How Will a Mixed-Initiative Level Editor that Predicts User Requirements Affect the Levels Created?

Tristan Barlow-Griffin

Abstract—Not sure what should go here Not sure what should go here Not sure what should go here Not sure what should go here Not sure what should go here Not sure what should go here Not sure what should go here Not sure what should go here Not sure what should go here Not sure what should go here Not sure what should go here Not sure what should go here Not sure what should go here

I. INTRODUCTION

THIS research project will look into how a prototyping tool with a Mixed-Initiative (MI) component that predicts user requirements will affect the process and design of a level. A prototype is the initial design of an object [1], the prototyping phase is used to quickly test certain aspects of a products' design so the designer can identify and clear up any problems before full production commences [2]. Fullerton *et all* [3, p. 150] state there are two kinds of prototyping in games: Physical and Software prototypes. Since the books publication in 2004, the accessibility of tools that help designers prototype levels has increased. Fullerton *et al* [3, p. 164] also describes a level editor as a good way to prototype levels. The free to download game engine called *Unreal Engine 4* (UE4) has a level editor built into it. Within this editor, the designers can create basic geometry scaling them to fit their needs. In addition, designers can add custom meshes and programmable objects to turn their levels into games.

The prototyping phase is meant to test the design of a product, the less time and resources required to produce an artefact that can demonstrate the proposed design the better. Beyond the benefit of saving time, the less time a designer puts into a particular design the less attached to the design they become. When collaborating in a group, differing opinions can cause different constraints to be set on the design of a product. While a given design may satisfy the original designers set constraints, the prototype may have to be discarded as it did not meet the other requirements set by the team. Identifying and discarding concepts early in development can save a lot of time and energy [6, p.489]. Arguably, this will reduce the negative impacts to interpersonal relations that idea dismissal may have. Developing a level editor that is meant to only prototype a levels' design will allow for less focus on the polish of the level created as the aim of the prototype is just to test if it fits the users requirements. This paper will build upon a standard level editor by adding an MI component, this component aims to implement features requested from Alvarez *et al* [4] study on their MI level editor. The implementation

of these features is meant to increase the overall ease of designing a level through predicting what the user may require. In addition, this paper looks at the Sentient Sketchpad [5] an existing MI level editor, comparing their design to MI tool design theory and general user interface (UI) design theory. The hypotheses in this paper are only concerned with how the MI component interacts with the participants and what kind of impact theses interactions will have when trying to prototype a level. Looking at existing studies on the most effective prediction methods [40], [43], [46] this paper identifies and evaluates the methods proposed in the context of a MI level editor.

II. RELATED WORK

The lack of documentation about prediction methods being used in game design, required this paper to go beyond the scope of just game design examples. The methods of prediction used in this study must be effective for the hypotheses to be tested accurately. To this end, Section V looks at currently used prediction methods comparing the performance results of each method. In Section III, mixed-Initiative tools are group into two broad categories using Liapis *et al* [7] definition. The definition of mixed-initiative used in this paper will also be defined in Section III. Defining these groups makes it easier to distinguish between the most common types of mixed-initiative tools. This section will also explore the different ways MI tools are being leveraged both within and outside of the games industry. In Section IV there is an evaluation of the existing mixed-initiative level editors. There is also a discussion and comparisons of these MI editors in the context of more general MI and UI design theory.

III. MIXED-INITIATIVE

The term mixed-initiative was first introduced in *1970* by Jaime R [8]. It describes a process whereby a computer and a human designer work together to achieve a goal. Other definitions of mixed-initiative build on the idea of human-computer co-creativity. This paper will use the first definition of mixed-initiative presented, as trying to define creativity is a very complex matter in itself.

There are two broad categories that MI tools can be grouped into, Interactive evolution and Computer-aided design [7].

- *Interactive evolution*(IE) is where the designer has the idea, and the computer helps them realise it. The

computers' role is to evaluate the humans' design, presenting alternative solutions if any constraints are broken.

- *Computer-aided design* (CAD) is where the computer generates the content, but does not evaluate the quality of the produced work. In CAD, the human designer will evaluate the work and use these evaluations to move towards a more desirable product space.

The first documented mixed-initiative tool created helped students to learn the English language [8]. The uses of mixed-initiative tools have greatly expanded since 1970 and have been described as a backbone tool for designers [4]. The application of mixed-initiative systems in more complex environments have given mixed results [9]. Barnes *et al* [9] found that in most cases, systems that divide decision making between a human and an intelligent agent were generally more effective than if decisions were just dependent on the one. This can be seen with predictive texting, only choosing the words suggested to you by your phone can result in unexpected and hilarious results [10]. Barnes *et al* [9] found that human designers were better at making abstract decisions and inferring the significance of an object or event. Kantosalo *et al* [11] focus their MI tool on user-centred design, this means their AI agent played a back-seat role. Kantosalo *et al* [11] propose future work where the agent and the designer have an equal role in the system, they do not make any conjecture on the anticipated results.

PCG and CAD are closely related fields, the difference between CAD and PCG is in the evaluation period. If a designer were to use a PCG algorithm to generate a level and then look through the produced results evaluating and picking maps this would be considered CAD [7]. For it not to be considered CAD, the PCG algorithm would have to perform the evaluation itself. For example, this might include checking if the map is completable or if it is a certain size. The field of procedural content generation(PCG) has advanced significantly [12]. The uses of PCG are ever increasing as publishers seek to lower costs of production [13], [14].

One may choose to use PCG in games as it will increase the quantity and variation of levels produced ensuring replayability [15]. PCG algorithms can also be shipped with the games they are made for, this allows for an inexhaustible source of new maps [16]. Doing so will extend the games life-span giving the players an amount of content that would otherwise be impossible. Although there is no guarantee that this content will be interesting or unique. However, using CAD tools to generate maps there will always be a human element evaluating the maps. With this input the human designer may discard maps that are similar to existing maps, thus providing quality assurance not present in a PCG algorithm.

Prototyping a level should be a fast iterative process [17], the method used to prototype should allow for instant feedback on the design. This feedback needs to have an easy channel back to the designer so that amendments can be made with ease. Map generation algorithms, even in under ideal circumstance will only provide the designer with a set of parameters to change [18]. Small variations in the given parameters can produce large changes in the maps design [19]. This will make it difficult to make small changes when given feedback. As a result, reduce the effectiveness of a CAD approach when trying to prototype levels.

Within an IE environment, the core of the creative process relies on the human designer. As the main creative driver, the human has the most input. Ideally, the computer will add value by providing supplementary support, for example, alerting the designer when constraints have been broken. It can be argued as the constraints are determined more by the human, the size of the possible output space will be larger. Yanakakis *et al* [20] claim that CAD examples like PCG limit the designers' intentions as they follow their own algorithms. Doran and Parberry [18] inadvertently corroborate Yanakaki*et al* [20] view, as they say, a PCG algorithm should ideally, have a set of designer-centric parameters as their only form of control. This limited control over the creation of levels may, in some situations, be enough. During the prototyping phase, when the requirements needed from a map are changing the fastest, the level of control provided through CAD might not be enough to handle large changes in design.

IV. MIXED-INITIATIVE LEVEL EDITORS

A user interface should be intuitive to use and not require any additional helping systems [21]. Liapis *et al* [5] aim to achieve this by creating a design tool that allows users to create levels using a low-resolution graphical interface. Figure 1 shows the interface used to design levels for a strategy game. The designer can place tiles on the map which will colour in the given tile with the placed tiles colour. During a tile placement, the tool will test for the playability of the map, checking to see if placing the current tile will break any of the games' constraints. The Sentient sketchbook also provides alternative viewing modes, examples of these alternate viewing models can be seen in Figure 2. There is no mention in the study how often these tools were used, Galtiz [22, p. 752] warns against too many graphics on the screen.

Fig. 1. Liapis *et al* Sentient Sketchbook during a design session [5].

Baldwin *et al* [23] have also implemented a mixed-initiative dungeon designer called the evolutionary dungeon designer or (EDD) for short. EDD is closer to a CAD tool than the IE tool Sentient Sketchbook [5]. While the approaches may be different see Figure 3, both MI tools allow for large
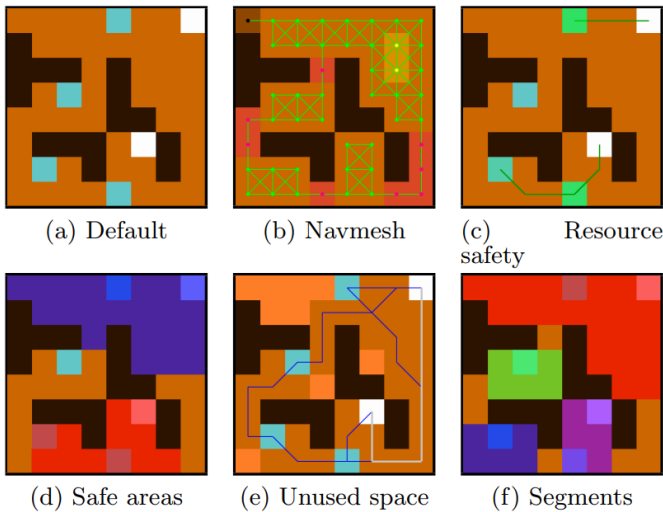
(a) Default   (b) Navmesh   (c)   Resource safety

(d) Safe areas   (e) Unused space   (f) Segments

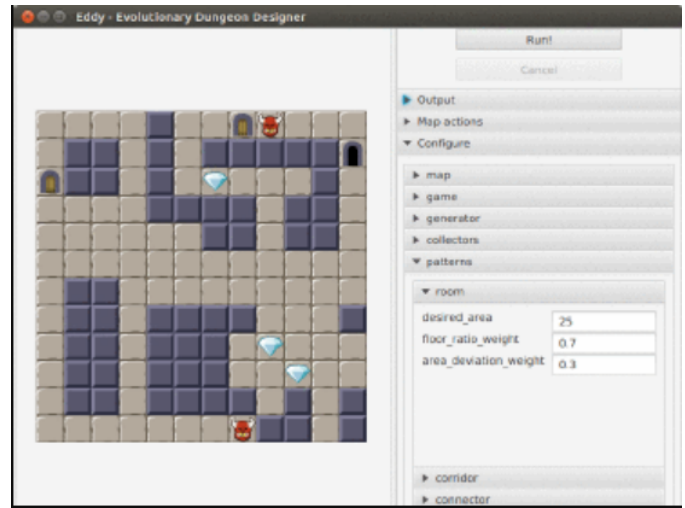Fig. 2. Liapis *et al* Sentient Sketchbook Different viewing modes [5].

Fig. 3. Baldwin *et al* Evolutionary Dungeon Designer user interface [23].

Fig. 4. Alvarez *et al* Evolutionary Dungeon Designer with modifications user interface [4].

customisation of the levels generated see Figure 3. Baldwin *et al* [23] take a different approach to Liapis *et al* [5]. Baldwin *et al* [23] core concept is to identify design patterns within the level design. These design patterns consist of multiple tiles that constitute common patterns found in games. Alvarez *et al* [4] builds upon the EDD suggested in [23] by adding an IE element to it. It could be argued that the new version of the EDD has an improved interface with lots of its excess drop-down menus gone see Figure 4. Beyond the ascetic differences, Alvarez *et al* [4] dungeon designer integrates key aspects of the Sentient Sketchpad [5]. The second edition of the EDD allows the user to design their own levels, it will then offer suggestions based upon the map the user created, this can be seen in the top right corner of figure 4. The results from Alvarez *et al* [4] experiment focused on whether their tool fostered creativity in the participants using it. Included with the results is a table of requested features made by the participants of the study. One key element highlighted is that the EDD should do a "bit more automated assistance when doing manual designs, which can reduce clicking around the program" [4, Table 2]. Another significant feature request was for the dungeon designer to take into account the pattern of the entire dungeon, using the map of the dungeon to generate new rooms.

Horvitz [24] proposes 12 critical factors to take into consideration when making a mixed-initiative user interface. While Horvitz [24] focuses on a mixed-initiative assistant for Microsoft Outlook (emailing software) it can be argued that some of these factors are relevant for a level designer. The first factor that is listed is that an MI tool needs to add significant value through the automation of services. An example of a service automated by emailing assistant is the sorting of a users' emails into different categories. Within the context of [5] they satisfy [24] first critical factor by allowing the computer to automate some of the map design services like checking for broken game constraints. Liapis *et al* [5] also allows their algorithm to take on a creative role all be it based on an original human designed map. Within this project,
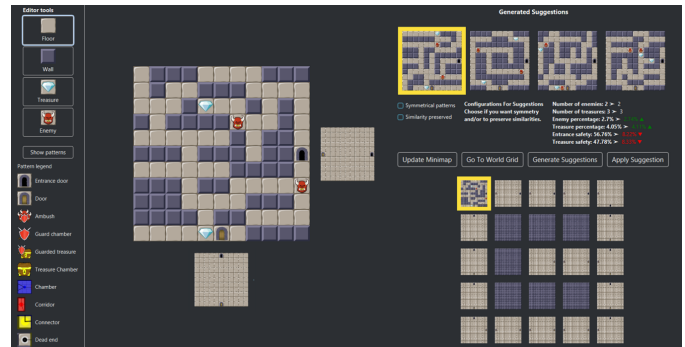
the focus will not be on the creative aspect as the definition of creativity is hard to for a computer to understand [25]. Alvarez *et al* [4] found their MI tool is better at providing controllability than expressivity, when the user imposes their vision, as it is hard for a computer to capture the designers' vision. It can be argued that an MI tool could not consistently add value if it cannot capture the designers' vision. Smith *et al* [17] believe that human designers strengths lay in creativity and their ability to evaluate good content and the computer lacks this ability.

Another factor raised by Horvitz [24] is that a tool must consider minimizing the costs of poor guesses about the users' goals. Even with an extensive history of the users' requirements, a novel use of the tool might be required. It is important for a system to recognise when something novel is happening and for it not to attempt predictions. Some authors find value in these missed guesses and even seek to find novelty search spaces [5]. Other authors [4], [20], [26] claim these kind of mistakes can foster creativity and alternate suggestions that do not aim to predict the user can be beneficial. On the right-hand side of Figure 1 the results of the guessing algorithm are shown. Clicking retry will remove the current maps and create new ones for the designer to evaluate.

None of the above examples [4], [5], [23] satisfy Barnes

et al [9] statement that an MI systems UI must provide transparency to the reasoning behind the agents actions. In all cases above [4], [5], [23] , when generating new suggestions the reasoning behind each suggestion was not given to the designer. The designer is presented with the statistics of the current map generated (density, number of resources etc.) but it is not clear from the interface how these statistics are being used. Lee and See [27] agrees with Barnes *et al* [9] that MI systems should be transparent, they go so far as to say it is required for a human to trust the automation process [27].

V. PREDICTION METHODS

Predictive texting increases the average message length users send to each other [28] as well as the speed the words are written [29]. The same theory may apply to game design. If patterns of a users game design are established, an AI system may be able to assist in design. In this section, this paper will look into different methods of predicting human requirements, this paper will also discuss the pros and cons of each technique.

The researchers of [30] have tested alternate methods for predicting human input so as to abstract the low-level movements of the robots the humans were controlling. They built on the idea that humans are good for high-level abstract tasks, but an AI agent was much better at performing low-level repetitive control tasks. In addition, they found that when trying to predict the next human input, trying to identify patterns in a history of events was not successful. Instead, using just the last event yielded much better results, hence their title "less is more". Instead of using current human inputs, [31] used the history of the users' social media page to predict the users' interests. Perhaps if the authors of [30] had looked less at the input history of the human and instead focused on grouping inputs together to create larger actions. Similar, to how modern day phones often predict entire sentences rather than just single words.

A. Markov Chains

Markov Chains implements a theory similar to the methods previously discussed by Bhatia and Hasija [31]. A Markov chain is a special kind of process that works under the assumption that the state at time *t+1* depends only on the current state. Another explanation is the state at time *t+1* is exclusively dependent on the state at time *t*. This means State *t+1* is not dependent on the history of the states leading up to it [32]. This technique has been proven useful for predicting anomalies in systems [32]–[34] where the states are heavily dependent on the latter state happening. However, it is hard to see how during a creative process, where the next state is dependent on the vision of the designer, the state at *t+1* can be predicted from just knowing the state at *t*. The introduction of a Markov model makes Markov Chains method more viable as a prediction technique. A Markov model can be used to describe the probabilistic relationship between the previous states in a Markov Chain [35]. Higher order Markov chains relax the rule of the next state being only dependent on the current state by allowing the network structure to look *n* number of states

back from the current state [36]. Snodgrass and Ontanon [37] used Markov models as a way to model level data. Using these Markov models Snodgrass *et al* [37] applied different sampling techniques, they found using their higher order Markov chains was more effective than using standard Markov chains. This conflicts with Chipalkatty *et al* [30] findings of "less is more", Snodgrass and Ontanon [37] found that using a history of states to predict the next state was more effective to achieve their goals.

B. Artifical Neural Network

An artificial neural network (ANN) consists of many nodes that are divided into separate layers. Each node receives inputs from other nodes, the value of these inputs depends on the weight of the connection between the nodes [38]. ANN are used in prediction methods focusing on outputting numerical values [38], [39]. Shepperd and Kadoda [40] found that case-based reasoning(CBR) outperformed a ANN on more occasions. However, Shepperd and Kadoda [40] highlighted the dependence of the respective techniques on the nature of the training sets used. Figure 5 shows stepwise regression procedure proving to be more accurate at prediction than both CBR and ANN on small datasets. A time series approach to neural networks has been found to increase their accuracy when dealing with world state predictions [41]. The number of data points in the training set that Hazarika and Lowe [41] used was much larger than in [40] at 500 points for training and 250 for validation. Looking at examples from [4], [5], [23] the maximum level dimensions are 12 tiles wide by 12 tiles tall. This means that the maximum number of data points provided by one map would be 124, which is far less than Hazarika and Lowe [41] use to train their ANN.

Dataset	Small training set (20)				Large training set (100)			
	SWR	RI	CBR	ANN	SWR	RI	CBR	ANN
Normal	9.90	23.00	20.03	15.25	9.77	13.41	17.90	38.96
	10.32	20.70	22.14	17.63	9.31	15.67	17.80	9.09
Normal + outliers	36.57	166.68	205.04	162.39	51.33	96.16	37.62	28.12
	63.95	256.05	57.64	160.30	40.19	88.11	34.39	32.57
Normal + multicollinearity	11.11	27.95	26.03	36.71	17.87	27.17	24.65	46.63
	20.65	28.23	34.07	38.36	12.04	19.96	21.33	14.16
Normal + outliers + multicollinearity	285.73	139.51	26.14	232.65	172.14	17.70	13.71	62.08
	140.59	149.36	22.50	261.63	148.22	18.52	14.55	52.57

Fig. 5. Shepperd *et al* Analysis of Accuracy (MMRE) for Continuous Model (Y1) [40].

C. Case-based Reasoning and Stepwise Regression procedure

Case-based reasoning (CBR) is a prediction technique that uses a history of completed products to predict solutions to problems in the current product. This is done by comparing features in the current problem description to problems already solved in existing projects. Usually the problem description that is most similar to the current problem is used to estimate

the solution [42], [43]. The core of a case -based reasoner is solving new problems by using or adapting solutions from old problems [44]. Watson [45] describes CBR as a methodology not a technique and provides several different techniques to apply CBR. Mendes and Mosley [43] have compared some of the techniques found in [45] as well as additional techniques. They found that using a CBR technique that used adaptation rules performed significantly better than techniques that did not use adaption rules. In addition, they found the CBR techniques that used weighted euclidean distance also gave the best predictions. Much like Shepperd and Kadoda [40], Mendes and Mosely [43] also found that stepwise regression gave the best prediction accuracy "for all measures of prediction accuracy" [43, p. 11]. The dataset used in [43] consisted of 34 data points which could be considered a small data set when comparing the number of data points used in [41]. Looking at Figure 5 there is a strong case for both SWR and CBR as prediction techniques with a small data set. Wen *et al* [46] also state CBR to be accurate with a small data sets ,validating the results found by Shepperd and Kadoda [40]. If CBR is described as being accurate with a small data set [46] and SWR techniques consistently out perform it with both small and large sets [40], [43]it can be inferred that SWR is also a good technique to use with small data sets. Stepwise regression adds to the prediction model the variables with the highest partial correlation to the response variable at each stage [47]. With the aim to have a set of variables (predictors) in the model to maximise F, F describes the association of all of the predictors to the response variable [47]. For a variable to be added to the model it must increase F by a constant specified amount a commonly described as the(Alpha-To-Enter) value. Similarly for a value to be removed from the model it is measured by its reduction in F and is also compared against a constant described as the (Alpha-To-Leave).

VI. METHODOLOGY

One key motivation for this experiment comes from Alverezs' *et al* [4] literature, in particular a users' feature request for more automated assistance with the aim to reduce clicking. To this end, this project proposes a tile based level editor which has a mixed-initiative component. The mixed-initiative component will reduce clicks by predicting what the users next action will be. In addition, this MI component will be used to discover if the predictive texting findings of Ling [28] and Dunlop and Crossan [29] are applicable to level design. The research question proposed in this project is: How Will a Mixed-Initiative Level Editor that Predicts User Requirements Affect the Levels Created?

A. Hypotheses

When creating a level editor the aim is to increase the ease at which levels could normally be created. For the scope of this paper, the focus is on the MI component within the context of the editor, not the editor itself. Table I shows the proposed hypotheses for this paper, it also shows from which source of data each hypothesis depends. Ling [28] found that predictive texting increased the speed at which messages are written.

Also, Dunlop and Crossan [29] found that predictive texting also increased the size of the messages written. Hypothesis 2 and 1 will see if these findings are also applicable to level design with the MI component replacing the predictive texting element. Hypothesis 3 will be investigating whether a predictive component will satisfy the feature proposed by Alverezs' *et al* [4]. Hypothesis 4 will build on the work of Alvarez' *et al* [4] and see if the MI level editor can make designers consider alternative level design approaches. Finally, hypothesis 5 will discover if designers with more experience are more likely to find the MI a negative influence than designers with less experience, the argument being that more experienced designers do not need help to realise their design.

B. Participants

Once a week, Falmouth Universities' Games Academy hosts different guild sessions for each game design route (design, art, animation, programming, writing and audio). All the participants for this experiment will be randomly sampled from these guild sessions through an online random name selector. Approaching the game design guild and another guild will ensure a range of level design ability. A power analysis for this experiment resulted in a sample size of 23. The power included a two-tailed t-test with an effect size of *0.8*, an alpha error probability of *0.05* and a *power(1- beta error probability)* of *0.95*. In this experiment all of the participants will be in one group, they will each be required to design levels using the level editor provided. By the end of the experiment, all of the participants will have designed the same number of levels. Everyone in this study will be asked to complete a consent form, given an information sheet and the option to quit the experiment at any time. The information sheet will only give the basic information of the project as we do not want to invite acquiescence bias [48]. The data collected in this experiment will not be enough to identify anyone, keeping the participants anonymous.

C. Design Session

The design session will require the participants to create five different levels each with different editor settings. Before the session starts the users are prompted with a multiple choice question. This question will require the participants' to self-assess their own game design experience, from "not a lot" to "a lot" of design experience. After this, the user will be presented with the main level editor interface. They will then be given as much time as they need to familiarize themselves with the interface. The different editor settings the participants will be working in can be seen in Table II. The order in which the different settings will occur will be random, this will reduce the impact of the practice effect on any data gathered. The computer will choose the order in which the editor setting will be applied. This means the observing researcher will not know what current settings are applied. By making this a double-blind study it will minimize the chance of investigator/observer bias [49]. Before each level starts an on-screen prompt will inform the user of the rules of the current level, for example: "During this level there will

TABLE I
HYPOTHESES TABLE

	Hypothesis	Null Hypothesis	Data Source
1	The MI component will effect the speed at which the participants create levels.	The MI component will not effect the speed at which the participants create levels.	Design Session Statistics
2	The MI component will effect the size of the levels the participants create.	The MI component will not effect the size of the levels the participants create.	Design Session Statistics
3	The MI component will effect the number of the clicks the designer makes.	The MI component will not effect the number of the clicks the designer makes.	Design Session Statistics
4	The MI component will influence participants to explore other level designs.	The MI component will not influence participants to explore other level designs.	Questionnaire
5	The participants with more experience will be Negatively impacted more often by the MI component.	There will be no correlation between experience and the negative impact of the MI component.	Questionnaire

be a component that predicts your requirements.". All of the levels will be silently timed so as not to influence participants to work faster than they would usually. At any point during this experiment the participants can withdraw from the study, doing so will cause any data gathered on the levels they have created to be classified as incomplete and will be deleted. After completion of each level, the designer will be asked a few multiple choice questions. Each question will have five possible answers, the answers are relative to the questions asked but will follow a similar scale from negative to positive. After the questions have been answered any data gathered will be stored along with the current configuration. This will later be sent to the server once the participant has completed the experiment. The questions are:

- How frequently did the level editor make you consider an alternative level design?
- How much impact did the level editor have on your work flow?
- What kind of impact did the level editor have on your work flow?
- How difficult did you find it to make the level you wanted?
- How pleased are you with the final design of the level?

An alternate method would be to have a questionnaire at the end of the experiment once all levels have been completed. However, as each set of responses correlates to a different editor configuration, it will make data management easier to get all the data for a single configuration in one go. In addition, leaving the questions until the end in one questionnaire may invite levelling and sharpening bias, this is where details are lost and others are heightened over time [50].

TABLE II
EDITOR SETTINGS

Editor Settings	Predictive Placements Enables	Unrestricted size	Exploration
Settings 1	X	X	
Settings 2		X	
Settings 3	X		
Settings 4			
Settings 5	X		X

D. The level editor

In this experiment there will be two kinds of game objects that can be placed in the level rooms and props. Rooms will occupy tiles on the map, whereas the props can be placed within any of the rooms. Figure 6 shows the first prototype design for the level editor. Barnes *et al* [9] suggests that any mixed-initiative component should provide reasoning behind the agents actions, to do this the prototype contains an information log, this information log will be used to tell participants why the level editor has made a particular decision. The level view will be a tile based view of the current map, it will allow users to place tiles and props according to their respective rules. Figure 6 also shows the layout of the level editor, on the left is where the buttons to switch between what prop/room will be placed when the participant next clicks. The check button will be used to make sure the game is completable. The checking algorithm will be simple, it will make sure there is a start and an end tile, it will also check these tiles are connected by a valid path. The level complete button will perform a check before finishing to make sure a valid level is being submitted. If a level were to be rejected, the information log will be used to inform the user why it has been rejected.

E. Prediction Technique

When deciding the prediction technique for this experiment the main limiting factor to consider is the number data points available. In the MI level editors discussed in Section IV, the number of tiles present within a level did not exceed 144 [4], [5], [23]. By treating each tile as one data point this gives a maximum number of 144 data points to use when training the MI component. It has been proven that CBR techniques have proven to be effective at prediction even with a small data set [40], [46]. However, SWR has been proven to out perform CBR on multiple occasions [40], [47]. It seems clear that using a stepwise regression model will increase the accuracy of the MI component, allowing the focus to be on how the participants interact with the MI component and less on the accuracy of the predictions.
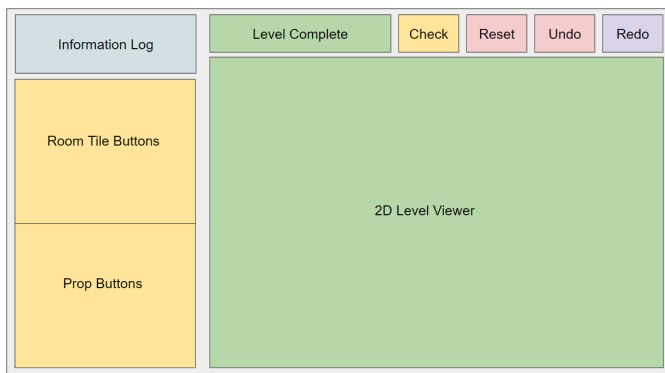
Fig. 6. The first iteration of the proposed level editor

REFERENCES

[1] A. H. Blackwell and E. Manar, "Prototype." accessed 12 Nov. 2018. [Online]. Available: http://link.galegroup.com/apps/doc/ENKDZQ347975681/SCIC?u=dclib_main&sid=SCIC&xid=0c8f739d

[2] R. Budde, K. Kautz, K. Kuhlenkamp, and H. Züllighoven, "Prototyping," in Prototyping. Springer, 1992, pp. 33–46.

[3] T. Fullerton, C. Swain, and S. Hoffman, Game design workshop: Designing, prototyping, & playtesting games. CRC Press, 2004.

[4] A. Alvarez, S. Dahlskog, J. Font, J. Holmberg, C. Nolasco, and A. Österman, "Fostering creativity in the mixed-initiative evolutionary dungeon designer," in Proceedings of the 13th International Conference on the Foundations of Digital Games. ACM, 2018, p. 50.

[5] A. Liapis, G. N. Yannakakis, and J. Togelius, "Sentient sketchbook: Computer-aided game level authoring." in FDG, 2013, pp. 213–220.

[6] J. Stempfle and P. Badke-Schaub, "Thinking in design teams-an analysis of team communication," Design Studies, vol. 20, no. 5, pp. 439–452, 1999.

[7] A. Liapis, G. Smith, and N. Shaker, "Mixed-initiative content creation," in Procedural content generation in games. Springer, 2016, pp. 195–214.

[8] J. R. Carbonell, "Mixed-initiative man-computer instructional dialogues." BOLT BERANEK AND NEWMAN INC CAMBRIDGE MASS, Tech. Rep., 1970.

[9] M. J. Barnes, J. Y. Chen, and F. Jentsch, "Designing for mixed-initiative interactions between human and autonomous systems in complex environments," in Systems, Man, and Cybernetics (SMC), 2015 IEEE International Conference on. IEEE, 2015, pp. 1386–1390.

[10] S. Ochs, "Just for fun: Show us your best quicktype poetry," 2014, accessed 12 Nov. 2018. [Online]. Available: https://www.macworld.com/article/2692885/just-for-fun-show-us-your-best-quicktype-poetry.html

[11] A. Kantosalo, J. M. Toivanen, P. Xiao, and H. Toivonen, "From isolation to involvement: Adapting machine creativity software to support human-computer co-creation." in ICCC, 2014, pp. 1–7.

[12] R. Van der Linden, R. Lopes, and R. Bidarra, "Designing procedurally generated levels," in Proceedings of the the second workshop on Artificial Intelligence in the Game Design Process, 2013.

[13] L. Doherty, M. Whitney, J. Shakya, M. Jordanov, P. Lougheed, D. Brokenshire, S. Rao, S. Menon, and V. Kumar, "Mixed-initiative in computer games: Algorithmic content creation in open-ended worlds," in AAAI Fall Symposium on Mixed-Initiative Problem-Solving Assistants, 2005, pp. 46–56.

[14] J. M. Font, R. Izquierdo, D. Manrique, and J. Togelius, "Constrained level generation through grammar-based evolutionary algorithms," in European Conference on the Applications of Evolutionary Computation. Springer, 2016, pp. 558–573.

[15] D. Karavolos, A. Bouwer, and R. Bidarra, "Mixed-initiative design of game levels: Integrating mission and space into level generation." in FDG, 2015.

[16] L. Johnson, G. N. Yannakakis, and J. Togelius, "Cellular automata for real-time generation of infinite cave levels," in Proceedings of the 2010 Workshop on Procedural Content Generation in Games. ACM, 2010, p. 10.

[17] G. Smith, J. Whitehead, and M. Mateas, "Tanagra: Reactive planning and constraint solving for mixed-initiative level design," IEEE Transactions on Computational Intelligence and AI in Games, vol. 3, no. 3, pp. 201–215, 2011.

[18] J. Doran and I. Parberry, "Controlled procedural terrain generation using software agents," IEEE Transactions on Computational Intelligence and AI in Games, vol. 2, no. 2, pp. 111–119, 2010.

[19] J. Regier and R. Gresko, "Random asset generation in diablo 3," Invited Talk, UC Santa Cruz, 2009.

[20] G. N. Yannakakis, A. Liapis, and C. Alexopoulos, "Mixed-initiative co-creativity." in FDG, 2014.

[21] R. Oppermann, "User-interface design," in Handbook on information technologies for education and training. Springer, 2002, pp. 233–248.

[22] W. O. Galitz, The essential guide to user interface design: an introduction to GUI design principles and techniques. John Wiley & Sons, 2007.

[23] A. Baldwin, S. Dahlskog, J. M. Font, and J. Holmberg, "Mixed-initiative procedural generation of dungeons using game design patterns," in Computational Intelligence and Games (CIG), 2017 IEEE Conference on. IEEE, 2017, pp. 25–32.

[24] E. Horvitz, "Principles of mixed-initiative user interfaces," in Proceedings of the SIGCHI conference on Human Factors in Computing Systems. ACM, 1999, pp. 159–166.

[25] A. Jordanous, "Defining creativity: Finding keywords for creativity using corpus linguistics techniques." in ICCC, 2010, pp. 278–287.

[26] A. Liapis, G. N. Yannakakis, C. Alexopoulos, and P. Lopes, "Can computers foster human users' creativity? theory and praxis of mixed-initiative co-creativity."

[27] J. D. Lee and K. A. See, "Trust in automation: Designing for appropriate reliance," Human factors, vol. 46, no. 1, pp. 50–80, 2004.

[28] R. Ling, "The length of text messages and the use of predictive texting: Who uses it and how much do they have to say," Association of Internet Researchers, Chicago, IL, 2005.

[29] M. D. Dunlop and A. Crossan, "Predictive text entry methods for mobile phones," Personal Technologies, vol. 4, no. 2-3, pp. 134–143, 2000.

[30] R. Chipalkatty, G. Droge, and M. B. Egerstedt, "Less is more: Mixed-initiative model-predictive control with human inputs," IEEE Transactions on Robotics, vol. 29, no. 3, pp. 695–703, 2013.

[31] V. Bhatia and V. Hasija, "Targeted advertising using behavioural data and social data mining," in Ubiquitous and Future Networks (ICUFN), 2016 Eighth International Conference on. IEEE, 2016, pp. 937–942.

[32] N. Ye et al., "A markov chain model of temporal behavior for anomaly detection," in Proceedings of the 2000 IEEE Systems, Man, and Cybernetics Information Assurance and Security Workshop, vol. 166. West Point, NY, 2000, p. 169.

[33] W.-H. Ju and Y. Vardi, "A hybrid high-order markov chain model for computer intrusion detection," Journal of Computational and Graphical Statistics, vol. 10, no. 2, pp. 277–295, 2001.

[34] R. Gwadera, M. Atallah, and W. Szpankowski, "Markov models for identification of significant episodes," in Proceedings of the 2005 SIAM International Conference on Data Mining. SIAM, 2005, pp. 404–414.

[35] A. Markov, "Extension of the limit theorems of probability theory to a sum of variables connected in a chain," 1971.

[36] W.-K. Ching, M. K. Ng, and E. S. Fung, "Higher-order multivariate markov chains and their applications," Linear Algebra and its Applications, vol. 428, no. 2-3, pp. 492–507, 2008.

[37] S. Snodgrass and S. Ontanon, "Learning to generate video game maps using markov models," IEEE Transactions on Computational Intelligence and AI in Games, vol. 9, no. 4, pp. 410–422, 2017.

[38] Q. Lai, H. Yu, H. Chen, and Y. Sun, "Prediction of total power of agricultural machinery using artifical neural networks," in Computing, Control and Industrial Engineering (CCIE), 2010 International Conference on, vol. 2. IEEE, 2010, pp. 394–396.

[39] U. Akdag, M. A. Komur, and A. F. Ozguc, "Estimation of heat transfer in oscillating annular flow using artifical neural networks," Advances in Engineering Software, vol. 40, pp. 864–870, 2009.

[40] M. Shepperd and G. Kadoda, "Comparing software prediction techniques using simulation," IEEE Transactions on Software Engineering, vol. 27, no. 11, pp. 1014–1022, 2001.

[41] N. Hazarika and D. Lowe, "A neural-network extension of the method of analogues for iterated time series prediction," in Neural Networks for Signal Processing VIII, 1998. Proceedings of the 1998 IEEE Signal Processing Society Workshop. IEEE, pp. 458–466.

[42] I. Watson, Applying case-based reasoning: techniques for enterprise systems. Morgan Kaufmann Publishers Inc., 1998.

[43] E. Mendes and N. Mosley, "Further investigation into the use of cbr and stepwise regression to predict development effort for web hypermedia applications," in null. IEEE, 2002, p. 79.

[44] C. K. Riesbeck and R. C. Schank, Inside case-based reasoning. Psychology Press, 2013.

[45] I. Watson, "Case-based reasoning is a methodology not a technology," in *Research and Development in Expert Systems XV*.   Springer, 1999, pp. 213–223.

[46] J. Wen, S. Li, Z. Lin, Y. Hu, and C. Huang, "Systematic literature review of machine learning based software development effort estimation models," *Information and Software Technology*, vol. 54, no. 1, pp. 41–59, 2012.

[47] L. D. Schroeder, D. L. Sjoquist, and P. E. Stephan, *Understanding regression analysis: An introductory guide*.   Sage Publications, 2016, vol. 57.

[48] D. Watson, "Correcting for acquiescent response bias in the absence of a balanced scale: An application to class consciousness," *Sociological Methods & Research*, vol. 21, no. 1, pp. 52–88, 1992.

[49] M. R. Phillips, B. D. McAuliff, M. B. Kovera, and B. L. Cutler, "Double-blind photoarray administration as a safeguard against investigator bias." *Journal of Applied Psychology*, vol. 84, no. 6, p. 940, 1999.

[50] A. Koriat, M. Goldsmith, and A. Pansky, "Toward a psychology of memory accuracy," *Annual review of psychology*, vol. 51, no. 1, pp. 481–537, 2000.