追跡問題における 状態数とメモリ量の削減に着目した Q 学習*

伊木美太輔[†]・松本啓之亮[†]・森 直樹[†]

Q-Learning Focusing on Partial States and Reducing Amount of Memory of Tracking Problems*

Taisuke IKIMI[†], Keinosuke MATSUMOTO[†] and Naoki MORI[†]

Recently, application of a multi agent system is expected from the viewpoint of the parallel and distributed processing of systems. Reinforcement learning attracts attention as an implementing method of the multi agent systems. However, there is a problem that the more the number of agents to deal with increases, the slower the speed of learning becomes. To solve this problem, we propose a new reinforcement learning that can learn quickly and reduce the amount of memory. It tries to increase efficiency of the learning on a tracking problem by using a method paying attention to partial states of two agents among a large number of agents.

1. はじめに

近年, 急速なインターネットやマルチメディアの発展などにより, 情報が分散化かつ巨大化してきており, システムの大規模化や複雑化が問題となっている. そこで, システムの並列処理や分散処理といった観点から, 複数のエージェント [1] がエージェントどうしでやりとりをするマルチエージェントシステムの応用が期待されている. このマルチエージェントシステムにおいて一番重要になるのはエージェントが自ら学習するということである. なぜなら, あらかじめ人間が設計したエージェントでは環境の変化に追従できず, 適切な行動をするのに限界があるためである. この場合, エージェントが自らの経験や他のエージェントから集めた情報から適切な判断基準を学習する必要がある [2].

エージェントの学習の実装方法として強化学習 [3] が注目されている. 目標とする状態がわかっていれば, 報酬を設定するだけで自律的に学習するので, マルチエージェントシステムにおける各エージェントの学習の実装方法として非常に有効な手段といえる. 本研究では, マルチエージェント問題に対して強化学習を適用し, 得ら

れたエージェント間の行動に対して評価する. 問題として, マルチエージェント環境下での協調問題として広く用いられている追跡問題を扱う. ここで, 扱う問題が複雑になり, エージェントの個体数が多くなるにしたがって状態数が指数的に多くなってしまい, 学習の速度が遅くなってしまふといった問題がある.

この問題を解決するために Modular Q-Learning という学習方法 [4] がある. これは, 存在するすべてのエージェントを考慮して状態を決定するのではなく, 特定のエージェントに着目して状態数を減らすという手法である. 状態数が減ったことにより, 学習のスピードが速くなるといった利点がある. しかしながら, この学習方法には 1 体のエージェントが複数の学習器を持つことによって消費メモリ量が多くなってしまふことや, 不完全知覚であることから学習精度が劣化してしまふといった問題がある.

そこで, 本研究では 1 体のエージェントにつき学習器を一つとして消費メモリ量を減らして, 切り換え学習という手法を用いることにより学習精度の劣化を防ぎ, 追跡問題における強化学習の効率化を図ることを目的とした. なお, 本論文における消費メモリ量は, 「学習開始時から終了時まで使用メモリ総量を考慮した, 単位時間当たりの消費メモリ量」を指す.

* 原稿受付 2011 年 12 月 19 日

[†] 大阪府立大学 大学院 工学研究科 Graduate School of Engineering, Osaka Prefecture University; 1-1, Gakuen-cho, Naka-ku, Sakai, Osaka 599-8531, JAPAN

Key Words: reinforcement learning, Q-Learning, tracking problem, partial states, entropy.

2. 強化学習

2.1 強化学習とは

強化学習とは、さまざまな報酬を手がかりとしながら、単位時間あたりの受け取る報酬値を最大化する政策を試行錯誤を通して獲得するものである [3]。学習者がどの行動をとるべきかは教えられず、その代わりに、どの行動をとればよりいっそうの報酬に結びつくかを見いだす必要がある。これにより、今まで誰も思いつかなかったような行動パターンや未知の問題に対しての解決策が見つけれられる可能性がある。

2.2 Q 学習

本研究ではブートストラップ型の手法の一つである Q 学習 [5] に着目した。マルコフ決定過程の環境では、学習率が適切に調整されれば、無限時間での最適解の収束が証明されている。以下に、状態 s_t から s_{t+1} に遷移したときの Q 値の更新式を定義する。

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

ここで、 s は状態、 a は行動、 $Q(s_t, a_t)$ は状態 s_t において行動 a_t をとるときの価値、 $\max_a Q(s_{t+1}, a)$ は遷移先の状態 s_{t+1} における最大 Q 値、 α は学習率 ($0 < \alpha \leq 1$)、 r は報酬、 γ は割引率 ($0 \leq \gamma \leq 1$) である。

3. 追跡問題

3.1 追跡問題の定義

追跡問題は、追跡役の複数のエージェント（ハンタ）が 1 体の逃走役のエージェント（獲物）を追いかけて捕まえるゲームである。本研究における追跡問題の定義を以下に示す。

- ハンタが獲物を捕まえるとは、獲物の現在位置と獲物の周囲 8 マスの計 9 マスのどこかに 3 体以上ハンタが存在することとする。
- フィールドは二次元格子状盤面で、トーラス状の盤面とする。
- 一つのマスに複数のエージェントが存在することを許す。
- 各エージェントは上下左右に 1 マス移動、または停止の 5 通りの行動ができる。
- ハンタは完全な視界を持っており、どの位置に他のハンタと獲物があるかが獲物からの相対座標でわかる。
- 各エージェントが一度ずつ行動する単位時間を“ステップ”という。
- 初期状態から目標状態までを一つの“エピソード”という。

3.2 状態数の爆発

本研究では、この追跡問題に 2. で述べた強化学習を適用する。その際、観測可能な状態 s と可能な行動 a と

の対 (s, a) について、行動をするたびにその行動を評価する。そして、同じ状態が観測されたときには、その評価値を行動選択に利用する。これより、強化学習の効果をも十分に発揮するためには、すべての観測可能な状態・行動対について、十分な回数評価する必要がある [6]。

フィールドのサイズを $m \times m$ 、ハンタ数を n 体とし、各ハンタの位置を絶対座標で知覚できるとき、1 体のハンタが観測できる状態数は m^{2n} となり、各状態において、上下左右と停止の 5 通りの行動ができるので、状態・行動対は $5m^{2n}$ となる。罫の肩にハンタ数の n があることから、ハンタ数が増加するにしたがって指数的に状態数が増加するので、状態数の爆発が起ってしまう。

この状態数の爆発により、学習の速度が遅くなってしまふ。よって、マルチエージェント環境における強化学習の実装において、状態数をいかにして削減していくかが課題となってくる。

3.3 Completely Perceptual Q-Learning

追跡問題における強化学習では、具体的にどのように状態数の爆発が起ってしまうのかを説明する。3.1 で述べたように、ハンタは完全な視界を持っている。この状況下であるハンタの状態を定義するときに、すべてのエージェントを考慮し、その状態を基に Q 学習を適用することを本論文では Completely Perceptual Q-Learning (CPQL) とする。いま、4 体のハンタが獲物からの相対座標でそれぞれ $s_1: (x_1, y_1), s_2: (x_2, y_2), s_3: (x_3, y_3), s_4: (x_4, y_4)$ という位置にすることが知覚できているとする。このとき、ハンタの現在の状態を考慮すると、 $\langle s_1, s_2, s_3, s_4 \rangle$ を考慮しなければならない。

3.4 Modular Q-Learning

状態数の爆発を防ぐために、Modular Q-Learning (MQL) という手法が小野らによって提案されている。この手法では、自分自身と他の 1 体のエージェントから構成される部分状態を用いる。これにより、フィールドサイズ $m \times m$ 、ハンタ数 n 体のときの状態数は m^4 となり、罫の肩が 4 という定数であることから、ハンタ数に影響されないで状態数の爆発を防ぐことができる。

Fig. 1 に 3.3 で説明した 4 体のハンタの例で考えたときの MQL におけるハンタの構成を示す。 s_1 の位置にいるハンタの状態を定義するときに、三つの部分状態 $\langle s_1, s_2 \rangle, \langle s_1, s_3 \rangle, \langle s_1, s_4 \rangle$ を学習に用いる。これらを別々の学習器に渡す。そして各学習器は Q 学習をして、Q 値を行動選択器に渡す。以下に示すように、行動選択器では受け取った Q 値の合計値がより大きな値となる行動 a_s を優先して行動する。

$$a_s = \arg \max_{a \in A} \sum_{i=1}^{n-1} Q^{k,i}(S_{k,i}, a) \quad (2)$$

ここで、 a は行動、 A は行動の集合、 $\arg \max_{a \in A} Q$ は Q を最大にするような引数 a を意味し、 $S_{k,i}$ はエージェ

Fig. 1 Architecture of the modular Q-Learning

Fig. 2 Architecture of the HSCMQL

ント k とそれ以外のエージェント i から構成される部分状態を表す．

このように，MQL は状態数の爆発を防ぐことができる手法ではあるが，欠点もある．一つ目として，部分状態を観測しているために完全知覚ではなく不完全知覚状態に陥ってしまい，学習精度が劣化してしまうことが挙げられる．二つ目として，ハンタ数が n 体のとき部分状態が $n-1$ 個存在し，それぞれに対して一つずつ学習器を割り当てるため，学習器の数が $n(n-1)$ 個となり，ハンタ数が多くなると多数の学習器が必要になるので，消費メモリ量が多くなってしまうことが挙げられる．

4. 提案手法

3.4 で述べた MQL を拡張し，問題点を解決する“Hybrid Switchable Centralized Modular Q-Learning (HSCMQL)”を提案する．

4.1 HSCMQL

HSCMQL では，不完全知覚による精度の劣化を切り換え学習によって解決する．また，この手法は CPQL と MQL の学習器を両方利用するので，メモリ量が膨大になる．そこで，メモリ量が多くなってしまうと指摘した MQL を中央集権型 Modular Q-Learning (中央集権型

MQL) という学習器に変えることにより，メモリ量を削減した．Fig. 2 に 3.3 で説明した 4 体のハンタの例で考えたときの HSCMQL におけるハンタの構成を示す．

MQL の構成である Fig. 1 と比較してみると，学習器が中央集権型 MQL 学習器と完全知覚学習器 (Completely Perceptual Learning module) の 2 個になっており，各学習器から行動パターンを Decision module に渡し，行動を決定している．

4.2 提案手法の特徴

従来の MQL に切り換え学習と中央集権型 MQL を用いて提案手法を構成した．そこで，それらの特徴を詳述する．

4.2.1 切り換え学習

切り換え学習は，知覚精度を下げて得られる知識を減らすことにより，学習の後期における行動選択の精度が低下してしまい，CPQL と比べて劣ってしまうことを解消するために伊藤らが提案した手法 [6] である．

学習器の切り換え

追跡問題での強化学習において，特に時間のかかる初期の学習を 2 体のハンタの部分状態で状態を定義した中央集権型 MQL により知覚精度を下げて高速化し，ある程度学習が進んだ段階ですべてのハンタを考慮して状態を定義する完全知覚に戻すことを考える．これは適切なタイミングで切り換えることで達成できる．しかし，初期の知覚精度を下げた影響を完全になくすることはできないため，行動判断の精度を完全に保つことはできない．

そこで，この問題を解決するために並行学習を，適切な切り換えのタイミングを得るために平均学習残エントロピーを定義した．この学習器の切り換えは Fig. 2 の Decision module に相当し，図中の I は平均学習残エントロピー， T は閾値である．

並行学習

並行学習は初期の短期的な学習に長けている中央集権型 MQL と長期的な学習に長けている CPQL を同時に学習させる手法である．初期において，行動選択の学習器を中央集権型 MQL にして「そこそこ良い」行動系列を見つけ，学習させる．そして，適切なタイミングで行動選択の学習器を CPQL に切り換えることによって更なる収束が期待できる．

平均学習残エントロピー

切り換え学習のタイミングを決定するための評価量として平均学習残エントロピーを利用する．以下に，内部状態 S のときに行動 a を選択する確率 $p(S, a)$ を示す．

$$p(S, a) = \frac{Q(S, a)}{\sum_{i \in A} Q(S, i)} \quad (3)$$

ただし $Q(S, a)$ は状態 S における行動 a の Q 値， A は選択可能な行動の集合である． Q 値が 0 以下になる可能性があるため，そのような場合は補正值 δ を以下のように定義して，状態 S におけるすべての行動の Q 値:

$Q(S, \forall a)$ に δ を加算して確率を求めた．

$$\delta = |\min_a Q(S, a)| + 0.01 \quad (4)$$

以下に $p(S, a)$ から定義される学習残エントロピー $I(S)$, およびエピソード E に含まれる状態とエージェントすべてについて $I(S)$ を平均した平均学習残エントロピー I を定義する．

$$I(S) = -\frac{1}{\log n_a} \sum_a p(S, a) \log p(S, a) \quad (5)$$

$$I = \frac{1}{n_p |E|} \sum_{S \in E} I(S) \quad (6)$$

ただし $n_a = |A|$, n_p はハンタ数 , $|E|$ はエピソード E に含まれる状態数である．この平均学習残エントロピーが未学習状態のときは 1 , 学習が進むにつれて 0 に近づいていく．

4.2.2 中央集権型 MQL

切り換え学習をするとき , 学習の初期での行動選択に MQL を用いる．MQL ではハンタが n 体存在するとき , ハンタ 1 体につき部分状態は $n-1$ 個となり , 全部で $n(n-1)$ 個の学習器が必要である．さらに , 並行学習をするので CPQL の学習器も必要である．これにより学習器の数が非常に多くなってしまい , 消費メモリ量が多くなってしまふことが考えられる．

これを解決するために , MQL を中央集権型に改良した．これは 1 体のハンタにつき学習器の数を 1 個とし , ハンタが n 体存在するとき全部で n 個の学習器だけで済むようにしている．3.3 の例題と同様に , 4 体のハンタが獲物からの相対座標でそれぞれ $s_1 : (x_1, y_1)$, $s_2 : (x_2, y_2)$, $s_3 : (x_3, y_3)$, $s_4 : (x_4, y_4)$ という位置にすることが知覚できるとする．1 体のハンタの状態を考えるとき , MQL と同様に部分状態 $\langle s_1, s_2 \rangle$, $\langle s_1, s_3 \rangle$, $\langle s_1, s_4 \rangle$ を考える．これらを MQL のように別々の学習器に渡すのではなく , 同じ一つの学習器に渡す．そして次式に示すように , 行動選択器で受け取った Q 値がより大きな値となる行動 a_s を優先して行動する．

$$a_s = \arg \max_{a \in A} \sum_{i=1}^{n-1} Q(S_{k,i}, a) \quad (7)$$

このようにして切り換え学習で必要になる消費メモリ量を削減した．

4.3 切り換え手法

中央集権型 MQL と CPQL の Q 値を同時に更新する並行学習を利用すると , 二つの学習方法の学習器をメモリ上に存在させなければならないので , 消費メモリ量が多くなってしまふ．そこで , 学習を切り換える前では中央集権型 MQL の学習器のみ , 切り換えた後では CPQL の学習器のみをメモリ上に存在させる．そうすることにより , 切り換え前と切り換え後を通して常に CPQL 以

下の消費メモリ量で学習器が用意できる (切り換え器などのメモリ量を考慮しない場合) . この手法を実現するための切り換え時における Q 値の受け渡し手法を示す．

3 体のハンタが存在し , ハンタ 1 , ハンタ 2 , ハンタ 3 の状態をそれぞれ $s_1(x_1, y_1)$, $s_2(x_2, y_2)$, $s_3(x_3, y_3)$ とする．ハンタ 1 の中央集権型 MQL の Q 値表現 Q_m は $Q_m(s_1, S, a)$ であるのに対し , CPQL の Q 値表現 Q_c は $Q_c(s_1, s_2, s_3, a)$ である．ただし , S は任意の他の 1 体のハンタの状態 , a は取った行動の種類である．このように表現形式が違ふため容易に Q 値がコピーできない．そこで , 以下の式のように Q 値を配列形式で受け渡すようにした．

$$Q_c[x_1][y_1][X][Y][All][All][a] = Q_m[x_1][y_1][X][Y][a] \quad (8)$$

$$Q_c[x_1][y_1][All][All][X][Y][a] = Q_m[x_1][y_1][X][Y][a] \quad (9)$$

ただし , X, Y は任意の他の 1 体のハンタの状態の x 座標と y 座標 , All は考えられるすべての座標の値である．これにより , CPQL でペアとなるすべての組合せに 2 体に注目する中央集権型 MQL の同じ Q 値を受け渡し , 両者の表現形式の差異を吸収する．

5. 計算機実験

提案手法の有効性を検討するために , 追跡問題に HSCMQL を適用した．提案手法の適用実験内容とその結果について示す．

5.1 実験概要

実験対象の問題は 3. で述べた追跡問題である．この問題においてハンタは個別に強化学習をし , その実装方法として提案手法である HSCMQL の他に CPQL , MQL を適用した．また , 以下に本実験における収束の定義式を示す．

$$\frac{1}{n} \left| \sum_{k=t-n+1}^t S(k) - \sum_{k=t}^{t+n-1} S(k) \right| < \mu \quad (10)$$

なお , t はあるエピソードのタイミング , $S(t)$ は t に対応するステップ数 , n は観察するエピソード数 , μ を閾値とした．

これにより , 提案手法の有効性や切り換え学習のタイミングを見いだすことが目的である．

5.2 実験 1: パラメータによる影響と提案手法の有効性

5.2.1 実験方法

基本的な問題設定を以下に示す．

- 問題: 追跡問題
- ハンタ数: 3 体
- フィールドの大きさ: 7×7
- 1 ステップのコスト: 0.05

つぎに , Table 1 に本実験の各学習方法とパラメータを示す．なお , 行動選択法には (3) 式で示した Q 値に

Fig. 3 Learning graph with various parameters

Fig. 4 Learning graph with various thresholds

Table 1 Learning methods and parameters

Learning method	reward	α	γ
CPQL	5.0	0.8	0.5
MQL	1.0	0.8	0.4
HSCMQL	1.0	0.8	0.4
MQL	5.0	0.5	0.5
HSCMQL	5.0	0.5	0.5

よるルーレット選択法を適用し、HSCMQL では切り換え学習をしておらず、中央集権型 MQL のみを利用して

5.2.2 結果と考察

Fig. 3 に各学習方法とパラメータで試行した結果のグラフを示す。横軸に 300 エピソードを 1 目盛、縦軸に 300 エピソードごとの平均ステップ数をグラフにとって表示した。

中央集権型 MQL の有効性

2 組の同じパラメータを用いた MQL と HSCMQL の学習の様子の違いを考察する。この 2 組の MQL と HSCMQL にほとんど差異が見られないので、同等の学習性能であることがわかる。また、HSCMQL に関しては切り換え学習をしていないので、中央集権型 MQL と MQL が同等の学習性能であることがわかる。これより、MQL よりも消費メモリ量が少ない中央集権型 MQL が有効であることがわかる。

パラメータによる HSCMQL への影響

つぎに、異なるパラメータでの HSCMQL の学習の様子の違いを考察する。パラメータの設定がうまくいくと、CPQL とほぼ同じステップ数に収束することがわかる。しかし、パラメータの設定がうまくいかないとステップ数が跳ね上がってしまい、学習精度の劣化が見受けられる。そこで、3.4 でも述べたように、この学習精度が劣化してしまう問題は部分状態を観測することによる不完全知覚が原因であるので、完全知覚である CPQL へ切り換える学習によって解消されることが考えられる。

5.3 実験 2: 切り換え学習のタイミング

5.3.1 実験方法

問題設定は実験 1 と同様である。パラメータは HSCMQL (最終報酬: 5.0, 学習率: 0.5, 割引率: 0.5) で統一、CPQL は実験 1 と同様にした。

4.2 で述べたように、学習が進むにつれて平均学習残エントロピーは 0 に近づく。よって、平均学習残エントロピーが大きければ大きいほど学習の初期段階であることがわかる。そこで平均学習残エントロピーの閾値を設けて、その閾値を下回ったときに HSCMQL の行動選択の学習器を中央集権型 MQL から CPQL に切り換えるようにした。本実験では閾値として 0.95, 0.80, 0.50 を設定した。

5.3.2 結果と考察

Fig. 4 に各閾値で学習器の切り換えを試行した結果のグラフを示す。実験 1 と同様に、横軸に 300 エピソードを 1 目盛、縦軸に 300 エピソードごとの平均ステップ数をグラフにとって表示した。

閾値 $T = 0.50$ で切り換えた場合

CPQL と一番遅く学習器が切り換わる閾値 0.50 のグラフについて着目し、切り換えるタイミングが遅いときの考察をする。学習器が切り換わる前では精度が劣化してしまい、長期間その行動パターンを学習するため、その行動パターンを良策であるという考えが記憶されてしまう。そして、切り換え時に急激にステップ数が跳ね上がってしまっていることがわかる。さらに、若干ではあるが CPQL よりも悪いステップ数で収束してしまっていることもわかる。これは中央集権型 MQL によって学習して記憶してしまった行動パターンを CPQL に受け渡してしまったため、劣化した行動パターンを修正できていないことが原因として挙げられる。これより、さらに早いエピソード数での切り換えが必要であると考えられる。

閾値 $T=0.80$ で切り換えた場合

CPQL と閾値 0.80 で切り換えたグラフについて着目する．閾値 0.50 より早いタイミングで切り換えをしているが，中央集権型 MQL で学習精度の劣化が起きているので，その学習結果を引き継いでしまい，閾値 0.50 で切り換えた場合と同様に切り換え時に急激にステップ数が跳ね上がっている．しかし，あまり学習精度が劣化しているときの学習が記憶されていないため，収束するステップ数は CPQL とほぼ同じである．これより，学習精度が劣化する前に切り換える必要があると考えられる．

閾値 $T=0.95$ で切り換えた場合

CPQL と中央集権型 MQL の学習精度が劣化する直前あたりで切り換わっている閾値 0.95 のグラフについて着目する．切り換え時には，急激にステップ数が跳ね上がることもなく学習器の切り換えができており，切り換え後は CPQL と同等の収束ステップ数になっている．これらより，閾値 0.95 で切り換えた場合，常に CPQL よりも少ないステップ数，かつ CPQL 以下の消費メモリ量で学習器が用意でき，それを利用して効率的に学習できていることがわかった．よって，HSCMQL によって追跡問題における強化学習の効率化ができるといえる．

6. おわりに

本論文では，追跡問題における強化学習で，問題が複雑になるにつれて学習の速度が遅くなってしまうといった問題について，初期の短期的な学習に長けている MQL と長期的な学習に長けている CPQL を並行学習させて，途中で行動選択の学習器を MQL から CPQL に切り換えることを考え，その際に消費メモリ量を少しでも削減するために MQL を改良した中央集権型 MQL を用いて，切り換えの前後において，メモリ上に存在する学習器がどちらか一方のみにした HSCMQL を提案した．

そして適用実験により，中央集権型 MQL の有効性と HSCMQL の効率性を確認できた．また，学習残エントロピーによる影響や学習器の切り換えのタイミングによる影響を示した．

今後の課題として，切り換えのタイミングは中央集権型 MQL の学習精度が劣化する手前であるとわかったので，そのタイミングを自動的に見つける新たな切り換え指標を提案していきたい．また，他の追跡問題の条件に対する提案手法の有効性，問題拡張に対する柔軟性の調査と行動選択方法にボルツマン選択法 [7] を採用することなどが挙げられる．

謝 辞

本研究の一部は JSPS 科研費 24560501 の助成を受けたものであり，記して謝意を表す．

参 考 文 献

- [1] S. Russell and P. Norvig: *Artificial Intelligence: A Modern Approach*, Prentice-Hall International (1995)
- [2] 高玉：マルチエージェント学習 相互作用の謎に迫る，コロナ社 (2003)
- [3] R. S. Sutton and A. G. Barto: *Reinforcement Learning: An Introduction*, MIT Press (1998)，(邦訳) 三上，皆川 共訳：強化学習，森北出版 (2000)
- [4] N. Ono and K. Fukumoto: Multi-agent reinforcement learning: A modular approach; *AAAI, ICMAS-96*, pp. 252–258 (1996)
- [5] C. J. C. H. Watkins and P. Dayan: Technical note Q-learning; *Machine Learning*, Vol. 8, No. 3-2, pp. 279–292 (1992)
- [6] 伊藤，金淵：知覚情報の粗視化によるマルチエージェント強化学習の高速化 ハンターゲームを例に；電子情報通信学会論文誌 D-I, Vol. J84-D-I, No. 3, pp. 285–293 (2001)
- [7] 永吉，村尾，玉置：連続行動空間への適用を考慮した Switching 強化学習；電気学会論文誌 C, Vol. 131, No. 5, pp. 976–982 (2011)

著 者 略 歴

い き み たい すけ
伊 木 美 太 輔

1988 年生．2011 年 3 月 大阪府立大学工学部知能情報工学科卒業，同年 4 月大阪府立大学大学院工学研究科電気・情報系専攻博士前期課程入学，現在に至る．強化学習に関する研究に従事．

まつ ちと けい の すけ
松 本 啓 之 亮 (正会員)

1978 年 3 月 京都大学大学院工学研究科精密工学専攻修士課程修了．同年 4 月三菱電機株式会社入社．1996 年大阪府立大学工学部情報工学科教授．現在，大阪府立大学大学院工学研究科教授．おもに，知能情報処理やソフトウェアに関する研究に従事．工学博士．1984 年電気学会論文賞，2005 年電気学会進歩賞受賞．電気学会，情報処理学会，IEEE などの会員．

もり き なお 直 樹 (正会員)

1994 年 3 月 京都大学大学院工学研究科原子核工学専攻修士課程修了，1997 年 3 月同大学院工学研究科電気工学専攻博士後期課程指導認定退学．同年 4 月大阪府立大学工学部情報工学科助手，現在，同大学院工学研究科准教授．進化的計算，マルチエージェントシステム，人工市場などに興味を持つ．博士(工学)．計測自動制御学会，IEEE などの会員．