US-China Education Review B 5 (2011) 613-626
Earlier title: US-China Education Review, ISSN 1548-6613

Pair Programming and LSs in Computing Education: Its Impact on Students' Performances

Tie Hui Hui
SEGi College Penang, Penang, Malaysia

Irfan Naufal Umar
Universiti Sains Malaysia, Penang, Malaysia

Learning to programme requires complex cognitive skills that computing students find it arduous in comprehension. PP (pair programming) is an intensive style of programme cooperation where two people working together in resolving programming scenarios. It begins to draw the interests of educators as a teaching approach to facilitate learning and improve programming performance. The approach of PP, its model, benefits and limitations as well as the LS (learning style) preference are presented in the first part of this paper. The research findings and discussion on the application of PP involving 96 first year computing students are incorporated in the second part of this paper. The participants in these two intact classes were randomly assigned either to the experimental group that received PP or to the control group that received DI (direct instruction) method only. In PP group, students worked in pairs based on the visual-verbal LS dimension and those of DI group work individually. During a seven-week treatment, both groups applied program flowcharts and pseudocode in solving programming tasks. This study used two assessment methods—the formative and summative to examine the students' programming achievements. Two programming assignments were used as a formative assessment tool, also the CPPT (computer programming performance test) as the second tool which comprises of a pre-test, an immediate post-test and a delayed post-test, was administrated to assess the students' programming recall and retention. The result findings indicated that students in PP group significantly outperformed those in DI group for both the formative and summative assessments. However, only the visual and verbal students performed significantly better in recall than the retention. The analysis on the interaction effects revealed that learning is within inner self with regard to the instructional strategies applied and LS preference in classroom environment. In this case, the effectiveness of instructional strategies adopted to foster learning somehow depends on the type of learners. Therefore, educators should reflect on individual learning abilities while applying PP to stimulate students' engagement and critical thinking skills that subsequently will have positive influence on academic performance.

Keywords: PP (pair programming), cooperative learning, LS (learning style), computer programming

Introduction

Computer programming being the essential element of the computing courses curriculum in science education is the main building block for most students who are pursuing programming courses. It takes courage and audacity for the first year computing students as novices to learn how to programme. As programming demands complex cognitive skills, planning, logical reasoning and problem-solving eventually play their roles

Tie Hui Hui, Ph.D., senior lecturer, Centre for Postgraduate Studies, SEGi College Penang.
Irfan Naufal Umar, Ed.D., associate professor, Centre for Instructional Technology and Multimedia, Universiti Sains Malaysia.

in the process of learning. Understanding the basic programming concept is one of the three primary pedagogical goals of learning and teaching programming, besides creative, logical and critical thinking skills (Miliszewska & Tan, 2007; Fincher, 2006). McGill and Volet (1997) noted that computing students encounter intricacies in acquiring conceptual understanding over the novel concepts while learning the most fundamental syntaxes. In fact, problem-solving and critical thinking skills which these novices lack are the ultimate skills sought after in following programming courses. To increase productivities, programming skills and higher quality of codes and minimize errors and deduce coding anxiety, programmers in software development industry frequently cooperate in their daily tasks on the design, coding and testing.

PP (pair programming) is a structured mode of cooperative learning approach that could be an added benefit when it is introduced and used in a form of method as the instructional strategy in educational environment (Kuittinen & Sajaniemi, 2004). Meseka, Nafziger, and Meseka (2010) as well as Hamer (2006), revealed that cooperative activities in classrooms typically promote richer social interaction, enrich critical thinking and enhance attitude of students in the process of programme development. Pairing activity assists both members to interact, discuss and brainstorm on the given problem scenarios in order to accomplish common programming goals. With an effective interaction between pair partners, it improves conceptual understanding on the abstract concepts in each learning opportunity. This further develops higher cognitive knowledge that leads to improve the students' programming proficiency and subsequently reveals positive impact on their academic performance. Through in-depth discussions and details explanation with their peers, Ballantine and Larres (2007) found that the high achievers benefited the most from their participation in the heterogeneous pairs, specifically with the diversity of thought that fosters the learning process. Besides, social interaction is another skill that computing students are lacking. Thus, these skills are crucial for students who are to be successful in future software development profession.

In general, LS (learning style) is described as an individual preference in approaching and acquiring new knowledge. It is a relatively stable indicator of how learners receive, organize, interact with and present information in the context of learning (Ladd & Ruby, 1999; Kolb, 1984). Empirical findings revealed that differences in LSs are correlated with students' academic progress and subsequently affect their programming performances (Pallapu, 2007). Educators are persistently looking for alternatives in ensuring that students attain better results in programming comprehension and logical thinking which are vitally needed in the software industry. PP as cooperative learning in due course could nurture these key skills that eventually inspire students to thrill while learning programming.

PP in Education

PP, as cooperative learning, is a style in which two people work together at one computer station (Beck, 2005). They consistently cooperate on the design, algorithm, code and test. One person acts as the "driver", writing the design of codes manually. While, the other person acts as the "navigator" who observes, guides his partner and actively engages in selecting the best methods for coding the programme as well as simultaneously scans for errors in the programming codes. Intermittently, these roles are interchanged between the "driver" and "navigator" for the benefits of both partners. Through pair activities, the "driver" and the "navigator" communicate and brainstorm at any one time on the problem scenarios towards the development of independent learning in problems-solving.

In this study, students were to work in pairs and solve programming problems by applying program flowchart and pseudocode without using computers. Figure 1 shows that the students work face to face, discuss and interact in pairs to solve programming problems or complete computer programming assignments in class sessions (without practical sessions). During the pair discussion (social interaction), a shared set of common associations or attributes developed by both partners is used as a communication tool which allows the transferring of knowledge between both members (see Figure 2). Once the common terms used for the narrative of the problems scenarios are established and created in the mental model of both the driver and navigator, the interaction of conceptual schemata between them become possible. Working cooperatively in pairs that interactively engage to verbalize problems scenarios builds complex mental schemas which trigger higher logical thinking and reasoning skills (Gibbons, 2008; Gillies, 2007; Ballantine & Larres, 2007). This in turn improves the students' confidence levels and attitudes towards programming.

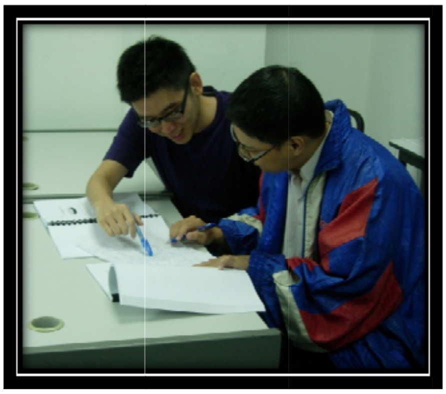

Figure 1. Pair discussions to solve programming tasks.

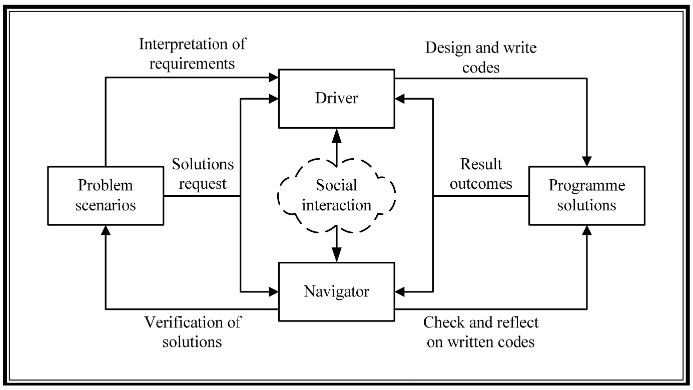

Figure 2. Process flow of PP (Chong, Plummer, Leifer, Klemmer, Eris, & Toyer, 2005).

Roles of driver and navigator. PP applies the concepts of "driver" and "navigator" for two students to work cooperatively in solving and completing programming tasks. The terms "driver" and "navigator" elaborate the role of each partner in the pair. The student who is the "driver" has current control on writing the program codes, while the "navigator" contributes to the tasks verbally by checking through the program logic of the written codes (Bryant, du Boulay, & Romero, 2006). The student, who acts as the "driver", writes the design and develops the program codes based on the problem scenarios. He/She converts the novel problems into program solutions through the use of program flowcharts and pseudocode. At the same time, the other student who acts as the "navigator", observes, guides the "driver" and provides constructive feedback on the programming codes (Adams, Lubega, Walmsley, & Williams, 2004). This could be further concluded that the "driver" works on the syntactic/conceptual knowledge, while the "navigator" contributes to the strategic/conditional knowledge as indicated in the McGill and Volet's (1997) conceptual framework.

The "driver" and "navigator" communicate and brainstorm at any one time in order to stimulate higher cognitive thinking and make their comprehension strategy explicit which enhance the learning process. The role rotation is done periodically between the driver and the navigator so to further promote effective learning where the knowledge is constantly exchange between the members in pairs (Xu & Rajlich, 2005; Srikanth, Williams, Wiebe, Miller, & Balik, 2004).

The benefits from pairing activities. A wide range of potential benefits to PP have been identified and reported in the literature. In an academic environment, working in pairs helps the students have deeper understanding of the design and solve complex programming tasks in shorter timeframe with less errors (Bryant, Romero, & du Boulay, 2007; Tomayko, 2002). Research results indicated that students working in pairs develop higher quality codes resulted in better programming performance, increasing satisfaction, reducing frustration, increasing confidence on program outcomes and superior results on graded assignments than those who work individually (VanDeGrift, 2004; Williams & Kessler, 2003). Discussion between partners in pairs enforces peer learning which offers richer social interaction, thus, helps each member to solve novel problems and complete the given tasks together. These results have proven that PP positively impact students' learning. Alternatively, this pairing activity is another promising instructional method to be considered for effective learning programming skills.

The limitation of PP. Despite of the fact that PP shows positive effects on learning processes and programming skills, there are still some foreseen obstacles in this instructional method. The first is the technique used to pair students. Pairing the students based on their past experience and knowledge is one of the options, which could maximize the individual programming abilities in solving novel situations. This further allows the students to learn from each other in a team of two. Another option is to let the students decide whom to pair with and which is to minimize the incompatibility between members in the pair. The second concern is the scheduling. The two partners need to have the common working schedule. They may not be able to work in pairs on the given tasks when their schedules overlap. The third drawback is the level of difficulties of programming scenarios. Theoretically, PP approach is ideal for solving novel problems that generate a medium programme size of codes (Xu & Rajlich, 2005). However, some novel problems may positively encourage and motivate students to comprehend better than that of others. Costing (the fourth issue) is the increase in labor cost. In practice, people are paid to do the job of one, while working in pairs requires extra manpower and in turn generates additional labor cost (Begel & Nagappan, 2010). Lastly is the difficulty in assessing the students' individual effort and programming abilities. It is not easy to measure the contribution of each partner in PP situations, which the partners may receive undeserving credits for the successful completion of programmes (Hahn, Mentz, & Meyer, 2009; McDowell, Hanks, & Werner, 2003). As such, two different assessment methods could be taken into consideration: One is to assess the individual performance while working in pairs, and the other is for pair assessments (Verhaart, Hagen, & Giles, 2005; Parsons, 2004).

Learning Style (LS)

LS is a preferred learning mode in which students respond to and use stimuli in the context of learning. For James and Gardner (1995) and Kolb (1984), it is the way in which students perceive, process, store and recall facts. In educational psychology, styles have been classified and acknowledged as the prime construct that serve as relatively stable indicators of how students respond to the learning environment (Deborah, 2005; Felder, 1993). This particular style reflects students' cognitive styles and shapes their own approach in acquiring and processing information and strongly influences their programming progresses when they approach a learning task.

Visual LS

Visual learners get more information and remember best from what they see, such as pictures, diagrams, flowcharts and demonstrations. Most students undertaking science courses are visual learners (Richardson, 1984; Barbe & Milone, 1981), while most lectures in basic computing curriculum are verbal—The information presented is predominantly auditory or visual presentation of auditory information (Tie & Irfan Naufal Umar, 2010). It has no surprise that students fail to reproduce information which was presented to them not long before as it may have been probably forgotten.

Verbal LS

Verbal learners remember and understand best from what they hear and then say. They enjoy activities which emphasize on discussion and get a lot out of it. Furthermore, these students learn effectively by explaining things to others. Information presented in most courses is verbal—spoken words in lectures, written words, formulae in texts and on the whiteboard. As such, both visual and verbal modalities representation are to be added to accommodate all students irrespective of their LSs preference in order to reinforce effective learning.

Research Questions

In this study, six primary research questions have been identified and formulated to address the research outcomes that:

(1) Are there any significant differences in the quality of codes written between students who received PP treatment and those who received DI (direct instruction) only?

(2) Are there any significant differences in duration for completing the programming assignments between students who received PP treatment and those who received DI only?

(3) Are there any significant differences in the number of errors made between students who received PP (PP) treatment and those who received DI only?

(4) Are there any significant differences in recall and retention between students who taught in PP and DI instructional methods?

(5) Are there any significant differences in recall and retention between visual and verbal students in learning programming?

(6) Are there any interaction effects between instructional methods and students' LSs preference for recall and retention performance?

Research Methodology

The purpose of this study was to investigate the effects of PP strategy in learning the basic programming concepts on the students' programming performances in terms of assignment comprehension, recall and retention amongst computing students. This study was further extended to include LS factor that served as the moderating variable for examining the differences in recall and retention between both visual and verbal students in learning programming. The interaction effects between instructional method and LS preference of students in programming performance were also analyzed. The emphasis of this research was on whether cooperative learning through PP in classroom environment is an effective alternative solution in programming education as compared to the DI approach that involves individual learning.

Research Design

A 2 × 2 quasi-experimental design was used to examine the effects of PP on the five dependent variables with LS (visual and verbal) being the moderating variables. The three dependent variables—quality of codes, time taken and errors made were used to assess the students' performances in programming assignment. The recall and retention being the last two dependent variables were based on the immediate and delayed post-test scores. A total of 96 first year undergraduate computing students participated in this study. They were randomly assigned to the two treatment groups. The experimental group consisting 48 students received the PP treatment and the control group (*n* = 48) received the DI treatment.

Research Instruments

In assessing the continuous learning progress of acquiring the programming concepts and skills, two programming assignment questions were used to measure the students' achievements during the learning activities. This formative assessment is to determine their learning efficiency throughout the course duration.

The Felder-Soloman's ILSQ (index of learning style questionnaire). This instrument based on the Felder and Silverman's (1988) LSs model was used to access the students' LS preference prior to the treatment. The existing ILSQ instrument consists of four learning dimensions. In this study, only one dimension measuring the visual and verbal LS is applied (Cronbach's Coefficient alpha of 0.78) (Zywno, 2003). This instrument uses a multiple choice format in presenting options. The students who responded mostly "a" in the questionnaire were classified as visual learners and those who respond mostly "b" were identified as verbal learners.

The CPPT (computer programming performance test). A pre-test was used to assess any initial differences in terms of programming performance between the two treatment groups prior to the treatment. The immediate and delayed post-tests of CPPT as summative assessments were administered to measure the students' recall and retention. The immediate post-test was administered immediately after the treatment and delayed post-test was conducted one month after the treatment. Both tests were designed based on the McGill and Volet's (1997) conceptual framework to assess the theory and practical knowledge of the basic programming concepts, especially in the topic of sequence and selection.

Data Collection Procedures

The experiment was conducted for a seven-week duration to the two intact classes. These classes from the first year computing programme were randomly selected and assigned to one of the two treatment groups: the PP group or the DI instructional strategy group in learning the basic programming concepts. The ILSQ questionnaire was administrated to these two groups prior to the treatment in order to classify the students as visual or verbal learners. The pre-test was first administered to these two groups of students before the treatment in order to collect the baseline data on their programming knowledge. During the treatment, the students in both groups received program flowchart and pseudocode in learning the basic programming concepts: (1) sequence; and (2) selection constructs.

In the PP group, the roles of "driver" and "navigator" were explained to both members of the pairs. The students worked in pairs (one visual and one verbal learner) to discuss and solve novel problems by applying the program flowchart and/or pseudocode in the given tasks. They were to persistently cooperate on the same design, algorithm, coding and testing. From time to time, they were expected to switch roles between the "driver" and the "navigator". On the other hand, there were no cooperative learning activities involving the students taught in the DI group. Students of this group were to work individually on the similar programming tasks as to the PP group. During this seven-week of treatment, both groups were to complete two programming assignments in regard to either working as pairs or individual.

Immediately after the treatment, the immediate post-test was administered to the two groups. The delayed post-test was carried out a month later. The students' programming performance on recall and retention in the basic programming concepts were measured using the CPPT instrument.

Findings

In this study, the pre-test score was used as the covariate to ensure the homogeneity between two participating groups in terms of knowledge status prior to treatment. The MANCOVA (multiple analysis of covariance) was selected to test the research hypotheses as it involved five dependent variables: (1) quality of codes; (2) time taken; (3) errors made; (4) recall; and (5) retention performance.

The results of the analysis are shown in Tables 1-3. Tables 1 and 2 indicate the MANCOVA findings, while Table 3 shows the descriptive statistics. Meanwhile, the graphs revealing the interaction effects between instructional methods and LS preference are illustrated in Figures 3 and 4. The results of MANCOVA analysis in Table 2 clearly reveal no significant difference in recall and retention performance between two treatment groups and visual-verbal LS. This finding shown here involves only between the visual and verbal students taught in the treatment groups. Therefore, it is used to reveal on the interaction effects between instructional methods and students' LS preference. However, the post hoc test in comparing between the visual and verbal students in both the PP and DI groups, between the visual students in the PP and DI groups, and between the verbal students in both the treatment groups were not the scope of this discussion.

Table 1

MANCOVA Analysis for Dependent Variables in the Two Treatment Groups

Source	Dependent variables	Degree of freedom (*df*)	Mean square	Fixation indices (*F*)	Significant
Group	Quality of Codes Written	1	1,390.45	26.65	0.00*
	Time Taken in Completion	1	235.03	43.54	0.00*
	Errors Made in Codes	1	127.70	50.16	0.00*
	Recall (immediate post-test)	1	5,554.46	48.64	0.00*
	Retention (delayed post-test)	1	6,102.45	59.02	0.00*
ILSQ	Recall (immediate post-test)	1	463.75	4.06	0.05*
	Retention (delayed post-test)	1	261.45	2.53	0.12

Note. * at the significant level of 0.05.

Table 2

MANCOVA Results for the Recall and Retention Scores of the Two Treatment Groups and LS

Source	Dependent variables	Degree of freedom (*df*)	Mean square	Fixation value (F-value)	Probability value (*P*-value)	Result*
Group*	Recall (immediate post-test)	1	7.69	0.07	0.80	Not significant
ILSQ	Retention (delayed post-test)	1	62.56	0.61	0.44	Not significant

Note. *at the significant level of 0.05.

Table 3

Descriptive Statistics of the Dependent Variables for the Two Treatment Groups With Different LSs

Source	Dependent variables	Group	Mean	Standard deviation (*SD*)	*N*
Group	Quality of codes written	PP	79.81	8.71	48
		DI	72.08	6.58	48
		Total	75.88	8.75	96
	Time taken in completion	PP	9.81	1.82	48
		DI	12.99	2.78	48
		Total	11.41	2.84	96
	Errors made in codes	PP	3.69	1.65	48
		DI	6.03	1.73	48
		Total	4.86	2.10	96
	Recall (immediate post-test)	PP	77.59	12.39	48
		DI	62.12	9.63	48
		Total	69.81	13.79	96
	Retention (delayed post-test)	PP	67.42	10.19	48
		DI	51.21	10.78	48
		Total	59.26	13.52	96
ILSQ	Recall (immediate post-test)	Visual	67.62	14.43	49
		Verbal	72.08	12.60	47
		Total	69.81	13.79	96
	Retention (delayed post-test)	Visual	57.64	14.58	49
		Verbal	60.99	12.06	47
		Total	59.26	13.52	96

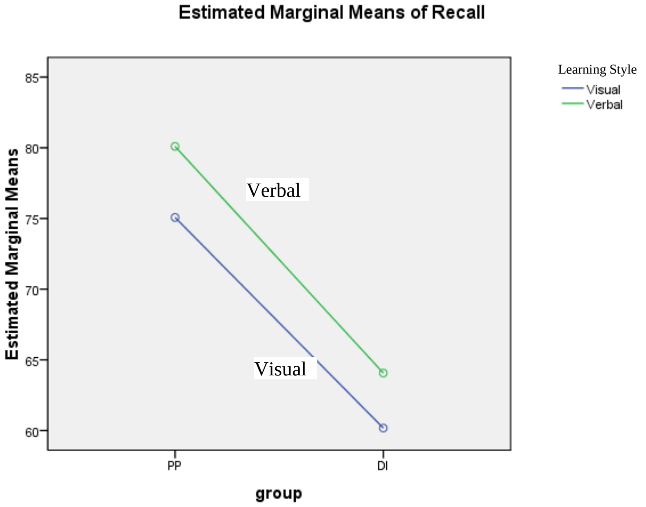

Figure 3. Interaction effects between instructional method and students' LSs on recall performance.

Hypothesis 1: There are no significant differences in the quality of codes written between the PP and the DI groups.

The MANCOVA results in Table 1 indicated a statistical significant difference in the quality of codes written between students who received PP, PP and those who received direct instruction, DI (F = 26.65; p = 0.00). Therefore, this finding has rejected the first hypothesis. As shown in Table 3, the students who received PP treatment performed significantly better than those of the DI treatment ($\overline{X}_{\text{pp}}$ = 79.81; $\overline{X}_{\text{DI}}$ = 72.08).

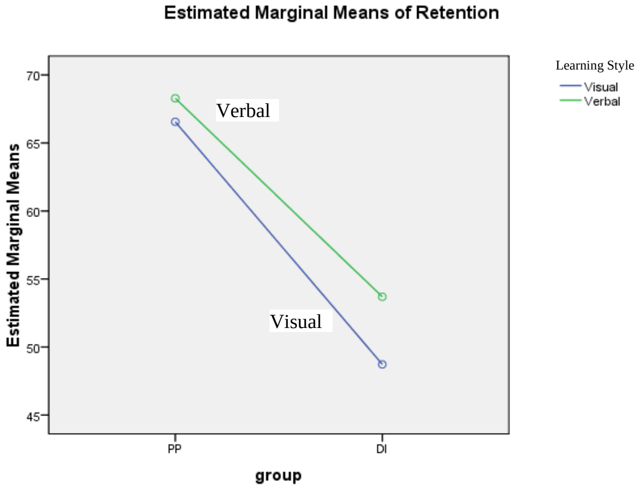

Figure 4. Interaction effects between instructional method and students' LSs on retention performance.

Hypothesis 2: There are no significant differences in duration taken between the PP and the DI groups.

The result analysis found a significant difference in the duration taken to complete the programming assignments between the students of PP and DI ($F = 43.54$; $p = 0.00$). The students who received PP treatment significantly outperformed those who received DI treatment ($\overline{X}_{\text{pp}} = 9.81$; $\overline{X}_{\text{DI}} = 12.99$). Thus, the second hypothesis has been rejected.

Hypothesis 3: There are no significant differences in the number of errors made between the PP and the DI groups.

The statistical analysis (see Table 1) also found a significant difference in the number of errors made in the programming assignments between the students taught in the PP and the DI methods ($F = 50.16$; $p = 0.00$; $\overline{X}_{\text{pp}} = 3.69$; $\overline{X}_{\text{DI}} = 6.03$). The descriptive statistics in Table 3 reveals that students who received PP treatment performed significantly better as compared to those with DI treatment. Therefore, this finding has rejected the third hypothesis.

Hypothesis 4: There are no significant differences in recall and retention between the PP and the DI groups.

The finding indicated that there is a significant difference in recall between the students who received PP and those who received DI ($F = 48.64$; $p = 0.00$). Similarly, a significant difference in retention performance between both groups of students was shown ($F = 59.02$; $p = 0.00$). The fourth hypothesis has been rejected as the students taught in the PP method significantly outperformed those taught in the DI method ($\overline{X}_{\text{pp-recall}} = 77.59$; $\overline{X}_{\text{DI-recall}} = 62.12$; $\overline{X}_{\text{pp-retention}} = 67.42$; $\overline{X}_{\text{DI-retention}} = 51.21$) for both the recall and retention performance.

Hypothesis 5: There are no significant differences in recall and retention between visual and verbal students in learning programming.

The statistical analysis also found a significant difference in recall performance between visual and verbal students ($F = 4.06$; $p = 0.05$). The descriptive statistics in Table 3 further indicated that the verbal students performed significantly better than those visual students ($\overline{X}_{\text{visual-recall}} = 67.62$; $\overline{X}_{\text{verbal-recall}} = 72.08$). Therefore, this finding has rejected the fifth hypothesis on recall. As for the retention performance, the descriptive statistics (see Table 3) indicated that verbal students obtained somewhat higher mean scores than those visual students ($\overline{X}_{\text{visual-retention}} = 57.64$; $\overline{X}_{\text{verbal-retention}} = 60.99$). However, the MANCOVA results revealed a non-significant difference in performance between both visual and verbal students ($F = 2.53$; $p = 0.12$; $p > 0.05$). As such hypothesis five on retention has been accepted.

Hypothesis 6: There are no interaction effects between instructional methods and students' LS preference for recall and retention performance.

The MANCOVA results (see Table 2) of the interaction effects on the two dependent variables were not statistically significant. The interaction graph in Figures 3 and 4 describes the relationship between the instructional methods (PP and DI) and the students LS preference (visual and verbal) across the two groups on recall and retention performance. Both graphs show no interaction effects between the instructional methods and the students' preferred LS on recall and retention across the two treatment groups ($F_{\text{recall}} = 0.07$; $p = 0.80$; $F_{\text{retention}} = 0.61$; $p = 0.44$). Thus, the finding has accepted the sixth hypothesis as no significant effects were observed.

Discussions

This study aimed to investigate the impact of PP instructional method and LS on students' programming assignments, recall and retention performances during fundamental programming course delivery. Two topics namely "sequence and selection" as a part of the basic programming concepts syllabus are vitally important for first year computer science students who are pursuing programming courses. The interaction effects between the instructional methods and LSs were also analyzed. The research findings indicated that the students taught in the PP method significantly outperformed their peers in the DI group on the course assignments, recall and retention performance. The PP has significantly assisted the students in programming comprehension and problem-solving skills. Likewise, the statistical results showed only a significant difference in recall performance between visual and verbal students in learning programming.

In the programming assignments performance, the findings revealed that the use of PP as cooperative learning approach did significantly improve the quality of codes developed, reduce the duration taken in completing the programming tasks assigned and minimize the number of errors made in codes. In particular, the students in the PP group obtained significantly better scores in programming assignments as compared to their counterparts in the DI group. This cooperative learning structure encourages students to acclimatize themselves to peers' approaches, perspectives and strategies in solving programming scenarios. Pair activities allow students to brainstorm on the novel problems that stimulate higher cognitive thinking and in turn cultivates "self troubleshooting" abilities (Xu & Rajlich, 2005; VanDeGrift, 2004). Through peer interaction, the discussions help each member to check for errors and complete the given tasks together in shorter duration with fewer mistakes found in the programming assignments than those working individually in the DI group (Bryant, Romero, & du Boulay, 2007; Kuppuswami & Vivekanandan, 2004). Discussions and explanations in peer learning environment assist the students to reflect upon and elucidate own actions and decisions for deriving the solution, thus, make thinking explicit. Furthermore, these findings on assignment performance have supported previous empirical studies in that the adoption of PP has yielded better quality, fewer defects of codes and shorter development cycles (Bryant, Romero, & du Boulay, 2008; Ballantine & Larres, 2007).

For the recall performance, this finding reported that PP has significantly influenced the students' immediate recall. In other words, the students in the PP group have achieved higher scores in recall performance than their peers in the DI group. Working in pairs enhances the students' comprehensions of the programming concepts expression, as these students in the PP group are encouraged to talk. This manner of learning, allowing them to discuss and self-explain that somehow facilitate problem-solving processes, aid in making arguments and accepting constructive criticisms, and, thus, develop higher thinking skills. This type of verbalization has resulted in achieving greater level of understanding and developing clearer “mental model” of the abstract concepts which are essentially critical to problem-solving (Goel & Kathuria, 2010; Bryant, du Boulay, & Romero, 2006).

This finding also revealed a significant difference in the students’ retention performances. PP creates richer social interaction which allows the students in the PP group to clarify uncertainty and identify alternative and generate solutions with their pair partners. In this pairing approach, students were consistently guided to be active learners where they were sanctioned to think accountable for self and peers’ learning and to foster better relationship with others. Hence, these students taught in the PP group shown significant improvement in their learning efficiency on the retention performance and eventually cultivating the interest towards programming as compared to those of the DI group when they are persistently cooperate on the given tasks. This leads to higher order thinking skills which subsequently show positive constructive influence on memory retention. Through peer interaction, students in the PP group cross check individual programming tasks, provide positive constructive criticisms and assist one another on the task assigned, which then significantly influence their long term learning process as compared to those who work individually. As such, the students are presumed to exchange role (driver and navigator) in working on the design, algorithm, coding and testing of the novel scenarios. In line with the results of several previous studies, this finding demonstrated that PP facilitates and improves learning through pair efforts (Gibbons, 2008; Newton & de Villiers, 2007).

In LS preference, the findings revealed a significant difference between visual and verbal students in learning programming for recall performance. In classroom environment, students were taught in auditory (lecturing) form where visual presentation of auditory information is organized into simple understandable chunks. This enables verbal students to forms better images of new concepts that thus assist them to easily digest the conceptual theories during the learning process as compared to visual students. When auditory information presented in pictorial, diagrammatical and charter forms, visual students may have forgotten it as their preferred style is in graphical manners. On the other hand, verbal students remember better when they hear and then say. As such, PP which emphasizes on discussions helps these verbal students effectively by explaining concepts, theories and approaches used in producing the completed programming tasks to their partners (Simon & Hanks, 2007; Felder, 2002). However, there was no significant difference in retention performance between the visual and verbal students during learning process for later information retrieval. Disregard to LS preference, these students performed equally well in their retention tests. This finding supports Fenrich’s (2006) and Davis’s (2006) studied that even though LSs serve as a relatively stable indicator, it is predictable to change from one learning environment to another. In this case, learning still occurs when visual students have adopted the new learning approach and continue to process in perceiving, interacting with and responding to the programming problems.

Based on the interaction effects between the two instructional methods and the visual-verbal LSs dimension, this study revealed no interaction effect in recall and retention between instructional methods adopted and the students’ LS preference. In other words, visual and verbal students taught in either PP or DI method benefited equally in programming performance. During the learning process, the effects of the instructional methods on programming recall and retention did not depend on the different LSs preference of individual student. This finding supports previous empirical studies that with regard to the preferred style, students somehow change their learning processes from one learning dimension to another when responding to acquiring knowledge (Tie & Irfan Naufal Umar, 2010; Price, 2004; Ayersman, 1993). Perhaps this is the reason why their programming achievements were not affected as these students in both groups were able to comprehend the novel concepts and enhance their problem-solving skills by adapting themselves in different presentation environment. However, the gap in the "Estimated Marginal Means" for the retention performance between visual and verbal students taught in PP group is small. This indicated that the effectiveness of one method particularly depends on the intact group of learners.

Conclusions

The study has emphasized the importance of adopting PP in learning the basic programming concepts in order to solve novel problems. PP has played a significant role in assisting students to develop self-reasoning and solutions on abstract situations, especially for problem-solving. This pairing activity with social interaction element has been used in software industry to enhance programming skills and increase productivities. Thus, in education, PP significantly yields a very positive impact on students' programming efficiencies and subsequently to achieve greater recall and retention performance. Besides, LS is a predictable stable indicator that could also influence the learning approach of students in perceiving, interacting with and responding to programming concepts. Considering LS components in learning the basic programming construct for classroom delivery may cause significant impact on students' performance in terms of recall and retention. To reinforce learning, blending both the visual and verbal representation of novel concepts into course deliveries may enhance cognitive knowledge and promote logical thinking skills with respect to programming performance. With deeper understanding on the LS dimensions, it benefits students on the learning outcomes through working with either a heterogeneous or homogeneous pair. For novices, interactive learning through PP creates positive impact on programming comprehension as well as generates fun learning environment to reduce anxiety during coding. It is also suggested that pairing activities to be incorporated into programming classes as this interaction element is vitally crucial in the working environment that the current students are lacking.

References

Adams, A., Lubega, J., Walmsley, S., & Williams, S. (2004). The effectiveness of assessment learning objects produced using pair programming. *Electronic Journal of E-learning, 2*(2), 247-256. Retrieved June 8, 2010, from http://www.ejel.org/volume-2/vol2-issue2/v2-i2-art1-adams.pdf

Ayersman, D. (1993). An overview of the research on LSs and hypermedia environments. Paper presented at *the 1993 Annual Convention of the Eastern Educational Research Association*. Clearwater Beach, Florida.

Ballantine, J., & Larres, P. M. (2007). Cooperative learning: A pedagogy to improve students' generic skills? *Education and Training, 49*(2), 126-137.

Barbe, W. B., & Milone, M. N. (1981). What we know about modality strengths. *Educational Leadership,* February, 378-380. Retrieved November 12, 2011, from http://www.ascd.org/ASCD/pdf/journals/ed_lead/ el_198102_barbe.pdf

Beck, K. (2005). *Extreme programming explained: Embrace change* (2nd ed.). Boston: Addison Wesley.

Begel, A., & Nagappan, N. (2010). Pair programming: What's in it for me? In A. Begel, & N. Nagappan, *Software Engineering Seminar,* March, 7. Retrieved June 8, 2010, from http://se.inf.ethz.ch/teaching/2010-s/0276/slides/senn.pdf

Bryant, S., du Boulay, B., & Romero, P. (2006). XP and pair programming practices. *Psychology of Programming Interest Group (PPIG) Newsletter, September,* 1-7.

Bryant, S., Romero, P., & du Boulay, B. (2007). *Pair programming and the mysterious role of the navigator*. IDEAS laboratory, University of Sussex, UK. Retrieved June 8, 2010, from http://www.cogs.susx.ac.uk/users/bend/paper/ijhcs07.pdf

Bryant, S., Romero, P., & du Boulay, B. (2008). Pair programming and the mysterious role of the navigator. *International Journal of Human-Computer Studies, 66*(7), 519-529.

Chong, J., Plummer, R., Leifer, L., Klemmer, S. R., Eris, O., & Toyer, G. (2005). Pair programming: When and why it works. In P. Romero, J. Good, E. Acosta Chaparro, & S. Bryant (Eds.), Proceedings of *the 17th Psychology of Programming Interest Group* (pp. 43-48).

Davis, G. A. (2006). LS and personality type preferences of community development extension education. *Journal of Agricultural Education, 47*(1), 90-99.

Deborah, G. (2005). Meeting differing LSs of non-traditional students in the second language classroom. *Journal of College Teaching and Learning, 2*(8), 1-7.

Felder, R. M. (1993). Reaching for the second tier: Learning and teaching styles in college science education. *Journal of College Science Teaching, 23*(3), 286-290.

Felder, R. M. (2002). Learning and teaching styles in engineering education. In R. M. Felder (Ed.), *Engineering Education, 78*(7), 674-681.

Felder, R. M., & Silverman, L. K. (1988). LSs and teaching styles in engineering education. *Engineering Education, 78*(7), 674-681.

Fenrich, P. (2006). Getting practical with LSs in "live" and computer-based training settings. *Issues in Informing Science and Information Technology, 3,* 233-242.

Fincher, S. (2006). *Studying programming*. The computing education research group, England: Palgrave Macmillan.

Gibbons, M. T. (2008). *The year in numbers* (In the 2007 Profiles of Engineering and Engineering Technology Colleges). Washington, D.C.: The American Society for Engineering Education.

Gillies, R. M. (2007). *Cooperative learning: Integrating theory and practice.* Los Angeles: Sage Publication.

Goel, S., & Kathuria, V. (2010). A novel approach for collaborative pair programming. *Journal of Information Technology Education, 9,* 183-196.

Hahn, J. H., Mentz, E., & Meyer, L. (2009). Assessment strategies for pair programming. *Journal of Information Technology Education, 8,* 273-284.

Hamer, J. (2006). Some experiences with the "contributing student approach". Proceedings of *the 11th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education* (pp. 68-72). Bologna, Italy. Retrieved April 29, 2010, from http://portal.acm.org/citation.cfm?id=1140123.1140145

James, W., & Gardner, D. (1995). Learning styles: Implications for distance learning. *New Directions or Adult and Continuing Education, 67,* 19-31.

Kolb, D. A. (1984). *Experiential learning: Experience as the source of learning and development*. Englewood Cliffs, N. J.: Prentice-Hall.

Kuittinen, M., & Sajaniemi, J. (2004). Teaching roles of variables in elementary programming courses. In *the 9th Annual Conference on Innovation and Technology in Computer Science Education* (pp. 57-61). Association for Computing Machinery.

Kuppuswami, S., & Vivekanandan, K. (2004). The effects of pair programming on learning efficiency in short programming assignments. *Informatics in Education, 3*(2), 251-266.

Ladd, P. D., & Ruby, R. (1999). LS and adjustment issues of international students. *Journal of Education for Business, 74*(6), 363-367.

McDowell, C., Hanks, B., & Werner, L. (2003). Experimenting with pair programming in the classroom. Proceedings of *the 8th Annual Conference on Innovation and Technology in Computer Science Education* (pp. 60-64). *ACM SIGCSE Bulletin, 35*(3).

McGill, T. J., & Volet, S. E. (1997). A conceptual framework for analyzing students' knowledge of programming. *Journal of Research on Computing in Education, 29*(3), 276-297.

Meseka, C. A., Nafziger, R., & Meseka, J. K. (2010). Student attitudes, satisfaction, and learning in a collaborative testing environment. *The Journal of Chiropractic Education, 24*(1), 19-29.

Miliszewska, I., & Tan, G. (2007). Befriending computer programming: A proposed approach to teaching introductory programming. *Issues in Informing Science and Information Technology, 4,* 277-289.

Newton, A., & de Villiers, J. (2007). Thinking while talking: Adults fail false-belief reasoning. *Psychological Science, 18*(7), 574-579.

Pallapu, P. (2007). Effects of visual and verbal LSs on learning. *Institute for LSs Journal, 1,* 34-39.

Parsons, D. (2004). Justice in the classroom: Peer assessment of contributions in group projects. Proceedings of *the 17th NACCQ* (pp. 145-152). Christchurch. Retrieved May 20, 2010, from http://www.naccq.ac.nz/conference05/proceedings_04/parsons.pdf

Price, L. (2004). Individual differences in learning: Cognitive control, cognitive style, and LS. *Educational Psychology, 24*(5), 681-698.

Richardson, J. (1984). *Working with people*. Associate Management Institute, San Francisco, California.

Simon, B., & Hanks, B. (2007). First year students' impressions of pair programming in CS1. Proceedings of *ICER*, September, Atlanta, G. A.

Srikanth, H., Williams, J. G., Wiebe, E., Miller, C., & Balik, S (2004). On pair rotation in the computer science course. Proceedings of *the 17th Conference on Software Engineering Education and Training* (pp. 144-149), March 1-3.

Tie, H. H., & Irfan Naufal Umar. (2010). The impact of LSs and alternative instructional approach on students' programming performance: A pilot study. Proceedings of the *2nd TARC International Conference on Learning and Teaching*, October (pp.18-19, 140-147).

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

VanDeGrift, T. (2004). Coupling pair programming and writing: Learning about students' perceptions and processes. Proceedings of the *6th Annual Conference on Innovation and Technology in Computer Science Education* (pp. 1-6). *ACM SIGCSE Bulletin, 33*(3).

Verhaart, M., Hagen, K., & Giles, O. (2005). Best practice assessment methods for evaluating an individual's performance in group work. Proceedings of the *18th Annual Conference of the National Advisory Committee on Computing Qualifications*. New Zealand, 113-120. Retrieved May 20, 2010, from http://www.naccq ac.nz/conference05/proceedings_05/paper/verhaart.pdf

Williams, L., & Kessler, R. (2003). *Pair programming illuminated*. Boston: Addison-Wesley.

Xu, S., & Rajlich, V. (2005). Pair programming in graduate software engineering course projects. *The 35th ASEE/IEEE Frontiers in Education Conference*, October 19-22, Indianapolis, India, F1G-7-F1G-12.

Zywno, M. S. (2003). A contribution to validation of score meaning for Felder-Soloman's index of learning styles. Proceeding of *the 2003 American Society for Engineering Education Annual Conference & Exposition*, American Society for Engineering Education.