効率のよい2つの図形の誤差計算アルゴリズム

An Efficient Algorithm for Calculating the Error of Two Figures

大槻 正伸・小泉 康一・大塩 智規*
福島工業高等専門学校電気電子システム工学科
*福島工業高等専門学校専攻科 産業技術システム工学専攻
Masanobu Ohtsuki, Koichi Koizumi, Tomonori Ohshio
National Institute of Technology, Fukushima College, Department of Electrical and Electronic System Engineering

（2018年8月25日受理）

An efficient algorithm for calculating the error of two closed figures is proposed. It is very useful for constructing efficient algorithms that solve figure puzzles such as “Tangram”, with some approximation means like SA(Simulated Annealing), GA(Genetic Algorithm).

In this paper we define the error of two figures mathematically, as the area of the symmetric difference of the two figures, and construct an efficient algorithm with Green’s Formula for calculating the error of simple two figures, and analyze the complexity of it.

Key words: Computational Geometry, Symmetric Difference, Error

1. はじめに

計算幾何学 [3)4)]等で図形を扱う様々な問題において「平面上の 2 つの図形の誤差（あるいは一致度）」を求める必要がよく出てくる。特にあるデータ構造をもって 2 つの図形が表現されているとき、そのデータを入力し、2 つの図形の誤差を効率よく計算するアルゴリズムが必要となる [10)]。

本論文では、「平面上の、有限個の線分で囲まれた単純な（i. e. 有限図形を囲む線分同士が交わらない）閉図形（線分および内部の点集合）F_1,F_2についての「F_1 と F_2の誤差を計算する」という計算問題に対して、効率のよいアルゴリズムを構成し、最後に構成したアルゴリズムの応用について述べる。図形 F_1 と F_2の誤差は $S(F_1 \triangle F_2)$で定義される（後述）。ここで△は対象差、S( )は面積を意味する。また F_1,F_2の周囲の線分の集合を∂F_1,∂F_2で表すこととする。

この計算問題が出てきた背景として、文献 7)の「2 つの閉軌道の誤差を計算する問題」や、文献 10)の「タングラム [6) 8) 9) 10) 13)]等の図形構成問題において、真の解である図形と SA（焼きなまし法）[2) 12)]や GA（遺伝的アルゴリズム）[5)]で求めた近似解の図形との誤差を計算する問題」等があげられる。これらの文献では、部分的には工夫があるものの、基本的には「2 つの図形を囲む範囲を、解像度 h で格子点 $\{(a+i*h,b+j*h)$; i=0,n-1, j=0,m-1$\}$ を探索し、図形内部に入った点を数え、領域の面積≒領域内部の点の個数×h^2 とする」等の操作を含み、高い精度を求めれば解像度 h を小さくしなくてはならず、そうすると計算量 [1)]が多くなり効率的とはいえないものであった（Fig.1）。この問題に対して効率的なアルゴリズムを構成するのが本論文の目的である。

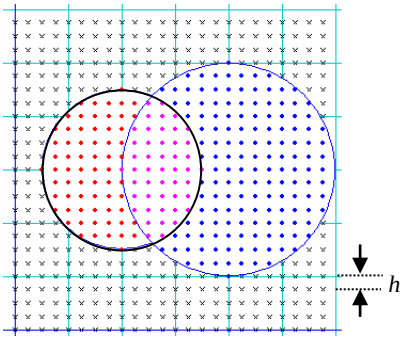

Fig.1 Calculation of Area of a Figure by Search with Resolution h

2. 図形の表現と計算問題の定式化

今回扱う図形は「有限個の線分で囲まれた単純な閉図形（周囲の線分とその内部の点集合）」とする。

そして、図形は、自然に「図形内部を常に左に見るように頂点を順番に訪れるものとし、その順番での頂点座標の列」というデータで表現されるものとする。

例えば、Fig.2 の 2 つの図形 F_1,F_2 は次のように表現される。

F_1＝{$P_{11}(0, 0)$, $P_{12}(0.4, 0)$, $P_{13}(1, 0.6)$, $P_{14}(1, 1)$, $P_{15}(0, 1)$}

F_2＝{ $P_{21}(0.1,-0.1)$, $P_{22}(0.7,- 0.1)$, $P_{23}(1.3, 0.7)$, $P_{24}(0.2,0.7)$}

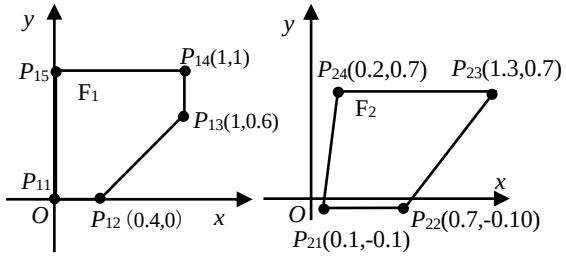

Fig.2　Examples of figures treated in this paper

また、F_1は{ P_{12}, P_{13}, P_{14}, P_{15}, P_{11}}とも{ P_{13}, P_{14}, P_{15} , P_{11}, P_{12}}と表現してもよい。

そして、最後の頂点と最初の頂点は結ばれているものとする。すなわち「最後の点の『次の点』は最初に戻って初めの点」と解釈する。

なお、図形の表現はタングラム問題の場合などは「どの連続する3点をとっても一直線上にない」ように無駄を省いて表現することが多いが、今回は、２軌道の誤差計算[7]も考えるため、特に図形表現のこの無駄についてはどちらでもよいこととする。

このような単純な閉図形が２つあった場合、その誤差は次のように計量することとする[7) 9)]。

【定義１】（図形の誤差）

単純な閉図形 F_1,F_2 に対して、誤差 $d(F_1,F_2)$を次で定義する。

$d(F_1,F_2)=S(F_1 \triangle F_2)=S((F_1- F_2)\cup(F_2- F_1))$

ここで、S( )は面積、△は対象差である。

すなわち、２つの図形の誤差（あるいは一致度とも解釈できる）は、一致しない部分の面積で計量するものとする（Fig.3）。

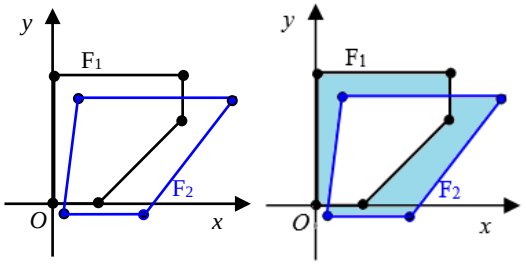

Fig.3　Error of 2 figures F_1,F_2 in Fig.2

【性質1】Uを、有限個の線分で囲まれた単純な閉図形の集合とすると、＜U, d＞は疑距離空間となる。

＜証明＞略　□（性質１）

これで、本論文で扱う図形の集合（空間）に疑距離が導入され、完全に２つの図形が合同であれば、疑距離＝0となり、図形の誤差（一致度）が計量できるようになる。

この誤差（疑距離）は、例えば、「Fig.2 の図形 F_1 を一回見て、その後 F_1 を見ず、思い出して F_1 を再現したつもりで F_2 を描いたとすると、どの程度正確に F_1 を描けたか」という視覚心理学的な心理物理実験の評価に必要になる[7)]（文献 7）では単純閉曲線の近似データとして頂点数は数百個となっている）。

また、例えば、タングラム等の図形復元問題（与えられたピース図形を組み合わせて、与えられた問題シルエットを構成するパズル）において、どの程度正確な近似解が得られたかの評価にも必要となる[10)]。

特にタングラム解法のアルゴリズム全体を考えると、この誤差計算アルゴリズムは、全体のアルゴリズムの速度に直結するため、効率のよい誤差計算アルゴリズムの開発が必要となるのである。

図形の誤差計算問題は次のように定式化される。

【2 つの図形の誤差計算問題】

$$F_1=\{P_{11}(x_{11},y_{11}),P_{12}(x_{12},y_{12}),\cdots,P_{1m_1}(x_{1m_1},y_{1m_1})\}$$

$$F_2=\{P_{21}(x_{21},y_{21}),P_{22}(x_{22},y_{22}),\cdots,P_{2m_2}(x_{2m_2},y_{2m_2})\}$$

とする。すなわち、F_1,F_2 はそれぞれ、m_1,m_2 個の頂点を持つ、頂点番号順に線分で結んでできる単純な閉図形とする。

F_1, F_2 の上記データを入力し、$d(F_1, F_2)$を計算することが本論文で扱う計算問題となる。ただし、今回はアルゴリズムの見通しを重視し、「F_1（F_2）の線分は高々1 つの F_2（F_1）の線分としか交わらない」ことを仮定し簡単化して論を進める。これは、Fig.4 のような状況はないものと仮定することである。

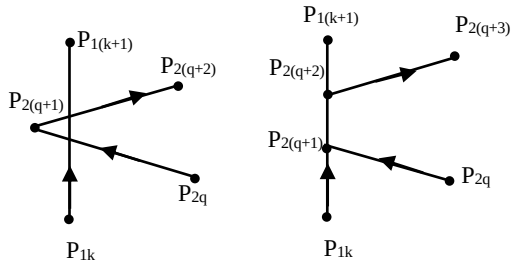

Fig.4 Examples of Inhibited Condition Supposed by the Algorithm Construction

この仮定は、文献 7)の軌道推定等において、図形 F_1,F_2 の輪郭の曲線が十分細かい直線で近似されている場合自然な仮定となる。しかし、タングラム問題における、真の解と解候補の誤差計算においては、Fig.4 の状況はよく起こりうる。したがって、今回構成したアルゴリズムは、Fig.4 の状況に対応した拡張が必要であるが、そのためには、やや複雑なデータ構造や処理が必要となる。この拡張アルゴリズムの構成、実際のパソコンへの実装等は今後の課題として残されている。

以下では、上記仮定のもとで正しく動作する、図形の誤差の計算問題に対する効率のよいアルゴリズムを構成する。

3. 図形の誤差計算の効率のよいアルゴリズム

3.1 基本的事項

まず１つの図形 F_1 の面積を計算するアルゴリズムを構成する。

次の２つの事実を確認しておく。

（１）Green の公式 [11] $\iint_D \frac{\partial f(x,y)}{\partial x} dxdy = \int_{\partial D} f(x,y)dy$

より $S(F_1) = \iint_{F_1} 1 dxdy = \int_{\partial F_1} xdy$

（２）線分 L を(x_0,y_0)－(x_1,y_1)　とし、L に対する線積分 $\int_L xdy$ を考える（∂F_1 全体の線積分の一部）。 パラメータ　$t \in [0,1]$を用いて、

$x(t) = x_0 + (x_1 - x_0)t,\ y(t) = y_0 + (y_1 - y_0)t$ として、

$$\int_L xdy = \int_0^1 x(t)\frac{dy}{dt}dt = \int_0^1 \{x_0 + (x_1 - x_0)t\}\cdot(y_1 - y_0)dt$$

$$= \frac{(y_1 - y_0)(x_0 + x_1)}{2}$$　となる。

したがって、F_1 の面積 $S(F_1)$を計算するアルゴリズムは次のように構成できる。

```
function calculateS(F1) ;
begin
  s:=0;
  for i:=1 to m1 do
    begin
      s:=s+(y1(i+1)−y1i)＊(x1i+x1(i+1))/2
    {＊ x1(m1+1) は x11 と解釈する y1(m1+1) も同様＊}
    end ;
  calculateS:= s
end.
```

この面積計算アルゴリズムの時間計算量は $O(m_1)$である。

3.2 ２つの図形の誤差計算の基本的な考え方

アルゴリズムの正確な記述に入る前に、見通しをよくするためにおおよその考え方を説明する。$S(F_1 \triangle F_2)$ の計算手順は以下のとおりである。

（１）$S(F_1)$を前節3.1のアルゴリズムで計算する（時間計算量$O(m_1)$）。

（２）$S(F_2)$を前記3.1のアルゴリズムで計算する（時間計算量$O(m_2)$）。

（３）$S(F_1 \cap F_2)$を次節で述べるアルゴリズムで計算する。（時間計算量$O(m_1 m_2)$）

（４）$S(F_1 \triangle F_2) = S(F_1)+S(F_2)-2S(F_1 \cap F_2)$より、誤差を求める。（時間計算量$O(1)$）

全体としてこの計算アルゴリズムの時間計算量は $O(m_1 m_2)$となり、計算量は文献7)のような解像度hに依存するアルゴリズムではないものが構成できたことになる。

また、計算の誤差も文献7)のものとは異なり解像度hに依存せず正確な値が求まる（より正確にいうと、実行するコンピュータの実数変数の精度まで正確な値が出ることになる）。

問題は上記（３）の$S(F_1 \cap F_2)$を計算する部分である。

3.3 ２つの図形の共通部分の面積計算の考え方

２つの図形F_1,F_2に対して$S(F_1 \cap F_2)$の計算アルゴリズムを記述する前に、見通しを明らかにするために、まずFig.5をもとに、その考え方を説明する。Fig.5では、典型的な２つの場合が示されている。

Fig.5で、各図形F_1,F_2は曲線で描かれているが、実際は、多数の点集合（を連結した線分）でデータ表現されている（Fig.5左 P_{11},P_{12},P_{13}…等）。そして、各点には番号がついており、番号順に点をたどると図形内部を常に左に見るように移動することに注意する。

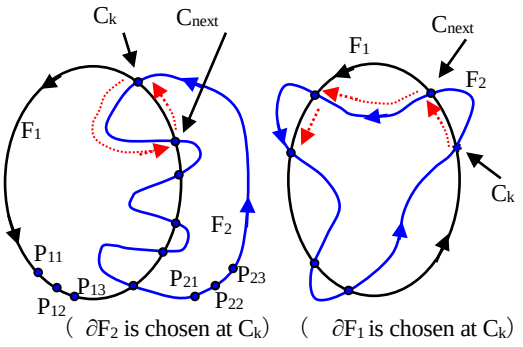

Fig.5 The Curve of Line Integral for Calculation of $S(F_1 \cap F_2)$ （red dotted lines）

まずは、∂F_1と∂F_2交点を任意に選びC_kとする。交点C_kからC_kを通る線分のうちでF_1,F_2の頂点（頂点番号の大きいもの）に移動したとき、相手の図形の内部に入っていく線分を持つ図形（F_1かF_2）を選びそれをFとする。交点C_kから、選んだFの∂Fに沿って、次の交点C_{next}まで線積分を計算し変数に足し込んでいく。そうしてC_{next}でも同じこと（相手の図形内部に入る線分をもつ図形を選び、次の交点まで線積分）をしていき、もとのC_kに戻るまでこれを繰り返す。

これで、$S(F_1 \cap F_2)$の一部（Fig.5左）または全部（Fig.5右）が計算されている。$S(F_1 \cap F_2)$が全部計算されつくされていない場合は、同じことを繰り返していく。

3.4 ２つの図形の共通部分の面積計算アルゴリズム

計算アルゴリズムを構成する前に、「線分の交差パターン」について考察する。

線分が交差する場合、交差パターンは Fig.6（上）にしたがい分類できる。すなわち、Fig.6 下（左）のような場合、F_1 の線分 P_{1k}→$P_{1(k+1)}$は、P_{1k} は図形 F_2 の内部（交差する F_2 の線分 P_{2q}→$P_{2(q+1)}$の左側）にあり、$P_{1(k+1)}$は F_2 の外部（線分 P_{2q}→$P_{2(q+1)}$の右側）にある。したがって線分 P_{1k}→$P_{1(k+1)}$は「in→out」のパターン「1」に分類される。

図形 F_2 からすると、P_{2q}→$P_{2(q+1)}$は、相手の図形 F_1 の外部から内部へ移動するから「out→in」のパターン「2」に分類される。

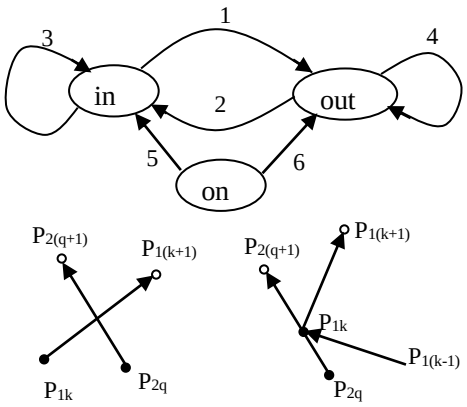

Fig.6 Cross Pattern and 2Examples

このように、各線分の交差パターン分類をするが、例えば Fig.6 下（右）のような状況の場合、P_{1k}→$P_{1(k+1)}$は「on（相手の図形の線分上）→out」のため、パターン「6」に、P_{2q}→$P_{2(q+1)}$の線分は。「in→in」のパターン「3」に分類される。

なお、今回は「F_1（F_2）の線分は高々1 つの F_2（F_1）の線分としか交わらない」という仮定をおいているから、Fig.6 の下（右）の交点は、線分 P_{1k}→$P_{1(k+1)}$（ベクトルの出発点）に属することとし、線分 $P_{1(k-1)}$→P_{1k} には属さないこととする。

すなわち、線分 P_{1k}→$P_{1(k+1)}$という場合、点 P_{1k} と線分上の点集合を意味し、点 $P_{1(k+1)}$は含まないものとして考える（そのため Fig.6 で線分の矢印の先の点は白丸で表現してある）。また、この仮定から「on→on」のパターンは自動的に排除される。

以上のように、交点パターンの扱いには細心の注意が必要となる。

さて、$S(F_1 \cap F_2)$の計算は次のアルゴリズムによる

begin

Read(F_1,F_2の頂点)；

∂F_1と∂F_2の交点をすべて求め

｛（交点C_i,pat_{i1},pat_{i2}）；i=1,2,…,交点数｝

という3次元ベクトルの集合 SET_{Cross}を求める。

ここで、C_iはi番目の交点、pat_{i1}(pat_{i2})は交点C_iのF_1（F_2）の線分に対する交差パターンである。

V：＝SET_{Cross}；

S:=0；

while (V≠ φ（空集合）) **do**

begin

①Vから交点を一つ選び、それをC_kとする；

②V：＝V－｛C_k｝
③C_0：＝C_k；
④n：＝C_0における交差パターンが2or3or5の値を持つ図形番号；
//*「C_0から相手の図形の内部に入っていく軌道」を輪郭線としてもつ図形F（それはF_1またはF_2）を選ぶ。
⑤C_{next}：＝C_kから∂Fの線分に沿って移動する場合の次の交点；
⑥ S:=S+ (C_0から∂Fの線分に沿ってのC$_{next}$までの線積分値)；
⑦C_0：＝C_{next}；
while($C_0 \neq C_k$) **do**
begin
V：＝V－｛C_0｝；
④；⑤；⑥；⑦｛＊上記④⑤⑥⑦を行う＊｝
end;
end；
Write(s)
end.

なお、このアルゴリズムの時間計算量は、F_1 の線分とF_2の線分の交点すべてを求める部分が主となり、アルゴリズム全体では O(m_1m_2)である。

3.4　２つの図形の誤差計算アルゴリズムの実装

実際に上記S ($F_1 \cap F_2$) ,S($F_1 \triangle F_2$) 計算アルゴリズムをDeiphi2010でプログラム化し、いくつかのデータで実行してみた（Fig.7）。
実際に正しく計算されているのが確認された。例えばFig.7中段の例でいうと、図形F_1（F_2）は黒色（青色）の線で囲まれた図形であり、それぞれの面積は方眼を数えると、24,26であることがわかるが、それが画面のSf1,Sf2のところに正しく計算され出力されている。$F_1 \cap F_2$の面積は方眼を数え2.75とわかるが、それが画面上Sf12のところに正しく計算され出力されている。
最終的に2つの図形の誤差は24+26-2×2.75=44.5であり、それが画面のSSdfのところに出力されている。

今回は述べなかったが、$F_1 \cap F_2 = \phi$（空集合）の場合、$|F_1 \cap F_2| < \infty$（２つの図形が有限個の点で接している場合）、$F_1 \subset F_2$（あるいは$F_2 \subset F_1$）の場合等にも対応できるように、ここで構成したアルゴリズムを拡張することは容易である。

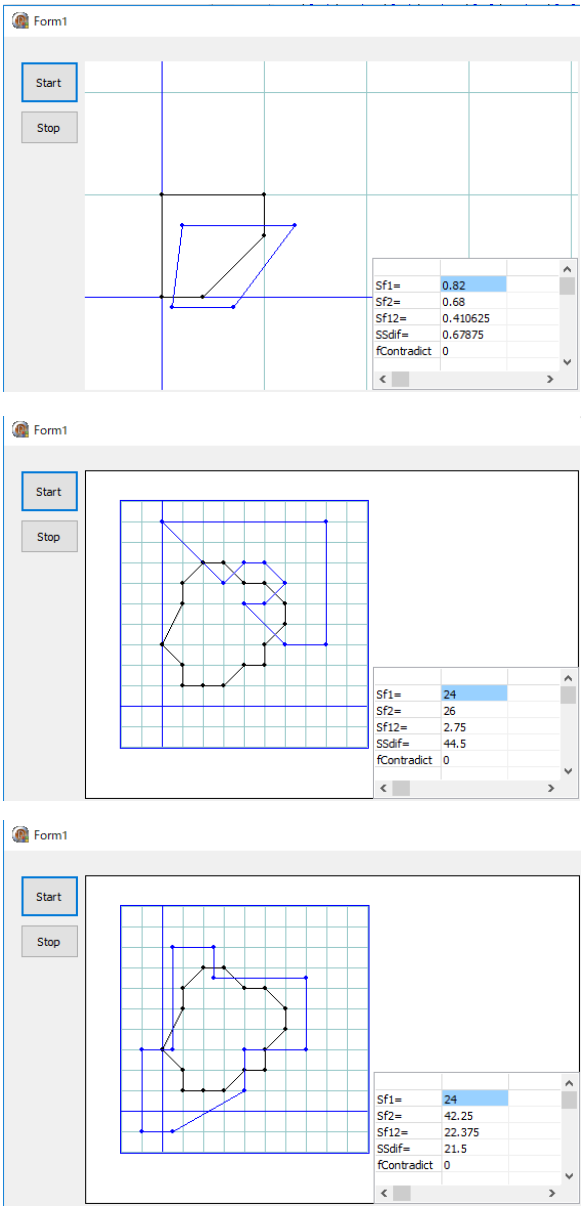

Fig.7 Results of the Algorithm Execution

4. 今後の課題

「$F_1(F_2)$の各線分は、$F_2(F_1)$の線分と高々1か所でしか交点をもたない」という条件のもとで、S($F_1 \triangle F_2$)を計算する効率の良いアルゴリズムを構成した。
今後の課題として、次のことがあげられる。

（１）上記の「高々１か所でしか交点を持たない」という条件をはずした、より一般的な場合にも対応できるアルゴリズムへ拡張すること。
その場合、交差パターン分類が非常に複雑になるため、精密な考察が必要となる。
（２）実際に、軌道推定問題の誤差計算、タングラム問

題の近似解法のプログラムに取り入れて、これらのアルゴリズムの高速化を実現すること

などがあげられる。

参 考 文 献

1) A.V. Aho，J.E.Hopcroft，J.D.Ullman，The Design and analysis of Computer Algorithms，pp364-404，Addison－Wesley Publishing Company,1974

2)浅居　喜代治編著，基礎　システム工学　p143、オーム社，2001

3) 浅野 哲夫，計算幾何学，朝倉書店，1990

4) M.ドバーグ，O.チョン, M.ファンクリベルド, M.オーバマーズ著（浅野 哲夫訳）, コンピュータ・ジオメトリ, 近代科学社，2010

5)北野 宏明，遺伝的アルゴリズム，産業図書，1993

6)中野良樹、大槻 正伸、数理パズル“タングラム”の洞察的問題解決における視線移動の分析、日本認知科学会第 34 回大会講演論文集、pp.1004-1010、2017

7)大槻 正伸、中野 良樹、R^2 内の 2 閉軌道の誤差計算アルゴリズム、福島高専研究紀要 No.42、pp35-42、2002

8)大槻 正伸，中野 良樹，新井 広，パズルゲーム「タングラム」解法の基本アルゴリズム，福島高専研究紀要 第 56 号　pp 19-24,2016

9) 大槻 正伸，小泉 康一，中野 良樹、新井 広，パズルゲーム「タングラム」の解法アルゴリズム，福島高専研究紀要 第 57 号　pp 15-20,2017

10) 大槻 正伸, 小泉 康一, 中野 良樹、新井 広, シミュレーテッドアニーリングを用いたパズルゲーム「タングラム」の近似解法アルゴリズム，福島高専研究紀要 第 58 号　pp 15-22 ,2018

11) 岩波数学入門辞典、pp174-175、岩波書店 2005 年

12) 焼きなまし法に関するホームページ，https://ja.wikipedia.org/wiki/%E7%84%BC%E3%81%8D%E3%81%AA%E3%81%BE%E3%81%97%E6%B3%95

13)タングラムに関するホームページ URL http://ja.wikipedia.org/wiki/%E3%82%BF%E3%83%B%E3%82%B0%E3%83%A9%E3%83%A0