American Journal of Engineering Research (AJER)
e-ISSN : 2320-0847 p-ISSN : 2320-0936
Volume-3, Issue-9, pp-73-86
www.ajer.org
Research Paper Open Access

# A Systematic Exploration of Mutation Space in a Hybridized Interactive Evolutionary Programming for Mobile Game Programming

Jia Hui Ong[1], Jason Teo[2]
[1](Evolutionary Computing Lab, Universiti Malaysia Sabah, Malaysia)
[2]Evolutionary Computing Lab, Universiti Malaysia Sabah, Malaysia))

***ABSTRACT :*** *In this study, a systematic exploration of mutation space in interactive evolutionary programming was conducted to investigate the effects of the game synthesis process using different mutation rates. Evolutionary programming is the core Evolutionary Algorithm (EA) used in this study where it is hybridized with Interactive Evolutionary Algorithm (IEA) to generate different rulesets that was played on a custom arcade-type mobile game. The experiment was initially conducted by utilizing different mutation rates of 10, 20, 30, 40, 50, 60, 70, 80, and 90 percent. From the optimization results obtained, the single best individual was selected from each mutation rate to further analyze its quality. It was discovered that higher mutation rates were able to yield faster and better solutions and lower mutation rates generally yielded results that were below average.*

***KEYWORDS :*** *Mutation space, Evolutionary Programming (EP), Interactive Evolutionary Algorithm (IEA), mobile games, arcade-type game*

## I. INTRODUCTION

EAs are optimization algorithms with operational processes that are inspired by nature. There are four different classes of EAs which are Genetic Algorithms (GA), Evolutionary Programming (EP), Evolution Strategies (ES), and Genetic programming (GP) [10].Interactive Evolution Algorithms (IEAs) are a branch of EAs where it uses human users to evaluate the quality of the individual solutions [9] as opposed to the traditional EAs, where the quality of the solutions are based on mathematical formulas and objective calculations that relate explicitly to the problem being solved. Music [6], games [7], graphical arts [8], are among problem domains that have used IEAs as the evaluation paradigm to solve the optimization problem. The major operating systems for smartphones are Apple iPhone Operating Systems (iOS), Android OS, Palm OS, Blackberry OS, and Microsoft Windows Mobile. The Android OS open features has enabled different device manufactures to use it as their device's operating systems. Moreover, Android has also opened up its resources for applications developer to develop applications with zero to a small minimal fee if they decided to post their application into the Android market. Hence, this was the motivation for this investigation to develop the custom arcade-type game using the Android OS platform.

To investigate the effects from the usage of different mutation rates, an Android mobile game was created and incorporated with the hybridized Evolutionary Programming (EP) with Interactive Evolutionary Algorithm (IEA) method. The time needed for the optimization result to converge [9] is one of the primary concerns in IEA due to its effect on users' fatigue. Thus, by searching for a suitable mutation rate that can yield a faster convergence, or in this case, the identification of a better game rule set, is the main objective in this paper. By identifying a suitable mutation rate, the time needed to get a good quality of novel rules set for the games is decreased and hence it will lower the users' fatigue level as well since users' fatigue level is dependent on the time invested in the interactive evolution process [9].The organization of this paper is as follows. Section II draws out the methods that we have used in this study, a more in-depth explanation of the game mechanism and how EP and IEA are implemented into the game. Section III describes the experimental setup that we have used and the results and discussion will be given in Section IV. We will conclude our study and discuss some future work recommendations in the last section.

## II. METHOD

Procedural content generation (PCG) is a method that has been used to automatically generate game contents. Contents that are involved here does not count the creation of artificial intelligence for non-player character (NPC) [3] but it is more on the terrain, stories, maps, and others elements that made up the game. Studies have been done using PCG on generating platform levels [5] and even some used it to generate maps in a large scales game like Real-time strategy (RTS) game genre [3]. Togelius and Schmidhuber [4] had conducted a study that involved generating a game rules instead of the environments. This has given us the idea to create a game that contains no rules and hence letting the PCG to work on the rules generations.

**Game Design :** This game is created to be able to run on Android OS 2.2, the screen size of the game is set to fit in a HVGA mobile display with a dimension of 430 x 320 pixels. The game is built upon a few components such as elements, walls, collision, and scoring. Below are the details of elements and walls as follow.

- Elements
  - Red elements
  - Blue elements
  - Green elements
  - Cyan elements
  - Yellow elements
- Walls
  - 20 x 320 pixels upper and lower border
  - 430 x 10 pixels left and right border
  - 30 x 30 pixels of square walls

Each element is in a round image with their respective colors, and the size of the image is 30 x 30 pixels. The yellow element is used by the users to navigate in the game environment. As for the position of each element is place randomly at the beginning of the game except for yellow element's position is fixed in the center of the environment. Walls served as a restriction for all the elements, upper, lower, left, and right border walls will restrict elements and player from moving out from the game environment. The 30 white square walls will be place as shown in Fig. 1 it will form a simple moving obstacle. Fig. 2 shows how these elements and walls placement looks like when they are place together into a mobile environment.

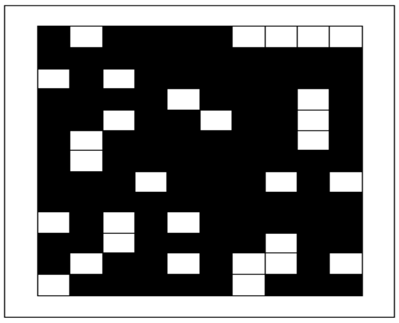


Figure. 1 Walls Placements

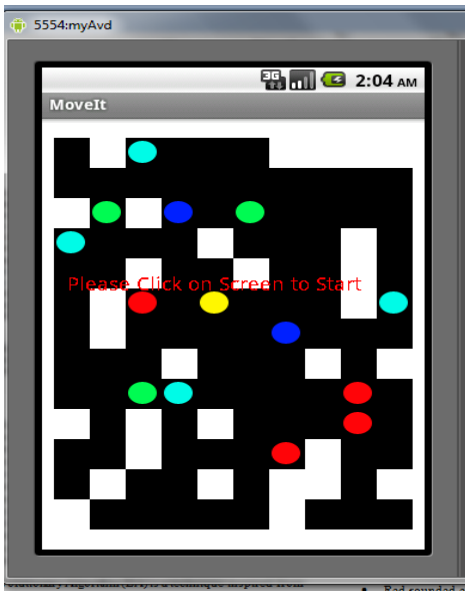


Figure. 2 Elements Wall Placement in Mobile Environment

Each element's movement has been set accordingly where red and blue will be static while green element can only move in vertical directions and cyan elements can only move in vertical directions. Table 1 shows the overall movement for the elements in the game.

Table 1 Elements Movement.

| Elements | Movement |
|---|---|
| Yellow | Vertical and horizontal |
| Red | Static |
| Blue | Static |
| Green | Vertical |
| Cyan | Horizontal |

Moving to the collision component, there are three events that might occur after each elements collide with each other.

- None (no effect) – 0
- Death (elements is deleted from the environment) – 1
- Teleports (elements get teleport back to a locations) – 2

Notice that the number at the end of 0, 1, and 2 it represents the effect in our chromosome. In order for these collisions to take effect, we have structured a collision effect table that will enable a lookup for each collision that happens and hence giving the proper effects that associate with it. Table 2 shows the collision structure that we have created.

Table 2 Collision Effect.

| **Elements** | **Yellow** | **Red** | **Blue** | **Green** | **Cyan** |
|---|---|---|---|---|---|
| **Yellow** | ■ | C1 | C2 | C3 | C4 |
| **Red.** | C1 | ■ | ■ | C5 | C6 |
| **Blue** | C2 | ■ | ■ | C7 | C8 |
| **Green** | C3 | C5 | C7 | ■ | C9 |
| **Cyan** | C4 | C6 | C8 | C9 | ■ |

C1: Yellow and Red element collision

C2: Yellow and Blue element collision

C3: Yellow and Green element collision

C4: Yellow and Cyan element collision

C5: Green and Red element collision

C6: Cyan and Red element collision

C7: Green and Blue element collision

C8: Cyan and Blue element collision

C9: Green and Cyan element collision

The black square represent collision effect that has been taken out, the reason that we took it out is due to the movement of the elements for example, is not possible for red and blue to collide with each other since they are in a static position. Another important component for the game is the scoring systems. Each collision will have a score linked to it as shown in Table 3. The score are 0, 1, or -1.

Table 3 Elements Movement Link to Score.

| **Elements** | **Movement** |
|---|---|
| C1 | S1 |
| C2 | S2 |
| C3 | S3 |
| C4 | S4 |
| C5 | S5 |
| C6 | S6 |
| C7 | S7 |
| C8 | S8 |
| C9 | S9 |

**Evolutionary Algorithm :** The EA method that we have applied in this study is evolutionary programming. Number of elements that can presented in the game, the collision effect, the score of each collision, and the winning point, the losing point of the game and the number of each elements in the game. As mention earlier that collision effect will be represented by 0, 1 and 2 while score of each collision is between -1, 0 or 1. Winning point range is from 1 to the maximum of available elements presents in the game as well as the losing point. The number of elements of each type is range from 1 to 5 meaning that each color elements will have none to a maximum of 5 that can be present in the game. Population size is set to three as we do not want to increase the fatigue of the human tester (Takagi, 2001) as the larger the population size increase, more evaluation has to be done by a tester in order to complete a full run. The same goes to the number of generations as we want to keep the time durations lower, hence the number of generations is set to be 20. Below is the flow of the overall EP:

1.0 Start

2.0 Random initialization for parent chromosome. The value of the each phenotype is illustrate below

2.1 Phenotype value for position from 0 to 8 – range from 0 to 2

| C1 | C2 | C3 | C4 | C5 | C6 | C7 | C8 | C9 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |

2.2 Phenotype value for position from 9 to 17 – range from 0 to 2

| S1 | S2 | S3 | S4 | S5 | S6 | S7 | S8 | S9 |
|---|---|---|---|---|---|---|---|---|

2.3 Phenotype value for position from 18 to 21 – range from 0 to 5

| R | B | G | C |
|---|---|---|---|

2.4 Phenotype value for position from 22 to 23 – range from 0 to max number of total elements

| W | L |
|---|---|

3.0 Parent is loaded into the game environment and evaluated

3.1 Repeat step 1.0 to 3.0 until the number of individual parent reach 3

4.0 Select the best individual parent to seed for next generations offspring

5.0 Generate offspring from parent

6.0 Offspring is loaded into the game environment and evaluated

6.1 Step 5.0 to 6.0 is repeated until the number of offspring reaches 3

7.0 Select the best offspring from the populations pool to be parent for next generations

Step 5.0 to 7.0 is repeated until the number of generations reached 20

**Interactive Evolutionary Algorithm :** IEA has two different evaluation methods which is reactive and proactive feedbacks. In reactive feedbacks algorithms, it requires human evaluator to give their feedbacks after the game or it can also allow the human evaluator to intervene the autonomously running algorithm [2]. Proactive feedbacks algorithm allows human evaluator to pause the algorithm at stagnation stage and alters the parameters in the algorithm before allowing it to continue with its process [2]. Reactive feedback has been chosen to be the IEA feedback method in this paper. Human evaluator has been given a score range of 0 to 7 where 0 represent the lowest score value and 7 represent the highest score value for the particular individual.

## III. EXPERIMENT SETUP

The experiment has been conducted with the help from a human tester from a faculty. The tester has been brief with the information of how to play the game and how to assign a score for each game generated. Below is the procedure that he needs to go through to complete a full run of the game

[1] Start
[2] A game rules is loaded into the game environment
[3] Tester played with the game rules and assign score at the end of the game.
[4] Step 1.0 to 3.0 will be repeated 60 times since each generation has 3 individuals and the number of generations has been set to be at 20.

Nine different experiments will be conducted with different mutation rate. The first experiment is start off with a 10 percent mutation rate and the following experiment mutation rate will be increase to another 10 percent which mean experiment 2 will have 20 percent, experiment 3 with 30 percent and so forth. In addition, an individual chromosome was selected from each mutation rate. The criteria for the individual to be selected are:-

- It has to belongs to the user that has the highest average rating in the particular mutation rate
- It is the last highest score attain from the 20 generations

10 different players were asked to test on these selected individual.

## EXPERIMENT RESULTS

Table 4 below is a summary of the results of the average score given by each human evaluator for each mutation rate.

Table 4: Average Score for Each Mutation Rate

| User \ Mutation Rate | **0.1** | **0.2** | **0.3** | **0.4** | **0.5** | **0.6** | **0.7** | **0.8** | **0.9** |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2.17 | 2.28 | 1.75 | 2.07 | 2.28 | 1.95 | 2.75 | 2.67 | 1.85 |
| 2 | 1.97 | 1.97 | 2.00 | 1.92 | 1.67 | 2.00 | 2.93 | 1.85 | 1.77 |
| 3 | 2.65 | 1.73 | 1.92 | 1.98 | 1.95 | 1.92 | 1.75 | 2.22 | 2.22 |
| 4 | 2.18 | 2.08 | 2.05 | 2.07 | 2.13 | **1.58** | 2.53 | 1.77 | 1.85 |
| 5 | 2.13 | 2.32 | 2.23 | 1.70 | 1.83 | 1.80 | 1.80 | 1.77 | **3.17** |
| 6 | 1.88 | 2.05 | 1.95 | 1.95 | 2.25 | 1.90 | 1.92 | 2.13 | 2.08 |
| 7 | 1.98 | 2.18 | 2.15 | 1.98 | 1.85 | 1.83 | 2.33 | 2.00 | 2.35 |
| 8 | 1.77 | 2.05 | 2.07 | 2.07 | 1.93 | 2.00 | 2.12 | 2.17 | 2.07 |
| 9 | 1.95 | 1.62 | 2.35 | 2.07 | 1.85 | 1.70 | 2.18 | 1.88 | 1.62 |
| 10 | 2.17 | 1.95 | 2.15 | 2.10 | 2.03 | 1.67 | 1.78 | 2.18 | 2.53 |
| ***Average*** | *2.09* | *2.00* | *2.06* | *1.99* | *1.98* | *1.83* | ***2.21*** | *2.06* | ***2.15*** |

Table 5: Highest Score of Each Generation in Mutation Rate 0.9 for User No.5

| Generations | Score |
|---|---|
| 1 | **5** |
| 2 | **5** |
| 3 | **5** |
| 4 | 3 |
| 5 | 4 |
| 6 | **5** |
| 7 | **5** |
| 8 | **5** |
| 9 | **5** |
| 10 | 4 |
| 11 | 3 |
| 12 | 1 |
| 13 | 4 |
| 14 | **5** |
| 15 | 4 |
| 16 | 3 |
| 17 | 4 |
| 18 | 4 |
| 19 | 4 |
| 20 | 3 |

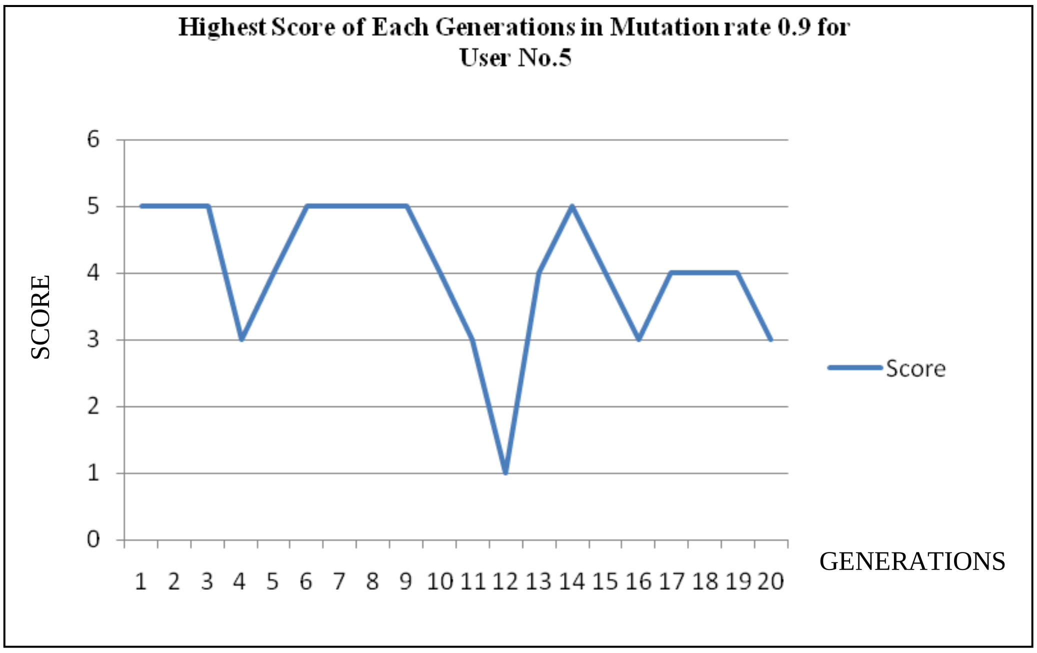


Figure 3: Graph for Highest Score of Each Generation in Mutation Rate 0.9 for User No.5

Table 6: Highest Score of Each Generation in Mutation Rate 0.7 for User No.2

| **Generations** | **Score** |
|---|---|
| 1 | 1 |
| 2 | 4 |
| 3 | 3 |
| 4 | 5 |
| 5 | 4 |
| 6 | 3 |
| 7 | 5 |
| 8 | 3 |
| 9 | 4 |
| 10 | 5 |
| 11 | 3 |
| 12 | 4 |
| 13 | 4 |
| 14 | 5 |
| 15 | 5 |
| 16 | 3 |
| 17 | 4 |
| 18 | 5 |
| 19 | 5 |
| 20 | 5 |

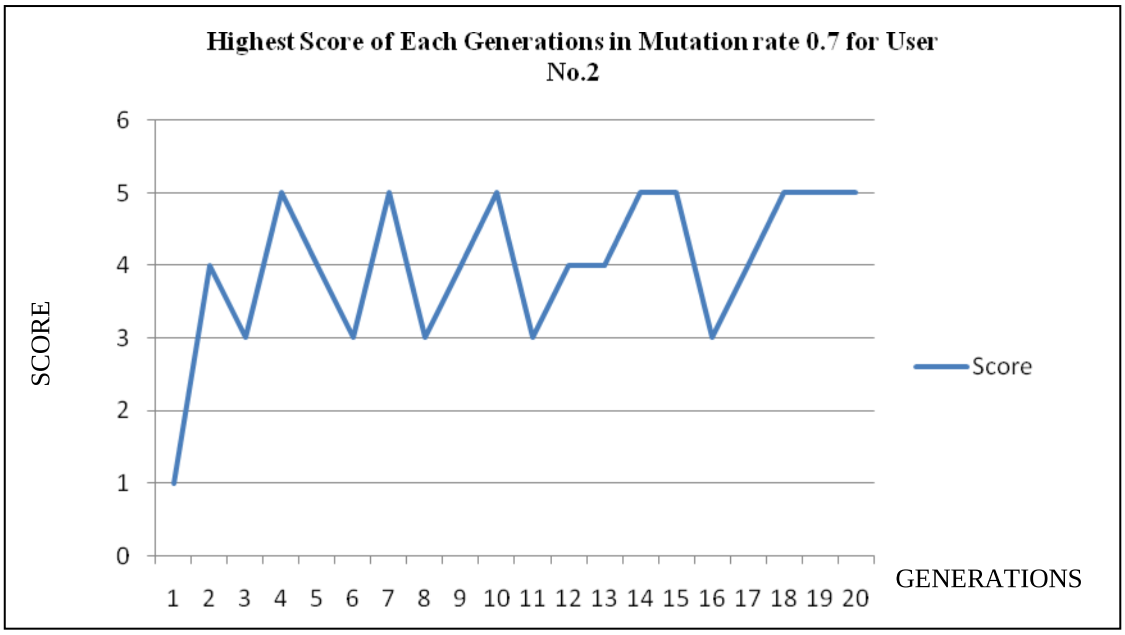


Figure 4: Graph for Highest Score of Each Generation in Mutation Rate 0.7 for User No.2

Table 7: Highest Score of Each Generations in Mutation Rate 0.6 for User No.4

| Generations | Score |
|---|---|
| 1 | 3 |
| 2 | 3 |
| 3 | 2 |
| 4 | 2 |
| 5 | 2 |
| 6 | 1 |
| 7 | 3 |
| 8 | 2 |
| 9 | 3 |
| 10 | 2 |
| 11 | 3 |
| 12 | 1 |
| 13 | 3 |
| 14 | 1 |
| 15 | 1 |
| 16 | 3 |
| 17 | 4 |
| 18 | 3 |
| 19 | 2 |
| 20 | 1 |

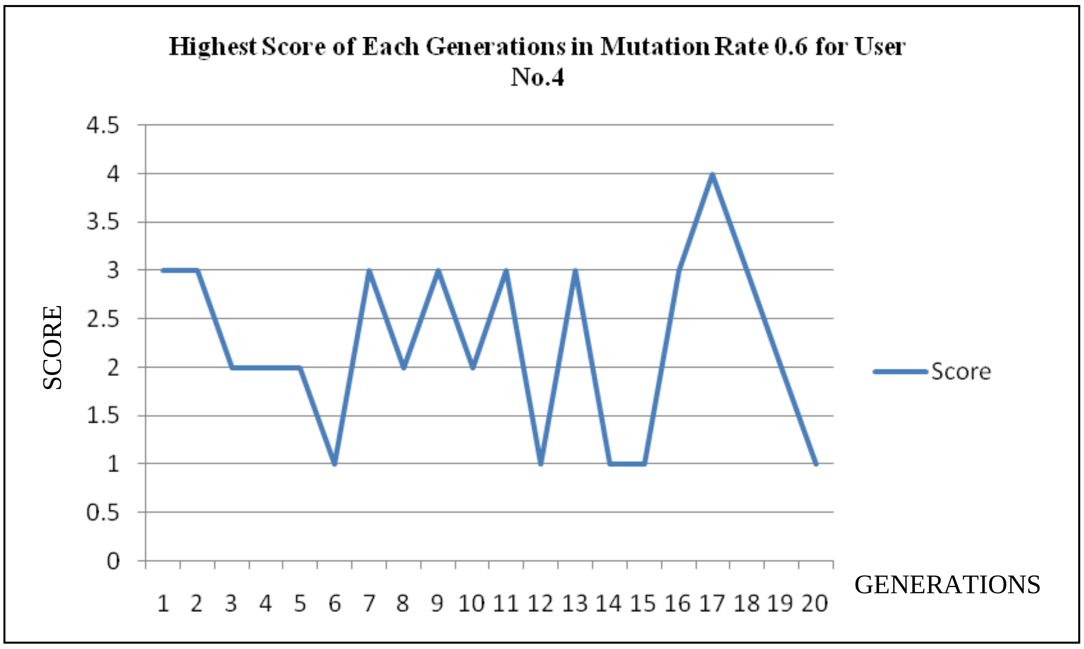


Figure 5: Graph for Highest Score of Each Generations in Mutation rate 0.6 for User No.4

Table 4 show the average results obtained where the highest average score obtained was 3.17 by user No.5 and is under the mutation rate of 0.9 while the lowest is 1.58 by user No.4 obtained from mutation rate of 0.6. Mutation rate of 0.9 has shown the best result compared to the other mutation rates. From Table 8, 9 out of 10 evaluators gave the highest score before the 10th generation and it reflects that by using mutation 0.9 the solutions can propel out of local optima and hence achieving a higher score from the evaluator. Table 5 and Figure 3 graph shows the highest score given by user No.5 in each generation for mutation rate 0.9. It is observed that 8 out of 20 generations have a high score of 5 and most of the other generations' score are at least 3 or more except for the 12th generation where the highest score in that generation is only 1. This reflects that most generation can generate good ruleset. Meanwhile Table 7 and Figure 5 graph shows the highest score in each generation for mutation rate 0.6 for user No.4. Most of the scores are below 3 hence this future supports that mutation rate 0.6 could not generate ruleset that reaches the user's satisfactory level.

Mutation rate 0.7 has the second best average score given by the 10 evaluators where 6 out of 10 average scores are over 2.00 and Table 9 shows in which generations the first highest score was attained. Sixty percent of the time the evaluators assigned the maximum score within the 10th generation. This proves that the individuals generated reached the evaluator's satisfactory level very quickly. Table 6 and Figure 4 graph shows highest score for each generation in mutation 0.7 given by user no.2. All except the first generation's score were more than 3 with a significant number of ruleset scoring the maximum of 5. Hence this further supports that the ruleset generated reached the user's satisfactory level.

Table 8: High Score Given by Each User in Mutation Rate 0.9

| **Evaluator** | **Highest Score** | **Generation** |
|---|---|---|
| 1 | 5 | 9th |
| 2 | 4 | 6th |
| 3 | 5 | 16th |
| 4 | 5 | 2nd |
| 5 | 5 | 1st |
| 6 | 5 | 2nd |
| 7 | 5 | 2nd |
| 8 | 4 | 3rd |
| 9 | 5 | 5th |
| 10 | 5 | 1st |

Table 9: High Score Given by Each User in Mutation Rate 0.7

| Evaluator | Highest Score | Generation |
|---|---|---|
| 1 | 5 | 2nd |
| 2 | 5 | 3rd |
| 3 | 5 | 2nd |
| 4 | 5 | 3rd |
| 5 | 5 | 18th |
| 6 | 5 | 15th |
| 7 | 5 | 4th |
| 8 | 5 | 13th |
| 9 | 5 | 1st |
| 10 | 5 | 15th |

Table 10: High Score Given by Each User in Mutation Rate 0.6

| Evaluator | Highest Score | Generation |
|---|---|---|
| 1 | 5 | 16th |
| 2 | 5 | 10th |
| 3 | 5 | 6th |
| 4 | 4 | 16th |
| 5 | 5 | 11th |
| 6 | 5 | 6th |
| 7 | 5 | 20th |
| 8 | 4 | 2nd |
| 9 | 5 | 14th |
| 10 | 4 | 11th |

Although in mutation rate 0.9 contains one of the highest average score, the total sum of all the scores attained for mutation rate 0.9 is only 1290 while for mutation rate 0.7 is 1326. The individual created by mutation rate 0.7 shows a higher and consistent score compared to mutation rate 0.9 based on this observation. Mutation rate 0.6 contains the lowest average score for this experiment and its total score is 1101 and Table 10 lists the highest score it receive from each evaluator. Table 10 very clearly shows that most of the evaluator only gives the first highest score at the very late stage of the generation.Throughout this experiment, it was found that the higher the mutation rate, the probability of generating a better result increases. In other words, the possibility of leaving locally-optimal solutions increases. By having three individuals in each population also helps to decrease the searching time for a good set of novel game rules and yet maintaining good individuals through each generation. Since the number of individual has increased significantly from the previous total of 20 individuals to 60 individuals, it has affected the human fatigue in this experiment. As the fatigue increased, it probably decreased the accurate judgment of the human evaluator. Hence, the overall lower averages obtained as compared to the preliminary experiment. Figures 6 to 32 below are some screenshots of each mutation rate experiment with a table summarizing each rule set generated.

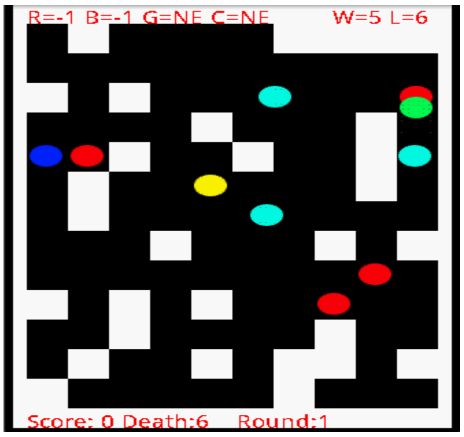
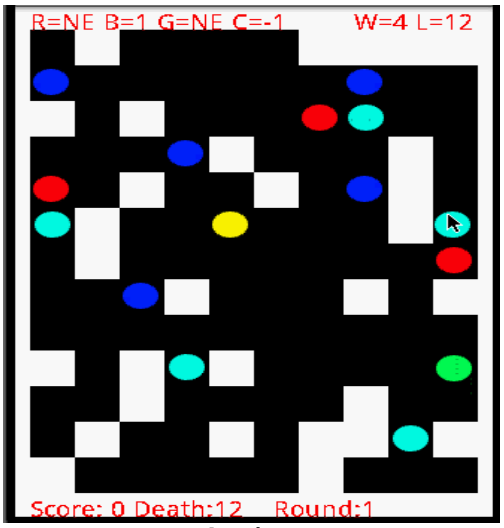


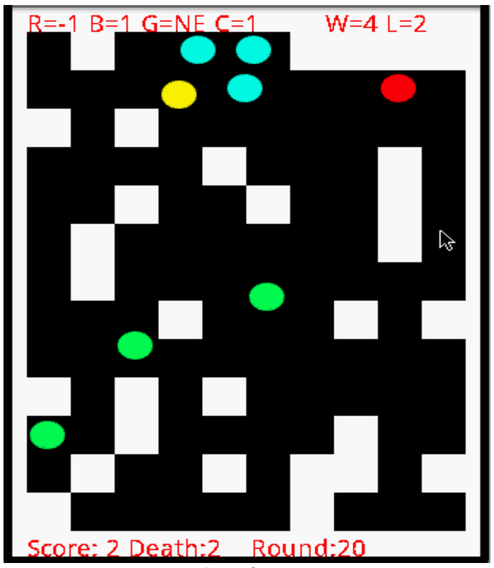
Figure 6: Screenshot for Mutation Rate 0.1



R=NE B=1 G=NE C=-1 W=4 L=12

Score: 0 Death:12 Round:1

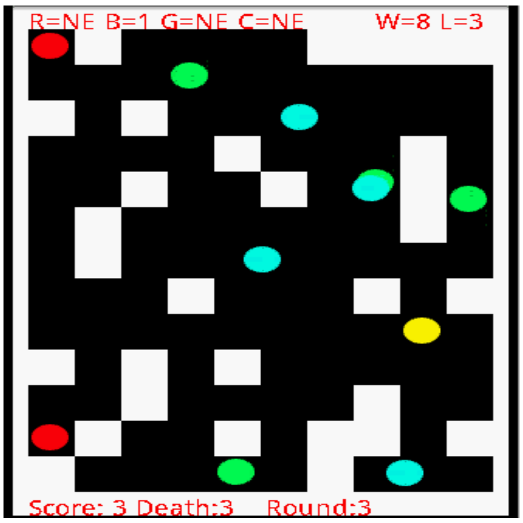
Figure 7: Screenshot for Mutation Rate 0.2



R=-1 B=1 G=NE C=1 W=4 L=2

Score: 2 Death:2 Round:20

Figure 8: Screenshot for Mutation Rate 0.3



R=NE B=1 G=NE C=NE W=8 L=3

Score: 3 Death:3 Round:3

Figure 9: Screenshot for Mutation Rate 0.4

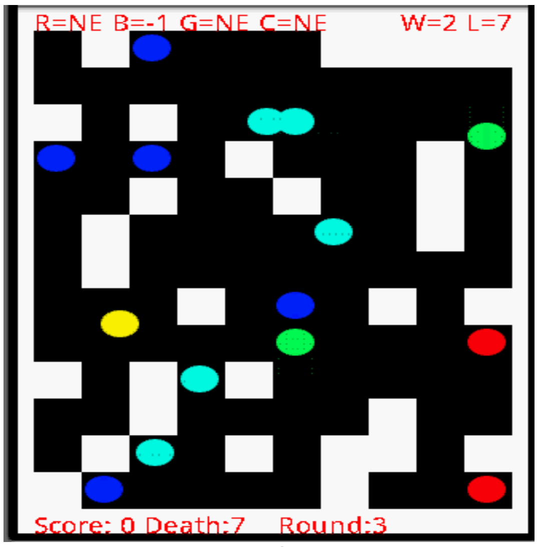


Figure 10: Screenshot for Mutation Rate 0.5

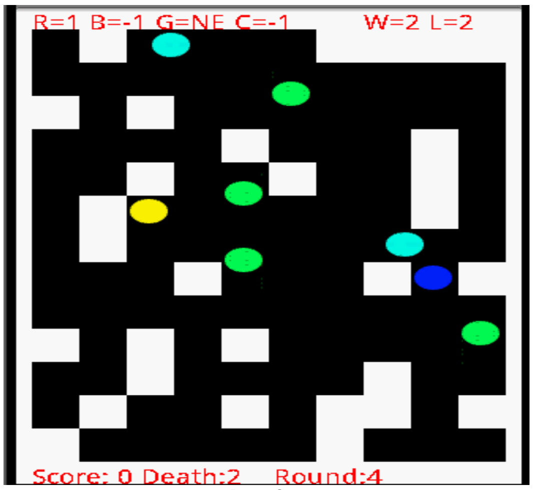


Figure 12: Screenshot for Mutation Rate 0.6

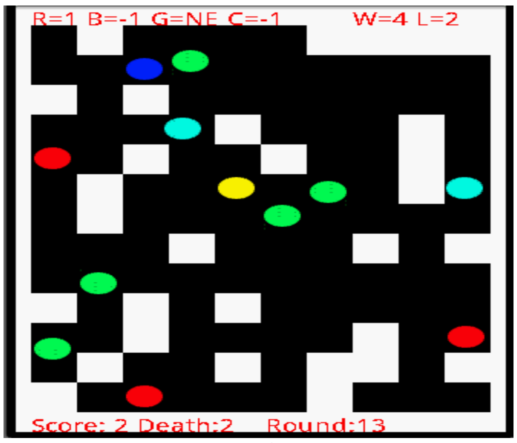


Figure 13: Screenshot for Mutation Rate 0.7

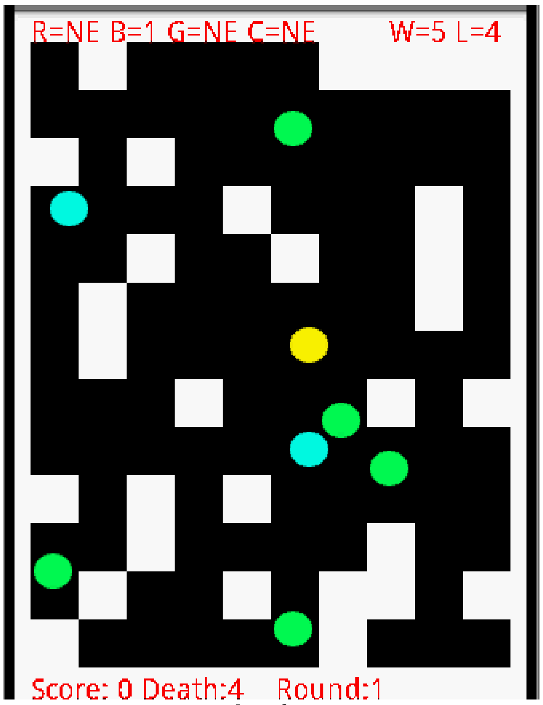


Figure 14: Screenshot for Mutation Rate 0.8

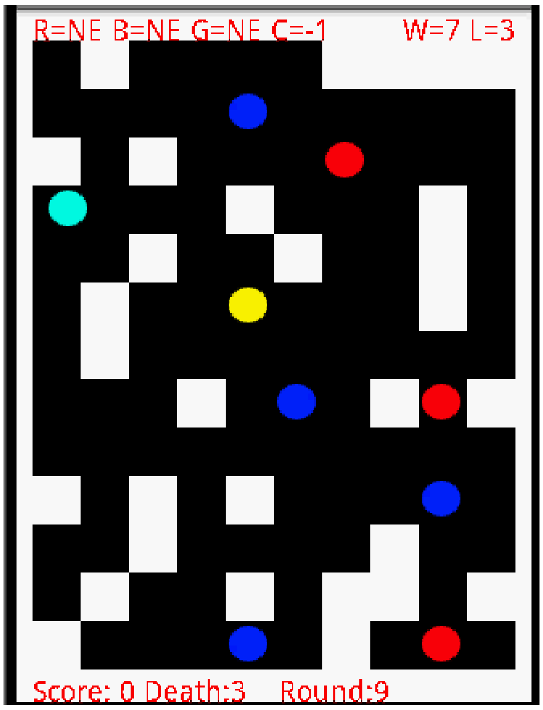


Figure 15: Screenshot for Mutation Rate 0.9

## IV. Conclusion And Future Work

Implementation of IEA in mobile games has been introduced in this paper. Searching for a suitable mutation rate in the main concern in this paper and from the results obtained, it shows that using a higher mutation rate will tend to result in better solutions. Since IEA involves a human evaluator and we have to keep this process running fast and timely, getting a suitable mutation rate will help to yield a faster and better convergence rate. Future work for this research is to extend the preliminary results to a larger pool of human evaluators to get a more statistically significant result. Another aspect that should be looked into for future work is searching for a better population size that is suitable with IEA and also the score rates.

## V. ACKNOWLEDGEMENTS

This research is funded by the FRGS Research project FRGS0213-TK-2010 granted by the Ministry of Science, Technology and Innovation, Malaysia.

## REFERENCES

**Proceedings Papers:**

[1] J.Togelius, G.N.Yannakakis, K.O.Stanley, C.Browne. Search-based Procedural Content Generation. In EvoStar Conference, 2010.

[2] R.Breukelaar, M.Emmerich, T.Bäck. On Interactive Evolution Strategies. In EvoWorkshops, 530-541, 2006

[3] J.Togelius, M. Preuss, G.N.Yannakakis.Towards Multiobjective Procedural Map Generation. In Workshop on Procedural Content Generation in Games, 2010.

[4] J. Togelius, J. Schmidhuber. An experiment in automatic game design. In IEEE Symposium on Computational Intelligence and Games, 2008.

[5] K.Compton, M.Mateas. Level Design for platform Games. In Second Artificial Intelligence and Interactive Digital Entertainment International Conference (AIIDE) Marina del Rey, 2006.

[6] Horowitz. Generating rhythms with genetic algorithms. In International Computer Music Conference, 142-143, 1994.

[7] E.J.Hastings, K.G.Ratan, K. O. Stanley. Evolving Content in the Galactic Arms Race Video Game.In IEEE Symposium on Computational Intelligence and Games (CIG09). Piscataway, NJ:IEEE, 2009

[8] P.J. Angeline. Evolving Fractal Movies. In: 1stAnnual Conference on Generic Programming Standford, Ca, USA, 503-511, 1996.

[9] H.Takagi. Interactive Evolutionary Computation:Fusion of the Capabilities of EC Optimization and Human Evaluation. . In IEEE 2001,vol.89, 1275-1296, 2001.

**Books:**

[10] A.E.Eiben, J.E.Smith.Introduction to Evolutionary Computing. 2003.