Volume 7 • Issue 3 • September 2013

Editor-in-Chief
Dr. Chen-Chi Shing

INTERNATIONAL JOURNAL OF

COMPUTER SCIENCE AND SECURITY (IJCSS)

ISSN : 1985-1553

Publication Frequency: 6 Issues / Year

CSC PUBLISHERS
http://www.cscjournals.org

INTERNATIONAL JOURNAL OF COMPUTER SCIENCE AND SECURITY (IJCSS)

VOLUME 7, ISSUE 3, 2013

EDITED BY
DR. NABEEL TAHIR

ISSN (Online): 1985-1553

International Journal of Computer Science and Security is published both in traditional paper form and in Internet. This journal is published at the website http://www.cscjournals.org, maintained by Computer Science Journals (CSC Journals), Malaysia.

IJCSS Journal is a part of CSC Publishers
Computer Science Journals
http://www.cscjournals.org

INTERNATIONAL JOURNAL OF COMPUTER SCIENCE AND SECURITY (IJCSS)

Book: Volume 7, Issue 3, September 2013
Publishing Date: 15 - 09 - 2013
ISSN (Online): 1985 -1553

IJCSS Journal is a part of CSC Publishers
http://www.cscjournals.org

Published in Malaysia

Typesetting: Camera-ready by author, data conversation by CSC Publishing Services – CSC Journals, Malaysia

CSC Publishers, 2013

EDITORIAL PREFACE

This is Third Issue of Volume Seven of the International Journal of Computer Science and Security (IJCSS). IJCSS is an International refereed journal for publication of current research in computer science and computer security technologies. IJCSS publishes research papers dealing primarily with the technological aspects of computer science in general and computer security in particular. Publications of IJCSS are beneficial for researchers, academics, scholars, advanced students, practitioners, and those seeking an update on current experience, state of the art research theories and future prospects in relation to computer science in general but specific to computer security studies. Some important topics cover by IJCSS are databases, electronic commerce, multimedia, bioinformatics, signal processing, image processing, access control, computer security, cryptography, communications and data security, etc.

The initial efforts helped to shape the editorial policy and to sharpen the focus of the journal. Started with Volume 7, 2013, IJCSS appears with more focused issues. Besides normal publications, IJCSS intend to organized special issues on more focused topics. Each special issue will have a designated editor (editors) – either member of the editorial board or another recognized specialist in the respective field.

This journal publishes new dissertations and state of the art research to target its readership that not only includes researchers, industrialists and scientist but also advanced students and practitioners. The aim of IJCSS is to publish research which is not only technically proficient, but contains innovation or information for our international readers. In order to position IJCSS as one of the top International journal in computer science and security, a group of highly valuable and senior International scholars are serving its Editorial Board who ensures that each issue must publish qualitative research articles from International research communities relevant to Computer science and security fields.

IJCSS editors understand that how much it is important for authors and researchers to have their work published with a minimum delay after submission of their papers. They also strongly believe that the direct communication between the editors and authors are important for the welfare, quality and wellbeing of the Journal and its readers. Therefore, all activities from paper submission to paper publication are controlled through electronic systems that include electronic submission, editorial panel and review system that ensures rapid decision with least delays in the publication processes.

To build its international reputation, we are disseminating the publication information through Google Books, Google Scholar, Directory of Open Access Journals (DOAJ), Open J Gate, ScientificCommons, Docstoc and many more. Our International Editors are working on establishing ISI listing and a good impact factor for IJCSS. We would like to remind you that the success of our journal depends directly on the number of quality articles submitted for review. Accordingly, we would like to request your participation by submitting quality manuscripts for review and encouraging your colleagues to submit quality manuscripts for review. One of the great benefits we can provide to our prospective authors is the mentoring nature of our review process. IJCSS provides authors with high quality, helpful reviews that are shaped to assist authors in improving their manuscripts.

Editorial Board Members
International Journal of Computer Science and Security (IJCSS)

Malaysia

Assistant Professor Vishal Bharti
Maharishi Dayanand University
India

Dr. Parvinder Singh
University of Sc. & Tech
India

Assistant Professor Vishal Bharti
Maharishi Dayanand University,
India

TABLE OF CONTENTS

Volume 7, Issue 3, September 2013

Pages

A Simple Method to Build a Paper-Based Color Check Print of Colored Fabrics by Conventional Printers

Sajjad Fashandi sfashandi@aut.ac.ir
MSc./Department of Textile Engineering
Amirkabir University of Technology (Tehran Polytechnic)
Tehran, 15914, Iran

Seyed Hossein Amirshahi hamirsha@aut.ac.ir
Professor/Department of Textile Engineering
Amirkabir University of Technology (Tehran Polytechnic)
Tehran, 15914, Iran

Shahram Peyvandi peyvandi@aut.ac.ir
PhD/Department of Textile Engineering
Amirkabir University of Technology (Tehran Polytechnic)
Tehran, 15914, Iran

Abstract

An open loop color management system is implemented to reproduce an analog color of a set of colored fabrics by a digital inkjet printer. A tetrahedral interpolation technique is designed for mapping between device-dependent (RGB) and device-independent (CIELAB) color spaces. A set of 3164 color patches are used as training set in 3-D LookUp Table (LUT) to characterize the color printer. Then, the designed color management system is examined by the colorimetric reproduction of a set of 30 colored fabrics using the conventional inkjet printer. The performance of the system is numerically evaluated by measuring the color difference values between the original and the reproduced samples. The results showed that the color reproduction system appropriately works for both groups of samples located inside the color gamut of output device, i.e. printer, and those out of gamut samples while the later logically leads to greater errors.

Keywords: Color Management System (CMS), Inkjet Color Printer, Colorimetric Reproduction, Lookup Table (LUT), Paper Check Print.

1. INTRODUCTION

The most applicable color matching algorithms in the traditional analog color reproduction, in which the colorant concentrations and the corresponding tristimulus values continuously change, were developed by Allen based on the single and two constant Kubelka-Munk theories [1, 2]. While the suggested methodologies try to match the colorimetric tristimulus values of target under a given set of viewing conditions, some methods based on the least squares fitting of reflectance spectrum of target was developed and is known as spectrophotometric matching [3].

In the recent decades, the modern digital instruments have been developed for color reproduction of scenes and objects. The more advanced color reproduction devices with digitally adjustable user controls have become more popular in the past few years and widely used in different media such as paper, plastic, textile as well as displaying units [4]. The simplicity of producing colors in the digital instruments especially in the forms of printed papers and display units has led to introducing some systems for successful transformation of colors within different digital devices like scanners, cameras, printers and monitors. The accurate reproduction of colors between different digital instruments would be guaranteed by an appropriate color management system (CMS). By using the CMS, all input digital signals (mostly in RGB color space) are mapped into a standard color space (CIEXYZ or CIELAB) and finally digital signals (RGB or CMYK) are delivered in output devices. The intermediate analog color space is called as profile connection space (PCS) and known as the "heart of CMS" [5, 6].

A type of conversion between the analog and digital colors occurs in digital devices such as monitors and printers. Several methods, such as polynomial transforms [7], physical models [8-10], artificial neural networks (ANNs) [11] and lookup tables (LUTs) [12] have been proposed to establish such mutual connections. Zhang *et al.* [7] employed the polynomial transforms for mapping XYZ tristimulus values of color patches to those of CMYK signals. They used IT 8.7/2-1993 color chart with 286 patches to specify the coefficients of polynomial transforms. Eq. 1 simply shows the first order polynomial transformation in matrix form, in which the tristimulus values *XYZ* are predicted from *CMYK* signals. Results of utilizing the second, third and fourth order polynomial transformations were also reported in mentioned article.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 & a_4 & a_5 \\ a_6 & a_7 & a_8 & a_9 & a_{10} \\ a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \end{bmatrix} \begin{bmatrix} C \\ M \\ Y \\ K \\ 1 \end{bmatrix} \tag{1}$$

Bezerra *et al.* [8] used partitive color mixing theory as a physical model to predict the colors on paper using a paper ink-jet printer. The mathematical approach of the model is shown by Eq. 2,

$$\begin{aligned} X_{mix} &= \sum_i a_i X_i \\ Y_{mix} &= \sum_i a_i Y_i \\ Z_{mix} &= \sum_i a_i Z_i \end{aligned} \tag{2}$$

where, X_{mix}, Y_{mix}, Z_{mix} and X_i, Y_i, and Z_i are the tristimulus values of the mixture and the i[th] colors, respectively and a_i are the fractional areas of the colors, i.e., $\sum_i a_i = 1$. As shown by Eq. 3, the resultant color $(X_{mix}, Y_{mix}, Z_{mix})$ of the overall image could be then calculated by the Neugebauer equations and the summation of the weighted tristimulus values of all fractional areas that could be eight for a CMY printer.

$$\begin{aligned} X_{mix} &= (1-c)(1-m)(1-y)X_W + c(1-m)(1-y)X_C + m(1-c)(1-y)X_M + y(1-m)(1-c)X_Y + my(1-c)X_R + \\ &\quad cy(1-m)X_G + cm(1-y)X_B + cmyX_K . \\ Y_{mix} &= (1-c)(1-m)(1-y)Y_W + c(1-m)(1-y)Y_C + m(1-c)(1-y)Y_M + y(1-m)(1-c)Y_Y + my(1-c)Y_R + \\ &\quad cy(1-m)Y_G + cm(1-y)Y_B + cmyY_K . \\ Z_{mix} &= (1-c)(1-m)(1-y)Z_W + c(1-m)(1-y)Z_C + m(1-c)(1-y)Z_M + y(1-m)(1-c)Z_Y + my(1-c)Z_R + \\ &\quad cy(1-m)Z_G + cm(1-y)Z_B + cmyZ_K . \end{aligned} \tag{3}$$

in which, the subscripts C, M, Y, R, G, B, W and K respectively stand for cyan, magenta, yellow, red, green, blue, white and black to represent the corresponding *X*, *Y* and *Z* tristimulus values. The areas covered by such subtractive primaries cyan, magenta, yellow are shown by *c*, *m* and *y*, respectively.

Zuffi *et al.* [9] employed a physical model to spectrally characterize the printers. They used the Yule-Nielsen spectral Neugebauer equation to model the printer. The authors also used the genetic algorithm to estimate the model's parameters. The genetic algorithm was also employed to tune a spectral printer model based on the Yule-Nielsen modified Neugebauer equation [10]. Three different types of printers as well as different papers and printer drivers were used in this research. An artificial neural networks (ANN) was used by Vrhel [11] to approximate the color characterization of multilayer lookup table (MLUT). While the MLUT could be logically large for such embedded systems, the ANN was used to provide a more compressed version of function approximation. Vrhel *et al.* [12] used the MLUT technique to map the device-dependent color space (RGB values) to the device-independent standard CIELAB color space ($L^*a^*b^*$ values).

In the present paper, a standard 3D-LUT and a tetrahedral interpolation technique are used to map the device-independent color space (CIELAB) of colored fabrics to device-dependent color space (RGB). In fact, the CIE $L^*a^*b^*$ colorimetric values of colored fabrics are converted to an RGB color space and reproduced by a commercial printer on paper. Whereas some input colors are not in the gamut of destination space that is the color inkjet printer, they are clipped into the gamut of the printer by a centroid clipping color gamut mapping algorithm. Since the color reproduction of textile through a coloration process requires a time and energy consuming procedure, the main goal of the present research is to reproduce the colors of textiles on paper using an inkjet printer for presenting a color check print with known CIELAB values.

2. DESIGNING A TYPICAL CMS

To set up a typical color management system (CMS), some principles for printer mapping, inverse printer mapping and the gamut mapping were employed in this research.

2.1. Printer Map

For mapping between the RGB and the CIELAB values, a tetrahedral interpolation technique was used [13]. The technique is a precise method for interpolating the regularly sampled LUTs. As Figure 1 shows, the tetrahedral interpolation technique divides a cube into six tetrahedrons. The interpolated values are the weighted sum of the values of the function at the four vertices of the tetrahedral enclosing the desired points. The formulation of the method could be described by Eq. 4,

$$P = P_{000} + P_x \frac{x - x_0}{x_1 - x_0} + P_y \frac{y - y_0}{y_1 - y_0} + P_z \frac{z - z_0}{z_1 - z_0} \tag{4}$$

where, the surrounding eight nodes in the RGB color space are [n_{000} n_{001} n_{010} n_{011} n_{100} n_{101} n_{110} n_{111}] and the corresponding points in the CIELAB space are [p_{000} p_{001} p_{010} p_{011} p_{100} p_{101} p_{110} p_{111}], while the $[x_0\ y_0\ z_0]$ and $[x_1\ y_1\ z_1]$ are the coordinates of n_{000} and n_{111} respectively. The expressions for P_x, P_y and P_z depend on the location of P with respect to the six tetrahedrons and as the result, the output P could be found through the proposed $[x\ y\ z]$ input values.

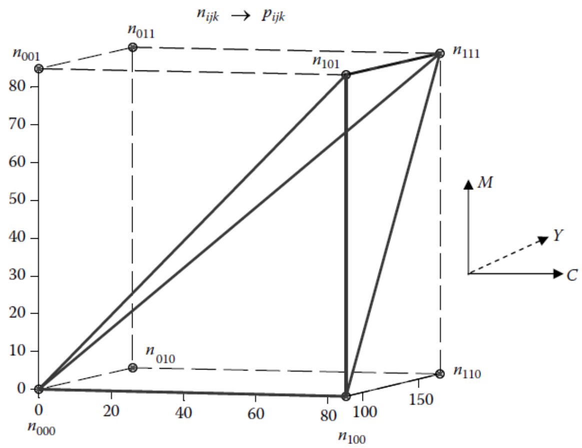

FIGURE 1: The Tetrahedral Interpolation [14].

The printer inverse model could be theoretically viewed as an inverse LUT and simply could be achieved from the inverse of the forward LUT; however, the method is not practically applicable. In fact, this kind of LUT may not be well defined for the colors at the boundary of gamut and may lead to multivalued outputs [14].

2.2. Inverse Printer Map

Multidimensional interpolation approaches, such as tetrahedral, conjugate gradient (CG) and iteratively clustered interpolation (ICI) algorithms are often used to produce inverse printer map [14]. In this research, the ICI algorithm is used to invert the forward model of printer. The algorithm is a gradient-based optimization method that improves the initial points by using an iterative technique [15].

2.3. The Printer Color Gamut Mapping

The printer gamut is usually restricted to a significantly smaller range in comparison to the gamut of the source digital image due to the physical limitations of the printer's primaries. The colors that could be found in the source gamut and would not be available in the output gamut are said to be out of gamut and should be converted to printable colors through a transformation technique called gamut mapping. Different techniques such as gamut clipping and gamut compression were suggested to deal with such problem [14]. In this paper, the centroid clipping color gamut mapping algorithm was used to map the out of gamut colors into the printer gamut [16].

3. EXPERIMENTAL

Three essential algorithms, i.e. tetrahedral interpolation for printer mapping, inverse printer map and color gamut mapping were designed in this research to perform the experimental study. Unlike the typical color management systems in which PCS is an intermediate analog color space, the PCS was considered as the initial space for input in the designed CMS in this research. Actually, the major difference between the employed and the classical CMSs is the profile connection space that opposed to classical method, it uses the analogue L*a*b* colorimetric data as inputs and provides digital RGB values in the outputs. In fact, the model could be considered as an abridged version of the general CMS model. In the other words, instead of using a digital image as input, analog tristimulus values of a colored fabric are used as inputs of designed CMS. In fact, the designed CMS could be considered as a digital approach for color matching in textile industry.

3.1. Training Color Charts

In this research, the training color charts were produced with Adobe Illustrator CS5. The color management system was turned off in this software to produce color charts. A tonal range of a unique hue was considered to create 126 color patches on each page. Subsequently, the saturation and lightness of colors of a given hue respectively varied from left to right and top to bottom for each hue. Accordingly, with 25 different hues on each page and 14 gray samples (gray ramp), the total of 3164 color patches were prepared to characterize the printer.

In order to achieve digital images from vector-base objects that were designed in Adobe Illustrator CS5, the produced colored samples were exported to Adobe Photoshop CS5. The color management system of software was again deactivated to avoid any possible change in the original values of color specifications. A low price commercial CMYK color printer named Epson Stylus T27 was used as the printing device. The color charts were printed on 260 $g\,m^{-2}$ A4 quality photo glossy papers that is commercially named EUNP5080 and supplied by UNIK Int. In fact, the small dot gain and good reproducibility can be expected by this type of paper. The original inks from Epson were used in printing process whose spectral reflectances are presented in Figure 2. The reflectance spectra of printed samples were measured from 400 nm to 700 nm at 10 nm intervals using a portable spectrophotometer named Eye One Pro from GretagMacbeth. Figure 3 shows the CIE a^*b^* and L^*C^* scatter-plots of the printer inks under D65 illuminant and 1964 standard observer.

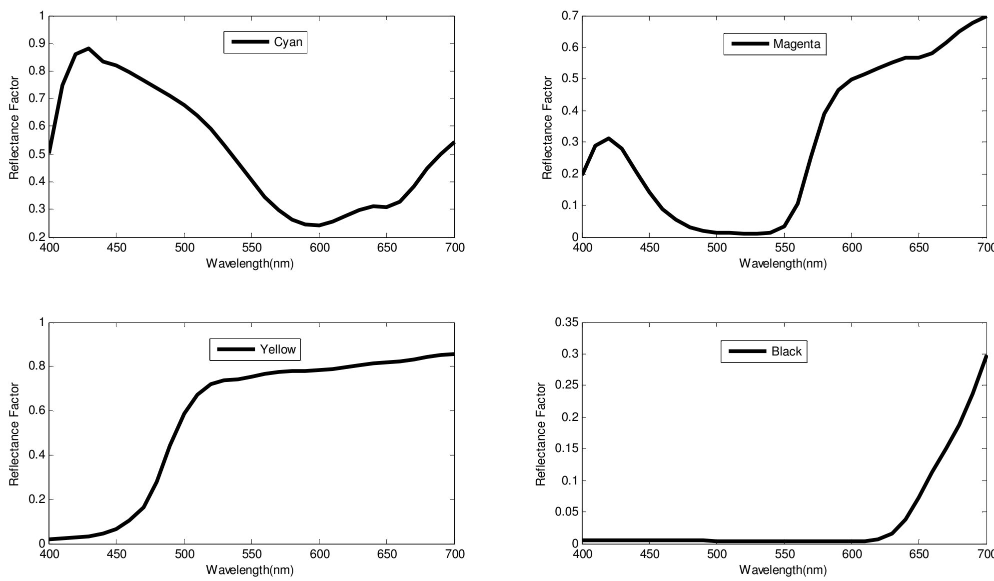

FIGURE 2: The Reflectance Spectra of Printer Primaries.

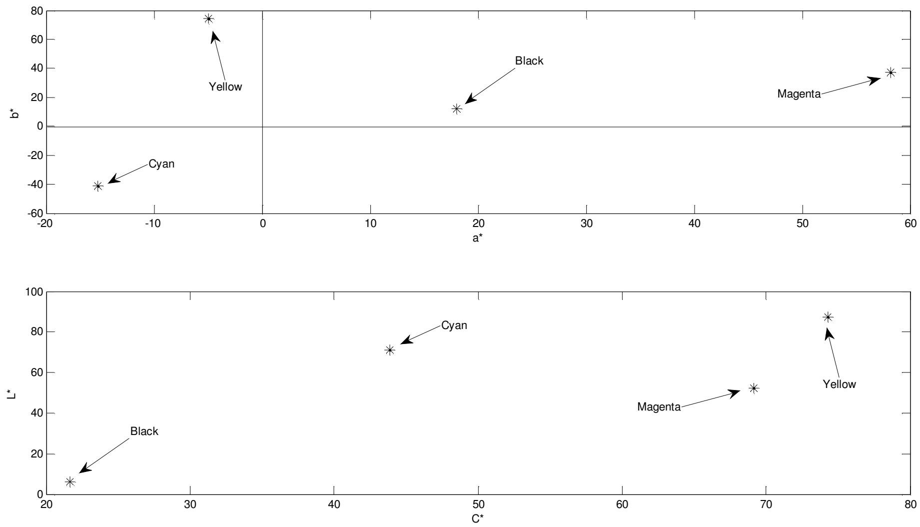

FIGURE 3: The CIE *a*b** and *C*L** Plots of Printer Primaries.

Since the Epson Stylus T27 is not a postscript printer, the RGB values were used as device dependent color space [17]. The produced color charts were firstly converted to TIFF format, with no embedded ICC profile in the Adobe Photoshop CS5, to determine the RGB values of color patches for further computations. It should be emphasized that the RGB values in TIFF format are as same as PSD format and the Adobe Photoshop CS5 was only an intermediate platform for producing and printing of training charts. Figure 4 shows the block diagram of the training charts production, printing and measuring attempts that were made in this work.

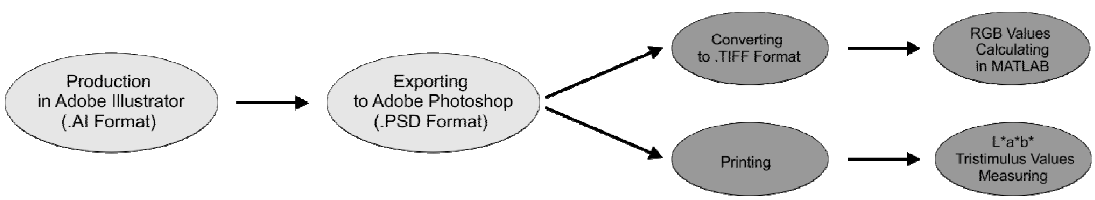

FIGURE 4: The block diagram of production of training charts, printing and color measuring attempts.

3.2. Test Color Chart

In order to evaluate the performance of the designed color management system and the efficacy of printer characterization, the spectral reflectances of 30 colored fabrics were measured using the GretagMacbeth Color-Eye 7000A spectrophotometer. The spectrophotometer benefited from integrated sphere with d/8 measurement geometry that makes it suitable for measuring the reflectance spectra of textured surfaces such as textiles. The spectral range was from 400 nm to 700 nm at 10 nm intervals. The specular component of reflectance was excluded and the medium aperture was used. Similar to the training charts, the CIELAB color values of samples were computed under D65 illuminant and 1964 standard observer. The CIE a^*b^* and L^*C^* scatter plots of colored fabrics are shown in Figure 5. The colorimetric values of the colored fabrics were considered as the target color for reproduction on paper media. To do so, the computed colorimetric values were imported as input data to the designed color management system and the resultant images were exported in TIFF format as output. Same as training color charts, Adobe Photoshop CS5, was employed to print the images. Figure 6 shows the block diagram of the designed CMS.

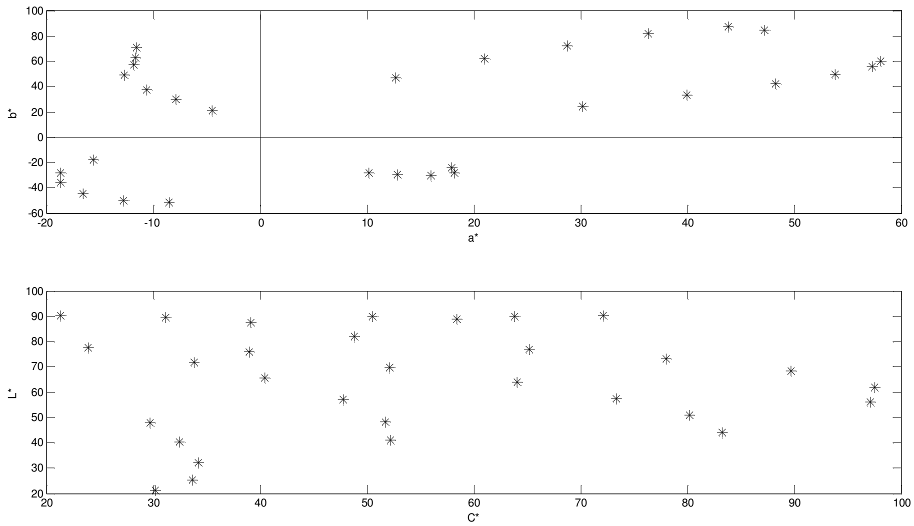

FIGURE 5: The CIE *a*b** and *C*L** Scatter Plots of 30 Colored Fabrics.

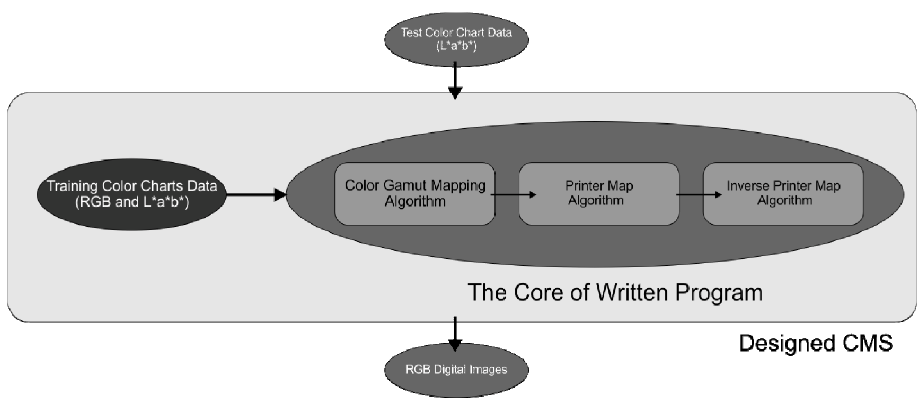

FIGURE 6: The Block Diagram of Designed CMS.

3.3. Evaluation of Color Management System

The performance of designed color management system was evaluated by calculating the CIELAB color difference values between the mapped initial CIELAB textile samples as the target colors (mapped $L^*a^*b^*$ in Figure 7) and the measured CIELAB values of color patches printed on the paper as the test chart (measured $L^*a^*b^*$ in Figure 7) using D65 standard illuminant and 1964 standard observer. While ICI algorithm was used for producing output RGB values, the calculated $L^*a^*b^*$ was the resultant of these RGB values. Hence the calculated ΔE^*_{ab} is the CIELAB color difference value between the mapped $L^*a^*b^*$ and the calculated $L^*a^*b^*$ values. In the case of calculated $L^*a^*b^*$ values tend to the mapped $L^*a^*b^*$, then the color difference between the mapped $L^*a^*b^*$ and the measured $L^*a^*b^*$ values becomes smaller. Figure 7 shows the workflow of the evaluation process.

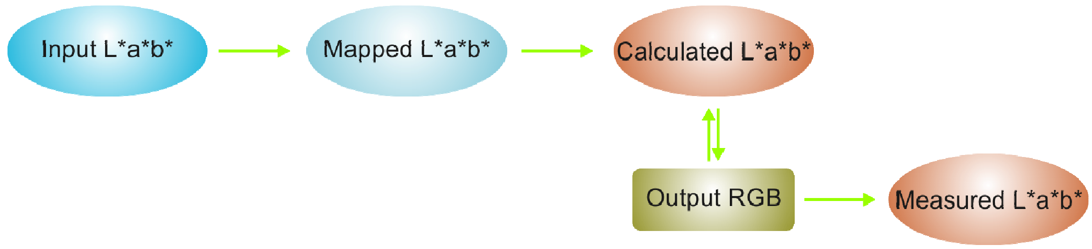

FIGURE 7: The Workflow for the Evaluation of Designed CMS.

4. RESULTS AND DISCUSSIONS

To evaluate the designed CMS, the colors of 30 colored fabrics were reproduced by the employed printer on the paper using the assembled color management system and the color difference values between the pairs were computed. The suggested recipes by the system were then employed for practical reproduction of colorimetric matches of colored fabrics on the paper and the color difference values were also measured. Since the printer inks were different from the textile dyestuffs, some colored fabrics were out of gamut and could not be reproduced by the printer primaries. Figure 8 shows the color gamut of the printer as well as the color specifications of targets, i.e. colored fabrics, in 3D space. It is evident in Figure 8 that some samples are out of printer gamut, hence the color gamut mapping process was performed before estimating the output RGB by intended CMS. So, the designed CMS model provided the mapped samples of those out of gamut colors before printing such samples on paper.

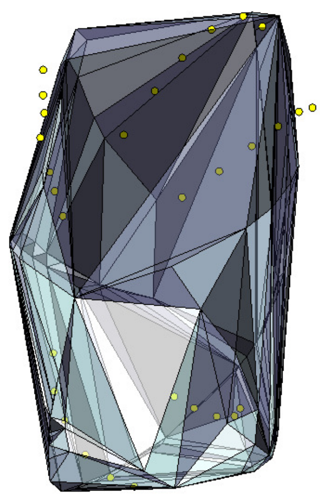

FIGURE 8: Color coordinates of the target fabrics (yellow dot points) with respect to the printer color gamut.

The average of the computed color differences between the targets and the estimated samples by the model as well as those between the targets and the physically printed samples are presented in Table 1. The table also shows the minimum, maximum, mean, median and the standard deviation of the color difference values. The reported results in Table 1 were achieved by creating the CMS with all 3164 available samples as the training chart. As the results shows, the median of the measured color differences is equal to 4.55 while, the minimum of 1.25 indicates to somehow large color difference value. The problem originates from the fact that the spectrophotometric and colorimetric properties of employed inks in printers were far from those dyestuffs that were employed in the textile industry. Furthermore, two different instruments with different geometries were employed for spectral and colorimetric measurement of papers and fabrics due to different surface properties. Besides, the reproducibility of the employed commercial printer was not high enough that could lead to some types of reproducibility problem.

Type of calculation	ΔE^*_{ab}				
	Min	**Max**	**Mean**	**Median**	**Std***
Calculated	0.85	10.14	4.55	4.48	2.48
Measured	1.25	10.45	4.70	4.55	2.29

* Standard Deviation

TABLE 1: The minimum, maximum, mean, median and standard deviation of the calculated and the measured color difference values between the colored fabrics and the reproduced samples on the paper under D65 and CIE 1964 standard observer.

Figure 9 shows the reflectance spectra of the six randomly selected targets and the corresponding matches on paper. Table 2 also shows the color differences values of these targets and the corresponding matches under D65, D50 and A standard illuminants. According to this table, different spectral behaviors of employed primaries in printers and dyes in the textiles have led to metameric (parameric) match that are more evident for samples #3, 5 and 6 by greater color difference values under different illuminants.

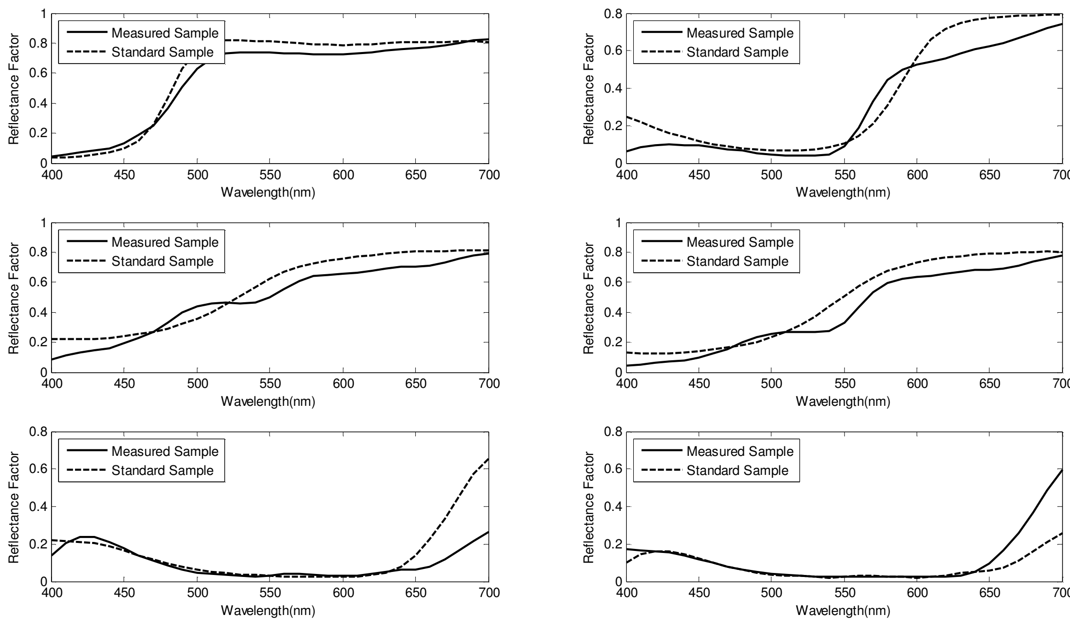

FIGURE 9: The reflectance spectra of 6 randomly selected targets and their corresponding matches.

Sample #	**Illuminant**		
	D65	**D50**	**A**
	ΔE^*_{ab}		
1	10.07	7.74	4.32
2	11.32	8.96	6.94
3	4.46	6.75	6.94
4	6.27	6.24	7.13
5	3.74	2.81	5.06
6	1.29	2.09	6.43

TABLE 2: The measured CIE ΔE^*_{ab} color difference values of 6 arbitrary selected colored targets and the corresponding matches under D65, D50 and A illuminants.

The effect of the number of training samples on the achieved color difference values was also studied. In fact, when the training samples are sparse, the chance of finding of samples in the region of target color would decrease. In order to investigate the effect of the number and the suitability of training samples on the performances of model, the reproducing procedure was carried out by using three different training sets. It is interesting that while the regression technique strongly depends on the specification of all samples in the training dataset, the interpolation technique based on LUT works locally and is only affected by the specifications of neighborhood samples around the desired sample. In fact, increasing the number of samples in training charts decreases the size of tessellation. So adding some extra samples in LUT could probably have no effect on the majority of other areas and therefore, on the final result. The examples of this issue are shown in Table 3 by reporting the color difference values of 10 randomly selected samples. The highlighted cells in the table show the color difference values that have not been affected by increasing the number of samples in the training charts. As mentioned earlier, increasing the color samples in the training chart at a given area does not guarantee the better performances for those in other areas. Figure 10 shows the scatter plot of a^* versus b^* values of color patches used in training sequence. As the figure shows, different numbers of training samples were used in LUT. The a^*b^* positions of the selected arbitrary samples are also shown by red circles in the plots. As Figure 10 demonstrates, the blank spaces in the LUT decreased by increasing of samples in training charts that could lead to more reliable results by improving the precision of the interpolation method. Thus, the appropriate learning based color reproduction system with acceptable results can be constructed by suitable as well as sufficient samples in the training charts whose color coordinates are properly distributed over the space.

Sample #	Sample specifications in CIEL*a*b* color system	ΔE^*_{ab} Number of samples in the training set 1400	2408	3164
1	L^*=50, a^*=80, b^*=40	10.64	3.55	3.55
2	L^*=50, a^*=-80, b^*=40	7.90	7.90	4.12
3	L^*=50, a^*=80, b^*=-40	10.75	4.41	4.41
4	L^*=50, a^*=-80, b^*=-40	4.81	4.81	4.81
5	L^*=80, a^*=50, b^*=-20	5.22	3.40	3.40
6	L^*=20, a^*=-40, b^*=60	6.54	6.54	5.62
7	L^*=60, a^*=20, b^*=80	14.75	7.17	5.19
8	L^*=60, a^*=40, b^*=80	22.84	6.60	6.60
9	L^*=60, a^*=-20, b^*=-80	15.35	15.35	8.58
10	L^*=60, a^*=-20, b^*=80	6.21	6.21	4.88

TABLE 3: The calculated color difference values against the number of samples in the training set.

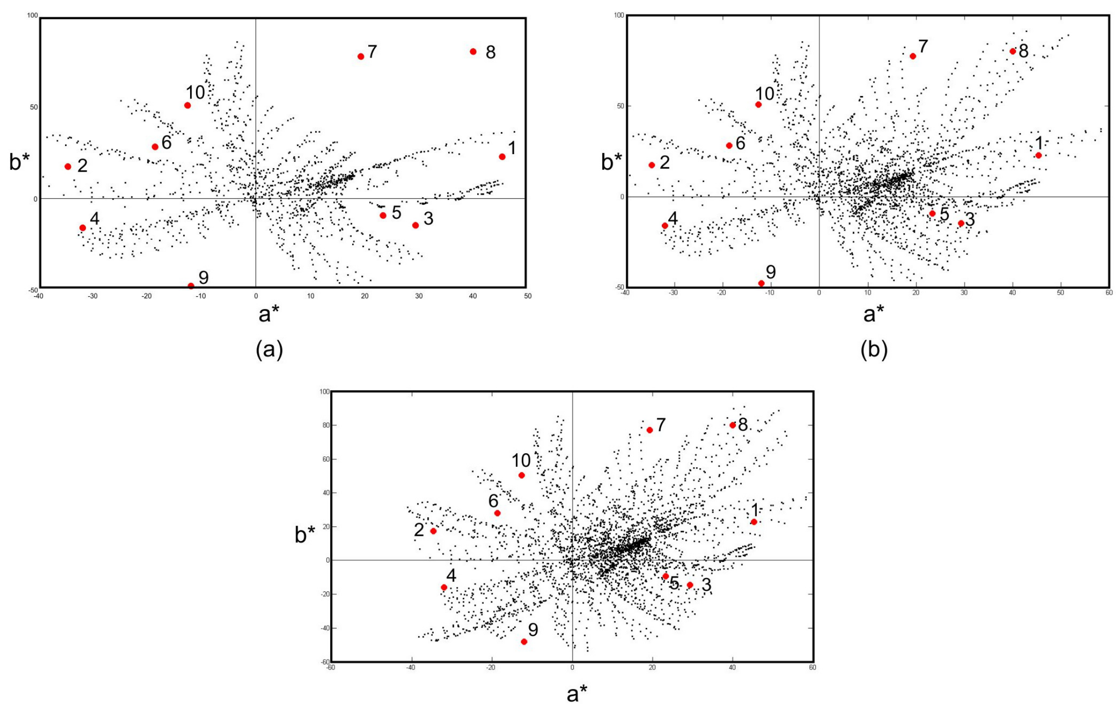

FIGURE 10: The CIE *a*b** scatter plot of color patches of the training charts. a) 1400 training samples, b) 2408 training samples and c) 3164 training samples. The red dots are the CIE a^*b^* specifications of 10 samples that their colors differences are reported in Table 3.

5. CONCLUSION

Implementing of an open loop color management system was suggested for colorimetric reproduction of a set of colored fabrics. To fulfill such plan, a color management system was designed to convert the CIELAB colorimetric coordinates to RGB values. By this arrangement, the system converted the analog CIE $L^*a^*b^*$ values of samples to digital RGB signals of printer.

The purpose was practically tested by designing a 3-D LUT technique to map between the color spaces. A set of color charts including of 3164 color patches was prepared to characterize the printer.

The designed color management system was assessed by colorimetric reproduction of a set of 30 colored fabrics on paper by a low price conventional printer. The CIELAB color coordinates of fabrics were introduced as inputs to a color management system and their colors were reproduced on desired paper using a characterized commercial inkjet printer. Then, the performance of the employed color reproduction system was practically examined by evaluating the color difference values between the targets and the corresponding matches reproduced on paper. The results of color reproduction were generally acceptable; nevertheless some samples suffered from somewhat high color difference values. It was shown that the accuracy of the system strongly depends on the number of samples in the training set together with their distribution in the color space.

It is essential to note that because of the process non-linearity, using the tetrahedral interpolation instead of polynomial transformation is either conventional or necessary in digital paper printing. But in comparison with other prevalent textile digital color management researches, Using the tetrahedral interpolation, gamut mapping, making custom training charts and results investigation are the novelties of this research in textile scope.

For future research it is suitable to use a color appearance model instead of a color model. A color appearance model matches the appearance of color and is independent from viewing condition. So having these kinds of models, the color reproduction process will be completed.

Also the result can be extended to paint, plastic and cosmetic industries for color reproduction using digital media.

6. REFERENCES

[1] E. Allen. “Basic Equations Used in Computer Color Matching”. J Opt. Soc. Am., vol. 56, pp.1256-1259, 1966.

[2] E. Allen. “Basic Equations Used in Computer Color Matching II”. J Opt. Soc. Am., vol. 64, pp. 991-993, 1974.

[3] P.H. McGinnis. “Spectrophotometric Color Matching with the Least Squares Technique”. Col. Eng., vol. 5, pp. 22-27, 1967.

[4] R.S. Berns, “A generic approach to color modeling”. Color Res. Appl., vol. 22, pp. 318-325, 1997.

[5] D.R. Wyble and R.S. Berns. “A Critical Review of Spectral Models Applied to Binary Color Printing”. Color Res. Appl., vol. 25, pp. 4-19, 2000.

[6] R.S. Berns, Billymer and Saltzman's Principles of Color Technology, 3rd ed., New York: Wiley and Sons, 2000.

[7] Y. Zhang, V. Cheung, S. Westland and K.J. Beverley. “Colour Management of a Low-Cost Four Colour Ink-Jet Printing System on Textiles”. Color. Technol., vol. 125, pp. 29-35 ,2009.

[8] C.D.M. Bezerra, C.J. Hawkyard, H.M. Kulube and R.S. Reyner. “Colour Matching for Ink-Jet Prints on Paper”. Color Res. Appl., vol. 23, pp. 18-26, 1998.

[9] S. Zuffi, R. Schettini and G. Mauri. “Spectral-based Printer Modeling and Characterization”. J Electron. Imaging, vol. 14, pp. 023008, 2005.

[10] S. Zuffi, R. Schettini and G. Mauri. "Using Genetic Algorithms for Spectral-based Printer Characterization". In Proceedings of the IS&T's Conference on Image Quality, 2003, pp. 268-275.

[11] M.J. Vrhel. "Approximation of Color Characterization MLUTS with Artificial Neural Networks". In Proceedings of the IEEE International Conference on Image Processing, 2003, pp. 465-468.

[12] M.J. Vrhel and H.J. Trussell. "Color Printer Characterization in MATLAB". In Proceedings of the IEEE International Conference on Image Processing, 2002, pp. 457-460.

[13] G. Sharma. Digital Color Imaging Handbook, Boca Raton FL: CRC Press, 2003.

[14] L.K. Mestha and S.A. Dianat. Control of Color Imaging Systems, Analysis and Design", 1st Ed., New York: CRC Press, 2009.

[15] D. Viassolo and L.K. Mestha. "A practical algorithm for the inversion of an experimental input-output color map for color correction". J Opt. Eng., vol. 42, pp. 625, 2003.

[16] K. Braun, R. Balasubramanian and R. Eschbach. "Development and evaluation of six gamut-mapping algorithms for pictorial images". In Proceedings of the IS&T and SID's 7th Color Imaging Conference, 1999, pp. 144-148.

[17] "RGB Printer Profiles: Adobe Photoshop CS3/CS4, Need To Know: X-Rite Color Service".Internet:http://xritephoto.com/documents/literature/en/RGBPrinterNTK_EN.pdf, 2011.

Empirical Design Guidelines for Enhanced Incorporation of Task Management in Web Browsing

Saad Alharbi *stharbi@taibahu.edu.sa*
College of Computer Science and Engineering
Taibah University
Medina, Saudi Arabia

Abstract

Task management features have become a necessity in web browsing, especially with the high proliferation of pages and information in the web. This paper presents a novel approach called TaskBar which helps manage pending tasks during web browsing. It works as a to-do list in the web browser and provides various task management features such as reminders and priorities to help decide which tasks should be dealt with first. A two-session controlled experiment was carried out to evaluate TaskBar and compare users' performance with and without task management features. The obtained data were analyzed in terms of task accomplishment time, rate of completion, and users' satisfaction. The results showed that incorporating task management features in web browsing, particularly TaskBar, significantly improved users' performance in terms of task completion time, completion rate, and satisfaction. These results were interpreted into a set of design guidelines for the employment of task management features in web browsers.

Keywords: Browser, Design, Guidelines, Revisit, Task Management, Web.

1. INTRODUCTION

The importance of adopting task management in web browsers increases every day, especially with the vast growth of web pages and information in the web. The web browser has become the place where we perform many of our daily tasks such as flight booking and arranging a meeting. However, this dramatic change in using the web faces a relatively slow change in web browsers. For instance, bookmarks and history are still considered the main ways of saving web pages for later use. In addition, recent studies showed that users sometimes print and email themselves web pages for later use [1]. Many studies, as a result, have been carried out to investigate incorporating web browsers with task management features. Most of these studies focused mainly on augmenting bookmarks and other browser functionalities with reminding features, marking pages, and linking pages together. Pages in such approaches are usually grouped without a valid categorization where tasks are most likely overlapped. Also, because the majority of these approaches are designed based on bookmarks or at least implementing the bookmark concept, some pages may become obsolete.

Other studies, on the other hand, focused on studying factors influencing the use of web browsing and navigation as well as classifying tasks in the web. For instance, web tasks can be categorized into multiple tasks (i.e. tasks can be performed in a single session) and multiple session tasks (i.e. tasks that span into multiple sessions).

Various approaches that support task management in web browsing have been presented in the literature [2]. However, several issues in such approaches are still controversial, including the way of presenting and grouping tasks as well as properties and functionalities that should be provided in these approaches. Furthermore, literature at present lacks of design guidelines for designing and developing such approaches. In an effort to investigate the effect of incorporating task management in the web, this paper presents a comparative study between TaskBar, a tool that works with Internet Explorer to help manage pending tasks in the web and the status quo web browser. The paper also presents a set of empirically derived guidelines for designing task management approaches for web browsers.

The rest of the paper is organized as follows: related work on task management in the web is reviewed first. Then the design and implementation of TaskBar is described. The methodology adopted to investigate the effect of incorporating task management particularly TaskBar in web browsing is reported next. Then the analysis of data obtained from the study is presented and is followed by a discussion of the findings and implications of the study and suggestions for future work directions.

2. RELEVANT WORK

With the vast growth of web pages in the World Wide Web and their ubiquitous use, returning to previously seen web pages has become one of the main activities in the web. For instance, Tauscher and Greenberg pointed out that 58% of the visited web pages have been visited before [3]. Cockburn and McKenzie carried out a similar study after three years and found out that the rate of web revisit was increased to 81% [4]. Therefore, a vast number of studies have been devoted to investigating how web browsers can incorporate revisiting features. Several studies [5-9] focused on augmenting browsers' navigation components such as back buttons, bookmarks, and history to support web revisit better. Also, various web revisit approaches employing graphical representation and visualization techniques were proposed in the literature [10-14].

Task management has also become one of the highly required features in the web. Users usually use various techniques to keep information for later use while browsing, such as email, saving pages, bookmarks, printing, and writing notes [1, 15, 16]. However, with such methods, valuable information may get lost or become obsolete.

A limited number of studies have been carried out to investigate suitable solutions for such issues. Jones et al. [1], for instance, developed a simple prototype called "Add Favorite 2," which provides the same functionalities of bookmarks with the ability to add descriptions about the web page in addition to the option of sending the web page by email.

Furthermore MACKAY, KELLAR and WATTERS in [17] developed an add-in tool that works together with a web browser, called Landmark, to help users re-find information in a web page. In this tool, users can mark information on a web page and return back to this information later. An experiment was carried out to investigate the effectiveness of Landmark in re-finding information. The results demonstrated a reduced time for re-finding information using Landmark when compared to typical methods. However, these approaches simply provide reminding features only and do not provide any functionality for dealing with multiple tasks.

Various studies were carried out to investigate how task management can be incorporated into web browsing. For instance, Melanie, Carolyn and Michael conducted a study to understand the factors influencing the usage of web browsing and navigation [18, 19]. It was found that web usage can be categorized into several tasks: fact finding, information gathering, browsing and transactions. It was also found that various factors influence the use of the web, such as task type, session, and individual differences.

Recently, novel and more advanced approaches for supporting web browsers with task management were proposed in the literature. For instance, Natalie and Kari-Jouko developed an approach for supporting task management in the web browser [20]. They introduced a workspace that can be opened alongside the browser presenting a collection of URLs of interest based on short-term usage. Pages marked as tasks are presented in thumbnails in the workspace. One of the limitations of this approach is that users were required to drag and drop URLs manually into the workspace. Also, task management functionalities other than drag–and-drop were not implemented. However, the results demonstrated an overall users' satisfaction.

Hupp and Miller, on the other hand, proposed a different type of web task management approach called Smart Bookmark [21]. It only records users' transactions in the web, for example flight booking, and enables replaying the actions of each transaction later. Bookmarked actions can be displayed here graphically, textually, and by using screen shots. The accuracy of extracting and recording actions in website was evaluated using twenty-five well-known websites. The results demonstrated a low correct extraction rate of actions (i.e. more than half of the websites). Furthermore, it only focused on one type of browsing activities, while others were neglected.

Morris, Meredith and Venolia also conducted a field study to investigate users' search habits in the web and found that search queries most likely span into multiple sessions and for long times [22]. Therefore, they proposed a system called SearchBar, which supports the management of search queries in an interrelated manner. Users could create a new topic and insert all relevant search queries in this topic. Each page could be marked to show special relevance. Also, a summary of each topic could be presented by clicking on the topic title. This summary shows several types of information such as user notes and special related pages. SearchBar was evaluated in terms of usability and the results demonstrated that it was easy to use and users used it extensively. However, it only supports search queries and neglects other browsing activities.

Web tasks can broadly be categorized in the literature into multiple tasks (MT), which are the tasks performed during a session, and multiple session tasks (MST), which are tasks that span multiple sessions. MacKay and Watters carried out a diary study and field study to understand how users perform MSTs [23]. They found users mostly performed similar actions in such tasks, such as opening new windows, searching, opening bookmarks, and using history. The results of these studies also help in the classification of tasks. More specifically, web tasks were classified into eight tasks which are school work, general topic search, research, travel, projects, actions, shopping, and status checking. Three main factors that should be taken into account when incorporating task management in browser were highlighted here. These factors were reminding features, tabbed browsing features, and managing tasks during sessions.

One year later, they developed three prototypes based on these guidelines [23]. These prototypes were similar in terms of easy access to multi-session tasks and in the way they were presented in the browser. The main difference among them is the functionality. For instance, the first prototype was the simplest, which provided creation of new tasks as well as stopping and resuming saved tasks. Beside these functionalities, the second prototype enabled the addition of web pages in active tasks. On the other hand, the third prototype consisted of additional four features. These features were specifying completion date for tasks, saving pages for later use, displaying pages according to the time viewed, and the ability to deal with them after the completion of tasks. An experiment was conducted to evaluate the usability of each prototype and the results demonstrated that the first and second prototypes did not significantly enhance users' browsing behavior, whereas the third prototype significantly reduced the usage of browsing-supporting tools such as bookmarks and history [23].

Wang and Chang [2] developed an approach called Multitasking Bar (MB) that worked together with Firefox to support browsing with multitasking management features. Tasks presented in MB as tabs with their name and status in the title of each tab. Pages related to each task also presented in tabs which could be only presented when the task was selected. Four attributes could be set when creating new tasks: name, status, end date, and active time. The results of the field study demonstrated that MB helped users complete tasks more effectively than traditional browsers. However, presenting tasks and their related web pages using tabs can be confusing and difficult to track, especially when the number of pages and tasks is high.

Recently, many efforts have been made to develop tools that better support task management in web browsing [17, 21, 23, 24]. However, the literature at present lacks guidelines for developing such tools and employing task management features in web browsers. To fill this gap in the literature, this paper presents an experimental work to define a set of guidelines for better employment of task management in web browsing.

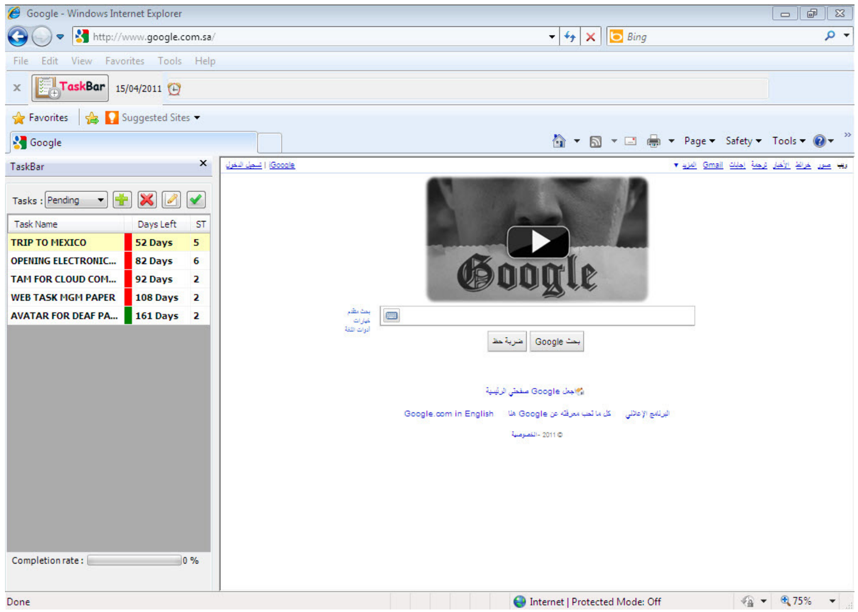

FIGURE 1: TaskBar Running the Browser with Pending Tasks Presented.

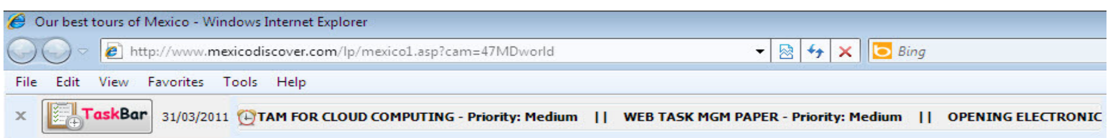

FIGURE 2: Pending Tasks Reminder Bar.

3. TASKBAR

To achieve the aims of this study, an experimental task manager was developed, called TaskBar, which works together with MS Internet Explorer. TaskBar enables users to manage ongoing (pending) tasks while browsing the web. The guidelines derived by MacKay and Watters for incorporating multi-session tasks in the browser were considered when developing TaskBar [24]. Furthermore, tasks here have the same four attributes (i.e. name, status, end date, and active time) of the Multitasking Bar developed by Wang and Chang [2] in addition to priority and notes. Unlike Multitasking Bar, tasks are presented in TaskBar in a list to reduce errors and confusion that could occur from a tabbed tasks view. It can also provide users with important features of task management such as reminding and priorities.

Previous tools allowed only the grouping of pages within multiple sessions but no valid classification between pages was presented. Archiving of completed tasks was also supported. The tool plays a to-do list in the browser and helps users decide which tasks should be dealt with first and which of them should be postponed.

One of the most important features of TaskBar is providing users with full structure of task management. Similar tools in the literature only allowed users to group related web pages in one tab as a task. However, such a method can overload users with tasks and hence cause difficulties in accomplishing and recalling tasks. On the other hand, tasks in TaskBar were categorized into main tasks, subtasks and web pages. It runs automatically with IE (as an IE Add-on) in two sidebars connected to the main window and the tool bar (see Figure 1). The latter is used for pending tasks reminding (see Figure 2). Main tasks were categorized into three types based on their completion date: pending, overdue, and completed tasks.

FIGURE 3: TaskBar with All Tasks Presented.

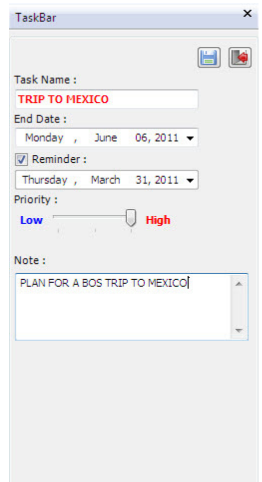

FIGURE 4: New Task Creation Form.

TaskBar automatically presents pending tasks in the sidebar when it is started and shows tasks reminder in the toolbar. The priority of tasks is encoded into colors (red for high, green for medium, and blue for low priority), number of days before the tasks are due, and number of sub tasks presented with each task. These types of tasks were selected to be presented only in the side bar to reduce the complexity of presentation, especially with the small area used for TaskBar and because of the frequent use of these types of tasks. Other types of tasks (i.e. overdue and completed tasks) can be displayed by selecting them from the dropdown menu (see Figure 3). A new task can be added in TaskBar by clicking on new task button where a new window will be presented that enables users to enter the task name, end date, priority, and notes (see Figure 4). A reminder can also be set to a task by checking the reminder field and setting the date on which the reminder should be started. Such tasks start moving in the tool bar (Figure 2) with their priority starting from the reminder date until it was disabled or the task was marked as completed. Tasks shown in the reminder bar are ordered based on priority and due date.

Moreover, a completion rate of each task is presented below the task list when it is selected. This completion rate is the percentage of subtasks completed under a main task. A task can be marked as completed by clicking on the "mark this task as complete" button when it is selected. It also can be edited using a similar window to (i.e. Figure 4) creating new task.

Subtasks under each task can be presented by double-clicking on the name of the main task. They are displayed in a separate window in a similar way to main tasks. However, the task name and number of days in which the task should be due only are presented here. The completion rate of subtasks is not presented because of the difficulties of calculating it with these types of tasks. In a similar way to main tasks, subtasks can also be deleted, edited, and marked as completed by clicking on the appropriate button after selecting the required subtask. The only difference in editing subtasks is the possibility of moving a subtask from a main task to another. This can be done by selecting a main task from the dropdown list in the editing form and then clicking on the save button. This feature gives users the flexibility of moving tasks to the appropriate category, especially with the large amount of information in the web that makes classification of information difficult.

Web pages can be added to subtasks in TaskBar by opening the page tab, double-clicking on the required subtask, and then selecting "add this page." The title and URL of the selected web page are written automatically in the appropriate fields (see Figure 6). A short note can be added to the web page and the title of the web page can be changed according to users' objectives. Many web pages can be added under a subtask where users can delete and edit them in a similar way to tasks. Web pages can be opened any time by double-clicking on their titles in the list (see Figure 5).

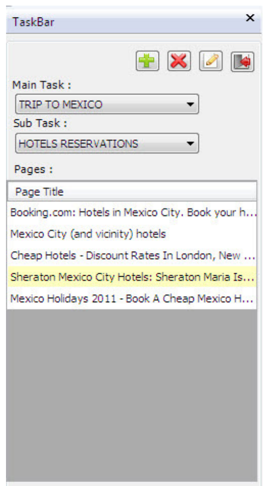

FIGURE 5: TaskBar with Pages Presented.

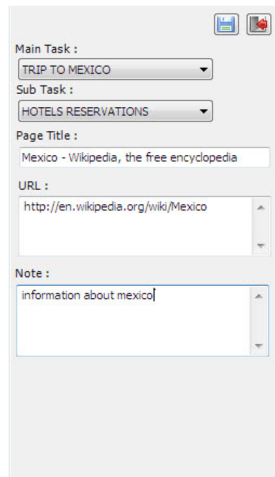

FIGURE 6: Form of Adding Pages to Tasks.

4. METHODOLOGY

The methodology used by Morris et al. in [22] was adopted here with some modifications to suit our aims. An experiment was carried out which consisted of two sessions scheduled one week apart; each session lasted half an hour. Two groups of thirty subjects each were involved in this experimental study. The controlled group used Internet Explorer without any tools for supporting task management (from now on it will be referred to as IE) to perform experimental tasks. The experimental group also used Internet Explorer but with the TaskBar. The experimental group was given a fifteen-minute demonstration about TaskBar prior to the start of the first session.

At the beginning of the first session, subjects were told that they would be asked to perform two main tasks and might be asked to accomplish some tasks instantly while performing the main tasks. These tasks were performed several times in a pilot study prior the experiment to estimate the time needed to complete each task. The minimum time required to complete all tasks was fifty minutes, meaning subjects most likely would not finish all tasks in one session.

Main tasks were also adopted from [22]and modified to suit the study aims. For instance, subjects were asked in the first task to book a flight to Sao Paulo, Brazil on given depart and return dates and find the best two offers in terms of cost and flight duration. They were also asked to write the airline name, route, number of stops, price, and duration in the answer sheet provided to them. Furthermore, subjects were asked in the second task to prepare a report about cloud computing. Parts of the report were already prepared and they were asked to complete the missing parts.

After ten minutes from the start of the session, subjects were asked to stop working in the main tasks and carry out two tasks instantly in a seven-minute interval. First, they were asked to find three pizza restaurants in Chicago. Restaurant name, phone number, and address were required to be written in the answer sheet. Second, they were asked to find two offers for digital cameras with prices less than $75. Camera brand name, price, resolution, color, and main features were required in the answer sheet.

At the end of this session, subjects were asked to fill a questionnaire soliciting demographic information, information about browsing habits, and information about experimental tasks. Files, including bookmarks, history, and TaskBar files in all computers were saved to be used in the second session. However, answer sheets were collected from subjects before leaving.

In the second session, the setup was the same as in the first session and all files remained as they were left by subjects. At the beginning of this session, subjects were asked to report on their progress of tasks in the first session by writing the task name (i.e. description), deadline, and status (i.e. completed or not). Ten minutes later, they were asked to stop working on the report and all reports were collected.

The subjects were then reminded about the tasks requested in the first session and were asked to perform four new tasks. These tasks were required to be performed in the same order at five-minute intervals. In each task, subjects were asked to find information related to the tasks in the first session. For instance, subjects in were asked in the first task to find the price and airline name of the cheapest offer for the Sao Paulo trip. In the second task, they were also asked to find three applications of cloud computing. The name and phone number of one of the pizza restaurant found in the first session were required in the third task. Finally, the brand name and price of the cheapest digital camera found in the first session were required in the fourth task. At the end of the session, subjects were asked to complete a questionnaire soliciting information about experimental tasks and overall satisfaction.

Measure	Items	Frequency	Percent
Gender	Male	60	100%
Education	Bachelor	44	73.3%
	Master	14	23.3%
	PhD	2	3.3%
Browser Used	IE	60	100%
Daily usage of the internet	1-3	8	13.3%
	4-6	25	41.7%
	7-9	21	35%
	more	6	10%
Ways of dealing with pending tasks	Bookmarks	38	63.3%
	Print pages	15	25%
	Other	7	11.7%

TABLE 1: Descriptive Statistics of Sample.

5. SAMPLE

Sixty employees working in Taibah University at Medina in the Kingdom of Saudi Arabia were voluntarily recruited to participate in this experimental study. All of them were males aged between 24 and 32 years old with an average age of 28 years. Table 1 shows descriptive statistics of the sample characteristics. It shows that all users used Internet Explorer as the main browser used for surfing the internet. Moreover, the majority (i.e. 41.7%) of the sample spent approximately four to six hours a day using the Internet. Also, the majority of the users used bookmarks as the main way of dealing with tasks in the web whereas none of them indicated that they emailed themselves as a reminder of pending tasks.

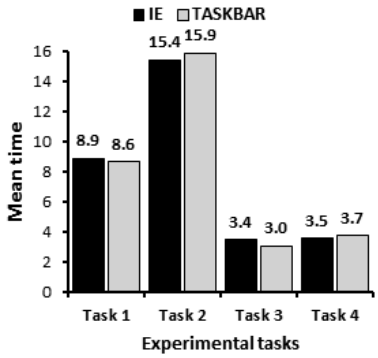

FIGURE 7: Mean Time Taken to Complete Tasks in Session 1.

Tasks	Results
Task 1	(t_{58}=0.79,p= 0.43)
Task 2	(t_{58}= - 1.29,p=0.2)
Task 3	(t_{58}= 1.52,p=0.13)
Task 4	(t_{58}= - 0.57,p=0.57)

TABLE 2: T-test Results of Time Taken to Complete Tasks in Session 1 at '0.05' Significance Level.

6. RESULTS

To compare between TaskBar and the control condition used for managing tasks in the web, the time taken to complete tasks and rate of task completion were calculated in the two sessions. Users' satisfaction was also taken into account when comparing the experimental conditions. The obtained data from the two sessions was analyzed independently.

Figure 7 shows the mean time taken to accomplish each task using TaskBar and IE in the first session. The total time taken was not calculated and used for comparison because the session length was set to be the same and therefore the total time of completion would almost be the same in the two conditions. Furthermore, the experimental tasks were designed to have different complexity levels, which helps explain the distribution of session time among tasks.

Figure 7 demonstrates that the time taken to the complete first session's tasks fluctuated between TaskBar and IE. For instance, the mean time taken to accomplish the first and third tasks was slightly higher in IE than in TaskBar. On the other hand, users required longer time to complete second and fourth tasks in TaskBar than using IE.

A T-test was applied here to investigate the significance of this difference. The results are shown in Table 2. The time taken to accomplish the experimental tasks was not significantly reduced in the two conditions in the first session.

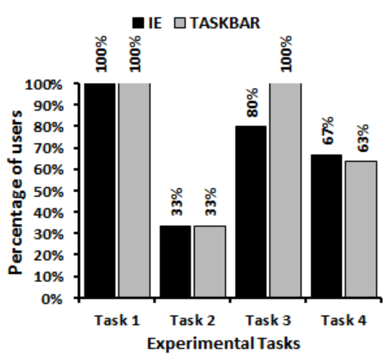

FIGURE 8: Rate of Tasks Completion in Session 1.

Tasks	Results
Task 3	(X^2=6.7, df=1,p=0.01)
Task 4	(X^2=0.73, df=1,p=0.78)

TABLE 3: Chi-square Results of Completion Rate in Session 1 at '0.05' Significance Level.

Figure 8 shows the percentage of users who successfully completed each task using IE and TaskBar in the first sessions. The number of users who completed each task also fluctuated between IE and TaskBar. For instance, the same number of users completed the first and second tasks (30 and 10 users) in IE and TaskBar. However, only 80% of users completed the third task in IE whereas 100% completed it in TaskBar. On the contrary, the percentage of users who completed the fourth task was slightly higher (67%) in IE than in TaskBar (63%).

A Chi-square test was applied to the number of users who completed each experimental task to investigate the difference between the two conditions. The results are shown in Table 3. The number of users who completed task 3 in TaskBar was significantly higher than IE, while no significant difference in other tasks was shown.

TABLE 4: Mean Users' Response On Questionnaire Statements In Session 1.

Statements	Condition	
	IE	Taskbar
How easy was it to perform the required tasks?	2.67	3.90
It was easy to create a new task.	2.23	3.87
It was easy to set a deadline for tasks.	1.90	4.00
It was easy to name tasks.	2.53	4.17
It was easy to deal (sort, move, and delete) with tasks.	1.67	3.57
It was easy to priorities for tasks.	1.73	4.03
It was easy to set the task reminder.	1.83	4.03
It was easy to deal with multiple tasks.	2.27	3.83
It was easy to understand the progress of tasks.	1.83	3.93
In overall, what is your overall satisfaction with performing experimental tasks?	2.27	3.93
Overall mean of users' responses:	2.09	3.93

The obtained data from questionnaires distributed at the end of the session was also analyzed independently. Table 4 shows the mean users' responses to each statement for both experimental conditions. The mean was calculated because these questions were set to measure users' satisfaction and according to other research [25], Likert scale questionnaires can be analyzed quantitatively when they are combined in a single composite.

The mean users' response to TaskBar was higher in all statements than IE, although their performance during the session was almost the same. For instance, Table 4 shows the mean users' responses ranged from 3.57 to 4.17, whereas performance ranged from 1.67 to 2.67. A T-test was applied to the overall mean of users' responses to all statements to investigate the significance of the difference between IE and TaskBar. The results indicated that users were significantly more satisfied with TaskBar than IE ($t58=11.35$, $p< 0.01$).

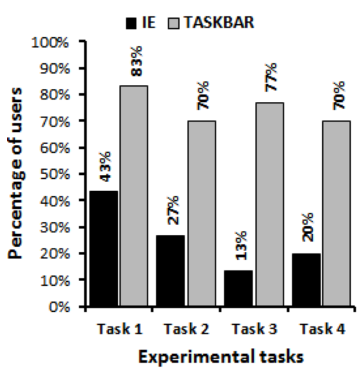

FIGURE 9: Rate of Users' Progress of Tasks Performed In Session 1.

Tasks	Results
Task 1	(X^2=10.34, df=1, p<0.01)
Task 2	(X^2=11.28, df=1, p<0.01)
Task 3	(X^2=24.31, df=1, p<0.01)
Task 2	(X^2=15.15, df=1, p<0.01)

TABLE 5 : Chi-square Results of Users' Progress of Tasks In Session 1 at 0.01 Significance Level.

As mentioned previously, users were required at the beginning of the second session to report on their progress in session 1. To measure the progress rate of tasks, users who answered all questions related to each task were considered to have reported on the task successfully. Figure 9 shows the percentage of users who successfully reported on each task. It shows the percentage of users who used TaskBar reported their tasks dramatically higher than those who used IE. For instance, only 43% of the users who used IE in the first session could report successfully on the first task whereas 83% of those who used TaskBar reported on it successfully. Furthermore, TaskBar helped 70%, 77%, and 70% of users successfully reporting tasks 2, 3, and 4 respectively. However, only 27%, 13%, and 20% of users who used IE completely reported on the same tasks. A Chi-square test was applied here to investigate the significance of this difference. The results are shown in Table 5. The number of users who reported successfully on each task using TaskBar was significantly higher than IE.

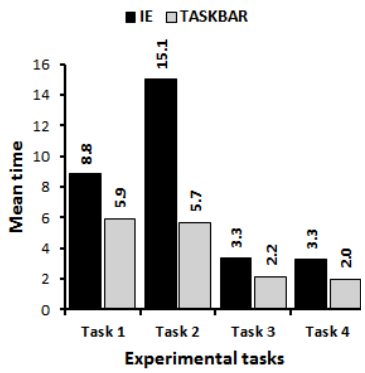

FIGURE 10: Mean Time Taken to Complete Tasks In Session 2.

Tasks	Results
Task 1	(t_{58}=13.19,p< 0.01)
Task 2	(t_{58}= 35.64,p<0.01)
Task 3	(t_{58}= 4.41,p<0.01)
Task 4	(t_{58}= - 5.37,p<0.01)

TABLE 6: T-test Results of Time Taken to Complete in Session 2 Tasks at 0.01 Significance Level.

The mean time taken to complete each task and the completion rate was also calculated in session 2. Figure 10 shows the mean time taken to accomplish each task using IE and TaskBar in session 2. Unlike session 1, it shows that time taken to accomplish all tasks in TaskBar was dramatically reduced when compared to IE. For instance, users required 8.8 minutes to complete the first task in IE, but only 5.9 minutes on average were required to complete it in TaskBar. Furthermore, users who used TaskBar successfully completed task 2 with only 38% (i.e. 5.7 minutes) of the time taken (i.e. 15.1 minutes) by those who used IE. Figure 10 also shows that users required 3.3 minutes to complete tasks 3 and 4 in IE whereas only 2.2 and 2 minutes were required to complete the two tasks in TaskBar, respectively. A T-test was applied to the time taken to complete each task to investigate the significance of this difference. The results are presented in Table 6. TaskBar helped users complete all tasks in session 2 with significantly reduced times when compared to IE.

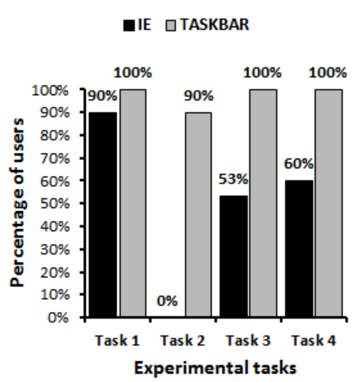

FIGURE 11: Rate of Tasks Completion in Session 2.

Tasks	Results
Task 1	(X^2=3.16, df=1, p>0.05)
Task 2	(X^2=60.00, df=1, p<0.01)
Task 3	(X^2=18.26, df=1, p<0.01)
Task 2	(X^2=15.00, df=1, p<0.01)

TABLE 7: Chi-square Results of Completion Rate in Session 2 at 0.01 Significance Level.

Figure 11 shows the percentage of users who completed each task in session 2 using TaskBar and IE. It shows the percentage of users who completed tasks using TaskBar was also noticeably higher than IE. For example, none of the users completed task 2 using IE while 90% of the users completed it using TaskBar. Moreover, all users completed tasks 3 and 4 in TaskBar whereas only 53% and 60% completed these tasks in IE, respectively. A Chi-square test was applied to these results to investigate the significance of the difference. The results demonstrated that the number of users who completed task 1 in TaskBar was not significantly higher than those who completed it using IE (see Table 7). Table 7, on the other hand, shows that the number of users who completed tasks 2, 3, and 4 in TaskBar was significantly higher than IE.

Statements	Condition	
	IE	Taskbar
How easy was it to perform the required tasks?	1.67	3.67
It was easy to create a new task.	1.93	4.13
It was easy to set a deadline for tasks.	1.47	3.93
It was easy to name tasks.	2.00	4.03
It was easy to deal (sort, move, and delete) with tasks.	1.70	3.47
It was easy to prioritize tasks.	1.33	4.03
It was easy to set the task reminder.	1.37	4.17
It was easy to deal with multiple tasks.	1.80	3.60
It was easy to identify task progress.	1.20	4.23
It was easy to remember pending tasks.	1.20	3.87
It was easy to identify important tasks.	1.17	4.03
It was easy to identify task deadlines.	1.23	4.03
It was easy to resume and carry on pending tasks.	1.37	3.90
What is your overall satisfaction in performing the experimental tasks?	1.67	3.80
Overall mean of users' response	1.5	3.9

TABLE 8: Mean Users' Responses to Questionnaire Statements in Session 2.

Users' responses regarding IE and TaskBar in this session were analyzed in a similar way to session 1. Table 8 shows the mean users' responses to each statement of the two conditions. It also shows that users' responses to TaskBar were higher than IE in all statements. The mean users' responses about TaskBar ranged from 3.47 to 4.23, while there responses about IE ranged from 1.17 to 2. A T-test was applied to the overall mean users' responses to investigate the significance of this difference. The results indicated that TaskBar was significantly more satisfying for users than IE ($t_{58}=18.39$, $p< 0.01$).

7. DISCUSSION

The results of the first experimental session demonstrated that TaskBar did not help users complete tasks with reduced time or with higher completion rates when compared with IE (i.e. controlled condition). On the contrary, users of IE sometimes outperform those using TaskBar in terms of time and completion rate. These results were expected because TaskBar was designed to facilitate dealing with pending tasks, which required users to perform extra work in the first session, such as entering tasks' names, setting priorities, and deadlines. On the other hand, users performing tasks using IE in the first session worked on tasks immediately without using a particular way of saving tasks for future use. Though, the results obtained from the questionnaire at the end of the session demonstrated that TaskBar was more satisfactory than IE.

The potential of TaskBar clearly appeared in the second session. Some users also used bookmarks to save web pages so they could return to them the next session, where others used notepad or other word-processing software to save page content or links. For instance, the majority of users who performed experimental tasks without using a task manager in the first session could not report on their progress at the beginning of the second session. This is because of the difficulties faced in finding required pages in bookmarks and related files. TaskBar, on the other hand, helped increased (nearly double) the percentage of users who reported successfully on their progress of the first session.

The majority of users who used the control condition repeated the tasks required in the first session to complete the tasks of the second session. Consequently, the results showed that the time taken to complete tasks using TaskBar was dramatically reduced when compared to the control condition. Storing task information in the TaskBar helped most of the users complete all the tasks. More specifically, almost all users (except 3 who could not complete task 2) finished all tasks within the time specified for the session. The results obtained from questionnaires at of this session confirmed that using a task manager in the web, particularly TaskBar, was more satisfactory than dealing with tasks without using any assistant tools. For instance, the majority of users found that setting task deadlines, priorities, and dealing with multiple tasks was easier in TaskBar but not in the controlled condition.

8. GUIDELINES

These results can be interpreted into various design guidelines for incorporating task management features in web browsing. First, task management approaches in web browsers should be developed independently with browsers' navigation functionalities, particularly bookmarks. Various studies in the literature demonstrated bookmarks are not used by the majority of web users and bookmarked pages usually become obsolete.

Second, pages should not be marked or dragged to a particular area of the screen without being classified into tasks. Otherwise, various tasks will overlap or be scattered in different places. Tracking task progress will become difficult and valuable information may get lost.

Third, the way to present tasks plays an important role in facilitating dealing with multiple tasks. A tabbed view is one of the ways used for presenting tasks and their web pages. Many limitations were observed using tabbing for presenting tasks in web browsers. Users most likely become overwhelmed with the web pages presented, especially with large tasks. Also, the whole area used for presenting tasks in such a way may be occupied with a limited number of tasks and important task information may not be displayed due to the limited space of tabs. For example, a task containing a large number of subtasks can easily occupy the whole area for presenting tasks. The list view used in TaskBar (see Figure 1 and Figure 3) helped present multiple tasks with important information such as due date and priority without overloading users with pages included in each task because they could be displayed by expanding the required task.

Fourth, presenting all tasks can also overload users with unimportant information and take the place of more important information, especially with the limited presentation area. Tasks with high priority and approaching their due dates should have the priority of presentation in the main view, while unimportant, completed, low priority, and distant due date tasks may be presented in a separate view and displayed when needed.

Fifth, tasks must have various properties to facilitate dealing with multiple tasks. For example, task name, due date, priorities, and notes are some of the important attributes of web tasks that can be considered when designing web task manager approaches.

Sixth, as suggested by Mackay and Watter [24], a reminder is one of the main features that must be taken into account when designing a web task manager. However, important tasks may also be forgotten with a normal task reminder, especially when a large number of tasks exists. To avoid missing important tasks, users must be aware of such tasks continually. Therefore, a nonstop reminder such as the one implemented in TaskBar (see Figure 2) should be taken into account when developing a task manager for web browsers.

9. CONCLUSION

This paper presented the design and evaluation of an approach called TaskBar, developed to improve web browsing by incorporating task management features in web browsers. TaskBar works together with Microsoft Internet Explorer and enabled dealing with selected web pages as tasks by providing several functionalities of task management such as reminders, and setting deadlines and priorities. It differs from currently available approaches mainly in two aspects: the way it presents tasks and the impartiality from web browser navigation features (i.e. bookmarks and history). A two-session controlled experiment was carried out to investigate the effect of employing task management features in web browsers. In this experiment, users' performance in TaskBar and a status quo web browser (Internet Explorer) were compared in terms of task completion time, completion rate, and users' satisfaction. The results showed that incorporating task management features in web browsing, particularly TaskBar, helped users return to and accomplish pending tasks in the web with significantly improved accomplishment time and higher completion rate when compared with a typical web browser. The results also demonstrated a dramatically high rate of users' satisfaction with TaskBar. Finally, the paper presented a set of design guidelines for incorporating task management in web browsing which was derived based on these results.

10. ACKNOWLEDGEMENT

This research project was funded by Taibah University‘s Deanship of Research under Grant number 432/38. I would also like to thank Ali Alsaedi for his help and support in implementing the proposed tool.

11. REFERENCES

[1] W. Jones, H. Bruce, and S. Dumais, *How Do People Get Back to Information on the Web?: How Can They Do it Better?*: IOS PRESS, 2003.

[2] Q. Wang, and H. Chang, “Multitasking bar: prototype and evaluation of introducing the task concept into a browser,” in Proceedings of the 28th international conference on Human factors in computing systems, Atlanta, Georgia, USA, 2010, pp. 103-112.

[3] L. Tauscher, and S. Greenberg, “How people revisit web pages: empirical findings and implications for the design of history systems,” *International Journal of Human Computer Studies,* vol. 47, no. 1, pp. 97-138, 1997.

[4] A. Cockburn, and B. McKenzie, “What do web users do? An empirical analysis of web use,” *International Journal of Human-Computer Studies,* vol. 54, no. 6, pp. 903-922, 2001.

[5] S. Greenberg, and A. Cockburn, “Getting back to back: Alternate behaviors for a Web browser's back button,” in Proceedings of the 5th Annual Human Factors and the Web Conference, NIST, Gaithersburg, Maryland, USA, 1999.

[6] S. Kaasten, and S. Greenberg, “Integrating back, history and bookmarks in web browsers,” in CHI '01 extended abstracts on Human factors in computing systems, Seattle, Washington, 2001, pp. 379-380

[7] A. Cockburn, S. Greenberg, S. Jones, B. McKenzie, and M. Moyle, “Improving web page revisitation: Analysis, design and evaluation,” *IT & Society,* vol. 1, no. 3, pp. 159-183, 2003.

[8] S. Kaasten, A. Cockburn, S. Greenberg, B. Mckenzie, and M. Jasonsmith, “WebView: A graphical aid for revisiting Web pages,” in `Proceedings of the 1999 Computer Human

Interaction Specialist Interest Group of the Ergonomics Society of Australia (OzCHI'99), Wagga Wagga, Australia, 1999, pp. 15-22.

[9] D. Abrams, R. Baecker, and M. Chignell, "Information archiving with bookmarks: personal Web space construction and organization." pp. 41-48.

[10] S. Kaasten, S. Greenberg, and C. Edwards, “How people recognize previously seen Web pages from titles, URLs and thumbnails,” *People and Computers XVI: Memorable Yet Invisible: Proceedings of HCI 2002*, pp. 247-266, 2002.

[11] R. R. Hightower, L. T. Ring, J. I. Helfman, B. B. Bederson, and D. H. James, “Graphical multiscale Web histories: a study of padprints,” in Proceedings of the ninth ACM conference on Hypertext and hypermedia : links, objects, time and space---structure in

hypermedia systems: links, objects, time and space---structure in hypermedia systems, Pittsburgh, Pennsylvania, United States, 1998, pp. 58-65.

[12] R. R. Hightower, L. T. Ring, J. I. Helfman, B. B. Bederson, and J. D. Hollan, “PadPrints: graphical multiscale Web histories,” in Proceedings of the 11th annual ACM symposium on User interface software and technology, San Francisco, California, United States, 1998, pp. 121-122.

[13] C. Sřensen, D. Macklin, and T. Beaumont, “Navigating the World Wide Web: bookmark maintenance architectures,” *Interacting with computers,* vol. 13, no. 3, pp. 375-400, 2001.

[14] H. Obendorf, H. Weinreich, E. Herder, and M. Mayer, “Web page revisitation revisited: implications of a long-term click-stream study of browser usage,” in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, San Jose, California, USA, 2007, pp. 597-606.

[15] R. Boardman, and M. A. Sasse, “"Stuff goes into the computer and doesn't come out": a cross-tool study of personal information management,” in Proceedings of the SIGCHI conference on Human factors in computing systems, Vienna, Austria, 2004, pp. 583-590.

[16] W. Jones, H. Bruce, and S. Dumais, “Keeping found things found on the web,” in Proceedings of the tenth international conference on Information and knowledge management, Atlanta, Georgia, USA, 2001, pp. 119-126.

[17] B. MacKay, M. Kellar, and C. Watters, “An evaluation of landmarks for re-finding information on the web,” in CHI '05 extended abstracts on Human factors in computing systems, Portland, OR, USA, 2005, pp. 1609-1612.

[18] M. Kellar, C. Watters, and M. Shepherd, “The impact of task on the usage of web browser navigation mechanisms,” in Proceedings of Graphics Interface 2006, Quebec, Canada, 2006.

[19] M. Kellar, C. Watters, and M. Shepherd, “A field study characterizing Web-based information-seeking task,” *Journal of the American Society for Information Science and Technology,* vol. 58, no. 7, pp. 999-1018, 2007.

[20] N. Jhaveri, and K.-J. Räihä, “The advantages of a cross-session web workspace,” in CHI '05 extended abstracts on Human factors in computing systems, Portland, OR, USA, 2005, pp. 1949-1952.

[21] D. Hupp, and R. C. Miller, “Smart bookmarks: automatic retroactive macro recording on the web,” in Proceedings of the 20th annual ACM symposium on User interface software and technology, Newport, Rhode Island, USA, 2007, pp. 81-90.

[22] D. Morris, M. R. Morris, and G. Venolia, “SearchBar: a search-centric web history for task resumption and information re-finding,” in Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems, Florence, Italy, 2008, pp. 1207-1216.

[23] B. MacKay, and C. Watters, “Building support for multi-session tasks,” in CHI '09 Extended Abstracts on Human Factors in Computing Systems, Boston, MA, USA, 2009, pp. 4273-4278.

[24] B. MacKay, and C. Watters, “Exploring multi-session web tasks,” in Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems, Florence, Italy, 2008, pp. 1187-1196.

[25] J. Harry N. Boone, and D. A. Boone, “Analyzing Likert Data,” *Journal of Extension,* vol. 50, no. 2, 2012.

Review on Sorting Algorithms A Comparative Study

Khalid Suleiman Al-Kharabsheh khalidkh@bau.edu.jo
Aqaba College , Balqa Applied University
Aqaba, Jordan

Ibrahim Mahmoud AlTurani Traini111@bau.edu.jo
Aqaba College , Balqa Applied University
Aqaba, Jordan

Abdallah Mahmoud Ibrahim AlTurani Traini2001@yahoo.com
IT college,Jordan University of Science and Technology
Irbid,Jordan

Nabeel Imhammed Zanoon Dr.nabeel@bau.edu.jo
Aqaba College , Balqa Applied University
Aqaba, Jordan

Abstract

There are many popular problems in different practical fields of computer sciences, database applications, Networks and Artificial intelligence. One of these basic operations and problems is sorting algorithm; the sorting problem has attracted a great deal of research. A lot of sorting algorithms has been developed to enhance the performance in terms of computational complexity. there are several factors that must be taken in consideration; time complexity, stability, memory space. Information growth rapidly in our world leads to increase developing sort algorithms.a stable sorting algorithms maintain the relative order of records with equal keys This paper makes a comparison between the Grouping Comparison Sort (GCS) and conventional algorithm such as Selection sort, Quick sort, Insertion sort , Merge sort and Bubble sort with respect execution time to show how this algorithm perform reduce execution time.

Keywords: Sort, Grouping Comparison Sort, Quick Sort, Merge Sort, Time Complexity.

1. INTRODUCTION

Sorting is a process of rearrangement a list of elements to the correct order since handling the elements in a certain order more efficient than handling randomize elements [1].Sorting and searching are among the most common programming processes, as an example take database applications if you want to maintain the information and ease of retrieval you must keep information in a sensible order, for example, alphabetical order, ascending/descending order and order according to names, ids, years, departments, etc.

Information growth rapidly in our world leads to increase developing sort algorithms. Developing sort algorithms through improved performance and decreasing complexity, it has attracted a great deal of research; because any effect of sorting algorithm enhancement of the current algorithms or product new algorithms that reflects to optimize other algorithms. Large number of algorithms developed to improve sorting like merge sort, bubble sort, insertion sort, quick sort ,selection sort and others, each of them has a different mechanism to reorder elements which increase the performance and efficiency of the practical applications and reduce time complexity of each one. When comparing between various sorting algorithms, there are several factors that must be taken in consideration; first of them is the time complexity, the time complexity of an algorithm determined the amount of time that can be taken by an algorithm to run [3][7][27]. This factor different from sorting algorithm to another according to the size of data that we want to reorder, some sorting algorithm inefficient and too slow. The time complexity of an algorithm is generally written in form big O(n) notation, where the O represents the complexity of the algorithm and a value n represent the number of elementary operations performed by the algorithm [8].The second factor is the stability[26], means; algorithm keeps elements with equal values in the same relative order in the output as they were in the input. [2][3][9]. Some sorting algorithms are stable by its nature such as insertion sort, merge sort, bubble sort, while some sorting algorithms are not, such as quick sort, any given sorting algorithm which is not stable can be modified to be stable [3]. The third factor is memory space, algorithm that used recursive techniques need more copies of sorting data that affect to memory space [3][9].Many previous researches have been suggested to enhance the sorting algorithm to maintain memory and improve efficiency. Most of these algorithms are used comparative operation between the oldest algorithm and the newest one to prove that.

2. PERFORMANCE IN AVERAGE CASE BETWEEN SORTING AIGORITHMS

the following studies are previous study on the same research which make a comparative between different type of sorting algorithms:

(Pooja Adhikari,2007) The performance of any computation depends upon the performance of sorting algorithms.Like all complicated problems, there are many solutions that can achieve the same results. This paper choose two of the sorting algorithms among them selection sort and shell sort and compares the various performance factor among them.

(Davide Pasetto Albert Akhriev,2011) In this paper we provide a qualitative and quantitative analysis of the performance of parallel sorting algorithms on modern multi-core hardware. We consider several general-purpose methods,which are widely regarded among the best algorithms available, with particular interest in sorting of database records and very large arrays (several gigabytes and more), whose size far exceed L2/L3 cache.

(ADITYA DEV MISHRA & DEEPAK GARG,2008)Many different sorting algorithms have been developed and improved to make sorting fast. As a measure of performance mainly the average number of operations or the average execution times of these algorithms have been investigated and compared. There is no one sorting method that is best for every situation. Some of the factors to be considered in choosing a sorting algorithm include the size of the list to be sorted, the programming effort, the number of words of main memory available, the size of disk or tape units, the extent to which the list is already ordered, and the distribution of values

This paper implemented of Selection sort, Quick sort, Insertion sort , Merge sort ,Bubble sort and GCS algorithms using C++ programming language, and measure the execution time of all programs with the same input data using the same computer. The built-in function (clock ()) in C++ is used to get the elapsed time of the implementing algorithms, execution time of a program is measured in milliseconds [6].The performances of GCS algorithm and a set of conventional sort algorithms are comparatively tested under average cases by using random test data from size 10000 to 30000. The result obtained is given in Table 1 to Table 6 for each Algorithm and the curves are shown in figure 1.

Selection sort

Selection sorts the simplest of sorting techniques. It's work very well for small files, also It's has a quite important application because each item is actually moved at most once [4]. It has O (n2) time complexity, making it inefficient on large lists. Selection sort has one advantage over other sort techniques[15][16]. Although it does many comparisons, it does the least amount of data moving. That means, if your data has small keys but large data area, then selection sorting may be the quickest.[8] .In Table 1 the execution time and number of elements as follow:

Number of elements	Running time (ms)
10000	2227
20000	5058
30000	8254

TABLE 1: Running Time for Selection Sort.

Insertion sort

Insertion sort is very similar to selection sort. It is a simple sorting algorithm that builds the final sorted list one item at a time[18]. It has O (n2) time complexity, It is much less efficient on large lists than more advanced algorithms such as quick sort, heap sort, or merge sort. However, insertion sort provides several advantages Simple implementation and, Efficient for small data sets [10][17] .In Table 2 the execution time and number of elements as follow:

Number of elements	Running time (ms)
10000	1605
20000	3678
30000	6125

TABLE 2: Running Time for Insertion Sort.

Merge sort

Merge sort is a divide and conquer algorithm .It's Divide the list into two approximately equal sub lists, Then Sort the sub lists recursively[19]. It has an O (n log n) Time complexity .merge sort is a stable sort, parallelizes better, and is more efficient at handling slow-to-access sequential media. Merge sort is often the best choice for sorting a linked list [11][20]. In Table 3 the execution time and number of elements as follow:

Number of elements	Running time (ms)
10000	728
20000	1509
30000	2272

TABLE 3: Running time for merge sort

Quick sort

In this sort an element called pivot is identified and that element is fixed in its place by moving all the elements less than that to its left and all the elements greater than that to its right. Since it partitions the element sequence into left, pivot and right it is referred as a sorting by partitioning. It's an O (n log n) Time complexity in average case[21][22]. In Table 4 the execution time and number of elements as follow:

Number of elements	Running time (ms)
10000	489
20000	1084
30000	1648

TABLE 4: Running Rime for Quick Sort.

Bubble sort

Bubble sort is a simple sorting algorithm that works by repeatedly; it's comparing each pair of adjacent items and swapping them if they are in the wrong order. This passing procedure is repeated until no swaps are required, indicating that the list is sorted [13][23]. It has a O (n2) Time complexity means that its efficiency decreases dramatically on lists of more than a small number of elements [12][24]. In Table 4 the execution time and number of elements as follow:

Number of elements	Running time (ms)
10000	1133
20000	3103
30000	5730

TABLE 5: Running Time for Bubble Sort.

Grouping Comparison sort

In this sort we divide the list of elements into groups; each group contains three elements that compare with the first element of next groups. Performance has been decreased by GCS algorithm, mainly if the input size more than 25000 elements that returned increasing number of comparison, the performance have been improved when size of input is less than 25000 elements. It has a time complexity O (n2) [14]. In Table 6 the execution time and number of elements as follow:

Number of elements	Running time (ms)
10000	1124
20000	3374
30000	6687

TABLE 6: Running Time for Comparison Sort.

3. COMPARATIVE STUDY AND DISCUSSION

All the six sorting algorithms (Selection Sort, Insertion sort, Merge sort, Quick sort, Bubble Sort and Comparison sort) were implemented in C++ programming languages and tested for the random sequence input of length 10000, 20000, 30000, All the six sorting algorithms were executed on machine Operating System having Intel(R) Core(TM) 2 Duo CPU E8400 @ 3.00 GHz (2 CPUs) and installed memory (RAM) 2038 MB. The Plot of length of input and CPU time taken (ms) is shown in figure 1. Result shows that for small input the performance for the six techniques is all most nearest, but for the large input Quick sort is the fastest and the selection sort the slowest. the grouping comparison sort for small input (10000) is the third sort and in the large input (30000) is the fifth sort in order between the six sorting algorithms.

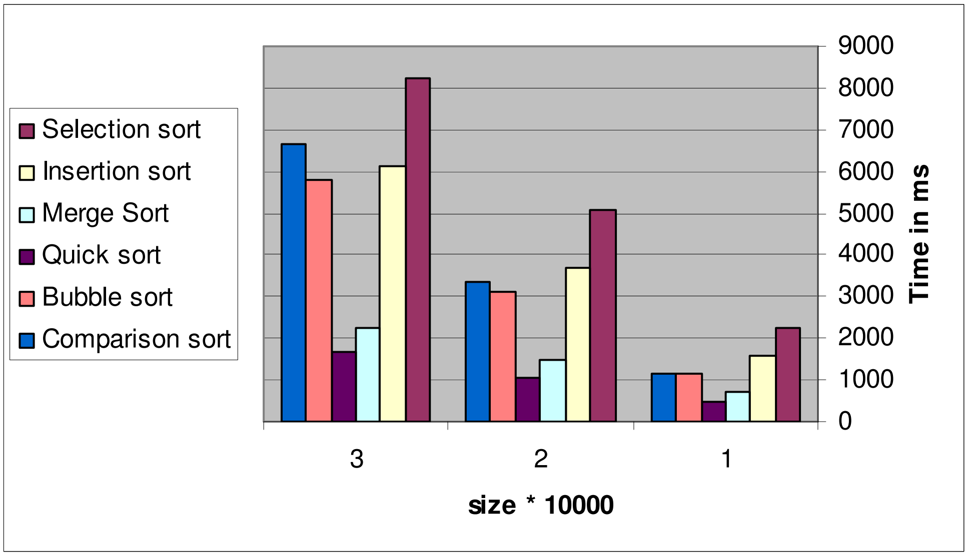

FIGURE 1 : Plot of Number of Input vs CPU.

3.1 Complexity Comparison between Typical sorting algorithms

The comparison of complexity between GCS and conventional sort algorithms are listed in table 7[5]. Table 6 determines the time complexity of new algorithm is equivalent to some conventional sort algorithms[25][28]. GCS gave an additional method to manipulate information.

Algorithm	Average case	Worst case
Selection sort	O (n2)	O (n2)
Insertion sort	O (n2)	O (n2)
Merge Sort	O (n log n)	O (n log n)
Quick sort	O (n log n)	O (n2)
Bubble sort	O (n2)	O (n2)
Comparison Sort	O (n2)	O (n2)

TABLE 7: Time Complexity of Typical Sorting Algorithms.

4. CONCLUSION AND FUTURE WORK

This paper discuss a comparison between the new suggested algorithm (GCS) and selection sort, Insertion sort, merge sort, quick sort and bubble sort. It analysis the performance of these algorithms for the same number of elements (10000, 20000, 30000). For small input the performance for the six techniques is all most nearest, but for the large input Quick sort is the fastest and the selection sort the slowest. Comparison sort in average and worst case have the same time complexity with selection, Insertion and bubble sort This research is initial step for future work; in the future we will improve our algorithms Grouping Comparison Sort algorithms (GCS) to optimize software's in searching method and retrieve data.

5. REFERENCES

[1] P.Adhikari, Review on Sorting Algorithms, "A comparative study on two sorting algorithm", Mississppi state university, 2007.

[2] T. Cormen, C.Leiserson, R. Rivest and C.Stein, *Introduction To Algorithms*, McGraw-Hill, Third Edition, 2009,pp.15-17.

[3] M. Goodrich and R. Tamassia, Data *Structures and Algorithms in Ja*va,John wiley & sons 4th edition, 2010,pp.241-243.

[4] R. Sedgewick and K. Wayne, *Algorithms*,Pearson Education, 4th Edition, 2011,pp.248-249.

[5] T. Cormen, C.Leiserson, R. Rivest and C.Stein,*Introduction to Algorithms*, McGraw Hill, 2001,pp.320-330.

[6] H. Deitel and P. Deitel, *C++ How to Program*, Prentice Hall, 2001,pp.150-170

[7] M. Sipser, *Introduction to the Theory of Computation*,Thomson,1996,pp.177-190.

[8] S. Jadoon , S.Solehria, S.Rehman and H.Jan.(2011,FEB). " Design and Analysis of Optimized Selection Sort Algorithm".11. (1),pp. 16-21. Available: http://www.ijens.org/IJECS%20Vol%2011%20Issue%2001.html.

[9] Aditya Dev Mishra & Deepak Garg.(2008,DEC). "Selection of Best Sorting Algorithm ", *International journal of intelligent information Processing*. II (II).pp. 363-368. Available: http://academia.edu/1976253/Selection_of_Best_Sorting_Algorithm.

[10] http://en.wikipedia.org/wiki/Insertion_sort

[11] http://en.wikipedia.org/wiki/Merge_sort

[12] http://en.wikipedia.org/wiki/Bubble_sort

[13] http://www.techopedia.com/definition/3757/bubble-sort

[14] I. trini, k. kharabsheh, A. trini, (2013,may)."Grouping Comparison Sort", *Australian Journal of Basic and Applied Sciences*.pp221-228.

[15] R. Sedgewick, *Algorithms in C*++, Addison–Wesley Longman,1998,pp 273–274.

[16] A. Levitin, *Introduction to the Design & Analysis of Algorithms*, Addison–Wesley Longman, 2007, pp 98–100.

[17] http://corewar.co.uk/assembly/insertion.htm

[18] T. Cormen, C.Leiserson, R. Rivest and C.Stein, *Introduction To Algorithms*, McGraw-Hill, Third Edition, 2009,pp.15-21

[19] Katajainen, Jyrki; Pasanen, Tomi; Teuhola, Jukka (1996,MAR). "Practical in-place mergesort". *Nordic Journal of Computing.* (3). pp. 27–40.

[20] Kronrod, M. A. (1969). "Optimal ordering algorithm without operational field". Soviet Mathematics - Doklady (10). pp. 744.

[21] T. Cormen, C.Leiserson, R. Rivest and C.Stein, *Introduction To Algorithms*, McGraw-Hill, Third Edition, 2009,pp.145-164.

[22] A. LaMarca and R. E. Ladner.(1997), "The Influence of Caches on the Performance of Sorting." Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms,. pp. 370–379.

[23] D.Knuth, The Art of Computer Programming,Addison-Wesley,Third Edition 1997,pp. 106–110 .

[24] http://www.sorting-algorithms.com/bubble-sort.

[25] T.Niemann, Sorting and Searching Algorithms:A Cookbook, 2008 , pp.11-14.

[26] H.Ahmed,H.Mahmoud, N.Alghreimil," A Stable Comparison-based Sorting Algorithm with worst Case complexity of O(n log n) Comparisons and O(n) Moves" ,*World Academy of Science, Engineering and Technology(22)*,pp.970-975.

[27] C.Cook , D.Kim. "Best sorting algorithm for nearly sorted lists*". Commun. ACM*, 23(11),pp.620-624.

[28] D.Garg," Selection O. Best Sorting Algorithm*", International Journal of Intelligent Information Processing* 2(2),pp.363-368.

,

INSTRUCTIONS TO CONTRIBUTORS

The *International Journal of Computer Science and Security (IJCSS)* is a refereed online journal which is a forum for publication of current research in computer science and computer security technologies. It considers any material dealing primarily with the technological aspects of computer science and computer security. The journal is targeted to be read by academics, scholars, advanced students, practitioners, and those seeking an update on current experience and future prospects in relation to all aspects computer science in general but specific to computer security themes. Subjects covered include: access control, computer security, cryptography, communications and data security, databases, electronic commerce, multimedia, bioinformatics, signal processing and image processing etc.

To build its International reputation, we are disseminating the publication information through Google Books, Google Scholar, Directory of Open Access Journals (DOAJ), Open J Gate, ScientificCommons, Docstoc and many more. Our International Editors are working on establishing ISI listing and a good impact factor for IJCSS.

The initial efforts helped to shape the editorial policy and to sharpen the focus of the journal. Started with Volume 7, 2013, IJCSS is appearing in more focused issues. Besides normal publications, IJCSS intend to organized special issues on more focused topics. Each special issue will have a designated editor (editors) – either member of the editorial board or another recognized specialist in the respective field.

We are open to contributions, proposals for any topic as well as for editors and reviewers. We understand that it is through the effort of volunteers that CSC Journals continues to grow and flourish.

IJCSS LIST OF TOPICS

The realm of International Journal of Computer Science and Security (IJCSS) extends, but not limited, to the following:

- Authentication and authorization models
- Computer Engineering
- Computer Networks
- Cryptography
- Databases
- Image processing
- Operating systems
- Programming languages
- Signal processing
- Theory
- Communications and data security
- Bioinformatics
- Computer graphics
- Computer security
- Data mining
- Electronic commerce
- Object Orientation
- Parallel and distributed processing
- Robotics
- Software engineering

CALL FOR PAPERS

Volume: 7 - **Issue: 5**

i. Submission Deadline : November 30, 2013 **ii. Author Notification:** December 25, 2013

iii. Issue Publication: December 31, 2013

CONTACT INFORMATION

Computer Science Journals Sdn BhD
B-5-8 Plaza Mont Kiara, Mont Kiara
50480, Kuala Lumpur, MALAYSIA

Phone: 006 03 6207 1607
006 03 2782 6991

Fax: 006 03 6207 1697

Email: cscpress@cscjournals.org

COMPUTER SCIENCE JOURNALS SDN BHD
B-5-8 PLAZA MONT KIARA
MONT KIARA
50480, KUALA LUMPUR
MALAYSIA

PHONE: 006 03 6207 1607
006 03 2782 6991

FAX: 006 03 6207 1697
EMAIL: cscpress@cscjournals.org