Computing Creativity: Divergence in Computational Thinking

Vicki Bennett
Department of Communication
University of Colorado at Boulder
Boulder, CO 80309
+1 (303) 995-2802
Vicki.Bennett@colorado.edu

Kyu Han Koh
Department of Computer Science
University of Colorado at Boulder
Boulder, CO 80309
+1 (303) 495-0357
kohkh@colorado.edu

Alexander Repenning
University of Colorado at Boulder
430 UCB
Boulder, CO 80309
+1 (303) 492-1349
ralex@cs.colorado.edu

ABSTRACT

Conventionally creativity is often conceived as an aptitude to be discovered in an individual that cannot be mathematically measured. But the concept of creative thinking as a divergence from a standard "norm" is used in creativity research for the purpose of assessing creativity and is also linked to non-traditional or creative processes that lead to unique and divergent artifacts [1,2]. Using Computational Thinking Pattern Analysis (CTPA)[3], the divergence between implemented computational thinking patterns in a student-created game, and that game's tutorial "norm" is calculated as an indicator of creativity. Through a case study of one teacher using three unique learning conditions, CTPA's computed divergence is explored as a valid measurement of creativity in these student games.

Categories and Subject Descriptors

K.3.2 [**Computers and Education**]: Computers and Information Science Education

Keywords

ACM proceedings, Computational Thinking, Computational Thinking Pattern Analysis, Creativity Measurement, Game Design

1. INTRODUCTION

In our previous research, we reported that a game design class could foster student creativity in public schools [4]. The aspects of game design documented were character/agent design and level/worksheet design. A visual inspection of authored agents and worksheets, plus classroom observations yielded the data for our conclusions. This type of data assessment was necessary due to the lack of a computational tool capable of measuring creativity in game design programming solutions.

Measuring the creative aspects of programming decisions that students employ can reveal teaching protocols for successful increases in the quality and quantity of those creative solutions. Measuring creativity in programming presents a unique obstacle. Since manually analyzing game programming patterns for creative divergence is labor intensive, a computational tool for creative assessment was sought. CTPA was developed as a tool for teachers and researchers to gauge game design accuracy. We surmised that a further expansion of CTPA could possibly assess indications of creativity as "divergence from the norm," where the "norm" is defined as the standard tutorial for each specific game.

1.1 Creativity Defined as Divergence

Defining a multi-faceted concept such as creativity presents measurement difficulties. Research investigations into creative processes usually describe them as the identifying or discovery of a problem and/or its solution. The solution or outcome must be significantly unique or divergent from other possible solutions [5], especially from the predetermined solution of the test. Herring et al [6], report that creativity can be produced regularly through the exposure to multiple visual examples of a similar nature, and multiple examples usually support a more divergent perspective of the problem, which in turn fosters more creative or divergent solutions. Currently, the most common measure of creativity for creativity research purposes is based on tests of divergence [1, 5, 7]. Tasks designed to distinguish divergence are currently the most recognized creative assessments [2, 8]. Although the quality and quantity of the solutions to the task must be considered, divergence from the accepted "norm" is commonly considered to be a significant indicator of creativity. The solution does not need to be totally unique or of high quality to be assessed as creative.

For example, the figure below (Fig. 1) illustrates several different Frogger design examples from different Frogger games created by 6th graders. All of them are divergent from the standard solution, provided by their teachers (Fig. 2). Similar to other skills, creativity can be learned and practiced. Consequently, creative efforts are not always of high quality, especially in the beginning. But, can creativity be measured computationally?

Inspired by the notion of divergent thinking tasks, we devised a method or tool (CTPA), divergence calculation [9], to more accurately assess creativity in programming solutions. Using comparative divergent thinking tasks as a measurement criteria for creativity in programming solutions should yield valid results, since computer programming lends itself to computational solutions and assessing divergent elements in programming solutions should be possible through a mathematical calculation.

Throughout the 2010/2011 school year, some early stages of mathematical measurement were investigated to calculate divergence in game programming in the Scalable Game Design project [10, 11]. One teacher presented lessons in three different conditions during 2010/2011 school-year. Since these three conditions lent themselves to a traditional experimental set-up, already accounting for the teacher influence as a variable, the game divergence from the tutorial norm was calculated with the addition of the implications of the creativity measurement.

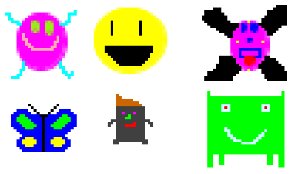


Figure 1. Frogger Design by Students

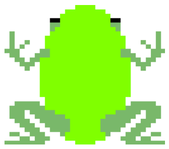


Figure 2. Standard Frogger Design

1.2 Scalable Game Design Project

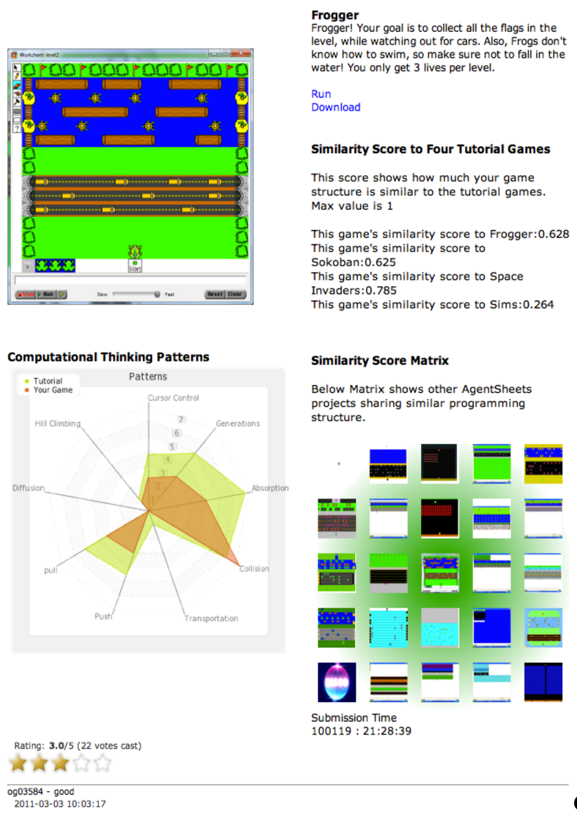
Although game design is often believed to be a creative endeavor, for the programming novice, this is not usually the case. A student's first experience can often be boring or disappointing at best and debilitatingly frustrating at worst. The Scalable Game Design (SGD) Project investigates the use of game design as a means to encourage interest in programming and computer science in middle school students. As members of SGD project research team, the authors often have opportunities to visit and observe middle school teachers during project implementation classes. During these classes the authors often observed an outpouring of creativity. Even though participating classes follow specific curriculum content and upload their self-created games to the Scalable Game Design Arcade (SGDA) at the completion of each course module, individual teachers add their own style to the curriculum parameters. This is observed in person and from the submitted games' programming analyses with the CTPA. A visual inspection for agent and worksheet creativity is also conducted.

1.3 Scalable Game Design Arcade

In support of the Scalable Game Design project, we designed and operated a cyberlearning infrastructure, Scalable Game Design Arcade (SGDA). SGDA works as an online repository for collecting games/simulations that are created by middle school students from participating schools. SGDA consists of three parts: a main page, an assignment gallery, and an individual page. The main page displays recently submitted games, the most downloaded games, and the most played games, while the assignments gallery shows multiple submitted games/simulations that are submitted by participant school students in one unit class (not shown). The individual page displays a screenshot of a submitted game/simulation, the Computational Thinking Pattern Analysis (CTPA) graph [[9], links for playing and downloading games/simulations, and a similarity score between a given game/simulation and four tutorial games (Fig 3). Any user who accesses SGDA can play, download, and/or rate a game without a time lag after the submission is made.

1.4 Using Examples for Creativity

Examples are one of the most commonly employed methods of facilitating creativity. The use of multiple examples of other people's creative activities, often give rise to ideas and solutions we would not otherwise consider. Most project teachers choose to have their students play the games on SGDA before formal instruction, as a motivational tool to engender interest and encourage familiarity with a specific game. Promoting creativity in their students' games was an unexpected effect. The teachers saw a creative increase in their students' abilities in comparison to other project lessons [4]. The use of multiple examples on SGDA is consistent with stimulating creativity, while providing motivation, as intended.


Frogger
Frogger! Your goal is to collect all the flags in the level, while watching out for cars. Also, Frogs don't know how to swim, so make sure not to fall in the water! You only get 3 lives per level.

Run
Download

Similarity Score to Four Tutorial Games

This score shows how much your game structure is similar to the tutorial games. Max value is 1

This game's similarity score to Frogger:0.628
This game's similarity score to Sokoban:0.625
This game's similarity score to Space Invaders:0.785
This game's similarity score to Sims:0.264

Computational Thinking Patterns

Similarity Score Matrix

Below Matrix shows other AgentSheets projects sharing similar programming structure.

Submission Time
100119 : 21:28:39

Rating: **3.0**/5 (22 votes cast)

og03584 - good
2011-03-03 10:03:17

Galler

y Page of SGDA

Figure 3. The Individual Page of SGDA. Individual page illustrates the screenshot of the game (upper left), Run and Download button (upper right), the game's similarity score compared to four tutorial games (middle right), a similarity score matrix showing similarly programmed games to the submitted game, and the CTPA graph (bottom right & left).

2. METHODS

Methodology for this case study incorporated multiple classroom observations, the CTPA of submitted student games, and the divergence calculation to assess creativity. Inspired by Latent Semantic Analysis [12], we developed the concept of Semantic Subject Analysis (SSA) that uses multiple high dimensional cosine calculations to analyze semantic meanings of a given context with several pre-defined subjects.

Latent Semantic Analysis (LSA) was devised to compute the similarity between two given essays: a student-written essay and a sample essay [12]. LSA can be used as a grading-aid tool for text assignments by calculating the high dimensional cosine value between the student-created text and sample/standard text. However in LSA, the high dimensional cosine value calculations between the text and the target-text can express the similarity between these two texts as a single value, only. On the other hand, SSA can show the semantic meaning of a given text by calculating multiple high dimensional cosine calculations. These value calculations compare the similarity of the given context to the pre-defined subjects semantically within that context.

2.1 Computational Thinking Patterns

Computational thinking is a high level concept, and still has not been clearly defined in one sentence [13]. So we conceptualized Computational Thinking Patterns within the game design context to help students and teachers understand how CT can be practically utilized [10]. A Computational Thinking Pattern (CTP) is an abstract form of programming, which can be easily found in game and simulation programming. For example, the CTP, Generation, represents one agent creating another agent (i.e. a gun shoots a bullet or one cell splits in two). While CTP, Absorption represents that same bullet 'disappearing' as it enters the target. In this way, each CTP represents only one complete phenomenon or behavioral concept in a game or science simulation design.

2.2 Computational Thinking Pattern Analysis

The Computational Thinking Pattern Analysis (CTPA) is a specified version of SSA, within which nine canonical computational thinking patterns work as pre-defined subjects within a given game's context. The SSA structure appeared to be a good fit for comparing computational thinking patterns within student-submitted games to the tutorial standard used to teach those specific games. Consequently, SSA was revised to analyze the semantic meaning of the computational thinking pattern concept. To reflect the tool's analysis capabilities, we named the tool, the Computational Thinking Pattern Analysis [3].

CTPA tool consists of two parts: a computational thinking pattern analysis (CTPA) and a computational thinking pattern analysis graph (CTPAG). CTPA compares the similarity between a given programmed artifact and the nine pre-defined subjects (nine canonical CT patterns). For each programmed artifact nine high dimensional cosine calculations are computed which results in nine values between 0 and 1. The result of this CTPA is visualized as a form of spider graph with nine values displayed graphically in comparison to the same nine values calculated for the standard tutorial. The CT patterns for the standard tutorial of a given game overlap the CT patterns of the submitted game in different colors. The graphic display is called the CTPA graph or CTPAG. The CTPAG (Fig 4) visualizes the semantic similarity between a given game/simulation through each computational thinking pattern available. The equation below illustrates a high dimensional cosine calculation for the CTPA [3].

In this equation (below), u and v mean a given game/simulation and one canonical computational thinking pattern respectively. Also, n means he vector size of a game/simulation or CT pattern, and m means the number of computational thinking patterns that are applied to CTPA, currently 9. The calculated result of CTPA through CTPA (1) to CTPA (*m*) could be represented as an *m* length vector.

$$\text{CTPA}\ (m) = \left[\frac{\sum_{i=1}^{n} u_i v_i}{\sqrt{\sum_{i=1}^{n} u_i^2}\sqrt{\sum_{i=1}^{n} v_i^2}} \right]_1^m$$

Specifically for the above equation, the Computational Thinking Pattern Analysis (CTPA) is computed by calculating the value of cosine □ between two n-dimensional vectors that represent a given game and one computational thinking pattern. Thus, the CTPA graph in Fig 4 requires nine high dimensional cosine calculations between the Frogger game and the nine computational thinking patterns: Cursor Control, Generation, Absorption, Collision, Transportation, Push, Pull, Diffusion, and Hill Climbing. The CTPA graph visually depicts the difference or divergence between the norm (tutorial standard-green) and each individual student's submitted artifact (brown) by overlapping the two images in different colors in one graph.

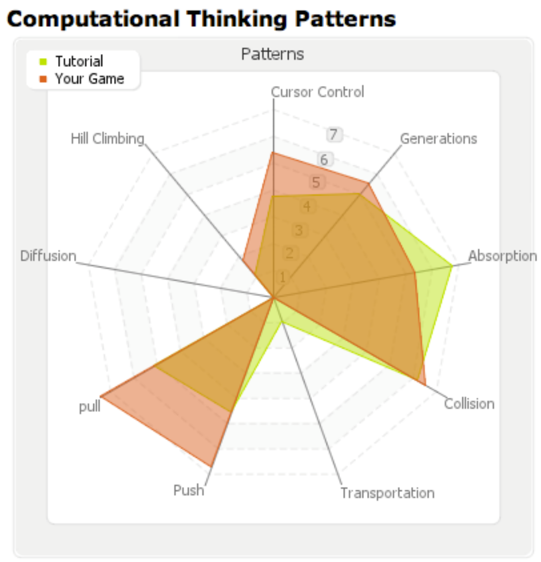


Figure 4. CTPA Graph Example 1. This graph is an actual student's example of a game of Frogger.

2.3 Divergence Calculation

The data previously collected from our class observations and interviews illustrate that the students' game artifacts showed increased creativity within the Scalable Game Design project. Previously some of these students referred to the making of the game (agent & worksheet design) as "all their own" [4]. This type of ownership appears to be consistent throughout the design of the agent depictions and worksheets. Our classroom observations confirmed that this "ownership" was related to divergence from the norm and/or creativity. We wanted to find evidence of creative ownership in the programming element of the student game artifacts, as well. With a mathematical method of calculating the divergence in the game programming, it seemed possible. Divergence in programming could be assessed through the evaluation of the differing approaches each student employed within the specified design parameters. In other words, when faced with a difficult challenge for agent behavior design, each student defined that programming challenge in a way that specified an accurate programming solution [7]. Consequently, creativity in game programming could be described as the divergence of a specific programming solution utilized by one student and not another in comparison to the SGD online tutorial standard. From this we devised the divergence calculation to demonstrate student-programming creativity as a divergence calculation from the "norm" or SGD online tutorial standard.

We recognized that computing creativity as divergence from the "norm" within the framework of CTPA could be possible. CTPA was already used to validate the students' game artifact solution divergence in comparison to the tutorial. So, it seemed logical to investigate its use for computing creativity. Figure 4 depicts a student-submitted game in comparison to the standard SGD online tutorial [3]. In figure 4, the difference or space between the tutorial (green) and the submitted artifact (brown) visually displays the divergence of a student's Frogger game from the tutorial "norm." Since each axis on the CTPA graph represents one element in a vector, the CTPA graph represents a nine-element vector, where each element represents a CT programming pattern that could be chosen by the student as part of his/her programming solution. Thus, the divergence of a student-created artifact from the tutorial "norm" can be calculated as the difference between two nine-element vectors; one is from the tutorial and the other is from the submitted artifact. The Divergence Score is calculated from the length of vector difference of the nine-element vector. The equation of the Divergence Calculation is depicted below.

$$Divergence(x) = \frac{\sqrt{\sum_{i=1}^{n}(u_i - v_i)^2}}{\sqrt{n}}$$

In this equation, u and v represent a tutorial and a given game respectively, and n represents the number of computational thinking patterns, which is equal to 9. At the current point in our research, 9 computational thinking patterns are used to accommodate the Agentsheets software, as well as the nine patterns taught within the Scalable Game Design project curriculum.

For example in Fig 4, the student-submitted game and the tutorial can be represented as nine dimensional vectors respectively (0.525, 0.557, 0.432, 0.641, 0, 0.687, 0.721, 0, 0.197) and (0.373, 0.499, 0.679, 0.623, 0.096, 0.455, 0.51, 0, 0.106). The difference of those two vectors is (0.152, 0.058, -0.247, 0.018, -0.096, 0.232, 0.211, 0, 0.091). The normalized (divided by the value of rooted n) length value of that vector is 0.15, and this is the value of divergence score of the given game.

Equally, when a student-submitted game is exactly the same as the tutorial, there is no difference between those graphs. So the submitted game and the tutorial can be represented as following vectors, (0.373, 0.499, 0.679, 0.623, 0.096, 0.455, 0.51, 0, 0.106) and (0.373, 0.499, 0.679, 0.623, 0.096, 0.455, 0.51, 0, 0.106). The difference of those two vectors, of course, is (0, 0, 0, 0, 0, 0, 0, 0, 0). The value of 0 represents no difference between a given game and the tutorial, so the game's programming is identical.

2.4 Three Class Conditions

Sheryle (pseudonym), the teacher selected for this study, taught three unique class conditions. Initially she taught the project class based on the SGD online tutorial. Subsequently, she designed her own tutorial for her regular in-class students and then transferred her tutorial adaptation to an online version of the project class. This offered a rare opportunity to compare three unique class conditions without having to consider teacher influence as a random variable. In all three class conditions the uploaded student games were noticeably divergent from the SGD tutorial "norm."

Common factors identified from all three class conditions taught by Sheryle are as follows:

- Sheryle is the teacher of record for all classes
- She alone helps the students complete their games
- She followed the project curriculum content parameters
- Frogger is the first project game taught to all classes
- Frogger is the game uploaded by all students
- Frogger is the only uploaded game analyzed

3. CREATIVITY IN GAME DESIGN

Runco and Okuda [5] describe creativity as divergence from the norm. Through observations of the project classes and the sets of games produced by these classes, we conceptualized that the differences between the individual student-created games and the tutorial could logically indicate creativity as computed divergence from the "norm." So, the CTPA processed data comparisons were utilized for measuring/computing creativity from the games uploaded to the Scalable Game Design Arcade (SGDA).

3.1 Game Dimensions and Creativity

The game design process allows for variations and/or individual divergence along three main dimensions: character, level and behavior (descriptions below). Creativity can be expressed within all three of these described game dimensions. Assessing the creativity or divergence from the "norm" in agent and level design can be detected by a simple visual inspection of the artifacts. In our previous research [4], we demonstrated that divergence in character design and level design can indicate creativity.

Characters: The characters or agents in AgentSheets [14], make up the entire game worksheet. If an agent doesn't exist for a specific object, that object does not exist in that particular game. In Frogger, agent depictions include frogs, trucks, streets, turtles and a river.

Levels: Levels may vary enormously. The only constant is that each succeeding level should be more difficult than the proceeding one. In AgentSheets, each game level is represented by a worksheet, similar to a blank painter's canvas. The student lays out his/her created agents into a configuration that is most pleasing to them. The game level sequence, as well as the difficulty of the levels can show a students' creativity or divergence from the tutorial "norm."

Behavior: The programming that students create, determines the behavior of characters, and is the most complex aspect of the game-design process.

Our current study anticipates the clarification and/or development of assessment possibilities for the behavioral dimension as well.

3.2 Creativity: Programming (Behavior)

Programming an agent as the student envisions can be problematic due to multiple available programming solutions. The specific approach an individual student uses to program his/her game differently than the tutorial is a divergence from that standard "norm" and an indication of creative programming. Although a game's programming (behavior) can also be examined through a visual inspection, the process is labor intensive and time-consuming. Consequently, when evaluating programming, originality and/or "divergence from the norm" the process is not as simple as the assessment of the agents and levels. Manually inspecting each game for creativity in the same manner would be an extremely time-consuming process. So in order to find a method for calculating divergence in game programming that was less time consuming the divergence calculation was developed.

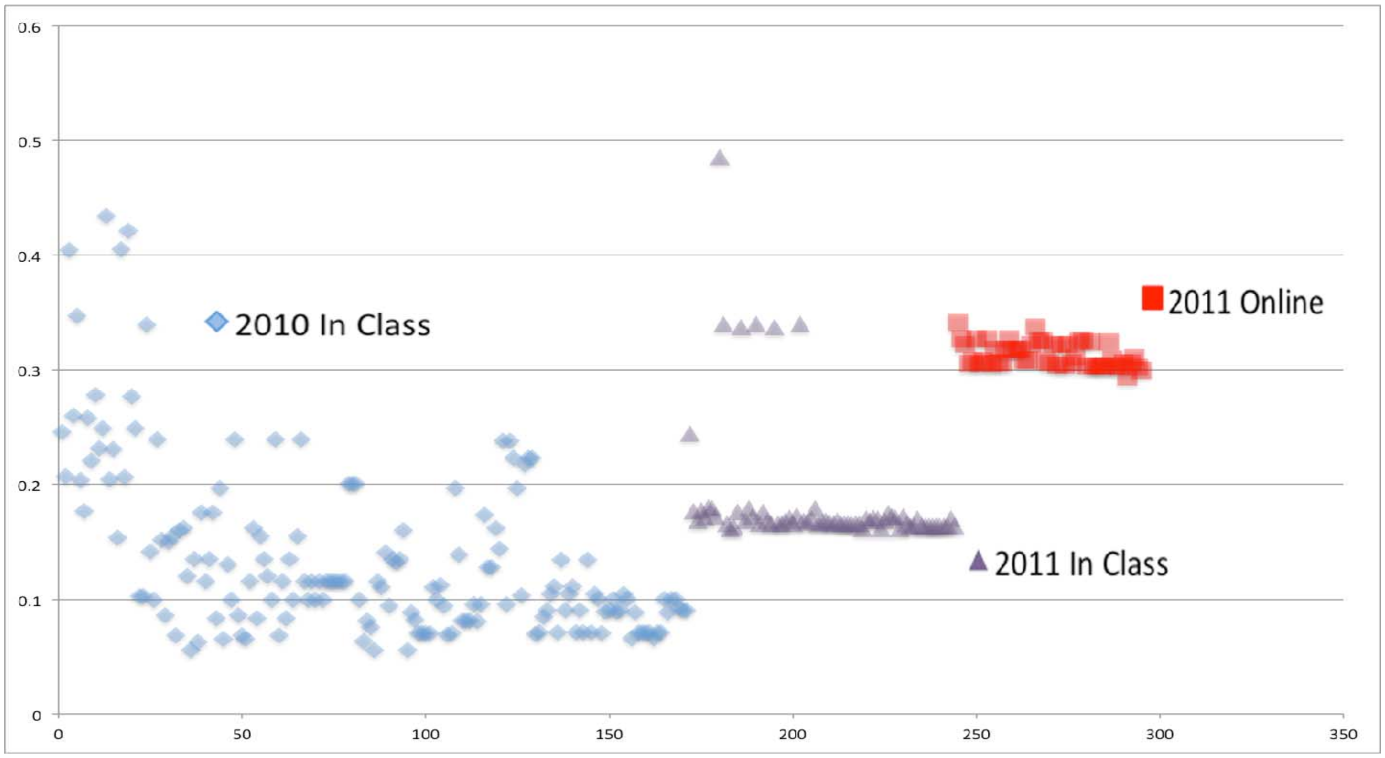


Figure 5. Scattered Divergence Calculation Graph: X-axis represents time by order of submission. Y-axis represents Divergence Score. Each dot means individual submission. 296 Frogger games are displayed in this graph.

4. FINDINGS

For this investigation we chose three unique class conditions that were taught by a single teacher. This allowed us to keep the experimental focus on the divergence calculation as an indication of creativity, as opposed to teacher influence. Findings show a marked difference between the three class conditions in regards to programming divergence, as well as class difference, as represented in Figure 5.

4.1 Divergence Calculation Graph

In Figure 5 (above), the graph depicts the collected data of all Frogger games during 2010-2011 academic year from Sheryle's three-class conditions, using the divergence calculation above. Each individual student-submitted game is placed on the graph according to the calculated divergence represented by his/her game. The different colors represent the three class conditions. The three distinguishable clusters accurately represent the three distinct class conditions. The left cluster (blue) displays a sparser more scattered pattern than the middle (purple) and right (red) clusters. In this analysis, 296 Frogger games are represented by this graph. Until the 171st game submission, the students' Frogger games were highly divergent from each other. From the 172nd submission the games started to converge into each other. Coincidentally, the 172nd game submission was the exact time frame when Sheryle started using her own tutorial in place of the official SGD online tutorial. It appears that although her teaching style was unchanged, her presentation of the material had evolved in some fashion.

Table 1. Standard Deviation Calculation

$$f(x;\mu,\sigma^2) = \frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

Using the standard deviation equation above we also calculated the class standard deviation, as well as the class divergence average. Those are displayed in Table 2 (below). It appears that while the "2010 In-class" condition (blue) with the widest pattern spread in Figure 5, also shows the largest standard deviation within the class, while the "2011 Online class" (red) with the narrowest spread, has the smallest standard deviation within the class. This means that the games in the "2010 In-class" condition are more divergent to each other than the games in "2011 Online" class condition.

Table 2. Divergence Calculation Score in Each Class

Divergence Score	Standard Deviation	Average
In Class 2010	0.074	0.135
In Class 2011	0.057	0.186
Online 2011	0.011	0.314

We conjecture that not only is the revised tutorial a significant factor in the represented divergence between class conditions, but that the in-class/online condition comparison also appears to be a significant factor affecting the divergence calculation for at least two class conditions, effecting calculated creativity.

5. DISCUSSION

Previously, we have documented indications of creativity in two game design aspects, agent design and level/worksheet design [4]. Showing creativity in the programming aspect of game design through the divergence calculation is the goal of this study. Currently, the most common indication of creativity is based on divergence assessments [1,5,7]. Using this standard, we have developed a mathematical measure of divergence for game programming, that calculates the difference of each programming sequence of a submitted game from the game tutorial "norm." The main difference in our measurement of creativity, is the use of a mathematical calculation in place of the subjective appraisal by a trained rater. The "norm" for standard creativity tests is usually a predetermined standard solution, similar to the SGD online tutorial. Since, the SGD tutorial is commonly used by most project teachers, it is the "norm" for the SGD curriculum or standard by which the teachers judge or grade their students' work. So, based on traditional measures of creativity, individual game divergence from the SGD tutorial should indicate the presence of individual and/or group creativity.

Although the same teacher conducted the three, class-conditions discussed in this paper, operationally eliminating teacher influence as a mitigating variable in the divergence calculation, we recognize a teacher rarely teaches multiple classes exactly the same. However, for the present purpose of beginning a validation process for the divergence calculation as a measurement method of creativity, we believe we have documented compelling support. So, when looking at the Scattered Divergence Calculation graph (Fig. 5) the three separate clusters, representing these three class conditions, are obvious. Even without the designating colors, the three class conditions stand apart from each other, as distinct clusters. So, although the implications and meanings of the different cluster spreads is not clear at present, the most significant feature of the Scattered Divergence Calculation graph in Figure 5 is that each of the three separate class-learning conditions generates a unique divergence pattern. Since each of the separate conditions is unique, then the first validity test should be to show that the divergence calculation can demonstrate that uniqueness in the game programming analyses and it does. The amount of creativity from divergence that each cluster pattern represents is not as important as the fact that it displays programming distinctiveness within the divergence calculation model, in general, in three separate learning conditions.

The divergence calculations are supported by other data sources, such as the teacher's unsolicited comments about her students' creativity and our observations of student enthusiasm and creativity in the physical classroom settings. Space limitations prevent us from more disbursing more details of these sources. Previous research on the use of examples for promoting creativity would also tend to support the divergence calculation since SGDA examples are a common teaching tool for student motivation. We additionally conducted a manual inspection of the programming code to support the actual divergence calculation within a random sample of the chosen uploaded games as a fail-safe.

Current creativity research for showing indications of creativity is founded on common comparisons for divergence. Similarly, our divergence calculation was utilized for three separate class conditions and resulted in three separate divergence patterns. Indications show that these divergence calculations could demonstrate, not only creativity indications within programming solutions, but that these calculations are subtle enough to differentiate between different class learning conditions of the same teacher.

So grounded on the basis that creativity can be shown through divergence in thinking [1,5,7], it is logical to conclude that creativity in game programming solutions can be exposed through a similar divergence in programming pattern solutions from a consistent format, in this case the SGD tutorial "norm." Consequently, we believe that the divergence calculation model can significantly inform the measurement of creativity within programming and possibly other scientific areas. We look forward to further investigating the more specific indications and meanings of the divergence calculations, cluster spreads and how these might relate to the quality of the creative solutions employed, especially to gain an understanding of how individual students determine the solutions they use to program their games and how that relates to the tutorial "norm" within the curriculum.

6. ACKNOWLEDGMENTS

This work is supported by the National Science Foundation under Grant Numbers DLR-0833612 and IIP-0848962. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

7. REFERENCES

[1] Torrance, E.P. "Creative teaching makes a difference." In J.C. Gowan, J. Khatena, & E.P. Torrance (Eds.), Creativity: Its Educational Implications (2nd ed., pp. 99-108). Dubuque, IA: Kendall-Hunt, 1981.

[2] Kaufman, J. C., Plucker, J. A., & Baer, J. "Essentials of creativity assessment." Hoboken, NJ: Wiley, 2008.

[3] Koh, K. H., Basawapatna, A., Bennett, V., Repenning, A., "Towards the Automatic Recognition of Computational Thinking for Adaptive Visual Language Learning." IEEE International Symposium on Visual Languages and Human-Centric Computing 2010, Leganés-Madrid, Spain, September 21-25, 2010

[4] Bennett, V., Koh, K. H., Repenning, A., "CS Education Re-Kindles Creativity in Public Schools." ITiCSE '11: Annual Conference on Innovation and Technology in Computer Science Education, Darmstadt, Germany, June 27-29, 2011.

[5] Runco, M.A. & Okuda, S.M. "Problem discovery, divergent thinking, and the creative process." Journal of Youth and Adolescence, 17:3, 211-220, 1988.

[6] Herring, S.R., Chang, C.C., Krantzler, J. & Bailey, B.P. "Getting inspired! Understanding how & why examples are used in creative design practice." CHI2009, 87-96, 2009.

[7] Csikszentmihalyi, M., and Getzels, J. W. "Discovery-oriented behavior and the originality of creative products: A study with artists." J. Personal. & Social. Psychol. 19: 47-52, 1971.

[8] Williams, F.E. "Creativity assessment packet examiner's manual." Austin TX: PRO-ED, 1993.

[9] Koh, K. H., Bennett, V., Repenning, A., "Computing Indicators of Creativity." ACM Creativity & Cognition 2011, The High Museum of Art · Atlanta, Georgia, USA, November 3-6, 2011

[10] Basawapatna, A., Koh, K. H., Repenning, A., Webb, D., Marshall, K., "Recognizing Computational Thinking Patterns." SIGCSE 2011: Reaching Out The 42nd ACM Technical Symposium on Computer Science Education, Dallas, Texas, USA, March 9-12, 201

[11] Koh, K. H., Bennett, V., Repenning, A., "Inspiring Collaborative Benefits: An Interaction between a Virtual and a Physical Group Learning Infrastructure." Western Canadian Conference on Computing Education (WCCCE 2010), Okanagan, B.C., Canada May 7-8, 2010.

[12] Landauer, T.K., Foltz, P.W. & Laham, D., "Introduction to Latent Semantic Analysis. Discourse Processes." 25, 1998, 259-284.

[13] Wing, J.M. "Computational Thinking." Communications of the ACM, 49:3, 33-35, 2006.

[14] Repenning, A. "AgentSheets®: an Interactive Simulation Environment with End-User Programmable Agents." In Proc. Interaction 2000, Tokyo, Japan, 2000.