# Multi-touch Table Technology for Enhancing Collaborative Programming and Learning

Ahmed Alzahrani
Leeds University
scaaal@leeds.ac.uk

Eric Atwell
Leeds University
e.s.atwell@leeds.ac.uk

Andy Bulpitt
Leeds University
a.j.bulpitt@leeds.ac.uk

**Abstract:**

Our research goal is to provide a new use for multi-touch table technology by developing new resource software to be used as an environment for collaborative pair programming. Such a project will yield new opportunities and provide improved quality within the field of collaborative learning (Nosek 1998; Williams et al., 2000a, 2000b; Williams et al., 2001; Cockburn et al., 2000; Mcdowell et al. 2002; Gallis et al. 2003; Cao and Xu, 2005; Brereton et al., 2009; Sanjay and Goel and Kathuria, 2010; Rick et al., 2011). For instance, the proposed project (1) is time-effective and leads to a better product, (2) is more enjoyable for developers, (3) may increase work satisfaction, (4) promotes student learning, (5) leads to better student activity engagement, (6) contributes to enhancing the level of student understanding with regard to programming modules, which makes students more confident in their work and more positive about programming and, finally, and (7) supports the awareness of others' actions and the ability to support concurrent input; it also encourages the development of problem-solving skills, efficiency, quality and teamwork.

The project begins with a survey of the available tools for collaborative pair programming, such as Scratch Programming (Scratch, 2014), and then these tools will be evaluated to determine their suitability for use in multi-touch tables. The most suitable tool will then be selected for development to be run on a multi-touch table. Alternatively, new resource software will be created for the project. In the final stage, the usability of new software, in terms of collaborative pair programming, will be evaluated.

**References**

Brereton, P., Turner, M. and Kaur, R. (2009). Pair programming as a teaching tool: a student review of empirical studies. pp.240--247.

Cao, L. and Xu, P. (2005). Activity patterns of pair programming. pp.88--88.

Cockburn, A. and Williams, L. (2000). The costs and benefits of pair programming. Extreme programming examined, pp.223--247.

Gallis, H., Arisholm, E. and Dyba, T. (2003). An initial framework for research on pair programming. pp.132--142.

Goel, S. and Kathuria, V. (2010). A novel approach for collaborative pair programming. Journal of Information Technology Education: Research, 9(1), pp.183--196.

McDowell, C., Werner, L., Bullock, H. and Fernald, J. (2002). The effects of pair-programming on performance in an introductory programming course. 34(1), pp.38--42.

Nosek, J. (1998). The case for collaborative programming. Communications of the ACM, 41(3), pp.105--108.

Rick, J., Marshall, P. and Yuill, N. (2011). Beyond one-size-fits-all: How interactive tabletops support collaborative learning. pp.109--117.

Scratch.mit.edu, (2014). Scratch - Imagine, Program, Share. [online] Available at: http://scratch.mit.edu/ [Accessed 9 Oct. 2014].

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

Williams, L., Kessler, R., Cunningham, W. and Jeffries, R. (2000). Strengthening the case for pair programming. IEEE software, 17(4), pp.19--25.

Williams, L. (2001). Integrating pair programming into a software development process. pp.27--36.