Measuring Strategic Depth in Games Using Hierarchical Knowledge Bases

Daan Apeldoorn
Technische Universität Dortmund
Dortmund, Germany
Email: daan.apeldoorn@tu-dortmund.de

Vanessa Volz
Technische Universität Dortmund
Dortmund, Germany
Email: vanessa.volz@tu-dortmund.de

***Abstract*—This paper presents a measure intended to quantify the relative strategic depth of games as experienced by human players. The measure is based on the complexity (number and specificity of rules) of a hierarchical knowledge base that is extracted from playtraces. As a proof-of-concept, we compute the proposed measure for three arcade-style games and compare the results to the strategic depth reportedly perceived by human players in a survey. We find that our measure is practicable and able to capture the differences in strategic depth very accurately.**

I. Introduction

Many approaches from the field of computational intelligence in games, e. g. (procedural) content generation [1] and automatic game balancing [2], require a measure for desirable qualities of games. These measures are then used to guide the search or optimisation process towards solutions (e. g. levels or parametrisations) that are in accordance with the intention of the game designer.

One of these qualities that game designer and players often look for in games is (strategic) depth or elegant design after Blizzard's principle: "easy to learn, hard to master"[1]. The concept of depth is often intuitively understood by players, but difficult to capture formally. Besides strategic depth, there is also narrative depth which we do not be address in this paper. A definition for strategic depth frequently used in popular culture is from the YouTube channel Extra Credits[2]:

Definition 1 (Strategic Depth) *[Strategic] depth is the number of emergent experientially different possibilities or meaningful choices that come out of one ruleset.*

Quantifying the strategic depth of a given game can help game designers understand what skills are requested from their players and which audience the game targets. Additionally, elegant games according to the principles of *shibui* "combine outer simplicity with inner depth" [3], i. e. they have a relatively small ruleset of a low complexity while still allowing multiple winning strategies. Go and Hex are among the most popular examples for games with an elegant design.

Therefore, strategic depth and measuring it is clearly an important issue in game design. In this paper, we propose a model using hierarchical knowledge base extraction based on playtraces from human players. In contrast to previous work on strategic depth (discussed in section II-A), with our approach, we are able to specifically take the player experience into account. Additionally, our approach does not necessitate any previous knowledge of a game and is thus generally applicable.

We provide a proof-of-concept for our measure by applying it to a set of selected games and comparing the measured strategic depth with the depth experienced by the players. The results are based on a small survey we conducted to provide anecdotal evidence of the practicability and validity of our approach. The survey will of course be extended in the future.

In the following section II-A, we first introduce some existing approaches to the quantification of strategic depth. Section II-B covers the basic concepts of hierarchical knowledge bases and their extraction, which the proposed measure relies on. Afterwards we give an explicit formal definition of our approach to measure strategic depth in section III as well as an explanation of the reasoning behind it. Based on the definition, in section IV, we analyse the presented measure using games from the GVGAI[3] framework (section IV-A) and describe the survey we conducted to collect playtraces from human players (sections IV-B and IV-C). We compare and discuss the obtained measurements with previously existing measures as well as validate the results using feedback from human players on the perceived strategic depth in section V with very promising results. We summarise the paper and give an overview of future work in section VI.

II. Related Work

A. Measures for Strategic Depth

As was shown in [4], it is possible to generate heuristics based on strategies developed with different methods (some based on AI players and some on expert knowledge). In the paper, the authors suggest that comparing the resulting strategies and perfect play could be used as a way to quantify the depth of a game. While the method we propose in this paper is also based on generated heuristics for playing a game, instead of expressing depth by some form of distance measure to perfect play, we analyse the realised (i.e. experienced) complexity of game play. Measuring strategic depth was not

[1] http://www.wolfsheadonline.com/bushnells-theorem-easy-to-learn-difficult-to-master/#8db01

[2] https://www.youtube.com/watch?v=jVL4st0blGU

[3] https://gvgai.net

investigated further in [4], however, since the focus is on generating understandable heuristics for novice human players.

Instead, a recent paper [5] takes up this idea and proposes to measure what is intuitively understood as depth using a modification of the concept of skill chains (cf. [6] or earlier complexity numbers). The basic idea is to identify the best solution strength (e. g. score in a game) that can be achieved with a specific amount of computational resources. Given a step threshold, the number of improvements in solution strength with increasing computational resources can be measured. The authors propose to use this number as a measure for depth. The reasoning is that only games that allow a high number of intermediary, non-perfect strategies result in a high depth. Games that are either very easy to master (i. e., play perfectly) or very difficult to learn (first solution already requires lots of computational resources) thus would have a small depth.

However, as the authors also acknowledge, their proposal is to be understood as a theoretical concept and difficult to apply in practice. First of all, determining the absolute best result for a given amount of computational resources is very difficult. On top of that, finding the perfect strategy is only possible for simple games.

In [3], a depth score for board games is proposed along with numerous similar properties such as simplicity, clarity and efficiency in order to create elegant games according to the *shibui* design principle. The author measures potential depth or game tree complexity based on the number of legal finishing states that can be reached from the initial state of a game. The computed depth score is then a ratio between the strategic depth (game tree complexity) and state-space complexity (all possible states) of a game.

Again, this approach is not practicable for many non-trivial games since the required measures are difficult to compute. However, there are heuristics to approximate them instead. Besides computational issues, basing a measure on potential depth can be misleading when some strategies in a game are never played and thus, resulting finishing states are not reached in practice. With our approach, we intend to measure the realised depth as experienced by players instead.

B. Hierarchical Knowledge Bases and Their Extraction

1) Hierarchical Knowledge Bases (HKBs): According to [7], an HKB is defined in the context of an agent and consists of rules which are organised on different levels of abstraction. Each rule consists of a premise and a conclusion. The premise is a conjunction over a subset of state values which are perceived by the agent through its sensors at a given time. The conclusion is an action from the agent's action space. The rules can be applied following defined priorities to guide an agent through the context-forming task. HKBs represent knowledge in form of rule-based heuristics instead of a *structured plan* that can be executed procedurally. More formally, in [7], two different types of states and two different types of rules are distinguished:

Definition 2 (Complete States/Partial States) *A complete state is a conjunction $s := s_1 \wedge ... \wedge s_n$ of all values s_i currently perceived by an agent's sensors, where n is the number of sensors (and every perceived sensor value $s_i \in \mathbb{S}_i$ of the corresponding sensor value set $\mathbb{S}_i$ is assumed to be a fact in the agent's current state). A partial state is a conjunction $s := \bigwedge_{s' \in S} s'$ of a subset $S \subset \{s_1, ..., s_n\}$ of the sensor values of a complete state.*

Definition 3 (Complete Rules/Generalised Rules) *Complete rules and generalised rules are of the form $p_\rho \Rightarrow a_\rho \, [w_\rho]$, where p_ρ is either a complete state (in case of a complete rule) or a partial state (in case of a generalised rule), the conclusion $a_\rho \in \mathbb{A}$ is an action from an agent's action space $\mathbb{A}$ and $w_\rho \in [0, 1]$ is the rule's weight.*[4]

Thus, according to Definition 3, complete rules map complete states to actions and generalised rules map partial states to actions. The weights are used to decide between rules from the same level that fire at the same time (rules with higher weights take precedence in execution) and are computed as part of the knowledge extraction described in section II-B2.

Following [7], a HKB is now defined as follows:

Definition 4 (Hierarchical Knowledge Base) *A Hierarchical Knowledge Base (HKB) is an ordered set $\mathcal{KB} := \{R_1, ..., R_{n+1}\}$ of $n + 1$ rule sets, where n is the number of sensors (i. e., the number of state space dimensions). Every set $R_{i<n+1}$ contains generalised rules and the set R_{n+1} contains complete rules, such that every premise $p_\rho = \bigwedge_{s \in S_\rho} s$ of a rule $\rho \in R_i$ is of length $|S_\rho| = i - 1$.*

According to Definition 4, set R_1 contains the most general rules (with empty premises) and the set R_{n+1} contains the most specific (i. e., the complete) rules. Every level $R_{i>1}$ of an HKB contains the exceptions of the level R_{i-1}. Rules in lower abstraction levels are applied prior to more general ones.[5]

Example 1 We consider an agent which is equipped with $n = 2$ sensors to determine its position (x and y coordinates) in a small two-dimensional grid world. The agent has to navigate around a wall from a starting point A to a target point B using the four cardinal directions. The agent's sensors have the value sets $\mathbb{S}_x = \{x_0, ..., x_7\}$ and $\mathbb{S}_y = \{y_0, ..., y_5\}$, and the agent's action space is $\mathbb{A} = \{\mathrm{North}, \mathrm{South}, \mathrm{East}, \mathrm{West}\}$ (see Figure 1 (a)). The HKB shown in Figure 1 (b) represents the agent's knowledge needed to navigate around the wall: The agent knows that in general, it should decide to go to the east (see level R_1). An exception to this rule is made in case the x-sensor value $s_x \in \mathbb{S}_x$ is equal to x_0 or to x_7 (see level R_2). According to R_3, if the agent is located at (x_0, y_5), it will go east. The weights in this example do not affect the policy since no rules of the same level fire at the same time in this case.

If the underlying environment is of a more complex nature, more rules are required on levels $R_{i>1}$ to properly reflect the knowledge needed to act in the new environment successfully.

[4]Note that in [8], complete rules are called *elementary rules*.

[5]Note that the order of the sensors does not influence the rule sets that are created.

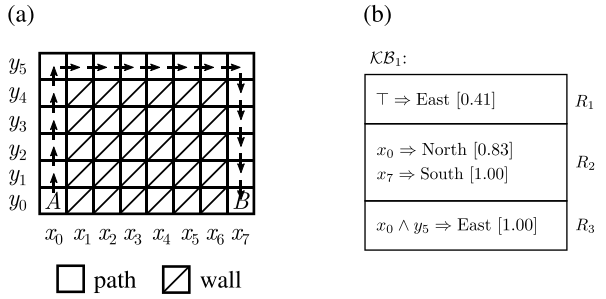


Fig. 1. Grid World Navigation Example

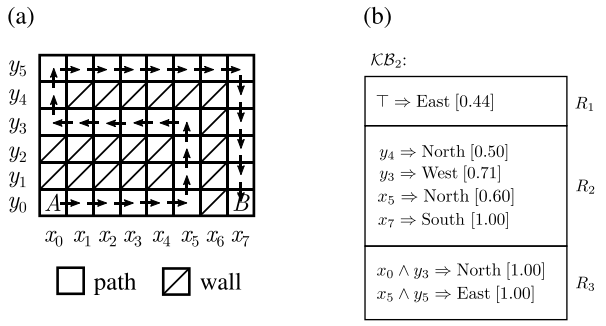


Fig. 2. Grid World Navigation Example with Different Structure

Example 2 We consider again the agent from Example 1, this time in a grid world of the same format but with different obstacles (see Figure 2): It can be seen that the HKB $\mathcal{KB}_2$ comprises more rules on the lower abstraction levels $R_{i>1}$ than HKB $\mathcal{KB}_1$ from figure 1, since more exceptions are required to navigate from starting point A to the destination point B in this case. In this example, the weights do influence the decision. For instance, if the agent is located at (x_7, y_4), it will go to the south rather than to the north. This is because the rule corresponding to the former action (last rule in R_2) is weighted higher than the rule for the latter (first rule in R_2).

Note that not all levels of an HKB have to contain rules: Depending on the underlying structure of the environment, it is possible that the knowledge needed to act successfully in the environment can be expressed without any rules on some levels $R_{i>1}$.

2) Extraction of Hierarchical Knowledge Bases: After having introduced the concept of HKBs in the previous section, this section now briefly outlines how HKBs can be created. In [8], an algorithm is described which extracts an HKB from an $(n+1)$-dimensional matrix of weights for state-action pairs, where n is the number of state space dimensions, i. e., the number of the agent's sensors). The additional dimension corresponds to the action space. For knowledge extraction in the experiments for this paper, we use a variant of the matrix-based version of the algorithm from [8] (a faster version of the algorithm is provided in [7]). The main steps of the extraction algorithm (see algorithm 1) will be briefly outlined in the following (for details the reader should refer to the original literature).

Knowledge base extraction is intended to represent the stored knowledge in a compact form. Therefore, the algorithm removes rules that are not relevant to the policy resulting from a knowledge base. First, all rules worse (according to the weight) than others on the same level of abstraction R_j are removed (lines 11-16). Afterwards (line 18-23) those rules on all levels $R_{j>1}$ which are more specific, but worse than a corresponding more general rule on a level $R_{j'<j}$ are deleted. The algorithm then removes those rules on all levels $R_{j>1}$ which are redundant (same action) of a more general rule on a level $R_{j'<j}$ (lines 25-32).

The extraction algorithm starts with an input set of state-action pairs $\mathcal{SA} = \{(\mathbf{s}_1, a_1), ..., (\mathbf{s}_m, a_m)\}$. Every $\mathbf{s}_1, ..., \mathbf{s}_m$ is a vector of the agent's state space, i. e., $\mathbf{s}_k \in \mathbb{S}_1 \times ... \times \mathbb{S}_n$ where every $\mathbb{S}_1, ..., \mathbb{S}_n$ is a value set of one of the agent's sensors.The output of the algorithm is an HKB $\mathcal{KB}^{\mathcal{SA}} := \{R \in \mathcal{KB}^{\hat{Q}} | R \neq \emptyset\}$ containing the knowledge represented in $\mathcal{SA}$ in a compact form, where every level $R_{j>1}$ can be considered to contain the exceptions of the level R_{j-1}.[6]

```
01 % Create (sparse) weight matrix
02 Q̂ := (q̂_{s_1,...,s_n,a})
03 for each (s, a) ∈ SA do
04     q̂_{s_1,...,s_n,a} := 1   % where (s_1, ..., s_n) = s
05 end for
06
07 % Create initial HKB (the weight of
08 % a rule is calculated from Q̂ as the
09 % rel. frequency of it firing)
10 KB^Q̂ := {R_1, ..., R_{n+1}}
11
12 % Removal of worse rules
13 for each rule ρ on each level R ∈ KB^Q̂ do
14     if ∃σ ∈ R: p_σ = p_ρ, w_σ > w_ρ then
15         R := R \ {ρ}
16     end if
17 end for
18
19 % Removal of worse more specific rules
20 for each rule ρ on each level R_{j>1} ∈ KB^Q̂ do
21     if ∃σ ∈ R_{j'<j}: S_σ ⊂ S_ρ, w_σ ≥ w_ρ then
22         R := R \ {ρ}
23     end if
24 end for
25
26 % Removal of too specific rules
27 for each rule ρ on each level R_{j>1} ∈ KB^Q̂ do
28     if ∃σ ∈ R_{j'<j}: a_σ = a_ρ, S_σ ⊂ S_ρ and
29         (∄τ ∈ R_{j-1}: a_ρ ≠ a_τ, S_τ ⊂ S_ρ or
30          ∃υ ∈ R_{j-1}: a_υ = a_ρ, S_υ ⊂ S_ρ, w_υ > w_τ) then
31         R := R \ {ρ}
32     end if
33 end for
34
35 % Remove unused rules according to SA
36 filter( KB^Q̂ )
```

Algorithm 1: HKB Extraction

[6] A step-by-step example of the knowledge base extraction algorithm can be found in [7].

III. Measure for Strategic Depth

We define a measure for strategic depth based on the complexity of a HKB extracted from a set of state-action pairs. The measure is the sum of the relative complexity per level in the HKB weighted exponentially by the depth of the level. Thus, a game is considered more complex if it has more rules (i. e. more exceptions), especially on deeper levels (i. e. more sophisticated exceptions). This approach is a way to measure the difficulty of searching for a successful strategy to solve a game (cf. [5], where the difficulty is estimated using a limited computational budget). The assumption is that more general heuristics (i e., rules) on the upper levels are identified even by novice players, while advanced players will spend some time to define nuanced exceptions to these heuristics. Of course, a maximally deep game according to this measure (i. e., a game having an exception rule for every single state) would not be entertaining or even playable for a human player, since then, there would exist no observable pattern or sense in the game.

More formally, we define our measure for strategic depth based on HKBs as follows:

Definition 5 (Strategic Depth Measure d_s) *Let $\mathcal{KB} = \{R_1, ..., R_{n+1}\}$ be a HKB and $\mathfrak{S} = \{\mathbb{S}_1, ..., \mathbb{S}_n\}$ be the set of all sensor value sets of an agent, then the strategic depth is defined as a function*

$$d_s(\mathcal{KB}, \mathfrak{S}) := \sum_{i=1}^{n+1} \binom{n}{i-1} b^{i-1} \frac{|R_i|}{\sum\limits_{\substack{\mathbf{S}\subseteq\mathfrak{S} \\ |\mathbf{S}|=i-1}} \prod\limits_{\mathbb{S}\in\mathbf{S}} |\mathbb{S}|} , \qquad (1)$$

where $n+1 = |\mathcal{KB}| = |\mathfrak{S}| + 1$ is the number of levels in $\mathcal{KB}$, b is a weighting constant, and $R_i \in \mathcal{KB}$ is the i-th level of $\mathcal{KB}$.

In equation (1), the fraction on the right measures the relative complexity of a level by dividing the realised by the potential complexity. More precisely, $|R_i|$ is the number of rules contained in the HKB on level i. This number is divided by the number of rules that would be possible at level i, which is the same as all possible partial states (premises) on that level (since all rules have unique premises after HKB extraction step 3, see section II-B2). The number of partial states is computed by iterating through all possible sensor combinations of $i-1$ sensors ($\mathbf{S} \subseteq \mathfrak{S}$) and then finding the number of all possible sensor value sets ($\mathbb{S} \in \mathbf{S}$) combinations (i. e. the product of the magnitude of the respective sensor value sets $\prod\limits_{\mathbb{S}\in\mathbf{S}} |\mathbb{S}|$).

The left part of the first sum in equation (1) are the weights for the level depth. They are intended to increase in the same order of magnitude as the potential complexity of the level. The number of sensor combinations on level i is computed by the binomial coefficient $\binom{n}{i-1}$, i. e. the possible ways of drawing $i-1$ sensor from the n available sensors. This approach takes the potential complexity resulting from the dimensionality of the state space n into account. This is because we assume that n (i. e., the number of observable sprites with their state space dimensions), does affect the perception of depth for human players. This is in line with the description of strategic depth above (cf. sections I and II-A), in that depth can be "bought" by extending the ruleset.

TABLE I
Weights and number of partial states according to definition 5 for games with $n = 2$, $b = 2$

Level	Weight	# Partial States
1	$\binom{2}{0} 2^0 = 1$	1
2	$\binom{2}{1} 2^1 = 4$	$\lvert\mathbb{S}_x\rvert + \lvert\mathbb{S}_y\rvert = 6 + 8 = 14$
3	$\binom{2}{2} 2^2 = 4$	$\lvert\mathbb{S}_{x\times y}\rvert = \lvert\mathbb{S}_x\rvert \cdot \lvert\mathbb{S}_y\rvert = 6 \cdot 8 = 48$

Similarly, b^{i-1} increases in the same order of magnitude as the number of possible sensor value set combinations. For the experiments in this paper, we use $b = 2$. The parameter b could be increased in cases where the sensor value sets are very large. However, we conjecture that the amount of possible sensor values plays a comparatively smaller role than the dimensionality n. To understand our reasoning, imagine the grid worlds in the examples 1 and 2 in section II-B1 would be extended by one column/row in each dimension. The resulting game would not become much more complex. Rather, general heuristics would work for more states as before. In contrast, adding a third dimension to the problem transforms the task to 3D navigation which is considerably more complex.

The proposed weights therefore grow with the depth of the level, but not as quickly as the partial state space. In general, our measure is intended to be sensitive to the size of the state space (as long as it is not normalised), since a game with a larger state space usually leaves more room for strategic depth. Still, if one would want to measure the elegance of game design instead, the measure d_s could be divided by $\sum_{i=1}^{n+1} \binom{n}{i-1} b^{i-1}$. This would measure the "intrinsic" depth of the game, i. e. the amount of realised depth in comparison to the potential depth introduced through rule complexity (or in our case, number of dimensions). This could be especially useful when doing multi-objective optimisation to maximise "intrinsic" depth and at the same time minimise complexity.

Example 3 As an example, the weights and partial state space for the relevant levels from the examples in section II-B1 with $n = 2$ are displayed in table I. For the examples we therefore obtain:

$$d_s(\mathcal{KB}_1, \{\mathbb{S}_x, \mathbb{S}_y\}) = 1 \cdot \frac{1}{1} + 4 \cdot \frac{2}{14} + 4 \cdot \frac{1}{48} = \frac{139}{84} \approx 1.65$$
$$d_s(\mathcal{KB}_2, \{\mathbb{S}_x, \mathbb{S}_y\}) = 1 \cdot \frac{1}{1} + 4 \cdot \frac{4}{14} + 4 \cdot \frac{2}{48} = \frac{97}{42} \approx 2.31$$

The measure attributes a larger strategic depth to the second example and thus behaves as intended for the grid world examples. In the following, we validate the measure on more realistic examples.

IV. Experimental Setup

To evaluate the proposed measure d_s, we selected video games from the general video game playing competition framework (GVGAI [9]) to be presented to 8 players. The selected games are described in (section IV-A). We then conducted a survey to collect two types of data: (1) feedback on the perceived strategic depths and (2) playtraces from the respective players. Details on the survey and the post-processing of the obtained data can be found in (section IV-B) and IV-C, respectively.

A. Game Selection

We have selected two levels each from three different games for our survey to evaluate the proposed measure of strategic depth.[7] The games are taken from the GVGAI competition [9] were partly modified to fit our purpose. The selected games and levels are shown in Figure 3.

In the game *Camel Race* (see Figure 3 (a) and (b)), the player controls the yellow camel which must reach the goal on the right before the others.[8] In the game *Run* (see Figure 3 (c) and (d)), the player controls the girl who must reach the cave's exit before being reached by the flood. The key is needed for unlocking the doors blocking the passage. In the game *Eighth Passenger* (see Figure 3 (e) and (f)), the player controls the elf which must reach the goal. The locked passages can be opened by reaching the dark green button.[9] The player loses the game on collision with the green ogre. The red and blue passages can exclusively be traversed by the ogre and the player, respectively. While the red passage is used by the ogre or while the blue passage is used by the player, the ogre is invisible to the player.

The games were mainly selected and modified regarding the following two criteria:

- a manageable state space that is easy to represent,
- the difference in intuitively perceived strategic depth.

We chose mechanically similar games on purpose in order to have similar basic rule sets of a game. However, *Run* adds more rule complexity by introducing locked passages and a corresponding key. For *Eighth Passenger*, the rule complexity is even higher since passages that are only traversable by one type of avatar are implemented on top of that. However, the number of meaningful choices (cf. definition 1) naturally also increases through the increase in rules.

Still, intuitively, *Camel Race* seems to have only minimal strategic depth since it only requires us to follow the obviously optimal path. While this path is a bit more obscure in level 2, it is still very apparent that the game was created as a usecase where lengthy exploration by game AIs is punished. For *Run*, a good solution for the shortest path is still easily recognisable for human players, but the game seems to be a little deeper since the player has to plan further ahead. This is because the win-state has two conditions and the locked passage needs to be opened first.

In contrast to both of the previously described games, for *Eighth Passenger*, even a good path to the button and then to the goal is not immediately obvious. The gameplay also does not focus on pathfinding, but rather includes reactive planning in order to avoid collision with the ogre. The game even supports multiple strategies based on the amount of usage of the exclusive passages to hide from the opponent.

Note that, although, the selected games are obviously all 2D games, they are of higher dimensionality in terms of their state space (cf. table II). Thus, in principle, also 3D games could have been used here in a similar way instead.

Since the games of the GVGAI competition are mainly designed to be played by artificial agents, some of the levels run rather fast and are therefore hard to be played by humans. To overcome this, all games have been throttled slightly for the survey. This also shifts the focus more to the strategy aspect, and away from a player's reactivity.

In order to apply the measure according to definition 5, the games need to be formalised first so that they can be represented as an HKB (cf. section II-B1). Therefore sensors with corresponding sensor value sets need to be defined that completely express the state of the game at any given time (cf. definition 2). The state encodes the knowledge that the agent has in a given situation which is the basis for the decision of a specific action as a response.

In the GVGAI competition, AI agents usually have perfect information in the game, i. e. the existence and movement of all sprites in the game is observable. This is true for almost all games except where imperfect information is part of the game mechanic, e. g., the exclusive passages in *Eighth Passenger* (which make the ogre invisible). It is important to note that despite these observations, the AI in the competition has incomplete information since it has no knowledge on the structure of the game or the intention of other players/NPC opponents. For the sake of consistency and also to preserve the opportunity to scale up our experiments, we chose to adhere to this decision. Therefore, the sensors provided for the agent are the same an AI in the competition would have. The action space is also always the same in the GVGAI framework to facilitate comparisons. The resulting state-action spaces for the games described above are listed in table II. For *Eighth Passenger*, the previous position of the opponent NPC is recorded in addition to its position at the time of measurement. This is done in order to provide information for strategic decisions depending on the movement direction of the NPC and in case its position is not observable due to the usage of the exclusive passages.

B. Survey

For our survey, we selected participants with a self-declared affinity for games so that we could safely assume that they would have an understanding of the meaning of strategic depth in games. We had only eight participants (one psychologist,

[7] Note that the levels are treated as separate games here.

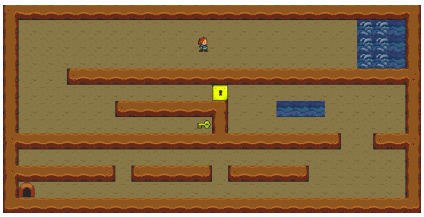
[8] In the original version of the game, there were more than three camels which have been removed for our survey.

[9] In the original version of the game, it was possible for both the elf and the ogre to pass through closed doors. This has been fixed for our survey.

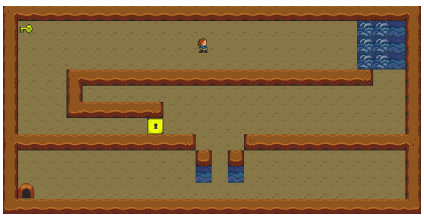
(a) Camel Race (Level 0)

(b) Camel Race (Level 2)

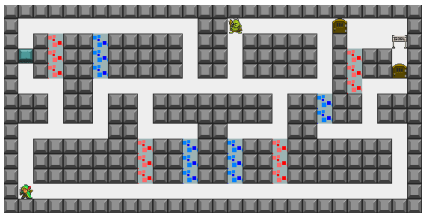
(c) Run (Level 1)



(d) Run (Level 2)



(e) Eighth Passenger (Level 0)



(f) Eighth Passenger (Level 3)

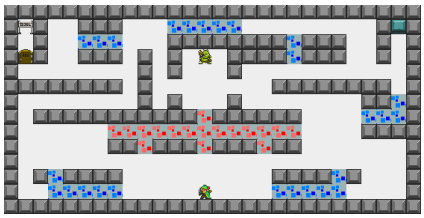


Fig. 3. Selected Games

TABLE II
STATE SPACES AND ACTIONS OF GAMES

Game	State Space	Actions
Camel Race	PLAYER_X × PLAYER_Y × CAMEL_PINK_X × CAMEL_GREEN_X	{UP, DOWN, LEFT RIGHT, NIL}
Run	PLAYER_X × PLAYER_Y × WATER_PROGRESS × KEY	{UP, DOWN, LEFT RIGHT, NIL}
Eighth Passenger	PLAYER_X × PLAYER_Y × ORC_X × ORC_Y × PREV_ORC_X × PREV_ORC_Y × DOOR	{UP, DOWN, LEFT RIGHT, NIL}

one industrial engineer and six computer scientists) since our survey was intended to obtain interpretable results used for qualitative evaluation of our approach.

The software for the survey is based on the GVGAI framework [9] since it contains the games as well as features to collect playtraces by storing all actions taken during gameplay. The survey starts with an introductory text explaining the format of the survey which is as follows:

- The levels described in section IV-A are presented to the participant in random order.
- Before each level starts, a short description of the rules is displayed. The human participants therefore have complete information of the level.
- After each level finishes, the participant is asked to provide a quantification for the perceived strategic depth of the level just played using a slider from 0 to 10 with a granularity of 100 steps.

In order to enable the participants to give an accurate representation of the perceived differences in depth, the players were asked to play the levels in the survey repeatedly and adjust the sliders until they were content with their answers.

To capture what the players did actually experience in the game, besides their final survey answers, we also collected their playtraces. The GVGAI framework already offers a feature to store the action taken by the player at each game tick. This was extended to also include the state space information of the game so that an HKB could be extracted from the results with the algorithm described in section II-B2. For each player, only the fastest run per level was kept. This decision was made to reduce the playthroughs to those where the strategy applied by the player was rather successful. The fastest playthrough is therefore taken as a representation of the respective player's solution to finding the best strategy for a given game. Since the perceived strategic depth hinges on the difficulty of finding this strategy and for reasons of comparability, we excluded answers related to unsuccessful runs from the evaluation. This was another reason for choosing survey participants who are relatively proficient game players. Note that our measure is in principle also applicable to unsuccessful runs, however, in this case it hard to define a proper criterion to compare the strategic depth among different (successfully and unsuccessfully played) games.

C. Post-Processing of Survey Data

From the survey, we obtain two data sets:

- the measured strategic depths according to d_s (see def. 5) from the corresponding playtrace
- the perceived strategic depths according to the participants' estimates

In order to avoid bias introduced by different approaches to the quantification of perceived depth [1] and in order to generalise from the raw values of individual participants, we only intended to capture the proportions of the strategic depths across the levels. We therefore normalised both data sets for each player according to the following approach: Let $\mathbb{D}^{\mathrm{r}}_p = \{v^{\mathrm{r}}_a, ..., v^{\mathrm{r}}_f\}$ be the (raw) values of either measured or perceived depth for player p and levels (a) through (f) (cf.

figure 3). For each level l and player p we then obtain the normalised depth value $v_l^{\mathrm{n}} \in [0,1]$ as

$$v_l^{\mathrm{n}} := \frac{v_l^{\mathrm{r}} - \min(\mathbb{D}_p^{\mathrm{r}})}{\max(\mathbb{D}_p^{\mathrm{r}}) - \min(\mathbb{D}_p^{\mathrm{r}})} \ . \tag{2}$$

However, a problem with the normalisation occurs for survey participants that were not able to finish some of the levels successfully. These player were able to give an answer regarding the perceived strategic depth, but there was no playtrace to calculate d_s from. Nevertheless, removing these missing values before normalisation would be an incorrect representation, because the remaining levels would be normalised to the range of $[0,1]$. For that reason, for the normalisation, we replaced any missing values by the averages obtained through other players so that the normalisation was applied to all levels. Of course, in order to avoid distortions or imbalances, we removed the inserted data afterwards as well as the corresponding perceived depth values. The obtained results therefore still only included accurate data, but we were able to normalise appropriately. Thankfully, with a missing value rate of ≈ 0.0417 for each data set, the described imputation only needed to be applied very rarely and should have minimal influences on the results.

V. Results

In the following we describe the results from our analysis of the measure. In section V-A we compare the measurements with the answers of the survey participants regarding perceived strategic depth in order to validate the measure. After that, in section V-B we compare the results with the approaches to measuring strategic depth presented in section II-A. Finally, we discuss alternative ways to obtain HKBs in section V-C.

A. Validation of the Measure

Our measure is intended to capture the intuitive judgement of strategic depth, so therefore, to validate it, we compare the obtained measurements with the survey answers regarding intuitively perceived depth. The mean values (omitting missing values) of the post-processed survey answers aggregated per level are visualised in the six leftmost bars in figure 4. The mean of the strategy depth estimates according to our measure d_s computed on the best playtraces of each player are shown in the middle in figure 4.

The results are very convincing: Although the ranking according to depth is not exactly the same, the measurement seems to have captured the differences in perceived strategic depth rather well. The first level in *Camel Race* was rated the least deep by all players as well as by our measure d_s. It can be clearly seen that in both bar plot groups, both levels for *Run* and *Eighth Passenger* are similar to each other respectively, while the second level in *Camel Race* was considered strategically deeper than the first one. The mean squared error is $\approx 8.2282 \cdot 10^{-3}$ and the standard deviation is $\approx 9.0703 \cdot 10^{2}$. Especially, the amount of difference in terms of strategic depth between the games in our experiment seems to be captured very accurately.

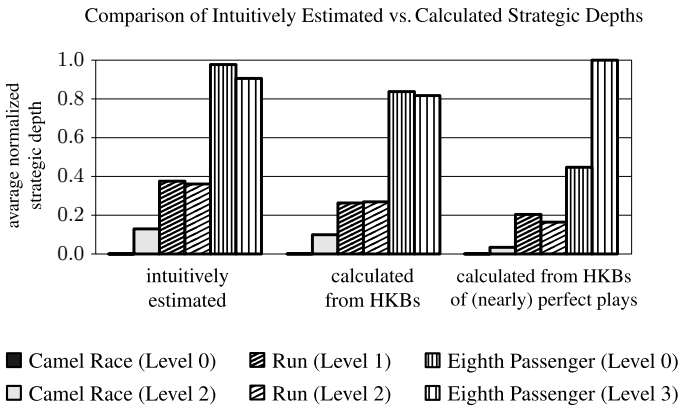


Fig. 4. Evaluation of the Strategic Depth Measure

A minor blemish is that, in our experiment, d_s seems to have a tendency of underestimating the strategic depth. This also seems to be the main reason for the measured standard deviation. Since the average in the plots is based on normalised values, part of this can be explained by the fact that the ranking of the games per player according to our measure was not always the same for each player as in the averaged survey results. Additionally, the differences between the games appear to be less distinctive. However, this could be adapted, e. g., by choosing a slightly higher weight constant b (instead of $b = 2$) and by further optimisations of the weighting of d_s.

B. Comparison to Existing Measures

For comparison of both the results as well as practicability, we also (aim to) compute the measures proposed in related work as described in section II-A. However, we find that a comparison to existing measures for the selected games is only possible within narrow limits.

In order to compute the depth according to [5], we require scores from perfect plays as well as partial solutions generated with AIs using different computational budgets. We were able to obtain the required playtrace for (an approximation of) perfect play by computing the shortest paths for all games and taking it. Only in *Eighth Passenger* moving along the shortest path might not be possible because of the opposing NPC. However, we found that for the level 0 (figure 3 (e)), the ogre did not collide with the player who followed the shortest path. For level 3 (figure 3 (f)), however, this was the case leaving only one alternative route as a viable option.

Surprisingly, the main difficulty in applying this measure was finding AIs that play the game successfully. Since for both *Camel Race* and *Run*, feedback is only given when the game finishes, e. g., MCTS-based AIs have trouble finding a good solution even with a large budget and rollout depth. We were therefore not able to obtain any meaningful data to use with the approach from [5]. However, we were able to find at least approximations for both game tree and state-space complexity for the selected games to compute the depth scores as defined in [3].

However, if we compute the depth score according to these estimates, for the respective first levels, we obtain ≈ 0.0311

for *Camel Race*, ≈ 0.0138 for *Run* and ≈ 0.0094 for *Eighth Passenger*. The ranking according to these scores is obviously counter-intuitive and the opposite of what was experienced by the players. *Run* and *Eighth Passenger* are not significantly far apart considering the level of approximation, but the depth score for *Camel Race* is far too high in comparison. This might be an effect of the relatively small state-space for this game. However, in general, the computed scores are very low across all games. We assume that the measure proposed in [3] is not suitable for non-board games and especially games that are as strategically simple as the selected ones.

C. Applying the Measure to AI Playtraces

Theoretically, the measure described in section III could also be computed based on playtraces generated by an AI player. In case a player AI exists, this would of course require considerably less effort than conducting even a small survey. For our experiments we, however, chose to base the measurements on playtraces from human players despite the fact that the results would only provide anecdotal evidence. The reasoning behind this decision hinges on the assumption that an experience-driven approach will better correspond to how humans perceive a game and its strategic depth.

To show that the proposed measurement does not work equally well on any playtrace, we also compute d_s for the (nearly) perfect playthroughs created as described in section V-B. The results are visualised as the rightmost bars in figure 4. It is very clear from the plot that the results do not align with the survey results nearly as well as the values computed from playtraces of human players. The obtained mean squared error was $\approx 6.1225 \cdot 10^{-2}$ and thus significantly larger than the one of $\approx 8.2282 \cdot 10^{-3}$ from the experience-driven approach.

While this result shows clearly that not any playtrace can be used, it is reasonable to assume that an aggregate over multiple playtraces from AI players, especially more explorative ones, might fare better. It seems especially interesting to investigate the performance of measures based on playthroughs from believable AIs (see e. g. [10]). This is of course only viable if these AIs are able to solve the respective game.

VI. Conclusion

In this paper, we proposed and provided a proof-of-concept for a new approach to measure strategic depth in games. Our approach was specifically focused on player experience as well as practicability. We thus tested it on three different arcade-style games with two levels each and obtained very promising results. The measure was able to capture rather accurately which games were perceived to be of similar strategic depth according to a survey as well as the perceived differences.

The validation in this paper was only based on a qualitative, small-scale survey in order to obtain a proof-of-concept. This is why we also refrained from using statistical analysis of the results. It is clear that the accuracy of the measure should be verified with a larger survey as well as more games. We therefore intend to extend the survey to more games contained in the GVGAI framework. This would only require a state-space representation like the ones in table II for the considered games. Obtaining the representations automatically based on the level descriptions and observations provided by the framework could be another option.

Furthermore, it would be interesting to see how well the measure fares in terms of computational effort for games with significantly larger state-action spaces (efforts to compute HKBs faster have been made in [7] already). For a very complex game, the space might however be reduced again to model a human player that is not able to observe everything either. Additionally, the measure should likely be validated on games with different main mechanics as well.

The measure could also be adapted to choose strategies instead of actions, thus hopefully reflecting a higher degree of long-term planning which is not possible for the games in this survey. The sensors could be abstracted as well. However, the application on this meta-level would require a large amount of context knowledge, automatic detection mechanisms, or both. For games in a continuous space, some level of abstraction, i.e. discretisation, would already be required.

On top of that, as hinted at in section V-C, despite the comparatively weak results for the measure applied to artificially created playtraces, it might be worth investigating further what results can be obtained from playtraces recorded from (believable) AI playthroughs. If evaluation based on AI players is viable, we plan to eventually use the proposed measure in the context of automatic game balancing (cf. [2]).

References

[1] N. Shaker, G. Smith, and G. N. Yannakakis, "Evaluating content generators," in *Procedural Content Generation in Games: A Textbook and an Overview of Current Research*, N. Shaker, J. Togelius, and M. J. Nelson, Eds. Springer, 2016, pp. 211–218.

[2] V. Volz, G. Rudolph, and B. Naujoks, "Demonstrating the Feasibility of Automatic Game Balancing," in *Genetic and Evolutionary Computation Conference*. ACM Press, New York, NY, USA, 2016, pp. 269 – 276.

[3] C. Browne, "Elegance in game design," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4, no. 3, pp. 229 – 240, 2012.

[4] F. de Mesentier Silva, A. Isaksen, J. Togelius, and A. Nealen, "Generating heuristics for novice players," in *Computational Intelligence and Games Conferece (CIG)*. IEEE Press, 2016, pp. 158–165.

[5] F. Lantz, A. Isaksen, A. Jaffe, A. Nealen, and J. Togelius, "Depth in strategic games," in *AAAI Workshop on What's next for AI in games*, 2017.

[6] G. S. Elias, R. Garfield, and K. R. Gutschera, *Characteristics of Games*. Cambridge/London: MIT Press, 2012.

[7] D. Apeldoorn and G. Kern-Isberner, "Towards an understanding of what is learned: Extracting multi-abstraction-level knowledge from learning agents," in *Proceedings of the Thirtieth International Florida Artificial Intelligence Research Society Conference*, V. Rus and Z. Markov, Eds. Palo Alto, California: AAAI Press, 2017, pp. 764–767.

[8] ——, "When should learning agents switch to explicit knowledge?" in *GCAI 2016. 2nd Global Conference on Artificial Intelligence*, ser. EPiC Series in Computing, C. Benzmüller, G. Sutcliffe, and R. Rojas, Eds., vol. 41. EasyChair Publications, 2016, pp. 174–186.

[9] D. Perez-Liebana, S. Samothrakis, J. Togelius, T. Schaul, and S. Lucas, "General video game ai: Competition, challenges and opportunities," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*. AAAI Press, 2016.

[10] C. Holmgård, A. Liapis, J. Togelius, and G. N. Yannakakis, "Generative Agents for Player Decision Modeling in Games," in *Conference on Foundations of Digital Games (FDG)*, 2014.