Toward Cooperative Level Generation in Multiplayer Games: A User Study in Overcooked!

In-Chang Baek[1], Tae-Gwan Ha[2], Tae-Hwa Park[2], Kyung-Joong Kim[1,2,*]
[1]*AI Graduate School*
[2]*School of Integrated Technology*
Gwangju Institute of Science and Technology (GIST)
Gwangju, South Korea
{inchang.baek@gm., hataegwan@gm., taehwa-p@gm., kjkim@}gist.ac.kr

***Abstract*—Multiplayer contents play an essential role in extending the lifespan of the game and positively affect the player's experience. However, previous studies on procedural content generation focused on single-player games and few for multiplayer games. Multiplayer games have different ethics and mechanics associated with competition or cooperation. Thus, content designers are concerned about this interaction. In this paper, we propose a new method for generating multiplayer levels that encourage diverse cooperation among game players. Our contributions are summarized as follows: 1) We designed four cooperation patterns usable in controllable generation literature. 2) We proposed a controllable level generator to deploy the proposed patterns in the two-player cooperative cooking game, Overcooked!. 3) We discussed effective methods for leading players' cooperation experience. Consequently, we found that the players' interaction is most fundamental at the multiplayer level. The results of our study will lead to diverse cooperation experiences in future multiplayer game content generation studies.**

***Index Terms*—procedural content generation, genetic algorithm, cooperative, multiplayer**

I. Introduction

Procedural content generation (PCG) studies have shown remarkable approaches in generating single-player game levels. To leverage the effectiveness of automated content generation, several methods have been proposed for generating content with several conditions set by a game designer [1–3]. Presently, studies on controllable PCG have achieved success in placing game resources, e.g., designing a level layout and allocating items. Furthermore, there are several approaches for adjusting the difficulty of a game with respect to the player's level [4–6], and involving the game design theory in traditional PCG approaches [7, 8]. These level generation studies are only related to enhancing a player's game experience; few are considered in multiplayer games.

Several attempts to diversify multiplayer experience in both industry and research have been considered because cooperative mechanisms in multiplayer games lower entry barriers and positively affect the player's experience [9–11]. Notably, there has been a rapid increase in multiplayer game scale in the game industry, and team-based games have emphasized teamwork differently in cross-game design. For instance, a multiplayer online battle arena, such as *League of Legends* and *Dota 2* puts players in several different roles. Similarly, games like *Portal 2* and *Overcooked!* merge players with the same abilities, leading to flexible coordination with their interaction. Different team-playing types and team-building processes using various combinations have been in the spotlight and increased the impact of the esports industry. Additionally, training human-collaborative artificial intelligence [12–14] and developing cooperative environments for these works are promising topics on multiagent research in the game research community [15–17].

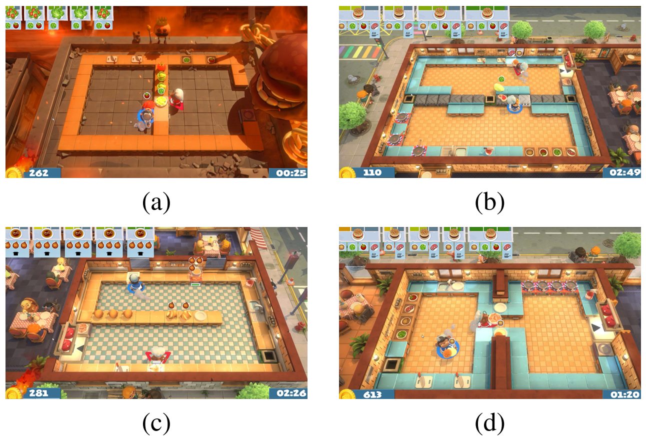


Fig. 1: Game levels from the commercial Overcooked! game.

Compared to single-player games, more elements are considered in designing a multiplayer content generator. Various relationship and interaction issues, such as competition and cooperation applied to multiplayer game design are considered. To solve this problem, research has been conducted to standardize and use core characteristics of cooperation in multiplayer games. These core characteristics of cooperation could be used as team performance measurement [18], cooperative performance metrics [19], and educational tools [20]. Generally, the core elements of cooperation were applied to game mechanisms in the form of characters and levels. Specifically, to induce cooperation in commercial online games, roles are explicitly determined by varying the ability of game characters [19]. These different roles are irreplaceable and coordinate players reliant on each other's ability to solve a challenging problem. Otherwise, a method that implicitly assigns the player role through the game components is employed [11, 21]. From a level generation perspective, the generator controls the movement and resources inside the level, so that the problem cannot be solved by a solo player. That is, the level is designed to intend natural cooperation among players.

Addressing the present PCG research on the multiplayer domain, our contribution can be described as follows: First, we studied core mechanics of cooperative playing and incorporated four methods for inducing cooperation. Then, we presented a controllable level generator for the *Overcooked!* game to deploy the cooperation methods in a specific game by integrating the methods into the mechanism of this game. Finally, we investigated the characteristics of each method using user studies and suggested effective methods for emerging cooperation in multiplayer levels. Our work could be an affordable approach in multiplayer level generation research under the current limitations on measuring cooperation discussed in the following section.

The remainder of the paper is organized as follows: Section II introduces related works and multiplayer game design patterns. Section III presents various cooperation emerging methods. We demonstrate the user study experiment in Section IV. Finally, Section V presents the conclusion and future studies.

II. Background

A. Controllable Generation for Single-Player Games

Controllable generation has been an active research area in the game PCG community. Several methods have been used to generate a game level under some conditions. In the search-based approach, the authors of [2, 22] suggested a physics-based game PCG to build game-level architectures for *AngryBirds*. They presented a parameterized fitness function for a genetic algorithm to control the game difficulty and the stable and feasible levels. For the controllable level generator of [22], the authors of [23] presented a dynamic difficulty adjustment method using agent-based evaluation. Moreover, [24] first addressed PCGRL (PCG via RL), by applying reinforcement learning (RL) to generate game layouts. Recently, the authors of [1] suggested a reward shaping method for generating controllable game content that the in-game features (e.g., distance and happiness of non-player characters) fit a designer's setting. Additionally, the authors of [4] proposed an adversarial RL architecture in which a generator model competes with a solver agent and controls the difficulty in a level from easy to challenging using a factorized parameter. Meanwhile, the authors of [8, 25] designed an objective function on the principle of Koster's *Theory of Fun* [26] to enhance a player's game experience. Given this, studies on controllable PCG are not limited to generating game content according to an author's preferences, thus advancing research in other ways, such as improving individual players' game experience. However, present studies are focused on single-player games such as *GVGAI* [3], *SuperMarioBros* [8], *AngryBirds* [2, 22, 23], *Zelda*, *RollerCoasterTycoon*, and *SimCity* [1]. Therefore, studies on content generation for multiplayer games are at their early stage.

B. Content Generation for Multiplayer Games

Compared to single-player games, additional relationships should be considered when designing a multiplayer game. In single-player games, there is only a relation between a player and non-player objects; however, not only the objects, other players can be an ally or an enemy in multiplayer games. This difference makes the game designer concerned about how to design a method for emerging cooperation between players. To apply a multiplayer game mechanism to present PCG research, finding design patterns representing core characteristics of cooperation is necessary. Thus, we introduce several game design patterns for multiplayer games and discuss the consideration of multiplayer PCG.

Several attempts have been made to assess team cooperation and to design the principles of cooperation in multiplayer games. For example, six principles for solving a common problem in a cooperative multiplayer game and a team performance process to examine the team adaptation of individual members were proposed in [18]. Among these principles, role differentiation induces complementary between players and gives the players individual responsibility for their task. Furthermore, the authors of [19] also suggested six design components (e.g., complementary, synergy, shared goal, and rules for enforcing cooperation) which lead to cooperation in games. To apply these principles in a game, we can design inefficiency in players' movements, limit a player's available resources, and/or use game entities to promote interaction between players. Additionally, five design components for an educational multiplayer game were proposed in [20]. To enhance the play experience (PX), they assigned different abilities (e.g., healing, nuking, tanking) to each player to make them accountable. Compared with single-player games, fun and difficulty in multiplayer games come from the level of team coordination and interaction between players (e.g., conversation) instead of individual players' performance.

Due to the nature of controllable PCG that requires measurable indicators, the abstract feature in multiplayer games, cooperation made it challenging to formalize and evaluate. The difficulty in content generation in multiplayer games comes from the proposed methodology. Recent PCG studies use an objective function to evaluate generated content and search for better candidates or train a generative model. To generate cooperative content using this mechanism, the cooperation between players should be measurable in a heuristic or agent-based testing. Although related studies [12, 15] used planning algorithms for agents in the game, this method pursues near-optimal policy by considering only the effectiveness of a solution. For this reason, it is not an exact method to multiplayer PCG algorithms as a level evaluator when we consider the traditional multiplayer game design patterns. Moreover, several attempts have been made to measure cooperation between players in other domains. The data-driven method [21] requires a large dataset. However, it is not easy to generalize. Additionally, since self-assessment [18] must be conducted with a human on every generated content, using this method in a PCG study is difficult. Accordingly, in this study, we suggest several design methods that can be used in multiplayer games and demonstrate a content generation method using them.

C. A Cooperative Game: Overcooked!

The game "*Overcooked!*"[1] is a two-player cooperative cooking game where every player gets an equal reward for playing. The game has a simple goal, two or more players cook dishes to get a score. The cooking process follows a sequence of delivering ingredients, cooking on a pot, putting it on a dish, and delivering the dish to the serving area. To get a high score, players build cooperation strategies on the principle of divide and conquer. While the players build a strategy, they discuss their roles and movements through conversation and their intuition. Fig. 1 shows the commercial *Overcooked!* game. The levels are designed to promote cooperation between players. The cooperation methods differ from the layout form of a level and placement of game objects. In other words, different cooperation settings can be conveyed with different level layouts and placement of the game objects. Moreover, because the game is a pure cooperative game, players only have a common goal. For this reason, it is an appropriate environment to emerge cooperation and evaluate our generated levels. This game has been widely used in the multiagent domain, e.g., human-robot coordination [12, 27] and inference mental-state [17].

However, there are a few studies on generating multiplayer game levels for *Overcooked!*. To generate cooperative levels that are stylistically similar to hand-designed levels, the authors of [15] investigated the mixed-integer linear programming (MIP) method for searching latent spaces of generative adversarial networks. They assumed that the team fluency metric is each player's contribution to the team and proper workload assignment; team fluency is the coordinated meshing of joint activities in teamwork. To evaluate and control the team fluency, they specified coordination behaviors measurable by considering the distribution of workload on each other. Here, the differences between the numbers of subtasks (dishes, ingredients, and orders) handled by each player were used as indicators to measure the robot's contribution to the team. However, to properly assess the distribution of workload, the constraints that both players must reach all subtasks degrade the layout diversity. They dealt with only one shared kitchen and did not consider the game experience from level layout and role assignment. In other perspectives, the authors in [16] sampled solvable levels from heuristically generated levels. They investigated how the diversity of environment affects the generalization of multiagent RL. These studies focusing on just generating solvable levels did not consider the cooperation design patterns for human game experience.

There are several ways to design the cooperation method in the *Overcooked!* game. The key to designing this game is giving responsibility to each player and leading them to set their individual goals. To solve this game efficiently, players plan their movement and divide their workload, observing their available resources and distances (*situation assessment* and *plan formation* in [18]). On these principles, the game designer leads the cooperation by **(1) implicitly inducing the discussion** (e.g., meshing and long way), as shown in Figs. 1(c) and (d). Specifically, the level designer gives players specific roles, **(2) explicitly giving a responsibility**, as shown in Figs. 1(a) and (b). We denote these approaches as implicit and explicit methods, respectively. The difference between these methods is that cooperation is nonmandatory for solving the level in the implicit method, whereas it is required in the explicit method. The design methods for multiplayer games diversify the cooperation strategies and present various cooperation PXs. In this study, we suggest several cooperation designs for level generation and measure the actual effect on the PX perspective. Following the controllable PCG literature, we suggest several parameters for generating cooperative levels. We investigate how these levels affect players' emerging coordination in a multiplayer game.

[1]https://www.team17.com/games/overcooked

III. Cooperative Level Generation

A. The Cooperation Emerging Methods

As stated in previous section, we suggested implicit and explicit methods for designing cooperative levels. All resources for cooking a dish were evenly spread in a map named *Messy* after generating an unorganized level. Here, the levels do unintentionally address the players to cooperate. Therefore, the tasks are spread widely, making it difficult for players to decide their tasks and strategies. Subsequently, the strategies are owned to the players' preferences and playing level.

Furthermore, there are two implicit (*IM-*) ways to address cooperation. In implicit methods, cooperation is nonmandatory, but the level designer leads players to recognize that dividing their workload or traffic effectively solves the problem. First, *IM1* gives the players a shared kitchen such that tasks are unevenly placed. Note that the players easily collide when doing the same task. Thus, team fluency is poor when they collide frequently. Also, the players are led to dividing their tasks or traffic line. Second, *IM2* restricts the players' traveling with a wall. However, the tasks are spread widely similar to *Messy*; each player can access at least one resource for each task. Although the players could solve the problem individually, the wall induces inefficient traffic when traveling all tasks alone. For that reason, the players could selectively process their tasks to build an optimal strategy.

Meanwhile, there is a way to give their role explicitly (*EX*), i.e., place the players in separate kitchens and make them forcibly cooperate, passing game items. The proposed methods are summarized as follow:

- **Messy**: Every task is spread near-randomly in a shared kitchen. Players have to decide their strategy fully.
- **IM1**: Same tasks are grouped and spread in a shared kitchen. It is easier to collide between players than *Messy*.
- **IM2**: A wall separates the players' traffic. However, cooperation is nonmandatory for serving a dish.

- **EX**: A wall separates the players' traffic, dividing tasks by the wall. Cooperation is mandatory.

To deploy four cooperation methods, we designed a controllable generator with a parameterized genetic algorithm. The method is widely used in controllable PCG to manipulate a level's characteristic [2, 22], even on difficulty [23]. This can be achieved by parameterizing a fitness function and giving some variables representing features in a level. We implemented our generator on the clone game proposed in [12].

B. Controllable Generation for Overcooked!

Level Representation: Genetic algorithm is an evolutionary searching method to find optimal chromosomes with the genetic procedure: selection, crossover, and mutation. To represent grid-like game levels as chromosomes, tabular representation has been used to encode game tiles into genes. From Fig. 2, we define six-game blocks into numbers and encode the game layout to numerical matrix. A game level is represented using a 10 by 10 size two-dimensional (2D) array chromosome. The chromosome can be loaded on the environment using an auxiliary code, which converts the array into a text file.

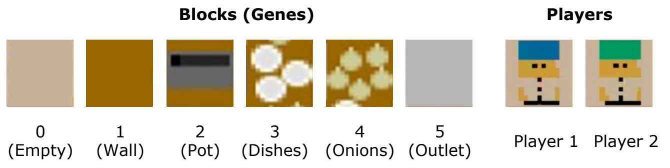
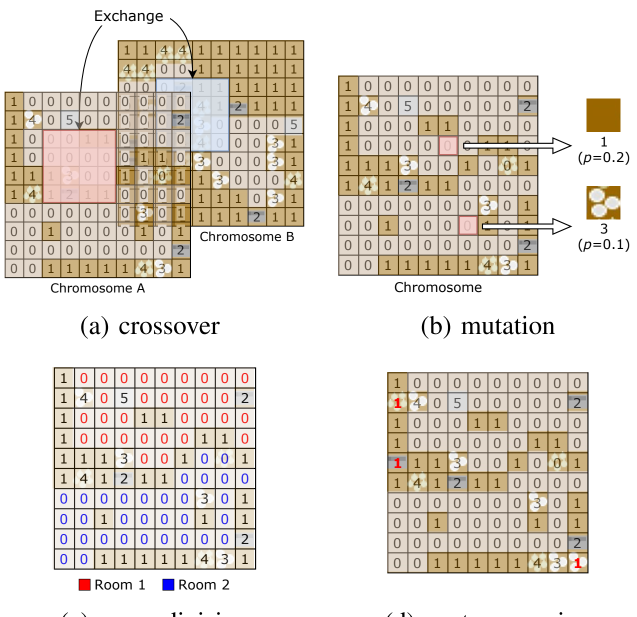


Fig. 2: Six blocks to build a level. We define the genotype into an integer value ranging from 0-5.

Population Initialization: A carefully designed initialization remarkably reduces the searching space in this domain (e.g., *AngryBirds* [2]). We also use a population initializer that selects game blocks with a predefined rule. Instead of placing previously described six blocks using an uniform random distribution, we use different probabilities that weight how much each block is used – $[0.7, 0.1, 0.05, 0.05, 0.05, 0.05]$. The probability sets how many each block is used in general. This method benefits in searching for optimal levels robustly.

Crossover and Mutation: To find more optimal chromosomes, crossover and mutation are used as the key procedure when enhancing a population. Figs. 3(a) and (b) show the example of the genetic process in visual. Thus, game levels are represented using a 2D array, and we use a tile-based one-point crossover method. Once we sample two parents for crossover, the 4 by 4 size tiles are randomly sampled for each parent. Next, the tiles are exchanged for the sampled location of each parent. In mutation, the random-resetting method is used to replace a gene with a random block. Here, the random block is sampled with a weighted probability for effective searching, according to the following probabilities – $[0.4, 0.2, 0.1, 0.1, 0.1, 0.1]$.

Fitness Function: There are two strategy models for generating multiplayer game levels. One is designing the layout that affects the player's route directly. The other is placing



(a) crossover (b) mutation

(c) room division (d) post-processing

Fig. 3: (a) and (b) show the crossover and mutation examples on the genetic algorithm. (c) shows how the rooms are separated and counted in the fitness function. (d) shows the post-processing that removes unusable resources.

the game resources in some positions that make the players decide their route indirectly. To convey these models, we build the parameterized fitness function, following to the controllable PCG literature. Therefore, we design several terms for controlling the level's layout and task placement and to evaluate a chromosome, as expressed in Eqs. 1-6.

$$f_{RC} = \mid c - C \mid \tag{1}$$

$$f_{RS} = \begin{cases} \sigma(s_1, s_2, \cdots, s_c), & \text{if } c > 1 \\ 1, & \text{otherwise} \end{cases} \tag{2}$$

Eqs. 1-2 are related to a level's layout. One way of generating a cooperative level is to design a game level into separate rooms for interaction between players. This design should be such that the room size is enough to get around. To satisfy these conditions, we calculate the room count and size as shown in Fig. 3(c). The room count and size are measured by separated spaces and the empty block in each space, respectively. Then, we design fitness for room count f_{RC} to current count c close to the target count C; f_{RS} lowers variance of room sizes $(s_1, s_2, \cdots, s_c)$ into zero, making each room size equal. This term alleviates the unequal distribution in the size of play areas.

$$f_{BC} = \frac{1}{T} \cdot \sum_{t=1}^{T} \mid R(T_t) - B_{t+1} \mid \tag{3}$$

From Fig. 4, let task (T) and resource (R) represent the game mechanism. A task (T_t) represents a procedure for cooking a dish to tasks (e.g., four action sequences in Section II-C); the number (t) is allocated by the sequence order. The resource list $(R(T_t))$ represents the available block list for

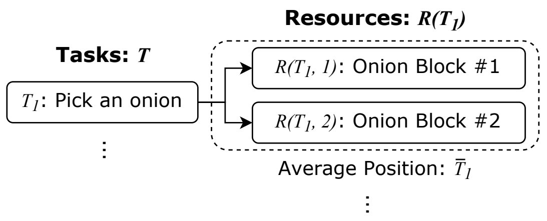

Resources: $R(T_1)$
Tasks: T
T_1: Pick an onion
$R(T_1, 1)$: Onion Block #1
$R(T_1, 2)$: Onion Block #2
Average Position: $\bar{T}_1$

Fig. 4: Task and resource diagram with the notations.

finishing a task (T_t). Moreover, each block in the resource list is represented as $R(T_t, r)$; r is the index in the list. Thus, tasks consist of resource blocks for finishing them. Note that the resources in a task ($R(T_t)$) is intuitively measured by the blocks in a level. Then, the f_{BC} (Eq. 3) is designed in a similar matter to f_{RC} and makes the quantity of the resources in each task ($R(T_t)$) to fit our block count parameters B_i, where i is the genotype for a block. Tasks 1-4 (T_{1-4}) use genotype (B_{2-5}) as resources, respectively. Moreover, this term is regularized by dividing it into four tasks (T) so that f_{BC} do not overwhelm other terms when the number of tasks increases.

$$f_{RD} = \sum_{t=1}^{T} \left(\frac{1}{R(t)} \sum_{r=1}^{R(t)-1} D(R(t,r), R(t,r+1)) \right) \tag{4}$$

$$f_{TD} = -\sum_{t=1}^{T-1} D(\bar{T}_t, \bar{T}_{t+1}) \tag{5}$$

Eqs. 4-5 are related to the task and resource placements. Observe that f_{RD} determines the concentration of resources in a task and f_{TD} determines the distances between tasks. f_{RD} (Eq. 4) measures the distances between resources in each task, locates them close if w_{RD} is positive. It engages on whether the players collide when they are doing the same task for a shared kitchen. f_{TD} (Eq. 5) measures the abstracted position of the tasks ($\bar{T}_t$), by calculating the average position of the resources in T_t. Then, it sums the distances between $\bar{T}_t$s, locates them far from each other if w_{TD} is positive. As an effect, it induces inefficient traffic lines or separation of tasks in the next section. Consequently, all terms with weight parameters w are summed to balance them. For the detailed implementation, refer to this code[2].

$$\begin{aligned} Fitness = & -(w_{RC} \cdot f_{RC} + w_{RS} \cdot f_{RS} \\ & + w_{BC} \cdot f_{BC} + w_{RD} \cdot f_{RD} + w_{TD} \cdot f_{TD}) \end{aligned} \tag{6}$$

C. Level Generation Experiment

We used our generator to generate these cooperative levels, mentioned in the previous controllable generation. The parameters for generating each level are described in Table I. The level generation experiment is conducted according to the following sequence: (1) Initialize a population with 100 chromosomes. Each chromosome has 100 genes and is initialized with the weighted random function. (2) Repeat the evolutionary process during 500 generations with our fitness function in the previous section. (3) Sample the best chromosome in the population, and save it after post-processing in Fig. 3(d). We repeat the above procedure for each cooperation emerging method. Next, we set the probabilities of crossover and mutation as 0.9 and 0.01, respectively. Roulette selection and linearly incremental elitism (0 to 0.5) were used for genetic algorithm hyper-parameters. We generated five levels for each cooperation method, as shown in Fig. 5.

TABLE I: Parameters used to generate levels using the cooperation emerging methods. The values are passed into the fitness function.

Parameter	Value			
	Messy	**IM1**	**IM2**	**EX**
B_5	1	1	2	1
C	1	1	2	2
w_{RD}	0	0.3	0	0.3
w_{TD}	0	0	0	0.09
B_{2-4}	3			
w_{RC}	1.5			
w_{RS}	0.2			
w_{BC}	3.0			

IV. Experiment and Result

A. User Study Setup

Participants: We recruited 22 participants (ages between 21 and 31, M=26.00, SD=3.06) in the local students and online game community. Most participants (86.4%) have never played this game or have played it for less than an hour. Each participant was compensated $50 for completing the study, which lasted for about 90 min, including tutorial and rest.

Procedure: The experiment was conducted in the order of (1) preliminary survey, (2) tutorial, (3) play and post-game survey for each session (method), and (4) interview after all games. Two participants participated in each experiment as a team. First, both participants conducted a preliminary survey asking about the player's cooperative game experiences. In the tutorial, each participant and the researcher played a level unrelated to the proposed methods to understand basic control and cooperation tasks. Then, the team plays a session in a shuffled order using the balanced Latin square method to reduce learning effects. For each session, two levels are sampled and played among the five seeds in Fig. 5, and the play duration is limited to 2 min for each level. The in-game conversation between players is not controlled. Thus, players were allowed to discuss the level naturally. After a session, each player individually conducted the game experience questionnaire (GEQ) survey [28]. Then, we conducted an open questionnaire about their strategy, the number of strategy changes, and when their strategy was decided. Finally, after all sessions, we asked participants what level they wanted to play again.

Survey: GEQ has several modules for surveying a game. Here, we used the core and social presence modules in our user study experiment. The core and social modules consist of seven and three indicators, respectively, each having detailed

[2]https://github.com/bic4907/Cooperative-Overcooked-PCG

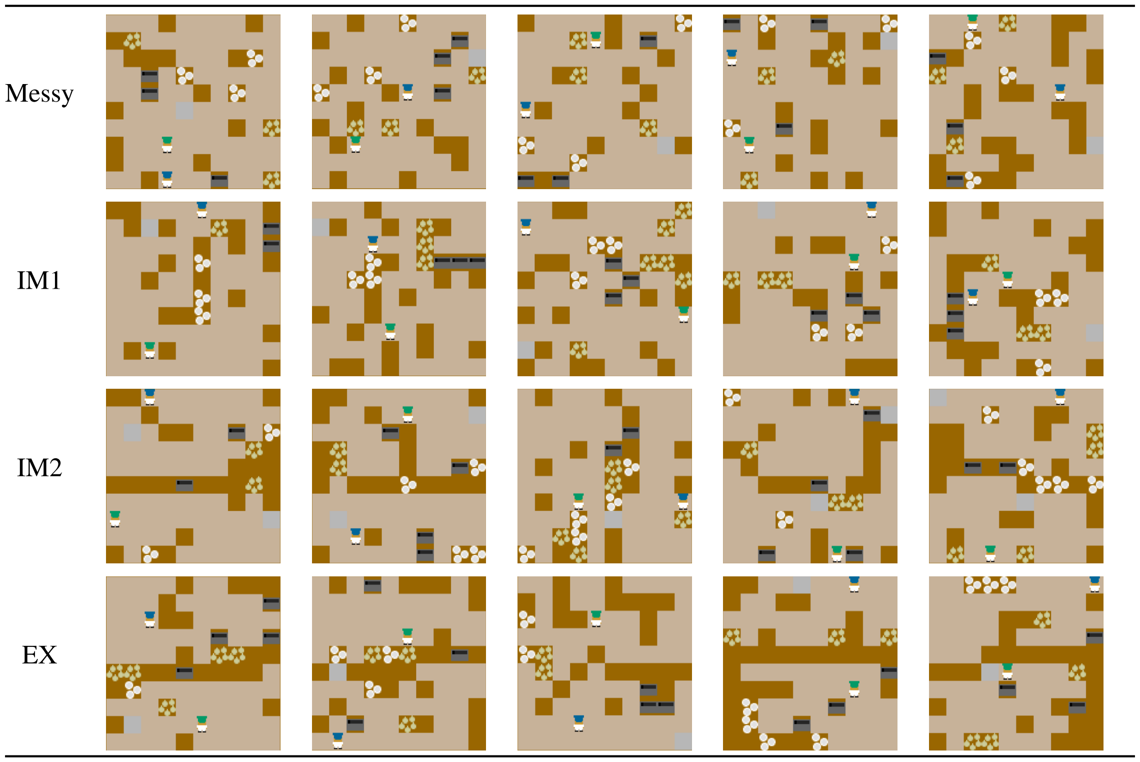


Fig. 5: Example of generated levels for each cooperation method. We use these levels on the user study experiment by loading on the Overcooked! environment. Each player is randomly placed in a shared kitchen and placed in each room in a separate kitchen.

intuitive 3-6 questions and measured according to the scoring guide. The core and social modules mainly ask players about their individual PX and interactions (e.g., behavioral, rapport), respectively. The last two sessions (among four) were used in the analysis. Most participants felt confused initially. However, meaningful clues appeared after they have adapted to the task. It is because most participants have little experience with the game. They needed time for adapting to the game despite the tutorial. We conducted the post-game survey and the overall results are summarized in Table II. For the statistical comparison, we selectively conducted a one-way ANOVA or Kruskal-Wallis rank-sum test according to the data normality. Then, the post-hoc test was conducted using Tukey HSD and Conover test, respectively.

There is a consideration when we apply this questionnaire in this game. The original purpose of the Psychological Negative is to measure the dissension between players. However, this intent changed because of the characteristic of this game, i.e., pure cooperation game. The reason is that Q.11-12 asked each player how much they influenced or were influenced by the other's mood. These questions affected this indicator. The players understood them as the connectivity with others in a neutral manner, rather than screws. Accordingly, this indicator measures how much interaction was found instead of the original meaning.

Hypothesis: To investigate how the proposed methods affected players' strategies and PX, we built the following research question (RQ) to check through.

- **RQ1**: What are the effective methods for emerging cooperation?
- **RQ2**: How does the characteristic of levels affect the player experience?

B. Result and Discussion

RQ1. What are the effective methods for emerging cooperation?

We suggest two directions for designing a cooperation level. *IM1* and *EX* showed higher trends than the others in the challenge and categories related to interaction. Through the case study with statistical differences, we found useful patterns for cooperation levels. To check the statistical differences visually, see Fig. 6.

First, giving dependent tasks to players leads to a high connectivity experience. A significant difference was shown between *EX* and *IM2* in Psychological Negative ($p < 0.01$, $H = 11.645$). The major difference between both levels is the necessity of cooperation. A player's work was affected by the other. Both players have their explicit roles and depended on the task sequence. Some players told us that they were satisfied by clear responsibility and chances for interaction, *"I felt content because role division was clear and teamwork was fluent."* (P9), *"There were lots of chances for communication and cooperation, and I liked it."* (P10). Moreover, several conversations followed when they discussed the timing of passing an item and their requirements for processing a task.

TABLE II: Descriptive statistics, variance analysis on GEQ (Core and Social module). The arrow marks indicate the direction of the positive effect in cooperation. For Psychological Negative, refer to the survey consideration in Section IV-A.

Questionnaire	Mean (±SD)				ANOVA		Kruskal-Wallis	
	Messy	IM1	IM2	EX	F	p	H	p
GEQ - Core Module								
Competence	4.28 (0.82)	3.64 (0.87)	3.96 (0.90)	4.20 (0.77)	-	-	5.154	.160
Sensory and Imaginative Immersion	3.40 (0.64)	3.21 (0.87)	3.01 (0.97)	3.41 (0.78)	.568	.639	-	-
Flow	3.86 (0.82)	4.04 (0.53)	3.58 (0.92)	3.98 (0.61)	.780	.511	-	-
Tension/Annoyance	1.52 (0.82)	2.10 (0.77)	1.43 (0.52)	1.66 (1.00)	-	-	5.457	.141
Challenge	2.70 (0.91)	3.74 (0.59)	3.12 (0.79)	3.41 (0.91)	3.180	.034*	-	-
Negative affect	1.60 (0.75)	2.07 (0.77)	1.90 (0.85)	1.58 (0.61)	-	-	2.896	.407
Positive affect	4.36 (0.80)	3.90 (0.75)	4.14 (0.76)	4.38 (0.51)	-	-	3.326	.343
GEQ - Social Module								
Psychological Empathy (↑)	3.79 (0.87)	4.23 (0.57)	3.80 (0.96)	4.30 (0.62)	1.404	.255	-	-
Psychological Negative (↑)	2.68 (0.53)	3.02 (0.70)	2.10 (0.88)	3.22 (0.56)	-	-	11.645	.008**
Behavioural Involvement (↑)	4.15 (0.60)	4.41 (0.62)	3.60 (1.21)	4.61 (0.44)	-	-	6.989	.072

significant at $^*p < .05$, $^{**}p < .01$

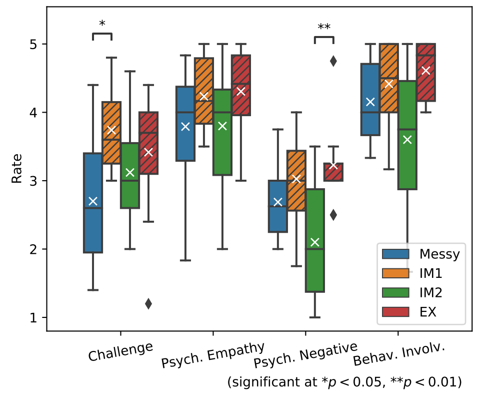


Fig. 6: Statistical comparison on Challenge, Psychological Empathy/Negative and Behavioral Involvement in GEQ. The white cross marks denote the average, and the dashed bar denotes our suggestions.

Thus, giving a clear responsibility to each player is a major design method in multiplayer games as mentioned in [20].

Second, the collision of chefs on the movements made the players feel the level positively challenging. The statistical difference was shown between *Messy* and *IM1* in Challenge ($p < 0.05$, $F = 3.180$). The collision made it difficult for players to decide their strategy, thereby leading to the most strategy changes, as shown in Table III. The reason follows from the difficulty to predict collisions, and this induced many trials and errors. Half of the participants chose *IM1* as they mostly want to try again. Some participants told us that they wanted to try another strategy, *"It was interesting that the collisions in movements made the cooperation strategy frequently changing. I want to try again because I think of a more efficient way."* (P1), *"I felt it was difficult to find an optimal strategy. However, I can think out a better one if I try it again."* (P19). Moreover, in the trial and error process, they actively discussed their movement strategies, leading to a high interaction experience along with *EX*.

Meanwhile, *Messy* and *IM2* show relatively low rates in interaction categories because they had a weak necessity for cooperation. The players responded that they decided to cooperate relatively late in these methods, even individually playing in some cases (*Messy*: 1 team, *IM2*: 2 teams; zero for others).

As such, there is a valuable finding from the results. Authors of [15] attempted to enhance the team fluency by dividing the cooking tasks equally since the level was evaluated by robots and their performance was equal. Thus, the fun that came from interactions is not considered. Our results show that the cooperative level must be considered for the interactions between players in the future. This is a new direction in a multiplayer game level generation.

RQ2. How does the characteristic of levels affect the player experience?

The major characteristic was the number of possible strategies provided to players. In the divided kitchen levels (*EX*, *IM2*), players were able to determine optimal strategies faster than in the shared kitchen levels (*IM1*, *Messy*) because the divided kitchen level had fewer cases where players considered others. The open question statistics in Table III show that dividing space helps the player to adapt to the game and easily build an optimal strategy.

TABLE III: Average number of strategy changes during play.

	Messy	**IM1**	**IM2**	**EX**
Mean (±SD)	1.45 (0.93)	1.72 (0.90)	1.00 (0.63)	0.81 (0.75)

However, the level was unsuccessful in some respects. Some participants told us that these levels help them adapt to the task and were comfortable to playing *"It was clear to divide the role and I felt immersed in cooperative play."* (P4, P12), *"I felt this level was a good stage to aim for the highest score."* (P13). Conversely, other participants said that the strategy was coercive and repetitive *"This level was not enjoyable because it had fewer strategies than other levels."* (P3), *"Since the tasks were forcibly divided on the level, it was impossible to play with my routine."* (P8), *"The optimal strategy was quickly built, so I could not share meaningful communication with the another player."* (P14).

Consequently, a player with low expertise in the game felt confused and dissatisfied with levels that had several strategies. Here, reducing the number of strategies that players can build, such as *EX* and *IM2*, was most efficient. The restriction on chefs' movements (i.e., divided kitchen) limits the players' available tasks and traveling; it reduces the searching space and burden of players. Otherwise, when players became experts at the game, there were positive assessments to *IM1* and *Messy*. The levels provided the players more choices for their strategy, leading to relatively diverse communication to build a strategy. Thus, we suggest providing a level that has intuitive or selective strategies depending on the players' expertise.

V. Conclusion and Future Work

In this paper, we introduced new directions for generating cooperation levels. To validate the proposed method, we used a parameterized genetic algorithm to generate four kinds of methods with different cooperation experiences. We applied these methods to the *Overcooked!* game and conducted the user study to investigate actual players' PXs. Furthermore, we suggested two methods for emerging cooperation effectively; the key is leading conversations with game levels, e.g., giving a dependent role to players and inducing collisions in their movements. Our result implies that team fluency is not the only key for multiplayer level generation; we will additionally consider several interactions between players in the future.

Empirically, we observed the performance gaps between team players (e.g., one is an expert and the other is at a novice level), which affected the team fluency. In the future, adaptive distribution of workload that considers asymmetric player levels could be valuable work. Additionally, we will investigate the relationship between the players' behavior (e.g., verbal and actions) and the generated content.

VI. Acknowledgment

This research was supported by the National Research Foundation of Korea (NRF) grant funded by the MSIT(2021R1A4A1030075). The authors would like to thank Youjin Choi for the constructive discussion on the user study.
*corresponding author

References

[1] S. Earle, M. Edwards, A. Khalifa, P. Bontrager, and J. Togelius, "Learning controllable content generators," in *2021 IEEE Conference on Games (CoG)*, IEEE, 2021, pp. 1–9.

[2] L. N. Ferreira and C. F. M. Toledo, "Tanager: A generator of feasible and engaging levels for angry birds," *IEEE Transactions on Games*, vol. 10, no. 3, pp. 304–316, 2017.

[3] A. Sarkar and S. Cooper, "Dungeon and platformer level blending and generation using conditional vaes," in *2021 IEEE Conference on Games (CoG)*, IEEE, 2021, pp. 1–8.

[4] L. Gisslén, A. Eakins, C. Gordillo, J. Bergdahl, and K. Tollmar, "Adversarial reinforcement learning for procedural content generation," in *2021 IEEE Conference on Games (CoG)*, IEEE, 2021, pp. 1–8.

[5] M. Kaidan, T. Harada, C. Y. Chu, and R. Thawonmas, "Procedural generation of angry birds levels with adjustable difficulty," in *2016 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, 2016, pp. 1311–1316.

[6] D.-F. H. Adrian and S.-G. C. A. Luisa, "An approach to level design using procedural content generation and difficulty curves," in *2013 IEEE Conference on Computational Inteligence in Games (CIG)*, IEEE, 2013, pp. 1–8.

[7] T. Shu, J. Liu, and G. N. Yannakakis, "Experience-driven pcg via reinforcement learning: A super mario bros study," in *2021 IEEE Conference on Games (CoG)*, IEEE, 2021, pp. 1–9.

[8] G. N. Yannakakis and J. Togelius, "Experience-driven procedural content generation," *IEEE Transactions on Affective Computing*, vol. 2, no. 3, pp. 147–161, 2011.

[9] M. Schmierbach, Q. Xu, A. Oeldorf-Hirsch, and F. E. Dardis, "Electronic friend or virtual foe: Exploring the role of competitive and cooperative multiplayer video game modes in fostering enjoyment," *Media Psychology*, vol. 15, no. 3, pp. 356–371, 2012.

[10] A. Beznosyk, P. Quax, K. Coninx, and W. Lamotte, "The influence of cooperative game design patterns for remote play on player experience," in *Proceedings of the 10th asia pacific conference on Computer human interaction*, 2012, pp. 11–20.

[11] J. B. Rocha, S. Mascarenhas, and R. Prada, "Game mechanics for cooperative games," *ZON Digital Games 2008*, pp. 72–80, 2008.

[12] M. Carroll *et al.*, "On the utility of learning about humans for human-ai coordination," *Advances in Neural Information Processing Systems*, vol. 32, pp. 5174–5185, 2019.

[13] B. Cui, H. Hu, L. Pineda, and J. Foerster, "K-level reasoning for zero-shot coordination in hanabi," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[14] A. Lupu, B. Cui, H. Hu, and J. Foerster, "Trajectory diversity for zero-shot coordination," in *International Conference on Machine Learning*, PMLR, 2021, pp. 7204–7213.

[15] M. Fontaine, Y.-C. Hsu, Y. Zhang, B. Tjanaka, and S. Nikolaidis, "On the Importance of Environments in Human-Robot Coordination," in *Proceedings of Robotics: Science and Systems*, Virtual, 2021. DOI: 10.15607/RSS.2021.XVII.038.

[16] K. R. McKee, J. Z. Leibo, C. Beattie, and R. Everett, "Quantifying environment and population diversity in multi-agent reinforcement learning," *arXiv preprint arXiv:2102.08370*, 2021.

[17] R. E. Wang, S. A. Wu, J. A. Evans, J. B. Tenenbaum, D. C. Parkes, and M. Kleiman-Weiner, "Too many cooks: Coordinating multi-agent collaboration through inverse planning," in *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, 2020.

[18] M. A. Rosen, W. L. Bedwell, J. L. Wildman, B. A. Fritzsche, E. Salas, and C. S. Burke, "Managing adaptive performance in teams: Guiding principles and behavioral markers for measurement," *Human resource management review*, vol. 21, no. 2, pp. 107–122, 2011.

[19] M. Seif El-Nasr *et al.*, "Understanding and evaluating cooperative games," in *Proceedings of the SIGCHI conference on human factors in computing systems*, 2010, pp. 253–262.

[20] N. P. Zea, J. L. G. Sánchez, F. L. Gutiérrez, M. J. Cabrera, and P. Paderewski, "Design of educational multiplayer videogames: A vision from collaborative learning," *Advances in Engineering Software*, vol. 40, no. 12, pp. 1251–1260, 2009.

[21] Z. O. Toups, J. Hammer, W. A. Hamilton, A. Jarrah, W. Graves, and O. Garretson, "A framework for cooperative communication game mechanics from grounded theory," in *Proceedings of the first ACM SIGCHI annual symposium on Computer-human interaction in play*, 2014, pp. 257–266.

[22] M. Stephenson and J. Renz, "Generating varied, stable and solvable levels for angry birds style physics games," in *2017 IEEE Conference on Computational Intelligence and Games (CIG)*, IEEE, 2017, pp. 288–295.

[23] M. Stephenson and J. Renz, "Agent-based adaptive level generation for dynamic difficulty adjustment in angry birds," in *Workshop on Games and Simulations for Artificial Intelligence at AAAI-2019*, Honolulu, HI, USA, 2019, pp. 1–8.

[24] A. Khalifa, P. Bontrager, S. Earle, and J. Togelius, "Pcgrl: Procedural content generation via reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 16, 2020, pp. 95–101.

[25] J. Togelius and J. Schmidhuber, "An experiment in automatic game design," in *2008 IEEE Symposium On Computational Intelligence and Games*, Citeseer, 2008, pp. 111–118.

[26] R. Koster, *Theory of fun for game design*. "O'Reilly Media, Inc.", 2013.

[27] R. Zhao *et al.*, "Maximum entropy population based training for zero-shot human-ai coordination," *arXiv preprint arXiv:2112.11701*, 2021.

[28] W. A. IJsselsteijn, Y. A. de Kort, and K. Poels, "The game experience questionnaire," *Eindhoven: Technische Universiteit Eindhoven*, vol. 46, no. 1, 2013.