＜報文＞

大規模迷路探索シミュレーション

― その２　立体の場合 ―

関東職業能力開発大学校　　加　部　通　明

The search simulation of a large scale Maze　　-No2 In the case of a solid-

Michiaki KABE

要約　以前、平面上の大規模な迷路探索シミュレーションについて報告した(1)。今回は、それを基にして大規模な立体迷路について、平面上で行ったのと同様な探索方法とそれ以外の探索方法とにより迷路探索シミュレーションを行った。具体的には、出口が異なる3種類の迷路に対して、準最短距離、初回距離、総探索距離、試行回数を調査した。その結果、総探索距離が長いほど準最短距離が短いことや、出口方向に進む拡張左手法に基づいた方法が初回探索距離が短い等が分かったので、それらについて報告する。

Ⅰ　はじめに

今回、考察する立体迷路とは、平面の場合と同様に以下の条件を満たす立体形状(注1)の迷路である。

［立体迷路の条件］

①入口と出口がそれぞれ1箇所ずつある。

②１つの区画にきてはじめてその区画の情報（進路）がわかる。

③区画は外見上同一であり区別できない。

④入口から全ての区画に到達可能である。

一般に、立体迷路を探索する場合、同一平面上の迷路が途中で途切れるので、それが上下とどのように繋がっているのかを見極め、目的地までどのような路を辿ればよいかを探し出すのが難しい。従って、たとえ立体迷路が小規模であっても、全体像を一目で把握するのが困難なので、その最短経路を目視で発見するのは苦労する。それが、例えば、縦×横×高さが40×64×40位の規模になると、その最短経路を目視による試行で見つけるのはかなり難しくなる。

今回取り上げた立体迷路の規模は、パソコンのメモリ容量の制約により、40×64×40という立体としては中規模ではあるが、全体では102,400区画あり、大規模な迷路と見なすことができる(注2)。また、限られた情報と少ない試行回数で最短経路を発見するのは困難であるから、平面の場合と同様に最短経路に近い準最短経路の発見を取り上げることにする。

大規模な立体迷路探索の応用としては、例えば、客船の沈没などの海難災害や、今後予想される宇宙船内での不慮の事故による船内災害などで、船室に閉じこめられた人々の救助活動が挙げられる。それは、災害により船室内部が倒壊して、船内が立体迷路のようになり得る場合があるからである。

Ⅱ　立体迷路の表示方法と作成方法

立体迷路を扱う場合、迷路自体を作成するよりも迷路をパソコン画面上に表示する方に工夫が必要である。壁が視界を遮りる中、移動する物体を上手に見やすくするにはどうしたら良いかが問題なのである。

１　大規模迷路の表示法

限られた画面の大きさの中で、最大40×64×40なる規模の迷路を表示させる為に、階層的な表示法を用いる。それには、３次元表示と２次元表示の２種類を用いる。

(1)３次元表示（図１）

第１階層は、立体迷路全体を縦、横、高さそれぞれについて２等分した４個の立体から構成され、それを画面左下に３次元表示させる。そして、着目する区画がどこにあるのかを明示するために、その箇所の立体

論文受付日　H18.6.8

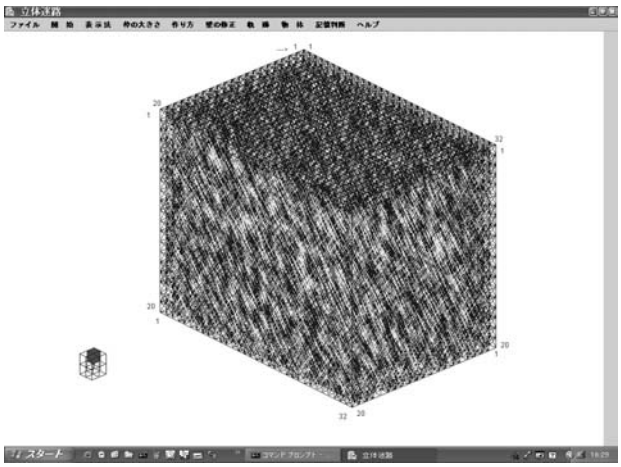


図１　立体迷路40×64×40の３次元表示

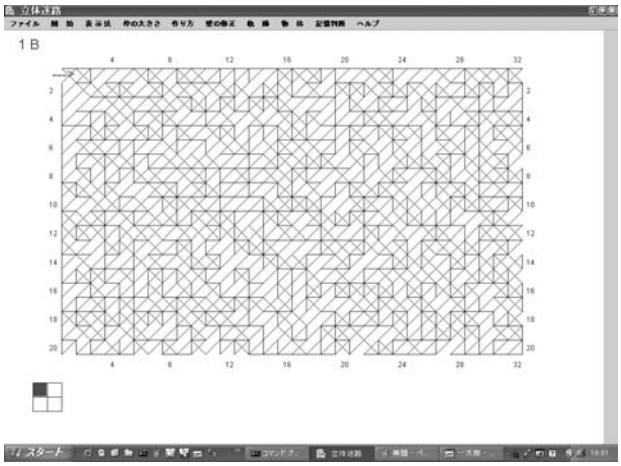
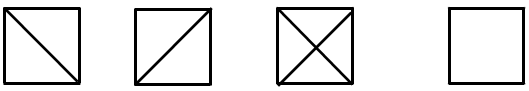


図２　立体迷路40×64×40の2次元表示



(a)床　(b)天井　(c)天井と床　(d)天井と床なし

図３　２次元表示における天井と床の表し方

格子に赤色を付けて示している。次の第２階層は、20×32×20の立体格子で構成され、第１階層の着目する立体格子の詳細な立体迷路表示となっている。

(2)２次元表示（図２）

立体迷路を上方から見た平面図で、各階層を平面迷路と同じ手法により表示し、壁、天井、床を直線で表す。特に、天井と床の有無の様子を図３に示した。

また、ある階層の迷路を２次元表示させる為には、先ず、目的の階に移り、次に、目的の箇所の迷路を表示させる操作を行う。具体的には、次のような操作手順に従って行う。

[2次元表示の操作手順]

①メニューで表示方法、２次元表示を選択する。

②表示したい階を含む画面左下にある第１階層の２×２格子の枠内をクリックする。

③第２階層の平面迷路で、縦の区画位置を表す左側に配置された番号の近辺をクリックする。即ち、この番号は階層の階数の意味をも兼ねている。

④上記③で目的とする階に辿り着いたので、また、第１階層の２×２格子に戻り、目的の第２階層を表示させたい箇所の格子をクリックする。

以上の操作手順を、具体的な例を挙げて図４で示した。その図では、階層23Bの(1～20)×(33～64)なる迷路を２次元表示させたものである。

2　作成方法

迷路自動生成方法にはいろいろとあるが[2]、以前用いた平面迷路と同じ手法を立体迷路に応用する[1]。即ち、次のような手順で自動的に迷路を作成し、その後、必要に応じて手動で修正する。

2.1　自動生成

入口を一番上（図１、２の矢印)、出口を一番下にして次の手順で自動的に迷路を作成する。

①出口に到るまでの道を乱数を使用し１つ確保する。規模が大きい場合には騙し道を１本確保する。

②各区画からこれら２本の道に通ずる道を作る[1]。

③周囲が全て進行可能な区画が集まっている所では、道を塞がないよう壁、天井、床を乱数を用いて作る。

2.2　手動による修正

立体迷路の一部の区画で、壁、天井、床を追加又は削除したりする場合、３次元表現でその修正作業を行うにはプログラム上かなり煩雑な処理がいる。それを回避するために、次のような手順で修正作業を行い、必要に応じて①から③の操作を繰り返す。

①修正区画を2次元表示させる。

②追加又は削除の指示を行う。

③修正する箇所の直線近辺をマウスでクリックする。

3　区画情報

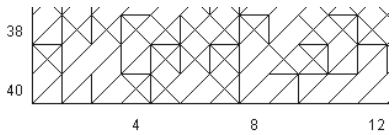
各区画には、前後左右上下それぞれについて、次のような情報を持たせ、その活用を図る。

①壁の有無：進行の可否

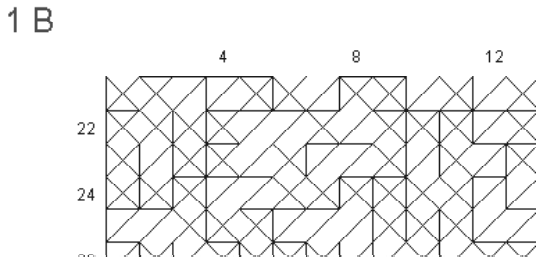
②通過回数：進路の選択基準

③距離：準最短経路の距離計算

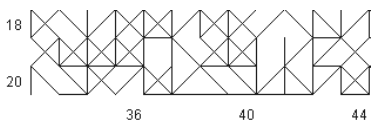
①表示法→2次元表示をクリック



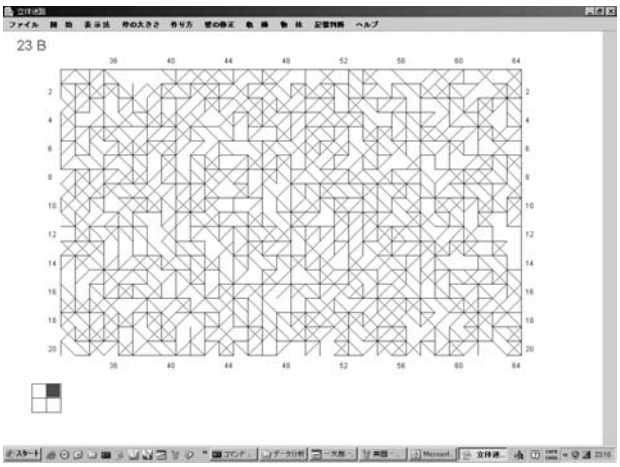
②第1階層2×2格子の23Bに該当する格子をクリック



③左側の縦に並んだ番号の23近辺をクリック



④2×2格子で(1～20)×(33～64)に当たる格子を
クリック



⑤目的の23B(1～20)×(33～64)迷路が表示された

図4　立体迷路の2次元表示操作手順

④行き止まり：無駄な探索の回避
⑤進行方向：ループの判断基準

Ⅲ　探索方法

1　探索方法

迷路探索法にはいろいろとあるが[3]、[4]、[5]、今回は次の２方法により、立体迷路探索を行う。

(1)ある判断基準に基づいたトレモー法[4]

平面迷路と同じように進路決定に際し、以下で述べる判断基準によりトレモー法で探索する。

(2)下(上)方向優先拡張左手法[注3]

出口が一番下(上)にあるという設定を利用して、平面迷路での左手法により通過回数を考慮した探索を行う。行き止まり以外、今来た路（直前の路）には戻らないと言う制約の下で、直前の区画を除いた周辺区画を見て次のような手順で進路を決定する。

[平面上での拡張左手法[4]]

最少通過回数の区画[注4]を左手方向から捜し、最初に見つけた最少通過回数の区画に進む。

[下(上)方向優先拡張左手法]

最少通過回数の区画を基にして、平面上での拡張左手法を利用し、次の手順により進路を決定する。

①最少通過回数の区画が複数ある場合、その中に下(上)方向があれば下(上)に進む。もしなければ、平面上のある方向の通過回数が上(下)方向の通過回数以下である場合には、平面上を拡張左手法で進み、それ以外の場合には、上(下)方向に進む。

②最少通過回数の区画が１つしかない場合には、その区画に進む。

2　知能の設定

迷路探索における岐路での判断基準とは何かを、以下の値を基にして調査検討した。この判断基準は人間の知能に相当し、知能の違いにより準最短経路に纏わる諸値に違いが生じることをシミュレーションにより確かめる。

[記号の意味]

Ｄ：準最短距離

D_n：試行nにおける出口までの距離

Ｎ：準最短経路が求まるまでの試行回数

ＴＤ：試行N回までの各試行の距離の和。即ち、

$TD=D_1+D_2+\cdots+D_N$

調査する値は､10回準最短経路を算出した場合のＤ、

N、ＴＤ及びD_1の平均値である。そして、これらの数値が小さいほど、知能レベルが高いと判断される。

平面迷路のときと同じトレモー法を用いる場合には、次の①～③のような知能レベルを設定した(1)。更に、それらの方法を別な方法と比較する為に④と⑤を追加した。

［知能レベルの設定と探索方法］

①進行方向の自由度を考慮し、通過回数を決定する。

②特別の機能を持たせない。

③通過回数を不確定にする。

④下方向優先拡張左手法による。

⑤上方向優先拡張左手法による。

以下、①～③について、やや詳細に説明する(注5)。

①では、進行方向が多い区画を優先して探索させるようにする。その為に、具体的な操作として、６方向に進行可能な場合１を、５方向の場合２を、以下順に２方向の場合５を通過回数に加え、１方向つまり行き止まりの場合には、通過回数をHighValueにする。

②では、純粋な通過回数だけを基にして探索する。

③では、乱数を用い確率0.2の割合で、通過回数を加えずに通過回数を乱れさせる。これは人間の忘却を擬似的に行っている。

Ⅳ　シミュレーションについて

１　シミュレーション方法とその内容

まず最初に、上で説明した①～③のトレモー法と④、⑤の方法をそれぞれ独立に行う。次の２回目以降では、それぞれの最初の結果を元に、それぞれ独立に②のトレモー法による準最短経路を求めるシミュレーションを行い、併せて試行回数Ｎと総探索距離ＴＤをも調査する。ここで、準最短距離Ｄが決定したとは、連続して5回同じ探索距離となった場合に、距離が収束したと見なして決定の判断を下した。その理由は、ある区画を通過する際、多くとも５箇所の未通過区画が隣接するから、それら全てを通過するためには5回試行を行えば十分であると考えたからである。

具体的には、40×64×40なる規模の出口だけが異なる３つの立体迷路Ａ、Ｂ、Ｃを調査対象にした(注6)。そして、シミュレーションを行うプログラムをJavaで作成し、主に次のような実験環境で実験を行った(注7)。そのシミュレーションプログラムの処理内容を表1に示す。

［主な実験環境］

ＯＳ：Windows XP

ＣＰＵ：Pentium(R)4 3.40GHz

メモリ：0.99GB

２　シミュレーション結果とその評価

上記Ⅳ-1で説明したシミュレーションを行い、その様子を図５に、又、その結果を表２、３に示す。

2.1　知能レベルの設定と探索方法について

3つの立体迷路をシミュレーションした結果を総合的に判断すると、準最短距離Ｄが短いものと総探索距

表１　シミュレーションプログラムの処理内容

メニュー	処 理 内 容
ファイル	開く、保存、終了
開　始	トレモー法→初回 トレモー法→経路決定 ダイクストラの方法(n≦2まで) 下(上)方向優先拡張左手法→初回 下(上)方向優先拡張左手法→2回以降
表示法	2次元表示、3次元表示
枠の大きさ	$(5\times2^{n-1})\times(8\times2^{n-1})\times(5\times2^{n-1})$, $1\leqq n\leqq 4$
作り方	自動生成、手作業
壁の修正	追加、削除
軌　跡	有り、無し
物　体	虫(平面表示のみ)、ボール
判断基準	やや高い、普通、やや低い、
ヘルプ	概要、　遊び方

表２　各迷路における判断基準等と各値との関係(注8)

判断基準等		①	②	③	④	⑤
迷路Ａ	Ｎ	6.4	5.3	8.5	5.5	2.0
	ＴＤ	20.3万	37.8万	43.3万	31.9万	62.2万
	Ｄ	219	204	188	193	159
迷路Ｂ	Ｎ	3.5	4.8	5.3	5.5	5.8
	ＴＤ	22.1万	36.6万	45.7万	32.6万	34.5万
	Ｄ	196	178	175	179	177
迷路Ｃ	Ｎ	3.0	3.4	5.4	3.8	6.3
	ＴＤ	29.3万	48.7万	44.9万	56.6万	38.4万
	Ｄ	170	166	168	171	163

表3　初回探索距離と各判断基準や探索方法との関係

迷路	①	②	③	④	⑤
A	126, 838	172, 358	196, 432	1, 331	621, 359
B	167, 134	238, 236	251, 402	67, 283	51, 267
C	237, 723	355, 114	307, 541	374, 320	1, 804

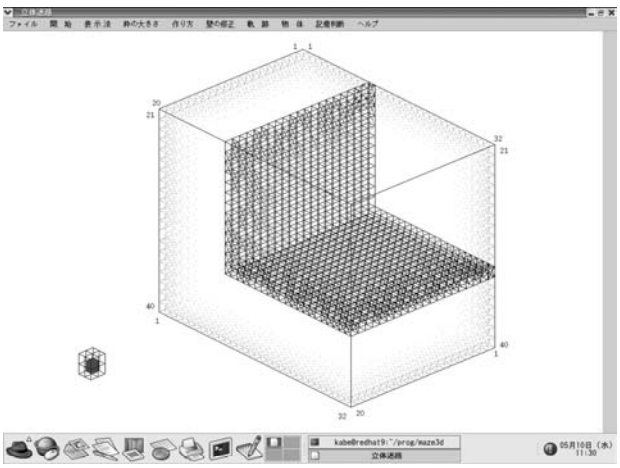


図５　迷路探索シミュレーション風景(注9)

離ＴＤが長いものが②、③、④、⑤の場合である。ＴＤが長いとは、それだけ数多くの区画情報を収集し、それによって最短距離に近い探索距離を求めることができると考えられる。即ち、ＴＤとＤとは互いにトレードオフの関係にあり、その最たるものが、ダイクストラの方法である。即ち、全ての区画情報を手に入れて、はじめて最短距離が決定できるのである。

また、ＴＤとＤのトレードオフに関する対偶的な関係として、ＴＤが短ければＤが長いとなる。この状況を表しているのが①と考えられる。いずれにせよ、今回調査した探索方法全てにＴＤとＤについては、トレードオフの関係にあるといえる。

更に、①については、周辺区画の通過回数とループを判断して同じ道を何回も繰り返して通らないようにしているので、②、③に比べるとＴＤやＤ1が短いと思われる。このことは、賢くて要領の良い人間が準最短経路を求める場合と似ている。つまり、そうした人間は多大な労力と時間を掛けずにそこそこの結果を出そうとするからである。良い結果を出すためには、それなりの労力と時間を掛け地道な努力をしなければならないと言うことを、この迷路シミュレーションでも示していると思われる。

次に、③について結果を検討してみると、準最短距離Ｄ以外全ての値が大きい。それは、通過回数を不確定にしているので、判断が曖昧になり、無駄な探索を行っているからである。更に、初回の探索がその後の探索までにも、影響を及ぼしているものと考えられる。以上のことは、当初の予想通りであった。

そして、決まり切った方法である④と⑤について考察すると、Ｄに関しては、出口方向に優先的に進んでもトレモー法の②、③と比べてあまり顕著な差がないことがわかる。ところが、迷路Ａでの⑤による探索では、出口と反対方向に優先的に進めば、多大な距離を進まなければならないが、Ｄが極端に短くなった。その理由は、ＴＤが飛び抜けて大きな値になったことによる。これは正しくＴＤとＤとのトレードオフの関係を如実に表している。

２.２　その他の値について

初回探索距離Ｄ1については、当然ながら出口方向に進む拡張左手法が断然短い。他の方法が乱数で進路を決定するのに対して、拡張左手法は決まり切った規則により決定するので、余り効率良いとは言えないが、Ｄ1の面では確実に短くなっている。

試行回数Ｎについて考えると、迷路Ａの⑤以外その値はまちまちなので、ここで一定の判断を下すのは難しいと思われる。

最後に、探索方法とは直接関係ないが、ここで準最短距離の収束条件について、実際にシミュレーションした結果から言及する。迷路や探索方法をいろいろと変えて、実験で５回連続して同じ値を示した後に、十数回その後の探索距離を数回調べたが、全て同じ値になった。この結果から、連続して５回同じ値が出れば、準最短距離が収束したと見なして妥当であると言える。ただし、150回の全実験では、４回連続して同じ値になっても５回目で異なる値が出たことが３回あった。

２.３　結果と評価のまとめ

以上の内容をまとめると、次のようになる。

(1)準最短距離と総探索距離とはトレードオフの関係にある。

(2)出口が下又は上にある場合、出口方向に優先的に進む拡張左手法が初回探索距離については短い。

(3)初回探索距離と総探索距離から判断すれば、知能の設定①、②、③の順に数値が大きくなるので、知能の高さとしては ①〉②〉③と捉えることができる。

(4)準最短距離の収束は、同じ値が5回連続した場合に収束したと見なすことができる。これを一般化すれば、n次元迷路では、3次元迷路と同じ理由により、同じ値が2n-1回連続して出れば準最短距離が収束したと見なすことができる。

Ⅴ　おわりに

今回、立体迷路探索シミュレーションプログラムを

作成し、探索手法や知能レベルの設定と準最短距離等の各種数値との関係を調査して、その評価を行った。

ただし、今回の実験では、出口だけが異なる同じ迷路を対象としたので、実験対象が少ない。今後、多くの迷路を取り上げ､上記の結果を検証する必要がある。

また、シミュレーションプログラム以外に、関連として、人間自らが探索体験することができる立体迷路探索シミュレーションゲームをも作成した。ゲームそのものは平面に比べ遙かに難しいが、3次元重力空間でのシミュレーションゲームプログラムは平面の場合を少し拡張した程度ですむので(注10)、本稿では取り上げなかった。ところが、3次元を超え、更に、無重力空間になると、進行方向の区別が全く付かなくなるので、区画選択が極度に難しくなる(注11)。また、そのプログラムも複雑になりそうで、作成し甲斐がありそうである。Webサイトには、特に、迷路ゲームがいろいろと載せられているが[6]､[7]、無重力空間の迷路を取り扱ったものは少ないようである。

［注］

(注1)立体迷路と称する迷路の中には、他の意味に捉えられ、区画の集合が立体をなさいものがある。例えば、以下のようなものである。

①ゲームシミュレーション画像が立体的に表現されている平面迷路。

②平面迷路の拡張で、立体の表面だけに迷路が施されている。即ち、立体内部には迷路がない。

③立体交差する道路のように、高さが2の迷路で、上方から見れば、平面迷路と同じように迷路全体を一望できる。

(注2)今回の実験環境で試行調査すると、迷路規模は最大約750,000区画数位になる。

(注3)下方向優先拡張左手法と上方向優先拡張左手法との説明は、単に下と上の文字を入れ換えるだけで済むので、両者を一括して記述した。

(注4)周辺区画の中で通過回数が最少の区画を指す。

(注5)進路判定用の通過回数の観点から述べると、②が純然たる通過回数そのもので、①は②の通過回数に自由度を加え、③は通過回数そのものを確率0.2の割合で不確定にしている。

(注6)立体迷路Ａの出口は(39,64,40)で、Ｂは(39,64,20)、Ｃは(39,64,1)である。特に、④と⑤では、出口が上又は下、或いは中間にあるのかによりＤ$_1$が大きく異なるが、それらは予想された結果である。

(注7)今回の実験では、シミュレーションプログラムをJavaで作成したことから、ＯＳやＣＰＵの違いにより結果が左右されないので、複数台のパソコンを使用した。

(注8)Ｎは回数、ＴＤ、Ｄ、Ｄ$_1$は区画数を表し、特にＴＤについては単位を万で表示した。

(注9)Windows系のＯＳでは、シミュレーション風景が上手く描画されないので、本稿では、RedHat Linux ver9.0によりそれを表示させた。

(注10)重力がある場合、以下のような姿勢規則に基づいて立体迷路を移動する。

［重力空間での姿勢規則］

①平面上を移動する場合、直立姿勢を保ちながら、進行方向を正面に見据え迷路を探索する。

②上下に移動する場合、エレベータに乗ったようにそのまま直立姿勢を保って移動する。

(注11)重力がない場合、迷路の定義から前後左右上下の区別が付かないので、人間が前に進む時には、次のような姿勢規則を設定し、それに従って進むようにする。移動に際しては、背中に推進エンジンを背負って移動する。

［無重力空間での姿勢規則］

①頭を進行方向に、足をその反対方向に向け、胴体を進行方向と平行に保って移動する。

②進行方向と平行な同一平面上を移動する場合、背面を常に一定に保ち前後左右に移動する。

③進行方向と垂直に移動する場合、ある種の背面の方向変化規則に基づいて背面を決定し移動する。

［参考文献］

(1)加部通明、大規模迷路探索シミュレーション　-その１　平面の場合-、職業能力開発報文誌、第18巻第1号、2006年、P1-6

(2)http://www5d.biglobe.ne.jp/ ~stssk/maze/make.html

(3)http://ja.wikipedia.org/wiki/%E8%BF%B7%E8%B7%AF#.E8.BF.B7.E8.B7.AF.E3.81.AE.E8.A7.A3.E6.B3.95

(4)http://www.informatics.tuad.ac.jp/tenji/tenji03/shiragami-lab/199970120/3.doc

(5)http://www.inet-lab.org/ted/program/prog065.html

(6)http://computers.yahoo.co.jp/download/vector/win95/game/puzzle/maze/

(7)http://www.informatics.tuad.ac.jp/tenji/tenji03/shiragami-lab/199970120/gaiyou.html