芸術科学会論文誌 Vol. 21, No. 4, pp. 199 – 212 (2022)

おはなし迷路の制作を支援するツールのデザインと実装

岡野稀央隆 (非会員)　山中克久 (非会員)　松山克胤 (正会員)

岩手大学

Design and Implementation of Tools Making Storytelling Mazes

Kiotaka Okano　Katsuhisa Yamanaka　Katsutsugu Matsuyama

Iwate University

m18u @ iwate-u.ac.jp

概要

本研究では，おはなし迷路の制作を支援するツールのデザインと実装を行う．従来のおはなし迷路の制作が，紙のような静的なメディアで行われてきたのに対し，本研究ではコンピュータを活用した制作ツールを開発し，その実現性と有用性を議論する．おはなし迷路の制作プロセスは複雑であり，迷路の作成と並行して場当たり的に作文を行う必要がある．本研究では，このような複雑さを踏まえた上で，制作支援の方針を示し，対話的ツールをデザインする．具体的には，ストーリーの作成，ストーリーの迷路上への配置，そして，迷路パラメータ変更の３つのブロック間を行き来して作品を制作するユーザインタフェースを提案する．また，ストーリーの迷路配置を最適化問題ととらえて，確率的アルゴリズムを用いて解く手法を提案する．そして，本提案システムを使用して実際に作品を制作し，有用性に関する議論を行う．

Abstract

We design and implement tools to support the creation of storytelling mazes. While the conventional process of creating storytelling mazes has been done on static media such as paper, this study develops computer-based tools and discusses their feasibility and usefulness. The process of creating storytelling mazes is complex, the creator have to make stories in parallel with the construction of the maze. Based on this complexity, we present a direction for creation support, and design interactive tools. Specifically, we propose a user interface that allows users to create works by moving back and forth between three blocks: making a story, placing it on the maze, and changing the maze parameters. We also propose a method to solve the maze placement of the story as an optimization problem by using a stochastic algorithm. We show some results actually created using the proposed tools and discuss usefulness.

1 はじめに

「おはなし迷路」とは,「おはなし」すなわちストーリーが書き込まれている「迷路」のことである．筆者らが調べた限りでは，おはなし迷路は児童書作家でおもちゃ作家の杉山亮が考案したと考えられる．杉山は,「さんびきのこぶた」や「うさぎとかめ」のような，日本において広く知られている童話を題材としたおはなし迷路作品を多数発表している [1].

杉山のおはなし迷路では，迷路の１つのマス目に１つの文字が割り当てられている．おはなし迷路のプレーヤーは，スタートから１文字ずつ辿り，文章を読み進めながら迷路を進んでいく．そして，ゴールに到着すると同時に，題材である童話のストーリー（メインストーリー）が完成するようになっている．

普通の迷路には，通路の分岐がいくつもあり，これらの分岐点で間違った方向に進むと「行き止まり」に突き当たる．おはなし迷路においては，通路の分岐点で，ストーリーの分岐も発生する．そして，ゴール以外の行き止まりに到達したときには，メインストーリーとは別のストーリー（サブストーリー）が完成する．

1.1 おはなし迷路の制作

おはなし迷路の制作に関する文献に，府川の研究報告 [2] がある．府川は，学生におはなし迷路を作成してもらい，どのような作文が行われたかを国語教育の観点から考察している．報告には，学生がおはなし迷路を作成する様子の記述もあり，以下のような内容である．

- 多くの学生は,「スタート」の位置から，メインストーリーの文章を書き始め，迷路のような通路になるように折り曲げながら配置して,「ゴール」に到達させて，正解ルートを完成させる．その後で，メインストーリーから分岐点を作って，サブストーリーを付け加えていく手順で作成していた．
- 最初にメインストーリー（正解ルート）を完成させずに，サブストーリーを同時並行的に書き込みながら迷路を作っていく学生もいた．
- あらかじめゴールの位置を決めていなくて，思いがけないところがゴールになったケースや，文字数をうまく調整できず，マス目の一部を塗りつぶして処理するケースがあった．

1.2 制作の難しさ

作文と配置: おはなし迷路の制作プロセスは，事前に固定された文章を用意するのではなく，迷路の作成と並行してその場で作文を行う必要がある．文章の作成そのものが場当たり的に行われるため，途中までのストーリーや，迷路の都合による文字数の制約の状況下で文章を作成する状況が生じる．これは，制作方法として，思いがけない面白さが生まれる可能性もあるが，作品として制作に失敗する可能性も含んでいる．

一方，作文した文章を迷路に配置することは計算量的に難しいタスクであることが示されている．固定された（分岐のある）文章を迷路のマス目に配置する問題でさえ NP 完全であることが既に示されている [3, 4]．よって，迷路サイズが大きい場合，コンピュータを用いたとしても現実的な時間で正確に解くことは難しい．このことからも文章を迷路のマス目に配置するタスクの難しさがわかる．

編集・修正: 制作上の特徴として，迷路の修正が難しいことも挙げられる．例えば，制作の途中で，あるルートの配置を変更したいと考えた時に，もし，そのルートの変更先に別のルートがすでに存在する場合は，その別ルートの配置や文章も変更する必要がある．このように，あるルートを修正するためには，隣接するルートの修正を連鎖的に行う必要が生じることもあり，このことが，迷路の修正を困難なものにする．他の修正の例としては，制作途中での挿絵の挿入や，迷路サイズの変更，ストーリーやゴール位置の変更などが挙げられる．

このように，おはなし迷路の制作に関する問題は，制作過程が複雑で失敗に結びつく要因が多いこと，そして，失敗した場合の修正が困難であることにある．良い作品を制作するためには，高度な作文能力と想像力を必要とするだけでなく，場合によっては再び最初から制作する必要もあるなど，多大な労力と時間も必要になる．

本研究は，おはなし迷路の制作を支援するツールのデザインと実装を行うものである．従来のおはなし迷路の制作が，紙のような静的なメディアで行われてきたのに対し，本研究ではコンピュータを活用した制作ツールを開発し，その実現性と有用性を議論する．

1.3 本論文の貢献

本論文の芸術科学コミュニティへの貢献を以下に挙げる．(a) おはなし迷路の制作プロセスに NP 完全のタス

クが含まれていることに言及し，それを踏まえた支援の方針を示したこと．(b) おはなし迷路の制作を支援する対話的ツールをデザインしたこと．(c) ストーリーの迷路配置を行う問題を最適化問題ととらえて，確率的アルゴリズムを用いて解く手法を提案したこと．

2 関連研究

2.1 迷路の自動生成に関する研究

迷路生成アルゴリズムの研究は古くから行われており，「棒倒し法」や「穴掘り法」など多数の迷路生成アルゴリズムが存在する [5]．本研究では，迷路形状はグリッド，すなわち，マス目の隣接関係は上下左右の４方向であることを想定している．上記の迷路生成アルゴリズムを使うことで，グリッド形状の迷路を自動的に生成することが可能であるが，通路の長さや分岐の位置などを操作することはできない．

コンピュータグラフィックスに関連する迷路生成の研究に，入力された画像に類似する迷路を生成する手法 [6, 7, 8] がある．それぞれ，基本図形上の点に揺らぎを加える [6]，入力画像の各ブロックを３つのタイプ（方向，螺旋、ランダム）に振り分ける [7]，そして，反応拡散系のシミュレーションに基づく [8] 手法により，迷路の壁を入力画像のストロークに似せることで，画像に類似する迷路を生成する．これにより，迷路に対して画像という付加的な要素を追加できるようになる．これに対し，本研究は，迷路にストーリーの分岐を含む文章を付加するところに独自性がある．加えて，上記の手法は通路の長さや分岐の位置などを操作することはできないために，本研究に直接的に適用することはできない．

正解ルートの形状が，入力画像に類似するような迷路を生成する手法 [9, 10] も存在し，画像の二値化を行った後で，前景色の領域に対して，ハミルトン路に基づく手法 [9] や，サイクルの拡大に基づく手法 [10] により，正解ルートを生成する．本研究ではストーリーの分岐を行うなど，より詳細な制御を必要とするため，正解ルートかどうかを二値的に考える上記のアプローチは適用できない．

難易度を考慮した迷路生成の研究も存在し [11, 12]，分岐の数や通路の長さのような，迷路の難易度に関するパラメータを指定して，グリッドグラフのパス探索 [11] や遺伝的アルゴリズム [12] により，パラメータに応じた迷路を生成する．しかしながら，上記の研究は，迷路単位でパラメータを指定するものであり，通路の長さや分岐の位置などを個別に操作することはできない．通路構造に対応するテクスチャ画像を用いて，スタイルを考慮した迷路をレンダリングする手法も提案されている [13] が，構造の生成はスパニングツリーに基づくものであり，具体的な構造を操作できないために，本研究の目的を達成することはできない．

2.2 おはなし迷路に関する研究

おはなし迷路の研究は，府川の研究報告 [2] と，筆者らの研究報告 [14] の他には見当たらない．府川の研究報告は，先述のように，作成された文章を考察するものであり，制作の支援を行うものではない．

おはなし迷路の制作プロセスの一部である，固定長の木構造を迷路グリッドに配置する問題は NP 完全であることが知られている [3, 4]．一方，著者らは，この迷路グリッドへの配置を遺伝的プログラミングを用いて行ったところ，6×6 マスという極めて小さいサイズの場合には可能であることを示した [14]．

しかし，このことは，より大きな迷路においては，遺伝的プログラミングによる配置では，現実的な計算時間で収束することが極めて難しいことを示している．さらに，おはなし迷路の制作プロセスには，ストーリーを迷路へ配置するだけでなく，ストーリーの作成も含まれるため，[14] のような木構造データを迷路上に配置することに着目するだけでは，ツールとしては不十分である．

おはなし迷路とは作品の方向性は異なるが，迷路上に文章を配置することで，正しい文法を導く迷路生成手法 [15] が存在する．この研究では，迷路の分岐付近に文字を配置して文法の選択肢を提示するために，作品の性質上，文字が配置されないマス目が多くても問題ないのに対し，おはなし迷路は，ストーリーを読み進めた上で迷路を分岐させるものであり，全てのマス目に文字が割り当てられることを基本とする．また，[15] では迷路生成に「壁のばし法」を採用しているために，本研究に必要な通路の長さや分岐の位置などを操作することはできない．

3 ユーザインタフェースとデータ構造

本研究のユーザインタフェースデザインの方針は，大局的には従来の制作プロセスを踏襲しつつ，部分的に支

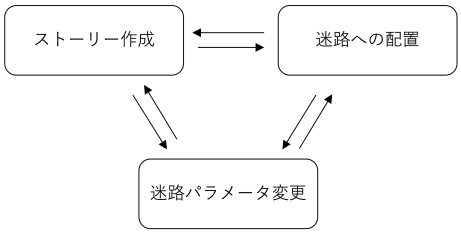

図 1: 提案ユーザインタフェースの概要

援可能なところをツールとして提供するものである．第 2 章で述べたように，分岐を含むストーリー文章を迷路上に配置する実用的なアルゴリズムの開発が困難であることが，計算量的な観点から証明されている．このことは，ストーリー作成や迷路配置に関しては，制作者が場当たり的に行うことそのものは変更できないことを示している．したがって，本研究の目的は，制作者の場当たり的に行われる迷路制作において，コンピュータを活用した制作ツールを開発することであり，従来的に行われてきた紙メディアでの制作における難しさの緩和を目指すものである．本章では，提案ユーザインタフェース，および，実現に必要なデータ構造について記述する．

3.1 提案ユーザインタフェースの概要

図 1 は，本提案ユーザインタフェースの概要を表すブロック図である．先述のように，一般的なおはなし迷路の制作は，ストーリーの作成と，ストーリーを迷路上に配置するプロセスを行き来して行われる．本提案のユーザインタフェースは，これに，迷路パラメータの変更も加える構成とし，これら 3 つのブロック間を行き来して作品を制作する．各ブロックについて次節で説明する．

3.2 迷路パラメータ

本研究で扱う迷路パラメータは，迷路サイズ（幅と高さ），スタート位置，そして，各マス目に対する文字配置の可否である．本提案ユーザインタフェースでは，迷路パラメータを制作の途中で動的に変更できるようにする．紙メディアのような従来の制作環境では，最初に迷路サイズやスタート位置を決定する必要があるが，本提案のように，迷路パラメータを動的に変更できるようにすることで，より柔軟に迷路制作を行えるようにする．

動的な迷路パラメータ変更の有用性を示す例として，メインストーリーの文字数と，スタート・ゴール間の迷路長さの偶奇が異なる場合は，どうやってもゴール位置にメインストーリーの最後の文字を配置することができない．このような場合に，迷路パラメータを変更して偶奇や長さを同じにすることで，ストーリーを変更しなくても解決できる可能性がある．

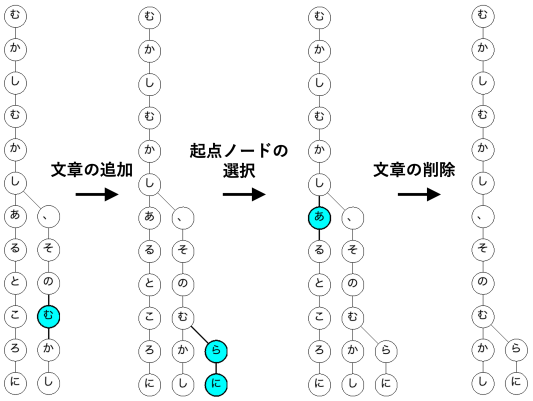

図 2: ストーリー修正の様子

また，各マス目に対して，文字の配置を許可するかどうかも設定できる．これにより，迷路の外形デザインや，画像の配置などが可能となる．

3.3 ストーリーの作成と修正

おはなし迷路のストーリーをコンピュータで表現するには，ストーリーの分岐を考慮したデータ構造を採用する必要がある．本研究では，ストーリーを表すデータ構造に，１文字を１つのノードとする木構造を採用する．ストーリーの分岐は，木構造の枝分かれ，すなわち，子ノードを複数持たせることで表現する．

ストーリーの作成や修正を行うユーザインタフェースは，木構造データのビューワーを基に行う（図 2）．ストーリーの作成は，開始点となるノードを選択して，テキストを入力することで行う．システムは，選択されたノードを起点に，入力されたノードを追加する形で木構造データを更新する．また，ユーザは，任意の部分木を指定することも可能であり，その部分木を削除することで，ストーリーを削除できる．

3.4 ストーリーの迷路配置

3.4.1 迷路配置のデータ表現

3.3 節で作成した木構造データを，迷路上にどのように配置するかを表現するためには，各ノード n_i に位置データを持たせる必要がある．

杉山の作品を含め，一般的に見られるおはなし迷路は，格子状（グリッド）のマス目が用いられており，本

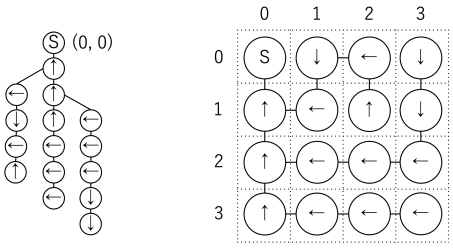

図 3: 迷路配置のデータ表現．左: ルートノード以外のノード n_i に親ノードの相対的な位置を表す変数 d_i を持たせる．右: 木構造の走査により迷路上に配置する．

研究でも迷路のマス目はグリッドであるものとする．したがって，各マス目の位置座標は，二次元の整数で $\boldsymbol{x} = (x, y) \in \mathbb{Z}^2$ と表すことができる．ここで，スタートのノード（木構造データのルート）以外のノードは，必ず 1 つの親ノードを持つことに着目し，かつ，

- エッジで接続されたノード間は，迷路上でかならず隣接させる．すなわち，隣接ノード間の迷路上でのマンハッタン距離は常に 1 とする．
- 迷路形状はグリッドなので，隣接関係は上下左右の 4 方向に限られる．

ことを制約として定める．本研究では，この制約を常に満足するように各ノードの位置を表すために，ルートノード以外の各ノード n_i に，その親ノードが迷路上で上下左右のどこにいるかを表す変数 $d_i \in \{\uparrow, \downarrow, \leftarrow, \rightarrow\}$ を持たせる（図 3 左）．なお，ルートノードの位置座標 $\boldsymbol{x}_0$ だけは迷路パラメータとして指定する（3.2 節）．

実際の迷路配置，すなわち，各ノードの迷路上への配置は，ルートノードを $\boldsymbol{x}_0$ に配置した後に，木構造の走査を行い，各ノード n_i を d_i に従って迷路上に配置していけば良い（図 3 右）．

ノードが作成された時の初期値として，$\{\uparrow, \downarrow, \leftarrow, \rightarrow\}$ のいずれかをランダムで割り当てる．ただし，初期値を割り当てた直後に，作成されたノードに対して，4 章の最適化を実行する．

3.4.2 エラーの可視化

前節のデータ表現を採用することで，常に隣接関係を保つ迷路配置が可能となる．しかし，原理的に，迷路サイズの外側にノードが配置されたり，同じマス目に複数のノードが配置されたりするなど，迷路としては破綻している可能性もある．

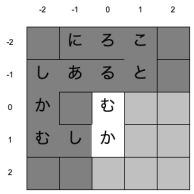

図 4: Deviate エラーの可視化例

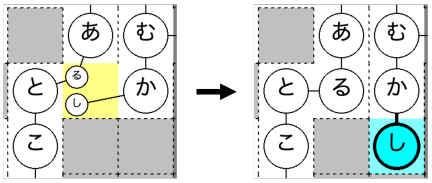

図 5: 手動での配置の修正

本提案システムでは，現在の迷路配置に不具合が生じている場合に，その情報をユーザに提示する．本提案ユーザインタフェースでは，以下の可視化を行う．

Deviate: ノードが迷路サイズの外側に位置していることを示すために，迷路の外側のマス目を，内側と比べて暗い色で表示する (図 4).

Overlap: ノードの重複が生じていることを示すために，ノードの重複が生じているマス目を通常とは異なる色（本論文では黄色）で表示するとともに，木構造データの配置を表す描画モードを実装する．この描画モードでは，木構造データをエッジ付きで描画することで，接続関係を視認できるようにする（図 5(左)）．重複しているノードの可視化は，マス目を再分割して，それぞれのノードを，分割領域内で異なる位置に配置することで視覚的に重なることを避ける．

Rock: 文字を配置できないように設定（3.2 節）したマス目にノードが侵入していることを示すために，マス目を通常とは異なる色（本論文では黄色）で表示する．

3.4.3 手動での配置の修正

本提案システムでは，迷路の不具合を解消するために，あるいは，迷路配置をより意図するものに近づけるために，迷路配置を手動で修正できるようにする．

ユーザインタフェースとしては，迷路グリッド上で修正したいノードを選択した後に，移動先のマス目を選択して設定する．具体的な操作手段として，ノードのドラッグによる移動や，選択されたノードのプロパティ

画面でドロップボックス選択する方法も実装する．このユーザ操作で，各ノード n_i が持っている変数 $d_i \in \{\uparrow, \downarrow, \leftarrow, \rightarrow\}$ を変更させる（図 5）．

本提案ユーザインタフェースは，3.4.1 節の制約を満たし続けるために，移動先のマス目の選択において，親ノードとのマンハッタン距離が 1 になるマス目以外は選択できないようになっている．具体的には，ドロップボックス選択においては，選択肢は $\{\uparrow, \downarrow, \leftarrow, \rightarrow\}$ とする．そして，ノードのドラッグ移動においては，親ノードとのマンハッタン距離が 1 のマス目にドロップした時だけ移動させる．なお，本節の操作では，ノードの移動先が迷路サイズの外側であっても良く，また，移動によって同じマス目に複数のノードを配置することも可能であり，その場合は 3.4.2 節のエラーの可視化が行われる．

3.5 ターゲット座標の指定

これまでに説明したユーザインタフェースを用いることで，手動による迷路配置を行うことができる．しかしながら，これだけだと迷路配置のプロセスが全て手動であり，エラーの解消には多大な労力が必要であるため，ユーザの負担はまだ大きい．

他にも，例えば，ユーザが，あるノード n_i を迷路座標 $\boldsymbol{x}_i$ に配置したいとする．この配置を手動で行うためには，ルートノードから 1 つずつノードを辿る形で，ノード n_i の位置が $\boldsymbol{x}_i$ に来るような修正を検討する必要がある．

本研究では，このような問題に対して，ユーザの負担を軽減するために，手動による迷路配置に加えて，最適化に基づく自動迷路配置の機能も実装する．具体的な最適化の手法は次章で説明するが，本節では，最適化の際に使用するパラメータである「ターゲット座標」の指定について説明する．

本研究では，各ノード n_i に，ターゲット座標を保持する変数 $\boldsymbol{t}_i \in \mathbb{Z}^2$ を持たせて，最適化を適用した時に，ノード n_i をできるだけ $\boldsymbol{t}_i$ の近くに配置するようにする．

ターゲット座標の指定を行うユーザインタフェースとしては，指定したいノードを選択した後に，ターゲット座標のマス目を選択して設定する．また，座標の数値的な入力もできるように実装する．なお，ターゲット座標は，迷路サイズの範囲内であればどこでも指定できる．また，指定されないノードの $\boldsymbol{t}_i$ は未定義とする．

ターゲット座標が設定されていることを表現するために，ターゲット座標が定義されているノードが選択された時に，そのノードの現在の座標（本論文では水色）と，指定された位置 $\boldsymbol{t}_i$（本論文では紫色）をハイライトさせる（図 6）．このハイライトは，現在の座標と $\boldsymbol{t}_i$ との差分（エラー）の可視化としても機能する．本研究で実装するエラーの可視化は，3.4.2 節の 3 つに，本節の可視化を加えた 4 つである．

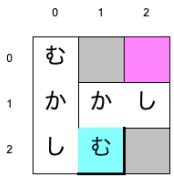

図 6: ターゲット座標 $\boldsymbol{t}_i$（紫色）の表示例

ターゲット座標を設定できるノード数に制限は無いが，本提案ユーザインタフェースでは，ゴールの位置を指定したり，迷路上の演出を行うために比較的少数のノードに設定されることを想定している．

4 最適化を用いた迷路配置

前節のターゲット座標の指定によって，いくつかのノード n_i にはターゲット座標 $\boldsymbol{t}_i$ が設定されている．本研究では，設定されたターゲット座標をできるだけ反映させるような迷路配置を自動で行う問題を最適化問題ととらえ，この最適化手法を設計する．加えて，ターゲット座標が指定されているかどうかにかかわらず，最適化によって，破綻していない迷路を自動的に生成できるようにする．

本提案ツールでは，ユーザが，最適化を適用する範囲を選択し，「最適化」ボタンを押下することで最適化が実行されるようにする．最適化の適用範囲は，文章の木構造全体を選択することができるのに加えて，部分木を選択できるようにして，最適化の適用範囲を絞ることができるようにする．また，最適化の実行は制作の途中でいつでも行えるようにする．

本論文が対象とする最適化問題は，組み合わせ最適化問題に分類できるものであるが，組み合わせ爆発が生じるために全探索を行うことが困難である．一般に，このような問題は，遺伝的アルゴリズム（Genetic Algorithm: GA）や擬似アニーリングのような確率的アルゴリズム

を用いて準最適解を求めることが多く，本研究でも確率的アルゴリズムを用いて最適解を求める．

4.1 コーディング

最適化を適用させるためには，木構造全体あるいはユーザが選択した部分木について，迷路配置の方法をコーディングする必要がある．筆者らは [14] で，迷路の制作途中を含む木構造データが可変長であること，そして，ストーリー分岐があるために子ノードの数が不定であることから，遺伝的プログラミング（Genetic Programming: GP）を用いた木構造データのコーディングを試行した．しかし，ノード数が 50 程度でも実用的な時間で収束しなかった．

そこで，本研究では，最適化の対象範囲を比較的少数（30 程度）のノードで構成される部分木に限定する方針を採用して，遺伝的プログラミングではなく，より高速に収束が期待できる遺伝的アルゴリズムを使用する．そして，ストーリーの木構造を，制作過程全体を通した可変長のコードとして表現するのではなく，ユーザが最適化を行う対象範囲として選択された部分だけを，固定長のコードとして定義するアプローチをとる．すなわち，最適化を実行する度に，その最適化のための固定長のコードを定義する．

具体的には，3.4.1 節で説明した迷路配置のデータ表現（図 3 左）のうち，ユーザが指定した範囲（木構造全体や部分木）を深さ優先順で平坦化することで 1 次元配列を作成し，これをコードとして使用する．

4.2 評価関数の定義

本研究では，評価関数を

$$E = w_T E_T + w_D E_D + w_O E_O + w_R E_R \qquad (1)$$

と定義する．各項は以下のようである．

Target: E_T は，指定されたターゲット座標と，現在の座標との差分を評価する．

$$E_T = \sum_i T_i$$
$$T_i = \begin{cases} 0 & (\boldsymbol{t}_i \text{は未定義}) \\ \mathrm{dist}(\mathrm{pos}(n_i), \boldsymbol{t}_i)^2 & (\text{それ以外}) \end{cases} \qquad (2)$$

ここで，関数 dist() はマンハッタン距離を計算する関数，関数 pos() は引数ノードの現在の座標を取得する関数である．

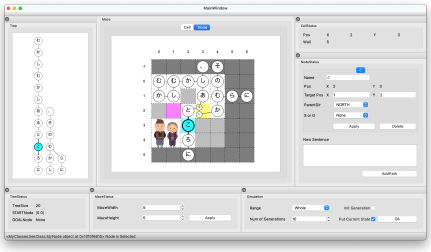

図 7: ユーザインタフェースのスクリーンショット

Deviate: E_D は，ノードが迷路パラメータで定められたサイズの内側に配置されていることを評価する．

$$E_D = \sum_i \mathrm{dev}(\mathrm{pos}(n_i))^2 \qquad (3)$$

ここで，関数 dev() は，引数の座標が迷路サイズの外側に位置する場合に，どれくらい外側に位置しているかをマンハッタン距離で算出する関数である．

Overlap: E_O は，迷路のマス目でノードの重複があるかどうかを評価する．

$$E_O = \sum_{x=0}^{W-1} \sum_{y=0}^{H-1} \max(\mathrm{num}((x,y)) - 1, 0)^2 \qquad (4)$$

ここで，関数 num() は，引数の座標のマス目に存在するノードの個数を取得する関数である．また，W と H は，迷路サイズの幅と高さをそれぞれ示す．

Rock: E_R は，ユーザが文字を配置しないように設定（3.2 節）したマス目に対する評価を行う．

$$E_R = (\sum_{x=0}^{W-1} \sum_{y=0}^{H-1} \mathrm{rock}((x,y)) \cdot \mathrm{num}((x,y)))^2 \qquad (5)$$

ここで，関数 rock() は，引数の座標のマス目が文字配置しない設定であれば 1 を，それ以外なら 0 を返す関数である．すなわち，文字配置しない設定のマス目に侵入しているノード数で評価する．

重み係数: 式 (1) の w_T, w_D, w_O, w_R は重み係数であり，本論文の制作例では全て $w_T = w_D = w_O = w_R = 1$ としている．最適化の実行は，主に作品制作の途中で行われることを想定しており，マス目の余白が十分にある状態でノード数 30 程度の部分木を対象に最適化を実行すれば，ほとんどの場合は $E = 0$ で（ターゲット座標に原理的に到達しない場合は $E = E_T$ で）収束する．この

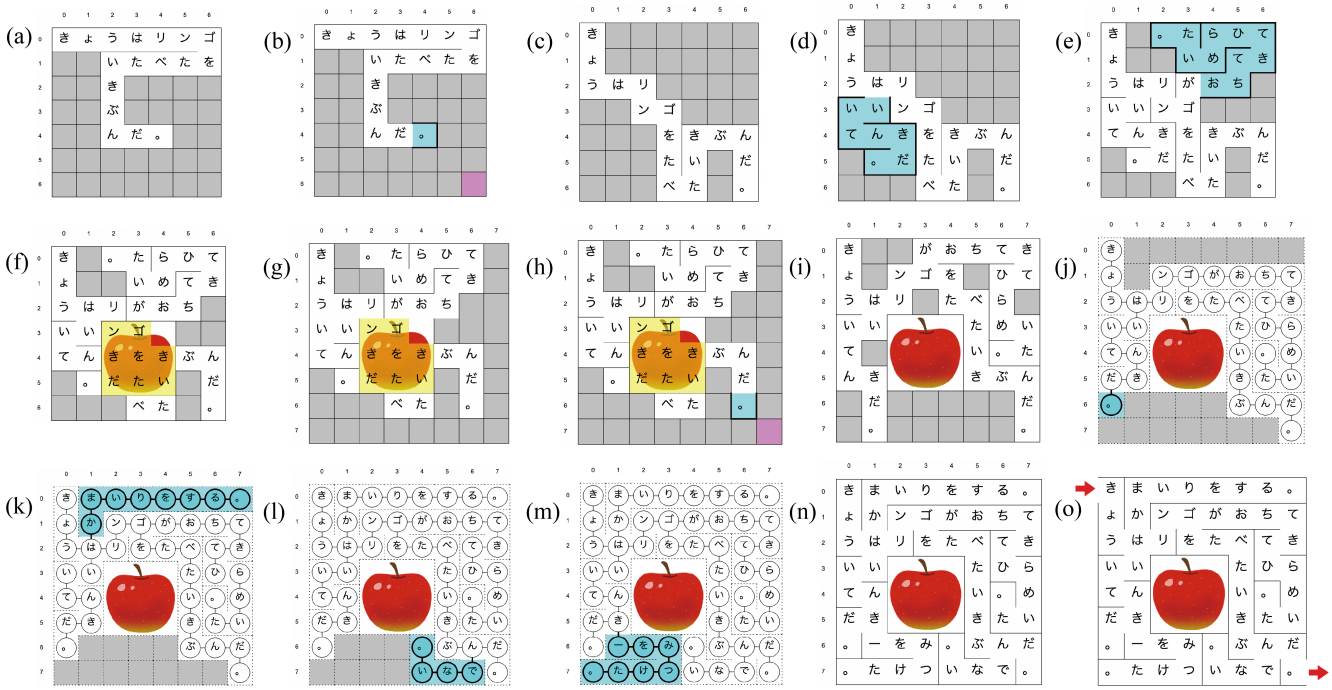

図 8: 作品例「リンゴ」.

ような場合は，評価関数の項目の間にコンフリクトが生じることなく収束するため，本論文では重み係数は 1 としている．

5 実行結果

上記のユーザインタフェースと最適化処理を行うシステムを実装した．本研究では最適化に GA を使用した．実装は Python 言語で，木構造データに node と anytree，GUI に PyQt5，GA に DEAP ライブラリをそれぞれ使用した．これまで示した図 2，4，5，6 は，実装した UI のスクリーンショットである．

一部の GA のパラメータは UI 上でユーザが任意に値を設定できる．具体的には，最適化の対象となる部分木，世代数，前回シミュレーションの結果を引き続き使用するかどうか，現在の個体を初期世代に追加するか否かを選択できる．

他にも，制作途中のデータのセーブとロード機能，そして，迷路上に画像を配置する機能も実装した．ユーザインタフェース上の各パネルはフローティング可能であり，パネルの配置は自由にカスタマイズできる（図 7）．

5.1 作品制作例

本提案システムを用いた迷路制作の様子を以下に示す．迷路の制作には，MacBook Pro（CPU: Apple M1 Pro，メモリ: 32GB）を使用した．

画像の挿入: 図 8 は，画像の挿入を含む作品制作の例である．まずは，メインストーリーの文章を入力 (a) する．次に，ゴールの位置として，メインストーリーの最後の文字が，迷路の右下に配置されるようにターゲット座標を設定 (b) する．ここで，文章全体に対して最適化を実行 (c) させて，ゴールの文字が右下に配置されたことを確認する．次に，分岐する別の文章を 2 つ入力 (d, e) する．そして，画像の挿入 (f) を行い，もう少し文章を追加するために迷路サイズを変更 (g) する．迷路サイズの変更に伴い，ゴールの位置を再設定 (h) する．ここで，木構造全体に最適化を実行 (i) し，画像と文字との重複が解消され，ゴールの文字が右下に配置できたことを確認した．そして，4 箇所ある 1 マスの空白が存在しないようにするために，手動での修正 (j) を行う．8 マスの空白を埋めるストーリー (k) と，11 マスの空白を埋めるストーリー (l, m) を入力して，迷路を作成 (n) した．最後に，画像処理ソフトウェアを使用して，入口と出口の矢印を追加 (o) して，作品が完成した．作品制作 (a)-(n)

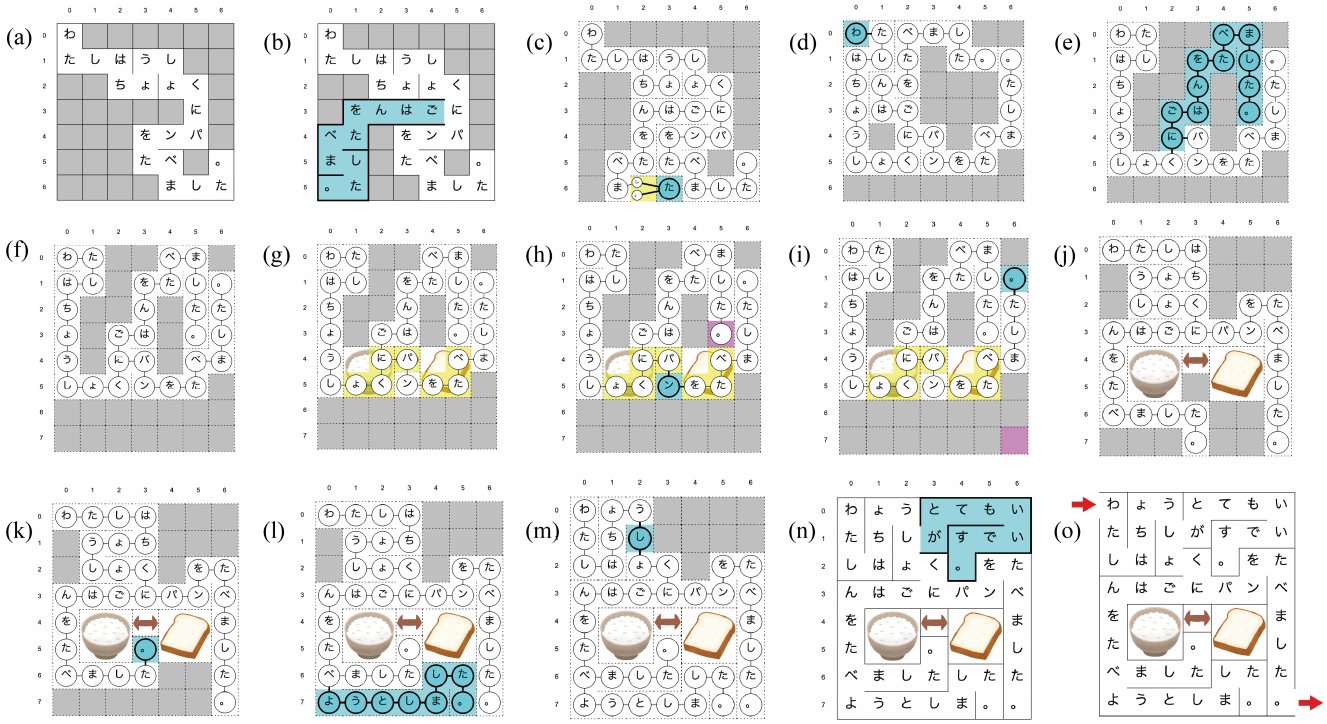

図 9: 作品例「パンとごはん」.

に要した時間は 10 分程度である．各ステップはすべて 1 分未満で行われており，(c) の最適化は 2 秒，(i) の最適化は 50 秒程度で収束した．メインストーリーは事前に考案済みで，画像も取得済みであり，これらは制作時間には含めていない．

特定の位置に文字列を配置: 図 9 は，迷路上の特定の位置に文字列を配置する例である．はじめに，メインストーリーの文章 (a) と，分岐するサブストーリー (b) を入力する．サブストーリーの近くに，2 つの余白領域があるので，1 つになるように手動での修正を試みたところ，ノードの重複が発生した (c)．そこで，文章木構造全体に対して最適化を実行 (d) し，その結果生じた 1 マスの余白を調整するためにも，試しにサブストーリーに対して最適化を実行した (e)．ここで，ゴールは右下とし，メインストーリーの最後の文字が右下に配置できるように，迷路サイズを変更 (f) しておく．ここで，パンとごはんの画像を挿入 (g) し，画像のすぐ上のマス目に「パン」と「ごはん」の文字をそれぞれ配置したいので，この 5 文字 (h) とメインストーリーの最後の文字 (i) にターゲット座標を設定する．ここで，木構造全体に最適化を実行 (j) し，画像と文字との重複が解消され，指定したターゲット座標の位置に文字が配置できたことを確認した．1 マスの空白を手動で修正 (k) し，8 マスの空白を埋めるストーリーを入力 (l) する．2 マスの空白を手動で修正 (m) し，9 マスの空白を埋めるストーリーを入力 (n) した．最後に，画像処理ソフトウェアを使用して，入口と出口の矢印を追加 (o) して，作品が完成した．作品制作 (a)-(n) に要した時間は 10 分程度である．各ステップはすべて 1 分未満で行われており，(d) の最適化は 5 秒，(e) の最適化は 2 秒，(j) の最適化は 10 秒程度で収束した．メインストーリーとサブストーリーの 1 つは事前に考案済みで，画像も取得済みであり，これらは制作時間には含めていない．

サイズの大きな作品の制作例: 図 10 は，本提案ツールを使用して，大きなサイズの作品を制作する過程の例を示す．制作プロセス自体は上記の 2 作品と同様であり，メインストーリーとサブストーリーの作成，画像の配置，迷路配置の修正を行い，空白のマスを埋めるように制作を進める．最初の部分を抜粋して説明すると，メインストーリーの冒頭部分を作成 (a) し，画像の配置とともに，「あかずきん」などの文字が画像に沿って配置されるようにターゲット座標を設定した後に最適化を実行 (b) す

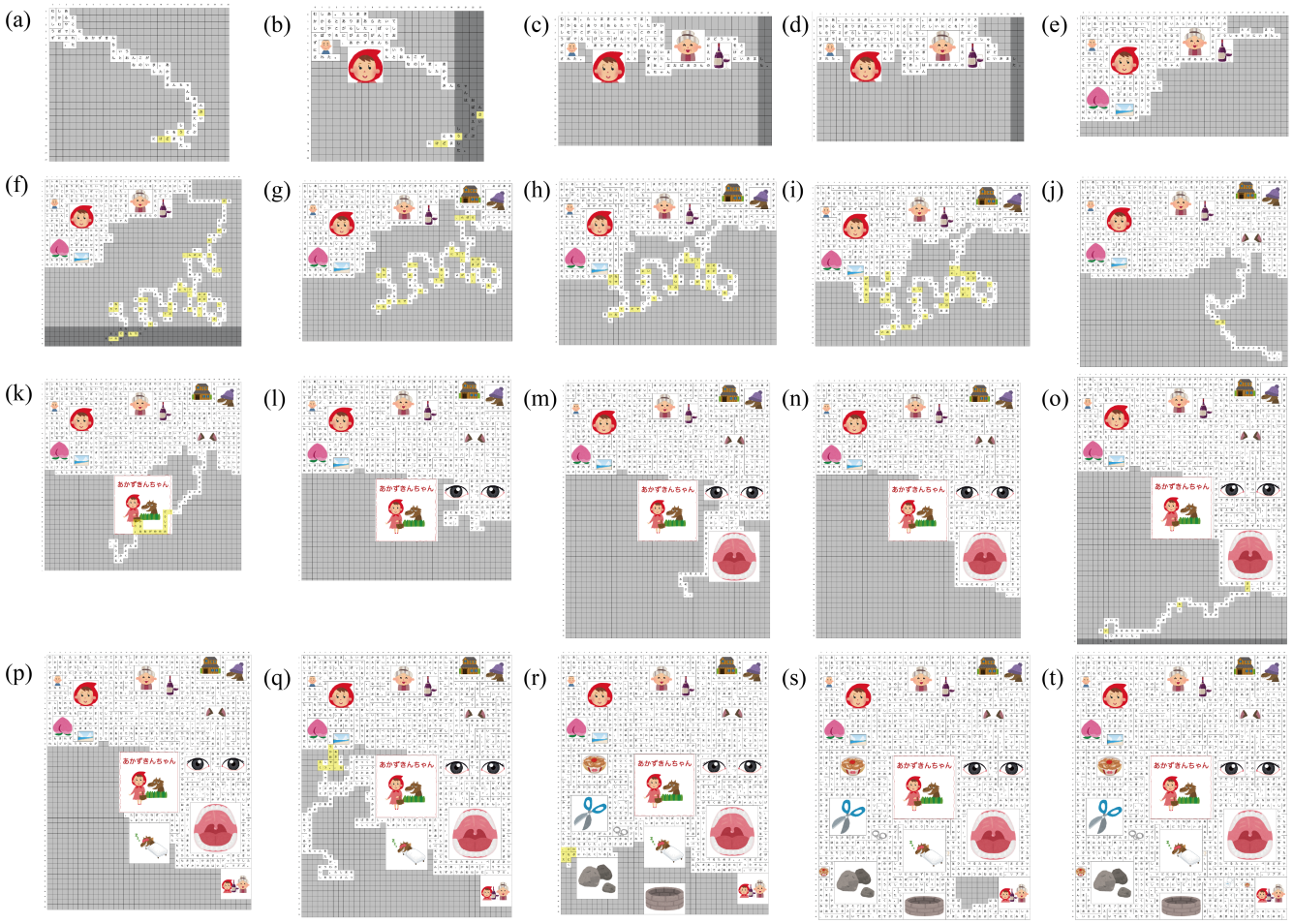

図 10: 作品制作過程例「あかずきんちゃん」.

る．「おばあさん」や「ぶどうしゅ」などもターゲット座標を設定して最適化を行った (c)．サブストーリーの考案や配置 (d, e) も同様に行い，メインストーリーの冒頭部分に関する迷路配置が完成した．そして，メインストーリーを延長 (f) し，それに伴うサブストーリーと画像の追加，ターゲット座標の設定，最適化と手動の修正を適宜行い，重複を解消しつつ空白のマスを埋めるように制作を進めていく．以降の制作途中のスクリーンショットを (g)-(t) に示す．メインストーリーの延長とともに迷路サイズの拡大 (e, g) も行い，最終的には 40x50 の作品となった．最後に，画像処理ソフトウェアを使用して，入口と出口の矢印を追加して，作品が完成した（図 13）．本作品の制作には，6 時間程度を要した．ただし，メインストーリーとサブストーリーを考案する時間や，挿絵の画像を Web 検索して取得する時間も制作時間に含めている．内訳は，おおよそではあるが，挿絵に使用する画像の検索と加工に 1 時間，ストーリーの考案に 2.5 時間，最適化や手動での修正に 2.5 時間程度を要した．実際には，限られた文字数でどのような文章を配置するかのように，配置とストーリーの考案が混在する工程が多く，制作時間の正確な測定は難しい．

5.2 最適化に関する実験と議論

本提案手法の最適化は，適用対象のノード数が 30 程度で，マス目の余白が十分にあれば（図 8(i) や図 9(j)），50 秒程度で収束する．GA は確率的探索手法であるため，最適化を実行する毎に異なる迷路が生成される．例えば，図 11 は，図 8(h) の状態から，木構造全体に最適化を実行した例であり，いずれも $E = 0$ で収束している．ユーザは，最適化を何回も実行することで，意図した迷路配置が得られるかどうかを試行することができる．

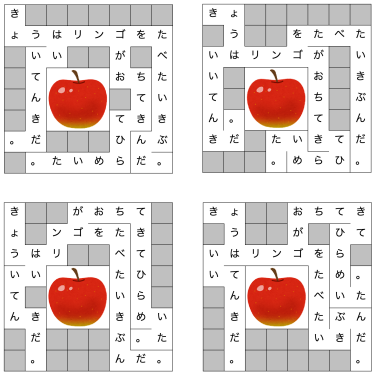

図 11: 最適化による迷路生成結果例.

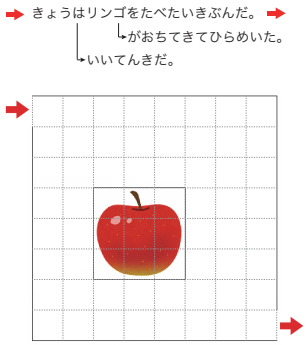

図 12: 人が配置する場合の時間計測に用いたシート.

ユーザは，最適化の実行時に，現在の配置情報を，初期値として GA の個体に追加するかどうかを選択できる．本手法の最適化は，初期値が最適解に近い配置であるほど，より高速に収束する傾向がある．また，初期値に近い配置結果となる傾向がある．なお，図 8(h) や図 9(i) の状態で，初期値を完全にランダムとしても 50 秒程度で収束する．

ここで，図 8(i) から (j) の最適化を対象に，人が文字を迷路配置した場合にかかる時間を計測した．計測は，図 12 を A4 サイズで印刷し，被験者に，おはなし迷路について説明した後で，ストーリーを分岐させて迷路上に配置すること，リンゴの画像上には配置しないこと，スタートとゴールの位置にそれぞれメインストーリーの「き」と「。」を配置する必要があることを説明する．また，図 12 で使用するストーリーの全文字数（36 文字）は，配置可能なマス目の数（55 マス）より少ないために，マス目全部は埋まらないことも説明する．そして，印刷した紙と鉛筆，消しゴムを使用して，分岐のあるストーリーを迷路のマス目に配置・記入してもらい，完成までにかかる時間を計測した．なお，印刷した用紙は何枚でも使用できることとした．この実験を 10 代-50 代の男女 8 名に対して行った結果，4 名が滞りなく記入できて，1 分 30 秒-2 分 43 秒（平均 2 分 9 秒）であった．他の 4 名のうち，2 名は事前に配置を計画した後で記入を行い，それぞれ 3 分 25 秒, 6 分 12 秒であった．そして，残り 2 名は当てはめの途中で完成しないことが発覚してやり直しがあり，それぞれ 6 分 38 秒，11 分 35 秒であった．本手法の最適化は，上記の作業を 1 回のクリックと 50 秒程度の待ち時間で実行することができるため，本最適化は迷路制作の効率化に資するものであるといえる．

本提案の最適化手法は，図 8(n) や図 9(n) のような，いちど完成した迷路の木構造全体に対して最適化による配置を実行しても $E = 0$ での収束に失敗する．ノード数 N に対して，迷路配置の組み合わせ数は高々 4^{N-1} 通り存在する．木構造を迷路のマス目に配置する問題は NP 完全であることが知られているため，この問題を現実的な時間で解くことは難しい．

5.3 議論

5.1 節に，「リンゴ」と「パンとごはん」の作品制作過程を示した．これらの例には，迷路サイズの変更や，迷路配置の修正が含まれているが，もし，この修正を紙メディア環境で制作する場合を考えると，制作開始時点までロールバックが必要である．これに対し，本システムの機能を用いることで，比較的短時間で修正できた．本提案ユーザインタフェースにおける迷路パラメータの動的変更や最適化による配置機能が，従来の紙のような静的メディア環境における制作上の難しさを緩和できたといえる．

本研究のアプローチは，おはなし迷路の制作が場当たり的に行われることを前提としており，おはなし迷路の制作プロセスそのものを変革するものではない．したがって，特に制作の終盤で，残されたマス目を全部埋めるような文字数でストーリーを考える必要があることや，手動による配置の試行錯誤が必要であることは，本質的には従来の制作と同様である．ただし，マス目の余白が十分にある場合では，30 文字程度の文章の配置を自動で行うことができるため，特に序盤から中盤にかけては，文字数や空白マスの形状を気にすることなく文字を入力することができ，ユーザはストーリー制作だけに焦点を当てることができる．これは，言葉遊びの性質が強

いおはなし迷路において大きなメリットである．

最適化による迷路配置の対象範囲として，任意の部分木を選択できるようにすることで，序盤は木構造全体を対象に，終盤は特定のルートのみを対象にするように使い分けて効率的に制作を行うことができた．ただし，やはり理想的には巨大な木構造全体に対して最適化が収束できれば良い．これについては，今後，評価関数や最適化手法の改良を行っていく．

6 まとめ

本研究は，おはなし迷路の制作を支援するツールのデザインと実装を行った．おはなし迷路の制作が場当たり的に行われる必要があることを前提として，３つのブロックで構成される対話的ツールのデザインを行い，ストーリーの迷路配置を行う確率的アルゴリズムを開発した．そして，作成したシステムを使用して実際に作品の制作を行い，有用性に関する議論を行った．今後，評価関数や最適化手法の改良，および，制作プロセスの変革に関する検討をおこなっていく．

参考文献

[1] 杉山亮, なぞなぞ工房, https://sugiyama-akira.jp, Retrieved May. 31, 2022.

[2] 府川源一郎, 分岐展開型作文の可能性 –「お話し迷路」という場の設定–, 横浜国立大学国語教育研究, No.45, pp.35-46, 2020.

[3] Bhatt, S. N., Cosmadakis, S. S., The Complexity of Minimizing Wire Lengths in VLSI Layouts, Inf. Process. Lett., Vol.25, No.4, pp.263-267, 1987.

[4] Gregori, A., Unit-Length Embedding of Binary Trees on a Square Grid, Inf. Process. Lett., Vol.31, No.4, pp.167-173, 1989.

[5] Pullen, W., Think Labyrinth!, http://www.astrolog.org/labyrnth.htm, Retrieved May. 31, 2022.

[6] Pedersen, H., Singh, K., Organic labyrinths and mazes, In Proceedings of the 4th international symposium on Non-photorealistic animation and rendering (NPAR '06), pp.79-86, 2006.

[7] Xu, J., Kaplan, C., Image-guided maze construction, ACM Trans. Graph., Vol.26, No.3, 2007.

[8] Wan, L., Liu, X., Wong TT., Leung CS., Evolving mazes from images, IEEE Trans Vis Comput Graph, Vol.16, No.2, pp.287-297, 2010.

[9] Okamoto, Y., Uehara, R., How to make a picturesque maze, 21st Canadian Conference on Computational Geometry, 2009.

[10] Wong, F., Takahashi, S., Flow-Based Automatic Generation of Hybrid Picture Mazes, Computer Graphics Forum, Vol.28, No.7, pp.1975-1984, 2009.

[11] Adams, C., Louis, S., Procedural maze level generation with evolutionary cellular automata, 2017 IEEE Symposium Series on Computational Intelligence (SSCI), pp.1-8. 2017.

[12] Kim, P. H., Grove, J., Wurster, S., Crawfis, R., Design-centric maze generation, Proceedings of the 14th International Conference on the Foundations of Digital Games, 2019.

[13] Chou, W.-S., Rectangular Maze Construction by Combining Algorithms and Designed Graph Patterns, GSTF Journal on Computing (JoC), Vol.5, pp.1-5, 2016.

[14] Okano, K., Matsuyama, K., A Method for Generating Mazes with Length Constraint using Genetic Programming, NICOGRAPH International 2020, pp.39-42, 2020.

[15] 平石真也, 郭清蓮, 文法と迷路を融合したデジタルコンテンツに関する研究, モバイルコンピューティングとユビキタス通信研究会研究報告, Vol.2013, No.20, pp.1-6, 2013.

岡野 稀央隆

2020 年岩手大学理工学部卒業．2022 年岩手大学大学院総合科学研究科修士課程修了．現在は株式会社ピー・ソフトハウスに勤務．

山中 克久

岩手大学理工学部教授．博士（工学）．群馬大学大学院工学研究科博士後期課程修了．電気通信大学助教，岩手大学助教，同大准教授を経て，現在，岩手大学理工学部教授．アルゴリズム理論，グラフ理論などの研究に従事．

松山 克胤

岩手大学理工学部准教授．博士（工学）．岩手大学大学院工学研究科博士後期課程修了．公立はこだて未来大学助教，岩手大学理工学部助教を経て，現在，岩手大学理工学部准教授．インタラクティブシステム，ユーザインタフェースデザイン，情報可視化，コンピュータグラフィックスなどの研究に従事．

図 13: 作品例「あかずきんちゃん」.