Computers in Human Behavior 64 (2016) 423–431

Contents lists available at ScienceDirect

Computers in Human Behavior

journal homepage: www.elsevier.com/locate/comphumbeh

Full length article

The impact of social factors on pair programming in a primary school

Baichang Zhong [a,*], Qiyun Wang [b], Jie Chen [c]

[a] Collaborative Innovation Center for Talent Cultivating Mode in Basic Education, Nanjing Normal University, China
[b] National Institute of Education, Nanyang Technological University, Singapore
[c] Bixi Primary School of Changshu City, China

ARTICLE INFO

Article history:
Received 19 March 2016
Accepted 18 July 2016
Available online 26 July 2016

Keywords:
Primary education
Pair programming
Social factors
Gender
Partnership

ABSTRACT

Pair programming (PP) is a usefulness approach to fostering computational thinking (CT) for young students. However, there are many factors to impact the effectiveness of PP. Among all factors, the social factors are often ignored by researchers. Therefore, this study aimed to explore the impact of two social factors (gender and partnership) on PP in a primary school setting. To that end, we conducted PP experiments in four classes from the sixth grade in a Chinese primary school. The research results indicated: (a) there was no significant difference on compatibility among the gender pairs, but a significant difference among partnership pairs; (b) there was no significant difference on programming achievement and confidence among different pairs, and girls became more productive and confidence in PP; and (c) PP tightened up the partnership within pairs. These findings suggest that teachers should take partnership into account as an important factor in PP or other collaborative learning, and adopt PP as an effective approach to decrease the gender gap in programming courses, and make students socialize.

1. Introduction

Programming for K-12 can be traced back to the 1960s when Logo programming was first introduced as an intellectual thinking educational tool for teaching mathematics (Feurzeig, Papert & Lawler, 2011). After Logo, the use of programming to teach thinking skills in K-12 was scarcely reported. In recent years, however, there has been renewed interest in introducing programming to K-12 students (Grover & Pea, 2013; Kafai & Burke, 2013). This was fuelled by the availability of easy-to-use visual programming languages such as Scratch (Brennan & Resnick, 2012; Burke, 2012; Lee, 2010), Stagecast Creator (Denner, Werner, & Ortiz, 2012) and Alice (Bishop-Clark, Courte, Evans, & Howard, 2006; Graczyńska, 2010; Kelleher & Pausch, 2007).

During programming, students are exposed to computational thinking (CT), a term popularized by Wing (2006). CT involves solving problems, designing systems, and understanding human behaviors, by drawing on the concepts fundamental to computer science (Wing, 2006). The nuts and bolts in CT are defining abstractions, working with multiple layers of abstraction and understanding the relationships among the different layers (Wing, 2008). Many researchers thought CT is a fundamental skill for almost everyone in a digital age, not just for computer scientists (National Research Council, 2011, 2010, pp. 3–4; Wing, 2006). More importantly, CT is in line with many aspects of 21st century competencies such as creativity, critical thinking, and problem solving (Binkley et al., 2012). Thus, it is not surprising that many educators claim that programming provides an important context and set of opportunities to develop CT for K-12 students (Brennan & Resnick, 2012; Kafai & Burke, 2013; Lye & Koh, 2014; Resnick et al., 2009).

This revived interest in programming for K-12 settings suggests a need to consider how CT can be fostered effectively via programming. Studies showed that students taught with pair programming (PP) performed better in CT than solo programming (Lye & Koh, 2014; Werner & Denning, 2009; Werner, Denner, Campe, & Kawamoto, 2012). The PP is a practice in which two people work side-by-side at one computer, and intensely collaborate to create a program. One is normally the “driver”, who is responsible for using a computer to key in codes. The other is usually known as the “navigator”, and takes the responsibility for observing the driver's work and providing support by pointing errors or offering ideas in solving a problem (Williams & Kessler, 2000).

In view of the usefulness to foster CT, we have used PP as a pedagogical teaching technique in a primary school for two years. Meanwhile, we identified some questions with putting PP in

* Corresponding author. School of Education Science, Nanjing Normal University, No. 122, Ninghai Road, Nanjing, 210097, China.
E-mail address: zhongbc@163.com (B. Zhong).

http://dx.doi.org/10.1016/j.chb.2016.07.017
0747-5632/

practice. One main question is how to pair up students to get effective teaching, which is about how the variation of pair formations may lead to different teaching results. There are several factors to consider for pairing up students, including students' skills, experiences, personalities, genders and partnership. In this study, we focused on the social factors including gender and partnership when pairing up students in the PP practice.

2. Literature review

Many studies have showed that PP has obvious benefits over solo programming, including but not limited to the aspects as follows. PP can (1) significantly improve individual programming skills and promote productivity or program quality since it provides students with a clearly defined method to help one another understand the context of the problem and to reflect on the contribution of both programmers (Braught, Eby, & Wahls, 2008; Cliburn, 2003; Hannay, Dybå, Arisholm, & Sjøberg, 2009; Li, Plaue & Kraemer, 2013; Tomayko, 2002; Williams & Kessler, 2000; Williams, Kessler, Cunningham, & Jeffries, 2000); (2) reduce frustration experienced by novice programmers, increase student satisfaction, enjoyment, and foster positive attitudes in programming (McDowell, Werner, Bullock, & Fernald, 2006, 2002; Bishop-Clark, Courte, Evans, & Howard, 2006; DeClue, 2003; LeJeune, 2006; Preston, 2005); (3) increase retention of students (especially for female students) in computer science courses (Li et al., 2013; McDowell et al., 2006); and (4) emphasize the importance of communication, teamwork, cooperation, and adaptability in a technical environment to better prepare students to work as a team (Cliburn, 2003; Williams & Kessler, 2000).

However, the above benefits do not occur automatically. Some experiments and empirical studies reported contradictory results to PP (Balijepally, Mahapatra, Nerur, & Price, 2009; Sfetsos, Stamelos, Angelis, & Deligiannis, 2009). There were many factors impacting the effectiveness of PP such as:

(a) Task complexity (Arisholm, Gallis, Dybǎ, & Sjoberg, 2007; Hannay, Arisholm, Engvik, & Sjøberg, 2010);
(b) Partners' skills and experiences (Hannay et al., 2010; Lui & Chan, 2006; Williams, Layman, Osborne, & Katira, 2006);
(c) Partners' learning styles (Salleh, Mendes, & Grundy, 2011; Williams et al., 2006; Zualkernan, Allert, & Qadah, 2006); and
(d) Partners' personalities and temperaments (Choi, Deek, & Im, 2008; Hannay et al., 2010; Katira et al., 2004; Sfetsos et al., 2009; Thomas, Ratcliffe, & Robertson, 2003; Williams et al., 2006).

The factors (b), (c) and (d) are directly related to the pair formation. For example, students seemed to do their best work when paired with students of similar levels of self-esteem (Thomas et al., 2003). We found, however, that most of these empirical studies were based only on competency or personality factors, largely ignoring the social factors (gender, partnership, race, and culture etc.) involved. This one-sided consideration possibly induced inadequacies in the pair formation selection, and illustrated an incomplete picture for PP.

In a family of newer studies, social factors especially gender aspect were included in the experiments' variables system. Choi (2015) claimed that one particular topic that has not received much attention is the gender topic which is about how men and women interact and collaborate in programming and dyad team work. Using a pool of university programming courses students as the experiment participants, the study examined three gender pair types: female–female, female–male, and male–male. The result revealed that there was no significant gender difference in the PP coding output. But the same gender pair exhibited significantly higher levels of pair compatibility than the mixed gender pair.

Katira, Williams, and Osborne (2005) conducted a study involving 361 software engineering students at North Carolina State University to understand and predict pair compatibility. The results indicated that pairing up a female student with a partner who has a similar SAT/GRE/GPA will likely result in a compatible pair. Minority students perceived compatibility with a partner who has a similar GPA. Pairs with different genders were less likely to report compatibility. When both students in the pair were minority students, they were more likely to perceive compatibility. Minority students were comfortable working with students of the same gender.

Lewis (2011) conducted a study to investigate differences between PP and collaborative learning in two summer enrichment classes for students entering the sixth grade. Although this study did not investigate the partnership, they thought the interruptions and opportunities for discussion reinforced the pair relationship because they observed that a student would infrequently consult another student who was not his/her partner, even if he/she was the same distance away as the student's partner.

McDowell et al. (2006) collected data from 554 students who attempted the programming course at the University of California-Santa Cruz. Each participant was paired with a preferred partner. The study reported (1) those who paired produced significantly better programs than those who worked alone, but there was no significant gender difference as a whole in average programming scores; (2) men were significantly more confident than women in solo programming, however, the 24% increase in confidence that pairing afforded women was even greater than the 15% confidence boost experienced by men who had the benefit of PP. The result was a significant decrease of a gender gap in confidence when working in PP.

In this section we summarized important findings from empirical studies on PP, especially from those studies that include psychology and gender factors in their research frameworks. There still exist many issues that have not been empirically investigated, such as the impact of student partnership on compatibility, confidence and performance in PP. Considering the fact that PP is one of the major human-centric software development paradigms, social factors, especially gender and partnership, need to be further addressed. Therefore, this paper reported an experimental investigation of two social factors (i.e. gender and partnership) and their interaction on PP with an expectation of making students have a better effectiveness toward programming. In addition, PP showed promise for reducing gender differences among college students, while this study was initiated to examine this promising practice in the primary school setting.

3. Research aim and questions

The study aimed to explore the impacts of two social factors on PP effectiveness. PP effectiveness is expressed in terms of PP result, measured in this study by compatibility of pairs, performance of student programming, confidence toward programming, and partnership of pairs. Therefore, four research questions were to be answered:

(a) Is there any difference on compatibility among the gender pairs and partnership pairs?
(b) Is there any difference on learning performance among the gender pairs and partnership pairs?
(c) Is there any difference on confidence toward programing among the gender pairs and partnership pairs? and

(d) Does the PP tighten up the partnership within pairs?

4. Method

4.1. Participants

Participants were the sixth grade pupils coming from a primary school in China (it is called "school C" in this paper). We selected randomly 4 of 8 classes in the sixth grade and 154 pupils participated in the experiment (see Table 1). These students took the Alice introductory course for 12 class hours in the 2014 autumn semester before this study, and had basic knowledge and skills about Alice programming.

4.2. Procedure

We conducted a PP teaching experiment scheduled in the 2015 spring semester. The experiment lasted for 13 weeks in all (see Table 2).

In the first week, the participants firstly took part in a prior knowledge quiz. Then they were grouped into 77 pairs according to their scores of the quiz. To even out pairs' general experience level, the most skilled student were to pair up with the lowest skilled student, and the second best skilled student with the second lowest skilled student (Müller, 2005). This naturally yielded three different group categories: boy-boy, girl-girl, and boy-girl. We also conducted a partnership survey to get the partnership information within pairs, and divided pairs to three partnership levels: general, good, and very good. Therefore, we got a new sample distribution in the taxonomy of gender pairs and partnership levels (see Table 3). Finally, the participants were given a brief introduction to PP terminology and guidance prior to the Alice advanced course.

In the 2nd to 9th weeks, the Alice advanced course was implemented as a PP session. The participants learned the course for a class hour each week taught by the same teacher. Besides the PP, the teacher also used a three-stage learning progression called Use-Modify-Create to describe a pattern of engagement (Lee et al., 2011). During the PP, each pair had to exchange the role of driver and navigator after finishing a programming task.

In the 10th to 13th weeks, we conducted the posttests including achievement tests, partnership survey, compatibility survey, and confidence survey.

4.3. Materials

4.3.1. E-textbook

To implement this experiment, we developed a school-based curriculum "learning to storytelling by programming" based on the three-dimension framework of computational thinking (Brennan & Resnick, 2012). This curriculum included two sessions: one is the Alice introductory course for 12 class hours, and the other is the Alice advanced course for 8 class hours as mentioned above. For the convenience of improving the curriculum at any time, we developed it as an e-textbook via 3DPageFlip (see Fig. 1).

Table 1
Sample characteristics.

Classes	Class size	Boys	Girls
A	40	22	18
B	38	21	17
C	36	20	16
D	40	23	17
Total	154	86	68

The 3D programming language Alice2.4 developed at Carnegie Mellon was used in this curriculum. Alice (http://www.alice.org/) is an easy-to-learn environment which allows users to build 3D virtual worlds. Instead of creating traditional text-oriented programs which display meaningless messages such as "Hello World" to the screen, Alice allows the programmer to create and manipulate interesting objects (such as an ice skater) and create interesting environments in a short period of time, thereby increasing their satisfaction and motivation (Bishop-Clark, Courte, & Howard, 2006).

4.3.2. Handbook for PP

To make PP effective, we designed a student handbook to manage the process of PP, which included pair information, PP guidance, a sheet for the progress of PP (see Table 4), and a task design table. The handbook was delivered before each class hour and took back at the end of each class hour. The teacher checked the handbook each time, and interviewed the students individually if they could not fill the handbook in detail or had obstacles in PP.

4.4. Measures

A 3×3 factorial design was employed to examine the impacts of the types of gender pairs and levels of partnership on learning effectiveness in the PP (see Fig. 2). The different instruments (see Table 5) were given in the pretest and posttest besides the partnership survey which were used twice. The partnership survey and prior knowledge quiz were given at the beginning of the PP experiment (pretest), and the partnership survey, compatibility survey, confidence survey, and achievement test were conducted after the PP experiment (posttest). All students completed the pretest and posttest individually.

4.4.1. Partnership survey

The partnership survey was administered to get the partnership information within pairs. It included only one question described in the pretest as how about the partnership between the partners according to the newly grouping, and described in the posttest as how about the partnership after the PP course. There were three options behind the question including general, good and very good, which indicated different partnership levels. In order to get the accurate partnership level, we validated each student's answer. The partnership level within a pair was valid if two partners in a pair selected the same option. There were 70.13% pairs in the pretest and 73.85% pairs in the posttest select the same options, and the inter-rater reliability was 0.83 for the pretest and 0.87 for the posttest (see Table 3). For the inconsistent pairs, we individually interviewed the pairs to get an agreed partnership level.

4.4.2. Prior knowledge quiz

The prior knowledge quiz that included 11 multiple choices was used to measure the students' understanding of programming concepts. Participants' scores of the quiz were an important reference to group. The validity of the quiz was determined by experts' evaluation, and the reliability coefficient of the quiz was 0.62. For instance, one question read "In order to write one piece of code to draw a square with different length of side, we can set the length of side as a: a) variable, b) constant, c) parameter, d) instruction."

4.4.3. Compatibility survey

The participants were asked to evaluate their compatibility in PP. There were five thinking questions which were not counted in grading followed by one overall rating question which did count in grading. The instrument was developed by Kaufman et al. (1999). A 3-point scale was used in the thinking questions ranging from Very

Table 2
Schedule of the experiment.

Weeks	Contents
1rd	Prior knowledge quiz and students grouping, partnership survey, and PP guidance
2nd	Magic show (A magician waves the wand to make animals float.)
3rd	Confused kangaroo (Help the kangaroo to jump smoothly, and add new scenario and animation to it.)
4th	
5th	Frog escape (A snake is swimming to a frog basking in the sunshine.)
6th	Glider flying (Control the glider to fly via keyboard.)
7th	New typist (Design 3D characters and trigger its special effect via keyboard.)
8th	Squirrel the explorer (Control the squirrel action via mouse click.)
9th	Skating girl (Control the girl skating via event or keyboard.)
10th	Achievement test 1
11th	Achievement test 2
12th	Achievement test 3
13th	Partnership survey, compatibility survey, and confidence survey

Table 3
Sample distribution in 3 × 3 factorial design.

		Gender pair			
		Girl-girl	Boy-boy	Boy-girl	Total
Partnership level	General	10	26	40	76
	Good	14	14	8	36
	Very good	18	20	4	42
	Total	42	60	52	154

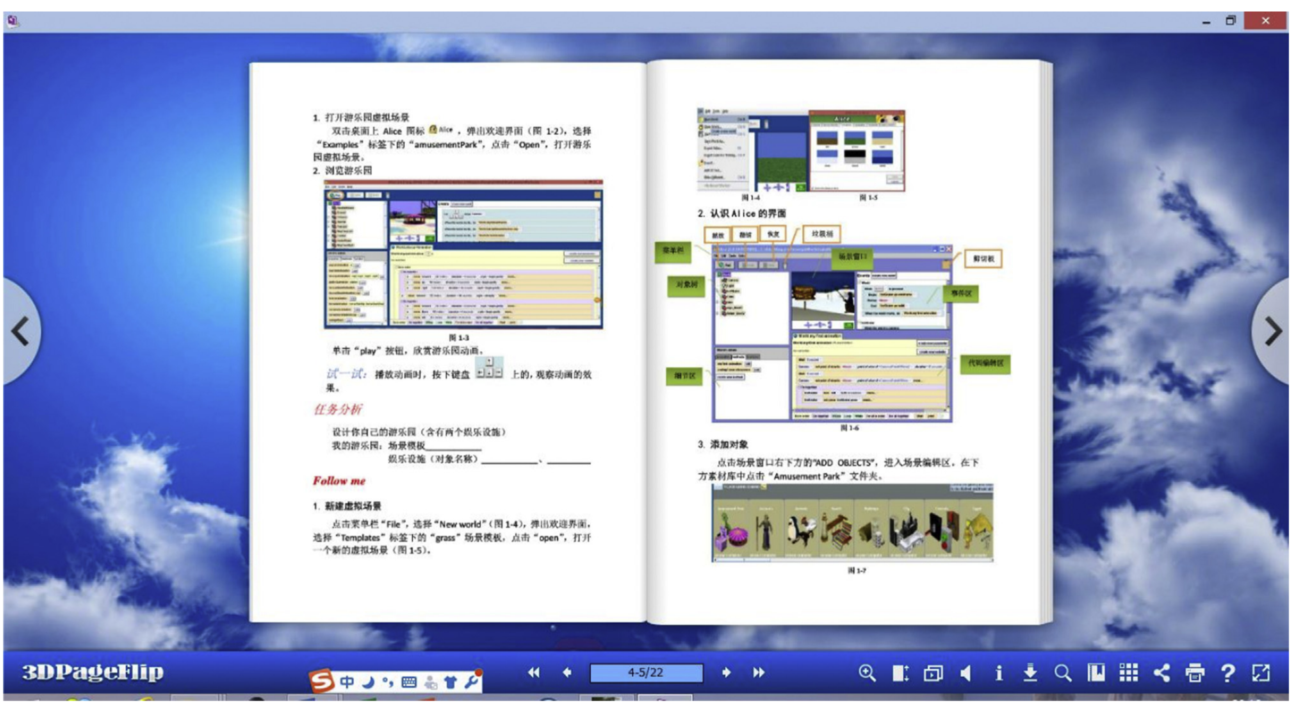

Fig. 1. A snapshot of the e-textbook.

Often (3), Occasionally (2) to Never (1), and in the overall rating question ranging from Very compatible (3), OK (2), to Not compatible (1). The reliability coefficient of the questionnaire was 0.76.

4.4.4. *Confidence survey*

The confidence survey was conducted to assess participants' confidence toward programing. The questionnaire was adopted from a previous PP study conducted by Williams et al. (2006). The participants were asked to rate themselves on a 9-point Likert-type scale with response options ranging from 1 (low) to 9 (high). The questionnaire consisted of five questions, of which the reliability coefficient was 0.99.

4.4.5. *Achievement test*

The achievement test was conducted after the PP course to assess participants' programming knowledge and capability. Each participant must finish the achievement test individually. The test tasks consisted of five tasks adopted from a previous study by Zhong et al. (2016) and covered three tasks, namely two closed tasks with a defined outcome and a defined process solely, two semi-open tasks with a defined outcome and a defined process solely, and one open task with an open outcome and open process.

These tasks have a common story context described as follows: The fairy princess comes to a forest, but a devil appears suddenly and takes her away. Then many fairies try their best to rescue the fairy princess (see Fig. 3 and Fig. 4), and finish five tasks.

Table 4
Sheet for the progress of PP.

Tasks[a]	Roles	Times (minutes)	Task completion	Comments
1	driver			
	navigator			
2	driver			
	navigator			
3	driver			
	navigator			
4	driver			
	navigator			
5	driver			
	navigator			

Note.
[a] There were a maximum of 5 tasks in a class hour. This sheet must be filled in class by the navigator.

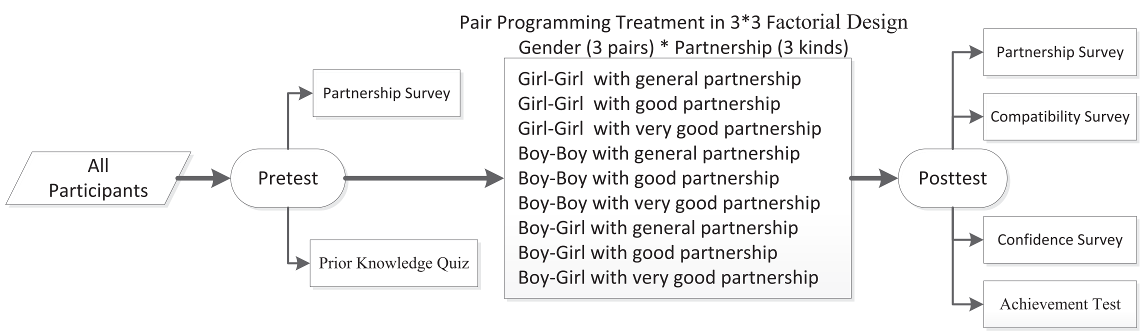

Fig. 2. Overview of experimental design.

Table 5
Instruments for data collections.

Construct	Source	Pretest/Posttest	Cronbach's alpha
Prior Knowledge (11 questions)	Designed by ourselves	Pretest	0.62
Partnership (1 question)	Designed by ourselves	Pretest	0.83
		Posttest	0.87
Compatibility (5 questions)	Kaufman, Felder, & Fuller, 1999	Posttest	0.76
Confidence (5 questions)	Williams et al., 2006	Posttest	0.99
Achievement (5 tasks)	Zhong, Wang, Chen, & Li, 2016	Posttest	0.79

(a)Two closed tasks: One is to make the female fairy fly to the devil, and the other is to make the male fairy choose the biggest apple from the tree and give it to the devil as a give-and-take condition.

(b)Two semi-closed tasks: One is to make the female fairy fly to the waterfall by event-driven, and the other is to make the female fairy catch the fish in the waterfall as the other give-and-take condition.

(c)One open task (Task 5): The devil, however, does not fulfill the promise to release the fairy princess and assign a fiery dragon to make the rescue more difficult. Students design a scenario to describe the fairies how to defeat the fiery dragon.

Each closed task and semi-open task worth 5 points, and the open task worth 20 points. The validity of the test was determined by experts, and the reliability coefficient of the test was 0.79.

All participants were also asked to provide demographic data including gender, class, and group number. Participants' responses were coded to uniquely identify them so that data could be compared in different pairs.

5. Results

5.1. Compatibility of pairs

The compatibility scores were subjected to a 3 × 3 independent-groups ANOVA with gender pair and partnership pair as between-subjects factors. A significant main effect was found for partnership pair, $F(2, 145) = 5.25$, $p < 0.01$, $\eta 2 = 0.161$. Table 6 shows that the very good partnership pairs were more compatible ($M = 2.78$) than the general partnership pairs ($M = 2.28$) and the good partnership pairs ($M = 2.45$). It was not significant for gender pair, $F(2, 145) = 1.47$, $p > 0.05$. The interaction also fell short of significance, $F(4, 145) = 0.39$, $p > 0.05$.

5.2. Performance of student programming

We conducted a 3 × 3 independent-groups ANOVA on the achievement test. Both main effects and interaction were not significant, for partnership pair, $F(2, 145) = 0.61$, $p > 0.05$, for gender pair, $F(2, 145) = 0.03$, $p > 0.05$, for interaction, $F(4, 145) = 0.32$, $p > 0.05$. There was also no any significant difference on the achievement test scores between the boys and the girls ($t = 0.68$, $p > 0.05$).

5.3. Confidence toward programming of pairs

We conducted a 3 × 3 independent-groups ANOVA on the programming confidence. Both main effects and interaction were not significant, for partnership, $F(2, 145) = 0.23$, $p > 0.05$, for gender pair, $F(2, 145) = 2.25$, $p > 0.05$, for interaction, $F(4, 145) = 1.31$, $p > 0.05$.

5.4. Partnership of pairs

A paired-samples T test revealed the difference of partnership between the pretest and the posttest was significant, $t = -3.67$, $p < 0.001$ (see Table 7). The partnership within pairs became closer after PP than before.

Because the gender pairs had been considered desirable, we divided the partnership into six groups, girl-girl pretest, boy-boy pretest, boy-girl pretest, girl-girl posttest, boy-boy posttest, and boy-girl posttest. We conducted a supplementary univariate ANOVA on the six groups of partnership. A significant main effect was found, $F(1, 148) = 20.04$, $p < 0.001$, $\eta^2 = 0.824$. Table 8 shows that there were 4 homogeneous subsets, the partnership within boy-girl pairs became significantly closer after PP ($M = 1.52$) than

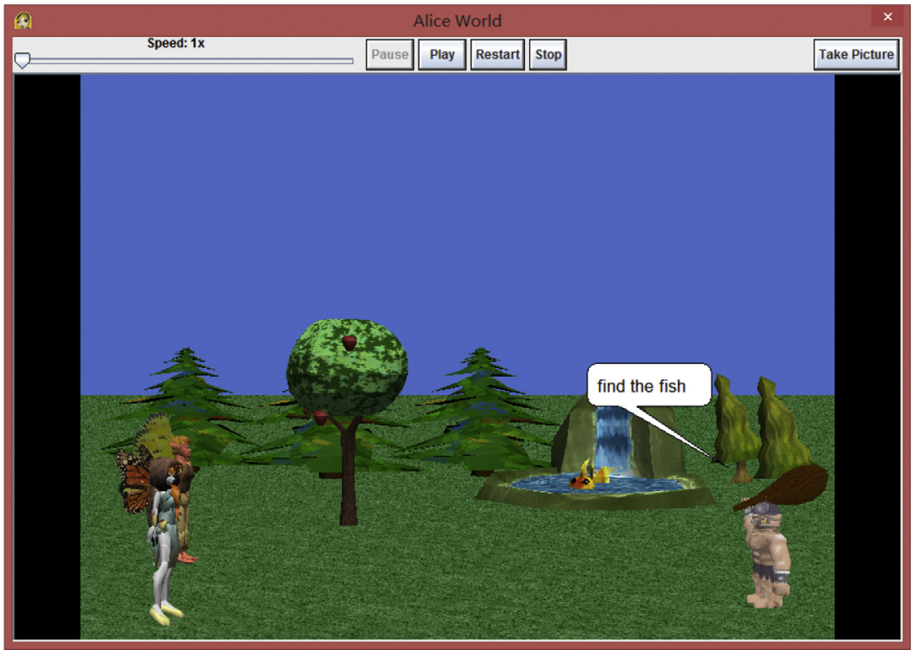

Fig. 3. Scenario of fairy princess rescue for tasks 1 to 4.

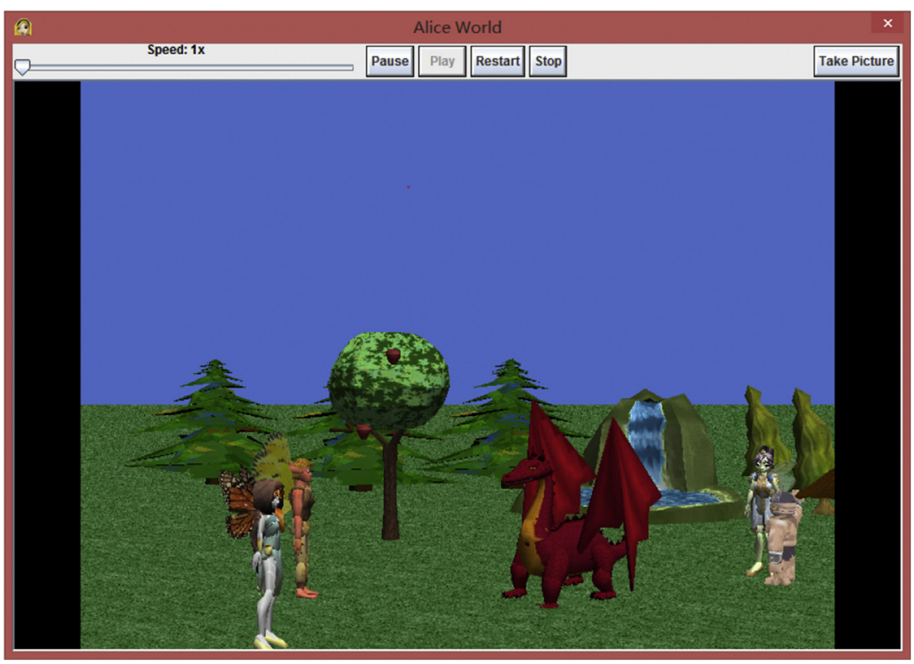

Fig. 4. Scenario of fairy princess rescue for tasks 5.

Table 6
SNK post Hoc test for partnership (pretest).

Partnership pair	N	Subset*	
		1	2
General	80	2.28	
Good	38	2.45	
Very good	36		2.78
Sig.		0.169	1.00

*$p < 0.05$.

Table 7
Difference of partnership between pretest and posttest.

Partnership	N	Mean	Std. Dev.	t	Sig.
Pretest	154	1.73	0.827	−3.67	0.000***
Posttest	154	1.96	0.808		

***$p < 0.001$.

Table 8
SNK post Hoc test for partnership (pretest &posttest).

Partnership	N	Subset*			
		1	2	3	4
Boy-girl pretest	52	1.15			
Boy-girl posttest	52		1.52		
Boy-boy pretest	60			1.92	
Boy-boy posttest	60			2.00	
Girl-girl pretest	42			2.17	
Girl-girl posttest	42				2.45
Sig.		1.00	1.00	0.192	1.00

*$p < 0.05$.

before ($M = 1.15$), and the partnership within girl-girl pair was the closest than other pairs. In general, however, the same gender pairs had closer partnership than different gender pairs in the pretest and the posttest, and there were no effect of opposites attract for the pupils.

6. Discussion

On compatibility of pairs, the result of this study is inconsistent with others studies. Choi (2015) found the same gender pair (female–female and male–male) exhibited significantly higher levels of pair compatibility than the mixed gender pair (female–male). Katira et al. (2005) also claimed that pairs with different gender were less likely to report compatibility. In this study, however, there was no significant difference on compatibility among three gender pairs, but a significant difference among pair partnership which implied more close partnership and more compatible in PP.

This issue needs further exploration. There are two possible reasons for this result. Firstly, perhaps partnership is a mediating variable which the previous studies did not take into account. In other words, partnership may be a more essential factor than gender toward compatibility in PP. Secondly, the student participants in the existing studies were adults, but the student participants were under age in this study. Pair studies from psychology, behavioral science, and cognitive science disciplines reported the fact that there was a significant gender effect on human interaction and collaboration (Carli, 1989; Underwood, McCaffrey, & Underwood, 1990). Some studies on gender diversity also showed that, instead of male dominating groups, groups with high percentage of women produced more favorable financial returns (Campbell & Minguez-Vera, 2008; Francoeur, Labelle, & Sinclair-Desgagne, 2008; Herring, 2009). These studies, however, were also conducted based on adult participants. There is no evidence indicating that a gender effect exists between female pupils and male pupils. That is to say, the age may be a suppressor variable toward compatibility in PP. To confirm these possible reasons, we need more PP experiments in different ages to test the two social factors. Anyway, partnership is an important factor to consider when pairing up students in PP or other collaborative learning settings.

On achievement and confidence of student programming, many studies reported that males outperform females in solo programming related fields. Yelland (1993) found that there was a gender difference in programming performance: girls were less likely to take risks to achieve a goal than boys when programming with LOGO. McDowell et al. (2006) reported those who paired produced significantly better programs than those who worked alone, and men were significantly more confident than women in solo programming. Rubio, Romero-Zaliz, Mañoso, and de Madrid (2015) also found that male and female students have different perceptions and learning outcomes: male students found programming easier, had a higher intention to program in the future, and showed higher learning outcomes than female students. However, the gender difference on programming performance and confidence in this study is not significant, which is in line with the earlier studies on undergraduate PP (Choi, 2015; McDowell et al., 2006).

An interesting finding of this study is that girls became more confident and get higher performance in PP than solo programming. In the context of PP, conflicts and different opinions on the coding strategy or practice are expected. The situation also demands the compromise of two different programmers, and can additionally pose “pair pressure” (Wray, 2010). To achieve a sound PP environment, these difficulties must be managed. The girls demonstrate better skills than boys to mitigate these difficulties. That is why we observed a phenomenon that boys tended to blame partners for code errors more frequently than girls especially in the initial weeks of PP. This hints that gender can be an issue in PP context. In fact, studies revealed that female students tend to focus on intimacy and solidarity during collaboration, and support others with effective persuasive messages, but male students to focus on hierarchy and independence (Gefen & Straub, 1997; Rosenberg-Kima, Plant, Doerr, & Baylor, 2010). In short, this finding suggests that PP is a good way to decrease the gender gap in programming courses, and teachers should make no discriminative expectations, opportunities and concerns for boys and girls on learning programming.

On partnership of pairs, this study extends the perception since it was rare to focus on partnership in previous studies. Partnership is an inevitable factor to consider in PP. On the one hand, good partnership helps to improve compatibility of PP as mentioned above. On the other hand, PP tightens up the partnership within a pair as indicated in Table 7. That is to say, PP is a useful approach to making students socialize and get a good friend in learning. An important reason for PP reinforcing the pair relationship is that students often need to communicate with their partners frequently as observed by Lewis (2011).

We also found the partnership within girl-girl pair was closer than other pairs as indicated in Table 8. We believe that there was a high level of intimacy between the two identical gender partners. Meanwhile, we also observed that there were much more talk and discussion within girl-girl pairs than other pairs. It is in line with the opinion that talk is the essence of relationships for women than it is for men (Johnson, 1996). Certainly, it is also an important reason to make girls maintain a good partnership since they tend to focus on intimacy and solidarity as mentioned above.

7. Conclusions

Limited studies have focused on social factors. Considering the fact that PP is one of the major human-centric software development paradigms, social factors should not be ignored. Therefore, this study aimed to explore the impact of two social factors (gender and partnership) on PP in the primary school setting. To that end, we conducted a PP experiment in four classes from the sixth grade in a primary school. The main results are summarized as follows:

(a) There was no significant difference on compatibility among the gender pairs, but a significant difference among partnership pairs which implied more close partnership and more compatible in PP. Thus, partnership is an important factor to consider when pairing up students in PP or other collaborative learning settings.
(b) There was no significant difference on programming achievement and confidence among different pairs, and girls

became more productive and confidence in PP. This finding suggests that PP is a good way to decrease the gender gap in programming courses, and teachers should make no discriminative expectations, opportunities and concerns for boys and girls on learning programming.

(c) PP tightens up the partnership within pairs. Therefore, PP can be a useful approach to make students socialize and get a good friend in learning.

8. Limitations and future research

In this research, the participants were primary school pupils, and the conclusions are suitable for the particular group. The results may be different if the participants are middle school students or high school students. We will further explore other options in future research.

Acknowledgements

The authors disclose and gratefully acknowledge the receipt of the following financial support for the research: the project "Collaborative Innovation Center for Talent Cultivating Mode in Basic Education, Nanjing Normal University, China", and the Priority Academic Program Development of Jiangsu Higher Education Institutions in China.

References

Arisholm, E., Gallis, H., Dybå, T., & Sjoberg, D. I. (2007). Evaluating pair programming with respect to system complexity and programmer expertise. *IEEE Transactions on Software Engineering, 33*(2), 65–86.

Balijepally, V., Mahapatra, R., Nerur, S., & Price, K. H. (2009). Are two heads better than one for software development? the productivity paradox of pair programming. *MIS Quarterly, 33*(1), 91–118.

Binkley, M., Erstad, O., Herman, J., Raizen, S., Ripley, M., Miller-Ricci, M., et al. (2012). Defining twenty-first century skills. In P. Griffin, B. McGaw, & E. Care (Eds.), *Assessment and teaching of 21st century skills* (pp. 17–66). Netherlands: Springer.

Bishop-Clark, C., Courte, J., & Howard, E. V. (2006). Programming in pairs with Alice to improve confidence, enjoyment, and achievement. *Journal of Educational Computing Research, 34*(2), 213–228.

Braught, G., Eby, L. M., & Wahls, T. (2008). The effects of pair-programming on individual programming skill. *ACM SIGCSE Bulletin, 40*(1), 200–204.

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 annual meeting of the American educational research Association, Vancouver, Canada*. Retrieved from http://scratched.gse.harvard.edu/ct/files/AERA2012.pdf.

Burke, Q. (2012). The markings of a new pencil: Introducing programming-as-writing in the middle school classroom. *The Journal of Media Literacy Education, 4*(2), 121–135.

Campbell, K., & Mínguez-Vera, A. (2008). Gender diversity in the boardroom and firm financial performance. *Journal of business ethics, 83*(3), 435–451.

Carli, L. L. (1989). Gender differences in interaction style and influence. *Journal of Personality and Social Psychology, 56*(4), 565.

Choi, K. S. (2015). A comparative analysis of different gender pair combinations in pair programming. *Behaviour & Information Technology, 34*(8), 825–837.

Choi, K. S., Deek, F. P., & Im, I. (2008). Exploring the underlying aspects of pair programming: The impact of personality. *Information and Software Technology, 50*(11), 1114–1126.

Cliburn, D. C. (2003). Experiences with pair programming at a small college. *Journal of Computing Sciences in Colleges, 19*(1), 20–29.

DeClue, T. H. (2003). Pair programming and pair trading: Effects on learning and motivation in a CS2 course. *Journal of Computing Sciences in colleges, 18*(5), 49–56.

Denner, J., Werner, L., & Ortiz, E. (2012). Computer games created by middle school girls: Can they be used to measure understanding of computer science concepts? *Computers & Education, 58*(1), 240–249.

Feurzeig, W., Papert, S. A., & Lawler, B. (2011). Programming-languages as a conceptual framework for teaching mathematics. *Interactive Learning Environments, 19*(5), 487–501.

Francoeur, C., Labelle, R., & Sinclair-Desgagne, B. (2008). Gender diversity in corporate governance and top management. *Journal of business ethics, 81*(1), 83–95.

Gefen, D., & Straub, D. W. (1997). Gender differences in the perception and use of e-mail: An extension to the technology acceptance model. *MIS Quarterly, 21*(4), 389–400.

Graczyńska, E. (2010). ALICE as a tool for programming at schools. *Natural Science, 2*(02), 124.

Grover, S., & Pea, R. (2013). Computational thinking in K-12: A review of the state of the field. *Educational Researcher, 42*(1), 38–43.

Hannay, J. E., Arisholm, E., Engvik, H., & Sjøberg, D. I. (2010). Effects of personality on pair programming. *IEEE Transactions on Software Engineering, 36*(1), 61–80.

Hannay, J. E., Dybå, T., Arisholm, E., & Sjøberg, D. I. (2009). The effectiveness of pair programming: A meta-analysis. *Information and Software Technology, 51*(7), 1110–1122.

Herring, C. (2009). Does diversity pay?: Race, gender, and the business case for diversity. *American Sociological Review, 74*(2), 208–224.

Johnson, F. (1996). Friendships among Women: Closeness in dialogue. In J. T. Wood (Ed.), *Gendered relationships* (pp. 79–94). Mountain View, CA: Mayfield.

Kafai, Y. B., & Burke, Q. (2013). Computer programming goes back to school. *Phi Delta Kappan, 95*(1), 61–65.

Katira, N., Williams, L., & Osborne, J. (2005). Towards increasing the compatibility of student pair programmers. In *Proceedings of the 27th international conference on Software engineering* (pp. 625–626). ACM.

Katira, N., Williams, L., Wiebe, E., Miller, C., Balik, S., & Gehringer, E. (2004). On understanding compatibility of student pair programmers. *ACM SIGCSE Bulletin, 36*(1), 7–11.

Kaufman, D. B., Felder, R. M., & Fuller, H. (1999). Peer ratings in cooperative learning teams. In *Proceedings of the 1999 American society for engineering education, Charlotte, NC*.

Kelleher, C., & Pausch, R. (2007). Using storytelling to motivate programming. *Communications of the ACM, 50*(7), 58–64.

Lee, Y. J. (2010). Developing computer programming concepts and skills via technology-enriched language-art projects: A case study. *Journal of Educational Multimedia and Hypermedia, 19*(3), 307–326.

Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., et al. (2011). Computational thinking for youth in practice. *ACM Inroads, 2*(1), 32–37.

LeJeune, N. F. (2006). Teaching software engineering practices with Extreme Programming. *Journal of Computing Sciences in Colleges, 21*(3), 107–117.

Lewis, C. M. (2011). Is pair programming more effective than other forms of collaboration for young students? *Computer Science Education, 21*(2), 105–134.

Li, Z., Plaue, C., & Kraemer, E. (2013). A spirit of camaraderie: The impact of pair programming on retention. In *Software engineering education and training (CSEE&T), 2013 IEEE 26th conference on* (pp. 209–218). IEEE.

Lui, K. M., & Chan, K. C. (2006). Pair programming productivity: Novice–novice vs. expert–expert. *International Journal of Human-computer studies, 64*(9), 915–925.

Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior, 41*, 51–61.

McDowell, C., Werner, L., Bullock, H., & Fernald, J. (2002). The effects of pair-programming on performance in an introductory programming course. *ACM SIGCSE Bulletin, 34*(1), 38–42.

McDowell, C., Werner, L., Bullock, H., & Fernald, J. (2006). Pair programming improves student retention, confidence, and program quality. *Communications of the ACM, 49*(8), 90–95.

Müller, M. M. (2005). Two controlled experiments concerning the comparison of pair programming to peer review. *Journal of Systems and Software, 78*(2), 166–179.

National Research Council. (2010). *Report of a workshop on the scope and nature of computational thinking* (pp. 3–4). Washington, D. C.: National academies press.

National Research Council. (2011). *Report of a workshop on the pedagogical aspects of computational thinking* (pp. 6–7). Washington, D. C.: National academies press.

Preston, D. (2005). Pair programming as a model of collaborative learning: A review of the research. *Journal of Computing Sciences in colleges, 20*(4), 39–45.

Resnick, M., Maloney, J., Monroy-Hernandez, A., Rusk, N., Eastmond, E., Brennan, K., et al. (2009). Scratch: Programming for all. *Communications of the ACM, 52*(11), 60–67.

Rosenberg-Kima, R. B., Plant, E. A., Doerr, C. E., & Baylor, A. L. (2010). The influence of computer-based Model's race and gender on female students' attitudes and beliefs towards engineering. *Journal of Engineering Education, 99*(1), 35–44.

Rubio, M. A., Romero-Zaliz, R., Mañoso, C., & de Madrid, A. P. (2015). Closing the gender gap in an introductory programming course. *Computers & Education, 82*, 409–420.

Salleh, N., Mendes, E., & Grundy, J. (2011). Empirical studies of pair programming for CS/SE teaching in higher education: A systematic literature review. *IEEE Transactions on Software Engineering, 37*(4), 509–525.

Sfetsos, P., Stamelos, I., Angelis, L., & Deligiannis, I. (2009). An experimental investigation of personality types impact on pair effectiveness in pair programming. *Empirical Software Engineering, 14*(2), 187–226.

Thomas, L., Ratcliffe, M., & Robertson, A. (2003). Code warriors and code-a-phobes: A study in attitude and pair programming. *ACM SIGCSE Bulletin, 35*(1), 363–367.

Tomayko, J. E. (2002). A comparison of pair programming to inspections for software defect reduction. *Computer Science Education, 12*(3), 213–222.

Underwood, G., McCaffrey, M., & Underwood, J. (1990). Gender differences in a cooperative computer-based language task. *Educational Research, 32*(1), 44–49.

Werner, L., Denner, J., Campe, S., & Kawamoto, D. C. (2012). The fairy performance assessment: Measuring computational thinking in middle school. In *Proceedings of the 43rd ACM technical symposium on computer science education* (pp. 215–220). NY: ACM.

Werner, L., & Denning, J. (2009). Pair programming in middle school: What does it look like? *Journal of Research on Technology in Education, 42*(1), 29–49.

Williams, L. A., & Kessler, R. R. (2000). All I really need to know about pair programming I learned in kindergarten. *Communications of the ACM, 43*(5), 108–114.

Williams, L. A., Kessler, R. R., Cunningham, W., & Jeffries, R. (2000). Strengthening the case for pair programming. *IEEE software, 4*, 19–25.

Williams, L., Layman, L., Osborne, J., & Katira, N. (2006). Examining the compatibility of student pair programmers. In *Agile conference, 2006*. IEEE.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33–35.

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A, 366*(1881), 3717–3725.

Wray, S. (2010). How pair programming really works. *IEEE software, 27*(1), 50–55.

Yelland, N. (1993). Young children learning with LOGO: An analysis of strategies and interactions. *Journal of Educational Computing Research, 9*(4), 465–486.

Zhong, B., Wang, Q., Chen, J., & Li, Y. (2016). An exploration of three-dimensional integrated assessment for computational thinking. *Journal of Educational Computing Research, 53*(4), 562–590.

Zualkernan, I., Allert, J., & Qadah, G. Z. (2006). Learning styles of computer programming students: A middle eastern and american comparison. *IEEE Transactions on Education, 49*(4), 443–450.