複数タスクに対するロボットの行動学習

- タスクの重要度に基づく行動選択手法の提案 -

三浦　丈典

第１章　序論

１．１　ロボットの学習について

近年では，ロボットの性能の進化や汎用性の向上から，ロボットの適用・活躍範囲は一般家庭やエンターテイメントなど多方面に広がっている．iRobot 社で製作された「ルンバ」に代表される家庭用ロボット掃除機は一般家庭に広く普及している．また，パートナーロボットとしては本田技研工業が製作している ASIMO が有名であり，その他介護福祉の分野でも介護支援ロボット「RIBA」等の開発が進められている (図 1.1)．このような現状から，ロボットは人間の生活空間のような多様な環境の中で複雑で柔軟性のある振る舞いを要求されるようになってきている．この期待に答えるべく，ロボットに対して人間のような学習機能を持たせる研究が行われている．人間は，周りの環境が変化した場合でも過去の経験からある程度の予測を行ない行動することが可能である．ロボットも人間のように過去の経験を蓄積し環境と試行錯誤することで環境変化に対応した行動を行なうというものである．

ロボットに学習を行わせる研究は様々な方法で試みられている．ロボットの学習手法で最も一般的なのは機械学習である．機械学習とは，ロボットが経験した情報を基にロボットの行動を決定する手法であり，教師あり学習，教師なし学習，強化学習[1]などがある．本研究では，機械学習の中で最もロボットへの適応が一般的な強化学習に注目する．

強化学習が適応されるロボットにおいては，人間がロボットに対してロボットに行ってもらいたい仕事（タスク）を与える．ロボットはそのタスクを達成するためにどのような行動を行えばよいのか学習し，試行錯誤の後にタスク達成を目指すものである．

「ASIMO」本田技研工業　　「ルンバ」iRobot　　「RIBA」東海ゴム人間共存ロボット連携センター

図 1.1 社会で活躍するロボット

１．２　複数タスクに対する学習

通常の強化学習においては，ロボットは人間から１つのタスクが与えられ，そのタスクの達成を目標としている．工場などで１つの作業を専門的に行うようなロボットであれば１つのタスクを達成できれば十分であるが，近年のロボットの能力の向上によってロボットの活躍の場所が広がっていることから，１つのことを専門的に行うロボットだけでなく，複数のことを汎用的に行うロボットの研究が進められている．

実際に様々なタスクをこなすことができるロボットを考えれば，ロボットは人間に与えられたタスクの他に，自身の存続を保つためにエネルギーを確保する必要や危険を回避する必要などもある．これか一つ一つを１つのタスクと考えれば，ロボットは常に複数のタスク下で行動決定を行う必要がある．タスクにはそれぞれ達成したい目標があり，そのために取るべき行動がある．当然，複数のタスクを同時に有する場合には，それぞれのタスク毎に取るべき行動が存在する．複数のタスク間で取るべき行動に相違が無い場合には，ロボットは１つの行動を選択することができるが，タスク毎に取るべき行動が違う場合には，何らかの方法によってロボットが最終的に行う行動を決定する必要がある．

１．３　従来研究

複数のタスクについてロボットに学習させようとする研究[2-6]は多く存在する．ここでは，その中から２種類の例として挙げる．１つ目は，大きな１つのタスクを分割し複数のタスクとして扱い学習の効率をあげようとする研究．もう一つが，２つ以上の異なるタスクを同時に与えられたロボットが複数のタスクに対して，学習を行い行動すると言うものである．まず，前者の研究の例としては階層型強化学習[7-10]が挙げられる．階層型強化学習とは文字通り強化学習を階層化し，問題を分割することで学習収束までの時間を高速化する手法である．この手法では，強化学習を階層化することで複数の問題を含む大きな問題を分割し，学習する状態空間を小さくすることができる．後者の研究例としては，重み付き報酬関数を用いて複数タスクを学習する研究や，多目的最適化問題を強化学習に適応した研究等がある．

重み付き報酬関数を用いる研究とは，１つの報酬関数で複数のタスクを表す研究[11]である．この手法では，各タスクのために設計された報酬関数を足しあわせて，重み付けをすることで１つの報酬関数として表現し，１つの学習空間によって複数タスクの学習を行うという．重み付けは，タスク毎に付加される．この報酬関数に付加された重みを変化させることで，複数タスクの中で環境変化によってどのタスクを優先した行動選択を行うかを決定している．

ここでは，ロボットに対して３つのタスクが与えられていた場合を例にして説明する．各タスクに対する報酬をそれぞれr_1，r_2，r_3とし，それぞれのタスクに対して与えられる重みをw_1，w_2，w_3とすると，これらの報酬を統合した報酬関数は（1.1）式の様な形で表される．

$$R = w_1 r_1 + w_2 r_2 + w_3 r_3 \tag{1.1}$$

（1.1）式のように，複数のタスクを１つの報酬関数に統合することで，１つの学習空間で学習を行なうことが可能となる．１つの学習空間で学習が可能であれば，ある状態においてロボットが選択可能な行動に対する評価値も 1 つずつしか存在しないので，従来の強化学習で用いられる行動選択手法を適用することができる（図 1.2）．

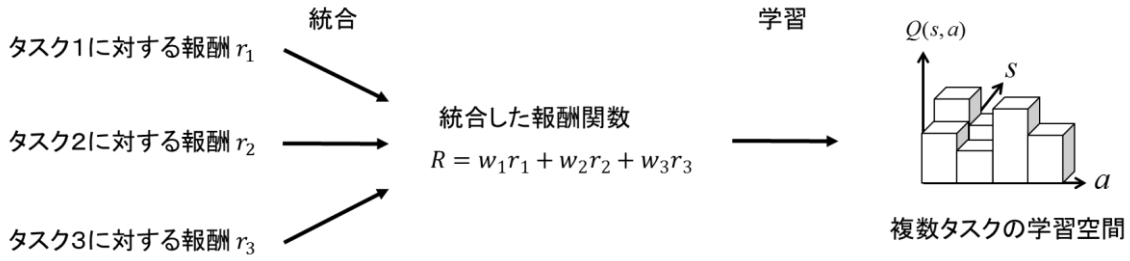

図 1.2　重み付き報酬関数を用いた手法概要図

多目的最適化問題を強化学習に適応した研究[12-14]は，ロボットの行動がどの行動をとるべきかを解とする多目的最適化問題とみなし，ロボットの行動決定を行うものである．この手法は，経済学やシステム工学の分野で古くから研究されている多目的最適化問題を，複数タスク下でのロボットの行動選択に適応した研究である．多目的最適化問題では，この手法では１つのタスクを１つの目的関数として定義し，多目的最適化を行う．例としては，強化学習の一種である Actor-Critic アーキテクチャに基づく強化学習法を提案したものがある．この研究では，複数のタスクに対して 1 つの価値関数を学習するのではなく，タスク毎に与えられた報酬関数から各タスクに対する価値関数を学習し，それらのパレート最適方策を得るというものである． パレート最適解とは，多目的最適化問題においてどの目的関数も他の目的関数を劣化させずにそれ以上最適化できないような解である．このパレート最適解は複数存在する場合があるが，その場合には多目的最適化問題を 1 つの目的関数の最適化問題へと変換する，スカラー化を行い最終的な行動を決定している．スカラー化において用いる重みベクトルについては，人間が設定して与えることになる．

１．４　従来研究の問題点

前節では，複数タスクに対するロボットの行動学習についていくつかの従来研究を挙げ説明した．本節では，従来研究の問題的について説明する．

複数のタスクそれぞれの報酬関数を重み付けして足し合わせ，１つの報酬関数によって表現する手法では，学習空間は１つだけ存在しその学習空間から行動選択を行う．しかし，１つの報酬関数によって学習を行う手法では，タスク間の関係性が変化すると報酬関数に付加されている重みも変化する．重みが変化すると当然報酬関数によって算出される報酬値が変化するので，ロボットの行動に対する評価値が変わってしまう．従って，１つの報酬関数によって学習する手法では，タスク間の重みが変化した場合には，再学習を行なう必要がある．また，新たなタスクを追加する場合や，不要となったタスクを削除する場合にも，報酬関数が変化してしまうので再学習をする必要がある．ロボットに与えられるタスクと使われる環境が限定されている場合には，この手法は有用であるがロボットに多種多様なタスクを与え様々な環境で用いる場合にはこの手法では適応に時間がかかってしまう．

多目的最適化を複数タスクの強化学習に適用した研究では，各タスクの報酬関数を多目的最適化問題における目的関数とみなし，パレート最適解を導出する．パレート最適解を導出した後は，スカラー化を行い１つの最適化問題へ変換するが，このときに用いる重みベクトルの値は人間が各タスクの関係を考え適切に設定する必要がある．従って，最終的には人間が事前に決めた重みによってロボットの行動が決定されることになる．ロボットに与えられるタスクが複数あり，その組み合わせも変化する状況においては，ロボットに与えられるタスクの組み合わせや，どのタスクが必要であるのかに応じて，人間が最適な重みベクトルの値を設定することは難しい．

上記の２つの従来研究の問題点をまとめると，ロボットに与えられるタスクが事前に決まっており，各タスクの重要性の関係が変化しない場合では，従来の手法が有用である．しかし，ロボットが学習し行動している間に，各タスクの重要性や優先順位が変動するような状況では，各タスクの重みを再設定しなければならない問題点がある．

１．５　研究目的

本研究の目的は，複数のタスクが与えられるロボットにおいて，ロボットの学習途中で各タスクの重要性や優先度が変化した場合でも変化に応じた行動決定可能な行動学習・行動選択手法を提案することである．これによって，複数タスクが与えられた場合にも，状況に応じた行動選択が可能であり，かつ新たなタスクの追加や削除が容易に行なうことが可能となる．

１．６　アプローチ

ロボットに複数のタスクが与えられる場合には，従来研究の問題点で説明したようにタスクの重要性や優先度が途中で変化することが考えられる．ロボットの行動学習を１つの報酬関数で行っていた場合には，重みが変化する毎に再学習が必要となる．これを防ぐために，本手法では，タスク毎に別々の報酬関数を設計し，タスク毎に学習空間を構成する．タスク毎に学習空間を構成することで，１つの学習空間には１つのタスクについてのみの行動価値が蓄積する．従って，新たなタスクの追加や不要なタスクの削除する場合にはその報酬関数と学習空間だけを追加または削除するだけで良い．しかし，タスク毎に強化学習を適用して個別に学習空間を構成すると，ある状態でロボットが選択すべき行動候補がタスク毎に存在してしまう（図 1.3）．

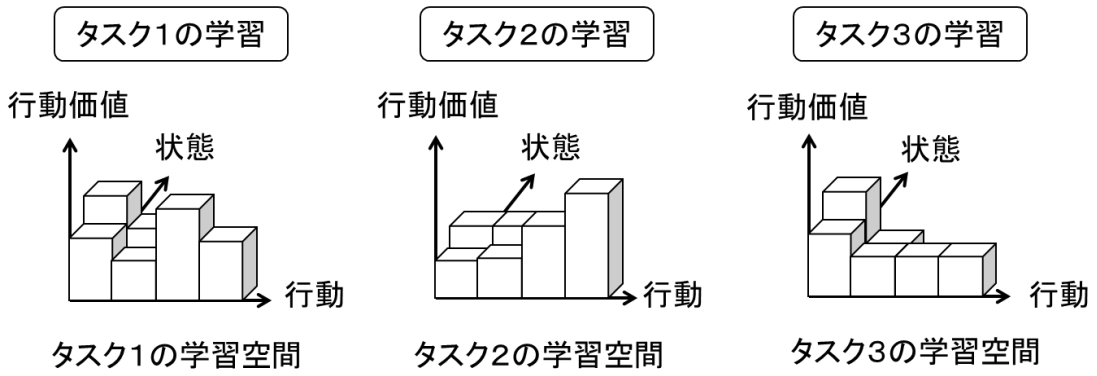

図 1.3　タスク毎の学習空間の構成

タスク毎に存在する行動候補から，最終的なロボットの行動を決定するために本研究では各タスクの重要性に注目する．タスクには，それぞれ重要性が高くなる状況や低くなる状況があり，タスク毎の重要性を考慮し最終的な行動を決定すべきである（図 1.4）．ロボットに搭載されたバッテリーの残量を一定以上に保つようなロボットが生得的に持つタスクの場合には，バッテリー残量が少なくなるほどタスクの重要性は高くなるといったように事前に状態に応じた重要性を設定することができる．また，人間に後天的に与えられるタスクに対しては，人間がどれほどそのタスクの遂行を望んでいるかによってタスクの重要性が変化する．そこで，本研究ではこれらタスク毎に重要性を設定し，その重要性に基づいて最終的な行動決定を行なう（図 1.5）．これによって，複数のタスクが与えられた場合でも，それぞれのタスクの重要性から行動選択が行える行動学習・行動選択手法を実現できると考える．

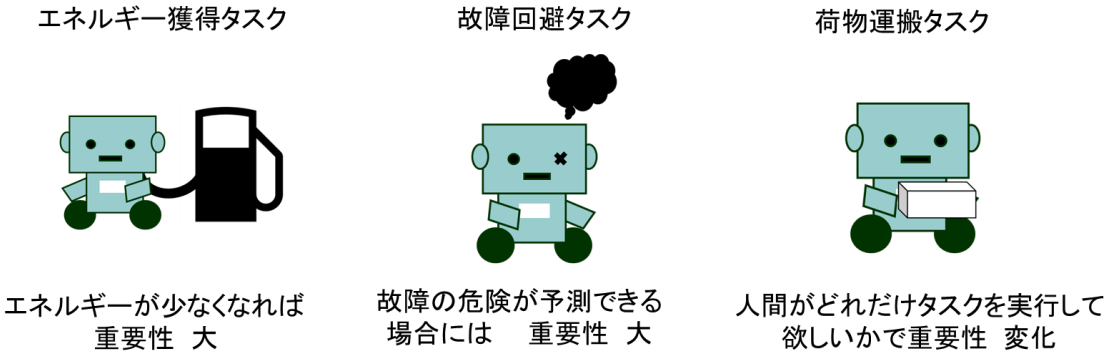

図 1.4　各タスクの重要性の変化

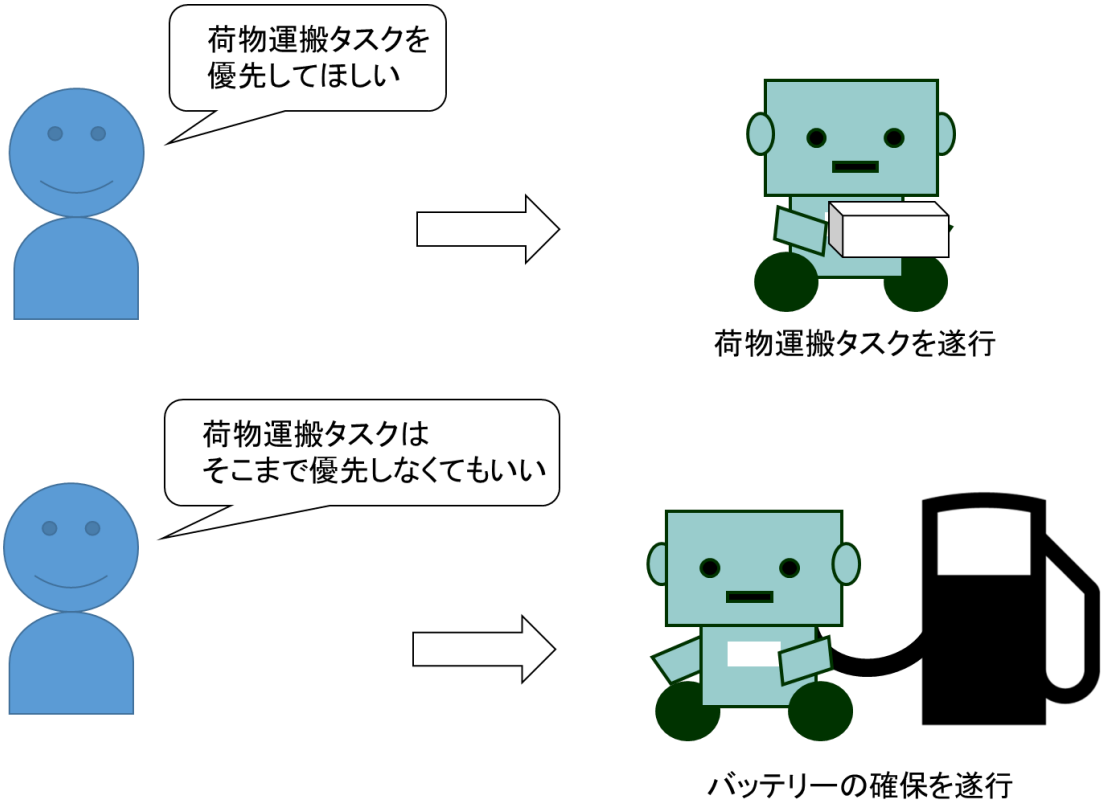

図 1.5　各タスクの重要性の変化に対するロボットの行動の変化

１．７　本論文の構成

第 1 章では，ロボットに学習を適用した研究と複数タスク下での強化学習について説明し，複数タスク下での強化学習ではタスク毎の学習が有効であり，そのためには複数の行動候補の中から状況に応じた行動選択が必要であることを述べた．そして，ロボットに与えられるタスクの重要性に注目し，各タスクの重要性に基づく行動学習・行動選択手法を提案することを目的とした．

第 2 章では，本研究で対象とする強化学習について説明し，ロボットへ適用した場合にどのような流れで学習を行うのかを説明する．また，複数タスクの学習についても触れる．

第 3 章では，提案する行動学習・行動選択手法について説明する．初めに手法の全体的な概要，構成，対象とするタスクについて述べる．その後，提案する手法における学習の方法，行動選択の方法についてそれぞれ詳しく述べる．

第 4 章では，本研究で提案する学習手法の検証実験について述べる．また，結果について考察する．

第 5 章では，検証実験の結果をもとに全体のまとめを述べ，今後の課題についても説明する．

第２章　強化学習

２．１ 強化学習の概要

強化学習[]とは，環境に対する試行錯誤を通じて適切な行動を獲得する機械学習の一種である．ロボットに強化学習を適用することで周りの環境に適応し，より良い行動を選択することが可能となる．

強化学習では，報酬と呼ばれるスカラー値を用いて学習を行う．ロボットが周囲の環境に対して行動することで環境から報酬を獲得することができる．ロボットと環境の関係図を図 2.1 に示す. ロボットは環境との試行錯誤を通して獲得する報酬量を最大化するための行動を選択する．ロボットが環境から与えられる報酬は人間が事前に設定する．従って，ロボットに行ってほしいタスクに応じて，報酬を設定する必要がある．

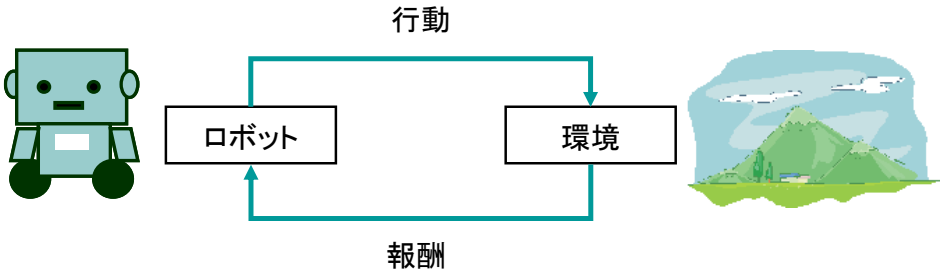

図 2.1　ロボットと環境の関係図

強化学習では，ロボットが行うタスクに対して何が最適な行動なのかを与える必要はない．人間がタスクに対してロボットに行ってほしい行動ほど高い報酬を与え，好ましくない行動に対して低い報酬を与えることで，ロボットは自動的に最善の行動を獲得することができる．従って，強化学習を適用したロボットは未知の環境を扱うことが可能であり，実ロボットに用いられることが多い．

２．２ 学習の流れと学習方法

前節では，強化学習の概要について述べた．本節では強化学習における学習の流れを説明し，具体的な学習方法について述べる．

強化学習で対象とするロボットにはセンサが搭載されており，センサを通して周囲の状態を認識することが可能である．また，ロボットは認識した状態に対して何らかの行動を取ることができる．このようなロボットに対する強化学習の概念図を図 2.2 に示す．

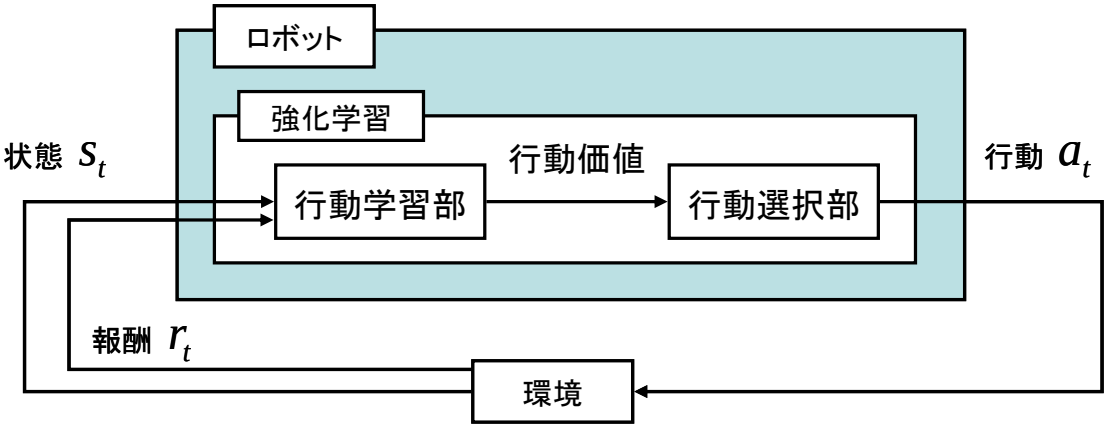

図 2.2　強化学習の概念図

時刻t においてロボットが認識した状態をS_tとする．ロボットは認識した状態S_tに対して，過去の学習結果から行動a_tを選択する．このとき選択した行動に対して環境から報酬r_tを獲得し，獲得した報酬r_tとロボットが認識した状態S_tは行動学習部に入力される．行動学習部では状態S_tと行動a_tの組み合わせに対する行動価値を更新する．この状態S_tと行動a_tの組み合わせを状態行動対と呼ぶ．行動価値とは，ロボットが取った行動によって獲得できる報酬の期待値である．報酬はロボットが取った行動に対して即時的な意味合いで良し悪しを示すものであるのに対し，行動価値は最終的な行動の望ましさを示している．行動価値の更新方法は，用いる行動学習法のよって異なるため詳しくは 2.3 節で説明する．行動学習部で算出された行動価値は行動選択部へ入力される．行動選択部では，行動学習部で更新された行動価値から次の行動を選択する．この行動の選択方法は，用いる行動選択手法によって異なるので詳しくは 2.4 節で説明する．ロボットはこのサイクルを繰り返すことでロボットは目的遂行のために最適な行動を学習することが可能となる．この流れをまとめた図を図 2.3 に示す．

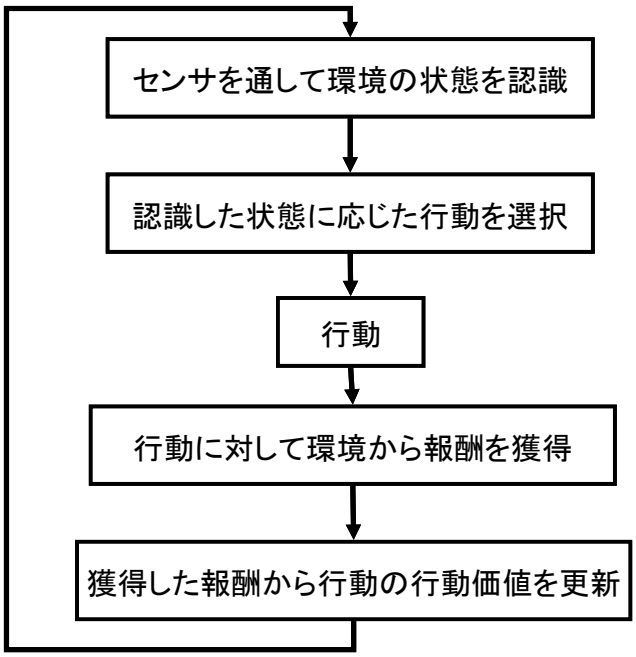

図 2.3　強化学習の流れ

２．３ 強化学習における行動学習手法

行動学習手法は 2.2 節で説明した行動学習部において, 環境から獲得した報酬から行動価値を評価する手法である．ここでは，行動学習法手法の例として Q 学習と加重平均法について説明する．

２．３．１　Q 学習

Q 学習[15]では，行動評価値である Q 値（$Q(s,a)$）が，期待割引収益になるように，評価値を更新する．ある時刻tでのエージェントの状態を状態S_tとし，状態S_tにおいてエージェントが行動a_tを取る．このときエージェントの状態が，状態S_{t+1}に遷移した場合の行動価値を$Q(s_t,a_t)$とすると更新式は（2.1）式のようになる．更新を続けることで，Q 値は$r_{t+1}+\gamma\max_a Q(s_{t+1},a_{t+1})$に収束する．$0<\gamma<1$の場合，最大の Q 値を持つ行動を選択し続けることで，獲得報酬は最大化される．

$$Q(s_t,a_t)\leftarrow Q(s_t,a_t)+\alpha\left[r_{t+1}+\gamma\max_a Q(s_{t+1},a)-Q(s_t,a_t)\right] \qquad (2.1)$$

r_{t+1}：新たに獲得した報酬

α：学習率（$0<\alpha<1$）

γ：割引率（$0<\gamma<1$）

２．３．２　加重平均法

加重平均法は，遠い過去に受け取った報酬を考慮するのか，近い時刻に受け取った報酬のみを考慮するのかを重み付けによって変更し，行動価値を更新する方法である．ある時刻tにおいて状態S_tで行動a_tを取った場合の行動価値$Q(s_t, a_t)$の更新式を式(2.2)に示す．αはステップサイズパラメータと呼ばれる定数で，この値を変化させることによって与える重みが変化する．α の値を大きく設定した場合には，近い時刻に受け取った報酬の影響が大きくなり，α を小さく設定した場合には遠い過去に受けとった報酬も考慮した行動価値の更新となる．

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} - Q(s_t, a_t)] \tag{2.2}$$

$Q(s_t, a_t)$：時刻t，状態S_tで行動a_tを選択した場合の行動価値

r_{t+1}：新たに獲得した報酬

α ：ステップサイズパラメータ（$0 < \alpha < 1$）

２．４ 強化学習における行動選択手法

行動選択手法は 2.2 節で説明した行動選択部において, 過去の行動価値から最適な行動を選択するための方法である．手法の例としては，最も価値の高い行動を選択する方法や，価値の割合に応じて選択する方法などがある．ここでは，代表的な手法について説明する．

２．４．１　ランダム選択法

ランダム選択法とは，行動の評価値とは関係なく，選択できる行動の中からランダムに１つの行動を選択する方法である．評価値の情報を用いないため，学習の進行具合とは関係なく行動を選択することになる．行動の選択確率に偏りがないのが特徴である．未知の環境下においては，どの行動が報酬獲得に必要であるのか，また不必要かがわからないため，偏りのある他の選択手法よりも優れている場合がある．

強化学習で用いる選択手法において，学習初期の行動選択はランダム探索であることが多い．全行動一律の選択確率から始まり，学習が進行するにつれて選択確率が偏る．

２．４．２　greedy 法

ある状態において，最も行動評価値の高い行動を選択する方法が，greedy 法である．ある状態において，各行動の価値が適切であれば最も価値の高い行動が最適な行動であり，greedy 法では最適な行動を選び続けるため，最適な政策となる．各行動の価値を推定する学習方法では，学習の収束時に各行動の価値が最適な値に収束するため，収束時には greedy 法が有効である．しかし，学習の途中において価値の大小が最適な場合と異なる場合，greedy 法は有効ではない．

２．４．３　ε -greedy 法

ε -greedy法は，過去の行動価値の中からεの確率でランダムな行動を選択し，1－εの確率で行動価値の最も高い行動を選択する行動選択の方法である（図2.4）．εの確率でランダムな行動を取ることで，現在最も高い行動価値となっている行動よりも更に良い行動があるかどうかを調べることができる．ε＝0 の場合，greedy法と同等になる．そのため，学習開始時は，εを適度な数字から開始し，学習が進むにつれてε→0と近づけることが多い．

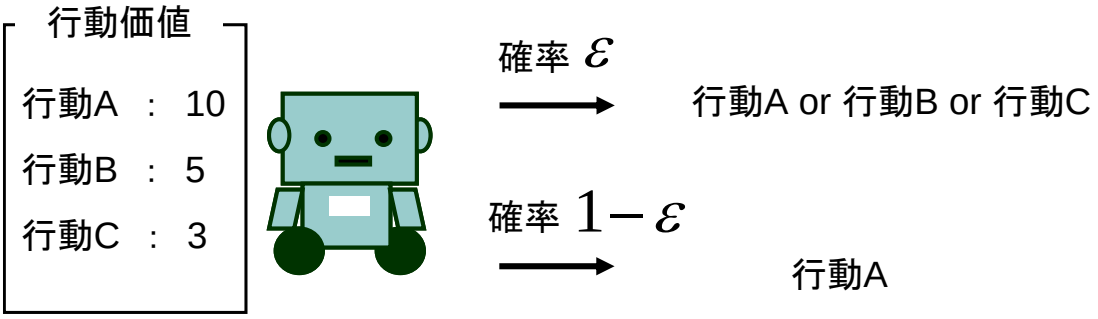

図 2.4　ε -greedy 法の概念図

２．４．４　softmax 法

softmax 法は，評価値の割合に応じて確率的に行動選択を行う方法である．行動を選択する確率を行動確率という．softmax 法では，行動価値によって行動確率を変化させる方法である．つまり，行動価値の高い行動には最も高い行動確率が与えられ，その他の行動は，行動価値の高い順に行動確率が与えられる．時間tにおいて状態sで行動aを選択する確率$\pi_t(s,a)$は（2.3）式で与えられる．τは温度と呼ばれる正定数である．

$$\pi_t(s,a) = \frac{e^{Q(s,a)/\tau}}{\sum_{b=1}^{n} e^{Q(s,b)/\tau}} \tag{2.3}$$

$\pi_t(s,a)$ ： 時間t，状態sで行動aを選択する確率

$Q_t(s,a)$ ： 時間t，状態sで行動aを行った場合の行動価値

τ ：温度

温度τが高い場合には，全ての行動がほぼ同程度に選択される．また，温度τが低い場合には行動価値の高低による選択確率の差が大きくなる．

以上が，現在一般的に使われている代表的な行動選択方法である．それぞれ選択方法に特徴があり，一概にどの選択方法が一番優れているかは決定できない．そのため，学習方法に合った選択方法を用いるのが一般的である．

第３章　複数タスク下での行動決定手法

３．１　提案手法概要

本論文では，複数のタスクを有する場合のロボットの行動学習，行動選択手法について提案する．本手法は，大きく分けて行動学習部と行動選択部に分かれる．提案手法全体の概要図を図 3.1 に示す．行動学習部では，各タスクの行動学習を行なう．ここで，各タスクの行動学習を行い学習空間に行動価値を蓄積する．その後，学習した行動価値を行動選択部へと渡し，行動選択部において各タスクの重要度を基にロボットの最終的な行動選択を状況に合わせて行なう．ロボットの行動学習と行動選択はロボットの行動毎に行う．

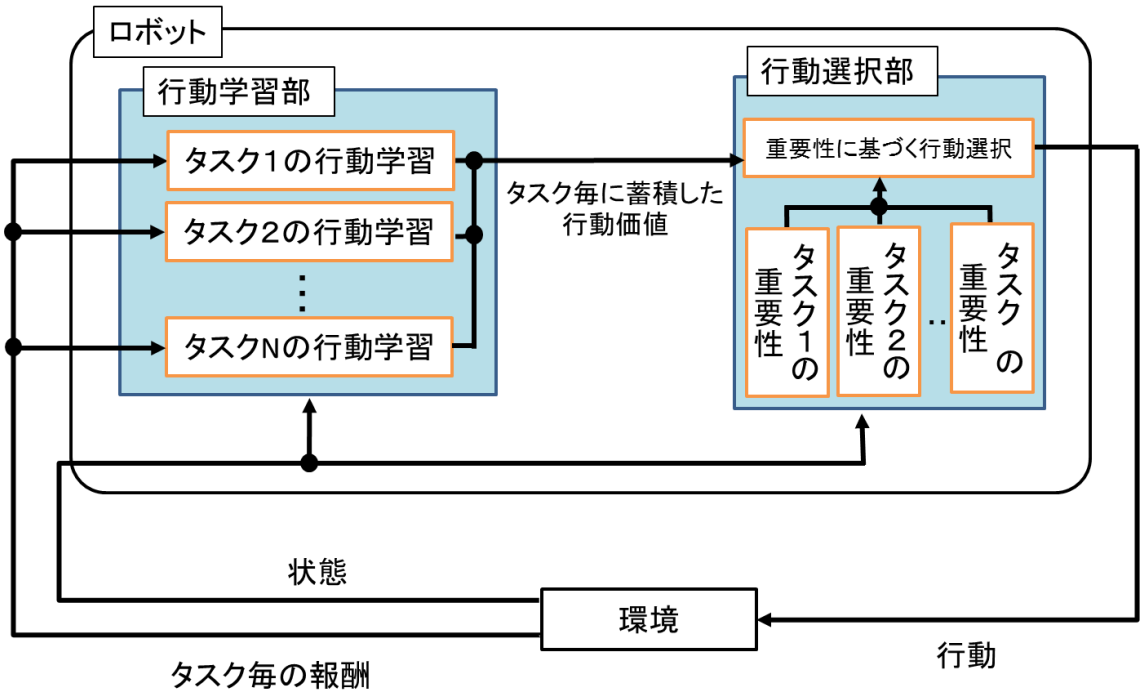

図 3.1　提案手法概念図

提案手法の流れ図は図 3.2 に示す．ロボットは環境に対して行動する．その行動に従って環境を認識した状態と正規化された報酬がロボットに入力される．正規化された報酬はロボットに与えられているタスク毎に与えられる．入力された状態と報酬は行動学習部でタスク毎に学習する．学習した結果として蓄積された行動価値は行動選択部へ渡される．行動選択部では，タスク毎の重要度を計算し，状態と各タスクの行動価値，重要度から最終的な行動を出力する．このサイクルを繰り返すことで．複数タスク下での行動学習と行動選択を行う．各部の詳しい説明は以下の節で示す．

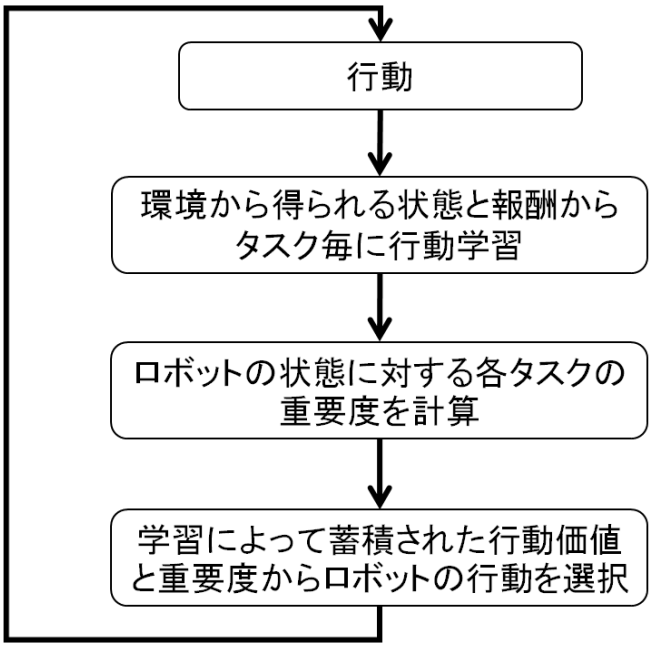

図 3.2 提案手法を適用した場合の流れ図

３．２　ロボットが持つタスクの種類

本論文では，ロボットが持つタスクを２種類のタスクに分けて定義する．ロボットには，自身の安全確保やエネルギーの保持など自分自身のためのタスクと人間から与えられる仕事としてのタスクの２種類が存在する．これら２種類のタスクについて，本論文では前者を生得的タスク，後者を後天的タスクと定義する．生得的タスクとはロボットは生まれながらにして持つタスクという意味であり，後天的タスクとはロボットが後から人間に与えられるタスクという意味である（図 3.3）．この２種類のタスクは，人間で例えれば一次的（基本的）欲求と二次的（派生的）欲求[16-17]に相当する（図 3.4）．一次的欲求とは，個体の生命や種の存続を維持するための身体的・生理的欲求のことである．ロボットにおいて個体の生命の存続を維持するということは，自らが活動停止とならないようにエネルギー残量を一定以上に保ったり，故障しないよう危険を回避したりすることになる．また二次的欲求とは，一次的欲求から派生した欲求である．生後，経験や学習を通じて新たに獲得されるものであり，金銭欲求・承認欲求・達成欲求などが挙げられる．ロボットにおける金銭欲求や達成欲求を個々に考えるのは難しいが，大まかには人間のために人間に与えられたタスクを達成することであると定義する．本論文において複数のタスクを持つロボットとは，これら２種類のタスクを，１つずつまたは複数持つロボットとする．

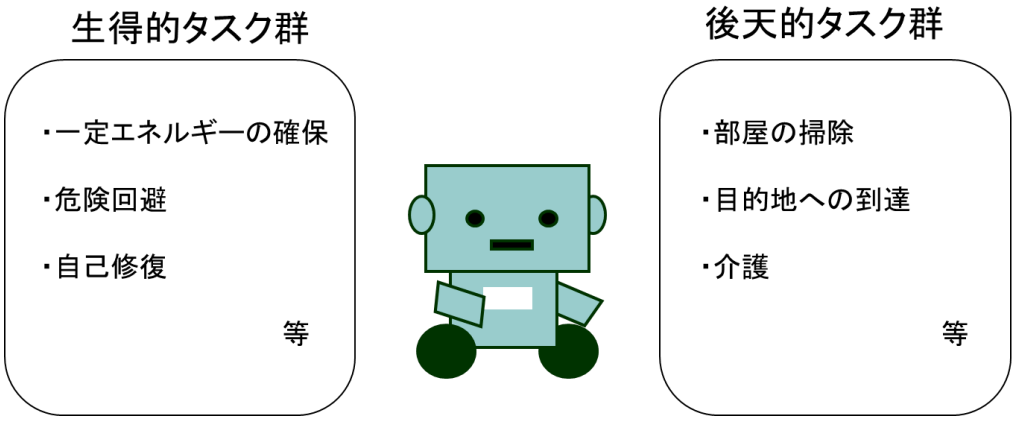

図 3.3　ロボットにおけるタスクの種類

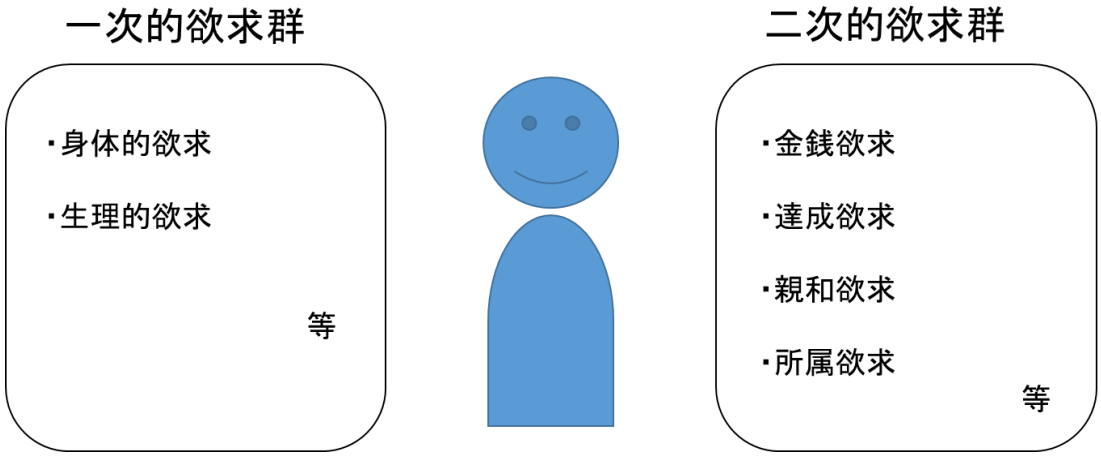

図 3.4　人間における欲求の種類

３．３　複数タスク下での行動学習手法

ロボットが与えられたタスクの達成に必要な行動を行なうためには，各タスクについて学習を行なう必要がある．ロボットに与えられるタスクが複数存在する場合には，当然複数のタスクに対する行動学習が必要となる．強化学習において１つのタスクを学習する場合には、１つの学習空間に，報酬関数から得られる行動価値を蓄積し学習を行なう．しかし，複数のタスクを１つの学習空間で学習するためには，複数のタスクを考慮した報酬関数を設計する必要がある．この方法では，新たにタスクを追加したり削除したりした場合には，報酬関数の再設計が必要となる．そこで，本手法では，複数のタスクに対して行動学習は別々に行なう．つまり，１つのタスクに対して１つの学習空間を用意し，それぞれ個別に学習を行なう．この方法では，新たにタスクが追加された場合であっても，個々の

タスクのみを考慮した報酬関数と学習空間を追加するだけで良い．また，不要となったタスクを削除する際にも，そのタスクに関する学習空間だけを削除すれば他のタスクの学習には影響しない．

提案する行動学習手法では，個々のタスクに対して個別に学習した後に行動選択を行なう．提案する行動学習手法の概要図は図 3.5 に示す．通常の強化学習では，報酬関数によって設定される報酬値に制限はないが，本手法では各タスクで設計される報酬関数が取り得る値の範囲を-1~1 の範囲に正規化する．従って，ロボットに対して負の報酬を与える場合には最低で”- 1”，正の報酬を与える場合には最高で” 1”となる．報酬関数によって得られる値を正規化することで，複数のタスク間での報酬のスケールを統一し，行動学習によって得られる行動価値による行動選択を可能にする．タスク毎に学習した行動価値は，行動選択部へ渡される．

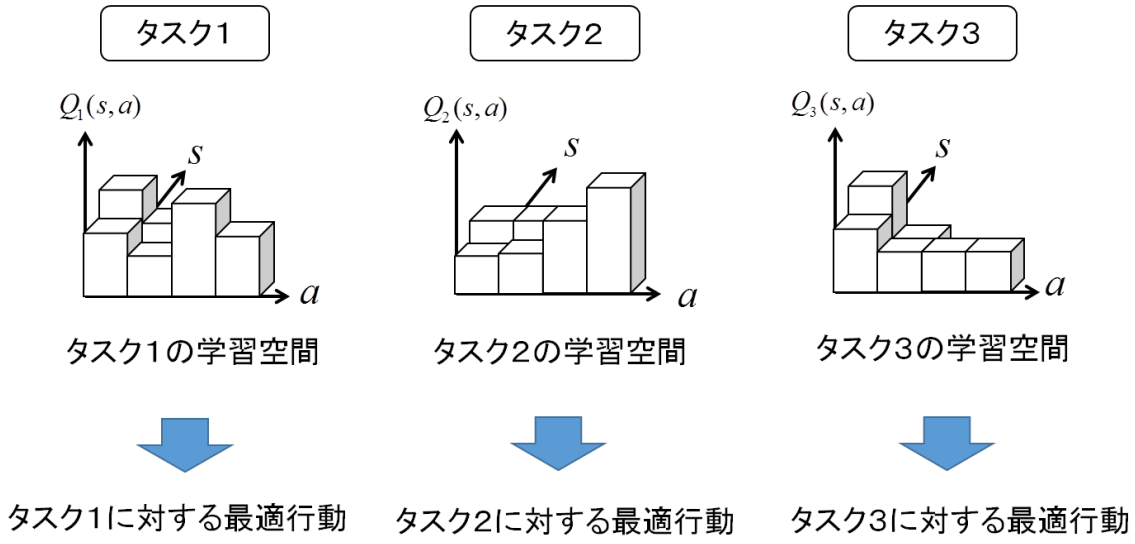

図 3.5　行動学習部の概念図

3．4　複数タスク下での行動選択手法

行動選択手法では，行動学習手法によって個別に学習し，蓄積した行動価値からロボットが最終的に行なう行動を決定する．ロボットの行動選択は，ロボットが直面する状態においてどの行動をすべきか決定するものである．通常の強化学習では学習空間が 1 つなので，ε-greedy 法や softmax 法のように取りうる行動の行動価値から行動選択を行う．しかし，本手法では，行動学習を個別に行っているためにある状態においてロボットが取りうる 1 つの行動に対して，タスク毎に行動価値が存在する．従って，従来のような行動選択手法を用いることはできない．そこで，本手法では，生得的タスク，後天的タスクそれぞれにおいてタスクの重要性を表す指標として重要度を定義し，この重要度を基に行動選択

を行う．重要度の算出とそれを基にした行動選択は，ロボットの一行動毎に行う．

３．４．１　重要度の定義

あるタスクがある状況でどの程度重要であるのかを表す指標として重要度を定義する．その範囲は-1~1 までとする．重要度“１”は，そのタスクの重要度が最も高い状態を表す．また，重要度 ”0” はそのタスクの重要性は無いことを表す．重要度が ”-1” となると，むしろそのタスクを達成しないほうが良い事を表す．従って重要度が負である場合には，ロボットはそのタスクを達成しないような行動を選択することが望ましい．

各タスクの重要度の変化に対してロボットが選択する行動がどのように変化するのか，タスクが２つの場合の例を基に説明する．タスク１の重要度，タスク２の重要度がともに正の値である場合，重要度が正であるということは２つのタスクの重要性がともに高いことを意味している．従って，ある状態においてロボットが選択できる行動群の中でタスク１，タスク２ともに達成できるような行動が選択されることになる．また，タスク１の重要度が正の値，タスク２の重要度が負の値である場合，タスク１の重要度が正の値であるということは，タスク１に対する重要性が高いことを意味しており，逆にタスク２の重要度が負であるということは，タスク２を達成しないような行動に対して重要性が高いということを意味している．最後にタスク１の重要度，タスク２の重要度がともに負の値である場合，それぞれのタスクを達成から最も離れた行動をロボットは選択する．

３．４．２　生得的タスクの重要度

重要度は，タスク毎に設定する．設定方法は，生得的タスクと後天的タスクで異なる．生得的タスクの場合には，ロボットが予め持っている指標によって重要性が決定されると考える．これは生得的タスクの重要性が周りの環境に依存するものではなく，ロボット内部の状態から決定されることを意味する．例えば，ロボットのバッテリー残量を一定以上に保つという生得的タスクをロボットが持っていた場合，このタスクの重要度は周りの環境からではなく，ロボットのバッテリー残量によって変化する．バッテリー残量が少なければ重要度は高くなり，バッテリー残量が十分あれば重要度は低くなる．従って，生得的タスクの重要度に関しては，ロボット内に重要度を計算するための指標を予め設計する．

生得的タスクにおける重要度が，ある関数$f(x)$によって決定されるとする．このある関数$f(x)$は，生得的タスクがエネルギー残量の確保であったり，危険回避であったりそのタスクに依存した形で設計する．この時，生得的タスクの重要度をp_1とすると，行動毎に更新される重要度p_1'は（3.1）式によって更新される．（3.1）式では，ある時点での重要度だけでなく過去の重要度の推移も考慮するために過去の重要度に対して重み θ を用いた加重平均を適用している．

$$p_1' \leftarrow p_1 + \theta\{f(x) - p_1\} \quad (3.1)$$

３．４．３　後天的タスクの重要度

後天的タスクの場合には，そのタスクの重要度を事前に設定することはできない．なぜなら後天的タスクは，生得的タスクとは反対に周りの環境によって重要度が変化するためである．後天的タスクは，人間に与えられるタスクであるので，そのタスクの重要性も人間が与えるものであると考える．そこで，本手法では人間がロボットに対して－１~１の範囲で値を与え，その値を基に重要度を算出する．この報酬は，学習に用いるものとは関係ないものであり，ロボットの行動に対して与える．イメージとしては，ロボットの行動に対して人間が褒めたり，叱ったりするようなものである．これを数値として与えることとする．また，この値が人間から与えられる頻度が高いほどそのタスクに対して人間が注目しているということであり，頻度が高いほど重要度が高くなるよう設計する．この－１~１の範囲の値によってロボットは自分に与えられている後天的タスクの重要度を計算し，行動選択を行う．

後天的タスクにおける重要度は，生得的タスク同様関数$f(x)$によって決定する．生得的タスクの場合には，そのタスク毎に関数を設計するが，後天的タスクの場合には人間から与えられる数値とその頻度を基に設計する．人間から与えられる数値をr_hとすると，関数$f(r_h)$は（3.2）式で算出する．（3.2）式では，重要度の急激な変化を抑えるためにシグモイド関数を適用している．

$$f(r_h) = \frac{-2}{1+e^{(\delta * r_h + \sigma)}} + 1 \quad (3.2)$$

δ,σ：定数

また，後天的タスクの重要度をp_2とすると，更新される重要度p_2'は生得的タスク同様（3.3）式によって更新される．（3.3）式では，ある時点での重要度だけでなく過去の重要度の推移も考慮するために過去の重要度に対して重みμを用いた加重平均を適用している．　この式によって，ロボットが報酬関数によって報酬が得られた場合のみ値を更新することで，人間に数値が与えられた頻度によって重要度が変化することになる．

$$p_2' \leftarrow p_2 + \mu\{f(r_h) - p_2\} \quad (3.3)$$

３．４．４　重要度に基づく行動選択

ロボットは各タスク別々に行動学習を行なう．従って，ある状態において取ることのできる行動には，タスク毎に行動価値が存在する．この各タスクの行動価値を軸とした空間を構成する．この空間をタスク間行動価値空間と呼ぶ．タスク間行動価値空間では，軸となる各タスクの行動価値の交点が１つの行動を表し，ロボットが直面する状態において取ることのできる行動の数だけ空間上に行動を表す点がプロットされる．ロボットの行動学習に用いられる報酬関数の値の範囲は-1~1 であるので，各行動に対する行動価値の範囲も同様に-1~1 となる．次に，タスク間行動価値空間に算出した重要度をプロットする．重要度は各タスクで算出されるので，各タスクでの重要度の交点が，ロボットが直面する状態において各行動の重要度を表す点となる．次に，プロットした重要度の点と各行動との間のユークリッド距離を計算する．計算したユークリッド距離が最も小さい行動をロボットが選択する行動として決定する．

次に，重要度を基にした行動選択について，ロボットに２つのタスクを同時に与えた場合を例に説明する．例としては，２タスクの場合を挙げているが，本手法はタスクの数が増えた場合にも，構成する空間の次元を拡張することで，対応可能である．

ここでは，ロボットに対し，タスク１とタスク２の２つのタスクが与えられているものとする．この時，ロボットはそれぞれのタスクについて学習し行動価値を蓄積する．ロボットがある状態で取ることができる行動をa_0~a_4とし，直面する状態Sにおけるタスク１の行動価値を$Q_1(S, a_i)$，タスク２の行動価値を$Q_2(S, a_i)$とする．この時，各タスクの行動価値を軸とするタスク間行動価値空間を構成する．構成された空間には，各行動に対する行動価値が交わる点が存在する．この点はロボットがある状態で取ることができる行動を意味し，その行動の数だけ存在する．また，本手法では行動学習においてタスクを学習するための報酬関数の取りうる値の範囲を-1~1 に正規化しているので，この空間における各軸（行動価値）の範囲も同様に-1~1 の範囲となる．構成した空間の例を図 3.6 に示す．例では２タスクの場合であるので２次元の空間となるが，３タスク，４タスクとなれば空間の次元も３次元，４次元と増えていく．

行動	タスク1の行動価値 $Q_1(S,a_i)$	タスク2の行動価値 $Q_2(S,a_i)$
a_0	$Q_1(S,a_0)$	$Q_2(S,a_0)$
a_1	$Q_1(S,a_1)$	$Q_2(S,a_1)$
a_2	$Q_1(S,a_2)$	$Q_2(S,a_2)$
a_3	$Q_1(S,a_3)$	$Q_2(S,a_3)$
a_4	$Q_1(S,a_4)$	$Q_2(S,a_4)$

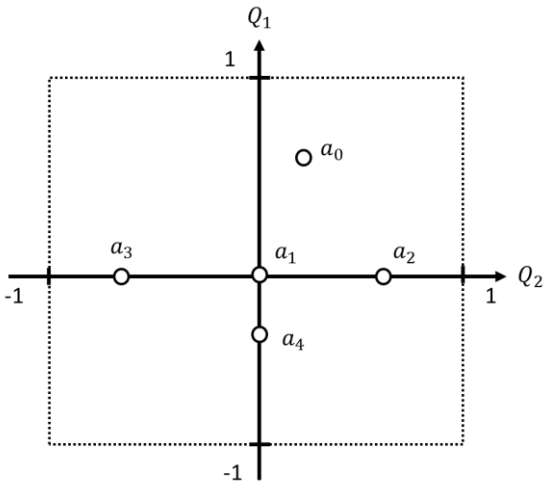

図 3.6　２タスクの場合のタスク間行動価値空間構成の例

次に，構成した空間上に重要度をプロットする．重要度はタスク毎に存在するのでこの各重要度の交点をP'とする．重要度の取りうる範囲は，行動価値と同様-1~1 の範囲であるので，空間上にプロットすることが可能である．タスク１の重要度をp_1，タスク２の重要度をp_2とすると，空間上にそれぞれの重要度の交点として点P'がプロットされる．

タスク１の重要度が“１”，タスク２の重要度も”１”である場合に空間上に点P'をプロットしたものを図 3.7 に示す．

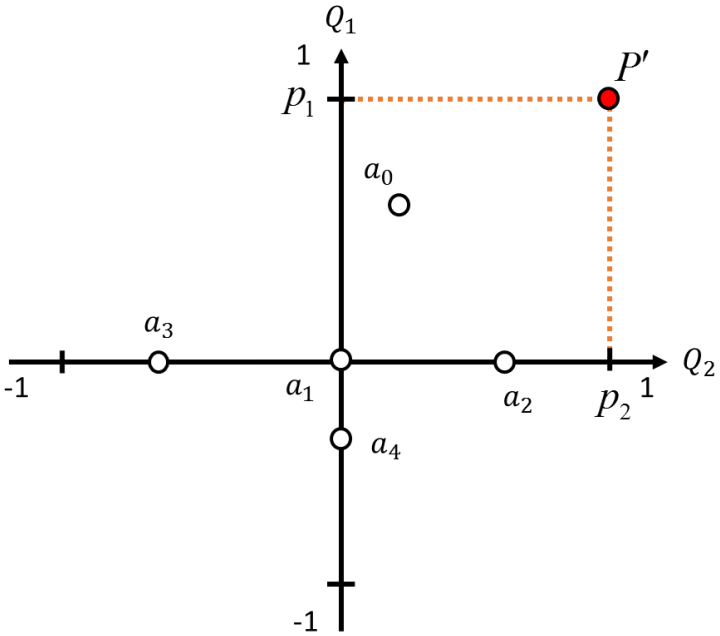

図 3.7　２タスクの場合の重要度点プロット

次に，空間上にプロットした重要度点と，ある状態においてロボットが取ることのできる行動との間のユークリッド距離を計算する（図 3.8）．重要度の交点を$P'(p_1, p_2)$とし，重要度の交点と行動との距離を距離d_i とすると，d_i は（3.4）式によって求める．

$$d_i = \sqrt{\left(p_1 - Q_1(S, a_i)\right)^2 + \left(p_2 - Q_2(S, a_i)\right)^2} \tag{3.4}$$

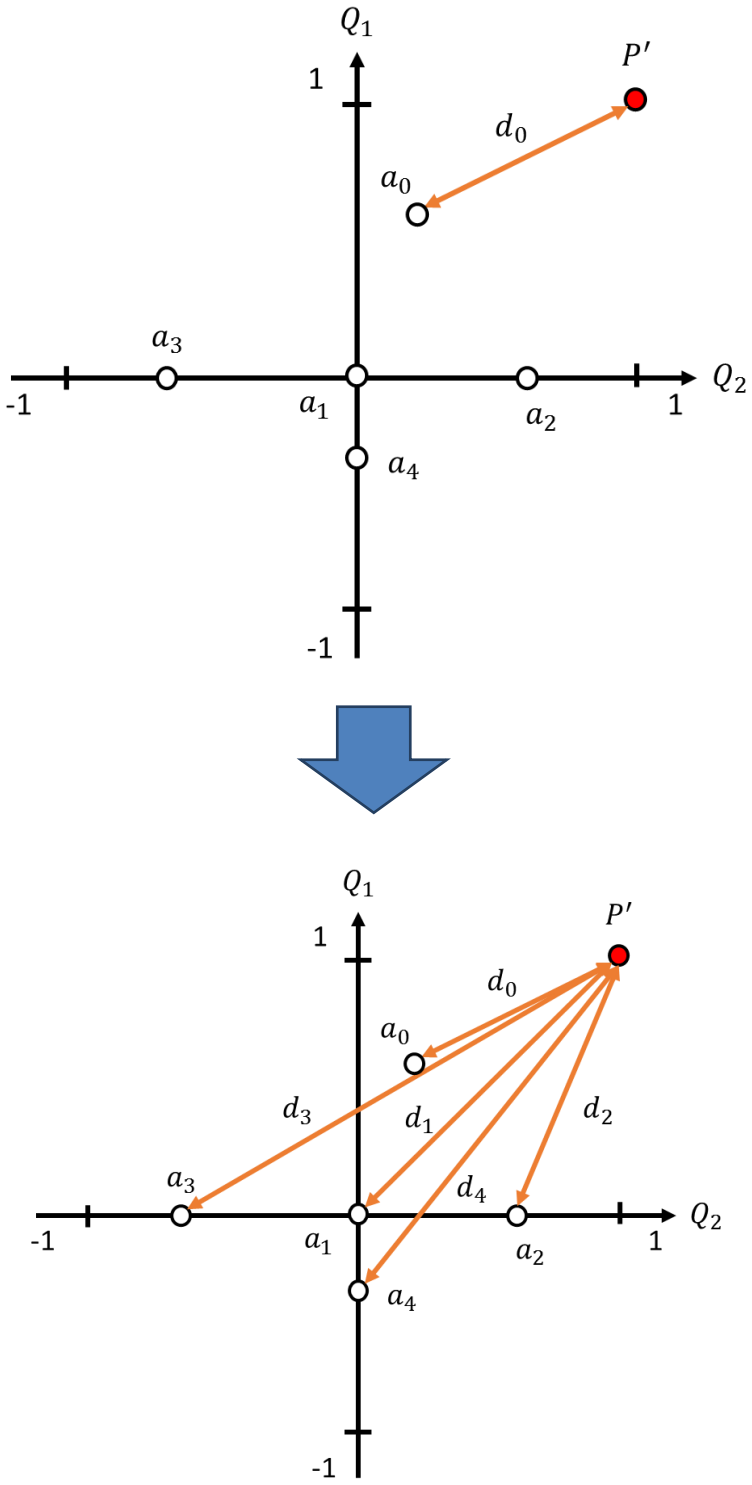

図 3.8　２タスクの場合の各重要度の交点と各行動とのユークリッド距離の算出

上記の空間と計算式は，２タスクでの例であるので２次元空間でのユークリッド距離の計算式である．これが，ｎ個のタスクへ増えた場合には，ｎ次元の空間を構成し，ｎ次元でのユークリッド距離を計算することになる．ｎ次元での重要度の交点と行動とのユークリッド距離の計算式は，(3.5）式となる．

$$d_i = \sqrt{\sum_{k=1}^{n} \{p_k - Q_k(S, a_i)\}^2} \tag{3.5}$$

この（3.5）式によって算出されるユークリッド距離の最も短い行動をロボットの最終的な行動に決定する．ただし，行動価値と重要度からの行動選択のみでロボットが行動すると，局所解に陥る可能性がある．通常の強化学習では，ε-greedy 法のように一定確率でランダムな行動を取ることによって局所解に陥ることを防いでいる．そこで，本手法でもεの確率でロボットはランダムな行動を取り，1-εの確率でユークリッド距離が最も短い行動を選択することとする．これによって，本手法でもロボットの行動が局所解に陥ることを防ぐ．

第４章　２タスクを有するエージェントを用いたシミュレーション実験

４．１　実験目的

提案した複数目的下での行動決定手法において，重要度による行動選択が可能であるか検証するためコンピュータ上に仮想的なロボット（これ以降エージェントとする）を設定し，シミュレーション実験を行う．エージェントには，２つのタスクを同時に与え，各タスクの重要度を変化させることで，エージェントが重要度に従って行動可能であることを検証し，提案手法の有用性を示す．

４．２　実験概要

本手法を適用したエージェント一体を仮想的な実験空間へ配置しシミュレーション実験を行なう．エージェントは，エネルギーを持っており，エネルギーを消費して行動することができる．このエネルギーはバッテリーのようなものである．このエージェントに対して，生得的タスクと後天的タスクをそれぞれ１つずつ同時に与え，エージェントが状況に応じた行動選択が可能であるかどうか検証する．

提案手法を適用したロボットに対して生得的タスクと，後天的タスクをそれぞれ１つずつ計２つのタスクを同時に与えた場合のロボットの行動学習についてシミュレーション実験を通して検証する．実験はシミュレーションによって行なうのでこれ以降ロボットのことをエージェントと呼ぶ．エージェントは，一定のエネルギーを保有しており，このエネルギーを消費することで行動することができる．このエージェントに対して，生得的タスクとしてエネルギー量の確保を目的とするタスクを与え，後天的タスクとしてゴール位置への到達を目的とするタスクを与える．

４．３　実験設定

実験環境は，図 4.1 のような 5×5 マスのグリッド状の環境を用いる．環境内に壁や障害物はない静的環境である．また，環境内にはスタート位置，ゴール位置，エネルギー充電ポイントが存在する．スタート位置はエージェントの初期位置であり，５×５マスの環境内では左下の位置となる．ゴール位置はエージェントに与える後天的タスクのゴールとな

る位置で，環境内では右下の位置となる．最後にエネルギー充電ポイントとは，エージェントがエネルギーを回復することができる唯一の位置であり，環境内では左上の位置となる．この実験環境の中で，エージェントは，自身がいるマスの位置 S ，自身のエネルギー残量を認識することができる．エージェントが持つエネルギーの総量は E とし，0~100 まで，1 刻みで変化する．エージェントは，環境内のマスの位置 S と，エネルギーの総量 E のから成る状態空間についてタスク毎に学習を行う．

ある状態においてエージェントが選択可能な行動は，図 4.2 に示すように前後左右に 1 マス移動するかその場で静止するかの 5 つとする．エージェントが行動した結果壁にぶつかる場合には，その場に静止する．

また，エージェントは 1 行動ごとに一定のエネルギーを消費する．ただし，エージェントが取る行動がその場に静止するのか，前後左右に移動するのかで消費するエネルギー量は異なる．エージェントがあるマスで静止するときの消費エネルギーを ΔE_s とし，前後左右に移動した場合の消費エネルギーを ΔE_m とする．このとき，エージェントがあるマスで静止するときの消費エネルギー ΔE_s は，エージェントが前後左右に移動するときに消費するエネルギー ΔE_{m} よりも小さくなる．1 行動あたりの消費エネルギーは，1 行動あたりのエネルギー変位量として表現できる．従って，エージェントは 1 行動ごとにエネルギー変化量 ΔE を得ることになる．ここでは，エージェントが前後左右に 1 マス移動したときのエネルギー変化量を，$-\Delta E_m$ とし，そのマスで静止した場合のエネルギー変化量を $-\Delta E_s$ とする（表 4.1）．このとき，その場に静止する行動の方が，前後左右に移動する行動よりも消費エネルギーは小さいので，エネルギー変位に変換すると， $\Delta E_m > \Delta E_s$ となる．

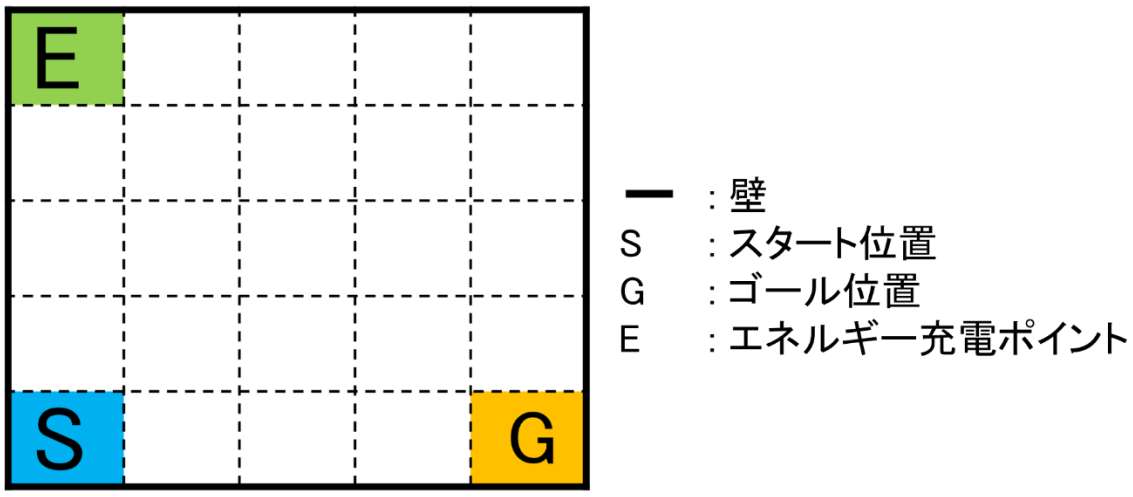

図 4.1　実験環境

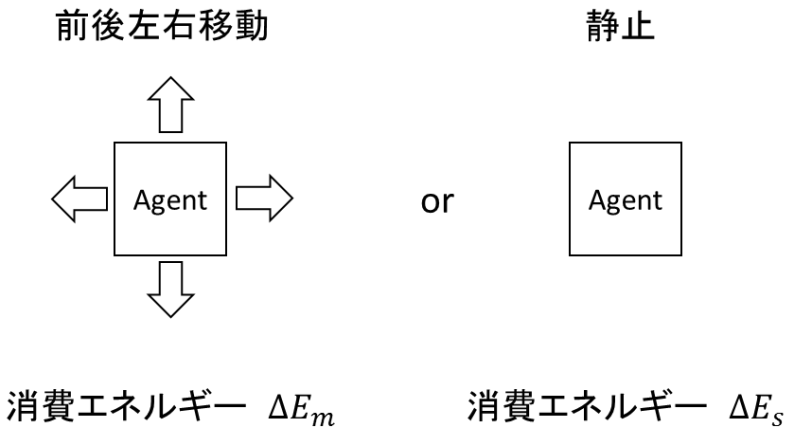

図 4.2 エージェントの選択可能な行動と消費エネルギー

表 4.1　エージェントの行動とエネルギー変位の対応

エージェントの行動	エネルギー変位
前後左右	$-\Delta E_m$
静止	$-\Delta E_s$

次に, 図 4.1 の実験環境に示すようにエージェントがエネルギーを獲得できる充電ポイントについて説明する．今回の実験設定では，左上をエネルギー充電ポイントとしている．エージェントがエネルギー充電ポイントに移動した場合，エージェントはエネルギーを充電することができる．従って，エネルギー充電ポイントにおいては，エージェントはエネルギー変化量$+\Delta E_c$を獲得できることになる．各行動でのエネルギー変位量やエネルギー充電ポイントでのエネルギー変位量の具体的な値は実験パラメータとして後述する．

上述した様な実験環境下においてエージェントには，2 つのタスクを与える．エージェントに与えるタスクは，生得的タスクと後天的タスクそれぞれ 1 つずつとする．それぞれのタスクの詳しい説明は次節で説明する．

４．３．１　生得的タスクの設定

今回のシミュレーション実験でエージェントに与える生得的タスクは，エネルギー獲得を目的とする．これを今回の実験環境に合わせて考えれば，１回の行動で獲得するエネルギー変位量を最大化することになる．このタスクにおいて，エージェントはエネルギー変位量が最も大きくなるような行動を学習する．

まずは，この生得的タスクに対する報酬関数の設計について説明する．生得的タスクに対する報酬をr_1とし，エージェントの一度の行動で変位するエネルギー量をΔEとすると報酬関数は（4.1）式で設定する．行動学習に用いる報酬関数は-1~1 の範囲に正規化するので，この時ΔE の最大値は 1 となる．また，前節で説明したようにエージェントの行動に対して消費するエネルギー量についても報酬関数の値が-1~1 の範囲となるように設計する．実際に消費するエネルギー量が-1 以下であったり，充電ポイントにおいて充電されるエネルギー量が+1 以上となる場合においても，報酬関数によって得られる報酬値は-1~1 の範囲を超えないよう調整する．

$$r_1 = \Delta E \tag{4.1}$$

次に，生得的タスクの重要度の設計について説明する．生得的タスクの重要度は，人間の食欲を基に事前に設計する．人間の場合には，お腹が減れば減るほど食欲が大きくなり，お腹が満たされれば，食欲は消滅する．また，満腹状態に近い場合には，これ以上食欲は発生せず逆に食欲以外のことが優先される．これを基に生得的タスクにおける重要度算出関数$f(E)$は（4.2）式で求める．（4.2）式においてEは，現在のエージェントのエネルギー残量を表している．この式では，エネルギー残量が低いほど重要度は高くなり，エネルギー残量が増えるにつれて重要度は下がる．また，エネルギー残量が満タンに近づけば逆にエネルギーを消費する傾向になる．また，δ と σ は定数であり，この値を変化させることでエネルギー獲得に積極的か，消極的というエージェントの傾向を変化させることができる．

$$f(E) = \frac{-2}{1+e^{(\delta * E+\sigma)}} + 1 \tag{4.2}$$

δ , σ：定数，E：エネルギー残量

今回の実験では，このδ と σ を３つのパターン用意しそれぞれ実験する．１つ目のパターンは$\delta = -0.2, \sigma = 10$の場合とする．このときの，重要度算出関数$f(E)$は図 4.3 のようにな

る．図 4.3 からわかるように，このパターンではエネルギー残量が 50 となったとき，重要度が 0 となる．従って，エネルギー残量が調度半分となったときに，重要度も±0 となるような標準的な関数になっている．

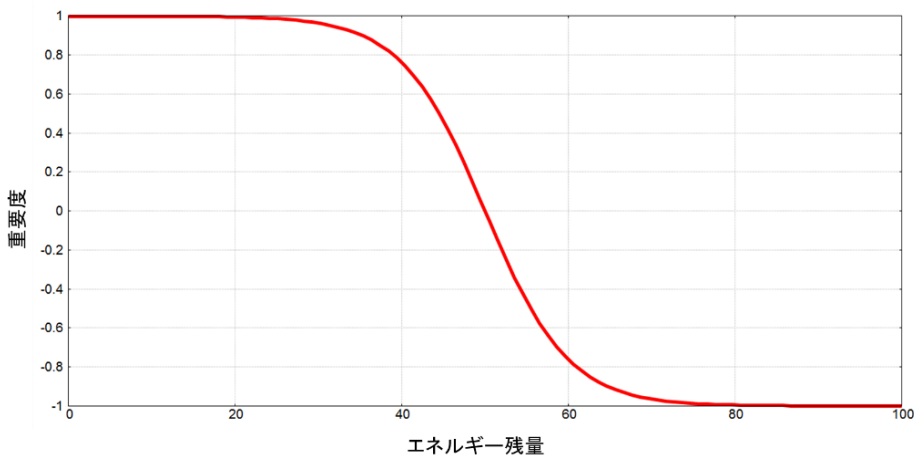

図 4.3　エネルギー残量と重要度の関係（$\delta = -0.2, \sigma = 10$）

２つ目のパターンは$\delta = -0.13, \sigma = 10$の場合とする．このときの，重要度算出関数$f(E)$は図 4.4 のようになる．このパターンではエネルギー残量 70 付近で重要度が 0 となり，それ以下のエネルギー残量となると急速に重要度が上昇する．従って，エージェントはエネルギー確保により積極的な傾向を持つことになる．

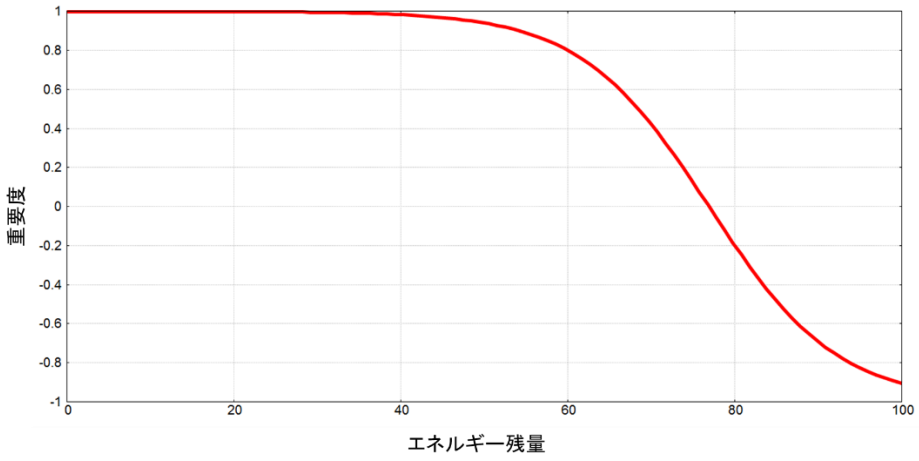

図 4.4　エネルギー残量と重要度の関係（$\delta = -0.13, \sigma = 10$）

3つ目のパターンは，$\delta = -0.3, \sigma = 10$とする．このときの，重要度算出関数$f(E)$は図 4.5 のようになる．このパターンではエネルギー残量 35 付近で重要度が 0 となり，それ以下のエネルギー残量となると急速に重要度が上昇する．従って，エージェントはエネルギー確保により消極的な傾向を持つことになる．

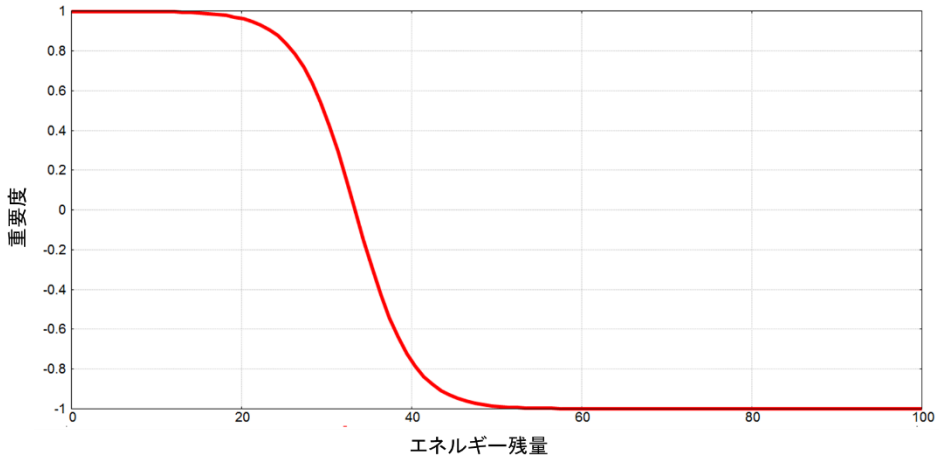

図 4.5　エネルギー残量と重要度の関係（$\delta = -0.3, \sigma = 10$）

上記のように設定した，重要度算出関数$f(E)$を基に生得的タスクにおける重要度p_1を，エージェントの行動毎に（4.3）式で更新する．　この式については，提案手法において説明しているのでここでは詳しい説明は行わない．

$$p_1' \leftarrow p_1 + \theta\{f(E) - p_1\} \tag{4.3}$$

４．３．２　後天的タスクの設定

今回のシミュレーション実験において，エージェントに与える後天的タスクは，ゴール位置への到達を目的とする．このタスクにおいてエージェントは，ゴールへ到達するための経路を学習する．

まずは，後天的タスクに対する報酬関数の設計について説明する．今回エージェントに対して与える後天的タスクは，ゴールへの到達であるので，報酬はエージェントがゴール位置に到達したときのみ与える遅延報酬とする．エージェントが後天的タスクにおいて獲得する報酬をr_2としたとき，報酬は（4.4）式で与えられる．

$$r_2 = \begin{cases} 1 & (\text{ゴールへ到達した時}) \\ 0 & (\text{それ以外}) \end{cases} \tag{4.4}$$

次に，後天的タスクにおける重要度の設定について説明する．第３章の提案手法でも説明したように，後天的タスクの重要度は人間がエージェントに対して与得るによって数値によって算出する．この時の，数値を与える頻度とその値の大きさを変化させることで，重要度を変化させることが可能である．今回のシミュレーション実験では，人間が数値を与える頻度とその値の大きさについて数パターン用意し実験を行う．まず，頻度については基本的にはエージェントがゴール位置に到達した場合に毎回一定の数値を与える．これは，エージェントがゴールするたびに人間がエージェントを褒める動作を表しているといえる．

もう一つのパターンとして，エージェントがゴールした場合２分の１の確率で数値を与える．こちらは，エージェントがゴールする毎に与える場合に比べ，タスクに対する人間の関心が低いことを表している．頻度についてはこの２パターンで実験を行なう．また，与える値は”＋１”，”0.2”,”－1”の３パターンで設定する．これらそれぞれのパターンで実験を行い算出される重要度とエージェントの行動について検証する．

人間に与えられた数値は，第３章の提案手法において説明したように，関数$f(r_h)$を用いて（4.5）式で算出する．（3.2）式におけるr_hの与え方が，今回用意した３パターンに変化することになる．

$$f(r_h) = \frac{-2}{1+e^{(\delta * r_h + \sigma)}} + 1 \tag{4.5}$$

δ,σ：定数

また，後天的タスクの重要度をp_2とすると，更新される重要度p_2'は（4.6）式によって更新される．この式については，第３章において説明してあるので詳しい説明は省略する．

$$p_2' \leftarrow p_2 + \mu\{f(r_h) - p_2\} \tag{4.6}$$

４．３．３　検証する重要度の組み合わせ

今回の実験では，エージェントに与えるタスクの重要度を変化させることで，重要度に合わせた行動選択が可能かどうかを検証する．ここでは，シミュレーション実験で検証する生得的タスクにおける重要度算出パラメータと，後天的タスクに対して人間が与える－１~１の範囲の数値の組み合わせについて説明する．

生得的タスクの重要度については，実験設定で記述したように３パターン用意し検証を行う．この３パターンはそれぞれ，エネルギー残量が最大の半分が重要度０になり，そこからエネルギー残量が減れば重要度が＋１に近づき，逆にエネルギー残量が植えれば重要度が－１に近づく標準的な設定が１パターン．標準的な設定よりもエネルギー残量の保持に結局的なパターン，エネルギー残量の保持に消極的なパターンの３パターンである．

後天的タスクの重要度は，エージェントの行動中に人間が与える数値を基に決定される．従って，人間が値を与える頻度やその値の大きさを変化させることで，タスクの重要性を変化させることができる．人間が与える値については，＋１，＋0.2，－1 の３パターン設定する．また人間が数値を与える頻度についてはエージェントがゴール位置に到達した場合に毎回与えるパターンと，ゴール到達時に２分の１の確率で与える２パターン設定する．

上記の生得的タスクにおけるパラメータのパターンと人間が与える数値のパターンから以下のような組み合わせで検証実験を行う．

実験パターン１：　人間はエージェントがゴールするたびに＋１を与える．
生得的タスクのパラメータは$\delta = -0.2\ \sigma = 10$

実験パターン２：　人間はエージェントがゴールするたびに＋１を与える．
生得的タスクのパラメータは$\delta = -0.13\ \sigma = 10$

実験パターン３：　人間はエージェントがゴールするたびに＋1 を与える．
生得的タスクのパラメータは$\delta = -0.3\ \sigma = 10$

実験パターン４：　人間はエージェントがゴールするたびに＋0.2 を与える．
生得的タスクのパラメータは$\delta = -0.2\ \sigma = 10$

実験パターン５：　人間はエージェントがゴールするたびに＋0.2 を与える．
生得的タスクのパラメータは$\delta = -0.13\ \sigma = 10$

実験パターン６：　人間はエージェントがゴールするたびに＋0.2 を与える．
生得的タスクのパラメータは$\delta = -0.3\ \sigma = 10$

実験パターン７：　人間はエージェントがゴールするたびに－１を与える．
生得的タスクのパラメータは$\delta = -0.2\ \sigma = 10$

実験パターン８：　人間はエージェントがゴールするたびに－１を与える．
生得的タスクのパラメータは$\delta = -0.13\ \sigma = 10$

実験パターン９：　人間はエージェントがゴールするたびに－1 を与える．
生得的タスクのパラメータは$\delta = -0.3\ \sigma = 10$

実験パターン１０：　生得的タスクのパラメータは$\delta = -0.13\ \sigma = 10$
人間から与える数値は行動回数 10000 回毎に変化させる．
行動回数 10000 回まではゴールする毎に＋１を与え，
行動回数 10001~20000 回までは，2 分の 1 の確率で＋１を与える．
さらに，行動回数 20001~30000 回では，再びゴールする毎に＋１を与える．

実験パターン１１：　生得的タスクのパラメータは$\delta = -0.13\ \sigma = 10$
人間から与える数値は行動回数 10000 回毎に変化させる．
行動回数 10000 回まではゴールする毎に＋１を与え，
行動回数 10001~20000 回までは，2 分の 1 の確率で＋１を与える．
さらに，行動回数 20001~30000 回では，再びゴールする毎に＋0.2 を与える．

実験パターン１２：　生得的タスクのパラメータは$\delta = -0.13\ \sigma = 10$
人間から与える数値は行動回数 10000 回毎に変化させる．
行動回数 10000 回まではゴールする毎に＋１を与え，
行動回数 10001~20000 回までは，2 分の 1 の確率で＋１を与える．
さらに，行動回数 20001~30000 回では，再びゴールする毎に－１を与える．

４．３．４　実験パラメータ

今回の実験において使用する各種パラメータや細かい設定を説明する．実験パラメータは表 4.2 にまとめる．

表 4.2　実験パラメータ

行動回数	30000 回
行動学習手法	Q 学習
初期行動価値	0.0
ステップサイズパラメータ α	0.1
割引率 γ	0.9
ランダム行動の確率ε	0.05
エージェントの初期エネルギー残量	100
最大エネルギー残量	100
最低エネルギー残量	0
充電ポイントでのエネルギー変位	+3.0
移動時のエネルギー変位	−0.25
静止時のエネルギー変位	−0.05
θ	0.5
μ	0.5
δ（３パターン）	−0.13,−0.2, −0.3
σ	10
δ'	5
σ'	0

今回の実験では，エージェントの行動回数を 30000 回とする．エージェントがゴール位置に到達した場合には，スタート位置に戻る．また，エージェントのエネルギー残量が 0 となった場合には，エネルギーを満タン状態まで回復し，スタート位置に戻る．

提案手法のタスク毎の学習には，ロボットの学習としてよく用いられる Q 学習を適応する．

ある時刻tでのエージェントの状態を状態S_tとし，状態S_tにおいてエージェントが行動a_tを取る．このときエージェントの状態が，状態S_{t+1}に遷移した場合の行動価値を$Q(s_t, a_t)$とすると更新式は（4.7）式のようになる．初期行動価値は 0.0 とする．

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right] \tag{4.7}$$

r_{t+1}：新たに獲得した報酬

α：学習率（$0 < \alpha < 1$）

γ：割引率（$0 < \gamma < 1$）

４．４　実験結果

以下にシミュレーション実験の実験結果を示す.

４．４．１　後天的タスクに対して人間が＋１を与えた場合の実験結果

はじめに，生得的タスクの重要度算出に関するパラメータを変化させ，後天的タスクの重要度についてはゴール位置到達毎に“＋１”を与えた場合の実験結果を示す．実験パターンとしては，1 から 3 に相当する．パラメータ設定が$\delta = -0.2, \sigma = 10$である場合のエージェントの行動回数に対するボール位置到達回数の推移を図 4.7 に，行動回数に対するエネルギー残量の推移を図 4.8 に示す．また，行動回数に対するゴール位置到達回数とエネルギー残量の推移を重ねあわせたものを図 4.9 に, 行動回数に対する各重要度の変化を図 4.10 に示す.

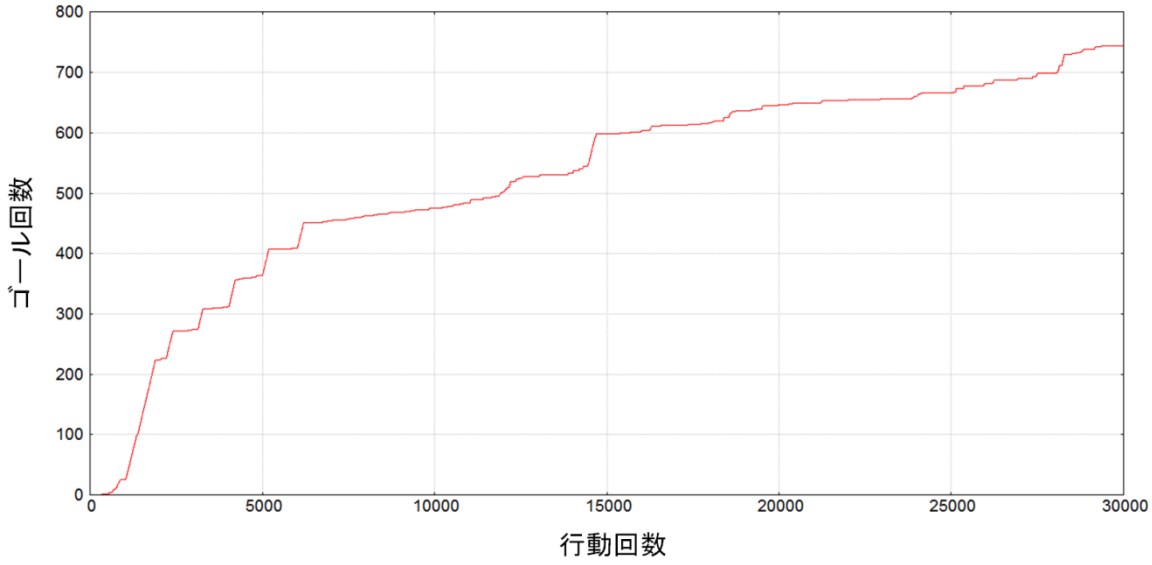

図 4.7　エージェントの行動回数に対するゴール位置到達回数の推移

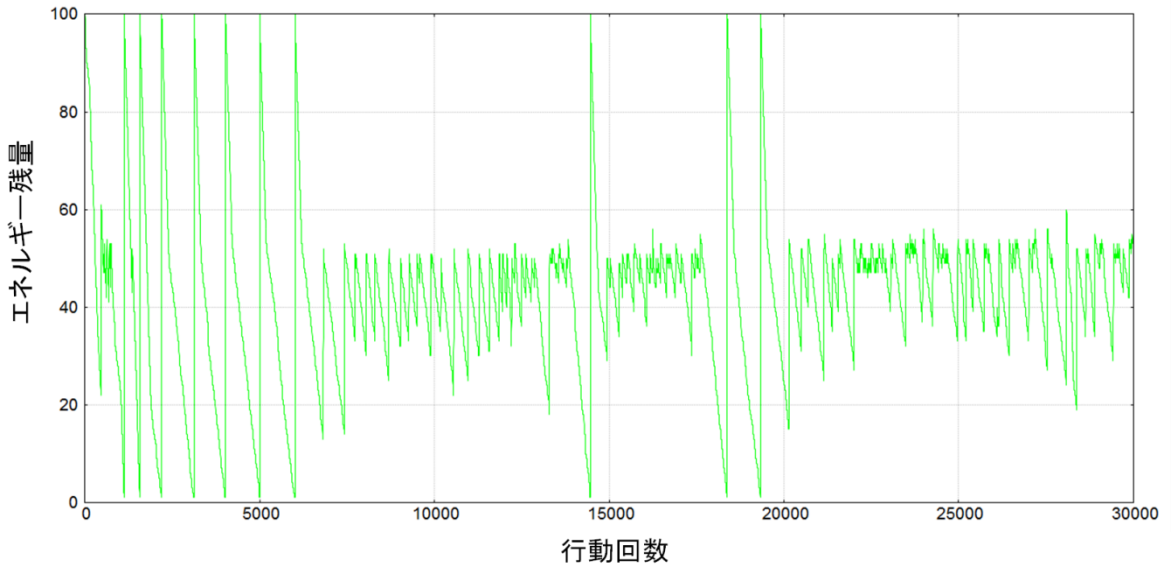

図 4.8　エージェントの行動回数に対するエネルギー残量の推移

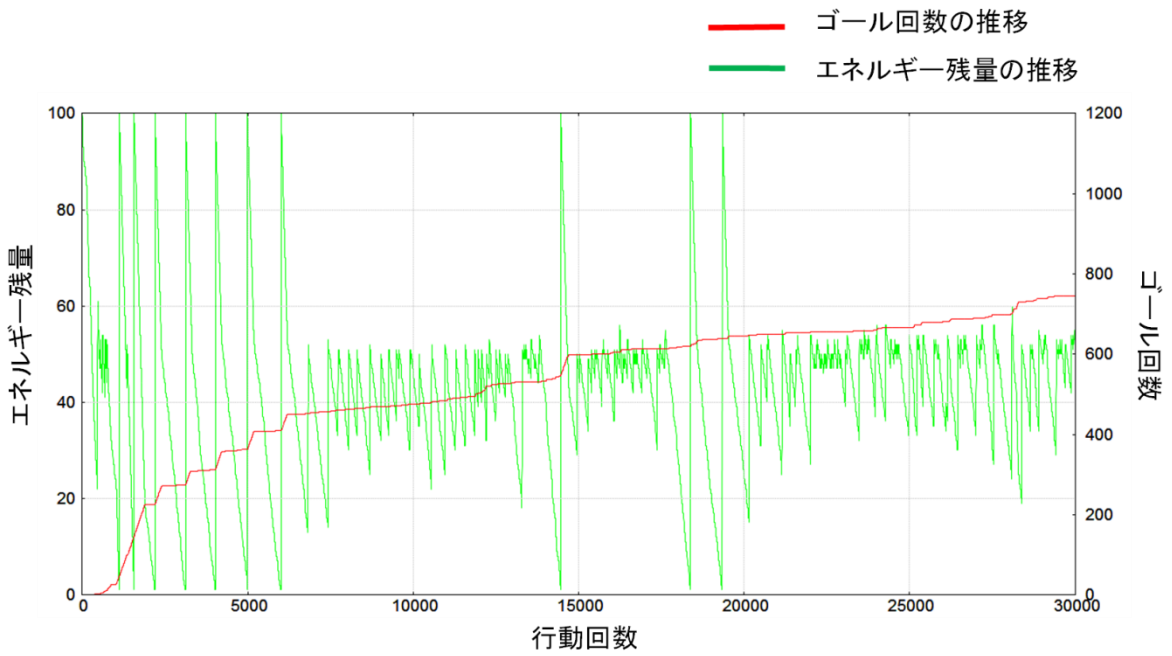

図 4.9　行動回数に対するゴール到達回数とエネルギー残量の推移

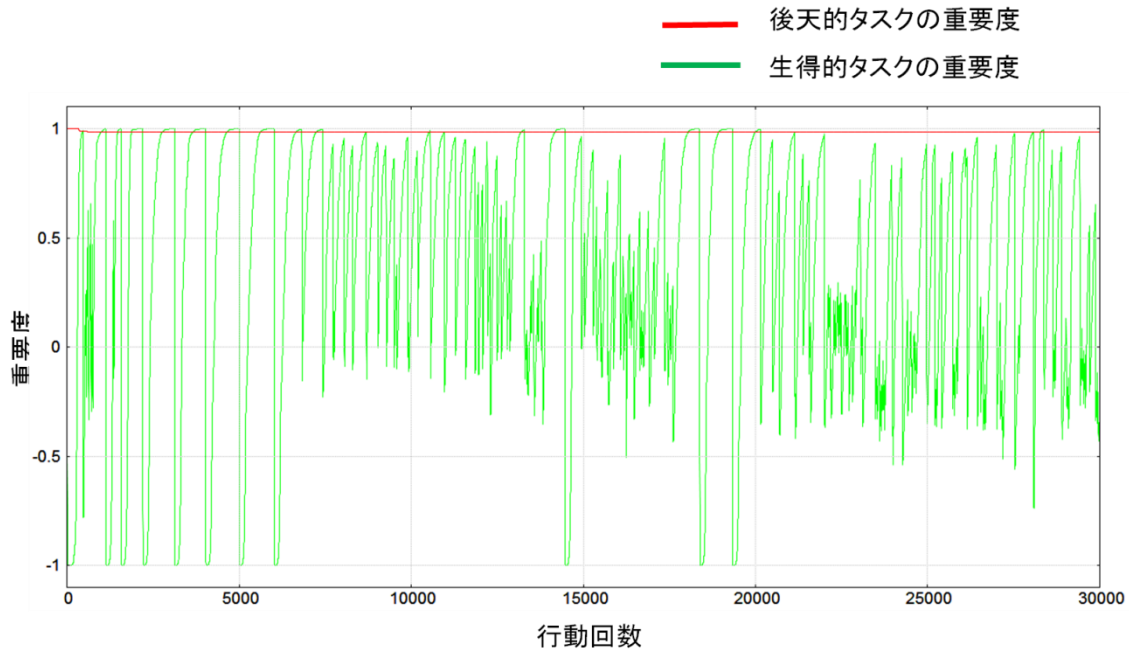

図 4.10　エージェントの行動回数に対する各重要度の変化

図 4.7 のエージェントのゴール回数の推移では，エージェントの行動開始から行動回数 6000 回付近までゴール回数が急速に増加し，それ以降はゆるやかに増加している．次に図 4.8 のエージェントのエネルギー残量の推移では，エージェントの行動開始から行動回数 6000 回付近までエネルギー残量は 0 から 100 まで何度も大きく変化している．エージェントのエネルギー残量が 0 になった場合には，エネルギー残量は 100 まで回復する．従って，これはエージェントのエネルギー残量が何度も 0 になり 100 まで回復していることを表している．また，行動回数 6000 回以降は，エネルギー残量 50 前後を上限としてエネルギー残量が減る度に回復している．

図 4.10 の各重要度の変化については，生得的タスクの重要度は常に大きく変化しており，後天的タスクの重要度は，ほぼ 1 に収束している．

次に，パラメータ設定が$\delta = -0.13, \sigma = 10$である場合のエージェントの行動回数に対するボール位置到達回数の推移を図 4.11 に，行動回数に対するエネルギー残量の推移を図 4.12 に示す．また，行動回数に対するゴール位置到達回数とエネルギー残量の推移を重ねあわせたものを図 4.13 に，行動回数に対する各重要度の変化を図 4.14 に示す．

図 4.11 のエージェントのゴール回数の推移では，エージェントの行動開始から行動回数 17000 回付近までゴール回数が徐々に増加し，それ以降はゆるやかに増加している．次に図 4.12 のエージェントのエネルギー残量の推移では，全体的にエネルギー残量は 0 から 100 まで何度も大きく変化している．エージェントのエネルギー残量が 0 になった場合には，エネルギー残量は 100 まで回復する．従って，これはエージェントのエネルギー残量が何

度も 0 になり 100 まで回復していることを表している．また，行動回数 17000 回~24000 回付近では，エネルギー残量 80~60 前後を保っている．

図 4.14 の各重要度の変化については，生得的タスクの重要度は常に大きく変化しており，後天的タスクの重要度は，ほぼ 1 に収束している．

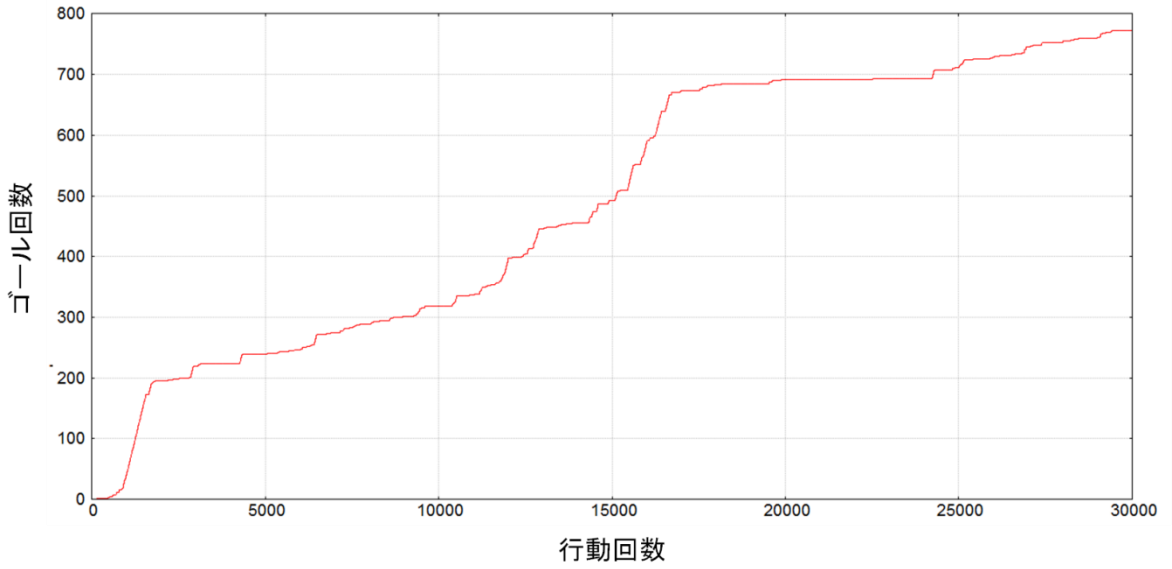

図 4.11　エージェントの行動回数に対するゴール位置到達回数の推移

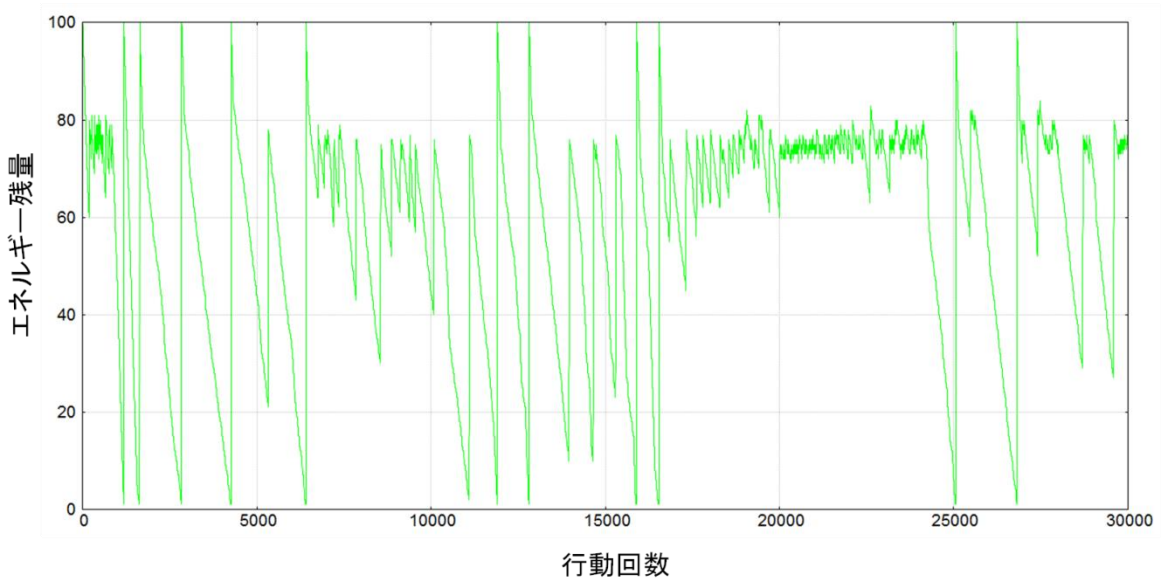

図 4.12　エージェントの行動回数に対するエネルギー残量の推移

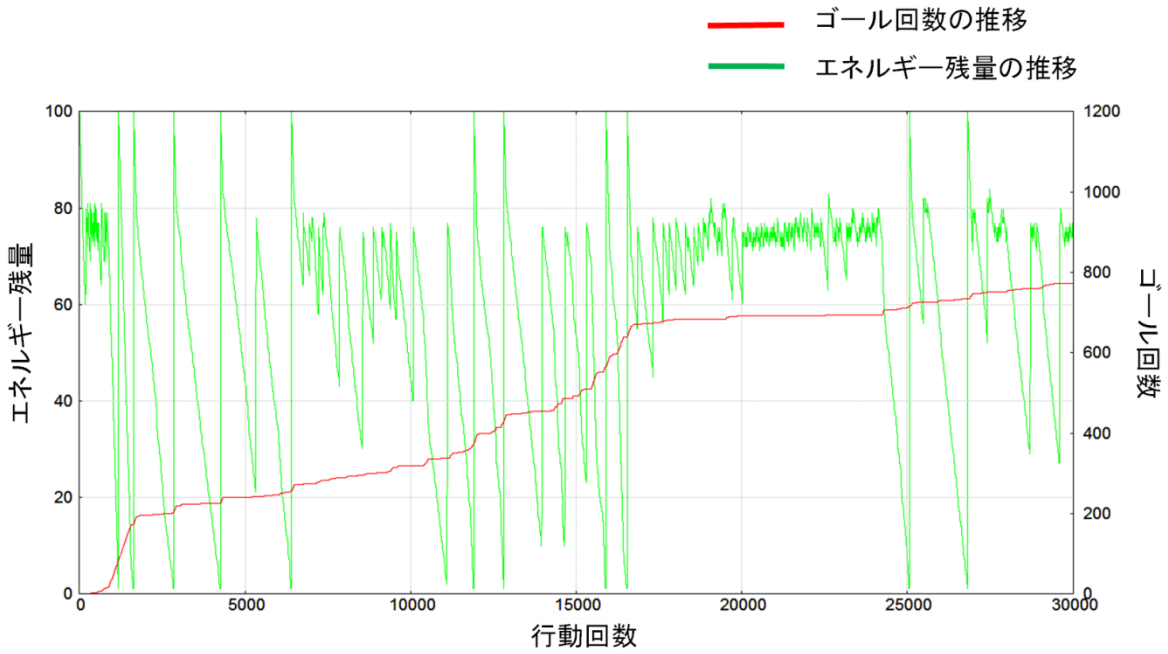

図 4.13　行動回数に対するゴール到達回数とエネルギー残量の推移

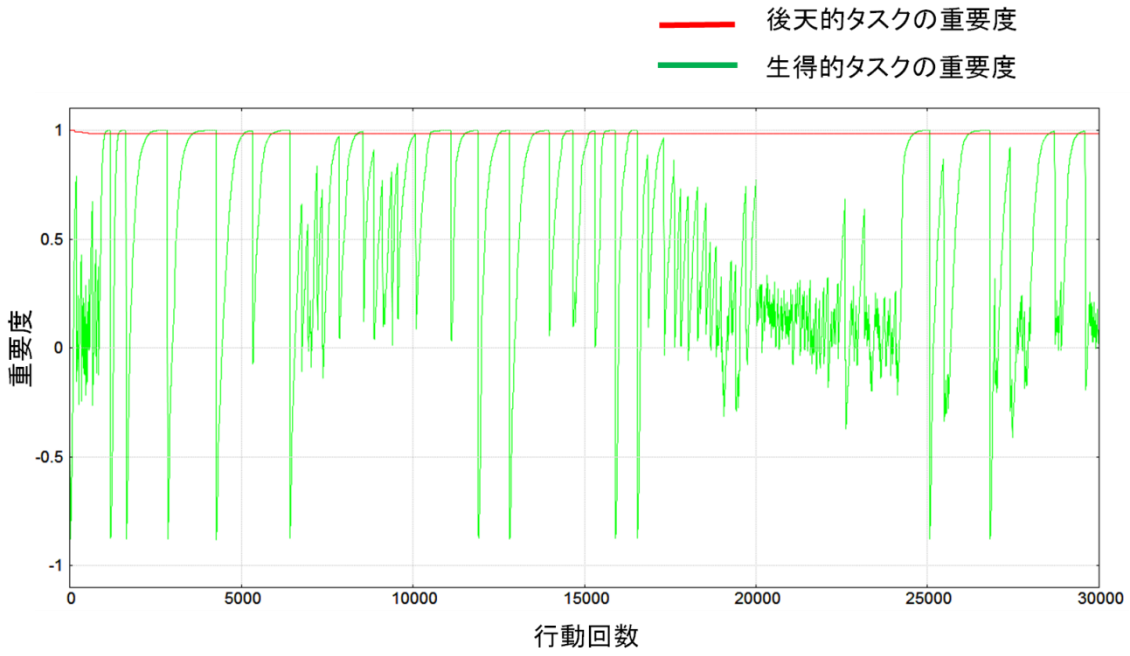

図 4.14　エージェントの行動回数に対する各重要度の変化

次に，パラメータ設定が$\delta = -0.3, \sigma = 10$である場合のエージェントの行動回数に対するボール位置到達回数の推移を図 4.15 に，行動回数に対するエネルギー残量の推移を図 4.16 に示す．また，行動回数に対するゴール位置到達回数とエネルギー残量の推移を重ねあわ

せたものを図 4.17 に，行動回数に対する各重要度の変化を図 4.18 に示す．

図 4.15 のエージェントのゴール回数の推移では，エージェントの行動開始からゴール回数が徐々に増加している．また，他のパラメータ設定時よりも合計ゴール回数が増加している．次に図 4.16 のエージェントのエネルギー残量の推移では，全体的にエネルギー残量は 0 から 100 まで何度も大きく変化している．エージェントのエネルギー残量が 0 になった場合には，エネルギー残量は 100 まで回復する．従って，これはエージェントのエネルギー残量が何度も 0 になり 100 まで回復していることを表している．また，ところどころではエネルギー残量が 40~20 に保たれている部分もある．

図 4.18 の各重要度の変化については, 生得的タスクの重要度は常に大きく変化しており，後天的タスクの重要度は，ほぼ 1 に収束している．

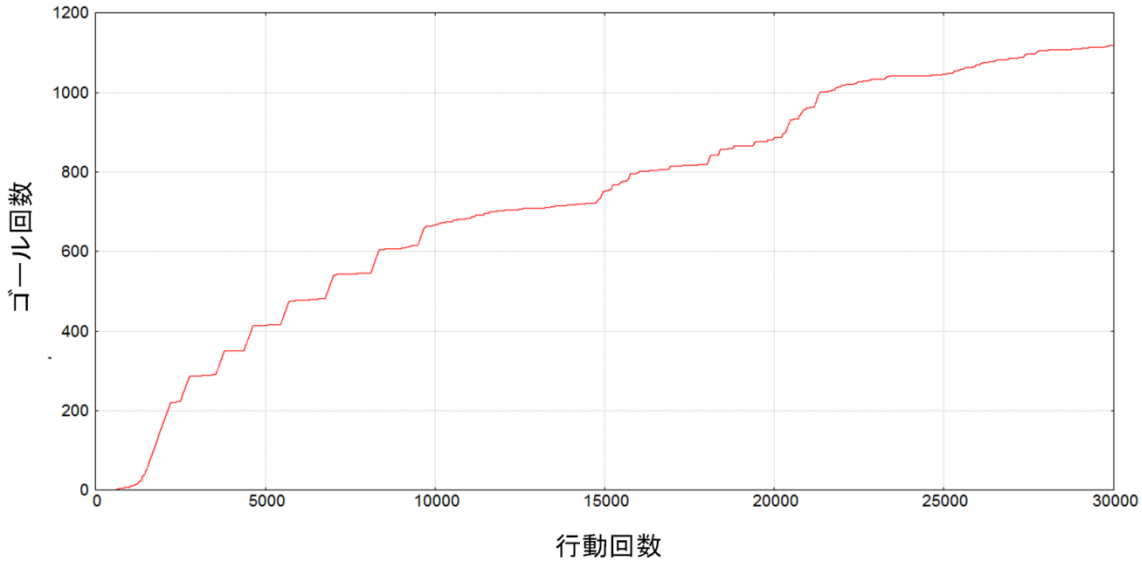

図 4.15　エージェントの行動回数に対するゴール位置到達回数の推移

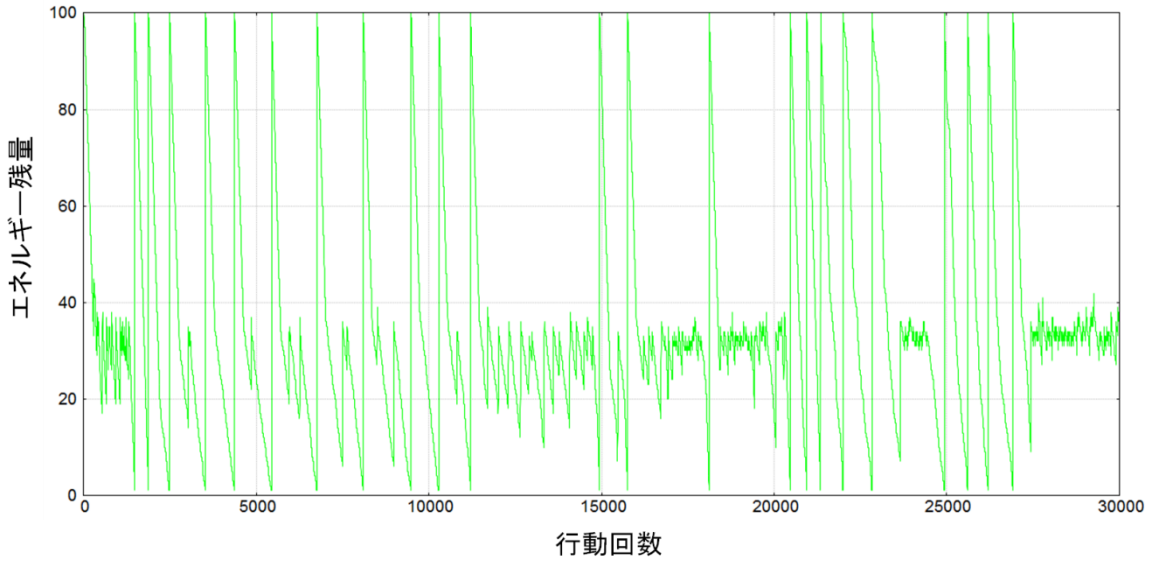

図 4.16　エージェントの行動回数に対するエネルギー残量の推移

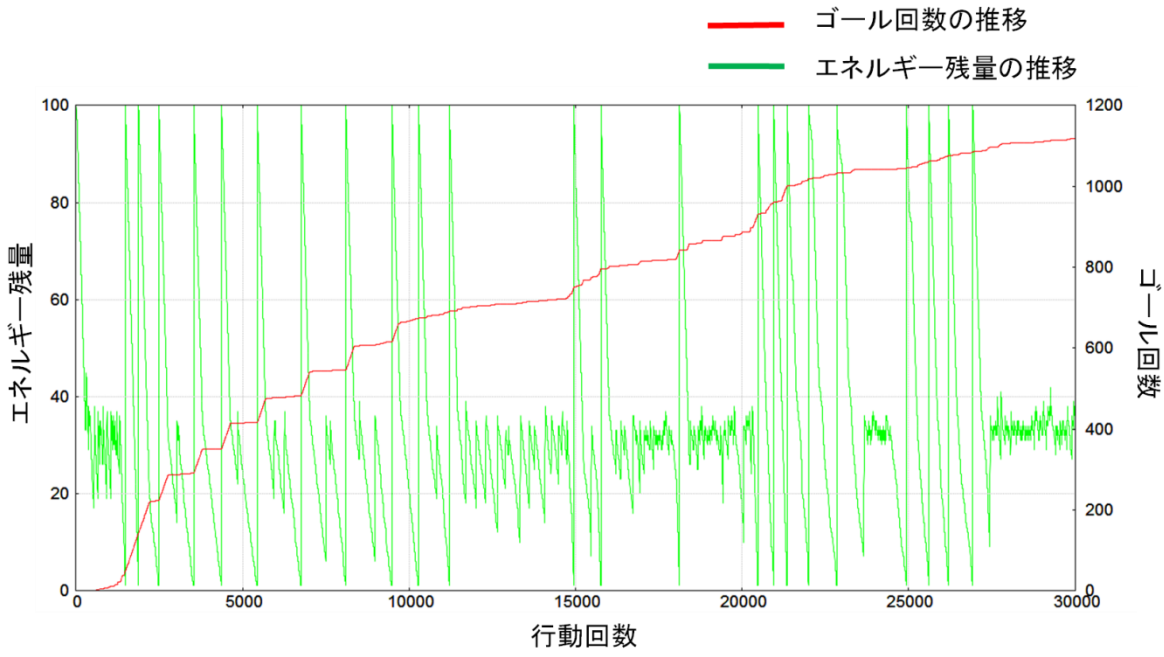

図 4.17　行動回数に対するゴール到達回数とエネルギー残量の推移

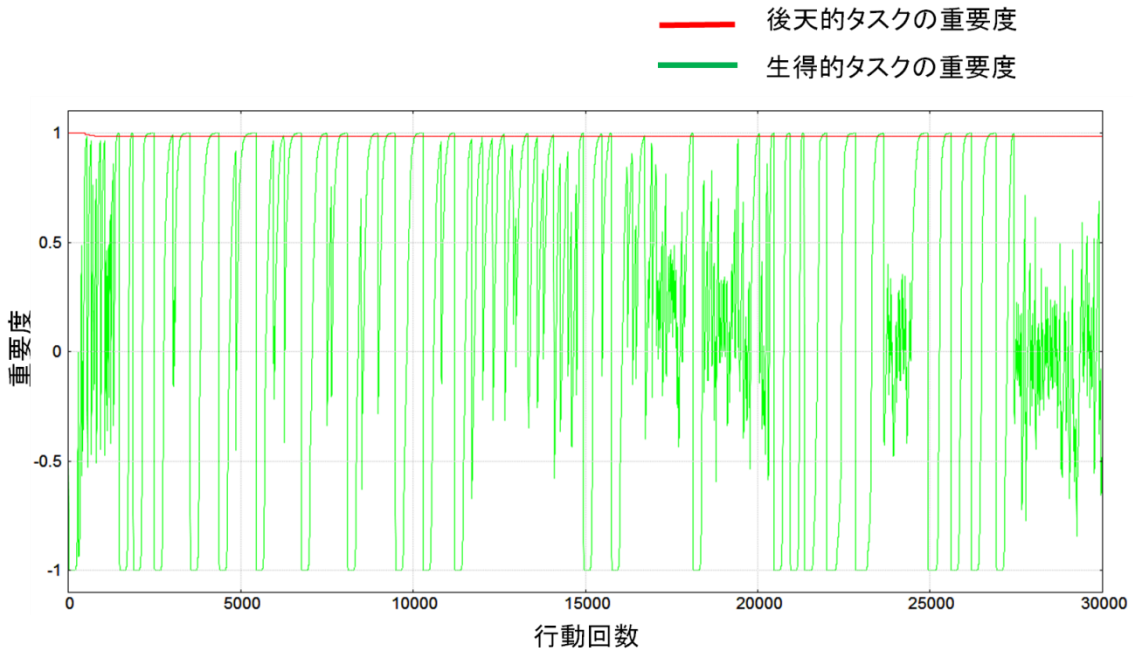

図 4.18　エージェントの行動回数に対する各重要度の変化

４．４．２　後天的タスクに対して人間が＋0.2 を与えた場合の実験結果

生得的タスクの重要度算出に関するパラメータを変化させ，後天的タスクの重要度についてはゴール位置到達毎に“＋0.2”を与えた場合の実験結果を示す．実験パターンとしては，4 から 6 に相当する．パラメータ設定が$\delta = -0.2, \sigma = 10$である場合のエージェントの行動回数に対するボール位置到達回数の推移を図 4.19 に，行動回数に対するエネルギー残量の推移を図 4.20 に示す．また，行動回数に対するゴール位置到達回数とエネルギー残量の推移を重ねあわせたものを図 4.21 に，行動回数に対する各重要度の変化を図 4.22 に示す．

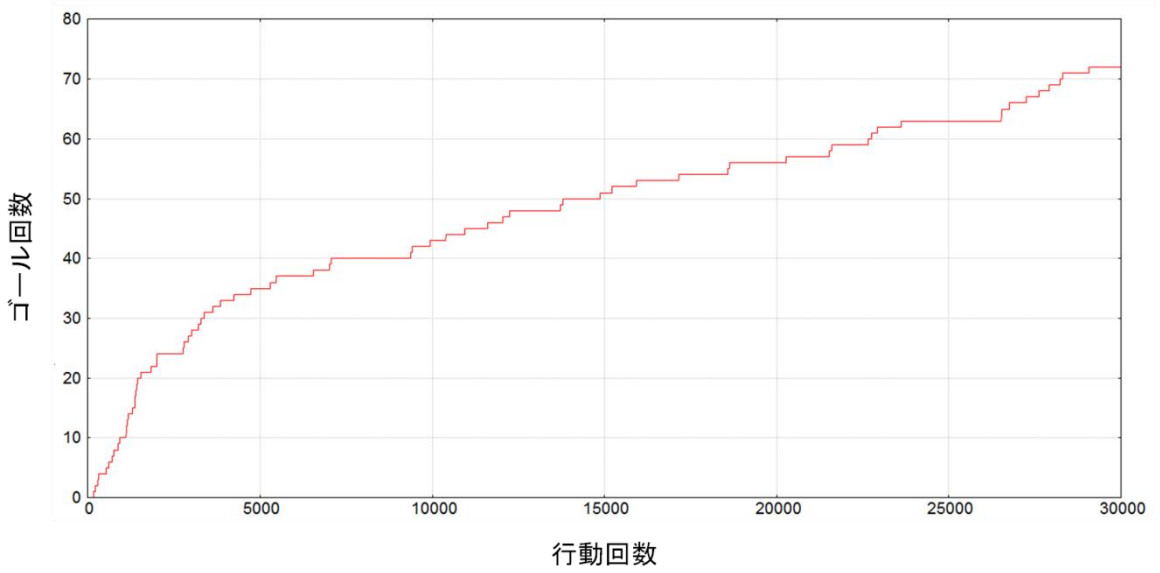

図 4.19　エージェントの行動回数に対するゴール位置到達回数の推移

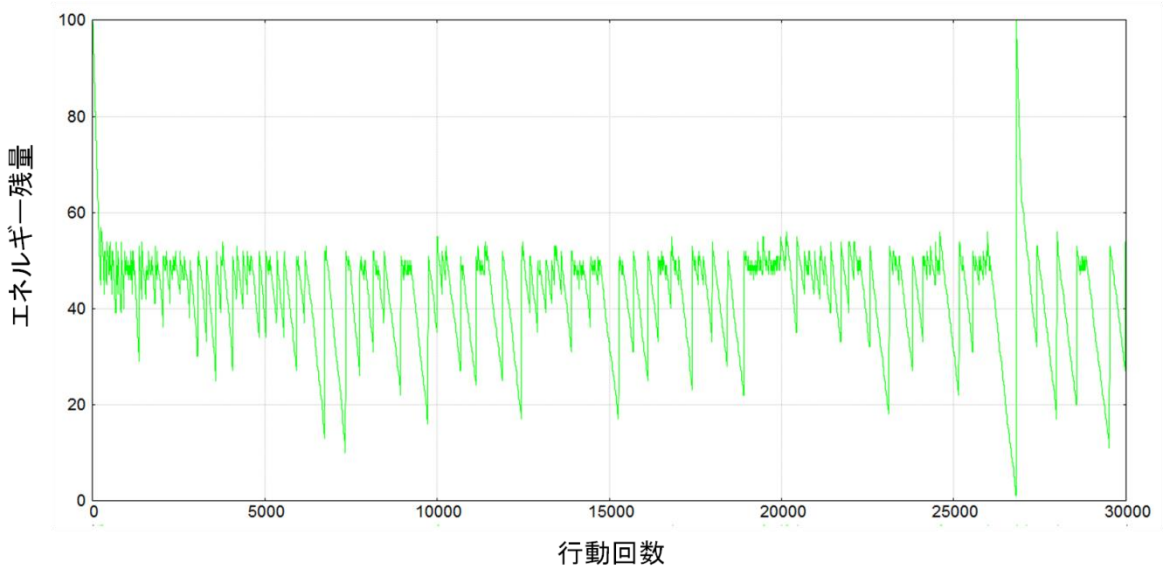

図 4.20　エージェントの行動回数に対するエネルギー残量の推移

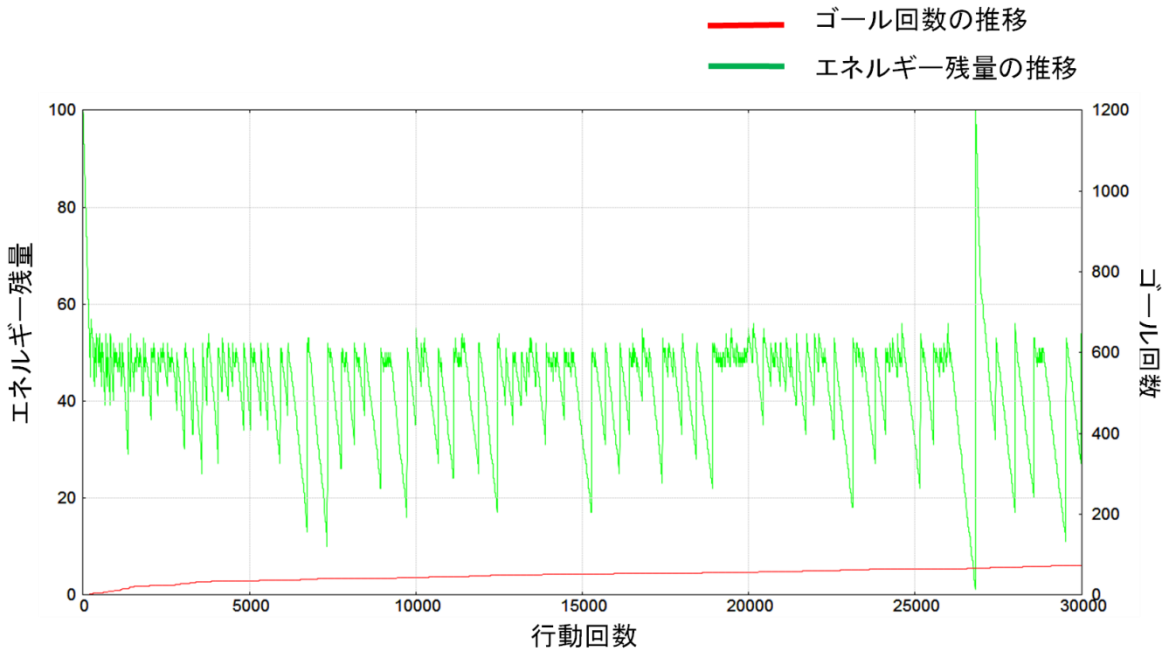

図 4.21　行動回数に対するゴール到達回数とエネルギー残量の推移

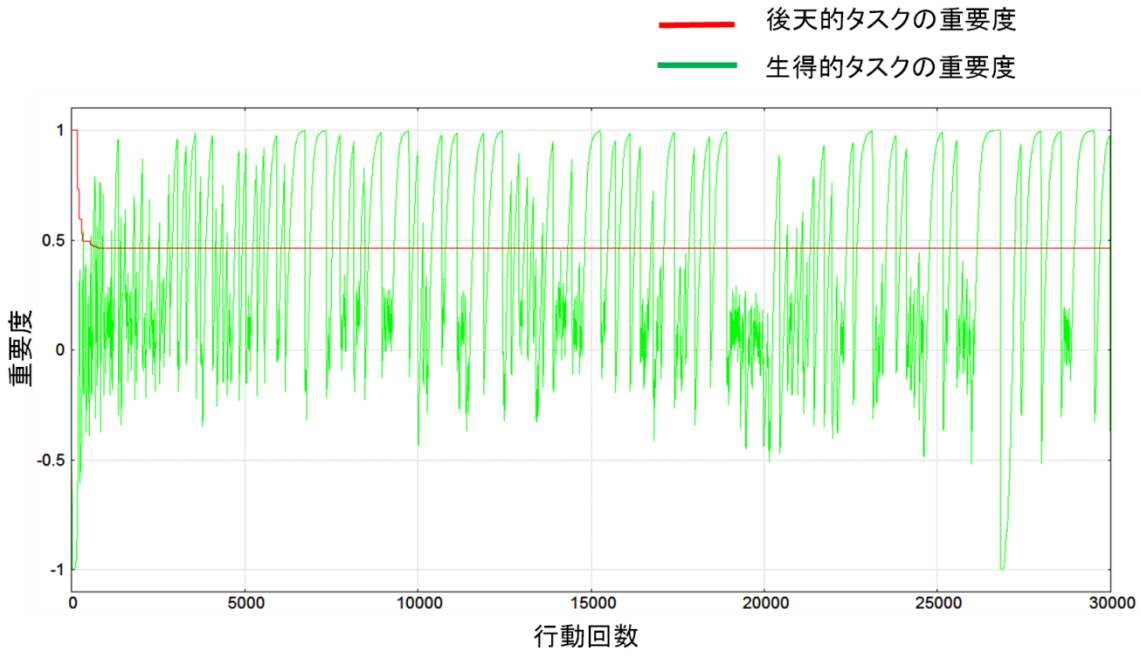

図 4.22　エージェントの行動回数に対する各重要度の変化

図 4.19 のエージェントのゴール回数の推移では，エージェントの行動開始からゴール回数が徐々に増加している．しかし，合計ゴール回数自体は 70 前後と少ない．次に図 4.20 のエージェントのエネルギー残量の推移では，全体的にエネルギー残量は 50 から 20 前後に保たれており，エージェントのエネルギー残量が 0 となったのは一度のみである．

図 4.18 の各重要度の変化については, 生得的タスクの重要度は常に大きく変化しており, 後天的タスクの重要度は, ほぼ 0.5 に収束している.

次に, パラメータ設定が$\delta = -0.13, \sigma = 10$である場合のエージェントの行動回数に対するボール位置到達回数の推移を図 4.23 に, 行動回数に対するエネルギー残量の推移を図 4.24 に示す. また, 行動回数に対するゴール位置到達回数とエネルギー残量の推移を重ねあわせたものを図 4.25 に, 行動回数に対する各重要度の変化を図 4.26 に示す.

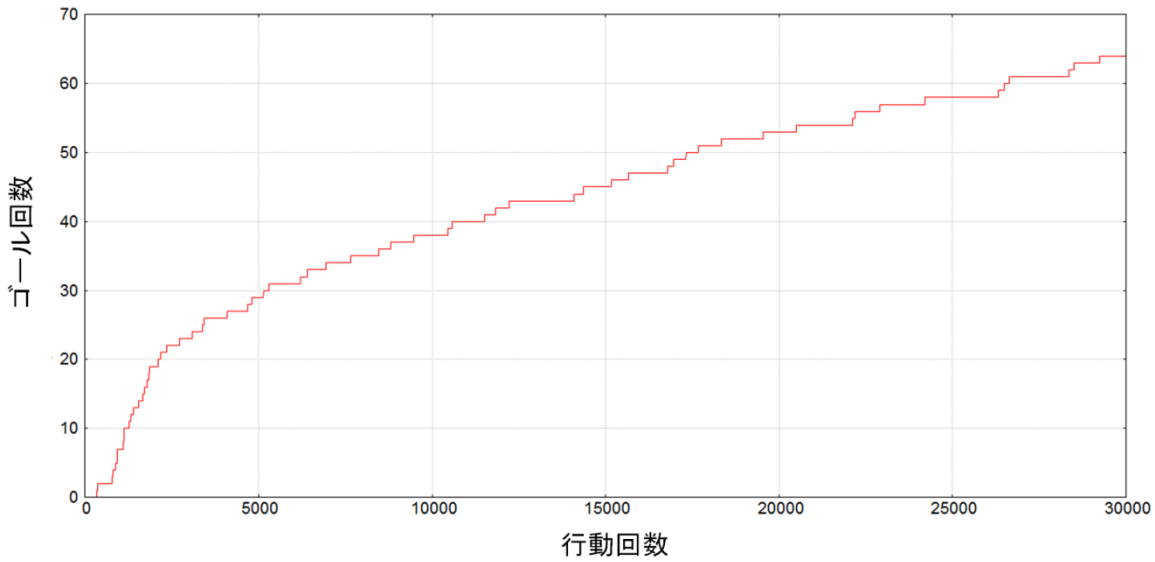

図 4.23　エージェントの行動回数に対するゴール位置到達回数の推移

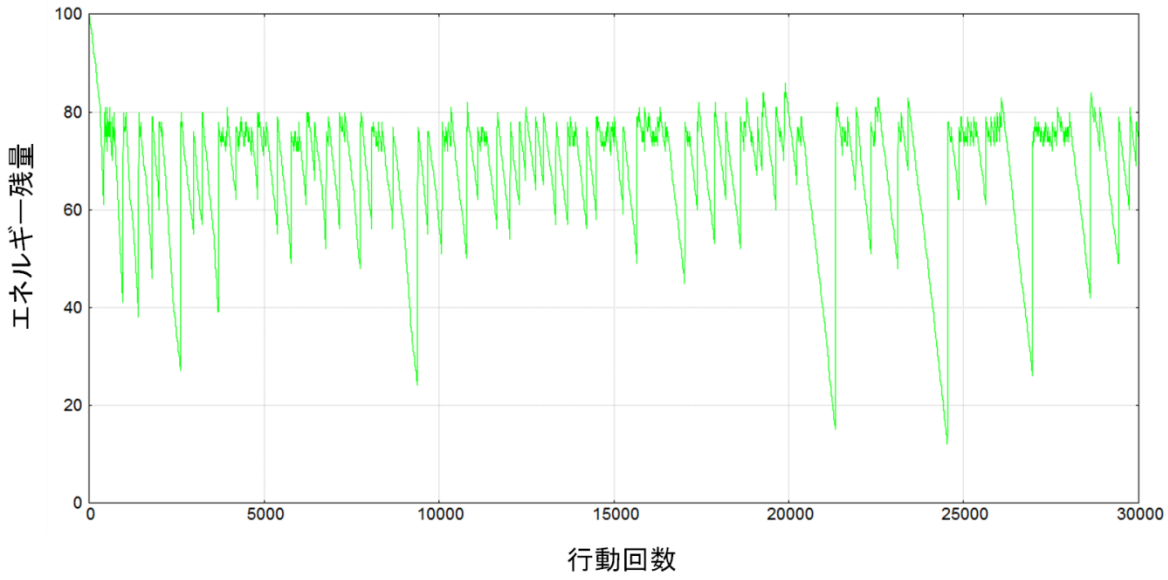

図 4.24　エージェントの行動回数に対するエネルギー残量の推移

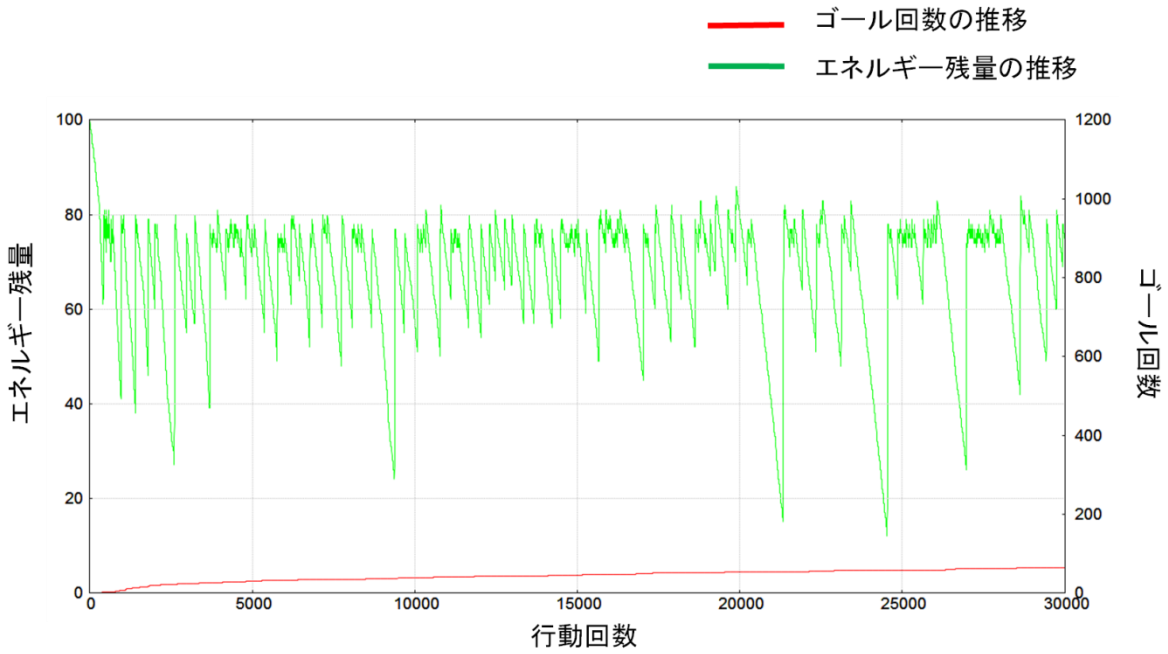

図 4.25　行動回数に対するゴール到達回数とエネルギー残量の推移

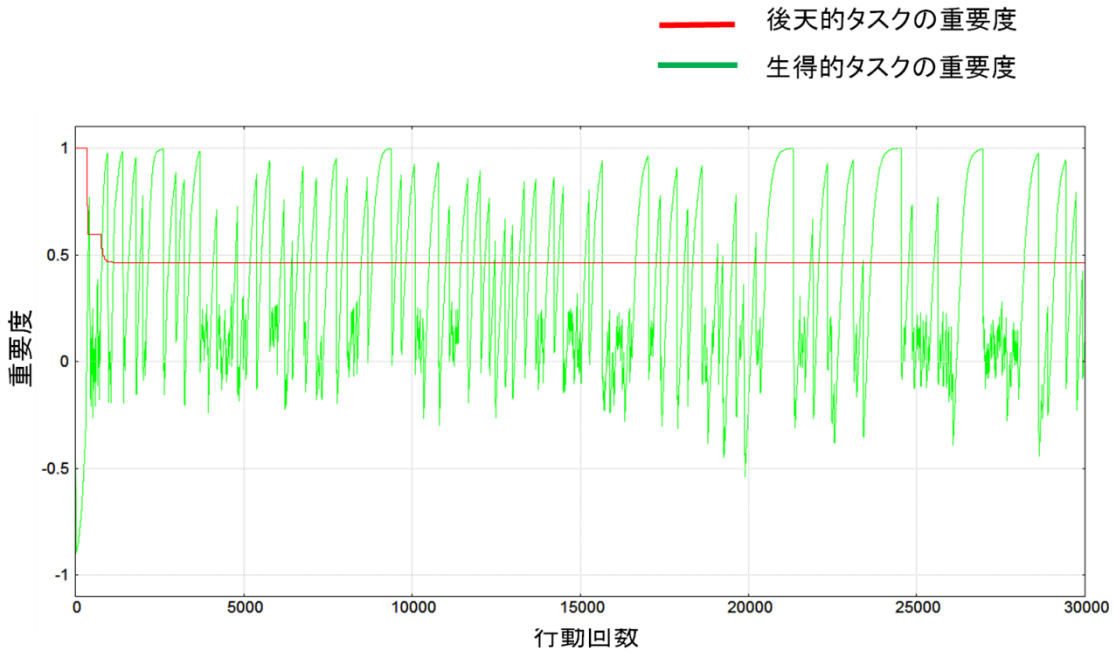

図 4.26　エージェントの行動回数に対する各重要度の変化

図 4.23 のエージェントのゴール回数の推移では，エージェントの行動開始からゴール回数が徐々に増加している．しかし，合計ゴール回数自体は 70 前後と少ない．次に図 4.24 のエージェントのエネルギー残量の推移では，全体的にエネルギー残量は 80 から 40 前後に保たれており，エージェントのエネルギー残量は一度も 0 になっていない．

図 4.26 の各重要度の変化については, 生得的タスクの重要度は常に大きく変化しており, 後天的タスクの重要度は, ほぼ 0.5 に収束している.

次に, パラメータ設定が$\delta = -0.3, \sigma = 10$である場合のエージェントの行動回数に対するボール位置到達回数の推移を図 4.27 に, 行動回数に対するエネルギー残量の推移を図 4.28 に示す. また, 行動回数に対するゴール位置到達回数とエネルギー残量の推移を重ねあわせたものを図 4.29 に, 行動回数に対する各重要度の変化を図 4.30 に示す.

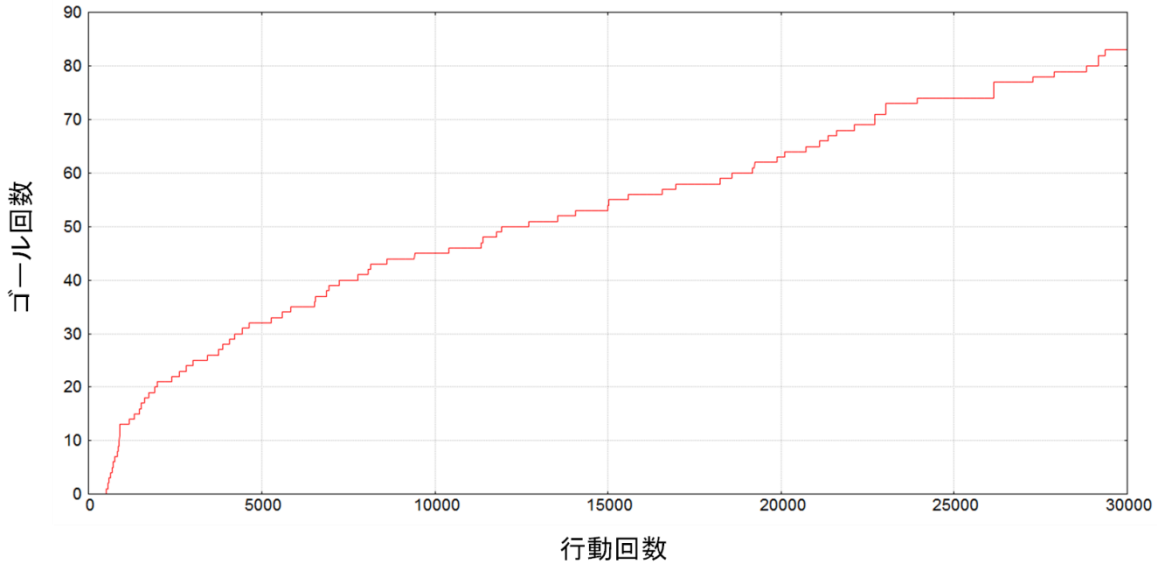

図 4.27　エージェントの行動回数に対するゴール位置到達回数の推移

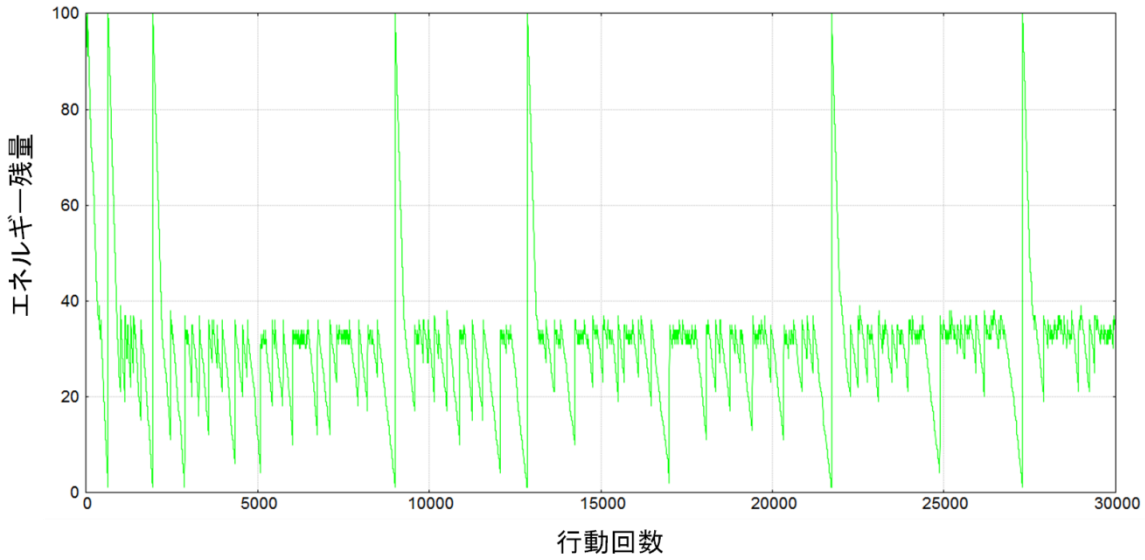

図 4.28　エージェントの行動回数に対するエネルギー残量の推移

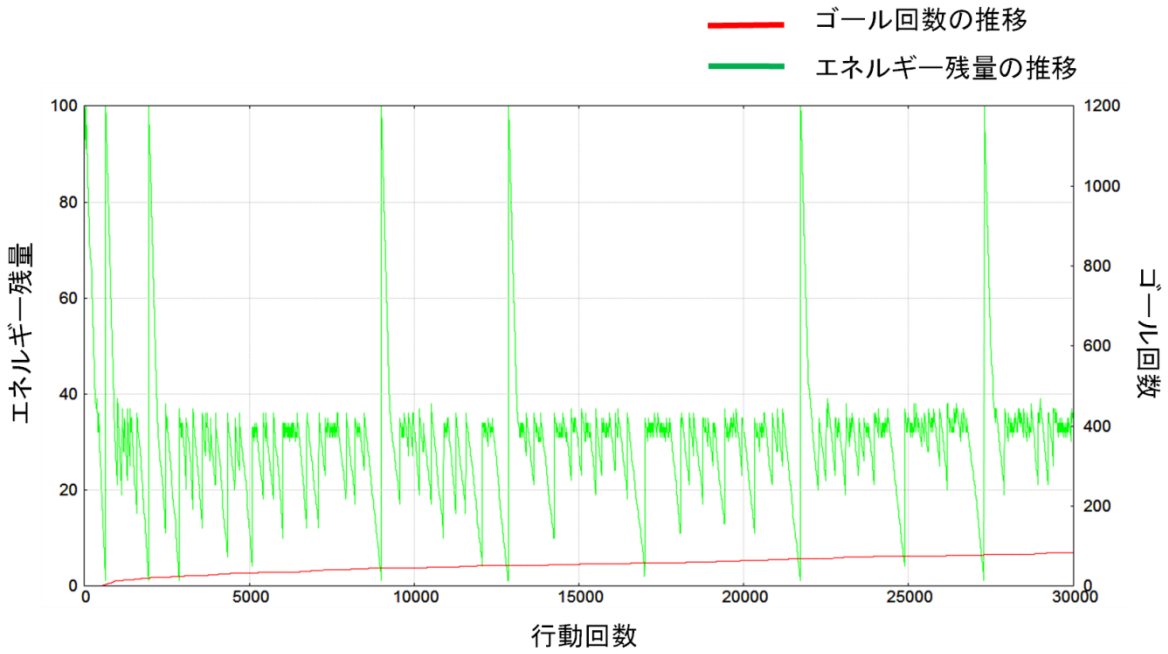

図 4.29　行動回数に対するゴール到達回数とエネルギー残量の推移

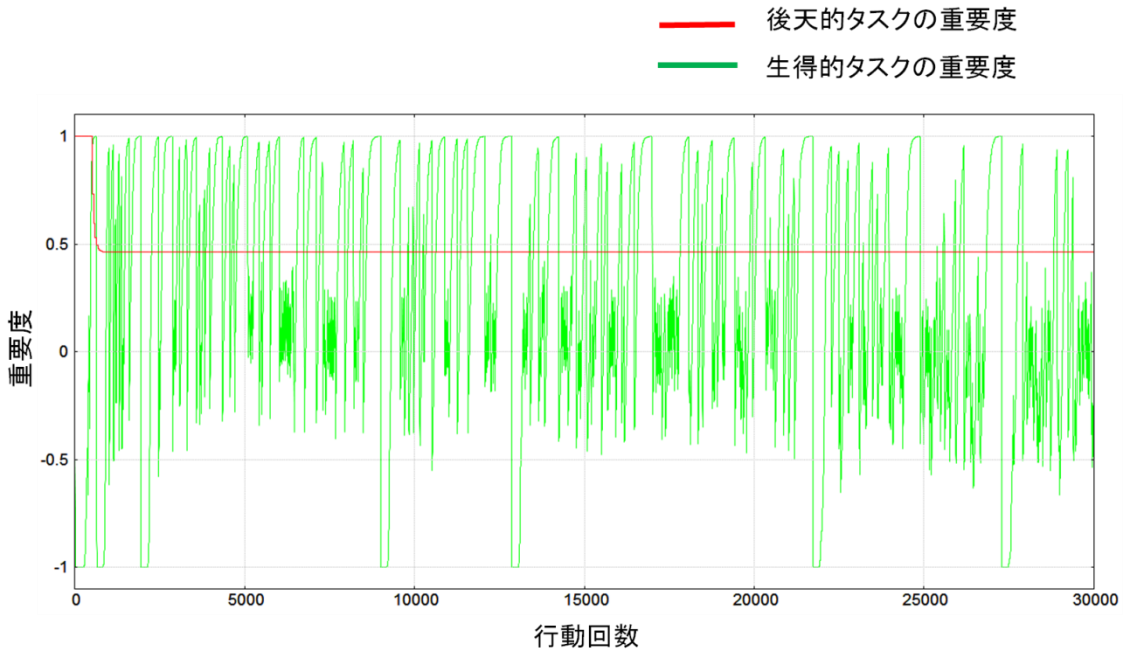

図 4.30　エージェントの行動回数に対する各重要度の変化

図 4.27 のエージェントのゴール回数の推移では，エージェントの行動開始からゴール回数が徐々に増加している．合計ゴール回数については，他のパラメータと比較すれば多少多いが 80 前後である．次に図 4.28 のエージェントのエネルギー残量の推移では，全体的にエネルギー残量は 40 から 10 前後に保たれているが，何度かエージェントのエネルギー

残量が 0 になっている．

図 4.30 の各重要度の変化については，生得的タスクの重要度は常に大きく変化しており，後天的タスクの重要度は，ほぼ 0.5 に収束している．

４．４．３　後天的タスクに対して人間が－１を与えた場合の実験結果

生得的タスクの重要度算出に関するパラメータを変化させ，後天的タスクの重要度についてはゴール位置到達毎に“−1”を与えた場合の実験結果を示す．実験パターンとしては，7 から 9 に相当する．パラメータ設定が$\delta = -0.2, \sigma = 10$である場合のエージェントの行動回数に対するボール位置到達回数の推移を図 4.31 に，行動回数に対するエネルギー残量の推移を図 4.32 に示す．また，行動回数に対するゴール位置到達回数とエネルギー残量の推移を重ねあわせたものを図 4.33 に，行動回数に対する各重要度の変化を図 4.34 に示す．

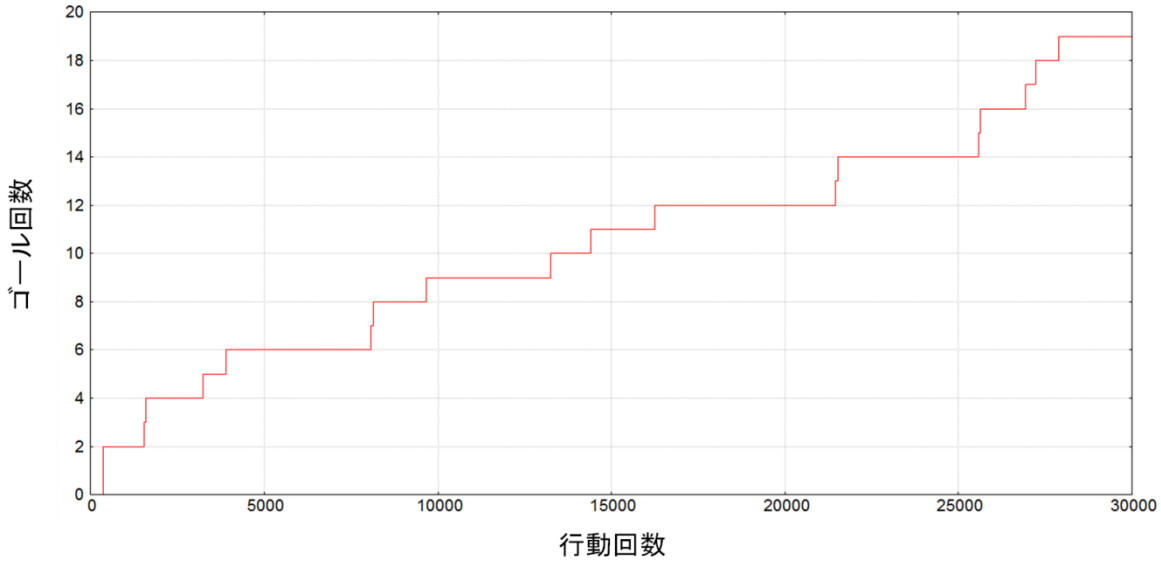

図 4.31　エージェントの行動回数に対するゴール位置到達回数の推移

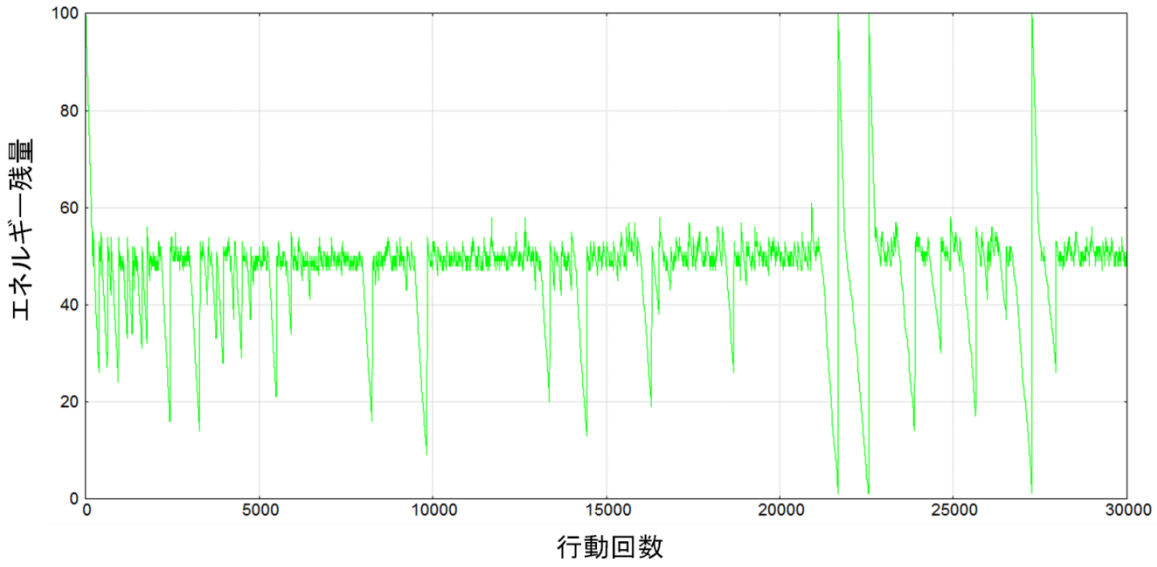

図 4.32　エージェントの行動回数に対するエネルギー残量の推移

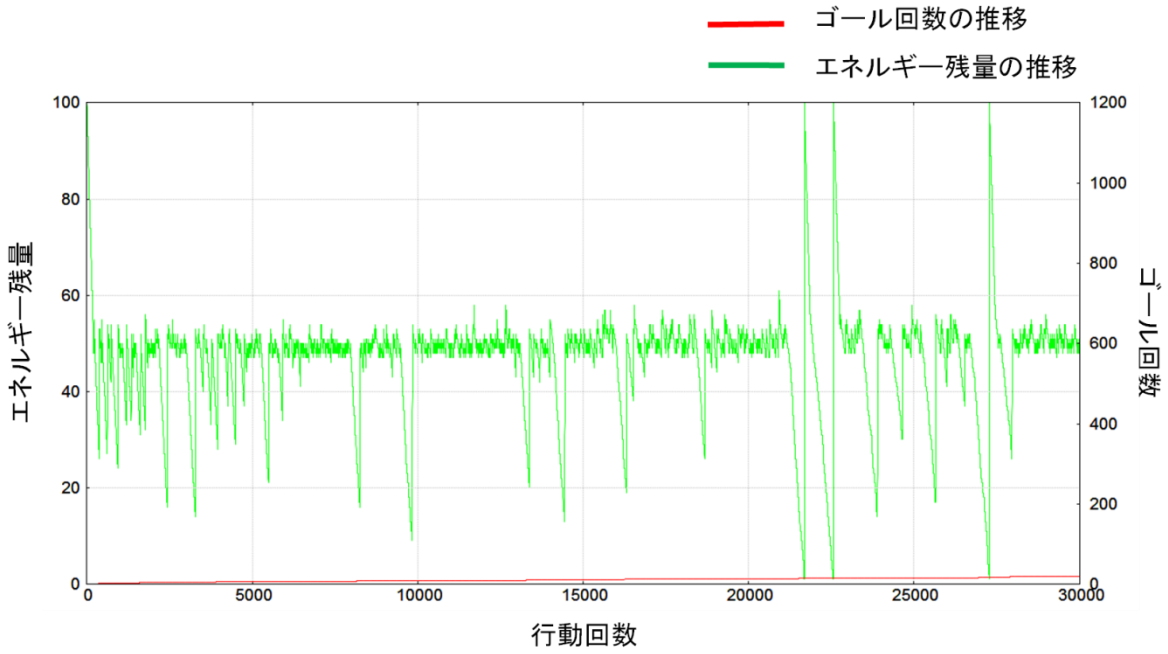

図 4.33　行動回数に対するゴール到達回数とエネルギー残量の推移

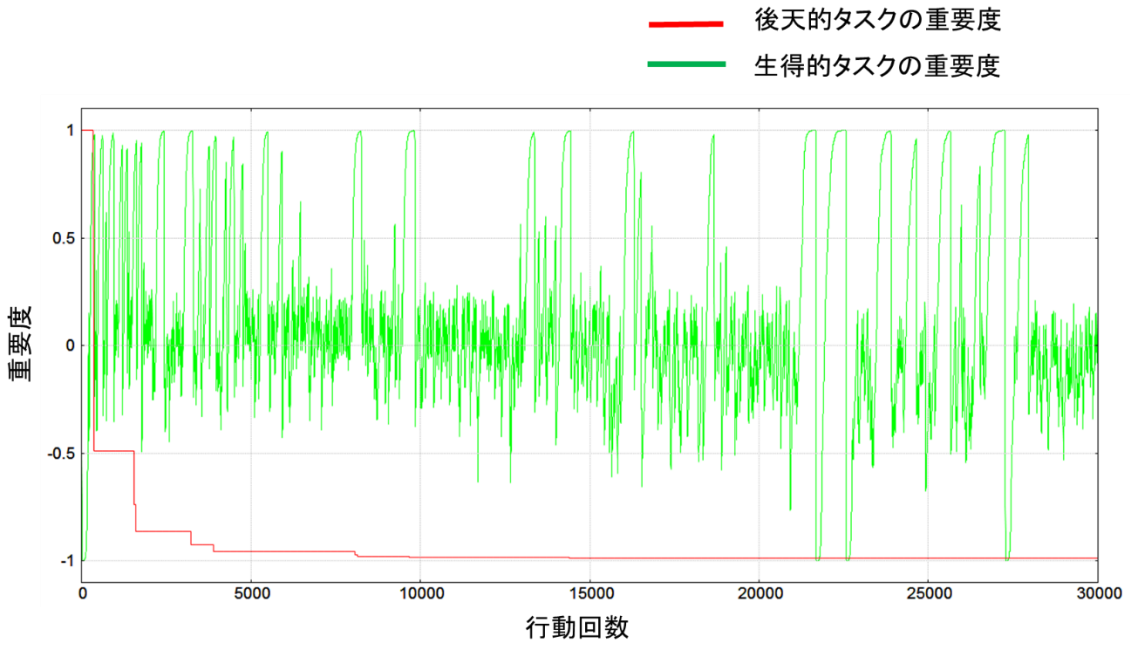

図 4.34　エージェントの行動回数に対する各重要度の変化

図 4.31 のエージェントのゴール回数の推移では，エージェントの行動開始からゴール回数が徐々に増加している．合計ゴール回数については，これまでの実験と比較しても少なく 20 前後である．次に図 4.32 のエージェントのエネルギー残量の推移では，全体的にエネルギー残量は 50 から 30 前後に保たれているが，何度かエージェントのエネルギー残量が 0 になっている．

図 4.34 の各重要度の変化については, 生得的タスクの重要度は常に大きく変化しており，後天的タスクの重要度は，1 から減少し，最終的にはほぼ−1 に収束している．

次に, パラメータ設定が$\delta = -0.13, \sigma = 10$である場合のエージェントの行動回数に対するボール位置到達回数の推移を図 4.35 に，行動回数に対するエネルギー残量の推移を図 4.36 に示す．また，行動回数に対するゴール位置到達回数とエネルギー残量の推移を重ねあわせたものを図 4.37 に，行動回数に対する各重要度の変化を図 4.38 に示す．

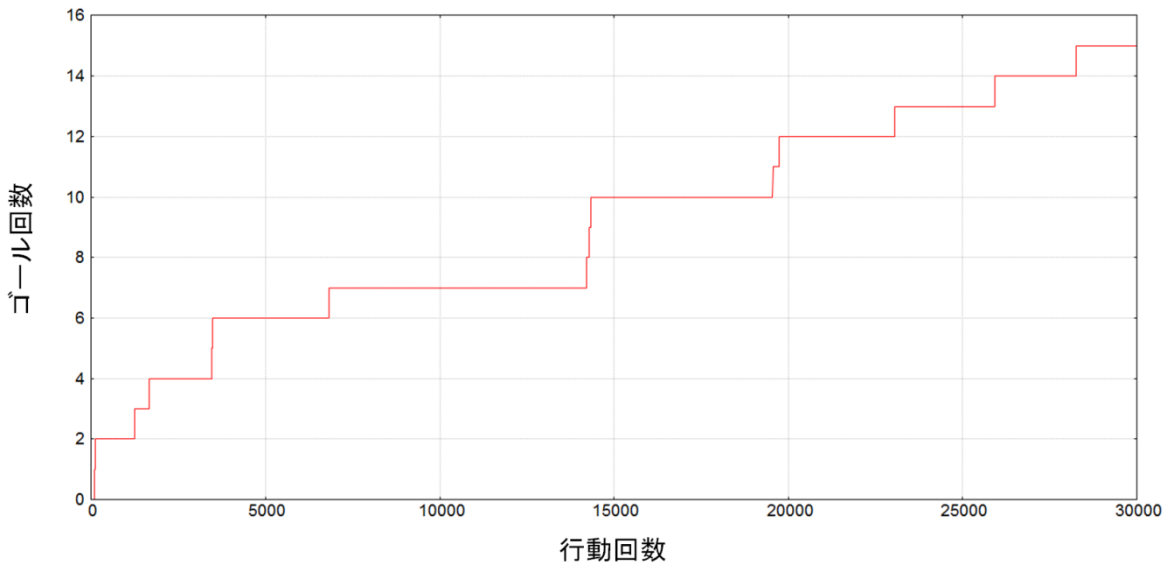

図 4.35　エージェントの行動回数に対するゴール位置到達回数の推移

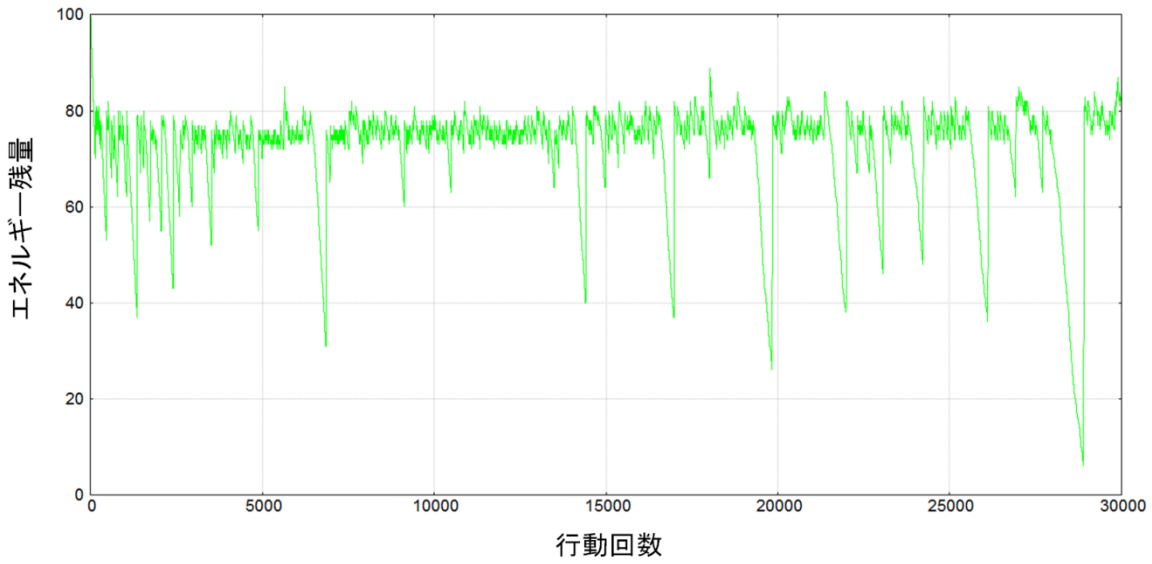

図 4.36　エージェントの行動回数に対するエネルギー残量の推移

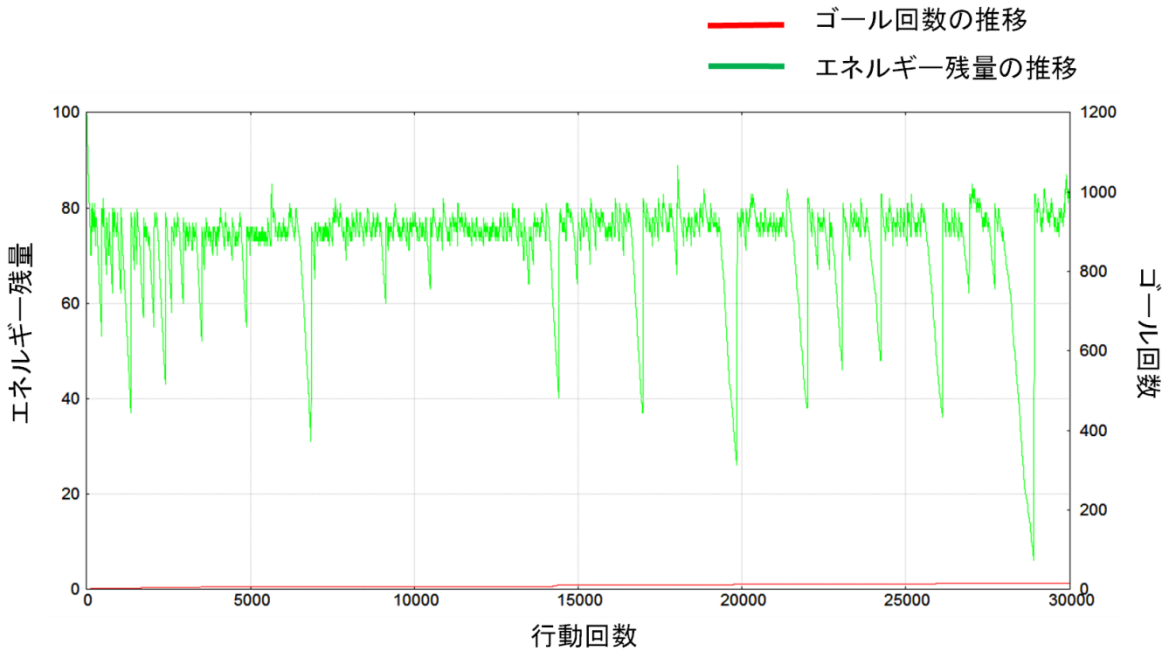

図 4.37　行動回数に対するゴール到達回数とエネルギー残量の推移

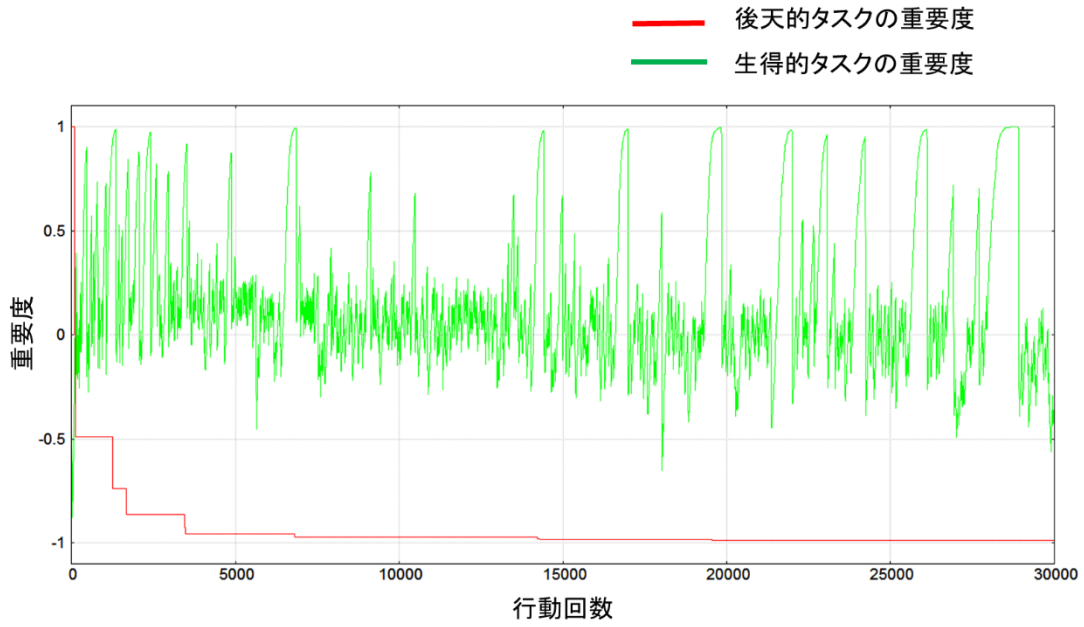

図 4.38　エージェントの行動回数に対する各重要度の変化

図 4.35 のエージェントのゴール回数の推移では，エージェントの行動開始からゴール回数が徐々に増加している．合計ゴール回数については，これまでの実験と比較しても少なく 18 前後である．次に図 4.36 のエージェントのエネルギー残量の推移では，全体的にエネルギー残量は 80 前後を上限として，エネルギー残量が減ると再び 80 前後まで回復している，エージェントのエネルギー残量は，一度もが 0 になっていない．

図 4.38 の各重要度の変化については，生得的タスクの重要度は常に大きく変化しているが，マイナス方向にはあまり振れていない．後天的タスクの重要度は，1 から減少し，最終的にはほぼ−1 に収束している．

次に，パラメータ設定が$\delta = -0.3, \sigma = 10$である場合のエージェントの行動回数に対するボール位置到達回数の推移を図 4.39 に, 行動回数に対するエネルギー残量の推移を図 4.40 に示す．また，行動回数に対するゴール位置到達回数とエネルギー残量の推移を重ねあわせたものを図 4.41 に，行動回数に対する各重要度の変化を図 4.42 に示す．

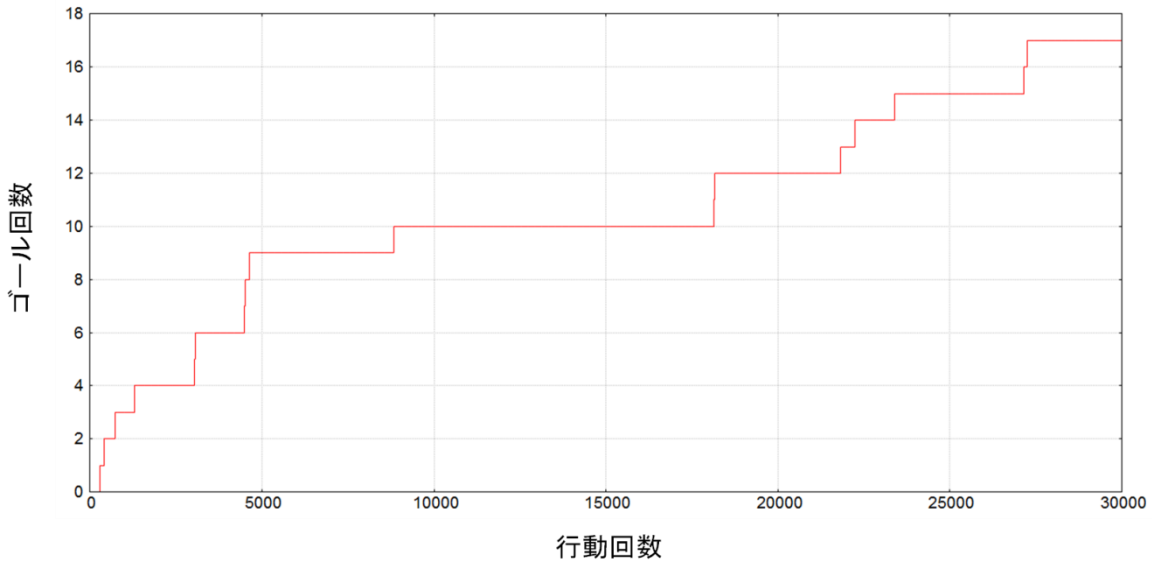

図 4.39　エージェントの行動回数に対するゴール位置到達回数の推移

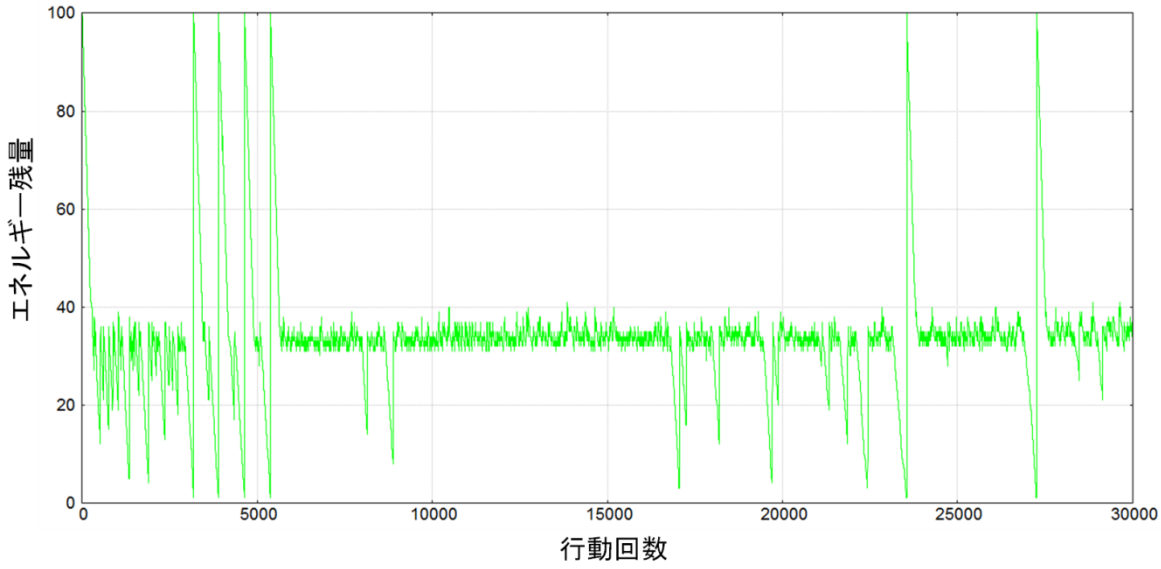

図 4.40　エージェントの行動回数に対するエネルギー残量の推移

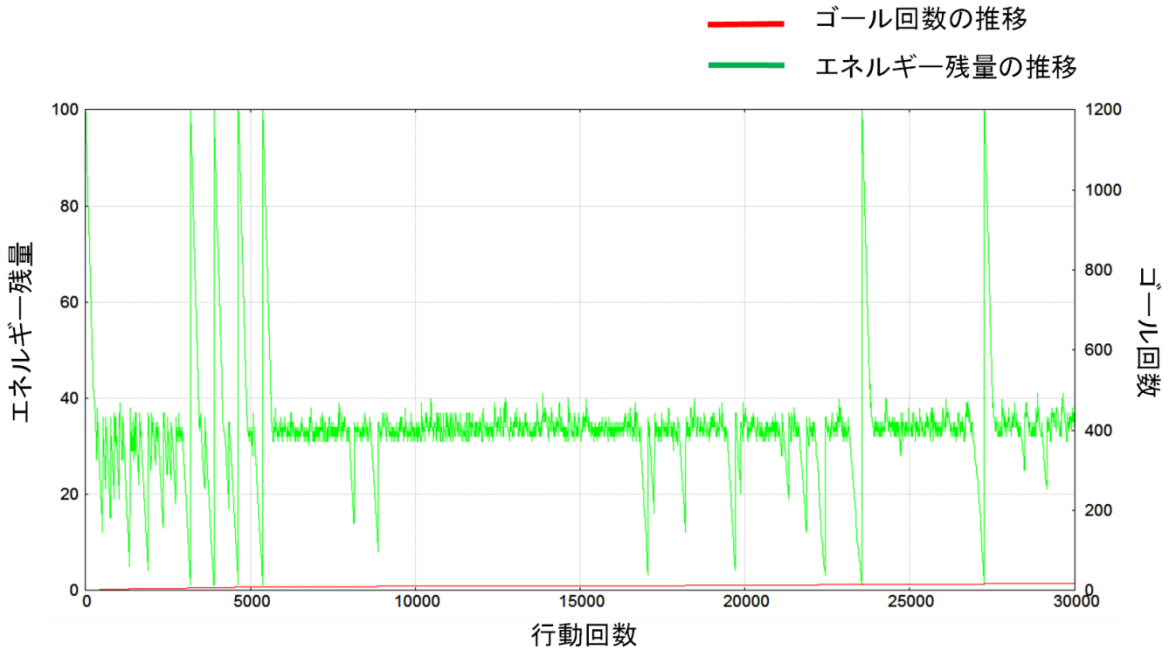

図 4.41　行動回数に対するゴール到達回数とエネルギー残量の推移

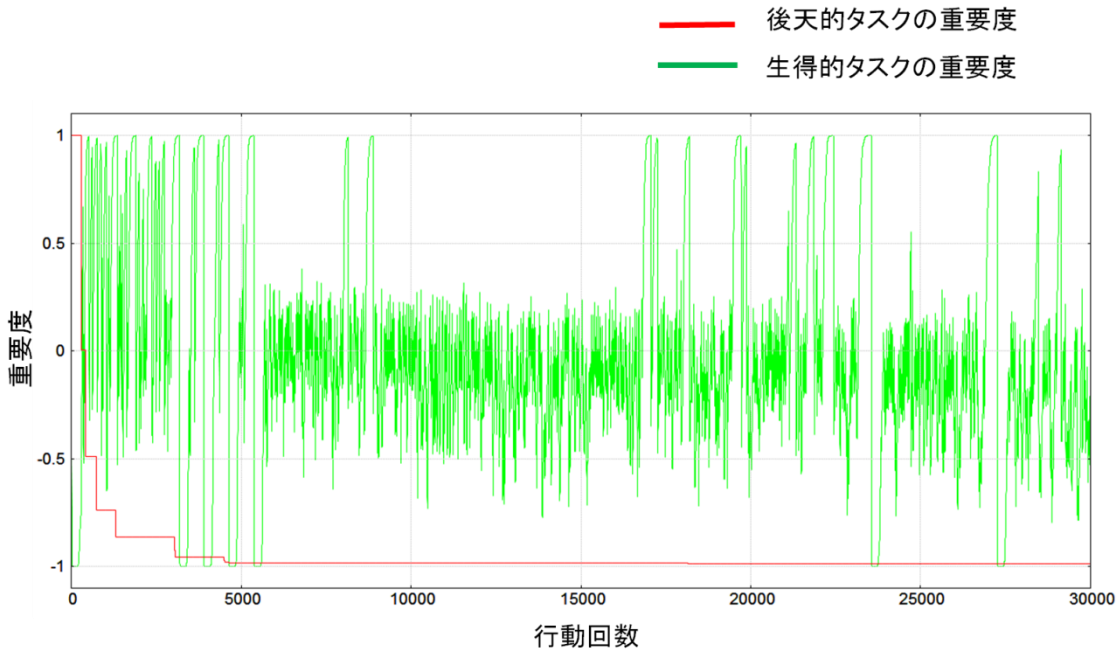

図 4.42　エージェントの行動回数に対する各重要度の変化

図 4.39 のエージェントのゴール回数の推移では，エージェントの行動開始からゴール回数が徐々に増加している．合計ゴール回数については，これまでの実験と比較しても少なく 18 前後である．次に図 4.40 のエージェントのエネルギー残量の推移では，全体的にエネルギー残量は 40 前後を上限として，エネルギー残量が減ると再び 40 前後まで回復しているが，エージェントのエネルギー残量は，何度か 0 になっている．

図 4.42 の各重要度の変化については，生得的タスクの重要度は常に大きく変化している．後天的タスクの重要度は，1 から減少し，最終的にはほぼ−1 に収束している．

４．４．４　後天的タスクの重要度を途中で変化させた場合の実験結果

後天的タスクの重要度について，人間が数値を与える頻度や値の大きさを一定行動毎に変化させた場合の実験結果を示す．実験パターンとしては，10 から 12 に相当する．生得的タスクの重要度算出時のパラメータ設定はすべて$\delta = -0.13, \sigma = 10$とする．また，後天的タスクについては，エージェントの行動回数 10000 回毎に変化させる．まずは，行動回数 0~10000 回まではゴールする度に“＋１”を与え，10001~2000 回まではゴール時に 2 分の１の確率で“＋１”，20001~30000 回では再びゴールする度に“＋１”を与える．このときのエージェントの行動回数に対するボール位置到達回数の推移を図 4.43 に，行動回数に対するエネルギー残量の推移を図 4.44 に示す．また，行動回数に対するゴール位置到達回

数とエネルギー残量の推移を重ねあわせたものを図 4.45 に，行動回数に対する各重要度の変化を図 4.46 に示す.

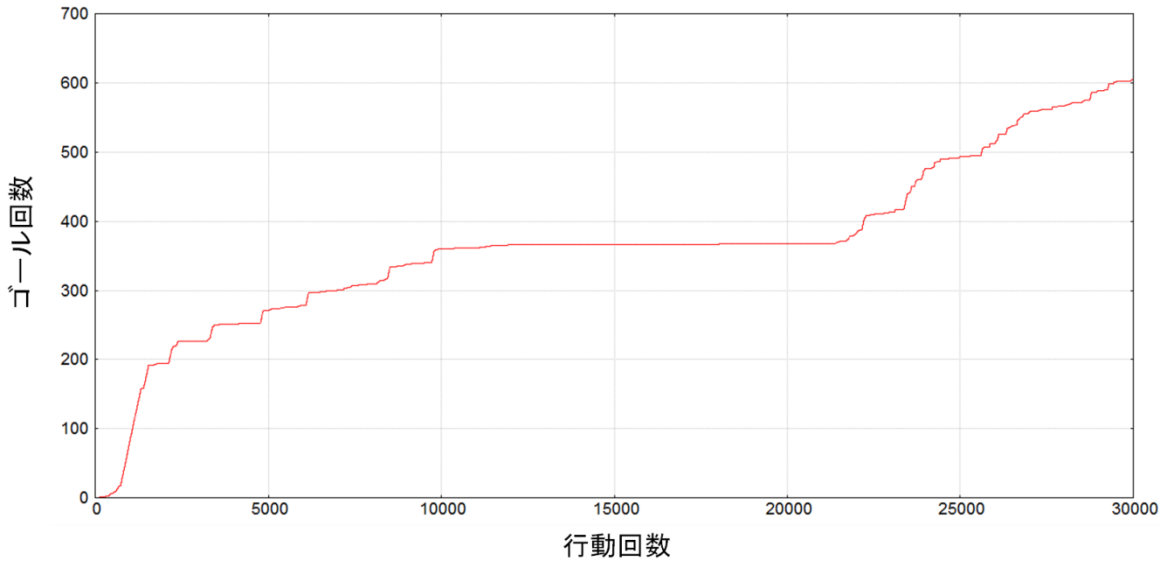

図 4.43　エージェントの行動回数に対するゴール位置到達回数の推移

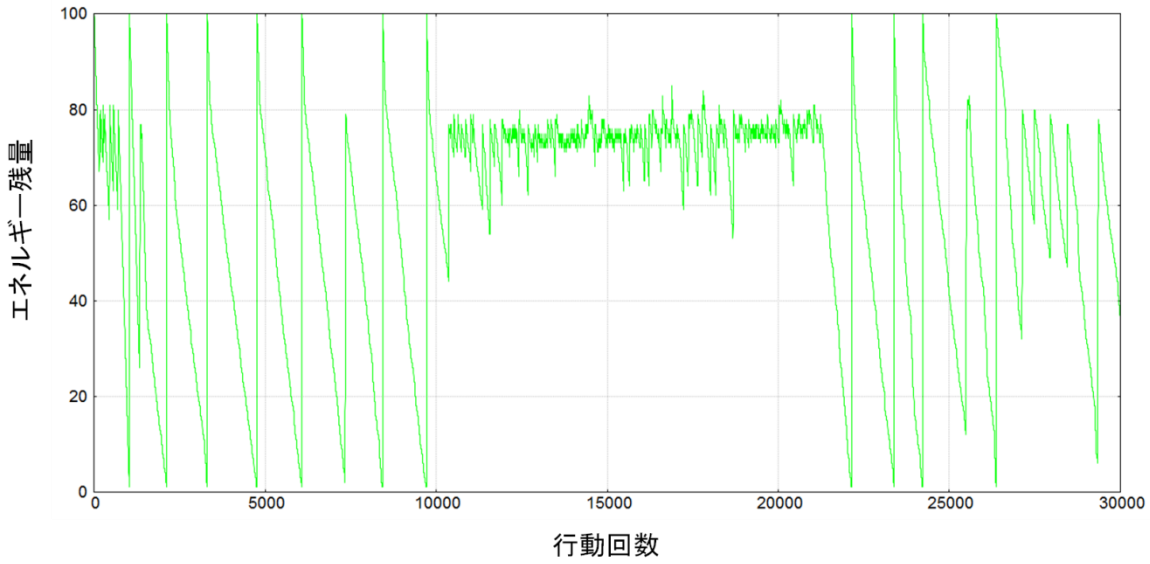

図 4.44　エージェントの行動回数に対するエネルギー残量の推移

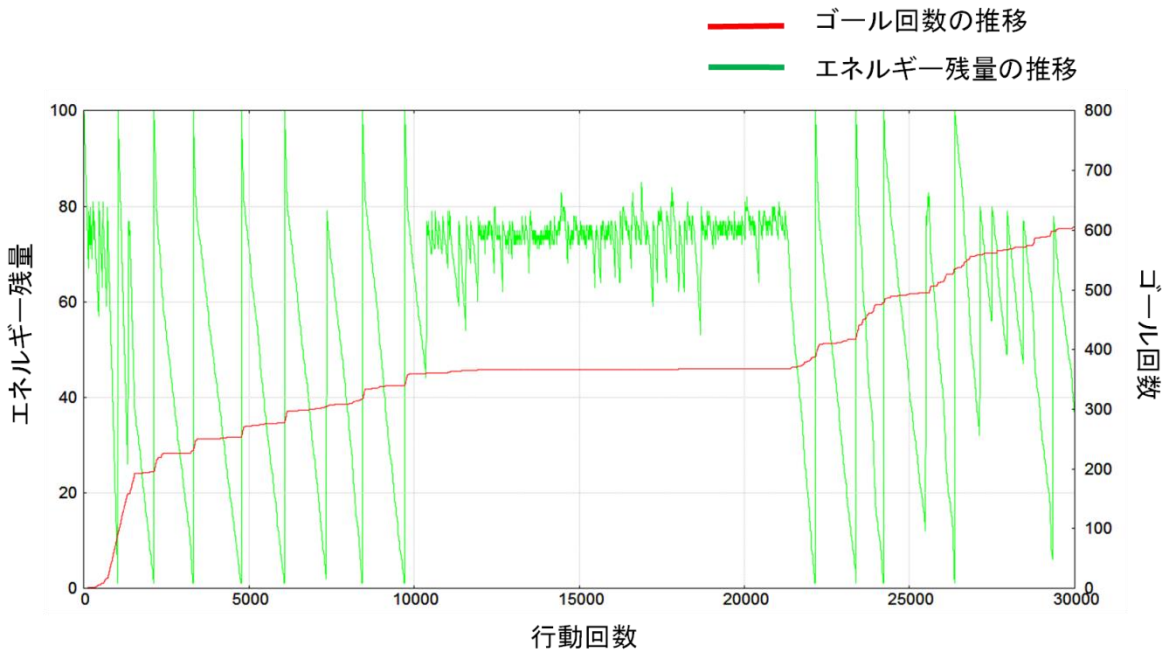

図 4.45　行動回数に対するゴール到達回数とエネルギー残量の推移

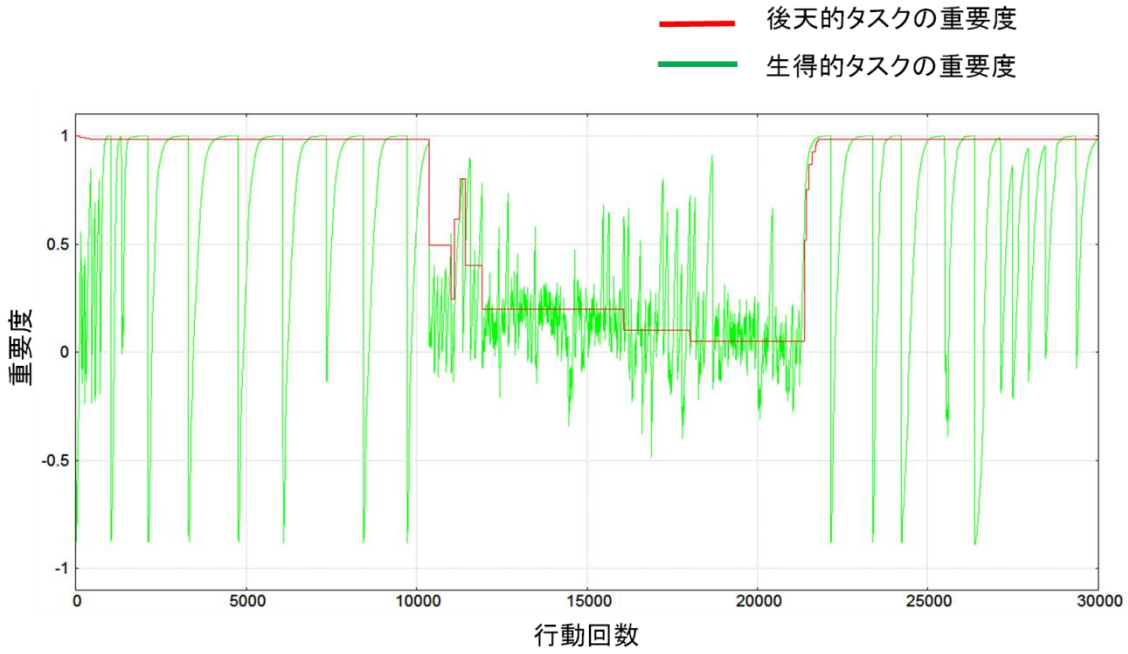

図 4.46　エージェントの行動回数に対する各重要度の変化

図 4.43 のエージェントのゴール回数の推移では，エージェントの行動開始から 10000 回前後まで徐々に増加し，その後 10000~20000 回前後ではほとんど増えていない．また，20000~30000 回前後では，再び徐々に増加している．全体でのゴール回数は 600 回前後となっている．

次に図 4.44 のエージェントのエネルギー残量の推移では，エージェントの行動開始から 10000 回前後までは何度もエネルギー残量が 0 となっている．その後 10000~20000 回前後ではエネルギー残量 80~60 を保っており一度もエネルギー残量 0 にはなっていない．また，20000~30000 回前後では，再び何度もエネルギー残量が 0 になっている．

図 4.45 の行動回数に対するゴール到達回数とエネルギー残量の推移については，ゴール回数が増加している部分では，エネルギー残量が何度も 0 となっており，逆にゴール回数がほとんど増加していない部分では，エネルギー残量が保たれており一度もエネルギー残量 0 にはなっていない．

図 4.46 の各重要度の変化については，エージェントの行動開始から 10000 回前後までと 20000~30000 回までは生得的タスクの重要度は－1 から＋1 まで大きく変化しており，後天的タスクの重要度はほぼ＋1 となっている．また，10000~20000 回前後では生得的タスクの重要度が 0.2 付近を中心に前後しており，後天的タスクの重要度は徐々に下がり最終的には 0 付近まで下がっている．

次に，行動回数 0~10000 回まではゴールする度に“＋１”を与え，10001~2000 回まではゴール時に 2 分の 1 の確率で“＋１”，20001~30000 回では再びゴールする度に“＋0.2”を与える．このときのエージェントの行動回数に対するボール位置到達回数の推移を図 4.47 に，行動回数に対するエネルギー残量の推移を図 4.48 に示す．また，行動回数に対するゴール位置到達回数とエネルギー残量の推移を重ねあわせたものを図 4.49 に，行動回数に対する各重要度の変化を図 4.50 に示す．

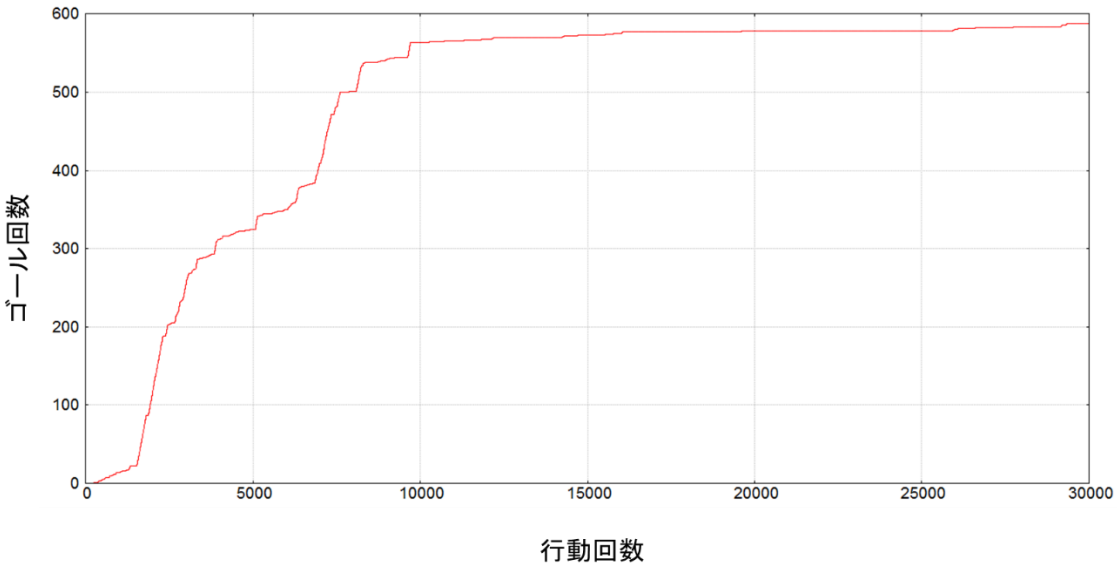

図 4.47　エージェントの行動回数に対するゴール位置到達回数の推移

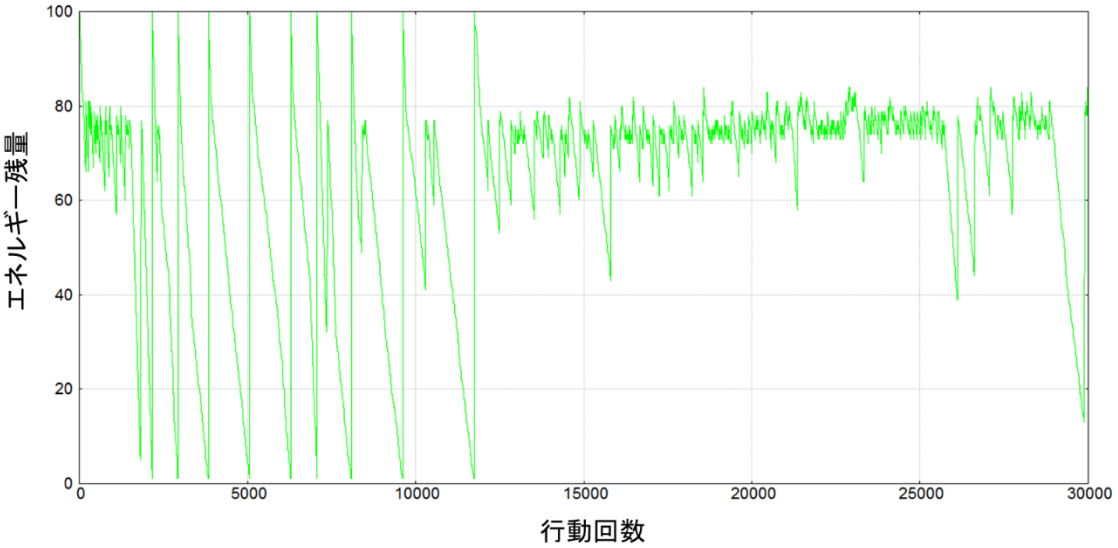

図 4.48　エージェントの行動回数に対するエネルギー残量の推移

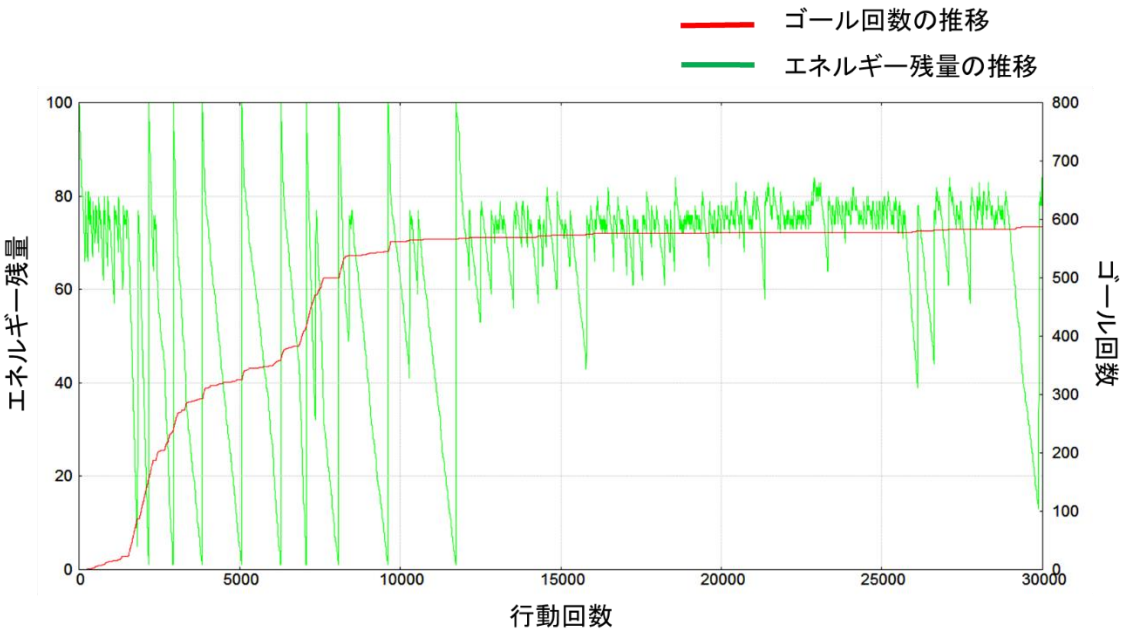

図 4.49　行動回数に対するゴール到達回数とエネルギー残量の推移

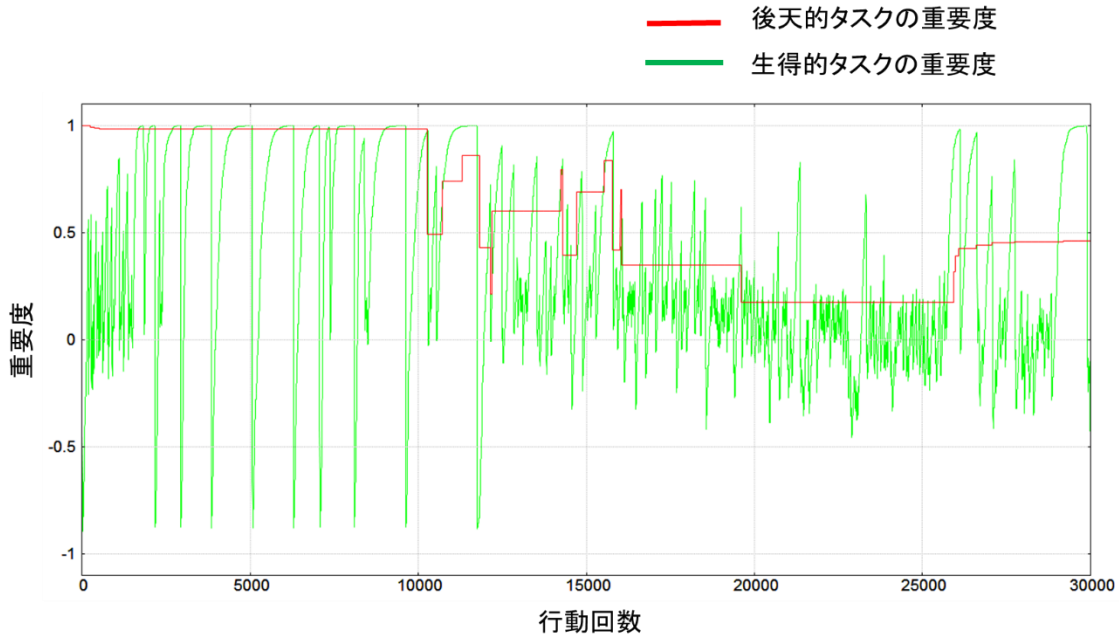

図 4.50　エージェントの行動回数に対する各重要度の変化

図 4.47 のエージェントのゴール回数の推移では，エージェントの行動開始から 10000 回前後まで徐々に増加し，その後 10000~30000 回前後ではほとんど増えていない．全体でのゴール回数は 600 回前後となっている．

次に図 4.48 のエージェントのエネルギー残量の推移では，エージェントの行動開始から 10000 回前後までは何度もエネルギー残量が 0 となっている．その後 10000~25000 回前後ではエネルギー残量 80~60 を保っており一度もエネルギー残量 0 にはなっていない．また，25000~30000 回前後では，エネルギーの減少幅が増えている．

図 4.49 の行動回数に対するゴール到達回数とエネルギー残量の推移については，ゴール回数が増加している部分では，エネルギー残量が何度も 0 となっており，逆にゴール回数がほとんど増加していない部分では，エネルギー残量が保たれており一度もエネルギー残量 0 にはなっていない．

図 4.50 の各重要度の変化については，エージェントの行動開始から 10000 回前後までは生得的タスクの重要度は－1 から＋1 まで大きく変化しており，後天的タスクの重要度はほぼ＋1 となっている．また，10000~25000 回前後では生得的タスクの重要度が 0.2 付近を中心に前後しており，後天的タスクの重要度は徐々に下がり最終的には 0.2 付近まで下がっている．25000 回以降は，後天的タスクの重要度が 0.5 付近まで上昇している．

最後に，行動回数 0~10000 回まではゴールする度に“＋１”を与え，10001~2000 回まではゴール時に 2 分の 1 の確率で“＋１”，20001~30000 回では再びゴールする度に“－1”を与える．このときのエージェントの行動回数に対するボール位置到達回数の推移を図 4.51 に，行動回数に対するエネルギー残量の推移を図 4.52 に示す．また，行動回数に対するゴール位置到達回数とエネルギー残量の推移を重ねあわせたものを図 4.53 に，行動回数に対する各重要度の変化を図 4.54 に示す．

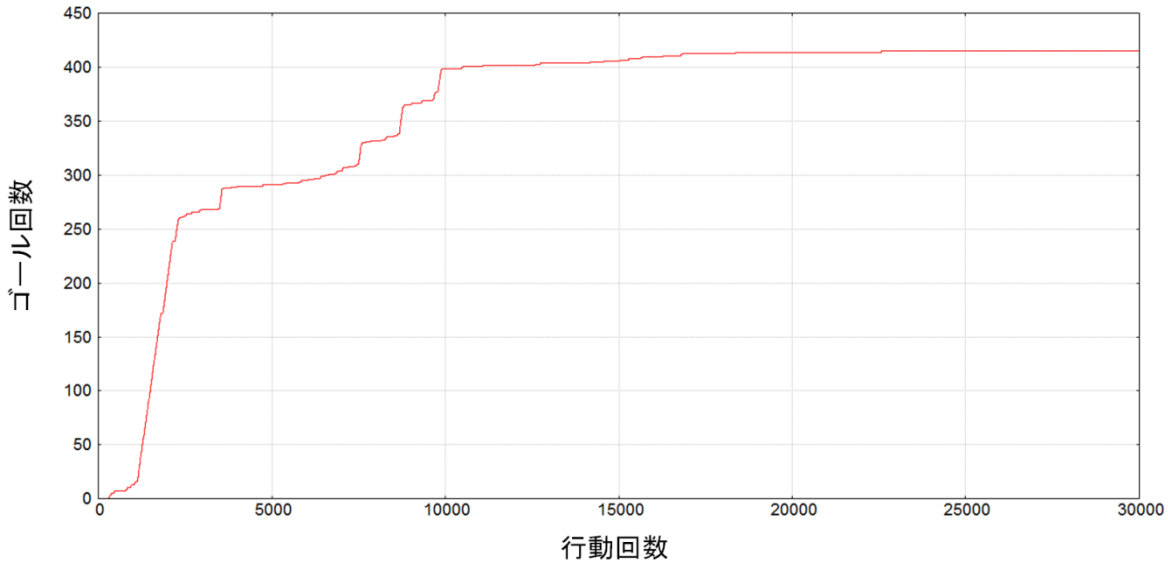

図 4.51　エージェントの行動回数に対するゴール位置到達回数の推移

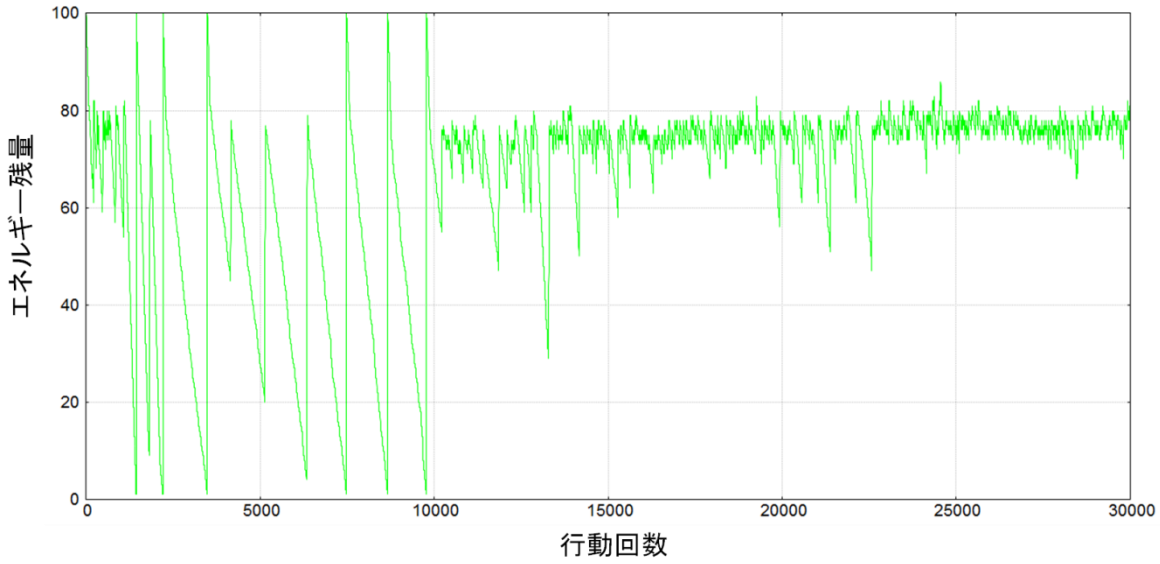

図 4.52　エージェントの行動回数に対するエネルギー残量の推移

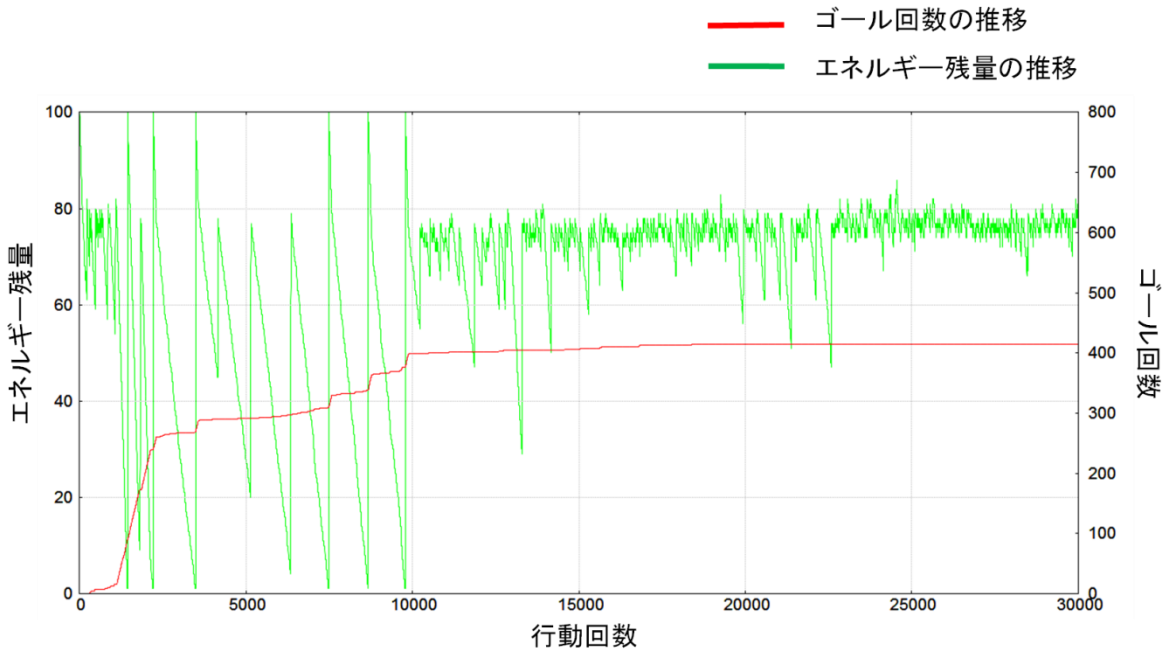

図 4.53　行動回数に対するゴール到達回数とエネルギー残量の推移

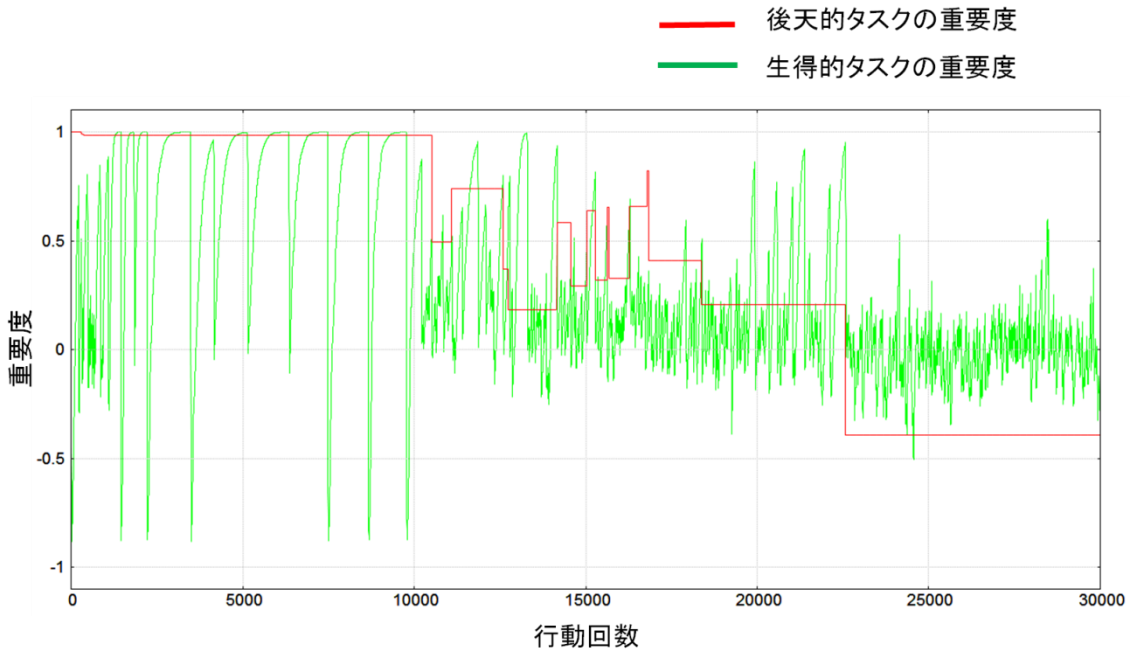

図 4.54　エージェントの行動回数に対する各重要度の変化

図 4.51 のエージェントのゴール回数の推移では，エージェントの行動開始から 10000 回前後まで徐々に増加し，その後 10000~30000 回前後ではほとんど増えていない．全体でのゴール回数は 400 回前後となっている．

次に図 4.52 のエージェントのエネルギー残量の推移では，エージェントの行動開始から 10000 回前後までは何度もエネルギー残量が 0 となっている．その後 10000~22000 回前後ではエネルギー残量 80~60 を保っており一度もエネルギー残量 0 にはなっていない．また，22000~30000 回前後では，エネルギーの減少幅が更に減っている．

図 4.53 の行動回数に対するゴール到達回数とエネルギー残量の推移については，ゴール回数が増加している部分では，エネルギー残量が何度も 0 となっており，逆にゴール回数がほとんど増加していない部分では，エネルギー残量が保たれており一度もエネルギー残量 0 にはなっていない．

図 4.54 の各重要度の変化については，エージェントの行動開始から 10000 回前後までは生得的タスクの重要度は－1 から＋1 まで大きく変化しており，後天的タスクの重要度はほぼ＋1 となっている．また，10000~22000 回前後では生得的タスクの重要度が 0.2 付近を中心に前後しており，後天的タスクの重要度は徐々に下がり最終的には 0.2 付近まで下がっている．22000 回以降は，後天的タスクの重要度が－0.4 付近まで減少している．

４．５　考察

シミュレーション実験で検証した実験パターン毎に，結果から考察を行う．

４．５．１　後天的タスクに対して人間が＋１を与えた実験についての考察

後天的タスクにおいて，エージェントがゴールする度に人間が＋１を与えた場合についての考察を行う．

まず，生得的タスクの重要度パラメータの設定が$\delta = -0.2, \sigma = 10$であったときの行動回数に対するゴール到達回数とエネルギー残量の推移（図 4.9）について考える．行動回数 0~6000 回付近までゴール回数が急激に増加し，エネルギー残量が何度も 0 となっているのは，エージェントが各タスクの学習を十分に行えていないために，エネルギー充電ポイントの存在がわからずゴール到達が優先されていると考えられる．それ以降は，エネルギー残量 50 前後を上限として，エネルギー残量が減る度に再び 50 前後まで回復している．同時にエージェントのゴール回数も増加していることから，エージェントがゴールへ向かう度にエネルギーが消費されエネルギー残量が低下し，その低下を補うために充電ポイントへ向かいエネルギー残量を回復していると考えられる．エージェントが保っているエネルギー残量が 50 付近であるのは，生得的タスクの重要度算出パラメータがエネルギー残量 50 を基準に変化するためである．

また，図 4.10 のエージェントの行動回数に対す各重要度の変化から，後天的タスクの重要度はほぼ 1 に収束している．生得的タスクの重要度については，エージェントの学習がある程度進んだ行動回数 6000 回以降，重要度－0.5 から＋1 の間を大きく上下している．後天的タスクの重要度が＋１に収束するのは，人間が数値を与える頻度が高く，その値も最大の＋１となっているためである．エージェントの行動は，後天的タスクの重要度が高いので，全体的にゴール到達重視となっているが，エネルギー残量がある程度減った場合には，エネルギー充電ポイントでのエネルギー残量回復も行えている．

次に，生得的タスクの重要度パラメータの設定が$\delta = -0.13, \sigma = 10$であったときの行動回数に対するゴール到達回数とエネルギー残量の推移（図 4.13）について考える．行動回数 16000~25000 回付近でエネルギー残量が 80 前後に保持されているが，その他の部分ではゴール回数が増加し，それに伴ってエネルギー残量が何度も 0 になっている．エージェントが保持しようとするエネルギー残量が 80 前後であるのは，生得的タスクの重要度算出に関するパラメータがエネルギー獲得に積極的な設定であるので，エネルギー残量 80 と高い基準でエネルギーの保持しようとした結果である．

また，図 4.14 のエージェントの行動回数に対す各重要度の変化から，後天的タスクの重

要度はほぼ 1 に収束している．生得的タスクの重要度については，エージェントの学習がある程度進んだ後は，重要度－0.5 から＋1 の間を大きく上下している．後天的タスクの重要度が＋ 1 に収束するのは，人間が数値を与える頻度が高く，その値も最大の＋ 1 となっているためである．エージェントの行動は，後天的タスクの重要度が高いので，全体的にゴール到達重視となっている．また，エネルギー残量がある程度減った場合には，エネルギー残量を 80 前後まで回復している．

最後に，生得的タスクの重要度パラメータの設定が$\delta = -0.3, \sigma = 10$であったときの行動回数に対するゴール到達回数とエネルギー残量の推移（図 4.17）について考える．エネルギー残量の傾向としては，40 から 20 の間で保持されているが，エネルギー残量が減少した時の回復が間に合わずに何度もエネルギー残量 0 になっている．エネルギー残量を保持している基準が 40 前後と低いのは，生得的タスクの重要度算出に関するパラメータがエネルギー獲得に消極的な設定であるのためである．

また，図 4.18 のエージェントの行動回数に対す各重要度の変化から，後天的タスクの重要度はほぼ 1 に収束している．生得的タスクの重要度については，重要度－1 から＋1 の間を大きく上下している．後天的タスクの重要度が＋ 1 に収束するのは，人間が数値を与える頻度が高く，その値も最大の＋ 1 となっているためである．エージェントの行動は，後天的タスクの重要度が高いので，全体的にゴール到達重視となっている．エネルギー残量については，保持しようとするエネルギー残量が低いので，エネルギー残量の回復が間に合っていない．

上記 3 パターンの考察を踏まえてまとめると，人間が与える数値が高くその頻度も多いと，後天的タスクの重要度がほぼ最高となり，エージェントの行動は後天的タスク重視となる．ただし，生得的タスクの重要度の設定次第では，エネルギー残量が減少した場合に，エネルギー充電ポイントへ向かいエネルギーを回復する行動も可能である．

４．５．２　後天的タスクに対して人間が＋0.2 を与えた実験についての考察

後天的タスクにおいて，エージェントがゴールする度に人間が＋0.2 を与えた場合についての考察を行う．

まず，生得的タスクの重要度パラメータの設定が$\delta = -0.2, \sigma = 10$であったときの行動回数に対するゴール到達回数とエネルギー残量の推移（図 4.21）について考える．全体的にエネルギー残量 50 前後を上限として，エネルギー残量が減る度に再び 50 前後まで回復している．エージェントのゴール回数については，ほとんど増加していない．エージェントが保っているエネルギー残量が 50 付近であるのは，生得的タスクの重要度算出パラメータがエネルギー残量 50 を基準に変化するためである．

また，図 4.22 のエージェントの行動回数に対す各重要度の変化から，後天的タスクの重要度はほぼ 0.5 に収束している．生得的タスクの重要度については，重要度－0.5 から＋1 の間を大きく上下している．後天的タスクの重要度が＋0.5 に収束するのは，人間が数値を与える頻度は高いが，与える値が＋0.2 と低いためである．エージェントの行動は，後天的タスクの重要度より，生得的タスクの重要度が高くなることが多いので，全体的にエネルギー獲得重視となっている．エネルギー残量がある程度多くなれば，後天的タスクの重要度が生得的タスクの重要度より高くなることもあるが，ゴールに到達する前に，再び生得的タスクの重要度のほうが上回るので，ゴール到達はあまり達成されていない．

次に，生得的タスクの重要度パラメータの設定が$\delta = -0.13, \sigma = 10$であったときの行動回数に対するゴール到達回数とエネルギー残量の推移（図 4.25）について考える．全体的にエネルギー残量が 80 前後を上限として，エネルギー残量が減る度に再び 80 前後まで回復している．エージェントが保持しようとするエネルギー残量が 80 前後であるのは，生得的タスクの重要度算出に関するパラメータがエネルギー獲得に積極的な設定であるので，エネルギー残量 80 と高い基準でエネルギーの保持しようとした結果である．

また，図 4.26 のエージェントの行動回数に対す各重要度の変化から，後天的タスクの重要度はほぼ＋0.5 に収束している．生得的タスクの重要度については，重要度 0 から＋1 の間を上下している．後天的タスクの重要度が＋0.5 に収束するのは，人間が数値を与える頻度は高いが，与える値が＋0.2 と低いためである．エージェントの行動は，後天的タスクの重要度より，生得的タスクの重要度が高くなることが多いので，全体的にエネルギー獲得重視となっており保持されるエネルギー残量の基準も高い．

最後に，生得的タスクの重要度パラメータの設定が$\delta = -0.3, \sigma = 10$であったときの行動回数に対するゴール到達回数とエネルギー残量の推移（図 4.29）について考える．エネルギー残量の傾向としては，40 から 20 の間で保持されているが，エネルギー残量が減少した時の回復が間に合わずに何度かエネルギー残量 0 になっている．エネルギー残量を保持し

ている基準が 40 前後と低いのは，生得的タスクの重要度算出に関するパラメータがエネルギー獲得に消極的な設定であるのためである．

また，図 4.30 のエージェントの行動回数に対す各重要度の変化から，後天的タスクの重要度はほぼ＋0.5 に収束している．生得的タスクの重要度については，重要度－0.5 から＋1 の間を大きく上下している．後天的タスクの重要度が＋0.5 に収束するのは，人間が数値を与える頻度は高いが，与える値が＋0.2 と低いためである．エージェントの行動は，後天的タスクの重要度より，生得的タスクの重要度が高くなることが多いので，全体的にエネルギー獲得重視となっており保持されるエネルギー残量の基準も高い．しかし，エネルギー残量については，保持しようとするエネルギー残量が低いので，エネルギー残量の回復が間に合っていない部分が見られる．

上記３パターンの考察を踏まえてまとめると，人間が数値を与える頻度は多いが値が低いと，後天的タスクの重要度が低下し，生得的タスクの重要度が後天的タスクの重要度を上回ることが増える．従って，エージェントの行動は生得的タスク重視となり，ゴール到達はほとんど達成できない．

４．５．３　後天的タスクに対して人間が－１を与えた実験についての考察

後天的タスクにおいて，エージェントがゴールする度に人間が－１を与えた場合についての考察を行う．

まず，生得的タスクの重要度パラメータの設定が$\delta = -0.2, \sigma = 10$であったときの行動回数に対するゴール到達回数とエネルギー残量の推移（図 4.33）について考える．全体的にエネルギー残量 50 前後を上限として，エネルギー残量が減る度に再び 50 前後まで回復している．エージェントのゴール回数については，ほとんど増加していない．エージェントが保っているエネルギー残量が 50 付近であるのは，生得的タスクの重要度算出パラメータがエネルギー残量 50 を基準に変化するためである．

また，図 4.34 のエージェントの行動回数に対す各重要度の変化から，後天的タスクの重要度はほぼ－１に収束している．生得的タスクの重要度については，重要度－0.5 から＋1 の間を大きく上下している．後天的タスクの重要度が－１に収束するのは，人間が数値を与える頻度は高いが，与える値が－１と最低であるためである．エージェントの行動は，後天的タスクの重要度より，生得的タスクの重要度が高くなることが多いので，全体的にエネルギー獲得重視となっている．エネルギー残量がある程度多くなれば，後天的タスクの重要度は負の値であるので，より後天的タスクの達成を避けるような行動になる．

従って，エージェントの行動はエネルギー獲得重視となり，ゴール達成は行わなくなっている．

次に，生得的タスクの重要度パラメータの設定が$\delta = -0.13, \sigma = 10$であったときの行動回数に対するゴール到達回数とエネルギー残量の推移（図 4.37）について考える．全体的にエネルギー残量が 80 前後を上限として，エネルギー残量が減る度に再び 80 前後まで回復している．エージェントが保持しようとするエネルギー残量が 80 前後であるのは，生得的タスクの重要度算出に関するパラメータがエネルギー獲得に積極的な設定であるので，エネルギー残量 80 と高い基準でエネルギーの保持しようとした結果である．

また，図 4.38 のエージェントの行動回数に対す各重要度の変化から，後天的タスクの重要度はほぼ－１に収束している．生得的タスクの重要度については，重要度－0.5 から＋1 の間を大きく上下している．後天的タスクの重要度が－１に収束するのは，人間が数値を与える頻度は高いが，与える値が－１と最低であるためである．エージェントの行動は，後天的タスクの重要度より，生得的タスクの重要度が高くなることが多いので，全体的にエネルギー獲得重視となっている．エネルギー残量がある程度多くなれば，後天的タスクの重要度は負の値であるので，より後天的タスクの達成を避けるような行動になる．
従って，エージェントの行動はエネルギー獲得重視となり，ゴール達成は行わなくなっている．

最後に，生得的タスクの重要度パラメータの設定が$\delta = -0.3, \sigma = 10$であったときの行動回数に対するゴール到達回数とエネルギー残量の推移（図 4.41）について考える．エネルギー残量の傾向としては，40 から 20 の間で保持されているが，エネルギー残量が減少した時の回復が間に合わずに何度かエネルギー残量 0 になっている．エネルギー残量を保持している基準が 40 前後と低いのは，生得的タスクの重要度算出に関するパラメータがエネルギー獲得に消極的な設定であるのためである．

また，図 4.42 のエージェントの行動回数に対す各重要度の変化から，後天的タスクの重要度はほぼ－１に収束している．生得的タスクの重要度については，重要度－0.5 から＋1 の間を大きく上下している．後天的タスクの重要度が－１に収束するのは，人間が数値を与える頻度は高いが，与える値が－１と最低であるためである．エージェントの行動は，後天的タスクの重要度より，生得的タスクの重要度が高くなることが多いので，全体的にエネルギー獲得重視となっている．エネルギー残量がある程度多くなれば，後天的タスクの重要度は負の値であるので，より後天的タスクの達成を避けるような行動になる．
従って，エージェントの行動はエネルギー獲得重視となり，ゴール達成は行わなくなっている．

上記３パターンの考察を踏まえてまとめると，人間が数値を与える頻度は高いが値が負の値であると，後天的タスクの重要度が負の値となり，後天的タスクの達成を避けるような行動を取る．従って，エージェントの行動は生得的タスク重視となり，ゴール到達を避けるような行動を取る．

４．５．４　重要度を一定行動回数毎に変化させた場合の実験についての考察

後天的タスクに対して人間が数値を与える頻度と値を途中で変化させエージェントの行動変化を検証した実験について考察を行う．

まずは，生得的タスクの重要度パラメータの設定が$\delta = -0.13, \sigma = 10$で，人間はエージェントがゴールする度に＋１，2 分の 1 の確率で＋１，＋１と順に変化させた場合について考察する．行動回数に対するゴール到達回数とエネルギー残量の推移については，行動回数 0~10000 回まではゴール回数が上昇いており，それに伴いエネルギー残量は減少し何度も 0 となっている．同時に各タスクの重要度の変化を見ると，後天的タスクの重要度が＋１となり，生得的タスクの重要度を超えている．従って，エージェントの行動は後天的タスクを重視するようになり，ゴール回数が上昇している．行動回数 10000~20000 回付近では，ゴール回数がほとんど増えずエネルギー残量は 80 を上限として，エネルギー残量が減少した場合には 80 付近まで回復している．この時の各タスクの重要度を見ると，後天的タスクの重要度が＋１から下がり最終的には 0 付近まで減少している．これは，人間がエージェントに対して数値を与える確率が減ったためである．この結果，生得的タスクの重要度が後天的タスクの重要度を上回り，エージェントの行動は生得的タスクを重視する結果となっている．行動回数 20000~30000 回では再び，ゴール回数が上昇いており，それに伴いエネルギー残量は減少し何度も 0 となっている．同時に各タスクの重要度の変化を見ると，後天的タスクの重要度が＋１となり，生得的タスクの重要度を超えている．従って，人間からのタスク達成の要求が高い場合には，エネルギー残量が減少した場合でも，人間に与えられたタスクの達成を優先し，人間からのタスク達成の要求が低い場合には，自身のエネルギー残量の確保を優先できていると考える．つまり，エージェントは生得的タスクと後天的タスクの重要度に合わせて行動可能であると判断できる．

次に，生得的タスクの重要度パラメータの設定が$\delta = -0.13, \sigma = 10$で，人間が与える数値をゴール時に毎回＋１，ゴール時に 2 分の 1 の確率で＋１，ゴール時に＋0.2 と変化させた場合について考察する．行動回数に対するゴール到達回数とエネルギー残量の推移については，行動回数 0~12000 回付近まではゴール回数が上昇いており，それに伴いエネルギー残量は減少し何度も 0 となっている．同時に各タスクの重要度の変化を見ると，後天的タスクの重要度が＋１となり，生得的タスクの重要度を超えている．従って，エージェントの行動は後天的タスクを重視するようになり，ゴール回数が上昇していると判断できる．行動回数 12000~25000 回付近では，ゴール回数がほとんど増えずエネルギー残量は 80 を上限として，エネルギー残量が減少した場合には 80 付近まで回復している．この時の各タスクの重要度を見ると，後天的タスクの重要度が＋１から下がり最終的には 0.2 付近まで減少している．これは，人間がエージェントに対して数値を与える確率が減ったためである．

この結果，生得的タスクの重要度が後天的タスクの重要度を上回り，エージェントの行動は生得的タスクを重視する結果となっている．行動回数 25000~30000 回では後天的タスクの重要度が 0.5 付近まで上昇し，その結果としてエネルギー残量が 80 付近から減少する減少幅が大きくなっている．ここで，後天的タスクの重要度が再び上昇したのは，人間が与える仮想的な報酬の値が増えたためである．また，人間が与える値を行動回数 20000 回から変更しているのにもかかわらず，重要度が変化したのが 25000 回付近となっているのは，その間にエージェントがゴール位置へ到達していないためである．

全体的に見ると，後天的タスクの重要度が 0.5 付近となると，ゴールへ到達しつつ，エネルギー残量が減少した場合には，エネルギー残量を回復する行動を取ることができる．

最後に，生得的タスクの重要度パラメータの設定が$\delta = -0.13, \sigma = 10$で，人間が与える数値をゴール時に毎回＋１，ゴール時に 2 分の 1 の確率で＋１，ゴール時に－１と変化させた場合について考察する．行動回数に対するゴール到達回数とエネルギー残量の推移については，行動回数 0~10000 回付近まではゴール回数が上昇いており，それに伴いエネルギー残量は減少し何度も 0 となっている．同時に各タスクの重要度の変化を見ると，後天的タスクの重要度が＋１となり，生得的タスクの重要度を超えている．従って，エージェントの行動は後天的タスクを重視するようになり，ゴール回数が上昇していると判断できる．行動回数 10000 以降では，ゴール回数がほとんど増えずエネルギー残量は 80 を上限として，エネルギー残量が減少した場合には 80 付近まで回復している．この時の各タスクの重要度を見ると，後天的タスクの重要度が＋１から下がり一度 0.2 付近で収束し，その後さらに減少して，最終的には－0.4 付近まで下がっている．一度 0.2 付近に収束しているのは，人間がエージェントに対して数値を与える確率が減ったためである．その後，人間がエージェントに対して与える値が－1 となったので，最終的な重要度は－0.4 付近まで下がっている．この結果，生得的タスクの重要度が後天的タスクの重要度を上回り，エージェントの行動は生得的タスクを重視する結果となっている．また，エネルギー残量については，行動回数 22000 回以降のほうが，行動回数 10000~22000 回よりもエネルギー残量の減少幅が小さくなっている．これは，後天的タスクの重要度が負の値になったことで，よりゴール達成を避けるような行動になり，エネルギーを消費する行動が減ったためだと考えられる．

第５章　結論

５．１　全体を通してのまとめ

本研究では，強化学習が適用された汎用的なロボットに注目し，このようなロボットに対して複数のタスクを与えた場合には，タスク毎の学習と状況にあわせた行動選択が必要であることを示した．そして，タスク毎の状況に合わせたロボットの行動決定を行う学習手法の実現を目的とした．提案手法では，タスク毎に別々の学習空間で学習を行い，新たなタスクの追加や不要となったタスクの削除を容易にした．また，複数の学習空間によって蓄積される行動価値とタスク毎の重要度を基に最終的なロボットの行動選択を行うことで状況に応じた行動選択を可能にした．

シミュレーション実験では，実際にロボットに２つの異なるタスクを与え，提案手法の有用性と重要度の変化による行動の変化について検証した．実験結果として，生得的タスクの重要度が高くなった場合には，ロボットが自身の存続を維持するような行動を優先的に選択し，後天的タスクの重要度が高くなった場合には，人間に与えられたタスクを達成するための行動を優先的に選択した．各タスクの重要度が変化しても，ロボットの行動学習自体の再学習は必要なく重要度に応じてロボットの行動傾向のみが変化する．以上のことから，本研究ではロボットに対して複数のタスクを与えた場合でも，各タスクの重要性から状況に応じた行動選択を実現できたと判断する．

５．２　今後の課題

本節では，本研究で今後想定される課題について説明する．

５．２．１　３つ以上のタスク下での検証実験

今回のシミュレーション実験では，生得的タスクと後天的タスクをそれぞれ１つずつ与え検証した．しかし，本手法ではロボットに与えられるタスクが更に増えた場合も想定している．従って，今後の課題としてロボットに対して３つ以上のタスクを与えた場合の検証実験を行う必要がある．今回行った実験設定を拡張したような実験環境でシミュレーション実験を考えるならば，生得的タスクの数を増やす場合は，危険回避等のタスク等が考えられる．また，後天的タスクの数を増やす場合には異なるゴール位置への到達等が考えられる．

５．２．２　タスクの途中追加と途中削除の検証実験

今回のシミュレーション実験では，実験開始から終了までロボットが有するタスク数の変化はない．しかし，提案手法ではタスク毎に報酬関数と学習空間を１つずつ設定しているので，タスクの追加と削除が容易であると考えられる．これを検証するためにも，実験中に新たなタスクの追加や削除を行った場合でも，その時々のタスクの組み合わせと重要度の関係からロボットの行動選択が可能であることを検証する必要がある．

５．２．３　実機を使用した検証実験

本研究では，シミュレーションを用いた実験しか行っていないが，最終的な目標としては実機に提案手法を適用することである．しかし，現段階では，実機に本手法を適用した場合に本手法が有効に機能するかどうかわからない．また，実機を用いる場合には，実機の作成やバッテリー残量の測定，人間から仮想的な報酬を受け取るためのチャンネルなど考慮すべき問題が多々あると考えられる．しかし，実機に適用することの問題点を検証するためには実機を用いた検証実験が必要である．

謝辞

本論文を結ぶにあたり，日頃より懇切なるご指導を賜りました倉重健太郎先生に深く感謝の意を表します．また，ご指導，ご助言をいただいた畑中雅彦先生，佐賀聡人先生，本田泰先生，に感謝の意を表します．そして，論文の査読や助言をしていただいた認知ロボティクス研究室の木島康隆さん，中南義典さん，宮崎愛央さん，梅津祐介さん，北山直樹さん，渋谷和さん，杉本大志さん，沼田利伸君，高泉昇太郎君，二階堂芳君，木村 敏久君，挟間 重直君，平間経太君，片山和宣君，小橋遼君，千葉秀平君に感謝いたします．

参考文献

[1] R.S. Sutton and A.G. Barto “Reinforcement learning An Introduction.” 1998
邦訳：三上 貞芳，皆川 雅章 “強化学習”，森北出版，2000

[2] 田中 文英，山村 雅幸“２次元のマルチタスク学習を行う強化学習エージェント”，人工知能学会全国大会論文集，Vol.15, No.2,2001

[3] 田中 文英，山村 雅幸 ”MDP 集団の上におけるマルチタスク強化学習”，電気学会論文誌.C，電子・情報・システム部門誌，Vol.123, No.5, pp.1004-1011, 2003

[4] 内部 英治, 銅谷 賢治 “複数の報酬によって与えられる拘束のもとでも強化学習”, 電子情報通信学会技術研究報告. NC, ニューロンコンピューティング, Vol.106, no.102, pp.1-6, 2006

[5] 小野寺 道寛，鈴木 輝彦，太原 育夫 “複数の目標を持つタスクに対する適格度トレースを用いた強化学習”, 情報科学フォーラム講演論文集，Vol.10, No.2, pp.509-514, 2011

[6] 青木 圭，佐久間 淳，浅井 孝宣，池田 心，小林 重信 “目標指向型探査に基づく多目的強化学習と２足歩行ロボットへの応用”，システム制御情報学会論文誌，Vol.18，No.10, pp.352-360, 2005

[7] M.Humphrys “Action Selection Methods Using Reinforcement Learning” , PhD thesis, University of Cambridge, 1997

[8] 高橋 泰岳，浅田 稔 “複数の学習器の階層的構築による行動獲得” ，日本ロボット学会誌，Vol.18, No.7, pp.1040-1046, 2000

[9] 高橋 泰岳，浅田 稔 “階層型学習機構における状態行動空間の構成”， 日本ロボット学会誌，Vol.21, No.2, pp.164-171, 2003

[10] 伊賀上 大輔，市村 匠 ”複数ターゲットによる階層型モジュラー強化学習結果からの知識獲得”，第 17 回日本知能情報ファジィ学会中国四国支部大会予稿集，2012

[11] 加藤 龍憲，鈴木 昭二，浅田 稔 “複数の報酬による強化学習を用いたサッカーロボッ

トのゴール守備行動の獲得”，ロボティクスシンポジア予稿集，Vol.4, pp.289-294, 1999

[12] Z.Gabor，Z.Kamlar and C.Szepesvari ”Multi-criteria reinforcement learning”，Proc. Of International Conference on Machien Learning, 1998

[13] D.-O. Kang and Z.Bien ”Multiobjective control problems by reinforcement learning”，HANDBOOK OF LEARNING AND APPROXIMATE DYNAMIC PROGRAMING, IEEE Press, chapter 17, pp. 433-461, 2004

[14] 上岡 拓未，内部 英治，銅谷 賢治 “複数の価値関数を用いた多目的強化学習”,電子情報学会信学技報，Vol.105, No.658, pp.127-132, 2006

[15] Watkins, C. and Dayan, P. “Technical note:Q-Learning, Machine Learning”, Vol. 8, pp. 279-292 ,1992

[16] 廣瀬 清人，菱沼 典子，印東 桂子 “マズローの基本的欲求の階層図への原点からの新解釈”，聖路加看護大学紀要, No.35, 2009

[17] 株式会社 AWG Minagine キャリア・サポート “モティペディア”，http://cs-d.awg.co.jp/csd/motivation

研究業績

[1] 三浦丈典，倉重健太郎，” Proposal of learning method which selects objectives based on the state”， IEEE SSCI 2013：IEEE Symposium Series on Computational Intelligence，Grand Copthorne Waterfront Hotel，シンガポール，2013．4．16-19