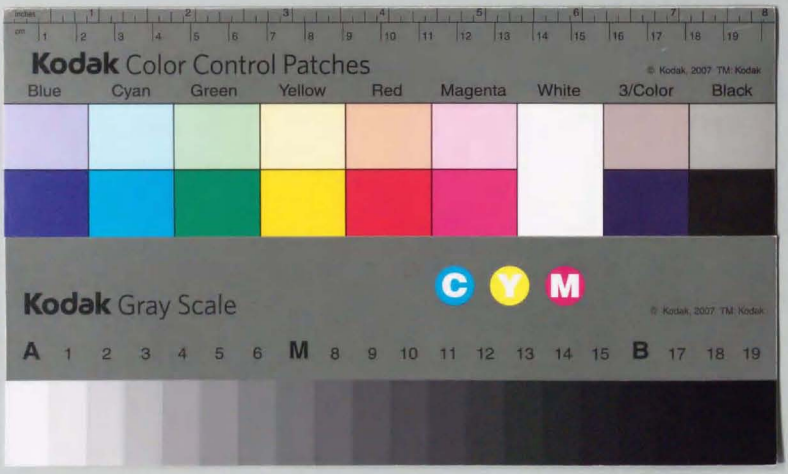

自律システムのための状況認識と
行動規則の同時学習
博士論文


Kodak Color Control Patches
© Kodak, 2007 TM: Kodak
Blue
Cyan
Green
Yellow
Red
Magenta
White
3/Color
Black
Kodak Gray Scale
© Kodak, 2007 TM: Kodak
A 1 2 3 4 5 6 M 8 9 10 11 12 13 14 15 B 17 18 19

1997 年度 学位請求論文

自律システムのための状況認識と行動規則の同時学習

上野 敦志

東京大学 大学院 工学系研究科 航空宇宙工学専攻

本論文の要旨

実世界中で，あるタスクを自律的に実行する認知行動システム（エージェント）を考える．エージェントは各瞬間に，センサからの知覚入力に基づいて，自らの行動を決定する．知覚入力には，通常，多様な情報が含まれていて，あまり重要でない情報もたくさん含まれている．エージェントの能力は実世界中の情報を全て扱うにはあまりにも限定されているので，情報処理の早い段階で不必要な情報を捨て去って必要な情報だけを抽出する処理が必要である．これは，抽象化と呼ばれる処理である．抽象化の利点としては，情報の圧縮や経験の一般化などが挙げられる．

しかし，システムとタスクにとって適切な抽象化の方法を定めるのは，非常に困難である．そして，抽象化が不適切であると，様々な問題が生じる．抽象化のレベルが低過ぎると，システムが不必要な情報を処理するための余分なコストが生じる．また逆に，抽象化のレベルが高過ぎたり，知覚入力の誤った特徴に注目してしまうと，タスクの実行のために必要な情報が抜け落ちて，タスクの遂行に度々失敗する．これらの問題は，限定された情報処理能力しかないシステムがその能力をはるかに上回る複雑性を持つ情報を扱おうとすることによって起こる問題であり，フレーム問題と呼ばれている．

人間もまた，環境の複雑性に比べれば，限られた情報処理能力しか持たないので，フレーム問題を解決することはできない．しかし，日常生活においては，フレーム問題にほとんど悩まされずに行動している．それは，日常的に環境中の膨大な情報のごく一部だけに注目する習慣を身につけているからである．また，フレーム問題によって判断を誤った場合にも，その誤りによる被害を最小限に抑える術を心得ているからである．これは，「フレーム問題の現実的な解決」と呼ばれている．人工的な認知行動システムにも，人間の持つ次の三つの機能を持たせることによって，フレーム問題の現実的な解決に近付けることができると考えられる．

1. 環境を頻繁に参照する．

2. 情報処理を並列化する．

3. 膨大な情報の中から注目すべき情報を取り出す方法を学習する．

1, 2の機能によって，システムは不慣れな環境でも，大きく破綻することなく行動することができる．そして3の機能によって，不慣れな環境に徐々に慣れていくことができる．本研究では，この3の機能に注目して，「抽象化の学習」（または「認識の学習」）を行うシステムを開発した．

抽象化の学習は，抽象化された表象の上での情報処理よりも，かなり大きなタイムスケールで行われるものである．そのため，従来のほとんどの人工知能システムでは，記号化などの抽象化のプロセスは固定してしまって，その上での情報処理を扱ってきた．その方が効率的であり，うまく記号体系を設計すれば，十分に実用的なシステムになるからである．しかし，実世界中の認知行動システムの場合には，環境の複雑性のために，この設計が非常に困難になる．また，真に自律性が求められているのは，惑星探査ローバーなどの人間の助力が困難なタスクにおいてである．そのようなタスクにおいては，通信が困難であるために自律性が必要とされ，さらに環境の事前の調査が難しいこと，故障した場合も修理できないことなどから，情報処理の大きな柔軟性が求められている．抽象化の学習という情報処理の低いレベルにおける学習をゆっくりと行いながら，同時に抽出された表象の上での学習を行うことによって，大きな柔軟性を持つことが期待できる．

本研究で開発した状況遷移ネットワークシステム（Situation Transition Network System，STNS）では，抽象化として，連続的な知覚入力から記号の抽出を行う．抽象化の手段として記号化を選んだのは，記号表現には強力な情報処理手法が存在するからである．また，将来的に他者と知識を共有することを考えても，記号表現を用いるのは望ましい．STNS は，知覚入力空間中の特定の領域，つまり「状況」を切り出して，その「状況」に対して記号を割り当てる．すなわち「状況」に相当する記号の意味を学習する．

このシステムが従来の認識の学習と大きく異なるのは，各状況の意味が，教師によって与えられるのではなく，行動の結果の類似性によって定義される点である．類似性の基準は環境から与えられる報酬とする．すなわち，一つには，ある特定の行動によって大きな報酬が得られる領域を状況として切り出す．また，もう一つには，ある特定の行動によって既存の状況に遷移する領域を状況として切り出す．このように行動の結果の類似性に注目することによって，エージェント自身の環境中での経験のみに基づいた記号の定義が可能になる．

STNS において，タスクを遂行する行動は，切り出された状況間でプランニングを行うことによって決定される．また，切り出された状況の形状は，その後のタスクを遂行する行動によって，常に調整され続ける．つまり，状況の形状が適切であれば行動はタスクを成功させるし，行動がタスクを成功させ続ければ状況の形状が適切に維持される．この状況と行動の相互依存関係によって，状況の形状が常にタスクにとって適切な抽象化を保つことが期待できる．

STNS の特徴をまとめると次の三つになる．

1. 環境からの報酬に基づいて，知覚入力空間中で状況を統計的手法を用いて切り出し，それを動的に維持する．(状況認識の学習)

2. 状況間の遷移確率とその時得られる報酬を過去の経験から最尤推定してワールドモデルを構築し，その上で部分的プランニングを行って行動を決定する．(行動規則の学習)

3. これらの二つの学習を，環境中でタスクを実行しながら同時に行う．

STNS の性能を調べるために五つの実験を行った．いずれもコンピュータシミュレーションである．2次元入力の簡単なナビゲーション問題では，STNS が，簡単な問題において，理想的な状況認識と行動規則を獲得できることが確かめられた．また，データが不十分な段階でも状況を切り出すことによって，学習が非常に速く進むことが確かめられた．アクチュエータが故障したローバーを用いた2次元入力のナビゲーション問題では，STNS が，超楕円体から離れた形状を持つ状況の認識，小さい状況の認識，他の状況に覆われた状況の認識などの点で弱点を持つことが確かめられた．しかし，問題を徐々に複雑にしていった時の性能の悪化は緩やかであることが確かめられた．2次元入力のナビゲーションの二つの柔軟性テストでは，STNS が，簡単な問題において，環境の小さな変化に柔軟に適応でき，大きな変化にもほぼ安定して対応できることが確かめられた．8次元入力の複雑なナビゲーション問題では，報酬に近い領域では，ある程度の状況認識と行動規則を獲得することができた．しかし，報酬から離れた領域では，状況認識が不十分であるために，適切な行動規則を獲得することができなかった．その原因は，主に，状況の形状を学習する能力の低さであると思われる．ただし，強化学習で最も一般的な入力空間をグリッドで分割する状態表現と比較すると，STNS における状況表現は抽象度が高く，柔軟性においても優れていると考察された．

STNS を実用的なタスクに用いるためには，状況学習能力の低さを補う仕組みが必要である．そのためのアプローチとしては，「階層化」と「属性の生成」の二つが考えられる．「階層化」では，大まかに認識して戦略を決定してからその戦略内で細やかに認識して行動を決定することによって，各レベルの状況の形状を簡単にすることができる．「属性の生成」を用いるアプローチでは，知覚入力空間上ではうまくまとまっていない領域を切り出すために，その領域がうまくまとまるような属性空間を生成して，知覚入力をその空間上にマッピングする．どちらも個々の状況の形状を簡単にすることによって，乏しい状況学習能力で複雑な状況表現を獲得する可能性を高めることができる．

環境と密接に関わりを持つ記号とは，STNS における状況のように環境中での経験に基づいて柔軟に定義されるものだと考えられる．しかし，この「状況」という記号は，まだ非常に初歩的なものである．このように経験に基づいて抽出される記号が，記号処理システム中で用いられるような複雑な記号となるためには，知識の構造化と再利用，物の認識，知覚と行動の時間的な抽象化などを考慮にいれてシステムを拡張していく必要がある．

また，STNS では，上述のフレーム問題を現実的に解決するための三つの機能のうち，3 の適切な認識を学習する機能しか扱っていない．フレーム問題に悩まされない環境に根付いたシステムとするためには，1 の環境を頻繁に参照する機能と，2 の情報処理を並列化する機能も合わせ持つように拡張していく必要がある．

本研究で開発した STNS では，認識の学習と認識された表象の上での強化学習を並列して実行することによって，従来の強化学習システムよりも柔軟な状態表現を実現している．その結果，与えられたタスクに特化しつつ，不慣れな環境に柔軟に適応できる認知行動システムを構成することができた．このシステムは，記号の学習としては知覚入力空間の分割，記号を用いた情報処理としては強化学習しか行わないし，アプリオリな知識はほとんど何も与えることができないので，認知行動システムとしては非常に初歩的なものである．しかし，認知行動システムによる認識の学習は，まだ始まったばかりの研究領域であり，もっと強力な記号体系，もっと複雑な情報処理，そしてアプリオリな知識の融合を目指してこの種の研究を発展させていくことは，実世界中の知能ロボットを実現する上で非常に有用であると思われる．

もくじ

もくじ

図一覧

図一覧

表一覧

表一覧

記号一覧

A	行動空間
B	決定空間
b_i	プラン上の i 番目の行動
b_k	離散時刻 k における決定
D	集団と点の間のマハラノビス（汎）距離
d	二点間のマハラノビス（汎）距離
d_i	プラン上の i 番目の目標状況
l	車輪間の間隔
m	ゴールまでの行動数の平均値
N_{init}^{R}	R 状況の抽出に必要な正事例の最小個数
N_{min}^{R}	R 状況の維持に必要な正事例の最小個数
N_{init}^{T}	T 状況の抽出に必要な正事例の最小個数
N_{min}^{T}	T 状況の維持に必要な正事例の最小個数
n	プランの長さ
n_{max}	IPRL によって前向きに展開されるプランの長さの最大値
n_i^b	状態 i において行動 b を実行した回数 履歴データベース中の状況 i において行動 b を実行したデータの個数

n_{ij}^{b}	状態 i において行動 b を実行した時に状態 j に遷移した回数 履歴データベース中の状況 i において行動 b を実行した時に状況 j に遷移したデータの個数
P	プラン
P	知覚入力空間
p	ゴールまでの行動数の 90% 点
$p(i,j;b)$	状態 i において決定 b を選んだ時の状態 j への遷移確率 状況 i において行動 b を実行した時に状況 j に遷移する確率
p_{min}	IPRL によって前向きに展開されるプランの成功確率の最小値
R_{min}^{T}	T 状況の意味の切り替えに必要な，その状況の意味に適合するデータの数に対する，その他のある意味に適合するデータの数の比の最小値
r	右車輪に対する左車輪の回転角速度の比
$r(i,b)$	状態 i において決定 b を選んだ時の利得 状態 i において行動 b を実行した時に直接得られた報酬の標本平均値
$r(i,j;b)$	状況 i において行動 b を実行して状況 j に遷移した時に得られる直接の報酬の期待値
$r_{\min}^{R}$	R 状況の正事例となるデータの持つ報酬の最小値
$U(i)$	状態 i のユーティリティ
$U_{\pi}(i)$	状態 i から政策 π をとった時の無限期間に渡る総利得の期待値
X	状態空間
x_k	離散時刻 k における状態
γ	報酬（利得）の割引率

μ	集団の平均
π	政策
Σ	集団の分散共分散行列
ω	右車輪の回転角速度

第 1 章

序論

1.1 はじめに

実世界中で，あるタスクを自律的に実行する認知行動システム（エージェントと呼ぶ）を考える．草原で獲物を探すライオン，都市を巡回するセールスマン，オフィスで書類を配達するロボット，惑星上を探査する自律ローバーなどである．エージェントは各瞬間に，センサからの知覚入力に基づいて，自らの行動を決定する．知覚入力には，通常，多様な情報が含まれていて，いくつかの情報は非常に重要であるが，あまり重要でない情報もたくさん含まれている．エージェントの能力は実世界中の情報を全て扱うにはあまりにも限定されているので，情報処理の早い段階で不必要な情報を捨て去って必要な情報だけを抽出する処理が必要である．これは，抽象化と呼ばれる処理である[1]．

抽象化の利点として，次の三つが考えられる．

1. 情報の圧縮
   外界の持つ情報は，エージェント内部に蓄え処理し得る情報よりもはるかに複雑である．外界の情報のうちの現在のタスクに必要な部分だけに注目することによって，エージェント内部で表現し処理することが可能になる．

2. 情報処理の時間の短縮
   1 の情報圧縮によって情報処理の問題空間が非常に小さくなり，エージェントが外界の変化に実時間で対応することが可能になる．

3. 経験の伝播，一般化
   実世界は連続的な空間なので，その空間中でエージェント自身が実際に経験す

[1]環境中の事象をセンサによって知覚入力に変換する際にも抽象化が行われている．しかし，本研究では知覚入力の抽象化を扱う．

る状態は非常にわずかである．その経験から，タスクにとって重要な部分を抜きだして一般化しておくことによって，未経験の状態や経験が不足している状態で適切な行動を決定することが可能になる．

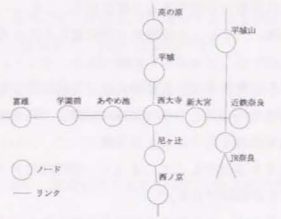
このように，実世界中の自律エージェントの情報処理（推論，プランニング，学習など）にとって，知覚入力の抽象化は欠かせないものである．

しかし，システムとタスクにとって適切な抽象化の方法を定めるのは，非常に困難である．そして，抽象化が不適切であると，様々な問題が生じる．抽象化のレベルが低過ぎると，システムが不必要な情報を処理するための余分なコストが生じる．また逆に，抽象化のレベルが高過ぎたり，知覚入力の誤った特徴に注目してしまうと，タスクの実行のために必要な情報が抜け落ちてしまう．その結果，状態を誤認してタスクの実行に失敗することもある．

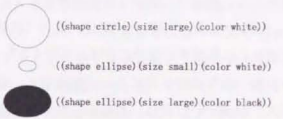
人工知能の領域でもっとも一般的に用いられている抽象化は記号化である．「記号（symbol）」とは，「要素の一つひとつが意味を持つような情報」のことであり [3]，図 1.1に示すように様々な記号表現が用いられている．従来の人工知能システムでは，記号および記号体系はシステムの設計者によって定義される．しかし，複雑なタスクにおいて記号を適切に定義することは非常に困難である（Articulation Problem [39]）．通常，システムがタスクの実行に失敗する事態を避けるために，記号は冗長に定義される．その結果，システムが関係のない情報を関係がないと推論するだけでも非常に時間を要するということが大きな問題として指摘されている[2]．

設計者があらかじめ定めた記号の上での情報処理における問題点を避けるために，パターン表現のままで情報を処理する手法もある．それは主にニューラルネットワークを用いる手法で，コネクショニズム（connectionism）や，parallel distributed processing（PDP）などと呼ばれている．この手法では，入力部のパターンが隠れ部において抽象化されたパターンとなり，そこから出力部のパターンを形成する．ニューラルネットワークによる情報処理は，記号処理に比べて強力な学習能力を持つが，1. 情報処理の仕組みがわかりにくいという問題点がある．そのため，記号処理で用いられている演繹的推論や，プランニングなどの強力な情報処理手法をそのまま用いることはできず，今のところ，記号処理で扱っているような複雑な構造を持つ対象を扱うタスクには適用できていない．またその問題点に関連して，2. 計算能力，学習能力の限界が明らかにされていない，3. 隠れ部のニューロンの数や結線，ニューロン内部の関数の形，ニューロン間の結合の重みの初期値などパラメータがたくさんあり，システムの設計法が確立していない，などの問題点もある．

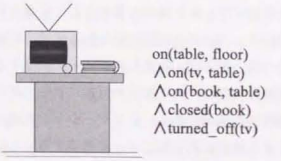
[2]この問題はフレーム問題（frame problem）と呼ばれる問題の一種である．フレーム問題については，第2章で説明する．



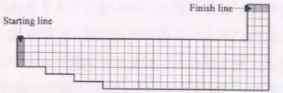
a. グラフ表現（ネットワーク表現）



b. リスト表現（[3] より改変）



c. 述語論理表現



d. 空間分割グリッド表現 [13]

図 1.1: 様々な記号表現

強力な情報処理手法を利用するためにも，また他者と知識を共有するためにも，記号表現を用いることは非常に有効であると思われる．しかし，上述のように，設計者が前もって記号を適切に（フレーム問題などの知覚入力の不適切な抽象化による問題を避けるように）定義することは非常に困難である．そこで，本研究では，記号そのものを学習することを目標とする．具体的には，「連続的な知覚入力空間を，その内部の点が類似した意味を持ついくつかの領域に分割し，各領域に記号を割り当てて，その記号を用いて情報処理を行うこと」を目標とした．このようにエージェント自身の経験に基づいて記号を学習することによって，その時のタスクにとって適切な記号表現が獲得されることが期待できる．

知覚入力空間中の各点は，ある特定の知覚入力すなわち「場面」と呼び得るものに相当する．従って，知覚入力空間中の領域は，特定の「場面」の集合すなわち「状況」と呼び得るものに相当する．本研究で開発した状況遷移ネットワークシステム（Situation Transition Network System，STNS）では，この「状況」に対して記号を割り当てる．すなわち「状況」に相当する記号の意味を学習する[3]．

各状況の意味は，行動の結果の類似性によって定義する．類似性の基準は環境から与えられる報酬とする．報酬というのは，各場面に対して数値で与えられる評価のことである（4.1節参照）．例えば，ナビゲーション問題の場合は，エージェントがゴールに到達した時に大きな報酬が与えられる．各状況は，知覚入力空間中の，同一の行動によって類似した結果が得られる場面の集合として定義される．すなわち，一つには，ある特定の行動によって大きな報酬が得られる領域を状況として切り出す．ナビゲーション問題の場合は，例えば，前進してゴールに到達する領域が状況として切り出される．もう一種類の状況は，ある特定の行動によってそのような報酬の見込みの高い既存の状況に遷移する領域である．ナビゲーション問題の場合は，例えば，先ほどの前進してゴールする状況に右回転によって到達することができる領域が状況として切り出される．このように行動の結果の類似性に注目することによって，エージェント自身の環境中での経験のみに基づいた記号の定義が可能になる．

さらに，切り出された状況の形状は，その後のタスクを遂行する行動によって，常に調整され続ける．また，タスクを遂行する行動は，切り出された状況間でプランニングを行うことによって決定される．つまり，状況の形状が適切であれば行動はタスクを成功させるし，行動がタスクを成功させ続ければ状況の形状が適切に維持される．この状況と行動の相互依存関係によって，状況の形状が常にタスクにとって適切な抽象化を保つことが期待できる．

[3]「物」に相当する記号の意味を学習するためには，各場面の中をさらに分割する必要がある．これについては7.2節で考察する．

STNSの特徴をまとめると次の三つになる．

1. 環境からの報酬に基づいて，知覚入力空間中で状況を統計的手法を用いて切り出し，それを動的に維持する．（状況認識の学習）
2. 状況間の遷移確率とその時得られる報酬を過去の経験から最尤推定してワールドモデルを構築し，その上で部分的プランニングを行って行動を決定する．（行動規則の学習）
3. これらの二つの学習を，環境中でタスクを実行しながら同時に行う．

このように記号の定義を連続的な空間中で学習する研究は非常に基礎的なものであり，人工的な認知行動システムが自らの経験に基づいて自らが用いる記号の定義を学習する研究はあまり行われてこなかった[4]．通常は，抽象化の学習は，抽象化された表象の上での情報処理よりも，かなり大きなタイムスケールで行われるものである．そのため，従来のほとんどの人工知能システムでは，記号化などの抽象化のプロセスは固定してしまって，その上での情報処理を扱ってきた．その方が効率的であり，うまく記号体系を設計すれば，十分に実用的なシステムになるからである．しかし，実世界中の認知行動システムに関しては，上述のように，この設計の困難さが指摘されている．また，真に自律性が求められているのは，惑星探査ローバーなどの人間の助力が困難なタスクにおいてである．そのようなタスクにおいては，通信が困難である[5]ために自律性が必要とされ，さらに環境の事前の調査が難しいこと，故障した場合も修理できないことなどから，情報処理の大きな柔軟性が求められている．抽象化の学習という情報処理の低いレベルにおける学習をゆっくりと行いながら，同時に抽出された表象の上での学習を行うことによって，大きな柔軟性を持つことが期待できる．本研究は，その可能性を探るための基礎研究であると位置付けられる．

1.2 本論文の構成

以下，まず第2章では，実世界中の自律エージェントである知能ロボットの研究について概観する．その中で状況認識の学習が重要であることを指摘する．第3章では，自律エージェントによる状況認識の学習を念頭において，そのために用いることのできる様々な分類法を紹介する．第4章では，まず機械学習の一種である強化学習につ

[4]機械学習の一種，強化学習の分野で扱われている「入力の一般化問題」で，同様のアプローチによる解法がいくつか研究されている．4.4節参照．

[5]火星上のマーズ・パスファインダーと地球との通信は，往復で約20分かかっていた．

いて簡単に説明し，認知行動システムによる状況認識の学習として強化学習における知覚入力の抽象化の学習を説明する．第5章では，環境中での経験に基づいて状況認識と行動規則を同時に学習する新しい機械学習法として「状況遷移ネットワークシステム（STNS）」を提案し，その説明をする．第6章では，STNSを用いた実験について述べ，考察を行う．第7章では，状況認識の学習を行う認知行動システムの将来の実用化に向けた展望と，環境に根付いた記号処理システムの実現に向けた展望を述べる．第8章で結ぶ．

第 2 章

知能ロボット

本研究で扱う自律エージェントは，環境から情報を採り入れ，内部で処理して，再び環境中に行動として出力するというフィードバックループを構成する．すなわち，認知行動システムである．実世界を環境としてその中で複雑なタスクを実行する認知行動システムは，知能ロボットと呼ぶことができる．

知能ロボットの研究は，人工知能の研究の初期の段階，1960年代の後半から始まった．人工知能は人間の知能をコンピュータ上で実現することを目指すものなので，人工知能に感覚器と手足を持たせるというのはごく自然な拡張である．その後，知能ロボットの研究は，その要素となる技術に対して，コンピュータビジョン，プランニング，マニピュレーションなどの様々な研究分野を生み出しながら発展してきた．個々の要素技術は進歩し，工業用ロボットなどで実用化されるレベルに達している．しかし，それらを統合した十分に知能的であるといえるロボットは今だ実現していない．

この章では，まず，昔から主流であった記号系仮説に基づくロボットについて説明し，そこで発生するフレーム問題を解決不能な問題として説明する．次に，近年注目されている物理基盤仮説に基づくロボットについて説明し，この種のロボットとフレーム問題の関連について述べる．最後に，フレーム問題を避ける手段として，抽象化の学習という情報処理の低いレベルでの学習が有効であることを説明する．

2.1 記号系仮説に基づくロボット

記号系仮説（symbol system hypothesis）とは，「記号系は知的なふるまいをするための必要で十分な道具である」[49]という言明である．従来のAIは，この仮説に基づいて，「与えられた問題とそれを解くための知識を記号としてコンピュータの中に表現し，その記号を操作することで問題の解答を得る，という形をとって」きた[69]．知能ロボットの情報処理も，この仮説に基づいて，環境中の情報をシステム内部の記

号で表象する感覚系，内部記号の操作によって推論する思考系，内部記号の出力にしたがってアクチュエータを動かす行動系を直列的に組み合わせることによって実現できると考えられてきた（図2.1参照）．

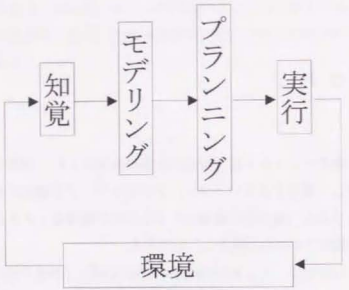


図 2.1: 直列型ロボット

ところが，あらかじめ定義された記号系を用いて，この仮説に基づいて知能ロボットを構築しても，実世界中で実用的に動くシステムは実現できなかった．1. 直列的なシステムでは，フィードバックループが大きくて，刺激を受けてから反応するまでのタイムラグが大きい．2. センサ，アクチュエータの誤差が避けられない．3. 実世界を完全に記述し推論することは時間的にも記憶容量的にも不可能である，などの原因で，システム内のモデルが現実から乖離してしまうのである．

これらの問題は，計算機の速度や，センサ，アクチュエータの精度などの個々の要素技術を改良していけばそのうち解消するような簡単な問題ではなくて，もっと本質的な問題である．なぜなら，計算機の速度がどんなに速くなって，センサ，アクチュエータの精度がどんなに上がっても，実世界中の情報を全てシステム内で表現することは，システムがその世界に含まれている限り，原理的に不可能であるからである．

現実的には，環境中の得られる限りの情報を記号で記述して，持っている限りの知識を用いて推論を行うことでさえ不可能である．記述の量，推論の分岐が膨大で，情報処理能力を軽く凌駕してしまうからである．そこで，現在のタスクに必要な情報，知識だけを枠で囲う必要がある．これは，序論で述べた抽象化と呼ばれる情報処理である．この枠を明確に規定することは，実は非常に困難な問題である．なぜなら，実

世界には非常に多くの事象が複雑に絡み合って存在しており，必要ないと切り捨てたことから，カオティックに重大な影響が生じることがあるからである．一方，十分な情報と知識を与えて派生する影響を十分に推論しようとすると，考えてばかりで少しも動かないシステムになってしまう．このことは，Dennett の書いた有名な論文 [25] の中で，ロボットたちのディレンマとして非常に端的に表現されている．それを要約すると次のようになる．

> 昔，R_1 と名付けられたロボットがあった．ある日，R_1 の予備バッテリーがしまってある部屋に次元爆弾が仕掛けられた．バッテリーはワゴンの上においてあったので，彼はワゴンを部屋から引き出せばバッテリーを部屋から救出できるだろうと考えた．そこで彼は直ちにそれを実行し，爆弾が爆発する前にバッテリーを運び出すことに成功した．ところが，不幸なことに爆弾もワゴンの上にあったので，バッテリーは爆発で破壊されてしまった．R_1 は爆弾がワゴンの上にあることを知っていたが，ワゴンを引っ張り出すことが爆弾も一緒に運び出すことになるということに気付かなかったのである．
>
> 設計者たちは，R_1 の反省から，次のロボット R_1D_1 には，自分の行動の帰結として，自分の意図したものだけではなく，副産物についての帰結も演繹できる機能を持たせた．ところが R_1D_1 も同じ苦境において失敗する．R_1D_1 は，直ちにワゴンを引き出すことに気がつくが，ワゴンを引っ張り出しても部屋の壁の色が変わらないだろうということを演繹し，つぎに，ワゴンを引けば車輪が回転するだろうという帰結の証明を行っている間に爆弾が爆発してしまったのである．
>
> 設計者たちは，次のロボット R_2D_1 には，関係のある帰結と関係のない帰結を区別し，関係のない帰結は無視する機能を持たせた．R_2D_1 を同じテストにかけてみたところ，彼はじっとうずくまったままであった．R_2D_1 は，無関係な帰結を探し出してそれを無視するのに懸命だったのである．そしてまたもや爆弾は爆発してしまった．

この問題は人工知能の分野で発見された問題で，フレーム問題（frame problem）と呼ばれている．このフレーム問題によって，現実とモデルの乖離は避けることができない．記号処理仮説に基づくシステムは，世界モデルを中心にして全ての行動をその上での推論で決定するので，この乖離の影響を強く受けるシステムであるといえる．

2.2 フレーム問題

ここで，フレーム問題の定義とその性質について説明する．フレーム問題には様々な見方がある．1969年にMcCarthyとHayesによって初めて指摘された時には，ある行動によって変化しない事象を記述しようとすると，記述の量が爆発してしまうという問題であった．彼らが用いた例は，電話帳で電話番号を調べて電話をかける，というものである．単調な記号論理でこの行為をモデル化しようとすると，「ある人が電話を持っていれば，電話帳で電話番号を調べてもまだ電話を持っている」とか，「電話帳で電話番号を調べても，電話帳の内容は変わらない」とかいう，人間にとっては自明の条件が山ほど必要になり，記述の量が簡単に爆発してしまう[33]．この問題は，環境中の膨大な情報を限られた容量のシステム内に取り込むことによる情報の部分性に起因している．すなわち，電話をかけることによる影響は，実環境中では様々に派生する可能性があるが，その全てを書き切ることも，反対に影響を受けない事象をすべて書き切ることも不可能なのである．

情報の部分性には，上述のような記述の部分性のほかに，処理の部分性もある．これは，システム内に記述されている情報の一部が，時間的な制約によって今だ処理されていないことによる情報の部分性である．この二つはシステム内部を完全に調べてみれば区別できるが，システム自体が情報処理を行っている最中に区別することはできない．そこで松原らは，これらの情報の部分性によって生ずる問題をひっくるめてフレーム問題と呼ぶことを提案している．すなわち，フレーム問題を，「限定された情報処理能力しかないシステムがその能力をはるかに上回る複雑性を持つ情報をどのように扱うかという問題」であると定義している[70]．また，この意味でのフレーム問題を，記述の部分性に関する狭い意味でのフレーム問題と区別する場合には，「一般化フレーム問題（generalized frame problem）」と呼んでいる[33, 68]．

この定義において，フレーム問題は解決不能である．フレーム問題は，本質的には解決不能な問題を，どのようにうまく扱うかという問題だからである．人間もまた，実世界の複雑性に比べると乏しい情報処理能力しか持っていないので，フレーム問題を完全に解くことはできない．いくつかの例が松原らによって示されているが，そのうち「この置き方は誰」という遊びの例を紹介する[68]．

「この置き方は誰」

何人かいるところで，まず出題者は鉛筆をいろいろな方向に向けて置いて，「この置き方は○○さんを指している」とか「この置き方は××さんを指している」とか何度か解答者の前でやってみせる．次に出題者は「この置き方は誰を指しているか？」と言ってから，鉛筆をある方向に向けて置く．

（ひっかかった）解答者は，どの置き方が誰を指しているかさっぱりわからず，しばらくからかわれ続ける．出題者の鉛筆を置く方向は誰を指すかとはまったく関係なく，鉛筆を置き終えてから最初に声を出した人を指しているのである．からかわれている人は，どの情報に注目すればよいかがわからないために，ひどい目に合う．

この例において，解答者には，必要な情報が与えられているのだが，与えられる情報があまりにも膨大であるために，その中のどの部分に注目したら良いのかがわからない．しかし，フレーム問題の解決不能性にも関わらず，日常生活においては，人間はフレーム問題にほとんど悩まされずに行動しているように思われる．それは，日常的に環境中の膨大な情報のごく一部だけに注目する習慣を身につけているからである．また，フレーム問題によって判断を誤った場合にも，その誤りによる被害を最小限に抑える術を心得ているからである．松原らは，これを「フレーム問題の現実的な解決」と名付けている[70]．

今のところ，人工的なシステムは，実世界中で行動するということに関しては，人間よりもはるかに劣る情報処理能力しか持っていない．そこで人工的な認知行動システムに，人間の持っている次の三つの機能を持たせることによって，フレーム問題の現実的な解決に近付けることができると考えられる．

1. 環境を頻繁に参照する．
2. 情報処理を並列化する．
3. 膨大な情報の中から注目すべき情報を取り出す方法を学習する．

1は，情報の部分性を認めた上で，不完全なモデルしか持てないのであれば，システム内に情報を貯めこまずに，なるべく新鮮な情報を用いて行動を決定しようというアプローチである．システムがフレーム問題の影響で判断を誤ったとしても，早期に発見して対処することが可能になる．

2の並列処理によって，システムは問題を多角的に捉えることができる．そのため，頻繁に環境を参照する部分とじっくり考える部分を共存させることができる．また，環境とシステムを結ぶフィードバックループが複数構成されるので，システム内のある部分で処理を失敗してもシステム全体が破綻することはなく，失敗の影響は緩やかに現れることになる．

1の即応性と2の並列性を合わせ持つことによって，システムは環境と密に相互作用することができる．これは，システムの「環境への埋め込み（embedding）」と呼ばれている．

3は，フレーム問題が解決不能であることを認めた上で，過去の経験や助言に基づく学習によって，自律的に適切な情報の枠を構成しようというアプローチである．1，2の機能によって，システムは不慣れな環境でも，大きく破綻することなく行動することができる．そして3の機能によって，不慣れな環境に徐々に慣れていくことが可能になる．

2.3 物理基盤仮説に基づくロボット

2.3.1 行動に基づく知能のアプローチ

物理基盤仮説（physical grounding hypothesis）とは，「知能を持ったシステムをつくるには，物理世界に根づいたものでなければならない」[49]という言明である．これは前述の記号系仮説に対して，Brooksによって提唱された考え方である[19]．記号系仮説に基づくアプローチでは専らコンピュータ内での記号処理に専心していたのに対して，物理基盤仮説に基づくアプローチではシステムと環境との相互作用を重視する．

そして，Brooksは知能ロボットのアーキテクチャとして，図2.1のような直列的なアーキテクチャを批判して，図2.2のような並列的なアーキテクチャを提唱した．このアーキテクチャでは，システムをタスクを実行する行動に基づいて分割する．このため，このアプローチは行動に基づく（behavior-based）知能のアプローチとも呼ばれる．各モジュールは必要なセンサとアクチュエータに直結され，システムと環境は複数のフィードバックループで結合される．このために前節のフレーム問題に対処する機能2で述べた情報処理の並列化が実現される．

さらに，このアーキテクチャでは，汎用的で環境をなるべく詳しく表現した環境モデルを作ることよりも，各々のモジュールがそれ自身のタスクを遂行するのに必要十分なだけの情報を処理することに集中する．そして，それぞれのモジュールがそれ自身のタスクに特化した内部モデルを持つ．そのため，情報処理を素早くして，刺激に対する反応を早くすることができる．また，この特化と即応性によって，記号系仮説に基づくシステムの汎用的な環境モデルに比べてフレーム問題による現実とモデルの乖離を小さくすることができる．

Brooksらのロボットでは，もう一つ反応を早くするための手段として，ごく簡単なタスクしか実行しないモジュールが設けられている．例えば，障害物回避や徘徊だけを実現するようなモジュールである[16]．このアーキテクチャでは，ロボットと環境が相互作用し続けることに何よりも重点を置くので，ごく簡単なタスクでもとにかく動き続けることが重要である．もっと複雑なタスクはこれらの低レベルのモジュール

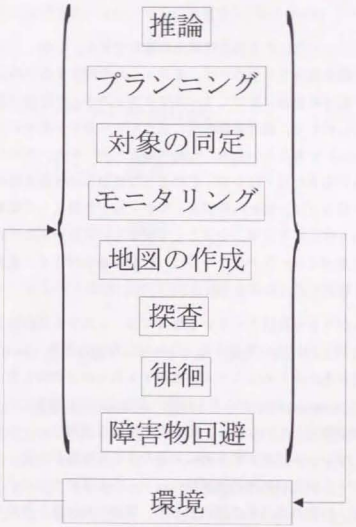


図 2.2: 並列型ロボット

と並列されるモジュールによって処理すれば良いのである．こうして，前節のフレーム問題に対処する機能1で述べた環境に対する即応性が実現される．

問題をいくつかのタスクに分解して各々を異なるモジュールで処理するアーキテクチャは，設計の面でも有利な点がある．個々のモジュールは比較的簡単なタスクを実行すればよいので，必要な情報，知識を枠で囲うのが容易になるという点である．従って，個々のモジュールに関しては，設計者は抽象化の仕様をかなり容易に規定することができる．

並列的なモジュール化による利点は以上の通りである．しかし，システム全体が整合性のとれた行動を実現するためには，モジュールを統合するために中心となるものが必要である．記号系仮説に基づくアーキテクチャの中心が記号で表現された環境のモデルであるのに対して，物理基盤仮説に基づくアーキテクチャの中心は行動の融合（behavior-fusion）であるといえる．行動の融合といっても，そのためのスーパーバイザ的な仕組みがあるわけではない．そのような仕組みは物理基盤仮説に反するからである．行動の融合とは，個々のモジュールがそれぞれ独立して環境（および他のモジュール）と相互作用する結果，全体として望ましい行動が実現されることである．物理基盤仮説に基づくアーキテクチャでは，行動の融合がうまく実現するように個々のモジュールを設計しなければならない．

各モジュールがうまく設計されている場合には，システムは設計者が設計時にあらかじめ想定した以上の行動を実現する．これは，行動の創発（emergence）と呼ばれている．設計者があらかじめシステムが遭遇するあらゆる環境を想定することは不可能なので，この behavior-fusion による行動の創発は非常に重要である．行動の創発によって，設計者が想定していなかったような不慣れな環境でも，システムが適切に情報を処理して，フレーム問題を現実的に回避できる可能性がある．ロボットの行動の複雑性は，ロボット内部の処理の複雑性のみから生じるのではなく，環境の複雑性からも生じるので，行動の創発を念頭におくと，環境の複雑性を複雑なタスクを実現するために利用することもできる．

しかし，ここには大きな壁がある．複雑なタスクに対しては，全体としての整合性が保たれるように各モジュールをうまく設計することが非常に困難なのである．環境からの情報の選択，統合は個々のモジュール内では比較的簡単に設計できるが，その代わり，異なるモジュールで処理されてきた情報を行動を融合する際に選択，統合しなければならない．しかも，それを一つのスーパーバイザが行うのではなくて，全てのモジュールの協調の結果として実現されるように，必要な知識をプリコンパイルした形で設計しなければならないのである．これは容易ではない．

2.3.2 服属アーキテクチャ

Brooks は，並列型のアーキテクチャにおいてこの協調動作を簡単に設計できるような仕組みとして，服属アーキテクチャ（subsumption architecture）[1]を提唱した [16]．このアーキテクチャでは，図2.3に示すように，上位のモジュールが下位のモジュールの入出力を抑制したり置換したりすることができる．逆に，下位のモジュールが上位のモジュールに対して直接的に作用を及ぼすことはできない．このような意味において，下位のモジュールが上位のモジュールに服属する（subsume）といえる．

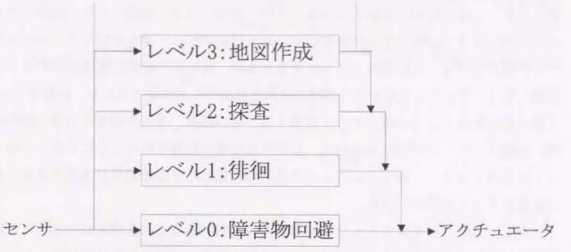


図 2.3: 服属アーキテクチャ（[16] より改変）

このように情報の流れに制限を加えることによって，下位のモジュールは，より上位のモジュールを気にすることなく設計することができるので，設計が容易になる．また，システムを拡張する時には，最上位のモジュールを新たに付け加えればよく，既存のシステムを書き換える必要がないので，拡張性に優れている．しかし，情報の流れを制限することは，複雑な環境中で複雑なタスクを実行するために必要な柔軟性を損なうことも意味している．現在までに実現されているシステムは，ジュースの缶を集める [17] などの比較的単純なタスク[2]を実行するものに限られているので，この服属アーキテクチャが複雑なタスクに適用できるかどうかは明らかでない．

また，情報の流れの制限によって設計が容易になるとはいえ，いろいろなタスクに対して手作業でシステムを設計するのはかなり厄介な仕事である．自律的に服属アー

[1] 文献 [49] の用語に従う．“包摂アーキテクチャ” と訳す場合もある．
[2] ただし，このような簡単なタスクであっても，従来のアーキテクチャでは，実世界中でうまく実現できなかったのである．

キテクチャを構成する手法としては，中須賀らによって帰納学習を用いる手法 [90, 110] が，Koza によって遺伝的プログラミング（Genetic Programming）を用いる手法 [54] が，それぞれ提案されている．

2.3.3 行動ネットワーク

Maes は，柔軟な情報の流れを可能にする並列型のアーキテクチャとして行動ネットワーク（behavior network）を提案した [64]．行動ネットワークは，図 2.4 に示すように，複数の行動モジュールと，行動間の因果関係を表すリンクで構成される．各行動モジュールは STRIPS-like なオペレータで，条件リスト，追加リスト，削除リストの三つのリテラルのリストを持っている．ある行動モジュールの追加リスト中のリテラルが他のモジュールの条件リストに含まれる時，前者から後者に因果関係のリンクが張られる．そして，知覚からの情報の流れと目標からの情報の流れを，行動モジュール間の活性化エネルギの流れとして表現する．その結果，現在の知覚と目標に関連の深い行動モジュールが活性化されて，対応する行動が起動されることになる．このような仕組みによって，再プランニングや環境の変化に対する頑健性が実現できることが実験によって示されている．

しかし，このアーキテクチャでは知識が STRIPS-like な命題論理表現で表されるので，連続的な知覚や行動をあらかじめ定義された記号で抽象化しなければならない．そのために，条件リスト，追加リスト，削除リストに書くべき記述の量がうまく限定されずに爆発してしまう恐れがある．これはフレーム問題に他ならない．また Maes 本人によって，各モジュールの活性化の閾値や，リンクを流れるエネルギを決定するパラメータを適切な値に調整するのが難しいことが指摘されている [64]．

Maes は，行動モジュール間の因果関係を学習によって決定する手法も提案している [65]．この手法を使えば，実際の経験に基づいて必要なリンクだけが形成されるのでリンクの量（すなわち記述の量）を限定することが期待できる．また，各リンクのエネルギの伝搬しやすさも，因果関係の強さに対応させることによって，自律的に調整される．ただし，連続的に実行された行動間で因果関係がない場合や，複数の行動によって初めて環境中に明確な変化が現れる場合などにはうまく働かないという欠点がある．

2.3.4 ハイブリッド・システム

複雑な環境中の複雑なタスクを扱えるようにするために，行動に基づく知能のアーキテクチャの中に記号処理システムを組み込むハイブリッド・システムの研究も行わ

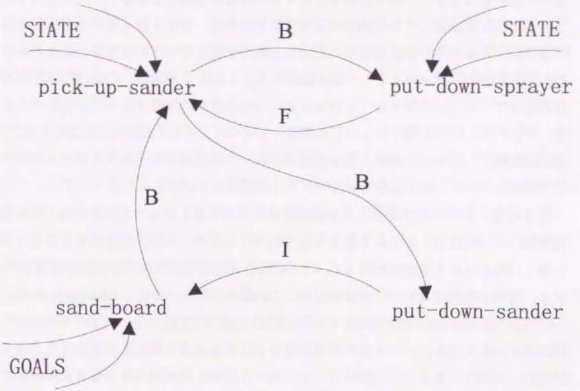


図 2.4: 行動ネットワーク [64]

B は後ろ向き，F は前向きの活性化エネルギの伝搬を表す．I は抑制関係を表す．

れている [4, 71]. 行動に基づく知能のアーキテクチャは並列的なので，記号処理などの複雑で遅い情報処理を行うモジュールを含んでも，並列性，即応性，行動の創発などの利点は，損なわれることはない．小さいフィードバックループで環境と密に相互作用する反射的なモジュールと，プランニングなどの強力な情報処理を行う熟考的なモジュールを組み合わせることによって，大局的にも局所的にも合目的的な行動を実現できるシステムとなることが期待できる．

ただし，記号処理モジュールをうまく機能するように組み込むのは簡単な問題ではない．記号処理モジュールと反射モジュールの出力が矛盾する場合に，良く思案されているが現実と乖離している恐れのある前者の出力と，現実に即しているが大局的には盲目的である可能性のある後者の出力のどちらを優先すべきかは簡単には決められないからである．現在のところ，記号処理モジュールは，大局的なプランの作成 [4] や新たなリアクティブ・ルール（反射モジュールの相当）の作成 [71] などに使われている．そのため，記号処理モジュールは反射モジュールに対して絶対的な影響を与える．逆に反射モジュールは直接的に記号処理モジュールに影響を与えることはできない．この意味において，記号処理モジュールの方が優越しているといえる．

記号処理モジュールで実行される情報処理について考えてみると，大局的プラン作成と新ルール作成は，どちらもかなり大きなタイムスケールで，環境の変化を見ながら徐々に実行される情報処理である．そのため，ある程度時間のかかる記号処理を行っても，環境の変化に遅れずに追随できることが期待できる．しかし，Dennett のロボットで用いられていた記号処理システムを，行動に基づく知能のアーキテクチャの中に組み込んだとしても，バッテリーを救出するタスクには全く役に立たないことは間違いない．このような形のハイブリッド・システムでは，環境中のある変化に反射的に反応することは可能になるが，フレーム問題のためにプランニングが破綻するのは避けられないからである．結局このタスクにおいては，ハイブリッド・システムに反射的に爆弾を遠ざける仕組みが組み込まれていれば救出できるし，そうでなければ爆発する．

2.4 ロボットにおける認識の学習

Dennett のロボットの問題で正しくプランニングするにはどうしたら良いだろうか．一つには設計者が，「ワゴンの上にバッテリーと爆弾の両方が乗っている場合は，爆弾を下ろしてからワゴンを引き出せば良い」という知識をロボットに与えて（設計して）やれば良い．しかし，ロボットが実環境で遭遇するであろうすべての状況に対処する知識をあらかじめ組み込んでおくことはできない．そこで，ロボット自身が環境

中で得た経験から，前述のような知識を学習する仕組みが重要である[3]．

しかし，記号体系の上での学習では，あるタスクを実行するために必要な知識だけを学習していっても，複雑なタスクの場合は，フレーム問題によって記述の量が爆発してしまうだろう．なぜなら，従来用いられてきた述語論理表現などの記号体系は非常に冗長で，現実には存在し得ないような状況をたくさん表現できてしまったり，同じとみなせる状況を何通りもの異なる記号で表現してしまったりするからである．従来用いられてきた記号体系では，後から自律的に記号を加えることができないので，あらかじめ十分に記号を用意しておかなければならない．細かい区別が必要な状況のために記号を用意することで，細かい区別が必要ない膨大な状況が記述可能になってしまう．Dennett のロボットの例でいうと，ロボットはもしかしたら部屋を区別するために壁の色という情報を利用しているかも知れない．そのために壁の色を表すたくさんの記号を持っているかも知れない．しかし，ロボットが既にバッテリーを発見してワゴンの引き出し作業にとりかかっている状況では，壁の色が何色であろうとも関係ない．それにも関わらず，状況を正確に表現しようと思えば，「壁の色が白で，ロボットがワゴンを引き出している状況」，「壁の色が黄色で，ロボットがワゴンを引き出している状況」…，などはすべて異なる状況として考慮しなければならなくなる．さらにたくさんのあり得ない状況，例えば「ワゴンを引き出している最中に壁の色が白から緑に変わる状況」なども考慮の対象に入ってくる．

デフォルト推論のように記述されないものは変化しないとみなす方法もある．2.3.3節で述べた STRIPS-like な条件リスト，追加リスト，削除リストの表現もそうである．しかし，変化するものをすべて書き尽くすことができないというのもフレーム問題の一つの現れであった．記号体系の記述の自由度が大き過ぎることはこの記述の量の爆発も促進する．その状況においてあまり重要でないことであっても，記述上で変化するものは全て書き尽くす必要があるからである．例えば，ワゴンの車輪が回転して床にかすかなわだちを残したとしても，バッテリー救出作戦には，通常，何も関係はない．ところが，わだちが残ることを表すリテラルが定義されていれば，そのような関係のないことも書き尽くさなければならない．

このような記述の爆発を避けるためには，記述の自由度を小さくすれば良い．現在のタスクを実行する上で区別する必要のない状況は全て同じ表現で記述し，あり得ない状況は記述することもできなければ良いのである．すなわち，現在のタスクの遂行に必要なことだけを十分コンパクトに書けるような記号を用意すれば良いのである．しかし，必要十分な記号をあらかじめ設計者が定義しておくことは非常に困難である

[3] もちろん，少々の失敗ではロボット自身が破壊されないという条件が必要である．

(Articulation Problem [39]). そこで，記号そのものをロボットが環境中で学習するというアプローチが考えられる．これは「抽象化の学習」，または「認識の学習」とみなせる．

このアプローチにおいては，情報処理に用いる記号は，ロボット自身がタスクに対する有用性に基づいて環境中から抽象して作り出す．環境の変化などで記号が不適切になった時には，適切になるように修正する．これは，2.2節で述べたフレーム問題に対処する機能の3.「注目すべき情報を取り出す方法の学習」に相当する．すなわち，タスクに関して考慮すべき情報の限定の仕方を学習することによって，フレーム問題を徐々に避けることができるようになる．

このアプローチは，2.3節で述べた物理基盤仮説と対立するものではなく並立するものである．情報処理の並列性，即応性を実現する反射モジュールと組み合わせて記号処理モジュールを用いるハイブリッド・システムにおいては，記号処理モジュールが優越していること，そのためにプランニングの際のフレーム問題が避けられないことが問題になっていた．そこで，記号処理モジュールで用いる記号を，反射モジュールと環境との相互作用の中から切り出すようにすれば，反射モジュールから記号処理モジュールへも情報が流れるようになる．そして，現在のタスクに必要な記号だけを処理することが可能になる．このような反射モジュールから記号処理モジュールへの情報の流れを作ることによって，以前から存在した記号処理モジュールから反射モジュールへの情報の流れと合わせて，柔軟な情報処理が可能になる．

従来の記号体系は，人間が設計する都合上，人間に理解しやすいものである必要があった．しかし，このように自律的に獲得される記号体系は，必ずしも人間にわかりやすい記号である必要はない．人間とのコミュニケーションを考えると，人間に理解しやすいという点も重要であるが，それよりもまず，ロボット自身がフレーム問題を現実的に解決できることの方が重要である．人間自身も内部で思考する時に，必ずしもすぐに口に出して言えるような明確な記号を用いているわけではないと考えられている．

従来のロボットでは，記号の学習などの認識の学習はほとんど行われなかった．記号の学習は記号を用いた情報処理と並列的に行われるもので，しかも非常にゆっくりと進むものなので，近似的には記号体系を固定してしまうことが可能だからである．記号体系を固定する方が効率の面では優れている．また，記号体系を上手に設計してやれば，記号の学習を行う必要はないと考えることもできる．しかし，現実には記号の冗長性によって，フレーム問題がうまく対処できない難問となっている．

一方，記号などの抽象化の学習を全くの白紙状態から行うのも非現実的なアプローチであろう．人間のように，生まれた時は無力で，十分に時間をかけて学習を行うシ

ステムであっても，初期の脳細胞の配置や神経繊維の結線の中に，非常に長い時間をかけて遺伝的に行われてきた学習の成果が先天的に備わっている．ロボットは遺伝的な資産は持っていないし，通常，学習にそれほど時間はかけられない．また人間のような汎用性は求められておらず，とりあえず，あるタスクに特化した自律性が求められている．そのため，何らかのアプリオリな知識を設計時に埋め込んでやることは，当然の選択である．その上で，実際の経験に基づいて，新たな認識を獲得したり，従来の認識に修正を加えたりするのが現実的なアプローチであろう．

しかし，認知行動システムによる認識の学習は，まだ始まったばかりの研究領域であり，アプリオリな知識として何をどのように与えたら良いのか，経験に基づく認識の獲得，修正はどのように行ったら良いのかなどは，まだほとんど何もわかっていない．後者の経験に基づく認識の獲得，修正は，機械学習の一種，強化学習の分野でいくつかの研究が行われている（4.4節参照）．それはまだ，記号の学習としては知覚入力空間の分割，記号を用いた情報処理としては強化学習しか行わないし，アプリオリな知識はほとんど何も与えることができない非常に初歩的なものである．しかし，もっと強力な記号体系，もっと複雑な情報処理，そしてアプリオリな知識の融合を目指してこの種の研究を発展させていくことは，実世界中の知能ロボットを実現する上で非常に有用であると思われる．

第 3 章

認識の学習

3.1 認識について

計算機科学における認識とは，「外界に存在する対象のパターンを，計算機がすでに知っているものとして認めること」である [3]．「計算機がすでに知っているもの」とは計算機の内部表象のことなので，計算機による認識とは，知覚入力パターンを抽象化して計算機の内部表象におけるいずれかのカテゴリ（またはクラス）に分類することだと考えられる．ただし，本論文においては，認識とは，ただ分けることを目的とするのではなくて，分けられた結果が何か別の基準で意味を持つような分類であると考える．つまり良い認識であるかどうかの基準は，システム外部になければならない．従って，認識では，分類した結果をどのように解釈するか，どう用いるかが重要である．

従来のコンピュータによる認識では，分類されるカテゴリの意味が，多くの場合，人間によって与えられている．知能ロボットの研究の大きな一分野であるコンピュータビジョンにおいても，静止画（もしくは動画）中から，人間によって定義された記号に相当する対象（例えば，コップ，机など）をいかにして見つけ出すかが重要なテーマである．ニューラルネットワークを用いた文字などのパターン認識の学習においても，人間によって定められたカテゴリへの収束が目標とされている．これは，認識の部分だけを切り出してシステムを構成しているからである．

しかし，知能ロボットなどの認知行動システムで行われる認識では，カテゴリの意味を人間が決めてやる必要はない．認知行動システムの場合は，分類の善し悪しを行動の結果から判断することができるからである．すなわち，システムが環境中でうまく行動できるようなカテゴリが構成できれば良いのである．そこで，2.4節で述べたロボット自身による認識の学習というアプローチが考えられる．このような学習を行うことにより，初めて，人間を含まないロボット内部で閉じた認識系が構成される．

2.4節の最後で述べたように，認知行動システムによる認識の学習はまだほとんど研究されていない．しかし，従来から研究されてきた様々な分類法および分類の学習法を，この用途の一部として用いることができる．そこで，この章では，認知行動システムによる認識の学習を念頭において，様々な分類法を紹介する．そのうちのいくつかは，認知行動システムによる認識の学習の一例である強化学習の入力の一般化の学習のために用いられている（4.4節参照）．

3.2 分類の学習

分類について考える場合には，分類のための内部表現と分類法，および内部表現の設計法を考えなければならない．ある分類法に対して内部表現は一意に決まるといって良い．またその内部表現の設計法は複数存在する場合もあるが，通常はほぼ限定されるので，分類法と内部表現とその設計法の三つを合わせて分類法として扱うことにする．

内部表現の設計法の中には，機械学習によって内部表現を構成する方法もある．本研究では認識の学習に焦点を絞るので，ここでは学習によって内部表現を構成するような分類法について紹介する．

分類の学習法には，教師付き学習と教師なし学習の二種類がある．教師とは正解のカテゴリを与えてくれるものである．この節では，それぞれの学習を順に説明する．

ここで，以下の説明において使用する用語の説明をしておく．「事例」とは分類される対象のことである．認知行動システムの認識では，個々の時点での知覚入力に相当する．「訓練事例」とは学習のために用いられる事例のことである．「テスト事例」とは学習の成果を確認するために用いられる事例のことである．「属性」とは個々の事例が持つ性質のことで，この属性を元にして事例が分類される．認知行動システムの認識では，個々の時点での個々のセンサの値（出力）に相当する．「属性空間」とは属性を軸として張られる空間のことで，各事例はこの空間内の点として表現される．認知行動システムの認識では，知覚入力空間に相当する．

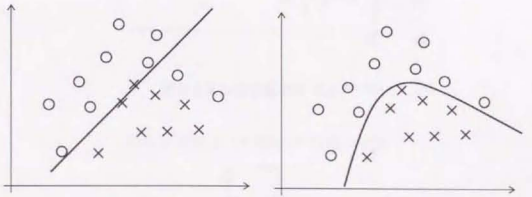
3.2.1 教師付き分類学習

教師付き学習とは，教師によって正解が与えられる学習のことである．分類の学習の場合は，正解としてカテゴリ (典型的には正，負) が与えられる．そして，その分類法を用いてカテゴリが不明の事例（テスト事例）を正しいカテゴリに分類できるようになるように，学習が進められる．

教師付き分類学習の場合は，分類に意味を与える基準は教師が持っているカテゴリ

である．訓練事例に関しては与えられたカテゴリに分類すること，テスト事例に関しても教師がその事例を分類するであろうカテゴリに分類することが認識の目的である．すなわち，教師が implicit に持っているカテゴリの形を，なるべく忠実に，なるべくコンパクトな表現で再現することが学習の目的である．教師の持つカテゴリは，通常，設計者によって定められているので，認識の意味はカテゴリを定めた設計者の頭の中にある．

統計学の分野で古くから用いられている分類法として判別分析がある [96]．これは，訓練事例が定められたカテゴリにうまく分類されるように，多次元の属性空間を，図3.1に示すように，超平面（線形判別関数）もしくは二次超曲面（マハラノビス距離による判別）を用いて分割する手法である．テスト事例は，分割された属性空間中のその事例が属する領域に対応するカテゴリに分類される．内部表現（境界面）は関数で表されているので非常にコンパクトで，分類のための計算も非常に簡単である．また分類の学習法としては，訓練事例を用いた簡単な計算で関数の係数を決定するだけである．しかし，母集団が複雑な形状のカテゴリを持つ場合は，誤分類が避けられない．



a. 線形判別関数による判別　　b. マハラノビス距離による判別

図 3.1: 判別分析による分類

パターン認識の分野で盛んに研究されている分類法として NN 識別法（nearest neighbor classification）がある [76, 121]．これは，最も単純な形では，学習時にすべての事例とそのカテゴリを記憶しておいて，分類時にテスト事例をその事例からもっとも近い訓練事例のカテゴリに分類する手法である．この手法による分類の様子を図3.2に示す．この手法では，学習は訓練事例の属性値を記憶するだけなので非常に簡単であり，また非常に複雑な形状のカテゴリを表現することもできるなどの利点がある．し

かし，全事例を記憶するために記憶容量が大きく，また分類時に全事例との距離を計算するためのコストが大きいという欠点がある．そのために，記憶する事例を選択して減らす手法や，距離計算の対象となる事例を限定する手法が盛んに研究されている [76]．またこの手法を発展させて，近い順にk個までの事例を集めて，その中で多数決をしてカテゴリを決定するk-NN識別法 [76, 121] や，カテゴリを重なりあった超直方体で表現して，最も近いカテゴリに分類するNGE（nested generalized exemplars）algorithm [126]（図3.3参照）などの分類法も提案されている．応用としては，記憶に基づく推論（Memory-Based Reasoning, MBR）や，事例に基づく学習（Instance-Based Learning, IBL）などがある．

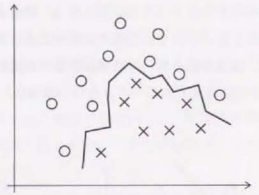


図 3.2: NN識別法による分類

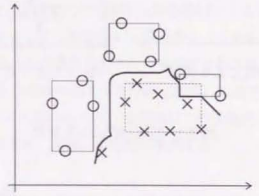


図 3.3: NGE algorithm による分類

判別木（決定木）は，図3.4のような木によって分類を行う．根ノードから出発し，各中間ノードで，ある属性に関する判別テストを行う．ノードから出る枝がその属性の値に対応している．そして，到達する木の葉が分類先のカテゴリに対応している．

判別テストは図3.4のような属性値に当てはまるかどうかのテストだけではなく，不等式も含み得るので，連続的な属性も扱うことができる．ただし，各ノードでは，通常，ある一つの属性に関する判別を行うので，属性空間は，図3.5のように，各属性の軸に垂直な超平面によって分割される．

図 3.4: 判別木（十字路での行動の分類）

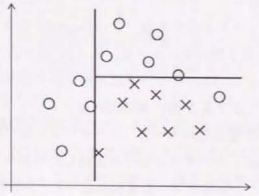


図 3.5: 判別木による分類

訓練事例は有限なので，必ず全訓練事例を正しく分類できる判別木を生成（学習）

することが可能である．しかし，正しく分類できる木は一つだけではなくて何通りもあり得るので，なるべくコンパクトに表現することが判別木を生成する上で重要である．木構造が単純な方が，分類も速いし，未知の事例を分類する上でも誤りが少ないことが期待できる．なぜならば，複雑な判別木はその時の訓練事例に特化し過ぎているからである．

Quinlan は，情報量の点からもっとも単純な判別木を生成する手法として，ID3 を提案した [99]．ID3 では，判別テストの順序と，判別式が不等式の場合はどこに境界を定めるかということを，情報量を用いて決定する．具体的には，カテゴリが異なるデータが混在するノードにおいて，判別テストを行った時に得られる情報の量が最大となる判別テストを採用する．この操作を繰り返すことによって，判別木が生成できる．この手法は，計算量が事例の数に対して指数関数のオーダーにはならない [3] ので，大量の事例の分類に用いることも可能である．しかし，もっとも単純な判別木でも，軸に垂直な超平面で切るという制約を持っているので，真のカテゴリの形状を近似するためには，図 3.5に示すようにあまり効率は良くない．

ニューラルネットワークの領域でも，教師付き分類学習を行うことができるアーキテクチャが数多く存在する[1]．文字などのパターン認識には，図 3.6に示すようなパーセプトロン型の階層的ニューラルネットワークがよく用いられている[2]．その学習には，誤差逆伝播アルゴリズム（error back propagation algorithm）がよく用いられている．このアルゴリズムでは，出力誤差の二乗和の最急降下方向に学習が進むことが保証されている．ネットワークが二層である場合は，このアーキテクチャによって属性空間は超平面で分割される．ネットワークが三層以上の場合は，境界を明示的に表現することは非常に難しい．

非パーセプトロン型のニューラルネットワークとしては，1982 年に Hopfield によって提案されたネットワークが有名である．この Hopfield のネットワークは，パーセプトロンと同様のしきい素子型のユニットを，図 3.7に示すように対称的に相互結合したネットワークである．このネットワークは相関学習や直交学習を使って自己想起的な連想記憶に用いることができる [9]．具体的には，ネットワークの各結合の重みを学習することによって，記憶したパターン（「パターン」は「事例」に相当する）がアトラクタとなって，類似したパターンをその記憶したパターンに引き込むような動作をする．そこで，記憶したいカテゴリの典型的なパターンを学習させることによって，

[1] ニューラルネットワークの詳しい説明は本論文の趣旨を外れるので省略する．詳しい説明は，文献 [3, 9, 101] などを参照のこと．

[2] パーセプトロンとは，1958 年に Rosenblatt によって提案された学習する神経回路網様のパターン識別機械の元祖である．

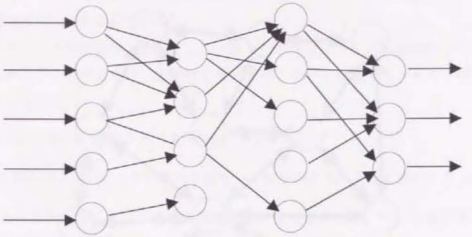


図 3.6: 階層的ネットワーク

それに類似したパターンを記憶したカテゴリに分類するという使い方ができる．

1983 年に Hinton らによって提案されたボルツマン・マシン（Boltzmann Machine）は，Hopfield のネットワークと同様の対称的な相互結合型のネットワークで，各素子の動作を確率的にしたものである．そのために，パターン空間中の全てのパターンが，ローカルなアトラクタではなくて最も強いアトラクタに高い確率で引き込まれる．そこで，ネットワークの一部（入力部）に入力パターンを固定して，あるパターンをネットワークの別の部分（出力部）に想起させるという相互想起的な連想記憶に用いることができる [9]．出力部のパターンをカテゴリに対応させれば，類似した入力パターンをカテゴリに分類する手法として用いることもできる．

これらのしきい素子型のニューロンを用いたニューラルネットワークでは，計算が非線形であること，パラメータが非常に多いことなどから，上述の統計学的な手法と比べると，計算能力および学習能力の解析が非常に難しい．学習されたカテゴリの形状もわかりにくい．しかし，経験的には，かなり複雑なカテゴリでもうまく近似できることがわかっている [3, 9, 101]．

Kohonen によって提案された学習ベクトル量子化（Learning Vector Quantization, LVQ）アルゴリズム [53, 1] は，ニューラルネットワークモデルの一種であるが，今までに説明したニューラルネットワークとはかなり異なる処理を行うニューロンを用いる．LVQ では，各カテゴリに一つのニューロンを対応させる．各ニューロンは属性空間中に参照ベクトル（「ベクトル」は「事例」に相当する）を持つ．新たな事例は上述の NN 識別法を用いて，最も近い参照ベクトルを持つカテゴリに分類される．そして，分類されたカテゴリが教師が定めたカテゴリと等しい場合は，参照ベクトルを

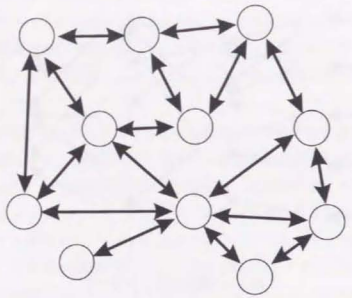


図 3.7: 対称的な相互結合型ネットワーク

その事例のベクトルに若干（ベクトル間の距離に比例した短い長さ）近付ける．逆に，分類されたカテゴリが教師の定めたカテゴリと異なる場合は，同じ長さだけ遠ざける．このようにして，漸次的に教師の持つカテゴリの形状を近似することができる．この手法は，ニューラルネットワークとはいっても統計学的な手法にかなり近く，NN 識別法の参照点を非常に減らしたものとみなすこともできる．参照点を減らした分だけ内部表現は圧縮され，計算も速くなるが，近似精度は落ちる．

3.2.2 教師なし分類学習

教師なしの分類の学習においては，訓練事例を分類すべきカテゴリが定められていない．訓練事例の集団は，分布に基づいて，いくつかの良い部分集団に分割され，各々の部分集団がカテゴリとなる．このカテゴリの学習過程は，「クラスタリング」とも呼ばれる．この学習法では，個々の事例が分類されるべきカテゴリの正解が定められていないので，テスト事例は存在しない．新しい事例は，分類法に従って，それまでに生成されたカテゴリの一つに分類される．学習が漸次的に行われている場合は，その新しい事例も訓練事例となって，カテゴリの修正に用いられる．

教師なし分類学習の場合は，カテゴリは訓練事例の分布にのみ基づくので，そのままでは分類に意味を与える基準は存在しない．その分類が良い認識法になっているかどうかは，分類した結果をどのように解釈するかに依存している．通常，分類に意味を与える基準がシステムの外部，設計者の頭の中に存在して，その基準の上で良い認

識になるように設計者によって学習法が選ばれたり，パラメータが調整されたり，時には属性を変えたりする必要がある．

クラスタリングの代表的な手法として，K 平均アルゴリズム（K-means clustering）が挙げられる．この手法では，全体を K 個のクラスターに分割する．その手順は次の通りである [78]．

1. 適当な位置に K 個のクラスターの中心を定める．
2. NN 識別法を用いて全ての訓練事例をいずれかのクラスターに分類する．
3. それぞれのクラスターで属する訓練事例の重心を求めて，それを新たなクラスター中心とする．
4. 重心の位置が動かなくなるまで 1-3 を繰り返す．

分類をうまく行うためには，クラスター内の事例間の距離の平均などの基準を用いて，各クラスターがなるべくコンパクトにまとまるようにパラメータを調整しなければならない．パラメータとしては，クラスターの数 K と，初期のクラスター中心の位置などがある．獲得された内部表現を用いて新たな事例を分類するには，NN 識別法などを用いる．しかし上述のように，この手法は漸次的ではないので，新たな事例を加えると全ての訓練事例を用いて再計算する必要がある．

クラスタリングには K 平均アルゴリズムの様に複数のカテゴリに分類するだけではなくて，様々な類似度のレベルで階層的にクラスタリングを行う手法もある．うまく階層的クラスタリングを行うと，各階層のカテゴリを人間の持つ概念に対応させることができ，複雑な認識を行うことも可能になる．例えば，ある人を人間であると認識するだけではなくて，動物界の脊索動物門の哺乳綱の霊長目のヒト科のヒトであると認識することもできる．

もっとも，この概念階層も自然界にはっきりとした正解が存在するようなものではなく，人間の目から見て自然に思えるようなカテゴリ構成に収束しているだけであるとみなすこともできる．ヒトの分類にしても，上述したもの以外に脊椎動物亜門や，四足上綱など様々な中間的なカテゴリが存在するし，日本の奈良県の奈良市の中山町西に住んでいる A 君といった全く異なるカテゴリの階層で認識を行うこともできる．結局，様々な属性およびパラメータを用いて，色々な意味での「良い」クラスタリングが可能であり，それに認識としての意味を与えるのは，その分類をどのように用いるかというシステムの外の基準である．

事例を，それぞれが異なる概念に対応するように階層的に分類することを目的として行う階層的クラスタリングのことを，一般に，概念クラスタリング（conceptual clus-

tering）という．概念クラスタリングでは，事例がどのような概念の例になっているかということが重要である．そのため，通常のクラスタリングの事例のような距離が定義できる連続的な属性ではなくて，あらかじめ人によって離散化された属性（色が「赤」「青」とか，背が「高い」「低い」など）が用いられる．すなわち，認識の学習といっても，属性空間を分節する目的ではなくて，分節して切り出された記号間の関係を定める目的で行われる学習である．概念クラスタリングには，カテゴリ内部の事例がなるべく密になるように（疎性度がなるべく小さくなるように）調整しながら，トップダウンに階層を構成していく手法 [3] や，カテゴリ内部の類似度とカテゴリ間の相違度から計算されるカテゴリの有用性（Category Utility Value, CU 値）に基づいて，逐次的にカテゴリの学習を行う手法（Fisher の COBWEB [27, 28]）などがある．

大量のデータの分析のためにクラスタリングを行うことを，クラスター分析という[3]．クラスター分析は数値分類法の一種で，主観をまじえずに，一定のアルゴリズムにしたがって分類することだけを目的とする．そのために，抽出されたクラスターをどう使うかということはあまり問題ではなく，データを人間が見やすいように要約して，事例間の関係をとらえようとする記述統計学的手法であるというべきである [96]．従って，認識を行うのはシステムではなくてシステムを使用する人間の側であるといえる．

クラスター分析では，データの分析が目的なので，漸次的な処理は行わず，全てのデータを一括して処理する．クラスター分析にも，階層的手法と非階層的手法がある．階層的手法としては，事例間の距離に基づいてボトムアップに階層を構成していく手法がよく用いられる．非階層的手法では，初めに適当にクラスターを構成して，ある基準に関して最良のクラスター構成となるように徐々に改良していく手法が用いられる．どのような手法を用いても，局所的な最適点に落ちついてしまう恐れがあるので，パラメータや初期値を変えたり，複数の手法を組合せて最良の構成を探索することが必要である．

ニューラルネットワークの領域にも，教師なし分類学習を行うアーキテクチャがいくつか存在する．パーセプトロンと同様のしきい素子型のユニットを用いるネットワークでは，図3.8に示すようなクラスター化された階層的ネットワークでの競合学習（competitive learning）がその一つである．

このネットワークでは，各層内のクラスター間にはリンクが存在しない．また，入力クラスター内のユニット間にもリンクが存在しない．隠れ層および出力層のクラスター内のユニット間には抑制性のリンクが存在して，各クラスター内で一つのユニッ

[3] クラスター分析に関しては，文献 [96] に詳述されている．

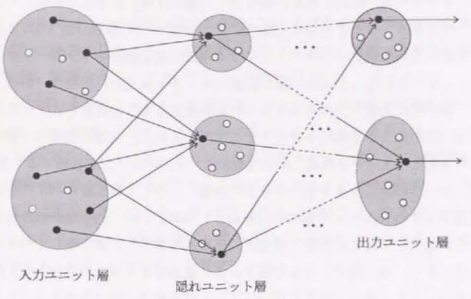


図 3.8: クラスター化された階層的ネットワーク

トしか発火しないように設定される．そして，入力層から出力層に向かって，隣あった層のユニットに興奮性の結合がある．この興奮性の結合を，発火したユニット間の結合を強くするように学習すると，そのうちに，入力層のある範囲の類似した入力に対して，出力クラスター内の特定のユニットが発火するようになる．出力クラスター内ユニットをカテゴリに対応させると，これは入力をクラスタリングしていることに相当する．この学習も，隠れ層の数や，隠れ層内のクラスターの数，各クラスター内のユニットの数，および各結合の初期値などを変えて，望ましいクラスタリングができるように調整する必要がある．

相互結合型のネットワークとして教師付き分類学習のところで説明した Hopfield のネットワークの学習では，通常，学習したパターン以外にもローカルなアトラクタができてしまう．このようなアトラクタは，学習に用いるパターンとその与え方によって決定されるが，これを一種の教師なし分類学習の成果とみなすこともできる．また，ある特定のパターンを学習させようとするのではなく，訓練事例をランダムな順序で与えて学習を行った場合も，様々な強さのいくつかのアトラクタが構成される．このネットワークでは，ある入力をネットワークの初期状態として与えてやると，必然的に一つのアトラクタに収束する．各アトラクタをカテゴリに対応させると，これは入力をクラスタリングしていることに相当する．この手法でも，「良い」カテゴリを構成するためには，結合の初期値や訓練事例の与え方などに工夫する必要がある．

Grossberg らによって提案された ART（Adaptive Resonance Theory, 適応共鳴理

論）ネットワークは，仮説と検定の繰り返し（適応共鳴）によってパターン認識を行うニューラルネットワークである [23, 1]．ネットワークの構成は図3.9のように表される．各層はHopfieldのネットワークのような相互結合型のネットワークで，短期メモリとして用いられる．特に第二層の内部のユニットは互いに抑制性の結合で結ばれていて，競合的にただ一つのユニットだけが発火するように調節されている．第一層と第二層の間の結合が長期的なメモリとして用いられる．入力はまず第一層において短期メモリ上にパターンを構成（仮説生成）し，そのパターンが長期メモリを介して，第二層上の一つのユニットを発火させる（分類）．そして第二層のパターンを第一層に逆伝達することによって分類の検定を行う．ARTでは，いくつかの特殊なユニットとルールを用いて，この仮説と検定を，自律的に素早く，繰り返して行えるようにしている．また，第二層で入力を分類するのに適当なカテゴリが存在しない時には，まだカテゴリとして使用されていないユニットが自動的に新たなカテゴリとして使用される．そして，カテゴリが定まると長期メモリの中に記憶される．このアーキテクチャは他のニューラルネットと比べて，学習が速く，メモリを効率良く利用していて，既存のカテゴリに属さないデータも特別な処理をすることなく扱えるなどの利点を持っている [23]．ARTは「一般的な問題解決を行う自己組織系」になり得るアーキテクチャであるが，具体的なインプリメントが難しく，簡単なパターン認識以外のシステムにはまだあまり用いられていない [1]．

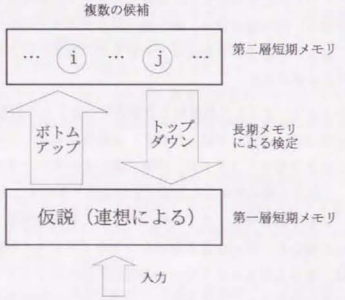


図 3.9: ARTの構成 [1]

Kohonenによって提案されたSOM（Self-Organizing Map, 自己組織化マップ）アルゴリズム [53] は，通常のクラスタリングとは少し違ったカテゴリを用いる．通常のクラスタリングでは，カテゴリに関する制約はほとんどなくて，せいぜい数が決まっているくらいであるのに対して，SOMアルゴリズムでは，カテゴリ集合に構造が定められている．具体的には，SOMでは，上述のLVQのニューロンと同様に，カテゴリに一対一で対応するニューロンを使用する．ニューロンは属性空間内に参照ベクトルを持つが，その参照ベクトルとは全く関係なく，ニューロン自身も2次元や3次元などの格子状に配列されている．そして，属性空間中の入力は，NN識別法を用いて最も近い参照ベクトルを持つカテゴリに分類され，ニューロンの格子の上にマッピングされることになる．その様子を，図3.10に示す．この図では，大きな三角形が属性空間中の入力が与えられる領域を表し，格子点がニューロンの参照ベクトルの配置を表す．格子はニューロン自身の配列を表す．

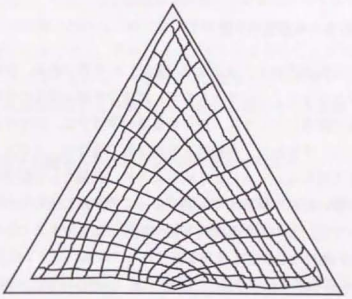


図 3.10: 属性空間中の参照ベクトルの配置 [53]

SOMの学習は，次のように漸次的に行われる．入力が与えられて，あるカテゴリに分類された時，そのカテゴリに対応するニューロンの参照ベクトルを，若干，その入力ベクトルに近付ける．さらに，その発火したニューロンにニューロン配列上で近接しているニューロンの参照ベクトルも，少しずつその入力ベクトルに近付ける．こうして，ある入力が与えられた時に，その周辺のニューロンが，ゴムの幕を引っ張るように，その入力に近付くことになる．

その結果として，与えられた訓練事例の分布密度に応じたカテゴリの分割ができる．訓練事例の密度が大きいところではカテゴリが細かく，密度が小さいところではカテ

ゴリが粗くなる．そこで，このアルゴリズムは，頻繁にデータが得られるところほど細かく認識したい場合に適しているといえる．

また，このアルゴリズムを用いると，多次元の入力をより次元の低い格子上にマッピングすることができる．例えば，2次元の格子を3次元的にくしゃくしゃにまるめることによって，3次元の立体の内部を満たすこともできるわけである．これは，人間が，身体の内部感覚などの3次元的な情報を，2次元の脳の表面にマッピングするのに類似している．そこで，入力が多次元でも実際の問題が本質的にはより少ない次元で表し得る場合に，その少ない次元でニューロンの格子を作ってやることによって，入力をずっと簡単な内部表現にマッピングしようとする研究も行われている [55, 58]．

3.3 本研究における認識の学習

3.3.1 本研究における認識の学習

認識の学習は属性空間のカテゴリ分け（分節）の学習である．分類されるカテゴリは，一種の記号とみなすことができるので，認識の学習は記号の学習とみなすことができる．この章で説明したように，従来の認識の学習では，記号の意味は人間によって与えられていた．すなわち，教師付き分類学習の場合は，カテゴリの意味は分類の結果を利用する人間によってあらかじめ与えられ，教師なし分類学習の場合は，カテゴリの意味は分類の結果を解釈する人間によって分類後に決められるのである．このようなシステムでは，どのような意味を持つ記号が必要であるのかということは，システムの設計者の頭の中に存在している．

3.1節で述べたように，本研究では，認識とは，ただ分けることを目的とするのではなくて，分けられた結果すなわちカテゴリが何か別の基準で意味を持つような分類であると考える．従来の認識の学習においては，上述のように，分類の結果を利用する人間によって，カテゴリに意味が与えられていた．従って，システム内部だけでは認識系が完結せず，カテゴリに意味を与える人間を含めて認識系が構成されていると考えられる．

しかし，認識系に思考系と行動系を加えた認知行動システムでは，システム全体が環境中でうまく行動できるという基準で，カテゴリに意味を持たせることができる．この場合には，うまく行動できるようなカテゴリを構成するための指標をシステムに与えてやる必要があるが，人間が直接カテゴリに意味を与えるのと比べると，はるかに自律的なシステムとなる．本研究で扱う認識の学習は，このような認知行動システムに含まれる自律的な学習である．

3.3.2 認識の学習を行う認知行動システム

2.4節で述べたように，従来の記号処理に基づく認知行動システムにおいては，認識の学習すなわち記号の学習はほとんど行われなかった．記号処理に基づく認知行動システムで行う学習は，ルールの学習や論理に基づく学習などの認識された表象の上での学習が一般的である．これは，固定された記号体系の上での記号間の関係の学習とみなすことができる．記号の学習は記号間の関係の学習よりもはるかにゆっくりと進むものなので，近似的に記号体系を固定することができる．その方が効率の面では優れているといえる．

しかし，実世界中の認知行動システムを構成する場合には，記号体系をあらかじめ適切に定めるのが非常に困難であることが，フレーム問題として認識されている．すなわち具体的には，不適切な記号体系を用いると，記号間の関係では記述できないような知識が存在したり，知識の量が記述しきれないほど膨大になってしまったりする．このようなフレーム問題を避けるためには，環境やタスクにとって必要十分な記号体系を構成することが有効である．そのために，タスクを遂行しながら認識の学習を行うというアプローチが考えられる．本研究では，従来の認知行動システムでは十分に行われてこなかった認識の学習を自律的に行う認知行動システムを扱う．

3.3.3 認識の学習と認識された表象の上での学習の並列実行

従来の記号処理システムの学習では，認識の学習と認識された表象の上での学習が別々に扱われてきた．しかし，両者を組み合わせて認知行動システムの学習とすることによって，上述のように，両者の欠点を補うことができる．すなわち，認識の学習には，タスク遂行の必要性に応じて記号を学習するという基準が得られ，自律的な認識系を構成することができる．一方，認識された表象の上での学習だけではフレーム問題に十分に対処しきれない場合にも，両者を組み合わせることで，学習の柔軟性を高め，フレーム問題に対処できる可能性を高めることができる．本研究で扱う認知行動システムは，このように認識の学習と認識された表象の上での学習を同時に行う認知行動システムである．

具体的には，図3.11に示すように，環境からの知覚入力を知覚入力空間上で分節して記号空間にマッピングし，記号空間上で推論を行って行動を出力するシステムを考える．学習としては，行動出力とそれによる環境の変化に基づいて，知覚入力空間上での認識の学習と記号空間上での記号間の関係の学習を行う．認識の学習としては，タスク遂行の必要性に応じて，記号に対応する知覚入力空間中の領域を絶えずゆっくりと調整し続けながら，さらに大きな変更が必要な時には，記号の生成や削除を行う．

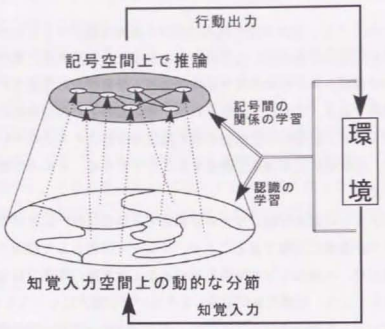


図 3.11: 認識の学習と記号間の関係の学習を行う認知行動システム

このように，認識の学習と認識された表象の上での学習を，タスクを遂行しながら同時に並列して行うことによって，環境とタスクにとって必要十分な記号体系を学習することができる．また，未知の環境や環境の変化に柔軟に対応することが可能になる．これは，2.2節で述べたように，フレーム問題を現実的に解決するために重要な機能である．一方，認識の学習を常に行い続けることは，効率の面では短所となる．また，タスクをうまく遂行できるような分節を行うための指標を定めなければならないという新たな問題が生じる．これらは，フレーム問題を現実的に解決するためには，積極的に取り組まなければならない問題であると考えることができる．

本研究で扱うような認識の学習と記号間の関係の学習をともに行う認知行動システムは，2.4節で述べたように，機械学習の一種，強化学習の領域でわずかに研究が行われている．そこで，次章では，まず強化学習について説明し，続いて強化学習の前処理として行われている認識の学習（すなわち知覚入力の抽象化の学習）について説明する．その中で，本研究の扱う研究領域をさらに絞り込んでいくことにする．

第 4 章

強化学習

2.4節で述べたように，認知行動システムによる認識の学習は，機械学習の一種，強化学習の領域でわずかに研究が行われている．この章では，従来の強化学習，および強化学習における知覚入力の抽象化の問題について概説する．

4.1 強化学習とは

強化学習（reinforcement learning）とは，本来，動物行動学の分野で用いられてきた用語で，「条件付け」といわれる一連の適応現象を実現する学習のことである [49]．機械学習の分野では，動物における強化学習をモデル化して，「学習者が環境中で行動を起こし，それに対して環境から与えられる「報酬」に基づいて，徐々に環境に適した行動パターンを獲得していく学習」のことを強化学習と呼んでいる．強化学習における「報酬」とは，システムの行動に対して数値で与えられる評価のことで，「強化信号」とか，負である場合には「罰」とも呼ばれている．報酬は，通常，各行動ごとに与えられるのではなく，環境中で何か評価し得る出来事が起こった時に与えられる．例えば，ロボットがゴールに到達したら 10 点とか，障害物にぶつかったら −2 点などである．この値は，理想的には，システム内のエネルギの増減とか，システムの損傷などから計算すべきであるが，通常は設計者によって明示的に定められる．

行動の学習は，教師付き学習でもよく行われている．教師付き学習の場合は，各行動ごとに，教師から出力すべき正解の行動が教えられる．それと比較すると，強化学習の特徴は，次の 2 点にまとめられる [49]．

- システムが出力すべきデータが教師からは与えられず，システムが実際に行った出力に対する評価という形で与えられる．
- システムの出力に対する評価が即座に与えられず，行為の系列に対する評価が遅れて与えられる．

このように，強化学習は，教師付きの学習よりもはるかに少ない情報から学習を行うことを目指す研究領域である．

強化学習システムは，一般的に，図4.1に示すように，実行要素と学習要素に分けることができる．実行要素では，その時の知覚入力と経験的知識に基づいて行動を決定し実行する．学習要素では，知覚入力と選択された行動と環境からの報酬に基づいて，行動の選択に影響を与える経験的知識を修正する．学習において，経験（すなわちデータ）に基づいて出力が決定されるのは当然であるが，強化学習においては，逆に経験の内容も出力（すなわち行動）に基づいて決定される．

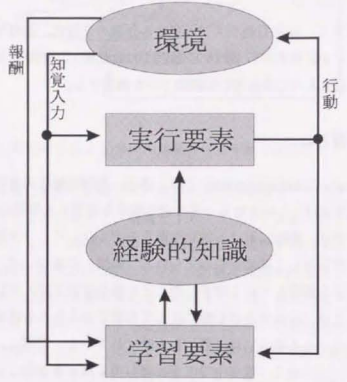


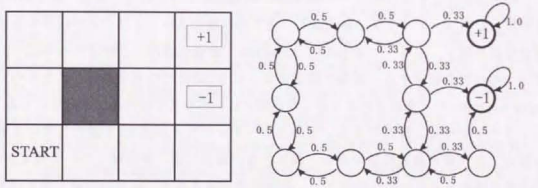
図 4.1: 強化学習システムの構成（[49] より改変）

強化学習では，システムに入力されるベクトルのことを「状態（state）」と呼ぶことが多い．特に，離散的な入力に関しては，一般的に状態と呼ばれる．しかし，連続的な入力に関してはその他にも，input signal, sensor output, input vector, perception, input pattern, perceptual state, state vector, vector of observations, 属性, 入力データなど，様々な呼び方が用いられる．また，連続的な入力をシステム内で離散化したものを「状態（state）」と呼ぶことも多い．この入力を離散化したものは，その他に，cluster, subspace, category, class, situation などと呼ばれている．本論文では，特に断らない場合は，システムに対する入力については連続的なものも離散

的なものも含めて「知覚入力」または「知覚入力状態」と呼ぶことにする．単に「状態」という場合は，離散的な入力または連続的な入力を抽象化したものを指すことにする．また，知覚入力が配置される空間のことを「知覚入力空間」または「入力空間」と呼ぶことにする．

4.2 マルコフ決定問題

強化学習では，通常，離散入出力，離散時間，マルコフ性が仮定される．典型的には，図4.2a に示すようなグリッド表現を用いた環境中での学習である．図4.2b はこの問題の状態遷移図の一例である．



a. グリッド環境　　b. 状態遷移図

図 4.2: 強化学習で扱う典型的な環境 [102]

この仮定によって，強化学習は，決定理論（decision theory）の分野におけるマルコフ決定問題（Markov decision problem）に帰着する．それは次のように定式化される [79].

> 状態空間を X，決定空間を B，$i \in X$ において $b \in B$ を選んだとき，$j \in X$ への遷移確率を $p(i,j;b)$ と記す．初期状態 $x_0 = i$ から始め，離散時刻 $k = 1, 2, \ldots$ の状態 x_k において決定 b_k を選択すると，無限期間に渡る総利得の期待値は
>
> $$U_\pi(i) = E_\pi\{\sum_{k=0}^{\infty} \gamma^k r(x_k, b_k) \mid x_0 = i\} \qquad (4.1)$$
>
> となる．ただし，γ $(0 \leq \gamma < 1)$ は割引率（discount factor），$r(x_k, b_k)$ は x_k で b_k を選んだときの利得，π は政策（すなわち各状態における決定

の選択規則）である．このとき総利得の期待値を最大にする政策（最適政策，optimal policy）は，動的計画法（dynamic programming, DP）の最適性方程式

$$U(i) = \max_{b \in B}[r(i,b) + \gamma \sum_{j \in X} p(i,j;b)U(j)],\ i \in X \qquad (4.2)$$

を解くことによって求められる．$p(i,j;b)$ と $r(i,b)$ は外界の性質を表す関数で，合わせてワールドモデルと呼ばれる．$U(i)$ は最適政策をとった時の無限期間に渡る総利得の期待値であり，状態 i のユーティリティ（utility）と呼ばれる．

ワールドモデルが既知である場合には，この方程式を解くために決定理論の分野で二つの手法が存在する [102]．一つは value iteration と呼ばれる手法で，式4.2の右辺を計算して各ユーティリティを更新し，全ての状態のユーティリティが収束するまで更新を繰り返す．もう一つは policy iteration（政策反復法）と呼ばれる手法で，まず一つの政策を仮定する．政策を固定すると式4.2はただの連立方程式になるので，value iteration や（変数があまり多くない時には）ガウスの消去法などを用いて各ユーティリティが容易に計算できる．求めたユーティリティが式4.2を満たしている場合は仮定した政策が最適政策である．満たさない場合には，各状態について式4.2の右辺で最大値を与える行動を政策とし，最適政策が求まるまで政策の更新を繰り返す．どちらも繰り返し過程を含んでいるので，初期値が最適値に近い場合は早く収束するが初期値が最適値から遠い場合は収束に非常に時間がかかる．この二つを比べると，value iteration が連続値であるユーティリティを収束させるのに対して，policy iteration は離散値である政策を収束させるので，後者の方が早く収束することが期待できる．policy iteration は，かなり大規模な問題まで解くことができる実用性の高い方法である [79]．

4.3 様々な強化学習法

強化学習では，通常，ワールドモデルが未知のマルコフ決定問題を扱う．これは，不完全情報マルコフ決定問題（Marcov decision problem with incomplete information）と呼ばれる問題で，ワールドモデルを学習しながら同時に政策の学習を行う．強化学習においては，通常，利得は報酬（または強化信号）と呼ばれ，決定は行動（または行為）と呼ばれるが，その他の用語，式は決定理論と同様である．

不完全情報マルコフ決定問題の最も単純な解法は，ワールドモデルの推定を行いながら，value iteration または policy iteration で最適政策を求める方法である．ワールドモデルは，過去の経験に基づいて最尤推定する．つまり，

$$p(i,j;b) = \frac{n_{ij}^b}{n_i^b}$$

ただし，n_{ij}^b は過去に状態 i において行動 b を実行した時に状態 j に遷移した回数，n_i^b は状態 i において行動 b を実行した回数である．また，$r(i,b)$ は状態 i において行動 b を実行した時に直接得られた報酬の標本平均値である．

ワールドモデルを更新するたびに iteration を収束するまで繰り返すのは非常にコストがかかる．また，不完全なワールドモデルに対して最適な政策を求めてもあまり意味はない．そこで，ワールドモデルと状態のユーティリティを行動のたびに共に少しずつ更新していく手法が最もよく使われている．この手法は DP-based learning と呼ばれている [13]．この手法には，式4.2によってユーティリティを更新していく Barto らの Adaptive Real-Time Dynamic Programming（Adaptive RTDP）[13]，各遷移の前後の状態のユーティリティの差（temporal difference, TD）を基にして状態のユーティリティを更新していく Sutton の TD method [112]，同じく TD を基にして各状態における行動の評価値（action-value, Q-value）を更新していく Watkins らの Q-learning [125] などがある．Q-learning では行動の評価値を学習するので，ワールドモデルを必要としない．DP-based learning では最適政策への収束が証明されているが，一般的に学習の収束が遅いことが指摘されている．学習を促進するために，直前の状態のユーティリティだけではなくてそれ以前の状態のユーティリティも少しずつ更新する方法（Barto らの eligibility trace を用いる方法 [14]，Sutton の recency parameter λ を用いる方法 [112]），経験をリプレイしながら学習する方法（Lin の experience replay [61, 63]），ワールドモデル上での仮想的な経験から学習する方法（Sutton の relaxation planning [113]，Lin の relaxation planning [62, 63]），教師が与える正しい行動に基づいて学習する方法（Lin [61, 63]）などが提案されている．

その他によく用いられる強化学習法を三つ紹介する．Barto らによって提案された actor-critic learning [14] は，一種の TD method によって状態のユーティリティを学習する adaptive critic element（ACE）と，そのユーティリティに基づいて政策を学習する associative search element（ASE）を結合したものである．ASE によって行動を決定するのでワールドモデルを必要としないが，単なる TD method と比べると，ASE の学習を行う分だけ学習の収束が遅くなる．bucket-brigade algorithm は，Holland の分類子システム（classifier system）[38] の一部として提案された強化学習法で，プロダクション・システム的なルールの強さを，DP-based learning のユーティリティの学習と類似した手法で調節する．同じく分類子システムの枠組の中で研究されてきた profit sharing 法は，獲得した報酬をその報酬を得る過程で使用された行動

規則に分配して，行動の評価値（Q値と同一）を更新する [72]．この手法では，マルコフ決定問題の最適政策への収束は保証されないが，DP-based learning に比べて学習が非常に速く進むことが指摘されている [75]．

4.4 強化学習における入力の一般化問題

4.4.1 入力の一般化問題

強化学習では，環境中で経験し得るすべての知覚入力状態で適切な行動を獲得することが目的である．しかし，実世界のような環境で用いる場合は，知覚入力空間は，通常，連続空間になるので，すべての知覚入力状態を別々に扱うことはできない．そこで，知覚入力を抽象化することが必要である．第1章で述べたように，抽象化によって，情報の圧縮，処理時間の短縮，経験の一般化などの利点が得られる．

しかし，抽象化を適切に行うことは非常に困難である．強化学習において，抽象化のレベルが低過ぎると，学習の収束が非常に悪くなってしまう．逆に，抽象化のレベルが高過ぎると，多くの知覚入力状態において適切な行動を選択できなくなってしまう．この知覚入力の抽象化を適切に行うという問題は「入力の一般化問題（input generalization problem）」と呼ばれている．これは，第2章で述べたフレーム問題の一種である．すなわち，実環境の持つ複雑性が強化学習システムの持つ情報処理能力をはるかに上回るために起こる問題である．

第2章では，フレーム問題を現実的に解決するために，抽象化の学習，すなわち認識の学習を行うことが有効であることを説明した．強化学習の領域では，入力の一般化問題に対処するために状態認識の学習を行う研究がいくつか存在する．それらの研究はまだ基礎的なものであり，扱う問題は簡単のものに限られている．また，定番として確立された手法は存在しておらず，様々な手法が模索されているところである．しかし，2.4節で述べたように，フレーム問題に対処する糸口として非常に重要な研究領域である．

この節では，それらの手法を整理して説明する．まず，それらの学習の性質を表すのにキーとなる三つの軸をここで説明しておく．

1. **オンライン学習 vs. オフライン学習**

   状態認識の学習には，タスクを実行しながら漸次的に行うオンライン学習と，環境中で主にランダムに行動したデータを大量に集めて，そのデータを用いて一括して状態空間を構成するオフライン学習の二種類がある．

   オンライン学習では，状態認識と行動選択の相互依存関係が成立する．従って，

良い状態認識によって良い行動選択が行われ，良い行動選択によって良い状態認識が保たれることが期待できる．これは，システム内の情報の流れが多様になるとみなすこともできる．しかし，このように柔軟な枠組みで精密な状態空間を構成することはかなり困難である．

一方，オフライン学習は，状態空間の設計という意味合いが強い．すなわち，大量のデータを用いて，できる限り精密に状態空間を構成することに重点がおかれる．また，固定した状態空間の上で行動選択の学習を行えるので，学習の効率の面では，オンライン学習よりも優れている．ただし，柔軟性に欠けるので，環境中の未知の状況や環境の変化には弱い．

フレーム問題を現実的に解決するという観点から考えると，環境中で未知の状況に遭遇しても，徐々に適応することができるオンライン学習の方が優れているといえる．

2. **連続入力 vs. 離散入力**

   強化学習における入力の一般化問題という場合，入力が連続ベクトルである問題と，離散ベクトルである問題の両方を含む．離散入力の問題は，ゲーム戦略の学習のように，もともと離散的である場合もあるが，実世界中の連続的な問題を，問題設定の時点で離散化してしまう場合もある．例えば，環境をあらかじめグリッドで分割しておく（空間分割グリッド表現．図1.1d，図4.2a参照）場合などである．このような抽象化の中には，既に設計者の恣意が含まれてしまうので，システム自身が経験に基づいてタスクの遂行に適した状態認識を学習するという点では不十分である．実世界中でタスクを自律的に実行するシステムとしては，連続的な知覚入力の抽象化の学習の方が望ましいといえる．

3. **報酬の類似度に基づく抽象化 vs. 基づかない抽象化**

   抽象化の全ての手法において，共通する大前提として「距離的に近いデータは類似した性質を持つ」という知識が用いられている．しかし，それだけでは適切な抽象化を行うことはできない．適切さの基準は，認知行動システムの場合は，行動の結果に置くことができる．ある認識に基づいて行動した時にタスクを効率良く達成できるような認識が，適切な認識なのである．

   3.3節で述べたように，タスクを効率良く達成できるような状態認識を獲得するためには，知覚入力空間を分節するのに何らかの指標が必要である．強化学習の場合は，システムのタスクを定めているのは報酬である．将来の期待される報酬をなるべく多くする行動が，効率良くタスクを達成する行動である．そこ

で，入力空間の分節のための指標として，この報酬を用いることができる．すなわち，報酬の面から見て類似した知覚入力状態をまとめることによって，タスクの遂行にとって適切な抽象化が行うことができる．強化学習は，そもそも環境からの報酬に基づいて行動規則の学習を行うものなので，このような抽象化の学習とは非常に相性が良いといえる．

次に，実際の強化学習における抽象化について説明する．強化学習は，状態の内部表現の違いによって二つに分けることができる．一つは知覚入力空間をいくつかの部分空間に分割して，その各々の部分空間を，内部が一様な性質を持つ離散的な状態とみなす方法である．これを，状態の「分割表現」と呼ぶことにする．もう一つは，状態の範囲を明示的に定めずに，知覚入力状態のなんらかの抽象化された性質を知覚入力の関数として表す方法である．これを，状態の「陰的表現（implicit representation）」と呼ぶことにする．

分割表現では，学習の収束を早くするために状態の数はなるべく少ない方が良い．一方，知覚入力の細かい違いによってタスクの遂行に影響が出る部分では，状態の形状をなるべく精密に表現できた方が良い．この両者は，相反するわけではないが，両立させるためには状態の形状を複雑にする必要があり，それはなかなか困難である．

分割表現として，最も単純でよく用いられているのは，入力空間を均等な（もしくは恣意的な）間隔の格子で分割し，離散化する方法である．この表現は，空間分割グリッド表現（またはグリッド表現）と呼ばれる（図 1.1d，図 4.2a 参照）．この表現では，入力空間は入力自身の類似性と設計者の恣意に基づいて分割される．入力空間は，タスクに合わせて，場所によって様々な粒度で分割される必要があるが，均等に分割する場合には，タスクに成功するために，最も細かいところに合わせて全体を分割しなければならない．その結果，余分な情報を大量に含むことになり，強化学習の収束が非常に遅くなることが問題となっている．また，恣意的に分割する場合にも，タスクにとって適切に分割することは設計者にとっても困難であること，および格子で分割するということが大きな制限となることなどから，強化学習の収束にかなりの時間がかかる場合があり，問題となっている [6, 7, 92]．この表現は，設計者によって固定的に定められるので，認識の学習の範疇には入らない．

分割表現を学習で構成する手法としては，報酬の類似度を考慮した手法と考慮しない手法の両方がある．上述のように，報酬の類似度を考慮した方が，タスク実行の観点から適切に抽象化を行うことが期待できる．この手法については，次の 4.4.2節で詳しく説明する．報酬の類似度を考慮しない手法としては，報酬が得られた地点の周りを細かく分割する手法（Kröse らの adaptive state space quantisation [56, 57]，畝見

の実例に基づく強化学習法 [122]），データの密度が高いところを細かく分割する手法（Moore の Variable Resolution Dynamic Programming, VRDP [77]，Kröse らの self-organizing state-space quantisation [55, 58]），入力データのクラスタリングを用いる手法（中村らの behavior-based map generation method [84]），入力される属性間の関係に基づいて分割する手法（高橋らの incremental state space segmentation [114]）などが提案されている．これらの手法では，報酬の面から見て同一とみなせる状態でも分割してしまうので，報酬の類似度に基づく分割法に比べて，抽象化のレベルが低くなりがちである．

陰的表現では，通常，各知覚入力状態のユーティリティを知覚入力の関数として表す．ユーティリティ関数が求まれば，行動の結果を 1 ステップだけ先読みすることによって最適の行動を決定することができるからである．ユーティリティとは将来の無限期間に渡る総報酬の期待値のことなので，ユーティリティ関数を学習することは，報酬の類似度を考慮した一般化を学習することに相当する．そこで，この表現を用いた抽象化については，次の 4.4.2節「報酬の類似度に基づく一般化」の中で詳しく説明する．

4.4.2 報酬の類似度に基づく一般化

報酬の類似度に基づく一般化は，タスクに応じて細かく見るところは細かく，粗くても良いところは粗く抽象化することができるので，強化学習の入力の一般化法として有望である．報酬の類似度に基づく一般化には，前節で述べたように，陰的表現を用いる場合と分割表現を用いる場合がある．

陰的表現による一般化は，通常，真のユーティリティ関数をニューラルネットワークを用いて近似することによって実現される．ニューラルネットワークは強力な学習能力をもつので，陰的表現では，通常，オンラインで学習が行われる．また，連続的な知覚入力ベクトルも扱うことができる（ただし，後述のように問題はある）．

この一般化のためのニューラルネットワークとしては，パーセプトロン型の階層的ニューラルネットワークが最もよく用いられる．この場合，ユーティリティ関数の近似は，各行動ごとにユーティリティの TD（4.3節参照）を逆伝播して学習される（誤差逆伝播学習）．この手法の例としては，まず，代表的な強化学習法の一つである actor-critic learning [14] が挙げられる．actor-critic learning では，入力の一般化のために 2 層のニューラルネットワークを用いている．また，有名な TD method [112] でも，ユーティリティの陰的表現が用いられている．具体的な陰的表現法は限定されていないが，文献 [112] では，2 層のニューラルネットワークを用いた場合の収束性の証明

などの理論的な解析が行われている．ただし，2層では表現能力も学習能力も限られるので，複雑な凹凸を持つ関数の近似は難しい．3層以上の多層ニューラルネットワークを用いた手法には，ニューラルネットワークとTD methodを組み合わせたTesauroのTD learning network（TD net）[116]，ニューラルネットワークとQ-learningを組み合わせたLinのconnectionist Q-learning（QCON）[63]，ニューラルネットワークとactor-critic learningを組み合わせたLinのconnectionist AHC-learning（AHCON）[63]，およびMartínらの手法[67]などがある．多層のネットワークの場合は，2層のネットワークと比べると，表現や学習の柔軟性は明らかに増すが，次のような問題点がある．

1. 最適性，収束性などの理論的な解析が非常に困難である．今のところは行われていない．

2. 層の数，各層のニューロンの数，ニューロン内部の関数の形，ニューロン間の結合の重みの初期値などパラメータが非常に多く，それらの設計法が確立していない．今のところは，それらを経験的に決めなければならない．

3. 複雑なタスクでは，連続的な知覚入力を適切に一般化することができない．複雑なタスクを扱う場合には，設計者によって定められた方法で，連続的な知覚入力を離散化したり[63]，離散的な知覚入力を二値化して[116]からニューラルネットワークに入力している．

4. 状態の内部表現がimplicitなので，抽象化された表現を記号として扱うことができない．

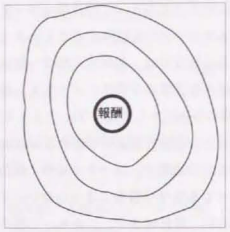
ユーティリティ関数の近似のためにニューラルネットワークを用いる手法には，パーセプトロン型のネットワークの他に，CMAC（Cerebellar Model Arithmetic Computer）を用いる手法（齋藤[104]）や，RBF（Radial Basis Function）を用いる手法（銅谷[26]）が提案されている．CMACはボックス型のステップ関数の重み付き和，RBFはガウス関数の重み付き和で関数を近似する方法である．従って，どちらも三層のニューラルネットワークの入力層と隠れ層の間の結合を固定したようなネットワークとみなすことができる．そのため，多層のパーセプトロン型ネットワークよりも扱いやすく，上述の問題点の1, 2に関しては改善される．しかし，表現や学習の柔軟性は劣るので，問題点の3に関しては悪化することが予想される．

次に，分割表現による報酬の類似度に基づく一般化，すなわち「報酬の類似度に基づく分割」について説明する．この報酬の類似度に基づく分割には，二つの方向がある．一つは，類似したユーティリティを持つ知覚入力を一つにまとめる方向の分割で，「ユーティリティに基づく分割（横の分割）」と呼ぶことにする．もう一つは，報酬に到達するための適切な行動が類似している知覚入力を一つにまとめる方向の分割で，「行動に基づく分割（縦の分割）」と呼ぶことにする．

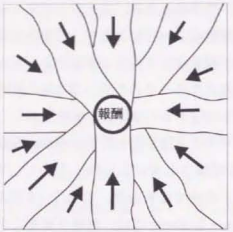
ユーティリティに基づく分割では，図4.3aに示すように，状態の境界はユーティリティの等高線となる．ユーティリティを適度に量子化すると，この表現はユーティリティ関数のよい近似となり，行動の結果を1ステップだけ先読みすることによって最適の行動を決定することができる．しかし，この表現では，各状態の形は一般に非常に複雑になる．そのため，第3章で説明した様々な認識の学習法によっても，状態の区別を学習するのは非常に困難である．この種の分割法としては，Yeeによって，ユーティリティを量子化するtop-down abstractionと，状態の認識を学習するaddress learningが提案されている[142]．しかし，まだ初歩的な段階で，ユーティリティ関数を近似する手法ではあるが，この二つを組み合わせても強化学習法にはならない．また，address learningでは，問題を簡単にするために，教師付きで，しかも，二値の知覚入力に限定して学習を行っている．

行動に基づく分割では，図4.3bに示すように，各状態は報酬への経路のような形になる．異なるユーティリティを持つ知覚入力が一つの状態に含まれるので，ユーティリティ関数の近似にはならない．従って，この分割法で離散化された状態表現の上で強化学習を行っても，マルコフ決定問題の最適政策が獲得できる保証はない．各状態は，報酬に到達するための適切な行動が類似している知覚入力を一つにまとめたものなので，あらかじめ適切な行動を持っており，通常は，この状態表現の上で強化学習を行う必要はない．利点としては，この表現を用いると，行動を時間的にまとめて抽象化することができる（7.2節参照）．状態の形状も，ユーティリティに基づく分割に比べると，まとまった塊となることが多い．そのため，抽象度の高い状態を簡単な表現法を用いて表すことが期待できる．

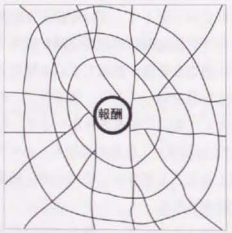
行動に基づく分割表現を構成するには，二種類の方法がある．一つは，報酬から逆向きに状態を構成していく方法である．この方法では，知覚入力空間中で，連続した同じ行動で報酬または既存の状態に到達する領域を新たな状態として切り出す．その結果，ある行動で報酬に到達する状態A，ある行動で状態Aに到達する状態B，ある行動で状態Bに到達する状態C…というように，徐々に状態空間が分割されていく．この方法で構成する場合は，最適な行動を獲得するというよりも，確実に報酬に到達するための行動を早く獲得するという学習になる．行動の最適性は重視しないので，通常の，マルコフ決定問題の解を求める強化学習よりも学習は簡単である．その分，連続的な知覚入力空間を分割するところに重点がおかれる．この種の手法として，浅



a. ユーティリティに基づく分割
（横の分割）



b. 行動に基づく分割
（縦の分割）



c. ユーティリティおよび行動に基づく分割
（縦横の分割）

図 4.3: 報酬の類似度に基づく分割

田らは超楕円体で分割する手法（action-based sensor space categorization）を提案している [6, 7, 92]．また，矢入らは重なりあったボックスで分割する手法（属性空間内サブゴール列学習）を提案している [129, 130]．どちらの手法も，ランダムな行動の結果を蓄えて，オフライン学習によって新しい状態を構成する．

もう一つの方法は，細かい粒度の状態表現の上で強化学習などを用いて最適な行動政策を獲得した後で，同じ最適行動を持つ隣あった状態を融合して大きな状態を構成する方法である．この種の手法として，Simons らは，知覚入力空間をタスクに合わせて場所によって適切な大きさのボックスで分割して，行動政策を学習した後で，ボックスを融合して複雑な形状の状態を構成する手法を提案している [111]．この手法では，行動学習の後で状態を融合するので，学習を促進する効果は全くないが，状態の内部表現がコンパクトになり，状態認識の計算が速くなることが期待できる．また，状態を融合することによって，状態の持つ意味を明確にすることができる．この手法はオフライン学習であり，状態表現は学習後に固定されてしまう．

ユーティリティに基づく分割と行動に基づく分割を共に行う分割表現もある．この表現は「ユーティリティおよび行動に基づく分割（縦横の分割）」表現と呼ぶことにする．この表現を用いると，ユーティリティに基づく分割と同様に，ユーティリティ関数を近似することができるので，最適政策を追求する強化学習を行うことができる．また，行動に基づく分割と同様に，各状態が図 4.3c に示すような適度にまとまった形になり易いので，状態を簡単な表現法で表すことが期待できる．しかし，上の二つの分割表現よりも抽象化のレベルは低くなる．この表現においては，各状態は，他とユーティリティかつ行動によって区別される領域なので，行動の評価値である Q-value を学習する Q-learning と特に相性が良い．次に紹介する五つの手法は，すべて Q-learning を行いながら，Q-value の類似度を基に状態を分割または融合する．Chapman らの G algorithm では，二値の入力空間を軸に垂直な超平面で階層的に分割していく [24]．Mahadevan らは，二値の入力空間において Q-value が類似した状態を統計的クラスタリングによって融合していく手法を提案している [66]．石黒らは，連続的な入力空間を empirically obtained perceiver（EOP）と呼ばれる線形判別関数を用いて，超平面で階層的に分割していく手法を提案している [44, 107]．橋本らの CIQ-learning（concept-intensive Q-learning）では，概念クラスタリング（COBWEB．3.2.2節参照）によって求めた知覚対象の概念木を利用して，離散的な入力空間中の Q-value のばらつきが大きい状態を分割する [109, 48]．これらの学習法は，Q-learning を行いながら状態分割（または融合）を行うという点ではオンライン学習であるといえるが，分割の境界線を固定してしまうので，状態の形状の柔軟な調整はできない．

次に，報酬の類似度に基づく一般化の陰的表現と分割表現の比較をする．上述の陰

的表現の問題点は，分割表現を用いた場合にはいくらかやわらげることができる．「1. 最適性，収束性の証明が困難である」に関しては，分割表現でも同様に問題となる．しかし，状態の形状がはっきりしているので，学習がどのように進んでいるのかが把握しやすい．また，統計的な分割法を用いている場合には，理論的な解析もいくらか可能である．「2. 設計法が確立していない」に関しては，分割表現を用いた手法では，通常，カテゴリの数や粒度の限界などは固定されておらず，表現能力の限界を定めるような決定的なパラメータは，あまり用いられていない．すなわち，状態内で行動の結果がばらつくようならば，いくらでも分割して，適度な抽象化のレベルを達成することができる．そのため，除的表現を用いる場合よりも，パラメータ設定の苦労は少ないと思われる．ただし，最適性を追求するためには，分割表現でもパラメータの設定に十分な労力をかける必要がある．「3. 複雑なタスクで連続的な知覚入力を扱えない」に関しては，いくつかの研究 [7, 44, 111, 129] で連続的な入力空間の分割の問題を扱っている．それぞれの分割法で複雑な問題でもうまく状態が切り出し得るか，除的表現と比べてどちらの表現能力および表現学習能力が優れているかは今後の検討課題である．「4. 記号を抽出できない」に関しては，分割表現では，切り出した状態を記号として扱うことができる．状態を記号として強化学習以外の目的に使用している研究はまだないが，人間が状態に意味づけができる程度の抽象化は実現されている．また，階層的な状態を切り出す手法 [24, 44, 48, 66, 109, 142] は，概念の学習とみなすこともできる．

逆に，分割表現が除的表現に劣る点としては，一つには，分割表現を用いた場合には切り出した状態が空間的に不連続であるということが挙げられる．そのため，連続的な関数を近似した時にはどうしても埋められない誤差が生じる．もう一つの劣る点として，分割表現を用いた手法の方が，学習後の表現が固定的であるということが挙げられる．ここに挙げたいくつかの分割法では，オフライン学習しか許していないし，その他の分割法でも，一度分割した状態の境界を修正する方法は含まれていないので，抽出された状態の形状が固定されてしまう．そのため，環境が変化する場合や学習に用いたデータに偏りがあった場合には初めから学習し直さなければならない．

4.5 本研究における強化学習システム

本研究では，フレーム問題の現実的な解決に近付くために認識の学習と認識された表象の上での学習を同時に行う認知行動システムを開発することを目標としている．強化学習は，そのうちの認識された表象の上での学習に相当する．本研究で強化学習を用いるのは，強化学習が，認識された表象の上での学習としては，認識の学習と組

み合わせる上で簡単で扱いやすいからである．また，認識の学習のための指標として，強化学習で用いられている報酬を利用することができるからである．報酬はタスクを端的に表現しているので，認識の学習のための指標として用いるのに都合が良い．しかし，強化学習システムは，記号として「状態」しか用いることができないので，記号処理システムとしては非常に単純なものである．将来的には，さらに複雑な記号処理システムと認識の学習を組み合わせる方向に拡張していく必要がある．

強化学習システムと組み合わせる認識の学習としては，フレーム問題を現実的に解決するという観点から考えると，4.4.1節で述べた状態認識の学習の三つの軸において，連続入力から，オンラインで，報酬の類似度に基づいて学習を行うことが望ましい．連続入力が可能であるという性質は，実世界からの知覚入力のためには欠かせない．オンラインで学習を行うという性質は，環境中で未知の状況に遭遇した時に徐々に適応するという機能のために欠かせない．また，強化学習においてはタスクは報酬の中に表現されているので，入力空間の分節のための指標として報酬を用いることは，タスクの遂行にとって適切な抽象化を行うために望ましいといえる．

強化学習のための状態表現には，4.4.1節で述べたように，分割表現と除的表現の二種類がある．状態表現として分割表現を用いた場合には，抽象化として記号化を行うことに相当する．一方，除的表現を用いた場合には，パターン表現のままで抽象化を行うことに相当する．抽象化として記号化を行った方が，抽象化の処理の部分と表象の上での処理の部分がはっきりと分けられるので，

1. 抽象化された表象に意味付けしやすい．
2. 情報処理の過程がわかりやすい．
3. 知識の再利用がしやすい．
4. 既存の強力な記号処理手法を用いることができる．

などの利点が得られる．そこで，本研究では，強化学習システムの状態表現として，分割表現を用いることにする．

結局，本研究で扱う強化学習システムは，報酬に基づく抽象化の学習，連続入力，オンライン学習，状態の分割表現という特徴を持つといえる．それは，図4.4のように表すことができる．

報酬に基づく抽象化の学習を行う強化学習システムは，4.4.2節で述べたように数多く存在するが，連続入力，オンライン学習，状態の分割表現ということを考慮に入れると，そのほとんどが除かれてしまう．上述した手法の中で，唯一該当するのが，石黒らの EOP を用いる手法 [44, 107] である．しかし，この手法では，タスクを遂行し

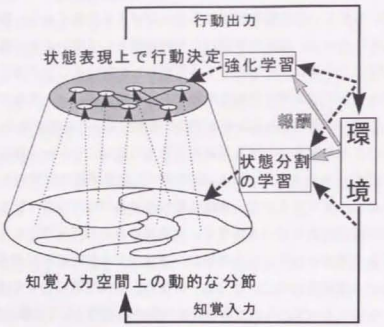


図 4.4: 認識の学習を含む強化学習システム

ながら状態を分割することは想定しているが，一度分割した境界線は固定してしまうので，不慣れな環境や環境の変化に適応するための柔軟性にはやや欠けるといえる[1]．本研究では，タスクを遂行しながら状態を切り出し，切り出した後も状態の形状を調整し続けるような認識の学習を行うことにする．この柔軟性は，不慣れな環境に徐々に慣れてフレーム問題に悩まされないようになるために非常に重要である．次章では，そのような認識の学習を行う強化学習システムとして本研究で開発した状況遷移ネットワークシステム（Situation Transition Network System，STNS）について説明する．

[1]一方，柔軟性を犠牲にすることによって，安定して緻密な状態表現を獲得することができる．石黒らの研究では，それを利用して，画像入力という非常に多次元の入力からの状態表現の獲得を実現している．

第 5 章

状況認識と行動規則の同時学習

この章では，まず本研究で開発した状況遷移ネットワークシステム（Situation Transition Network System，STNS）の概要について説明し，続いてそのシステム中で用いる二つの学習法について順に説明する．

5.1 状況遷移ネットワークシステム – STNS

STNS は，図 5.1に示すように，状況認識器（situation classifier）と，状況遷移ネットワーク（situation transition network, STN）と，複数の行動モジュール（behavior module）からなる．一回の動作では，センサから与えられる連続的な知覚入力から状況認識器を用いて現在の状況を認識し，状況遷移ネットワーク上で部分的プランニングを行って，一つの行動モジュールを作動させる．

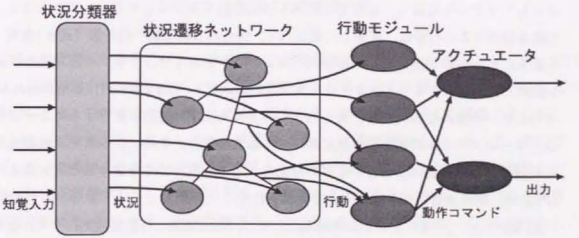


図 5.1: システムの構成

STNS は，環境中で何か評価し得る出来事が起こった時，すなわち，いくつかの特

定の知覚入力状態から特定の行動をして特定の知覚入力状態に遷移した時に，環境から報酬を得る．報酬というのは，一般に強化学習で用いられている報酬と同様に，数値で与えられる評価のことである（4.1節参照）．

STNS は，この報酬に基づいて，状況認識の学習と行動規則の学習の両方を行う．そのため，STNS は，第4章で説明した状態認識の学習を行う強化学習の一種であるとみなすことができる．本研究では，状況認識の学習のために「Adaptive Situation Recognition based on Rewards（ASRR）」を，行動規則の強化学習のために「インターリーブプランニングに基づく強化学習（Interleave Planning-based Reinforcement Learning，IPRL）」を提案する．

ASRR では，知覚入力，報酬，および行動の類似度に基づいて同一とみなし得る領域，すなわち「状況」を連続的な知覚入力空間から切り出し，その後の経験に基づいて状況の形状を調整しながら維持する．そのため，状況は，4.4.2節で述べた「行動に基づく分割表現」かまたは「ユーティリティおよび行動に基づく分割表現」で表されることになる．この分割表現では，固有の意味を持った抽象度の高い領域を表すことができる．通常の強化学習で用いられる抽象度の低い「状態」と区別するために，ASRR では，この領域のことを「状況」と呼んでいる．ASRR では，一つのデフォルトの状況から，報酬の見込みに基づいて状況抽出を繰り返し行うことにより，状況遷移ネットワークを構成する．この過程は報酬を上流に逆伝播させながら状況認識を学習しているとみなせる．ASRR は，状況認識器上で実現される．

IPRL では，ASRR によって抽象化された状況表現の上で行動規則の強化学習を行う．具体的には，状況間の遷移確率と各遷移で得られる報酬の期待値を最尤推定してワールドモデルを構成し，その上で前向きの部分的プランニングを行うことによって行動を決定する．行動は，探索した範囲内で，得られる総報酬の期待値（式4.1参照）を最大にするものが選ばれる．部分的プランニングとしては，プランの成功確率が高い範囲だけを探索する「インターリーブプランニング」（5.3.2節参照）が用いられる．そのため，不慣れな環境でも，考え込まずにとりあえず即応的に反応することができる．ワールドモデルは過去の経験に基づいて最尤推定されるので，システムは環境中で行動しながら，各状況においてどの行動をとるのが良いかを学習することになる．IPRL は，状況遷移ネットワーク上で実現される．

STNS では，ASRR と IPRL を組み合わせて，状況認識と行動規則を，タスクを実行しながら同時に学習する．そのため，状況の形状が適切であれば行動はタスクを成功させるし，行動がタスクを成功させ続ければ状況の形状が適切に維持される．この状況と行動の相互依存関係によって，タスクに特化しながら環境やタスクの多少の変化に柔軟に対応できる認知行動システムが実現できる．また，このオンライン学習に

よって，入力空間中の，タスクを実行する上で頻繁に経験する領域について，特に詳しく学習することができるので，学習が効率的に進む．

ASRR と IPRL における学習は，図5.2に示すように，システムがタスクを実行しながら観測した知覚入力（センサ値），認識された状況，選択された行動，得られた報酬をデータとして行われる．このデータは，一定期間，履歴データベースに蓄えられて，その後，消去される．そのため，環境やシステムの内部表現が変化した場合にも，古過ぎるデータに惑わされることなく柔軟に対応することが期待できる．

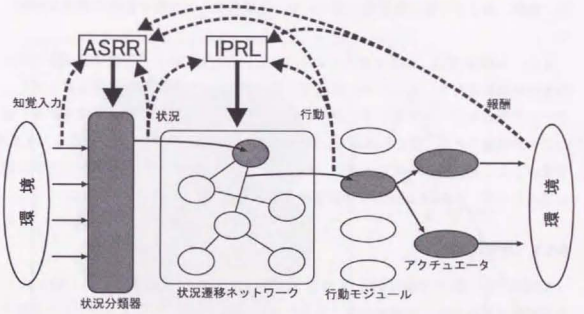


図 5.2: 学習のための情報の流れ

以上で説明したように，4.4.1節で述べた状態認識の学習の三つの軸においては，STNS は，連続入力から，オンラインで，報酬の類似度に基づいた学習を行うシステムであるといえる．この意味では，本研究は，最も柔軟でタスクに適応した抽象化を目指している研究であるといえる．また，状況の表現として分割表現を用いるので，除的表現を用いた手法よりも扱いやすく，抽出した記号を知識の共有などの強化学習以外の情報処理に利用することが期待できる．分割表現を用いた手法では，抽出された状態が固定的になってしまうことが問題点の一つであったが，STNS は，状況を切り出した後もその形状を調整し続ける点で他の分割表現の手法とは異なり，より柔軟な学習を可能にすることが期待できる．この柔軟性は，不慣れな環境に徐々に慣れてフレーム問題に悩まされないようになるために非常に重要である．しかし，フレーム問題に対処するために状態認識の学習を行う研究はまだ始まったばかりであり，抽出された

記号を用いて複雑な情報処理を行うような段階には達していない．本システムで扱う環境やタスクも非常に簡単なものである．本研究は記号としての状態の認識の学習を，既存の学習システムよりも柔軟にするための基礎的研究であると位置付けることができる．

5.2 状況の認識とその学習 – ASRR

STNS では，ASRR を用いて状況の認識とその学習を行う．ASRR では，知覚入力，報酬，および行動の類似度に基づいて，連続的な入力空間を複数の状況に分割する．

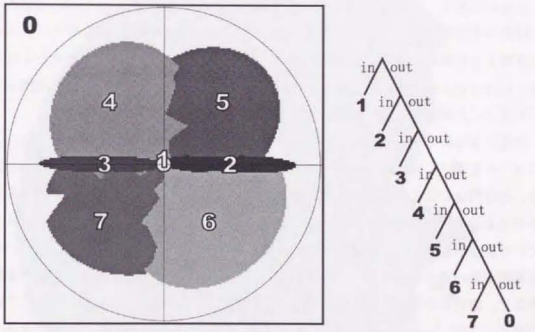
また，ASRR では，前節で述べたように，状況と行動の相互依存関係によって状況の形状の調整を行う．そのため，状況を，データが不十分な早期の段階で切り出し，データが増えるにつれてタスクに適応させることができる．その結果，学習が速く進むことが期待できる．また，状況を，環境の変化にも対応させることができる．4.4.2節で述べたように，強化学習の入力の一般化法として状況の形状の調整を行う手法は他に見当たらず，ASRR が斬新な手法であるといえる．

5.2.1 状況の認識

ASRR では，入力空間は図 5.3a のように分割される．状況の認識には，図 5.3b に示す判別木を用いる．判別木の各ノードにおいては，現在の知覚入力が，入力空間中の各ノードが対応する領域に入るか，入らないかで判別を行う．各状況は重なりあっていて，上にある状況から順に判別が行われる．状況として切り出されていない余白部分の領域は，状況 0 と名付ける．

ASRR においては，状況 0 以外の各状況が固有の意味（特定の行動とその特定の結果）を持っている．つまり「その状況内から特定の行動をすると，特定の結果が得られる」という意味である．特定の行動のことを「条件行動」と呼ぶことにする（状況の意味の中で，「条件部」となっている行動だからである）．特定の結果というのは，「大きな正の報酬が得られる」という結果と，「他の特定の状況（「親状況」と呼ぶ）に遷移する」という結果の二種類がある．前者の結果を意味として持つ状況を R 状況（Reward-based Situation），後者の結果を意味として持つ状況を T 状況（Transition-based Situation）と呼ぶ．T 状況は報酬に直接的に依存していないが，状況の抽出はまず R 状況から始まるので，帰納的に報酬に依存しているといえる．

状況 0 以外の状況の形状は，その状況の意味に対して定められる正事例と負事例によって決定される．正事例とは，履歴データベース中の全データのうち，条件行動を



a. 入力空間（2 次元入力の場合）　　b. 判別木

図 5.3: 入力空間と判別木（0 ～ 7 は状況を示す）

して特定の結果が得られたデータのことである．負事例とは，その状況に分類されたデータのうち，条件行動をして特定の結果が得られなかったデータのことである．

STNS では，タスクを実行しながら学習を行い，その際に頻繁に経験される入力空間中の領域を，ASRR を用いて状況として抽出する．従って，状況の正事例は，タスクを達成する過程で経験される知覚入力状態である．また，負事例は，「この状況でこの行動をとったらこういう結果が得られて報酬に近付くことができる」と期待してその行動をとった時に，期待に反する結果が得られた場合の知覚入力状態である．すなわち，タスクに失敗した時の知覚入力状態である．従って，システムが良い性能を示している時には，大量の正事例と，小量の負事例が集まり続けることになる．

状況の形状は，図 5.4に示すように，「齧られた超楕円体（bitten hyper-ellipsoid）」と呼ぶべき形になる．これは，図 5.5に示すように，正事例集団からのマハラノビス距離による巨視的な認識と，NN 識別法による微視的な認識を組み合わせたものである．超楕円体とは，多次元空間内の楕円体のことで，入力空間中で正事例集団からのマハラノビス距離（後述）がある値以下の領域に相当する．

状況に入るかどうか判別する際には，まず現在の知覚入力が超楕円体の内側に入るかどうかを調べ，内側に入る場合は NN 識別法によりその状況に入るかどうかを調べる．超楕円体による巨視的な認識法では，様々な形状，姿勢の超楕円体を用いることができるので，かなり柔軟に状況を表現することができる．また，簡単なパラメータだけで表現されるので，認識のための計算と学習のための計算が非常に速い．しかし，境界面が整い過ぎているので，図 5.5a に示すように，誤分類を避けることができない．そこで，微視的な認識法を組み合わせる必要がある．巨視的な認識法としては，3.2.1節で述べた 2 群（および多群）を分類するための様々な手法を用いることができる．しかし，STNS では，上述のように，正事例を多く集め続けることが良い性能に直結するので，通常，負事例よりもかなり多くの正事例が得られる．また，負事例は，その状況に分類されたデータのうちの失敗例のことなので，その状況の周辺でしか集まらず，負事例集団の分布は図 5.5に示すようにドーナツ型になる．そこで 2 群を同等に扱うような分類法よりも，正事例集団の分布に基づいて超楕円体で区別する手法の方が適していると思われる．

NN 識別法による微視的な認識では，3.2.1節で述べたように，非常に精密に判別することができる．また，学習としては事例の位置を記憶するだけなので，非常に速い．反面，記憶容量と認識のための計算時間がかなり大きいということが欠点である．ASRR では，NN 識別法を用いる前に巨視的な認識で足切りを行うことによって，属する見込みの無い状況で NN 識別法を行う手間を省くことができる．これは，計算時間をかなり短縮する効果がある．また，上述のように，状況は大量の正事例を小量の負事例

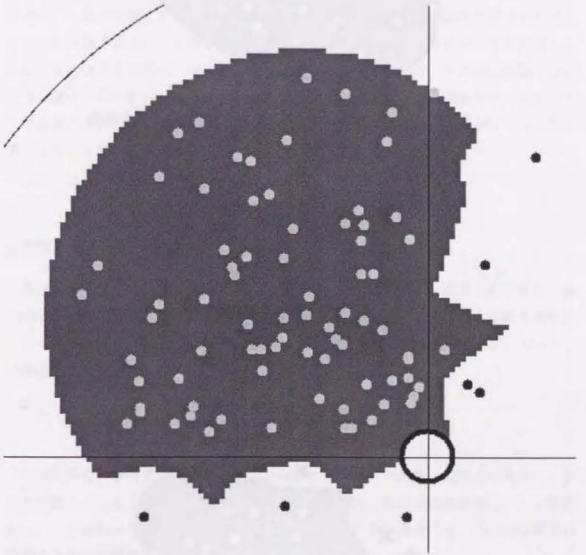
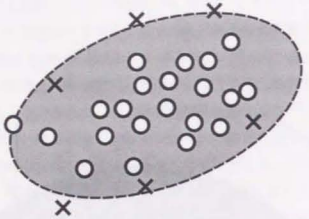
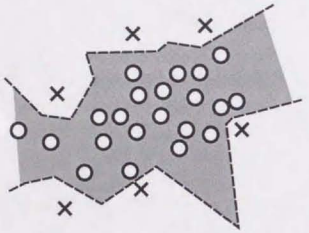


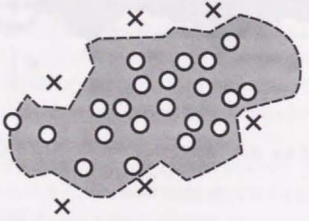
図 5.4: 状況（薄灰色点は正事例，黒点は負事例）



a. 正事例集団からのマハラノビス距離を用いた巨視的な認識



b. NN 識別法による微視的な認識



c. a と b を組み合わせた ASRR による認識

図 5.5: ASRR による認識 (○は正事例，×は負事例)

で包み込むような形になるが，負事例が不足すると，図 5.5b に示すように，NN 識別法だけではあちこちに破れが生じる．巨視的な認識を組み合わせることによって，図 5.5c に示すように，この破れを補うことができる．

ASRR では，このように多くの事例（特に正事例）によって状況の形状を維持している．これらの事例は，システムが報酬を集める過程で頻繁に経験されたものである．そのため，ASRR における状況は，報酬に到達するために通過する経路としての役割を持つ．人間が認識して利用していると思われる状況には，この報酬に到達する目印としての状況のほかに，危険な場面を避けるための警告としての状況があると考えられる．そのような状況は，一度大きな罰を受けた時に記憶して，その後は経験しないように避けながらしかも維持しなければならない．ASRR では，現在のところそのような仕組みを持っていないので，環境から大きな負の報酬が与えられるようなタスクは，うまく扱うことができない．

次に，巨視的な認識と微視的な認識について，順に細かく説明する．

正事例集団からのマハラノビス距離を用いた巨視的な認識

集団と点の間のマハラノビス（汎）距離とは，集団の平均から点までの距離を，集団の標準偏差で標準化した距離のことである．点の集団と任意の点との距離を表すのによく用いられる．集団の平均を$\boldsymbol{\mu}$，分散共分散行列を$\boldsymbol{\Sigma}$とすると，任意の点$\boldsymbol{x}$からこの集団までのマハラノビス距離 D は，次式で表される．

$$D^2 = (\boldsymbol{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu})$$

この式からわかるように，ある集団からのマハラノビス距離一定の点の集合は，多次元空間上ではある傾きを持った超楕円面で表される．集団が正規分布している場合には，この面は分布密度の等高線である．正規分布でない場合にも，この面は集団の輪郭の大まかな特徴を表している．

正事例となるべき知覚入力の母集団の分布は明らかでないので，平均と分散共分散行列は，正事例集団から求まる不偏推定量を用いる．そして，正事例集団から最も離れた正事例までのマハラノビス距離を境界とする超楕円体を作る．正事例が十分多く存在する場合には，この超楕円体は，母集団の大まかな輪郭を表している．従って，母集団のほとんどをその内部に含み，かつ分布密度の低い場所をあまり含まないことが期待できる．この手法による認識は，母集団の分布が正規分布に近いほど正確である．

NN識別法による微視的な認識

NN識別法による認識では，正事例と負事例を合わせたデータ集団のうち，現在の知覚入力に最も近いデータが正か負かで判別を行う．距離は，NN識別法で通常用いるユークリッド距離ではなく，正事例集団の標準偏差で標準化した距離を用いる．この距離も（2点間の）マハラノビス距離と呼ばれている．正事例集団の標本分散共分散行列を$\boldsymbol{\Sigma}$とすると，任意の点$\boldsymbol{x}$からある事例$\boldsymbol{i}$までのマハラノビス距離dは次式で表される．

$$d^2 = (\boldsymbol{x} - \boldsymbol{i})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{i})$$

この距離を用いることにより，正事例集団を主成分分析にかけた時に寄与率の大きい主成分の方向ほど，各事例の影響を強く伝播させることができるようになる．

認識は次の手順で行われる．

1. 現在の知覚入力から最も近い負事例までのマハラノビス距離を計算する．
2. 現在の知覚入力から正事例までのマハラノビス距離を順に計算し，最近接負事例よりも近い正事例が一つでも見つかると現在の知覚入力はその状況に含まれると判断する．
3. すべての正事例が最近接負事例よりも遠い場合，現在の知覚入力はその状況に含まれないと判断する．

ASRRでは，上述のように，負事例は正事例に比べて十分少ないことが期待できるので，この手法によって，単純にすべての正事例，負事例を調べるよりも計算時間がかなり抑えられる．

5.2.2 状況認識の学習

ASRRでは状況の抽出と動的な維持によって，状況認識の学習を行う．

状況の抽出

ASRRにおける状況抽出の条件は，次の二つである．

1. **R状況** 全入力空間中で，ある特定の行動によって特定の大きな（r^R_{min}以上）報酬が得られたデータが，N^R_{init}個以上存在する．

   T状況 状況0の中で，ある特定の行動によって状況0以外の特定の状況に遷移し，大きな（r^R_{min}以上）報酬が得られなかったデータが，N^T_{init}個以上存在する．

2. 既存の状況の中に，同じ意味を持つ状況が存在しない．

R状況の抽出条件1は，正事例となり得るデータがたくさんある場合に状況を抽出するという単純なものである．それに比べると，T状況の抽出条件1は少し特殊である．T状況の抽出条件1で，データを数えあげる領域を状況0の内部に限っている理由は，次のように説明される．T状況の抽出の際にR状況と同様に正事例となり得るデータを全入力空間中で数えると，多くの状況が大きく重なり合って存在することになる．すると，他の状況に大きく覆われた状況では，状況遷移確率を正しく計算できなくなってしまう．それは具合が悪いので，T状況の正事例となり得るデータは，状況0の中でのみ数えることにしている．同じくT状況の抽出条件1で，状況0に遷移するデータを数えない理由は，状況0は固有の意味を持たない「余白」の状況で，状況0から適切な行動を選択することは難しいからである．状況0に遷移する領域を切り出しても，あまり意味が無く，他の適切な意味を持つ状況の抽出や維持を阻害するので，かえって有害である．また同じくT状況の抽出条件1で，大きな報酬が得られるデータを数えない理由は，大きな報酬が得られるデータは，R状況で確実にカバーするので，さらにT状況でカバーする必要がないからである．このようなデータを，無理にT状況としても抽出した場合，報酬に基づくR状況と状況遷移に基づくT状況が大きく重なりあって存在することになる．これは，上述のように，状況遷移確率を計算する上で具合が悪い．

抽出条件2で，同じ意味を持つ複数の状況の抽出を禁じているのは，同じ意味を持つ状況が複数存在しても，完全に重なりあってしまい無意味だからである．

上の二つの条件を満たした領域は状況として抽出される．抽出する際に，抽出条件中の特定の行動を「条件行動」にして，特定の行動の結果と共に，新たな状況の固有の「意味」として定められる．そして，その意味にしたがって，履歴データベース中に含まれる正事例となり得るデータを，すべてその状況の正事例集団に集める．T状況の正事例も入力空間全体から集める．負事例は，抽出時には存在しない．それは，負事例とは，その状況に分類されたデータのうち条件行動をして適切な結果が得られなかったデータのことだからである．過去のデータの中には，新たな状況の領域内から条件行動をして，新たな状況の意味に反する（負事例の意味に適合する）結果が得られたデータも存在するかも知れない．しかし，状況を切り出すことによって，同じ知覚入力状態から行動を始めても，状況を切り出す前とは異なる結果になり得る（6.1.2節参照）ので，そのようなデータを負事例として集めることはしない．ただし，行動が固定長である場合には，状況を切り出しても行動の結果が変わらないので，履歴データベースから負事例となり得るデータを集めても良い．

ASRRでは，入力空間全体が状況0であるところから学習を始めて，ある行動で報酬に到達するR状況A，ある行動でR状況Aに到達するT状況B，ある行動でT状況Bに到達するT状況C…というように，次々に状況を抽出し，入力空間を分割しながら学習が進む．

状況の維持

状況を動的に維持するために，次の四つの手法を用いる．

正事例集団，負事例集団の更新

新しいデータは，そのデータが正事例になり得るすべての状況の正事例集団に加える．また，そのデータが属する状況の負事例になり得る場合には，その状況の負事例集団に加える．新しいデータが得られると履歴データベース中の一番古いデータが消去されるが，その時に，各状況の正事例集団，負事例集団からもそのデータが除かれる．このように正事例集団，負事例集団が更新されるたびに，状況の形状が変化する．

超楕円体の大きさの更新

正事例集団から各正事例までのマハラノビス距離を定期的に調べて，最も遠い正事例までの距離を超楕円体の境界とする．

状況の重なり方の更新

状況の重なり方（判別木のノードの順序）は，強化学習における各状況のユーティリティを定期的にチェックして，ユーティリティが大きい状況が上に来るように並べ替えられる．これは，複数の状況が重なる部分では，報酬に到達するために複数の経路が存在するが，なるべく報酬に近い経路を選択した方が良いからである．

新しい状況が抽出された時にも，その状況を含めて並べ替えを行う．

状況の意味の切り替えと状況の削除

次の三つの場合には，状況の意味の切り替えを試みる．

1. 正事例の数が，R状況の場合は N^R_{min} 個，T状況の場合は N^T_{min} 個を下回る場合
2. T状況に含まれるデータのうち，状況の意味に適合する（条件行動で親状況に遷移する）データの R^T_{min} 倍以上のデータが，それ以外のある意味に適合する場合

3. T状況において，親状況が消滅した場合

1は，正事例が少な過ぎて，適切な状況の形状を維持できない場合である．2は，状況の意味が，タスクを遂行する際の事実上の意味とずれてしまっている場合である．STNSでは，状況は固有の条件行動とその結果に基づいて維持されているが，実際の行動の選択は強化学習によって行われている．従って，状況を維持するための意味が，タスクを遂行する際の事実上の意味とずれてしまうことがある．そのようなずれが見つかった場合には，意味を切り替えて，状況の形状も事実上の意味での事例に基づいて維持すべきある．3は，T状況の親状況が消滅してしまって，それ以降，状況を維持するための正事例が集まらなくなってしまった場合である．

意味を切り替える場合には，必ずT状況に切り替えることにする．そして，新しい条件行動として，その状況からの行動のうち最も頻繁に実行された行動を選び，新しい親状況として，その条件行動で最も多い遷移先の状況を選ぶ．こうして選ばれた新しい意味が，次の五つの条件のいずれかに適合する場合は，その意味は不適切であるとして，意味の切り替えを行わずにその状況を削除する．五つすべてに適合しない場合に，その新たな意味に切り替える．

1. 親状況が，状況0である場合
2. 親状況が，その状況自身である場合
3. 親状況が，その状況自身を親状況とするT状況である場合
4. 同じ意味を持つ状況が，既に存在する場合
5. 履歴データベース中の新しい意味において正事例となり得る事例の数が，N^T_{init} 個を下回る場合

1で状況0に遷移するT状況を作らない理由は，抽出条件でも説明したように，状況0が固有の意味を持たない「余白」の状況だからである．状況0から適切な行動を選択することは難しいので，状況0に遷移するT状況は有害になり得る．2で「その状況自身」に遷移するT状況を作らない理由は，明らかに不必要なループを形成するからである．状況間の遷移がループを形成しても，そのループの中で大きな報酬が得られれば有用である．繰り返し大きな報酬が得られる領域にはR状況が抽出されるので，そのような有用なループの中には必ずR状況が含まれる．2の場合は明らかにR状況を含まないので，不必要なループとなる．3で「その状況自身を親状況とするT状況」に遷移するT状況を作

らない理由も，同様に，明らかに不必要なループを形成するからである．4で同じ意味を持つ複数の状況を作らない理由は，上述の抽出条件2に関して説明したように，同じ意味を持つ状況は完全に重なってしまうので無用だからである．5は，上述のT状況の抽出条件1に準じている．

R状況への意味の切り替えを行わない理由は，次のように説明される．まず，R状況への意味の切り替えを試みる際に，T状況と同様に上述の状況削除条件4, 5を考慮に入れることにする．条件5に従うと，新たなR状況となるためには，正事例となり得るデータが N^R_{init} 個以上必要である．しかし，上述のR状況の抽出条件より，そのような場合には，必ず既にその意味を持つR状況が抽出されているはずである．従って，条件4よりその状況は削除される．このように，R状況への意味の切り替えを試みる必要はないといえる．

状況の意味を切り替えた時には，状況の形状が不連続的に変化するのを避けるために，切り替える以前の正事例集団，負事例集団をそのまま保つことにする．その後に得られるデータから，新たな意味に基づいて正事例，負事例を集める．

また，政策にしたがって行動した場合に期待される報酬が負である状況は，維持しても意味がないので削除する．具体的には，各状況のユーティリティを定期的に調べて，ユーティリティが負である状況を削除する．

5.2.3 行動の時間的な離散化と状況表現

この節では，システムが出力する行動の性質と状況表現との関連について説明する．システムの出力は与えられるタスクに深く依存する問題であって，あまり一般的に扱うことはできないが，システムの出力のある性質が，ASRRによる状況分割表現に深く関連している．その性質とは，行動をどのようにして時間的に離散化しているかという性質である．

認知行動システムが出力する行動は，通常，時間的に連続な行動である．ロボットが前進する場合も，ロボットハンドで積木を積む場合も，最終的なモーターの出力は連続的である必要がある．しかし，強化学習やプランニングで行動を扱う場合は，通常，行動が時間的に離散化されている必要がある[1]．そのためには，連続的な行動を適当にまとめて離散化する必要がある．

「積木を積む」とか「ボールを投げる」というそれだけで完結する行動に関しては，全体を一まとめにして離散化することが多い．そして，その一まとめの行動を実現するための運動制御の問題と，一まとめにした行動を用いたプランニングなどの問題（例えばいわゆる「積木の世界」の問題）に分けて扱われる．

[1]強化学習において時間的に連続な出力を扱う研究もある [26].

一方，「前進する」とか「回転する」という行動に関しては，普通は終わりが定められていない．このような行動を扱う場合によく用いられるのが，一定のタイムステップの間はその行動を続けて，タイムステップの区切りと同時に行動も終了させるというアプローチである．この場合，行動が一定の長さに区切られることになるので，「固定長行動アプローチ」と呼ぶことにする．固定長行動アプローチは，すべての行動を等質に扱うことができる点で優れている．例えば，「3回前進して，2回右にまわって，さらに5回前進する」という行動列で，10タイムステップ必要とすることも意味することができる．さらに，同じ報酬が得られるならば，行動回数が少ない方がよいと単純に考えることができる．一方，作業空間中の各点を，タスクに関係なく異質化してしまうという欠点がある．一回の行動で移動できる点も限られ，ほんの近くの点であっても，行動の区切りに合わなければ，移動するのに非常な遠回りを強いられたり，遠回りしても行きつけなかったりする．複雑なタスクを実行する際にこのような事態が起こるのを避けるためには，行動を非常に細かく離散化する必要がある．

もう一つのアプローチは，終了条件（または継続条件）を定めて，その条件が満たされるまで（または継続条件が満たされている間だけ）行動を継続するというアプローチである．これを，「条件つき行動アプローチ」と呼ぶことにする．このアプローチでは，固定長行動アプローチのように行動を等質に扱うことはできないし，終了条件を扱うタスクやシステムに依存して考案する必要がある．終了条件としては，例えば移動行動に関しては，「CLOSEな空間からOPENな空間に出たとき」または「OPENな空間からCLOSEな空間に出たとき」などを用いることができる [10]．このアプローチの利点としては，タスクごとに終了条件を適切に設定してやれば，行動をかなり大きな塊として扱うことができる点があげられる．これも作業空間の異質化であるが，タスクに適した異質化であれば，上述のような問題は起こらない．この行動の表現によって，情報は非常にコンパクトに圧縮され，情報処理も容易になる．例えば，連続的な空間であっても，「三つめの信号まで直進して，角を右折して，ガソリンスタンドが見えるまでさらに直進する」というような抽象化のレベルの高い表現が可能になる．

次に，この二つのアプローチとSTNSとの関連について述べる．STNSでは，時間的に離散化された行動を出力する．出力された行動がどのように終了するのかは，行動の定義の中に，すなわち行動モジュールの内部の制御に押し込めてしまっている．従って，固定長行動であっても，条件つき行動であっても，または「積木を積む」といった抽象度の高い行動であっても，STNS自体は同様に扱うことができる．

しかし，出力する行動の性質によって，ASRR による状況表現の性質は異なってくる．固定長行動を用いた場合には，4.4.2節で述べた三つの分割表現のうちの「ユーティリティおよび行動に基づく分割表現」に相当する状況表現になる．それは，各行動が一定時間で終了し，報酬も式4.1に示すように，一定の離散時間ごとに割り引かれるからである．ASRR によって，同じ行動で同じ状況に遷移する知覚入力状態を一まとめにしても，図5.6に示すように，この行動の長さ分ずつ鱗状に入力空間が分割される．

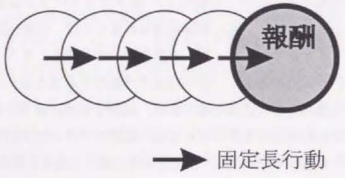


図 5.6: 固定長行動に基づく入力空間分割

一方，条件つき行動を用いた場合には，4.4.2節で述べた三つの分割表現のうちの「行動に基づく分割表現」に相当する状況表現になる．それは，終了条件をうまく定めてやることによって，入力空間中の最適行動を同じくする領域を一まとめにすることができるからである．ASRR では，各行動の終了条件を「報酬が得られたとき」と「他の状況に遷移したとき」などと定めることによって，図5.7に示すように，行動の経路に沿った領域を一つの状況として抽出することができる．

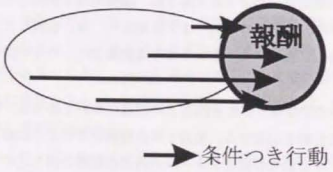


図 5.7: 条件つき行動に基づく入力空間分割

STNS において固定長行動を用いた場合の利点は，4.4.2節の「ユーティリティおよび行動に基づく分割表現」のところで述べたように，状況表現がユーティリティ関数の近似となるので，マルコフ決定問題の意味での最適化が可能となる点である．反面，固定長行動では，作業空間上に到達しづらい点が数多くできるので，確実に報酬に到達するためには，行動をかなり細かく離散化しなければならない．その結果，状況がかなり細かく，数多くなり，学習の収束に，非常に時間がかかる恐れがある．また，STNS では，報酬から離れるごとに状況の形状のゆがみが蓄積されていくので，図5.6は実際には，図5.8に示すような分割表現になる．これは，行動を細かくすればするほど，状況の形状が悪くなり，誤った行動の選択の機会が増えてしまうことを意味する．

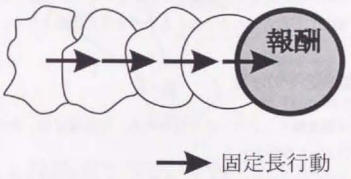


図 5.8: 固定長行動に基づく入力空間分割 2

一方，STNS において条件つき行動を用いた場合には，最適政策が獲得できる保証はないが，1.「かなり高いレベルで抽象化できるので，確実にゴールに到達する行動を早く獲得することが期待できる」，2.「報酬までに通過する状況の数を減らせるので，状況の形状のゆがみによる行動選択の誤りを減らすことができる」，3.「切り出した状況の意味（行動とその結果）がはっきりしているので，記号として扱いやすい」などの利点がある．利点の 3 は，例えば，固定長行動を用いた場合には，「一回前進すればゴールに到達する状況」，「二回前進すればゴールに到達する状況」…などをすべて区別しなければならないが，条件つき行動を用いれば，「前進すればゴールに到達する状況」としてまとめることができるということである．

このような利点を考慮すると，STNS では，条件つき行動アプローチの方が望ましいと思われる．次章で説明する実験でも，条件つき行動アプローチを用いている．STNS で汎用的に用いることのできる行動の終了条件としては，

1. 報酬を得た．
2. プラン中で目標としている状況に到達した．

の二つが考えられる．さらに，状況の内部では，正事例がなるべく散らばることが望ましいので，目標の状況に到達してもすぐに行動を終了せずに，ランダムな深さまでその状況に侵入してから行動を終了するなどの条件が有効である．その他にも，報酬につながる整った形状の状況を作り維持するためには，問題に依存した条件を含む様々な細かい条件が必要である（6.1.2節参照）．

STNSでは，システムの内部を汎用的なものとして整えるために，行動モジュールの中にこのような問題に依存する知識を埋め込んでいる．実際，STNSは行動モジュールの内部に対しては不干渉で，行動が結果的に思った通り実行されたかどうかも気にしない．STNSにとっては，実際に動くことではなくて，動こうと試みることが行動なのである．目の前に壁があって，前進しようとしても一歩も動けなかった場合でも，STNSでは，「"前進"したらぶつかった」というように認識される．

5.3 行動の選択とその学習

行動の選択は状況遷移ネットワークで行われる．行動規則は，状況遷移ネットワーク上の強化学習によって学習される．

STNSで条件つき行動アプローチを用いた場合，5.2.3節で述べたように，状況は行動に基づく分割表現で表される．行動に基づく分割表現では，各状況は「特定の行動によって特定の結果が得られる知覚入力状態の集合」という意味を初めから持っている．しかし，STNSでは各状況における行動政策を強化学習で決めなおす．それは，STNSでは状況の認識の学習をオンラインで漸次的に行うからである．STNSは，4.4.2節で述べた行動に基づく三つの分割手法[7, 111, 129]とは違って，1. データが正確な状況の形状を表現するには不十分な数しかない早い段階で状況を切り出し，2. 新たなデータを用いて状況の形状を調整しながら，3. 強化学習によって各状況に適切な行動を学習する．状況を不十分な数のデータから切り出す場合，必ずしもその領域にとって最適な行動が割り当てられないので，3の強化学習が必要である．

5.3.1 状況遷移ネットワーク－STN

状況遷移ネットワーク（Situation Transition Network，STN）とは，状況間の遷移確率モデル，および各遷移の直接報酬モデルからなるワールドモデルである（図5.9参照）．これはマルコフ決定問題におけるワールドモデルと同じものである．状況iにおいて行動bを実行した時に状況jに遷移する確率を$p(i,j;b)$，その遷移によって得られる直接の報酬の期待値を$r(i,j;b)$と表す．

STNSにおいては，状況は，システム内部で動的に抽出，維持されるものなので，

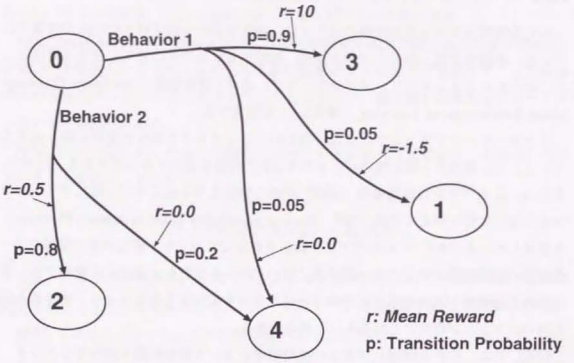


図5.9: STN（状況0からの遷移のみ示す）

ワールドモデルをあらかじめ与えることはできない．そこでワールドモデルを推定しながら適切な行動を獲得していく必要がある．これは，4.3節で説明した不完全情報マルコフ決定問題である．この問題は，強化学習によって解くことができる．

不完全情報マルコフ決定問題は，離散時間の問題である．すなわち，1タイムステップごとに，情報を取り入れ，処理し，行動を出力することを前提としている．しかし，条件つき行動アプローチを用いた場合には，行動は，離散時間ごとに終了するのではなく，終了条件が満たされるまで続けられる．このような行動を用いる場合には，正確な意味でのマルコフ決定問題を解くこと，すなわち「離散時間ごとに割り引いた報酬の合計の期待値の最大化」を行うことはかなり困難である．そこで，この場合の強化学習では，「行動の切り替えごとに割り引いた報酬の合計の期待値の最大化」を目的とすることにする．多くの場合，これではマルコフ決定問題の意味での最適行動政策を獲得することはできないが，なるべく簡単な手順で報酬を得る行動政策を獲得することができる．

5.3.2 インターリーブプランニングに基づく強化学習 – IPRL

不完全情報マルコフ決定問題に対しては，強化学習の分野で様々な手法が考案されている．本研究では，動的に状況を抽出，維持するシステムに適した強化学習法として，あらたに「インターリーブプランニングに基づく強化学習（Interleave Planning-based Reinforcement Learning，IPRL）」を提案する．

インターリーブプランニングとは，部分的プランニングと実行を交互に繰り返すことによって，熟考性と即応性のバランスのとれた行動を実現する方法である．山田は，先読みによるプランの成功確率（後述）がある閾値よりも低くなった時にプランニングから実行に切り替える手法「SIP（Success probability based Interleave Planning）」を提案している[140]．この手法では，閾値を固定していても，変化の速い環境では，各行動の成功確率が低いので，即応的に短いプランを実行し，安定した環境では，各行動の成功確率が高いので，熟考的に長いプランを実行することになり，熟考性と即応性のバランスが自然にとれるという利点がある．

IPRLでは，SIPと同様にプランの成功確率によって探索範囲を限定するインターリーブプランニングを用いる．SIPにおいては，環境の変化が速く，プランニングの最中にも徐々にプランの成功確率が落ちていくような問題を扱っている．IPRLで対象とする不完全情報マルコフ決定問題では，そのような急激な変化は仮定していない．しかし，成功確率で探索範囲を限定することによって，不確かな環境では無駄なプランニングをせずに即応的に行動し，確かな環境では可能性の高いプランだけを詳しく検討するという意味で，熟考性と即応性のバランスが自然にとれると言える．

IPRLでは，決定理論のvalue iterationと同様に，動的計画法の最適性方程式を用いて，各状況のユーティリティの近似値を計算する．状況遷移ネットワークでは直接報酬を各行動ごとではなくて各遷移ごとに割り当てているので，式4.2の最適性方程式は次のように変更される．

$$U(i) = \max_{b \in B} \sum_{j \in X} p(i,j;b)\{r(i,j;b) + \gamma U(j)\},\ i \in X \qquad (5.1)$$

状況遷移確率$p(i,j;b)$と，直接報酬$r(i,j;b)$は，過去の経験に基づいて最尤推定によって求めることにする．すなわち，

$$p(i,j;b) = \frac{n_{ij}^b}{n_i^b}$$

ただし，n_{ij}^bは履歴データベース中の状況iにおいて行動bを実行した時に状況jに遷移したデータの個数，n_i^bは状況iにおいて行動bを実行したデータの個数である．$n_i^b = 0$の場合は，存在するすべての状況への遷移確率を等しく1/(状況の数)とする．また，

$r(i,j;b)$は履歴データベース中のこの遷移によって直接得られた報酬の標本平均値である．この2つの値は，一つの行動が終了するたびごとに更新される．

value iterationでは全状況のユーティリティが収束するまで繰り返し計算をするが，IPRLでは現状況からプランを前向きに成功確率が高い範囲内で展開して，その範囲の外の状況のユーティリティを0とすることによって，繰り返し計算なしにユーティリティの近似値を求める．

IPRLにおけるプランPは，行動と，目標とする状況（「目標状況」と呼ぶ）のペアのリストである．すなわち，

$$P = ((b_1,d_1),(b_2,d_2),...,(b_n,d_n))$$

ただし，nはプランの長さ，b_iはプラン上のi番目の行動，d_iはその行動による目標状況である．プランの成功確率は，プラン上のすべての遷移確率の積であらわされる．

IPRLにおけるプランニングの手順は次のように表される．

1. 現状況からプランを前向きに展開する．プランの成功確率がp_{min}以上，かつプランの長さがn_{max}以内の範囲で，展開を続ける．

2. 展開された木の葉に相当する状況のユーティリティを0とする．

3. 展開された木の全ての葉から現状況に向かってバックトラックしながら，式5.1を用いてユーティリティを計算する．式5.1の右辺で最大値を与える行動と，それに最も貢献する遷移（$\sum$の中身で最大値を与えるj）をプランとして親ノードに伝える．

4. 根に戻った時に，唯一のプラン（マスタープラン）が決定される．

上述のように，探索範囲外の状況のユーティリティを0とみなすので，各状況のユーティリティの平均値が，0に近い方が望ましい．そこで，環境から与えるデフォルトの直接報酬は0として，何か特別の出来事に対して正または負の報酬を与えるような報酬の与え方が適切である．

プランニングが終了すると，マスタープランの実行に移る．マスタープランの最後に達する（マスタープランの終了）か，またはマスタープラン上の行動以外の行動を実行した場合とマスタープラン上の目標状況以外の状況に遷移した場合（マスタープランの失敗）は，実行を停止してプランニングに切り替える．

5.3.3 IPRL と他の強化学習法との比較

IPRL は，離散入出力，離散時間，マルコフ性が仮定された上で適切な行動政策を獲得することを目的とする「不完全情報マルコフ決定問題」を扱う手法である．この問題に対して，強化学習の領域では様々な手法が提案されている．この節では，それらの手法を代表的な三つのカテゴリに分類し，それらと IPRL を比較する．ここまでは，IPRL に対する離散入力を「状況」と呼んでいたが，この節では，他の強化学習法に合わせて「状態」と呼ぶことにする．

カテゴリとしては，「iteration」と「DP-based learning」と「profit sharing 法」を考える．iteration とは，ワールドモデルを記憶しておいて，状態のユーティリティは，必要な時にその都度，ワールドモデル上の繰り返し計算によって求める手法である．このカテゴリには，4.3節で説明した，ワールドモデルを最尤推定しながら value iteration または policy iteration でユーティリティを計算する手法が含まれる．DP-based learning とは，動的計画法（DP）の最適性方程式（式 4.2）を用いて，記憶しているワールドモデルと状態のユーティリティを行動のたびに共に少しずつ更新していく手法である．このカテゴリには，Barto らの Adaptive RTDP [13]，Sutton の TD method [112]，Watkins らの Q-learning [125] などが含まれる．profit sharing 法とは，報酬を獲得する度に，その報酬を途中で使用された行動規則に分配して，記憶している行動規則の評価値を更新していく手法である．報酬の分配の仕方に何種類かの方式がある [72]．

IPRL は，学習としてはワールドモデルの構築しか行わず，各状態のユーティリティは，必要な時にその都度計算するという点で，iteration に類似している．しかし，iteration では，状態のユーティリティ（または行動政策）が収束するまで繰り返し計算するのに対して，IPRL では，前向きプランニングの探索範囲を限定することによって，繰り返し計算なしに状態のユーティリティの近似値を求める．

各手法の性質は表 5.1のようにまとめられる．まず，学習速度に関して説明する．iteration と IPRL では，学習としてワールドモデルの最尤推定しか行う必要がないので，学習は非常に速く進む．profit sharing 法では，報酬を得た時に，その影響を途中で使用された行動規則に一度に分配するので，学習はかなり速く進む．ただし，学習を促進するために，報酬から時間的にかなりさかのぼった行動にも大きな割当てを与えると，無用な行動規則も強化されてしまうことが指摘されている [72]．このような事態を避けるためには，急速な報酬の割当てを控えて，ゆっくりと学習する必要がある．DP-based learning では，報酬の影響は，漸化式（式 4.2）を用いて一回の行動ごとに一段階ずつしか伝達されないので，学習速度はかなり遅いといえる．この欠

点を改善するために，4.3節で述べたように，学習を促進するいくつかの手法が研究されている [14, 61, 62, 63, 112, 113]．しかし，DP-based learning では，通常，ワールドモデルの学習とユーティリティの学習の両方を行わなければならないので，ワールドモデルの学習しか行わない IPRL（および iteration）より学習速度を速くすることはできない．

表 5.1: 強化学習法の比較

	学習速度	計算速度	最適性への収束
iteration	非常に速い	非常に遅い	保証される
IPRL	非常に速い	遅い	近似的に保証される
DP-based learning	遅い	非常に速い	保証される
profit sharing 法	速い	非常に速い	なし

次に，行動選択のための計算の速度に関して説明する．profit sharing 法では，各状態において最強の行動規則を選択するだけなので，計算は非常に速い．DP-based learning では，高々一段階の先読みしか行わないので，こちらの計算も非常に速いと言える．IPRL では，前向き探索を行うので，探索の深さに対して計算時間が指数関数的に増加する．従って，計算にはかなり時間がかかる．ただし，IPRL では全状態のユーティリティを求める必要はなく現状態のユーティリティだけを計算すること，その計算中に報酬に到達するためのプランを立てて，プランにしたがって行動する間はユーティリティの計算を行わないこと，実現する確率の高い範囲に探索を限ることなどで計算時間を節約している．iteration では，行動選択のための計算に繰り返し計算を含むので，一般的に，非常に時間がかかるといえる．ただし，iteration の初期値として最適値に近い値を与えられる場合は，計算速度はかなり速くなる．学習が進んでいる段階では，ワールドモデルがかなり収束しているので，初期値として直前に求めたユーティリティや行動政策を用いることによって，計算時間をかなり短縮することができる．特に policy iteration を用いた場合は，連続的な評価値ではなく離散的な行動政策を収束させるので，収束自体が速いし，最適値に近い初期値を比較的簡単に与えられることも期待できる．

次に，最適性への収束について説明する．iteration と DP-based learning では，各状態を訪れる確率が永遠に 0 にならないことを保証すれば，最適政策に収束することが証明されている [13]．IPRL では，探索する範囲内での最適性が保証される．確率が低くて非常に大きい報酬が得られるようなプランがない場合には，良い近似になる

ことが期待できる．profit sharing 法では，学習は経験を強化する方向に進み，最適性を目指す方向には進まない [141, 75]．

以上のことから，IPRL は，報酬が得られる間隔があまり長くない環境で素早く学習する問題に向いているといえる．ASRR では，抽象度の高い状態を切り出すので，プランがあまり長くならないことが期待できる．また，ASRR では状態が生成したり消滅したりするので，行動の学習が早く収束することは非常に重要である．従って，IPRL は，ASRR と組み合わせるのに適した強化学習法であるといえる．

一方，DP-based learning を ASRR と組み合わせることを考えた場合，学習速度が遅いので，状態の生成や消滅の影響の伝播にかなり時間がかかることなどが問題になると思われる．profit sharing 法の場合は，そもそも最適性を目指さないので，状態の生成や消滅に合わせて，どのように新たな経路を開拓していくのかを考えなければならない．

iteration を ASRR と組み合わせる際には，繰り返し計算による行動選択の遅さが問題になる．ただし，上述のように，policy iteration を用いる場合は，ワールドモデルの学習が進んだ段階では，かなり計算時間が短くなることが期待できる．また，iteration では，状態の数が増加すると飛躍的に計算時間が伸びるが，ASRR では，抽象度の高い状態を切り出すので，状態の数が少なく抑えられることが期待できる．これらを考慮すると，実際にはそれほど計算時間が長くならないようにも思われる．また，iteration を用いると最適性が保証されるのは，大きな利点である．一方，学習がかなり進んだ段階でも，ゴールから離れた状態（特に「余白」である状況 0）では，行動による報酬の期待値の差があまりないので，最適政策に収束させるために，何度か繰り返し計算をする必要がある．また，状態の生成や消滅の直後などは普段に比べて著しく計算時間がかかる可能性もある．これらは，policy iteration を ASRR と組み合わせる際に，不利な方向に働く．iteration（特に policy iteration）と IPRL のどちらがより ASRR と組み合わせるのに適しているのかは，今後，検討しなければならない課題である．

iteration と IPRL の性質で明らかに異なるのは，iteration では全体の状態の数が増加すると計算時間が飛躍的に伸びるのに対して，IPRL では全体の状態の数の増加はほとんど影響がないという点である．そこで，「状態の数は多いけれど，比較的頻繁に報酬が得られるような問題」に関しては，IPRL の方が iteration よりも圧倒的に適していると考えられる．このような問題の一種としては，知覚入力空間中の多数の点で報酬が与えられるようなタスクがあげられる．また，IPRL では，プランを作成するという点が iteration とは異なっている．作成したプランをさらに処理するような問題では，IPRL のプランを作成するという機能が大きな意味を持ってくるだろう．そ

のような問題としては，例えば，複数のエージェントが各々のプランを比較して交渉するような問題が考えられる．これは，知識の共有に関する問題で今後の課題の一つである．

第 6 章

実験

6.1 2 次元入力のナビゲーション – 簡単な問題

6.1.1 問題設定

STNS の最初のアプリケーションとして，簡単な 2 次元入力のナビゲーション問題を扱う．問題の設定を図 6.1に示す．ゴールを置く領域の大きさ (72×72) は，ローバーが任意の方向からゴールに近付いた時に，ゴールに到達する前に，壁に衝突することがないように定められている．作業空間を図 6.2に示す．この図において，空間の比率は正しく保たれている．STNS の目的は，このナビゲーション問題を繰り返し解くことによって，ゴールに到達するための適切な状況認識と行動規則を獲得することである．

タスク:
100x100の連続的な平面上で，16x12のローバーを，唯一のゴール（半径5の小円）に導くこと
　ゴールは，部屋の中央の72x72の正方形内の任意の場所におかれる．
　障害物は周囲の壁のみ．

知覚入力:
ローバー固定の実数座標系で見たゴールの(x, y)座標
　原点をローバーの重心とし，ローバーの正面をx軸の正の方向とする．

報酬:
1. ゴールに到着する．(+10)
2. 壁にぶつかる．(-1)
3. 90度以上回転しようとする．(-1)
　90度というのは，後述の行動の対称性より，90度回転すれば全方位に向くことができるからである．

行動:
前進，後進，左回転，右回転
　回転時には，壁に衝突することなく回転できると仮定する．

図 6.1: 2 次元入力のナビゲーション問題

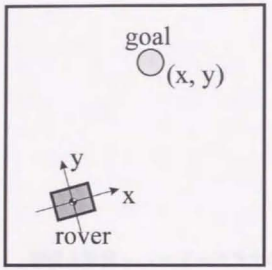


図 6.2: 作業空間

6.1.2 ナビゲーション問題における行動

ナビゲーション問題における行動は，5.2.3節で述べた「条件つき行動」を用いる．その終了条件は以下のように定める．

行動の終了条件

次の二つの終了条件のいずれかが満たされるまで，行動を続ける．

1. 報酬を得た．
2. マスタープラン中の目標状況に到達した．

報酬が得られるのは，「ゴール到達」，「衝突」，「90°回転」の三つの場合である．「ゴール到達」と「衝突」の場合にはその場で停止する．「90°回転」の場合には，その場で停止してしまうと，頻繁に同じ方角を向いて停止してしまうことになるので，次の三つの手法を用いて確率的に決まる停止位置まで戻ってから，停止することにする．

1. スタート地点の状況および状況0と異なる，条件行動が現在の行動と一致していない状況のうち，最初に通過した状況の中で，一様分布で停止位置を決定する．
2. 1に相当する状況がない場合は，スタート地点の状況と異なる，条件行動が現在の行動と一致している状況のうち，最初に通過した状況の中で，一様分布で停止位置を決定する．

3. 1, 2に相当する状況がない場合は，$(0°, 90°]$の中で一様分布で停止位置を決定する．

この手法では，主に三つのヒューリスティクスを使用している．一つめは，「スタート地点と同じ状況内で止まってしまうのは好ましくない」というヒューリスティクスである．スタート地点と同じ状況で止まってしまうと，再び同じ行動が選択される可能性が非常に高いので，他の状況を優先した方が良い．二つめは，「状況0の中で止まってしまうのは好ましくない」というヒューリスティクスである．状況0は余白の状況で，適切なプランニングが難しいので，他の状況を優先した方が良い．三つめは，「現在の行動を条件行動とする状況内で止まってしまうのは好ましくない」というヒューリスティクスである．現在の行動を条件行動とする状況内で停止すると，現在の行動が再び選択される可能性がやや高いので，他の状況を優先した方が良い．一つめと二つめの優先順位は等しく，三つめの優先順位はそれよりも低い．

目標状況に到達した時には，その場ですぐに停止してしまうと，事例が状況の境界面上にばかり集中して，タスクの達成や状況の形状の維持に悪影響を与える．タスクの達成のためには，通常，状況のなるべく中心近くで停止するのが良いが，正事例が状況の中心に集まってしまうと，状況の形状が維持できなくなる．タスクの達成と状況の形状の両方を考慮に入れると，事例は状況の中でなるべく散らばっているのが良いと言える．このような分布を実現するために，行動を境界面上で停止しないで，目標状況の中に侵入し，その経路上で「目標状況の正事例集合からのマハラノビス距離が極小の地点」，または「目標状況から出る地点」，または「スタート地点から90°回転した地点」のいずれかに遭遇するまで，行動を継続する．そして，目標状況に最初に入った地点と行動を終了した地点の間で，一様分布で停止位置を決定し，その位置まで戻ることにする．

目標状況なしで行動している場合，またはスタート地点と同じ状況を目標状況としている場合は，行動の終了条件2を用いることができない．そこで，行動を開始して最初に遭遇する「スタート地点の状況および状況0と異なる，条件行動が現在の行動と一致していない状況」を新たに目標状況と決め直すことにする．

行動がこのように目標状況に依存する場合，状況間の遷移確率や遷移による報酬の期待値も目標状況に依存する．従って，状況遷移確率としては，正確には「ある目標状況を持っている場合に，ある状況からある状況に遷移する確率」を記憶しなければ

ならない．しかし，IPRLでは，簡単のために，「ある状況からある状況に遷移する確率」を記憶している．遷移による報酬の期待値も同様に目標状況に依存しないものとみなして記憶している．そのため，ユーティリティの計算は，一種の近似計算であるといえる．しかし，政策が収束すれば，各状況から常に同じ目標状況を持って行動することになるので，状況遷移確率と報酬の期待値もその政策に関する正しい値に収束するはずである．

行動の選択は，次の二つ規則に基づいて行う．上の規則の方が強い規則で，上の規則でまず実行可能な行動を選んだ後で，下の規則で実行する行動を決定する．

行動選択の制限

新たにプランを立てる際，またはランダムに行動する際には，直前に実行した行動と対称的な行動を実行しないように行動の選択を制限する．これは，「今来た道をそのまま戻っても，あまり報酬は期待できない」というヒューリスティクスに基づいている．また，壁に衝突した場合，および同じ場所で360°以上回転した場合には，再び同じ行動を実行しないように制限する．これは，そのような場合に同じ行動を繰り返しても，明らかに無駄だからである．ただし，マスタープランの実行が継続中の場合は，何らかの報酬の見込みに基づいて行動しているはずなので，これらの制限を受けないことにする．

貪欲な行動選択

行動選択において，実行可能な最も報酬の見込みの高い行動を常に選択する行動政策は，「貪欲政策（greedy policy）」と呼ばれている．強化学習においては，学習の初期の段階では，通常は，確率的な行動選択が用いられる．その理由は，最適な行動政策が獲得されていない時点では，色々な行動を試してみる必要があること，また，学習の初期の段階で貪欲政策を用いると，政策にループが形成された時になかなか抜け出せなくなることなどである．政策が間違っている場合に，ループを繰り返すことによって政策を学習し直すのは悪いことではないが，そのために，報酬の収集が一時的に著しく減少することは，オンライン学習としては好ましくない，また，ループ内の知覚入力状態ばかりを著しく偏って経験すると，システムがその状態に特化してしまうので，タスク全体の学習という点では好ましくない．そこで，確率的な行動によってループに陥らないようにすることは有用である．しかし，学習が収束した時点では，貪欲政策の方が良い性能を得られるので，学習の収束とともに徐々に貪欲政策に収束させていく手法がよく用いられる [13, 61, 63]．しかし，どのような割合で貪欲政策に収束させていくのかは，いまだ経験的に定められている段階である．

STNSでは行動選択法を特に定めていないが，このナビゲーション問題に関しては，学習の初めから貪欲な行動選択を行う．すなわち，各状況において，マスタープラン中で指定されている行動を必ず選択する．その理由の一つは，貪欲政策を用いても色々な行動を試みることができるからである．このナビゲーション問題では，上述の行動選択の制限によって，貪欲政策を用いても次善の策がまれに選択される．そして，IPRLでは学習が速いので，まれに選択される行動でも良い結果につながれば，急速に政策が変化する．その結果，各状況で色々な行動が試されることになる．また，未経験の行動が選択されやすくなるように，未経験の遷移に小さい正の報酬（+1）を与えることにした．もう一つの理由は，状況認識にASRRを用いる場合は，政策にループがあっても行動のループには陥らないからである．ASRRでは微視的な認識法としてNN識別法を用いているので，貪欲政策で失敗した点はすぐに状況に含まれない点となる．そして，次に同じ点から行動する時には，異なる状況の点としてプランニングされるので，貪欲政策であっても異なる行動が選択されやすい．従って，同じループを繰り返すことはない．

ただし，状況0においては，貪欲政策を用いずに，常にランダムに行動を選択することにした．その理由の一つは，状況0で貪欲政策を用いると，行動のループに陥りやすく，上述の性能の一時的な悪化や，偏った経験につながるからである．状況0は固有の意味を持たない余白の状況なので，貪欲政策で失敗した点でも状況0に留まり続ける．従って，政策のループが行動のループに直結してしまう．またもう一つの理由は，状況0では，各行動の結果にあまり違いがない（行動によって，特色のある結果が得られるのならば，その領域は新しい状況として切り出されるはずである）ので，ランダムに行動を選択しても，性能にほとんど悪い影響を与えないからである．状況0は，特色ある領域を切り出した残りなので，学習が進めば，状況0に含まれるデータはかなり少なくなり，性能に対する悪影響はますます小さくなる．

6.1.3 結果と考察

上のナビゲーション問題のための行動政策を学習するSTNSをつくり，シミュレーションで実験を行った．各定数値は，

$$N^R_{init} = N^R_{min} = 3,\ N^T_{init} = 15,\ N^T_{min} = 7,\ r^R_{min} = 5.0,\ R^T_{min} = 1.5,$$
$$\gamma = 0.7,\ p_{min} = 0.1,\ n_{max} = 7$$

と定めた．履歴データベース中のデータ数は1000個とし，超楕円体の大きさと状況の重なり方の更新は100回の行動ごとに行うと定めた．N^R_{init} と N^R_{min} は，2次元の空間の中で2次元の部分空間を保つために最小限必要なデータ数である．ただし，$N^R_{init} = N^R_{min} = 5$ として同じ実験を行った場合にも，性能にほとんど差は見られなかった．その他の値は，恣意的に定めているが，特に細かく調整する必要はなかった．

シミュレーションは一回の試行で10000回の行動を実行する．入力空間中に，ゴールに到達した状況（図6.3の状況1）と残りの全体を占める状況0の二つの状況のみが存在するところから，学習を開始する．ゴールに到達した時には，任意の位置にゴールを，ゴール以外の任意の位置にローバーを置き直して，学習を続けた．それを20試行行った．

典型的な収束後（累積行動数10000回）の入力空間を図6.3に示す．説明のために，各状況に状況0～状況7の名前をつけている．前進，後進，左回転，右回転で矢印の方向にゴールが移動する．中央の小円（半径5）がゴールに到達した状況で，周囲の大きい円（半径103.2）はゴールとローバーがもっとも離れた場合を表す．この図から，入力空間のほぼ全体が，状況0以外の意味のある状況によって覆われていることがわかる．また，理想的な入力空間は，図6.4に示すような空間なので，かなり良い状況認識が獲得されていることがわかる．入力空間の周辺部が状況0のまま残されるのは，周辺部はデータが十分に集まらないからである．周辺部のデータが得られるのは，図6.2の作業空間の対角線上にローバーとゴールが位置する時などの限られた場合だけである．20試行のデータで，十分に収束していると思われる累積行動数9001～10000回のデータのうち，状況0のデータはわずか2.3%であった．従って，周辺部が状況0であっても，性能に大きな悪影響は与えないといえる．

図6.5と図6.6に，入力空間の典型的な変遷の例を示す．これは，図6.3に示した入力空間が収束する過程である．図6.5a～fは，新たな状況が切り出された直後の空間である．各状況が，大まかな形状で切り出されて，その後，適切な形状に調節されていく様子がわかる．$N^R_{init} = 3$ と $N^T_{init} = 15$ は，かなり小さい数であるが，このようにデータが不十分な段階で状況を切り出しても，抽出後の修正によって，適切な状況形状を獲得することができる．累積行動数1200回の図6.6hまでには，状況の形状がほぼ収束している．20試行全てにおいて，一度このような形に収束すると，多少ゆがむことはあっても，二度と崩れることはなく，シミュレーションの終わり（累積行動数10000回）まで安定して状況が維持された．

図6.7にゴールまでの平均行動数の変化を示す．図6.7aでは各試行を100回の行動ごと，図6.7bでは250回の行動ごとに区切り，それぞれの中で平均した．このグラフから，学習がかなり速く進むことがわかる．初めの状況が切り出される辺り（平均の

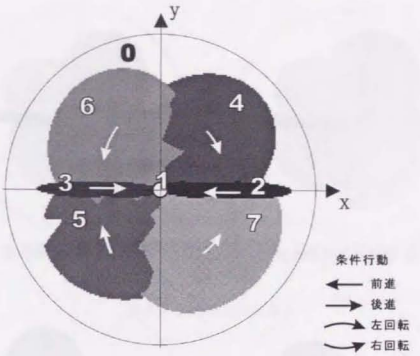


図 6.3: 収束後の入力空間（0 ～ 7 は状況を示す）

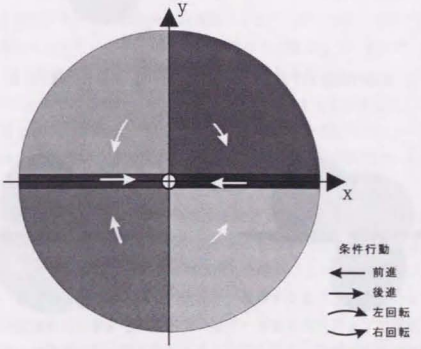
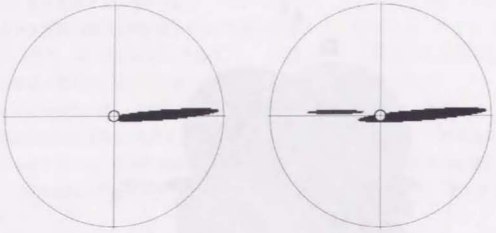
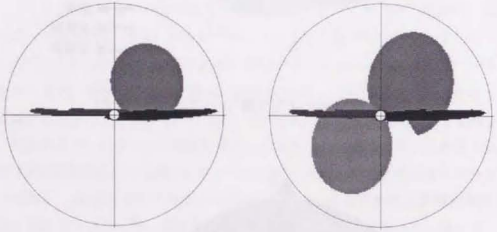


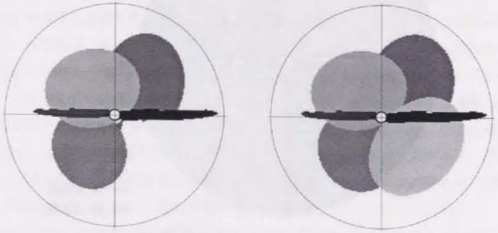
図 6.4: 理想的な入力空間



a. 累積行動数 280 回　b. 累積行動数 293 回

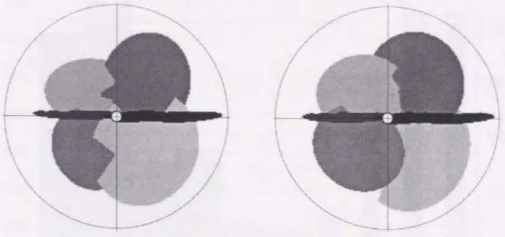


c. 累積行動数 517 回　d. 累積行動数 767 回



e. 累積行動数 803 回　f. 累積行動数 814 回

図 6.5: 入力空間の変遷 1



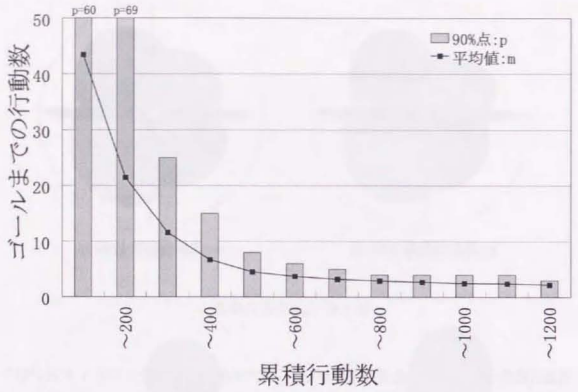
g. 累積行動数 1000 回　h. 累積行動数 1200 回

図 6.6: 入力空間の変遷 2

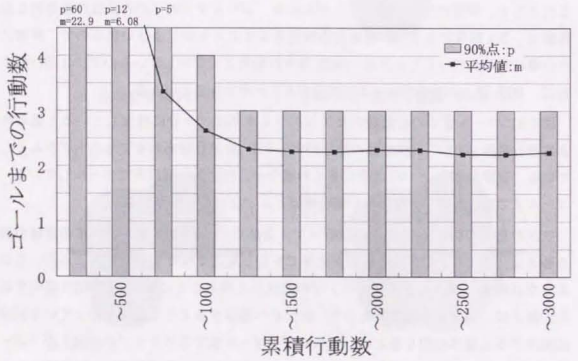
累積行動数 211 回）で，急速に学習が進み，四つの象限の最後を占める T 状況が切り出される辺り（平均の累積行動数 1058 回）で収束する．平均的には，累積行動数 1200 回程度で収束した．このように，データが不十分な早期の段階でも状況を切り出すことによって，学習が急速に進むことがわかる．ゴールまでの平均行動数の最終的な収束値は，2.2 前後であった．理論的な最適値は 2 よりも少し小さい値なので，非常に良い値に収束しているといえる．90% 点も行動数 3 で安定しているので，3 回行動すれば，90% 以上の確率でゴールに到達することができるといえる．

図 6.8にゴールまでの行動数のヒストグラムを示す．十分に収束していると思われる累積行動数 9001 ～ 10000 回の部分のデータを 20 試行分あわせてヒストグラムとしている．全体の 87% という非常に多くの場合が，行動数 2 回以内でゴールしている．ゴールまでの行動数が非常に多い場合がほとんどないこともわかる．

状況の意味の切り替えは，20 試行中で，全部で 12 回行われた．すべての意味の切り替えは，ゴールまで 3 回の行動が必要であると見込まれる T 状況で起こった．このような状況を「ゴールまで 3 ステップの状況」と呼ぶことにする．図 6.3で説明すると，例えば，状況 4 の位置にあって，状況 6 へ遷移するような意味を持っている状況に相当する．意味の切り替えは，すべて，「ゴールまで 3 ステップの状況」が「ゴールまで 2 ステップの状況」に切り替わるように行われた．先ほどの例で説明すると，状況 4 の位置にある状況が，状況 2 へ遷移するような意味に切り替えられたことに相当する．従って，すべての場合で，前より適切な意味に切り替わったといえる．また，



a. 累積行動数 100 回刻み



b. 累積行動数 250 回刻み

図 6.7: ゴールまでの平均行動数の変化

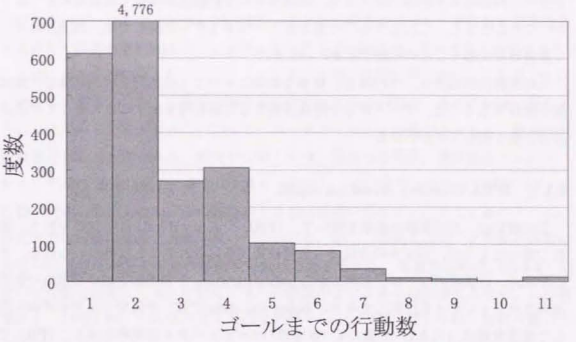


図 6.8: ゴールまでの行動数のヒストグラム

意味の切り替えは，12 回中 8 回が累積行動数 1500 回以内で起こった．これは，履歴データベース中のデータ数 1000 個と比べると，かなり早いといえる．最も遅い場合では，累積行動数 4092 回で起こった．

状況の消滅は，20 試行中で，全部で 2 回起こった．どちらの場合も，「ゴールまで 3 ステップの状況」と「ゴールまで 2 ステップの状況」が，入力空間中の同じ象限内で競合して，「ゴールまで 3 ステップの状況」が消滅した．状況の消滅もかなり早い時期（累積行動数 1256 回と 1687 回）に起こった．

この状況の意味の切り替えと消滅によって，状況の意味はすべて，図 6.4に示す理想的な入力空間と等しくなった．この状況の意味に適合する状況遷移の確率も軒並高く，収束後のワールドモデルでは，90% ～ 95% の間に集中していた（累積行動数 9001 ～ 10000 回の部分のデータでは，R 状況で平均 94.3%，T 状況で平均 92.1% であった）．このことから，入力空間上で理想的な状況認識と行動規則が獲得されていることがわかる．この問題では，行動選択が貪欲政策で行われているのにも関わらず，最適な行動規則の探索が適切に行われているといえる．

収束後のワールドモデルから，policy iteration を用いて最適政策を計算したところ，すべて，図 6.4に示した理想的な行動政策と一致した．このことから，ワールドモデルが適切に獲得されていることがわかる．また，同じワールドモデルから，IPRL

を用いて行動政策を計算したところ，状況0からの行動以外は全て最適政策と一致した．このことから，このようなごく簡単なワールドモデルに関しては，IPRLによって最適政策を導くことが可能であるといえる．

この実験の結果から，STNSが，簡単な問題において，理想的な状況認識と行動規則を獲得できること，データが不十分な段階でも状況を切り出すことによって学習が非常に速く進むことがわかる．

6.1.4 IPRLとpolicy iterationの比較

この節では，上の実験の結果を用いて，IPRLとpolicy iterationの比較を行う．前節で述べたように，いずれの方法でも最適政策が獲得されることは確かめられた．そこで次に，収束後のワールドモデルからpolicy iterationとIPRLを用いて求めた各状況のユーティリティを比較してみる．policy iterationでは，そのワールドモデルに関して最適方程式（式5.1）を満たす“本物の”ユーティリティが求められる．IPRLでは，5.3.2節で述べたように，その近似値が求められる．

比較の結果を表6.1に示す．表6.1の左には，全20試行の収束後のワールドモデルからpolicy iterationを用いて計算したユーティリティの平均値と標準偏差を示す．ゴールまで1ステップの状況，ゴールまで2ステップの状況，および状況0に分けて，平均値と標準偏差を求めた．状況0以外は，かなりばらつきが少ないことがわかる．

表6.1: ユーティリティとIPRLによる近似値との誤差

	ユーティリティ		IPRLの誤差	
	平均値	標準偏差	平均値	標準偏差
ゴールまで1ステップの状況	9.67	0.106	0.235	0.0788
ゴールまで2ステップの状況	6.46	0.0990	0.432	0.0867
状況0	4.08	0.988	0.811	0.724

表6.1の右には，同じく収束後の20個のワールドモデルからIPRLを用いて計算した各状況のユーティリティの近似値と本物のユーティリティとの誤差の平均値と標準偏差を示す．IPRLによる近似値は，5.3.2節で述べたように計算するが，探索木の根においては，式5.1の右辺において最大値を与える行動ではなく，最適政策における行動を選択した場合の値をユーティリティの近似値とした．これは，本物のユーティリティとIPRLによる近似値を，同じ行動を選択した場合の報酬の期待値として比較するためである．IPRLで計算した場合は，全ての状況において，本物のユーティリ

ティよりも若干小さく見積もられた．各状況ごとにその差を求め，ゴールまで1ステップの状況，ゴールまで2ステップの状況，および状況0に分けて，平均値と標準偏差を求めた．誤差の平均値はゴールまで1, 2ステップの状況で，それぞれユーティリティの2.4%，6.7%とやや大きい．ユーティリティ自身の標準偏差と比べても無視できない大きさである．しかし，誤差の標準偏差がかなり小さいので，全ての状況で同じようにずれていることがわかる．従って，ユーティリティに誤差があっても，最適政策が計算できることがわかる．状況0に関しては，誤差の平均値，標準偏差ともにユーティリティの20%に近く，非常に大きい．しかし，STNSは，状況0ではランダムに行動を決定しているので，この誤差の大きさは問題にならない．また上述のように，状況0では行動の結果にあまり違いがないこと，かつ，状況0はデータの数が非常に少ないのでワールドモデルの信頼性が低いことなどから，最適性方程式（式5.1）を用いて行動を決定しても，それほど良い行動選択ができるとは期待できない．この結果から，IPRLが，この2次元入力のナビゲーション問題に関しては，適切に働いているといえる．

しかし，計算時間の点では，policy iterationの方が圧倒的に優れている．Sun Ultra2を用いた[1]時に，policy iterationでは，繰り返し一回あたり，1/10,000秒のオーダーで計算できる．また，収束までの繰り返し回数は，初期値の政策が最適政策と全く異なる場合でも，ほぼ三回以内である．そして，全体の計算は，ほぼ1/1,000秒のオーダーで終了する．一方，IPRLを用いた場合は，上述の設定で，プランニングに1/10秒のオーダーの時間がかかってしまう．実世界中で行動する場合には，ほとんど問題にならない時間だが，計算の質の面でも時間の面でもpolicy iterationに劣るといえる．ただし，IPRLの計算時間は，プランの長さに対して指数関数的に変化するので，パラメータに大きく依存している．実際のところ，この問題では，プランニングの足切り確率$p_{min} = 0.7$として，さらにプランの最大長$n_{max} = 2$としても，IPRLの計算結果は状況0以外は全く変化しないが，計算時間は1/100,000秒のオーダーにまで短縮することができる．従って，あらかじめプランの最大長が予想できる場合には，IPRLの計算速度がpolicy iterationを上回るように設定することも可能である．

この問題に関しては，最適性が保証され，計算時間も問題にならない程度に十分短いので，強化学習としてpolicy iterationを用いた方が良いのではないかと推測される．

[1]以下，計算時間の測定には，すべて，Sun Ultra2を用いている．

6.2 2次元入力のナビゲーション－少し複雑な問題

6.2.1 問題設定

STNSの少し複雑なアプリケーションとして，アクチュエータが故障したローバーを用いた2次元入力のナビゲーション問題を扱う．アクチュエータの故障としては，左車輪の回転数が低下した場合を想定する．

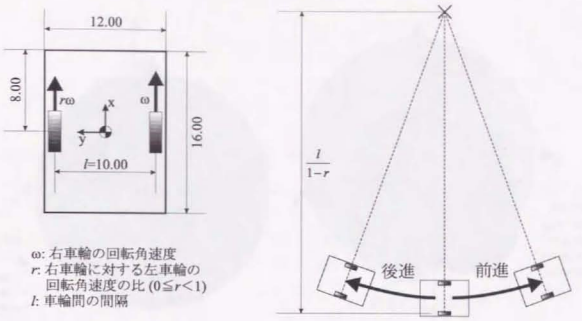
問題設定，およびシステムは6.1節の2次元入力のナビゲーション問題で用いたものと全く同じものを使用する．ただし，行動出力は，左右の車輪が独立に駆動される駆動系を想定して，図6.9aに示すように，左車輪の回転角速度が右車輪のものと比べて低下しているものを用いる．知覚入力空間の座標系において，左車輪は$(0,5)$，右車輪は$(0,-5)$の位置にあるものとする．右車輪に対する左車輪の回転角速度の比rは，95%，90%，85%の三つの場合について，それぞれ10試行ずつ実験を行った．結果として，ローバーは，「前進」および「後進」の場合に，図6.9bに示すような対称な円軌道を描くことになる．今回は回転角速度の低下を想定しているが，左車輪が右車輪よりも小さい場合や，車軸が重心からずれて曲がっている場合にも，同様の円軌道を描くことになる．この故障によって，「左回転」および「右回転」の場合にも重心の移動などの影響が現れることが予想されるが，今回は簡単のために無視することにする．

この問題では，行動の対称性が崩れるので，前節の問題よりも少し複雑な問題となる．左車輪の回転数が低下するにつれて（rの値が小さくなるにつれて），複雑さが増加する．

6.2.2 結果と考察

シミュレーションは一回の試行で10000回の行動を実行する．入力空間中に，ゴールに到達した状況（図6.10，図6.11，図6.12の状況1）と残りの全体を占める状況0の二つの状況のみが存在するところから，学習を開始する．ゴールに到達した時には，任意の位置にゴールを，ゴール以外の任意の位置にローバーを置き直して，学習を続けた．それを，$r=95\%, 90\%, 85\%$の各々の場合について10試行ずつ行った．

$r=95\%, 90\%, 85\%$の場合の典型的な収束後（累積行動数10000回）の入力空間と理想的な入力空間を，それぞれ，図6.10，図6.11，図6.12に示す．説明のために，各状況に状況0～状況9の名前をつけている．収束後の入力空間は，それぞれの場合の10試行の中から，状況が比較的うまく認識されているものを選んだ．理想的な入力空間と比較してみると，$r=95\%, 90\%$の場合には，ゴールまで2ステップ以内

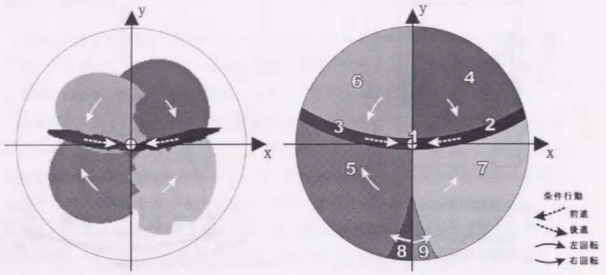


a. ローバーの駆動系　　b. ローバーの軌道

図 6.9: 左車輪の回転角速度の低下

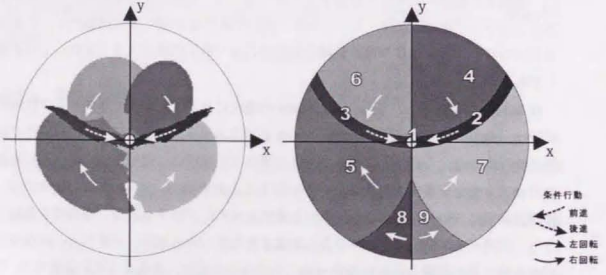
の状況はかなりうまく認識されている．ゴールまで3ステップの状況8と状況9は認識されていない．$r=85\%$の場合には，状況8に相当する状況も抽出されている．また，状況4と状況6の重なりがかなり大きくなって，理想的な状況形状からの歪みが大きくなっている．これは，ローバーが一回の行動で最大90°まで回転することができるために，状況4になり得る領域と状況6になり得る領域が大きく重なっているからである．

10試行全体を見ると，$r=95\%, 90\%$の場合には，ゴールまで2ステップ以内の状況は，全ての試行で認識された．ゴールまで3ステップの状況は，全ての試行で認識されなかった．これは，ゴールまで3ステップの状況は，領域が狭く周辺部にあるために，十分な正事例が集まらないからであると思われる．$r=85\%$の場合には，R状況は常に二つとも認識され，ゴールまで2ステップのT状況は，全部で4個あるうち，平均で3.5個認識されていた．認識されなかった5回は，いずれも，状況4と状況6の一方が認識されていなかった．そのうち3回は，状況4（または状況6）の位置に，状況6（または状況4）に遷移するゴールまで3ステップのT状況が認識されていた．残りの2回は，状況4（または状況6）の位置に，状況が認識されていなかった．これは，状況4になり得る領域と状況6になり得る領域が大きく重なっているために，一方が先に認識されると，もう一方が十分なデータ（正事例と負事例）を



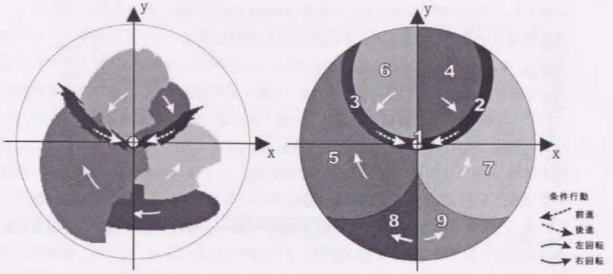
a. 収束後の入力空間　　b. 理想的な入力空間

図 6.10: r=95% の場合の入力空間（0 ～ 9 は状況を示す）



a. 収束後の入力空間　　b. 理想的な入力空間

図 6.11: r=90% の場合の入力空間（0 ～ 9 は状況を示す）



a. 収束後の入力空間　　b. 理想的な入力空間

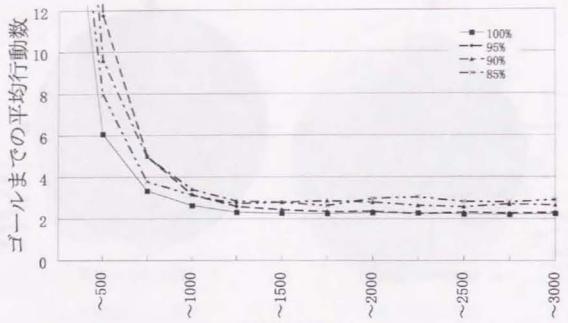
図 6.12: r=85% の場合の入力空間（0 ～ 9 は状況を示す）

集めにくくなるためであると思われる．ゴールまで3ステップの状況のうち，状況8か状況9に相当する状況は，10試行中で5回認識され，そのうち三つがシミュレーションの終わり（累積行動数10000回）まで維持された．$r = 85\%$ の場合には，状況5と状況7の間の空間がかなり広くなるので，このように，状況8か状況9に相当する状況が時々認識されることになる．累積行動数9001 ～ 10000回のデータのうち，状況0のデータは，$r = 95\%$ の場合には2.2%，$r = 90\%$ の場合には4.0%，$r = 85\%$ の場合には4.8%であった．このことから，行動の非対称性が強くなるほど状況0の割合が大きくなることがわかる．

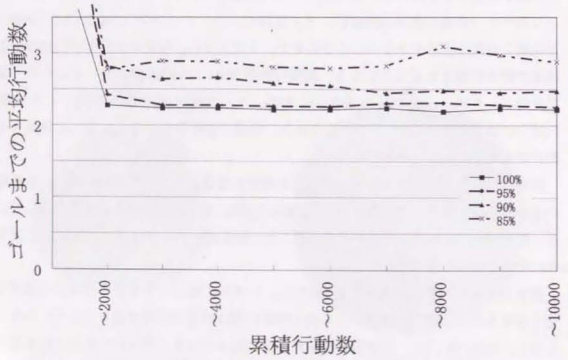
図6.13に，各場合のゴールまでの平均行動数の変化を示す．比較のために，左車輪の回転角速度が低下していない $r = 100\%$ の場合（6.1節の実験結果）も図示している．図6.13aでは各試行を250回の行動ごと，図6.13bでは1000回の行動ごとに区切り，それぞれの中で平均した．

図6.13aより，故障のない場合と比べて，いずれの場合でも学習の進み具合は若干遅くなるものの，累積行動数にして高々250回遅れ程度で学習が進むことがわかる．しかし，図6.13bより，左車輪の回転角速度の低下が大きくなるにつれて，収束後の性能が悪くなることがわかる．

$r = 95\%$ の場合には，累積行動数2000回程度で収束している．収束値は故障のない場合よりもわずかに悪い2.2 ～ 2.3程度である．不適切な意味を持つ状況の意味の切り替えもかなり早い段階で行われていて，累積行動数3500回までには，全ての試



a. 累積行動数 250 回刻み



b. 累積行動数 1000 回刻み

図 6.13: ゴールまでの平均行動数の変化

行の全ての状況で理想的な意味を獲得していた．

$r = 90\%$ の場合には，累積行動数 1250 回程度までは急速に学習が進み，その後はゆっくりと学習が進んで，累積行動数 6000 回程度で収束している．四つの象限の最後を占める T 状況が切り出されるのは，平均で累積行動数 1561 回の辺りであったが，状況の意味が不適切である状態が長く続き，全ての状況が理想的な意味を獲得するのは，平均で累積行動数 3979 回，最も遅い場合では累積行動数 9120 回の時であった．状況の意味の切り替えの回数も多く，10 試行中で全部で 14 回行われた．故障のない場合には，6.1.3節で述べたように，20 試行中で 12 回であったので，$r = 90\%$ の場合に適切な状況の意味を獲得するのがやや困難であることがわかる．そのために学習の収束に時間がかかる．ただし，一度理想的な意味を獲得した状況は安定し，そのような状況全てが，シミュレーションの終わり（累積行動数 12000 回）まで削除されることも意味を切り替えられることもなく維持された．ゴールまでの平均行動数の収束値は 2.4 ～ 2.5 程度で，故障のない場合よりも若干悪い程度である．

$r = 85\%$ の場合には，累積行動数 1250 回程度までは急速に学習が進み，その後は 10000 回行動した後でも，ゴールまでの平均行動数がふらついていて，他の三つの場合のようには収束しなかった．上述のように，累積行動数 10000 回の段階で，ゴールまで 2 ステップの状況でさえ十分な状況認識と行動規則が獲得されていないので，この後学習を続ければ，さらに性能が上がる可能性はある．しかし，図 6.13b の学習曲線を見る限りでは，性能向上はかなり困難であると予想される．従って，この場合には，理想的な状況認識と行動規則の獲得はかなり困難であるといえる．しかし，ゴールまでの平均行動数はふらつくものの 2.7 ～ 3.1 の範囲に収まっており，故障のない場合の 2.2 と比べても，それほど悪くない性能が得られているといえる．

表 6.2に，各場合の，状況の意味に適合する状況遷移の確率の平均値を示す．各場合について，累積行動数 9001 ～ 10000 回の部分のデータを 10 試行分（$r = 1.00$ の場合は 20 試行分）集めて，平均値を求めている．この表から，行動の非対称性が強まるにつれて，特に R 状況からの遷移の確率が大きく低下しているのがわかる．これは，弧状となる R 状況の弧の曲率が大きくなるにつれて，ASRR で状況形状を正確に学習するのが困難になることを表している．ASRR では，巨視的な認識において，超楕円体で状況を表現するので，状況の形状が超楕円体に近いほど状況を表現しやすい．R 状況の弧の曲率が大きくなると，状況の形状が超楕円体から離れ，微視的な認識に大きな比重がかかるようになる．すなわち，状況を表現するのに，多くの負事例を必要とするようになる．その結果，状況遷移確率が低下する．

ここまでに述べたように，左車輪の回転角速度の低下が大きくなるにつれて学習が困難になり，収束に時間がかかったり，収束後の性能が悪くなるなどの悪影響が大き

表 6.2: 状況の意味に適合する状況遷移の確率の平均値

r	R状況からの遷移	ゴールまで2ステップのT状況からの遷移	ゴールまで3ステップのT状況からの遷移
1.00	94.3%	92.1%	存在しない
0.95	92.9%	92.1%	存在しない
0.90	87.9%	90.5%	存在しない
0.85	80.4%	88.6%	91.9%

く現れるようになる．その原因としては，

1. 状況5と状況7の間に2回の行動でゴールに到達できない領域ができ，理論的なゴールまでの平均行動数が増加する．その領域にゴールまで3ステップの状況が抽出されない場合も多く，その場合には，その領域は状況0の一部となって，実際のゴールまでの平均行動数はさらに増加する．

2. R状況が弧状になり，ASRRで状況形状を正確に学習するのがやや難しくなる．その結果，R状況の形状が歪んで，状況遷移確率が低下する．

3. 状況4になり得る領域と状況6になり得る領域が大きく重なっており，一方が他方を覆ってしまう．その結果，覆われた方は十分なデータ（正事例と負事例）を集められず，状況の抽出や適切な行動規則の獲得が困難になる．

などが考えられる．

この実験の結果から，STNSが，超楕円体から離れた形状を持つ状況の認識，小さい状況の認識，他の状況に覆われた状況の認識などの点で弱点を持つことがわかる．そして，状況認識が不十分になるにつれて，行動規則の学習も困難になることがわかる．しかし，問題を徐々に複雑にしていった時の性能の悪化はゆるやかで，図6.12に示すような三つの弱点にかなり適合する問題の場合にも，ある程度良い性能を獲得することができた．

6.3 2次元入力のナビゲーション－柔軟性テスト

この節では，STNSの柔軟性をテストするために行った二つの実験の説明をする．いずれの実験も，6.1節の2次元入力の簡単なナビゲーションのためのシステムにおいて，十分に学習を行った後で，システムに故障が生じるケースを想定している．一

つめの実験は，センサの取り付け角度が変化してしまうケースであり，二つめの実験は，左車輪の回転数が低下してしまうケースである．

6.3.1 センサの故障に対する柔軟性テスト

問題設定

STNSの柔軟性をテストするための一つめの実験として，学習が収束したSTNSの知覚入力を変化させて，新たな知覚入力で再学習する実験を行った．問題設定，およびシステムは6.1節の2次元入力の簡単なナビゲーション問題で用いたものと全く同じものを使用する．ただし，STNSの初期設定と知覚入力を次のように変える．

STNSは，初めから，学習済みのものを用いる．6.1節の実験で学習した20試行のSTNSのうち，累積行動数9001～10000回の間のゴール数が平均に近く，状況がバランスよく認識されているものを一つ選んだ．そのSTNSの入力空間は，図6.3に示したものである．

知覚入力は，ローバー固定の座標系でのゴールの(x, y)座標で，x軸の正の方向がローバーの正面からずれたものを用いる．x軸の正の方向が，図6.14に示すように，ローバーの正面に対して時計回りに，10°，15°，20°,25°，90°，180°回転した各々の場合について，10試行ずつ実験を行った．

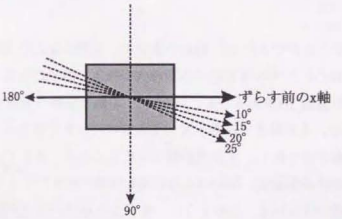


図 6.14: 知覚入力のずれ（x軸のみ示す）

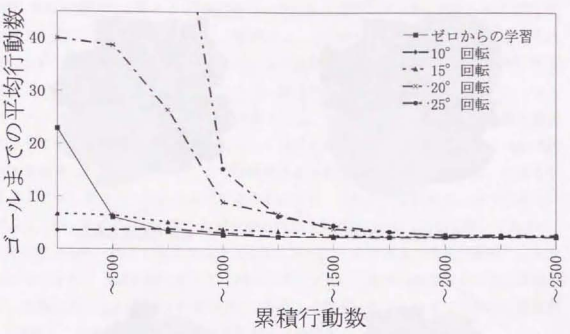
STNSの初期設定は，x軸の正の方向がローバーの正面をちょうど向いている状態で学習済みのものなので，この問題は，ローバーの学習が収束した後に，何かにぶつかるなどしてセンサの取り付け角度が変化してしまった場合の，STNSの適応性を調べるものである．

結果と考察

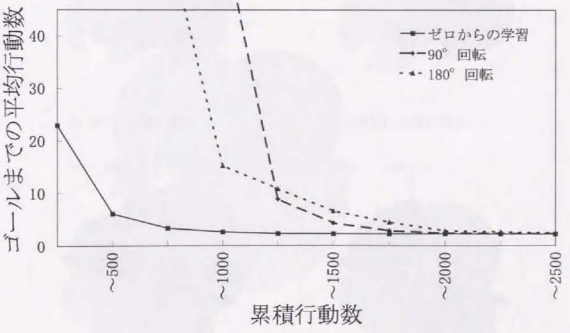
図6.15に，知覚入力を変化させた後のゴールまでの平均行動数の変化を示す．図6.15aでは10°，15°，20°，25°回転の各場合について，図6.15bでは90°，180°回転の各場合について，それぞれゼロから学習したもの（6.1節の実験結果）と比較して図示している．全試行を250回の行動ごとに区切り，それぞれの中で平均した．このグラフから，15°以内の回転の場合にはかなり柔軟に適応しているといえる．この場合には，全ての状況が消滅することなく，うまく形状を変化させた．また，20°以上の回転の場合には柔軟に適応することはできなかった．この場合には，特にT状況が消滅するケースが多かった．もともとのR状況2個とT状況4個のうち維持できた状況の数の平均は，20°回転時にはそれぞれ1.3個と1.5個，25°回転時には1.3個と0.8個，90°回転時には0.1個と0.1個，180°回転時には1.2個と1.5個であった．この数は，学習の早期の性能にほぼ対応している．180°回転の場合は，入力空間の対称性のために約半分の状況が生き残るので，状況がほぼ全滅する90°回転の場合よりも適応が速いが，収束は90°回転の場合の方が良くなっている．これは，180°回転の場合には，各状況がそこそこ対応するものの，過去の余韻を引きずって状況の形状のゆがみがなかなかとれず，かつその状況が邪魔になって新しく整った形状の状況が作られないためだと思われる．ただし，どの場合でも，ゼロから学習した場合に比べて，遅くとも累積行動数約1000回遅れでほぼ同等の収束状態に至るので，安定性はあるといえる．

うまく適応することができた15°回転の場合の，典型的な入力空間の変遷の例を，図6.16に示す．初めに二つのR状況に多数の穴が空き，形状が次第にゆがんでいく．R状況は，累積行動数1000回までに，ほぼ15°の傾きを持つ形状を形成する．四つのT状況の形状は，その時までに若干変化するものの，まだ傾きは小さい．その後，R状況の形状が整うにつれて，T状況の傾きも大きくなる．そして，累積行動数2000回までに状況の変化は安定し，初めの入力空間を反時計回りにちょうど15°回転させたような入力空間が得られる．このように，まず，R状況の変化が先行し，T状況の変化はそれに追随して起こる．

状況の意味の切り替えは，10°回転の場合には，10試行中で一度も起こらなかった．すなわち，理想的な意味を維持したまま，状況の形状の変化だけで適応することができた．15°回転の場合には，10試行中一つの試行で一度，理想的な意味と異なる意味に変更された後で再び理想的な意味に変更された．理想的な意味を獲得したのは，累積行動数2090回の時であった．その他の9試行では，理想的な意味を維持したまま適応することができた．

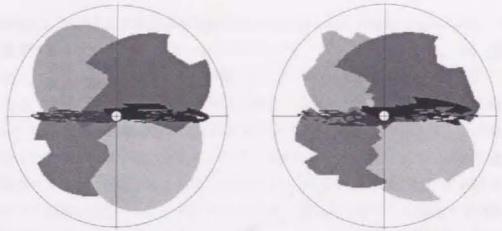


a. 10° ～ 25° 回転

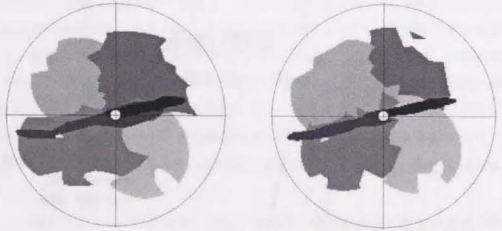


b. 90°，180° 回転

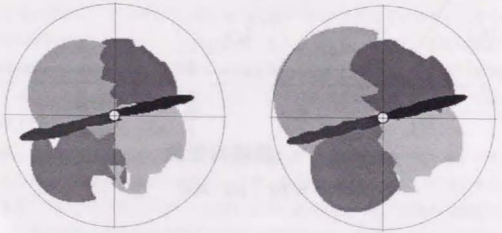
図 6.15: ゴールまでの平均行動数の変化



a. 累積行動数 300 回　b. 累積行動数 600 回



c. 累積行動数 1000 回　d. 累積行動数 1300 回



e. 累積行動数 1600 回　f. 累積行動数 2000 回

図 6.16: 15° 回転後の入力空間の変遷

20° 以上の回転の時には，10000 回行動しても理想的な意味を獲得できないケースがあった．そのようなケースは，20° 回転の場合には 10 試行中で 2 試行，25°，90°，180° 回転の場合には各々の 10 試行中でいずれも 1 試行であった．これらの 5 試行は，いずれも図 6.17に示すのと類似した理想的な入力空間の分割を獲得していたが，この図において，状況 6 の位置にあって状況 5 に遷移するのと同様な意味を持つゴールまで 3 ステップの T 状況が存在していた（4 試行では一つ，1 試行では二つ）．このナビゲーション問題では，6.1.2節に述べたように，マスタープラン中の目標状況に到達するまで行動を継続するので，状況 6 から状況 5 を目標状況として行動する途中で状況 3 を通過したとしても，そこでは行動が停止されないのである．そして，状況 5 まで進んでから折り返して状況 3 に戻ることになる．このような行動は，入力空間中でアルファベットの J の字に似た軌道を通るので，説明のために「J 型行動」と名付け，J 型行動を条件行動とする T 状況を「J 型 T 状況」と名付ける（図 6.17参照）．入力空間中に J 型 T 状況が存在すると，ゴールまで 2 回の行動で到達できない場合が増えて，性能が悪化する．J 型 T 状況が一つ存在する場合にはゴールまでの平均行動数が 2.5 程度，二つ存在する場合には 2.8 程度にまで増加した．

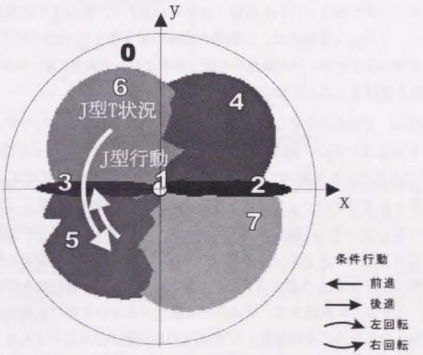


図 6.17: J 型行動と J 型 T 状況（0 ～ 7 は状況を示す）

J 型 T 状況の持つ意味は，理想的な意味と同じ条件行動を持つ．図 6.17の状況 6 の場合では，いずれも右回転が条件行動である．STNS では，6.1.2節で述べたように，

貪欲政策を用いていても，行動選択の制限によって他の行動が選択され，行動政策の探索を行うことができるが，目標状況についてはうまく探索されない．従って，J型T状況の持つ意味は，一度安定してしまうとなかなか理想的な意味に切り替えられない．しかし，安定した後でも，状況6の位置にあるJ型T状況の意味が，状況4を目標状況とする意味に切り替わることは稀に起こる．この意味では，理想的な意味と異なる条件行動を持つので，その後，状況3を目標状況とする理想的な意味に切り替わることがあり得る．このような形で二段階で理想的な意味に切り替わるケースが，累積行動数10000回までに1回起こっていた．さらに学習を続けた場合，累積行動数20000回までに3回起こった．すなわち，20°回転の場合から180°回転の場合まで合わせて全部で7個あったJ型T状況のうち4個が，累積行動数20000回までに理想的な意味に切り替えられていた．一度理想的な意味を獲得できた状況は二度と他の意味に切り替わることはなかったので，十分に長く学習を続ければ，いつかは理想的な意味に収束すると考えられる．ただし，この二段階の意味の切り替えのうち一段めでは，ゴールまで3ステップの状況から同じくゴールまで3ステップの状況に切り替わるので，それぞれの場合の将来の報酬の期待値は大差がなく，かなり起こりにくい切り替えであるといえる．さらに，累積行動数20000回の段階でも残ったJ型T状況のうちの二つは，同じ試行の中で隣あっている状況（図6.17において，状況4と状況6に相当する）であった．このような場合は，二段階の意味の切り替えの一段めがゴールまで3ステップの状況から4ステップの状況への切り替えとなって非常に起こりにくいので，理想的な意味を獲得することは非常に困難である．

J型T状況は，ゼロから学習した場合には一度も生成されなかったので，この柔軟性テストで生成されたのは，過去の影響をひきずっているためであると思われる．J型T状況ができるケースを調べてみたところ，いずれも環境の変化によってR状況とそれに遷移するT状況が一つだけ残されている場合に，新たな状況としてJ型T状況が抽出されていた．さらに詳しく調べてみると，図6.18に示す三つの現象が，J型T状況生成の主な原因となっていると思われる．(1)は「R状況3が先に変化に適応して傾いた時に，T状況5の傾きはまだ不十分であるために一部がはみ出してしまう」という現象，(2)は「R状況3は，周辺部では変化が大きいために正事例を集められず，中央に縮んでしまう．その結果，T状況5の周辺部がはみ出してしまう」という現象，(3)は「R状況3の形状の変化によってT状況5に多数の穴があき，穴の内部は状況0となる」という現象である．いずれの現象も状況0から右回転によってT状況5に遷移する事例を増やす方向に働く．(1),(2)の現象は，状況0から右回転によってR状況3に遷移する事例を減らす方向にも働く．その結果，状況6の場所に，右回転によってT状況5に遷移する意味を持つJ型T状況が抽出されることになる．(1),

(2),(3)はいずれもT状況の適応がR状況よりも遅れることによって生じる現象で，ゼロからの学習では起こらない．従って，ゼロからの学習では，J型T状況は生成されない．

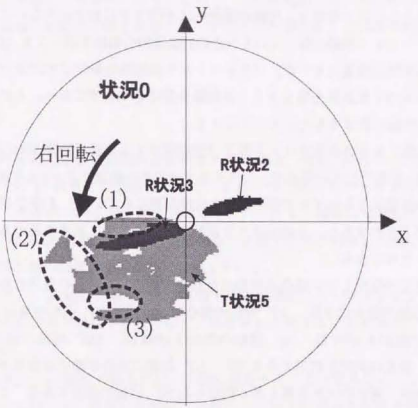


図 6.18: J型T状況生成の原因

この柔軟性テストにおいては，多くの場合は理想的な状況の意味を獲得できたが，J型T状況というローカルミニマムが存在し，そこに落ち込んでしまう場合があり得ることがわかった．上述のように，R状況とそこに遷移するT状況が一つだけ残る場合にだけこの現象が生じるので，過去の状況を利用するか，状況を削除して一から学習し直すかの境目で，J型T状況が生成されやすいということができる．この現象を避けるためには，

1. ワールドモデルの内部で，状況遷移確率と遷移による報酬の期待値を，目標状況ごとに分けて記憶する．
2. プラン中で，連続して対称的な行動を選択することを禁止する．
3. 対称的な行動を条件行動として持つT状況に遷移するT状況の抽出を禁止する．

4. 目標状況の選択を確率的にする．

などの対策を用いることができる．しかし，1は，6.1.2節でも述べたように，ワールドモデルが大きくなり過ぎるので，非現実的である．2, 3は，対称的な行動を連続して出力して初めて報酬が得られるようなタスク（例えば，釘を打つタスクなど）への適応性を失うことにつながる．行動の選択法はSTNSでは特に定めないことにしたので，ナビゲーション問題に関していくつかの行動選択の制限を設けても（6.1.2節参照）STNSの汎用性に問題はないが，プランニングや状況抽出条件はSTNSの内部の問題なので，なるべく汎用性を損なうような制限を設けるべきではない．4は，確率的にした分だけ性能の低下をもたらすことになる．

この柔軟性テストの結果では，J型T状況が存在する累積行動数5000回の段階で，20°～180°回転の全ての場合で，ゴールまでの平均行動数2.2～2.3を達成しており，ゼロからの学習よりもわずかに悪い程度の性能が得られている．J型T状況の生成を無理に避けようとすると，上述のような問題が生じるので，敢えて避ける必要はないと考えることもできる．

J型T状況が発生しない場合の理想的な意味を獲得した時点の累積行動数の平均値は，10°回転の場合は0回，15°回転の場合は209回，20°回転の場合は2757回，25°回転の場合は3778回，90°回転の場合は1944回，180°回転の場合は2510回であった．過去の状況を利用できる10°，15°回転の場合が適応が最も早いが，その次に早いのは，過去の状況を最も多く削除した90°回転の場合である．この結果からも，過去の状況を利用するか，状況を削除して一から学習し直すかの境目で，適応が最も困難になることがわかる．

このセンサの故障に対する柔軟性テストでは，STNSが，環境の小さな変化[2]には柔軟に適応でき，大きな変化にもほぼ安定して対応できることがわかった．小さな変化と大きな変化の境目で最も適応が難しく，J型T状況のようなローカルミニマムに陥る恐れはあるが，その可能性は低く，平均的にはゼロからの学習に匹敵する性能が常に得られることが分かった．小さな変化への対応と大きな変化への対応も，自律的に，おおむねスムーズに切り替えることができるといえる．

6.3.2 アクチュエータの故障に対する柔軟性テスト

問題設定

STNSの柔軟性をテストするためのもう一つの実験として，学習が収束したSTNSの行動出力を変化させて再学習する実験を行った．問題設定，およびシステムは6.2節

[2]センサやアクチュエータの故障は，STNSにとっては，環境の変化とみなすことができる．

のアクチュエータが故障したローバーを用いた2次元入力のナビゲーション問題と全く同じものを使用する．ただし，STNSの初期設定として，6.3.1節の「センサの故障に対する柔軟性テスト」と同じものを用いる．すなわち，図6.3に示す入力空間を持つSTNSを初期状態とする．このSTNSの初期設定は，「前進」および「後進」の場合に直進する設定で学習済みのものなので，この問題は，ローバーの学習が収束した後に，左車輪の駆動系の調子が悪くなってしまった場合の，STNSの適応性を調べるものである．

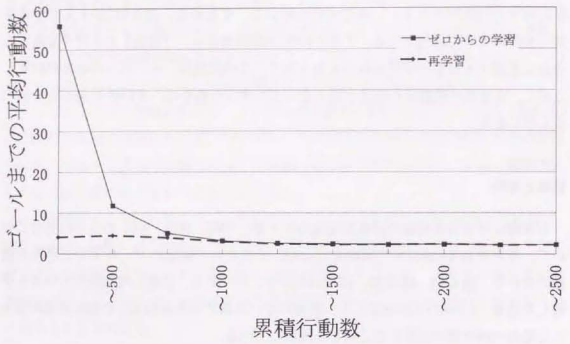
結果と考察

右車輪に対する左車輪の回転角速度の比rが，95%，90%，85%の三つの場合について，それぞれ10試行ずつ実験を行った．それぞれの場合のゴールまでの平均行動数の変化を，図6.19，図6.20，図6.21に示す．いずれも，故障した状態でゼロから学習した結果（6.2節の実験結果）と，故障のない状態で学習が収束した後に故障が発生した場合の再学習の結果とを比較して図示している．

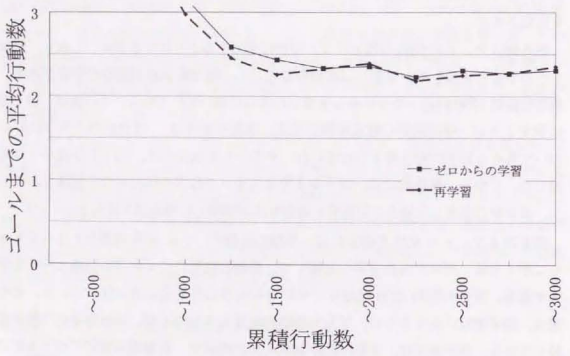
図6.19より，$r = 95\%$の場合には，かなり柔軟に適応しているといえる．この場合には，全ての状況が，消滅することなく，意味を切り替えることもなく，うまく形状を変化させた．

図6.20より，$r = 90\%$の場合には，柔軟に適応することはできない．しかし，ゼロからの学習と比較して，早期には累積行動数にして約250回遅れ程度で学習が進み，累積行動数1750回辺りでゼロから学習した場合に追い付いている．その後は，図6.20bに示すように，ほぼ同等の収束状態に至る．状況の意味は，10試行のうち2試行において各々4回ずつ切り替えられていた．そのうち1試行では，元のT状況が一つ消滅した．しかし，最終的には，ゴールまで2ステップ以内の状況は全て認識されており，ゼロから学習した場合と同程度に適切な状況認識と行動規則が獲得されていた．

図6.21より，$r = 85\%$の場合には，早期の性能が，$r = 90\%$の場合と比べてもさらに悪くなる．ゼロからの学習と比較して，累積行動数にして約750回遅れ程度で学習が進み，累積行動数2250回辺りでゼロから学習した場合に追い付いている．その後は，図6.21bに示すように，どちらの学習曲線もふらつくが，ほぼ同等の性能を維持している．状況表現は，累積行動数10000回の段階で，R状況は常に二つとも認識され，ゴールまで2ステップのT状況は平均で3.6個，図6.12の状況8, 9に相当するゴールまで3ステップのT状況は10試行合わせて2個認識されていた．ゼロからの学習の場合には，それぞれ2個，3.5個，3個なので，ほぼ同等の状況認識と行動規則が獲得されているといえる．この学習の場合に，早期の性能が著しく悪化するのは，

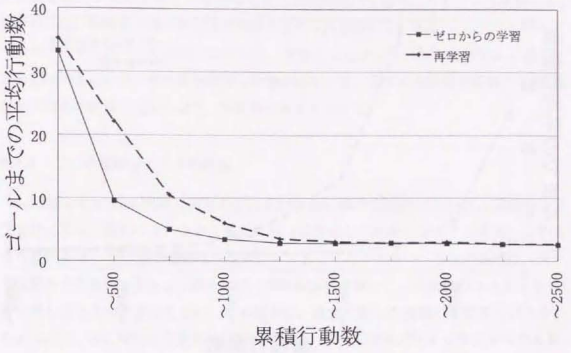


a. 累積行動数 250 回刻み

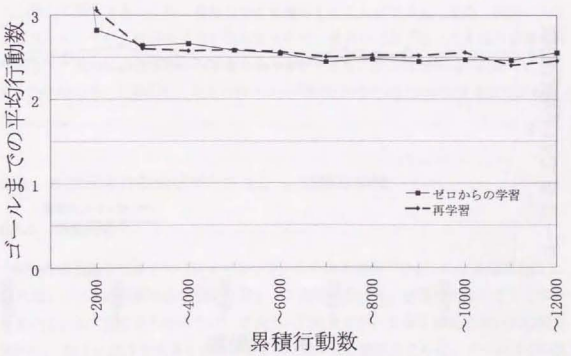


b. 累積行動数 250 回刻み

図 6.19: r=95% の場合のゴールまでの平均行動数の変化

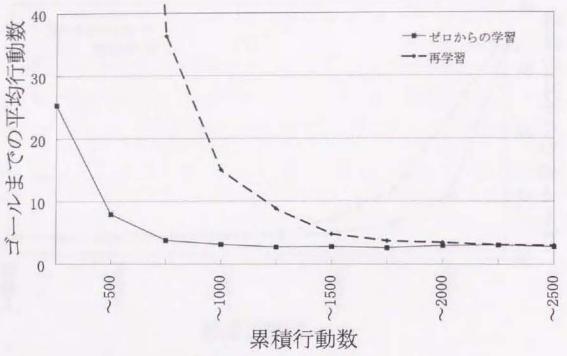


a. 累積行動数 250 回刻み

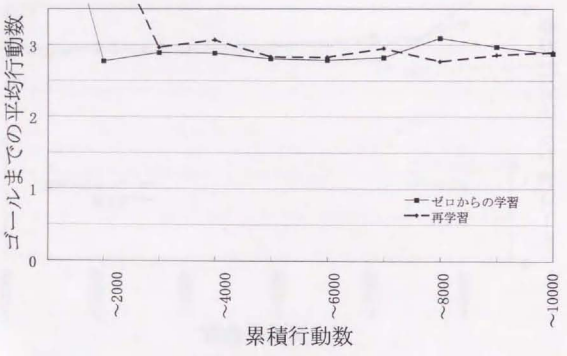


b. 累積行動数 1000 回刻み

図 6.20: r=90% の場合のゴールまでの平均行動数の変化



a. 累積行動数250回刻み



b. 累積行動数1000回刻み

図 6.21: r=85% の場合のゴールまでの平均行動数の変化

故障に対応できない状況が消滅するのに，少し時間がかかるためである．もともとのR状況2個とT状況4個のうち維持できた状況の数の平均は，それぞれ1.3個と1.1個であった．報酬から遠い状況ほど消滅するケースが多いといえる．

このアクチュエータの故障の場合でも，環境の変化が大きい場合には柔軟に適応することはできないが，ゼロから学習した場合に比べて，遅くとも累積行動数約1000回遅れで同等の性能に達するので，安定性はあるといえる．

6.3.3 二つの柔軟性テストの結論

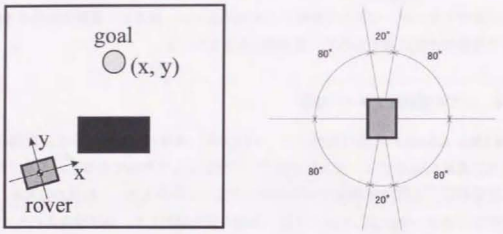
6.3.1節と6.3.2節の実験の結果から，STNSが，簡単な問題において，環境の小さな変化に柔軟に適応でき，大きな変化にもほぼ安定して対応できるといえる．このような柔軟性は，ASRRの動的な状況表現によるところが大きい．状況の表現を，グリッド表現などを用いて固定した場合には，環境の変化に対して，各状況のユーティリティを学習し直さなければならない．この場合は，過去の誤った知識の影響をしばらく引きずるので，特に報酬から離れた状況の場合は，学習の遅れが大きくなりがちである．一方，STNSでは，環境の小さな変化に対しては，状況の形状を変化させるだけで対応できるので，学習が速く進む．学習中の性能も，過去の知識を利用することによって，著しく悪化することなく維持したまま適応することができる．また，環境の大きな変化に対しては，既存の状況が消滅するので，過去の大きく誤った知識の影響を断ち切って，新たに白紙状態から学習することができる．小さな変化への対応と大きな変化への対応も，自律的に，おおむねスムーズに切り替えることができることがわかった．

6.4 8次元入力のナビゲーション－複雑な問題

6.4.1 問題設定

STNSの複雑なアプリケーションとして，8次元入力のナビゲーション問題を扱う．図6.22aに，この問題の作業空間を示す．この図において，空間の比率は正しく保たれている．この図に示したように，2次元入力のナビゲーション問題で用いた作業空間中に，33.3×16.7の大きさの障害物を一つ置いた．簡単のために，ゴールはこの図の位置に固定した．ローバーには，図6.1に記したゴールセンサからの知覚入力（ローバー固定座標系でのゴールの(x, y)座標）に加えて，図6.22bに示す障害物センサからの知覚入力を与える．知覚入力は合わせて8次元となる．障害物センサは図に示した各々のセンサの視野内で最も近い障害物（周囲の壁を含む）までの距離を出力する．

その他のタスク，報酬，行動の設定は，すべて6.1節の問題と全く同じものを使用する．



a. 作業空間　　b. 障害物センサ

図 6.22: 8次元入力のナビゲーション問題

6.4.2 結果と考察

上のナビゲーション問題のための行動政策を学習するSTNSをつくり，シミュレーションで実験を行った．各定数値は，

$$N^R_{init} = N^R_{min} = 9,\ N^T_{init} = 20,\ N^T_{min} = 12,\ r^R_{min} = 5.0,\ R^T_{min} = 1.5,$$
$$\gamma = 0.7,\ p_{min} = 0.2,\ n_{max} = 7$$

と定めた．履歴データベース中のデータ数は10000個とし，超楕円体の大きさと状況の重なり方の更新は100回の行動ごとに行うと定めた．N^R_{init} と N^R_{min} は，8次元の空間の中で8次元の部分空間を保つために最小限必要なデータ数である．ただし，$N^R_{init} = N^R_{min} = 12$ として同じ実験を行った場合にも，性能にほとんど差は見られなかった．履歴データベース中のデータの数は，5000個では状況が安定せず良い性能が得られなかったので，10000個に定めた．p_{min} は，2次元入力の実験で用いた値0.1よりも大きく設定されているが，これについては後で詳しく述べる．その他の値は，恣意的に定めているが，特に細かく調整する必要はなかった．

シミュレーションは一回の試行で28000回の行動を実行する．入力空間中に，ゴールに到達した状況（図6.23，図6.24の状況1）と残りの全体を占める状況0の二つの

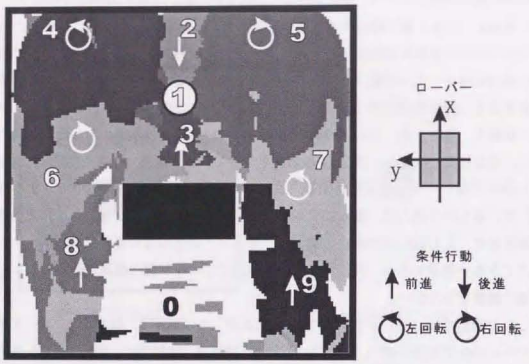
状況のみが存在するところから，学習を開始する．ゴールに到達した時には，ゴール以外の任意の位置にローバーを置き直して学習を続けた．それを10試行行った．

シミュレーション終了時の作業空間の分割の例を，図6.23，図6.24に示す．この図は，ローバーの姿勢を固定することによって，入力空間中の各状況を作業空間に射影したものである．この問題ではゴールの位置を固定しているので，ローバーの姿勢を固定すると，作業空間中の一点を入力空間中の一点に対応させることができる．ローバーが真上，真下，右，左の四つの方角を向いている場合のみ図に示した．説明のために，主な状況に状況0～状況11の名前をつけている．前進，後進，左回転，右回転で矢印の方向にローバーが移動する．この図の分割例は，10試行の中でうまく認識されているものを選んだ．図6.25には，比較のために，理想的と思われる作業空間の分割を示す．この図の中の余白の部分は，ローバーがその時の姿勢では，存在することのできない領域である．従って，この領域内でどんな状況と認識されるとしても，行動に影響を与えない．

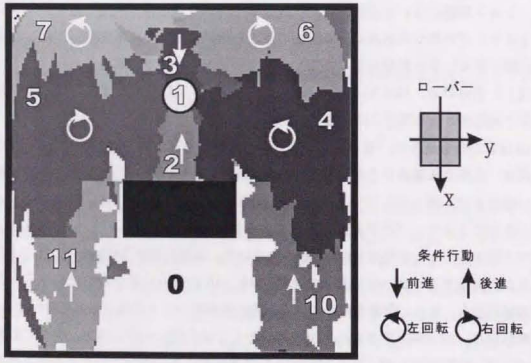
この分割例では，主な状況として，R状況が二つ（状況2，状況3），ゴールまで2ステップのT状況が四つ（状況4，状況5，状況6，状況7），ゴールまで3ステップのT状況が四つ（状況8，状況9，状況10，状況11）認識された．これらの10個の状況は安定して維持された．ゴールまで2ステップ以内の状況は2次元入力のナビゲーション問題において認識された状況と全く同じ役割を持つ状況である．ゴールまで3ステップの四つの状況は，障害物の陰から抜け出すために重要で，これらの状況が認識されていると性能が良くなる．これらの状況を，以下の説明のために，「回廊状況」と名付ける．図6.25と比べると，これらの10個の状況に関しては，理想的な状況と対応がとれる程度には認識されているといえる．

10試行全体を見ると，R状況は常に二つとも認識され，ゴールまで2ステップのT状況は，全部で4個あるうち，平均で3.6個認識されていた．このT状況が四つ揃わない場合には，図6.26に示すように，回廊状況（状況7）が縦に長い形状となって，ゴールまで2ステップのT状況が欠けている領域をカバーしていた．回廊状況は，全部で4個あるうち，平均で2.6個認識されていた．累積行動数28000回までに，図6.23のように，回廊状況が四つとも認識されたのは，10試行中わずかに1試行であった．他の試行でも，さらに学習を続ければ，回廊状況が四つとも認識される可能性はある．回廊状況は，ゴールまで3ステップの状況としては安定しやすいので，学習に時間をかければ，次第に図6.23，図6.24のような空間分割に落ち着いていくと予想される．ゴールまで3ステップの状況で安定して維持されていたのは，回廊状況のみであった．また，ゴールまで4ステップ以上の状況が安定して維持されることはなかった．

図6.27にゴールまでの平均行動数の変化を示す．図6.27aでは各試行を500回の行

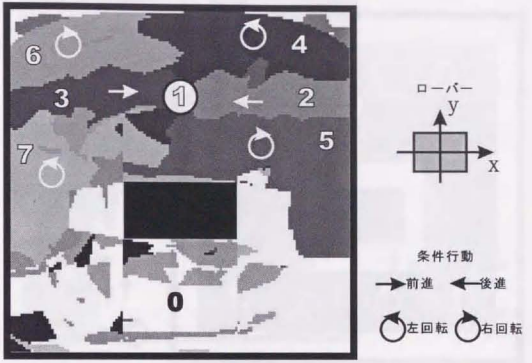


a. ローバーが真上を向いている場合

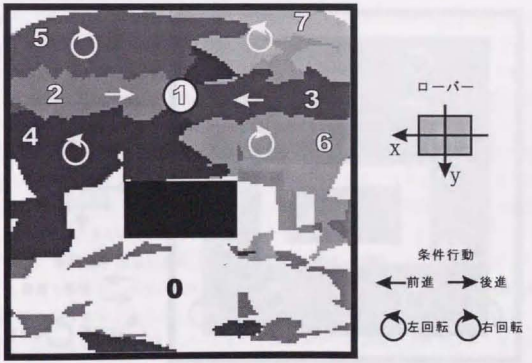


b. ローバーが真下を向いている場合

図 6.23: シミュレーション終了時の作業空間分割 1 （0 ～ 11 は状況を示す）

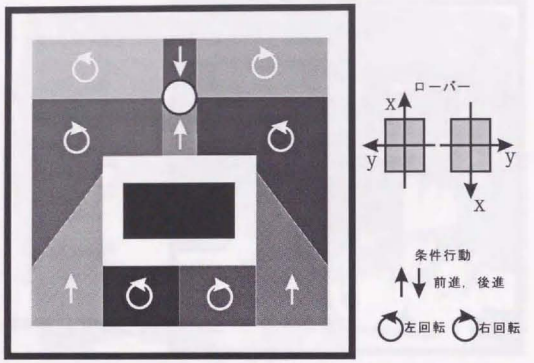


c. ローバーが右を向いている場合

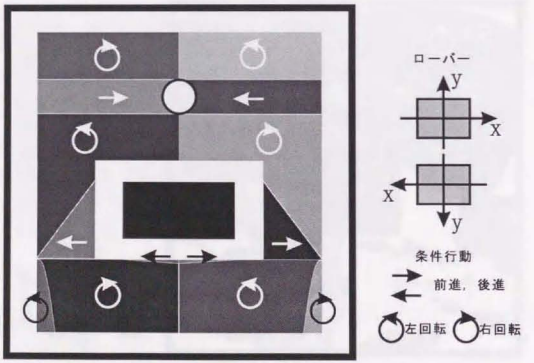


d. ローバーが左を向いている場合

図 6.24: シミュレーション終了時の作業空間分割 2 （0 ～ 7 は状況を示す）



a. ローバーが上または下を向いている場合



b. ローバーが右または左を向いている場合

図 6.25: 理想的な作業空間分割

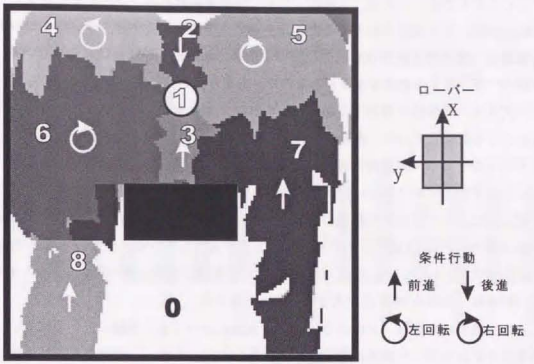


図 6.26: ゴールまで2ステップの状況が欠けている例（0～8は状況を示す）

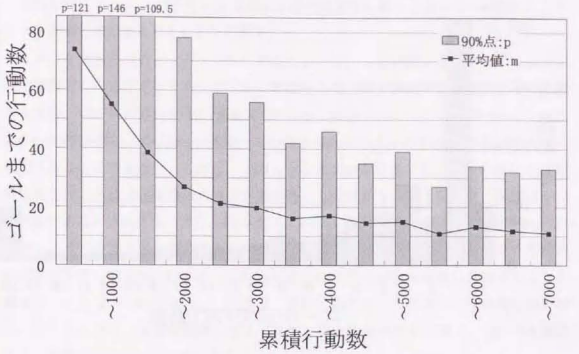
動ごと，図6.27bでは2000回の行動ごとに区切り，それぞれの中で平均した．学習は2次元入力の場合と比べると非常にゆっくりと進む．これは，一つには，次元が上がったせいで，各状況の形状を維持するのに非常に多くの事例を必要とすることが原因である．8次元入力の場合は，状況抽出に必要な正事例の数も多く，また各状況を維持するために2次元入力の場合の4～5倍程度の数の事例を用いている．もう一つには，8次元入力の場合は問題が難しく，ゴールに到達するデータや役に立つデータの密度が薄いことが原因である．図6.23，図6.24に示したように，8次元入力の問題では，状況0の領域も大きく残されており，また，状況がゆがんでいて望ましいデータが得られない領域もあるので，得られたデータのかなりの部分が役に立たないデータとなってしまう．例えば，累積行動数27001～28000回のデータのうち，状況0のデータは8.7%もある．また，状況の意味に適合する状況遷移の確率は，累積行動数28000回の段階で70%前後であまり良くなく（累積行動数27001～28000回の部分のデータでは，R状況で平均69.8%，ゴールまで2ステップのT状況で平均70.8%，ゴールまで3ステップのT状況で平均62.0%であった），状況の形状がゆがんでいることを示している．これらの原因で，過去のデータをかなり長く（累積行動数にして10000回分の長さ）残す必要があるので，環境やシステムの内部表現の変化に対する対応も遅くなる．図6.27bから，累積行動数28000回の辺りでもまだゆっくりと学習が進んで

いることがわかる．これは，上述の「さらに学習を続ければ，ゴールまで3ステップの状況が四つまで抽出されてくるだろう」という予想と一致する．ゴールまでの平均行動数は，累積行動数28000回の段階で7前後であった．理論値はわからないが，2回以内でゴールする領域もかなりあるので，あまり良い値とはいえない．90%点はさらに大きく，平均値の倍以上であった．これは，ゴールまでの行動数にばらつきが大きいことを意味している．実際の行動を観察しても，障害物に邪魔されない場所では，早くゴールするが，障害物の裏に入ってしまったり，障害物の角に引っかかってしまうとなかなかゴールできない様子が観察された．

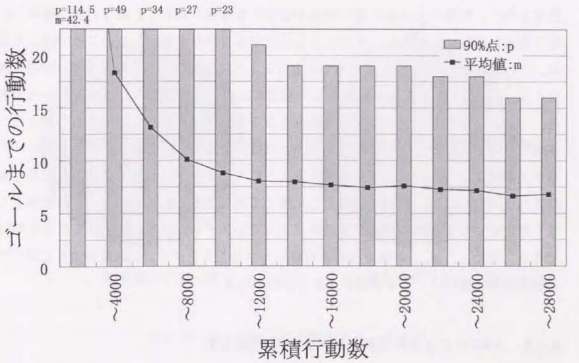
図6.28にゴールまでの行動数のヒストグラムを示す．累積行動数27001～28000回の部分のデータを10試行分あわせてヒストグラムとしている．全体の40%が行動数2回以内でゴールしている反面，ゴールまでの行動数が非常に多い場合もかなりある．この結果は，上述の90%点が大きいことと一致する．

この実験の結果から，STNSが，複雑な問題においても，報酬に近い領域では，ある程度の状況認識と行動規則を獲得できるといえる．しかし，報酬から離れた領域では，状況認識が不十分であるために，適切な行動規則を獲得することができなくなる．その原因は，主に，ASRRの状況形状を学習する能力の低さであると思われる．入力空間における状況の形状は容易に想像できないが，その作業空間への射影（図6.23，図6.24，図6.26）を見る限りでは，2次元入力の場合の4～5倍の事例を用いているにも関わらず，その形状はゆがんでおり適切であるとはいえない．行動規則に適合する状況遷移の確率は，上述のように70%程度であるが，そのために，3回遷移しただけで，成功確率はすでに34%に落ちてしまう．報酬に近い状況の形状がゆがんでいると，その状況に遷移してくる状況全てに悪影響が伝わり，図5.8に示したように，報酬から離れるごとに状況の形状のゆがみが蓄積されていくので，問題は深刻である．ただし，人間の場合も，報酬から離れるに従って，状況の認識は曖昧になっていくと思われるので，この問題は本質的に避けられないものであるかも知れない．例えば，図6.22aに示す作業空間上の各点での各姿勢において，最適な行動を計画するのは，ゴールまで4ステップ以上の行動を要する場合は，人間にとってもそれほど簡単な問題ではない．

最後に，先読みプランニングの足切りの確率p_{min}の設定について説明する．この値を小さくすると，成功確率が低いプランも先読みし，その状況における行動の最適性という意味では，システムは改善される．しかし，その場合，かえって性能が悪くなるという現象が観察された．それは，過度に深いプランニングをすると，得られる確率の低い報酬に基づいてあまり重要でない状況が長期間存続して，有意義な状況ができるのを妨げるからである．STNSでは報酬がなるべく多く得られるような行動を選



a. 累積行動数500回刻み



b. 累積行動数2000回刻み

図6.27: ゴールまでの平均行動数の変化

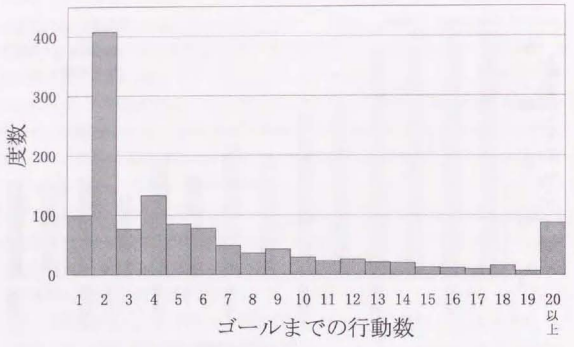


図 6.28: ゴールまでの行動数のヒストグラム

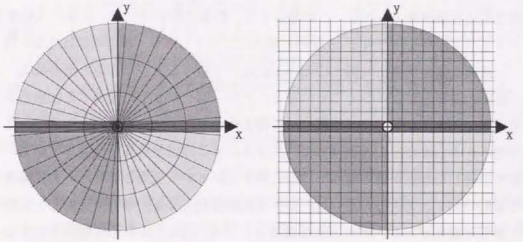
択するので，報酬の見込みの薄い状況が安定して維持されることはないが，履歴をかなり長くとっているために，そのような状況でもなかなか消滅しないことは起こり得る．この問題において，$p_{min} = 0.2$ に上げることにしたのは，2次元入力の問題と同じく $p_{min} = 0.1$ として深く先読みした場合には良い性能が得られなかったからである．しかし，プランニングで深く先読みすることと，状況を長く維持することとは，本来は分けて考えることができる．STNS では，消滅条件を簡単にするためにユーティリティが負になった時に状況を消滅するようにしているが，状況が必要以上に長く維持されないためには，消滅の境界を色々変えて試してみる必要がある．この境界をうまく設定することができれば，深読みによってシステムの性能が向上することも期待できる．ただし，p_{min} の値を決める時には，プランニングの計算時間が深さに対して指数関数的に延びることも考慮に入れる必要がある．

6.4.3 ASRR による状況表現とグリッド表現の比較

この節では，STNS の動的な状況認識法である ASRR による状況表現と，固定的なグリッド表現との比較を行う．グリッド表現は，4.4.1節で述べたように，強化学習法において最も一般的な状態表現の一つである．STNS が，グリッド表現を用いた強化学習に比べて大きな柔軟性を持ち，環境の小さな変化にも大きな変化にもより早く

適応できるであろうことは，6.3.3節で考察した．ここでは，8次元入力のナビゲーション問題を例にとって，ASRR による状況表現の抽象度の高さと抽象化の適切さを，グリッド表現と比較することによって示す．

まず，8次元入力のナビゲーション問題において，入力空間をグリッドで分割して強化学習を行うことを考える．グリッド表現として，ゴールおよび障害物までの距離を3段階，ゴールの方角を10°単位で36段階に離散化したとする．ゴールセンサ空間の分割の様子を，図 6.29a に示す．この図からわかるように R 状況の部分をうまく表現することはできない．しかし，このような大まかな分割でも，状態の数は 78,732 個になる．これは強化学習のための状態の数としてはかなり大きいので，学習はかなり困難であると思われる．また，別のグリッド表現として，ゴールセンサ空間を，ゴールがちょうど収まる大きさの正方形で分割してみる．この場合のゴールセンサ空間の分割の様子を，図 6.29b に示す．この場合は，R 状況の部分も正確に分割することができる．しかし，この場合は，各状態から回転行動によって移動した時の遷移先がばらつくことになり，学習が困難になる．また，状態の数はさらに増え，321,489 個になり，学習はほとんど不可能であるといえる．



a. 極座標による分割　　b. 直交座標による分割

図 6.29: ゴールセンサ空間のグリッド表現

一方，ASRR を用いた場合は，不完全な状況認識とはいえ，10 個程度の状況で全入力空間を表現することができた．ASRR よる状況表現と同様の「行動に基づく分割表現」で理想的な状況認識が実現された場合には，数十個の状況で全入力空間を表現

することができると推測される．これは，十分に強化学習が可能な数であり，ASRRよる状況表現が，グリッド表現に比べて，非常に抽象度の高い表現であるといえる．

しかも，ASRRでは，入力空間を自由度の高い曲面で切ることができる．そのため，様々な知覚入力に対して適切に抽象化を行うことができる．例えば，6.3.1節のセンサの故障に対する柔軟性テストで示したように，センサが0°から360°までのどんな角度で取り付けられていたとしても，ASRRでは全く同じように入力空間を表現することができる（図6.16f参照）．一方，グリッド表現では，図6.29のa, bのどちらの表現を用いても，任意のセンサ取り付け角度に対応することは，明らかに無理である．aの極座標を用いた場合は，10°周期で性能が大きく変化することが予想される．bの直交座標を用いた場合は，取り付け角度のずれが0°，90°，180°，270°の時以外では，かなり性能が悪化すると思われる．また，6.2節の左車輪が故障した場合にも，ASRRでは，若干精度は悪くなるものの，弧状のR状況を表現することができた（図6.10，図6.11，図6.12参照）．一方，グリッド表現では，図6.29のa, bのどちらの表現を用いても，弧状の状況を表現すると非常に精度が悪くなってしまう．

もっとも，現在の知覚入力がASRRに有利なのは確かである．8次元入力のナビゲーション問題は，ゴールを固定しているので，実際の自由度は3である．従って，最も単純な状態表現としては，作業空間をグリッドで分割して，さらにローバーの方角を離散化する表現が考えられる．この場合では，作業空間をゴールの大きさ（直径10）のグリッドで分割し，方角を36段階に離散化したとしても，状態の数は3,600個なので，一般的な強化学習法（例えばQ-learning）で扱うことができると思われる．ただし，行動の遷移先がばらつくことになり，かつ，ゴールまでに通過する状態の数が最大ではかなり多くなるので，学習には時間がかかり最適政策の獲得も困難であることが予想される．また，この表現では，あまりにもこの環境に特化し過ぎているので，学習で獲得した知識が他の環境では全く使えないという欠点がある．一方，障害物センサを用いた8次元入力の場合は，少々環境が変化しても，学習で獲得した知識を利用できる可能性がある．そのような環境としては，図6.22aに示す作業空間を少しゆがませた環境や，部分的に類似している環境などが考えられる．

上述の3次元の知覚入力（作業空間上の位置と方角）からSTNSで学習を行った場合は，うまく状況を抽出することができなかった．それは，ASRRでは状況表現の中で平均と分散を用いているので，方角のような循環する属性をうまく扱えないためである．ASRRでは，属性空間中で正事例が正規分布に近い分布で存在し，状況が超楕円体に近い形状をしている場合に，最も正確に状況を表現することができる．このように，もちろんASRRにも得意な属性と不得意な属性があり，性能を上げるためには，知覚入力を得意な属性を用いて表現するなどの工夫が必要である．

6.4.4 IPRLとpolicy iterationの比較

この8次元入力のナビゲーション問題でも，IPRLとpolicy iterationを比較してみる．まず，この実験で獲得されたワールドモデルから，policy iterationを用いて最適政策を計算してみた．ゴールまで2ステップ以内の状況では，最適行動は，すべて，図6.25に示した理想的な行動政策と一致した．しかし，障害物の除から抜け出すのに重要な「回廊状況」は，全部で26個のうち，20個もの状況の最適行動が，理想的な行動政策とは異なっていた．そのうち，10個の回廊状況では，履歴データベース中で一度も実行されていない行動が最適行動として選ばれた．一度も実行されていない行動では，すべての状況に同じ確率で遷移すると仮定していて，さらに一度も経験されていない遷移には+1の報酬が与えられているので，このようなことが起こる．この場合は，一度その行動を実行して失敗すれば，その行動の評価が大きく下がって，最適行動が理想的な行動と一致する可能性がある．また，図6.26に示したようなゴールまで2ステップの状況が欠けている場合の縦に伸びた回廊状況（状況7）において，ゴールまで2ステップとなる行動（状況7の場合では，左回転）が最適行動として選ばれている例も一つあった．しかし，残りの9個の回廊状況においては，理想的な行動以外の行動が，何度か試みて失敗しているにも関わらず，最適行動として選ばれていた．そのような状況では，理想的な行動によるゴールまで2ステップの状況への遷移の確率が，4～6割とあまり高くなかった．また，最適行動として選ばれた行動では，図6.25に示した理想的な状況分割が実現されていれば有効な遷移は存在しないはずだが，いくつかの状況へ2～6割程度のやや高い確率で遷移していた．これは，ワールドモデルが十分に学習されていないこと，および状況の形状のゆがみがゴールから離れるにしたがって大きくなることが原因である．

一方，同じワールドモデルから，IPRLを用いて行動政策を計算したところ，ゴールまで2ステップ以内の状況では，すべて，図6.25に示した理想的な行動が選択された．また，回廊状況でも，全部で26個のうち，22個の状況で，理想的な行動が選択された．残りの4個のうちの3個の状況では，ゴールまで2ステップの状況が欠けている場合の縦に伸びた回廊状況において，ゴールまで2ステップとなる行動が最適行動として選ばれるケースであったので，これも理想的な行動が選ばれたとみなすことができる．結局，理想的な行動以外の行動が選択されたのは，わずかに一つの回廊状況のみであった．そのケースでは，回廊状況から右回転によってゴールまで2ステップの状況に高い確率（6割6分）で遷移していて，そのために行動政策として右回転が選択されていた．図6.25に示した理想的な状況分割では，そのような遷移は起こらないので，これは状況の形状のゆがみによるものである．しかし，policy iterationを

用いた場合に比べると，このような状況の形状のゆがみなどによって理想的な行動以外の行動が選択されるケースが，非常に少ないことがわかる．

この原因は，次のように説明される．policy iterationでは，ある行動による遷移先が分散していて各々への遷移確率が低くても，各々がそれなりに良いユーティリティを持っていれば，それらを足し合わせることによってその行動の評価が高くなる．一方，IPRLでは，確率の高い遷移だけを考慮するので，ある行動による遷移先が分散している場合には，各々がそれなりに良いユーティリティを持っていても，その行動の評価が高くならない．T状況の理想的な行動というのは，図6.25に示したように，ある決まった状況への遷移となるので，遷移先は分散しない．従って，policy iterationにおいては，理想的な行動以外の行動が，分散している遷移の評価を足し合わせることによって，理想的な行動の評価を上回る評価を得る可能性が高いといえる．

STNSはある行動をして類似した（つまり分散していない）結果が得られる領域を状況として切り出すというシステムなので，policy iterationよりもIPRLの方がSTNSに向いていると考えることもできる．ただし，状況の形状が収束して来れば，理想的な行動以外の行動では結局報酬につながるような状況に遷移できないので，policy iterationでも理想的な行動に収束してくると推測できる．はっきりといえるのは，IPRLを用いた場合よりもpolicy iterationを用いた場合の方が，より積極的に探検を行う手法になるということで，それ以上の比較，すなわち，学習の収束性や獲得される政策の最適性などの比較は，今後の課題である．

次に，8次元入力のナビゲーション問題で獲得されたワールドモデルを用いて，policy iterationとIPRLで求めた各状況のユーティリティを比較してみる．policy iterationでは，そのワールドモデルに関して最適方程式（式5.1）を満たす“本物の”ユーティリティが求められる．IPRLでは，5.3.2節で述べたように，その近似値が求められる．ただし，上述のように，そのワールドモデルに関して最適であることが，その問題に対する政策の最適性を意味するわけではないことに注意する必要がある．

比較の結果を表6.3に示す．表6.3の左には，全10試行のワールドモデルからpolicy iterationを用いて計算したユーティリティの平均値と標準偏差を示す．ゴールまで1ステップの状況，ゴールまで2ステップの状況，ゴールまで3ステップ以上の状況，および状況0に分けて，平均値と標準偏差を求めた．ゴールまで3ステップ以上の状況には，ゴールまで3ステップの状況が30個と，4ステップの状況が4個含まれる．これらの間に大きな違いは見られなかった．この表から，ゴールから離れるほど，ユーティリティのばらつきが大きくなっていることがわかる．これは，ゴールから離れるほど，状況の形状のゆがみが大きいところと小さいところの差が広がってくることを表している．ある状況の形状のゆがみは，上流の状況すべてに影響を与えるので，

ゴールから離れるほどゆがみが蓄積されて，この差が大きくなるのであると思われる．

表6.3: ユーティリティとIPRLによる近似値との誤差

	ユーティリティ		IPRLの誤差	
	平均値	標準偏差	平均値	標準偏差
ゴールまで1ステップの状況	7.81	0.174	0.834	0.0818
ゴールまで2ステップの状況	4.19	0.330	0.784	0.112
ゴールまで3ステップ以上の状況	2.54	0.659	1.84	0.955
状況0	1.08	0.765	0.956	0.650

表6.3の右には，同じく10試行のワールドモデルからIPRLを用いて計算した各状況のユーティリティの近似値と本物のユーティリティとの誤差の平均値と標準偏差を示す．IPRLによる近似値は，6.1.4節で述べた2次元入力の問題の場合と同じように計算した．IPRLで計算した場合は，ほとんどの状況において，本物のユーティリティよりも若干小さく見積もられた．状況0の中に一つだけ本物のユーティリティが負になる状況ができていて，その状況においてのみ，IPRLの方がユーティリティを高く見積もった．その場合にも，誤差は正であるとした．各状況ごとに誤差を求め，ゴールまで1ステップの状況，ゴールまで2ステップの状況，ゴールまで3ステップ以上の状況，および状況0に分けて，平均値と標準偏差を求めた．誤差の平均値はゴールまで1, 2ステップの状況でも，それぞれユーティリティの10.7%，18.7%と非常に大きい．しかし，誤差の標準偏差がかなり小さいので，全ての状況で同じようにずれていることがわかる．従って，ユーティリティに誤差があっても，最適政策が計算できることがわかる．ゴールまで3ステップ以上の状況，および状況0では，誤差の平均値もますます大きくなるが，それよりも，誤差のばらつきが急に大きくなっているのが目につく．これは，ちょうどこの辺りで先読みプランニングの枝刈りが行われていて，ゴールまで先読みできる場合とできない場合で，IPRLによる近似値が大きく異なっていることを表している．ワールドモデルに関する政策の最適性という点では，この誤差は無視することができないほど大きい．従って，このように行動の結果が大きくばらつくような問題では，IPRLはあまり適切な行動政策を学習できないといえる．しかし，上述のように，この問題の理想的な行動という点では，かえってIPRLの行動選択の方が適切である場合も多い．STNSの内部に組み込む強化学習法としては，ユーティリティ計算の精度よりも，トータルとしての政策の最適性の方が重要であり，その点に関してIPRLとpolicy iterationのどちらがより優れているのかは，ま

だ明らかではない．

次に，計算時間を比較してみる．policy iteration では，状況の数によっても異なるが，繰り返し一回あたり，ほぼ 1/1,000 秒のオーダーで計算できる．また，収束までの繰り返し回数は，初期値の政策が最適政策と全く異なる場合でも，三回から多くて五回である．そして，全体の計算は，長い時でも 1/100 秒のオーダーの時間で終了する．一方，IPRL を用いた場合は，計算時間が大きくばらつく．上述の設定で，短い場合には 1/10,000 秒のオーダーで計算できるが，長い場合には 1/10 秒のオーダーの時間がかかる．平均的にはほぼ 1/1,000 秒のオーダーで計算できるので，policy iteration と同等であるといえる．パラメータを調整して，プランの最大長 $n_{max} = 3$ とした場合には，IPRL の計算結果にほとんど影響を与えることなく，計算時間を最長でも 1/1,000 秒のオーダーに短縮することができた．従って，2 次元入力の問題と同様，あらかじめプランの最大長が予想できる場合には，IPRL の計算速度が policy iteration を上回るように設定することも可能であるといえる．

この問題において IPRL と policy iteration を比較した場合，計算時間はほぼ同等であり，学習の収束性や政策の最適性に関してはまだ明らかではない．従って，STNS に組み込む強化学習法として，IPRL と policy iteration のどちらがより適しているのかは，今後さらに詳しく検討する必要がある．

6.5 実験のまとめ

この章では，STNS の性能を調べるために，2 次元入力のナビゲーション問題で四つの実験，8 次元入力のナビゲーション問題で一つの実験を行った．いずれもコンピュータシミュレーションである．6.1節で説明した 2 次元入力の簡単なナビゲーション問題では，STNS が，簡単な問題において，理想的な状況認識と行動規則を獲得できることが確かめられた．また，データが不十分な段階でも状況を切り出すことによって，学習が非常に速く進むことが確かめられた．6.2節で説明したアクチュエータが故障したローバーを用いた 2 次元入力のナビゲーション問題では，STNS が，超楕円体から離れた形状を持つ状況の認識，小さい状況の認識，他の状況に覆われた状況の認識などの点で弱点を持つことが確かめられた．そして，状況認識が不十分になるにつれて，行動規則の学習も困難になることが確かめられた．しかし，問題を徐々に複雑にしていった時の性能の悪化はゆるやかであることが確かめられた．6.3節で説明した二つの柔軟性テストでは，STNS が，簡単な問題において，環境の小さな変化に柔軟に適応でき，大きな変化にもほぼ安定して対応できることが確かめられた．このような柔軟性は，ASRR の動的な状況表現によるところが大きく，環境の変化が小さい時

には，過去の知識を利用して素早く滑らかに適応し，環境の変化が大きい時には，自律的に過去の大きく誤った知識の影響を断ち切って，新たに白紙状態から学習することが確かめられた．6.4節で説明した 8 次元入力の複雑なナビゲーション問題では，報酬に近い領域では，ある程度の状況認識と行動規則を獲得することができた．しかし，報酬から離れた領域では，状況認識が不十分であるために，適切な行動規則を獲得することができなかった．その原因は，主に，状況の形状を学習する能力の低さであると思われる．ただし，強化学習で最も一般的な入力空間をグリッドで分割する状態表現と比較すると，ASRR による状況表現は抽象度が高く，柔軟性においても優れていると考察された．

第 7 章

今後の展望

7.1 状況認識の学習を行う認知行動システムの実用化に向けて

STNS を実用的なタスクに用いるために，一番の障害となるのは，やはり ASRR による状況形状の学習能力の低さであろう．今のところ，限られたデータから複雑な形状の状況をうまく切り出す手法は存在しない．そのため，限られた状況学習能力で，タスクをうまく処理することを考えた方が，実現性が高いであろう．そのためのアプローチとしては，「階層化」と「属性の生成」の二つが考えられる．それぞれを順に説明する．

階層化

STNS の一様な状況認識法では，複雑な形状の状況の認識を学習するのに限界がある．そこで，状況認識および行動選択を階層化することを考える．すなわち，まず大まかに認識して戦略を決定してから，次にその戦略内でさらに細かく認識して行動を決定する．このように認識を多段階にわけ，さらに戦略内ではその戦略に適した属性を用いて認識することによって，各レベルの状況の形状を簡単にすることができ，乏しい状況学習能力で複雑な形状の状況を表現できる可能性が高まる．

これを実現するためには，まず初めに，要素となる基本的な行動と，それを組み合わせた抽象的な戦略を決める必要がある．例えば，ナビゲーション問題の場合には，基本行動としては「前進，後進，右回転，左回転」など，戦略としては「目標追跡行動，障害物回避行動，壁沿い行動」などが考えられる．そして，戦略のレベルと，戦略を決めた時の行動のレベルで，おのおの状況認識と出力選択の同時学習を行うわけである．ただし，戦略のレベルの学習は，出力の結果が下のレベルに大きく依存していて，特に学習の初期には非常に曖昧に

なるので，各レベルを同時に学習した場合にはうまく学習が進まないことが予想される．この困難を避けるために，学習を二段階に分けて，まず各戦略における行動規則をしっかりと獲得してから，全体のタスクを達成するための戦略選択の学習を行う必要があるかも知れない．

行動戦略の切り替えは，國吉が主張するところの「注意」の機能の一つに相当する [59]．階層化されたSTNSの上位の戦略決定の部分は，一種の注意機構とみなすこともできる．

属性の生成

知覚入力空間上でうまくまとまっていない領域を切り出すためには，その領域がうまくまとまるような属性空間を生成して，知覚入力をその空間上にマッピングする手法が有効である．現在のSTNSでは，6.4.3節で述べたように，性能を上げるためには，得意な属性を設計者が生成する必要がある．得意な属性をシステム内部で自律的に作り出すことができれば，状況の学習能力はかなり向上するはずである．

岡田らは，ロボットコマンド学習システムAcorn-IIにおいて，新属性を生成するために，FC(Feature Construction)法と呼ばれる手法を用いている [95]．この手法は，簡単に説明すると，基本属性と属性を結合するオペレータを用意して，それらを組み合わせて手当り次第に新属性を生成し，望ましい属性を選択するというものである．Acorn-IIでは，FC法を，NGE algorithm（3.2.1節参照）に類似したGTI（Generalization To Interval）法と呼ばれる手法と組み合わせることによって，「大きく回れ」といったような抽象度の高いロボットコマンドを効果的に学習する．

ここで，STNSにFC法を組み合わせることについて考察する．FC法はかなり盲目的な探索法なので，STNSと組み合わせても，複雑な形状の状況を学習するのは困難であると思われる．すなわち，新たな属性となり得る組み合わせは無限大であり，何らかの制限を設けなければ，探索に非常に時間を要するが，逆に制限を設けると，良い属性を生成できない可能性が高まるのである．Acorn-IIでは，「新しい属性を生成するのに使用した属性は，属性集合から削除する」という制限を設けて探索空間を大幅に縮小しているが，これでは，状況の表現能力も大きく制限されてしまう．STNSを複雑なタスクに用いるためには，適切な新属性を発見するのに，もっとヒューリスティックな探索法を用いる必要があるかも知れない．

STNSにおいて，属性が望ましいかどうかを判断する基準としては，「状況の意味に基づく遷移の確率が低い場合は，いずれかの属性が望ましくない」という基準や，「正事例の分布が正規分布に近いほど望ましい」という基準などを用いることができる．ただしこれだけでは，判断に時間がかかり不十分であるので，何か即座に判断できる基準を考える必要があると思われる．

属性の生成と関連して，各状況を異なる属性を用いて表現するという手段も考えられる．属性を設計者が決める場合には，各状況が生成される前に，各状況に異なる属性を用意するのは不可能であるが，属性を自律的に生成する場合には，状況が抽出された後で，属性を切り替えることが可能なのである．現在のSTNSでは，すべての状況を同じ属性を用いて表現しているが，STNSではある入力が各状況に含まれるかどうかを順に調べていくので，各状況が異なる属性を用いて表現されていても問題はない．また，上述の「階層化」と組み合わせて，各レベルのSTNSで異なる属性を用いることもできる．これらの手法を用いた場合は，各状況の表現がさらに柔軟になることが期待できる．

STNSを実用的なタスクに用いるためには，その他にもいろいろな改良すべき点がある．そのいくつかを挙げてみる．

連続的な行動空間

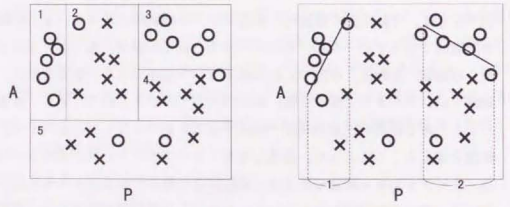
STNSでは，行動空間が離散的であるのが一つの制約となっている．行動空間が離散的であるというのは，STNSで出力される行動が，例えば，「前進，後進，右回転，左回転」の四つに限られるという意味である．実際のロボットの制御では，モーターに様々な電圧をかけることができるので，同じ「前進」にしても，ある範囲の任意の速度で前進することができるし，左右のモータの回転数を違えることによって，前進しながらカーブするといった運動も可能である．ナビゲーション問題の場合は，離散的な行動はそれほど大きな制約にならないが，ボールを投げたり歩いたりするような運動制御の問題では，連続的な出力が欠かせない．

4.3節で説明したactor-critic learningのように，出力の計算にニューラルネットワークを用いる強化学習法では，連続的な行動空間を実現することが可能である[1]．Williamsは，出力を確率的に正規分布させて，分布の平均と標準偏差をニューラルネットワークで強化学習することによって，連続的な行動出力を実

[1]Bartoらによって提案されたもともとのactor-critic learningでは，離散的な各々の行動の評価値（action-value）をニューラルネットワークで計算して，最大の評価値を持つ行動を選択するものであり，連続的な行動空間は扱えない [14]．

現する手法を提案している [127]. Martín らは，2リンクのマニピュレータの制御の学習に，その手法を応用した actor-critic learning を用いている [67]. しかし，複数のアクチュエータの協調動作が必要な場合には，個々の出力を別々に学習するこの方法では，学習が困難であることが予想される．また，この方法を STNS のような状況を分割表現で表す手法と組み合わせる場合には，連続的な行動出力とその結果から状況の分割表現を獲得する手法を，新たに考案しなければならない．

Salganicoff らの IE-ID3 では，知覚入力空間（P）と行動空間（A）を合わせた知覚行動空間（$P \times A$）において，行動の成功例と失敗例を，ID3（3.2.1節参照）を用いて分類し，現在の知覚入力に対して最も成功率の高い行動を選択するという方法で，連続的な行動空間を実現している [105]. これは，図 7.1a に示すような，行動空間を知覚入力空間と同様に分割していく手法の一種である．また，図 7.1b に示すような，知覚入力空間中においてある特定の結果を得るための行動が線形的に表される範囲を状況として切り出すという手法も考えられる．行動は，自ら直接操作することができるという点で，知覚入力とは異なっているので，後者の方がより正確に適切な行動を決定できる可能性がある．



a. 知覚行動空間の分割による表現　　b. 行動の線形近似による表現

図 7.1: 知覚行動空間上での状況表現（1 ～ 5 は状況を示す）

STNS でこれらの状況表現を用いることを考えた場合，行動空間を考慮に入れると問題の次元が大きくなるので，いずれの表現を用いても，学習がますます困難になることが予想される．従って，実用的なタスクにおいて，現在の状況の表現法，学習法をそのまま用いて連続的な行動出力を実現することは，不可

能であると思われる．STNS は，とりあえず戦略選択の学習のような抽象度の高い学習に用いることを考え，運動制御などのより細かい学習への適用は，もっと強力な状況表現法が得られてから考えた方が良いと思われる．

時間の導入

STNS では，5.2.3節で述べたように，固定長行動よりも条件つき行動を用いた方が，状況の抽象度を高くでき，報酬までに通過する状況の数を少なくできるので，良い性能を得られることが期待できる．しかし，条件つき行動を用いる場合には，行動が時間的に抽象化されるので，ワールドモデル上のプランニングでは時間を定量的に考慮できなくなってしまう．すなわち，いかに早くゴールに到達するかという最適化ではなくて，いかに少ない行動（切り替え）でゴールに到達するかという最適化を行うシステムになってしまう．第6章の実験で用いた STNS は，そのようなシステムであった．

条件つき行動を用いながら時間を考慮に入れるためには，一つには，入力空間中でシミュレーションを行うことによって，行動をプランニングする方法が考えられる．しかし，このプランニングは，現在の STNS のワールドモデル上でのプランニングに比べると，かなり抽象度の低いプランニングになるので，計算時間が大きく増大することが予想される．実世界中のエージェントでは，長期的なプランニングには比較的時間をかけることが可能であるが，環境の変化に反射的に反応する必要もある．従って，このように内部でシミュレーションを行うような場合には，即応的なシステムと並列的に組み合わせて用いる必要があると思われる．

学習の高速化

STNS は，非常に少ない正事例から状況を切り出すことができるので，他の知覚入力空間の分節法と比べて，非常に速く学習が進む．しかし，それでも初期のランダムな行動で正事例を集めるのには，かなりの時間がかかる．学習の初期の段階では，ほとんどのデータが状況 0 に遷移するが，状況 0 に遷移するデータは正事例になり得ないので，ほとんどのデータが無駄になっているのである．例えば，第6章のナビゲーション問題においては，状況 0 とゴール状況しか存在しない時には，図 6.2に示す障害物がない環境では 100 回の行動で約 2 回，図 6.22a に示す障害物がある環境では 100 回の行動で約 1.5 回しかゴールに到達しない．残りのデータは，全て状況 0 に遷移するデータで，学習の役には立たない．このような段階が，最初の R 状況が抽出されるまで，すなわち 2 次元の問題では

最小で3回，最大で5回，8次元の問題では最小で9回，最大で17回ゴールに到達するまで続く．

一方，人間の場合は，失敗から学ぶことによって，学習時間を驚異的に短縮することができる．例えば，バスケットボールをゴールにシュートして，少し右に外れた場合には，今度は少し左を狙って投げるということができる．これは，空間の一様性を利用して，ゴールでない場所をゴールと仮定することによって，失敗例から成功例と同じように学習しているとみなすことができる．この学習によって，ほとんど成功しない時にも学習が進む．

STNSにおいても，「空間が一様である」という知識や，実際の経験に基づいて入力空間中でシミュレーションを行う機能を持たせることによって，失敗例から学習することが可能になると思われる．このような学習を行うことによって，効率良く正事例を集めることが可能になり，数回の学習で適切な行動規則が獲得されることも期待できる．ただし，このように仮想的な経験に基づいて学習を行うということは，実世界と乖離する恐れも含んでいる．空間の一様性を仮定して学習していたら，実際には一様でない部分が存在して，かえって再学習に手間がかかるということも起こり得る．このようなことを避けるためには，どういう場合に仮想的な経験から学習を行い，どのように現実の経験と組み合わせるのか，そのためにはどのようなアプリオリな知識を与えるのが適切があるのかなどについて，十分に検討する必要がある．

学習を高速化する手法として，もう一つ有力なのが，教師付きの学習である．強化学習の分野でも，Linによって，教師が与える正しい行動を何度か繰り返すことによって正しい行動政策を獲得する方法（Teaching）が提案されている[61, 63]．強化学習では，何度も繰り返して高い報酬につながる行動は自然に強化されていくので，教師付き学習のための特別な機構は必要ない．しかし，各時点での適切な行動を，ロボットの内部のコマンドで教え続けることは，教師にとって簡単なことではない．

この困難さを無くすためには，教師が実演するのを観察し，それを模倣する機能が有効である．そのようなシステムとしては，國吉らが開発した「実演による作業教示」システムがある[59]．このシステムでは，教師の作業を，抽象度の高い行動記述に変換してから，ロボット固有のコマンドに変換するので，機構的に異なる教師の動作でも模倣することができる．このシステムを用いて，適切な行動を教師から獲得し，強化学習の行動選択の部分で，教師を模倣する行動を繰り返し選択することによって，教師がロボット内部のコマンドを知らなくても高速に学習することができる．

学習の高速化とは少し異なるが，別のタスクのために学習した知識を再利用したり，他者が学習した知識を共有して，システムに学習の初めから知識を持たせてやることができれば，学習時間を大幅に短縮することが期待できる．6.3節の二つの柔軟性テストで，環境の変化が小さい場合に，学習の初期から高性能を得られたのは，その例であると考えられる．この知識の再利用と共有については次節で考察する．

知的な忘却

STNSの履歴データベースでは，有効なデータも冗長なデータも同様に一定期間保持した後に捨てられる．これを，有効なデータは長く保持するように改良することによって，状況の形状などをより良く表現できる可能性がある．具体的に考えると，STNSでは，図5.4に示したように，正事例に比べて負事例が少ししか得られず，しかも状況の境界を決定する上でかなり重要な役割を持っている．そこで，負事例に限って，一度負事例が消去された場所の近傍に再び負事例が得られた場合には，少し寿命を長くする方法などが考えられる．しかし，環境が変化した場合には，事例の寿命が短いほど素早く反応することができるので，どのように寿命を長くしていくのかは，十分に検討する必要がある．

多次元入力

ロボットにおいては，画像入力装置がよく用いられている．画像の持つ情報量は非常に大きく，無駄な情報も多いが画像でしか得られない重要な情報も多く含まれているので，複雑なタスクを実現するためには，画像入力は欠かせない．また，このような膨大な情報にこそ，適切な抽象化の手法が切実に求められているといえる．従って，STNSでも，さらに大きな次元の知覚入力を扱うことを目指さなければならない．

ところが，実際には，STNSは，8次元入力の問題でも扱うのに苦労している．しかも，この8次元入力というのも，あらかじめ設計者が抽象化したものである．すなわち，STNSで用いた障害物センサは，図6.22bに示したように，前方と後方のみが指向性が高いものである．これは，入力の次元を抑えるために，重要なところだけを細かくして，残りを粗くした結果である．できれば，もっと多くのセンサを用いて，このような抽象化も自律的に獲得されることが望ましい．

STNS で大きな次元の知覚入力を扱うためには，次元を圧縮する手法と組み合わせるのが一つの方法である．石黒らは，生の画像データを，解像度を落して，さらに relief algorithm と principal component analysis（主成分分析）を順に用いて次元を落とすことによって，分割のための入力空間を構成する手法を提案している [44, 107]．STNS では，今のところ，石黒らの EOP による分割表現（4.4.2節参照）よりも低い次元の入力空間の問題しか扱えないが，石黒らの次元を圧縮する手法と組み合わせることによって，STNS で画像のような高次元のデータを扱うことができるかどうかは，今後試してみる価値がある．ただし，relief algorithm と principal component analysis を用いると，与えられる事例集合の分布に応じて，入力空間の軸が変化してしまう．石黒らの研究では，固定した事例集合からのオフライン学習で入力空間を分割するので問題はないが，STNS では，タスクを実行しながらのオンライン学習で絶えず事例集合が更新されることになるので，微妙に変化し続ける入力空間の軸をどのように固定して状況表現を得るのかを十分に考察する必要がある．

樋木らの CIQ-learning（4.4.2節参照）では，概念クラスタリング（COBWEB. 3.2.2節参照）によって求めた知覚対象の概念木をもとに，知覚対象を区別する上で重要な属性から順に状態分割に利用する [109]．この手法では，多数の知覚入力の次元のうちの，必要な次元だけに注目することができる．ただし，この手法は離散的な属性しか扱うことができないので，連続的な知覚入力を扱うためには，連続的なクラスタリング手法を用いるように拡張する必要がある．

知覚的見せかけ問題

現在の STNS では，マルコフ的な環境で用いられることを仮定している．マルコフ的な環境とは，ある時点での適切な行動を，その時点での知覚入力のみから決定することができる環境のことである．しかし，知覚入力は環境中の事象のほんの一部しか表現していないので，実問題ではこの仮定が成り立たないことが多い．すなわち，センシング能力の制約から，実際には異なる状態を同じ状態とみなしてしまうことがよく起こる．このような問題は，知覚的見せかけ問題（perceptual aliasing problem，または不完全知覚問題）と呼ばれている．知覚的見せかけ問題が起こった時には，マルコフ性が成り立たなくなり，現時点での知覚入力からは適切な行動を決定できなくなる．

知覚的見せかけ問題が起こっても，過去の知覚入力の履歴を用いると適切な行動を決定できる場合も多い．それは，過去の数タイムステップの知覚入力を合わせると，マルコフ的な問題になる場合である．しかし，知覚入力を合わせると入力空間の次元が大きくなり，状況表現の学習はそれだけ困難になる．そこで，上述の多次元入力に対応する手法と組み合わせて用いることが考えられる．岩津らは，履歴の利用と CIQ-learning に類似した手法を組み合わせることによって，知覚的見せかけ問題に対処するシステムを提案している [48]．

避けるべき状況の認識

5.2.1節で述べたように，STNS では，危険な場面を避けるための警告としての状況の認識を学習することはできない．従って，環境から大きな負の報酬が与えられるようなタスクは，うまく扱うことができない．

このような状況を認識するためには，人間の場合は，一度大きな罰を受けた時に記憶して，その後は，罰を受けなくてもその状況に接近するだけで，避ける行動が強化されていくように思われる．STNS では，繰り返し経験することによって，状況および行動が学習されるので，大きな罰につながる行動が直接的に強化されることはないが，大きな報酬につながるかも知れないし大きな罰につながるかも知れないというギャンブル的な行動が強化される可能性は十分にある．また，状況 0 において大きな罰につながる行動をランダムに何度も選択してしまう可能性も十分にある．このような事態を避けるためには，少ない経験から大きな罰につながる状況を切り出し，その状況を避けながら維持するような仕組みが必要である．

中村らは，衝突を避ける行動の強化学習法として，衝突する行動を学習して，ユーティリティの正負を反転する手法を提案している [82, 83]．この手法と同様にすれば，STNS でも危険な場面を避けるための状況を切り出すことは可能であると思われる．ただし，一度，ユーティリティの正負を反転した後には，危険な場面に向かう行動は選択されなくなるので，タスクを実行しながらオンラインで状況と行動の学習を行うことはできない．STNS でこの状況を維持するためには，やはり何か特別な仕組みが必要になる．

一度大きな罰を受けた時に，それを避けるような行動に対して自律的に正の報酬を設定し，それによって避ける行動を学習する方法も考えられる．しかし，避ける行動は一意に定まらない場合も多く（右にも左にも避けられる場合など），各行動に対応する状況が重なってしまうという問題がある．また，避ける行動に自律的に報酬を設定する際に，もともとの正の報酬を追求する行動の学習を妨げないようにバランスをとるのはそう容易なことではないと思われる．この方法を STNS に組み込むためには，これらの問題を解決しなければならない．

7.2 環境に根付いた記号処理システムの実現に向けて

STNS は，連続的な知覚入力空間から意味のある領域（「状況」）を切り出して，その上で簡単なプランニングを行うシステムである．「状況」の持つ意味は環境から与えられる報酬に依存している．記号処理システムがフレーム問題を現実的に解決するためには，この「状況」のように環境中での経験に基づいて柔軟に定義される記号を用いて知識を表現する必要があると考えられる．しかし，STNS で扱っている「状況」という知識は，まだ非常に初歩的なものである．記号処理システム中で用いられるような複雑な記号を適切に定義するためには，次のような方向に拡張していく必要がある．

知識の構造化

STNS の「状況」は一様である．しかし，記号処理システムで一般的に用いられている記号的知識は，様々な抽象化のレベルを持つ構造化された知識である．様々な抽象化のレベルを持つことによって，異なるタスクおよび各タスクの異なる段階で，それぞれ必要十分なレベルで認識を行うことが可能になる．

強化学習の分野において，状態空間を構成する際に，分割もしくはクラスタリングを階層的に繰り返す手法は，一種の構造化された状態表現を獲得しているとみなすことができる．そのような手法は非常に多い．しかし，ほとんどすべての手法では，状態表現の構造を使用しない．分割法では，一様とみなせない状態を分割し，各々の分割された状態を別々に使用する．それらの状態がもともとは同じであるとみなされていたということは，忘れ去られる．クラスタリング法では，同じ状態とみなせる複数の状態を融合して，新しい状態を生成する．その下位構造であるもともとの状態は忘れ去られる．構造的知識を利用しない理由は，強化学習では，状態は単に区別できれば良く，構造を用いる必要がないからであると思われる．上述の各手法においては，現在のところ抽出された「状態」という記号的知識は強化学習にしか用いられておらず，強化学習においては，タスクの遂行のために必要十分なレベルの一様な状態表現で情報処理を行うので，中間概念や下位概念を用いて思考する必要はない．ただし，あるタスクのために獲得された構造的知識を，他のタスクで再利用する場合には，もともとのタスクとは異なるレベルで情報処理を行う必要が生じるであろう．これについては次の項目「知識の再利用」で考察する．

また，構造的知識として，状態表現以外の知識を利用する強化学習法もある．樋木らの CIQ-learning（4.4.2節参照）では，あらかじめ概念クラスタリング（COB-

WEB．3.2.2節参照）によって知覚対象の概念構造を求めておいて，状態を分割する際に，その概念構造を利用する [109]．この学習法では，学習の初めには，すべての知覚対象を区別しないで一つの概念として扱う．そして，知覚的見せかけ問題が発生した時に，一つ下位の概念を用いて知覚対象を認識し直すことによって，その問題を解決することができる．このようにして，各時点で必要十分なレベルで知覚対象を認識することが可能になる．

7.1節で述べた STNS の階層化も，一種の知識の構造化に相当する．上位の戦略のレベルの知識と，下位の戦略内の行動のレベルの知識は，それぞれ独立して，他のタスクでの再利用および他のエージェントとの共有が可能である．ただし，このような戦略レベルと行動レベルの知識の構造化は固定的であるので，もともとのタスクと異なるレベルで情報処理を行うことはほとんど不可能である．すなわち，あるタスクの戦略レベルの学習による状況の切り方，行動規則などの知識を，他のタスクの行動レベルで利用することはまず考えられない．

知識の再利用

知識の再利用とは，あるタスクのために獲得した知識の一部を，別のタスクで利用することである．実ロボットで用いる知識を，あらかじめシミュレーションで学習しておく場合も，一種の知識の再利用であるとみなすことができる．知識を再利用することによって，あるタスクについて学習する前から，関連する知識を組み合わせてある程度の予想を立てることができる．記号的知識では，知識が意味のある塊に抽象化されているので，知識の再利用をしやすいという利点がある．強化学習の分野では，今までに，あるタスクのために獲得した知識の一部を，別のタスクで利用するという研究は（シミュレーションから実ロボットへの適用以外は）ほとんど行われていないが，前節でも述べたように，学習時間の短縮のためには非常に有効であると思われる．

まず，状況認識のための知識の再利用を考える．これは，状況の形状や意味，状況表現の階層構造の再利用に相当する．このためには，「状況形状の柔軟性」と「状況表現の構造化」の二つが重要なポイントになると考えられる．

「状況形状の柔軟性」は，STNS の状況表現法である ASRR のように，状況の形状を経験に基づいて常に調整し続けることによって得られる．ある意味を持つ状況を他のタスクで再利用する場合，状況の形状が正確に一致することはあまり期待できないので，形状を調整する機能が必要である．6.3節の二つの柔軟性テストでは，この ASRR における状況形状の柔軟性によって，少し異なる環

境で学習した知識を再利用することで学習の初期から高性能が得られることを示した．強化学習のための状態の分割表現法として，状態形状の調整を行う手法は，今のところASRR以外にはほとんど見当たらないが，環境やタスクが変化した場合の適応性を考えると，今後重要になってくるであろう．ただし，状況の形状を調整するには，当然，そのためのコストが伴う．従って，効率の良い学習のためには，ASRRのような一様な調整法ではなくて，変化の激しいところを中心に調整するような仕組みが必要となるであろう．

「状況表現の構造化」は，知識の再利用の機会を大きく拡大する．例えば，「手紙を投函する」というタスクにおいては，ポストは目的地であり，電柱は障害物であるが，「買いものにいく」というタスクにおいては，ポストも電柱も障害物である．一つめのタスクにおいて，「ポストがある状況」と「電柱がある状況」を全く別々に認識していると，二つめのタスクにおいて「障害物がある状況」を学習し直す必要があるが，一つめのタスクにおいて，ポストと電柱を合わせて「何か物がある状況」として認識していれば，二つめのタスクでは，その知識を再利用することができる．前項目で述べた分割もしくはクラスタリングを階層的に繰り返す手法によって構成される状況表現は，このような目的に用いることができる可能性がある．

次に行動選択のための知識，すなわち行動規則の再利用について考える．異なるタスクにおいて，類似した意味を持つ状況が存在している場合は，行動規則を再利用できる可能性がある．行動規則の再利用でも，「状況表現の構造化」が重要なポイントになる．知識が構造化されていれば，低いレベルでの細かい知識は利用できなくても，大まかな知識は利用できる可能性がある．例えば，「オフィスで空き缶を集める」というタスクと「火星で石を集める」というタスクは，知覚入力に対する行動の割り当ては全く異なっているかも知れないが，目標物を探し，発見して獲得し，収集場所まで運ぶという戦略の切り替えの知識は，共通して用いることができる可能性がある．また逆に，細かい知識が一つのスキルとしてまとまっている場合には，その部分だけ一括して抜き出して利用できる可能性がある．例えば，あるタスクのために学習した「壁沿い行動」というスキルを，他のタスクにおいてそのまま利用することが期待できる．行動規則の再利用法としては，与えられた行動規則から最適であると予想される行動を，学習時に選択されやすくしたり，政策の初期値として用いたりして，学習を促進する方法が考えられる．

知識の共有

知識の共有とは，知識の再利用の一種で，あるエージェントが獲得した知識を，異なるエージェントが再利用することである．従って，前項目の知識の再利用で説明したことがそのまま当てはまる．特に，機構の異なるエージェントが同じタスクを実行する場合には，抽象度の低いレベルでは，それぞれの機構に応じたローカルな学習が必要であるが，戦略選択などの抽象度の高いレベルでは，異なるエージェント間で知識を共有できる可能性が高いと思われる．

知識の共有に人間が介在しない場合には，エージェント間のコミュニケーションが必要となる．この場合にも，情報の塊としての記号が重要な役割を持つ．現在のSTNSの「状況」はシステム内部の記号に過ぎないが，この記号が他者と情報交換するために用いられると，エージェント間の言語としての意味を持つことになる．そして，言語による情報交換によって，実際の経験のみに基づく学習よりも，はるかに効率良く学習することが可能になると思われる．この言語による情報交換は人間の学習でも大きな役割を果たしていると考えられる．

このように，複数のエージェントで共有される言語を生成するためには，共通体験が必要である．中野らは，言語発生のプリミティブモデルとして，二つのロボットが，共通体験と連想によって，次第に共通の言語を獲得していくモデルを示している [85]．また，小野は，雄雌という異種のエージェントの集団間で，個々のエージェントが接触を繰り返すことによって共通の言語が獲得されていくモデルを示している [97]．状況を共有する雄雌の間のコミュニケーションにおいて，集団内の他のエージェントと同じ言語を用いている場合に高い確率で報酬が得られるという当然の原理にしたがって，言語が次第に収束していく．國吉は，実世界中での知能のためには，共同注意と模倣が重要であることを指摘している [59]．これは，共通体験の内容をさらに細かく指摘したものであるとみなせる．STNSで言語を獲得する際にも，エージェント間での情報交換や共通体験，模倣が鍵になってくると思われる．

また，中川らは，他者から伝達される音声情報と，その時点での視覚情報を統合して概念を獲得するシステムを提案している [80]．このシステムでは，自然言語と記号を結び付ける学習を行うが，言語発生システムの中に組み込んで，言語理解の部分に用いることが可能であると思われる．

物の認識

STNSでは状況に関する記号しか抽出できない．しかし，現実に用いられている記号の多くは物に対して割り当てられている．昔から画像認識や記号推論の

分野では，当然のように，物に対して記号を割り当てることが有効であると仮定されてきた．これは，物を単位として環境の認識を行うことが，人間にとって自然であるためだと思われる．人間および猿や犬などの自然淘汰に生き残ってきた動物の一部が，物の認識を利用している（と思われる）ことは，物の認識の有効性を表していると思われる．

状況の認識は，環境を時間的に切ることに相当する．従って，状況は知覚入力の関数とすることができる．一方，物の認識は，環境を空間的に切ることに相当する．従って，物は知覚入力の関数とすることができず，ある知覚入力に対して，無限の物を切り出すことが可能である．各知覚入力に対して，現在のタスクにとって必要な物だけを認識するように制限しないと，簡単にフレーム問題に陥ってしまう．

この従来の記号処理システムと同じ失敗を繰り返さないためには，状況の認識と同様に，環境から得られる報酬を基に物の認識を行うような仕組みが有効であると考えられる．ただし，状況の認識とは異なって，報酬が得られたからといって，知覚入力全体を強化するわけにはいかない．まず，報酬が知覚入力中のどの部分に起因するものなのかを明らかにする必要がある．また，知覚入力中のどの部分が物として認識することができて，その物がどのように動いた時に報酬が得られるのかという形で知識を表現しなければならない．物の認識のためには，「空間中で塊となっている領域は物である」とか「動かした時に一緒に動く領域は物である」などのアプリオリな知識を与える必要があるであろう．もちろん，動きの概念もあらかじめ与えておく必要がある．また，物と報酬を結び付けるためには，「報酬の得られた位置の近くの物が報酬と関係している」とか「報酬が得られた時に動いていた物が報酬と関係している」などのアプリオリな知識を与える必要があるであろう．このような知識を用いて，報酬に関係する物の認識を次第に確実にし，報酬に関係が薄い物の認識は次第に曖昧にしていくことによって，フレーム問題を避けながら，物の概念を用いた知識表現が獲得できると思われる．

物の概念を用いて知識を表現する場合には，状況の概念を用いた場合とは異なる問題が，まだたくさん存在する．一つには，物は存在しているのだが，センサの観測域の影響で，観測できない場合がある．例えば，他の物の陰に入っている場合などである．しかし，この場合でもタスクに対するその物の影響は無視できない場合があるので，知識表現の中にはその物を記しておく必要があるかも知れない．また，そのように隠れている物を意識的に観測することもでき

る．能動視覚の問題である．そのような場合には，観測コストを考慮する必要も出てくるであろう．さらに，物と一口にいっても，その性質には様々なものがある．色，形（3次元的，見え方），大きさ，位置，手触り，味など，物の性質の認識にも切りがない．これも，タスクに関係する性質だけを認識するようにしなければ，簡単にフレーム問題に陥るであろう．

物の概念を用いて表現された知識を，どのような情報処理に用いるのかも検討しなければならない問題である．物の概念を用いた知識は，従来の人工知能で用いられてきた述語論理による推論やエキスパートシステムなどの強力な情報処理手法によって扱うことが可能である．これは大きな魅力の一つであるが，そのような情報処理手法と環境からの報酬に基づく強化学習をどのように組み合わせていくのかは，大きな問題である．

物の認識と，STNS で行うような状況の認識をどのように組み合わせるのかも問題である．人間も，どちらか一方だけではなく，両方を使い分けていると思われる．一つには，大まかに状況を認識して戦略を決めて，細かく物を認識して行動を決定するという使い方が有効であると思われる．また，状況の認識は，物の認識の上に成り立っている部分もあると思われる．これらのことを含めて，色々と検討してみる必要がある．

環境中での経験に基づいて物の認識を学習する研究は，まだほとんど行われていない研究分野である．その中で，椹木らの CIQ-Learning の研究は，強化学習の枠組の中で，物の認識を扱う先駆的な研究である [109]．この研究は，物の概念構造から強化学習の状態表現を獲得したり，物を認識する時に生じる能動知覚（active sensing）の問題を扱っている点で，従来の強化学習とは大きく異なっている．ただし，物の認識に関しては，あらかじめ与えられた物と属性を用いていて，環境中での経験に基づいて物の認識を学習しているわけではない．物の認識の学習に関しては，「本当に物の認識が有効であるのか」，また「物の認識を行うためにはどのようなアプリオリな知識が必要となるか」という辺りからじっくりと考えていく必要がある．

知覚の時間的な抽象化

STNS では，知覚入力は各時点ごとに得られ，その知覚入力のみに基づいて状況の認識を行う．一方，知覚の時間的な抽象化とは，個々の時点の知覚入力を別々に扱うのではなく，時間的に連続して得られる知覚入力系列を意味のあるまとまりとして認識を行うことである．これは，前節で説明した「知覚的見せ

かけ問題が起こった時に過去の履歴を用いて状況の認識を行う手法」の延長であると考えられる．ただし，後者では，状況が区別できれば良いのに対して，前者では，知覚入力系列が意味のあるまとまりをなしていることが重要である．このような抽象化によって，他者の運動，および自己の運動を認識することが可能になる．

中村らは，大量の知覚入力系列をクラスタリングすることによって，局所的な環境の構造の認識を学習する手法を提案している [84]．この手法では，簡単な衝突回避行動から，固定長の知覚入力系列を大量に蓄積し，それらの系列をクラスタリングすることによって，「角を曲がっていく状況」とか「廊下の壁に斜めにぶつかっていく状況」などの局所的な環境構造の認識を獲得する．各知覚入力系列は固定長で切り出しているので，知覚の抽象化としては非常に初歩的であるが，衝突回避行動というルールに従って行動しているので，人間にも意味の理解できるクラスタを構成することができる．また，知覚入力系列が類似しているということは，自己の運動が類似していることを意味しており，自己の運動の認識を学習しているとみなすこともできる．ただし，学習はオフラインで行われていて，柔軟に修正したりすることはできない．

國吉らは，人間の作業を観察し，作業の記号記述を生成するシステムを提案している [60]．このシステムでは，行為を，外部に変化をもたらす因果的な過程であると定義して，行為対象の切換りを検出することによって，作業の時間的分節化を行っている．これは，知覚入力の時間的な抽象化の一種であるとみなせる．このシステムは学習システムではないが，知覚入力系列の時間的な分節法としては，中村らの手法よりも汎用性があり，認識の学習にも用いることが可能であると思われる．

行動の時間的な抽象化

行動の時間的な抽象化とは，時間的に連続な行動系列を意味のあるまとまりとして扱うことである．STNSで用いている「条件つき行動」（5.2.3節参照）も行動の時間的抽象化の一例である．しかし，この条件つき行動は，同一の行動を繰り返し実行するだけなので，抽象化としては初歩的なものである．同様の抽象化としては，野田らの強化学習システムで用いられている「可変長行動系列」がある [92]．

STNS を 7.1節で述べたように階層化した場合の「戦略」も行動の時間的抽象化の一種であるとみなすことができる．戦略の場合は，異なる種類の行動を組み

合わせて意味のある行動系列を作り出すので，条件つき行動に比べると抽象度が高い．しかし，7.1節で述べたSTNSの階層化では，戦略の意味はあらかじめ設計者によって決められるので，行動系列の分節の仕方が固定的であるといえる．

行動系列を適切な長さで抽象化してやると，効率良く情報処理を行うことができる．しかし，与えられたタスクに対して，どの程度の長さで行動系列を抽象化するのが最も良いのかは，設計段階では簡単にはわからない．そこで，幸島らは，一まとめにして扱う行動系列の長さを，タスクを実行する経験に基づいて調整し，適切な長さを学習する手法「RCA（Reinforcement learning with Chunking of Action）法」を提案している [106]．これは，行動系列の分節の仕方の学習に相当する．RCA 法は，一まとめにして扱う行動系列，すなわち抽象的な行動の長さを調整することによって，強化学習の性能を向上させる．RCA 法では，抽象的な行動は，条件つき行動と同様に，同一の行動の繰り返しなので，抽象化としては初歩的である．また，行動の長さの学習法は，同一行動の繰り返し回数をあらかじめ複数用意して，各々で強化学習を行うごとに各々の評価値を更新していくだけなので，かなり単純な学習である．しかし，行動表現の適切な粒度を学習する研究は他に見当たらず，斬新な研究であるといえる．

行動を制約に基づいて定義するような抽象化もある．この行動では，システムは，知覚入力値が常にある制約を満たすように動き続ける．行動をこのように定義すると，環境の様々な変化に対するシステムの様々な反応をわずかな制約式で記述することができるので，抽象度はかなり高い．橋田らは，この制約が，環境とシステムとを結ぶフィードバックループの束に相当し，フレーム問題の現実的な解決のための鍵になると主張している [33, 34]．岡田らは，この制約を用いて表現されるロボットコマンドの学習システム Arcon-II を提案している [95]．Arcon-II によって学習されるロボットコマンドは属性値の制約式である．この制約式を学習するために，属性空間上の正事例と負事例から，NGE algorithm（3.2.1節参照）と類似した手法で正事例が含まれる領域だけを切り出す GTI（Generalization To Interval）法と，7.1節で説明した新たな属性を生成する FC(Feature Construction) 法を組み合わせて用いている．STNS で用いている「条件つき行動」も，「ある条件が満たされている間は，同じ行動を続ける」という一種の制約に基づいているとみなすこともできる．

行動の外部的定義

STNSにおいては，行動はロボット内部のモーターコマンドに，直に対応していた．例えば，第6章のナビゲーション問題における「前進」とは，通常において前進するための電圧をモーターにかけることであって，例え壁にぶつかって進むことができなくても，例え急な坂道で後ろにずり落ちていたとしても，その電圧さえかけていれば「前進」であった．しかし，異なる環境，異なるエージェントで知識を再利用しようとする場合には，この定義では問題がある．内部的にはどのようなコマンドを用いていても，外部的には外から見て前進している行動を「前進」と名付けるべきである．

このような定義を実現するためには，知覚入力の変化によって行動を定義する手法が有効である．例えば，視覚センサの画像上のオプティカルフローが一様に左方向に流れるような行動を，「右回転」と名付けるわけである．中村らは，このようなオプティカルフローの情報に基づいて，ロボット内部のコマンド群をクラスタリングし，強化学習の学習時間の短縮に利用する研究を行っている[82, 83]．

このような抽象度の高い行動の概念は，知識の再利用だけではなくて，他のエージェントの行為理解や，模倣のためにも重要である．しかし，ロボット内部の固有の行動表現にも，この抽象度の高い概念を用いるべきかどうかは検討の余地がある．「目標追跡行動，障害物回避行動，壁沿い行動」などの戦略的な行動は，ロボットの内部でも，当然，抽象度の高い概念で表現されるが，その要素となる「前進」とか「後進」のような基本的な行動は，外部的に定義することも，モーターコマンドに直に対応させることも可能である．抽象度の高い概念を用いる場合は，当然，その行動を実現するための仕組みや，概念通りの実行に失敗した場合の処理などが必要であり，情報処理が複雑になる．また，基本的行動がモーターコマンドに直に対応していても，それを組み合わせた戦略のレベルでは，外部的な概念として定義することが可能である．これらを考え合わせると，ロボットの内部では，基本的な行動はモーターコマンドに直に対応する簡単な行動で十分であると思われる．

2.2節で述べたフレーム問題を現実的に解決するための三つの機能のうち，STNSで扱ったのは，3の「膨大な情報の中から注目すべき情報を取り出す方法を学習する」機能だけである．十分に環境に根付いたシステムとするためには，STNSに，1の「環境を頻繁に参照する」機能と，2の「情報処理を並列化する」機能を組み込む必要がある．1の機能は，多くの場合，2の情報処理の並列化と共に実現され，システムの「環境への埋め込み（embedding）」と呼ばれている．そこで，STNSにおいて情報

処理を並列化することを考える．

情報処理の並列化

情報処理の並列化は，一つには，7.1節で述べたSTNSの階層化によって実現することができる．この並列化では，システム全体の頑健性は高まらないが，処理を分割することによって，即応性を高めることができる．出力の計算や学習のための計算は，上位の戦略レベルのSTNSではゆっくりと行うことができるが，下位の行動レベルのSTNSでは素早く行う必要がある．階層化することによって，各STNS内の状況の数はある程度少なく抑えられるので，計算時間が短くなることが期待できるが，それでも十分な即応性が得られない場合は，下位の情報処理にはSTNSを用いずに，behavior-basedロボットで用いられているようなhand-codingされたサブシステムを用いる必要があるかも知れない．また，学習なしでも十分に機能するサブシステムが存在する場合には，余分なコストをかけて学習する必要はないといえる．

独立したアクチュエータが複数存在する場合には，それぞれに一つずつSTNSを割り当てて，対等な並列化を行うこともできる．このように並列化すると，いくつかのサブシステムが故障しても，残りのサブシステムで，タスクを可能な限り効率良く達成する行動を自然に再学習することができる．このフェールソフトの能力によって，システム全体の頑健性をかなり高めることができる．ただし，アクチュエータが複数あっても，それらが協調して動く必要があることが多い場合には，あらかじめ協調行動を定義しておいて，単独のSTNSで処理した方が，学習がはるかに容易になると思われる．並列されたSTNSで協調行動を学習する際には，他のアクチュエータの出力を，環境を通して間接的に知ることもできるが，リカレントニューラルネットワークのように，直接的に知覚入力の中に加えてやる方が，学習が容易になると思われる．処理速度に関しては，互いの動作が依存しない場合には，単独のSTNSで処理するよりも，分けて処理した方が，個々のSTNSの状況の数が抑えられて，素早く処理できると思われる．逆に互いの動作が依存する場合には，分けて処理する場合には上述のように知覚入力の次元が高くなるので，単独のSTNSの方が素早く処理できると思われる．

第 8 章

結論

本論文では，フレーム問題を避けるための認識の学習について論じた．フレーム問題とは，限定された情報処理能力しかないシステムがその能力をはるかに上回る複雑性を持つ情報を扱おうとすることによって起こる問題であり，情報処理の時間の爆発，記述の量の爆発，誤った情報処理によるタスクの失敗などの形で現れる．このフレーム問題は，環境中での経験に基づいた情報の適切な抽象化によって，回避することができると考えられる．そこで，本研究の目的は，情報処理に用いる記号を環境中での経験に基づいて定義することにより，タスクを表すのに適切な記号体系を獲得することとした．

第2章ではまず，実世界中の自律エージェントにおいて，このフレーム問題が大きな障害となっていることを説明した．その上でフレーム問題の現実的な解決に近付くためには，次の三つの機能

1. 環境を頻繁に参照する．
2. 情報処理を並列化する．
3. 膨大な情報の中から注目すべき情報を取り出す方法を学習する．

が重要であることを示した．1, 2の機能によって，システムは不慣れな環境でも，大きく破綻することなく行動することができる．そして3の機能によって，不慣れな環境に徐々に慣れていくことが可能になる．この内の 1, 2 は，物理基盤仮説に基づくロボットとして，近年，盛んに研究されている．しかし，3の抽象化の学習（認識の学習）は，認知行動システムにおける学習としては，従来はあまり研究されて来なかった．それは，抽象化の学習は，抽象化された表象の上での情報処理よりも，かなり大きなタイムスケールで行われるものなので，近似的に固定することができるからである．固定した方が効率的であり，うまく記号体系を設計すれば，十分に実用的なシス

テムになる．しかし，実世界中の認知行動システムの場合には，環境の複雑性のために，この設計が非常に困難になることが指摘されている．また，真に自律性が求められているのは，惑星探査ローバーなどの人間の助力が困難なタスクにおいてである．そのようなタスクにおいては，通信が困難であるために自律性が必要とされ，さらに環境の事前の調査が難しいこと，故障した場合も修理できないことなどから，情報処理の大きな柔軟性が求められている．抽象化の学習という情報処理の低いレベルにおける学習をゆっくりと行いながら，同時に抽出された表象の上での学習を行うことによって，大きな柔軟性を持つことが期待できる．本研究は，3 の認識の学習のこのような可能性を探るための基礎研究であると位置付けられる．

第 3 章では，従来の認識の学習を概観した．従来の認識の学習では，分類するカテゴリの意味は，教師から与えられるか，またはデータの分布のみに基づいていた．しかし，認知行動システムによる認識の学習では，行動の結果に基づいてカテゴリの意味を定義することができる．これは，環境中での経験に基づく抽象化の学習とみなすことができる．そのような学習は，機械学習の一種である強化学習の分野で，入力の一般化の学習として研究されている．この入力の一般化の学習の中でも，従来の認識の学習法のいくつかが使用されている．そこでこの章では，まず従来の認識の学習法を教師付き学習と教師なし学習に分けて説明した．そして次に，本研究における認識の学習として，認識の学習と認識された表象の上での学習を認知行動システムの中で同時に並列して行うことの重要性を指摘した．

第 4 章では，まず強化学習について簡単に説明した．強化学習では，状態表現の抽象化のレベルを適切に設定することが難しく，抽象化のレベルが低過ぎると，学習の収束が非常に悪くなってしまう．逆に，抽象化のレベルが高過ぎると，多くの知覚入力状態において適切な行動を選択できなくなってしまう．これは，フレーム問題の一種で，入力の一般化問題と呼ばれている．この入力の一般化問題を解決するためには，環境からの報酬に基づいて入力の一般化の学習を行うのが良いと考察された．強化学習は，そもそも環境からの報酬に基づいて行動規則の学習を行うものなので，このような抽象化の学習とは非常に相性が良いといえる．

次に，従来の様々な入力の一般化の学習法を紹介した．報酬に基づく入力の一般化は，大きく，分割表現を用いる手法と，陰的表現を用いる手法に分けられる．分割表現を用いた場合は，内部の情報処理が理解しやすく，また，切り出された状態表現を記号として扱うことができる．一方，陰的表現を用いた場合は，学習の柔軟性に優れると考察された．分割表現は，さらに，ユーティリティに基づく分割表現と，行動に基づく分割表現と，ユーティリティおよび行動に基づく分割表現の三つに分けられる．ユーティリティに基づく分割表現では，各状態の評価値を近似することができるが，

状態の形状が複雑になる．行動に基づく分割表現では，各状態の評価値は近似できないが，状態の形状が比較的まとまる．ユーティリティおよび行動に基づく分割表現では，各状態の評価値を近似することができ，状態の形状が比較的まとまるが，上の二つに比べると抽象度が低くなると考察された．

最後に，本研究におけるフレーム問題の現実的な解決に近付くための強化学習システムとしては，連続入力から，オンラインで，報酬の類似度に基づいて，状態の分割表現を学習しながら，同時に強化学習を行うシステムが望ましいことを指摘した．

第 5 章では，環境中での経験に基づいて状況認識と行動規則を同時に学習する新しい機械学習法として「状況遷移ネットワークシステム（Situation Transition Network System，STNS）」を提案した．STNS では，状況認識の学習のために「Adaptive Situation Recognition based on Rewards（ASRR）」を，行動規則の学習のために「インターリーブプランニングに基づく強化学習（Interleave Planning-based Reinforcement Learning，IPRL）」を用いる．

ASRR は，知覚入力空間中の特定の領域，つまり「状況」を切り出して，その「状況」に対して記号を割り当てる．すなわち「状況」に相当する記号の意味を学習する．抽象化の手段として記号化を選んだのは，記号表現には強力な情報処理手法が存在するからである．また，将来的に他者と知識を共有することを考えても，記号表現を用いるのは望ましい．ASRR において，各状況の意味は行動の結果の類似性によって定義される．類似性の基準は環境から与えられる報酬とする．すなわち，一つには，ある特定の行動によって大きな報酬が得られる領域を状況として切り出す．また，もう一つには，ある特定の行動によってそのような報酬の見込みの高い既存の状況に遷移する領域を状況として切り出す．切り出された状況は，その後の経験に基づいて動的に維持される．このように行動の結果の類似性に注目することによって，エージェント自身の環境中での経験のみに基づいた記号の定義が可能になる．このような手法によって，行動に基づく分割表現による抽象度が高い状況を切り出すことができる．

IPRL では，ASRR によって抽象化された状況表現の上で行動規則の強化学習を行う．具体的には，状況間の遷移確率と各遷移で得られる報酬の期待値を最尤推定してワールドモデルを構成し，その上で前向きの部分的プランニングを行うことによって行動を決定する．部分的プランニングとしては，プランの成功確率が高い範囲だけを探索する「インターリーブプランニング」を用いる．そのため，不慣れな環境でも，考え込まずにとりあえず即応的に反応することができる．ワールドモデルは過去の経験に基づいて最尤推定されるので，システムは環境中で行動しながら，各状況においてどの行動をとるのが良いかを学習することになる．この強化学習法では，状況表現が変化しても，行動規則を素早く収束させることができる．

STNSでは，ASRRとIPRLを組み合わせて，状況認識と行動規則を，タスクを実行しながら同時に学習する．そのため，状況の形状が適切であれば行動はタスクを成功させるし，行動がタスクを成功させ続ければ状況の形状が適切に維持される．この状況と行動の相互依存関係によって，タスクに特化しながら環境やタスクの多少の変化に柔軟に対応できる認知行動システムが実現できる．また，このオンライン学習によって，入力空間中の，タスクを実行する上で頻繁に経験する領域について，特に詳しく学習することができるので，学習が効率的に進む．

STNSは，分割表現を用いて入力の一般化の学習を行う強化学習の一種であるとみなすことができる．しかし，状況を切り出した後もその形状を調整し続ける点で他の分割表現の手法とは異なり，より柔軟な学習を可能にすることが期待できる．この柔軟性は，不慣れな環境に徐々に慣れてフレーム問題に悩まされないようになるために非常に重要である．

第6章では，STNSの性能を調べるために，2次元入力のナビゲーション問題で四つの実験，8次元入力のナビゲーション問題で一つの実験を行った．いずれもコンピュータシミュレーションである．2次元入力の簡単なナビゲーション問題では，STNSが，簡単な問題において，理想的な状況認識と行動規則を獲得できることが確かめられた．また，データが不十分な段階でも状況を切り出すことによって，学習が非常に速く進むことが確かめられた．アクチュエータが故障したローバーを用いた2次元入力のナビゲーション問題では，STNSが，超楕円体から離れた形状を持つ状況の認識，小さい状況の認識，他の状況に覆われた状況の認識などの点で弱点を持つことが確かめられた．そして，状況認識が不十分になるにつれて，行動規則の学習も困難になることが確かめられた．しかし，問題を徐々に複雑にしていった時の性能の悪化はゆるやかであることが確かめられた．2次元入力のナビゲーションの二つの柔軟性テストでは，STNSが，簡単な問題において，環境の小さな変化に柔軟に適応でき，大きな変化にもほぼ安定して対応できることが確かめられた．このような柔軟性は，ASRRの動的な状況表現によるところが大きく，環境の変化が小さい時には，過去の知識を利用して素早く滑らかに適応し，環境の変化が大きい時には，自律的に過去の大きく誤った知識の影響を断ち切って，新たに白紙状態から学習することが確かめられた．8次元入力の複雑なナビゲーション問題では，報酬に近い領域では，ある程度の状況認識と行動規則を獲得することができた．しかし，報酬から離れた領域では，状況認識が不十分であるために，適切な行動規則を獲得することができなかった．その原因は，主に，状況の形状を学習する能力の低さであると考察された．ただし，強化学習で最も一般的な入力空間をグリッドで分割する状態表現と比較すると，ASRRによる状況表現は抽象度が高く，柔軟性においても優れていると考察された．

第7章では，まず，状況認識の学習を行う認知行動システムの将来の実用化に向けた考察を行った．STNSを実用的なタスクに用いるために，一番の障害となるのは，状況形状の学習能力の低さである．今のところ，限られたデータから複雑な形状の状況をうまく切り出す手法は存在しない．そのため，限られた状況学習能力で，タスクをうまく処理することを考えた方が，実現性が高い．そのためのアプローチとしては，「階層化」と「属性の生成」の二つが考えられる．「階層化」では，まず大まかに認識して戦略を決定してから，次にその戦略内で細やかに認識して行動を決定する．このように認識を多段階にわけ，さらに戦略内ではその戦略に適した属性を用いて認識することによって，各レベルの状況の形状を簡単にすることができる．「属性の生成」を用いるアプローチでは，知覚入力空間上ではうまくまとまっていない領域を切り出すために，その領域がうまくまとまるような属性空間を生成して，知覚入力をその空間上にマッピングする．どちらも個々の状況の形状を簡単にすることによって，乏しい状況学習能力で複雑な状況表現を獲得する可能性を高めることができる．

STNSを実用的なタスクに用いるためは，その他にも，

1. 連続的な行動空間
2. 時間の導入
3. 学習の高速化
4. 知的な忘却
5. 多次元入力
6. 知覚的見せかけ問題
7. 避けるべき状況の認識

などの点で改良を行う必要があることを考察した．

次に，環境に根付いた記号処理システムの実現に向けた考察を行った．環境と密接に関わりを持つ記号とは，STNSにおける状況のように環境中での経験に基づいて柔軟に定義されるものだと考えられる．しかし，この「状況」という記号は，まだ非常に初歩的なものである．このように経験に基づいて抽出される記号が，記号処理システム中で用いられるような複雑な記号となるためには，次のような方向に拡張していく必要がある．

1. 知識の構造化

2. 知識の再利用

3. 知識の共有

4. 物の認識

5. 知覚の時間的な抽象化

6. 行動の時間的な抽象化

7. 行動の外部的定義

また，STNSでは，上述のフレーム問題を現実的に解決するための三つの機能のうち，3の「適切な認識を学習する」機能しか扱っていない．フレーム問題に悩まされない，環境に根付いたシステムとするためには，この機能に合わせて，1の「環境を頻繁に参照する」機能と，2の「情報処理を並列化する」機能も同時に持つように拡張していく必要がある．

本研究で開発したSTNSでは，認識の学習と認識された表象の上での強化学習を並列して実行することによって，従来の強化学習システムよりも柔軟な状態表現を実現している．その結果，与えられたタスクに特化しつつ，不慣れな環境に柔軟に適応できる認知行動システムを構成することができた．このシステムは，記号の学習としては知覚入力空間からの「状況」の抽出，記号を用いた情報処理としては強化学習しか行わないし，アプリオリな知識はほとんど何も与えることができないので，認知行動システムとしては非常に初歩的なものである．しかし，認識の学習と認識された表象の上での学習を同時に行う認知行動システムは，まだほとんど扱われていない研究領域であり，本研究で開発したSTNSは，記号に対応する知覚入力空間中の領域を，筆者の知る中で最も柔軟に調整することができる認知行動システムである．環境中の小さな変化に対して，記号に対応する領域の形状を調整するシステムも，環境中の大きな変化に対して，既存の記号を消去して新たな意味の記号を定義し直すシステムも，STNS以外には存在しない．もっと強力な記号体系，もっと複雑な情報処理，そしてアプリオリな知識の融合を目指してこの種の研究を発展させていくことは，実世界中の知能ロボットを実現する上で非常に有用であると思われる．本研究がこの発展に対し，何らかの貢献となることを期待したい．

謝辞

本論文を締めくくるに当たって，これまでお世話になった方々に感謝の気持ちを述べさせて頂きます．

指導教官である堀浩一助教授に，心から感謝致します．この研究を始めるに当たって，筆者の素朴なアイディアを，的確なアドバイスで研究に導いて頂きました．また，研究に行き詰まってしまった時には，いつも堀先生とのディスカッションで方向性を見い出すことができました．博士課程の三年間で研究をまとめることができず研究室を離れた後も，ずっと面倒を見て頂きました．本当にありがとうございました．

知能工学研究室の中須賀真一助教授には，専門的なアドバイスをたくさん頂きました．筆者が博士課程三年目に入った頃に，研究室が融合して，機械学習の専門家である中須賀先生と同じ研究室になれたのは，本当に幸運でした．周りに同じ分野の研究者が少なく，ともすれば独り善がりに陥りがちな研究を，機械学習の研究としてまとめることができたのは，中須賀先生のお導きのおかげです．心から感謝致しております．

本論文の審査委員である東京大学の河内啓二教授，鈴木真二教授，桐山孝司助教授には，本研究に対する有用かつ的確なご指摘，ご指導を賜わりました．また，鈴木先生には，学部時代に指導教官として卒業論文の指導をして頂き，筆者が研究者の道に進むきっかけを与えて頂きました．ここに深く感謝致します．

博士課程の一年目まで研究室の教授でいらっしゃった早稲田大学理工学部大須賀節雄教授には，研究会などで，厳しくて非常に有用なアドバイスをたくさん頂きました．大須賀教授が早稲田大学に移られてからも筆者を気にかけてご指導下さいました．深く感謝致します．

知能工学研究室の山内平行助手には，研究環境の整備などの面などでお世話になりました．また，研究会の折や日常の会話の中でも，貴重な助言をたくさん頂きました．ここに深く感謝致します．

知能工学研究室の皆さんには，特に研究環境の整備に関する面で大変お世話になりました．また，研究に関する助言を頂いたり，相談にのって頂きました．研究以外の

面でも，皆さんのおかげで楽しい生活を送ることができました．どうもありがとうございました．

知能工学研究室の秘書である二木晶子さんには，５年間に渡り，研究活動を様々な面から支えて頂きました．研究室を離れた後も気にかけて下さり，訪ねる度に相談にのって頂きました．また，社会人一年生の筆者に，社会生活に関する色々なアドバイスもして頂きました．深く感謝致します．

奈良先端科学技術大学院大学情報科学研究科の西田豊明教授，武田英明助教授には，筆者が奈良先端大学に赴任してからの一年半もの間，この研究を見守って励まして頂きました．心より感謝致します．

京都大学工学部の沢田篤史助手には，昨年一年間，研究環境の整備などの面で大変お世話になりました．深く感謝致します．

参考文献

[1] 阿江忠．もう一つのニューラルネット学習法 — 自己組織化．*bit*, Vol. 24, No. 9–11, 1992.

[2] Albus, J. S. ロボティクス: ニューロンから知能ロボットへ, 第 6 章, pp. 151–198. 啓学出版, May 1984. 小杉 幸夫, 林 巌, 亀井 宏行 訳.

[3] 安西祐一郎．認識と学習, 岩波講座ソフトウェア科学, 第 16 巻．岩波書店, Feb. 1989.

[4] Arkin, R. C. Integrating behavioral, perceptual, and world knowledge in reactive navigation. In Maes, P., editor, *Designing Autonomous Agents*, pp. 105–122. The MIT Press, 1990.

[5] 浅田稔, 野田彰一, 依積田健, 細田耕．視覚に基づく強化学習によるロボットの行動獲得．日本ロボット学会誌, Vol. 13, No. 1, pp. 68–74, Jan. 1995.

[6] Asada, M., Noda, S., and Hosoda, K. Non-physical intervention in robot learning based on LfE method. In *Proceedings of Machine Learning Conference Workshop on Learning from Examples vs. Programming by Demonstration*, 1995.

[7] Asada, M., Noda, S., and Hosoda, K. Action-based sensor space categorization for robot learning. In *Proceedings of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 96)*, Vol. 3, pp. 1502–1509, Nov. 1996.

[8] Asada, M., Noda, S., Tawaratsumida, S., and Hosoda, K. Vision-based reinforcement learning for purposive behavior acquisition. In *Proceedings of IEEE International. Conference. on Robotics and Automation*, 1995.

[9] 麻生英樹. ニューラルネットワーク情報処理 — コネクショニズム入門、あるいは柔らかな記号に向けて —. 産業図書, June 1988.

[10] 麻生英樹, 本村陽一, 松井俊浩, 速水悟, 原功. 対話ベース学習によるオフィス環境での自律ロボットナビゲーション. 合同研究会 "AI シンポジウム'95", pp. 40–47, Dec. 1995. 人工知能学会研究会資料 SIG-J-9501.

[11] 麻生英樹. 情報統合 — 課題と展望 —. 人工知能学会誌, Vol. 11, No. 2, pp. 185–192, Mar. 1996.

[12] 麻生英樹, 赤穂昭太郎, 本村陽一. 統計的推論と AI の推論. 人工知能学会誌, Vol. 12, No. 2, pp. 196–203, Mar. 1997.

[13] Barto, A. G., Bradtke, S. J., and Singh, S. P. Learning to act using real-time dynamic programming. *Artificial Intelligence*, Vol. 72, pp. 81–138, 1995.

[14] Barto, A. G., Sutton, R. S., and Anderson, C. W. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-13, No. 5, pp. 834–846, Sep./Oct. 1983.

[15] Barto, A. G., Sutton, R. S., and Watkins, C. J. C. H. Learning and sequential decision making. In M.Gabriel and Moore, J. W., editors, *Learning and Computational Neuroscience: Foundations of Adaptive Networks*, pp. 539–602. MIT Press, 1990.

[16] Brooks, R. A. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, Vol. 2(1), pp. 14–23, Mar. 1986.

[17] Brooks, R. A. Herbert: A second generation mobile robot, Jan. 1988. MIT AI Lab Memo 1016.

[18] Brooks, R. A. 自律移動ロボット. Grimson, W. E. L. and Patil, R. S., editors, MIT の人工知能（原題: "AI in the 1980s and Beyond"）, pp. 329–348. パーソナルメディア社, 1989. 伊庭 斉志 訳.

[19] Brooks, R. A. Elephants don't play chess. In Maes, P., editor, *Designing Autonomous Agents*, pp. 3–15. The MIT Press, 1990.

[20] Brooks, R. A. Artificial life and real robots. In *Proceedings of the First European Conference on Artificial Life*, pp. 3–10, 1991.

[21] Brooks, R. A. Intelligence without representation. *Artificial Intelligence*, Vol. 47, pp. 139–159, Jan. 1991.

[22] Brooks, R. A. and Stein, L. A. Building brains for bodies. A.I. memo No. 1439, Aug. 1993.

[23] Carpenter, G. A. and Grossberg, S. The ART of adaptive pattern recognition by a self-organizing neural network. *Computer*, Vol. 21, No. 3, pp. 77–88, Mar. 1988.

[24] Chapman, D. and Kaelbling, L. P. Input generalization in delayed reinforcement learning: An algorithm and performance comparisons. In *Proceedings of IJCAI-91*, pp. 726–731, 1991.

[25] Dennett, D. コグニティブ・ホイール: 人工知能におけるフレーム問題（原題: Cognitive wheels: the frame problem of AI）. 現代思想, Vol. 15, No. 5, pp. 128–150, 1990. 信原 幸弘 訳.

[26] Doya, K. Temporal difference learning in continuous time and space. In Touretzky, D. S., Mozer, M. C., and Hasselmo, M. E., editors, *Advances in Neural Information Processing System*, Vol. 8. MIT Press, 1996.

[27] Fisher, D. H. Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, Vol. 2, pp. 139–172, 1987.

[28] Fisher, D. H., Pazzani, M. J., and Langley, P., editors. *Concept Formation: Knowledge and Experience in Unsupervised Learning*. The Morgan Kaufmann Series in Machine Learning. Morgan Kaufman, 1991.

[29] Gachet, D., Salichs, M. A., Moreno, L., and Pimentel, J. R. Learning emergent tasks for an autonomous mobile robot. In *Proceedings of the IEEE/RSJ/GI International Conference on Intelligent Robots and Systems*, pp. 290–297. IEEE, 1994.

[30] Ginsberg, M. L. and Smith, D. E. Reasoning about action I: A possible worlds approach. *Artificial Intelligence*, Vol. 35, No. 2, pp. 165–195, 1988.

[31] Ginsberg, M. L. and Smith, D. E. Reasoning about action II: The qualification problem. *Artificial Intelligence*, Vol. 35, No. 3, pp. 311–342, 1988.

[32] Harnad, S. The symbol grounding problem. *Physica. D, Nonlinear phenomena*, Vol. 42, pp. 335–346, 1990.

[33] 橋田浩一, 松原仁. 知能の設計原理に関する試論 －部分性・制約・フレーム問題－, 1993. 日本認知科学会「認知科学の発展」原稿.

[34] Hasida, K. and Matsubara, H. Partiality of information and the structure of the frame problem. In *Proceedings of PRICAI '90*, pp. 711–716, 1990.

[35] Hinton, G. E. Mapping part-whole hierarchies into connectionist networks. *Artificial Intelligence*, Vol. 46, No. 1-2, pp. 47–75, 1990.

[36] 開一夫. Soar における学習－認知科学的側面－. 人工知能学会誌, Vol. 9, No. 4, pp. 497–504, July 1994.

[37] 開一夫, 松原仁. 機械学習からみたロボット学習－能動的学習機構に向けて－. 日本ロボット学会誌, Vol. 13, No. 1, pp. 5–10, Jan. 1995.

[38] Holland, J. H. Escaping brittleness. In Michalski, R. S., Carbonell, J. G., and Mitchell, T. M., editors, *Machine Learning: An Artificial Intelligence Approach*, Vol. II, pp. 593–623. Morgan Kaufmann, 1986.

[39] Hori, K. and Ohsuga, S. Articulation problem - a basic problem for information modelling. In Kangassalo, H., Ohsuga, S., and Jaakkola, H., editors, *Information Modelling and Knowledge Bases*, pp. 36–44. IOS Press, 1990.

[40] Horswill, I. Polly: A vision-based artificial agent. In *Proceedings of the Eleventh National Conference on Artificial Intelligence (AAAI-'93)*, pp. 824–829, 1993.

[41] 伊庭幸人. 基礎的問題から見た情報統合. 人工知能学会誌, Vol. 11, No. 2, pp. 193–200, Mar. 1996.

[42] 石田亨. 移動目標探索アルゴリズムとその性能改善. 人工知能学会誌, Vol. 8, No. 6, pp. 760–768, Nov. 1993.

[43] 石田亨. 実時間探索と強化学習：LRTA* から Q-Learning への系譜, 1994. けいはんな自律エージェント研究会報告書.

[44] Ishiguro, H., Sato, R., and Ishida, T. Robot oriented state space construction. In *Proceedings of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 96)*, Vol. 3, pp. 1496–1501, Nov. 1996.

[45] 磯田佳徳, 山田誠二, 豊田順一. 成功確率によるプランニングと実行のインターリーブ. 情報処理学会第 45 回全国大会, 第 2 巻, pp. 135–136, 1992.

[46] 磯田佳徳, 山田誠二, 豊田順一. インターリーブ・プランニングとその実験的評価. 情報処理学会第 86 回人口知能研究会, 1993. 93-AI-86-8.

[47] 伊藤哲郎, 葉玲如, 中島誠. 構造化領域での生成的概念形成と記述の扱い. 人工知能学会誌, Vol. 9, No. 6, pp. 917–926, Nov. 1994.

[48] 岩津賢, 椹木哲夫, 堀内匡, 片井修. 概念形成による履歴からの状態空間の再帰的構成に基づく行動学習. 第 24 回知能システムシンポジウム, 1997.

[49] 情報処理学会（編）. 新版 情報処理ハンドブック. オーム社, Nov. 1995.

[50] Kaelbling, L. P. On reinforcement learning for robots. In *Proceedings of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 96)*, Vol. 3, pp. 1319–1320, Nov. 1996.

[51] Kakazu, Y. and Hakura, J. Explorations of perceptional mechanisms for adaptive agents. In *Proceedings of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 96)*, Vol. 3, pp. 1699–1706, Nov. 1996.

[52] 北村新三, 片山修. ニューラルネットとロボットの学習. 日本ロボット学会誌, Vol. 13, No. 1, pp. 63–67, Jan. 1995.

[53] Kohonen, T. *Self-Organizing Maps*. Springer-Verlag, 1995. 徳高 平蔵, 岸田悟, 藤村 喜久郎 訳: 自己組織化マップ.

[54] Koza, J. R. Evolution of subsumption using genetic programming. In *Proceedings of the First European Conference on Artificial Life*, pp. 110–119, 1991.

[55] Kröse, B. J. A. and Eecen, M. A self-organizing representation of sensor space for mobile robot navigation. In *Proceedings of the IEEE/RSJ/GI International Conference on Intelligent Robots and Systems*, pp. 9–14. IEEE, 1994.

[56] Kröse, B. J. A. and van Dam, J. W. M. Adaptive state space quantisation : Adding and removing neurons. In Aleksander, I. and Taylor, J., editors, *Artificial Neural Networks,2*, pp. 619–624. North-Holland/Elsevier Science Publishers, Amsterdam, 1992.

[57] Kröse, B. J. A. and van Dam, J. W. M. Adaptive state space quantisation for reinforcement learning of collision-free navigation. In *Proceedings of the 1992 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1327–1332. IEEE, Piscataway, NJ, Raleigh, N.C., Jun. 1992.

[58] Kröse, B. J. A. and van Dam, J. W. M. Learning to avoid collisions: a reinforcement learning paradigm for mobile robot navigation. In *Proceedings of the 1992 IFAC/IFIP/IMACS Symposium on Artificial Intelligence in Real-Time control*, pp. 295–301. IFAC, Delft, the Netherlands, Jun. 1992.

[59] 國吉康夫. 実世界エージェントにおける注意と視点 — 情報の分節・統合・共有 —. 人工知能学会誌, Vol. 10, No. 4, pp. 508–514, July 1995.

[60] Kuniyoshi, Y. and Inoue, H. Qualitative recognition of ongoing human action sequences. In *Proceedings of IJCAI-93*, pp. 1600–1609, 1993.

[61] Lin, L.-J. Programming robots using reinforcement learning and teaching. In *Proceedings of the Ninth National Conference on Artificial Intelligence (AAAI-'91)*, pp. 781–786, 1991.

[62] Lin, L.-J. Self-improving reactive agents: Case studies of reinforcement learning frameworks. In *Proceedings of the First International Conference on Simulation of Adaptive Behavior: From Animals to Animats*, pp. 297–305, 1991.

[63] Lin, L.-J. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine Learning*, Vol. 8, No. 3/4, pp. 293–321, May 1992.

[64] Maes, P. Situated agents can have goals. In Maes, P., editor, *Designing Autonomous Agents*, pp. 49–70. The MIT Press, 1990.

[65] Maes, P. Learning behavior networks from experience. In *Proceedings of the First European Conference on Artificial Life*, pp. 48–57, 1991.

[66] Mahadevan, S. and Connell, J. Automatic programming of behavior-based robots using reinforcement learning. In *Proceedings of the Ninth National Conference on Artificial Intelligence (AAAI-'91)*, pp. 768–773, 1991.

[67] Martín, P. and del R. Millán, J. Reinforcement learning of sensor-based reaching strategies for a two-link manipulator. In *Proceedings of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 96)*, Vol. 3, pp. 1345–1352, Nov. 1996.

[68] 松原仁, 橋田浩一. 情報の部分性とフレーム問題の解決不能性. 人工知能学会誌, Vol. 4, No. 6, pp. 695–703, Nov. 1989.

[69] 松原仁. AIマップ — 辻 三郎先生の記事へのコメント. 人工知能学会誌, Vol. 8, No. 5, pp. 543–550, Sep. 1993.

[70] 松原仁. 人工知能におけるロボットの役割. 日本ロボット学会誌, Vol. 14, No. 4, pp. 478–481, May 1996.

[71] Mitchell, T. M. Becoming increasingly reactive. In *Proceedings of the Eighth National Conference on Artificial Intelligence (AAAI-'90)*, pp. 1051–1058, 1990.

[72] 宮崎和光, 山村雅幸, 小林重信. 強化学習における報酬割当ての理論的考察. 人工知能学会誌, Vol. 9, No. 4, pp. 580–587, July 1994.

[73] 宮崎和光, 山村雅幸, 小林重信. k-確実探索法：強化学習における環境同定のための行動選択戦略. 人工知能学会誌, Vol. 10, No. 3, pp. 454–463, May 1995.

[74] 宮崎和光, 山村雅幸, 小林重信. l-確実探索法：エージェントによる環境同定のための行動選択戦略 —k-確実探索法の不確実性下への拡張 —. 人工知能学会誌, Vol. 11, No. 5, pp. 804–808, Sep. 1996.

[75] 宮崎和光, 山村雅幸, 小林重信. MarcoPolo: 報酬獲得と環境同定のトレードオフを考慮した強化学習システム. 人工知能学会誌, Vol. 12, No. 1, pp. 78–89, Jan. 1997.

[76] 毛利隆夫. Nearest Neighbor 法と記憶に基づく推論. 人工知能学会誌, Vol. 12, No. 2, pp. 188–195, Mar. 1997.

[77] Moore, A. W. Variable resolution dynamic programming: Efficiently learning action maps in multivariate real-valued state-spaces. In *Proceedings of the Eighth International Workshop on Machine Learning (ML91)*, pp. 333–337, 1991.

[78] 長尾真. 知識と推論, 岩波講座ソフトウェア科学, 第 14 巻. 岩波書店, July 1988.

[79] 長尾真, 石田晴久, 稲垣康善, 田中英彦, 辻井潤一, 所真理雄, 中田育男, 米澤明憲（編）. 岩波 情報科学辞典. 岩波書店, May 1990.

[80] 中川聖一, 中西宏文, 古部好計, 板橋光義. 視聴覚情報の統合化に基づく概念の獲得. 人工知能学会誌, Vol. 8, No. 4, pp. 499–508, July 1993.

[81] 中川聖一, 升方幹雄. 視聴覚情報の統合化に基づく概念と文法の獲得システム. 人工知能学会誌, Vol. 10, No. 4, pp. 619–627, July 1995.

[82] 中村恭之, 浅田稔. 運動スケッチ：画像運動情報に基づく単眼視覚移動ロボットの行動 獲得. 人工知能学会誌, Vol. 11, No. 6, pp. 905–915, Nov. 1996.

[83] Nakamura, T. and Asada, M. Motion sketch: Acquisition of visual motion guided behaviors. In *Proceedings of IJCAI-95*, pp. 126–132, 1995.

[84] Nakamura, T., Takamura, S., and Asada, M. Behavior-based map representation for a sonar-based mobile robot by statistical methods. In *Proceedings of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 96)*, Vol. 1, pp. 276–283, Nov. 1996.

[85] 中野馨, 磯谷亮輔, 大森隆司. 情報交換機能を自己組織的に獲得するシステム — 言語発生のプリミティブモデル. 電子情報通信学会論文誌 A, Vol. J70-A, No. 5, pp. 806–815, May 1987.

[86] 中島秀之, 松原仁, 大澤一郎. 因果関係によるフレーム問題へのアプローチ. 人工知能学会誌, Vol. 8, No. 5, pp. 619–627, Sep. 1993.

[87] 中島秀之. 手足を持った人工知能. 日本ロボット学会誌, Vol. 14, No. 4, pp. 482–484, Mar. 1996.

[88] 中島秀之. 情報統合のための有機的プログラミング. 人工知能学会誌, Vol. 11, No. 2, pp. 201–208, Mar. 1996.

[89] Nakasuka, S. and Koishi, T. Automated extraction of attribute hierarchies for an improved decision-tree classifier. *Engng Applic. Artif. Intell.*, Vol. 8, No. 4, pp. 391–399, 1995.

[90] Nakasuka, S., Yairi, T., and Wajima, H. Autonomous generation of reflexion-based robot controller using inductive learning. *Robotics and Autonomous Systems*, Vol. 17, pp. 287–305, 1996.

[91] Nakasuka, S. and Yoshida, T. Dynamic scheduling system utilizing machine learning as a knowledge acquisition tool. *International Journal of Production Research*, Vol. 30, No. 2, pp. 411–431, 1992.

[92] 野田彰一, 浅田稔, 細田耕. 強化学習によるロボットの行動獲得のための状態空間の自律的構成. 第 5 回ロボットシンポジウム予稿集, 1995.

[93] 大澤一郎, 中島秀之. 因果関係と状況に基づく動作効果の動的計算. 人工知能学会誌, Vol. 10, No. 2, pp. 222–231, Mar. 1995.

[94] 大澤真幸. 知能の社会性. 日本ロボット学会誌, Vol. 14, No. 4, pp. 485–489, Mar. 1996.

[95] 岡田豊史, 開一夫, 安西祐一郎. ロボットコマンド学習システム Acorn-II とその評価. 人工知能学会誌, Vol. 9, No. 6, pp. 882–889, Nov. 1994.

[96] 奥野忠一, 久米均, 芳賀敏郎, 吉澤正. 多変量解析法. 日科技連, 1971.

[97] 小野典彦, Adel, T. R. 人工生物群による通信の自己組織化. 石田亨（編）, マルチエージェントと協調計算 II: 日本ソフトウェア科学会 MACC'92, レクチャーノート / ソフトウェア学 5, 第 13 章, pp. 181–195. 近代科学社, Dec. 1993.

[98] Pfeifer, R. Cognition, 1993. Trento ASI.

[99] Quinlan, J. R. Learning efficient classification procedures and their application to chess end games. In Michalski, R. S., Carbonell, J. G., and Mitchell, T. M., editors, *Machine Learning: An Artificial Intelligence Approach*, Symbolic Computation, chapter 15, pp. 463–482. Springer-Verlag, 1984.

[100] Quinlan, J. R. *C4.5: Programs for Machine Learning*. The Morgan Kaufmann Series in Machine Learning. Morgan Kaufmann, 1993.

[101] Rumelhart, D. E., McClelland, J. L., and the PDP Research Group. PDP モデル — 認知科学とニューロン回路網の探索 — (原題: Parallel Distributed Processing — Explorations in the Microstructure of Cognition—). 産業図書, Feb. 1989. 甘利 俊一 監訳.

[102] Russell, S. and Norvig, P. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 1995.

[103] 佐伯胖. 認知科学の方法, 認知科学選書, 第 10 巻. 東京大学出版会, Dec. 1986.

[104] 齋藤史倫, 福田敏男. 強化学習による実ロボットの運動制御. 日本ロボット学会誌, Vol. 13, No. 1, pp. 82–88, Jan. 1995.

[105] Salganicoff, M. Active learning for vision-based robot grasping. *Machine Learning*, Vol. 23, No. 2/3, pp. 251–278, May/June 1996.

[106] 幸島明男, 間一夫, 錦見美貴子, 仁木和久. 行動のチャンキングと強化学習. 合同研究会 "AI シンポジウム'94", pp. 71–78, Dec. 1994. 人工知能学会研究会資料 SIG-J-9401.

[107] Sato, R. Robot oriented state space construction. Master's thesis, Kyoto University, Feb. 1996.

[108] 槻木哲夫. 資源制約を有するエージェントの設計と概念学習. 日本ロボット学会誌, Vol. 13, No. 1, pp. 44–50, Jan. 1995.

[109] Sawaragi, T., Sawada, H., and Katai, O. Reinforcement learning for autonomous mobile robots by forming approximate classificatory concepts. In *Proceedings of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 96)*, Vol. 3, pp. 1337–1344, Nov. 1996.

[110] 白坂成功. 自律型ローバーシステムの構成法に関する研究. Master's thesis, 東京大学, Mar. 1994.

[111] Simons, J., Brussel, H. V., Schutter, J. D., and Verhaert, J. A self-learning automaton with variable resolution for high precision assembly by industrial

robots. *IEEE Transactions on Automatic Control*, Vol. AC-27, No. 5, pp. 1190–1113, Oct. 1982.

[112] Sutton, R. S. Learning to predict by the method of temporal differences. *Machine Learning*, Vol. 3, No. 1, pp. 9–44, 1988.

[113] Sutton, R. S. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Proceedings of the Seventh International Conference on Machine Learning*, pp. 216–224. Morgan Kaufmann, 1990.

[114] Takahashi, Y., Asada, M., and Hosoda, K. Reasonable performance in less learning time by real robot based on incremental state space segmentation. In *Proceedings of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 96)*, Vol. 3, pp. 1518–1524, Nov. 1996.

[115] 谷淳. ロボットにおける認知と自律性の構造：力学系の見地から. 日本ロボット学会誌, Vol. 14, No. 4, pp. 474–477, May 1996.

[116] Tesauro, G. Practical issues in temporal difference learning. *Machine Learning*, Vol. 8, No. 3/4, pp. 257–277, May 1992.

[117] 東京大学教養学部統計学教室 (編). 統計学入門, 基礎統計学, 第1巻. 東京大学出版会, 1991.

[118] 辻三郎. AI マップ — ロボットから見た AI. 人工知能学会誌, Vol. 8, No. 4, pp. 404–409, July 1993.

[119] 上野敦志, 堀浩一, 中須賀真一. 報酬に基づく状況認識と状況に基づく行動選択の同時学習. 第 24 回人工知能基礎論研究会, pp. 38–45, Mar. 1996. 人工知能学会研究会資料 SIG-FAI-9503.

[120] Ueno, A., Hori, K., and Nakasuka, S. Simultaneous learning of situation classification based on rewards and behavior selection based on the situation. In *Proceedings of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 96)*, Vol. 3, pp. 1510–1517, Nov. 1996.

[121] 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, May 1990.

[122] 畝見達夫. 実例に基づく強化学習法. 人工知能学会誌, Vol. 7, No. 4, pp. 697–707, July 1992.

[123] 畝見達夫. 強化学習. 人工知能学会誌, Vol. 9, No. 6, pp. 830–836, Nov. 1994.

[124] 畝見達夫. 強化学習法とロボットへの応用. 日本ロボット学会誌, Vol. 13, No. 1, pp. 51–56, Jan. 1995.

[125] Watkins, C. J. C. H. and Dayan, P. Technical note: Q-learning. *Machine Learning*, Vol. 8, No. 3/4, pp. 279–292, May 1992.

[126] Wettschereck, D. and Dietterich, T. G. An experimental comparison of the nearest-neighbor and nearest-hyperrectangle algorithms. *Machine Learning*, Vol. 19, No. 1, pp. 5–27, Apr. 1995.

[127] Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, Vol. 8, No. 3/4, pp. 229–256, May 1992.

[128] 谷内田正彦, 山口智浩. 視覚からのモデルと概念の学習. 日本ロボット学会誌, Vol. 13, No. 1, pp. 25–31, Jan. 1995.

[129] 矢入健久, 堀浩一, 中須賀真一, 輪島裕之. 自律ローバーのための属性空間分割によるサブゴール列学習. 第13回日本ロボット学会学術講演会予稿集, 第2巻, pp. 553–554, 1995.

[130] 矢入健久. 惑星上ローバーのための自律行動学習系. Master's thesis, 東京大学, Mar. 1996.

[131] 山田誠二. インターリーブによるリアクティブ・プランニング. 情報処理学会第79回人口知能研究会, 1991. 91-AI-79-8.

[132] 山田誠二, 磯田佳徳, 豊田順一. インターリーブによるリアクティブ・プランニングの枠組み — プランの評価とプランニング. 情報処理学会第43回全国大会, 第2巻, pp. 125–126, 1991.

[133] 山田誠二, 磯田佳徳, 豊田順一. インターリーブによるリアクティブ・プランニングの枠組み — 実行 / 観測のタイミング決定. 情報処理学会第43回全国大会, 第2巻, pp. 123–124, 1991.

[134] Yamada, S. Reactive planning with uncertainty of a plan. In *Proceedings of The Third Annual Conference on AI, Simulation and Planning in High Autonomy Systems*, pp. 201–208, 1992.

[135] 山田誠二. ベイジアン・ネットワークによるプランの表現と成功確率の計算. 人工知能学会全国大会 (第7回) 論文集, pp. 187–190, 1993.

[136] 山田誠二. リアクティブプランニング. 人工知能学会誌, Vol. 8, No. 6, pp. 729–735, Nov. 1993.

[137] 山田誠二. エージェントのプランニング. 人工知能学会誌, Vol. 10, No. 5, pp. 677–682, Sep. 1995.

[138] 山田誠二. リアクティブプランニングにおける学習. 日本ロボット学会誌, Vol. 13, No. 1, pp. 38–43, Jan. 1995.

[139] 山田誠二, 磯田佳徳, 豊田順一. 単純タイルワールドにおける SIP の実験的評価. 人工知能学会誌, Vol. 11, No. 5, pp. 744–751, Sep. 1996.

[140] 山田誠二. 動的環境における成功確率を用いた熟考の制御. 人工知能学会誌, Vol. 11, No. 4, pp. 645–652, July 1996.

[141] 山村雅幸, 宮崎和光, 小林重信. エージェントの学習. 人工知能学会誌, Vol. 10, No. 5, pp. 683–689, Sep. 1995.

[142] Yee, R. Abstraction in control learning. Technical report, Department of Computer and Information Science, University of Massachusetts, Amherst, MA, 1992. COINS Technical Report 92-16.

[143] 吉田典晃. サブサンプションアーキテクチャを用いたロボットの制御. 日本ロボット学会誌, Vol. 11, No. 8, pp. 1118–1123, Nov. 1993.

参考文献

発表文献リスト

- 上野 敦志, 堀 浩一, 中須賀 真一： 報酬に基づく状況認識と状況に基づく行動選択の同時学習, 人工知能学会 第 24 回人工知能基礎論研究会資料（SIG-FAI-9503）, pp. 38–45, March 1996.
- Atsushi Ueno, Koichi Hori, and Shinichi Nakasuka: Simultaneous Learning of Situation Classification based on Rewards and Behavior Selection based on the Situation, In *Proceedings of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 96)*, Vol. 3, pp. 1510–1517, November, 1996.
- 上野 敦志, 堀 浩一, 中須賀 真一： 自律エージェントのための状況認識と行動規則の同時学習, 人工知能学会 第 30 回人工知能基礎論研究会資料（SIG-FAI-9702）, in printing, September 1997.