Explaining Multi-threaded Task Scheduling using Tangible User Interfaces in Higher Educational Contexts

Clifford De Raffaele, *IEEE Member*
School of Science and Technology
Middlesex University Malta
Pembroke, PBK1776, Malta
cderaffaele@ieee.org

Serengul Smith, Orhan Gemikonakli
School of Science and Technology
Middlesex University
Hendon, NW4 4BT, UK
s.smith@mdx.ac.uk, o.gemikonakli@mdx.ac.uk

***Abstract*—Endorsing the advantages of computer-based interaction within the educational domain, this study analysis the potential for tangible interactive technology to mitigate the challenges faced by higher educational institutes in explaining abstracted technical concepts. Implemented within a novel within the educational domain, this paper evaluates the efficacy of adopting a tangible user interface (TUI) to aid in the conceptual understanding of multi-threaded task scheduling and programming by undergraduate IT students. Making use of physical object representations, a description is provided for the distinctive development of a collaborative system that allows students to interact with and visualize the scheduling of multiple software threads onto a computer processes. The paper quantitatively studies the usefulness of the proposed TUI system with respect to traditional lectures by deploying the system within a university computing degree. Evaluation analysis of the obtained results highlight a significant improvement in the students' abilities to grasp the abstract and complex notions of multi-threading, thus validating the potential of the proposed study.**

Keywords— Computer aided instruction; Higher Education; Multi-threaded Task Scheduling; Tangible User Interface

I. INTRODUCTION

The drive to include technology within the educational environment has garnered consistent momentum in the last years [1], with the premise of active learning leading the strategy in computer-based education [2]. Referred to as digital natives, today's children have particularly succeeded in embracing these techniques [3-4], and the coupling of education with digital games is becoming ever-more widespread due to its ability to further engage children with learning [5]. Several studies have supported this approach, with results highlighting an increase in knowledge and cognitive performance from students [6-8].

In tandem with this progress, market commoditisation has been shifting technical advancements such as multiple CPU integration from a previously specialised domain associated with supercomputers to proliferation in laptops and mobile systems [9]. As parallel computing gained increased significance in the industry, this has in turn directly affected the educational curricula [10], with students in computer science being compelled to familiarise themselves with the different approaches to managing parallelism in order to have the ability to eventually exploit future computing technologies [9].

With multi-thread programming becoming the method of choice in parallel computing [11], emphasis within the industry is ever-more focused on the "real performance" of systems and hence impelling programmers to consider the overall execution time of their software [12]. Thus, whilst teaching modern programming languages today, lectures must go beyond object oriented concepts and promptly introduce students to built-in thread functionality within languages to deal with concurrency and inter-thread synchronisation issues [10].

As explained by the authors in [13], introducing the paradigm of multi-threading poses significant challenges for both the lecturer to find the best way to the concepts as well as for students to understand what is happening to their programmes [13]. This difficulty was practically experienced in the study by [14] which reiterated previous claims, highlighting the need for changing the students' thinking paradigm from that adopted in sequential programming [15].

Further adding complexity to multi-threaded programming is the fact that debugging and analysis techniques which are commonly adopted in single-threaded applications do not provide the same relevant information to reconstruct the parallel execution of programs [10]. To overcome this limitation and mitigate the loss of students' confidence in understanding what is happening at runtime to their code [13] a number of software tools have been developed that log the concurrent execution of instructions on different threads and visually display these using UML and other software development tools [16-18]. Whilst these approaches provide a significant aid for students to understand multi-thread runtime execution [10], [13], they intrinsically require successive iterations of code programming and execution to generate useful results. The latter can in fact only be derived in post-execution of different multi-threaded configurations, upon which students can finally comparatively evaluate each simulation scenario [19].

In light of the above constraints and the limited adoption of computer-based technology within literature for explaining the concepts of concurrency and inter-thread synchronisation, this

paper proposes the adoption of a Tangible User Interface (TUI) system to provide students the ability to interactively comprehend the abstract nature of multi-threaded execution. This paper is organised so that following a brief analysis and review of TUI related articles in education, presented Section II, the proposed design for a novel multi-threaded scheduling TUI system is elaborated in Section III. Subsequently, Section IV explains the details and results derived from implementing this research within a university programme. Finally, Section V outlines a brief conclusion of the presented work.

II. Tangible User Interfaces

TUIs have been developed as innovative human-computer interaction technique that natively interlaces the physical and digital domains [20]. As opposed to conventional Graphical User Interfaces (GUI)s, TUI systems go beyond the limited use of computer peripherals (such as keyboards and mice) and permit users to interact with digital information through manipulation of physical objects and triggered behaviours [21],[22]. This enables users of TUI systems to take advantage of innate spatial and environmental skills [23] whilst interacting with and configuring physical objects [24], attributes which have been directly correlated with the enhancement of problem solving skills [25]. Making use of common physical items representing a users' everyday environment to interact with digital information [25] further affords TUI systems an assimilation paradigm that helps users focus on the subject at hand without being distracted by "control mechanisms" for interaction and feedback [26]. Studies report that with respect to conventional systems such as GUI and multitouch, such techniques provide TUI users a heighted sense of constructive behaviour, attractive engagement, and sociocultural learning by enabling collaborative use [25][27].

The integration of these learning benefits together with the inherent attractive and eye-catching aspects of TUI systems [28] led to positive feedback when integrated within children's learning environments. By allowing children to touch and manipulate tangible musical notes, the work in [29] reported significant progress through the "boring" phases of learning music. Moving tokens were used in [30] to familiarise students with astrological notions of moon phases and the motion of planets. Another system aimed to teach children rhetorical skills and hence facilitate their expression of complex arguments using TUI [31]. The substitution of numbers with tangible patterned blocks also allowed the successful mathematical teaching of pattern fitting and area comparison [32].

As a contrast, little work has been done in using TUI to explain programming related concepts. In [24], the authors employ a gamified approach to introduce basic concepts for object-oriented programming (OOP). The study employs the use and programming of Sifteo cubes [33], which are autonomous microcontroller-based devices, by students during their course. Whilst results illustrate that this technique augmented the interest levels and consequently the achieved marks by students [24], the technique does not focus on the tangible aspect of teaching and uses Sifteo cubes mainly as a code execution platform rather than the Integrated Development Environment (IDE) on a conventional computer. Another study aimed to introduce aspects of programming recursion in the Erlang functional language by linking a TUI system in conjunction with Augmented Reality (AR) [34]. Whilst succeeding to visualise programming stacks in single-tailed-recursive structural functions, the technological interactions of the system are not widely applicable [34] and the system makes use of tangible blocks merely as optical markers for controlling pure AR virtualisation.

More complex adaptions of TUI systems have alternatively been employed within the industrial setting to model physical situations and visualise abstract associations [28]. Within the air traffic control industry, the Strip'TIC TUI system allowed employees to visualise and understand events within the controlled airspace [35]. Employed for architectural building design, a TUI system provided architects the ability to analyse the effect of shading and illumination generated by different daylight stages [1]. Geospatial modelling using TUI was further successful in [36] which allowed users a flexible way to sculpt terrain as well as investigate stormwater runoff management options. The interactive control abilities of TUI were further exploited in [37] where physical objects, representative of design models, were employed to more intuitively and easily edit CAD and GIS designs [37] designations.

III. Proposed TUI Framework

The unique contribution of this paper lies at the confluence of the different streams investigated in literature on TUI implementations and multi-threading education. This research makes its contribution by analysing the suitability of TUI systems for integration with the domain of higher education. In particular, the proposed system will be evaluated for its efficacy and aptness in aiding the teaching and learning of abstract and complex concepts such as those present within the task scheduling of a multi-threaded environment.

A. System Overview

The system was designed to challenge students in allocating a number of computationally complex operations for execution on multiple threads using a TUI approach. Based on the MCRpd interaction model [38] illustrated in Fig 1(a), the proposed implementation adopts an interactive table top design in line with the ReacTIVision framework [39] depicted in Fig 1(b).

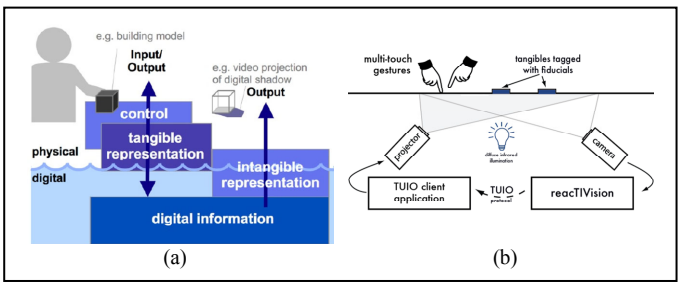

Fig. 1. a) Tangible interaction model. *(source: [38])*
b) ReacTIVision architectural framework *(source: [39])*

As shown in Fig 2, the physical construction was composed of a wooden table onto which a semi-transparent acrylic glass was placed. The height of the table, at 80cm, was designed so that the table top setup could be easily accessible and visible to a number of students standing around it. These dimensions provided an interactive area of 1m x 0.7m which was illuminated using a short-throw projection and captured via a wide-angle camera.

Fig. 2. Interactive table top hardware setup used in proposed system.

B. *Interactions*

Students engage interactively with the proposed system via a set of 3D physical objects which allow the control and setup of scenarios. These would be directly used to allow setup and control of the created multi-threaded scenarios. These were designed so that they inherently symbolise and express the different computationally complex procedures which would undergo multi-threaded execution within the system. Four commonly used processes in the computer science field were identified by these criteria so as to ensure that students are familiar with the scenarios. These were; downloading content from the internet, compressing files, image processing and content searching).

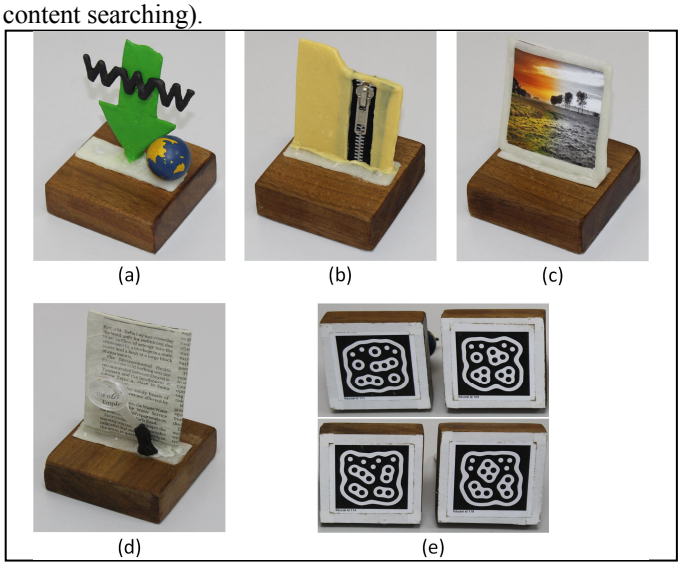

Fig. 3. Design of tangible objects representing distinct processes;
a) downloading internet content, b) file compression,
c) image processing, d) content searching,
e) reacTIVision 'amoeba' fiducials [39].

As seen in Fig 3, the tangible objects were designed to characteristically represent the aforementioned processes so that interaction would be evermore instinctive to students. The following descriptions highlight individual feature representation visualised in Fig 3:

- Internet Downloads (Fig 3a) – a green downward facing arrow was implemented, typical of internet browser icons for this process, together the symbols 'www' and a globe both common representatives of the internet.
- File Compression (Fig 3b) – a yellow folder shape was used onto which a physical runner was integrated to represent the widely employed file archiving technique using the 'zip' compression algorithm.
- Image Processing (Fig 3c) – one of the basic and most commonly the foremost operation on image processing algorithms entails greyscaling of the digital image. This was represented using a photo with two colour variants (full colour and greyscale) within a typical photo frame.
- Content Searching (Fig 3d) – a newspaper article is represented in miniature with a magnifying glass physically oriented on parts of the text to represent the commonly employed symbols for computer text searching.

Each object was mounted onto an 8cm x 8cm wooden platform which was chosen so as to allow comfortable physical control and interaction with the objects. The size of this wooden platform also enabled the scaling of reacTIVision 'amoeba' fiducials [39] to be attached underneath as seen in Fig 3e. These symbols are orthogonally optimized for unique identification of each object, its centre point as well as the rotation angle of the tangible device using the installed camera.

This setup provides users the ability to interact with the system using accurate fiducial positioning, whereby placement of the objects in specific areas on the table triggers different algorithms. A timer was employed for this purpose which locked-in a tangible object with the system once the former is placed on a location for more than five seconds. The proposed framework also makes intrinsic use of spatial shifts of tangible objects as an interaction domain. Once locked within a process, the TUI objects are consistently spatially tracked and the system reacts according to the direction of motion followed. The unique nature of the 'amoeba' fiducial symbols further provide rotational uniqueness, which the framework exploits to provide an additional domain of interaction whereby students can alter the attributes of a process by rotating the physical object in a clockwise or anti-clockwise direction.

C. *User Interface*

Inherent to the benefits conveyed by proposed TUI system is the ability to enhance and interweave the physical tangible objects with digital information. Perceptual coupling of interactive embodiment is achieved by projecting onto the table a dynamic GUI which receives and reacts to controls and inputs provided from the tangible devices. Information, colours and sounds are dynamically altered as students interact with the platform and these provide direct feedback and computational

cooping to the interactions undertaken by the tangible objects. Interfacing with the developed Java based software algorithms, the GUI was developed using the JavaFX library, which enabled the use of visual components such as gauges and dynamic charts to further explain the occurring processes.

The initial design upon program start-up, as illustrated in Fig 4, consists of four main areas; the status dashboard, the queue and CPU loading dashboard, the processes description listing, and the thread and tasks area.

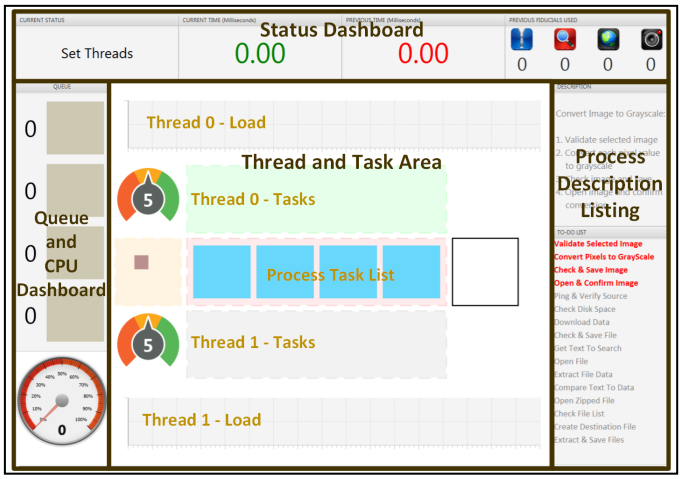

Fig. 4. GUI layout segmented according to the four main display areas

The top segment of the GUI, depicted in Fig 5, provides the student both information about the current state as well as data from previous system configurations, which can be used for real-time comparison. This dashboard affords students this by allowing direct comparison of both the current and previous simulation execution timings (in milliseconds) respectively. Furthermore, the user can also keep track of the amount of process executed in the previous run by means of representative icons on the top-right corner for more detailed comparison of the tasks undertaken. The Queue and CPU dashboard compliment this data by providing further information about the current simulation setup by enlisting process tasks which are queued for execution as well as a gauge measuring CPU load during runtime across both threads as seen in Fig 6b. The queue, illustrated in Fig 6a was designed to serve also as placeholders for the tangible objects being used so as the current processes queued can be visually associated using respective TUI objects.

Fig. 5. Status dashboard highlighting current system execution timings and state in direct comparison with previous process execution.

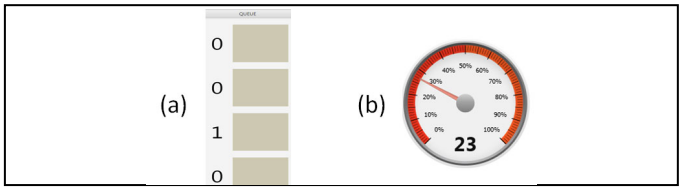

Fig. 6. Queue list for current simulation setup, together with CPU load monitoring gauge during multi-threaded execution.

The right section of the table interface contains a process description of the locked-in object, as well as a breakdown of the selected process into distinct sub-processes which need to be executed as depicted in Fig 7a. These are highlighted upon process selection and a red/green colour schema, captured in Fig 7b, is used to discriminate sub-processes which have been allocated on threads and others that still need to be scheduled. Moreover, this task list is also used during configured multi-threaded execution to highlight the sub-process which are currently being considered by the system. As shown in Fig 7c, the description section serves also as an area to explain to the students any execution error or exception encountered and thus provides formative feedback on the system status accordingly.

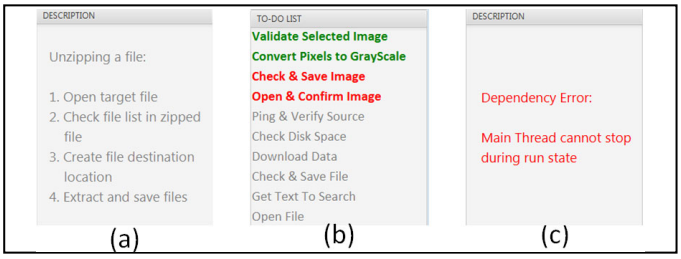

Fig. 7. Process description list with sub-processes breakdown used for scheduling and status feedback.

The central area within the system interface, highlighted in Fig 8, is the section in which the students will mainly interact with. To aid explaining multi-threaded scheduling concepts, the system adopted a two-threaded design which allowed a reduction in numerical complexity as well as aided better visualisation by students. This section is chiefly composed of the main and secondary threads, denoted as Thread 0 and Thread 1 in Fig 4 as well as a process breakdown section in the centre whereby locked-in processes are decomposed into sub-tasks and these are assigned onto individual threads using the tangible objects to spatial shift into location accordingly. The thread load area further allows the student to dynamically identify the sub-tasks that have been assigned to each thread, as depicted in Fig 8, with each process allocated a unique colour in harmonisation with the tangible object.

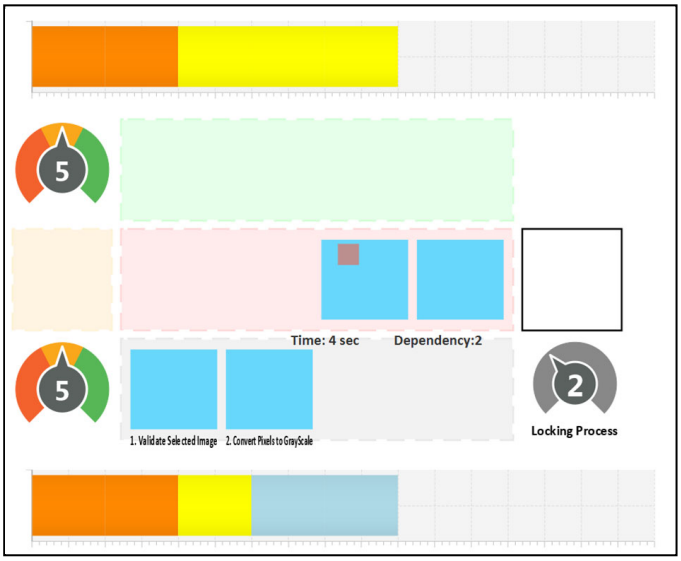

Fig. 8. Thread visualisation and sub-process allocation section.

D. Session

A complete session of the system is best understood as a series of stages within which the student undertakes to setup a multi-threaded environment. Making use of the TUI objects photographed in Fig 3, students are able to add a number of processes onto an execution queue which are then compiled and run with the execution time highlighted in the dashboard of Fig 5. As illustrated in Fig 8, once a tangible object is placed onto the process placeholder area, a lock-in five second countdown timer is commenced after which the individual process is decomposed into a set of sub-tasks placed in the middle of the process area as visualised with the blue components in Fig 4. The description and to-do list sections, illustrated in Fig 7, are concurrently updated accordingly.

Making use of the same tangible objects, students then allocate individual sub-process tasks onto either the main or secondary thread by physically dragging each task accordingly. In order to aid the understanding of task concurrency decisions, once locked-in on a sub-process, adjacent information is projected about the individualistic task duration and its relational dependency as seen in Fig 9a. Upon placing each sub-process into the thread task area, three circles highlighted in Fig 9b are displayed and the students can alter the task priorities by rotating and angling the tangible object to the selected value. This decision affects in turn the whole thread priority which will be assigned the highest allocated value as seen in the thread gauge of Fig 9b.

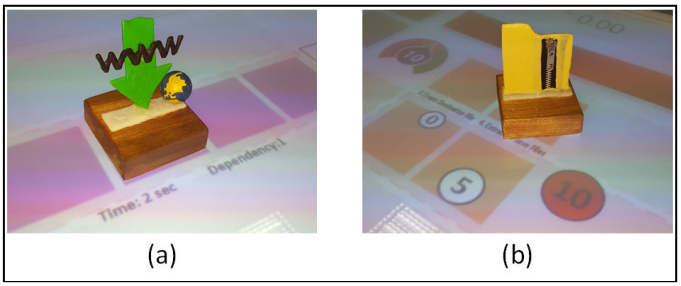

Fig. 9. Visualisation options for information relating to task dependency and user-allocated priority.

Following the successful allocation of the sub-tasks within a process, students can either opt to assign additional processes to the threads as illustrated in Fig 8, or allow a dedicated countdown timer of ten seconds to elapse following which simulation of the assigned tasks will commence. Upon runtime, the system will start completing the processes accordingly whilst updating the CPU gauge and the current time display in the status dashboard. Progress is tracked by students using the highlighting of the tasks being executed inside the to-do-list panel. Finally, a session ends with the execution duration and process details displayed to students in the status dashboard as shown in Fig 5, whereby details are also kept of previous runs for direct comparison of multi-threaded process allocation setups.

The system also detects exceptional instances when the main thread has been incorrectly allocated and scheduled to be idle whilst processes would still be running on the secondary thread. To allow students to visualise and understand the executional procedure undertaken in such an instance, the TUI system pauses the execution timer and an animation is displayed whereby progressive transfer occurs of the currently executing task from the secondary thread towards the main thread, after which the system execution carries on normally. Alternatively, if the student incorrectly allocated tasks which conflict at execution time due to internal process dependencies, an error is thrown by the system and execution is halted while details are displayed in the description area as shown in Fig 7c. In both these exceptional instances, the user attention is further attracted by the TUI system with the use of appropriate sounds and animations. These instances provide direct formative feedback to students and hence aid the conceptual learning better.

IV. Experimental Results and Discussion

A. Evaluation Methodology

The implementation of the TUI system was undertaken at Middlesex University Malta within a degree programme of Computer Science. Second year students reading a module in Engineering Software Development were selected for summative evaluation. These candidates had a prerequisite in object oriented programming basics and were introduced during their second year of study to concepts of computer architecture and operating systems. The topic of multi-threaded task scheduling was within syllabus of this module and the evaluation session was coordinated to coincide with the formal introduction to the multi-thread domain.

The class of nineteen (19) students aged between seventeen (17) and twenty-six (26) years old was recruited based on a convenience sampling technique. A random selection of seven (7) students was selected for the experimental group whilst the remaining twelve (12) students would compose the control group for the evaluation study. The selection of 7 students was based on the physical limitation on the amount of students that are able to successfully huddle around and interactively participate on the table-top system. The control group would be subject to a traditional lecture for introducing multi-thread task scheduling, whilst the experimental group would make use of the proposed TUI system for explanation of the same multi-threaded concepts. In order to reduce variable conditions between both cohorts, each session was allocated a fixed time and delivered by same module Lecturer on the same day.

The module was studied by students reading their degree in either full-time or part-time mode. This introduced a potential variation between students owing to their individualistic exposure and practical experience towards the subject of multi-threaded software development from potential industrial perspectives. Whilst the theoretical concepts of multi-thread task scheduling would be introduced within the coordinated session, an a-priori examination was undertaken by all students, so as to establish an individualistic knowledge baseline. This assessment was composed of ten (10) multi-thread scheduling related questions posed as a combination of open-ended and multiple choice questions.

Following this test, students were split into different rooms for the undertaking of their respective group session. Whilst the traditional lecture used conventional technological equipment

such as video projection and smartboard setups, the TUI session complimented the same projected slide material with the explanation on the proposed framework. Both sessions covered identical technical content and task examples, which involved mainly the understanding of various multi-threaded scheduling concepts and their potential related errors. Further to completing a tuition session, each cohort was provided with another questionnaire to answer. This test, whilst containing different questions than the first one, covered similar multi-threaded conceptual knowledge, using a combination of open-ended and multiple choice questions.

The evaluation process was hence designed to yield a quantitative analyses, whereby students would be evaluated on their answers to academic assessments. This provided the necessary data to objectively compare and quantify the ability of the proposed teaching methodology to convey the abstract notions of multi-threaded task scheduling procedures.

B. Results and Discussion

The data in Fig 10 represents the results obtained from all the participating students within their common pre-test technical questionnaire. The average mark obtained by the entire class in this initial test was of 27% with a standard deviation of 12%. The normally distributed data for this initial test, moreover outlines that students had in general a similar a-priori understanding of the subject.

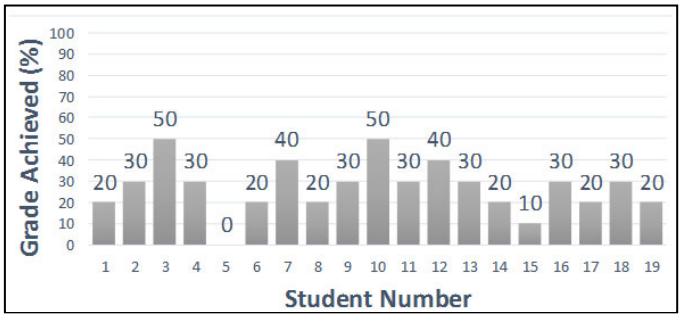

Fig. 10. Individual student grades during an a-priori examination.

The student cohort was divided randomly according to the predefined group sizes, and following each teaching intervention, a second technically similar test was provided to students. This allowed for a direct relational deduction of the each student's individual ability to understand the multi-threaded concepts conveyed in the respective session. The results in Fig 11 illustrate a personal comparison between the before and after grades obtained by individual students attending a traditional lecture on multi-threaded task scheduling (Fig 11a - blue) with respect to those being provided the same knowledge using the proposed TUI system (Fig 11b - red).

The comparative histograms in Fig 11a highlight that following the attendance to traditional lecture, in general students improved their understanding of multi-threaded programming, with an average mark improvement from 30 (SD 14.1) to an average mark of 48.3 (SD 19.9). This occurred even in light of students 10 and 12, who failed to understand the provided lecture and thus weren't able to answer the second set of questions correctly. Nevertheless, a paired sample t-test on the marks obtained by each student in this lecture-based group showed that an average grade increase of 18.3 marks has been registered with $p < 0.05$ and a test statistic of 2.99.

In relation to the control group, the students who learnt multi-thread concepts whilst using the proposed TUI system, Fig 11b, demonstrated an average mark increase from 22.8 (SD 7.6) to 77.1 (SD 17.0). A separate paired sample t-test on the a-priori and a-posteriori grades of the TUI learning cohort proved that the grade increase had a $p<0.001$ and a test statistic of 9.50, clearly highlighting the statistical significance of the obtained result. The mean difference in the grade improvement of both teaching techniques is depicted in Fig 12 together with the respective 95% confidence lower and upper bounds.

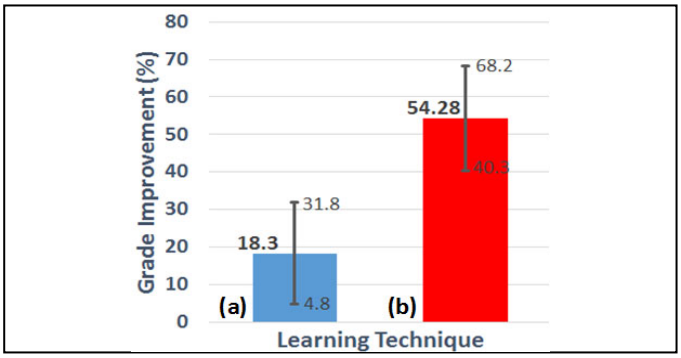

Fig. 12. Relative grade improvement obtained by students at 95% confidence bounds in a-posteriori examinations following; a) Traditional lecture session, b) Proposed TUI system session.

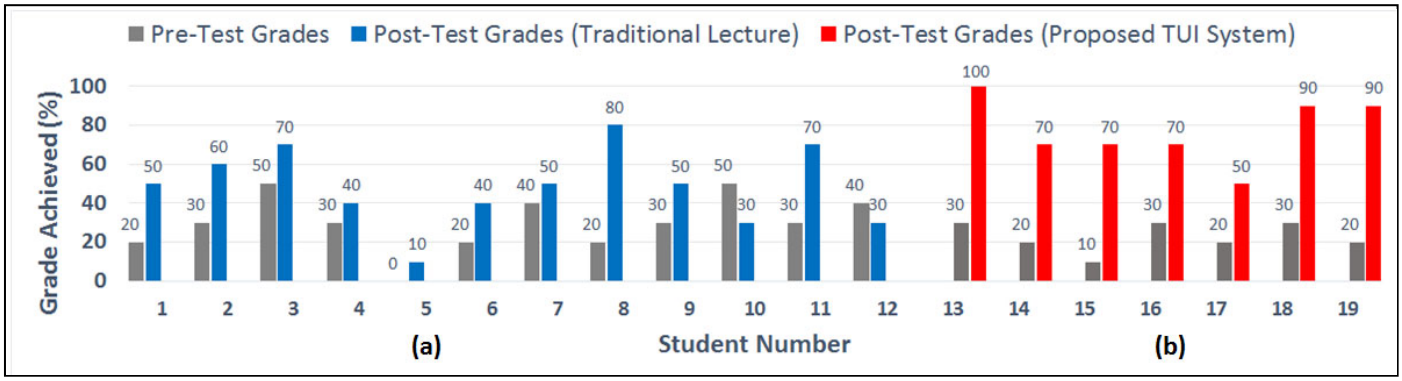

Fig. 11. Individualistic student comparison between test grades obtained before and after attending a learning session using; a) Traditional lecture session, b) Proposed TUI system session.

An independent sample t-test was further undertaken on the relative grade improvement from both examination marks between the two groups. Albeit, some variation can undoubtedly be attributed to the smaller group size tested for the TUI experiment, the results categorically endorsed the fact that students learning the abstract concepts of multi-threaded task scheduling using the proposed TUI system were able to attain 22.8 higher marks (SD 9.0) than the control group. This discrepancy was stated under Levene's test for equal population variance which proved the result statistically significant at $p<0.005$. The positive results from utilising the TUI system were also resounded subjectively by the students and lecturer alike, which reported a heightened sense of engagement whilst interacting with the proposed system. This encouraging phenomenon was observed both in terms of increased participation in discussions during the topic explanation as well as augmented group collaboration between students whilst configuring the system.

V. Conclusion

This paper has considered the use of a more practical approach to introduce second-year undergraduate students to the abstract concepts of multi-threaded task scheduling. A tangible user interface system was proposed which provided students the ability to physically interact and actively visualise the effects of scheduling on the execution of processes across different threads as well as appreciate better the situations leading to runtime execution errors. Through evaluation and analysis of the implemented experimental sessions, the effectiveness of the TUI system was objectively quantified with respect to a traditional lecturing approach. This concluded that TUI systems have great potential in aiding the delivery of higher educational concepts which are normally difficult to explain and challenging for students to comprehend.

Acknowledgment

The authors would like to thank Middlesex University Malta undergraduate students Mr. Daniel Sammut, Mr. Patrick Sammut, Mr. Matthew Mizzi, Mr. Keith Cassar and Mr. Carlos Spiteri for their valued contribution in the successful development and implementation of this work.

References

[1] J. Nasman and B. Cutler, "Evaluation of user interaction with daylighting simulation in a tangible user interface," *Automation in Construction,* vol. 36, pp. 117-127, 2013.

[2] J. Blasco-Arcas, I. Buil, B. Hernandez-Ortega and F. Javier Sese, "Using clickers in class. The role of interactivity, active collaborative learning and engagement in learning performance," *Computers & Education,* vol. 62, March, pp. 102-110, 2013.

[3] P. Thompson, "The digital natives as learners: Technology use patterns and approaches to learning," *Computers & Education,* vol. 65, July, pp. 12-33, 2013.

[4] V. J. Rideout, U. G. Foehr and D. F. Roberts, "Generation M2: Media in the Lives of 8- to 18-Year-Olds," Kaiser Family Foundation, Washington, D.C., 2010.

[5] R. R. Mellecker, L. Witherspoon and T. Watterson, "Active Learning: Educational Experiences Enhanced Through Technology-Driven Active Game Play," *The Journal of Educational Research,* vol. 106, no. 5, pp. 352-359, 2013.

[6] F. C. Blumberg and E. Altschuler, "From the playroom to the classroom: Children's views of video game play and academic learning," *Child Development Perspectives,* vol. 5, pp. 99-103, 2011.

[7] S. M. Jaeggi, M. Buschkuehl, J. Jonides and P. Shah, "Short and long-term benefits of cognitive training," *Psychological and Cognitive Sciences,* vol. 108, p. 10081–10086, 2011.

[8] K. Kellison and G. Font, ""Click, You're It!": The Role of Gaming in the K-12 Educational Setting," in *Design and Implementation of Educational Games: Theoretical and Practical Perspectives*, Hershey, PA: Information Science Reference, 2010, p. 278–292.

[9] G. Wolffe and C. Treffz, "Teaching parallel computing: new possibilities," *Journal of Computing Sciences in Colleges,* vol. 25, no. 1, pp. 21-28, 2009.

[10] Y. Bi and J. Beidler, "A visual tool for teaching multithreading in Java," *Journal of Computing Sciences in Colleges,* vol. 22, no. 6, pp. 156-163, 2007.

[11] M. Bedy, S. Carr, X. Huang and C.-K. Shene, "The design and construction of a user-level kernel for teaching multithreaded programming," in *Frontiers in Education Conference*, Puerto Rico, 1999.

[12] N. Giacaman, "Teaching by Example: Using Analogies and Live Coding Demonstrations to Teach Parallel Computing Concepts to Undergraduate Students," in *IEEE 26th Parallel and Distributed Processing Symposium,* Shanghai, 2012.

[13] G. Malnati, C. M. Cuva and C. Barberis, "JThreadSpy: teaching multithreading programming by analyzing execution traces," in *2007 ACM workshop on Parallel and distributed systems: testing and debugging*, London, United Kingdom, 2007.

[14] C.-K. Shene, "Multithreaded Programming in an Introduction to Operating Systems Course," in *29th SIGSCE Technical Symposium on Computer Science Education*, Atlanta, Georgia, 1998.

[15] H. Sutter, "A fundamental turn toward concurrency in software," *Dr. Dobb's Journal,* vol. 30, no. 3, 2005.

[16] R. Oechsle and T. Schmitt, "JAVAVIS: Automatic Program VIsualization with Object and Sequence Diagrams Using the Java Debug Interface (JDI)," *Ed. Lecture Notes in Computer Science,* vol. 2269, pp. 176-190, 2002.

[17] H. Leroux, A. Requile-Romanczuk and C. Mingins, "JACOT: a tool to dynamically visualise the execution of concurrent Java programs," in *2nd Int. Conf. on Principles and Practice of Programming in Java*, 2003.

[18] K. Mehner, "JaVis: A UML-Based Visualization and Debugging Enviroment for Concurrent Java Programs," *Ed. Lecture Notes in Computer Science,* vol. 2269, pp. 163-175, 2002.

[19] J. B. J. Trumper and J. Dollner, "Understanding complex multithreaded software systems by using trace visualization," in *5th international symposium on Software visualization*, Salt Lake City, Utah, USA, 2010.

[20] M. L. Maher and M. J. Kim, "Studying designers using a tabletop system for 3D design with a focus on the impact on spatial cognition," in *First IEEE International Workshop on Horizontal Interactive Human-Computer Systems (TableTop 2006)*, Sydney Univ., NSW, Australia, 2006.

[21] S. Wagner, Nusbaum, H and Goldin-Meadow, S, "Probing the mental representation of gesture: Is handwriting spatial?," *Journal of Memory and Language*, vol. 50, 2004.

[22] L. Garber, "Tangible User Interfaces: Technology You Can Touch," *Computer,* vol. 45, no. 6, pp. 15-18, June 2012.

[23] H. Ishii, "The tangible user interface and its evolution," *Communications of the ACM - Organic user interface,* vol. 51, no. 6, pp. 32-36, jUNE 2008.

[24] J. M. Rodríguez Corral, A. C. Balcells, A. M. Estévez, G. J. Moreno and M. J. Ferreiro Ramos, "A game-based approach to the teaching of object-oriented programming languages," *Computers & Education,* vol. 73, pp. 83-92, 2014.

[25] B. Schneider, P. Jermann, G. Zufferey and P. Dillenbourg, "Benefits of a

Tangible Interface for Collaborative Learning and Interaction," *IEEE Transactions on Learning Technologies,* vol. 4, no. 3, pp. 222-232, 2011.

[26] A. Dünser, J. Looser, R. Grasset, H. Seichter and M. Billinghurst, "Evaluation of Tangible User Interfaces for Desktop AR," in *2010 International Symposium on Ubiquitous Virtual Reality (ISUVR)*, Gwangju, 2010.

[27] T. Sapounidis and S. Demetriadis, "Tangible versus graphical user interfaces for robot programming: exploring cross-age children's preferences," *Personal and Ubiquitous Computing,* vol. 17, no. 8, pp. 1775-1786, 2013.

[28] D. Edge and A. Blackwell, "Correlates of the cognitive dimensions for tangible user interface," *Journal of Visual Languages & Computing,* vol. 17, no. 4, pp. 366-394, 2006.

[29] R. Waranusast, A. Bang-ngoen and J. Thipakorn, "Interactive tangible user interface for music learning," in *28th International Conference on Image and Vision Computing New Zealand*, New Zealand, 2013.

[30] H. Agrawal and K. Sorathia, "AstroGrasp: a tangible user interface for teaching basic astronomy concepts," in *11th Asia Pacific Conference on Computer Human Interaction*, Bangalore, India, 2013.

[31] M. Stringer, E. F. Toye, J. A. Rode and A. F. Blackwell, "Teaching rhetorical skills with a tangible user interface," in *Interaction design and children: building a community*, Maryland, USA, 2004.

[32] S. Yonemoto, T. Yotsumoto and R. Taniguchi, "A Tangible Interface for Hands-on Learning," in *International Conference on Information Visualisation*, London, England, 2006.

[33] D. Merrill, E. Sun and J. Kalanithi, "Sifteo cubes," in *Human Factors in Computing Systems*, Texas, USA, 2012.

[34] J. D. T. Vidarte, C. Rinderknecht, J. I. Kim and H. Kim, "A Tangible Interface for Learning Recursion and Functional Programming," in *International Symposium on Ubiquitous Virtual Reality (ISUVR)*, Gwangju, 2010.

[35] J.-L. Vinot, C. Letondal, R. Lesbordes, S. Chatty, S. Conversy and C. Hurte, "Tangible augmented reality for air traffic control," *Interactions,* vol. 21, no. 4, pp. 54-57, July 2014.

[36] L. Tateosian, H. Mitasova, B. Harmon, B. Fogleman, K. Weaver and R. Harmon, "TanGeoMS: Tangible Geospatial Modeling System," *IEEE Transactions on Visualization and Computer Graphics,* vol. 16, no. 6, pp. 1605-1612, 2010.

[37] A. Petrasova, B. Harmon, V. Petras and H. Mitasova, Tangible Modeling with Open Source GIS, Springer International Publishing, 2015.

[38] H. Ishii, "Tangible Bits: Beyond Pixels," in *2nd Int. Conf. on Tangible and Embedded Interaction*, Bonn, Germany, 2008.

[39] M. Kaltenbrunner and R. Bencina, "reacTIVision: a computer-vision framework for table-based tangible interaction," in *1st Int. Conf. on Tangible and embedded interaction*, Louisiana, USA, 2007.