメラの光軸から p_R,p_L までの距離をそれぞれ X_R,X_L として，X_R,X_L から点 P の位置 (x_p, y_p) を求める．

F_R と p_R，F_L と p_L を通る直線をそれぞれ l_1,l_2 とすると，

$$l_1 : y = \frac{f}{X_R}(x - D) \tag{6.2}$$

$$l_2 : y = \frac{f}{X_L}(x + D) \tag{6.3}$$

式 (6.2) と式 (6.3) を連立させて交点 P の座標 (x_p, y_p) を求めると，

$$x_p = \frac{X_L + X_R}{X_L - X_R}D, \quad y_p = \frac{2}{X_L - X_R}fD \tag{6.4}$$

式 (6.4) における X_R,X_L の単位は長さであるが，実際に画像処理を行なって p_R,p_L の位置を求める場合，その位置は [pixel] で表すことになる．また式 (6.4) では焦点距離 f を定数と

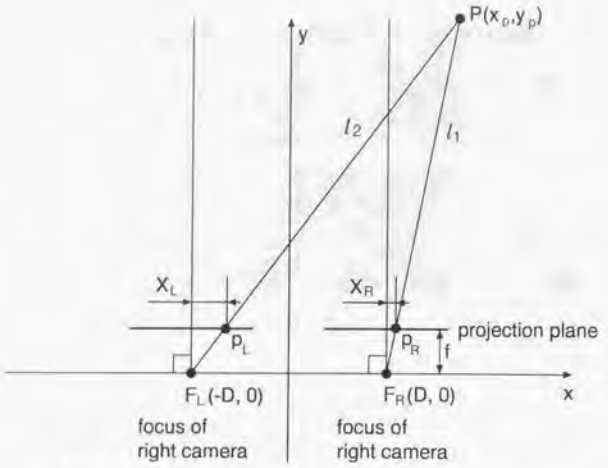


Figure 6.7 Transformation from disparity to distance.

して与える必要があるが，f を直接計測することは困難である．そこで，画素の番号 n_R, n_L，及びカメラの画角 γ (計測しやすい) を使って式 (6.4) から X_R, X_L, f を消去することにする．

カメラの画像面と光軸の交点を C，C から n=0,1,…,N-1 の N 個の画素を考え，その端点を E，CE の距離を w，C から E までの画角を γ とする．

$$\tan\gamma = \frac{w}{f} \tag{6.5}$$

一方，C から画素 n の中心までの距離 $X(n)$ は

$$X(n) = \frac{w}{2N} + \frac{w}{N}n \tag{6.6}$$

式 (6.6) に n_R, n_L を代入して，

$$X_R = X(n_R), \quad X_L = X(n_L) \tag{6.7}$$

式 (6.5)，式 (6.7) を式 (6.4) に代入して整理すると，

$$x_p = \frac{n_L + n_R + 1}{n_L - n_R} D, \quad y_p = \frac{2}{n_L - n_R} \frac{N}{\tan\gamma} D \tag{6.8}$$

N[pixel] 分の画角 γ，及びカメラの位置 D をあらかじめ計測しておき，p_R、p_L の位置を求めれば，式 (6.8) により対象 P の座標を計算できる．

式 (6.8) の定数のうちカメラの位置 D は長さを計測するだけで値が決定でき，結果は D=0.15[m] である．また，カメラ (SONY EVI-330 倍率 0.6 倍の広角レンズを使用) の N=304[pixel] 分の画角を計測する．$\tan\gamma$=0.58(γ = 30 [deg]) であった．

次に，ロボットを廊下に置いて視差を求め，三角測量でロボット前方の空間を検出する実験を行なった．まず粗密法を用いて視差を求めた結果が **Figure 6.8** である．次に 32 個のテンプレートについて，そのテンプレート中に映っている対象の位置を計算した．ロボットの右カメラの位置を原点として計算結果をプロットしたものが **Figure 6.9** 左図である．各点は，画面上のテンプレートの水平サイズの分だけの幅を持たせてある．また図中の点線は実際の廊下の形を示している．次に，カメラから各対象の位置までは物体が存在しないと仮定し，原点から各対象までの空間を塗りつぶす．この作業を 32 個すべてのテンプレートについて行なった結果が **Figure 6.9**右図で，ロボット前方に認識された開空間を示す．

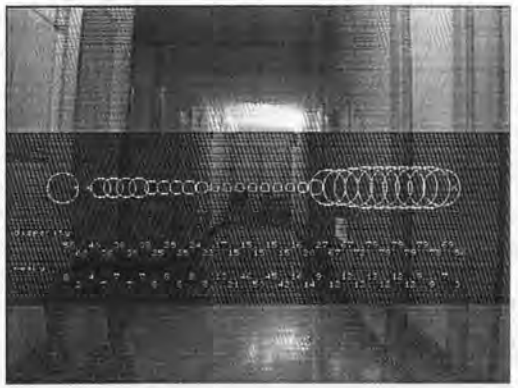


Figure 6.8 Result of disparity detection.

Figure 6.9では，ロボット正面の壁，つまり廊下の行き止まりの壁の位置は比較的正確に求まっているが，側面の壁の位置はあまり正確に求まっていない．しかしここでの目的は，ロボット前面の開空間の方向を求めることである．ロボット正面の壁の位置が求まれば，その壁の方向に開空間があることがわかるので，側面の壁の正確な位置が求まらなくても問題ないと思われる．

次に進行方向の決定方法を述べる．**Figure 6.9**右図を用い，図の上から開空間を見ていき，その幅がある値以上である直線 L を求める．見つけたライン上の，塗りつぶされた部分

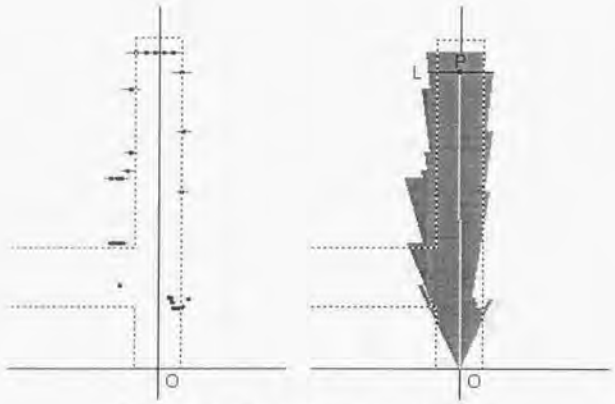


Figure 6.9 Result of open space detection.

の中心点をPとして，この点Pをロボットが前進するときの目標とする．すなわち，Pが画面上で左右の中心に一致するように制御する．「一定幅以上塗りつぶされている」という条件を加えたのは，

- ノイズのため，幅の狭い開空間が突発的に現れることがある．このような開空間を目標にすると，進行方向がふらつく．
- 開空間の幅がロボットの横幅より狭ければ前進できない．

という理由からである．

6.2.5 走行実験

ステレオ視覚により前方の開空間を検出して，その方向に前進する実験を **Figure 6.10**に示す廊下環境で行なった．廊下の中心軸上からスタートさせた場合，そのまま廊下の中央に沿って進んだ．またロボットを 50[cm] 横にずらしてスタートさせた場合，約 5[m] 前進すれば中心軸に収束することを確認した．ビジュアルフィードバックのゲインを大きくすれば収束が速くなると思われるが，一方振動も大きくなる．自律的に前進しながら全方位ビューを記憶する場合を考えると振動はないほうが良ため，ゲインは振動しない程度に小さくする．

Figure 6.11は壁の手前 3[m] の位置 (**Figure 6.10**の A 地点) での処理画像である．このように廊下の行き止まりが近付くと，カメラの画像の大部分が正面の壁の画像となる．すると

正面の壁とその他の部分を区別できなくなるので「廊下の行き止まりの壁に向かって進む」ということができなくなる．そこで本研究では，正面の壁までの距離を計測し，進行方向が決定できなくなるほど壁が接近する前に停止することにした．この距離は廊下の横幅にも依存するが，**Figure 6.10**の実験環境で行なうときは，壁の手前約 4[m] の位置 (**Figure 6.10**の S 地点) とした．

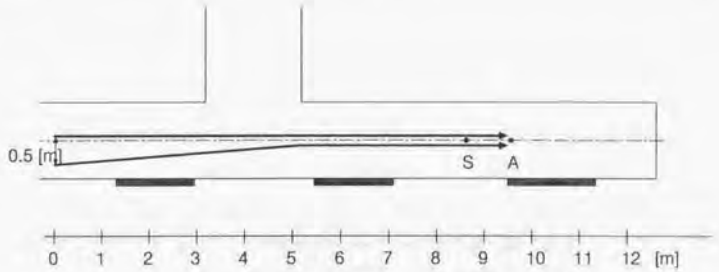


Figure 6.10 Experimental environment.

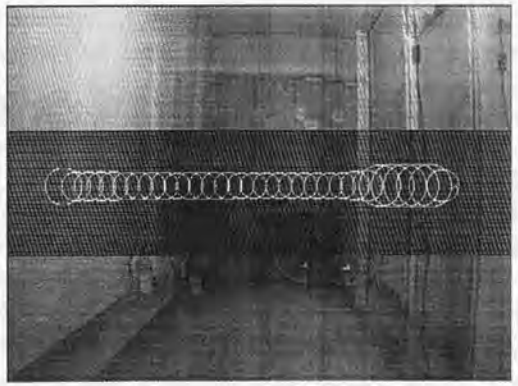


Figure 6.11 An image near the end of a corridor.

6.3 ロボット側方の空間の検出

ここでは，ロボットが廊下を走行しながら交差点を発見する方法について検討する．もし側方を向いた両眼カメラシステムがあれば，前節で述べた開空間の検出方法をそのまま利用して，交差点を認識することは可能である．しかしその場合，左右の廊下を検出するには，左右にそれぞれ両眼カメラシステムが必要となり，移動ロボットのシステムが大きくなる．また，前方を向いた両眼カメラシステムを必要に応じて左右に振ることは，ロボットの連続した動きの妨げになり，処理も複雑になるために好ましくない．

そこでここでは，走行しながら側方の開空間を検出する手段として，全方位画像上での運動視差を利用する．運動視差とは，カメラの移動によって発生する観察対象の位置の移動量であり，対象までの距離が近いほど同一の移動による運動視差は大きくなる．つまりロボットの側方が壁であれば運動視差は大きくなり，開空間があれば運動視差は小さくなる．

ここで求めたい運動視差は，開空間があるかないかだけを判断するのに用いるので，精度はそれほど必要ない．そのため，全方位ビュー (128×32) 上で，一定の走行距離ごとにオプティカルフローにより運動視差を求め，側方の空間を検出することにする．

6.3.1 全方位画像上の運動視差

オプティカルフローは現在位置 x_0 での全方位ビューと，l だけ手前の位置 $x_0 - l$ で記憶しておいた全方位ビューの間のマッチングを行い求める．現在位置での全方位ビューは，ロボットが位置 $x_0 + l$ に来たときのマッチングに利用するので記憶しておく．距離の計測にはロボットの車輪のエンコーダを用い，一定距離 $l/2$ 前進するごとに上記の処理をおこなう．実験では，l=30[cm] とした．

オプティカルフローの生成では，**Figure 6.12** のように位置 $x_0 - l$ で記憶しておいた全方位ビューの左右各側面からテンプレート画像 (16×16，水平方向の視野 45[deg] に相当) をとり，現在位置 x_0 での全方位ビューの左右各側面を探索画像 (32×32，水平方向の視野 90 度に相当) としてテンプレートマッチングを行うことで求める．この結果得られる左右のオプティカルフローの水平成分 (-8[pixel] ～ +7[pixel]) をそれぞれ f_l f_r とする．もしロボットが回転することなく前進しただけであれば，f_l f_r が運動視差となる．

しかし実際には $x_0 - l$ から x_0 に前進する間に多少回転する可能性があり，回転した場合 **Figure 6.12** のようにビュー全体が水平方向にずれるので単純に f_l f_r をロボットの前進による運動視差とみなすことはできない．そこでまず全方位ビューの前面の画像を利用して，回転量を求める．すなわち位置 $x_0 - l$ で記憶しておいた全方位ビューの前面からテンプレート画像 (16×16) をとり，現在位置 x_0 での全方位ビューの前面を探索画像 (32×32) としてマッチングをおこない，ロボットの回転量 f_u とする．前面の画像にずれ f_u が生じる理由として，

- ロボットの回転
- ロボットの直進経路に対する垂直方向のずれ

が考えられるが，この場合 l=30[cm] 前進する間にそれほど大きく直進経路からずれるとは考えられないので f_u をロボットの回転量とみなすことができる．そして，左右各側面のずれ f_l f_r から回転によるずれ f_u を引いた値をロボットの前進による運動視差とみなす．

また，ロボットの側面に廊下の壁のように一様で特徴の少ない対象が見えていると，マッチングが正しく行われない可能性がある．しかし側面に空間がある場合は，遠くの風景が見えるので画像中に特徴が多く，マッチングの信頼度は高くなる．そこでステレオマッチングで行なったのと同様にオプティカルフローにおいてもマッチングの信頼度を求めておき，横方向の空間を発見する際フローの値だけでなく信頼度も評価する．つまり，

- オプティカルフローの信頼度がある閾値以上である．
- 運動視差の大きさががある閾値以下である．

という 2 つの条件を満たす場合にのみ，側方に開空間が存在するとみなす．

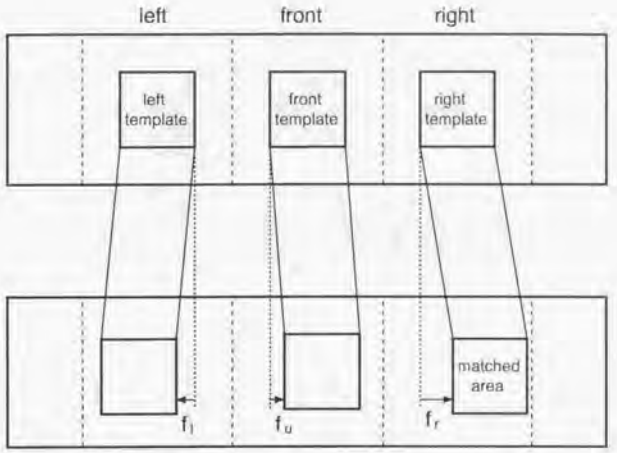


Figure 6.12 Optical flow detection on omni view.

6.3.2 交差点の検出実験

廊下の交差点付近でロボットを動かし交差点を検出する実験を行なった．その実験環境を Figure 6.13に示す．人間がロボットを後ろから押しながら前進させて左右各側面の運動視差を求めた．両側が壁の場合の実験結果が Figure 6.14，また交差点で左側に開空間がある場合の結果が Figure 6.15である．図中には，それぞれの場所でのマッチングに用いたテンプレート画像，探索画像，運動視差の値，オプティカルフローの信頼度を示してある．

この結果をみると，右側が壁の場合の運動視差は 7，左側が壁の場合の運動視差は -7 となっている．また左側に開空間がある場合には -1 となっており壁である場合との差ははっきりと出ている．オプティカルフローの信頼度は，開空間の場合 0.41 と高く，壁の場合はパターンによっては 0.2 程度の低い値をとることが分かる．

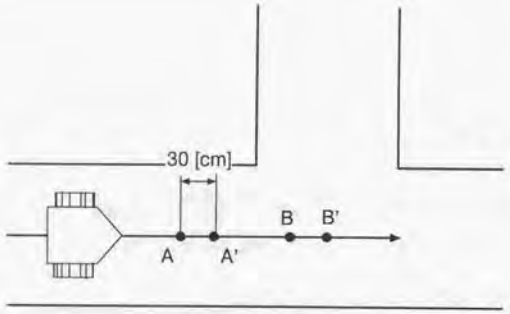
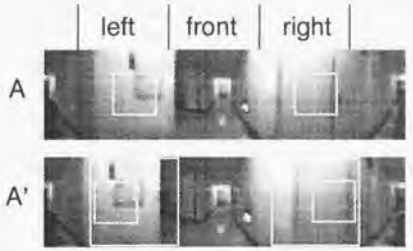
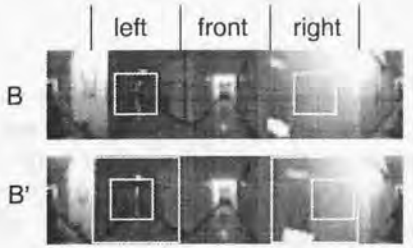


Figure 6.13 Experimental environment.



	left	right
flow	-7	+7
reliability	0.23	0.50

Figure 6.14 Detection of motion stereo disparity in corridor.



	left	right
flow	-1	+7
reliability	0.41	0.21

Figure 6.15 Detection of motion stereo disparity at crossing.

6.4 自律的地図獲得

6.4.1 地図作成の手順

ここでは地図表現の簡単化のため，「建物内の通路は直交する直線経路から構成される」と仮定し，各直線経路についてのビューシーケンスを生成し，交差点でリンクさせたものを地図とする．ビューシーケンスを生成する際，あるいは交差点でビューシーケンスをリンクする際，ビュー間のリンク（隣接関係）は北(N)，南(S)，西(W)，東(E)の四方向で記述する．その上でロボットは **Figure 6.16** に示す戦略に従い，

1. 前方の開空間を検出しながら前進する
2. 直前に記憶したビューと現在のビューを比較し変化があれば，新たなビューを記憶する
3. 横方向の開空間（交差点）を発見したら，その位置をスタックに保存する
4. 行き止まりにつきあたったら，スタックから交差点の位置を取りだし，その位置に戻って，未探索環境へ進む

という行動を繰り返しながら，地図表現を作成していく．この地図表現にはビューの列とその接続関係が蓄えられていくが，そのビューのうちの必要なものには後で名前（シンボル）を付けることで，人間がゴールとして指定できるようにする．

前進およびビューの記憶

ロボット自身で前方の開空間を見つけ，その方向に前進する．前進中得られる全方位ビュー I と直前に記憶したビュー M_i との間で 360 度マッチングを行ない，エラーがある閾値を越えたら I を M_{i+1} として記憶する．そしてビューを記憶したら，ビュー間のリンク（隣接関係）を張る．例えば現在の進行方向が北であれば，「ビュー M_i の北にはビュー M_{i+1} がある」と記憶する．

交差点の発見

前進中は車輪のエンコーダで距離を計測し，一定距離前進するごとにロボット側方の運動視差を求める．信頼度が高く，かつ小さい値の運動視差が得られれば，側方に開空間があるとみなし，その地点での全方位ビュー M_i を記憶する．またスタックに交差点のビューの番号 (i) と開空間の方向 (NEXT_i) を保存し，未探索の通路があることを記憶しておく．

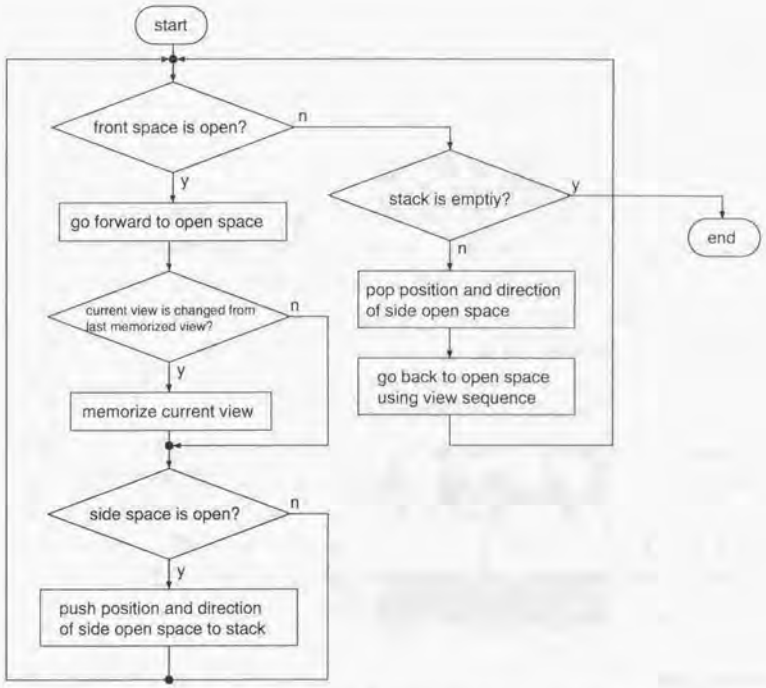


Figure 6.16 Flowchart of autonomous map acquisition.

新たな直線経路の探索

前進を続け前方に開空間がなくなれば，その直線経路についてのビューシーケンスの生成は終了する．そしてスタックから交差点の位置 (i) および開空間の方向 ($NEXT_i$) を取り出し，交差点の位置まで移動する．移動の際，地図全体は不完全であっても，その時点までに記憶したビューの中から，交差点までのビューを並べて経路を作成すれば自律移動できる．地図を利用した自律移動の方法については 6.4.2 で後述する．移動が終了したら，開空間の方向を向き，未探索の直線経路のビューシーケンス作成を開始する．交差点でのビューを M_i，未探索の直線経路に入って最初に記憶したビューを M_j とすると，M_j と M_i との間にリンクを張る．

面と方向の対応

ビューを記憶する際，ロボットから見た方向 (前面，後面，左右各側面) と絶対的な方角 (N,S,W,E) の対応を記憶する必要がある．なぜならば，記憶走行の際には各地点で一枚のビューを記憶するが，自律走行の際には進行方向に応じて，記憶したビューを水平方向にシフトさせて，位置推定のためのテンプレートとする必要があるからである．例えば，直線経路上で記憶走行時と逆方向に進む場合は，記憶したビューを 180[deg] シフトさせる．また右左折の際には，どの方向からどの方向に曲がるかに応じて，交差点で記憶したビューを 0,90,180,270[deg] の 4 通りにシフトする場合が考えられる．

Figure 6.17は交差点で記憶した一枚のビューを利用して，角を曲がる例である．記憶走行の進行方向は北向きなので，交差点でビューを記憶する際「ビューの前面は北である」と記憶しておく．自律走行の際，この交差点に東向きで進入し，南に右折する場合を考える．東向きに進入する際には，正面が東になるようにビューをシフトさせ，これをテンプレートとして一時停止する位置を決める．次に，正面が南になるようにビューをシフトさせ，このテンプレートと現在得られるビューが一致するまで回転を行なう．

進行方向の認識

ロボットはビュー間のリンクを張るために自分の進む方角 (N,S,W,E) を知っておく必要がある．これは最初の直線経路で前進を開始する際に人間がロボットに教えておくだけでよい．なぜならば，交差点で新たに通路を発見した場合，ロボットの左右どちらに通路が伸びているかでその通路が伸びている方向 (N,S,W,E) を判断できるためである．さらに，実際に初期値として与える方角は正しいものである必要もない．もし間違っていれば地図全体が回転するだけであるので，ロボットの自律移動に支障はないからである．

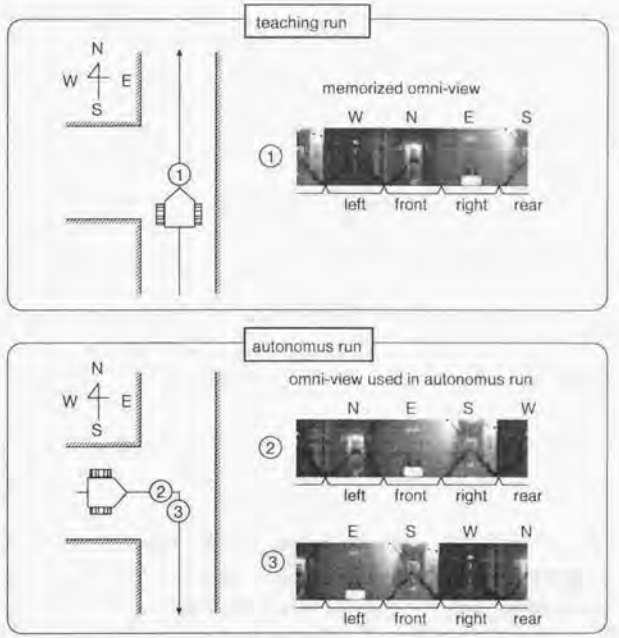


Figure 6.17 Rotation of memorized omni-view.

6.4.2 地図を用いた自律移動

地図を作成すれば，地図上の任意のスタートからゴールまでの自律移動が可能になる．ここでは，ロボットがどの位置にどのような姿勢でいようと，人間がゴールを指示すれば，そこまで自律移動させる手法を提案する．自律移動の処理過程を **Figure 6.18** に示す．ロボットはまず自分の現在位置を認識し，自分で作った建物内の地図を利用して，ゴールまで行く経路を考え，その経路に沿って実際に自律移動する．以下各処理について詳しく述べる．

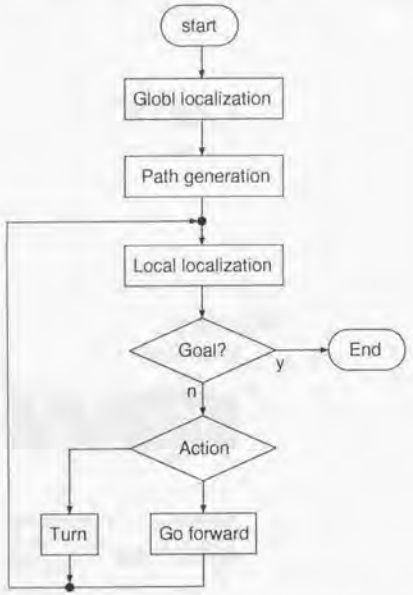


Figure 6.18 Flowchart of autonomous navigation using map.

初期位置の推定

移動を始める前には，ロボットは過去の履歴なしに自己の位置を知る必要がある．ここでは，現在の全方位ビューと，記憶したすべての位置でのビューの間で 360 度マッチングを行ない最も小さいエラーをとる位置を現在位置とする．単眼カメラのビューと異なり，全方位ビューでは周囲 360 度の画像を利用してマッチングを行なうため，ロボットの姿勢にかかわらず，信頼度の高い位置推定ができる．またロボット自身で進行方向に対する姿勢のずれを認識できるので，走行を開始する前に人間がロボットの姿勢を修正する必要はない．

経路計画

経路は，スタートからゴールに到達するためにたどるべきビューの番号を並べたものである．作成した地図はビューの番号をノード，隣接するビューをリンクでつないだグラフ構造となっている．スタート位置から，隣接するノードを探索していき，ゴール地点に到達できるパスを見つけてこれを経路とする．いま，スタート位置の番号を PATH_0 とし，たどるべきビューの番号を順に PATH_1，PATH_2，$\cdots$ とする．

経路中の自己位置の推定

ビューシーケンス内での自己位置の推定は前章で述べた方法と同様に行なう．すなわち，PATH_i および PATH_{i+1} での全方位ビューを参照画像，現在のビュー I を探索画像とした 360 度マッチングを行ない，より小さなマッチングエラーをとる方を経路中での現在位置とする．

動作の決定

人間が経路を教示する場合と異なり，ロボットが作成した地図には，その位置で行なうべき動作は記憶していない．しかし地図から経路を計画して，経路中の現在位置が分かれば，次の位置に進むように直進，回転あるいは停止の動作を決定できる．自己位置の推定の結果，現在位置が PATH_i と分かったとする．PATH_{i-1} から見た PATH_i の方向 NEXT_{i-1} と，PATH_i から見た PATH_{i+1} の方向 NEXT_i が一致していれば直進，異なっていれば NEXT_i の方向を向くまで回転を行なう．また PATH_i がゴールであれば停止する．直進，回転は前章で述べた方法と同様に行なう．直進は PATH_i でのビューのうち前方と後方の各ビューを参照画像，現在得られるビュー I を探索画像として 90 度マッチングを行ない，経路からのずれを検出して，ずれを小さくする方向にステアリング角を制御する．また回転は，PATH_i でのビューを適当にシフトさせて回転終了時のビュー V を生成し，I が V と一致するまで回転する．

6.4.3 地図獲得実験

地図作成

Figure 6.19のような廊下環境で，自律的に地図を獲得する実験を行なった．ロボットは A 地点で北を向けてスタートし，南北に伸びる廊下をステレオ視覚で前方の開空間を検出し前進しながらビューを記憶していった．その途中の C 地点では，左側に開空間があることを検出し，その位置を方向をスタックに記憶した．行き止まりの壁の手前約4[m] の位置Bでまで進んだところで，前方の開空間がなくなったと判断し停止した．次に，スタックから検出した交差点の位置を取りだし，その位置まで記憶したビューシーケンスを利用して戻ってきた．次に西向きに進行方向を変え，東西に伸びる廊下の探索を行なった．**Figure 6.20** は南北に伸びる廊下の探索を行なっている際の処理画像である．

また **Figure 6.22** はロボットが自律的に移動して作成した地図の形状およびそこに含まれるビューシーケンスである．南北に伸びる廊下の行き止まりまでのビューシーケンスが生成できていること，交差点のビュー (M_{12}) で2本の直線経路のビューシーケンスがリンクされていることが確認できる．各ビューはロボットが記憶した時の向きではなく，北向きが中心にくるように回転してから記憶されている．

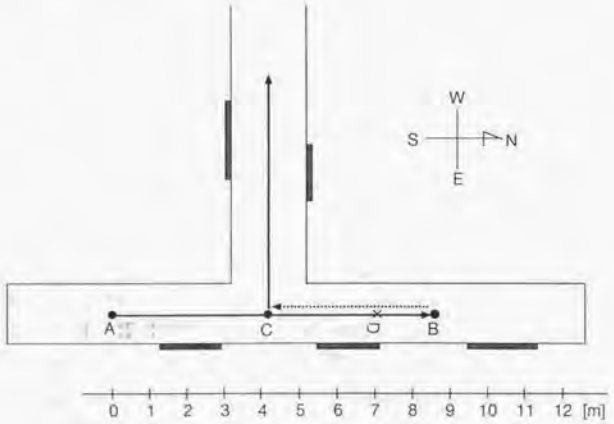


Figure 6.19 Environment for map acquisition.

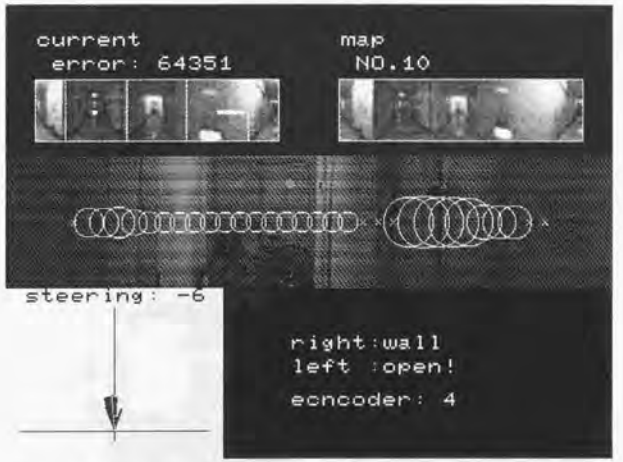


Figure 6.20 Display image when wondering in a straight corridor.

View	North	South	West	East	View	North	South	West	East
No. 0	1	--	--	--	No.23	24	22	--	--
No. 1	2	0	--	--	No.24	25	23	--	--
No. 2	3	1	--	--	No.25	--	24	--	--
No. 3	4	2	--	--	No.26	--	--	27	12
No. 4	5	3	--	--	No.27	--	--	28	26
No. 5	6	4	--	--	No.28	--	--	29	27
No. 6	7	5	--	--	No.29	--	--	30	28
No. 7	8	6	--	--	No.30	--	--	31	29
No. 8	9	7	--	--	No.31	--	--	32	30
No. 9	10	8	--	--	No.32	--	--	33	31
No.10	11	9	--	--	No.33	--	--	34	32
No.11	12	10	--	--	No.34	--	--	35	33
No.12	13	11	26	--	No.35	--	--	36	34
No.13	14	12	--	--	No.36	--	--	37	35
No.14	15	13	--	--	No.37	--	--	38	36
No.15	16	14	--	--	No.38	--	--	39	37
No.16	17	15	--	--	No.39	--	--	40	38
No.17	18	16	--	--	No.40	--	--	41	39
No.18	19	17	--	--	No.41	--	--	42	40
No.19	20	18	--	--	No.42	--	--	43	41
No.20	21	19	--	--	No.43	--	--	44	42
No.21	22	20	--	--	No.44	--	--	--	43
No.22	23	21	--	--		--	--	--	--

Figure 6.21 Map connection data.

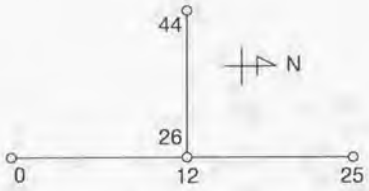


Figure 6.22 Acquired View-Sequenced Map.

場所の命名

ロボットに移動のゴールを指示する際に人間が地名（シンボル）を与えられるように，必要に応じて地名を決められるようにした．その時のインターフェースを **Figure 6.23**に示す．画面上には，全方位ビューから前後左右各面 90 度分の画像が並べられており，前進，向きの変更などをボタン操作で指示し，建物内を実際に動くような感覚でロボットが作成した地図を確認する．そして後でゴールとして用いる可能性がある場所では，キーボードから地名を入力する．また，同じインターフェースを用いて，人間がロボットにゴール地点を指示することができる．

Figure 6.23 Interface for naming a place.

地図を利用した自律移動

ロボットにゴールを指示すれば，任意の初期位置からゴールまで自律移動することを確認する実験を行なった．ここでは，初期位置を無作為に **Figure 6.19**の D 地点に置いて，M_{30} まで移動するように指示した．**Figure 6.24**は初期位置で得られるビューと，記憶した全てのビューと 360 度マッチングを行なって位置の推定をした結果である．M_{21} とのエラーが際だって小さくなっており，信頼度の高い位置推定ができていることが確認できる．

360 度マッチングの効果を調べるために，マッチした M_{21} と隣接するビュー M_{20}，M_{22} の 3 枚のビューを比較する (**Figure 6.25**)．側面の扉や廊下の行き止まりの形など，画像の一部の特徴を 3 枚のビューで比較してもそれほど大差はない．しかし 360 度全体で比較すると，側面の扉が見える方向などが 3 枚のビューで確実に異なっており，360 度マッチングで M_{21} にマッチしたことが妥当であることが確認できる．

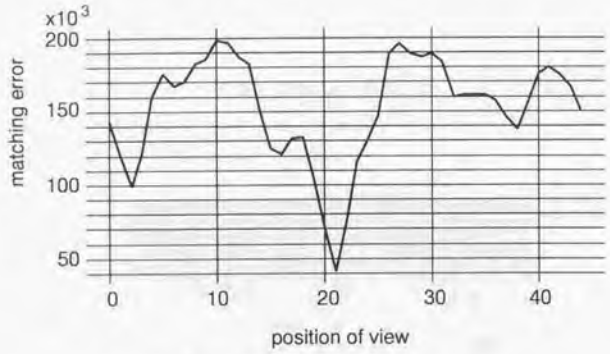


Figure 6.24 Result of global localization.

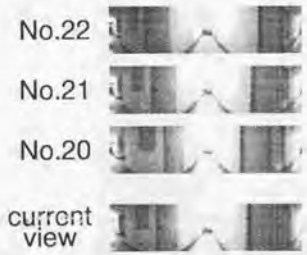


Figure 6.25 Matched view (21) and neighboring views.

次に，作成した地図を利用して，初期位置 M_{21} から人間が指示したゴール M_{30} までの経路をロボットに生成させた．結果は **Figure 6.26**のようになり，経路の生成が正しく行なわれていることを確認できる．各ビューは，進行方向が中央に来るようにそれぞれ回転してあり，また各ビューに対応する行動が付加されている．このビューシーケンスを用いて，ロボットは計画した経路に従ってまず南に前進し，交差点で西に右折し，M_{30} まで自律移動できることを確認した．**Figure 6.27**は自律移動している際の処理画像である．

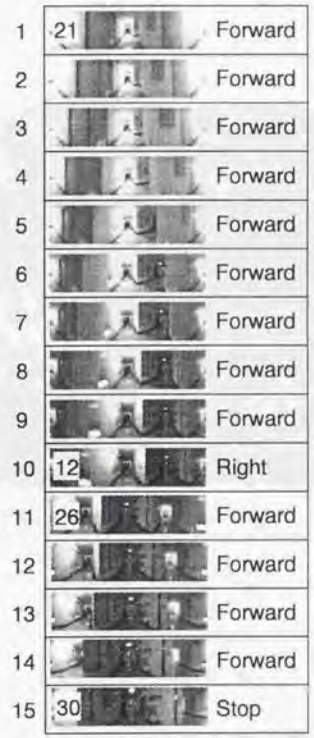


Figure 6.26 Generated View Sequence as a navigation path.

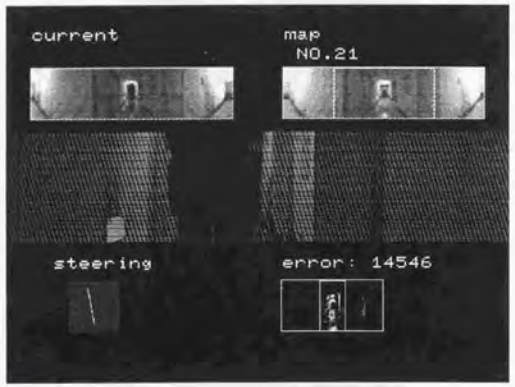


Figure 6.27 Display image in autonomous navigation.

第 7 章

仮想環境を用いたビュー・シミュレーション

7.1 ビューのシミュレーション

ロボットの行動生成のアルゴリズムを研究する場合には，様々な条件を任意に設定した上で実験を繰り返し，その性能を定量的に評価することが重要である．しかし，実環境では多くの条件を任意に設定したり，繰り返し実験することは非常に手間がかかるため，通常はアルゴリズムの評価にはシミュレータを用いた仮想的な実験が利用される．

これまでのシミュレータでは，環境を単純化し周囲の状況が完全に分かっていると仮定した上で行動生成のシミュレーションを行なっているものが多く，「センサの情報に基づく環境認識」を含めた行動生成アルゴリズム全体のシミュレーションを行なえるようなシステムは少ない．しかし，実世界で行動するロボットにおいては，環境認識の結果(どのような種類の情報がどのくらいの精度で得られるか)は行動生成に大きく依存する．そのため，環境認識アルゴリズムと行動生成アルゴリズムの設計や評価は，分離してそれぞれ別個に行なうことは，望ましくない．例えば視覚を用いて行動するロボットにおいては，視覚処理の過程を含めた行動生成のシミュレーションが不可欠であると考えられるが，そのようなシミュレータはこれまでにはなかった．

近年人工知能のテストベットとして注目されている，ロボットによるサッカー競技会RoboCup のシミュレーション部門[70]では，モデルから認識結果を得る過程において視野やノイズを設定することで得られるセンサ情報を制限し，現実のロボットに条件を近づける工夫をしている．しかし視覚処理をシミュレーションに含めているわけではないため，実機部門のロボットのプログラムをそのままシミュレータ上のロボットで用いることはできず，視覚処理まで含めたロボットのプログラムの検証を行えるシステムにはなっていない．

本章では，このような問題点を解決するため，コンピュータグラフィックス(以下CG)による視覚情報と画像処理システムを組み合わせた視覚移動ロボット用のシミュレータ View Simulation System(以下VSS)[71]を提案する．このシステムを用いると，環境認識処理(特に視覚処理)まで含めた移動ロボットのシミュレーションを行なうことができる．**Figure 7.1**はそのコンセプトを示す．図上は，実ロボットの処理ループである．図中はこれまでのシミュレーションの処理ループで，センサ情報から環境認識を行なう部分が省略され，仮想世界から環境情報を直接得て行動生成を行なっている．図下は我々が提案するVSSを用いたシミュレーションのループで，実ロボットと同じように環境認識と行動生成の処理が行なわれる．

CG を画像処理の対象とすることは，コンピュータビジョンの分野では古くから行なわれている[6]．これは，既知の3D モデルのCG を生成し，ステレオ視やオプティカルフローなど画像処理アルゴリズムの評価に用いるというものである．これらはあらかじめオフラインで生成された画像を用いている．

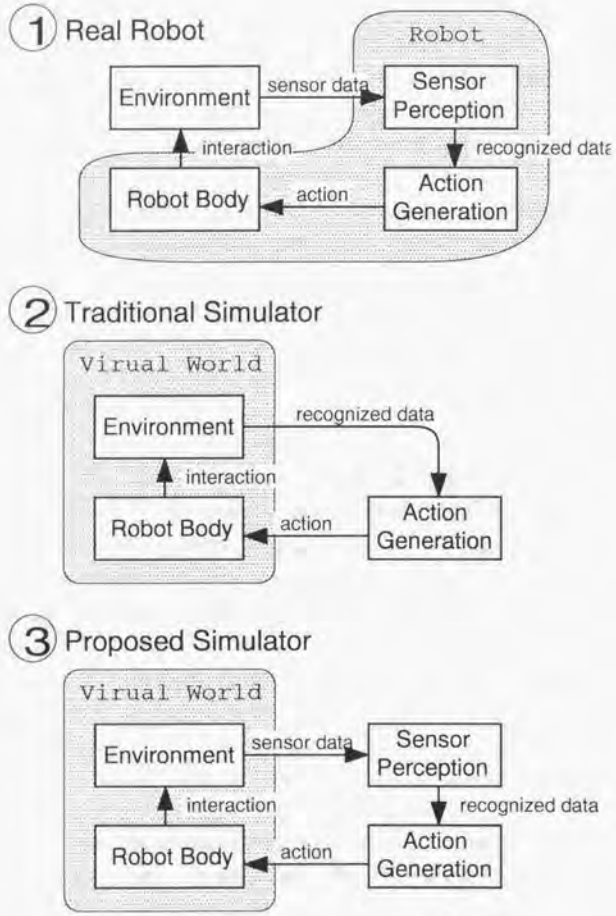


Figure 7.1 Real robot system (left) and VSS (right)

また，Demetri らはインタラクティブな3次元CGを画像処理に用いて，仮想世界の魚(Artificial Life) の視覚による行動アルゴリズムの生成を行なっている [72]．この研究ではCG を処理し，その結果から行動をシミュレートして，仮想世界を更新するというビジュアルフィードバックを実現している点が，コンピュータビジョンの分野での CG の利用法より一歩進んでいる．しかし，仮想世界内でのシミュレーションにとどまっており，実時間性や実世界での行動への応用に関しては特に考慮されていない．

ロボットのシミュレーションにおいて，CG は単に可視化のためにだけ利用されることが多い．例えば，移動ロボットのための仮想環境システムである VIRTUOUS[73] においては，CG は単にユーザに状態を見せるための手段に過ぎない．

ロボット用シミュレータで，CG を画像処理の対象とするものは，これまでにはなかった．これは，実環境のモデリングや，CG の高速な生成が困難であるためであると考えられる．しかし，近年のコンピュータの進歩，特にバーチャルリアリティのための画像表示ハードウェアの進歩により，画像処理での利用に耐え得るリアリティの高い画像のリアルタイムでの表示が可能となってきた．本稿では，CG 生成用のコンピュータを用いることで実現した，視覚処理を含めた移動ロボットのシミュレータの開発，およびそれを用いた移動ロボットのナビゲーション手法の評価について述べる．

7.2 ビュー・シミュレーション・システム

7.2.1 ハードウェア構成

Figure 7.2.Figure 7.3に，VSS のハードウェア構成を示す．実験システムは実際の視覚移動ロボットと全く同じユーザプログラムが動作するように構成されており，CG 生成部，画像処理部，インターフェース部の 3 つの部分からなる．CG 生成部は環境とロボットのモデルを持ちシミュレーションを行い，ロボットから見た視野画像を生成する．画像処理部は視覚処理と判断を行なう頭脳にあたり，実ロボットと同じ画像処理装置を用いる．インターフェース部は CG 生成部と画像処理部を仲介し，画像処理部のユーザプログラムが実ロボットと同じように動作できるような環境を提供する．これにより VSS 上と実ロボットとでソースレベルで同じプログラムが動作可能となる．

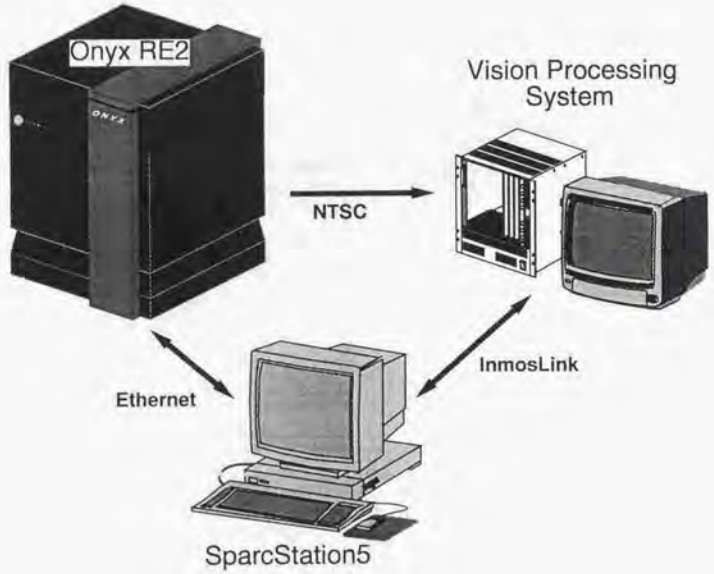


Figure 7.2 Overview of VSS

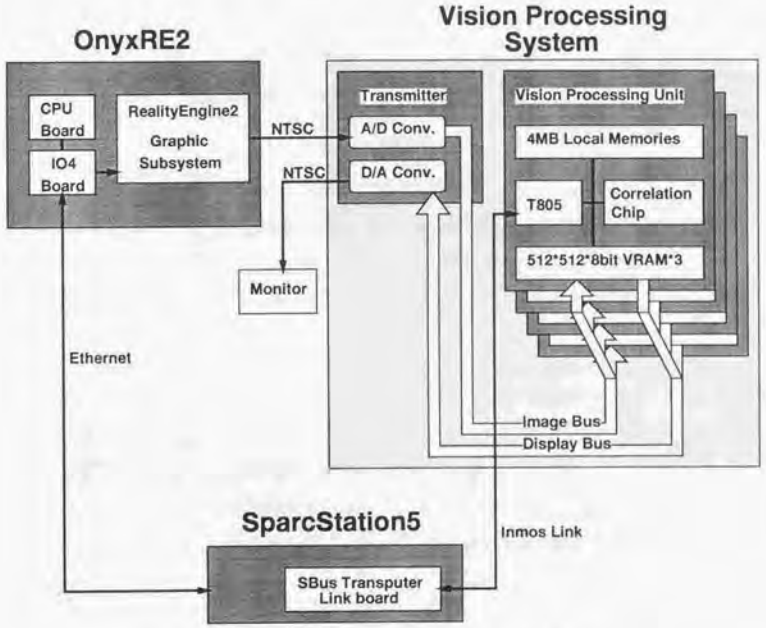


Figure 7.3 Hardware configuration of VSS

CG 生成部

CG の生成は計算機にとって非常に負荷の高い処理を必要とする．そのため，本システムでは，CG の生成にはグラフィックスエンジンと呼ばれる専用のハードウェアを持ったグラフィックワークステーション，SiliconGraphics 社の OnyxRealityEngine2 (以後 OnyxRE2) を使用している．OnyxRE2 には RGB 出力の他にビデオ出力が標準装備されており，これを画像処理部への入力として利用している．

Table 7.1 Specification of OnyxRE2

CPU	R4400 (150MHz) 2 個
最大処理能力	600MFLOPS
メインメモリ	256Mbyte
フレームバッファ	40Mbyte
アンチエイリアス (AA) 処理	200 万 vector/ 秒
AA・テクスチャ三角形メッシュ	930K/ 秒
AA・テクスチャ処理 描画レート	8000 万 pixel/ 秒
その他	NTSC/PAL/S-Video 出力

画像処理部

画像処理部には画像処理ボード jsk-ifm/mt-01 [45] を用いる．この画像処理ボードは CPU としてトランスピュータ T805，局所相関演算のための専用 LSI が搭載されたものである．このボードを複数枚用いることで，同一の入力画像に対する画像処理を並列に行うことができる．これは実ロボットに搭載されているものと同一のシステムである．

インターフェース部

インターフェース部には jsk-ifm/mt-01 のホストマシンとして，トランスピュータリンクボードを持つ SparcStation5(以下 SS5) を使用している．SS5 はイーサネットによって OnyxRE2 とつながれており，画像処理部と画像生成部の通信を行なう際にその仲介を行なう．また，画像処理部のプログラム (つまりロボット上で実行されるアプリケーションプログラム) のユーザインターフェース，キーボードやマウスのエミュレーションも行う．

7.2.2 ソフトウェア構成

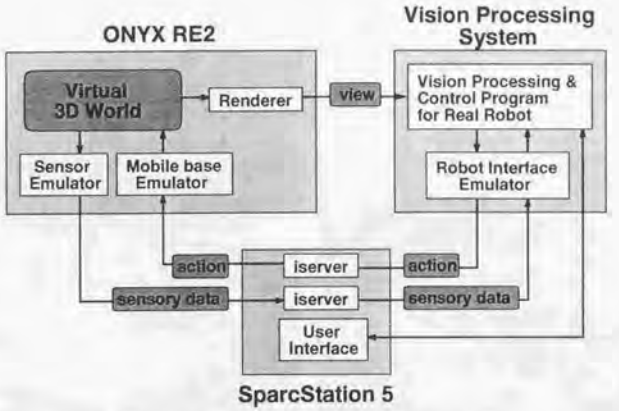


Figure 7.4 Software configuration of VSS

実験システムのソフトウェア構成を **Figure 7.4**に示す．ソフトウェア構成はハードウェア構成に対応した 3 つの大きな部分に分れる．

CG 生成部は，事前に与えられた環境およびロボットのモデルから 3 次元仮想環境を生成し，CG を生成して画像処理部に送る．画像処理部では，画像処理を行って，ロボットの動作コマンドを生成する．この動作コマンドは，インターフェース部を介して CG 部に送られ，ロボットの動作モデルを用いて移動量に変換される．そして仮想環境内のロボットの位置・姿勢が更新されたのち，画像に反映される．このようにして，CG 部と画像処理部をループが形成され，ビジュアルフィードバックが実現される．

CG 生成部

CG 生成部は環境とロボットの 3 次元モデルを持ち，画像を生成する．また，ロボットの動作命令をインターフェース部から受け取り，その 3 次元モデルを更新していくことで，実世界の実験環境をシミュレートする．移動ロボットにおける画像以外のセンサ，例えばシャフトエンコーダの情報も，仮想環境生成部のモデルから得ることができ，必要であればインターフェース部を介して画像処理部に送られる．

CG 部において，3D の CG を作成するプログラムを書くための API としては OpenInventor を使用する．OpenInventor は OpenGL 上に構築されており，現在多くの OS に移植されている．また，OpenInventor のデータファイルは，VRML とほぼ同等のデータ形式

であるため，市販の VRML 用モデリングツールにより作成した様々な仮想環境を実験に利用することができる．

Figure 7.5に作成した仮想環境の例を示す．左は VRML 用モデリングツールを用いて作成した屋内環境，右は簡単なモデルにテクスチャを貼ることで作成した屋外環境である．

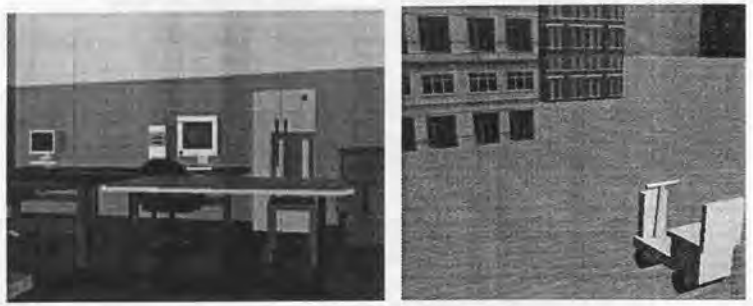


Figure 7.5 Examples of 3D world model

画像処理部

画像処理部ではユーザプログラムが動作する．ユーザプログラムは実ロボットとソースコードのレベルで同一のもので，この中には画像処理プログラムとロボットの動作を決定するプログラムが含まれる．ユーザプログラムにおいて生成されたロボットの動作命令は，実ロボットにおいてはトランスピュータ上のコントローラプロセスが受け取り，直ちに実行されるが，VSS においてはコントローラエミュレータが受け取った後，インターフェース部を介して仮想環境部へ転送される．

インターフェース部

インターフェース部は，画像処理部と仮想環境生成部の間の双方向の情報転送を仲介する他，実ロボット上のキーボードやマウスなどのユーザインターフェースをエミュレートする．

7.3 仮想世界のモデリング

7.3.1 環境モデル

実環境で動作するロボットの評価用に用いる仮想環境は実環境の複雑さを表現する必要がある．実環境の 3D 幾何モデルを自動的に詳細に獲得し CG に利用するという研究 [74] も行なわれているが，本研究では 3D 幾何モデルは単純なものにし，実画像テクスチャを組み合わせてリアリティを持たせた環境モデルを用いる．このモデルは仮想環境で実環境の複雑さを適度に表現でき，かつ描画に要する時間が短くてすむ．また，VSS では CG 表示における制限およびモデル作成の簡単化のために，

- 光源は環境光のみ存在
- 光の反射や影はない

としている．環境モデルの作成は，実ロボットの実験環境に応じて行う必要がある．次章では，その一例として廊下のモデルの作成について述べる．

7.3.2 ロボットの運動モデル

実験対象とする移動ロボットによってごとに運動モデルを用意しており，現在用意されているロボットモデルは 屋内実験用ロボット (2 輪速度差方式) および屋外実験用ロボット (全輪ステアリング方式) の 2 種類である．VSS ではロボットの運動モデルを簡単化するために，

- 車輪と床との間の摩擦・すべりはない
- 床の凸凹はない
- 制御の時間遅れは (VSS 内の通信時間を除き) ない
- ロボット全体および車輪の慣性はない
- 車輪の速度と指令値との誤差はない

と仮定している．

7.4 廊下環境のモデリング

7.4.1 幾何モデル

ここでは，実際に実ロボットが稼働している環境である廊下の仮想環境を作成した例について述べる [75]．仮想環境の構成方針として，簡単な 3 次元幾何モデルと実画像のテクスチャ

を用いることとする. 廊下の幾何モデルとしてはFigure 7.6に示す, 左右の壁, 天井, 床, 正面の5面構成される直方体を用い, 5つの面にそれぞれテクスチャを貼り付ける. Figure 7.6のwidth, heightは実測値を用いるが, depthは実際の廊下の長さよりも短くする. つまり, 廊下のある部分から先は, frontの画像一枚で代用することで, テクスチャのサイズを減らし, 表示の高速化を図る.

この幾何モデルは, 廊下モデルとしては非常に単純であり, 幾何モデルの細かい部分は省き, 実環境の複雑さはテクスチャによって補っている. 仮想現実感の研究でも3D幾何モデルの精度を上げることよりもむしろテクスチャとしての実画像を積極的に用いてリアリティを上げていこうという傾向がある[76]. これは, 複雑な3D幾何モデルを用いるよりも少ない手間でリアリティの高いものが生成可能であると考えられるためである.

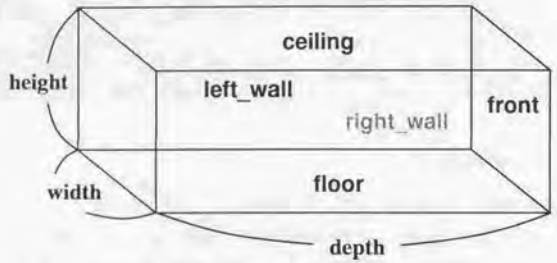


Figure 7.6 Geometric model of corridor

7.4.2 テクスチャの取り込み

テクスチャの取り込みには, パノラミックビュー[14]の画像取り込み手法を利用する. パノラミックビューとは画像中のあるラインに注目し, カメラを動かした時にその部分から得られる連続的な画像データをつなぎ合わせた画像のことである. ここでは, Figure 7.7上に示すように画像中の4スリットをスキャンのためのラインとし, カメラを前方に移動しながら床, 天井, 壁のテクスチャを同時にスキャンしていく.

ただし, この方法はカメラの正確な直線移動を必要とする. 実際のテクスチャの取り込みには移動ロボットに搭載したカメラを用いているが, ロボットは正確な直線移動は行えない. ここでは, ロボットの移動誤差による画像の見え方のずれを, 画像処理(トラッキング)を用いて補正することにした. まずテクスチャの取り込みを始める前に, 廊下の正面中央に見えるパターンとその位置を記憶する. そして, そのパターンをテンプレートマッチングを用いて追

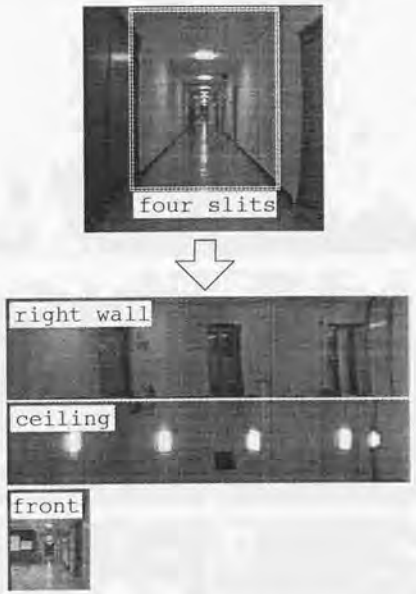


Figure 7.7 Texture generation

跡しつづけることで，ロボットの方向の正面からのずれを検出する．そのずれに合わせてテクスチャを取り込むためのスリットの位置を上下左右に移動させる．このような方法により，ロボットを廊下で一度走行させるだけで，容易にテクスチャを取り込むことができる．

取り込んだテクスチャの例(右壁，天井，正面)を Figure 7.7下に示す．テクスチャは解像度が一定であり，また，4ラインの画像を同時に取るため，奥行き方向に関して4面のズレがない．しかし，スリットは一定距離前方を見ているため，凹凸がある場合には正確なテクスチャを取ることができない．また，実際のカメラがピンホールモデルとは一致しないために画像が端にいくに従って歪んでおり，その影響でテクスチャにも歪みが見られる．

このテクスチャを用いて作成した廊下モデルの外観と，モデル内のロボットの視点から見た画像を Figure 7.8に示す．外観では，幾何モデルの単純化やテクスチャの歪みの影響が目立つが，モデル内のロボットから見た限りは特に問題にはならない程度のものであり，全体としてはかなりリアルなCG画像を生成できている．

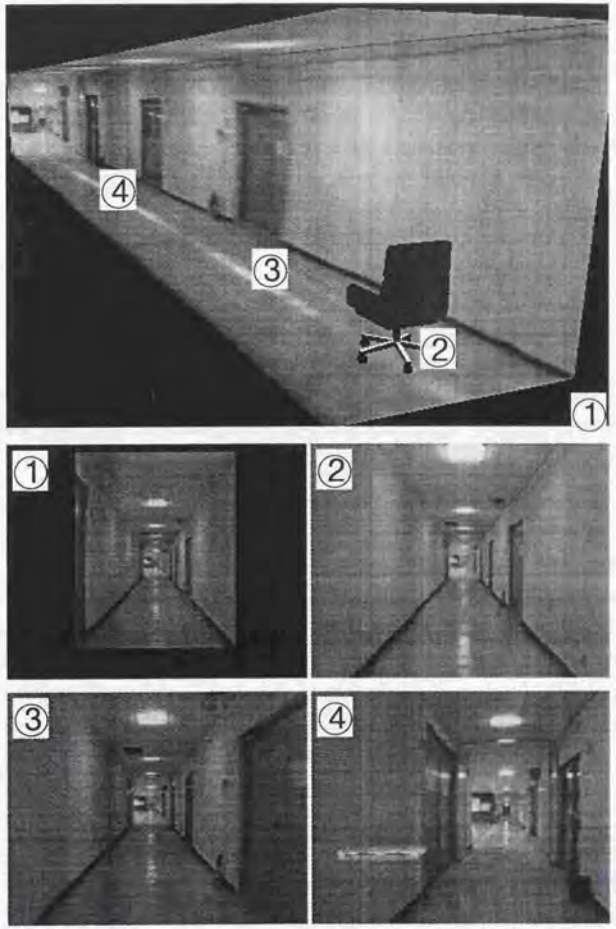


Figure 7.8 CG of reconstructed corridor model

Table 7.2 Four combinations of teaching and autonomous run using VSS

	教示走行	自律走行	実験内容
a	実ロボット	実ロボット	従来の走行実験
b	実ロボット	VSS	教示画像の分析
c	VSS	実ロボット	仮想教示による走行
d	VSS	VSS	仮想教示の結果予測

7.5 走行シミュレーション

本節では，View Simulation System を，ビューシーケンスによる経路誘導手に応用することで，本システムの有効性を確認する．ここで，教示走行と自律走行のそれぞれで VSS または実ロボットを用いることができるため，Table.7.2に示す4つの組み合わせが考えられる．

VSS で生成する画像列は，完全にまっすぐな走行を行った場合の画像列や，既知のずれを与えた場合の画像列など，実際の記録走行では不可能な任意の経路に沿った画像列を作成することができる [75]．実験には，前章で作成した廊下モデルを用いる．一度作成した仮想環境はある瞬間に固定したデータであるため，繰り返し同じものを用いることができる．これにより対照実験が可能であり，実験条件の変更による結果の変動のうち，この世界固有のデータによるものとそうでないものとを分離することが可能となる．

実画像のテクスチャを用いた廊下のモデルを利用すると，作成した CG は非常にリアルなものになる．さらに，画像列を用いた走行では実際のマッチングに用いる画像は解像度を下げた画像であるため，3D 幾何モデルやテクスチャの細部の誤差は問題にならない．ただし，CG と実画像のマッチングを取る場合には，明るさを実環境で得られる画像と一致させなければならないが，これはモデル内の照明や物体の反射率などのパラメータが複雑に関係するために難しい．そこで，この問題にはマッチングの際に画像全体の明るさを正規化することで対処する．

7.5.1 システムの評価実験

まずはじめに，画像列に基づく走行手法において，VSS を用いた走行が，実環境での走行をどの程度再現できるかを評価するための実験を行なった．はじめに実ロボットで教示 (意図的に直線からずらした) を行ない，得られた画像列を用いて実ロボットと VSS で自律走行実験を行って比較する．これは Table.7.2の a,b にあたる．この時の走行軌跡を **Figure 7.9** に示す．実ロボットの走行 (実線) は，教示が正確な直線経路でなかったため，細かくふらつきながら徐々に壁側に寄っている．VSS で同じ画像列を用いて走行を再現すると (破線)，ふらつきは無いものの徐々に壁側に寄る傾向を示している．

この実験により，VSS では実ロボットの走行時の細かなふらつきまでは再現できないが，軌跡のおおまかな傾向は再現できることが示された．VSS で再現できなかった誤差は，7.3節で導入したモデリングにおける仮定によって起きていると考えられる．再現性をより高めることは，より正確にモデリングを行なうための手間とトレードオフの関係である．本研究では，これ以上正確にふらつきを再現することに意味があるとは考えず，本実験で得られた再現性を，画像列に基づく走行手法のシミュレーションとして十分有用であると評価する．

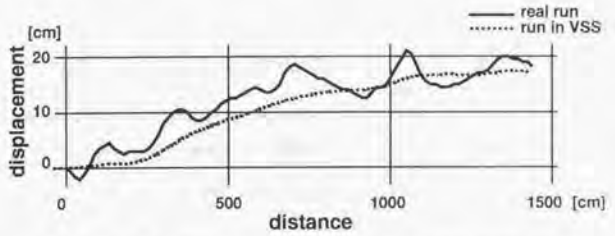


Figure 7.9 Evaluation of VSS using real view sequence

7.5.2 走行経路の予測

前節での走行実験では，実ロボットおよび VSS の両方で経路が壁に寄っていくという傾向が得られた．これは，教示走行における画像列の記憶が正確に行なわれていなかったためであり，ここでは画像列の誤差が走行経路に与える影響を VSS を用いて調べる．画像列の誤差の原因としては次の様なものが考えられる．

- 教示の際，カメラの位置が経路から横にずれた
- 教示の際，カメラは姿勢が進行方向正面からずれた

ここでは，VSS を用いて意図的にカメラの姿勢だけに既知の誤差を与えた画像列を生成し，走行軌跡にどのような影響を及ぼすかを VSS での走行に用いて予想した．またこの予想を確認するために，同じ画像列を用いて実ロボットでも走行を行なった．この実験は Table.7.2の c,d にあたる．

画像列としては，スタートから 250[cm] から 400[cm] の間の画像は 3[deg] 右を向いて見える画像，900[cm] から 1100[cm] の間は 3[deg] 左を向いて見える画像，その他の区間に関して

は正面を向いた画像を与えた．この画像列を用いて VSS で 10 回の仮想走行を行なった結果の走行軌跡の平均を Figure 7.10の破線で示す．画像に与えた姿勢の誤差はわずかなものであるにもかかわらず，走行軌跡に大きく影響が出ることがこの実験によって分かった．

また，Figure 7.10の実線は同じ画像列を用いた実ロボットの走行軌跡であり，VSS による仮想走行の軌跡予想が，実ロボットでの走行軌跡と誤差数 cm 程度で一致していることがわかる．

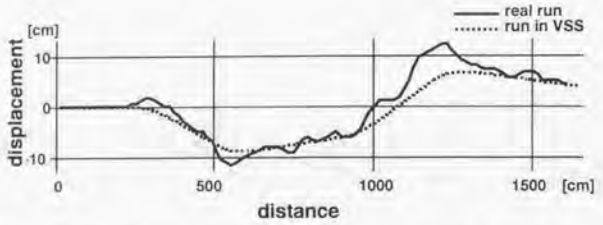


Figure 7.10 Predicted and real trajectory using virturl view sequence

7.5.3 実ロボットでの走行精度の改善

ここでは，VSS を用いて仮想走行を行ない，正確にまっすぐに進んだ場合の画像列を生成する．この画像列を用いて実ロボットの走行を行なえば，通常の教示走行で作成した画像列を用いるよりもよりまっすぐに進むことが期待できる．これは仮想環境を用いたロボットのオフライン教示と見ることができる．

Figure 7.11の実線が実ボットでの教示走行により作成した画像列を用いて，実ロボットで走行した軌跡で，破線が VSS を用いて仮想走行を行なって作成した画像列を用いて実ロボットで走行した軌跡である．VSS を用いて教示した場合，走行軌跡の直線性が大きく改善されているのがわかる．このように，VSS を用いることで，実ロボットを用いるよりも正確な教示画像列を作成することが可能であることが分かった．

Figure 7.12は，この 2 つの走行の際のマッチングエラー値を示す．値が小さいほど教示時と自律時の画像が似ていることになる．全般的に，仮想教示を行った場合の値が大きい．この原因としては，実環境と仮想環境の細かなモデルの差以外にも，

1. 前半では，スタート地点で生成できる CG がスリットの位置の関係で画面全体にはならない (Figure 7.8参照)

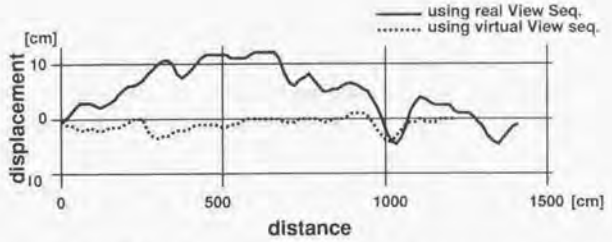


Figure 7.11 Improvement of linearity in autonomous run

2. 後半では，廊下モデルでの正面の一枚の画像が視野中に占める割合が大きくなってきた

といったものが考えられる．ただし，エラー値が多少高くなってもマッチングの結果として得られる 1) 画像列中の自己位置の認識，2) 画像とのずれの検出，に失敗することはなかった．

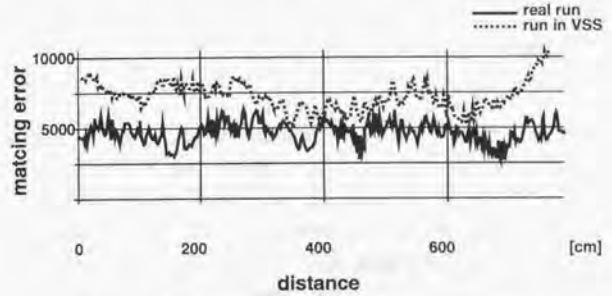


Figure 7.12 Matching error in autonomous run

7.5.4 考察

本シミュレータは，実ロボットの慣性や，滑べり，床面の凸凹等は考慮していないため，走行軌跡は実世界のものに比べて滑らかになりがちであるが，全体の傾向はよく一致していた．しかし，これは「ロボットが行っている画像処理(全体を粗く見る)と作成したモデルの相性がよかった」ためであるとも言える．全ての視覚移動ロボットに適した仮想環境を作ることは困難である．そのため，シミュレーションの対象となるロボットが利用する画像特徴をよく再現できるよう，場合に応じてモデリング手法を決める必要があるが，これは今後の課題である．

VSS での走行では，画像列以外の要素はほぼ理想的な状態である．このため，実ロボットでの教示によって作成した画像列を用いて VSS で走行を行なえば，作成した画像列の影響のみを調べることができる．実際には VSS による実験でも通信の時間遅れが存在するため，VSS による走行も多少のブレがあるが，繰り返しシミュレーションを行なうことで，この影響を含めた評価が可能である．

7.6 オンライン・ビュー・シミュレーション・システム

7.6.1 オンラインのシミュレーション

視覚処理を含めたロボットのシミュレーションを行うためには仮想環境の画像を生成する必要があり，このために CG を利用する．ここでは，これまでのロボットビジョンの研究における CG の利用法について考察する．ロボットビジョンと CG の間には利用目的によって以下の 3 つの関係が考えられる．

1. CG による視覚処理結果の可視化
   数値データの可視化は CG の元来の利用法である．ロボットビジョンにおいても，環境認識などの視覚処理の結果を CG により可視化することは多い．ここで，
2. CG を用いた視覚処理システムの評価
   CG によってあらかじめ生成した定量的な仮想画像を用いて視覚処理アルゴリズムの定量的評価を行う．コンピュータビジョンの研究分野では古くから行われていた利用法．
3. CG を用いたロボットシステムの評価
   仮想環境の中でロボットに視覚に基づく行動をさせることで視覚処理を含むロボットの行動生成アルゴリズムの評価を行なう．

3 の利用法は，CG の生成と視覚処理が閉ループを形成し，互いにフィードバックをかけているところが 1,2 と異なる．我々が開発した View Simulation System [71](以下 VSS) は，この利用法を実現するためのシミュレーションシステムである．VSS は実ロボット上のプログラムをそのまま仮想世界で実行することで仮想的な教示走行や自律走行実験を行なうことができるという特徴がある．しかし，このループは仮想環境内で閉じているため，実環境とは切り離されていて，実環境への働きかけを想定していない．

Figure 7.14上に，もう一度 3 の View Simulation System のループを，実環境でのロボットの行動のループと比較して示す．この 2 つのループをつなげると，実環境と仮想環境がそれぞれロボットを通るループによって結合されたロボットシステムになる．これが我々が本論文で提案する新たなロボットシステム Online View Simulation System (以下オンライン VSS) のコンセプトである [77]．

オンライン VSS を用いると，仮想的な走行と実際の走行を同時に行なうことができる点が，これまのロボットシステムでと大きく異なる．これら 2 つのループを同時に実行することは，ロボットが「自分が動いたら，環境の見え方がどう変化するか」を常に予想(シミュレーション)しながら行動を行なうことに相当する．

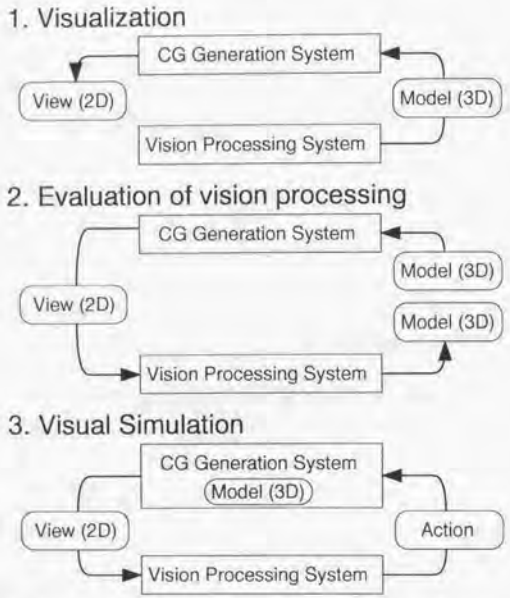


Figure 7.13 Robot vision with CG generation system

このような考え方はエッジベースのモデルマッチングを利用する研究では行なわれていた([24] など)が，ビューベースのマッチングを行なう研究ではこれまでにはなかった．

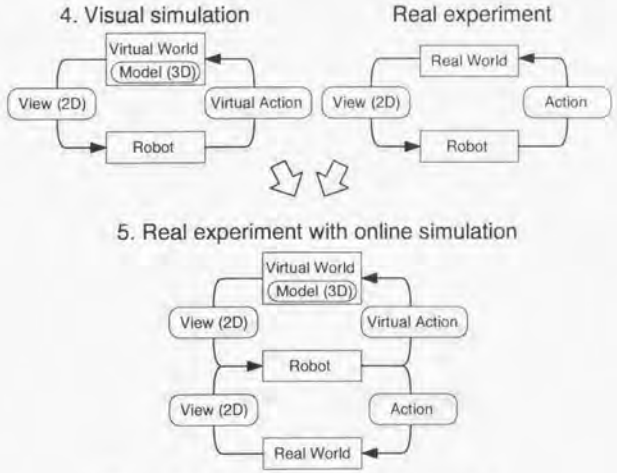


Figure 7.14 Real experiment with simulation

7.6.2 ハードウェア構成

オンライン VSS を用いた実験システムの構成は **Figure 7.15** に示すようにロボット本体に搭載された画像処理部と CG 生成部に分かれる．本来ならば，CG 生成部も移動ロボット本体に画像処理部とともに搭載するのが望ましい．しかし現在は CG 生成部としてグラフィクスワークステーションを利用しているために，サイズや消費電力の点でロボットへの搭載は不可能である．そこで CG 生成部は研究室内に据え置き，建物内を移動するロボットは，無線 LAN を介して CG 生成部とネットワーク接続することにした．

CG 生成部

CG の生成は計算機にとって非常に負荷の高い処理を必要とするため，本システムでは，CG の生成にはグラフィックスエンジンと呼ばれる専用のハードウェアを持ったグラフィックワークステーション，SiliconGraphics 社の OnyxRealityEngine2 (以後 OnyxRE2) を使用し

ている．VSS では OnyxRE2 のビデオ出力を画像処理部への入力として利用していたが，オンライン VSS では画像をネットワーク経由で送る．ビデオ信号を無線 (UHF) の送受信器を用いて送ることも考えられるが，1) 画像にノイズが乗りやすい，2) ロボットの行動範囲が無線ビデオ信号の届く範囲に限定されてしまう，という理由で採用していない．

画像処理部

画像処理部には画像処理ボード jsk-ifm/mt-01 [45] を用いる．この画像処理ボードは CPU としてトランスピュータ T805，局所相関演算のための専用 LSI が搭載されたものである．このボードを複数枚用いることで，同一の入力画像に対する画像処理を並列に行うことができる．

インターフェース部

画像部の CPU(Transputer T805) は，PCMCIA の デジタル IO カード (RATOC 社 REX5055) を用いて製作したトランスピュータリンク (10[Mbps]) を介してホストのノート PC と通信する．ノート PC は無線 LAN(2[Mbps]) と通常の LAN(10[Mbps]) を介して CG 生成部とつながっており，ソケット通信を用いてデータのやりとりを行なう．

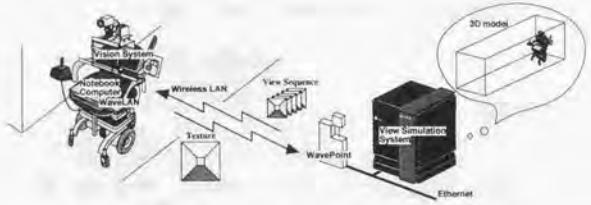


Figure 7.15 System configuration of online VSS

7.6.3 ソフトウェア構成

具体的には，画像処理装置上の CPU(Transputer T805) は，PCMCIA DIO (REX5055) を用いたインターフェース (10[Mbps]) を通して ホストのノート型 PC 上のサーバプログラムと通信する．このサーバは無線 LAN(2[Mbps]) と通常の LAN を介して VSS とつながっており，ソケット通信を用いてデータのやりとりを行なう．オンライン VSS を用いた実験では，

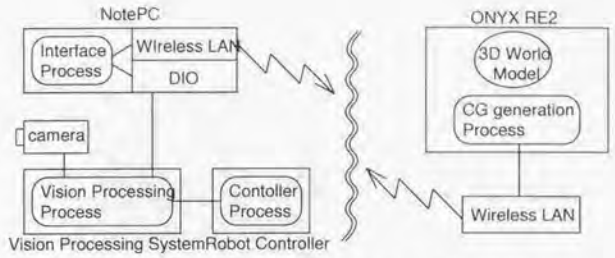


Figure 7.16 System configuration of online VSS

走行時にロボットから VSS 側へは廊下のテクスチャや画像列生成コマンドが転送され，VSS からロボット側へは生成した画像列が送り返されることになる．

7.7 オンライン経路生成

7.7.1 走行手順

オンライン VSS を用いた今回の実験システムでは，走行中に画像の転送が出来る．そこで，

1. ロボットから VSS に 1 枚の画像を転送する．
2. VSS で逆透視変換により廊下モデルのテクスチャを生成し，モデルを更新する．
3. VSS で 5m 程度の仮想教示走行を行ない，画像列を生成する．
4. VSS からロボットに画像列を転送する．
5. ロボットが画像列に従って自律走行を行ない，停止する．
6. さらに走行を続けるなら，1. に戻る．
7. 終了．

という手順により，オンラインのモデル更新，経路生成および自律走行を行なう．

7.7.2 幾何モデル

ここでは，実際にロボットの実験環境である廊下の仮想環境のモデリング方法について述べる．廊下のモデルは簡単な 3 次元幾何モデルと実画像のテクスチャを用いることとする．廊下の幾何モデルは **Figure 7.6** と同一である．

7.7.3 テクスチャの生成

それぞれの面のテクスチャは，廊下のある地点でロボットが得られる 1 枚の画像を逆透視変換することで生成する．逆透視変換の概要を **Figure 7.17** に示す．なお，この変換を行うにあたりロボットの位置・姿勢は既知としている．あるテクスチャ上の 1 画素 p の色を求める場合，まず p とカメラの焦点 o を結んだ直線を引く．この直線が画像と交差する交点 q の色が p の色となる．

全てのテクスチャ上の点についてこの計算を順次行うことでテクスチャ全体を生成する．**Figure 7.18** に生成された壁のテクスチャの例を示す．図右が手前，左が奥である．この例からも分かる通り，逆透視変換を用いる方法ではテクスチャの解像度は一定ではなく，奥に行くにしたがって粗くなる．

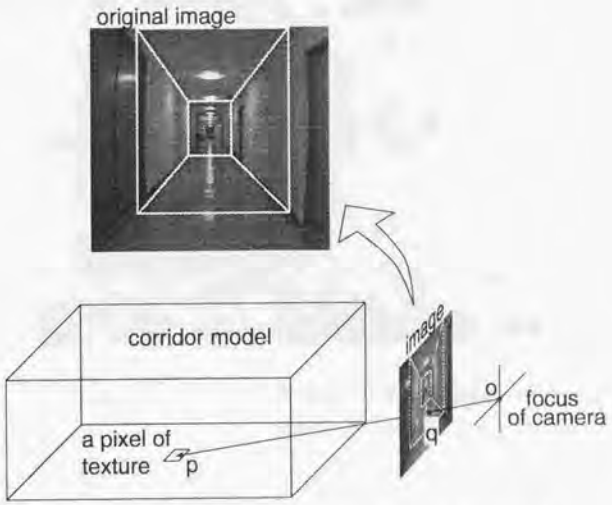


Figure 7.17 Texture generation by inverse perspective

7.7.4 テクスチャの更新

そのために，これまでの VSS で一度つくったテクスチャから生成される画像列を用いて自律走行できる距離は 5m 程度と短かかった．

逆透視変換によってテクスチャを生成するとき，ロボットは必ずしも廊下の中心で真正面を向いてるとは限らない．走行に用いる低解像度の画像では位置や姿勢のずれを正確に検出することは困難なためで，VSS で生成した画像列を用いた走行では，経路からの位置のずれは 5[cm]，姿勢のずれは 2[deg] 程度ある．ここで，位置のずれよりも姿勢のずれの方が画像への大きく影響するので，画像のずれは全て姿勢のずれによるとみなし，高解像度の画像を用いて画像のずれを検出し，逆透視変換するときに補正することにした．

VSS は任意の解像度の画像を生成することができるため，ロボットが停止した後に，その位置で見えるはずの画像をより高い解像度で生成してロボット側へ送る．この画像と，実際に

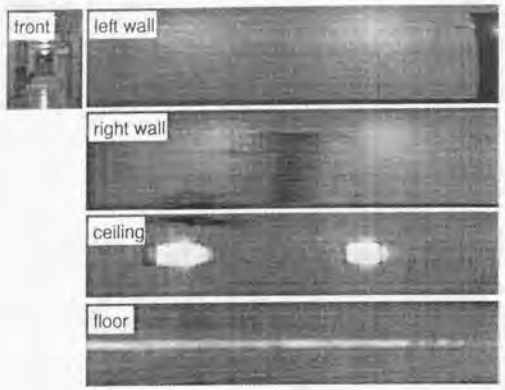


Figure 7.18 Generated textures

見えている画像を用いて相関演算を行ない，画像のずれを求める．この処理はフレームレートでは実行できないため，ロボットは停止してから処理を行なう．

7.7.5 仮想教示走行

経路生成は，VSS での仮想教示走行によって行なう．通常は，ロボット側から走りたい経路を VSS に送る．VSS はその経路に沿って仮想ロボットを進ませ，仮想教示走行を行なうことで画像列を生成する．生成された画像の例を **Figure 7.19**に示す．上の列は，テクスチャを作るために画像を取り込んだ位置を基準として，それぞれ0[m], 2.7[m], 4.5[m] の地点にロボットを進めた時に得られた仮想画像，下の列は実際の廊下における 0[m], 2.7[m], 4.5[m] の地点での画像である．テクスチャの解像度は一定ではなく，進むにつれて徐々に低下するため，4.5[m] の地点での画像ではテクスチャの粗さが目立っている．しかし，ここで生成された画像列は解像度を低げた後にマッチングに用いられるため，5[m] 程度までの画像は実画像とのマッチングに問題はなかった．

また，VSS は生成された画像列を使うとどのような走行軌跡が得られるかをシミュレートする機能を持っている [71] ので，必要に応じてあらかじめ VSS で仮想自律走行を行ない，走行軌跡を予想，評価してから実際の走行を行なうこともできる．

経路生成は，VSS での仮想教示走行によって行なう．通常は，ロボット側から走りたい経路を VSS に送る．VSS はその経路に沿って仮想ロボットを進ませる，仮想教示走行を行なう

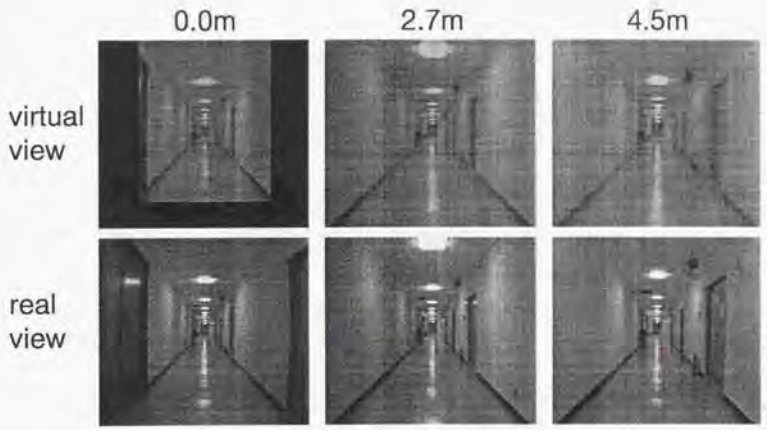


Figure 7.19 Generated virtual view

ことで画像列を生成する．

また，VSS は生成された画像列を使うとどのような走行軌跡が得られるかをシミュレートする機能を持っているので，必要に応じてあらかじめ VSS で仮想自律走行を行ない，走行軌跡を予想，評価してから実際の走行を行なうこともできる．

7.7.6 走行実験

実際の廊下において5[m] づつの走行を 4 回繰り返し，合計 20[m] の直線経路の走行を行なう実験を行なった．テクスチャを更新する時に，ロボットの姿勢のずれを考慮しない場合は誤差が蓄積されて 3 回目の更新時に廊下の中心線から約 40[cm] ずれることもあったが，姿勢のずれを検出して補正した場合には，走行の終了時にも 10[cm] 以内のずれに収まり，安定した走行結果が得られた．

第 8 章

屋外環境における経路誘導への展開

8.1 屋外でのビュー

前章までのビューシーケンスに基づく経路誘導は，建物内の廊下環境をターゲットとしていた．しかし，移動ロボットの行動範囲は，屋内だけでなく屋外にも広げることが必要であると考え，本章では移動ロボットの屋外移動について，ビューベーストアプローチがどの程度有効であるかを検討する．

屋内の廊下環境では「明るさなど，環境の見え方が時間によって大きく変化しない」という仮定が置けた．そのため，ビューのマッチングに濃淡画像の輝度値を使ったテンプレートマッチングを用いていることは，特に問題にはならなかった．しかし，視覚により移動する移動ロボットにとって，屋内環境と屋外環境の最大の違いは「明るさが時刻によって大きく変化する」ということであり，ビューシーケンスをそのまま屋外に適用しても，明るさの変化によりマッチングが失敗し走行できなくない．

画像の正規化により輝度の平均や分散を合わせることも試みたが，正規化は明るさの変化に対して多少は有効であっても，局所的な明るさの変化（太陽の向きによる影など）には対応することは困難であった．そこで本章では，明るさの変化に強いと考えられる，

- 微分画像
- 色相画像
- 視差画像

の 3 つを用いてそれぞれのビューを定義し，屋外での経路誘導実験を行う．このように生の画像を使わずに，別の特徴量に変換した画像をビューとして用いることになるわけであるが，本研究はビューベーストアプローチにより経路誘導を行うことを目的としており，これらの画像はできるだけ余計な特徴抽出を行わず生に近い画像であることが望ましい．特徴抽出を行い抽象的なデータを取り出すプロセスはノイズに弱く，ビューベーストアプローチの特徴であるロバスト性を失うことになるからである．そこでそれぞれの画像を定義するときにはデータの抽象化（シンボル化）を行わない．またデータの信頼性を定義しマッチングに用いることで，ロバストな環境認識を目指す．

8.2 エッジ画像による経路誘導

微分した画像は，もとの画像の明るさがシフト場合には全く変化しない．また局所的に画像の明るさが変化し，例えば画像中に極端に明るい部分が出たとしてもその部分に余分なエッジが出るだけで，正規化のように画像全体に影響を与えることはないため，正規化よりもよい結果が期待できる．

通常エッジ画像のマッチングを行う場合，微分画像から線分データを抽出してから線分同士のマッチングを行うことが多い．しかし，本研究ではデータの抽象化は行なわないという方針であるので，連続値を持つ 2 次元の微分画像をそのまま用いてマッチングを行う．

8.2.1 エッジ画像の生成

エッジ画像を生成するにあたり，微分フィルタとしては 3 × 3 の ラプラスフィルター，

$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

を用い，微分後に絶対値をとることにした．ただし通常の微分画像は，大部分が低い輝度の領域で，その中に幅の局所的な高い輝度の領域が存在することが多い．そのために，単純なテンプレートマッチングでは画像の変化により連続的にマッチングエラーが増加するという性質が得られにくい．

このことを簡単なパターンの例を用いて説明する．**Figure 8.1** は，画像中に 1 ピクセルの幅のエッジが 2 本存在するものをテンプレートとして用いた例である．この 2 本のエッジの間隔が徐々に広がる場合を考えると，探索画像は (1) ～ (4) のようになる．この画像の変化は，人間の目には連続的に見える．ここで，それぞれの探索画像とテンプレートを用いてマッチングを行ってみる．ただし画像の輝度は，白が 0，黒が 1 とする．するとマッチングエラーはそれぞれ 0, 10, 10, 5 となり，画像の変化に応じて連続に増加しないのである．

このような状況に対処するためには，エッジを膨張させることが有効である．**Figure 8.2** はテンプレートおよび探索画像のエッジをそれぞれ 1 ピクセル膨張させた場合を示す．これによりマッチングエラーはそれぞれ 0, 7, 21, 35 となり，画像の変化により連続的に増加するようになる．エッジ画像にはこのような性質があるため，エッジ画像の 2 次元的なマッチングを行う際にはエッジの膨張が不可欠である [78, 79]．

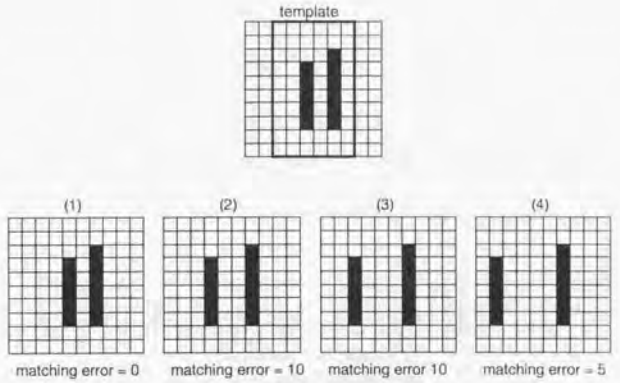


Figure 8.1 Matching of Edge Image without Expansion.

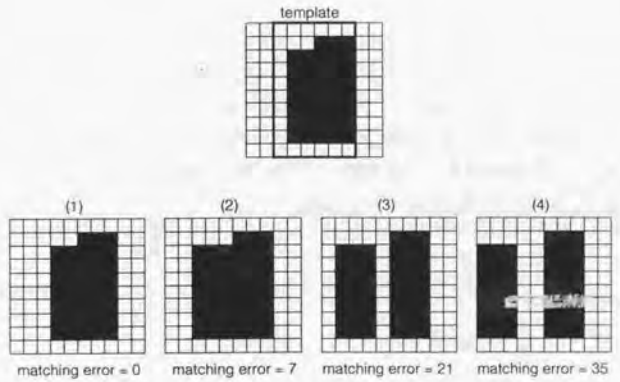


Figure 8.2 Matching of Edge Image with Expansion.

8.2.2 エッジ画像のマッチング

ここでは，まず予備的な実験として，屋内において明るさが変化しやすい窓際の廊下でエッジ画像を用いてマッチングを行った．そのマッチングの様子を **Figure 8.3** に示す．この場所では，窓から日がさしこむため，昼と夜で窓の部分の明るさが極端に変わる．画像の解像度は 32 × 32 で，まずカメラから取り込んだ画像を平滑化して通常のビューを生成した後に，の微分フィルタをかけた後に，その絶対値をとりエッジ画像とする．この実験では，同一の教示データ（夜に記憶）を用いて，昼夜両方で走行することができ，明るさの変化への適応性が示された [80]．

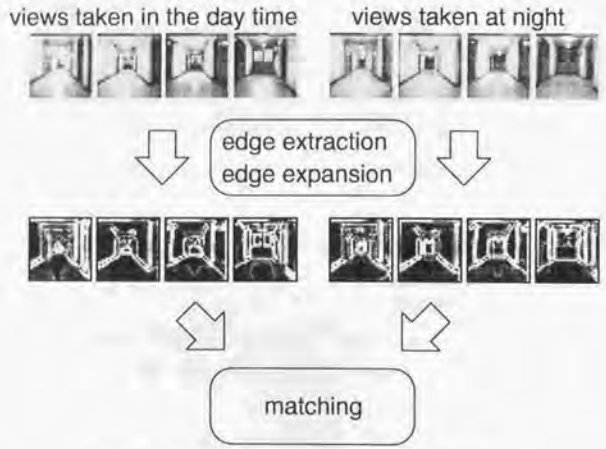


Figure 8.3 Matching of Edge View in Corridor.

次に，実際に屋外環境でマッチング実験を行い，どのような手順で生成したエッジ画像がビューシーケンスに適しているのかを調べた．エッジ画像を生成するには，

- 画像の縮小
- 微分
- エッジ膨張

の処理を行うが，これらの処理の順序やパラメータは何通りかのものが考えられる．これまでの研究では，エッジ画像の膨張処理を行っている例はあったが，いずれもどの程度の膨張が必要であるかを検討せずに行っている．ここでは，縮小と膨張の処理の順序，縮小した画像のサイズをいくつにするか，膨張するピクセル数をいくつにするか，について以下の組合せを試し検討する．

1. 画像のサイズは 32 × 24 か 64 × 48.
2. 処理順序は，縮小が先か微分が先かのどちらか．どちらでも膨張は最後に行う．
3. エッジ膨張のピクセル数は 0 ～ 3 の 4 通り．

(1) と (2) についてはそれぞれ 2 通り，(3) については 4 通りなので，合計で **Table 8.1** に示すように 16 通りの組合せがある．

Table 8.1 16 combinations for generation of edge-view.

	縮小画像のサイズ	微分と平滑化の順序	エッジの膨張数
a ～ d	32	微分が先	0~3
e ～ h	32	縮小が先	0~3
i ～ l	64	微分が先	0~3
m ～ p	64	縮小が先	0~3

実験を行った環境を **Figure 8.4** に示す．ロボットはこの画像が見える位置でテンプレートを記憶し，そこから徐々に移動しながらマッチングエラーを記録してゆく．このときのマッチングエラーの結果を a~d, e~ h を **Figure 8.5** に，i~l, m~p を **Figure 8.6** にそれぞれ示す．画像の善し悪しは，マッチングエラーが「滑らかに，長い区間を単調に増加する」かどうかで判断する．マッチングエラーの変化が，位置の変化に応じて滑らかに単調増加すればビューシーケンスに用いることができ，また単調増加する範囲が広いほど画像を記憶する間隔が長くてよい．この点に注意しながらこのグラフを分析すると，

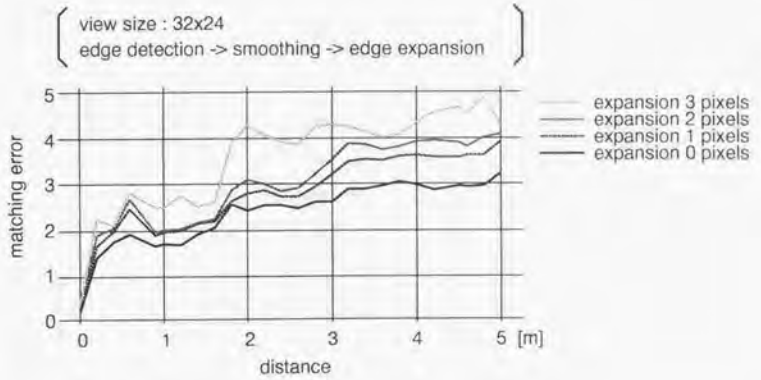
1. 画像の縮小を先に行う方が，微分を先に行うよりもよい．
2. テンプレートのサイズは 64 × 48 の方が，32 × 24 よりもよい．
3. 膨張の効果は 0 ～ 3 ピクセルであまり変わらない．

ということが分かる．(1),(2) から m ～ p の 4 つが残り，そのうち o (64 × 48，縮小が先，膨張 1 画素) を採用し，以後の実験に用いる．膨張の効果がほとんどみられなかった理由

としては，画像を縮小する際に元の画像の $10times10$ もしくは 5x5 画素の平均を取っているため，画像をなます効果がすでに入っているため，それ以上の膨張処理の効果が少ないことが考えられる．

Figure 8.4 Outdoor Environment for Basic Matching Experiment of Edge View

次にカメラの絞り値 (iris) を変え，疑似的に得られる画像の明るさを変えた場合のマッチングエラーの値を調べた．**Figure 8.7**にその結果を示す．基準となるテンプレート画像は，絞り値 10 で生成した．その後，絞り値を 10, 8, 6 と変えながらテンプレートを記憶した位置からロボットを移動させた．この図から，絞り値 10 で記憶したので当然 10 のままではロボットの移動によってマッチングエラーが単調増加してゆくだけでなく，絞り値 8 でも 2 [m] 程度まではマッチングエラーが増加してゆくことが分かる．ただしその傾きは絞り値 10 の場合よりもゆるくなっており，さらに絞り値を 6 にするとマッチングエラーはほとんど平坦になってしまい，移動距離との相関がほとんど見られなくなっている．



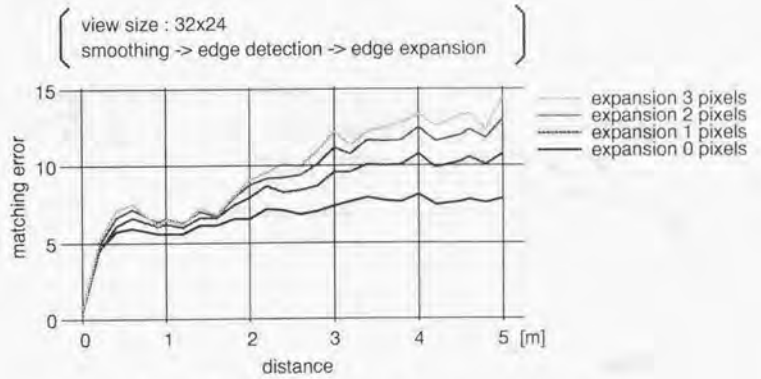


Figure 8.5 Result of Basic Matcing Experiments (1).

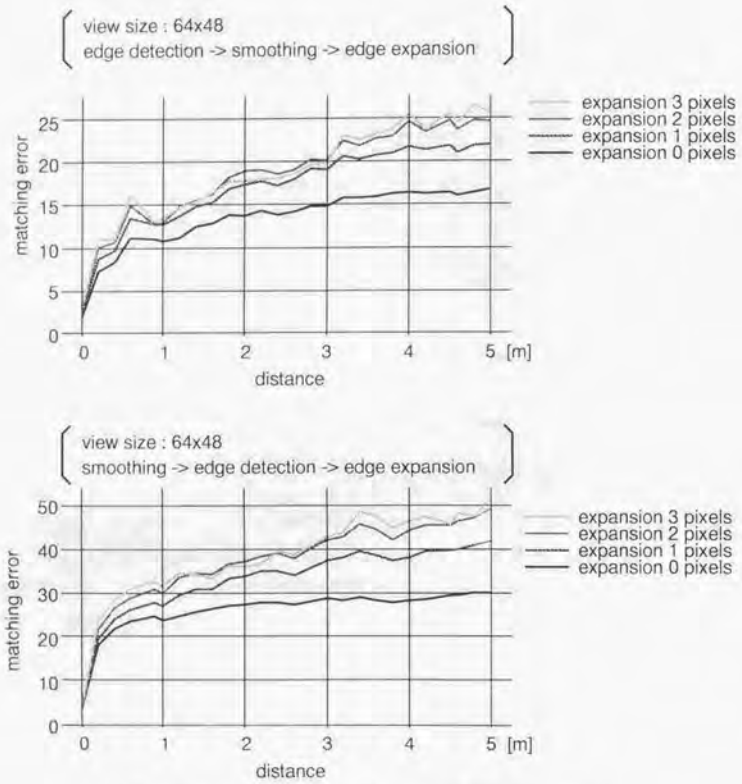


Figure 8.6 Result of Basic Matcing Experiments (2).

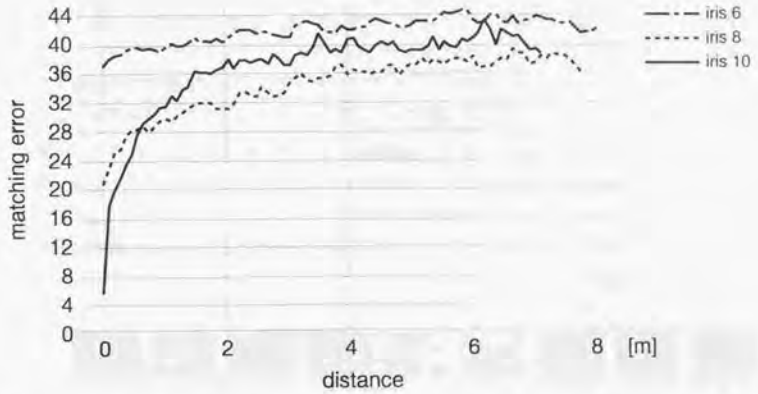


Figure 8.7 Result of Basic Matching Experiment with Changing the Parameter of Iris.

8.2.3 エッジ画像を用いた経路誘導

ここまでのマッチング実験により，エッジ画像はビューシーケンスに利用できることが分かったので，ここでは実際にエッジ画像を用いて屋外経路を教示し，走行時間を変えて明るさの変化に対応できるかどうかを調べた結果を **Figure 8.8**に示す．教示走行は 12 時に行ない，自律走行は 12 時と 16 時に行なった．**Figure 8.9** 上に示す 12 時の自律走行では，走行環境に変化がないのでマッチングは安定している．16 時の自律走行では，太陽が沈む時間に近くなっており，かなり暗くなっていた．そのため，そのままでは走行できなかったため，絞り値を 10 — 15 に変えて，得られる画像を明るくして走行を行なった．その結果が **Figure 8.9** 下のグラフである．マッチングエラーの値は上のグラフと比べて全体に大きく，また各曲線の最小値や，曲線同士が交差するときの値は上下に変動しており，マッチングがやや不安定になっていることが分かる．しかし各マッチングエラーの最小値をとる位置は $P1 \sim P11$ とほぼ一致しており，ロボットは正しく位置を認識しながら走行していることが分かる．

このように，エッジ画像を用いたビューシーケンスにより，明るさが変化した環境でも走行することは可能になったが，12 時の画像でそのまま 16 時に走行することができなかったように，やはり明るさが大きく変化してエッジの強度が変化する場合には走行は難しい．微分フィルタの大きさを変える，あるいはエッジ画像をさらに正規化するなど，改良の方法は幾つか考えられるが，ここまでで既に “画像の縮小と微分の順序”，“エッジ膨張の画素数” といったパラメータが出てきており，さらにパラメータを増やした上で最適なものを実験的に選択することはますます困難になると思われる．

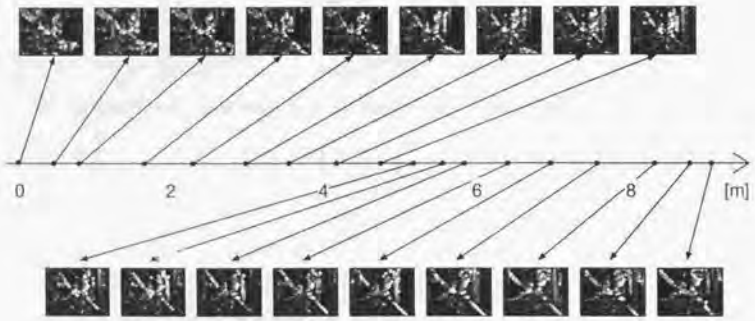


Figure 8.8 Memorized Edge View Sequence.

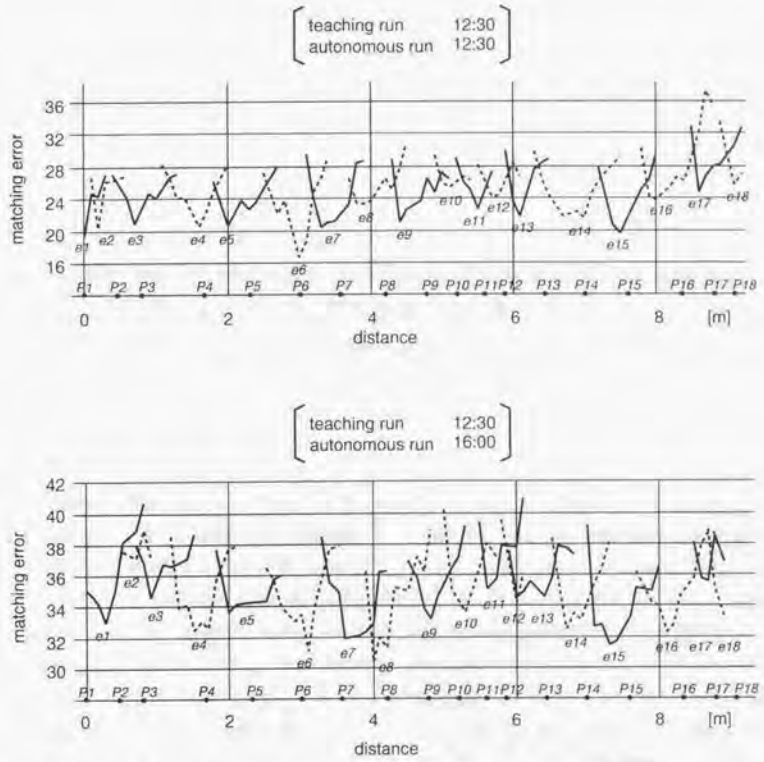


Figure 8.9 Experimental Result of Matching in Autonomous Navigation.

8.3 色相画像を用いたビュー

濃淡画像は，各画素は輝度のみの 1 次元の情報しか持っていないため，明るさの変化があった場合にはマッチングがとれなくなる．しかし，カラー画像は 3 次元の情報を持っており，通常は RGB，HSV，YCbCr などいくつかの表色系で表現される．このうちの HSV 表色系の H (Hue) は色相と呼ばれ，“赤”，“青” など色の種類を表現する値で，理想的には明るさの変化には影響されない成分である．また S (Saturation) は彩度とよばれ色の鮮やかさを，また V (Value) は輝度をそれぞれ表現している．ここでは，この明るさの変化に影響されにくい色相を，ビューのマッチングに用いることを試みる [81]．つまり，各画素に色相の値持つ画像をつくりマッチングに用いるということで，この画像のことを “色相画像” と呼ぶ．

8.3.1 色相画像の生成

HSV 表色系で表される各成分 (h, s, v) は，YCbCr 表色系 で得られるカラー情報 (y, cb, cr) のうち $\vec{c} = (cb, cr)^T$ によって表されるベクトルを用いて以下のように定義される．

$$
\begin{aligned}
s &= |\vec{c}| \\
h &= \arg\vec{c} \\
v &= y
\end{aligned}
$$

この定義から h のとる範囲は $0 \leq h < 2\pi$ である．また Cb, Cr から色相，彩度への変換と，色相の値が示す色領域を **Figure 8.10** に図示する．色相は Cb-Cr 空間において，赤 → 黄 → 緑 → 水色 → 青 → 紫 → 赤，の順に環状に分布している．また Cb-Cr 空間の原点は彩度が 0 であり，Y の値により黒，灰色，白といった色彩がない色に対応する点である．

また，画像が RBG 表色系で得られる場合，YCbCr 表色系への変換は

$$
\begin{pmatrix} y \\ cb \\ cr \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.299 & -0.587 & 0.886 \\ 0.701 & -0.587 & -0.114 \end{pmatrix} \begin{pmatrix} r \\ g \\ b \end{pmatrix}
$$

で表される．ただし本実験で用いるカラートラッキングビジョンボードは RGB 入力モード以外に YCbCr 入力モードを持つため，この変換は必要ない．入力画像から色相を取り出すためには，まずノイズの影響を低減するために Cb, Cr それぞれを画面全体で平滑化し，32 × 24 の画像を生成した後に，色相に変換する．

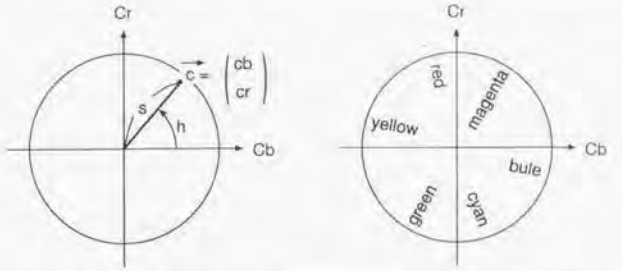


Figure 8.10 Hue and saturation in Cb-Cr space.

8.3.2 色相画像を用いたマッチング

色相は輝度と同じく 1 次元の情報であるが，**Figure 8.10** の右に示すように環状に分布しているので 0 と 2π は等しく，2 つの色相の距離は角度の差のうち小さい方をとることになる．そこでマッチングに用いる 2 枚の色相画像の画素 h_1, h_2 の距離 d は，

$$
d(\, h_1, h_2\,) = \min(\, |\, h_1 - h_2\,|\,,\ 2\pi - |\, h_1 - h_2\,|\,)
$$

と定義する．すると 2 つの色相画像のマッチングエラー e は

$$
e(i) = \sum_{x=u}^{xsiz\,e-u} \sum_{y=0}^{ysiz\,e} d(\, h_1(x, y), h_2(x + i\,, y))
$$

$$
e = \min_{i=-u}^{u} e(i)
$$

と定義できる．しかし，彩度が小さい (白，灰色，黒に近い) 領域の色相は，もとの画像の Cb, Cr 成分にのるノイズの影響によって大きく変化しやすく信頼性が低いので，このままではマッチングエラーはノイズに影響されやすい．そこで色相の信頼度を定義し，マッチングの際に重み付けを行うことで信頼性の低い情報の影響を抑える．信頼度には彩度を用いるのが妥当であるように見えるが，実は彩度は輝度によって変動する．同一物体でも，例えば明るさが異なって (r, g, b) がそれぞれ 2 倍になったとすると，そこから線形変換される (y, cb, cr) もそれぞれ 2 倍になるため，HSV 表色系の s, v もそれぞれ 2 倍になるのである．そこでここでは色相の信頼度 r を

$$
r = \frac{s}{v}
$$

と定義する．その上で 2 つの色相画像のマッチングエラー e は

$$e(i) = \frac{\sum_{x=u}^{xsize-u} \sum_{y=0}^{ysize} d(h_1(x,y), h_2(x+i,y)) \cdot \min(r_1(x,y), r_2(x+i,y))}{\sum_{x=u}^{xsize-u} \sum_{y=0}^{ysize} \min(r_1(x,y), r_2(x+i,y))}$$

$$e = \min_{i=-u}^{u} e(i)$$

と定義する．ただし，$xsize, ysize$ は色相画像の大きさ，u は探索範囲である．この場合は横方向にのみ探索領域を設定し，縦には探索しない．このように複雑なマッチングエラーの計算はこれまで用いてきたハードウェアによるテンプレートマッチング [45] では処理することができないため，ソフトウェアで行なう．

色相の信頼度による重み付けの効果を **Figure 8.11** に示す．ロボットは静止しており，同じ場所の画像をテンプレートと探索画像として用いている．図中の実線は重み付けをした場合のマッチングエラー，点線は重み付けをしなかった場合のマッチングエラーである．理想的には同じ場所の画像を与え続けているので，マッチングエラーは理想的には 0 になるはずだが，ノイズの影響でどちらも変動が見られる．変動の大きさは重み付けをした場合の方が小さくなっており，その効果が確認できる．重み付けの効果は場所によって異なり，画像中の有彩色の割合が少ないほど効果が大きい．なお，重み付けしたことにより，2 つのグラフは分散だけでなく平均も異なる．

次に，屋外においてマッチング実験を行った．実験環境を **Figure 8.12** に示す．ロボットはこの画像が見える位置でテンプレートを記憶し，そこから徐々に移動しながらマッチングエラーを記録してゆく．このときのマッチングエラーの結果を **Figure 8.13** に示す．実線は重み付けをしたもの，点線は重み付けしていないものである．この図からは，2[m] 程度はマッチングエラーが滑らかに単調増加しており，また重み付けをした方が全体にマッチングエラーが小さく安定していることが分かる．

8.3.3 色相画像を用いた経路誘導

次に，色相画像のビューシーケンスを記憶し，自律走行させる実験を行った．**Figure 8.14** は，教示走行のすぐ後に自律走行を行ったときのマッチングエラーである．教示走行では約 10[m] の経路に対して 9 枚の色相画像を記憶した．

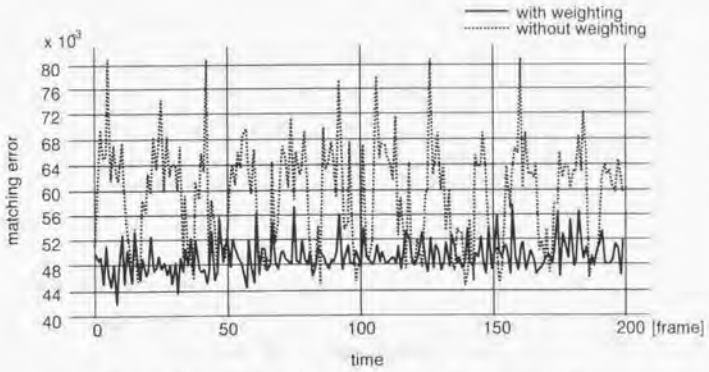


Figure 8.11 Effect of Weighting with Reliability of Hue.

Figure 8.12 Outdoor Environment for Basic Matching Experiment of Hue View

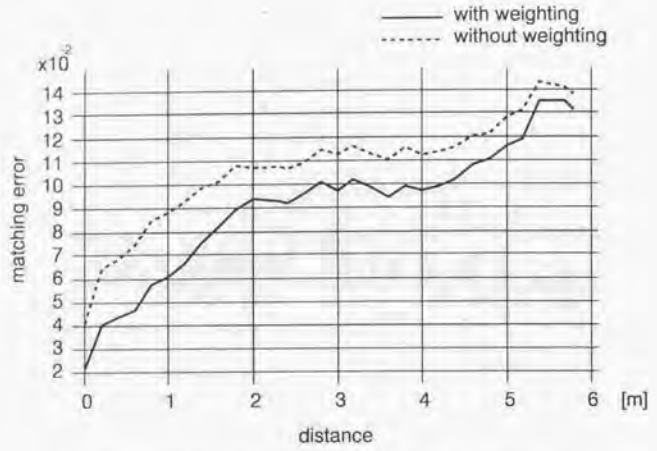


Figure 8.13 Result of Basic Matcing Experiment.

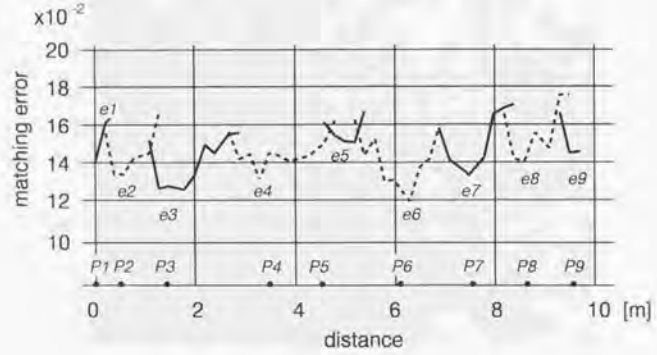


Figure 8.14 Experimental Result of Matching in Autonomous Navigation.

8.4 視差画像を用いたビュー

エッジ，色相ともに屋外における明るさの変化への適応度は期待したほど高くなかった．これは明るさの変動がカメラの性能（ダイナミックレンジ，色の再現性）を越えてしまっており，エッジや色相が見え方の不変量にはならなかったことを意味する．そこで，ここではこれまでの完全な 2 次元情報だけからなるビューの定義を拡張し，3 次元的な環境の情報を含む“視差”を用いてビューシーケンスを作ることを試みる．

視差とは，ステレオカメラを用いて両眼立体視をした時の，対象物体の画像中の位置の差（ずれ）のことである．平行ステレオ（カメラの光軸を平行に設定）の場合，視差が 0 であれば対象は無限遠に，また視差が大きくなればなるほど近くにあることになる．

視差は距離によって決まるものであるが，距離を記憶しマッチングに用いるのはビューベーストアプローチとは言えない．距離は幾何学的な情報であり「見え方」とは異なる．また，視差から距離への変換は線形でなく，視差が 0 に近くなると分解能が極端に下がるため，遠方の物体までの距離は大きな量子化誤差を含み，ノイズやキャリブレーション誤差にも弱い．これに対して視差は画像から直接得られる「見え方」の差であり，ビューベーストアプローチに適合した特徴量であると考えられる．**Figure 8.15** は廊下での視差（左）と，それを用いて復元した廊下の形状（右）である．このうち視差は比較的安定して得られるが，廊下の形状は分解能が低い上に時間的な変動が大きく，直接マッチングに用いることができるような特徴量ではない．

8.4.1 視差画像の生成

視差画像は，第 6 章で用いたのと同じ両眼立体視の手法により生成する．視差画像の大きさは 32 × 1 で，画面上の中央に横 1 ラインに視差を検出するテンプレート領域を配置する．**Figure 8.16** は屋外において取り込んだ画像（実際にはステレオ画像）と，そこから生成した視差画像 (stereo disparity) と，視差の信頼性 (reliability of stereo matching) である．視差画像は明るいほど大きな視差，つまり近い物体を示す．**Figure 8.16** では通路の右側近くの木とやや左側の離れた位置にある木の部分が明るくなっているのが分かる．また，視差の信頼度は第 6 章で定義したものを用いている．視差画像が 1 ラインしかないのは，処理時間の制限によるものである．ここでの視差画像の生成には，

- ハードウェアによるテンプレートマッチング機能を利用する．
- フィールド多重化によりステレオ画像を 1 画面に混合して入力する．
- 粗密法により，ステレオマッチングの回数を削減する．

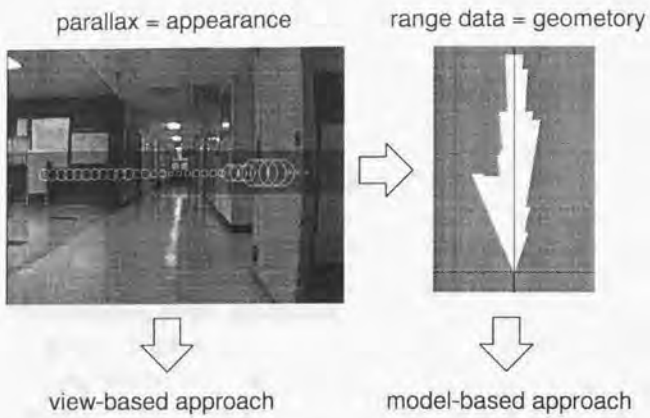


Figure 8.15 Stereo Disparity and Range Data.

などの工夫をしており，通常の画像処理システムを用いるよりは効率がよいが，それでも処理時間が 200[ms] 程度かかっている．移動ロボットでは処理のリアルタイム性が必要であるので，これより大きなサイズの視差画像を用いることは現状では無理であると考えられる．

8.4.2 視差画像のマッチング

記憶したテンプレート視差画像を T，走行時の視差画像を S とする．それぞれの画像の各画素の視差を $d_T(x), d_S(x)$ とおくと，視差画像だけを用いたマッチングエラーは以下のように定義できる．

$$e(i) = \sum_{x=u}^{xsize-u} |d_T(x) - d_S(x+i)|$$

$$e = \min_{i=-u}^{u} e(i)$$

しかし，色相画像で行ったのと同様にここでも視差画像の信頼度 $r(x)$ を用いてマッチングエラー e を以下のように定義した．

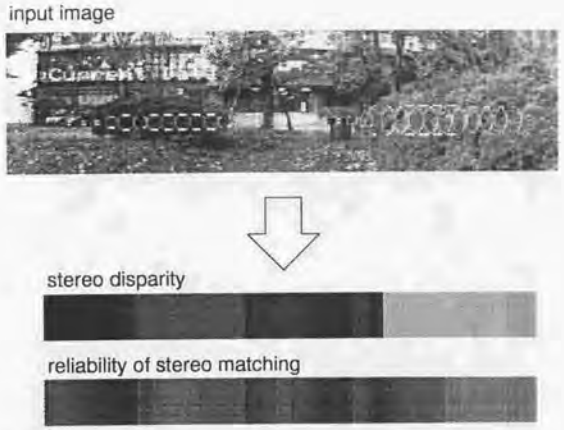


Figure 8.16 Stereo Disparity View.

$$e(i) = \frac{\sum_{x=u}^{xsize-u} |d_1(x) - d_2(x+i)| \cdot \min(r_1(x), r_2(x+i))}{\sum_{x=u}^{xsize-u} \min(r_1(x), r_2(x+i))}$$

$$e = \min_{i=-u}^{u} e(i)$$

次に，屋内環境においてロボットの移動距離とマッチングエラーの関係を調べた．実験を行った環境を **Figure 8.17** に示す．ロボットはこの画像が見える位置でテンプレートを記憶し，そこから徐々に前方に移動しながらマッチングエラーを記録してゆく．この実験の結果を **Figure 8.18** に示す．図中の実線は視差の信頼度により重み付けを行った場合，点線は重み付けを行わなかった場合の結果である．この実験結果では 4[m] 直前までマッチングエラーは滑らかに単調増加しており，極めて良い結果が得られている．これは，実験環境が **Figure 8.17** 示すようにはじめは画面全体が遠く，ロボットが前進するに従ってそれらが同時に近付いてくるという，変化の出やすい所であったためである．

次に，屋外環境においてロボットの移動距離とマッチングエラーの関係を調べた．実験を行った環境を **Figure 8.19** に示す．ロボットはこの画像が見える位置でテンプレートを記憶

Figure 8.17 Indoor Environment for Basic Matching Experiment of Hue View

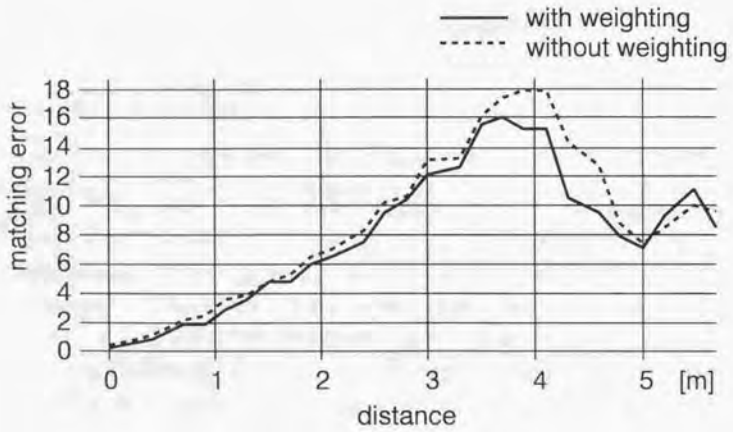


Figure 8.18 Result of Basic Matcing Experiment.

し，そこから徐々に前方に移動しながらマッチングエラーを記録してゆく．この実験の結果を Figure 8.20 に示す．図中の実線は視差の信頼度により重み付けを行った場合，点線は重み付けを行わなかった場合の結果であり，重み付けをした方がマッチングエラーの値は変動が小さく，この実験結果では 2.5[m] までマッチングエラーは単調増加しているとみなせる．屋内，屋外のマッチング実験の結果から，教示走行時に新たな画像を記憶するためのマッチングエラーの閾値（単調増加するとみなせる限界）は 6 と設定した．

Figure 8.19 Outdoor Environment for Basic Matching Experiment of Hue View

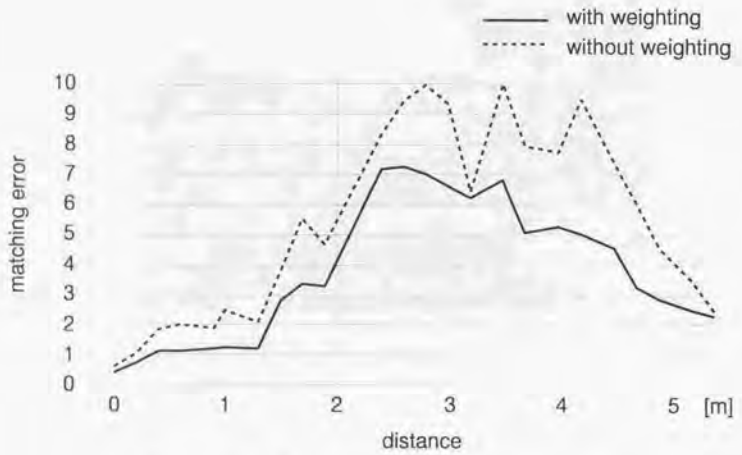


Figure 8.20 Result of Basic Matcing Experiment.

8.4.3 視差画像を用いた経路誘導

実際に屋外経路おいて，視差画像のビューシーケンスを教示したところ，約12[m] の経路に対して 9枚の画像を記憶した．記憶した視差画像のビューシーケンスと，その位置での写真，記憶した位置を **Figure 8.21**に示す．視差画像を並べてみると，写真に示した ① ～ ⑤ の物体（木など）が，視差画像中で中心近くに現れ，徐々に両端へ向かって移動していくのが分かる．

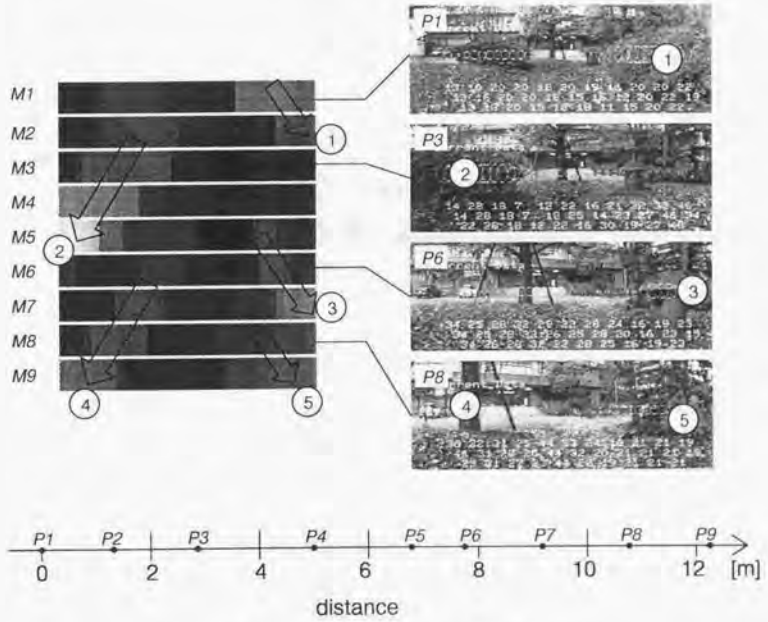


Figure 8.21 Memorized Disparity View Sequence.

この教示データを用いて，自律走行を行った時のマッチングエラーのグラフを **Figure 8.22** に示す．マッチングエラーは教示時の閾値 (6) を越えることなく，正しく推移していた．ただし，マッチングの基礎実験のグラフから分かるように，視差画像の生成自体が十分安定であるとは言い難く，まだ改良の余地がある．また，処理時間が 7[frame] かかっている点も移動ロボットにとって十分な処理速度が達成できているとは言えない．

しかし，この視差画像を用いたビューシーケンスは，本質的に明るさの変化には影響されない，環境の形状に関する情報を用いているものであり，エッジ画像や色相画像よりも将来性が高いと考えられる．ステレオマッチングを高速に行う研究は現在盛んに行われており，将来的にはより密で安定した視差を得ることができると期待できるからである．

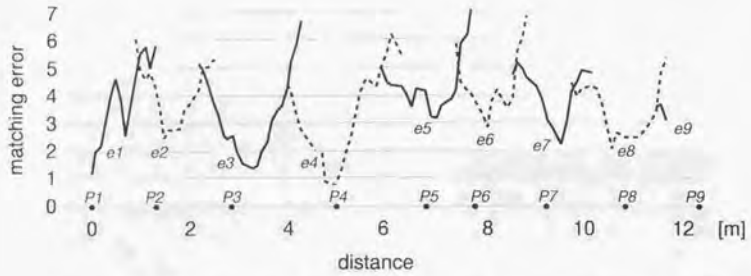


Figure 8.22 Experimental Result of Matching in Autonomous Navigation.

第 9 章

結論および考察

9.1 結論

本研究では，ビューベーストアプローチに基づき，新たな移動ロボットのための環境表現およびそれを用いたナビゲーション手法を提案し，実環境での走行を実現した．ビューベーストアプローチは，近年注目されている3次元情報を用いずに2次元画像をそのまま認識に用いるものである．このアプローチでは，予め認識対象の様々な見え方を記憶し，画像の照合により認識を実現する．従来のアプローチよりも2次元画像をより直接的に利用し，3次元モデルが必要ないため，実環境の複雑な対象の認識にも適用することができると期待されている．

第2章「移動ロボットにおける視覚に基づく環境認識」では，まず近年コンピュータビジョンの分野で注目されている，3次元情報を用いずに 2次元画像をそのまま認識に用いる“view-based” あるいは “appearance-based” と呼ばれるアプローチについて検討し，移動ロボットにおける “見え方” の記憶に基づく環境認識手法を提案した．見え方の記憶に基づく環境認識という考え方は，本研究を通して一貫して用いられるものであり，ここではこのアプローチで鍵となる「どのような見え方を記憶するか」「どのような照合(マッチング)の手法を用いるか」について検討した．次に，従来の視覚を持つ移動ロボットの研究の流れについて考察し，これまでの移動ロボットのために提案された地図および経路表現における問題点を整理した．

第3章「視覚移動ロボットのシステム開発」では本研究で用いる屋内実験用と屋外実験用の2台の移動ロボットの概要，およびそれらを構築するにあたり開発した以下の2つの技術について述べた．一つは，ロボットの共通のカーネル部分と個々のロボットに依存する部分に切り分け，カーネル部分を標準化することで，開発を効率的に行なうことができるようにすることを目的とした PC ベースの知能ロボット汎用カーネル，もう一つは，2つのビデオ信号をフィールド毎に切替えて画像処理システムに入力する手法(フィールド多重化)により移動ロボットに搭載できるコンパクトなステレオ視覚システムである．

第4章「ビューシーケンスに基づく経路誘導」では，移動ロボットのための “見え方” に基づく経路表現として “ビューシーケンス” を提案した．これまでの移動ロボットの研究においても，経路の見え方を記憶や学習に利用している例はあったが，本手法ではリアルタイムに経路上のどこにいるのかを認識しながら同時に誘導を実現することを可能とした．経路モデルとしては，走行時に見える進行方向の視野全体の画像の列を用い，画像のマッチングにはハードウェアによる高速なテンプレートマッチングを利用する．走行時には，記憶したビューシーケンスと現在の見え方のマッチングにより環境認識を行なう．また，見え方をモデルとして用いるために，実環境に対応できる経路モデルの作成が容易であるという特徴がある．

第5章「全方位ビューシーケンスに基づく経路誘導」では，全方位視覚センサを利用してこれまでのビューシーケンスを拡張した環境表現“全方位ビューシーケンス”を提案した．全方位視覚センサを用いて得られた画像を円筒状の投影面に変換し，マッチングにはテンプレートマッチングを用いる．全方位ビューシーケンスはこれまでのビューシーケンスよりも視野が広いために含まれている情報が多く，環境の変化に強いマッチングを実現できた．また，全方位視覚センサは複数(例えば前方と後方の2つ)の異なる方向を向いたカメラを持つのと同じ効果が得られため，単純な制御方法でこれまでより正確な誘導を行うことが可能となることを示した．

第6章「自律的な地図表現の獲得」では，ロボットが視覚により自律的に環境を動き回り，建物内の環境表現(地図)を獲得する手法について述べた．前章までで述べた経路誘導の手法では，あらかじめオペレータが何らかの形でロボットを移動させることで経路の見え方を教示する必要があった．また，ある地点からある地点までの経路を教示するのは容易であるが，建物全体の地図を作成するのは手間がかかる作業であった．しかし，ロボットが自律的に環境を動き回ることができれば，自動的に地図を生成することが可能となる．

ここでは，まずはじめにステレオ視覚および全方位視覚を用った，両眼立体視やオプティカルフローにより実現した，走行可能な空間を検出する手法について述べた．この機能を用いてロボットは建物内の自律移動を行ないながら，同時に全方位ビューシーケンスを記憶していくことで，建物内の環境全体の地図を獲得した．地図を獲得した後には，ロボットはゴールを指示されると獲得した地図から経路を生成して，自律走行することが可能となった．

第7章「仮想環境を用いたビュー・シミュレーション」ではグラフィックスワークステーションと画像処理システムを組み合わせて開発した，視覚を持つ移動ロボット用のシミュレータについて述べた．

本研究では環境の幾何情報を利用せず，見え方を直接利用して環境の認識を行いロボットの行動を実現しており，通常のミュレータでは行動をシミュレートすることはできない．なぜなら，通常のシミュレータではロボットや環境は2次元平面上に表現されており，ロボットの位置や障害物までの距離を直接得ることで行動を生成するので，見え方から行動を直接生成する本研究での手法とは合わないのである．

そこで，本章では見え方をグラフィックスワークステーションを用いて生成し，画像処理システムを用いてその画像を処理して仮想的な行動を行ない，見え方を更新する，という処理ループを持つ視覚移動ロボット用のシミュレータ“View Simulation System”を提案した．本システムを用い，まずビューシーケンスを用いた誘導手法のシミュレーションを行うことが出来ることを実験により確認した．また，仮想的な教示走行を行なうことで，実際の教示なしに実ロボットが経路を計画しながら走行を行なうことが可能となることを示し，最後にロボットが見え方を想像しながら行動する“Online View Simulation”という概念を提案した．

第8章:「屋外環境における経路誘導への展開」

前章までの経路表現では，見え方として濃淡画像をそのままの形で，記憶とマッチングに利用していた．しかし，この見え方は明るさの変化に弱いという欠点がある．本章では屋外環境での経路誘導を実現することを目指し，“見え方”の範囲を広げて，濃淡画像よりも明るさの変化に強い特徴量を用いた画像を 見え方として利用することを試みた．具体的にはエッジ，色相，ステレオの視差を利用し，それぞれについて評価，検討を行なった．

9.2 考察

本研究で提案した“ビューシーケンス”による経路誘導手法は，これまでに研究されてきた分析的に環境を認識する視覚による移動ロボットの誘導方法と比較して，単純でありながら現実世界の複雑さに対応できる現実的な方法であると考えられる．これはビューベーストアプローチの特徴であるとも言うことが出来る．また，本研究ではビューシーケンスに利用可能なビューの性質として「ビューのマッチングエラーが，ある区間で単調増加すること」が必要となることを指摘した．そしてこの基準にあうものであれば，これまでのビューベーストアプローチの研究において一般に用いられてきた濃淡画像以外に，微分画像や色相画像，さらに視差画像を用いても認識が可能であることを示し，このアプローチの発展の可能性を示した．

しかし，ある環境でビューシーケンスがどの程度利用可能であるかは実験を行なわないと分からなく，またある環境に最も適したビューがどれであるか定量的に判断する方法もまだ確立されていない．このような評価に関しては，今後の課題として残る．

ビューシーケンスによる経路誘導するという考え方は，Figure 9.1のように表現することができる．通常の経路誘導は1 と 2 の矢印だけを行なうループで，3次元モデルを介在しない．また View Simulation System を用いた仮想的な経路生成は 4 の矢印に対応する．現在3は人手で行なう，もしくは決まった位置にロボットを置いてから自動で行なっており，環境の形状に対する制限も多い．このモデリングの自由度を上げることは，今後の課題として残る．ここで，視差画像は環境の3次元構造と関係がある「見え方」であり，3次元モデルとビューベーストアプローチを統合する新たな手法に発展する可能性があると期待される．この場合，必要なモデルの精度は，必要となるビューの精度で決定できるようになるのではないかと考えている．

また，ビューシミュレーションシステムとして提案した，ロボットが見え方を自ら生成し想像しながら行動をするというアプローチは，移動ロボットだけに限らず，今後のロボット研究におけるコンピュータグラフィクスの新たな利用法として発展が期待される．3次元グラ

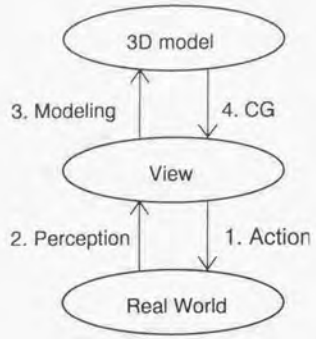


Figure 9.1 View and 3D model.

フィクス表示ボードは，パーソナルコンピュータのゲーム用途に急速に発展している分野であるため，高いコストパフォーマンスのものが容易に利用できるようになってきている．そのため，本研究のように無線ネットワーク経由でロボットと画像生成装置を接続するという大がかりなシステムでなく，ロボットのコントローラとしてのパーソナルコンピュータ 1 台だけで，CG を表示し，その画像を直接画像処理ボードに入力し現実の画像と比較するというシステムが現実になるのは近いと期待される．

謝辞

本論文は，東京大学 大学院工学系研究科 情報工学専攻に在学中，東京大学工学部 機械情報工学科 井上博允教授の御指導の下で行なわれた研究をまとめたものです．

井上教授には，学部4年からはじまり6年間にわたって，たいへん興味深いテーマを与えて頂き，研究を進め方，まとめ方，発表の仕方など，基礎的なところから指導して頂きました．筆者の研究の要所要所においてはっとするような素晴らしいアイデアを出して頂き，またロボットの購入の際には筆者を工場へ連れていって行って一緒に検討して頂いたりと，大変お世話になりました．心から感謝致します．

東京大学 大学院工学系研究科 情報工学専攻 田中英彦教授，同専攻 武市正人教授，機械情報工学専攻 三浦宏文教授，先端科学技術センター 佐藤知正教授には，本論文をまとめるにあたり有意義な御助言を頂き，大変感謝しております．

東京大学 工学部機械情報工学科 稲葉助教授には，研究室に入って以来数々の御意見・御助言を頂き，深く感謝致します．

近野敦助手には，英語の論文をまとめたり，海外に発表に行く際には大変お世話になりました．また，近野さんの人脈の広さから，学会での懇親会でたくさんの方々と知合いになることもできました．

また，先生方には2号館から遠く離れた9号館において，筆者ら学生たちの自主性を大きく尊重して頂き，自由な雰囲気の元で研究を進めることが出来たことを大変感謝しております．

また，藤田技官，戸塚技官，秘書の千代延さん，大澤さん，宮治さんには備品の購入などでお世話になりました．厚く感謝致します．

9号館で一緒に研究を行なってきた柴田さん（現 ERATO），寺岡君（現 NTT），稲邑君，宮田君（現 富士通），宮崎君（現 キーエンス），安藤君（現 セガ），坂井君，椛沢君，八重樫君，池田君，それに研究員として1年間9号館にいらっしゃっていた武田さん（コマツ）には，研究の面においても，生活の面においても大変お世話になりました．

特に9号館グループ設立当時からのメンバーの長である柴田さんにはある意味では先生方以上にお世話になりました．研究の考え方，サーベイの進め方からプログラミングの細かなアイデアに至るまですべての面で，たくさんのアドバイス，サポートを頂きました．柴田さんの幅広い知識と頭の回転の速さには，いつも驚かされましたし，9号館で安心して研究ができたのも柴田さんのおかげであり，大変感謝しています．本論文の随所で柴田さんとの議論から得た知見を生かせたと思います．

宮田君には，ロボットの製作を行なって頂きました．筆者が何かと忙しかった時期には，製作の大まかな指示を出すだけしか出来ませんでしたが，宮田君は MOUSE の回路を調べながら素早く完成させてくれました．このロボットは，システム構成を様々に変えながらも本論文

での実験に最後まで用いてきたものであり，大変感謝しております．稲邑君には，持ち前の明るいキャラクターで，9号館の雰囲気場を盛り上げて頂きました．また，武田さんにも様々な面でお世話になりました．本論文での最も基本的な走行方法である，画面全体を平滑化した粗い画像を用いて走行する手法は，武田さんがニューラルネットで廊下を走行する実験を見ていて思いついたアイデアです．この手法がうまくいったことが，筆者が新しいロボット MOUSE を作ることに，さらには博士課程に進学することになるきっかけになりました．宮崎君には，ONXY のプログラミングを一手に引き受けてもらい，シミュレーションシステムを構築してもらいました．もし修士で卒業していなければ，まだまだ ONYX を使った面白い実験が出来たのではないかと，とても残念に思っています．坂井君には，屋外でのエッジ，色相，視差のビューシーケンスの実験や，パンチルタの設計などで多大な成果を出してもらいました．池田君は，筆者が博士課程最終年度に 9 号館に来て，すばらしい実験成果をつぎつぎと出してくれました．全方位視覚やステレオ視，トラッキングビジョンなどの実験装置を使いこなし，予想以上のスピードで次々と結果を出すその才能には，本論文をまとめるに当たり大変貢献してもらいました．池田君なしでは，筆者は今年度で卒業することはありえなかっただろうと思っています．これらの 9 号館の皆さんのおかげで，楽しい研究生活を 9 号館で送ることができたことを感謝します．

また，研究室 OB の石綿さん（現 ETL），森さん（現 東京大学 先端研），相山さん（現 東京大学 精密工学科），岡さん（現 電通大），加賀美さんの先輩方には，プログラミングやハードウェアの知識，研究のしかたなどを飲み込みの悪い筆者に丁寧に教えて頂き，また様々なアドバイスを頂き，大変感謝しております．

同期の松島君，五十嵐君，石川君，滝口君とは，学部時代から修士課程までの 3 年間，一緒に楽しい研究室生活を送ることが出来ました．松島君とは同じ情報工学専攻，名簿順でも隣ということで，情報輪講などでは苦楽をともにしました．旭川で一般市民を前に一緒にデモをしたのもよい思いでです．また，松島君の優れたプログラミングセンスからは筆者も多くのことを学びました．五十嵐君とは，高校時代からの友人でずいぶんと長いつき合いになりました．まさか研究室まで一緒になるとは想像もしていませんでした．運動神経抜群の五十嵐君は JSC の中核としてずいぶんと活躍してくれました．何においても五十嵐君なら何とかするだろうという安心感がありました．また，4 年時に同期に研究室に入った斎藤君は，イギリスの Sussex 大学の修士課程を卒業し帰国してからというもの，隠れ 9 号館メンバとして活躍してくれました．斎藤君の linux に関する知識のおかげで，トラッキングビジョンのドライバを linux に移植することができ，実験効率が格段に上がりました．

金広君，長嶋君には，筆者が 9 号館にこもっている間，2 号館の学生の面倒をみてくれました．また，そのほか井上研の後輩である竹田君，二宮君，西村さん，長阪君，佐藤君，吉池君，星野さん，田宮君，香山君をはじめ，多くの方々に研究に対する助言を頂いたり，研究以外の活動で一緒に楽しんだりさせてもらいました．

このような素晴らしい研究室で学部 4 年からはじまり，修士課程 2 年間，博士課程 3 年間の合計 6 年間の研究を行ない，ここに博士論文を完成させることができたことを大変誇りに思います．

最後に，筆者をあたたかく見守ってくれた家族，友人の方々にも心から感謝します．

参考文献

[1] 吉田健一. 移動ロボット用視覚. 日本ロボット学会誌, Vol. 1, No. 4, pp. 298–303, 1983.

[2] S.Suzuki, J.Iijima, and S.Yuta. Design and Implementation of an Architecture of Autonomous Mobile Robot for Experimental Researches. In *Proc. of ICAR'93*, pp. 423–428, 1993.

[3] 谷内田正彦. 視覚を用いた3次元位置・形状の計測とその応用. システムと制御, Vol. 29, No. 10, 10 1985.

[4] 越川. 反射偏光を利用する面方向検出法. 計測自動制御学会論文集, Vol. 18, No. 10, pp. 1031–1033, 1982.

[5] R.B.Lewis and A.R.Johnston. A Scanning Laser Range Finder for a Robotic Vision. In *Proc. of the 5th IJCAI*, pp. 762–768, 1977.

[6] D. Marr. *Vision*. Freeman, 1982.

[7] 谷内田正彦. ロボットビジョンの概要と今後の展望. 日本ロボット学会誌, Vol. 10, No. 2, pp. 140–145, 1992.

[8] W.E.L.Grimson. *From Images to Surfaces*. MIT Press, 1981.

[9] 太田. 走査線間の整合性を考慮した動的計画法によるステレオ対応探索. 情報処理学会研究報告（コンピュータビジョン）29-7, 1984.

[10] N.Ayache and F.Lustman. Fast and reliable passive trinocular stereo vision. In *Proc. of 1st Int. Conf. Computer Vision*, pp. 422–426, 1987.

[11] 北村喜文, 谷内田正彦. 三眼視による三次元情報の計測. 日本ロボット学会誌, Vol. 5, No. 2, pp. 131–138, 1987.

[12] W.E.L.Grimson. Computer Experiments with Feature-Based Stereo Algorithm. *IEEE Trans. PAMI*, Vol. 7, No. 1, pp. 17–34, 1985.

[13] 金出武雄, 蚊野浩, 木村茂, 川村英二, 吉田収志, 織田和夫. ビデオレートステレオマシンの開発. 日本ロボット学会誌, Vol. 15, No. 2, pp. 261–267, 1997.

[14] J.Y.Zheng and S.Tsuji. Panoramic Representation for Route Recognition by Mobile Robot. *International Journal of Computer Vision*, Vol. 9, No. 1, pp. 55–76, 1992.

[15] K.Yamazawa, Y.Yagi, and M.Yachida. Obstacle Detection with Omnidirectional Image Sensor HyperOmni Vision. In *Proc. of IEEE Int. Conf. on Robotics and Automation*, pp. 1062–1067, 1995.

[16] 松本勉, 油田信一. 経路地図に従った移動ロボットの自律走行システム. 日本ロボット学会誌, Vol. 5, No. 5, pp. 19–27, 1987.

[17] 前山祥一, 大矢晃久, 油田信一. 街路樹をランドマークとしたポジショニングによる移動ロボットの屋外走行実験. ロボティクス・メカトロニクス講演会'94 講演論文集, pp. 305–308, 6 1994.

[18] S. Tsuji and J. Y. Zheng. Visual Path Planning by a Mobile Robot. In *Proc. the 10th IJCAI*, pp. 1127–1130, 1987.

[19] 浅田稔. センサ情報の統合と理解による移動ロボットのための世界モデルの構築. 日本ロボット学会誌, Vol. 8, No. 2, pp. 28–38, 1990.

[20] 石黒浩, パトリックステルマジック, 辻三郎. 注視制御による局所地図の獲得. 電子情報通信学会論文誌, Vol. J74-D-II, No. 7, pp. 926–932, 1991.

[21] Charles E. Thorpe, editor. *Vision and Navigation : The Carnegie Mellon Navlab*. Kluwer Academic Publishers, 1990.

[22] T.Tsubouchi and Shin'ichi YUTA. Map Assisted Vision System of Mobile Robots for Reckoning in a Building Environment. In *Proc.IEEE int. Conf. on Robotics and Automation*, pp. 1978–1984, 1987.

[23] Claude Fennema, Allen Hanson, Edward Riseman, J.Ross Beveridge, and Rakesh Kumar. Model-Directed Mobile Robot Navigation. *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. 20, No. 6, pp. 1352–1369, Nov/Dec 1990.

[24] Akio KOSAKA and Avinash C. KAK. Fast Vision-Guided Mobile Robot Navigation Using Model-Based Reasoning and Prediction of Uncertainties. *CVGIP: IMAGE UNDERSTANDING*, Vol. 56, No. 9, pp. 271–329, 1992.

[25] M. Meng and A. C. Kak. NEURO-NAV:A Neural Network Based Architecture For Vision-Guided Mobile Robot Navigation Using Non-Metrical Models of the Environment. In *Proc. of Int'l Symp. on Robotics Research(ISIR'93)*, pp. 750–757, 1993.

[26] M.J.Mataric. Environment Learning Using a Distributed Representation. In *Proc. of IEEE Int. Conf. on Robotics and Automation*, pp. 402–406, 1990.

[27] D.A.Pomerleau, J.Gowdy, and C.E.Thorpe. Combining Artificial Neural Networks and Symbolic Processing for Autonomous Robot Guidance. *Engng Applic. Artif. Intell.*, Vol. 4, No. 4, pp. 279–285, 1991.

[28] 小野口一則, 渡辺睦, 岡本恭一, 久野義徳. 移動視覚のための多重情報地図. 日本ロボット学会誌, Vol. 11, No. 3, pp. 401–409, April 1993.

[29] 柴田智広, 松本吉央, 稲葉雅幸, 井上博允. 人が搭乗しその場その場で行動を指示できるパーソナル視覚移動ロボット Hyper Scooter の開発. 日本ロボット学会誌, Vol. 14, No. 8, pp. 1138–1144, 1996.

[30] 黒須, 大矢, 油田. 画像による移動ロボットのポジショニングと誘導. ロボティクス・メカトロニクス講演会'94 講演論文集, pp. 1178–1183, 1994.

[31] 三河正彦, 升谷保博, 丸典明, 宮崎文夫. ビジュアルサーボを用いた自律移動ロボットの制御. 日本ロボット学会第 11 回学術講演会 予稿集, pp. 1309–1310, 11 1993.

[32] J.Y.Zheng, M.Barth, and S.Tsuji. Autonomous Landmark Selection for Router Recognition by A Mobile Robot. In *Proc. 1991 IEEE Int'l Conf. Robotics and Automation*, pp. 2004–2009, 1991.

[33] H.Ishiguro, T.Miyashita, and S.Tsuji. T-Net for Navigating a Vision-Guided Robot in a Real World. In *Proc. of IEEE Int. Conf. on Robotics and Automation*, pp. 1068–1073, 1995.

[34] D.A.Pomerleau. ALVINN: An Autonomous Land Vehicle in a Neural Network. Technical Report CMU-CS-89-107, CMU, 1989.

[35] C.Furlanello, B.Caprile, and L.Stringa. A memory based approach to navigation. *Biological Cybernetics*, Vol. 69, pp. 385–393, 1993.

[36] 北野宏明. 超並列人工知能. 人工知能学会誌, Vol. 7, No. 2, pp. 224–262, 1992.

[37] C. Stanfill and D. Waltz. Toward Memory-Based Reasoning. *Communications of the ACM*, Vol. 29, No. 12, pp. 1213–1228, 1986.

[38] B.V Dasarathy. *Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques*. IEEE Computer Society Press, 1991.

[39] 村瀬洋. 古くて新しい画像認識法 - 固有空間による画像認識 -. 情報処理, Vol. 38, No. 1, pp. 54–60, 1997.

[40] M.A. Turk and A.P. Pentland. Face Recognition Using Eigenfaces. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 586–591, 1991.

[41] Shree K.Navar, Hiroshi Murase, and Sameer A.Nene. Learining, Positioning, and Tracking Visual Apperance. In *Proc. of IEEE Int'l Conf. on Robotics and Automation*, pp. 3237–3244, 1996.

[42] 前田武志, 石黒浩, 辻三郎. 全方位画像を用いた記憶に基づく未知環境の探索. 電子情報通信学会技術研究報告. PRU92-10, pp. 73–80, 1995.

[43] 前田佐嘉志, 久野義徳, 白井良明. 固有空間解析に基づく移動ロボットの位置認識. 電子情報通信学会論文誌, Vol. J80-D-II, No. 6, pp. 1502–1511, 1997.

[44] I. Horswill. Polly: A Vision-Based Artificial Agent. In *Proc. of Int'l Conf. on AAAI '93*, pp. 824–829, 1993.

[45] 井上博允, 稲葉雅幸, 森武俊, 立川哲也. 局所相関演算に基づく実時間ビジョンシステムの開発. 日本ロボット学会誌, Vol. 13, No. 1, pp. 134–140, 1995.

[46] J.Dias, C.Paredes I.Fonseca, H.Araujo, J.Batista, and A.de Almeida. Simulating Pursuit with Machines : Experiments with Robot and Artificial Vision. In *Proc. of IEEE Int. Conf. on Robotics and Automation*, pp. 472–477, 1995.

[47] 大久保厚志, 田中麻紀, 丸典明, 宮崎文夫. アクティブなステレオ視覚システムによる移動物体の追跡. 第5回ロボットシンポジウム予稿集, pp. 11–16, 5 1995.

[48] M. Inaba, Hara T., and H. Inoue. A Stereo Viewer Based on a Single Camera with View-Control Mechanisms. In *Proc. of Int'l Conf. on Intelligent Robots and Systems*, pp. 1857–1864, 1993.

[49] K.Hosoda, K.Sakamoto, and M.Asada. Trajectory Generation for Obstacle Avoidance of Uncalibrated Stereo Visual Servoing without 3D Reconstruction. In *Proc. of Int'l Conf. on Intelligent Robots and Systems (IROS)*, pp. 29–34, 1995.

[50] 溝口博, 上瀬訪吉克, 森武俊, 佐藤知正. 行動処理のためのマルチカメラビジョンシステム, 第13回日本ロボット学会学術講演会予稿集, pp. 247–248, 11 1995.

[51] Y. Matsutmoto, T. Shibata, K. Sakai, M. Inaba, and H. Inoue. Real-time Color Stereo Vision System
for a Mobile Robot based on Field Multiplexing. In *Proc. of IEEE Int. Conf. on Robotics and Automation*, pp. 1934–1939, 1997.

[52] 森武俊, 松本吉央, 稲葉雅幸, 井上博允. 相関値分布の分類に基づく追跡注視点生成. 日本機械学会ロボティクス・メカトロニクス講演会'95 講演論文集, pp. 1076–1079, 1995.

[53] 武藤伸洋, 他. 特集「研究開発プラットフォーム」. 日本ロボット学会誌, Vol. 14, No. 1, 1996.

[54] 油田信一. 移動ロボット研究のためのプラットフォーム. 日本ロボット学会誌, Vol. 14, No. 1, pp. 14–17, 1996.

[55] 広瀬茂男, 有川敬輔. 普及型歩行ロボット TITAN-VIII の開発. ロボティクス・メカトロニクス '96 講演会講演論文集, pp. 275–278, 1996.

[56] 増田良介. 標準マニピュレータ構想 - ロボット制御理論の実用化のために -. 日本ロボット学会誌, Vol. 14, No. 1, pp. 6–9, 1996.

[57] 小笠原司. ロボット研究プラットフォームにおけるソフトウェア. 日本ロボット学会誌, Vol. 14, No. 1, pp. 2–5, 1996.

[58] 井上博允. ハイパーマシン：実用化を指向する知能ロボットのカーネル. 日本ロボット学会第10回学術講演会予稿集, pp. 699–701, 1992.

[59] 立川哲也他. 高速相関演算機能を持つビジョンシステム（第1報：ハードウェア）. 日本ロボット学会第9回学術講演会予稿集, pp. 839–840, 1991.

[60] 柴田智広, 松本吉央, 稲葉雅幸, 井上博允. ハイパーマシンによる車の自動運転. 日本ロボット学会第10回学術講演会予稿集, pp. 705–706, 1992.

[61] 柴田智広, 稲葉雅幸, 井上博允. ハイパーマシンの視覚処理：オプティカルフローの生成と前走車の追跡. 日本ロボット学会第10回学術講演会予稿集, pp. 703–704, 1992.

[62] 森田俊彦, 沢崎直之, 内山隆, 佐藤雅彦. カラートラッキングビジョン. 第14回日本ロボット学会学術講演会予稿集, pp. 279–280, 11 1996.

[63] 福島 E. 文彦, 妻木俊道, 広瀬茂男. PWM 制御方式 DC サーボモータ駆動回路の開発. 第13回日本ロボット学会学術講演会予稿集, pp. 1153–1154, 1995.

[64] 原口聡, 加賀美聡, 桃沢光隆, 近野敦, 稲葉雅幸, 井上博允. 脚型ロボット研究用プラットフォーム J.Rob-1 のハードウエア構成. 第15回日本ロボット学会学術講演会予稿集, 1997.

[65] 小島浩, 関進, 高橋勝彦, 岡隆一. 移動ロボットからの動画像による走行シーンの同定. 第12回日本ロボット学会学術講演会予稿集, pp. 709–710, 11 1994.

[66] 上坂吉則, 尾関和彦. パターン認識と学習のアルゴリズム. 文一総合出版, 1990.

[67] 松本吉央, 稲葉雅幸, 井上博允. 視野画像列の記憶に基づく移動ロボットのナビゲーション — 廊下環境における位置・姿勢および障害物の検出 —. 第13回日本ロボット学会学術講演会予稿集, pp. 313–314, 11 1995.

[68] 住友金属工業株式会社. IP9015(Sketch) 画像平均縮小 LSI. 画像処理 LSI/ モジュールデータブック, pp. 153–176. 住友金属工業株式会社, 1994.

[69] Yoshio Matsumoto, Masayuki Inaba, and Hirochika Inoue. Memory-Based Navigation using Omni-View Sequence. In *Proc. of Int. Conf. on Field and Service Robotics*, pp. 184–191, 1997.

[70] NODA Itsuki. Soccer Server: a simulator for RoboCup. In *JSAI AI-Symposium 95: Special Session on RoboCup*, 1995.

[71] 宮崎猛, 柴田智広, 松本吉央, 稲葉雅幸, 井上博允. 仮想ビュー生成器を用いた視覚移動ロボットのシミュレーション環境"View Simulation System" の構築. 第14回日本ロボット学会学術講演会予稿集, pp. 575–576, 11 1996.

[72] Demetri Terzopoulos, Tamer Rabie, and Radek Grzeszczuk. Perception and learning in artificial animals. In *ALife V Proceedings*, pp. 313–320, 1996.

[73] 岡田浩之, 関口実, 渡部信雄, 松尾和洋. 移動ロボットのための仮想環境システム. ロボティクス・メカトロニクス講演会'96 講演論文集, pp. 754–755, 6 1996.

[74] H.Y.Shum, K.Ikeuchi, and R.Reddy. Virtual reality modeling from a sequence of range images. In *Proc. of IEEE Int'l Conf. on Intelligent Robotics and Systems*, pp. 703–710, 1994.

[75] 松本吉央, 宮崎猛, 稲葉雅幸, 井上博允. 視野画像列の記憶に基づくナビゲーション – View Simulation System を用いた画像列の生成 –. 第14回日本ロボット学会学術講演会予稿集, pp. 291–292, 11 1996.

[76] 廣瀬通孝, 宮田亮介, 谷川智洋. 二次元実画像を用いた三次元仮想世界の構築. ロボティクス・メカトロニクス講演会'96 講演論文集, pp. 1047–1050, 6 1996.

[77] 松本吉央, 宮崎猛, 稲葉雅幸, 井上博允. 画像列に基づくナビゲーション –view simulation system を用いたオンライン経路生成 –. 日本機械学会ロボティクスメカトロニクス講演会'97, 1997.

[78] 喜多伸之, S.Rougeaux, 國吉康夫, 坂根茂幸. 仮想ホロプタを用いた実時間両眼追跡. 日本ロボット学会誌, Vol. 13, No. 5, pp. 101–108, 1995.

[79] 菅井賢, 堀洋一, 小笠原司, 築根秀男. ロボットマニピュレータを用いた構造化トラッキングによる能動視覚システム. 日本ロボット学会誌, Vol. 13, No. 3, pp. 411–419, 1995.

[80] 松本吉央, 柴田智広, 桑原太地, 稲葉雅幸, 井上博允. ハイパースクーターの研究：実画像列の記憶に基づく走行制御. 第12回日本ロボット学会学術講演会予稿集, pp. 1143–1144, 11 1994.

[81] 松本吉央, 稲葉雅幸, 井上博允. 視野画像列の記憶に基づく移動ロボットのナビゲーション：カラー画像を利用したマッチング. ロボティクス・メカトロニクス講演会'96 講演論文集, pp. 1205–1206, 6 1996.