修士論文

プレイヤを自然に誘導する 2D ゲームマップの自動生成

藤平 啓汰

主指導教員 池田 心

北陸先端科学技術大学院大学
先端科学技術研究科
（情報科学）

令和 4 年 3 月

Abstract

Artificial intelligence (AI) has been actively researched in various fields and benefited human society. For games, creating strong AI players has been mainly researched, and AI players have already achieved superhuman levels of plays in many games (e.g., AlphaGo for the game of Go). In addition, research on teaching, entertaining, and supporting human players is widely investigated, one of which is the automatic generation of game content.

Since video games have become more complex and diverse, development costs have increased significantly in recent years. In order to reduce development costs, a framework of automatic content generation, well known as Procedural Content Generation (PCG), has attracted attention from academia and industry. In the field of games, the term content covers various components such as maps and characters. Although PCG was researched for many game genres, research on role-playing games (RPG) is relatively few.

RPG is a classical genre in which players generally enjoy exploring maps, fighting with enemies, and growing characters. In many cases in RPGs, game designers provide guidance for players exploring maps to lead the players to appropriate directions, aiming to increase the fun of playing. However, too obvious guidance harms the excitement of exploring maps. Thus, it is better that RPG maps give players freedom in exploring while controlling their behaviors *naturally*. By natural, it means that players do not notice the control or guidance. Creating such desirable maps usually needs many game designers' efforts and time, making the development costs high.

This research aimed to clarify "what elements in maps influence the players' behaviors" and to generate two-dimensional maps that guide players naturally based on the clarified findings. Considering the complexity of RPG maps that contain many elements such as items and enemies, we targeted two kinds of simplified maps where elements were included step by step. The first target is mazes that contain only passages and walls but no items, enemies, or NPCs. The second target is dungeons that contain passages, walls, and rooms. For the dungeons, items and hit points (HP) were introduced to make the environment closer to RPG maps. We analyzed and predicted human players' behaviors when exploring maps and proposed a method to automatically generate maps that guide players naturally.

Our proposed method mainly consists of the following four phases: (1) Collecting play logs of human players, (2) Investigating human players' behavioral tendencies and predicting their behaviors by supervised learning, (3) Creating a test player considering human-likeness using the learned model, and (4) Generating maps by using playing results of the test player. This paper works on phases (1) to (4) for mazes and only (1) and (2) for dungeons.

For mazes, specifically, we aimed to automatically generate mazes with adjusted difficulty considering human players' path selection tendencies at branch points. In the first phase, we conducted subject experiments to collect play logs. A total of 20 players participated in the experiments. We used the play logs as training data for supervised learning and built a regression model to predict the human players' selection probabilities of proceeding directions at branch points. The model was based on the LightGBM, a decision tree-based gradient boosting model. The prediction results showed that the Pearson correlation coefficient between the predicted probabilities and actual values was 0.74, indicating a highly positive correlation. Although the prediction accuracy still had room to improve, we concluded that the prediction model was reliable to some extent.

Using the prediction model, we created a test player that simulates human players' behaviors. Furthermore, we automatically generated mazes with difficulty based on the number of steps that the test player played the mazes. The difficulty was classified into five groups such as "easy", "moderate", or "difficult". We conducted new subject experiments to evaluate whether the difficulty is suitable for human players. A total of 10 players, who were different from those in the previous experiments, played 35 prepared mazes. The experimental results showed that the estimated difficulty by the test player matched the play results of human players in general and that there were statistically significant differences between each difficulty group. From the results, we concluded that mazes generated with adjusted difficulty by our method were generally suitable for human players' playing in terms of the number of steps.

As the next step, we changed the target maps to dungeons. We assumed that only focusing on behaviors at branch points is not enough to properly guide players in dungeons because the maps contain not only passages and walls but also rooms and items. Thus, in addition to a prediction model for branch points, we built another two types of prediction models: one predicts the human players' selection probabilities of proceeding directions at rooms, and the other predicts probabilities that players turn back (turn back model).

We conducted new subject experiments to collect play logs for dungeons. A total of 18 players participated in the experiments. We used the play logs as training data for supervised learning as we have done for mazes. The correlation coefficients between the predicted probabilities and actual values for the branch point model and the room model were 0.62 and 0.67, respectively. Although each value showed a positive correlation, the prediction accuracy was lower than the case of mazes. The results indicated that human players' behaviors would change by just adding a few elements such as items and HP, making it difficult to predict.

We also checked the accuracy of the turn back model. To know whether the

model's predictions are similar to human players' behaviors, two important points are "places to turn back" and "the frequency to turn back." We first checked "the frequency to turn back." We obtained human players' frequency from the play logs and calculated the prediction model's frequency. The results showed that both of them turned back once out of approximately 20 actions. Furthermore, we observed that human players generally turned back at places with higher predicted probabilities and did not turn back at places with lower ones. From the results, we concluded that the outputs of the turn back model were similar to human players' behaviors in general.

Finally, we summarize this research. The results of mazes showed that our PCG method was effective in generating two-dimensional game maps that guided players naturally. Furthermore, from the results of dungeons, we verified that the method had room to improve in each phase. For example, the method would be enhanced using other types of techniques such as deep learning to build the prediction models.

概要

人工知能（AI）の技術は，画像処理や自然言語処理など幅広い分野で盛んに研究され，人間社会に利益をもたらしてきた．ゲームもまたAI技術の研究が盛んな分野であり，その成果として囲碁のAlphaGoなど，人間のトッププロを上回るほどの実力を持つコンピュータプレイヤが登場している．近年では，ゲームにおける強さ以外，例えば「教える」「楽しませる」「助ける」などに着目した研究も盛んに行われており，コンテンツの自動生成技術の研究もそのうちの1つである．

ビデオゲームの複雑化と多様化に伴う開発コストの増加が問題となっているゲーム分野では，学術・産業界を問わず，コンテンツの自動生成技術が注目を集めている．ゲームにおいてコンテンツとは，マップやキャラクタなど，ゲームを構成するあらゆる要素のことを指し，それらのコンテンツを自動生成する技術，あるいはその枠組みのことをProcedural Content Generation（PCG）と呼ぶ．PCGの研究ではさまざまなゲームジャンルが対象となるが，古典的ジャンルの1つであるロールプレイングゲーム（RPG）に焦点を当てたPCGの研究は比較的少ない．

RPGでは，ゲームの面白さの向上を目的に，マップを探索しているプレイヤに対して，適切な進行方向へ向かわせるための誘導を施すことが多い．しかし，それらの誘導があまりに露骨なものであれば，プレイヤが感じるはずのマップ探索の高揚感を喪失させ，ゲームの面白さを減じてしまうことに繋がる．そのため，プレイヤが「自分は能動的にマップを探索している」と感じつつも，実はゲームデザイナの意図に沿った行動をとる，つまりプレイヤが自然に誘導されるマップがしばしば求められる．プレイヤを自然に誘導するという複雑な特性を持つマップは，一般的に複数のゲームデザイナが多くの時間をかけて制作することが多く，ゲーム開発において高コストな作業とされている．

本研究では，マップの要素がプレイヤの行動選択に与える影響を解明し，その知見に基づき，プレイヤを自然に誘導する2Dゲームマップの自動生成を目指す．アイテムや敵など多くの要素を含むRPGマップの複雑さを考慮し，本研究では段階的にマップの要素を増やすアプローチをとる．研究対象のマップは2種類で，1つ目は壁と通路のみで構成される単純な迷路，2つ目は壁・通路・部屋から構成され，アイテムやHPの要素を含むなど，1つ目よりRPGに近い環境を持つダンジョンである．この2種類を対象としたうえで，マップにおける人間プレイヤの行動選択について分析・予測し，自然に誘導するマップの自動生成手法を提案する．

提案手法は主に，(1) 人間プレイヤのプレイログ収集，(2) 教師あり学習による人間プレイヤの行動選択予測モデルの構築と行動傾向の調査，(3) 予測モデルに基づく人間らしさを考慮したテストプレイヤの作成，(4) テストプレイヤの評価に基づく自然に誘導するマップの自動生成の4段階で構成される．本論文では，1つ目の迷路で（1）から（4）までを，2つ目のダンジョンで（1）と（2）のみを扱う．

1つ目の対象では，具体的には，人間プレイヤの分岐選択傾向を考慮して難易度を調整した迷路の自動生成を行った．自動生成に向けてまず，プレイログを収集するため，20人を対象に被験者実験を行った．実験から得た分岐点に関する学習

データをもとに，決定木ベースの勾配ブースティングモデル LightGBM を用いて，人間プレイヤの分岐選択確率を予測する回帰モデルを構築した．予測精度を検証した結果，実測値と予測値の相関係数は 0.74 とやや強い正の相関となり，人間プレイヤの迷路における分岐選択確率をある程度の精度で予測できることを示した．

この分岐選択予測モデルを用いることで，人間プレイヤの分岐選択傾向を模倣する，つまり人間らしい分岐選択を核とするテストプレイヤを作成した．さらに，このテストプレイヤが迷路をプレイしたときの歩数に基づく難易度推定によって，「簡単」「普通」「難しい」など，計 5 種類の難易度の迷路を自動生成した．テストプレイヤの推定した難易度と，人間プレイヤが実際に迷路をプレイしたときの結果が適合するのか検証するため，10 人を対象に新たな被験者実験を実施した．人間プレイヤによるプレイ結果は，テストプレイヤの推定に大まかに適合し，各難易度の結果には有意な差が見られた．これにより，プレイヤの歩数を難易度の指標に用いる場合，テストプレイヤの推定により難易度を調整した迷路は，実際に人間プレイヤが解いてもおおよそ推定通りの難易度であることが示され，迷路を対象とする研究は期待通りの結果となった．

迷路の次の段階として，アイテムや HP の要素を含みより複雑なマップであるダンジョンへ対象を移行した．迷路とは異なり，部屋やアイテムが存在するダンジョンでは，分岐点に着目して行動を予測するだけでは，人間プレイヤを適切に誘導することは困難であると考える．そのため，分岐点を対象とする予測モデルのほかに，部屋を対象とする分岐選択確率を予測するモデル，探索中にプレイヤが通路を引き返す確率を予測するモデル，これら 3 つを構築することにした．

ダンジョンのプレイログを収集するため，18 人を対象に新たな被験者実験を実施し，迷路と同様に実験から得た学習データをもとに LightGBM を用いて各予測モデルを構築した．分岐選択に関する 2 つのモデルの精度を検証した結果，分岐点を対象とするモデルでは相関係数が 0.62，部屋を対象とするモデルでは 0.67 となり，実測値と予測値に正の相関が見られたものの，迷路向けのモデルと比較した場合には劣る精度となった．この結果は，単純な迷路と比較すれば，アイテムや HP を要素に含むダンジョンでは人間プレイヤの行動がより複雑になり，その予測が困難になることを示唆している．また，引き返しに関するモデルについては，予測した結果の「引き返す場所」や「引き返す回数の多寡」が人間プレイヤに類似しているかが精度検証において重要になる．それらのうち，本論文では「引き返す回数の多寡」について検証を行った．実験データから算出した人間プレイヤによる多寡と予測結果を比較したところ，両者とも約 20 数回の行動のうち 1 回程度引き返していた．さらに，人間プレイヤが実際に引き返した場所は概ね予測確率も高く，逆に引き返してない場所は予測確率も低くなっていることが判明した．これにより，予測結果は人間プレイヤに類似していることが示された．

本研究の結果をまとめると，まず迷路の結果より，プレイヤを自然に誘導する 2D マップの自動生成に提案手法が有効であることが示された．さらに，ダンジョンの結果より，提案手法は部分ごとに改良の余地があることを検証でき，例えば，

予測モデルの構築に深層学習のような強力な手法を用いれば，提案手法がさらに効果的なものになると期待できる．

目 次

図 目 次

表 目 次

第1章　はじめに

人工知能（AI）の技術は，人間の知的作業を計算機に代替えさせることを目的に幅広い分野で研究され，目覚ましい進歩を遂げてきた．AI 技術の適用対象は自然言語処理 [1] や画像処理 [2] などさまざまだが，注目されている分野の 1 つにゲームがある．AI 研究の適用対象としてのゲームには，明確なルールが設定されていることや勝敗によって AI の性能を評価しやすいといった利点がある．

ゲーム分野における AI 研究は，主にチェスなどのボードゲームを対象に発展してきた．その成果として，チェスの Deep Blue[3] や将棋の bonanza[4]，囲碁の AlphaGo[5] のように人間のトッププロと同等以上の強さを持つほどのコンピュータプレイヤが登場している．このような成果はボードゲームだけにとどまらず，DeepMind が発表した Deep Q-Network という技術を利用したコンピュータプレイヤは，複数のビデオゲームにて人間以上のスコアを記録することに成功した [6]．このように，強さに着目した AI 研究は人間が十分に満足する水準を達成しつつある．現在では，強い対戦相手以外の用途，例えば「教える」「楽しませる」「助ける」などに着目した AI 研究が注目されている．そのうちの 1 つがゲームコンテンツの自動生成に関する研究である．

近年，ビデオゲームの複雑化と多様化に伴い，その開発コストの増加が問題とされている．この問題を解決するために注目されているのが，Procedural Content Generation（PCG）と呼ばれるコンテンツの自動生成技術である．ゲームにおけるコンテンツとは，マップや地形，アイテム，キャラクタ，文章，音楽，テクスチャ，ストーリー，ルールなど，ゲームを構成するあらゆる要素のことを指す．PCG の適用範囲は広く，横スクロールアクション [7] や First-person shooter（FPS）[8]，レーシングゲーム [9]，パズルゲーム [10]，ストラテジーゲーム [11] などのゲームに適用されてきた．一方，日本でよく親しまれている『ドラゴンクエスト』シリーズ [12] や『ファイナルファンタジー』シリーズ [13] に代表されるロールプレイングゲーム（RPG）を対象とする PCG の研究は比較的少ない．

多くの RPG においてプレイヤは，最後の大ボスを倒すといった大目的を果たし物語を完結させるため，キャラクタを操作しながらゲームマップの探索や敵との戦闘を行う．プレイヤは，大目的を達成する過程の中で「次の街へ行く」「アイテムを探す」といった多様な目的を持つ．それらの目的を達成しようとするとき，例えば，次の街へ行こうとしているとき，行き先がわからずに迷ったまま長時間経過してしまうと，プレイヤはゲームを進行する意欲を失い，ゲームプレイ自体を辞めてしまうことがある．ゲームデザイナは，プレイヤがマップ内で迷いすぎな

いように助けるため，マップ構造や敵・アイテムの配置方法，ノンプレイヤキャラクタ（NPC）による案内などを用いてプレイヤを適切な方向へ誘導することが多い．また，誘導の目的は，プレイヤを助けることだけに限らない．例えば，希少性の高いアイテムが容易に手に入ってしまうと，苦労して取得したときと比較し，得られる喜びや達成感は小さいはずである．そのため，プレイヤをアイテムからあえて遠ざけるための誘導などもあり得る．しかし，このような適切な方向への誘導ないしあえて遠ざけるような誘導は，ゲームデザイナの意図だけを考慮して仕掛けを施すと，ゲームの面白さを減じてしまうことがある．例えば，目的地や行き方の詳細を NPC が逐一提示して誘導する方法や，プレイヤの行動を不自然に制限する方法は，RPG の楽しさの 1 つであるマップ探索の高揚感を喪失させたり，プレイヤに不服を感じさせることに繋がったりする．

RPG では，プレイヤが「自分は能動的にマップを探索している」と感じつつも，実はゲームデザイナの意図に沿った行動をとる，つまりプレイヤが自然に誘導されるマップがしばしば求められる．プレイヤを自然に誘導するためには，プレイヤの行動傾向を理解・予測したうえで，マップをデザインする必要がある．通常，そのような複雑な特性を持つマップの制作は，複数のゲームデザイナが多くの時間をかけて取り組むため，ゲーム開発における高コストな作業とされている．

本研究の主たる目的は，プレイヤの行動選択がマップの要素にどのような影響を受けるのかを解明し，そこで得た知見をもとにプレイヤを自然に誘導する 2D ゲームマップを自動生成することである．マップ探索中のプレイヤの行動には，マップ構造やアイテム，敵，NPC など，マップ中のさまざまな要素が影響を与える．これにより，プレイヤ行動の分析・予測は複雑かつ困難になると予想される．本研究では，比較的要素の少ないゲームからはじめ，段階的に要素を増やし，各要素がプレイヤの行動に与える影響について 1 つずつ解明していくアプローチをとる．具体的には 2 つのゲームを対象とする．1 つ目は，通路と壁のみで構成され，敵やアイテムを含まない単純な迷路ゲームである．2 つ目は，通路と壁，そして部屋から構成され，なおかつアイテムを含むなど，1 つ目の迷路ゲームより RPG に近い環境のダンジョン探索ゲームである．

プレイヤを自然に誘導するマップは，以下の手順により自動生成する．まず，分岐選択などに着目しながら，人間プレイヤの行動選択予測モデルを教師あり学習によって構築する．次に，予測モデルに基づく人間らしさを考慮したテストプレイヤを作成し，さらにそのテストプレイヤを用いてマップの自動生成を行う．

本論文は以下の形で構成される．2 章では，迷路と RPG の一般的な概要を述べる．3 章では，PCG や人間らしさを考慮したテストプレイヤについての関連研究を紹介する．4 章では，プレイヤを自然に誘導するマップの自動生成に向けた具体的なアプローチを与える．5 章では，迷路における人間プレイヤの分岐選択確率予測モデルを構築し，続く 6 章にて予測モデルに基づくテストプレイヤを用いた迷路の自動生成を行う．7 章では，ダンジョンにおける人間プレイヤの行動予測モデルを構築する．最後に，8 章で本研究の成果や今後の展望をまとめる．

第2章　迷路とRPGについて

本章では，本研究が焦点を当てる「迷路」「RPG」の一般的な概要を述べる．

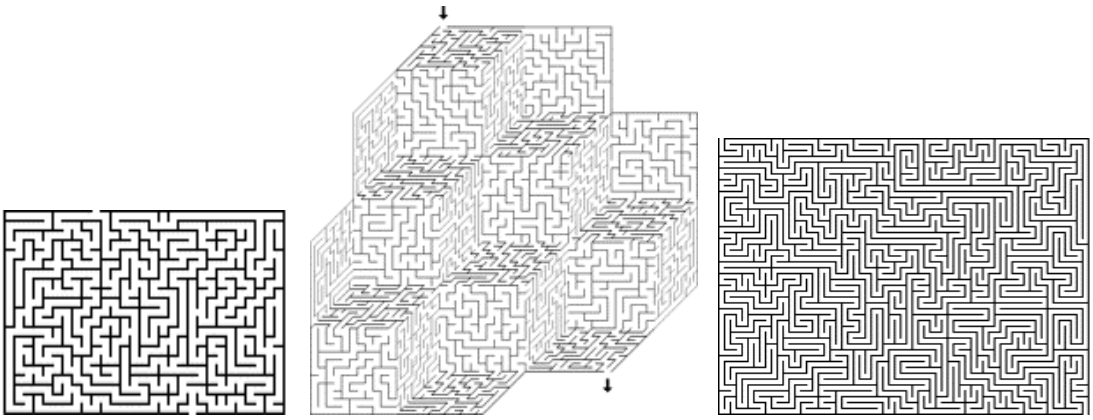
2.1　迷路

迷路とは，定められたルールに従い，スタートからゴールまでの道のりを探索する古典的パズルゲームである．迷路は，これまでに多くの種類が公開され，世界中で楽しまれてきた．図 2.1 に示すように，迷路の種類は次元数や正解路の一意性，トポロジ，分岐点の有無，外観など多様な観点から分類することができる [14]．迷路の中には娯楽的な側面だけではなく，絵画的迷路 [15] のように芸術的，あるいは数学迷路 [16] のように教育的な側面を持つ迷路も存在するが，いずれの迷路も「スタートからゴールへ辿り着くこと」がプレイヤの主目的となることが多い．また，その楽しみ方はプレイヤごとに異なり，素早くゴールへ向かうことや，不正解路へ進む回数を最小限に抑えることなどさまざまである．その中でも，ゴール到達時に得られる達成感は，多くのプレイヤに共通して見られる．

達成感を得る，つまり迷路の面白さを生み出すために必要な要素の 1 つとして，「難易度」が挙げられる．例えば，迷う余地がなく，ただ進むだけでゴールへ辿り着いてしまう簡単な迷路であれば，プレイヤが得る達成感はそれほど大きくならないはずである．一方，何度も間違えを重ね，ゴールへ辿り着けない状況が続いた場合には，最終的に得られる達成感よりも苛立ちのほうが大きくなり，面白さを減じてしまう可能性がある．

これらを踏まえると，マップデザインの観点では，ターゲットとなるプレイヤにとっての適切な難易度を見極めることと，その難易度になるように迷路を調整することが重要であると考える．

パズルゲームとしてそれ単体で遊ばれることも多い迷路だが，用途はそれだけに限らない．敵から逃げつつアイテムを回収することが目的のアクションゲーム『パックマン』シリーズ [17] では，ゲームマップとして迷路が採用されている．また，ゲーム分野以外にも，生理学分野の空間学習に関する研究 [18] や防災・減災分野の避難行動に関する研究 [19] において迷路は活用されている．このように，迷路には幅広い用途がある．



(a) 一般的な 2 次元迷路　(b) 特殊なトポロジの 2 次元迷路　(c) 分岐点のない 2 次元迷路

図 2.1: 異なる種類の迷路 [14]

2.2 RPG

RPG の元祖は『ダンジョンズ&ドラゴンズ（D&D）』[20] である．D&D においてプレイヤ達は，ルールに従い各々が作成したキャラクタを演じながら，与えられた物語を進行する．プレイヤ達の主な目的は，互いに協力し，キャラクタを成長させ，敵や罠が潜むダンジョンを踏破して財宝を手に入れることである．RPG に明確な定義があるわけではないが，このように与えられた舞台・物語上でキャラクタを演じ，成長しながら試練を乗り越え，目的の達成を目指すようなゲームが典型的な RPG とされる．

明確に言えば，D&D はテーブルトークロールプレイングゲーム（TRPG），ドラゴンクエストシリーズやファイナルファンタジーシリーズはコンピュータロールプレイングゲーム（CPRG）に区分される．日本では主に RPG と言えば CRPG のことを指し，本論文においても CRPG を RPG と呼ぶこととする．また，多種多様な作品が存在する RPG だが，ここではドラゴンクエストシリーズのようにシングルプレイであり，物語やキャラクタの成長，マップ，戦闘，ボスなどの要素を含む作品を想定する．

本節では，RPG の基本的な流れや用語について述べた後，RPG におけるマップ探索と戦闘，プレイヤの誘導についてそれぞれの概要を述べる．

2.2.1 基本的な流れ

RPG は，作品ごとにグラフィックやゲームシステムなど異なる部分があるものの，多くの作品において「プレイヤはマップ探索と戦闘を繰り返しながらキャラクタを成長させ，最後には大ボスを倒す」という共通の流れがある．

具体的には，物語を開始したプレイヤはまず「街の住人から悪い盗賊団の討伐を依頼される」といったように「何かを倒す」，あるいは「何かを探して取得する」「誰かに会う」などの目的を持つことになる．それらを達成するため，プレイヤはマップ内を探索して必要な情報・モノを収集したり，特定の場所へ向かったりする．その中で主な障害となるのが敵役の NPC である．プレイヤは，キャラクタの特徴やパラメータ，探索状況を踏まえながら戦略を考え，敵と交戦する．敵を倒すことによりプレイヤは経験値を獲得し，経験値を用いてキャラクタを成長させることができる．経験値のほかにも，マップ上で取得したアイテムなどを用いてキャラクタを成長・強化させることも可能である．マップ探索や戦闘の末に目的を達成すると，プレイヤは次の新しい目的を持ち，物語を進めることになる．例えば，先に挙げた「盗賊団の討伐」を達成したならば，次は「盗賊団を討伐したプレイヤの手腕を見込んだ王様から新たな依頼を承る」といった具合に，前の目的と次の目的への導入には物語上の関連性があることが多い．

このような一連の流れを幾度も繰り返しながら，最後に大ボスを倒すことでプレイヤは物語を完結させることができる．これが RPG の基本的な流れである．

2.2.2 基本的な用語

作品ごとに用いられる用語は異なるが，その中でも一般的だと思われる用語とその意味を以下にまとめる．

- **HP：**キャラクタの体力を示すパラメータであり，プレイヤが操作するキャラクタ，あるいはプレイヤの味方キャラクタ全員の HP が 0 になると，ゲームオーバーとなる．ゲームオーバーとなったプレイヤは，何らかのペナルティを受けた後，近くの街やゲームの進行状況を最後に保存した場所からゲームを再開することになる場合が多い．

- **MP：**キャラクタが持つ特殊な能力を使用するために必要なパラメータであり，MP を消費することで，例えば HP を回復したり，キャラクタを一時的に強化したりすることができる．MP が 0 になったキャラクタは，特殊な能力を使用できなくなる．

- **アイテム：**キャラクタが所持できる品物の総称．使用したり，身につけたりすることで，HP・MP の回復やキャラクタの強化などアイテムごとの効果を得ることができる．アイテムの取得方法として，購入する，マップ上で拾う，戦闘の報酬として貰うなどが挙げられる．アイテムの種類はさまざまで，取得の必要はないが探索や戦闘を有利に進められるアイテムのほかに，通常のアイテム以上の効果を得られるアイテム（以降，レアアイテム），ゲームの進行に必要不可欠なアイテム（以降，キーアイテム）などがある．

- **お金：**主にアイテムの購入などに用いられる．お金の取得方法として，マップ上で拾う，戦闘の報酬として貰うなどが挙げられる．

2.2.3　マップ探索

一口にRPGマップといってもその分類や機能は作品によって異なるため，明確に定義することは難しい．ここでは便宜上，マップを拠点・ダンジョン・フィールドの3つに分類し，それぞれの特徴を以下にまとめる．

- **拠点：**プレイヤの冒険の起点となる場所．主にHPやMPを回復したり，アイテムを購入したり，ゲームを進行するために必要な情報を収集できたりする．
- **ダンジョン：**敵やアイテムが配置されており，拠点以外でプレイヤの主な目的地となる場所．プレイヤはボスを倒したり，アイテムを探したりするため，あるいは次の拠点への通過点としてダンジョンを探索する．ゲームの進行上，探索する必要のないダンジョンも存在するが，レアアイテムを取得できるなど，探索することへの何らかのメリットが用意されている場合が多い．

  また，通常では通ることのできない通路，いわゆる隠し通路などの仕掛けが施されている場合もある．隠し通路を進むことによって，レアアイテムを取得できたり，新たに別のダンジョンを発見できたりする．
- **フィールド：**拠点やダンジョンを行き来するためにプレイヤが移動する必要のある場所．ダンジョンと同様に敵，場合よってはアイテムも配置されている．

この3つを大まかなマップ分類としたうえで，特にダンジョンに着目してマップ探索の概要を述べる．ここで想定するのは，ボスが配置されていないダンジョン，つまりアイテム探しの場所や次の拠点への通過点として機能するダンジョンである．物語の進行上，プレイヤは必ず探索する必要があるものとする．また，プレイヤは何度でもダンジョンに出入りすることが可能であるとする．

マップ探索においてプレイヤは，欲しいアイテムを発見したり，マップを踏破（出口に辿り着くなど）したりすることで，高揚感や達成感を得ることができる．また，それらに至るまでに，どのようなルートを辿るのか，HPやMPなどのリソースをどう管理するのか思案する部分が，マップ探索の面白さに繋がっていると考える．つまり，マップ探索の面白さを生み出すためには，プレイヤに適度な迷いが生じる状況やリソース管理が求められる状況が必要となる．

例えば，ダンジョン内でキーアイテムを探しているが，発見できていない状況を考える．探索が長引くとリソースが枯渇してしまうため，プレイヤは一旦ダンジョンを抜けて探索を仕切り直す必要がある．このような状況が特定のダンジョンに対して何度も繰り返される場合，プレイヤはゲームを進める気を失くし，ゲームプレイ自体をやめてしまう可能性がある．一方，欲しいアイテムを容易に発見で

きてしまうと，苦労して発見したときと比較し，得られる高揚感は小さくなるはずである．また，探索を進める中で，リソースの枯渇を心配する必要がなければ，「どこを優先して探索するべきか」「どこで引き返すべきか」など，思案する機会や探索の緊張感が失われてしまう可能性がある．ただし，常に厳密なリソース管理を要求するダンジョンは，物語の進行を阻害する障害にもなり得るため，ゲームデザインにもよるが多用することは望ましくないはずである．

これらを踏まえると，RPG のマップデザインでは，プレイヤがある程度迷うマップ構造にすることや，リソース管理に緊張感が生じるようにアイテム・敵を配置することが重要になると考える．なお，これらは一般的な傾向について考察した結果であり，全ての RPG 作品に共通するとは限らない．

2.2.4 戦闘

本研究ではマップ探索に焦点を当てているが，RPG においてマップ探索と戦闘は深い関連を持つ．本項では，RPG における戦闘の概要，マップ探索と戦闘の関連性について述べる．

プレイヤは主に，キャラクタの成長やアイテム・お金の取得，ゲームの進行のために戦闘を行う．作品によって，戦闘システムやプレイヤに求める技術・知識は異なるが，戦闘パートの基本的な流れは以下の通りである．

(1) 敵味方を問わず，戦闘に参加するキャラクタの特徴やリソースを考慮しながら戦略を練る．

(2) 戦略に基づいて攻防を繰り出す．

(3) 敵を倒す，またはプレイヤがゲームオーバーになるまで (1) と (2) を繰り返す．

(4) 敵を倒すことができた場合には何らかの報酬を得ることができる一方，ゲームオーバーになった場合には何らかのペナルティを受ける．ゲームオーバーになったプレイヤは，戦闘から得た知見による戦略の練り直しや，別の敵と戦うことによるキャラクタを成長を図る．

この一連の流れを通してプレイヤが面白いと感じる部分は作品によってさまざまであるが，戦略を練る部分が戦闘の面白さを構成する要素であることはどの RPG 作品にも共通すると考え，それはプレイヤが体験する成長と達成感が要因であるとも考える．プレイヤが自ら意思決定を行い（戦略を練る），試行する（攻防を繰り出す）．そのとき，戦略が上手くいけば達成感を得ることができ，失敗した場合には原因を分析し戦略を練り直す．再度試行したときに上手くいけば，自身の成長を実感することができる．これらの成長と達成感が，戦略を練ること，ひいては戦闘の面白さを生み出す要素の 1 つとなる．さらに，ある戦略を実現するには，

それに応じたキャラクタの成長や特徴づけ，戦闘用アイテムの収集などが必要となり，これらもまたRPGの面白さに繋がっている．

ただし，RPGにおいては，単一の戦闘の面白さのみを追求するだけでは不十分であることが多い．これはボス戦を含むいくつかの場合を除き，プレイヤがいくつもの戦闘を重ねて，その都度いくらかのリソースを消費しながらマップを探索するからである．例えば，プレイヤとの力量が拮抗している敵のみが出現するダンジョンを想定する．激闘の末プレイヤが勝利し，高い達成感を得られたとしても，そのような戦闘が何度も続くようであれば，プレイヤのリソースは枯渇し，マップを踏破できなくなる可能性が高まる．そのような状況は，ゲームの進行を阻害してしまうため望ましくないはずである．ほかにも，マップ探索中のプレイヤは，アイテムの取得などによってパラメータ・戦略を変化させることがあるため，同じ敵との戦闘であってもその面白さが変動してしまうことにも注意しなければならない．

このように，マップ探索と戦闘は相互に影響を与えるため，マップの規模や配置されるアイテムと敵の種類・数，プレイヤの移動経路などを考慮しながら，そのマップにおける戦闘部分をデザインする必要がある．

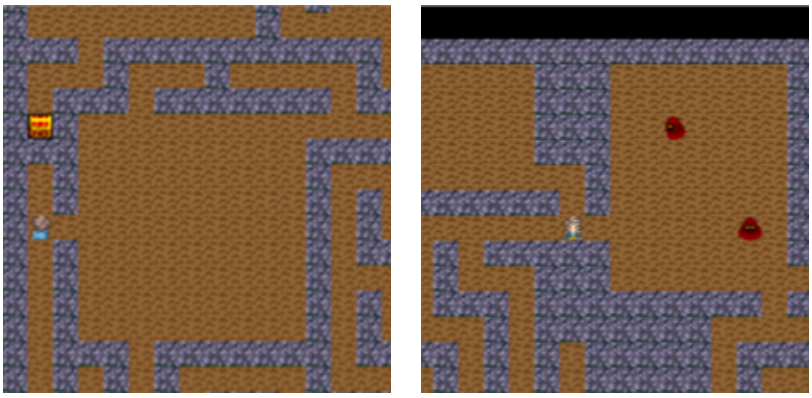
2.2.5　マップ探索におけるプレイヤ誘導

本研究では，プレイヤを自然に誘導する2Dゲームマップの自動生成を試みる．これに関連して本項では，マップ探索中のプレイヤを誘導する手法を紹介する．また，ここで想定するジャンルは，2DRPGとする．

プレイヤに対する誘導は,「プレイヤを助ける」「プレイヤの達成感を高める」など，ゲームデザイナの何らかの意図を反映するために行われ，さまざまな手法によって実現される．いくつかの具体例とその特徴を以下にまとめる．

- **配置：**アイテムや敵，NPCなどを配置することによって，プレイヤを引き付ける，あるいは遠ざける手法（図2.2）．例えば，アイテムならば「欲しいアイテムかもしれない，行ってみよう」とプレイヤに思わせ，その配置場所へ引き付けたり，配置場所の方向へ意識を向けさせたりすることが期待できる．敵の場合，その種類や強さによって，プレイヤの行動を変化させることができる．例えば，プレイヤにとって厄介な敵が多く出現するならば「あまり寄り道せずに進もう」「この敵を避けながら進もう」となったり，あるいは倒すことによって多くの経験値を獲得できる敵が出現するならば「もっと遭遇したいからマップを動き回ろう」となったりする．

- **追い返し：**ゲームによっては，物語の進行の観点から，ある時点では進むことが推奨されない領域が存在する場合がある．その領域へ進もうとするプレイヤを追い返す方法として，動かせないオブジェクトを配置したり，NPCが止めたりなど，明示的に進入を拒む手法がある．一方，敵を用いて非明示に



(a) アイテムの配置　　(b) 敵の配置

図 2.2: 配置による誘導

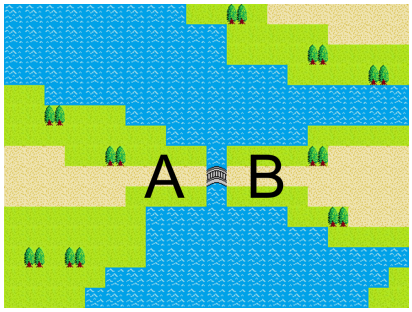


図 2.3: 追い返し

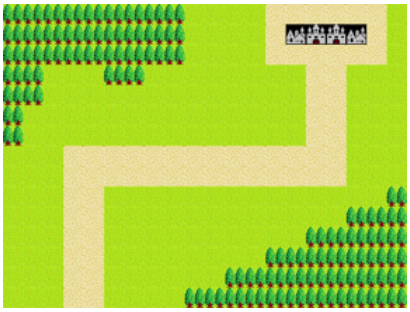


図 2.4: 道筋

追い返す手法もある．例えば，図 2.3 において，領域 A をプレイヤがいる領域，領域 B を進むことが推奨されない領域とする．このとき，領域 B には「場合によっては勝利できるが，プレイヤの勝てる見込みが薄い敵」を配置する．これにより，プレイヤが領域 B へ進入したとしても，領域 A へ引き返すよう非明示的に誘導することができる．また，キャラクタを成長させることによって将来的には進入することができるため，領域 B はプレイヤに成長を実感させる要素にもなり得る．

- **道筋：**地面の色や模様を変更することで道筋を作成し，目的地までの道のりをプレイヤに示す手法（図 2.4）．この手法は主に，プレイヤを適切な方向へ導くことに用いられることが多いが，プレイヤの意識を特定の場所から逸らすことにも利用できる．

- **強調：**マップ中の特定の場所を強調させ，その場所へプレイヤを引き付ける手法（図 2.5）．強調された場所には，隠し通路への入口やアイテムなどが隠されていることがある．これらの発見は，プレイヤの高揚感や達成感の向上に繋がることが多い．

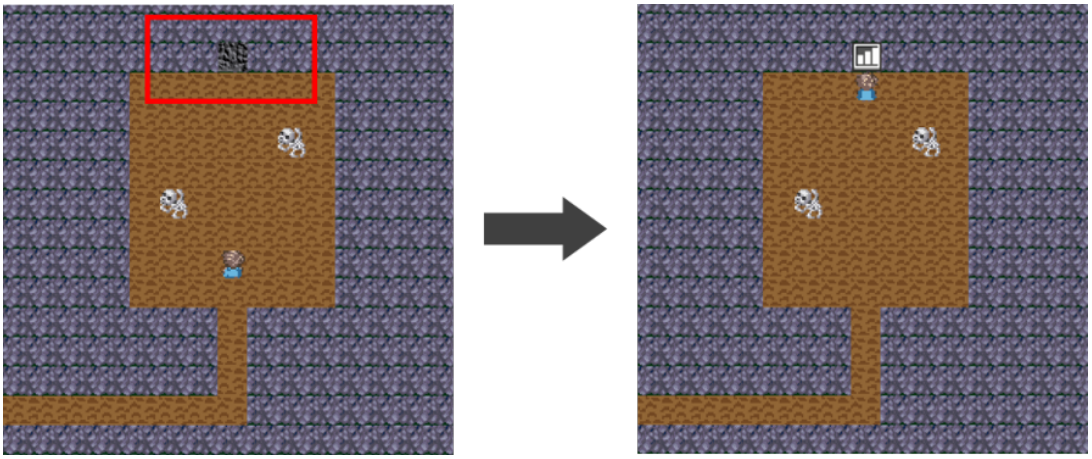


図 2.5: 強調による誘導．赤枠の中央に位置する壁が強調されている．この場合，強調された壁には通路が隠されている．

- **音：**背景音楽の変化を用いて誘導する手法．例えば，目的地への道のりから大きくはずれると音量が小さくなる，ダンジョン内ではボスに近づくにつれてテンポが速くなるなど，プレイヤの行動に応じて音を変化させ，その変化によってプレイヤにヒントを与えたり，適切な場所へ誘導したりする．
- **NPC との会話：**NPC との会話によって誘導する手法．この手法について説明するため，以下の状況を考える．
  - プレイヤは現在，街 A にいる．
  - プレイヤは，街 B に行くことが次の目的である．
  - 街 B に行くには，雨を降らせるアイテム（雨アイテム）を取得しなければならないが，プレイヤはそのことをまだ知らない．
  - 雨アイテムは，街 C で取得できる．

  目的を達成させるうえで，単純な誘導手法として「街 C にある雨アイテムを手に入れれば，街 B に行くことができる」などとプレイヤに伝えることが挙げられる．一方，ゲームの面白さを高めるため，誘導に謎解きの要素を取り入れた以下のような手法もある．まず，ある NPC が「街 B に行くには雨アイテムが必要になるそうだ」とプレイヤに伝える．次に，別の NPC が「街 C ではよく雨が降るそうだ」とプレイヤに伝える．

  この例では，雨というキーワードを複数の NPC の会話の中で登場させることにより，雨アイテムと街 C の関係をプレイヤへ示唆している．このように，いくつかのヒントを与え，プレイヤ自身に気づかせることにより，能動的にマップを探索している感覚を与えることができる．

ここで紹介したものは，誘導の手法とその使い方の一例である．ジャンルやシステム，ゲームデザインによって誘導の意味や採用するべき手法は異なる．また，2DRPG やその他のジャンルにおける誘導ないしゲームデザインについてまとめた文献もあるので，そちらも参考にされたい [21][22][23].

第3章　関連研究

本研究では，人間らしさを考慮したテストプレイヤを用いて，人間プレイヤを自然に誘導する迷路・ダンジョンの自動生成を試みる．これに関連して本章では，コンテンツの自動生成技術である PCG，迷路・ダンジョンの自動生成，人間らしさを考慮したテストプレイヤ，さらに人間の誘導に関する研究を紹介する．

3.1　Procedural Content Generation

本節では，はじめに PCG の概観を与える．次に，PCG の中でも注目されることの多い Search-based PCG と PCG via Machine Learning，PCG via Reinforcement Learning について述べる．

3.1.1　PCG とは

PCG とは，手続き的にコンテンツを自動生成する技術，あるいはその枠組みのことを指す．コンテンツとは，ゲームを例とすればマップや地形，キャラクタ，音楽，テクスチャ，ストーリー，ルールなど，ゲームを構成するあらゆる要素のことを指す．また，手続き的な生成というのは，何らかのアルゴリズムに基づいてコンテンツを生成することを意味する．PCG は，映像分野や芸術分野など幅広い分野で活用されているが [24]，ここではゲームを対象とする PCG について述べる．

コンテンツに求められる条件や品質は多岐にわたり，それらと対応するように PCG の手法もさまざまである．本項では，Shaker らの著書 [25] を参考に，PCG の基本的な分類とその特徴について述べる．

- **Online versus Offline**：コンテンツを生成するタイミングは，プレイヤがゲームをプレイしている途中（online），あるいはゲームの開発途中やプレイヤがゲームを始める前（offline）のいずれかである．Online でコンテンツを生成するゲームの例として，FPS ゲーム『Left 4 Dead』[26] が挙げられる．Left 4 Dead では，ゲーム内でリアルタイムにプレイヤの行動やステータスを分析し，その分析に基づき敵の配置や数を選定している [27]．Online の場合，プレイヤの特徴や好みを反映したコンテンツを生成することが可能だが，リアルタイムでの処理となるため，高速な生成が求められる場合が多い．

- **Necessary versus Optional**：生成されたコンテンツは，ゲームを進行するうえで必要不可欠なコンテンツ（necessary）であったり，ゲームの進行という点では必要のないコンテンツ（optional）であったりする．ストーリーを生成する場合，ゲームの進行に必要なメインストーリーは necessary に，ゲームの進行には影響のないサブストーリーは optional にそれぞれ該当する．

- **Degree and dimensions of control**：PCG では，何らかのアルゴリズムを用いてコンテンツを生成するが，生成するコンテンツを制御する方法はいくつかある．その 1 つにコンテンツの特徴をパラメータ化し，そのパラメータによって制御する方法がある．例えば，Shaker らは『スーパーマリオブラザーズ』[28] のレベルを自動生成する際，地面と地面の距離などをパラメータとして扱い，その下限と上限を定めることで，生成されるレベルを制御している [29]．一方，特徴のパラメータ化のほかに，ランダムシードを用いて制御する方法もある．『マインクラフト』[30] では，1 つのランダムシードを用いて，地形や気候，構造物などを含むマップが生成される．この方法では，マップの一部の特徴を指定するといったことができないが，同じランダムシードを用いることで，常に同じマップを生成することができる．

- **Generic versus Adaptive**：Generic PCG では，特定のプレイヤではなく，大勢のプレイヤに支持されるようなコンテンツを生成する．大勢のプレイヤが面白い，あるいは楽しいと感じるコンテンツの生成は商業ゲーム開発においても重要な課題とされており，大勢に支持されることを求めた結果，どのプレイヤにも満足されないコンテンツになってしまうことはあり得る．そういった事態の回避を期待できるのが Adaptive PCG である．Adaptive PCG では，特定のプレイヤの行動や好みに基づきコンテンツを生成する．ただし，対象ゲームやプレイする環境によっては，行動や好みの分析自体が困難な場合もある．一般的に，商業用ゲームでは Generic PCG を用いることが多いが，近年では Adaptive PCG も注目されている [31]．Left 4 Dead は，商業用ゲームにおいて Adaptive PCG を採用した例である．

- **Constructive versus Generate-and-test**：Constructive PCG では，コンテンツが一度だけ生成され，その品質評価は行われない．フラクタル図形を用いる地形の自動生成 [32] などは，Constructive PCG に該当する．Constructive PCG において注意するべき点は，ゲームの進行を妨げない程度の品質が生成アルゴリズムによって保証されているか否かである．例えば，スタートからゴールへ辿り着くことが目的の迷路を生成する場合，スタートとゴール，さらにその 2 点間を結ぶ通路が迷路に含まれている必要がある．一方，Generate-and-test PCG では，一定の品質基準を設定し，その基準を超える品質のコンテンツが生成されるまで，生成と評価を交互に繰り返し行う．

- **Automatic generation versus Mixed authorship**：PCG の手法は，コ

ンテンツの生成に人間がどの程度介入するかによっても分類できる．例えば，生成アルゴリズムによってのみコンテンツを生成することもあれば，人間が制作したコンテンツに対してアルゴリズムが補充したり，新たな部分を追加したりすること [33]，あるいはアルゴリズムにより生成したコンテンツに人間が手を加えることもある [34]．

このように，PCG はさまざまな観点で分類できるが，生成アルゴリズムや PCG の研究がそれぞれどのグループに属するのかを厳密に分類することは難しい．PCG では，対象となるゲームやコンテンツの種類，生成するタイミング，求められる生成速度，ターゲットとなるプレイヤ，プレイヤから取得できる情報など，多様な観点を考慮しながら適用する手法を選定し，組合わせる必要がある．

3.1.2 Search-based PCG

本項では，PCG の代表的な手法の 1 つである Search-based PCG について述べる．Search-based PCG は，3.1.1 項で述べた Generate-and-test PCG の一種である [35]．通常の Generate-and-test PCG では，コンテンツをランダムに生成した後，評価関数や AI プレイヤによる 1 回または複数回のテストプレイなどによって品質評価を行い，一定の品質基準を超えるコンテンツが生成されるまで生成と評価を繰り返す．そのため，生成対象のコンテンツが複雑であったり，品質基準が高かったりすると，求める品質を持つコンテンツが生成されるまでに多くの時間を要してしまうことがある．これに対して，進化的アルゴリズムや群知能に基づく最適化アルゴリズムなどのメタヒューリスティックスを用いることで，コンテンツを効率よく探索しようとするのが Search-based PCG である．

Search-based PCG では，遺伝的アルゴリズム（Genetic Algorithm，GA）を用いることが多い [36][37][38]．GA を用いた場合を例に，Search-based PCG におけるコンテンツ生成の大まかな流れを以下に示す．

(1) 解候補の個体（コンテンツ）を遺伝子として表現する．個体の表現方法として，ビット文字列や実数値の配列などが挙げられる．

(2) 複数の個体から成る集団を作成する．

(3) 事前に定義した評価関数により，集団内の各個体を評価する．

(4) 評価値が低い個体は淘汰する．評価値が高いいくつかの個体については，交叉（遺伝子の交換）によって新しい個体を作成したり，突然変異を生じさせたりする．

(5) (4) によって新たな集団が形成されるので，再び (3) へ．

この一連の流れを繰り返すことにより，求める品質を持つコンテンツの効率の良い生成が期待できる．

評価関数を適切に設計することができれば，ゲームデザイナの意図を十分に反映したり，ターゲットとなるプレイヤの特徴や好みに合わせたコンテンツを生成したりすることができる．一方，その評価関数の設計の難しさや個体の表現方法が Search-based PCG の課題にもなっている．個体の表現方法は探索空間の大きさに影響を与え，一般的に探索空間が小さいほどコンテンツの多様性は減少し，品質の上限は低下してしまう．一方，探索空間が大きくとも，その分探索にかかる時間が増加してしまう傾向にある．また，評価関数の時間計算量はコンテンツの生成速度に大きな影響を与える．そのため，対象となるゲームやコンテンツごとに，適切な表現方法や評価関数を考案・設計する必要があり，これらの課題に焦点を当てた研究も注目されている [39].

3.1.3 PCG via Machine Learning

本項では，Search-based PCG と同様に，PCG の代表的な手法の 1 つである PCG via Machine Learning（PCGML）について述べる．PCGML とは，既存のコンテンツを学習データとする生成モデルを機械学習により構築し，そのモデルを用いてコンテンツを生成する PCG のことである [40]．Search-based PCG などの場合，機械学習はコンテンツの評価や人間プレイヤの行動の模倣など，コンテンツを生成する過程の一部で用いられる．一方，PCGML は，機械学習によって構築した生成モデルがコンテンツそのものを出力するという特徴を持つ．以下では，PCGML の研究をいくつか紹介する．

Summerville らは，トレーディングカードゲーム（TCG）の『マジック：ザ・ギャザリング（MTG）』[41] を対象に，Long Short-Term Memory（LSTM）を用いて，カードデザインの半自動生成を試みている．それまでにも MTG のカードを自動生成した事例はあったものの [42]，カードの名前や能力などの情報について，その最初の 1 文字目のみを入力できる仕様であり，一部の情報のみ，例えば能力の内容だけを全て指定したうえでの生成などはできなかった．PCG では，コンテンツのある部分だけは事前に指定できるような仕様が望まれることも少なくない．これに対して，Summerville らの生成モデルは，一部のカード情報を入力，欠損した情報が補完された状態のカードを出力としている点が特徴である．学習の結果，カードとして適切に機能するものを生成することに成功しているものの，学習データが十分に確保できない種類のカード，あるいは別のカードゲームへの適用が難点であると報告している．

Summerville らは，LSTM を用いてスーパーマリオブラザーズのレベルの自動生成も行っている [43]．LSTM を用いるうえで，Summerville らはレベル中のタイル（ブロックや土管のパーツ，コインなど）を文字，レベルを文字列として扱っている．PCGML では，生成したコンテンツの外観に不自然な点があることもしばし

ばだが，Summerville らは文字の前後関係を学習することによって，生成したレベル中の隣り合うタイルの位置関係が自然になる確率を高めている．もう 1 つの特徴的な点として，A*アルゴリズムによって作成された AI プレイヤの経路情報を学習データに取り入れた点が挙げられる．一般的に，PCGML で生成されたレベルなどは，それがプレイ可能であるとは限らない．Summerville らは，学習データとして既存のレベルをそのまま用いるのではなく，そのレベルを AI プレイヤがクリアしたときの経路情報を取り入れたものを学習データとすることより，スタートからゴールまでの移動可能な経路が含まれる，つまりプレイ可能なレベルをより多く生成することに成功している．

ひとたび生成モデルを学習し終えると，高速でコンテンツを生成できるため，PCGML は online の生成に適しているとされる．しかし，十分な量の学習データを確保することは一般的に困難な場合が多い．また，学習データを十分に集められたとしても，そのデータに偏りがあれば生成されるコンテンツの種類に偏りが生じる．多様なコンテンツを生成可能になったとしても，実際に生成したコンテンツが求める特徴を持つのか，そもそもプレイ可能なのか判断する必要があるといった別の課題もあり，それらの解決は容易でない場合も多い．

3.1.4 PCGML と Search

PCGML では，生成モデルの学習後，実際にコンテンツを生成してみると，望ましい特徴を持つものがいつまでも生成されないといった場合がある．本項では，これらを解決するための工夫として，Search-based PCG で用いられるような探索手法を取り入れた PCGML の研究をいくつか紹介する．

Volz らは，スーパーマリオブラザーズのレベルの断片の自動生成を試みており，生成モデルとして Generative Adversarial Network（GAN），望ましいコンテンツの探索に Covariance Matrix Adaptation Evolution Strategy（CMA-ES）を用いている [7]．GAN は，敵対する 2 つのネットワーク「Generator」「Discriminator」を競わせるような形で学習を行う生成モデルであり，潜在変数というデータの特徴を示す値を Generator に入力することで新しいデータを生成する．Volz らは少ないデータ数での学習を試みており，既存の 1 つのレベルを 173 個の断片に分割，それぞれを GAN の学習データとして生成モデルを構築している．

GAN では，学習後に特定の特徴を持つデータを生成するとなった場合，そのデータを生成する潜在変数の値を見つける必要がある．これに対して，Volz らは CMA-ES を適用して，特定の難易度となる断片を生成する潜在変数の探索を行っている．探索時の評価値となるのは，A*アルゴリズムを用いた AI プレイヤが断片をプレイしたときの「断片の横幅に対して進んだ距離の割合」「ジャンプの回数」である．これによって，ジャンプの回数が多くなる，あるいは少なくなる断片を効率よく生成することに成功している．ただし，生成された断片には，図 3.1 のよ

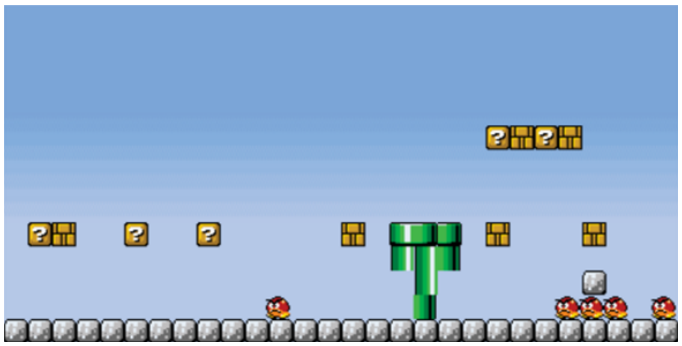


図 3.1: エラーがある断片の例 [7]

うに土管の一部が欠けていたり，土管のパーツが不適切に組み合わさっていたりなど，不自然な点（エラー）が存在している場合もある．

Volz らの研究に対して Shu らは，GAN によって生成された断片内のエラーを自動的に検出し，そのエラーを修正するための手法を提案している [44]．まず，エラーの検出には多層パーセプトロンによるモデルを用いている．断片を格子状に分割し，断片のある地点（i, j）に対してエラー検出を行う場合，モデルの入力は地点（i, j）の y 座標とその周囲 8 マスのタイルの情報，出力は地点（i, j）が特定のタイルである確率となっている．Shu らの研究では，11 種類のタイルを扱っているため，モデルから出力されるのは 11 種類それぞれに対する確率となる．そのうえで，未知の断片の地点（i, j）のタイルについて，対応する確率が事前に設定した閾値 θ を下回るならば，地点（i, j）はエラーであるとしている．

検出したエラーを修正するにあたり，例えば「モデルの出力確率が一番高いタイルに変更する」という単純な方法が挙げられる．しかし，Shu らは，地点（i, j）のエラー検出を行うとき，そもそも入力となる周囲 8 マスにエラーがあれば，エラーではない地点（i, j）をエラーだと誤判定してしまう可能性が高くなると報告している．また，場合によっては，CMA-ES などによってある特徴を持つ断片を生成したものの，修正を重ねるうちにその特徴が失われてしまうということもあり得る．Shu らは，エラーの誤判定を減らしたり，修正前の特徴を保持したまま修正を施したりするために,「エラーと判定された地点の数」「修正された地点の数」などを評価値とする GA をエラー修正に適用した．その結果，GAN によって生成された元の断片と比較すると，修正後の断片ではエラーが減少し，かつ元の特徴を保持できていることが示されている．

Giacomello らは，FPS『DOOM』[45] のレベルを生成対象として，Volz らと同様に GAN と CMA-ES を用いた自動生成を試みている [46][47]．Giacomello らは，1000 個以上の既存のレベルを学習データとして生成モデルを学習し，部屋の数や通路の大きさなどを評価値とする CMA-ES によって特定の特徴を持つレベルを生成する潜在変数を効率よく探索している．

3.1.5 PCG via Reinforcement Learning

本項では，近年注目されている PCG via Reinforcement Learning（PCGRL）について述べる．PCG の中でも，強化学習を用いてコンテンツを生成する手法を PCGRL と呼ぶ．ゲーム分野ではこれまで，強化学習は主にプレイヤとしての AI を作成するために用いられてきた．例えば，強化学習によってスーパーマリオブラザーズのコンピュータプレイヤを学習させる場合，ゲームの局面（画面）が「状態」，ジャンプや走る動作が「行動」となり，ゴールへ到達したり敵に当たったりするとそれに応じた「報酬」を与えるといった具合になる．PCGRL ではまず，コンテンツ生成を強化学習の問題へどのように変換するのかが肝となる．

Nam らは，ターン制 RPG のステージを生成対象とし，強化学習による自動生成を試みている [48]．Nam らが対象とするのは，敵や回復を表すマスが一直線に並ぶような，一般的なターン制 RPG のステージを単純化したものである．各種類のマスはそれぞれパラメータを持ち，例えば回復マスなら回復率，敵マスなら HP・体力といったパラメータを持つ．そのようなステージを生成するうえで，Nam らは「生成途中の未完成なステージ」を状態,「未完成なステージ内の次の 1 マスのパラメータを決める操作」を行動,「評価関数によるステージの面白さ」を報酬と定め，多様性と品質のバランス調整を可能とするステージの生成に成功している．

Khalifa らは，強化学習を用いて 2D ゲームのレベルの自動生成を行っている [49]．レベルは 2 次元配列で表現され，配列の各要素には配置されるオブジェクトの種類（例えば，壁や敵など）を表す整数が格納される．そのうえで，Khalifa らは「2 次元配列」を状態,「オブジェクトの削除・変更」を行動,「評価関数によるステージの完成度」を報酬としている．学習の結果，レベルの生成が全て成功しているわけではないものの，強化学習を用いたアプローチがコンテンツ生成に効果的であることが示されている．

PCGRL は，PCGML とは異なり学習データを必要としない．また，ひとたび学習を終えれば，高速でコンテンツを生成することができる．一方，Search-based PCG と同様，評価関数の設計の難しさについては課題となっている．

3.2 迷路とダンジョンの自動生成

本節では迷路とダンジョンついて，自動生成アルゴリズムや関連研究を紹介する．

3.2.1 迷路の自動生成

一般的な迷路生成アルゴリズムとして，穴掘り法 [50] や壁伸ばし法 [51]，棒倒し法 [52] などが挙げられる．2.1 節で述べたとおり，難易度は迷路の面白さに影響を与えうる要素の 1 つである．難易度に焦点を当てながら迷路を生成する場合，Search-based PCG が適用されることが多い [53][54][55][56]．

Ashlock らは進化的アルゴリズムと動的計画法を組み合わせ，多様な外観や特徴を持つ迷路を自動生成している [53]．Ashlock らは，袋小路の数や袋小路の始まりから終わりまでの距離，正解路長など迷路の特徴に関する 5 つの評価関数を用いており，その中でも特徴的なのがチェックポイントに関する 2 つの評価関数である．チェックポイントとは，事前に定められた迷路内のある地点のことである．例えば，ある評価関数では，チェックポイントに多くの通路が接続されるほど評価値が高くなる．これにより，迷路内にプレイヤが到達しやすい場所を作ることが可能になっている．さらに，異なる 3 つの個体表現を用いることにより，同じ評価関数を用いた場合でも，異なる外観・特徴を持つ迷路を生成することに成功している．

Kwiecień は，Cockroach Swarm Optimization を用いて複雑な通路構造を持つ難しい迷路を自動生成している [55]．具体的には，McClendon[57] や Bagnall ら [58] の議論を参考に正解路長や通路の方向転換回数，分岐点間の距離などを複雑さの指標とし，複雑なほど評価値が高くなる迷路を生成している．

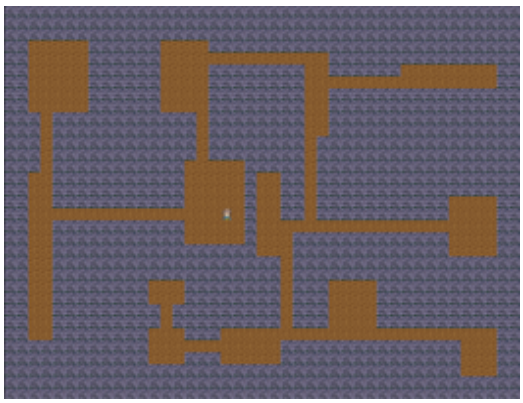
また，Adams らは，セルオートマトン（Cellular Automaton，CA）を用いて迷路を自動生成し，CA のルールを進化させるために GA を適用している [54]．評価値には正解路長や行き止まりの数など 3 種類の特徴を用いている．

Wu らは，主に難しさと美しさに着目しながら，パックマンで用いる迷路の自動生成を試みている．難しさの指標には分岐点の数や迷路の大きさ，美しさの指標には分岐点間の距離の標準偏差や迷路全体に対する通路面積の割合などを用いている．Wu らは，通路面積の割合などを美しさの指標として採用した具体的な理由を述べていないが，それらの指標について，パックマンで実際に用いられた迷路と生成した迷路の類似度が高いほど美しい迷路であるとしている．

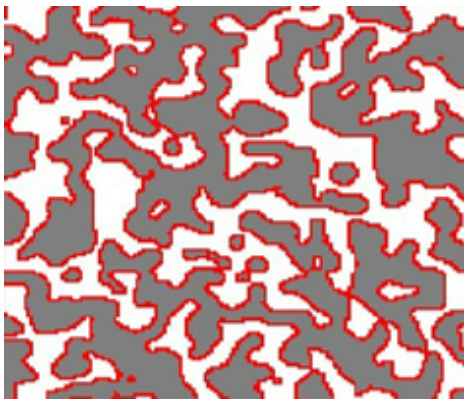
既存研究の多くは，難易度の評価値として正解路長や分岐点の数などの特徴を用いているが，それらの特徴と難易度の関連性については未解明な部分も多い．ただし，関連性が見られると報告する研究も少なくない [57][59][60]．例えば，分岐点について，その数が多いほどプレイヤの迷いが発生し得る地点が増えるため，難易度に関連すると報告されることは合理的であると考える．

3.2.2 ダンジョンの自動生成

敵やアイテムを含まず，純粋にマップのみを考えると，一般的に迷路は「通路」「壁」，ダンジョンは「通路」「壁」「部屋」から構成されることが多い．両者のマップ構造には共通する部分もあるため，穴掘り法などの迷路生成アルゴリズムをダンジョン生成に用いることもある．そのほか，ダンジョン生成によく用いられるアルゴリズムとして，空間分割（Space Partitioning）[61] や CA，生成文法（Generative Grammar）[62] などが挙げられる．各アルゴリズムには特徴があり，例えば，空間分割の一種であるバイナリ空間分割（Binary Space Partitioning，BSP）[63] によって生成したダンジョンは図 3.2(a) に示す人工物のようなマップに，CA によって生成したダンジョンは図 3.2(b) に示す自然物のようなマップになりやすい．ま



(a) BSP を用いて生成されたダンジョン



(b) CA を用いて生成されたダンジョン [64]

図 3.2: 人工物のようなダンジョンと自然物のようなダンジョン

た，生成文法には，ダンジョンのどこに何を配置するのか，どの部屋同士を接続するのかなど，生成規則を直感的に記述しやすいといった特徴がある．

Thompson らは，生成文法を用いたアプローチによって，2D アクションアドベンチャーゲーム用のダンジョンマップとマップ内でプレイヤに課すミッションの自動生成を試みている [65]．生成されたマップとミッションは，ゲームの進行を妨げることなく，プレイ可能であると報告されている．ただし，マップの構造やミッションの内容は，事前に用意されたテンプレートを自動生成に用いているため，多様性の観点では制限のある手法だと考える．

Green らは，マップの生成と敵やアイテムの配置，それぞれに CA など 3 種類のアルゴリズムを用いており，計 9 通りのアルゴリズムの組合せによってダンジョンを生成している [66]．生成されたダンジョンは，スタートとゴール間の距離や壁の配置など，アルゴリズムの組合せ方によって異なる特徴を持つ．また，Green らは，生成したダンジョンがプレイヤにどのようなプレイ体験を与えるのか，3 種類のテストプレイヤを用いて検証している．各テストプレイヤは「可能な限り少ない行動回数でゴールへ到達する」「敵の討伐を優先する」「アイテムの獲得を優先する」といった異なる目的を持つよう設計されており，同じテストプレイヤであっても，生成アルゴリズムが異なれば，敵の討伐数などのプレイ結果に顕著な違いが現れると報告されている．

ここまで紹介したアルゴリズムや研究は Constructive PCG に分類されるが，そのほかのアプローチによってダンジョンを生成する場合，学習データを必要としその模倣を得意とする PCGML ではなく，学習データを必要とせずゲームデザイナの要望を比較的反映しやすい Search-based PCG が用いられることが多い [67][68]．これは，ダンジョンというコンテンツを必要とするジャンルが，主に RPG やアクションアドベンチャーゲームであるからだと考える．例として RPG を挙げると，2.2 節で述べた通り，ダンジョンの構造や敵・アイテムの配置などはプレイヤの行

動，ひいてはゲームの面白さに大きな影響を与えうる．そのためマップをデザインするうえで，例えば，どの場所にどのアイテムを配置するのか，どのタイミングでプレイヤにアイテムを獲得させるのか，配置場所へどのようにプレイヤを誘導するのかなど，事細かな調整が必要となる場面が多い．こういった理由から，ダンジョン生成にはPCGMLよりもSearch-based PCGのほうが用いられることが多いのではないかと考える．ただし，PCGMLによる事例はいくつかあり[69][70][71]，近年ではPCGRLによるダンジョン生成も行われている[49]．

3.3 人間らしさを考慮したテストプレイヤとPCG

3.1節で述べた通り，Search-based PCGやPCGRLにはコンテンツの評価を行う過程が含まれる．評価方法として，例えば迷路であれば正解路長や分岐点の数などコンテンツの特徴から評価する方法，あるいはテストプレイヤによるシミュレーション結果から評価する方法などがある．

重要な評価基準として，例えば，人間プレイヤにとって適切な難易度であるのかといった点が挙げられる．これは難易度が高すぎれば，プレイヤはゲームをクリアできずに不満を覚えることがあり，逆に簡単すぎても，自身の成長や達成感を味わいにくくなるからである．また，ほかの評価基準としては，ゲームデザイナが想定したことを人間プレイヤが体験できるのかといった点が挙げられる．ゲームデザイナがある体験をプレイヤに与えようとしても，プレイヤが想定通りに行動しなければ，それはゲームデザイナの意図とは異なる体験になり，プレイヤに予期せぬ不満を抱かせることになり得るからである．このような理由から，テストプレイヤによるシミュレーションでは，敵が多すぎてゲームが進行できない状態になっていないか，テストプレイヤがどのような経路を辿って目的地に向かっているのかなど，さまざまな観点からゲームあるいはコンテンツを評価する．

ただし，テストプレイヤが人間プレイヤからかけ離れた行動をとると，適切な評価結果を得られない可能性が高まる．例として，敵が攻撃を開始したその1フレーム後に回避行動を取ることができる人間離れしたテストプレイヤを考え，そのテストプレイヤがあるステージを「1度もダメージを受けていない．だからこのステージは簡単である」と評価したとする．このとき，1度もダメージ受けていないのは人間離れした行動のためであり，人間に近い行動を取るテストプレイヤであれば評価が変動する可能性は十分にあり得る．このように，人間離れしたテストプレイヤによるシミュレーションが望ましくない場合も多いため，PCGにおいて人間らしさを考慮したテストプレイヤの需要は高い．以下では，PCGの評価部分において，人間らしさを考慮したテストプレイヤを用いた研究を紹介する．

Liapisらは，人間プレイヤが掲げるような目的を持たせたペルソナと呼ばれるテストプレイヤを作成し，ダンジョンの評価に活用している[72][73]．Liapisらは，ペルソナを作成するために7つの線形パーセプトロンを組合わせたネットワークを用いており，そのネットワークでは「アイテムとの距離」「ゴールまでの距離」

「敵との距離」などが入力，次に取るべき行動が出力となる．Liapis らは，ネットワークの重みを最適化するために進化的アルゴリズムを用いており，その評価値として 5 つの異なる目的関数を用いて，最適化を 5 試行することにより 5 種類の互いに異なるペルソナを獲得している．具体的には，各ペルソナは「可能な限り少ない行動回数でゴールへ到達する」「敵の討伐を優先する」「アイテムの獲得を優先する」など異なる目的を持って行動するように最適化されている．例えば，少ない行動回数でのゴールを目的とするペルソナであれば，移動した距離が大きいほど他のペルソナよりも評価値が低くなるよう設計されている．

このようにペルソナを作成した後，Liapis らは GA を用いてダンジョンを生成している．ダンジョンの生成と GA による最適化はペルソナごとに行われており，例えば，アイテムの獲得数が多くなるようなマップへと最適化する場合，評価関数はアイテムの獲得数などに関連するものとなり，その最適化を適切に進めるためにアイテムの獲得を優先するペルソナを用いている．ただし，そのように生成されたマップは「簡単に獲得できるアイテムが多く配置されているだけ」という場合も少なくないと報告されている．また，人間プレイヤによるマップの評価は行われていない．

Liapis らと同様に，Fernandes らもペルソナによるコンテンツ評価を行っている [74]．Fernandes らが対象としたコンテンツは，拠点を守りながら敵を全滅させることが目的となる 2D ゲームのレベルである．ペルソナはルールベースで複数の種類が作成されており，各ペルソナは「拠点の守りを優先する」「敵の討伐を優先する」などの目的を持つ．使用するペルソナによって，異なる特徴を持つレベルを生成することに成功しているが，それらが人間プレイヤにとって面白いのか，そもそもペルソナの行動は人間プレイヤに類似しているのかは検証されていない．

吉田は，シューティングゲームのステージの自動生成に向けて，行動に人間らしさを取り入れたテストプレイヤを用いている [75]．吉田は「人間プレイヤは正確で素早い操作が苦手」「人間プレイヤは将来あり得る危険を予測する」といった観点から，大まかな先読みや，ある程度の余裕を持った回避行動などを人間らしさとして取り入れている．その結果，テストプレイヤを用いて生成したステージは，ランダムに生成したステージと比較して，面白さが向上することが示された．

3.4 誘導に関する研究

人間に対する行動誘導は，ゲームのみならず現実世界のさまざま場面で必要となる．そのため，マーケティングや防災・減災など幅広い分野[1]で誘導に関する研究が進められてきた．本節では，明示的・非明示的を問わず，各分野における誘導手法やそれに関する研究を紹介する．

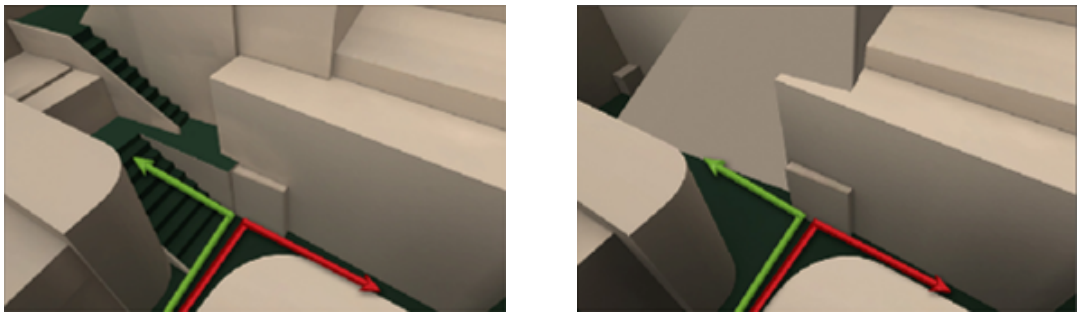
[1]誘導を意味する英単語はさまざまだが，著者が調査した範囲では，マーケティング分野の論文で “induction”，防災・減災分野やゲーム分野の論文で “guidance” といった単語が用いられていた．

マーケティング分野では，実店舗での売り上げ増加を目的とする顧客の行動誘導や誘導に必要な動線分析が行われている．スーパーマーケットなどの小売店を対象に顧客の動線分析を行った研究では，顧客は店舗中の特定の通路のみを移動する傾向があり，どの通路も満遍なく移動することは少ないという分析結果が報告されている [76][77]．ある通路へ顧客を向かわせたい場合，例えば店員の呼び込みのような明示的な誘導は，顧客に対して不快感を与えてしまう可能性もある．これに対して，ある通路へ顧客を非明示的に誘導したい場合には，その通路に配置された商品が良く見えるように視認性を高めることや，顧客の興味を引く商品を通路の端に配置することが効果的であると検証されている [78][79]．これらに類似する誘導手法はゲームにも存在しており，例えば，マップ上に配置されているアイテムはプレイヤの興味を引くオブジェクトの典型例である．

マーケティング分野と同様に，防災・減災分野においても誘導に関する研究が盛んに行われている．例えば，群衆の避難誘導を目的としたシステムの研究開発が進められており，シミュレーションよる出口や避難誘導灯の配置の最適化などが行われている [80]．防災・減災分野の研究と本研究では「誘導時に進むべき方向や目的地が明示されているか」が大きな違いとなる．災害時には，避難者に向けてわかりやすく誘導することが求められる場合も多いと考える．一方，本研究では，誘導の存在がプレイヤに認識されないように誘導することが重要になる．

ゲーム分野においても，これまでに紹介した分野と比較して事例は少ないものの，人間プレイヤの行動誘導に関する研究がある．Winters らは，どのようなマップ構造がプレイヤの興味を引き，誘導に効果的であるのか，異なる特徴を持つ 2 つの 3D マップを用いた被験者実験によって比較・検証している [23]．この実験では，マップ内のある地点に到達することが被験者の目的となる．2 つのグループに分かれた被験者は，グループごとに異なるマップを探索する．2 つのマップには図 3.3 に示すような差異が 6 か所あり，同じ状況下であっても，例えば図 3.3 の階段とスロープならば，階段であるほうがプレイヤの選択割合が有意に高くなるという結果になっている．ただし，そのほかの 5 つの差異については有意差は見られず，どのようなマップ構造が誘導に効果的であるのか明確には結論づけられていない．

また，Johanson らは，誘導の強度がプレイヤの快適さや探索能力の向上にどのような影響を与えるのかを検証するため，3D マップを用いた被験者実験を行っている [81]．実験におけるプレイヤの目的は，マップ上のある地点を目指したり，あるマークを発見したりすることである．各プレイヤは，トレーニング・テストの 2 回のマップ探索を行い，トレーニング中には（1）誘導なし,（2）マップ上における自身の現在地が示される（弱い誘導）,（3）行先を示す矢印が表示される（強い誘導）のいずれかの誘導を受ける．マップ探索後のプレイヤにアンケートを取った結果，誘導を受けなかったプレイヤと比較して，弱いあるいは強い誘導を受けたプレイヤは探索のフラストレーションが減少する傾向にあることが示されている．さらに，トレーニング中に誘導を受けたプレイヤであっても，テスト探索における探索能力の低下は見られなかったと報告されている．ただし，実験で使用



(a) 緑矢印の方向に階段があるマップ A　　(b) 緑矢印の方向にスロープがあるマップ B

図 3.3: 実験に使用されたマップの差異の例 [23]．プレイヤは緑または赤の矢印の方向へ進むことができる．

されたのは FPS のマップであることや，トレーニングとテストで同一のマップを使用していること[2]を考慮すると，本研究のように迷路やダンジョンを対象とした場合，同様の結果が得られるのかは疑問が残る．

このように，プレイヤの誘導に着目した既存研究はいくつかあるものの，そこで得た知見をもとにプレイヤを誘導するマップの自動生成を試みた研究は，著者が知る限りではあまり見かけない．

[2] トレーニングとテストでは，プレイヤのスタート地点が異なる．

第4章　提案手法

本章では，プレイヤを自然に誘導するマップの自動生成に向けた提案手法の概要を述べる．図 4.1 に示すように，提案手法は主に（1）人間プレイヤのプレイログ収集,（2）教師あり学習による人間プレイヤの行動選択予測モデルの構築と行動傾向の調査,（3）予測モデルに基づく人間らしさを考慮したテストプレイヤの作成,（4）テストプレイヤの評価に基づく自然に誘導するマップの自動生成の 4 段階で構成される．4.1 節と 4.2 節では，研究対象である迷路とダンジョン探索ゲーム（ダンジョン）のそれぞれについて，人間プレイヤのどの行動に着目して選択の予測や傾向の調査を行うのかを述べる．

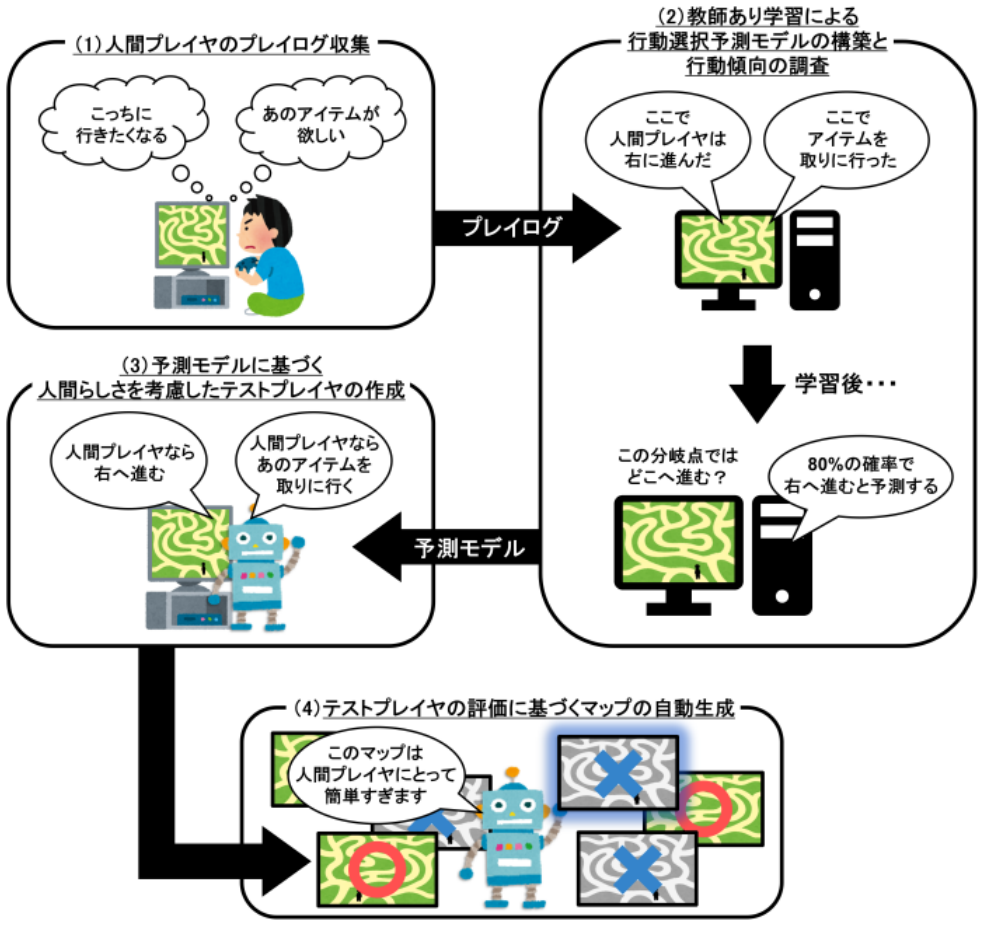


図 4.1: 提案手法の流れ

4.1　迷路の自動生成に向けた提案手法

2.1 節では，迷路の難易度がその面白さに影響を与えうることを述べた．また，3.2.1 項では，難易度に焦点を当てながら迷路を自動生成する場合，GA などのメタヒューリスティックが適用されることが多く，その評価値は迷路の特徴，例えば正解路長や分岐点の数などに関連すると紹介した．本研究では，それらの特徴の中でも分岐点に着目し，迷路の自動生成や難易度の調整を行う．

既存研究では，分岐点の数を難易度の指標の一つとして用いることがあるが，一口に分岐点といっても，人間だからこそ迷ってしまう，あるいは特定の方向へ進みたくなってしまう場所など，さまざまな分岐点が迷路には存在するはずであり，その点を考慮し，人間にとっての難易度こそを調整するべきだと考える．本研究では，そのような人間らしさを考慮した難易度の迷路を自動生成する．

迷路の自動生成に向けた提案手法ではまず，教師あり学習を用いて分岐選択確率予測モデルを構築することに加え，人間プレイヤの分岐選択の傾向を調査する．次に，予測モデルを用いることで傾向を模倣する，つまり人間らしい分岐選択を核とするテストプレイヤを作成する．最後に，テストプレイヤによる難易度評価を用いて，人間だからこそ難しい，あるいは簡単な迷路を生成する．

4.2　ダンジョンの自動生成に向けた提案手法

2.2.3 項では，RPG におけるマップ探索の面白さを生み出すために，プレイヤに適度な迷いが生じる状況やリソース管理を求められる状況が必要になる場合が多いと述べた．これを踏まえて，2 つ目の対象であるダンジョンでは，以下の条件を満たすマップの自動生成を目指す．

- HP とアイテムの要素を含む．
- HP は定期的に減少する（これによって擬似的な敵を表す）
- HP が多い状態でクリアできるほど簡単ではなく，しばしば HP が 0 になるほど難しくもない．
- アイテムは複数配置される．
- 無暗に探索してアイテム全てを獲得できるほど簡単ではなく，アイテムを 1 つも獲得できないほど難しくもない．
- 探索するプレイヤにはある程度の迷いが生じる．例えば，どの通路を進めばアイテムを獲得できるのか，簡単に把握できるほど単純なマップ構造ではない．また，プレイヤがマップ内における現在地を把握できなくなったり，ある分岐点へ戻る方法が分からなくなったりするほど複雑なマップ構造でもない．

これらの条件を満たすよう自然な誘導を施すが，そのために必要な人間プレイヤの行動予測や傾向調査をするうえで，いくつかの点に注意する必要がある．まず，1 つ目の対象ゲームである迷路は壁と通路のみで構成されるが，ダンジョンは壁と通路，そして部屋から構成される．加えて，ダンジョンには HP とアイテムの要素が含まれる．これにより，迷路のように分岐点の行動に着目するだけでは，ダンジョンにおける人間らしさを十分に考慮したテストプレイヤを作成することはできないと予想する．これは，例えば「HP が枯渇しそうだからアイテムは諦めて引き返す」「ゴールが見えるけれど，別の方向にアイテムがあるからそちらへ進む」など，ダンジョン特有の行動が発生するからである．

ダンジョンの自動生成に向けた提案手法ではまず，教師あり学習を用いて，分岐選択確率予測モデル，部屋における行動予測モデル，探索中の引き返しが発生するタイミングの予測モデルの 3 つを構築する．その後の人間プレイヤの傾向を模倣したテストプレイヤの作成と，テストプレイヤによるマップ評価と生成は，迷路と同様である．ただし，本論文では，予測モデルの構築までを扱い，テストプレイヤの作成とマップの自動生成は行わない．

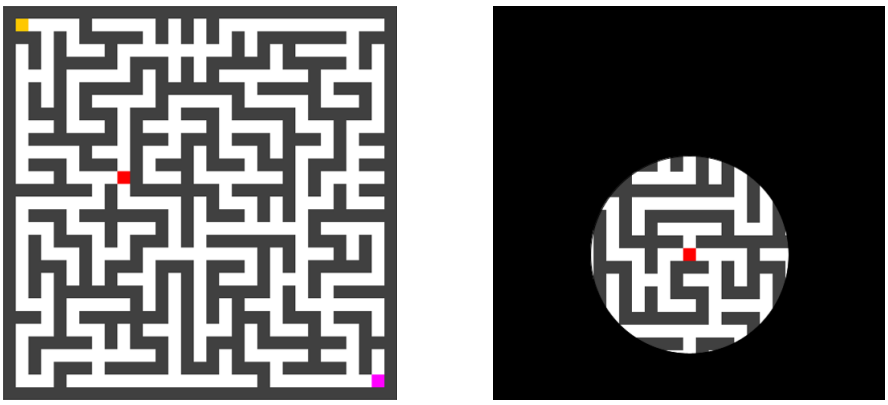
第5章　迷路における人間プレイヤの分岐選択確率の予測

本章では，教師あり学習によって，迷路における人間プレイヤの分岐選択確率予測モデルを構築する．5.1 節では，対象迷路の設定や学習データを収集するため実施した被験者実験の概要について述べる．5.2 節と 5.3 節では，学習の内容とその結果をそれぞれ与える．

5.1　学習データ収集のための被験者実験

本研究の対象は，迷路の中でも最も一般的と思われる 2 次元格子状の全域木型迷路である．本研究が対象とする迷路の具体的な特徴を以下にまとめる．図 5.1 はそのような迷路の例である．

- プレイヤの目的は，スタートからゴールに辿り着くことである．
- スタートは迷路の左上隅，ゴールは迷路の右下隅のマスに設置する．
- 迷路のサイズは 31×31，41×41，51×51 マスのいずれかである．以降では，それぞれのサイズを S，M，L と呼ぶ．
- 制限時間があり，制限時間を過ぎてもゴールに辿り着いていない場合，プレイヤはゲームオーバーとなる．制限時間は迷路サイズによって異なる．S は 80 秒，M は 100 秒，L は 150 秒である．
- 迷路は，通行可能なマスと通行不可能なマスのみで構成され，それぞれを通路マス，壁マスと呼ぶ．
- 迷路の外周部分は，壁マスで構成される．
- プレイヤは 1 回の行動につき 1 マス分のみ，現在地の上下左右に隣接するいずれかの通路マスに移動できる．
- マップ左上の頂点座標を $(1, 1)$ とすると，通路マスの配置候補となるのは，x 座標，y 座標，あるいはその両方が偶数となるマスである．言い換えれば，x，y 座標がどちらも奇数の場合，そこは必ず壁マスとなる．



(a) Wide　　(b) Narrow

図 5.1: 対象とする迷路の例．左上隅の黄色マスはスタート，右下隅のピンクマスはゴール，赤色マスはプレイヤの現在地をそれぞれ示す．

- ループ（閉路）は存在しない．到達できない通路マスも存在しない．これらの特徴によりプレイヤが同じ通路マスを 2 回以上訪れない限り，スタートとゴールを結ぶ経路（正解路）は必ず存在し，かつ一意に定まる．
- プレイヤは，以下のいずれかの視界状態を持つ．
  - Wide：図 5.1(a) のように迷路全体が見える状態．
  - Narrow：図 5.1(b) のようにプレイヤを中心とする円の内部のみ見える状態．円の直径は，迷路一辺の半分の長さである．例えば，S サイズの迷路の場合，円の直径は $31/2 = 15.5$ となる．本来であれば，人間プレイヤの実際の視界範囲を調査した後，円の大きさを定義する必要がある．しかし，人間プレイヤの視界範囲の定義は，本研究の主な目標には含まれない．そこで，人間プレイヤの視界や，商業用 RPG マップの視界範囲を考慮しつつ，不適切ではない程度の大きさになるよう設定した．
- 迷路はディスプレイ上に映り，プレイヤはキーボード入力による操作を行う．
- プレイヤが辿ってきた道のりは，ディスプレイ上に可視化されない．また，Narrow 迷路において，これまでのプレイヤの移動によって視認してきた領域があったとしても，その領域がプレイヤの視界範囲外にあるならば，その領域は可視化されない．

この設定のうえで，実験に用いる迷路は穴掘り法 [50] と壁伸ばし法 [51]，棒倒し法 [52] のいずれかを用いて生成した．実験に参加した被験者は 20 名（20〜40 代の男女）である．各被験者は，事前に指定された迷路の設定や解く順番に従い，サイズや視界状態を変更しながら計 21 個の迷路をプレイした．被験者の迷路を解いた順番については付録 A.1 を参照されたい．

5.2　学習内容

本節では，5.1 節で得たデータをもとに行う教師あり学習の内容について述べる．学習内容を述べる前にまず，本研究における「未確定」「分岐点」「分岐」という用語を定義する．この定義は，Wide 迷路と Narrow 迷路で共通である．どちらの場合も，Narrow 迷路の視界状態を用いて定義する．

迷路内において，プレイヤの現在地からある方向に注目したときに「その先に到達可能な通路マスが，1 つでも Narrow 視界範囲外にある」ならば，その通路は「未確定」とする．例えば，図 5.2(a) でプレイヤの右方向に注目すると，その先（緑斜線の部分）が行き止まりなのか，ゴールへ続くのか判断できないため，右方向に続く通路は未確定となる．一方，左方向の先（青縦線の部分）は，その通路を構成する通路マスが全て見えるため未確定ではない（以降，未確定でない通路を確定通路と呼ぶ）．なお，この「未確定」の判定は，現在の視界で得られる情報のみで行われることに注意する必要がある．つまり，これまでのプレイヤの移動によって得られた情報を覚えていれば行き止まりと分かる場合でも，上記の条件を満たしていれば未確定として扱う．

このうえで「プレイヤが進行してきた方向の通路を除き，未確定な通路が 2 つ以上接続されている通路マス」を「分岐点」と定義する．図 5.2(a) のプレイヤの現在地は，一般的な意味での分岐点ではあるが，本研究では分岐点とは呼ばない．一方，図 5.2(b) ではプレイヤの現在地は 2 つの未確定な通路と，図 5.2(c) では 3 つの未確定な通路と接続されているためそれぞれ分岐点となる．これに加えて，「プレイヤが進行してきた方向の通路を除き，分岐点に接続されている通路」を「分岐」と定義する．分岐であるための条件として，未確定であるかどうかは含めない．

この分岐点の定義を踏まえたうえで学習の内容を述べる．この教師あり学習では，人間プレイヤの分岐選択について予測する．目的変数となるのは，5.1 節の実験データから算出した被験者の分岐選択割合である．例えば，図 5.3 に示すような上と下方向に進行可能な分岐点において，20 人中 16 人のプレイヤが下方向を選ん

(a) 分岐点ではない　(b) 分岐点　(c) 分岐点

図 5.2: 分岐点である通路マスとそうでない通路マスの例

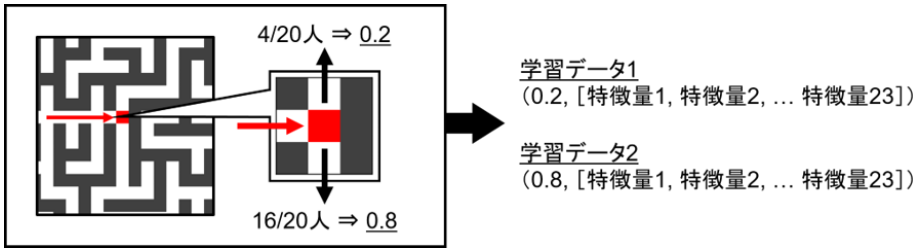


図 5.3: 学習データの作成方法

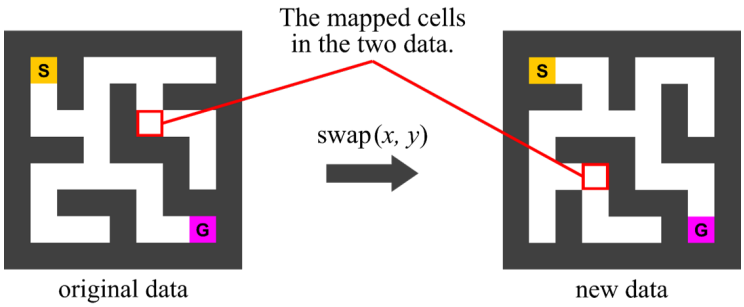


図 5.4: x 座標と y 座標の入れ替えによる学習データの水増し方法

だ場合，下方向の選択割合は $16/20 = 0.8$ となる．図 5.3 に示す分岐点の場合，目的変数が 0.2 のデータと 0.8 のデータ，計 2 つの学習データ得ることができる．ただし，学習データや予測対象とするのは，未確定な通路のみである．この選択割合を学習するために「分岐点の座標」「分岐の方向とゴールの方向のなす角度に対する $\cos\theta$」など計 23 個の特徴量を用いる．特徴量の詳細については，付録 A.2 を参照されたい．

実験データより，Wide 迷路は 147 個，Narrow 迷路は 197 個の学習データをそれぞれ得ることができた[1][2]．さらに，図 5.4 のように x 座標と y 座標を入れ替えたデータを作成することで学習データ数を 2 倍にしている．学習には決定木ベースの勾配ブースティングモデルである LightGBM[82][3]を，精度検証には leave-one-out 交差検証を用いる．

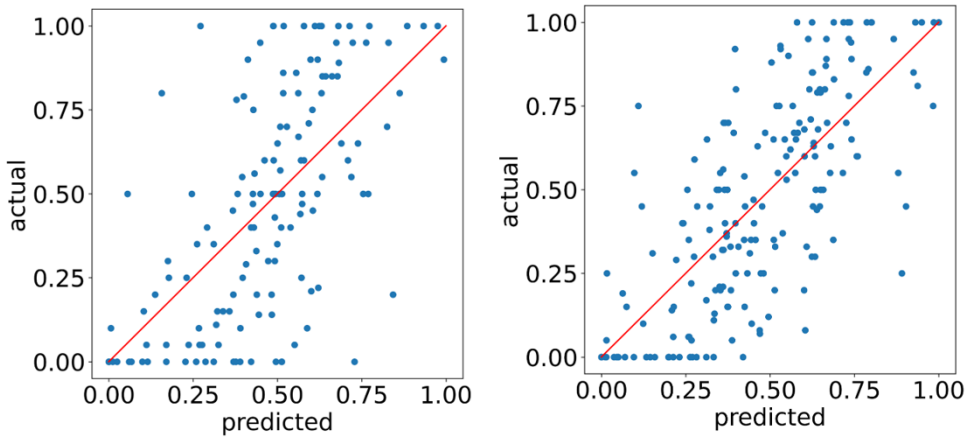
予測モデルは 1 つの分岐点に関する複数の選択割合を別々に予測するため，予測値の和が 1.0 になるとは限らない．そこで，1 つの分岐点に関する予測対象が n 個あるとき，予測値の和が 1.0 になるように次式を用いて各予測値 p_i を正規化し，確率分布へと変換する．

$$f(p_i) = \frac{p_i}{\sum_{k=1}^{n} p_k} \tag{5.1}$$

[1]学習データは，プレイヤが分岐点を初めて訪れたときのデータのみを収集対象としている．
[2]学習データ数は，分岐点の数ではなく，分岐点から得られた予測対象の分岐の数を示している．
[3]バージョンは 3.2.1 である．学習時のパラメータは以下のとおりである．objective：regression, learning_rate:0.01, boosting_type:gbdt, metric:rmse, num_leaves:7, max_depth:5

5.3　学習結果

図 5.5 は，横軸に学習データを水増しした予測モデルによる予測値を正規化した値，縦軸に実験データから得た実測値をとる散布図である．表 5.1 は，正規化や水増し処理を行わない場合の結果も含めて，予測値と実測値の二乗平均平方根誤差（RMSE），ピアソンの積率相関係数をまとめたものである．RMSE は小さいほど，相関係数は大きいほど予測精度が高いことを意味する．正規化と水増し処理を行った場合の相関係数は 0.66（Wide），0.74（Narrow）であり，特に Narrow においてはやや強い正の相関となっているため，予測精度に改善の余地はあるものの，人間プレイヤの分岐選択確率をある程度の精度で予測できていることが示されている．



(a) Wide　　(b) Narrow

図 5.5: 学習データの水増し処理を行った予測モデルによる分岐選択確率の予測結果（横軸：正規化後の予測値，縦軸：実測値）

表 5.1: 迷路を対象とする各予測モデルの精度比較

視界状態	予測モデル	RMSE	相関係数
Wide	正規化なし＋水増しなし	0.30	0.50
	正規化あり＋水増しなし	0.30	0.50
	正規化なし＋水増しあり	0.31	0.57
	正規化あり＋水増しあり	0.26	0.66
Narrow	正規化なし＋水増しなし	0.23	0.69
	正規化あり＋水増しなし	0.24	0.68
	正規化なし＋水増しあり	0.23	0.70
	正規化あり＋水増しあり	0.21	0.74

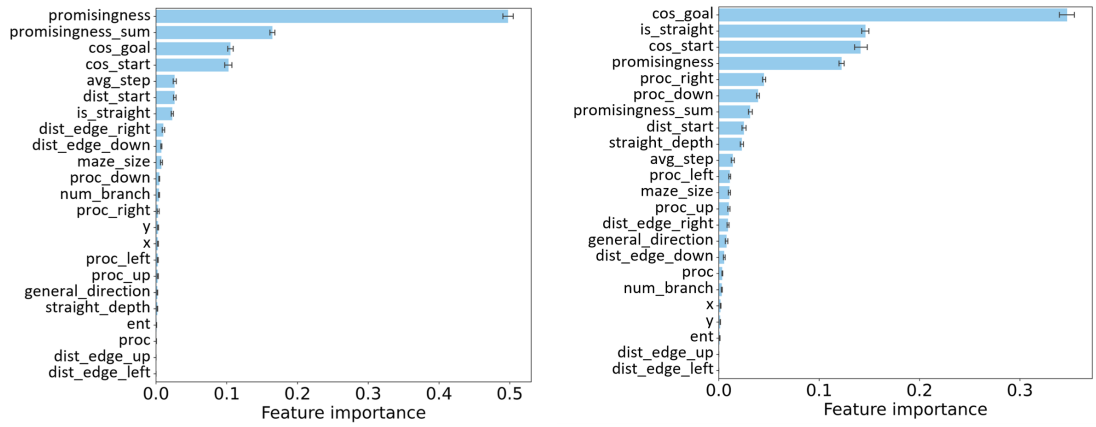
5.3.1 予測精度に関する分析

予測精度について，水増しを行った場合とそうでない場合の結果と比較すると，Wide・Narrow 迷路ともに水増しによる予測精度の改善が表 5.1 より確認できる．また，正規化についても，水増しと組合わせることによって予測精度の改善に貢献していることがわかる．

Wide 迷路と比較して，Narrow 迷路の予測精度が高いことには主に 2 つの理由があると推測する．1 つ目は，Wide 迷路よりも Narrow 迷路の学習データ数が多いため．2 つ目は，視界制限があることによって，Narrow 迷路におけるプレイスタイルが限られ，その分人間プレイヤの分岐選択が予測しやすくなったためだと推測する．2 つ目の理由について，まず視界制限がない Wide 迷路を考えると，プレイスタイルとして「ゴールまでの経路を先に把握したうえで分岐選択する」「事前の経路把握などはせずに分岐選択する」などが挙げられる．一方，視界制限がある Narrow 迷路では，ゴールまでの経路を先に把握することができないため，Wide 迷路と比較してプレイスタイルの種類が限られる．その分，人間プレイヤごとに生じていた分岐選択のばらつきが減少し，予測しやすくなった結果，予測精度が比較的高くなったと推測する．

5.3.2 人間プレイヤの分岐選択の傾向

人間プレイヤの分岐選択傾向を調査するため，水増し処理を行った予測モデルから特徴量重要度を算出する．特徴量重要度とは，学習に用いた特徴量が予測にどの程度貢献しているのか数値化したものである．特徴量重要度の算出には LightGBM の機能を用いている．図 5.6 は，Wide，Narrow 迷路それぞれに対する特徴量重要度を示したグラフである．



(a) Wide　　(b) Narrow

図 5.6: 迷路における分岐選択確率予測モデルの特徴量重要度

Wide 迷路すなわち視界制限のない迷路において，最も重要度が高い特徴量は promisingness（重要度 0.5）であった．特徴量の詳細は付録 A.2 で述べているため省略するが，promisingness は「予測方向の先にある到達可能な全ての通路マスの個数」を示す特徴量である．promisingness の値が小さければ，続く通路が行き止まりであるか，ゴールに繋がっているのか判断しやすくなる．一方，promisingness の値が大きければ，特に制限時間がある中では，その判断が比較的難しくなるはずである．そのため，promisingness の値が小さい不正解路に続く方向は選択確率が低く，値が大きい方向は正解・不正解にかかわらず選択確率が高くなるといった理由により，promisingness が重要視されたと推測する．

Narrow 迷路について結論から述べると，人間プレイヤには（1）上や左方向よりも下や右方向を選ぶ傾向と（2）下と右の両方向が選択可能な場合，直進方向を選ぶ傾向があると考える．図 5.7 は，形状の異なる分岐点ごとにまとめた，被験者たちの各方向の選択割合である．読みやすさの観点から割合が 0.5 未満のデータを，信頼性の観点からサンプル数が 20 未満のデータをそれぞれ除外している．

図 5.7 より，下や右方向の選択割合が高いことが分かるが，これはゴールが迷路の右下隅に固定されているためだと推測できる．また，Narrow 迷路において最も重要度が高い cos_goal（重要度 0.35）は，予測方向が右や下であれば値が大きくなり，そうでなければ値が小さくなる特徴量である．つまり，cos_goal が最も重要であるとする結果は，予測モデルが人間プレイヤの傾向（1）を十分に反映した結果であると言える．そのほかに，予測対象の方向を示す特徴量として proc も挙げられるが，proc は Wide・Narrow 迷路のどちらにおいても，cos_goal と比較して重要度が低い．予測対象の方向は重要であるが，例えば，右方向が選択可能な分岐点であったとしても，それが迷路の右上の領域に位置する場合と左下の領域に位置する場合とでは，右方向の選択割合に差が生じることがあり得る．予測対象の方向だけではなく，分岐点とゴールとの位置関係が重要であるため，位置関係を考慮していない proc は重要度が低い結果になったと推測する．

さらに，図 5.7 において，下と右の両方向が選択可能な場合（赤矢印）に注目すると，いずれも直進方向の選択割合が高いことが分かる．Narrow 迷路で 2 番目に重要度が高い特徴量は is_straight（重要度 0.15）であり，これも予測モデルが傾向（2）を十分反映した結果であると言える．

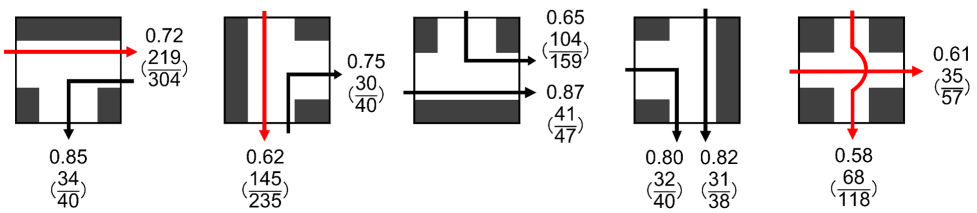


図 5.7: Narrow 迷路を対象とし，形状の異なる分岐点ごとにまとめた被験者の分岐選択割合

第6章　人間らしさを考慮したテストプレイヤによる難易度推定を用いた迷路の自動生成

人間の分岐選択傾向を考慮した難易度の迷路を自動生成するうえで，難易度推定には人間らしさを考慮したテストプレイヤを用いる．本章では，テストプレイヤの作成方法と難易度の定義，難易度の推定方法，さらに生成した迷路を評価するための被験者実験について述べる．

6.1　人間らしさを考慮したテストプレイヤ

本節では，人間らしさを考慮したテストプレイヤの作成方法について述べる．テストプレイヤの行動を説明するうえで，迷路の視界状態はNarrowと仮定することに注意されたい．テストプレイヤの行動は，その現在地となる通路マスが分岐点（5.2節で定義）であるかどうかで異なる．

通路マスが分岐点ではない場合の行動をアルゴリズム1に示す．アルゴリズム1では，スタートを根とする木に迷路を見立てたとき，テストプレイヤの現在地の親ノードを「前のマス」，子ノードを「次のマス」と呼ぶ．また，テストプレイヤは，図6.1(a)のようにゴールが見えている場合を除き，5.2節の図5.2(a)で示した青縦線部分のような確定通路には移動しない．アルゴリズム1において特に重要なのが8・9行目であり，条件（1）と（2）に該当する状況の例を図6.1(b)と6.1(c)にそれぞれ示す．これは人間プレイヤの行動により近づけるため導入したヒューリスティックである．例えば，条件（1）について，対象迷路はゴールが迷路の右下隅に設置されループも含まないため，外周に隣接する通路上で上や左方向に進むことはその通路が不正解路であることを意味する．被験者実験でも，そのような通路を突き進む被験者はあまり見られなかった．

分岐点におけるテストプレイヤの行動は，その分岐点を初めて訪れたかどうかで異なる．ある分岐点を初めて訪れた場合，5章で構築した予測モデルを用いて，確定通路を除く全ての分岐の選択確率を予測し[1][2]，その確率分布に従い進む先をラ

[1]図6.1(a)のようにゴールが見える場合には，予測ステップを省きゴールへ続く通路を進む．
[2]分岐点進入前にテストプレイヤがいた方向は予測対象から除く．

アルゴリズム 1 迷路の分岐点ではない通路マスにおける行動

Require: *state* : テストプレイヤの状態
1: **if** *state* = 通路を進む **then**
2: 次のマスに移動する
3: **else if** *state* = 最後に通過した分岐点まで戻る **then**
4: 前のマスに移動する
5: **end if**
6: **if** 分岐点に着いた **then**
7: *state* ← 分岐点にいる
8: **else if** （1）外周に隣接するマスをゴールとは逆方向に進んだ **or**（2）続く通路が行き止まりだと判明した **then**
9: *state* ← 最後に通過した分岐点まで戻る
10: **else**
11: *state* ← 通路を進む
12: **end if**

(a) ゴールが見えている状況　(b) 右側の外周に隣接する通路を上方向に動いた状況　(c) 続く通路の先にある全ての行き止まりが見えた状況

図 6.1: テストプレイヤの行動と状況の例

ンダムに選択する．その分岐点を訪れるのが 2 回目以降の場合には，未訪問の通路のみを予測対象としたうえで選択確率を予測し進む．ただし，不正解路上[3]のある分岐点で 2 回目以降の予測を行う場合に限り，「5.2 節で述べた正規化処理前の予測値が 0.5 未満ならば，その予測値を 0.0 に変換する」という条件を加える．この理由について，例えば人間プレイヤは，もし不正解の方向に進んだとしても，その探索結果から迷路構造の一部を把握することができる．そのため，次回以降の分岐選択時，人間プレイヤなら「こちらは不正解路っぽい」など見当をつけ，未訪問の通路であっても無視することがあるのではないかと推測した結果，この条件を適用することとした．また，この条件を正解路上の分岐点に適用すると，テストプレイヤがゴールに辿り着かない可能性があるため，不正解路上の分岐点のみに適用した．

[3]正解路と不正解路は区別できるものとする．

6.2 迷路の難易度とその推定方法

迷路の難易度の指標には，正解路長や分岐点の数といった迷路の特徴，あるいは解くのに要する時間などさまざまなものが考えられるが，本研究では，6.1 節で作成したテストプレイヤが迷路を解いた際の歩数をもとに難易度を推定する．具体的には，ある迷路のスタートからゴールに到達するまでに必要な最短歩数を $n_{shortest}$，テストプレイヤがその迷路を解いたときの歩数を n_{total}，その差を $n_{extra} = n_{total} - n_{shortest}$ としたとき，n_{extra} が大きいほど難しい迷路であると定義する．ただし，これはあくまでも同一サイズの迷路を想定したときの定義である．例えば，サイズが 31×31 の迷路と 101×101 の迷路では，n_{extra} が同じ値であったとしても，人間プレイヤの悩む程度などは異なるはずである．

また，テストプレイヤの行動にはランダム性があるため，1 試行ではなく複数回試行したときの n_{extra} の平均値と標準偏差をもとに難易度を推定する．これを踏まえて，6.3 節で述べる評価実験では，以下の 5 つの難易度を用意する．

- $\mathrm{easy}_{\mathrm{lowSD}}$ (n_{extra} の平均値：小，標準偏差：小)
- $\mathrm{moderate}_{\mathrm{lowSD}}$ (n_{extra} の平均値：中，標準偏差：小)
- $\mathrm{moderate}_{\mathrm{highSD}}$ (n_{extra} の平均値：中，標準偏差：大)
- $\mathrm{difficult}_{\mathrm{lowSD}}$ (n_{extra} の平均値：大，標準偏差：小)
- $\mathrm{difficult}_{\mathrm{highSD}}$ (n_{extra} の平均値：大，標準偏差：大)

テストプレイヤによる n_{extra} の平均値が同じだとしても，迷路によってその標準偏差には差が生じるはずである．標準偏差が大きい迷路を複数の人間プレイヤが解いた場合，同じ迷路であっても各プレイヤは全く異なる難易度を体験してしまう可能性が高くなる．このような観点から，標準偏差が小さいことを示す lowSD の難易度と大きいことを示す highSD の難易度に分けている．

6.3 迷路評価のための被験者実験

本節では，テストプレイヤの推定した難易度が，人間プレイヤが実際に解いたときの結果に適合するのか検証するために実施した被験者実験について述べる．

6.3.1 実験内容

本実験に参加した被験者は 10 人であり，公正な評価のためいずれの被験者も 5.1 節の実験とは異なる被験者を採用した．本実験で使用する迷路は，サイズを L，視界状態を Narrow に設定し，穴掘り法 [50] を用いて自動生成した．実験の準備段階で 3

万個の迷路を生成し，各迷路を 10 回ずつテストプレイヤが解いた．その結果，n_{extra} の平均値が $[0, 50)$ であれば難易度を easy，$[150, 250)$ であれば moderate，$[350,)$ であれば difficult とした．さらに，それらの中で標準偏差が上位 7 個を highSD，下位 7 個を lowSD とした．生成した迷路の難易度の分布を示すヒストグラムを図 6.2 に示す．この設定のうえ，各被験者は計 35 個の迷路を表 6.1 に記す順番で解いた[4]．

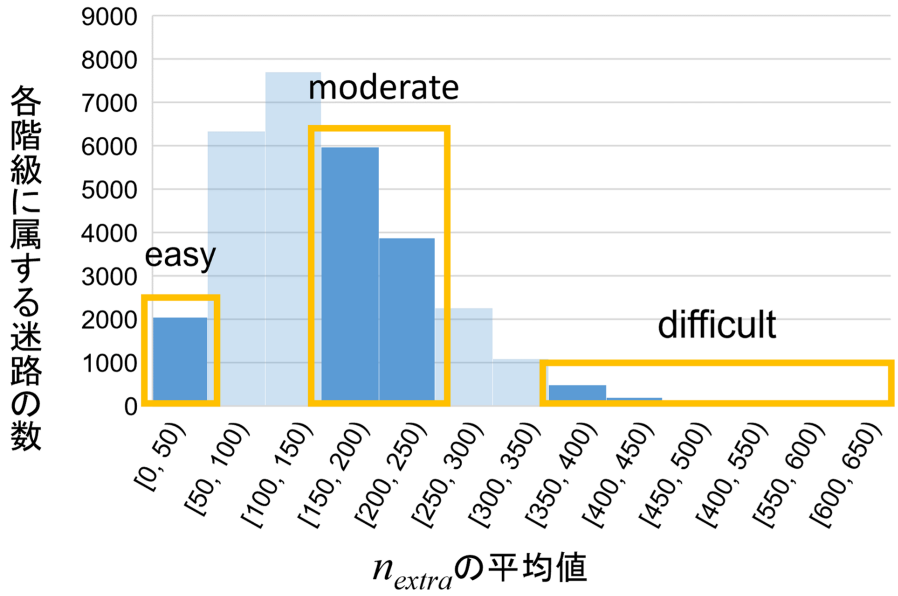


図 6.2: 難易度の分布を示すヒストグラム（横軸：テストプレイヤによる n_{extra} の平均値，縦軸：各階級に属する迷路の数）

表 6.1: 迷路を解いた順番と被験者による n_{extra} の平均値と標準偏差（SD）

$\text{easy}_{\text{lowSD}}$			$\text{moderate}_{\text{lowSD}}$			$\text{moderate}_{\text{highSD}}$			$\text{difficult}_{\text{lowSD}}$			$\text{difficult}_{\text{highSD}}$		
順番	mean	SD	順番	mean	SD	順番	mean	SD	順番	mean	SD	順番	mean	SD
5	200	194	1	71	43	3	53	113	4	299	173	2	257	180
9	72	131	10	241	197	6	243	206	7	364	123	8	231	162
14	69	44	12	285	114	11	195	209	13	371	121	17	447	162
18	77	116	19	220	137	15	135	214	16	406	150	20	368	200
21	175	151	23	228	140	24	232	201	22	262	133	25	265	234
26	48	47	29	178	152	27	228	180	28	517	159	30	104	173
33	27	25	32	235	42	31	314	291	34	374	77	35	140	184
Avg.	95	101	Avg.	208	118	Avg.	200	202	Avg.	370	134	Avg.	259	185

[4]5.1 節で述べた学習データ収集のための被験者実験と同様，1 つの迷路を解くときの制限時間は L サイズ用の 150 秒である．

6.3.2　実験結果

被験者による n_{extra} の平均値と標準偏差を表 6.1 の mean，SD の列にそれぞれ示す．まず平均値について，おおよそ easy < moderate < difficult となり，テストプレイヤの推定と人間プレイヤが解いた結果は適合することが示唆された．また，lowSD 迷路について Student の t 検定（$p < 0.05$）を行った結果，$\text{moderate}_{\text{lowSD}}$ は $\text{easy}_{\text{lowSD}}$ よりも平均値が大きく（$p = 0.0016$），$\text{difficult}_{\text{lowSD}}$ は $\text{moderate}_{\text{lowSD}}$ よりも平均値が大きく（$p = 0.0083$），いずれも統計的に有意差があることが示された．ただし，これらの平均値の中には，テストプレイヤによる推定区間から外れているものもあり，$\text{easy}_{\text{lowSD}}$ の平均値は区間 $[0, 50)$ より 45 程度大きく，$\text{difficult}_{\text{lowSD}}$ と $\text{difficult}_{\text{highSD}}$ の平均値は区間 $[350,)$ より 43，91 程度それぞれ小さくなった．この理由は 6.3.3 項にて考察する．

次に，lowSD と highSD 迷路の n_{extra} の標準偏差の差に注目する．lowSD と highSD 迷路の標準偏差の平均値について Student の t 検定（$p < 0.05$）を行った結果，$\text{moderate}_{\text{highSD}}$ は $\text{moderate}_{\text{lowSD}}$ よりも標準偏差の平均値が大きく（$p = 0.0142$），$\text{difficult}_{\text{highSD}}$ は $\text{difficult}_{\text{lowSD}}$ よりも標準偏差の平均値が大きく（$p = 0.0057$），いずれも統計的に有意差があることが示された．これは，迷路ごとの標準偏差の違い，つまり複数の人間プレイヤが同じマップを探索したときに体験し得る難易度の違いをテストプレイヤが適切に推定できていることを意味する期待通りの結果である．そのため，提案手法によって自動生成されたマップを用いることで，ゲームデザイナが意図した通りの体験を多くの人間プレイヤに与えることが見込める．

最後に，これらの結果を踏まえたうえで，実験に使用した $\text{easy}_{\text{lowSD}}$ と $\text{difficult}_{\text{lowSD}}$ の迷路を図 6.3(a) と図 6.3(b) にそれぞれ示す．色付きの通路マスはテストプレイヤの軌跡を示しており，灰色マスは正解路中の軌跡，赤色マスは不正解路中の軌跡である．この $\text{easy}_{\text{lowSD}}$ 迷路をある被験者が解くと，テストプレイヤと同じく図 6.3(a) の軌跡となる場合が見られた．また，ある被験者が図 6.3(b) の迷路を解いたときの軌跡は図 6.3(c) に示すものとなった．これらは，実験結果の中でもテストプレイヤと被験者の軌跡が特に一致した例である．

6.3.3　実験結果の考察

6.3.2 項の結果より，テストプレイヤの行動は人間プレイヤの傾向を反映している部分があり，その傾向に基づく難易度の迷路を生成できていることが分かる．また，テストプレイヤによる標準偏差の推定と人間プレイヤが解いた結果が大まかに適合したことは良い結果と言える．

しかし，人間プレイヤの結果に反して，テストプレイヤが難しさを（1）過大評価していた，あるいは（2）過小評価していた迷路もいくつか見られた．まず，(1) と（2）に共通する理由として，予測モデルに改善の余地があることが挙げられる．

(a) $\text{easy}_{\text{lowSD}}$ 迷路の例．テストプレイヤとある人間プレイヤの軌跡が完全に一致している例．

(b) $\text{difficult}_{\text{lowSD}}$ 迷路とテストプレイヤの軌跡の例

(c) $\text{difficult}_{\text{lowSD}}$ 迷路とある人間プレイヤの軌跡の例

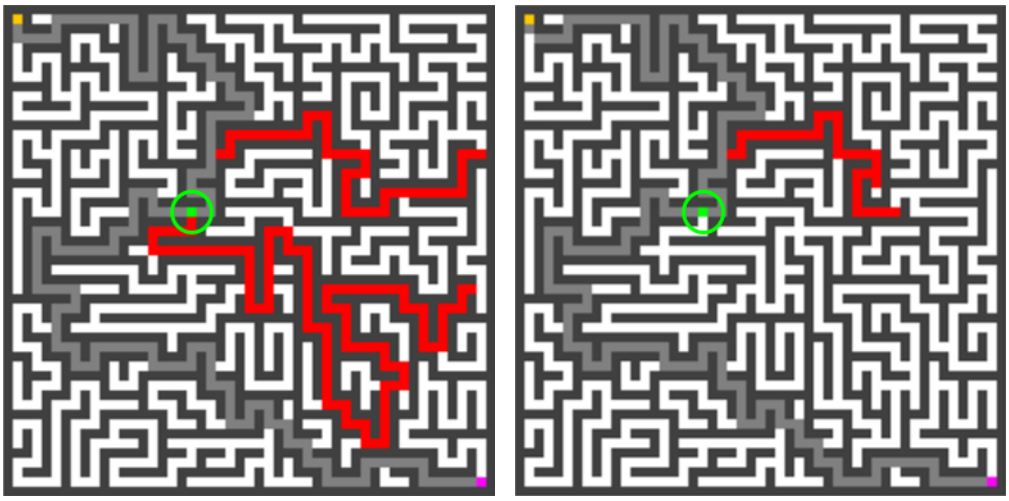
図 6.3: 自動生成した迷路の例．色付きの通路マスは迷路中のプレイヤの軌跡の例（灰色マス：正解路中の軌跡，赤色マス：不正解路中の軌跡）

図 6.4 は，ある分岐点（緑色マス）において，テストプレイヤと人間プレイヤが異なる分岐選択をした例である．

次に（1）について，これは制限時間に起因していると考える．被験者実験では制限時間を設けていたため，特に難しい迷路の場合，ゴールまで辿り着けずゲームオーバーになる被験者も少なからず見られた．一方，テストプレイヤが迷路を解く際には制限時間を設けず，必ずゴールまで辿り着くという条件のもと n_{extra} を計算していた．被験者とは異なり，テストプレイヤは際限なく迷路内を探索できてしまったことで，(1）が生じたと推測する．

（2）については，人間プレイヤの「思い込み」「勘違い」といった部分がテストプレイヤに十分反映されていないことが理由であると考える．例えば，非常に長い正解路を持つ迷路を人間プレイヤが解いているとする．プレイヤは正解路を進んでいる最中，長時間ゴールに辿り着けなければ「これは不正解路なのだろうか」と考え通路を引き返し，その分 n_{extra} が増加する可能性がある．一方，テストプレイヤにはそういった迷いがないため，(2）のような食い違いが生じたと推測する．また，ある人間プレイヤの思い込みが生じていた可能性のある例を図 6.5 に示す．ゲーム開始後，その人間プレイヤは正解路を進み続け，緑色マス[5]に到達後，青色マスへ進んだ．緑色マスへ引き返した人間プレイヤは，緑色マスの左方向に行くのではなく上方向，つまり来た道を戻っていった．迷路の左上の領域を長時間探索し続けた後，緑色マスの左方向の先へ進んだが，時間切れとなりゴールすることはできなかった．これは，緑色マスの左方向が「正解路ではない」「行き止まりになっている」といった思い込みが人間プレイヤに生じたためであると推測する．分岐選択のほかにも，このような「思い込み」「勘違い」といった人間らしさを取り入れることが（2）の解消に繋がると期待できる．

[5]図 6.5 の緑色マスは分岐点ではない．



(a) テストプレイヤの軌跡　　(b) 人間プレイヤの軌跡

図 6.4: テストプレイヤと人間プレイヤが異なる分岐選択をした $\mathrm{difficult}_{\mathrm{lowSD}}$ 迷路の例（灰色マス：正解路中の軌跡，赤色マス：不正解路中の軌跡，緑色マス：異なる分岐選択をした分岐点）

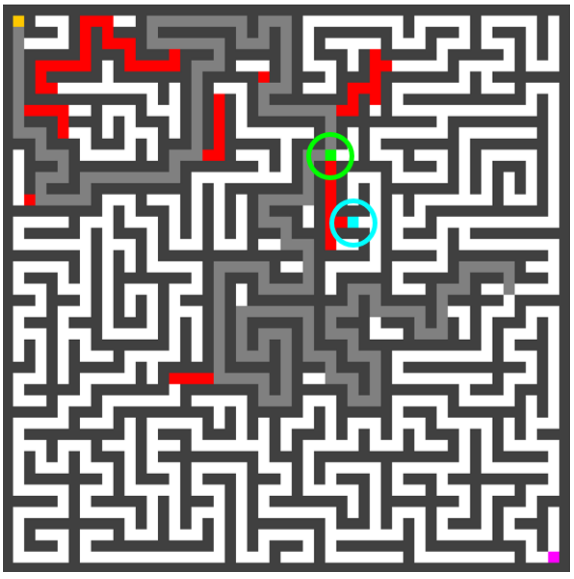


図 6.5: 人間プレイヤの思い込みが生じていた可能性のある $\mathrm{easy}_{\mathrm{lowSD}}$ 迷路の例（灰色マス：正解路中の軌跡，赤色マス：不正解路中の軌跡）

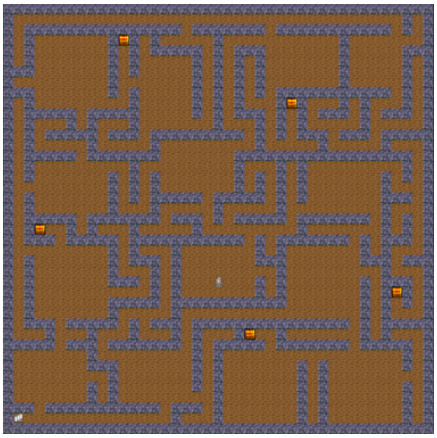
第7章　ダンジョンにおける人間プレイヤの行動選択の予測

6 章までの結果を受けて，研究対象をダンジョンへ発展させる．本章では，4 章で述べた通り，人間プレイヤの分岐点における分岐選択確率予測モデル，部屋における分岐選択確率予測モデル，探索中の引き返しが発生するタイミングの予測モデルの 3 つを構築する．7.1 節では，対象ダンジョンの設定や学習データを収集するため実施した被験者実験の概要について述べる．7.2 節と 7.3 節では，学習の内容とその結果をそれぞれ与える．

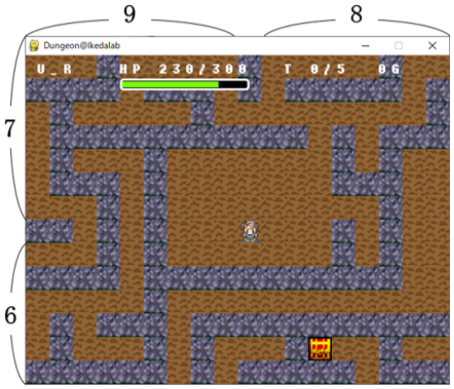
7.1　学習データ収集のための被験者実験

本研究の対象は，壁と通路，部屋から構成され，かつアイテムと HP の要素を含む 2 次元のダンジョンである．以下の点は，5.1 節の迷路と同じ特徴である．

- プレイヤの主な目的は，スタートからゴールに辿り着くことである．
- ゴールは，スタートの対角線上に位置する隅のマスに設置する．
- ダンジョンは，通行可能なマスと通行不可能なマスのみで構成され，それぞれを通路マス，壁マスと呼ぶ．
- ダンジョンの外周部分は，壁マスで構成される．
- 敵は存在せず，戦闘もない．
- プレイヤは 1 回の行動につき 1 マス分のみ，現在地の上下左右に隣接するいずれかの通路マスに移動できる．
- マップ左上の頂点座標を（1, 1）とすると，部屋内部を除き，通路マスの配置候補となるのは，x 座標，y 座標，あるいはその両方が偶数となるマスである．言い換えれば，x，y 座標がどちらも奇数の場合，そこは必ず壁マスとなる．
- ループ（閉路）は，部屋内部を除き存在しない．到達できない通路マスも存在しない．



(a) ダンジョンの全体像．マップの右上隅がスタート，左下隅がゴールである例



(b) 実際のプレイ画面

図 7.1: 対象とするダンジョンの例

- ダンジョンはディスプレイ上に映り，プレイヤはキーボード入力による操作を行う．
- プレイヤが辿ってきた道のりは，ディスプレイ上に可視化されない．

共通しないあるいは新しい特徴は以下のとおりである．図 7.1(a) はそのようなダンジョンの全体像の例であり，図 7.1(b) は実際のプレイ画面である．

- プレイヤは副目的として，できるだけ宝箱を探しその中のアイテムを取ることを目指す．
- スタートは，ダンジョンの四隅のマスのいずれかに設置する．
- ダンジョンのサイズは 41×41 マスである．
- ダンジョン内には，複数の通路マスで構成される長方形の空間が存在し，これを部屋と呼ぶ．
- ダンジョン内には宝箱が配置される．1 つのダンジョンに配置される宝箱の数は 3〜5 個である．また，配置場所はダンジョン内の行き止まり部分である．
- 宝箱の中には，ゴールドと呼ばれるアイテムのみが入っている．1 つの宝箱に入っているのは，10 ゴールド，20 ゴールド，30 ゴールドのいずれかである．
- プレイヤのパラメータとして HP が存在する．1 つのダンジョンにおいて，探索開始時点で与えられる HP の値は 300 である．
- プレイヤが 20 歩くごとに，HP が 10 減少する（これによって擬似的な敵を表す）．

- プレイヤは，HP が 0 になるとゲームオーバーとなる．
- HP を回復する手段はない．
- ダンジョンを探索するうえで，制限時間は設けられていない．
- マップ左上の頂点座標を（1, 1）とすると，部屋の左上頂点となるのは，x, y 座標の両方が偶数となるマスである．また，ある部屋のサイズは必ず奇数 × 奇数となる．
- プレイヤは，あるダンジョンにおいて，そのダンジョンに配置されている宝箱の総数および開封済みの宝箱の総数を知ることができる．
- プレイヤは，あるダンジョンにおいて，そのダンジョンで獲得したゴールドの累計を知ることができる．
- 図 7.1(b) に示すように，プレイヤの視界範囲は 14×18 マス（上方向に 7 マス，下方向に 6 マス，左方向に 9 マス，右方向に 8 マス）である．

この設定のうえで，付録 B.1・B.2 に記す手法により実験用のダンジョンを生成した．実験に参加した被験者は 18 名（20〜30 代の男女）で，9 人ずつの 2 グループに分けられた．各被験者は，左上隅スタート，左下隅スタート，右上隅スタート，右下隅スタートとなるマップをそれぞれ 20 個，計 80 個のマップを事前に指定された順番に従いプレイした．また，各グループは異なる 80 個をプレイしたため，計 160 個のマップのプレイログを収集できた．ただし，そのうち 2 個に不具合があったため，残りの 158 個のみを分析や予測モデルの学習に用いる．

7.2 学習内容

本節では，7.1 節で得たデータをもとに行う 3 つの教師あり学習の内容を述べる．

7.2.1 分岐点における分岐選択確率の学習内容

本項で述べる教師あり学習では，ダンジョン中の分岐点を対象とし，人間プレイヤの分岐選択を予測する．5.2 節の予測モデルと同様，目的変数となるのは，7.1 節の実験データから算出した分岐点における被験者の分岐選択割合である．5.2 節の予測モデルでは 23 個の特徴量を用いたが，ここではそのうちの 22 個に加えて，アイテムや HP に関する新たな 8 個を追加した計 30 個の特徴量を用いる．学習に用いた特徴量の詳細については，付録 B.3 を参照されたい．

実験データより，1482 個の学習データを得ることができた．また，5.2 節で述べたように学習データの水増しを行うが，迷路と異なりダンジョンのスタート位置

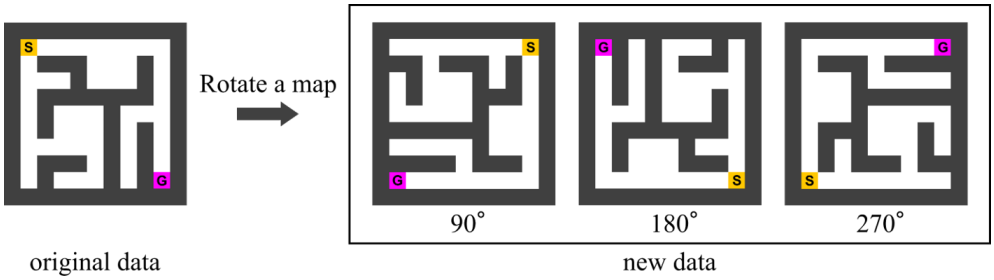


図 7.2: マップの回転による学習データの水増し方法

は左上隅に固定されていない．そこで，図 7.2 のようにマップを回転させることによる水増し方法を導入している．この回転による方法と，5.2 節で述べた方法を組み合わせることにより，学習データ数を 8 倍にしている．

5.2 節の予測モデルと同様，学習には LightGBM を，精度検証には leave-one-out 交差検証を用いて，かつ予測値の正規化も行う．

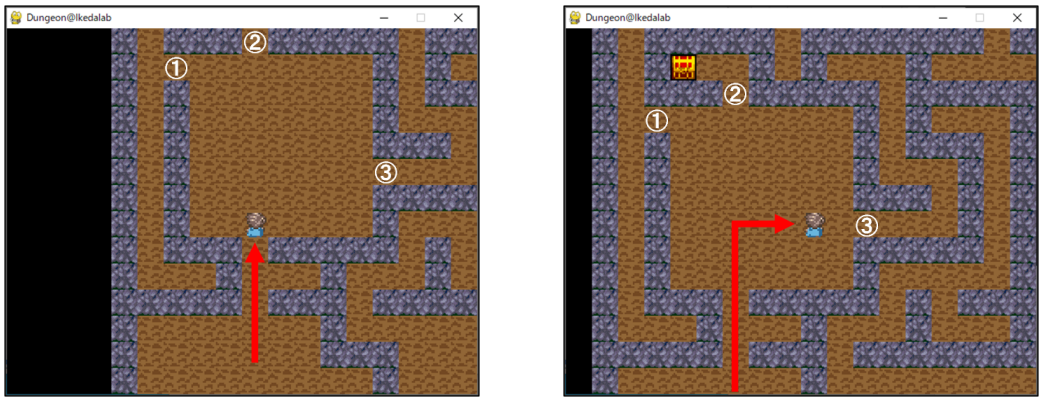
7.2.2 分岐部屋における分岐選択確率の学習内容

本項で扱う予測モデルの学習内容を述べる前にまず，本研究における「分岐部屋」という用語を定義する．

一口に部屋といっても，未確定な通路，あるいは確定通路，またはその両方が接続されている部屋など，いくつかの種類が存在する．また，1 つの通路マスから成る分岐点とは異なり，部屋は複数の通路マスからなる．そのため，部屋に接続される通路について，プレイヤが部屋のある地点から見た場合には未確定な通路であったとしても，同じ部屋の別の地点から見た場合には確定通路になることもあり得る．例えば，図 7.3 に示すように 1〜3 番の 3 つの通路と接続されている部屋を考える．図 7.3(a) において，プレイヤは下方向から部屋に進入し，部屋の入口に位置している状況にある．このとき，プレイヤから見れば，3 つの通路はいずれも未確定である．一方，図 7.3(b) のようにプレイヤが移動したとすると，2 番と 3 番の通路はどちらも確定通路となる．このように，部屋に接続される通路は，見る位置によって未確定であるかの判定が変化することがあり得る．

このうえで,「部屋の入口」から見たときに「プレイヤが進行してきた方向の通路を除き，未確定な通路が 2 つ以上接続されている部屋」を「分岐部屋」と定義する．つまり，部屋の入口以外の場所から見た場合に確定通路であったとしても，入口から見た場合に未確定であれば，その通路は未確定として扱う．また，部屋を含むダンジョンでは，分岐の定義を「プレイヤが進行してきた方向の通路を除き，分岐点または部屋に接続される通路」と改める．

この分岐部屋の定義を踏まえたうえで学習の内容を述べる．この教師あり学習では，ダンジョン中の分岐部屋を対象として，人間プレイヤの分岐選択を予測す



(a) プレイヤは部屋の入口にいる　　(b) プレイヤは部屋の中を移動している

図 7.3: 見る位置によって，接続されている通路の未確定の判定が変化する例

る．7.2.1 節の予測モデルと同様，目的変数となるのは実験データから算出した被験者の分岐選択割合であり，部屋への進入後に初めて踏まれた分岐のことを「選択された分岐」とする．学習に用いる特徴量は，7.2.1 項で述べた 30 個に新たな 4 個を加えた計 35 個である．特徴量の詳細については，付録 B.4 を参照されたい．

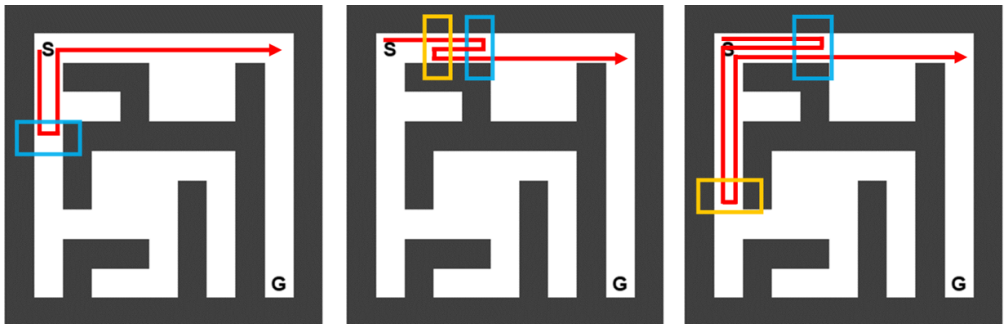
実験データより，2057 個の学習データを得ることができた．学習方法やデータの水増し方法，精度検証の方法については，7.2.1 節の予測モデルと同様である．

7.2.3　引き返すタイミングの学習内容

本項で述べる教師あり学習では，探索中の人間プレイヤが引き返すタイミングを予測する．具体的には，ダンジョンのある地点Aにある向きから進入したとき，地点 A や周囲の情報，そこを訪れたときのプレイヤの情報をもとに「地点 A を訪れたプレイヤがそこで来た道を引き返す確率」を予測するモデル（引き返しモデル）の学習を行う．本研究では，学習データを作成するにあたり，通路のある地点でプレイヤが引き返したか否かを以下の方法により判定する．

- あるダンジョンを探索するプレイヤの軌跡を見たとき，プレイヤが s 歩目で向いている方向と $s+1$ 歩目で向いている方向が逆であるならば，$s+1$ 歩目を「引き返した」とする（図 7.4(a) の青枠部分）．

- ただし，実際のプレイログには一度逆方向に進んだ後，間隔を空けずに再度逆方向へ進むようなデータが含まれることがある（図 7.4(b)）．このとき，1 度目の逆方向への進行が意図的なものであったという場合もあれば，単なる操作ミスだったという場合もあり得る．操作ミスは人間らしさを構成する要素の 1 つであり，学習データとして価値があると考える一方，本研究では意図的に引き返したデータを中心に予測モデルを学習させることを想定してい



(a) (b) (c)

図 7.4: 通路における「引き返した」と「引き返していない」の例（赤矢印：プレイヤの軌跡）

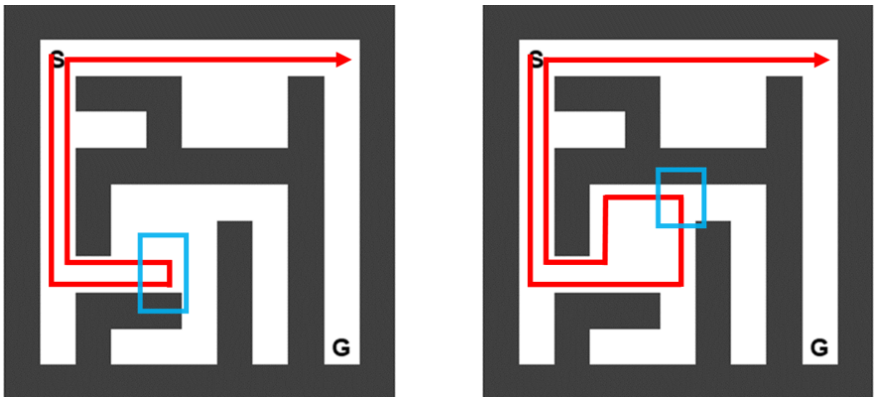
る．そこで本研究では，操作ミスから生じたものであれば「引き返していない」として扱うことにする．しかし，どのようなデータを操作ミスとするのか，言い換えると「ある逆方向への進行と次の逆方向への進行までに，どの程度の間隔が空いていれば前者を『引き返した』とするのか」は判断が難しいところである．

これに対して本研究では，間隔を表す変数 n を導入し，「$s+1$ 歩目で逆方向に進み，その後 n 歩以内で再び逆方向に進むことがあれば，$s+1$ 歩目は『引き返していない』とする」という条件を加えている．例えば，$n=5$ としたうえで，図 7.4(b) の軌跡を考える．図 7.4(b) では，4 歩目（青枠部分）で 1 度逆方向に進み，5 歩目（黄枠部分）で再び逆方向に進んでいる．この場合，5 歩以内に再び逆方向へ進んでいるため青枠部分は「引き返していない」となる一方，意図的に逆方向へ進んだと推測できる黄枠部分は「引き返した」と判定される．また，図 7.4(c) に示す軌跡では，4 歩目（青枠部分）で逆方向に進み，そこから 7 歩進んだ後（黄枠部分）に再び逆方向に進んでいるため，青枠・黄枠の両者とも「引き返した」と判定される．

以上が通路における判定方法である．次に，部屋における判定方法を述べる．

- 本研究では，部屋における引き返しについて，図 7.5(a) や図 7.5(b) のように「プレイヤが部屋に進入した後，どの分岐にも進まないまま，進行してきた方向の通路に戻ったとき」のみを引き返したときと見なす．この条件を満たしていない部屋内での行動は，全て「引き返していない」とする．これを踏まえたうえで，部屋における引き返しが発生した地点は次のように判定する．

- まず，通路の場合と同様，プレイヤが s 歩目で向いている方向と $s+1$ 歩目で向いている方向が逆であるならば，$s+1$ 歩目を部屋における「引き返した地点」とする（図 7.5(a) の青枠部分）．これに当てはまる場合，次の条件を調べずにこの地点のみを「引き返した地点」とする．



(a) (b)

図 7.5: 部屋における「引き返した」の例（赤矢印：プレイヤの軌跡）

表 7.1: 引き返しモデルの学習データ数とその内訳

n	学習データ数	「引き返した」の数	「引き返していない」の数
1	589,902	25,724	564,178
5	584,214	25,713	558,501
10	577,104	25,622	551,482

- 先の条件には当てはまらないものの，図 7.5(b) のように引き返す場合もあり得る．この場合，部屋の入口から最も遠かった地点を「引き返した地点」とする（図 7.5(b) の青枠部分）．

この判定方法を用いるうえで，本研究では n =1，5，10 とする 3 種類の学習データをそれぞれ作成した．実験データより得ることができた学習データ数や「引き返した」「引き返していない」の内訳は，表 7.1 に示す通りである．学習に用いる特徴量は,「プレイヤの現在地の座標」「現在進んでいる通路の先に未開封の宝箱が見えるかどうか」「現在進んでいる通路の先に行き止まりが見えるかどうか」など計 26 個である．特徴量の詳細については，付録 B.5 を参照されたい．

モデルの学習及び精度検証を行うために学習データを分割する必要があるが，今回は 158 個のマップを 10 個ずつ，18 人のプレイヤを 3 人ずつに分割する．そのうえで「マップ 148 個 ×15 人分のデータで学習，マップ 10 個 ×3 人分のデータで検証」を 48 セット繰り返していく[1][2]．また，学習データは，7.2.1 節の予測モデルと同様の方法で水増しする．

[1] 7.1 節で述べた通り，18 人のプレイヤを 9 人ずつの 2 グループに分割し，グループごとに 80 個のマップをプレイしているため，$(80/10) \times (9/3) \times 2 = 48$ セットとなる．

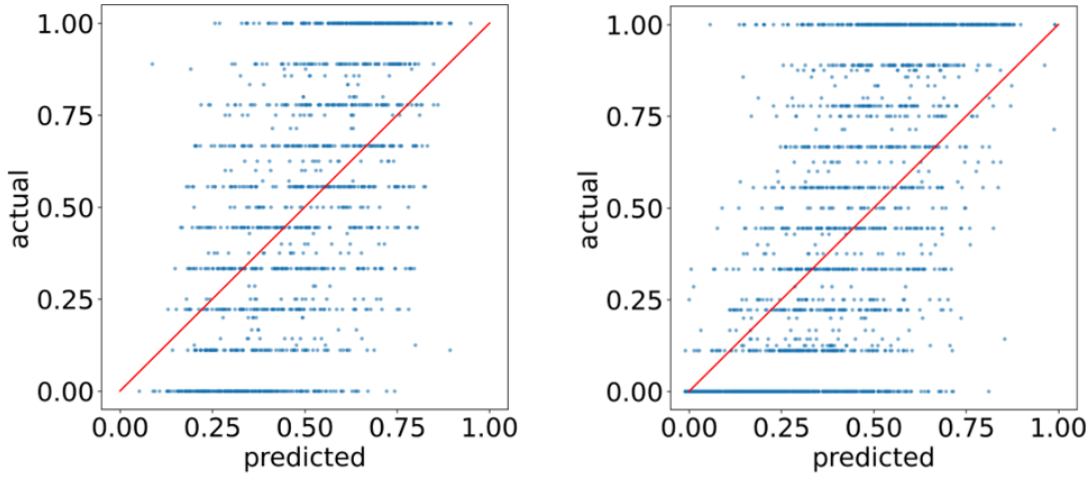
[2] 7.1 節で述べた通り，160 個のマップのうち 158 個のみを用いることとなったため，48 セットのうち 3 セットでは「マップ 150 個 ×15 人分のデータで学習，マップ 8 個 ×3 人分のデータで検証」となっている．

7.3　学習結果

本節では，7.2 節で述べた 3 つの予測モデルの学習結果を与える．

7.3.1　分岐点・分岐部屋における分岐選択確率の学習結果

本項では，ダンジョン中の分岐点を対象とする予測モデルと分岐部屋を対象とする予測モデルの学習結果を与える．図 7.6 は，横軸に学習データを水増しした予測モデルによる予測値を正規化した値，縦軸に実験データから得た実測値をとる散布図である．また，ダンジョン向けの予測モデルでは，アイテムや HP など迷路には存在しない要素に関する特徴量を用いているが，それらの特徴量を用いずに迷路向けの予測モデルと同様の特徴量のみ[3]を用いる学習も試みた．そのように特徴量を制限した場合や，水増し処理を施していない場合を含めた学習結果について，予測値と実測値の RMSE とピアソンの相関係数を表 7.2 にまとめる．対象が分岐点あるいは分岐部屋であることを問わず，予測精度が最も良いのは，ダンジョン向けの特徴量を含みかつ学習データの水増し処理を行った場合であった．相関係数について，分岐点向けモデルは 0.62，分岐部屋向けモデルは 0.67 といずれも正の相関を示しており，予測精度に改善の余地はあるものの，テストプレイヤの作成に活用できる見込みがある結果になったと考える．



(a) 分岐点向け予測モデル　　(b) 分岐部屋向け予測モデル

図 7.6: 学習データの水増し処理を行った予測モデルによる分岐選択確率の予測結果（横軸：正規化後の予測値，縦軸：実測値）

[3]maze_size を除く，22 個の特徴量である．

表 7.2: ダンジョンを対象とする各予測モデルの精度比較

対象	予測モデル	RMSE	相関係数
分岐点	迷路向けの特徴量のみ＋水増しなし	0.30	0.53
	迷路向けの特徴量のみ＋水増しあり	0.30	0.52
	ダンジョン向けの特徴量を含む＋水増しなし	0.28	0.61
	ダンジョン向けの特徴量を含む＋水増しあり	0.28	0.62
分岐部屋	迷路向けの特徴量のみ＋水増しなし	0.35	0.43
	迷路向けの特徴量のみ＋水増しあり	0.35	0.42
	ダンジョン向けの特徴量を含む＋水増しなし	0.29	0.67
	ダンジョン向けの特徴量を含む＋水増しあり	0.29	0.67

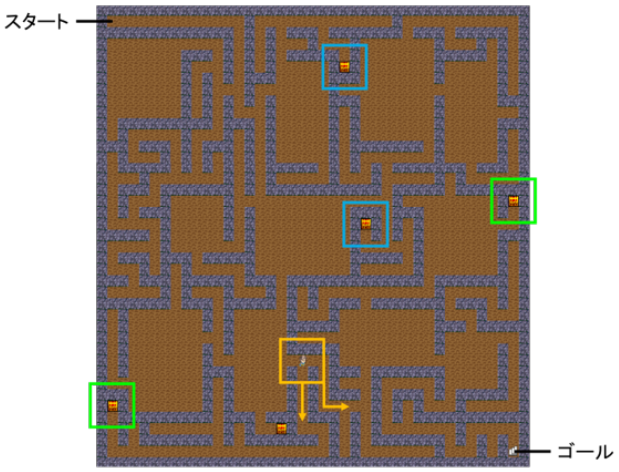
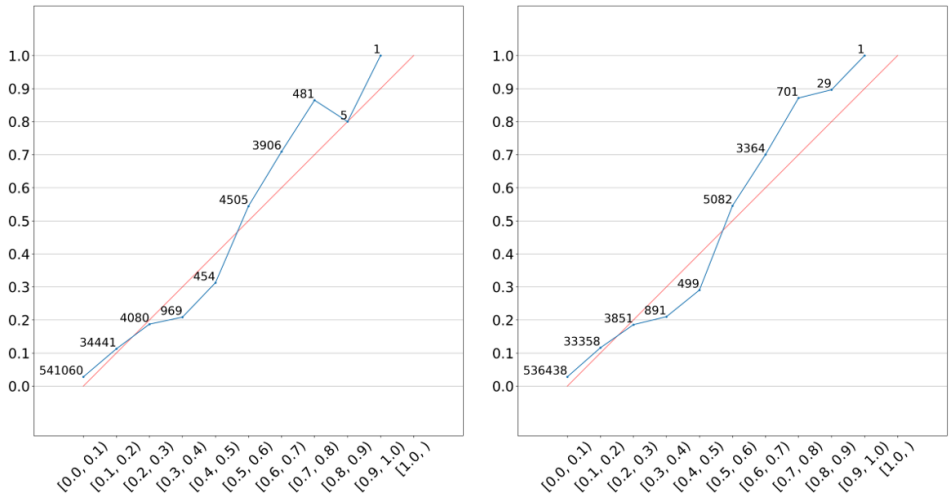


図 7.7: 実測値と予測値に差がある状況の例．プレイヤは左方向から分岐点に進入している（黄枠：予測対象の分岐点，青枠：初めて分岐点に到達する以前に取得可能な宝箱，緑枠：分岐点からさらに進むことで取得可能な宝箱）．

このように，ある程度良い学習結果となった一方，5.3 節で述べた迷路向けの予測モデル（相関係数 0.72）と比較した場合には劣る精度となった．この結果は，単純な迷路と比較すれば，アイテムや HP を導入しただけのダンジョンであっても人間プレイヤの行動はより複雑になり，その予測が困難になることを示唆したものであると考える．図 7.7 中の黄枠の分岐点は，そのように予測が困難であった状況の例である．この分岐点における選択を予測する場合，分岐点到達までにどの程度アイテムを獲得してきたのか，分岐点の先にあるアイテムの存在を把握しているのかといった点が重要になるはずである．これに対して今回構築した予測モデルでは，それらの点を直接表現する特徴量を含めることが困難であり，複数のプレイヤの状態を平均化したものを特徴量としていた．今後はそのように平均化した特徴量を用いるモデルのほかに，プレイヤ個々人の状態に焦点を当てたモデルについても構築する価値があると考える．

7.3.2　引き返すタイミングの学習結果

本項では，プレイヤが来た道を引き返す確率を予測するモデル（n =1，5，10）の学習結果を与える．図 7.8 は，予測値の分布を示すグラフであり，横軸は予測値の区間，縦軸は各区間に属するテストデータのうち実測値が「引き返した」であるデータの割合を表す．図 7.8(a) において，予測値の区間が $[0.7, 0.8)$ である箇所を例に説明すると，この区間に属するテストデータは 481 個存在し，そのうち 9 割弱のテストデータは実測値が「引き返した」であることを示している．そのため，図中の折れ線が赤斜線に沿っているほど良い分布であることを意味する．



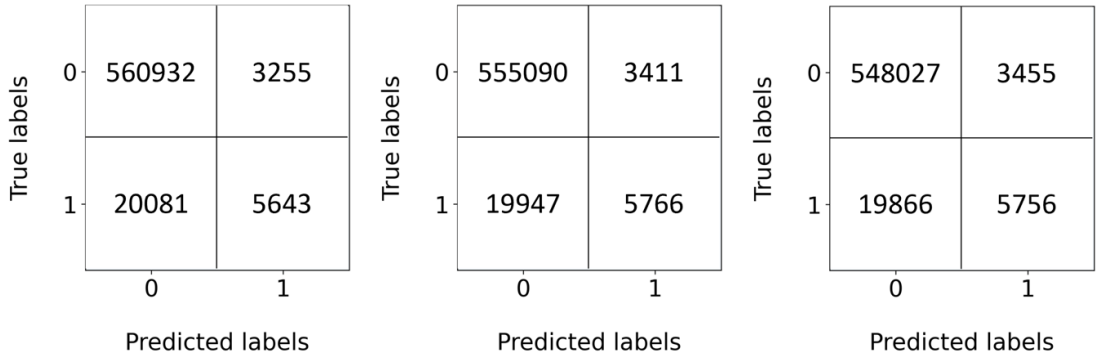
(a) n =1　　(b) n =5

(c) n =10

図 7.8: 引き返しモデルによる予測結果（横軸：予測確率の区間，縦軸：各区間に属するテストデータのうち実測値が「引き返した」であるデータの割合，各マーカーの左上にある数値：各区間に属するテストデータの総数）

図 7.8 に示す 3 つ全てのグラフにおいて，折れ線は赤斜線に沿った形になっていることから，いずれのモデルも引き返す確率をある程度の精度で予測できていることが示されている．また，各グラフを比較すると，n の値を 1，5，10 と変化させたとしても，予測値の分布に顕著な変化は見られない．この結果より，テストプレイヤの作成に向けて，変数 n を用いる引き返しの定義の調整を行った場合でも，モデルの精度について大きな障害が生じることは少ないと考える．

さらに，予測値が 0.5 未満ならば「引き返していない」，0.5 以上ならば「引き返した」とし，予測結果を 2 値に変換した場合の混同行列を図 7.9 に示す．各モデルによる f1 score は，いずれも 0.33 となった．通常の 2 値分類であれば f1 score が 1.0 に近いほど良い精度であることを意味するが，この予測モデルではテストプレイヤの作成に活用できるかどうかが肝になる．言い換えれば，(1) 引き返すタイミングや (2) 引き返す回数の多寡が人間プレイヤに類似するのかといった点が重要になる．本論文では (1) は扱わないが，(2) については次の方法により評価できる．例えば，$n = 1$ の場合，テストデータ数は計 589,902 個，その中でも実測値が「引き返した」であるデータは計 25,724 個である．このとき $25,724/589,902 \approx 0.044$ より，統計的に見れば人間プレイヤは約 23 回の行動のうち 1 回程度引き返していることになる．一方，予測モデルの予測値 p に従い確率的に引き返すとするならば，$\sum_{i=1}^{589,902} p_i/589,902 \approx 0.051$ であったため，約 20 回の予測のうち 1 回程度引き返すこととなり，人間プレイヤに近い値になることがわかる．このような引き返す回数の多寡の違いは，テストプレイヤの人間らしさや生成されるマップの良し悪しに少なからず影響を及ぼすはずであり，この多寡が人間プレイヤに近いことは良い結果である．



(a) n =1（f1 score：0.33）　(b) n =5（f1 score：0.33）　(c) n =10（f1 score：0.33）

図 7.9: 引き返しモデルによる予測結果の混同行列（横軸：予測値，縦軸：実測値，1：引き返した，0：引き返していない）

第8章　おわりに

本研究では，ゲームデザインにおける重要な課題でありながらも，学術研究では扱われることの少ない「プレイヤの自然な誘導」に着目し，プレイヤを自然に誘導する 2D マップの自動生成手法を提案した．研究対象としたマップは「迷路」「ダンジョン」の 2 種類である．提案手法は大まかに，教師あり学習による人間プレイヤの行動予測モデルの構築，予測モデルに基づく人間らしさを考慮したテストプレイヤの作成，テストプレイヤの評価に基づくマップの自動生成で構成される．

1 つ目の対象である迷路は，壁と通路のみで構成される単純なマップである．そのうえで，具体的には，人間プレイヤの分岐選択傾向を考慮しながら難易度を調整した迷路の自動生成を目指した．そのような迷路の自動生成に向けてまず，迷路内の分岐点に着目し，人間プレイヤの分岐選択について傾向調査および予測を行った．予測テストの結果，迷路における人間プレイヤの分岐選択確率は，ある程度の精度で予測できることが示された．次に，人間プレイヤの分岐選択傾向を模倣したテストプレイヤを作成し，そのテストプレイヤによる難易度推定を用いて，迷路の難易度調整および自動生成を行った．生成した迷路には，テストプレイヤにとって難しい迷路群，中程度の迷路群，簡単な迷路群が存在し，それらを人間プレイヤに解いてもらう被験者実験を実施した．ゲーム終了時の歩数などを用いて，人間プレイヤにとっての難易度を評価したところ，テストプレイヤにとっての難易度と人間プレイヤにとっての難易度は概ね一致していることが示された．

2 つ目の対象であるダンジョンは，壁・通路・部屋から構成され，なおかつアイテムや HP の要素を含むマップである．そのうえで，迷路の分岐点における分岐選択確率の予測に加えて，部屋における分岐選択確率の予測，探索中の引き返しが発生するタイミングの予測，これら 3 つを行った．予測テストの結果，迷路に向けたモデルと比較した場合には劣る精度となることが確認できた．この結果は，単純な迷路と比較すれば，アイテムなどの要素が増えただけであっても，人間プレイヤの行動は一層複雑になり，その予測が困難になることを示唆している．ただし，いずれの予測モデルについても，テストプレイヤの作成という観点では活用できる見込みがある結果となっている．

今後の展望としては，深層学習などによる予測部分の改良，人間らしさを考慮しつつ複数種類のマップをテストできる汎用的なテストプレイヤの作成を行う．また，敵や NPC などの要素も含むマップ，あるいは 3D マップなど，より複雑な環境のマップに提案手法を適用した場合の効果検証も行う．

謝辞

本論文は，筆者が北陸先端科学技術大学院大学 先端科学技術研究科 池田研究室に在籍していた期間に行った研究をまとめたものです．本論文をまとめるにあたり，多くのご支援とご指導を賜りました，池田心教授と HSUEH Chu-Hsuan 助教に深く感謝申し上げます．池田心教授には，常に温かく，時に厳しいご指導をいただきました．また，研究内容のみならず，研究者としての姿勢や物事の捉え方など多くのことを学ばせていただきました．これらの経験を糧とし，今後も精進していく所存です．HSUEH Chu-Hsuan 助教には，本研究を進めるにあたり，多くの議論を重ね，その都度的確な助言をいただきました．特に，論文執筆の際に丁寧かつ熱心なご指導をいただいたこと，大変感謝しております．

共に研究やゲームの面白さについて探求した池田研究室のメンバーに心より感謝申し上げます．特に同期の酒見真君，富永裕貴君，南基大君，早下雅弘君，山田直央君とは，研究内容から私生活の些細なことまで多くのことを話し合い，切磋琢磨しながら研究生活を過ごすことができました．皆さんと共に過ごした時間は，研究生活において大いに励みとなりました．

その他，被験者実験など，本研究にご協力いただいた全ての方々に感謝の意を表します．最後に，ここまで温かく見守り支えてくれた家族に感謝します．

発表論文リスト

[1] 藤平啓汰, 池田心. 2D 迷路における行動選択の予測と自然なプレイヤ誘導への応用. 第 46 回ゲーム情報学研究発表会（GI）, Vol. 2021-GI-46, No. 4, pp. 1-6, 2021.

[2] 藤平啓汰, シュエジュウシュエン, 池田心. 人間らしさを考慮したテストプレイヤを用いる迷路の自動生成と難易度評価. 第 26 回ゲームプログラミングワークショップ（GPW）, pp. 192-199, 2021.

[3] Keita Fujihira, Chu-Hsuan Hsueh, and Kokolo Ikeda. Procedural Maze Generation Considering Difficulty from Human Players' Perspectives. In *Proceedings of the Advances in Computer Games (ACG)*, 2021.

参考文献

[1] Ashish Vaswani, et al. Attention Is All You Need. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, pp. 5998–6008, 2017.

[2] Quoc V Le. Building High-Level Features using Large Scale Unsupervised Learning. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8595–8598, 2013.

[3] Murray Campbell, A Joseph Hoane Jr, and Feng-hsiung Hsu. Deep Blue. *Artificial Intelligence*, Vol. 134, No. 1-2, pp. 57–83, 2002.

[4] 保木邦仁. 局面評価の学習を目指した探索結果の最適制御. 第 11 回ゲームプログラ ミングワークショップ（GPW）, pp. 78–83, 2006.

[5] David Silver, et al. Mastering the Game of Go with Deep Neural Networks and Tree Search. *Nature*, Vol. 529, No. 7587, pp. 484–489, 2016.

[6] Volodymyr Mnih, et al. Human-Level Control Through Deep Reinforcement Learning. *Nature*, Vol. 518, No. 7540, pp. 529–533, 2015.

[7] Vanessa Volz, et al. Evolving Mario Levels in the Latent Space of a Deep Convolutional Generative Adversarial Network. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 221–228, 2018.

[8] Luigi Cardamone, Georgios N Yannakakis, Julian Togelius, and Pier Luca Lanzi. Evolving Interesting Maps for a First Person Shooter. In *Proceedings of the European Conference on the Applications of Evolutionary Computation (EvoApplications)*, pp. 63–72, 2011.

[9] Luigi Cardamone, Daniele Loiacono, and Pier Luca Lanzi. Interactive Evolution for the Procedural Generation of Tracks in a High-End Racing Game. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 395–402, 2011.

[10] Adam M Smith, Erik Andersen, Michael Mateas, and Zoran Popović. A Case Study of Expressively Constrainable Level Design Automation Tools

for a Puzzle Game. In *Proceedings of the International Conference on the Foundations of Digital Games (FDG)*, pp. 156–163, 2012.

[11] Julian Togelius, et al. Multiobjective Exploration of the Starcraft Map Space. In *Proceedings of the IEEE Conference on Computational Intelligence and Games (CIG)*, pp. 265–272, 2010.

[12] エニックス. ドラゴンクエスト, 1986.

[13] スクウェア. ファイナルファンタジー, 1987.

[14] Think Labyrinth! url: `https://www.astrolog.org/labyrnth.htm`, (accessed 2022-02-05).

[15] Conceptis Puzzles. *Picture This! Mazes*. Sterling Publishing.

[16] Ralph J. Colao. *Math Maze Puzzle: The Math Alternative to Crossword Puzzle!* Authorhouse.

[17] ナムコ. パックマン, 1980.

[18] Richard Morris. Developments of a Water-Maze Procedure for Studying Spatial Learning in the Rat. *Journal of Neuroscience Methods*, Vol. 11, No. 1, pp. 47–60, 1984.

[19] 山崎文雄, 永田茂, 横山秀史, 大槻明. 避難行動の迷路実験結果. 土木学会論文集, No. 441, pp. 203–206, 1992.

[20] Studies Rules. ダンジョンズ & ドラゴンズ, 1974.

[21] Christopher W Totten. *Architectural Approach to Level Design*. CRC Press, 2019.

[22] Ahmed Khalifa, Fernando de Mesentier Silva, and Julian Togelius. Level Design Patterns in 2D Games. In *Proceedings of the IEEE Conference on Games (CoG)*, pp. 1–8, 2019.

[23] Glenn Joseph Winters and Jichen Zhu. Guiding Players through Structural Composition Patterns in 3D Adventure Games. In *Proceedings of the Foundations of Digital Games (FDG)*, 2014.

[24] 宮田一乗. ゲームとエンタテインメント技術 (2) ゲームと CG - プロシージャル技術. 映像情報メディア学会誌, Vol. 63, No. 8, pp. 1107–1112, 2009.

[25] Noor Shaker, Julian Togelius, and Mark J Nelson. *Procedural Content Generation in Games*. Springer, 2016.

[26] Valve Corporation. Left 4 dead, 2008.

[27] Michael Booth. The AI Systems of Left 4 Dead. In *Proceedings of the Artificial Intelligence and Interactive Digital Entertainment Conference (AIIDE)*, 2009.

[28] 任天堂. スーパーマリオブラザーズ, 1985.

[29] Noor Shaker, Georgios Yannakakis, and Julian Togelius. Towards Automatic Personalized Content Generation for Platform Games. In *Proceedings of the Artificial Intelligence and Interactive Digital Entertainment Conference (AIIDE)*, pp. 63–68, 2010.

[30] Mojang Studios. マインクラフト, 2011.

[31] Georgios N Yannakakis and Julian Togelius. Experience-Driven Procedural Content Generation. *IEEE Transactions on Affective Computing*, Vol. 2, No. 3, pp. 147–161, 2011.

[32] Gavin SP Miller. The Definition and Rendering of Terrain Maps. *ACM SIGGRAPH Computer Graphics*, Vol. 20, No. 4, pp. 39–48, 1986.

[33] Gillian Smith, Jim Whitehead, and Michael Mateas. Tanagra: A Mixed-Initiative Level Design Tool. In *Proceedings of the International Conference on the Foundations of Digital Games (FDG)*, pp. 209–216, 2010.

[34] Ruben Smelik, Tim Tutenel, Klaas Jan De Kraker, and Rafael Bidarra. Integrating Procedural Generation and Manual Editing of Virtual Worlds. In *Proceedings of the Workshop on Procedural Content Generation in Games (PCGames)*, pp. 1–8, 2010.

[35] Julian Togelius, Georgios N Yannakakis, Kenneth O Stanley, and Cameron Browne. Search-Based Procedural Content Generation: A Taxonomy and Survey. *IEEE Transactions on Computational Intelligence and AI in Games*, Vol. 3, No. 3, pp. 172–186, 2011.

[36] Daniele Loiacono, Luigi Cardamone, and Pier Luca Lanzi. Automatic Track Generation for High-End Racing Games using Evolutionary Computation. *IEEE Transactions on Computational Intelligence and AI in Games*, Vol. 3, No. 3, pp. 245–259, 2011.

[37] Lucas Ferreira and Claudio Toledo. A Search-Based Approach for Generating Angry Birds Levels. In *Proceedings of the IEEE Conference on Computational Intelligence and Games (CIG)*, pp. 1–8, 2014.

[38] Danial Hooshyar, Moslem Yousefi, Minhong Wang, and Heuiseok Lim. A Data-Driven Procedural-Content-Generation Approach for Educational Games. *Journal of Computer Assisted Learning*, Vol. 34, No. 6, pp. 731–739, 2018.

[39] Mike Preuss, Antonios Liapis, and Julian Togelius. Searching for Good and Diverse Game Levels. In *Proceedings of the IEEE Conference on Computational Intelligence and Games (CIG)*, pp. 1–8, 2014.

[40] Adam Summerville, et al. Procedural Content Generation via Machine Learning (PCGML). *IEEE Transactions on Games*, Vol. 10, No. 3, pp. 257–270, 2018.

[41] Wizards of the Coast LLC. マジック：ザ・ギャザリング, 1993.

[42] billzorn. url: `https://github.com/billzorn/mtg-rnn`, (accessed 2022-02-05).

[43] Summerville Adam J. and Mateas Michael. Super Mario as a String: Platformer Level Generation Via LSTMs. In *Proceedings of the International Joint Conference of DiGRA and FDG*, 2016.

[44] Tianye Shu, Ziqi Wang, Jialin Liu, and Xin Yao. A Novel CNet-Assisted Evolutionary Level Repairer and Its Applications to Super Mario Bros. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–10, 2020.

[45] id Software. Doom, 1993.

[46] Edoardo Giacomello, Pier Luca Lanzi, and Daniele Loiacono. DOOM Level Generation using Generative Adversarial Networks. In *Proceedings of the IEEE Games, Entertainment, Media Conference (GEM)*, pp. 316–323, 2018.

[47] Edoardo Giacomello, Pier Luca Lanzi, and Daniele Loiacono. Searching the Latent Space of a Generative Adversarial Networks to Generate Doom Levels. In *Proceedings of the IEEE Conference on Games (CoG)*, pp. 1–8, 2019.

[48] Sanggyu Nam, Chu-Hsuan Hsueh, and Kokolo Ikeda. Generation of Game Stages with Quality and Diversity by Reinforcement Learning in Turn-Based RPG. *IEEE Transactions on Games*, 2021, doi:10.1109/TG.2021.3113313.

[49] Ahmed Khalifa, Philip Bontrager, Sam Earle, and Julian Togelius. PCGRL: Procedural Content Generation via Reinforcement Learning. In *Proceedings*

of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, pp. 95–101, 2020.

[50] Algoful. url: `https://algoful.com/Archive/Algorithm/MazeDig`, (accessed 2022-02-05).

[51] Algoful. url: `https://algoful.com/Archive/Algorithm/MazeExtend`, (accessed 2022-02-05).

[52] Algoful. url: `https://algoful.com/Archive/Algorithm/MazeBar`, (accessed 2022-02-05).

[53] Daniel Ashlock, Colin Lee, and Cameron McGuinness. Search-Based Procedural Generation of Maze-Like Levels. *IEEE Transactions on Computational Intelligence and AI in Games*, Vol. 3, No. 3, pp. 260–273, 2011.

[54] Chad Adams and Sushil Louis. Procedural Maze Level Generation with Evolutionary Cellular Automata. In *Proceedings of the IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1–8, 2017.

[55] Joanna Kwiecień. A Swarm-Based Approach to Generate Challenging Mazes. *Entropy*, Vol. 20, No. 10, p. 762, 2018.

[56] Zong-Han Wu, Kevin Lai, Li-An Lin, Ming-Han Huang, and Wen-Kai Tai. Procedurally Generating Game Level with Specified Difficulty. *Proceedings of the IEEE Games, Entertainment, Media Conference (GEM)*, pp. 1–9, 2018.

[57] Michael Scott McClendon. The Complexity and Difficulty of a Maze. In *Bridges: Mathematical Connections in Art, Music, and Science*, pp. 213–222. Citeseer, 2001.

[58] Anthony J Bagnall and Zhanna V Zatuchna. On the Classification of Maze Problems. In *Foundations of Learning Classifier Systems*, pp. 305–316. Springer, 2005.

[59] 吉田喜峰廣, 小泉康一, 大槻正伸. 難しい迷路の条件に関する研究. 第 37 回ゲーム情報学研究発表会（GI）, Vol. 2017-GI-37, No. 6, pp. 1–5, 2017.

[60] 横田和幸, 船瀬新王, 藤原清悦, 内匠逸. ヒトを対象にした迷路課題の難易度を定量的に決定するパラメータの検討. 生体医工学, Vol. 57, No. 2-3, pp. 58–67, 2019.

[61] OpenGenus IQ. url: `https://iq.opengenus.org/space-partitioning-trees/`, (accessed 2022-02-05).

[62] 小野隆啓. url: https://www.kufs.ac.jp/English/faculty/ono/hp-ono4.htm, (accessed 2022-02-05).

[63] RogueBasin. url: http://www.roguebasin.com/index.php/Basic_BSP_Dungeon_generation, (accessed 2022-02-05).

[64] Lawrence Johnson, Georgios N Yannakakis, and Julian Togelius. Cellular Automata for Real-time Generation of Infinite Cave Levels. In *Proceedings of the Workshop on Procedural Content Generation in Games (PCGames)*, pp. 1–4, 2010.

[65] Tommy Thompson and Becky Lavender. A Generative Grammar Approach for Action-adventure Map Generation in the Legend of Zelda. 2017.

[66] Michael Cerny Green, Ahmed Khalifa, Athoug Alsoughayer, Divyesh Surana, Antonios Liapis, and Julian Togelius. Two-step Constructive Approaches for Dungeon Generation. In *Proceedings of the International Conference on the Foundations of Digital Games (FDG)*, pp. 1–7, 2019.

[67] Roland Van Der Linden, Ricardo Lopes, and Rafael Bidarra. Procedural Generation of Dungeons. *IEEE Transactions on Computational Intelligence and AI in Games*, Vol. 6, No. 1, pp. 78–89, 2013.

[68] Breno MF Viana and Selan R dos Santos. Procedural Dungeon Generation: A Survey. *Journal on Interactive Systems*, Vol. 12, No. 1, pp. 83–101, 2021.

[69] Evan C Sheffield and Michael D Shah. Dungeon Digger: Apprenticeship Learning for Procedural Dungeon Building Agents. In *Proceedings of the Symposium on Computer-Human Interaction in Play (CHI PLAY) Companion Extended Abstracts*, pp. 603–610, 2018.

[70] Zisen Zhou and Matthew Guzdial. Toward Co-creative Dungeon Generation via Transfer Learning. In *Proceedings of the International Conference on the Foundations of Digital Games (FDG)*, pp. 1–9, 2021.

[71] Philip Bontrager and Julian Togelius. Learning to Generate Levels From Nothing. In *Proceedings of the IEEE Conference on Games (CoG)*, pp. 1–8, 2021.

[72] Christoffer Holmgård, Antonios Liapis, Julian Togelius, and Georgios N Yannakakis. Evolving Personas for Player Decision Modeling. In *Proceedings of the IEEE Conference on Computational Intelligence and Games (CIG)*, pp. 1–8, 2014.

[73] Antonios Liapis, Christoffer Holmgård, Georgios N Yannakakis, and Julian Togelius. Procedural Personas as Critics for Dungeon Generation. In *Proceedings of the European Conference on the Applications of Evolutionary Computation (EvoApplications)*, pp. 331–343, 2015.

[74] Pedro M. Fernandes, Jonathan Dahl Jørgensen, and N. T. Poldervaart. Adapting Procedural Content Generation to Player Personas Through Evolution. In *Proceedings of the IEEE Symposium Series on Computational Intelligence (SSCI)*, 2021.

[75] 吉田友太. 遺伝的アルゴリズムを用いた 2D シューティングゲームのステージ生成. 修士論文, 北陸先端科学技術大学院大学, 2020.

[76] Jeffrey S Larson, Eric T Bradlow, and Peter S Fader. An Exploratory Look at Supermarket Shopping Paths. *International Journal of Research in Marketing*, Vol. 22, No. 4, pp. 395–414, 2005.

[77] Herb Sorensen, et al. Fundamental Patterns of in-store Shopper Behavior. *Journal of Retailing and Consumer Services*, Vol. 37, pp. 182–194, 2017.

[78] Corinne H Mowrey, Pratik J Parikh, and Kevin R Gue. A Model to Optimize Rack Layout in a Retail Store. *European Journal of Operational Research*, Vol. 271, No. 3, pp. 1100–1112, 2018.

[79] Ram Bezawada, Subramanian Balachander, Pallassana K Kannan, and Venkatesh Shankar. Cross-category Effects of Aisle and Display Placements: a Spatial Modeling Approach and Insights. *Journal of Marketing*, Vol. 73, No. 3, pp. 99–117, 2009.

[80] Azhar Mohd Ibrahim, Ibrahim Venkat, K. G. Subramanian, Ahamad Tajudin Khader, and Philippe De Wilde. Intelligent Evacuation Management Systems: A Review. *ACM Trans. Intell. Syst. Technol.*, Vol. 7, No. 3, pp. 1–27, 2016.

[81] Colby Johanson, Carl Gutwin, and Regan Lee Mandryk. The Effects of Navigation Assistance on Spatial Learning and Performance in a 3D Game. In *Proceedings of the Symposium on Computer-Human Interaction in Play (CHI PLAY)*, pp. 341—-353, 2017.

[82] Guolin Ke, et al. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. *Advances in Neural Information Processing Systems*, Vol. 30, pp. 3146–3154, 2017.

付 録A

A.1　迷路を解く順番

迷路を対象とした学習データ収集のための被験者実験では，20 人の被験者を 2 つのグループに分けた．1 つ目のグループは 11 人，2 つ目のグループは 9 人である．各グループに属する被験者は，表 A.1 に示す設定と順番に従い迷路を解いた．また，19 番目から 21 番目の迷路については，両グループとも制限時間を 90 秒と設定した．

表 A.1: 迷路の設定と解く順番

順番	グループ 1	グループ 2
1	壁伸ばし法-S-Wide	壁伸ばし法-S-Wide
2	壁伸ばし法-M-Wide	壁伸ばし法-S-Narrow
3	壁伸ばし法-L-Wide	穴掘り法-S-Wide
4	穴掘り法-S-Wide	穴掘り法-S-Narrow
5	穴掘り法-M-Wide	棒倒し法-S-Wide
6	穴掘り法-L-Wide	棒倒し法-S-Narrow
7	棒倒し法-S-Wide	壁伸ばし法-M-Wide
8	棒倒し法-M-Wide	壁伸ばし法-M-Narrow
9	棒倒し法-L-Wide	穴掘り法-M-Wide
10	壁伸ばし法-S-Narrow	穴掘り法-M-Narrow
11	壁伸ばし法-M-Narrow	棒倒し法-M-Wide
12	壁伸ばし法-L-Narrow	棒倒し法-M-Narrow
13	穴掘り法-S-Narrow	壁伸ばし法-L-Wide
14	穴掘り法-M-Narrow	壁伸ばし法-L-Narrow
15	穴掘り法-L-Narrow	穴掘り法-L-Wide
16	棒倒し法-S-Narrow	穴掘り法-L-Narrow
17	棒倒し法-M-Narrow	棒倒し法-L-Wide
18	棒倒し法-L-Narrow	棒倒し法-L-Narrow
19	壁伸ばし法-L-Narrow	壁伸ばし法-L-Narrow
20	穴掘り法-L-Narrow	穴掘り法-L-Narrow
21	棒倒し法-L-Narrow	棒倒し法-L-Narrow

A.2　迷路の予測モデルの特徴量

迷路を対象とする教師あり学習に用いた特徴量を以下に記す．方向を表す特徴量は，右なら 0，下なら 1，左なら 2，上なら 3 に変換し，数値化している．

- **maze_size**：迷路のサイズ（S，M，またはL）．S なら 0，M なら 1，L なら 2 に変換し，数値化している．
- **x, y**：分岐点の x，y 座標.
- **ent**：分岐点から見たとき，分岐点へ進入する前にプレイヤがいた通路マスの方向.
- **proceeding direction（proc）**：分岐点から見たとき，予測対象となる分岐の方向.
- **proc_up, proc_down, proc_left, proc_right:** 分岐点から見たとき，ent を除く各方向に未確定な通路があるかどうか．ある方向が True で，その方向を 1 回以上訪れた場合でも，その方向は True のままとする．
- **dist_edge_up, dist_edge_down, dist_edge_left, dist_edge_right**：分岐点と各方向にある外周との直線距離．ただし，dist_edge_up と dist_edge_left は，特徴量 x，y と同じ値をそれぞれ持つためあまり意味のない特徴量である．
- **is_straight**：ent と proc が直進方向にあるかどうか.
- **straight_depth**：is_straight が True の場合，視界範囲に関係なく[1]，proc の方向に進んだときに直進できなくなるまでの通路マスの個数．is_straight が False なら 0 となる．
- **promisingness**：視界範囲に関係なく，proc の方向に進んだときに到達可能な全ての通路マスの個数．例えば，図 A.1 中の B を分岐点とし，下方向の選択確率を予測する場合，緑斜線で塗りつぶされた通路マス（ゴールを含む）の個数が promisingness の値となる．
- **promisingness_sum**：分岐点から見たとき，ent を除く各方向の先に続く通路マスの個数．例えば，図 A.1 中の B を分岐点とし，それに関連する選択確率を予測する場合，緑斜線と青縦線で塗りつぶされた通路マス（ゴールを含む）の個数が promisingness_sum の値となる．
- **num_branchpoints**：視界状態を Narrow と仮定し，視界範囲内にある分岐点の個数．ただし，現在いる分岐点は除く．

[1]Narrow 迷路では，本来プレイヤは視界範囲外の情報を取得できないが，straight_depth と promisingness，promisingness_sum を計算する際には視界範囲外の情報も用いている．

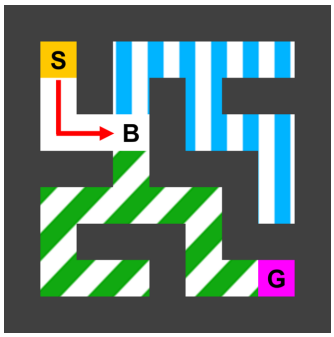


図 A.1: promisingness と promisingness_sum の例

- **dist_start**：スタートから分岐点までの最短歩数.
- **avg_step**：全被験者の分岐点到達時点の歩数の平均値．テストプレイヤが迷路をプレイする場合，分岐点到達時点の合計歩数を avg_step の値とする．迷路や分岐点の情報から作成される他の特徴量とは異なり，本特徴量はプレイヤの歩数をもとに作成される．歩数が多いほど迷路内を探索し，その構造を把握している可能性が高くなる．それが分岐選択に影響するだろうという観点から，本特徴量を用いることとした.
- **cos_goal**：分岐点から見たゴールの方向と proc の方向のなす角度に対する $\cos\theta$．図 A.2 に例を示す．図 A.2 中の B を分岐点，G をゴールとし，proc が右方向なら cos_goal の値は $\cos 30^\circ \approx 0.86$，下方向なら $\cos 300^\circ = 0.50$，左方向なら $\cos 210^\circ \approx -0.86$，上方向なら $\cos 120^\circ = -0.50$ となる.
- **cos_start**：分岐点から見たスタートの方向と proc の方向のなす角度に対する $\cos\theta$.
- **general_direction**：視界状態を Narrow と仮定し，proc の先に続く通路の大まかな方向．この特徴量を計算するうえで，分岐点を根，proc の先に続く視界範囲内の通路マスを節や葉とする迷路の部分木を考える．また，特徴量の候補値を決めるための変数 D_{LR} と D_{UD} を導入し，アルゴリズム 2 を用いて D_{LR} と D_{UD} を計算する．D_{LR} が正数であれば右方向を，負数であれば左方向を候補値とする．また，D_{UD} が正数であれば下方向を，負数であれば上方向を候補値に追加する．D_{LR} と D_{UD} の絶対値同士を比較し，値の大きい変数に由来する候補値を general_direction の値とする.

  例えば，図 A.3 中の B を分岐点とし，下方向の選択確率を予測する場合，$D_{LR} = -1 - 1 = -2$，$D_{UD} = 1 + 1 + 1 + 1 = 4$ となり，$|-2| < |4|$ となるため，general_direction は下方向となる．また，右方向の選択確率を予測する場合，$D_{LR} = 1 + 1 + \frac{1}{2}(1 + 1) = 3$，$D_{UD} = \frac{1}{2}(1 + 1 - 1 - 1) = 0$ となり，$|3| > |0|$ となるため，general_direction は右方向となる.

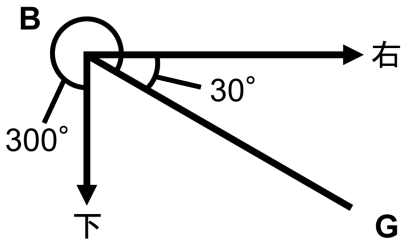


図 A.2: cos_goal の例

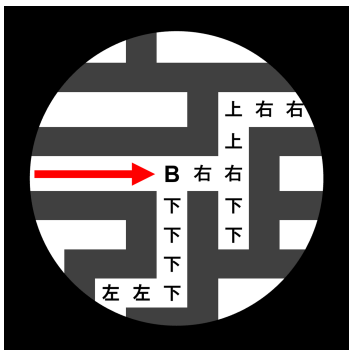


図 A.3: general_direction の例

アルゴリズム 2 D_{LR} と D_{UD} の計算

$D_{UD} \leftarrow 0$
$D_{LR} \leftarrow 0$
$S \leftarrow \{\}$　// 空のスタック
$v \leftarrow$ ルートノード // 分岐点
$r \leftarrow 1$
$S.push((v, r))$
while S が空ではない **do**
　$(node, r) \leftarrow S.pop()$
　for all $c \leftarrow node$ の子ノード **do**
　　$d \leftarrow$ 迷路上で $node$ から c を見たときの方向
　　if $d =$ 上 **then**
　　　$D_{UD} \leftarrow D_{UD} - 1 \times r$
　　else if $d =$ 下 **then**
　　　$D_{UD} \leftarrow D_{UD} + 1 \times r$
　　else if $d =$ 左 **then**
　　　$D_{LR} \leftarrow D_{LR} - 1 \times r$
　　else if $d =$ 右 **then**
　　　$D_{LR} \leftarrow D_{LR} + 1 \times r$
　　end if
　　if $c =$ 十字路または丁字路の中心地点 **then**
　　　$S.push((c, r/2))$
　　else
　　　$S.push((c, r))$
　　end if
　end for
end while

付 録B

B.1 不正解路の広さと通路の直進率の調節に着目したダンジョン生成アルゴリズム

6.3 節の実験後，被験者からマップに関する以下のような意見が挙げられた．

- 分岐点がなく，曲がり角が多いだけの通路はイライラする（図 B.1）．
- ある分岐を深い地点まで進んだ後，ゴールの直前で不正解だと判明する場合がある（図 B.2）．広い分岐を引き返すのは面倒であり，ゲームの面白さを損ねていると感じた．また，分岐を往復しただけで制限時間が迫ってしまい，ゲームクリアが困難になってしまっていることもあった．

これらの意見は納得できるものであり，解決すべき課題であると考える．これに対して本研究では，通路の直進率と不正解路の広さの調節に着目しつつダンジョンを自動生成するアルゴリズムを提案し，7.1 節の被験者実験で用いた．

提案手法では，部屋・正解路・不正解路の順にマップを構築する．部屋の作成は BSP[63]，正解路と不正解路の作成は穴掘り法 [50] をベースとしている．また，提案手法では，直進率に関するパラメータとして $\alpha_{\mathrm{str}} \in [1.0,)$，不正解路に関するパラメータとして $\gamma_{\mathrm{ext}} \in (0.0, 1.0]$，$\gamma_{\mathrm{pos}} \in (0.0, 1.0]$，$L_{\mathrm{min}}$，$L_{\mathrm{max}}$ を用いる．具体的なマップの構築手順を以下に記す．

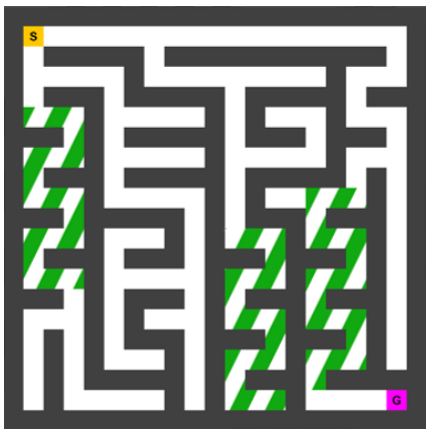


図 B.1: 曲がり角の多い通路

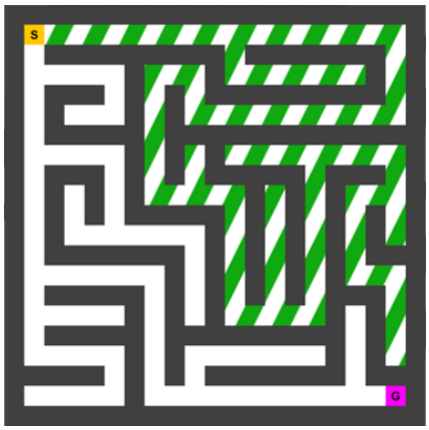


図 B.2: 広い不正解路

- **部屋の作成**

  (1) 奇数 × 奇数となるサイズのマップを生成する．

  (2) BSP を用いてマップ内に部屋を配置する（図 B.3）．このとき，マップ左上の頂点座標を $(1, 1)$ とすると，部屋の左上頂点の x，y 座標はともに偶数とする．また，部屋のサイズは奇数 × 奇数とする．

- **正解路の作成**

  正解路の作成は，直進率に関するパラメータ α_{str} を用いて以下の手順で行う．

  (1) スタートとゴールの座標を定める．

  (2) スタートを開始地点とし，穴掘り法と同じ要領で 2 マス分だけ通路を掘り進める（図 B.4）．掘り進める際の進行方向は，進行可能な方向の中から等確率で選ぶ．

  (3) 再び 2 マス分だけ通路を掘り進める．(2) と同様，進行方向は確率的に選ぶ．ただし，直進方向に進む選択肢があるならば，その方向の選択確率は他の候補の α_{str} 倍とする．例えば，図 B.5 の場合，右と下が選択肢となり，かつ右が直進方向となる．そのため，例えば $\alpha_{\mathrm{str}} = 2$ とすると，右の選択確率は約 0.66，下の選択確率は約 0.33 となる．掘り進めた後，ゴールへ到達していない，かつ現在地からどの方向にも進行できない場合には (4) へ進む．ゴールへ到達した場合，正解路の作成を終了する．それ以外の場合には (3) を繰り返す．

  (4) 現在地を除き，作成してきた正解路の中で最もゴールに近いマスへ移動し，それ以降の通路マスは全て壁マスに変更する（図 B.6(a)）．移動先の地点において，その地点からどの方向にも進行できない場合には（4）

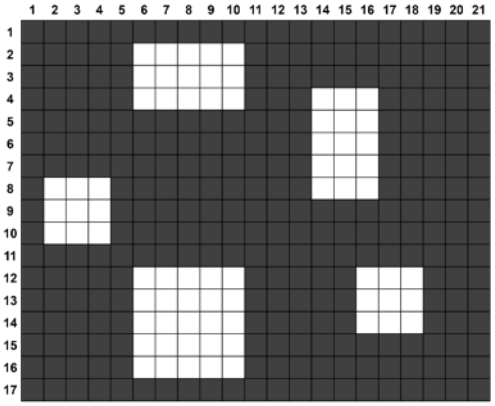


図 B.3: 部屋の配置が終了した状態

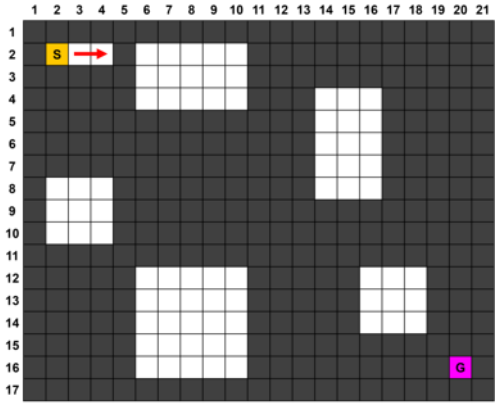


図 B.4: 正解路の作成 (2)

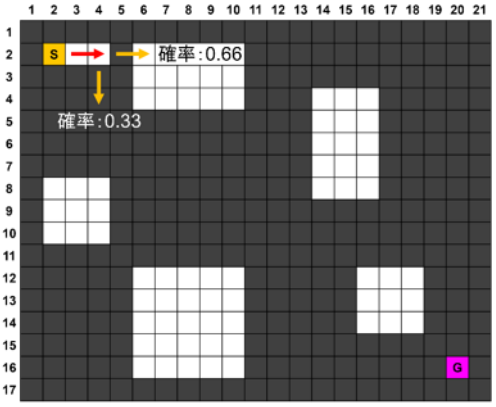
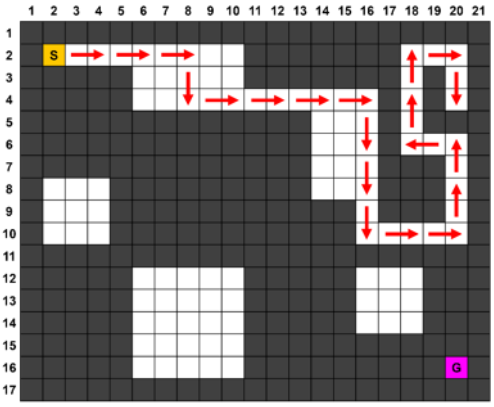
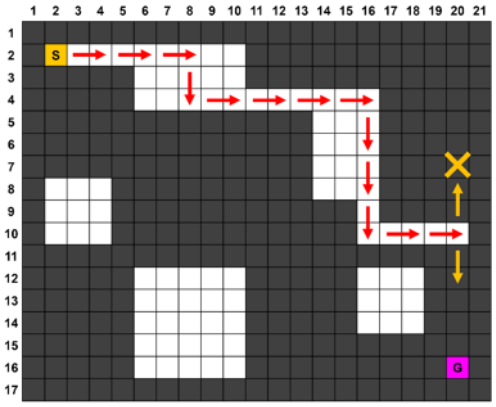


図 B.5: 正解路の作成 (3)



(a)



(b)

図 B.6: 正解路の作成 (4)

を繰り返す．それ以外の場合には（3）へ戻るが，(3）において進行方向を決める際，一度選んだ方向は選択肢から除外しておく（図 B.6(b)）．

提案手法における特徴手的な手順の 1 つがステップ（3）であり，これにより直進率を調整しながら通路を作成することが可能となる．また，ステップ（4）によって，正解路が完成しないことを防ぐこともできる．

ステップ（4）中の「最もゴールに近いマスへ移動する」条件により，単調な正解路が生じてしまう可能性もあるが，付録 B.2 で述べる検証実験の結果を踏まえると，提案手法の欠点にはならないと考える．例えば，最も単調な正解路の長さは，31×31 マスのマップの場合に 57 マス，91×91 マスのマップの場合には 177 マスとなる．表 B.1 の実験結果では，提案手法によって作成した正解路の長さは約 85 マス（31×31），約 350 マス（91×91）であり，最も単調な場合と比較するとそれぞれ約 1.49 倍，約 1.98 倍の長さとなっている．そのため，正解路長という観点で見れば，極端に単調な正解路が生じる可能性は低い．

- **不正解路の作成**

  不正解路の作成には，直進率に関する α_{str} と不正解路の広さに関する γ_{ext}，γ_{pos}，$L_{\min}$，$L_{\max}$ の 5 つのパラメータを用いる．ある不正解路について，γ_{ext} と γ_{pos} はその開始地点を定めるときの選択確率の減衰を，$L_{\min}$ と $L_{\max}$ はその広さを制御するために用いる．不正解路の作成手順は以下の通りである．

  (1) スタートとゴールを除く，正解路を構成する通路マスを対象とし，x，y 座標がともに偶数である地点を不正解路の開始地点候補として抽出する（図 B.7 の赤色マス）．

  (2) 各候補 c_i について，c_i の選択確率の算出に用いる変数 $p_i = 1.0$ をそれぞれ割り当てた後，組 $(c_i, p_i, L_{\min}, L_{\max})$ を作成する．さらに，作成した全ての組から成る集合 C を作成する．

  (3) C の要素数が 0 の場合，不正解路の作成を終了する．要素数が 1 以上の場合，C から 1 つの組 $(c', p', l'_{\min}, l'_{\max})$ を選ぶ．C の要素数を n とすると，各組の選択確率は次式によって算出される．また，選ばれた組は C から除外する．

  $$f(p_i) = \frac{p_i}{\sum_{k=1}^{n} p_k} \tag{B.1}$$

  (4) c' から掘り進めることができる通路マスの上限 $l \in [l'_{\min}, l'_{\max}] \cap \{\text{偶数}\}$ を定める．また，変数 $r = 1.0$ を用意する．この r は，ステップ (6) において，新たな開始地点候補に対する選択確率の減衰率を保存するために用いる．

  (5) 穴掘り法と同じ要領で，c' から 2 マス分だけ通路を掘り進める．正解路の作成時と同様，進行方向は α_{str} を用いて確率的に選ぶ．どの方向にも進行できない場合には (3) へ戻る．掘り進めることができた場合には (6) へ進む．

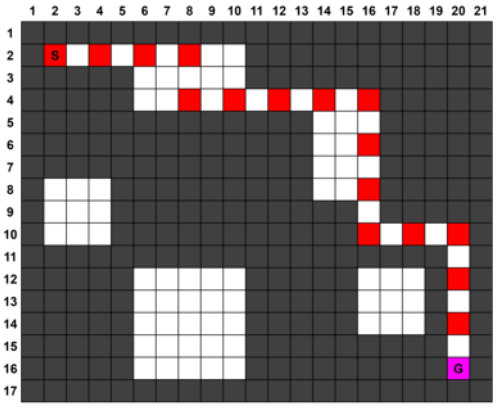


図 B.7: 不正解路の作成 (1)

(6) l を 2 減じ，$l = 0$ ならば (3) へ戻る．それ以外の場合には，c' に現在地 c を代入する．このとき，c の x，y 座標がともに偶数ならば，組 $(c, p' \times \gamma_{\mathrm{ext}} \times r, l'_{\mathrm{min}}/2, l'_{\mathrm{max}}/2)$ を C に追加し，r に $r \times \gamma_{\mathrm{pos}}$ を代入する．その後 (5) へ戻る．

これらの中でも，提案手法における特徴手的な手順となるのがステップ（4）と（3）（6）である．まず（4）において，l は掘り進めることができる通路マスの上限を表している．つまり，パラメータ L_{min} と L_{max} が大きくなるにつれて，不正解路の広さも大きくなることを意味する．

ある通路を一旦掘り終えると，新たな地点から再び通路を掘り始めることとなる．その開始地点を選択するのがステップ（3）であり，各開始地点候補の選択確率の算出に用いる値を定めるのがステップ（6）となる．(6) において，γ_{ext} は「既存の不正解路のさらなる拡大」を，γ_{pos} は「さらに拡大する場合の開始地点の位置」を制御しようとするパラメータである．例えば，$l = 0$ となり，ある不正解路の作成が一旦終了した後の状況を考える（図 B.8）．このとき，赤色マスの選択確率に対して，青色マスの選択確率は γ_{ext} 倍となる．つまり，γ_{ext} の値が小さくなるにつれて，既存の不正解路内のマスが開始地点となる確率，ひいては不正解路が拡大する確率は低下する．

さらに，γ_{pos} を用いることにより，図 B.9 に示すように，同じ不正解路内に存在する青色マス同士の選択確率を変化させることができる．具体的には，γ_{pos} の値が小さいほど不正解路の根に近い地点が選ばれやすく，葉に近い地点が選ばれにくくなるように選択確率を減衰させることができる．

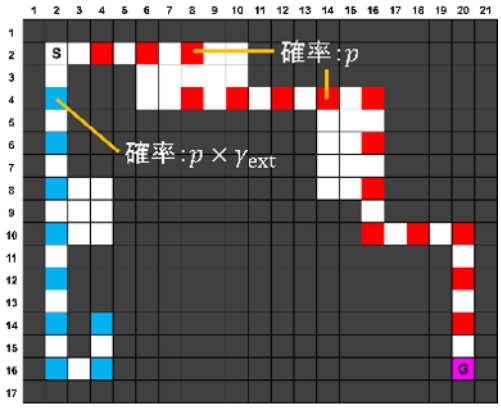


図 B.8: 選択確率とパラメータ γ_{ext}

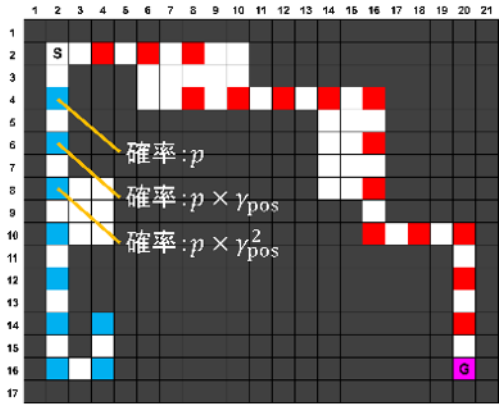


図 B.9: 選択確率とパラメータ γ_{pos}

B.2　ダンジョン生成アルゴリズムの検証実験

付録 B.1 で述べた提案手法の挙動を確かめるため，いくつかの実験を行った．本実験では，スタート地点を左上隅に設定したマップを 1000 個生成したうえで，それらの（1）正解路長,（2）正解路の直進率,（3）不正解路の直進率,（4）マップ中の全通路マスに対して部屋を構成する通路マスが占める割合（部屋の面積率）,（5）分岐点の数,（6）分岐部屋の数,（7）分岐点ではない交差点の数,（8）各不正解路を構成する通路マスの数の平均値（平均分岐広さ）,（9）最も広い不正解路を構成する通路マスの数（最大分岐広さ）,（10）分岐広さの標準偏差の 10 項目の平均値を確認した．パラメータ α_{str} に関する実験結果を表 B.1，L_{min} と L_{max} に関する実験結果を表 B.2 に示す[1][2]．実験結果は概ね期待通りであり，α_{str} を大きくすることで直進率を，また L_{min} と L_{max} によって不正解路の広さをある程度調節できることが示されている．

[1]パラメータについて，表 B.1 の実験では $l \in [20, 30]$，表 B.2 の実験では $\alpha_{\mathrm{str}} = 2$ で固定した．また，いずれの実験においても，$\gamma_{\mathrm{ext}} = 0.5$，$\gamma_{\mathrm{pos}} = 0.9$，BSP の深さは 5 で固定した．

[2]L_{min} と L_{max} は正解路に影響しないパラメータである．そのため，表 B.2 では正解路長と正解路の直進率を記載しない．また，部屋の面積率は α_{str} や L_{min}，L_{max} に影響されない項目であるため，$\alpha_{\mathrm{str}} = 2$ かつ $l \in [20, 30]$ の場合の結果を表 B.1 にのみ記載する．

表 B.1: パラメータ α_{str} の検証結果

項目	α_{str}	マップサイズ			
		31×31	41×41	61×61	91×91
正解路長	1	89.5	130.4	222.6	381.9
	2	85.5	123.1	212.0	356.1
	3	83.9	117.9	203.5	337.8
	5	80.6	115.7	185.9	310.7
正解路の直進率（%）	1	50.8	49.8	46.5	42.1
	2	60.6	59.8	57.8	56.1
	3	65.9	65.2	64.5	63.7
	5	72.0	71.3	72.2	72.8
不正解路の直進率（%）	1	46.7	44.4	41.4	40.4
	2	51.3	49.3	46.9	45.9
	3	53.5	52.0	49.7	48.9
	5	56.6	54.6	53.0	52.2
部屋の面積率（%）	2	53.5	51.8	43.0	28.0
分岐点の数 （括弧内：正解路内のみの場合）	1	2.3（1.9）	4.5（3.2）	13.7（7.9）	45.3（19.4）
	2	2.5（2.0）	5.0（3.5）	14.9（8.5）	48.7（20.4）
	3	2.6（2.0）	5.3（3.6）	15.8（8.8）	50.7（20.8）
	5	2.9（2.3）	5.6（3.8）	17.1（9.1）	53.8（21.9）
分岐部屋の数 （括弧内：正解路内のみの場合）	1	2.9（2.2）	5.5（3.6）	11.2（5.5）	16.7（5.7）
	2	2.9（2.1）	5.4（3.4）	11.0（5.2）	16.2（5.5）
	3	2.8（2.0）	5.3（3.3）	10.9（5.0）	16.3（5.4）
	5	2.8（1.9）	5.3（3.2）	10.8（4.8）	16.3（5.2）
分岐点ではない交差点の数	1	10.4	20.4	64.4	210.9
	2	10.3	20.3	63.7	209.4
	3	10.5	20.4	63.4	206.1
	5	10.2	20.0	63.1	202.9
平均分岐広さ	1	19.9	25.3	37.2	57.0
	2	21.3	26.7	38.8	59.0
	3	22.1	28.0	39.9	61.2
	5	24.0	29.4	43.3	64.4
最大分岐広さ	1	65.5	102.0	216.7	517.9
	2	65.6	102.0	221.8	523.1
	3	68.4	104.3	222.1	523.1
	5	70.1	107.3	228.6	518.0
分岐広さの標準偏差	1	19.8	28.4	51.2	98.7
	2	20.2	28.9	52.2	100.5
	3	21.0	29.6	52.8	101.4
	5	21.9	30.8	55.1	102.2

表 B.2: パラメータ $L_{\min}$ と $L_{\max}$ の検証結果

項目	$l \in [L_{\min}, L_{\max}]$	マップサイズ			
		31×31	41×41	61×61	91×91
不正解路の直進率（%）	$l \in [10, 20]$	51.0	48.5	46.6	45.8
	$l \in [20, 30]$	51.3	49.3	46.9	45.9
	$l \in [30, 40]$	52.1	49.8	47.1	45.8
	$l \in [40, 50]$	52.3	49.9	47.4	46.0
	$l \in [\infty, \infty]$	52.6	50.8	48.3	46.8
分岐点の数 （括弧内：正解路内のみの場合）	$l \in [10, 20]$	2.9（2.5）	5.6（4.4）	16.4（10.7）	50.7（24.7）
	$l \in [20, 30]$	2.5（2.0）	5.0（3.5）	14.9（8.5）	48.7（20.4）
	$l \in [30, 40]$	2.3（1.8）	4.6（3.1）	14.2（7.3）	46.6（17.6）
	$l \in [40, 50]$	2.2（1.7）	4.3（2.8）	13.5（6.7）	44.5（15.9）
	$l \in [\infty, \infty]$	1.9（1.5）	3.6（2.4）	10.0（5.3）	31.2（11.9）
分岐部屋の数 （括弧内：正解路内のみの場合）	$l \in [10, 20]$	2.9（2.3）	5.5（3.6）	11.3（5.4）	16.8（5.7）
	$l \in [20, 30]$	2.9（2.1）	5.4（3.4）	11.0（5.2）	16.2（5.5）
	$l \in [30, 40]$	2.7（2.0）	5.3（3.3）	10.5（5.0）	15.9（5.3）
	$l \in [40, 50]$	2.7（1.9）	5.1（3.1）	10.3（4.7）	15.5（5.2）
	$l \in [\infty, \infty]$	2.4（1.8）	4.5（2.8）	9.1（4.3）	13.3（4.7）
分岐点ではない交差点の数	$l \in [10, 20]$	12.7	24.6	72.4	228.1
	$l \in [20, 30]$	10.3	20.3	63.7	209.4
	$l \in [30, 40]$	8.8	17.5	56.3	190.5
	$l \in [40, 50]$	8.3	15.9	51.3	177.1
	$l \in [\infty, \infty]$	7.4	13.3	35.9	107.3
平均分岐広さ	$l \in [10, 20]$	17.7	22.6	32.8	49.7
	$l \in [20, 30]$	21.3	26.7	38.8	59.0
	$l \in [30, 40]$	22.9	28.7	42.2	64.0
	$l \in [40, 50]$	24.1	30.6	45.0	67.8
	$l \in [\infty, \infty]$	25.7	33.1	50.9	79.4
最大分岐広さ	$l \in [10, 20]$	55.9	91.5	201.8	491.2
	$l \in [20, 30]$	65.6	102.0	221.8	523.1
	$l \in [30, 40]$	76.1	115.8	237.5	548.8
	$l \in [40, 50]$	85.6	129.9	260.2	575.1
	$l \in [\infty, \infty]$	97.7	178.7	404.3	949.4
分岐広さの標準偏差	$l \in [10, 20]$	16.1	24.1	44.5	88.4
	$l \in [20, 30]$	20.2	28.9	52.2	100.5
	$l \in [30, 40]$	23.9	33.3	58.6	109.8
	$l \in [40, 50]$	26.9	37.6	65.1	118.1
	$l \in [\infty, \infty]$	30.9	49.9	94.8	182.8

B.3　ダンジョンの予測モデルの特徴量（分岐点）

ダンジョン中の分岐点における分岐選択確率の教師あり学習では，付録 A.2 で述べたうち，maze_size 以外の 22 個の特徴量を用いた．また，それらに加えて，新たな 8 個の特徴量も用いた．その 8 個の特徴量を以下に記す．

- **avg_exploring_rate**：「ダンジョンの通路マスの総数に対する，プレイヤが分岐点到達時までに通過した通路マス数の割合」の全被験者の平均値．図 B.10 は，スタートとゴールを含めて 53 個の通路マスが存在するダンジョンの例である．図中の B を分岐点とし，あるプレイヤが赤矢印に沿って移動しながら分岐点 B へ到達したとき，緑斜線で塗りつぶされた通路マスとスタート，分岐点 B を含む 22 マスが「通過したマス」となる．
- **avg_hp_rate**：「最大 HP に対する，分岐点到達時の HP の割合」の全被験者の平均値．
- **avg_item_rate**：「ダンジョン中の宝箱の総数に対する，分岐点到達時までに開封した宝箱の数の割合」の全被験者の平均値．
- **avg_gold**：全被験者の分岐点到達時のゴールドの平均値．
- **num_room**：視界範囲内にある部屋の数．視界範囲内で中心部が見える部屋のみカウントの対象であり，かつそれが分岐部屋であるかどうかは問わない．
- **num_item**：視界範囲内にある宝箱の数．各宝箱が未開封であるかどうかは問わない．[3]
- **can_see_item_in_general_direction**：視界範囲内で，general_direction が示す方向に宝箱が見えるかどうか．宝箱が未開封であるかどうかは問わない．

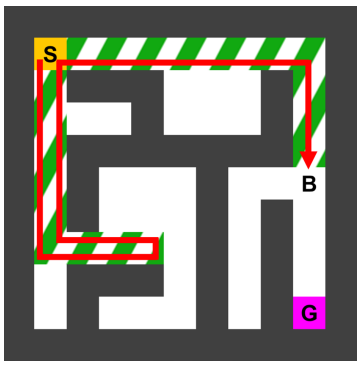


図 B.10: avg_exploring_rate における通過したマスの例

[3]未開封であるかどうかを問わない理由は，目的変数が被験者の分岐選択割合であり，かつ分岐点到達時までにどの宝箱を開封したのかは被験者によって異なるからである．

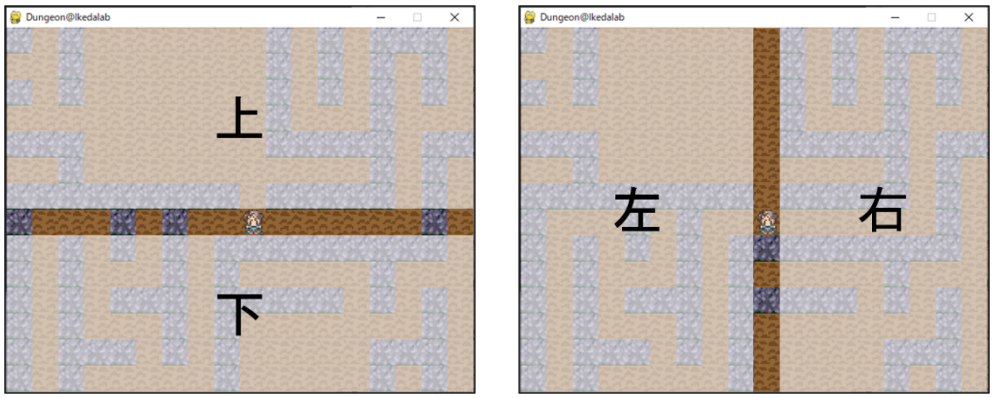


図 B.11: can_see_item_in_general_direction における各方向の領域

また，視界範囲内において，どの領域が上下左右のそれぞれを指すのかは図 B.11 に示す通りである．

- **is_item_in_proc**：視界範囲内で，proc の方向の通路に未開封の宝箱があるかどうか．[4]

B.4　ダンジョンの予測モデルの特徴量（分岐部屋）

ダンジョン中の分岐部屋における分岐選択確率の教師あり学習では，計 35 個の特徴量を用いた．そのうちの 30 個は，付録 B.3 で述べた特徴量とおおよそ同じものである．ただし，いくつかの特徴量については，分岐部屋に向けたモデルに用いるため内容の調整を施している．調整を施した特徴量の詳細を以下に記す．

- **x, y**：部屋の入口の x，y 座標．例えば，図 B.12 に示す部屋では，赤矢印に沿って部屋へ進入した場合，プレイヤの位置する場所が入口となる．また，予測対象となる各通路に対して，黄枠で囲った地点をそれぞれの通路へ向かう際の出口とする．
- **ent**：部屋に進入する前にプレイヤがいた通路マスの方向．
- **proceeding direction（proc）**：予測対象となる通路が接続されている方向．
- **proc_up, proc_down, proc_left, proc_right:** 部屋の入口から見たとき，ent を除く各方向に未確定な通路が接続されているかどうか．
- **dist_edge_up, dist_edge_down, dist_edge_left, dist_edge_right**：部屋の入口と各方向にある外周との直線距離．

[4] 5.2 節で述べたように「プレイヤが分岐点を初めて訪れたときのデータのみ」を学習データとしているため，is_item_in_proc については，未開封の宝箱と限定することができる．

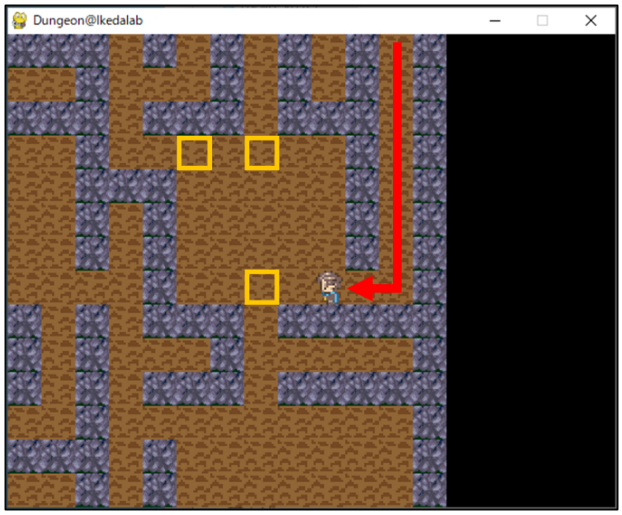


図 B.12: 部屋における入口と出口の例

- **promisingness**：視界範囲に関係なく，予測対象の通路に向かう出口から，proc の方向に進んだときに到達可能な全ての通路マスの個数.
- **promisingness_sum**：各出口の先に続く通路マスの個数.
- **num_branchpoints**：入口から見たとき，視界範囲内にある分岐点の個数.
- **dist_start**：スタートから入口までの最短歩数.
- **avg_step**：全被験者の入口到達時点の歩数の平均値.
- **cos_goal**：予測対象の通路に向かう出口から見たゴールの方向と proc の方向のなす角度に対する $\cos\theta$.
- **cos_start**：予測対象の通路に向かう出口から見たスタートの方向と proc の方向のなす角度に対する $\cos\theta$.
- **avg_exploring_rate**：「ダンジョンの通路マスの総数に対する，プレイヤが入口到達時までに通過した通路マス数の割合」の全被験者の平均値.
- **avg_hp_rate**：「最大 HP に対する，入口到達時の HP の割合」の全被験者の平均値.
- **avg_item_rate**：「ダンジョン中の宝箱の総数に対する，入口到達時までに開封した宝箱の数の割合」の全被験者の平均値.
- **avg_gold**：全被験者の入口到達時のゴールドの平均値.

- **num_room**：入口から見たとき，視界範囲内にある部屋の数．視界範囲内で中心部が見える部屋のみカウントの対象であり，かつそれが分岐部屋であるかどうかは問わない．

- **num_item**：入口から見たとき，視界範囲内にある宝箱の数．各宝箱が未開封であるかどうかは問わない．

- **can_see_item_in_general_direction**：視界範囲内で，予測対象の通路に向かう出口から見たとき，general_direction が示す方向に宝箱が見えるかどうか．宝箱が未開封であるかどうかは問わない．

- **is_item_in_proc**：視界範囲内で，予測対象の通路に向かう出口から見たとき，その通路に未開封の宝箱があるかどうか．

以下は，35 個の特徴量のうち，分岐部屋に向けた予測モデルのために新しく追加した 5 個である

- **room_width, room_height**：分岐部屋の横幅，縦幅．

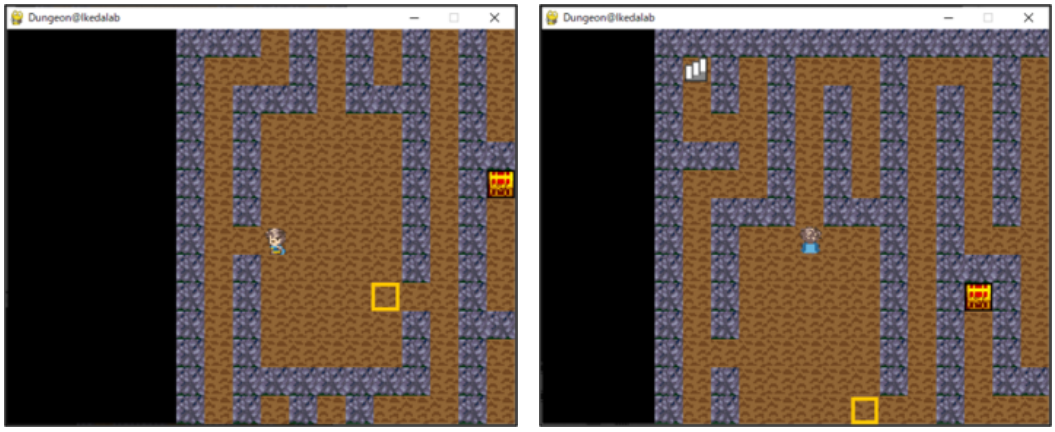
- **is_uncertain**：予測対象の通路に向かう出口から見たとき，その通路が未確定であるかどうか．例えば，図 B.13 に示す部屋では，黄枠の場所が入口，プレイヤの位置する場所が各出口であるとする．図 B.13(a) でプレイヤが向かっている通路であれば is_uncertain の値は True，図 B.13(b) で向かっている通路であれば False となる．

- **dist_between_ent_and_proc**：入口 $(x_e,\, y_e)$ から予測対象の通路に向かう出口 $(x_p,\, y_p)$ までの最短距離 $|x_p - x_e| + |y_p - y_e|$．

- **straight_rate**：dist_between_ent_and_proc において，直進方向に進む歩数の割合．図 B.14(a) に示す分岐部屋では，プレイヤは下方向へ進みながら部屋へ進入したため，直進方向は下となる．このとき，1 番の分岐について，dist_between_ent_and_proc の値は 6 であり，そのうちの 4 歩分は下に向かうものであるため，straight_rate の値は $4/6 \approx 0.67$ となる．2 番の分岐の場合，straight_rate の値は $0/4 = 0$ となる．また，図 B.14(b) に示すように，dist_between_ent_and_proc の値が 0 となる分岐についても，straight_rate の値は 0 となる．

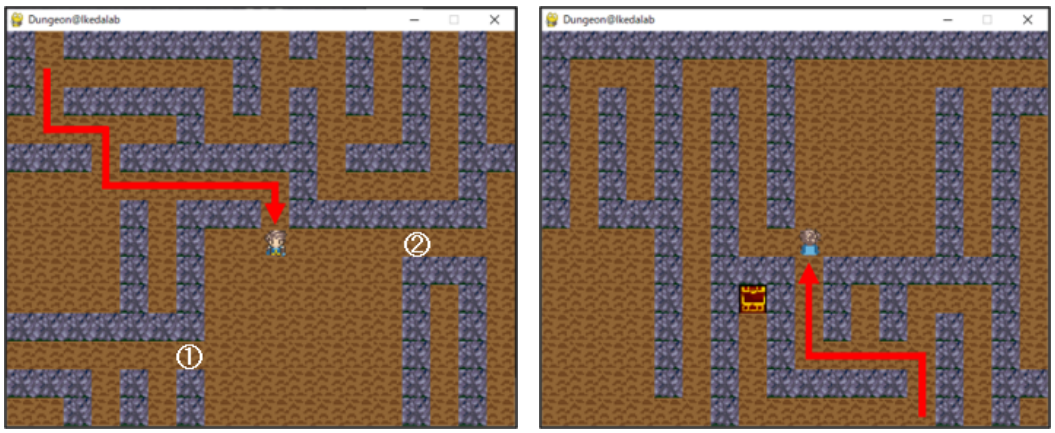
B.5　ダンジョンの予測モデルの特徴量（引き返し）

プレイヤが来た道を引き返す確率の教師あり学習では，計 26 個の特徴量を用いた．また，そのうちのいくつかの特徴量は，付録 B.4 や B.4 で述べたものと内容が類似する．そのような特徴量の詳細を以下に記す．



(a) True である分岐　　(b) False である分岐

図 B.13: is_uncertain の例



(a)　　(b)

図 B.14: straight_rate の例

- **x, y**：プレイヤの現在地の x, y 座標.
- **proceeding direction（proc）**：現在地に進入したとき，プレイヤが向いている方向.
- **dist_edge_up, dist_edge_down, dist_edge_left, dist_edge_right**：プレイヤの現在地と各方向にある外周との直線距離.
- **promisingness**：視界範囲に関係なく，proc の方向に進んだときに到達可能な全ての通路マスの個数.
- **num_branchpoints**：視界範囲内にある分岐点の個数.

- **num_room**：視界範囲内にある部屋の数．視界範囲内で中心部が見える部屋のみカウントの対象であり，かつそれが分岐部屋であるかどうかは問わない．
- **dist_start**：スタートからプレイヤの現在地までの最短歩数．
- **cos_goal**：現在地から見たゴールの方向と proc の方向のなす角度に対する $\cos\theta$．
- **cos_start**：現在地から見たスタートの方向と proc の方向のなす角度に対する $\cos\theta$．
- **general_direction**：，proc の先に続く通路の視界範囲内での大まかな方向．
- **exploring_rate**：ダンジョンの通路マスの総数に対する，プレイヤが現在までに通過した通路マス数の割合．
- **hp_rate**：最大 HP に対する，現在の HP の割合．
- **item_rate**：ダンジョン中の宝箱の総数に対する，現在までに開封した宝箱の数の割合．
- **gold**：現在のゴールド．
- **can_see_unopened_item_in_general_direction**：general_direction が示す方向に，視界範囲内で未開封の宝箱が見えるかどうか．
- **is_uncertain**：proc の方向に続く通路が未確定であるかどうか．
- **is_unopened_item_in_proc**：proc の方向へ続く通路に，視界範囲内で未開封の宝箱があるかどうか．

以下は，このほかに用いた 5 個の特徴量である．

- **num_unopened_items**：視界範囲内にある未開封の宝箱の数．
- **dist_branchpoint**：最後に通過した分岐点，あるいは最後に通過した分岐部屋の出口からプレイヤの現在地までの最短距離．
- **can_see_goal_in_general_direction**：general_direction が示す方向に，視界範囲内でゴールが見えるかどうか．
- **last_viewd_item_x, last_viewd_item_y**：最後に見た未開封の宝箱の x, y 座標．