University of Peradeniya

Sri Lanka

Proceedings of the Second Engineering Students' Conference at Peradeniya (ESCaPe) 2013

Organized by

Department of Computer Engineering,

Faculty of Engineering, University of Peradeniya

19th June 2013

Organizing Committee

Chairperson

Dr. R. G. Ragel

Technical Programme Committee

Mr. S.D. Deegalla

Dr. S. D. Dewasurendra

Dr. D. Elkaduwe

Dr. R. Krishanthmohan

Mr. Z. Maraikar

Dr. R. G. Ragel

Dr. K. Samarakoon

Dr. M. Sandirigama

Logistics and Ceremonial Arrangements

Mr. U. Senanayake

Mr. P. Herath

Ms. N. Pathirana

Ms. R. Yatawatte

Cover Page and Logo Design

Mr. U. Senanayake

Contents

Session I

Session II

Session III

Posters

Message from the Dean, Engineering

I am indeed very happy to write this message to mark the 2nd Engineering Students' Conference at Peradeniya (ESCaPe'13) scheduled to be held on the 19th June 2013 at the Faculty of Engineering, University of Peradeniya.

The Faculty remembers that the department of Computer Engineering coined the idea of a students' symposium back in 2010. The objective of the symposium was to raise awareness among industrial representatives and academics from other departments, faculties and universities on research and associated academic activities of the students, which is a timely requirement. This also provides a valuable opportunity for the final year students to showcase their projects to a wider audience, and certainly promote a research culture among the students. What you witness today is our second attempt on materializing the above goals and reaching the objective in a wider manner.

I observe that the ESCaPe'13 includes a number of high quality papers, technical articles initiating from, and based around, the projects conducted by the final year students of the Computer Engineering Department. These projects cover a wide range of research topics from Single-chip Cloud Computing to low-end embedded devices, from Artificial Intelligence to low-level memory management, etc. Some of these projects were conducted in collaboration with foreign Universities and some others with the Industry. I believe you will agree with me that such diversity, depth and breadth of topics, together with the volume of the work reinforce the strength of the Department, which has gained momentum to move incredibly forward since its humble beginning.

I would like to take this opportunity to congratulate the staff at the Department of Computer Engineering who made this event a reality.

Prof (Eng.) Leelananda Rajapaksha
Dean, Faculty of Engineering
University of Peradeniya

Message from the Director, International Research Centre, University of Peradeniya

It gives me great pleasure to pen this message congratulating the Department of Computer Engineering on the organization of this students' conference. Dealing in a field that reaches new frontiers on a daily basis the Department of Computer Engineering has proven with time that they are indeed capable and have produced some brilliant academics and professionals. As the first fully fledged Computer Engineering Department to be established in the University system of Sri Lanka, the Department has built a reputation over the years for its quality of education.

The International Research Centre is always keen on supporting the Department in their activities and we look forward to their next great venture. It is indeed an achievement that once again the annual student symposium had been organized, a highlight in the university calendar, and I wish you continued success and all the very best in your future endeavours.

Dr Nanda Gunawardhana
Director
International Research Centre
University of Peradeniya

Message from the Head, Computer Engineering

It is with pride I write this message to welcome you all to the 2nd Engineering Students' Conference at Peradeniya or ESCaPe'13, to be held on the 19th June 2013 at the Faculty of Engineering, University of Peradeniya. The main objective of the conference is to disseminate the knowhow of the projects conducted at the department to the rest of the world, including the Sri Lankan industry and the other Universities producing computing related graduates.

The Department takes careful steps towards promoting research among Undergraduates. Most staff members have linkages with overseas Universities through which the Department was able to expose students to cutting edge technologies. Even such high quality work would be useless if the results are not presented to the right audience in a manner they would accept. Events such as ESCaPe'13 provide the opportunity for our students to organize, present and get feedback on their work from a panel of experts. I humbly request all of you to provide constructive criticism on the work presented today.

The papers you would witness today are based around the projects completed by the final year undergraduates. All the projects are supervised by the Department staff, in some cases with an industrial supervisor. The Department encourages the industry to have close collaborations with the Department by means of joint projects of this nature. Our experience with such projects suggests that such activities are mutually beneficial.

I would like to take this opportunity to thank the dedicated staff of the Department who made this event a reality. Further on behalf of the Department of Computer Engineering I thank our exclusive sponsor 99x Technologies.

Dr. D. Elkaduwe,
Head, Department of Computer Engineering,
University of Peradeniya

Keynote Speech: Future Internet and Implications of Wireless

by

Dr. Dharma P. Agrawal
Ohio Board of Regents Distinguished Professor and
Director, Center for Distributed and Mobile Computing
School of Computing Sciences and Informatics, University of Cincinnati, OH, 45221-0008
Tel:513-556-4756 E-mail:dpa@cs.uc.edu; web: http://www.cs.uc.edu/~dpa

Abstract

Internet is being used by a vast majority of population and its acceptance is anticipated to grow. This keynote first introduces various underlying functional components of a typical Internet. We also project various new services expected in the near future and increase in expected number of websites worldwide. A key component that cannot be effectively supported by the Internet is the last-mile connectivity in an efficient way. We consider potential use of wireless technology to provide such flexibility. It may be remembered that neither Internet nor wireless technologies are designed for multimedia traffic. This talk considers what kind of services would be desirable and what will be the corresponding implications on the design techniques and performance requirements. Security in future systems is going to be a major concern and ought to be addressed very carefully. Finally, a new type of wireless devices known as sensor networks are discussed and their potential use in many different disciplines are also explored.

About the Speaker

Dharma P. Agrawal is the Ohio Board of Regents Distinguished Professor and the founding director for the Center for Distributed and Mobile Computing in the School of Computing Sciences and Informatics. He has been a faculty member at the ECE Dept., Carnegie Mellon University (on sabbatical leave), N.C. State University, Raleigh and the Wayne State University. His current research interests include resource allocation in wireless mesh networks, query processing and secured communication in sensor networks, environmental monitoring using sensor networks, and effective traffic handling in integrated wireless networks. His recent contribution in the form of a co-authored introductory text book on *Wireless and Mobile Computing* has been widely accepted throughout the world and third edition has just been published. The book has been has been reprinted both in China and India and translated in to Korean and Chinese languages. His co-authored book on *Ad hoc and Sensor Networks, 2nd edition,* has been published in spring of 2011. A co-edited book entitled, *Encyclopedia on Ad Hoc and Ubiquitous Computing,* has been published by the World Scientific and co-authored books entitled *Wireless Sensor Networks: Deployment Alternatives and Analytical Modeling*, and *Innovative Approaches to Spectrum Selection, Sensing,* *On-Demand Medium Access in Heterogeneous Multihop Networks*, and *Sharing in Cognitive Radio Networks* have being published by Lambert Academic.

He is an editor for the *Journal of Parallel and Distributed Systems*, founding Editorial Board Member, I*nternational Journal on Distributed Sensor Networks*, *International Journal of Ad Hoc and Ubiquitous Computing (IJAHUC), International Journal of Ad Hoc & Sensor Wireless Networks* and *the Journal of Information Assurance and Security (JIAS)*. He has served as an editor of the IEEE *Computer magazine*, and the *IEEE Transactions on Computers and* the *International Journal of High Speed Computing*. He has been the Program Chair and General Chair for numerous international conferences and meetings. He has received numerous certificates from the IEEE Computer Society. He was awarded a *Third Millennium Medal*, by the IEEE for his outstanding contributions. He has delivered keynote speech at 25 different international conferences. He has published over 592 papers, given 32 different tutorials and extensive training courses in various conferences in USA, and numerous institutions in Taiwan, Korea, Jordan, UAE, Malaysia, and India in the areas of Ad hoc and Sensor Networks and Mesh Networks, including security issues. He has graduated **62 PhDs and 52 MS students**. He has been named as an **ISI Highly Cited Researcher**, is a Fellow of the **IEEE,** the **ACM**, the **AAAS** and the **World Innovation Foundation,** and a recent recipient of **2008 IEEE CS Harry Goode Award**. In June 2011, he was selected as the **best Mentor for Doctoral Students** at the University of Cincinnati. Recently, he has been inducted as a **charter fellow of the National Academy of Inventers.** He has also been elected a **Fellow of the IACSIT** (International Association of Computer Science and Information Technology), 2013.

Online Sinhala News Aggregator with Automated News Article Classification

M.N.T. Perera, H.M.D.K. Mallikarachchi, T.M.G. Tennakoon and D.S. Deegalla

Department of Computer Engineering, Faculty of Engineering,
University of Peradeniya, Sri Lanka

Abstract— **There are many Sinhala news sites with manual news article classification. However, none of such sites or Sinhala news aggregators employs automated classification. News aggregation allows users to read news from several sites in one location, while automated classification reduces exhaustive manual work. The purpose of this study is to develop a Sinhala news reader tool with automated news aggregation and classification. In this paper, we propose a methodology to pre-process Sinhala text which combines several previous research works on Sinhala language processing and automatic news classification based on the Multinomial Naïve Bayes (MNB) algorithm. The classifier model proved to be a success with an average accuracy of 88% on randomly selected test samples of 500 news articles.**

I. INTRODUCTION

According to latest statistics, digital news media (online and mobile news) indicate a significant growth in capturing audience compared to other conventional news sources, over the past few years [1]. This also applies to Sri Lankan context, considering the fact that a large number of Sinhala news sites available online. This trend has caused continuously increasing amount of news articles in English, Sinhala as well as the other languages being added to the World Wide Web (WWW) every day, subsequently paving pathway to many research areas for automated techniques of aggregating and organizing news information on the web. Such aggregation of news provides the convenience of viewing news from several sources at one place and classified news allows category-specific advertising, market research, trend research and political campaigning.

In Sri Lanka, even though Sinhala news aggregation from several sources have been paid attention to, no researches related to automated Sinhala news classification are found during the literature, presumably due to the complicated nature of Sinhala language processing. Therefore, there is a need for an automated Sinhala news classification system, which will enable the organization of Sinhala news information on the web, and will support the growth of online news trend in Sri Lanka.

News article categorization, i.e., text categorization can be done using both supervised and unsupervised learning techniques. If categorized training data is available, supervised learning can be used and if not unsupervised learning can be used.

The most commonly used text clustering techniques include Hierarchical Agglomerative Clustering (HAC) [5], K-means and Expectation Maximization [5] whereas Multinomial Naïve Bayes (MNB), K-Nearest Neighbour (KNN), Support Vector Machine (SVM) and rule sets (RIPPER) [4] are frequently used in text classification.

In the following section, there is comprehensive description of related work on Sinhala language processing and news classification. In the methodology section, we describe the architectural details of the tool we developed, and how several previous works on Sinhala language processing were combined together to pre-process news articles. Next section provides the implementation details. In Section V, results of the empirical study on comparison of two text classification algorithms are presented. Finally, concluding remarks and potential future work are given in Section VI.

II. RELATED WORK

Even though, languages like English have extensive work done in the language processing field, with well-defined algorithms, there are some issues that arise when it comes to automated Sinhala text processing. Some of the issues are: lack of commonly available tools or algorithms in the Sinhala processing and lack of standard representation of characters.

There are a number of projects completed by the University of Colombo School of Computing (UCSC), in implementing a stemmer algorithm [10], suffixes list, etc., whereas in the paper by Hettige [3], there is a description on a computational model for Sinhala grammar that would ease the grammatical analysis of Sinhala text. In the field of Sinhala text classification, Meedeniya [2] has presented a research on document indexing and applying clustering techniques for document categorization.

There are also many researches done based models developed and experimented with, in different languages for news classification. One approach, as described in a paper by Darko et al. [6], uses a news article classification model trained with K-Nearest Neighbour. Here the documents are pre-processed by indexing articles using a set of predefined keywords. Another model described by Tenenboim et al. [7], uses hierarchical news ontology based approach, using the keywords defined by the International Press Telecommunications Council (IPTC) news codes. This approach facilitates multiple category assignment.

While categorization has been the primary focus on the above methodologies, the paper by Chan et al. [9] has concentrated on developing a news classification model that provides personalized news to users using the SVM. The model classifies articles into binary categories as positive and negative to user's preferences. In another paper [8], the researchers have described an intelligent news agent with personalization, where the classification is done using the user's long-term and short-term interests and feedback to each article.

The most widely available tools that provide automated classification for aggregated English language news are Google and Yahoo. In Google, the news articles are clustered based on their keywords into a topical keywords centroid, that develop concept relationships between articles and keywords [13]. Whereas in Yahoo, a semi-automatic hierarchy based technique is used for classification of aggregated articles [12]. In Sri Lanka, even though there are sites with news aggregators available (such as Sinhala24News [11]), none of them uses automated classification.

III. Methodology

A. Data Selection

In data selection, labelled news articles were acquired from few selected news sites with manual classification. The news articles were taken from Lankapuwath [14], Sri Lanka Mirror [15] and Divaina [16]. Arts & Culture, Business & Economics, Politics, Defence & Law and Sports were selected as news categories by considering the availability of the news articles. For each class, 1500 articles were selected for better accuracy and reliability.

B. Pre-processing

In pre-processing stage, unnecessary tokens such as stopwords were removed from the news articles and then the words were stemmed. The main issue was acquiring a Sinhala stopword list and a stemming algorithm. We have used the stopword list and the stemming algorithm developed by UCSC [18].

C. Data Transformation

Documents were indexed using the bag of words method [17]. The distinct list of stemmed words was used as attributes and the bag of words representation of articles were used in indexing to create the Attribute Relational File Format (ARFF).

D. Model Development

For the classification model development, supervised learning was used, due to availability of labelled data. The most commonly used supervised learning techniques; MNB, Decision Trees, Rule Based Classifiers, SVM and KNN were considered in selecting the most appropriate classification algorithm. As Decision Trees and Rule Based Classifiers are highly context dependent and KNN is slow with large data sets, the selection was narrowed down to SVM and MNB. For these two algorithms, 10 models for each algorithm were created by incrementing the size of the training dataset. For each of these models, a 10 fold Cross Validation [19] (10 fold CV) accuracy and the prediction accuracy for a test set of 500 articles were measured. Also the average prediction time for classifying the articles in test data, using these two algorithms, was calculated. Based on these performance measures, the most appropriate technique was selected.

E. Building the tool

The tool includes an aggregator that search for and download news articles on continuous basis. The articles were then classified into classes and stored in database with links. Content manager handles client requests including article updates user personalization etc. Interactions with database are performed through a data service layer. Fig. 1 illustrates the architecture of the tool.

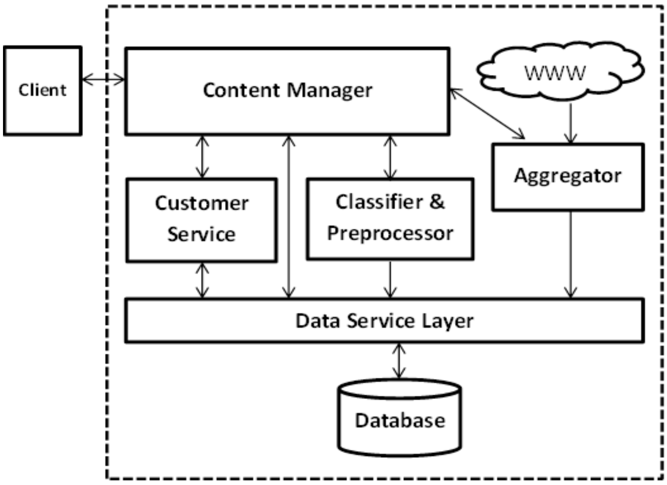


Fig. 1. Architecture of the proposed tool

IV. Implementation

In data acquisition, retrieving of news articles was automated using C# and Html Agility Pack, and downloaded articles were written to a text file in a pre-defined format.

The pre-processing filter was implemented as a C# module as described in the methodology. The stopword list that was received was further extended using word-frequency analysis, and the data driven stemming algorithm developed by UCSC was implemented in C#.

When it comes to data mining, the main reasons for selecting WEKA for classification model training are that it is freely available and portable. Also, the fully implemented WEKA java source could be used in c#, by creating a DLL.

The tool is designed to be developed as a client-server model, and therefore, for tool development .NET framework is proposed to be used, since Windows Communication Foundation (WCF) in .NET can be used for server side development. The client site was decided to be developed using ASP.NET. Microsoft SQL is proposed as the database system as it can be easily integrated with the .NET framework.

V. Results and analysis

As explained in the methodology, performance of ten 5-class models trained with SVM and MNB were compared, with respect to 10-fold CV accuracy, test set accuracy and average test set prediction time. The resulting accuracies, 10-fold CV accuracy for SVM and MNB and test set prediction accuracy for SVM and MNB, are shown in Fig. 2 and Fig. 3 respectively.

While both models demonstrate 10-fold CV accuracy of over 87%, accuracy of MNB is fairly higher than the accuracy of SVM, throughout all the models. Then, the models were evaluated using independent test sets. For the test sets, MNB has accuracy over 86% while SVM has accuracy over 81% which are always less than the accuracy of MNB.

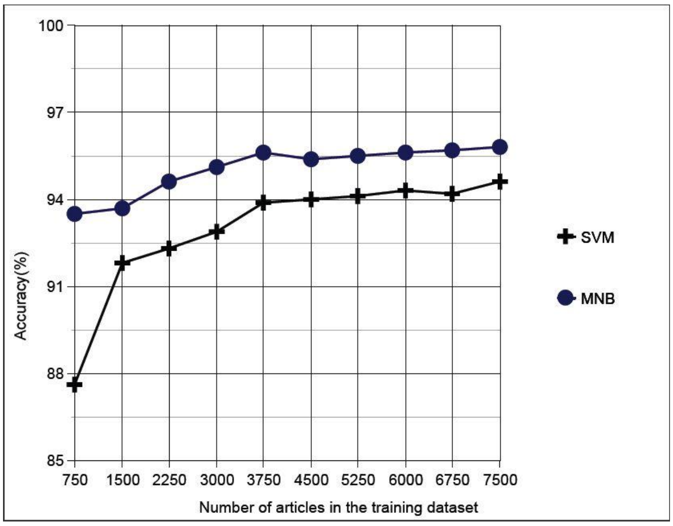


Fig. 2. 10-fold CV accuracy for SVM and MNB

The average time taken to make predictions on the test set of 500 articles is 6ms with a SVM model, while for a MNB model, it takes around 0.06ms. For the chosen dataset, MNB is approximately 100 times faster than SVM.

Based on the above results, Multinomial Naïve Bayes (MNB) was selected as the algorithm suited to train the automated news classifier, due to its small prediction time, and high predictive accuracy towards news data.

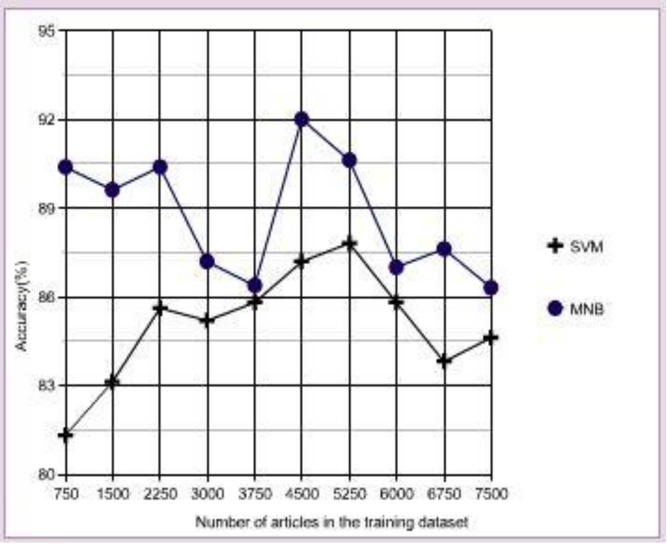


Fig. 3. Test set prediction accuracy for SVM and MNB

VI. Conclusion

In this paper, we introduced a methodology to pre-process the Sinhala text and automated classification of news articles using data mining techniques. Multinomial Naïve Bayes algorithm demonstrates an average classification accuracy of 88% for randomly selected test data sets. However, still there are some limitations with Sinhala text pre-processing, mainly caused by character representation issues (such as encoding) in the training data set. In addition, the existing classifier is limited to five news categories. Possible future extensions of the Sinhala news tool include personalization, continuous model development with user feedback, summarization of same news items from different sources and collaborative news article filtering.

References

[1] Pew Research Center for the People & the Press. (2011). Internet gains on television as public's main news source. [Online] Available: http://www.people-press.org/2011/01/04/internet-gains-on-television-as-publics-main-news-source/

[2] D. A. Meedeniya, D. A., "Investigating the applicability of partition-based clustering for Sinhala document", Master's thesis, University of Moratuwa, Sri Lanka, 2012.

[3] B. Hettige, "A computational grammar of sinhala for english-sinhala machine translation", Master's thesis, University of Moratuwa, Sri Lanka, 2010.

[4] R. J. Mooney, (2011). Text categorization. Unpublished raw data, University of Texas, Department of Computer Science, [Online] Available: http://www.cs.utexas.edu/~mooney/cs388/slides/

[5] R. J. Mooney, (2011). Text clustering. Unpublished raw data, University of Texas, Department of Computer Science, [Online] Available: http://www.cs.utexas.edu/~mooney/cs388/slides/

[6] G. Darko, I. S. Pandzic and H. Bacan, "Automated news item categorization", 2005.

[7] P. Shoval, B. Shapira and L. Tenenboim, "Ontology-based classification of news in an electronic newspaper", in Proc. of Inter. Conf. of intelligent information and engineering systems, Varna, Bulgaria, 2008.

[8] M. J. Pazzani and D. Billsus, D., "A hybrid user model for news story classification" in Proc. of 7th Int. Conf. User Modeling.,UM'99, Banff, Canada, 1999.

[9] C. Chan, A. Sun and E. Lim, "Automated online news classification with personalization", in Proc. of the 4th Int. Conf. of Asian Digital Library., ICADL2001, Bangalore, India, 2001.

[10] Language technology research, University of Colombo School of computing laboratory. (2011). [Online] Available: http://www.ucsc.cmb.ac.lk/ltrl/?page=projects&lang=en&style=default

[11] Sinhala 24 news. (n.d.). [Online] Available: http://www.sinhala24news.com/unicode/?isnew=1

[12] C. Sherman. (2002, Oct. 1). *Google news search leaps ahead*. [Online] Available: http://searchenginewatch.com/article/2068044/Google-News-Search-Leaps-Ahead

[13] "Celebrating pan l10n impact sri lankan component" (2012). [Online] Available: http://www.panl10n.net/english/2012PDF/PAN L10N - Sri Lanka_Last day.pdf

[14] Lankapuwath. (n.d.). [Online] Available: http://www.lankapuvath.lk/sinhala/

[15] Sri lanka mirror. (n.d.). [Online] Available: http://mirror.lk/sinhala/

[16] Divaina. (n.d.). [Online] Available: http://www.divaina.com/

[17] C. Elkan, (2013, May 18), "Text mining and topic models." [Online] Available: http://cseweb.ucsd.edu/~elkan/250B/topicmodels.pdf.

[18] Personal Communication

[19] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, Second Edition, 2005.

Implementation of an Autonomous Driver Assistance System Using Swarm Intelligence

W.M.M.P.B. Wickramasinghe[1], D.P.S. Gamachchige[1], U.D.G.T. Madushanka[1], S.D. Dewasurendra[1] and A. Kumarasinghe[2]

[1]Department of Computer Engineering, Faculty of Engineering
University of Peradeniya, Sri Lanka

[2]IFS Corporation,
IFS World, Kandy, Sri Lanka

***Abstract*— Modern world is more interested in using the time effectively rather than wasting time for unworthy activities. The time really matters for the regular travelling community and the ones with urgency. The road traffic is a tipping issue which leads to a lot of time and energy waste. The Swarm Intelligence is a segment in Artificial Intelligence which is used in the military land mine disposal, NASA Nanobots functionality, etc. This project is about researching on the capacity of swarm intelligence to form an autonomous driver system. In the simplest form, swarming is a decentralized control comprised with entities that works as an autonomous unit doing its own task.**

Enhancing a traffic light controlling mechanism does not provide an impeccable solution as traffic lights do not help in analysing the road traffic before accessing a busy route. It only helps on the junctions. This emphasizes the necessity of an Autonomous Driver Assistance system with contrast to a traffic light controlling system. Swarming of vehicles can be modelled as a Swarm Intelligence System and it may be possible to find a solution by using an appropriate model. If a solution model exists, then it is a matter of integrating a small intelligent device for each vehicle and creating a communication link among them. There will be more advanced communication system developed under this concept in the future than in the present.

Keywords*— *Swarm Intelligence

I. INTRODUCTION

Congestion in road traffic creates wastage of lots of time and money as well as many environmental issues. Usually, traffic congestion can be generated due to several reasons. For an example, concerning accidents, a traffic build-up cannot be seen at the time the accident takes place, thus, vehicles which do not know about the accident will create congestion. If these vehicles knew about the accident before entering that road, there would have been a possibility to avoid the traffic congestion.

The Drive Assistant System (DAS) is a system which helps to analyse the traffic condition before entering a road that gives an option for a driver to choose alternative routes. This system is to be developed based on Swarm Intelligence. The Drive Assistant System makes a network among the vehicles within the DAS to show the traffic condition of the roads to the driver on a dashboard display. A driver has the freedom to make decisions with the information available regarding the road traffic as is shown on the dashboard display. In order to develop this system, swarm elements to act as 'bots' are required. Simultaneously, the possibility of using mobile phones as 'bots' has to be analysed as a mobile phone has enough facilities to do this.

The key component of this project is Swarm Intelligence [1]. Swarm Intelligence (SI) describes the collective behaviour of decentralized, self-organized systems, natural or artificial. Because of using SI, there is no mechanism like central server and no need for high performance computing. In a swarm, even if one 'bot' did not function properly, still the system will work at a satisfactory level: progressive degradation.

II. LITERATURE SURVEY

Even though lots of research has been carried out on traffic modelling, it is possible to divide all of them into three categories.

1. *Optimizing the traffic light controlling system*: Enhancing the traffic light controlling mechanism seems not to be supporting that much as traffic lights do not help in analysing the road traffic before accessing a busy route. It only helps on the junctions.

2. *Introducing a central control mechanism (like central server)*: The control is given to the server to manage the traffic light or guide the driver. Obviously, such a server should have a high processor power and most importantly, the decisions taken by this system should be accurate. If the server does not work for few minutes, it will be troublesome. In this project, there is no central server. The advantage of it is even if a single bot fails to function well, the network system will still work.

3. *High performance satellite systems for identifying the traffic*: These satellites analyse the traffic and control the vehicles according to the traffic condition. This is really costly and need some advanced devices.

III. MATERIAL AND METHODS

The sole implementation language for the application is Python as it is a dedicated programming language for intelligent models.

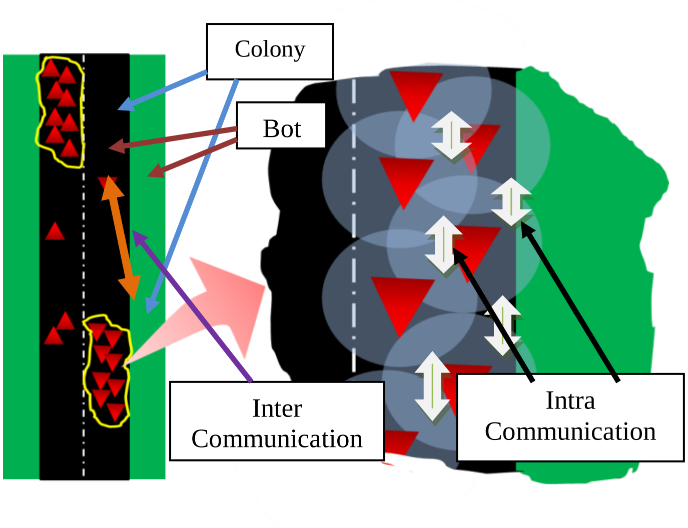


Fig. 1. System overview

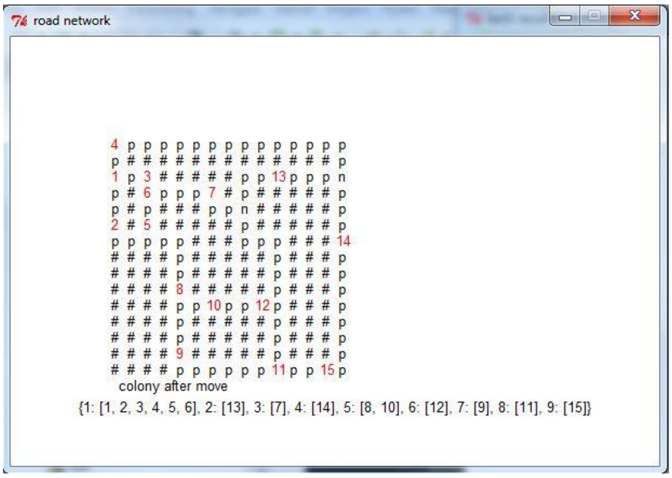
After the literature survey, a document was prepared about the different traffic scenarios and covered maximum number of traffic scenarios through the solution and came up with a model that is most suitable for the scenarios under study, with alternative solutions offered for others.

The vehicle with this solution is defined as a Bot. Each Bot has a unique ID. The Bot has two communication channels called inter communication and intra communication (see Figure 1). The intra communication channel is used for communication with other bots that are in the same colony (Bots' who are in the range of Intra communication channel will create a colony) and the Inter communication channel is used for communicate with the other colonies that are in predefined range (Inter communication channel range). In the Intra communication, Data Tables of the each bots' in the same colony communicates among them. Data Table is a string that holds the bot id, location and speed of bots' in the same colony. A colony has a token which gives right for broadcast (Using Inter communication channel) colony details (Average speed and location of the colony) to a bot by another bot. This token is passed throughout the colony randomly. This broadcasted information of other colonies is taken by each and every bot (Through Inter communication channel) and display the traffic density of the each road on a map in a user friendly format.

IV. Results and Analysis

A simulator was implemented for simulate the system and verified that the solution can be integrated in the real world scenario based on the simulated results. The simulation process can be explained as below.

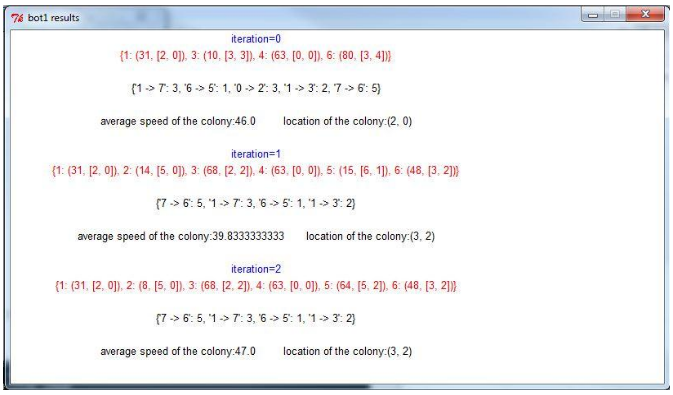
First view can be used to simulate the road network. This view is enabled to move the vehicle by mouse click on the view (Figure 2). The view shows the road network and the possible colonies that can be formed.



```
4 p p p p p p p p p p p p p p
p # # # # # # # # # # # # # p
1 p 3 # # # # # p p 13 p p p n
p # 6 p p p 7 # p # # # # # p
p # p # # # p p n # # # # # p
2 # 5 # # # # # p # # # # # p
p p p p p # # # p p p # # # 14
# # # # p # # # # # p # # # p
# # # # p # # # # # p # # # p
# # # # 8 # # # # # p # # # p
# # # # p p 10 p p 12 p # # # p
# # # # p # # # # # p # # # p
# # # # p # # # # # p # # # p
# # # # 9 # # # # # p # # # p
# # # # p p p p p p 11 p p 15 p
  colony after move
{1: [1, 2, 3, 4, 5, 6], 2: [13], 3: [7], 4: [14], 5: [8, 10], 6: [12], 7: [9], 8: [11], 9: [15]}
```

Fig. 2 View for simulate the road network

The second view (Figure 3) shows the results of the bot 1 in an understandable format (the bot that id =1).



```
iteration=0
{1: (31, [2, 0]), 3: (10, [3, 3]), 4: (63, [0, 0]), 6: (80, [3, 4])}

{'1 -> 7': 3, '6 -> 5': 1, '0 -> 2': 3, '1 -> 3': 2, '7 -> 6': 5}

average speed of the colony:46.0        location of the colony:(2, 0)

iteration=1
{1: (31, [2, 0]), 2: (14, [5, 0]), 3: (68, [2, 2]), 4: (63, [0, 0]), 5: (15, [6, 1]), 6: (48, [3, 2])}

{'7 -> 6': 5, '1 -> 7': 3, '6 -> 5': 1, '1 -> 3': 2}

average speed of the colony:39.8333333333        location of the colony:(3, 2)

iteration=2
{1: (31, [2, 0]), 2: (8, [5, 0]), 3: (68, [2, 2]), 4: (63, [0, 0]), 5: (64, [5, 2]), 6: (48, [3, 2])}

{'7 -> 6': 5, '1 -> 7': 3, '6 -> 5': 1, '1 -> 3': 2}

average speed of the colony:47.0        location of the colony:(3, 2)
```

Fig. 3 Results of Bot 1

For the simulation purpose, it is considered the range of the intra communication is 2 units. When considering bot 1 of the road network (Figure 2), it can be concluded that bots 4, 3 and 6 are in its range. When considering the bot 6, bot 5, 3, 2 and 1 in its intra communication range. Finally, bot 1, 2, 3, 4, 5, 6 forms a colony. The possible colonies that can be formed are shown under the label 'colony after move' in the road network view.

Bot 1 results view has a term called iteration. One iteration mean each and every bots in the road network do a one communication cycle (Both intra communication and inter communication). This view enabled to do iteration by click on the view. The data table of the bot 1 shown by the red colour and density table shown by the black colour. The average speed of the colony that calculate by the bot 1 and the location of the colony are shown below the density table.

According to the Figure 3, there are three iterations that have been done (intra communication + inter communication, three times). Results of the each iteration are shown below the iteration label. Initially, every bot has only its detail (id, speed, location). But after iteration 0, bot 1 have the details of bot 3, 4 and 6 because they are in the range of intra communication channel of bot 1. In the same iteration, bot 6 get the details from bot 1, 3, 5 and 2 since they are in the intra communication channel range of bot 6. Likewise, each and every bot are updating their data tables. In iteration 1, bot 1 takes the details from bot 6 and now it also has the details about bot 2, 5 since bot 6 has the details about them. After a few iterations, each and every bot has the details about other bots that are in the same colony which the particular bot belongs to. In the above simulation, bot 1, 2, 3, 4, 5, 6 are in the same colony. Thus, bot 1 has the details about all of them. Also, bot 1 can calculate the average speed and the location of the colony by using those details.

The road network shown in Figure 2 can be represented as in Figure 4. The number shown inside the node indicates the number assigned to the each node in this simulation.

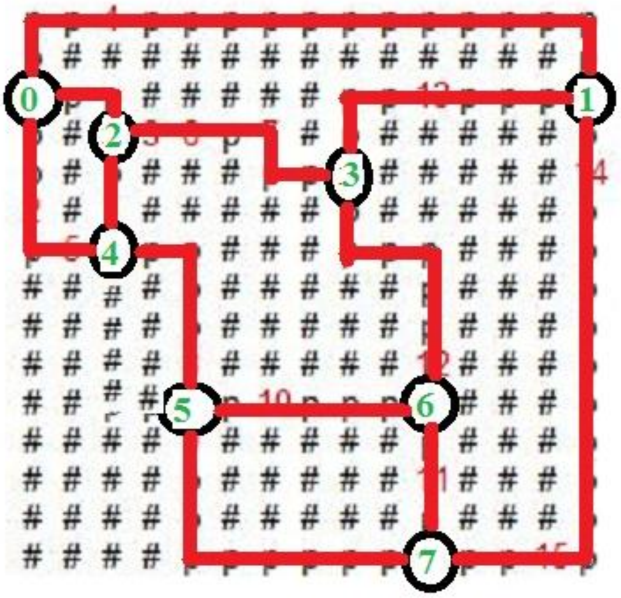


Fig. 4 Road network with node labels

After doing a inter communication cycle, bot 1 receives the output of the each and every colony. By using those colony outputs, bot 1 calculates the traffic density of the each edge. According to the bot 1, results shown in Fig. 2, ‘1 -> 7’:3 means the traffic density of the edge between node 1 and 7 is 3 and ‘6 -> 5’: 1 means the traffic density of the edge between node 6 and 5 is 1, and so on. The colony output is transmitted by the bot that has the token.

V. Discussion

The next step is to test the solution using mobile phones for verifying that it is working at an expected level in the real world. After verifying the system, an embedded device can be designed for vehicles.

One of the major problems when executing the project task is the high rate of functionality in the device, which will lead to a program with high resource occupancy. Further, there will be high power consumption due to signal broadcasting. These issues may be unavoidable as they directly deal with the technology used. Indefinable parameters such as incompatible conditions, accidents and bad road conditions can pose difficulties when it comes to the practical use in the real environment. They cannot be addressed adequately in our rationale and those issues can be avoided for better results by building up a knowledge base on the road networks with the help of data that can be acquired from other sources and transferred to the system as an extension to the project.

VI. Conclusion

The Driver Assistant System can be used to reduce the traffic congestion in cities and also in factory environments where forklifts and other carriers are being used for transportation of goods. There may be more advanced communication systems developed in the future. This system may be easily re-configured for those communication systems to bring the maximum advantage to the driver.

References

[1] E. Bonabeau, M. Dorigo, and G. Theraulaz. Swarm Intelligence: From Natural to Artificial Systems. New York: Oxford University Press, 1999.

[2] Renfrew, D.; Xiao-Hua Yu, "Traffic Signal Control with Swarm Intelligence," Natural Computation, 2009. ICNC '09. Fifth International Conference on , vol.3, no., pp.79,83, 14-16 Aug. 2009

[3] Alonso, L.; Oria, J.P.; Arce, J.; Fernandez, M., "Urban traffic avoiding car collisions fuzzy system based on ultrasound," Automation Congress, 2008. WAC 2008. World , vol., no., pp.1,6, Sept. 28 2008-Oct. 2 2008

[4] N. M. Kwok, Q. P. Ha and G. Fang: Motion Coordination for Construction Vehicles using Swarm Intelligence.

[5] Joanne Penner Ricardo Hoar Christian Jacob: Swarm-Based Traffic Simulation with Evolutionary Traffic Light Adaptation, 2002.

[6] Chen Yue-ming; Geng Hui; Sheng Gang, "Emergency Management System for Major Traffic Accidents Based on New Model and Algorithms," Computer Science-Technology and Applications, 2009. IFCSTA '09. International Forum on , vol.2, no., pp.367,371, 25-27 Dec. 2009

[7] Yang Ying; Wei Ximing, "The Research of Collision Avoiding System Based on Millimeter Wave and Image Processing Technique," Computational Intelligence and Security, 2006 International Conference on , vol.2, no., pp.1789,1792, 3-6 Nov. 2006

Adaptive Self-Learning GUI Quality Assurance Tool

I.R.P.D.I. Jayathilaka[1], H.D.S. Lakmali[1], H.M.P. Prasadini[1], H.M.P.C.K. Navarathna[2] and D.S. Deegalla[1]

[1]Department of Computer Engineering, Faculty of Engineering,
University of Peradeniya, Sri Lanka
[2]IFS R & D Internationals Pvt. Ltd., Sri Lanka.

***Abstract*—Since the Graphical User Interface (GUI) is the front end of software which facilitates a user to interact with the software, the visual layout of software highly affects end user performance and acceptance of software. However, validation of visual layout of GUIs has not paid enough attention. Even though there are some tools available for visual testing, they generally use a fixed set of predefined standards and are not customizable. Since there may be more than one GUI standard maintained in the same company and they sometimes introduce new GUI standards, existing tools may not be able handle such situations. In this study, we have developed an Adaptive Self-Learning GUI Quality Assurance Tool which is capable of learning the standards from a given set of example GUIs. The tool has shown 78% percentage of accuracy on the experimental dataset using Nearest Neighbour algorithm.**

***Keywords*— Graphical User Interface (GUI); Data Mining and Machine Learning; Instance Based Algorithm; Nearest Neighbour Algorithm**

I. Introduction

Graphical User Interface (GUI) is the visual layout which gives capability of accessing the functionality of the software. It plays major role in user performance and acceptance of software [3]. It is known that the first impression of the GUI is the last impression of the software system. Further, GUI based software systems represent more than 60% of software systems being developed today [5].

User requirement of a product or service will be guaranteed by the quality assurance process. Quality assurance of GUI falls into two main categories, namely functional and visual layout quality assurances. Functional quality assurance is validating functionalities of the GUI [3]. A GUI contains controls such as text boxes, text fields, buttons and dropdowns etc. Consistency of controls in GUI is considered in visual layout quality assurance. Mahajan and Shneiderman [4] say that a user interface is consistent or inconsistent with respect to standards or guidelines which may be within the individual application or for all the applications running on the particular system. For an example, some interfaces may have text field which are aligned only to left side. If that interface has right aligned text field it is inconsistent. Mahajan and Shneiderman [4] say that an inconsistent GUI could slow user performance between 10% and 25%. Most of the software companies do quality assurance of the functionalities of the GUI controls in the software but not the visual layout quality assurance.

Companies follow GUI guidelines or standards to create GUIs in order to keep the GUIs consistent. Those standards may be strict ones. Providing fully detailed GUI guidelines reduce the room for creativity. Therefore, GUI visual layout could be tested manually by GUI quality assurance person. Issa, Sallito and Garousi [3] had shown that the decision would be dependent on the person's preferences and it is very tedious, time consuming and error prone. Some of the companies use tools such as CROSSCHECK [1], Fighting-layout-bugs [2] and SHERLOCK [4], to detect the visual layout defects. Some of the tools were identical to companies and they used predefined standards or rules and they are not customizable. Some of the companies have several different standards in use and new GUI standards are introduced with time and also changes made to existing standards. To accommodate the changes, a tool must be customizable; else the tool would be useless.

The objective of the study is to develop GUI quality assurance tool using data mining approach. The tool should be able to extract the patterns of the standards of GUIs and test GUIs according to any given standards. This tool could take any changes to standards of GUIs in any company without changing implementation of the tool.

The remainder of this paper is structured as follows. Section II surveys previous work. Section III describes the proposed work. The implementation of the tool is described in section IV. Results and discussion is given in section V and finally, conclusion and future plans are outlined in section VI.

II. Previous Work

A tool called SHERLOCK [4] was implemented by Rohit Mahajan and Ben Shneiderman in 1997. That was designed to identify the inconsistency of visual design and textual properties of GUIs. However, the tool detects inconsistencies according to a set of hard coded standards defined by them [4]. They predefined some ratios to be fulfilled by the controls to provide suggestions for inconsistencies. Further, the tool was not customizable. Another tool called CROSSCHECK [1] was implemented to identify the web interface visual differences between different browsers which are known as Cross Browser

Incompatibilities (XBI). Five visual XBIs are considered in CROSSCHECK. They are size difference ratio, displacement, area, leaf Data Object Model [6] text difference, and image distance. Visual defect will only be reported if it is not identical in both browsers.

Fighting-layout-bugs [2] is a tool specialized in capturing layout defects in a web interfaces. It could detect invalid image Uniform Recourse Locators (URLs). It also detects text which is very near or overlaps a horizontal edge and vertical edge. It detects text which is not readable because of too low contrast. There is a research work [3] done on visual layout quality assurance of GUIs tried to show that importance of fixing visual defects of GUI. They have conducted a study of defects in four open source systems WordPress, Moodle, Joomla and Bugzilla. According to their work they have shown that visual defects are around 16% - 33% from reported GUI faults, i.e., both functional and visual. Objective of this research is to motivate the importance and the need for systematically conducting visual testing among researchers and practitioners.

III. Proposed Work

The problem in existing tools is they test GUIs according to a set of predefined standards and those standards are fixed and cannot change or customizable. However, the standards change with time and vary in different companies. Therefore existing tools become useless or reimplementation is needed. There is a need for a customizable tool which is capable of adapting to changes of standards.

A. *Training Phase*

At very first, the training data set was prepared and all the facts that research show are basically based on this set of GUIs. Generally, a GUI being faulty means it contains improperly spread and/or placed Controls. Thus one way to locate defects in GUIs is to search for faulty Controls. In order to predict whether a control is going to be faulty or not, we considered supervised learning. Since the prediction is given for individual Controls, a GUI Control should be the atomic unit or instance for learning. First, a Control should be represented in attribute-value pairs and class label should be provided whether the Control is faulty or not for the training Control set. So, a set of attributes which are going to represent the state of a Control on a GUI was decided. The details of selected attributes are as follows. Fig. 1, Fig. 2 and Fig. 3 represent details of the attributes for sample GUIs.

1). Type
2). noOfCtrls
3). sameLeft
4). sameTop
5). sameLeftWithType
6). sameTopWithType
7). sameRight
8).sameBottom
9).sameBottomWithType
10).sameLeftAndSameWidth
11).sameLftAndDfferentWidth
12).sameTopAndSameHeight
13).sameTopAndDffrntHeight
14). rightStarts
15). leftEnds
16). bottomStarts
17). topEnds
18). sameWidth
19). sameHeight
20). parellelLeft
21). parellelRight
22). Parellelup
23). parellelDown
24). horizontalRange
25). verticalRange
26).betweensHorizontal
27). betweensVertical
28).betwnsHrzontlWthType
29).betwnsVertcalWithType
30). topDiffWithLeft
31). topDiffWithRight
32). bottomDiffWithLeft
33). bottomDiffWithRight
34). horizontalDividers
35). verticalDividers
36). guiToCtrlsRatio
37). rowSpacing
38). columnSpacing
39). NearestHorizontalGap
40).NrstHrzntlNghbrLabel
41).NearestHrzntalUpAlign
42).NearestHrzntlDwnAlign
43). NearestVerticalGap
44). NearestVrtclLeftAlign
45).NearestVrtclRightAlign
46).NearestVrtclNghbrLabel
47). childOfChild
48). Class

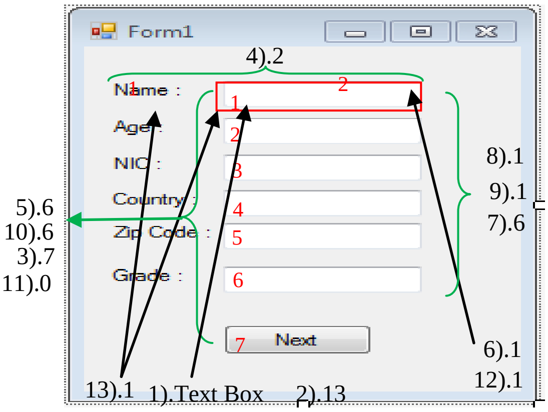


Fig. 1. Sample values for attributes from 1 to 13

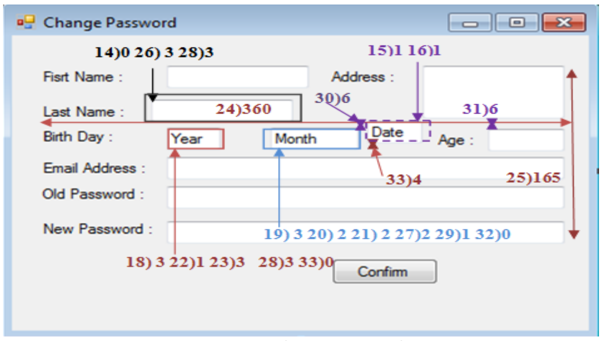


Fig. 2. Sample values for attributes from 14 to 32

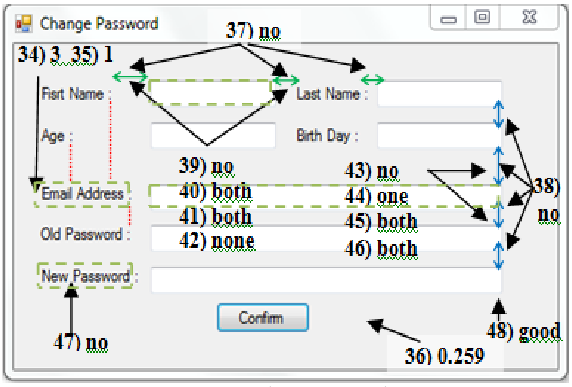


Fig. 3. Sample values for attributes from 17 to 48

As the next step, several machine learning algorithms were considered for training and the Nearest Neighbour algorithm [6] was selected as the suitable algorithm by comparing both the accuracy and complexity of the algorithm. When someone uses the tool, the user can select the desired standard by modifying the training data set. Then, the tool can adapt to new standards. If we selected a Decision Tree model, it should regenerate the model when training data set changes. Since Nearest Neighbour algorithm [6] does not create any model as such in the training phase, it could be used to change the structure by altering the training data. This was one of reasons to select Nearest Neighbour as the learning algorithm and it predicts good or bad GUI Controls by selecting majority class label of its nearest GUI Controls. Overall architecture for training phase is shown in Fig 4.

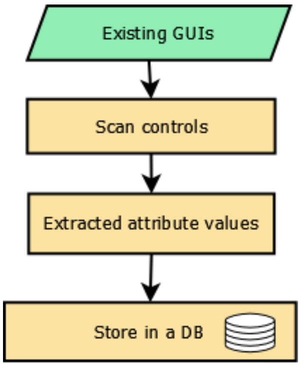


Fig. 4. Training Phase

B. *Testing Phase*

Then, the tool was implemented such that user can create a GUI guideline by adding example GUIs according to the guideline. Thus, the set of example GUIs become training data set to the guideline.

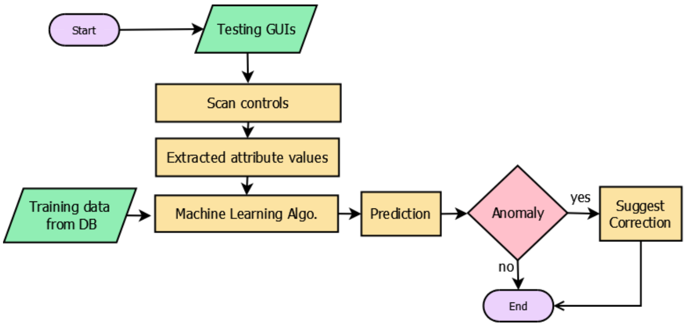


Fig. 5. Testing Phase

Basically the training data set for guideline represents the guideline by extracting the standards used in creation of those GUIs by learning the patterns in training data. Thereafter, the user can upload the set of GUIs to be tested to the tool. Then the tool predicts faulty Controls by running Nearest Neighbour algorithm on test GUIs, based on controls in the training data set. Overall architecture for testing phase is shown in Fig 5. The tool facilitate user to create more than one GUI guideline by adding example GUIs to each guideline since the same company may maintain more than one guideline. Different projects in the same company may employ different GUI standards. Therefore, training data for different guidelines are maintained separately.

IV. Implementation

Since we are checking faults of C# windows forms, C# was used as the programming language for extracting features of GUIs, processing data and writing the basic logic of the software. The entire example GUIs were maintained in a MySQL database. WEKA is used as the data mining tool for selecting a suitable machine learning algorithm. Different machine learning algorithms were evaluated on example GUIs by considering accuracy as the performance measure.

V. Results And Analysis

A test data set was developed considering visual defects of spacing, alignments and sizing in order to check the accuracy of the tool. We evaluated the tool on the test data based on the accuracy and considered the ratio of number of correctly classified controls to the total for each test GUI. The tool shows 78% average accuracy for the test set.

VI. Conclusions And Potential Future Work

Existing GUI quality assurance tools are mostly based on a pre-defined set of guidelines integrated in the tools. In this paper, we proposed a tool which can handle more than one set of guidelines by using data mining approach. Our proposed tool has shown 78% of average accuracy on the test set. To further develop this tool, one may consider removing irrelevant data in the training phase and integrate ensemble learning.

References

[1] Choudhary, & Shauvik, R., “Detecting cross-browser issues in web applications”. Waikiki, Honolulu, HI, USA: ACM, 2011.

[2] Fighting-layout-bugs, Retrieved from https://code.google.com/p/fighting-layout-bugs/ Accessed: 14 February 2013.

[3] Issa, A., Sillito, J., & Garousi, V., “Visual testing of Graphical User Interfaces: An exploratory study towards systematic definitions and approaches”. In Tilley, R. Scott, Distante, & R. Damiano (Ed.), pp. 11-15, IEEE, 2012.

[4] Mahajan, R., & Shneiderman, B., “Visual and textual consistency checking tools for Graphical User Interfaces”. 23, pp. 722-735, IEEE Computer Society, 1997.

[5] Robinson, P., B., and Memon, A., “An Initial Characterization of Industrial Graphical User Interface Systems”, in Proceedings of International Conference on Software Testing Verification and Validation. pp. 11-20, IEEE, 1999.

[6] Witten H., Frank I. E., and Hall M. A.,. Data Mining : Practical Machine Learning Tools And Techniques. 3rd Edition. Morgan Kaufman, 2011.

Database Standard Checker

M.H. Satheeq and Z. Maraikar

Department of Computer Engineering, Faculty of Engineering,
University of Peradeniya, Sri Lanka

***Abstract*— Code review is an important step during the process of software development. Generally, a team of expertise is involved in debugging process and code standard checking. Manually, reviewing millions of lines of code to enforce the code standards is a tedious job as that process follows a constant guideline and rules. Repeatedly checking for the same code standards in every review is ultimately as exhausting as it must be exhaustive. Also this consume much time and cost. This study investigated the effectiveness and importance of automating code review and code standard checking, and explains a comprehensive solution for automating code standard checking efficiently.**

***Keywords*— Code review; Code Standards; MS-SQL; Code Standard Automation; Standard Checking**

I. Introduction

There are several errors in software development. In case of databases, many tools are available to check syntax and logical errors. Members of a development team follow different standards for scripting [1]. So it is a part of the development process to check the violations of code standards by code review.

The main objective of code review is to catch bugs and also to enforce the code standards. With generally tight deadlines, it is really hard to spend time for a code review. Code reviews are costly and the reviewers are wasting time on catching standard violations. It is a bottle-neck in the release train to wait until the code reviews are finished properly [2]. Hence automated code checking is very useful to developers.

Companies have automated the code standard evaluation using their own service components and code review tools such as Code Collaborator (a code review tool). Even though evaluated comments are added to the review automatically, a formal code review process must run through and reviewers have to take part in the process. Unfortunately, reviews are not completed by the reviewers on time; developers never check the comments until the release. During the release time, incorporating the review comments consume great deal of time than creating the scripts, because the developer is completely far away from the time that the script was created and in a critical conjunction with the release train.

This research is targeted to Pearson Lanka (Pvt) Ltd. software development environment and automating T-SQL database script standard checking through SQL Server Management Studio (SSMS).

II. Related Work

Reviews are performed by review boards composed of highly skilled and experienced subject matter expertise (SME). For the ease of SME, software tools provide assistance with code review and inspection process. I observed and classified hundreds of review comments during my industrial training period at Pearson Lanka. The study revealed that most developers are doing the same mistakes again and again while coding. Also code reviewers and automated tools want to specify the warnings and comments repeatedly. To reduce the uncertainty of company code standards, standard checkers and code quality tools are provided to the developers. General code standard checking tools are not exactly fitting with company standards.

Pearson Lanka (Pvt) Ltd. maintains larger scale databases and has their own code standards for database scripting and software implementation. So reviewing all the code changes was consuming a huge amount of time. To overcome the situation, they created an automation tool to evaluate their own MS-SQL code standards. Figure 1 describes the usage of the automation tool.

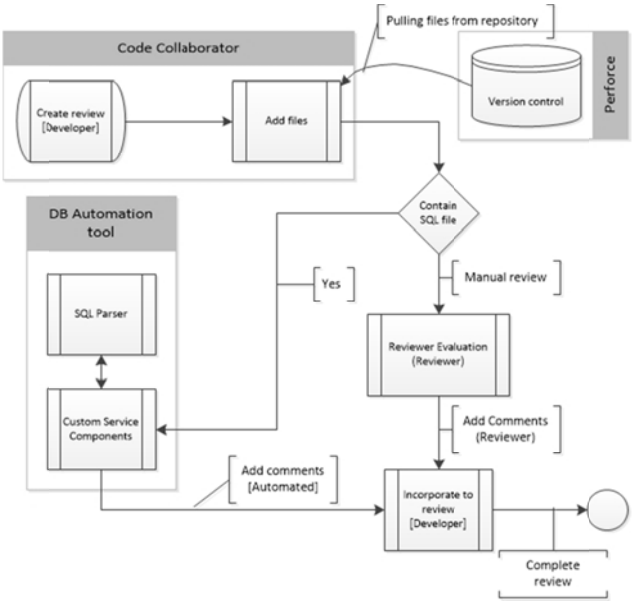


Fig. 1. Activity Diagram - Pearson database code review

The developer creates a new review in Code Collaborator and refers files from Perforce (a version control tool). Once the files are retrieved to Code Collaborator, the automation tool invokes if there are any SQL file present. Then, by creating the SQL tree parser, the automation tool traverses and find possible standard violations. The violations are mentioned as comments and added to the review through Code Collaborator back-end. Finally, the reviewers also inspect the code and add comments.

Somehow this process is similar to a manual code review. The only different is standard checking is automated. In addition to this requirement, there are many tools that are used to manage source code quality in the software development environment.

For an example, ‘Sonar’ is a tool used to manage source code quality and it offers reports on Duplicated code, Coding standards, Unit tests, Complex code, Potential bugs, Comments, Design and architecture in a general point of view. It is a web application where the admin can define the standards and rules. But this tool does not support MS-SQL and also the users have to use another external application (a web service) to validate the code standard.

‘JustCode’ is another tool which has the ability to use as an internal plugin with integrated development environment (IDE). This tool is used to analyse and refactor the source code. And a tool called ‘Mimer SQL validator’ is a web service which validates the MS-SQL script against ISO SQL99 standards. The purpose of introducing the above tools is to propose the necessity of a solution to overcome their shortcomings.

All the above models are made to improve code quality and most of them are used as an external application from the development environment. Some enterprises’ applications’ such as ‘JustCode’ and ‘ReSharper’ are internally used as a plug-in in the software development tools (Eg: Visual Studio). Although almost all the tools come with readymade rules that help in maintaining source code quality and code standards. The trick to a successful automated code review is to combine the built-in rules with custom ones.

As shown in Fig. 1, the automation tool is also treated as an external tool which is integrated with Code Collaborator. Even though this is highly advanced tool to evaluate the company standards, the developers never get the full satisfaction for the effort made. Therefore, combining those analysis tools with the same development environment is much appreciated by the developer, since they can easily access the tool and get validated.

III. Proposed Work

Considering the background, existing solutions and the cost of the code review, I came up with the following ideas and have done study about them with an internal survey in Pearson Lanka (Pvt) Ltd.

Plan A: Create a web application to validate company code standard.

Plan B: Create a plug-in with a web service.

A. *Create a web application to validate company code standards.*

Using a web application to check code standards is similar to Code Collaborator review, but the developers can have privacy while validating code standards. Again this procedure is similar to use of an external application in the development environment.

B. *Create a plug-in with a web service*

Through the study I have done in this situation, I realized that providing a tool which can be used as an internal / integrated application with IDEs and a centralized web service to customize the standards is a good solution. As I am conducting this research with Pearson Lanka (Pvt.) Ltd, it is useful to obtain a result by integrating the database code standard checking with software development environment. Since most of the developers use the SSMS and Visual Studio to develop database scripts, I decided to develop a tool which is compatible with the above IDEs. So that developers can easily evaluate the code standards simultaneously from the editor while they develop the script. Also the web service is a centralized web service which can be customisable according to company standards.

In addition to the plug-in and the web service, a comment analysis tool can be developed to analyse comments from the server side. Most of the companies have internal knowledge transfer sessions and motivational workshops to improve software development process. With the help of a web service, we can store all the comments as XML entries so that comments can be analysed for further study. This tool finds the most frequent mistakes developers made. This is to advise developers to minimize the mistakes that we found using the analysing tool. This will improve developer’s knowledge and decrease mistakes in scripting. Also using the results by using the tool, valid sample code snippets can be distributed among the developers. This makes the developments process easier.

IV. Implementation

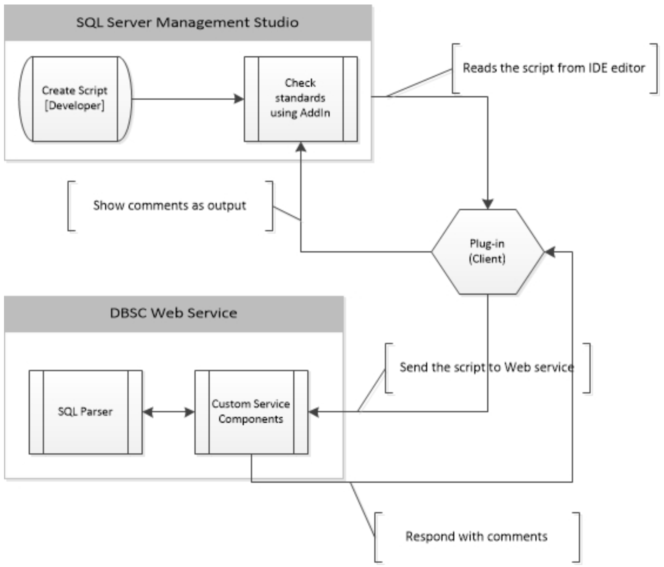

SQL Server Management Studio
Create Script [Developer]
Check standards using AddIn
Reads the script from IDE editor
Show comments as output
Plug-in (Client)
DBSC Web Service
SQL Parser
Custom Service Components
Send the script to Web service
Respond with comments

Fig. 2. Higher level diagram of Database Standard Checker

The plugin is an extension to the SSMS and Visual Studio. It can be developed using .Net framework 3.5 and C# [3]. Also the plugin access the SSMS or Visual Studio editor, output window and menu items so that we can read the script from the editor and then invoke with the web service. Once the web service responds with the comments, they are displayed on the output window. Figure 2 describes the process of a code standard checking using Database Standard Checker and the web service. The following statements describe in detail the standard checking using Figure 2.

1. Developer feeds the script to SSMS plugin to validate.
2. The plugin uploads the script to the web service.
3. Web service uses Pearson service components to validate the standards and respond with review comments.
4. The comments will be displayed in the SSMS output field.

It is clear that the process of validating the scripts as shown in Figure 1 is a big process and basically a code review. Thus, delivering a tool to validate scripts is providing the right solution.

At the beginning, SSMS 2012 and Visual Studio 2010 were used to create the plug-in and test. SSMS 2012 was the targeted environment. Microsoft DTE event handler libraries [4] are widely used in the plug-in development. Then different versions of the plug-in were created for SSMS2008 and SSMS2008R2. The web service is created using ASP.Net and Service Oriented Architectural Protocol (SOAP). It includes the Pearson custom service component to validate MS-SQL script and generate comments. It responds to the plugin with comments according to the script. The web service can be hosted in a centralized Windows server. Also the plug-in was packaged as Windows installer format so that the user can easily configure the tool.

Agile methodology was followed during the implementation process. Functional test and Unit Test were carried out by the developer and Usability Test was carried out by a team of Database Administrators and Specialist from Pearson Lanka (Pvt.) Ltd. I wanted to fix some bugs in the initial version and released a new version. During the Microsoft SQL Server Servicepack1 update, the tool was not capable of being loaded into SSMS2012. Then, the bug was identified and fixed.

The analysis tool was created using basic data mining techniques and C#. As mentioned earlier, all the comments generated by the web service were stored in XML data format and collected. Using the data and the analysing tool, the comments can be analysed.

V. RESULT

The web service deployed into the Internet Information Services (IIS) in an internal server in Pearson Lanka (Pvt.) Ltd. In addition, the plug-in was distributed among all the developers and DBAs in Pearson. An easy configuration method was documented and tested for installing the plug-in. Most of the MS-SQL script developers used the tool and showed a good feedback. A survey was carried out before and after the tool usage. The survey revealed that almost one week of code review time was reduced to 1-2seconds to evaluate by using the tool.

VI. CONCLUSIONS

According to the results and feedback from the tool users, it is crystal clear that this tool makes a significant effect in software development process and standard checking. The solution I proposed here was the most comprehensive solution for automating code standard checking.

VII. POTENTIAL FUTURE WORK

This is a sample application made to automate code standard checking. Here I have used MS-SQL as a test program. Going further, we can automate this process to other languages as well. Here I have automated only part of company standard that are used in Pearson. In future different techniques can be used to include all the other standards as well. The analyzing tool can be developed further to use advanced data mining techniques and can improve development processes drastically.

ACKNOWLEDGMENT

I wish to express my deepest gratitude to Mr. Nalina W. (Director & GM), Preethiviraj K. (Associate Database Architect, MVP), Industrial Supervisor, Mr. Dinesh Asanka (Database Specialist) and all the staffs of Pearson Lanka (Pvt.) Ltd. for their support in various ways, providing me necessary data with guidelines and allowing me to deploy the project in the company development environment.

REFERENCES

[1] Stacy Nelson, Johnn Schumann, “What Makes a Code Review Trustworthy?,” in *Proceeding of the 37th Anuual Hawaii Internaional Conference on System Sciences (HICSS'04) - Track 9 - Volume 9),* Washington, DC, USA, 2004.

[2] Alberto Bacchelli, Christian Bird, “Expectations, Outcomes, and Challenges of Modern Code Review,” in *Proceedings of the International Conference on Software Engineering*, 2013.

[3] M. H. Satheeq, “Techie Notes,” Blogger, 21 Jan 2013. [Online]. Available: http://satheeq.blogspot.com/2013/01/how-to-write-sql-server-management.html. [Accessed 10 February 2013].

[4] P. Eremeev, “Visual Studio Automation Object Model. EnvDTE interfaces,” 18 10 2012. [Online]. Available: http://www.viva64.com/en/b/0169/. [Accessed 15 02 2013].

Memory Management in CUDA

M.H.M.G. Dissanayaka, W.T. Madushan, and D. Elkaduwe

Department of Computer Engineering, Faculty of Engineering,
University of Peradeniya, Sri Lanka

Abstract- **Parallel computing is the modern approach for high performance computing. NVidia CUDA leverages the modern parallel computing by bringing GPU into general purpose programming. At present, CUDA architecture and the API provides all most all functions required for parallel computing in CUDA GPUs with the exception of handling pointer based data structures. The main issue with pointer based structures is that pointers are context dependent and therefore, structures that contain pointers cannot be copied from main memory into the memory of the GPU or vice versa. Once copied, the pointers stored within the structure are no longer valid. Currently, programmers are advised not to use data structures with memory pointers or to convert such structures into an array based representation. In this paper, we propose a solution to overcome this limitation. We have developed a memory management library which can be used to handle data structures with memory pointers. By using our library, the programmers can directly use structures with memory pointers within the GPU.**

Keywords – CUDA; Parallel Computing; GPGPU

I. Introduction

NVidia CUDA is an architecture coined by the NVidia Cooperation in 2007 [1]. CUDA architecture allows NVidia Graphic cards for general purpose computing – hence the term General Purpose Graphic Processing Unit or GPGPU. These GPGPUs are capable of parallel programming and some graphic cards contain more than hundreds of CUDA cores. These GPUs provide massive computing power which can be effectively used to handle the computational demand of modern applications.

Applications use various types of data structures – binary search trees, graphs etc. - to hold their internal data. Most of such data structures use memory pointers to connect components of the structure together. For example, a node (say node A) in a binary search tree will contain memory pointers to two other binary search trees. These memory pointers are then used for traversing the tree.

In brief, a memory pointer tells where the said object is located in the memory. For example, node A contains the memory location of the two trees. Using this location, information we can access the two sub-trees.

With this background, let us now turn our attention to basics of GPGPU programming. The GPU or the Graphic Processing Unit has its own memory which is used by the CUDA cores. CUDA cores use Single Instruction Multiple Threat (SIMT) method for the parallel computation. That is to say all processing elements of the GPU will execute the same instruction on different data items. These data items will be accessed from the GPU's internal memory and not the main memory of the computer.

Now suppose one needs to hand over a certain computational task to the GPU. Then we need to copy the data required for the computation from the main memory to the GPU's memory, or using the typical terminology from the host (i.e. main memory) memory to the device (i.e. the internal memory of card) memory.

Among other things, NVidia has provided libraries for memory management between host and device. Thus one can copy the data to the GPU, unleash the parallel computing elements and copy back the results.

Out of the many functions provided in the CUDA API, CUDAMalloc and CUDAMemcpy are central for our work. CUDAMalloc allocates memory in the device and CUDAMemcpy is used to copy data from the main memory to the device memory or from device memory to main memory.

This simple memory management scheme causes issues when the said data structure contains memory pointers. Going back to the example of the binary tree, when node A is copied into the device, it will still contain the memory addresses of the two sub-trees which are in the main memory (not the device memory). In theory, we can copy the two sub-trees into the device but it is highly unlikely that they can be copied to the same device memory address so that the pointers in node A are still valid. Further, one cannot modify the contents of node A and update the memory pointers without knowing the actual structure of node – thus making a generic implementation impossible.

II. Current Solution

Currently, any linked data structure that is to be used in the computation needs to be implemented using an array. Going back to our binary tree example, the tree should be implemented using an array so that sub tree locations can be calculated using the address of the current node.

Array based realization of sparse data structure – for example a sparse binary tree – is known to be inefficient. Thus the current solution is not ideal. The programmer either needs to avoid linked structures (possibly by modifying the algorithms) or have sub optimal realizations of them.

III. Our contribution

In this paper, we propose a solution to manage linked-list data structure in CUDA. The main drawback in linked structures with CUDA is once an object is copied to the device, the pointers it contains are no longer valid. We propose a method call memory offsets to overcome this issue.

The solution we present here is based on the ANSI C programming language and was compiled using NVidia's nvcc compiler [2] without any optimization flags.

IV. Proposed Solution

In this section, we present our solution. Before explaining the solution, we recall the main issue with memory pointers that causes this problem. Suppose we call the malloc function of the C programming language. Then we get an address or a memory location in the main memory which we can use to store our data. Later on, we can copy this memory to the device, but it will not be to the same memory address. Hence, any data structure (or a variable) that stores the previous address will not work.

In this work, we propose to use *memory offset* as opposed to memory address when accessing linked data structures. We link our C program against a slightly different implementation of *malloc*. This particular *malloc* function sets up a relatively large region as the heap and records the start address of that region (Let us call this address *start)*. When memory is requested, *malloc* will look for a *suitable* region, say the memory address of this region is *x*. Once a suitable region is found, our *malloc* function returns *(x – start)* instead of *x*. We call this value the *memory offset*.

The memory offset specifies the location of the data item relative to the *start*. Obviously, one cannot use the memory offset as a typical pointer. One needs to add the *start* to memory offset and use that value as the memory address. Our implementation provides number of C pre-processor macros and several in lined functions to deal with this conversion. An illustrative example of a typical C code using memory pointers and our memory offset method is shown in Figure 1-3:

```
node_t * temp;
temp = malloc(sizeof(node_t));
temp -> data = data;
```

Fig. 1. Typical C code

```
offset_t temp;
temp = malloc(sizeof(node_t));
MAKEPTR(node_t *, temp) -> data = data;
```

Fig. 2. Same code with memory offset

```
#define MAKEPTR(t,offset) ((t) (offset+start))
```

Fig.3. Implementation of a macro

As one can see from the above example, the difference is minor and it is just syntactical.

The reason for introducing memory offset is that it gives us a point of indirection. All the data structures store memory offsets – in other words relative position of the structure with respect to *start*. Suppose we want to copy this structure into the GPU. For this purpose, we find a sufficiently large area of memory within the device memory and copy the entire region which was used for memory allocation from the main memory to the device memory. Now the only thing we need to do is change the value of *start* to suite the new location of data structures in the device memory. Note that we do not need to change any memory offset values within any data structures because these offsets are still valid and can be used to compute the new address using the new *start*.

A. The malloc function

Our *malloc* function is very similar to a typical *malloc* except that:

- It returns the memory offset as opposed to the memory address.
- It always attempts to allocate objects in a dense manner so that copying objects from main memory to device memory would be efficient.

To emphasize on the second point above, the *malloc* function attempts to allocate the next object directly after the first object unless there is an alignment requirement. In a typical code that we experimented, it is very much possible to do a compact allocation since most of the time we will be allocating objects of the same type. Going back to our running example of the binary tree, when creating a binary tree, one allocates nodes all of which are of the same type and hence size. As such, all these nodes can be densely packed. Any requirement for alignment will be automatically satisfied if we start with aligned memory. Therefore, copying from main memory to device memory can be made efficient.

V. Overheads

The main overhead introduced by our method is that every dereferencing of a memory offset requires an additional memory reference to get the current value of start and an addition operation. Our macro level analysis shows that this overhead is negligible.

VI. Implementing Aho-Corasick

We implemented Aho-Corasick algorithm – a commonly used string matching algorithm – using the proposed method. We customised an existing implementation to use our library. The search function was changed to run in CUDA cores on a graph created by the CPU.

References

1. Kandrot, J. S. (2010). CUDA by Example. Boston: Pearson Education, Inc.
2. NVIDIA. (2008). CUDA Programming Guide GeForce 8 and 9 Series.

Dynamic Depth of Field for a 3D Graphics Engine Based on Eye Tracking

S. Paranagama, S. Jayasundara, M. Baminiwatta and K. Samarakoon
Department of Computer Engineering, University of Peradeniya, Sri Lanka

Abstract—**Virtual Reality aims to immerse a user in a simulated environment. To make the immersion convincing enough, various techniques are used to make the simulation as real to life as possible. While some techniques like post processing effects focus only on the visual presentation of a simulation, other techniques target the level of control or the level of impact a user has on the environment.**

This project aims to implement dynamic depth of field for a 3D Graphics engine based on eye tracking. It is based on enhancing the level of impact a user has on the environment by changing the visual simulation based on where the user is looking at. Similar advances have been done before, however the user was constrained in certain movements as discussed in the literature survey. This project aims to remove such constraints and enhance the user's experience in a virtual environment. It will track the user's eyes and head to obtain the user's gaze point in the display and approximate the focal distance. Based on the focal distance, the depth of field of the scene is made to change. The 3D simulation is built using the Ogre3D SDK and the eye tracking is done using OpenGazer.

I. Introduction

3D Virtual environments, consisting mainly of Games and Simulations have always amazed us over the last decade. They have been used to immerse players for hours and to train people on simulations. Thus, to make simulations as real to life as possible, the 3D Graphics Industry has developed major techniques to increase the visual realism. However, the trend has been just that; to increase the immersion only through visual realism. Only in the last few years has the Virtual Reality (VR) industry taken steps to increase the immersion of a user in special ways. One example would be the Microsoft®Kinect™Device [1]. Devices like that aim to increase a users immersion by his/her level of control over the environment. Our project aims to increase a users level of immersion in a similar way by giving him and letting him control a dynamic sense of depth in the 3D environment. In other words, we will be dynamically changing the depth of field based on where the user is looking at in the simulation. The users gaze point is derived by tracking his/her eye and head movements.

The literature survey discusses the history and current context of eye tracking and its use in Human Computer Interaction and a brief discussion on depth of field effects. After that, the methodology is presented outlining the steps followed to complete the project. Next, the implementation section gives a detailed account on the tools and technologies used to actually implement the project and the overall architecture of the program. The results section will come next describing the evaluation process and results followed by the analysis and conclusion.

II. Literature Survey

The study of eye movements began over hundred years ago [2]. However, information derived from those studies was mainly consumed by psychologists at that time to link eye tracking data to model cognitive processes. Scleral Contact Lens was one of the most precise eye movement measurement methods involving attaching a mechanical or optical reference object mounted on a contact lens which is then worn directly on the eye. Such early recordings used a plaster of paris ring attached directly to the cornea and through mechanical linkages to recording pens. This technique evolved to the use of modern contact lens to a which a mounting stalk is attached. Electro oculography [3] was the most widely applied eye movement recording method some 40 years ago (and still used today). It relies on measurements of the skin's electric potential differences, measured by electrodes placed around the eye. This technique measures eye movements relative to head position, and so is not generally suitable for point of regard measurements unless head position is also measured (e.g using a head tracker). Today, possibly the most widely applied eye movement technique, primarily used for point of regard measurements, is the method based on corneal reflection. In the advent of the computer era, eye tracking research found a new domain in Human Computer Interaction (HCI). As mentioned in the study [2], in the 1980's, research was carried out mainly on the interaction for differently abled users. Also, that time ushered in techniques for real time eye tracking as well. A good study on the current status of HCI has been carried out in [4] which points out two major usages of eye tracking in terms of HCI in the current context. One is in eye tracking as a "Research and Usability-Evaluation tool". It elaborates on the evaluation of eye tracking and movement data across areas of interest in the display. For example; a longer fixation on an area of interest would suggest that it would take awhile for the information in that area to process in the subjects brain; suggesting room for improvement. Another usage is tracking the eye "as an Input Device". [4] points out the ability for a person to interact with an interface based on eye tracking data. The user can register clicks based on his eye tracking input data. Another example points out the control of Virtual Reality environments, although it suggests using the user's gaze point to manipulate objects in the VR environment and trigger context-sensitive help. [5] points out that "the use of an eye tracker can increase the immersion of a video game and can significantly alter the gameplay experience." It details a scenario in which the users view orients to match where the user is looking at. [6] details a study which closely follows our project in which eye tracking data is used to apply depth of field effects to a 3D environment. [6] uses the players gaze point to compute the focal distance using an auto-focus zone

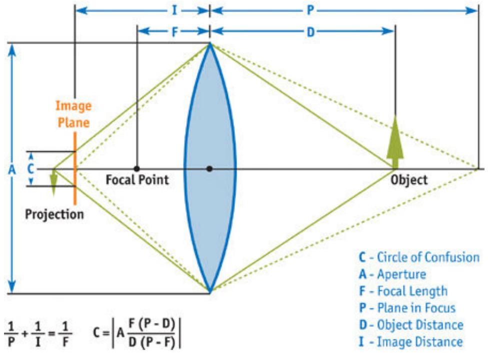


Fig. 1. Circle of Confusion

[7]. The user's gaze point is centered around this focus zone. Objects are given weights according to their importance and an average weighting technique is used to approximate which object would be in focus inside this focus zone. However a main drawback of this implementation is that it requires the user to keep his head resting motionless on a chin rest. In traditional computer graphics, a scene is rendered through a pinhole camera model. A pinhole model only lets in one ray of light from a specific point hence the whole scene appears in focus. [8] was the first study to propose the thin lens camera model which would enable a 3D scene to have camera effects like depth of field. Although it was computationally expensive, it gave good results at the time. This study spawned a lot of other research which would use this thin lens camera model to render realistic 3D images.

III. Methodology

The users gaze point was read giving a x and y coordinate of the screen. Using those coordinates, the object the user is looking at in the 3D simulation is judged. Based on the z-value of that object which corresponds to the focal plane in the 3D scene, the depth of field effect is added using post process effects. Depth of field is the effect in which objects within some range of distances in a scene appear in focus, and objects nearer or farther than this range appear out of focus. This happens in real life because objects which are focused correspond to a point in the retina and objects which are not in focus correspond to an area in the retina.

As illustrated in Figure 1, the object corresponds to a circular area (Image Plane) in the retina. This circle is called the Circle of Confusion (CoC). In typical computer graphics, a scene is not rendered using a lens. It is being done by letting light pass through a single point. Therefore, the Depth of Field (DoF) blur is only applied as an effect in most cases. The depth of field technique was implemented by following [9]. This is done by calculating a blur amount based on the approximated CoC. The CoC was represented by a variable size kernel filter. The filter size is computed per-pixel from the blurriness value of the center sample. The bluriness factor is the absolute distance from the focal plane normalised in the range from -1 to +1. A copy of the image was downsampled and blurred, while depth information was written to destination alpha in the image. Then, according to the depth information, it was blended with the original image according to the blurriness factor using the kernel filter.

Object distance was calculated by shooting a ray from the camera position towards the object the user is looking at such that the viewport projection of the ray intersection corresponds to the point the user is looking at. The z value of the intersecting point is the focal distance.

IV. Implementation

The estimation of the user's gaze point was initially done using ITU Gaze Tracker. However, due to lack of quality output in webcameras, that approach was abandoned. Therefore an open source alternative "OpenGazer" was used to approximate the user's gaze point. Opengazer opens a local UDP socket at port 20230 to which it sends the current estimates X and Y. Therefore a UDP listener was implemented to stream the output data into the main application. The network data was captured and formatted to obtain the X and Y coordinates.

The 3D engine was built using the object-oriented graphics engine library, Ogre3D. It presents an API to build one's own 3D application. Using Ogre3D's built-in functions, the object the user is looking at was judged. The selected object's coordinates was then passed to the shader program which implements the depth of field effect using the focal plane as the z coordinate of the said object.

```
Ogre::Ray depthRay =
mCamera->
getCameraToViewportRay(screenx,screeny);
RaySceneQuery* query = 0;
query = mSceneManager->
createRayQuery(depthRay);
query->setRay(depthRay);
RaySceneQueryResult& queryResult =
query->execute();
RaySceneQueryResult::iterator i =
queryResult.begin();
```

In Ogre3D, any change to be done to the scene is programmed inside the "frameStarted()" method. In this method, the current X and Y coordinates of the user's gaze point is read using a network port listener. Based on the coordinates, a ray is calculated originating from the camera's position and towards a location such that a viewport projection of the intersection point between the ray and an object would correspond to the X and Y coordinates just obtained. The Z coordinate of the intersection point is thought to be the focal distance and this value is passed to the shader program. A short shader code listing is below.

```
// compute texture coordinates
vec2 coordScene = gl_TexCoord[0].st +
(pixelSizeScene * poisson[i] * discRadiusScene);
vec2 coordBlur = gl_TexCoord[0].st +
(pixelSizeBlur * poisson[i] * discRadiusBlur);

// fetch depth
vec4 tapScene = texture2D(scene, coordScene);
float tapDepth = texture2D(depth, coordScene).r;
vec4 tapBlur = texture2D(blur, coordBlur);

// mix low and high res. based on blurriness
// put blurriness into [0, 1]
float blurAmount = abs(tapDepth * 2.0 - 1.0);
vec4 tap = mix(tapScene, tapBlur, blurAmount);
```

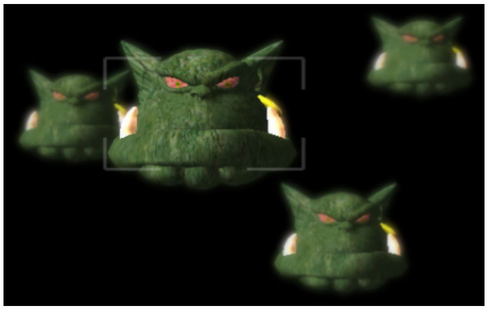


Fig. 2. Resulting Depth of Field effect

TABLE I. RESULTS OF EXPERIMENT

Player	Increased	Decreased	No Change
1	Yes	-	-
2	Yes	-	-
3	-	Yes	-
4	Yes	-	-
5	-	Yes	-
6	Yes	-	-
7	Yes	-	-
8	-	Yes	-
9	-	Yes	-
10	Yes	-	-

```
// "smart" blur ignores taps that
//are closer than the center tap and in focus
float factor = (tapDepth >= centerDepth)
? 1.0 : abs(tapDepth * 2.0 - 1.0);
```

V. RESULTS

The results of implementing the depth of field effect using [9] is shown in Figure 2. In the figure, the amount of blur applied to an object has increased as its depth increases.

The evaluation method for this project is based on a qualitative method where the user was subjected to a scenario which does not use the DoF effect versus a scenario which does apply the DoF effect. The user's output was recorded to be one of the following choices.

- Increased immersion
- Decreased immersion
- No change in immersion

A set of 10 participants of the same age were used in the experiment. They were subjected to observe the visual simulation without DoF for 15 seconds, then with eye tracking and DoF enabled, they were subjected to observe the simulation for another 15 seconds. Their responses are recorded I.

VI. ANALYSIS

Upon analysis of the results, 60% of users reported that the depth of field technique based on eye tracking improved their level of immersion in the visual simulation. The rest of the users reported that the technique decreased immersion. This could be attributed to the eye tracking accuracy. When the returned gaze point is not what the user is actually looking at, the user's gaze point location might appear blurred, since the calculated depth is erroneous. This could decrease his sense of realism and immersion in the simulation.

VII. CONCLUSION

In conclusion, a technique to improve a user's sense of immersion in a virtual environment was outlined in this paper. It relied on tracking a user's gaze point on a 2D monitor and using it to change the depth of field in the simulation. The usefulness of the technique however, is heavly dependent on the accuracy of the eye tracker as discussed in the analysis section. Although the user is not constrained to keep his stationary, severe head movement resulted in poor eye tracking and rendered this technique unusable. Overall, if an accurate eye tracking mechanism exists, the underlying methods in this paper can be certainly employed to improve a user's immersion in a 3D simulation.

REFERENCES

[1] Z. Zhang, "Microsoft kinect sensor and its effect," *Multimedia, IEEE*, vol. 19, no. 2, pp. 4–10, 2012.

[2] R. Jacob and K. Karn, "Eye tracking in human-computer interaction and usability research: Ready to deliver the promises," *Mind*, vol. 2, no. 3, p. 4, 2003.

[3] K. Hyoki, M. Shigeta, N. Tsuno, Y. Kawamuro, and T. Kinoshita, "Quantitative electro-oculography and electroencephalography as indices of alertness," *Electroencephalography and clinical Neurophysiology*, vol. 106, no. 3, pp. 213–219, 1998.

[4] A. Poole and L. J. Ball, "Eye tracking in human-computer interaction and usability research: Current status and future," in *Prospects, Chapter in C. Ghaoui (Ed.): Encyclopedia of Human-Computer Interaction. Pennsylvania: Idea Group, Inc*, 2005.

[5] J. D. Smith and T. C. N. Graham, "Use of eye movements for video game control," in *In Proceedings of the 2006 ACM SIGCHI International Conference on Advances in Computer Entertainment Technology (p. 20.* ACM Press, 2006.

[6] S. Hillaire, A. Lcuyer, R. Cozot, and G. Casiez, "Using an eye-tracking system to improve camera motions and depth-of-field blur effects in virtual environments," in *Virtual Reality Conference, 2008. VR'08. IEEE.* IEEE, 2008, pp. 47–50.

[7] S. Hillaire, A. Lécuyer, R. Cozot, and G. Casiez, "Depth-of-field blur effects for first-person navigation in virtual environments," *Computer Graphics and Applications, IEEE*, vol. 28, no. 6, pp. 47–55, 2008.

[8] M. Potmesil and I. Chakravarty, "Synthetic image generation with a lens and aperture camera model," *ACM Trans. Graph.*, vol. 1, no. 2, pp. 85–108, 1982.

[9] G. Riguer, N. Tatarchuk, and J. Isidoro, "Real-time depth of field simulation," *ShaderX2: Shader Programming Tips and Tricks with DirectX*, vol. 9, pp. 529–556, 2003.

Optimizing Power Usage in a Single Chip Cloud Computer

A.S. Gomez[1], L.S. Devamanthri[1], I. Nawinne[2] and R. G. Ragel[1]

[1]Department of Computer Engineering, Faculty of Engineering,
University of Peradeniya, Sri Lanka

[2]School of Computer Science and Engineering,
University of New South Wales, Australia

***Abstract* — Number of cores integrated to a chip is increasing in each computer generation. Intel Single-chip Cloud Computer (SCC) is a flexible multi core research platform designed to run parallel programs. It supports power management in the cores. It is an implementation of cluster based architecture. Therefore, SCC cores have both shared memory communication and message passing. Power management is an important area in multicore computer architecture research as energy consumption is always a concern. In this paper, we show how energy saving can be achieved on SCC that runs a bare metal boot loader (without an operating system) with a producer-consumer type of application.**

I. Introduction

Improving processor clock speed is a technique to increase the CPU performance. However, it affected negatively on energy efficiency. It always requires high power to operate on higher frequencies. This leads to a decrease in performance per watt. Due to these reasons, in the recent past, the computer architecture has been pushed forward to multicore architecture as the only solution to make effective use of the large number of transistors available. A key architectural challenge of multicore design is how to support increasing parallelism and scale performance, while being efficient in power and energy consumption.

Intel's 'Single-chip Cloud Computer' (SCC) [1] was introduced to overcome these challenges. This chip has the ability to schedule process execution on cores and manage voltage and frequency. However, it needs to have a reliable algorithm for scheduling for power and managing the frequency while running tasks in parallel.

Early researches on SCC have revealed power break down measurements, and comparison of its power consumption measurements for many state of the art CPUs. By optimizing voltage of cores according to tasks executions, frequency scaling can be used as a straight forward power management technique. Implementing such a technique and running it on SCC without a standard OS support would yield clear results on optimizing power usage of SCC.

II. Background

A. Overview of SCC

The SCC is the second generation processor design that resulted from Intel's Tera-Scale research. The SCC has 24 tiles and two cores per tile. The SCC core is a full IA P54C core and hence can support the compilers and OS technology required for full application programming.

B. Overview of tile and core

The SCC has 24 tiles, 48 cores, two cores to a tile. Each core has an L1 and L2 cache. The L1 caches are on the core; the L2 caches are on the tile next to the core. Each core has a 16KB, L1 instruction cache and a 16KB, L1 data cache. Each tile's L2 cache is 256KB.

C. Power and frequency management

SCC power controller (called the VRC for Voltage Regulator Controller) provides the capability of changing the voltage and frequency of the cores as well as other parts of the die. This capability supports research on power-aware applications. The SCC has seven voltage domains and 24 frequency domains. A program running on a core can change the voltage for all the members of a voltage domain; it can also change the frequency for all the members of a frequency domain. There are six voltage domains comprising four tiles of two cores each in a 2x2 array. The seventh voltage domain is the entire set of tiles. There are 24 frequency domains, one for each tile. Fig. 1 illustrates the voltage and frequency domains on the SCC.

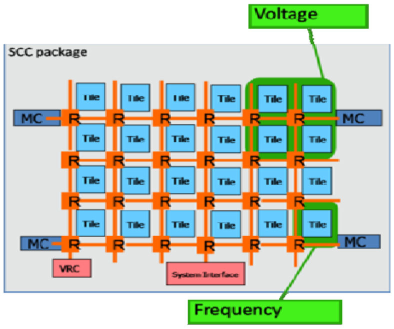


Fig. 1. Power and frequency domains [1]

D. SCC communication environment (RCCE)

The SCC architecture supports a variety of parallel programming models. At its foundation, however, the SCC processor is a message passing chip. The cores may interact through shared memory, but with the total lack of cache coherence between cores, the most natural and efficient programming models for this chip build on the ability to send messages between cores. The message passing library developed for this chip and used to analyse workloads as the chip was designed is called RCCE [2][3][8].

III. Related Work

There were several projects relevant to SCC power management. Those works characterize the performance behaviour of the chip with various power settings, mappings of processes/cores to memory controllers [12] and use various scalable applications to quantitatively compare and analyse the performance, power consumption and energy efficiency of different cutting-edge platforms that differ in architectural build [13]. These platforms include Intel's Single-Chip Cloud Computer.

IV. Implementation

The SCC architecture allows programmers to change the voltage and frequency of core domains by assigning appropriate control values for designated registers. However, voltage and frequency are not independent. Because of this reason, varying voltage and frequency should be done under safe conditions. RCCE library provides a safe power management API. This API is useful for changing voltage and frequency under safe conditions.

A. *Programming model*

The application would be run on SCC bare-metal. BareMichael [11] bare-metal boot loader will be used to make the SCC cores ready to run the application. RCCE message parsing library will be used to implement the scheduler program. Application will be emulated on OpenMP many core architecture.

B. *Development*

Bare-metal framework is the boot loader OS to use in SCC running programs without an OS. BareMichael is a minimalistic, open source bare-metal framework for launching C and/or assembly code on the SCC. This has only some amount of groundwork laid to allow one to design and add in these or other features as needed.

BareMetal_iAPI [5] is an improved version of BareMichael framework which is used to run RCCE applications. BareMetal_iAPI consist an example based on Consumer – Producer paradigm. The Consumer-Producer application is separated into two tasks, namely the Consumer task and the Producer task.

RCCE utility API contains core utility functions to initialize RCCE, RCCE_finalize() to finalize and shutdown the RCCE, RCCE_num_use() to return the number of units of execution etc. We used the above functions and checked for the output. After testing the RCCE utility functions, we changed the code to use the RCCE power management API.

Power management API is the same for both basic and more complex interfaces in RCCE. Since this API is not used frequently, it is not included in the RCCE build process.

To make the power management API available, we first compiled the RCCE and later compiled the RCCE application with PWRMGMT=1 in the makefile. Power management API also contains few more functions to manage the power domains in the SCC.

C. *Voltage and Frequency interrelation.*

The frequencies of the SCC cores are related to the operating voltage. Maximum frequency of a core is 1.6GHz. Intel has provided a way to decrease the frequency into 16 values. This is done by using a divider value. The divider value, when the core is running in the maximum frequency, is 1 and the frequency is 1.6GHz and when the divider is 2, the frequency is 800MHz. Likewise, the divider value can be set up to 16.

In the SCC voltage domains, by changing the value of the RPC register, we can change the value of the voltage. There are 5 main voltage levels and for each of these values there is a maximum frequency value that can be reached. Table I shows the voltage levels and their respective maximum frequencies.

Table I. Voltage Levels and the Maximum Frequency

Voltage Level	Voltage (volts)	Maximum Frequency (MHz)
0	0.7	460
1	0.8	598
2	0.9	644
3	1.0	748
4	1.1	875

Based on the frequency divider explained earlier and the voltage levels shown in Table I, a summary can be derived as in Table II.

Table II. Voltage Levels and Frequency Level Matching

Tile Frequency (MHz)	RCCE Frequency Divider	RCCE Voltage Level
800	2	4
533	3	1
400	4	0
320	5	0
266	6	0
228	7	0
200	8	0
178	9	0
160	10	0
145	11	0
133	12	0
123	13	0
114	14	0
106	15	0
100	16	0

D. *Implementation of voltage scheduling for Consumer – Producer* [7]

Due to restrictions in RCCE power API, power scheduling implementation should be executed on core 0, if we use first voltage domain. Consumer and Producer were made to run on core 01 and 00. Therefore, the varying of voltage takes place in the Producer algorithm.

E. *Checking states of cores.*

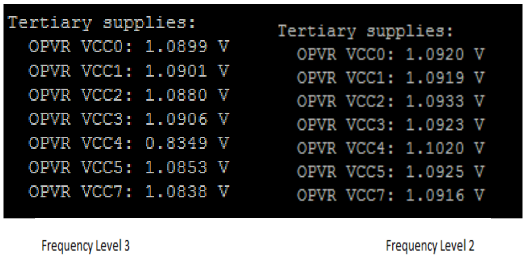
The command **'sccBmc –c status**'[6] returns the status of SCC board. The BMC is the board management controller. It is responsible for example downloading the bit stream into the FPGA. The BMC is basically a tiny ARM based embedded system that can bootstrap the whole SCC system by configuring the FPGA.

V. Testing and Results

After implementation of Consumer–Producer application and power management algorithm, few tests were performed to verify the correctness of the algorithm.

A. *Running image on SCC* [9][10]

With the usual Baremetal framework commands, image for application was build and run. With respects to implementation, status of BMC was recorded (Fig. 2).



```
Tertiary supplies:                Tertiary supplies:
  OPVR VCC0: 1.0899 V               OPVR VCC0: 1.0920 V
  OPVR VCC1: 1.0901 V               OPVR VCC1: 1.0919 V
  OPVR VCC2: 1.0880 V               OPVR VCC2: 1.0933 V
  OPVR VCC3: 1.0906 V               OPVR VCC3: 1.0923 V
  OPVR VCC4: 0.8349 V               OPVR VCC4: 1.1020 V
  OPVR VCC5: 1.0853 V               OPVR VCC5: 1.0925 V
  OPVR VCC7: 1.0838 V               OPVR VCC7: 1.0916 V
```

Fig. 2. Output status for frequency levels

B. *Calculations*

Expression for power and energy of a core [14]

Energy of core (E) $\propto$ (core voltage) ^{2}

Power of core (P) $\propto$ E * (core frequency)

Assuming the domains have same characteristic of cores:

Without power management:

Working core domain voltage = 1.1V

Non-working core domains (5) = 1.0V

$E_1 \propto [(1.0V)^2 *5] + (1.1V)^2$

$P_1 \propto [(1.0V)^2 *5] * 0 \text{ MHz} + (1.1V)^2 * 533 \text{ MHz}$

With power management:

Working core domain voltage = 0.8V

Non-working core domains (5) = 0.7V

$E_2 \propto [(0.7V)^2 *5] + (0.8V)^2$

$P_2 \propto [(0.7V)^2 *5] * 0 \text{ MHz} + (0.8V)^2 * 533 \text{ MHz}$

Therefore, scheduling efficiencies:

For energy: E_1/E_2 = 2.0

For Power: P_1 / P_2 = 1.9

C. *Results*

Table III and Table IV summarise the results.

TABLE III. Actual And Theoretical Voltages

Frequency Level	Actual Voltage /V	Theoretical Voltage/V
3	0.8349	0.8
2	1.1020	1.1

TABLE IV. Power and Energy Efficiency

Frequency Level	Energy E_1/E_2	Power P_1 / P_2
3	2.0	1.9
2	1.7	1.0

VI. Conclusion

In this paper, we have shown how energy saving can be achieved on SCC that runs a bare metal boot loader (without an operating system) with a producer-consumer type of application. The preliminary results show about a two-fold energy saving with our algorithm.

References

[1] The SCC Platform Overview, Revision 0.80

[2] The SCC Programmer's Guide, Revision 1.0

[3] (SSG), Tim Mattson (IL) and Rob van der Wijngaart. "RCCE: a Small Library for Many-Core *Communication." Jan 10, 2011.*

[4] Philipp Gschwandtner, Thomas Fahringer, Radu Prodan. "Performance Analysis and Benchmarking of the Intel SCC"

[5] Carsten Clauss, Stefan Lankes, Thomas Bemmerl, Jacek Galowicz, and Simon Pickartz, "iRCCE: A Non-blocking Communication Extension to the RCCE Communication Library for the Intel Single-Chip Cloud Computer", November 4, 2011.

[6] Using the RCCE Power Management Calls revision 1.1

[7] How to Modify a Core's Voltage by Writing the SCC RPC Register revision 1.1

[8] John-Nicholas Furst Ayse K. Coskun, "Performance and Power Analysis of RCCE Message Passing on the Intel Single-Chip Cloud Computer"

[9] Tim Mattson, "The Future of Many Core Computing: A tale of two processors", Intel Labs, January 2010

[10] Tim Mattson, Intel Labs, "Many core processors at Intel: lessons learned and a view of the future"

[11] Michael Ziwisky, Dennis Brylow, "BareMichael: A Minimalistic Bare-metal Framework for the Intel SCC"

[12] Philipp Gschwandtner, Thomas Fahringer, Radu Prodan, Institute of Computer Science, University of Innsbruck Technikerstrasse 21a, Innsbruck, Austria, "Performance Analysis and Benchmarking of the Intel SCC"

[13] Ehsan Totoni, Babak Behzad, Swapnil Ghike, Josep Torrellas, Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA, "Comparing the Power and Performance of Intel's SCC to State-of-the-Art CPUs and GPUs"

[14] Jjohn l. Hennessy, Davic a. patterson, "Computer Architecture: A Quantitative Approach, Fifth Edition". Morgan Kaufmann is an imprint of Elsevier , 225 Wyman Street, Waltham, MA 02451, USA. 2011

An Instruction Scheduler to Maximize both Data and Instruction Level Parallelism in a Multiprocessing System

R.S.M.S. Bandara, W.H.M.A.U. Kumara, B.S. Manuweera, S. Radhakrishnan and R. G. Ragel
Department of Computer Engineering, Faculty of Engineering,
University of Peradeniya, Sri Lanka

***Abstract*– This paper proposes a scheduler for DRMA (Dynamically Reconfigurable MPSoC Architecture), where we can configure the cores in a multicore system for parallel execution of tasks. Also DRMA is compared with the existing SIMD extension (MMX) targeted at multimedia applications. Unlike MMX, our scheduler deals with Instruction Level Parallelism and different word sizes. Our target is to prove that DRMA provides a performance improvement when compared with existing multimedia applications (MMX) and to enhance a compiler, which will efficiently deals with the vectorizable program structures typically found in multimedia applications. First, we had a detailed look at some standard benchmark applications, analyzed them manually and then automated the scheduler to show the performance improvement.**

***Keywords*— Scheduler; ILP; DLP; TLP; DRMA; MMX; SIMD**

I. Introduction

Multiprocessing systems are used in many places today with the growth of technology. Earlier it was only one processor in a system to do all the work and everything had to be done sequentially. But with the multiprocessing systems, performance of the computers increased with increased number of cores. Now there are multiprocessing systems with two, four, eight, etc. number of cores [13].

Since earlier programs used sequential execution, the problem of processing them in parallel was a need in order to increase the performance of the system. To do so, each core has to be assigned to a set of operations so that each core can execute them in parallel [5].

In a program, there can be different types of parallelism exploited such as instruction level parallelism (ILP) [7], data level parallelism (DLP) [6], loop level parallelism (LLP) [17], and task level parallelism (TLP) [15]. Here we consider mainly on ILP and DLP. Vectorization [4] comes under DLP that process data in parallel as vectors. SIMD (Single Instruction Multiple Data) architecture uses Vectorization [4] which is used in Multimedia Extensions. This increases the performance of the CPU significantly. However, it was only considered on DLP where only data is processed in parallel [10].

Our attempt is to perform both ILP and DLP in a system which does not have a multimedia extension. To do so the program needs to be modified in a way that all the cores can contribute to the execution of the program in parallel. It then gives higher performance to the system since it exploits both ILP and DLP. This can be achieved by using DRMA architecture.

DRMA, an ASIC based flexible MPSoC architecture, which can be configured as individual processors to execute separate tasks in parallel, can be combined together to execute single task with wide data widths or execute tasks with varying data widths [1].

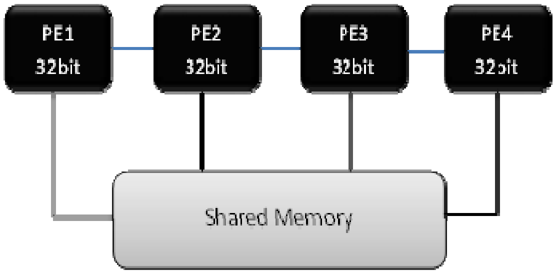


Fig. 1. Overview of the DRMA Architecture

Fig. 1 shows an overview picture of DRMA architecture system. Each processing element is connected with special bus and is connected to a shared memory. This special bus helps to transmit signals between processor elements. DRMA can be configured through instructions. So, it is capable of combining multiple processors at runtime to create various configurations. Here, we search for independent instruction and data to process in parallel and arrange them in a way that it utilizes all the cores for the execution.

Section 2 of this paper describes the previous work done up to now by others. Also the contribution part along with the previous work explains how the project contributes to the world and in what way. Section 3 describes the proposed work explaining the concept and the algorithms used. In the implementation part, the things accomplished up to now is described. What is implemented for instruction scheduling and how it is done are clarified. Results and Analysis part contains the results and in the last section we conclude the paper.

II. Previous Work And The Present Contribution

As a solution to the problem of speeding up the program execution time on executing a sequential program in parallel, parallel computing [14] came into practice. It is a form of computation in which many calculations are carried out simultaneously. Parallel computing uses multiple processing [13] elements simultaneously to solve problems. This is accomplished by breaking large problems into independent parts so that each processing elements can execute its parts of the algorithm simultaneously with others.

MMX [8] is the technology which is currently used to make the programs faster. MMX uses Single Instruction Multiple Data architecture [11]. MMX is proposed by Intel and there are several products from different companies based on MMX like Altivec [3] from Motorola and 3DNow [2] from AMD.

In our project, we consider not only DLP but also ILP. Therefore, if there are no DLPs left in the code, still ILP can be exploited, resulting in an efficient program execution.

III. Proposed Work

First task in this project is to verify that using DRMA to run required program we can get better results. To do so, we manually analyse three benchmark applications (AES128, ADPCM and FIR). Instruction Level Parallelism and Data Level Parallelism are applied manually to each of those applications and get the speed of the each program running in DRMA. For this task we use *valgrind* as the profiler for calculations in the program execution.

In the next phase, we implement an instruction scheduler for DRMA which has the functionality of vectorization and Instruction Level Parallelism. Fig. 2 shows an overview of our proposed concept. .

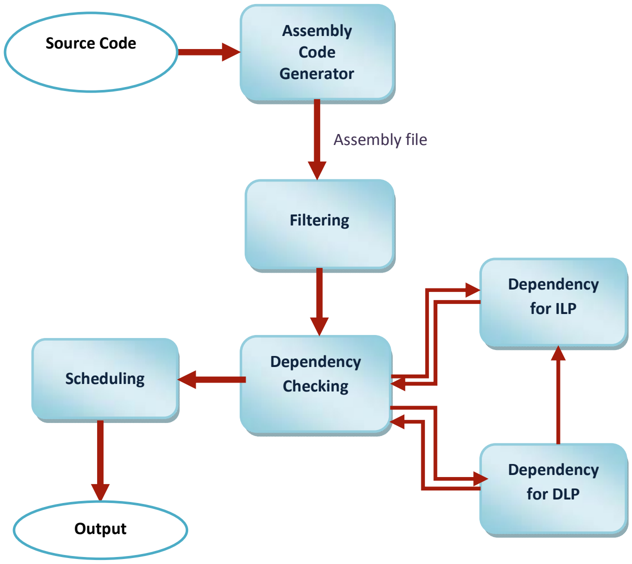


Fig. 2. Overview of Proposed Work

First of all, we did an analysis on what we are going to do. It was a manual code analysis on some benchmark applications. After the results of the analysis were obtained, the decision was made on doing this in the following way.

First we convert the source code into an assembly file and then to recognize each instruction by filtering the code. By filtering, we expect to get a data structure which contains all the data inside the assembly code in a manner that dependency checking can be done easily. So, all the register names, opcodes etc. are separated clearly for further use. This structure is used to find dependencies between instructions and data.

Then dependency checking should be done to identify ILPs and DLPs. As we consider both ILP and DLP, we hope to give priority to DLP. After identifying DLPs, ILPs are identified. Thereafter, the instruction scheduling is performed.

IV. Implementation

In the implementation of scheduler, the language used is Python and the scheduling is done for MIPS assembly architecture. The work on implementing scheduler is carried out according to the procedure shown in Fig. 2.

First the MIPS assembly code is generated using SimpleScalar [9] GCC compiler and then it is read by the program that is being generated for scheduling. While reading it line by line, the code is filtered to separate instructions, comments, directives etc. for further use. After filtering all the data in the assembly code, they are stored in lists that combined together.

Filtering is done thereafter. Filtering recognizes whether the line contains an instruction, directive etc. is by the way it is written. Likewise, all the functions are recognized and instructions and labels inside that function are stored under the function in a list.

Filtered output is used to check dependency between instructions inside the lists. Dependency checking is done using two instructions at a time. First check for dependency for ILP and while doing that, if a loop is recognized, see whether those iterations are independent to perform DLP. If not, instructions inside the loop are checked for ILP.

After the above mentioned process, scheduling is done. Scheduler decides which core is going to have instructions to execute and which is going to be idle. To do that, a calculation on instructions is used.

Finally, as output, four assembly files are generated to go to four cores in the system.

V. Results and Analysis

The analysis was done using four Benchmark applications. A Profiler was used to get the cycle estimation for each application was Valgrind [15]. The following table shows the initial clock cycles and final clock cycles after performing all the possible ILPs and DLPs.

TABLE I. FINAL RESULTS

Program Name	Initial Clock Cycles	Optimized Clock Cycles	Speed up
AES128	12923	6645.5	1.94
ADPCM	2260347660	2259868495.5	1.00
FIR	103918	37106	2.80

In the above results, CRC32 was not included since it does not have any DLP's and only has ILP's.

To speed up calculations of AES128 and ADPCM, only DLP was performed and in FIR calculations, both ILP and DLP were included. FIR contained more places than AES128 and ADPCM where ILP can be applied. In AES128 and ADPCM, no ILPs were significantly seen in the high level code. The differences are highlighted in the table showing that FIR gives 2.8 speeds up and it is relatively higher than the rest of the speed up values. The reason is it uses both the DLP and ILP with assumptions while AES128 and ADPCM use only DLP. ILP in FIR was considered assuming that each line will produce one instruction in assembly level and therefore, the values may not as accurate as in the real situation because ILP in high level code is quite difficult.

So it is clear that in assembly level, we can maximize the speed up since there are both ILP and DLP in assembly codes.

VI. CONCLUSION

It might not be possible to exploit significant amount of ILPs and DLPs from each and every program we encounter. However, as most of the programs are available for ILP this can be used for almost all the programs to increase performance. Our scheduler has the functionality to recognize both ILP and DLP. Therfore, it will decrease the execution time of almost all the programs. This program uses most of the functionalities of DRMA except processing multiple sized words. This program can be further increased if we can find the places where different sized words can be processed to optimize the code execution.

REFERENCES

[1]. Lawrance Zhang, Jude Angelo Ambrose, Jorgen Peddersen, Sri Parameswaran, Roshan Ragel, Swarnalatha Radhakrishnan, and Kewal K. Saluja. 2013. DRMA: dynamically reconfigurable MPSoC architecture. In Proceedings of the 23rd ACM international conference on Great lakes symposium on VLSI (GLSVLSI '13). ACM, New York, NY, USA, 239-244.

[2]. 3DNow!^{TM} Technology [online]. Available at: http://www.amd.com/us/products/technologies/3dnow/Pages/3dnow.aspx [Accessed: 02.12.2012]

[3]. AltiVec [online]. Available at: http://www.en.wikipedia.org/wiki/AltiVec [Accessed: 02.12.2012]

[4]. Auto-vectorization in GCC [online]. Available at: http://gcc.gnu.org/projects/tree-ssa/vectorization.html [Accessed: 01.12.2012]

[5]. Corbal, J., Valero, M., Espasa, R., Explioting a New Level of DLP in Multimedia Applications, Department of Computer Architecture, University of Politecnica de Catalunya, Spain.

[6]. Data parallelism [online]. Available at: http://en.wikipedia.org/wiki/Data_parallelism [Accessed: 02.12.2012]

[7]. Instruction-level Parallelism [online]. Available at: http://www.en.wikipedia.org/wiki/**Instruction-level_parallelism** [Accessed: 02.12.2012]

[8]. Intel MMX for Multimedia PC's [online]. Available at: http://ewh.ieee.org/r10/bombay/news1/4.htm [Accessed: 01.12.2012]

[9]. Introduction to SimpleScalar [online]. Available at: http://www.ecs.umass.edu/ece/koren/architecture/Simplescalar/SimpleScalar_introduction.htm [Accessed: 02.01.2012]

[10]. Mehrara, M., Jablin, T., Upton, D., August, D., Hazelwood, K., Mahlke, S., 2009, Multicore Compilation Strategies and Challenges, *IEEE Signal Processing Magazine* November 2009.

[11]. MMX (instruction set) [online]. Available at: http://en.wikipedia.org/wiki/MMX_(instruction_set) [Accessed: 01.12.2012]

[12]. Multi-core [online]. Available at: http://en.wikipedia.org/wiki/Multi-core [Accessed: 01.12.2012]

[13]. Multiprocessing [online]. Available at: http://en.wikipedia.org/wiki/Multiprocessing [Accessed: 01.12.2012]

[14]. Parallel Computing [online]. Available at: http://www.en.wikipedia.org/wiki/**Parallel_computing** [Accessed: 01.12.2012]

[15]. Task Parallelism [online]. Available at: http://en.wikipedia.org/wiki/Task_parallelism [Accessed: 02.12.2012]

[16]. Valgrind [online]. Available at: http://www.valgrind.org [Accessed: 03.12.2012]

[17]. What is loop-level parallelism [online]. Available at: http://insidehpc.com/2006/03/11/what-is-loop-level-parallelism/ [Accessed at: 02.12.2012]

String Matching with Multicore CPUs: Accelerating the Performance of the Aho-Corasick Algorithm

S. Arudchutha, T. Nishanthy and R.G. Ragel
Department of Computer Engineering, Faculty of Engineering,
University of Peradeniya, Sri Lanka

Abstract – **Multiple string matching is known as locating all occurrences of a finite number of keywords in an input arbitrary string. The use of multiple pattern matching is very significant in genomics where the algorithms are frequently used to locate gene or peptide sequence patterns in DNA. Huge amount of data (particularly keywords or patters) in the form of strings has to be handled in such bio-computing applications. Therefore, performance of multiple string matching algorithms has to be improved. Multicore architectures provide better performance by parallelizing the multiple string matching algorithms. Aho-Corasick is one of the commonly used exact multiple string matching algorithms. The focus of this paper is the acceleration of Aho-Corasick algorithm through a multicore CPU based software implementation.**

Keywords-Aho-Corasick; Multicore processor; POSIX threads

I. Introduction

Multiple string matching algorithms are used for locating all occurrences of a finite number of keywords in an arbitrary string. Multiple string matching is commonly required in information retrieval, text editing applications, spam filters and virus scanners. Although there are so many applications for multiple string matching process, string set matching has a vital importance in computational biology in places like DNA synthesis. Huge amount of data in the form of strings are being handled in bio-computing applications. Therefore, the performance of multiple strings matching algorithms has to be increased. In the current market, desktop and laptop computer architectures have evolved towards multicore. Aho-Corasick is a commonly used multiple string matching algorithm. What we are focusing on in this paper is a multicore CPU based software implementation for accelerating Aho-Corasick and improves the performance of existing implementation of Aho-Corasick on multicore.

The rest of the paper is organized as follows: In Section II we present the related work and in Section III we present the background details. In Section IV and Section V, we present the implementation and results. In Section VI we conclude the paper.

II. Background

A. The Aho-Corasick Algorithm

Aho-Corasick algorithm has been widely used for string matching due to its advantage of matching multiple patterns in a single pass. Mainly the algorithm consists of two steps. First step is constructing a string-matching machine from the given set of keywords. The second step is processing the input string in a single pass using the string matching machine. Fig. 1 shows the state machine of the four patterns, "AB", "ABG", "BEDE", and "EF". In Figure1, the solid lines represent the valid transitions whereas the dotted lines represent the failure transitions.

The failure transitions are used to back-track the state machine to recognize patterns in different locations. Given a current state and an input character, the Aho-Corasick machine first checks whether there is a valid transition for the input character; otherwise, the machine jumps to the next state where the failure transition points. Then, the machine regards the same input character until the character causes a valid transition. For example, consider an input stream which contains a substring "ABEDE". The Aho-Corasick state machine first traverses from state 0 to state 1 and then to state 2 which is the final state of pattern "AB". Because state 2 has no valid transition for the input "E," the AC state machine first takes a failure transition to state 4 and then regards the same input "E" leading to state 5. Finally, the AC state machine reaches state7 which is the final state of pattern "BEDE".

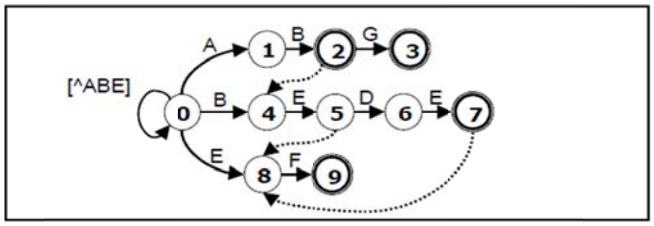


Fig. 1. Aho-Corasick state machine for patterns "AB", "ABG", "BEDE", and "EF" [2]

B. *Parallel Failure-less Aho-Corasick Algorithm*

In Parallel Failure-less Aho-Corasick (PFAC), for each byte of an input stream, a thread is allocated to identify any pattern starting at the thread starting location. The idea of allocating each byte of an input stream to a thread to identify any pattern starting at the thread starting location has an important implication on the efficiency. First, in the conventional state machine, the failure transitions are used to back-track the state machine to identify the patterns starting at any location of an input stream. Since in the PFAC algorithm, a thread only concerns the pattern starting at a particular location, the threads of PFAC do not need to back track the state machine. Therefore, the failure transitions of the state machine can all be removed. Fig. 2 shows the diagram of the PFAC which allocates each byte of an input stream a thread to traverse the state machine.

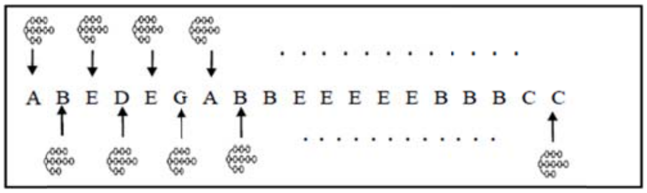


Fig. **2.** PFAC algorithm which allocates each byte of an input stream a thread to traverse the state machine [2]

III. Related Work

As it is necessary to derive efficient string matching algorithms, a large number of researches are being carried out in this area. Here we present two research projects carried out on string matching algorithms.

One of the researches [1] was done to test the hypothesis that multi-core architectures should provide better performance in this kind of computation, but still it would depend on the algorithm selected as well as the programming model being utilized. They explored the possibility of parallel execution through two different architectures: (1) splitting the input into separate chunks and processing them for the string matching in separate threads and (2) splitting the pattern file and developing different pattern machines in separate threads and passing the input to each machine.

In the Parallel Failure less–AC algorithm (PFAC) [2], the authors have tried to overcome the problems in the direct implementation of Aho-Corasick algorithm on GPU and increase the throughput by increasing the parallelism. In this method, they are removing the failure transitions so that there is no need to backtrack the state machine used, reducing the complexity of the algorithm. They have achieved this by allocating each byte of an input stream, a GPU thread to identify any virus pattern starting at the thread starting location.

In this paper, we are proposing a different method for parallel execution to Aho-Corasick algorithm by improving the performance of the existing implementation [1] using Parallel Failure less-AC algorithm [2] techniques. This method takes into consideration the fact that bio-computing applications use large pattern sets.

IV. DESIGN AND IMPLEMENTATION

A. Thread Assignment Methodology

Our design in this experiment is based on PFAC with Aho-Corasick algorithm on a multicore machine as opposed to a GPU. In PFAC, because of limited number of threads available on a multicore processor, we could not allocate each byte of an input stream a thread. The first step is to divide the pattern file into a given number of threads as explained in [1] and then the same numbers of threads will be created. Each different pattern chunk will be passed into a separate thread. We allocate first byte of an input stream to a thread. When we consider thread, when it terminates, it will be allocated to second byte of an input stream. Likewise each thread identifies any pattern starting at the thread starting locations.

B. Implementation Details

Initially, we found a suitable implementation of Aho-Corasick algorithm. We used a module called “Multifast-v0.6.2” for our implementation. This module is implemented in C programming language. It provides an easy to understand codebase and clear implementation. Basically, *multifast* takes an array of strings as a set of finite pattern strings and input string against which the patterns would be matched and output gives the details on the pattern matched and its position in the input string. Additionally *multifast* takes both the set of patterns and the input string from reading the files therefore both set of strings can be given from files. The approaches we took were to achieve our design via following the steps. Multifast-v0.6.2 contains failure link transitions in state machines for backtracking patterns starting at different locations. In our design, we rebuilt the state machines without the failure link transitions and changed the searching method with respect to the failure less transitions. We achieved parallelism via Pthreads. We used C programming language and POSIX Pthread library for thread handling.

V. EXPERIMENTAL EVALUATION

We analyzed the results mainly under the following aspects:

- Throughput: the number of pattern matches per unit time.
- Scalability: the effect of changing the input files size used in our implementation.

$$Throughput = \frac{Total\ number\ of\ patterns\ matched}{The\ average\ time\ taken} \qquad (1)$$

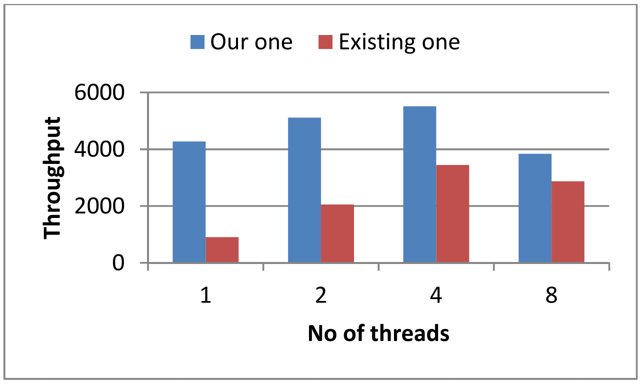


Fig. 3. Variation of throughput with number of threads for 20MB patter file

The experiments were carried out by changing the number of patterns from half a million (10MB) to one million (20MB) on a multicore machine that supports 4 threads. Fig. 3 shows the variations of throughput for different number of threads for half a million patterns and Fig. 3 depicts the same variables for one million patterns. In both cases, the input file size is kept at constant.

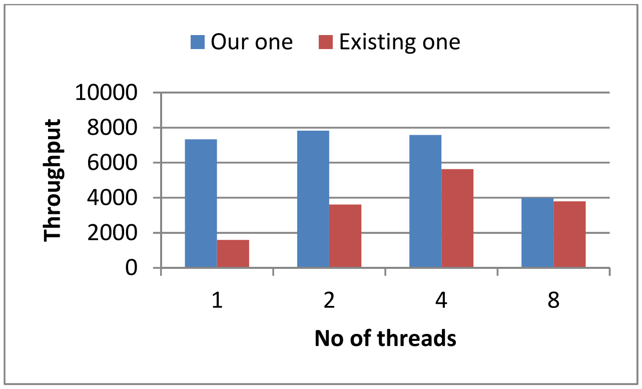


Fig. 4. Variation of throughput with number of threads for 10MB patter file

From the results, it can be seen that out method is performing better (with higher throughput), compared to the existing solution.

In addition, the throughput gain gets better with the increasing patter size.

VI. CONCLUSION

Large amount of data in the form of strings is needed to be handled in bio computing applications. Therefore, performance of multiple string matching algorithms has to be improved. In this paper, we have presented a methodology to achieve improved performance of Aho-Corasick pattern matching algorithm on a multi-core CPU through parallel manipulation of pattern files using POSIX thread utility. With our approach, we could conclude that our implementation of Aho-Corasick algorithm is performing better compared to the one presented in [1].

REFERENCES

[1] Damayanthi Herath, Chathurika Lakmali, Roshan Ragel, **Accelerating String Matching for Bio-computing Applications on Multi-core CPUs**, *Seventh International Conference on Industrial and Information Systems (ICIIS 2012)*, August 2012, Chennai, India

[2] Lin, Cheng-Hung. **Accelerating String Matching Using Multi-Threaded Algorithmon GPU,***Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE.* Nat. Taiwan Normal Univ., Taipei, Taiwan, 2010. 6-10

[3] *Charalampos S. Kouzinopoulos, Panagiotis D. Michailidis and Konstantinos G. Margaritis* . **Parallel Processing of Multiple Pattern Matching**, *ISBN 978-953-307-875-5, September 12, 2011*

[4] Daniele, S. P., Oreste, V., & Fabrizio, P. **High-Speed String Searching against Large Dictionaries**, 22nd IEEE International Symposium on Parallel and Distributed Processing, IPDPS 2008, Miami, Florida USA, April 14-18, 2008. IEEE 2008

Model-Based Design and Testing (H-I-L) of the Speed Control Sub-system of an Adaptive Cruise Control in an Automobile

W.D.D.N. Jayasundara[1] , L.P.S. Kumar[2] , S.D. Dewasurendra[1] , L. Samaranayake[2]
[1]Department of Computer Engineering, Faculty of Engineering,
University of Peradeniya, Sri Lanka
[2]Department of Electrical and Electronic Engineering, Faculty of Engineering,
University of Peradeniya, Sri Lanka

***Abstract*— In a conventional vehicle, the driver controls the speed by applying braking and acceleration according to driving conditions and hence, it can be a difficult and risky task when driving under heavy traffic conditions. Having an automated speed and braking control system is very useful for a driver to have a safe and comfortable journey. Adaptive Cruise Control (ACC) is such an automated system available in modern vehicles. However, ACC must be made error–free essentially. Our main objective has been to develop and validate a model-based design/testing procedure for a correct-by-design speed control sub-system of an ACC and draw general guidelines for other control modules. This would later evolve into an experimental platform. Currently available ACC systems can maintain a pre-set following distance from the preceding vehicle, control vehicle speed, etc. However, if the vehicle immediately behind the subject vehicle is too close and accelerating, the risk of an accident is still very high, because the trailing vehicles may not have enough time to react to a sudden braking action of the ACC. Incorporating this aspect into the controller has been another objective. The important issues of test coverage and compatibility of test results at different abstraction levels were addressed using Time Partition Testing (TPT) [2] to drive a XPC-Target [6] Hardware-in-the-Loop test on Matlab/ Simulink/ Stateflow. The system is tested at several testing levels using TPT test scripts before it is tested on the final hardware in order to identify the errors at early stages in the development cycle.**

***Keywords*— model-based design; model-based testing; adaptive cruise control; hardware-in-the-loop testing; speed control;**

I. Introduction

We have built a development platform for a correct-by-design speed-control sub-system (simplified) of an Adaptive Cruise Control (ACC) for automobiles using a Model-Based-Design (MBD) and Testing (MBT) strategy. Traditional ACC controls the vehicle speed by adjusting throttle position to maintain a speed set by the driver while maintaining a safe distance from the preceding vehicle. When a preceding vehicle is detected, the ACC controls acceleration and braking and maintains a set distance between vehicles without driver intervention. When there is no vehicle in front, it controls the vehicle speed by adjusting throttle position to maintain a speed set by the driver. ACC is expected to reduce the chance for accidents while providing driver comfort [1]. However, there is a problem here: if the vehicle immediately behind is too close and accelerating, the risk of an accident is still very high, because the trailing vehicles may not have enough time to react to a sudden braking action of the ACC. Incorporating this aspect into the controller maintaining the integrity of the speed-control has been another objective.

Model-based design (MBD) methodology builds a complex control system (modelling) using visual methods. For describing the system mathematically, system parameters, states and signals are identified and algebraic, logical, differential and difference equations are used to create the system model in a modelling environment like MATLAB Simulink/Stateflow. It is particularly suitable for designing complex reactive systems in industrial/bio-medical motion controls, avionics, aerospace, and automotive applications.

In Model-based testing (MBT), a test driver feeds the model with sets of inputs with expected outputs developed using the system model, to generate results that helps in detecting faults at an early stage in the development cycle and rearranging the model until the original system specifications are met. MBT is traditionally done at different levels, Model-in-the-Loop, Software-in-the-Loop, Circuit-in-the-Loop, Hardware-in-the-Loop, etc., and testers seem to specialise at their own testing levels, with dedicated test design processes and test models/platforms/rigs. However, in this approach, the results from different levels of testing may have to go through consistency checks, because they would have been based on different models of the system under test. In order to overcome this problem, the concept of Time Partition Testing (TPT) has been mooted [2]. The philosophy here is that the tests are all based on the same models that have been used in the model-based design process and hence, the test generation process remains consistent across testing levels by design. The tests themselves are organised on automata, with each state of an automaton corresponding to a feature to be tested. The state sequence corresponds to the temporal sequence of events corresponding to these features. Also, this provides means for establishing adequate test coverage.

In the current research, we are focus on Hardware-in-the-Loop (H-i-L) level of testing, but using the TPT techniques to facilitate consistent test development at other levels. In H-i-L level testing, the software runs on the final Electronic Control Unit (ECU), with the environment still being a simulated one. However, unlike in traditional simulation, we need real-time behaviour from the simulated environment to reveal faults in the low-level services of the ECU and I/O.

II. Litrature Survey and Research Methodology

A. Adaptive Cruise Control (ACC)

On average, at least one person dies in a crash. Automobile accidents injure at least 10 million people every year worldwide [4], two or three million of them seriously. Adaptive Cruise Control (ACC) could provide safety and comfort in driving. A good ACC system accelerates aggressively to the desired cruising speed without overshooting and maintains that speed with little deviation irrespective of the weight in the car or road inclination. Many automotive manufactures equip their products with ACC: Lexus, BMW, Mercedes Benz, Audi, Volvo, Acura, KIA. MBD methodology is used for designing safety-critical reactive systems like this.

B. Model-Based Design (MBD)

Model-Based Design (MBD) is a mathematical and visual methodology for addressing problems associated with designing embedded systems for safety and performance critical applications: aerospace, automotive, motion control, industrial process/equipment, etc. MBD is an efficient and cost effective way to develop products with improved quality and it also is reported to reduce the development time by 50% or more [8]. MBD paradigm is significantly different from the traditional design methodology. Rather than using complex structures and extensive software code, designers can use MBD to define models with advanced functional characteristics using continuous-time and discrete-time building blocks. These built models used with simulation tools can lead to rapid prototyping, software testing and verification. Implementation of the function on a Electrical Control Unit (ECU) is done by automatic production code generation.

Among several other tools developed for model-based design/testing of real-time control systems we opted for xPC Target for its' particular adaptability to perform H-i-L testing using Matlab/Simulink/Stateflow models and rapidly prototyping and implementing real-time control systems.

C. Model-Based Testing (MBT)

The model-based testing process (Fig. 1) describes the different activities within the automotive electronics development process from a testing point of view [5]. MBT complements model-based design with a method for systematic test definition. The starting point of the MBT process is the model, Simulink/Stateflow, for instance, of the function or controller to be developed (MBD). The control system designed in block diagram form in Simulink/Stateflow is compiled to an executable. It is then loaded onto a dedicated computer (Target-PC) for real-time execution. There are three major components in the xPC Target control system: The host PC running MATLAB, Simulink, and the Real-Time Workshop and xPC Target toolboxes, the Target-PC running the xPC Target real-time kernal, and the plant. A block diagram representing the controller is created on the host PC in Simulink. Special Simulink blocks are inserted into the diagram that represents different I/O hardware in the plant: special xPC Target blocks which send or read physical control or sensor signals from the I/O hardware. In order to perform automated and systematic testing, the constant and signal generator blocks are substituted by testlets of Time Partition Testing (TPT) [2]. To run the controller, the diagram is compiled to an executable and uploaded to the Target-PC. One can use xPC Target itself for real-time rapid control prototyping and hardware-in-the-loop simulation. However, for reasons given above, we use TPT to drive the tests. Based on the interface of the logical model, and by using a test coverage tree, the test scenarios can be systematically derived and represented graphically. Test coverage indicates, how well the test cases cover the range of possible test input combinations and is therefore one of the most important test metrics.

TPT supports semantics for the selection and documentation of test data in its testlets and the syntactic techniques direct definition, time partitioning and data partitioning which are used to build testlets. Test data can be characterised as a function that describes the time of termination and a signal course for each output channel depending on signal traces of input channels. A nonempty set of functions for sets of input and output channels is called a testlet. Testlets facilitate an exact description of test data and guarantee the automation of test execution and test evaluation. TPT uses an automaton-based approach to model the test behaviour and associates with the states pre- and post-conditions on the properties of the tested system (including continuous signals) and on timing. In addition, a dedicated run-time environment enables the execution of the tests. It is based on the script language Python extended with some syntactic test evaluation functions.

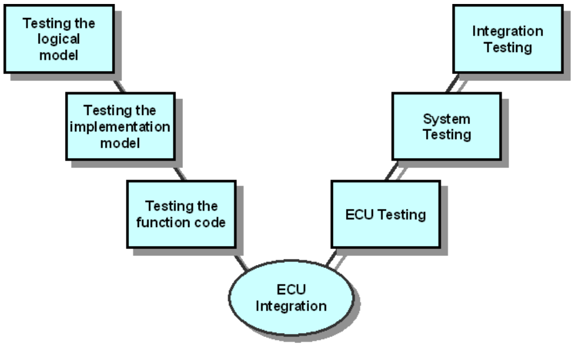

Testing the logical model
Testing the implementation model
Testing the function code
ECU Integration
ECU Testing
System Testing
Integration Testing

Fig. 1. Model-based testing process [5]

D. Statecharts

This is an extension of finite state diagrams. It provides a visual formalism for describing states and transitions of reactive or event-driven systems. State charts are compact and expressive. Matlab has an add-on called stateflow which implements a semantically restricted version of statecharts.

III. IMPLEMENTATION

The speed controller sub-system of an ACC measures the actual vehicle speed, compares it with the desired one and corrects any deviations by accelerating or decelerating the vehicle within a predefined time interval. Afterwards, the vehicle velocity should be maintained constant, if the desired speed does not vary. The speed of the vehicle is maintained using a PI controller to minimise the error between an actual vehicle speed and a desired vehicle speed by adjusting the throttle position to maintain a speed set by the driver. In this project, a simplified speed control subsystem is built using a model-based design process and tested to ensure the safety of the system.

Matlab Simulink/Stateflow is the main software used for computation, model implementation, and simulation (Fig. 2 and Fig. 3). Simulink/stateflow model of the controller has already been built: PI controller and State chart. The controller is parameterised to allow easy modification (tuning). The stateflow model has "on", "off", "set", "resume", "accelerate", and "coast" functions:

- On/off buttons activate/deactivate cruise system.
- Cruise system will be enabled by pressing Set or Resume button.
- Previous cruise speed will be loaded from the memory if cruise is enabled by the resume button.
- When cruise is enabled, cruise speed can be adjusted by Accelerate (up) and Coast (down) buttons.

Depressing the brake or the accelerator, the pedal will disable the system so the driver can change the speed without resistance from the system. The driver must bring the vehicle up to speed manually and use a button to set the speed control to the current speed. The system does not allow use of the speed control below 40 km/h.

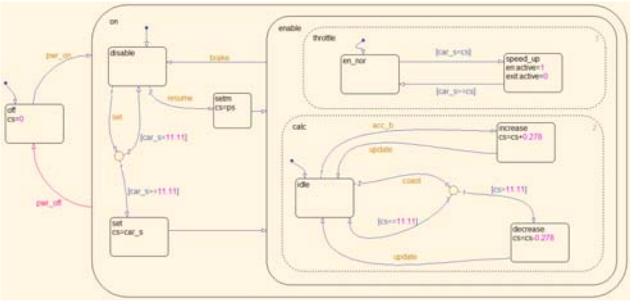


Fig. 2. State chart of the speed control sub-system

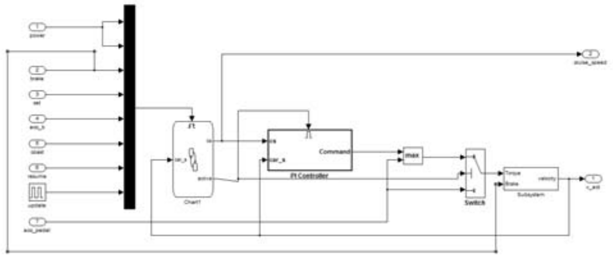


Fig. 3. Components of the speed control sub-system

IV. Results and Analysis

The speed controller works for discrete-time and continuous-time signals. The speed controller was simulated in MATLAB and then tested with the Time Partition Testing (TPT) tool (Fig. 4). The PI controller was tuned to get the expected results from the speed control sub-system. The simulation gives results in non-real time, but since this is a hard real-time system, we have to test the system in real time. This is achieved by letting the target-PC run the xPC Target real-time kernel and target processor (Xilinx FPGA) run the code for speed control.

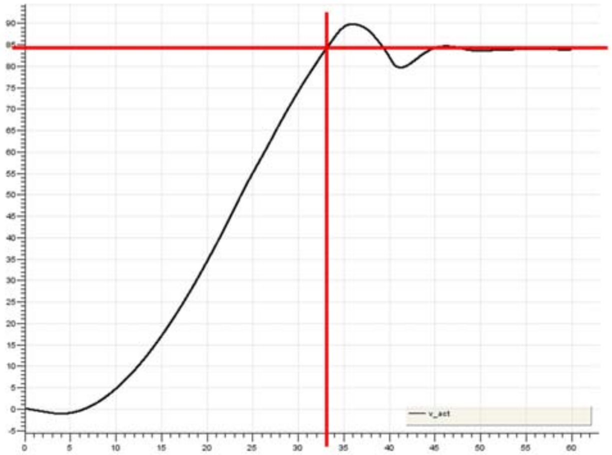


Fig. 4. Results generate from (TPT).

V. Conclusion

The objectives of this project were to develop a model for describing the behaviour of the speed control sub-system of an adaptive cruise controller, build a test bed for speed control sub-system and conducting the tests up to and including H-i-L level. The speed control sub-system has been developed and Model-in-the-Loop level tests carried out.

This project demonstrates a process for developing correct-by-design embedded controllers through the model-based design/testing of an ACC. The model provides a higher abstraction level than software code. Therefore, system complexity management becomes easier. Higher abstraction allows more software to be developed in the same time, change system requirements easily and find errors early through simulation (cheaper and faster to fix). Thus, software development time is reduced. It also eliminates hand coding, and hence, the resulting errors: the quality of the system is vastly improved.

References

[1] Justyna Zander-Nowicka, 2009. Model-based Testing of Real-Time Embedded Systems in the Automotive Domain [pdf] Available at: http://opus.kobv.de/tuberlin/volltexte/2009/2186/pdf/zandernowicka_justyna.pdf

[2] http://www.piketec.com.

[3] Jinghua Zhong, 2006. PID Controller Tuning: A Short Tutorial [pdf] Available at:

http://wwwdsa.uqac.ca/~rbeguena/Systemes_Asservis/PID.pdf

[4] http://www.who.int/violence_injury_prevention/road_traffic/en/

[5] K. Lamberg, M. Beine, M. Eschmann R. Otterbach, M. Conrad and I. Fey. Model-Based Testing Of Embedded Automotive Software Using Mtest. 2004 SAE World Congress, Detroit, Michigan, March 8-11, 2004.

[6] http://www.mathworks.in/products/xpctarget/description4.html.

[7] S. Kirstan and J. Zimmermann. Evaluating costs and benefits of model-based development of embedded software systems in the car industry results of a qualitative case study. Proceedings of the 6th European Conference on Modeling Foundations and Applications, 2010.

A Web-Based Content-Based Image Retrieval System

H.B.D. Himali, C.P.S. Kaluarachchi, W.M.D. Sajini, and D.S. Deegalla

Department of Computer Engineering, Faculty of Engineering,
University of Peradeniya, Sri Lanka

***ABSTRACT*—Content Based Image Retrieval (CBIR) is a popular approach for retrieving appropriate information from large image datasets. In this paper, we present a web-based CBIR system that takes an image as the query and retrieves images based on image contents, i.e., colour, texture and shape information. Most of the present retrieval systems may result less satisfaction to the user due to the usage of keyword based search for relevant images to a query. In this study, we address this issue by considering content based image features and getting Relevance Feedback (RF) from users. RF narrows down the gap between low-level visual features of an image and its semantic meaning. To further improve efficiency of the system clustering, grouping of similar images, was used. Precision-Recall cross over plot is considered as the performance evaluation measure for the CBIR system.**

***Keywords*: content-based image retrieval, clustering, relevance feedback, semantic gap, k-means**

I. Introduction

Automatic image retrieval systems provide an efficient way of navigating through the growing number of available images on the web. Traditional image retrieval systems allow users to retrieve images using associated metadata of an image [1]. This method finds images by matching keywords from a user query to keywords that have been manually added to the images. Therefore, this process becomes inefficient when searching through large collection of images since images may not be described using only keywords. In early 1990s, to overcome these difficulties and as a solution for this conventional image retrieval, content based image retrieval (CBIR) was introduced [1]. CBIR is based on matching a query image with a database of images through image to image similarity evaluation. However, still CBIR systems are not able outperform traditional keyword based systems. In this work, we have developed a web-based CBIR system which uses low level features as well as user feedback to improve the efficiency of the system by addressing the semantic gap [5] problem.

The rest of the paper is as follows. The next section describes previous work related to CBIR systems and then describes our proposed algorithms to improve the existing CBIR system. Next, we briefly describe the implementation. In section V, results and analysis are given. Finally, we provide concluding remarks of the study and further work.

II. Previous Work

Throughout the past decade, many general purpose image retrieval systems have been developed. Among them TinEye [5] claims that it is the first web-based image search engine which uses image identification technology. User can upload an image and get similar images based on the contents.

"Alta Vista", a search engine which allows content based image retrieval, both from special collections and from the web [6]. This system checks similarity between query and other images based on visual characteristics such as dominant colours, shapes and textures.

"SemQuery" system [13] is another CBIR system in which database images are clustered based on heterogeneous features in the images.

The first WWW image search engine which focused on relevance feedback in feature selection process was described by Scloroff, et al. [13].

III. Proposed Work

In representing each image in the system, we have considered three features; colour, texture and shape as the descriptors [2] of the images. Colour histogram was considered for colour feature extraction and it represents the number of pixels that have colours in each of the predefined colour ranges.Gray Level Co-occurrence Matrix (GLCM) [7] was considered for shape information and it statistically samples the image in a way that certain grey-levels occur in relation to other grey-levels. The elements of this matrix measure the relative frequencies of occurrence of grey level combinations among pairs of pixels with a specified spatial relation. To describe shape information, Chain codes and Horizontal Distance [7] are used. Chain codes represent the boundary of a binary image by a connected sequence of straight-line segments of specified length and direction. Horizontal distance vector describes the variance of the shape of the object from top to bottom.

In addition to selecting three features to represent content of the image, we also consider user preference of the output of the system. Therefore, we have considered relevance feedback (RF) [10] using Random Walks algorithm. The overall architecture is given in Fig. 1.

To improve the efficiency of the system we have used clustering [3] to identify similar images based on the three features chosen to represent images. K-means clustering algorithm [8] is chosen due to simplicity of the algorithm. RF uses Random Walks algorithm [10] in which it gets the user feedback on retrieved images and then re-weighted the features in the database [14]. In the next iteration, system retrieves images using re-weighted features in the database until user satisfies with the retrieved images.

In clustering, combination of the three features---colour, shape and texture features--- is chosen to represent each image.

Then we group similar images which represents each image in a group that is similar in their characteristics. When finding similar images to the query image, distance to each cluster's centroid is calculated instead of searching individual images. This cluster of images is used to find most similar images to a given query.

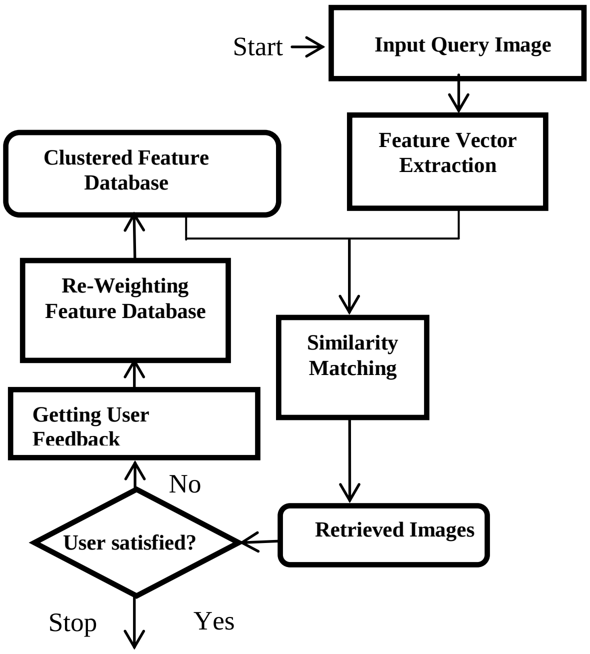


Fig. 1. Architecture of the proposed CBIR system.

During RF process, the images are ranked according to the weighted measure mentioned below. Similarity between the query image (Q) and the images in the database (I) is given by the following weighted measure $M(I,Q)$.

$$M(I,Q) = \sum_{i=1}^{3}(w_i) * |f_{iI} - f_{iQ}| \qquad (1)$$

Where, f_{iI} , f_{iQ} are i^{th} feature component of I and Q respectively. Here w_i is the initial weight of i^{th} feature component. With RF these initial weights are updated according to the user feedback on retrieved images to a query [14].

IV. Implementation

The method proposed for image searching and retrieval mainly focused on generation of feature vectors and calculating the Euclidean distance between feature vectors of the query and the images in the database. The Euclidean distance d(I,Q) between query image (I) and an image in the database (Q) is calculated using the following formula.

$$d(I,Q) = \sqrt{\sum_{i=1}^{3}(f_{iI} - f_{iQ})^2} \qquad (2)$$

Lower the value of d(I,Q) of an image in the database indicates higher relevance to the query image.

For clustering, we have used Matlab built-in function for K-Means [9]. We have selected K as 4 since database images are from four different classes. There each image can be considered as a point in the multi-dimensional space. Given two points, p_I and p_C for the query image and a cluster centre where each feature vector described by their feature component vectors, $p_I = (f_{I1}, f_{I2}, f_{I3})$ and $p_C = (f_{C1}, f_{C2}, f_{C3})$, the distance, d(I,C), between p_I and p_C is given by:

$$d(I,C) = \sqrt{\sum_{i=1}^{3}(f_{Ii} - f_{Ci})^2} \qquad (3)$$

The distances are calculated with each cluster centroid and the cluster centroid with least distance to the query image is chosen as the best cluster. Then, the most similar images to the query image are retrieved from the selected cluster.

To evaluate the performance of the CBIR system, we have considered a database of 300 images belongs to four different classes. After generating feature vectors for colour, texture and shape of each image, the Euclidian distance of all images with respect to the query image is calculated. The images with minimum Euclidian distance are considered as better match for the query image. For the evaluation Precision, and Recall graph is considered. The Precision, and Recall as defined in the equations 4 and 5 below, depicts the measure retrieval performance of the algorithm.

$$\text{Precision} = \frac{\text{Number of relevant images retrieved}}{\text{Total number of images retrieved}} (4)$$

$$\text{Recall} = \frac{\text{Number of relevant images retrieved}}{\text{Total number of relevant images in database}} (5)$$

V. Results And Analysis

Fig. 2 represents the Precision and Recall graph of the system.

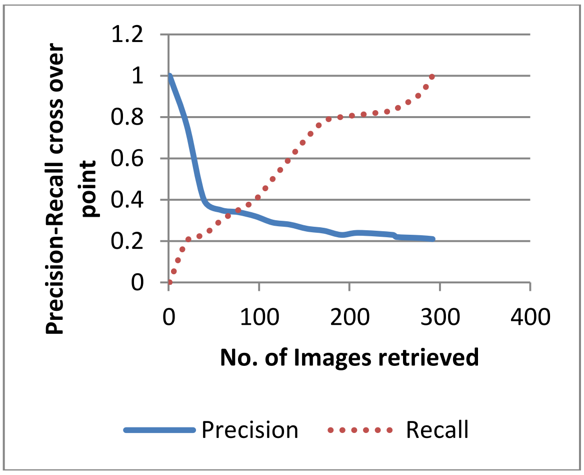


Fig. 2. Graph of Precision and Recall for a selected query image with database of 300 images

The point of cross section of Precision and Recall over the maximum number of relevant retrievals for database of 300 images is seen in Fig. 2. Here X axis represents the images retrieved and Y axis represents the Precision-Recall value. As shown in Fig. 2 the Precision decreases as the number of images retrieved from the database increases, while the Recall increases as the number of images retrieved from the database increases. Crossover point [14] in Precision and Recall is the point on the graph where both the Precision and Recall curves meet. Crossover point can be used to measure effectiveness of our CBIR system. According to [14], higher the crossover point, better is the performance of the system. The Precision-Recall crosses over point value obtained for 300 images in the database are 35%, i.e., there are 35% of relevant images retrieved in the output.

VI. Conclusion

In this study we have considered three visual feature descriptors, i.e., colour, texture and shape. We also employed clustering technique on target and query images to identify images with similar characteristics. We have also considered relevance feedback from the user to reduce the semantic gap between low level features and high level features. Precision-Recall graph was used to measure the accuracy of the CBIR system.

The accuracy of the current CBIR system could be improved using Meta data as another descriptor set to support the retrieval of images. Fuzzy logiccould be used to deal with the ambiguity of human judgement on image similarity.

References

1. "Image retrieval." Internet: http://en.wikipedia.org/wiki/Image_retrieval [Nov. 28, 2012].
2. A. Gupta and R. Jain, "Visual information retrieval," Communications of the ACM, vol. 40, no. 5, pp. 70–79, 1997.
3. A. K. Raikwar and S. Jain, "Content based image retrieval using clustering," International Journal of Computer Applications, vol. 41, no. 20, pp. 29-32, Mar. 2012.
4. X. S. Zhou and T.S. Huang, "CBIR: from low-level features to high-level semantics," in Proc SPIE Image and Video Communication and Processing, 2000, pp. 24-28.
5. "TinEye." Internet: http://en.wikipedia.org/wiki/TinEye. [Nov. 25, 2012].
6. "AltaVista." Internet: http://en.wikipedia.org/wiki/AltaVista. [Nov. 28, 2012].
7. I. Case, "Content based image retrieval feature vector selection," Nov. 2008.
8. M. Jain and S.K. Singh, "A survey on: content based image retrieval systems using clustering techniques for large data sets," IJMIT, vol. 3, no. 4, pp. 25-27, Nov 2011.
9. G. Chamundeswari et al., "An experimental analysis of Kmeans using Matlab," IJERT, vol. 1, pp. 1-5, Jul. 2012.
10. S. R. Bulo et al., "Content-based image retrieval with relevance feedback using random walks," Pattern Recognition, vol. 44(9), pp. 2109-2122, 2011.
11. S. Sclaroff et al., "Mix and match features in the image recover search engine," in Principles of Visual Information Retrieval, pp. 259-277, 2001.
12. H. Narasimhan and P. Ramaraj, "Contribution-based clustering algorithm for content-based image retrieval," ICIIS 2010, 2010.
13. G. Sheikholeslami et al., "SemQuery: semantic clustering and querying on heterogeneous features for visual data," IEEE Trans. Knowledge and Data Engineering, vol. 14, no. 5, pp. 988–1002, 1998.
14. Y. Wu and A. Zhang, "A feature re-weighting approach for relevance feedback in image retrieval," ICIP, vol. 2, pp. 581-584, 2002.

Spoken Language Identification

A.M.M.S. Adikari, A.L.S.M. Perera, W.G.S. Wiharagoda and D. Elkaduwe

Department of Computer Engineering,

University of Peradeniya, Sri Lanka

***Abstract* – This paper proposes acoustic features based method for automatic spoken language identification (LID). Identifying many languages is impractical task for human beings. Proposed system identifies the spoken language using the features of the speech. We demonstrate that the use of acoustic features, both in training and testing, significantly improves the accuracy of an acoustic features based LID system. Performance is further enhanced by using a classification method building the training model. Basically this approach is based on three major features. The system achieves a 66% of accuracy when it uses a language model which is containing features of four different languages.**

I. INTRODUCTION

Language identification (LID) is a human skill which has many practical applications. Necessity of automating the language identification using computer system is increasing as the communicational affairs are increasing day by day. Also, identifying an unknown language is desirable feature in the concept of a global village.

There are about 6,900 unique languages in the world. However, only 6% of these languages are spoken by 94% of the world population of 6.4 billion people [1]. Given these discrepancies it would be advantageous if artificial intelligence systems possessing the capacity to store models of known languages carry out the task of LID automatically.

In this paper, we propose a method for implementation of user independent language identification (LID) system using classification methods. LID has various applications where one application could be a telephone based front-end system whose main work is to route the call to the appropriate caller who is fluent in that language. Other applications of language identification system would be in the speech-to-speech translation, shopping, airports and other commercial areas. In this system, we extract some desirable unique features from the speech. Using the extracted features, we use a classification method to identify the language.

II. ACOUSTIC FEATURES APPROACH

Pre-processing – Most of the speech data used in our system is obtained from BBC news sound clips. The background music and not relevant sounds other than voice were removed from every sound clip as the first step of processing. Then audio files were converted into 16-bit with 44,100 Hz.

MFCC Vectors – In sound processing, the Mel-frequency cepstrum (MFC) is a representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear Mel scale of frequency [3]. Mel-frequency cepstral coefficients (MFCCs) are coefficients that collectively make up an MFC. As in Fig. 1, they are derived from a type of cepstral representation of the audio clip (a nonlinear "spectrum-of-a-spectrum"). The difference between the cepstrum and the mel-frequency cepstrum is that in the MFC, the frequency bands are equally spaced on the mel scale, which approximates the human auditory system's response more closely than the linearly-spaced frequency bands used in the normal cepstrum [2].

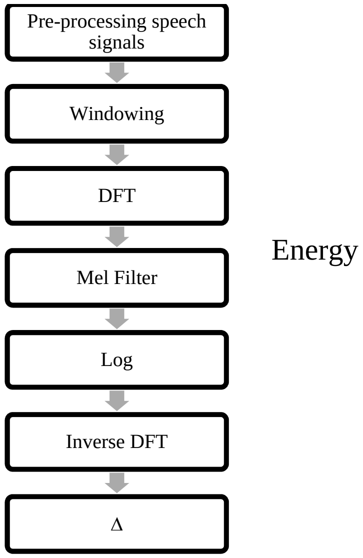


Fig. 1. MFCC Vectors generation process

Because of its ability to show the amplitude spectrum of an audio signal in a compact form, Mel Frequency Cepstral Coefficients (MFCC) is a one of the most prominent features to LID system. In order to find the energy of the sound, we use windowed segments as inputs of the next steps so that windowing function is applied to break the utterance into segments for further processing. Most of the time in speech processing, the Hamming window is used to segment the speech signal. Normally, the Hamming and Hann window are perceptually the same, and in our case we have also used the Hamming windowing function to segment the speech [6]. Then the windowed signal is given to the DFT (Discrete Fourier Transform) as an input. The output of the Fourier transforms shows information concerning the amount of energy at each frequency band. Then the output of the Fourier transform is converted into Mel scale. To compute the Mel frequency from the frequency, the following function is applied.

$$Mel\ Frequency = 1127 \ln\left(1 + \left(\frac{f}{700}\right)\right)$$

Pitch Contours – The human ear is sensitive for the variation of the pitch. So, this ability is used in this system as a feature of the sound. The Pitch contours can be taken as the variation of the pitch at each point. Fig. 2 shows the variation of the pitch with the spectrum of the signal [5].

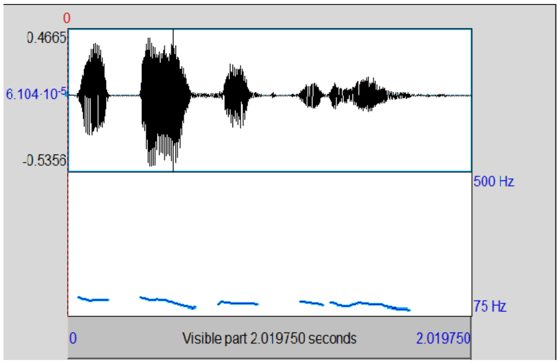


Fig. 2 Pitch Contour Sample

III. CLASSIFICATION

ARFF Generation – The Attribute-Relation File Format is a previously collected data set and it the file does not specify which of the data is to be classified [7]. Since this project is based on machine learning concepts, ARFF file has to be generated with the collected data. The attributes of ARFF file change when we select the features for the classification. As an example, for thirteen MFCC coefficients with the pitch contour, the number of attributes is fourteen.

Classification model generation – Separate classification model was generated for every feature that had been extracted from the speech signal. MFCC vectors can be distinguished using the number of coefficients used inside the vector. Five different MFCC vectors were generated from the training data. They are:

- MFCC-5 vectors
- MFCC-13 vectors
- MFCC-20 vectors
- MFCC-27 vectors
- MFCC-41 vectors

Each vector type was considered as separate feature. WEKA toolkit was used to build classification models feeding the generated ARFF files. J48 tree algorithm was used for building classification models. All classification models were tested using separate test sets.

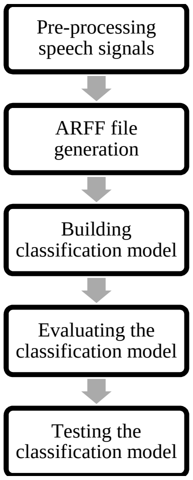

Pre-processing speech signals
ARFF file generation
Building classification model
Evaluating the classification model
Testing the classification model

Fig. 3 Classification model generation process

Predicting the language – The classification models that had been generated using the above process (Fig. 3) were used to predict the language class of the given speech signal. ARFF file was generated from the given speech signal extracting the same acoustic features which had been used to build the classification model. That ARFF file was fed to WEKA for predicting. Sample language prediction is given in below.

Sinhala Tamil English Hindi
Language: Sinhala occurrence: 1757
Language: Tamil occurrence: 251
Language: English occurrence: 335
Language: Hindi occurrence: 287

The Language: Sinhala

The developed system predicts the possible candidate for the language. The final class is estimated using the majority predicted class. In above example, 1754 instances had been predicted as Sinhala language, 251 instances of the input speech signal had been predicted as Tamil language, 335 instances had been predicted as English languages and 287 instances had been classified as Hindi language. The majority predicted class was Sinhala, so the final classification was given as "Sinhala".

IV. RESULTS

The accuracy of the system changes when we select different number of MFCC coefficients to build the model. The following tables and graphs (Table I and II; Fig. 4 and 5) show the variation of the accuracy with selected features.

TABLE I ACCURACY WITH MFCC VECTOR.

Number of MFCC Vectors	Percentage /(%)
5 MFCC	53
13 MFCC	69
41 MFCC	67

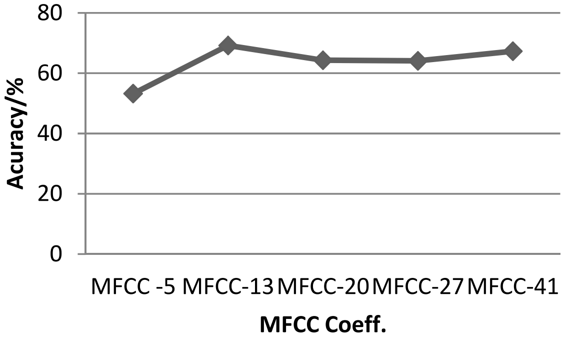


Fig. 4. Variation of the accuracy with MFCC Vector

The accuracy changes when select the pitch contour with the MFCC vectors.

TABLE II. ACCURACY WITH 5 MFCC COEFFICIENTS AND PITCH CONTOUR

Number of MFCC Vectors + Pitch	Percentage/(%)
5 MFCC	52
13 MFCC	66
41 MFCC	58

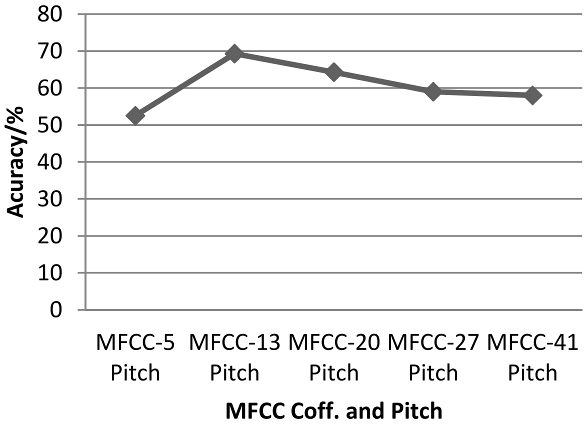


Fig. 5. Variation of the accuracy with MFCC coefficient and Pitch

V. CONCLUSION

As obvious in the results, the accuracy has been changed with the number of coefficients and it has changed when we consider the pitch contours and the MFCC vectors. In both the situations, the highest accuracy is achieved when the number of MFCC vectors is 13. Further study is required to find the reason for these variations.

Initial requirement was identifying the spoken language of the given speech. The system has been trained for four languages (Sinhala, Tamil, English, and Hindi) and the system is capable of identifying the spoken language out of the above mentioned four languages. The current LID system cannot identify the language accurately from speech signals recorded in noisy environments. The reason for that is the current classification modes have been trained for noiseless signals. Those classification models give incorrect results for signal with noises.

All of the classification models have been built using J48 tree algorithm in data mining. Every feature that has been selected building the classification model has numeric values. So J48 may not be the ideal classification algorithm with this kind of attribute. A different classification algorithm may increase the accuracy of the prediction.

Classification model training process is time consuming. MFCC-13, pitch model took approximately 3 hours for completion its' training time in WEKA tool kit. If the number of languages is large, time it takes for the classification process of the classification models will take long time.

VI. REFERENCES

[1] Adda-DeckerM. (2008). Automatic Language Identification. Wilay-ISTE: MarianiJ.

[2] Logan-B. (2000). Mel frequency Cepstral Coefficients or Musical Modelling.

[3] Mel Frequency Cepstrum, Wikipedia. (n.d.). Retrieved from Wikipedia: http://en.wikipedia.org/wiki/Mel-frequency_cepstrum

[4] Pitch Analysis, Praat. (n.d.). Retrieved from Praat manual: http://www.fon.hum.uva.nl/praat/manual/Intro_4__Pitch_analysis.html

[5] Pitch Contour, Wikipedia. (n.d.). Retrieved from Wikipedia: http://en.wikipedia.org/wiki/Pitch_contour

[6] Window function, Wikipedia. (n.d.). Retrieved from Wikipedia: http://en.wikipedia.org/wiki/Window_function

[7] Witten, I. &. (2005). Datamining Practical machine Learning Tools and Techniques, 2nd Ed. San Francisco: Morgan Kaufman.

Gesture Tracking For Android

P. Dhanushanth, M.Y.M. Firnas and Z. Maraikar

Department of Computer Engineering, Faculty of Engineering,
University of Peradeniya, Sri Lanka

Abstract — We describe the process of porting Microsoft's Kinect gesture tracking sensor for use on the Android platform, using existing Linux libraries and applications as a starting point. Gesture tracking on Android will open up a new interaction mode not just for gaming but also applications such as gesture control of set-top boxes and Smart TVs running the Android OS.

Keywords— Gesture tracking; Microsoft Kinect; Android porting

I. INTRODUCTION

Gesture recognition requires high quality stereoscopic cameras and complicated computer vision algorithms to recognize human gestures. The systems often turn out to be expensive and require extensive set-up [3].

Microsoft Kinect provides an inexpensive and easy way for real-time user interaction, this is an accessory to Xbox console provides excellent gesture recognition capabilities as well as user and environment data. These features can be used to develop intuitive user applications that provide a new dimension to user interaction.

We wish to make this gesture tracking device independent using the Android platform [2] since a large variety of electronic devices such as smartphones, televisions and tablets etc. come with Android Operating system. The implementation of Kinect library for android device will enable user interaction applications and games [4].

II. PREVIOUS WORK

Developers have already developed some native Android application that uses Microsoft Kinect on the Tegra Ventana [6] (Android 3.0) development kit . The project was proof of concept for data capturing and rendering using Kinect. The drawback of the Tegra Ventana project is they have not verified that their setup will run on other android platforms.

Yang [5] proposed a gesture recognition system using depth information provided by Kinect. It was able to recognize eight gestures to control the media player. This system demonstrates the applicability of using Kinect for gesture recognition. The algorithm for hand tracking is to first find the hand waving motion based on the assumption that a user tends to start an interaction on session with such a motion [3].

III. PROPOSED WORK

Fig. 1 shows the components of the Linux Kinect stack that need to be ported to have functional gesture recognition on Android.

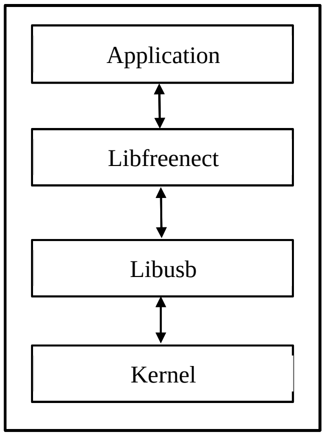


Fig. 1. Application Design

Libusb is a C library that gives applications easy access to USB devices on many different operating systems. This is used to make the communication between the android kernel and Kinect device driver. Libfreenect [7] implements a user-space a driver for Kinect. It implements the device communication protocol on top of Libusb.

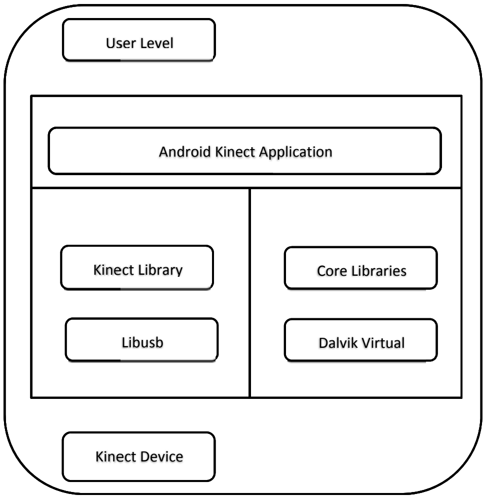


Fig. 2. Architecture Design

Fig. 2 shows the components mapped to four layers in Android: the Linux kernel, Native layer, Application framework layer and Application layer. The porting process has been done in two steps.

Native Layer: This contains the Hardware libraries used to communicate between kernel level and Application level. This will run on native layer for performing Kinect library calls.

Application Layer: All Android applications (native or third party) are built on the application layer using the same Android API. The Kinect user level interaction application will be implemented here. After the implementation of Libfreenect in the native layer the android application will be implemented on top of Libfreenect.

IV. Implementation

The development workflow summarized in Fig. 3 is described below.

1. Define abstract native method in java to call native code. The code is compiled with Javac to generate the native library class using JNI (Java native interface).
2. The compiled file is built with Javah tool that generates a C header file. Then these functions are implemented in the C in native layer.
3. Compile Native code with Android NDK. The Android runtime environment is a subset of the full Linux environment, so we need to port code. We first built the Libfreenect source file on Linux to confirm that it works. To port the Kinect Library from Linux to Android we write an Android.mk file to build the library against the NDK. The Android NDK allows you to write code-using C/C++ and then be linked into Java application to access the Kinect device.
4. Call native code from Java. The library is integrated with Android application layer using the native methods implemented in [1]. We use Android SDK to run and test android Kinect application on android SDK [1].

V. Evaluation

The Android x86 platform [10] running on a Virtualbox virtual machine was used for the porting work and testing. The Kinect was connected to the computer running Virtualbox and the USB connection was passed through to the VM.

To facilitate debugging during testing we added android debug logging to freenect library and the test application. This allowed us to quickly identify issues. We also hope to test the setup on Android x86 running natively on a PC.

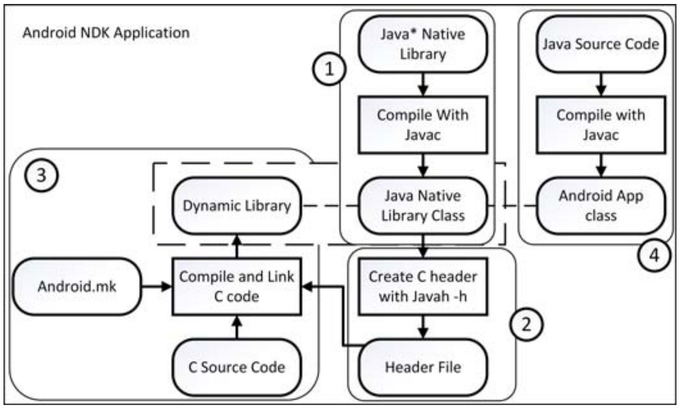


Fig. 3. Architecture Design

The library and sample application was also verified to compile and install on the Android ARM emulator. ARM-based hardware with USB host support such as tablets should be able to run our Kinect stack.

VI. Conclusion

Our project ported Kinect library to the Android platform version 2.3. Unlike the previous Tegra Kinect port, our project is platform-independent. This opens up gesture recognition to any device, which runs Android and provides USB host connectivity for the Kinect.

Reference

1. Android SDK. *Webopedia.* [Online] http://www.webopedia.com/TERM/A/Android_SDK.html.
2. Varghese, Allen Thomas. hasgeek. *www.funnel.hasgeek.com.* [Online] 12 06 2012. http://funnel.hasgeek.com/droidcon2012/497-microsoft-kinect-and-android.
3. Microsoft. *support.xbox.com.* [Online] 2013. http://support.xbox.com/en-US/xbox-360/kinect/accessibility-kinect.
4. SAMY, MINA. [Online] [Cited: 10 November 2010.] http://mobileorchard.com/introduction-to-android-development/.
5. Google. Android. [Online] 2013. http://developer.android.com/tools/sdk/ndk/index.html.
6. BLog. http://www.javabeat.net/. [Online] http://www.javabeat.net/qna/388-what-is-android-runtime/.
7. Aleksandar (Saša) Gargenta, Ken Jones, Marko Gargenta. Marakana Android Internals. *Androd Stack.* [Online] 2012. [Cited: 17 08 2013.] http://marakana.com/static/courseware/android/internals/index.html#_android_stack.
8. Raymondlo. *raymondlo84.blogspot.com.* [Online] 27 July 2011. http://raymondlo84.blogspot.com/2011/07/howto-using-microsoft-kinect-on-tegra.html.
9. Wiki. *Wiki.* [Online] 18 May 2013. http://en.wikipedia.org/wiki/Tegra.
10. Tips, Speaking. *www.speaking-tips.com/.* [Online] 15 December 2003. http://www.speaking-tips.com/Articles/Hand-Gestures.aspx.
11. *HAND GESTURE RECOGNITION USING KINECT.* Li, Yi. China : University of Louisville , 2010, Vol. I.
12. Android. android . *www.android.com.* [Online] android , 2013. [Cited: 28 May 2013.] http://www.android.com/about/.
13. Cheoljong Yang, Yujeong Jang, Jounghoon Beh, David Han, and Hanseok Ko. *Gesture recognition using depth-based hand tracking for contactless controller* . s.l. : 2012 IEEE International Conference,pages 297 -298, 2012.

Handwritten Tamil Text Recognition

S. Vinodharanie, S. Gobiga, and D.S. Deegalla
Department of Computer Engineering, Faculty of Engineering,
University of Peradeniya, Sri Lanka

***Abstract*- Handwritten text recognition has received extensive attention in academic fields. There are sufficient studies that have performed in foreign language scripts like Chinese, Japanese and Arabic characters, however, only a very few work can be traced for handwritten text recognition of Tamil language. Various methods have been proposed in each phase of the recognition process, whereas each approach provides solution only for few character sets. However, there is no general solution to identify all Tamil characters with reasonable accuracy. Challenges still prevails in the recognition of normal as well as abnormal writing, slanting characters, similar shaped characters, joined characters, curves and so on during recognition process. Almost all the existing handwritten character recognition techniques use neural network approach, which requires lot of preprocessing and a large dataset for training. In this study, we develop a nearest neighbour based handwritten text recognition system while evaluating several preprocessing techniques. The proposed system provides the prediction accuracy of 78% for tablet handwritten characters and 45% for scanned handwritten characters.**

Keywords—Optical Character Recognition (OCR), Handwritten Tamil Text Recognition

I. Introduction

The penetration of Information Technology (IT) becomes harder in a country such as Sri Lanka where the majority read and writes in their native language. Therefore, enabling interaction with computers in the native language and in a natural way such as handwriting is absolutely necessary.

Tamil is one of the oldest languages in the world. It is an official language in countries such as Sri Lanka, India, Singapore and Malaysia. Being the mother tongue of several million people, recognition of Tamil handwritten characters has a wide range of applicability.

Handwriting recognition is one of the challenging problems being investigated today. The main difference between handwritten and typewritten characters is in the variations that are associated with handwriting. Recognition of handwritten characters with respect to any language is difficult, since the handwritten characters differ not only from person to person but also according to the state of the mood of the subject person.

The difficulty of the recognition problem scales with the complexity and size of the character set in consideration. Tamil has a large alphabet size. There are 156 distinct symbols in Tamil [1]. The presence of angles, curves and holes in majority of the characters pose a great challenge in the recognition process.

Broadly, handwritten text recognition systems include four stages: image preprocessing, segmentation, feature extraction, and classification. Here preprocessing is primarily used to reduce variations of handwritten characters. Segmentation is where the image is decomposed into lines, words and individual characters. A feature extractor is essential for efficient data representation and extracting meaningful features for later processing. A classifier assigns the characters to one of the several classes.

The remainder of the paper is organized as follows: Section II briefly reviews the previous work on recognition of handwritten Tamil characters. The proposed system architecture is given in Section III. Section IV describes the implementation. The results of our experiment are given in Section V and conclusions are mentioned in Section VI.

II. Related Work

Suresh Kumar and Ravichandran [2] described a system for handwritten Tamil character recognition using RCS (Random Character Segmentation) algorithm. The extracted features considered for recognition are analysed using neural network concepts, where the characters are classified using supervised learning algorithm. These classes are mapped onto Unicode for recognition. Then the text is reconstructed using Unicode fonts. In this system, the overall efficiency was found to be 97% for a set of identified characters.

Seethalakshmi et al. [3] discuss the various strategies and techniques involved in the process of converting printed Tamil text documents into software translated Unicode Tamil text where extracted features are passed to a Support Vector Machine (SVM) in which the characters are classified by supervised learning algorithm. The recognition rate achieved was 50 % for 5 different numbers of fonts.

Kannan et al. [4] proposed a novel solution for performing character recognition in Tamil. They have used octal graph conversion for recognizing off-line handwritten. In this system overall efficiency was found to be 82%. The evaluation cases consisted of ten samples, which consist of 4 good samples, 4 misaligned samples and 2 extremely disfigured samples.

Almost all the existing handwritten character recognition techniques use neural network approach. In this paper, we develop a nearest neighbour based character recognition system for 156 distinct characters.

III. PROPOSED ARCHITECTURE

The proposed architecture includes a scanning state, a pre-processing block, and a segmentation and feature extraction classification sections. The architecture is briefly described in Fig. 1.

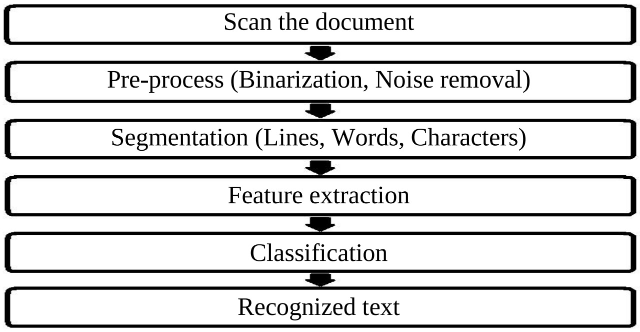

Scan the document
Pre-process (Binarization, Noise removal)
Segmentation (Lines, Words, Characters)
Feature extraction
Classification
Recognized text

Fig. 1. Proposed architecture of the system

A clearly written document is chosen for scanning. Several pre-processing steps are applied to the scanned image before segmentation. The objective of these steps is binarization and noise removal. Binarization transforms the grayscale image into a black and white binary image. After pre-processing, the image is passed to the segmentation phase, where the image is decomposed into individual characters. After segmenting into individual character images, feature extraction is performed on these images considering character height, width, horizontal lines, vertical lines and slope. Statistical techniques are considered for classification, which determines the probability of the observed pattern belonging to a certain class. Example: K-Nearest-Neighbour, Bayesian classifier, Support Vector Machine, Bagging, and Random Forest.

IV. IMPLEMENTATION

To train the system, we selected several existing databases. We have used hpl-tamil-iwfhr06-train-offline database [6] in our experiments. This dataset contains samples of the 156 character classes collected from different writers using a TabletPC application.

A. *Pre-processing*

Images captured often may be influenced by noise and may not be desirable for analysis. In addition, in images with acceptable quality, certain regions may need to be emphasized or highlighted. Therefore, pre-processing steps are performed on the scanned images. The binary image is pre-processed for noise removal. We considered the following techniques for pre-processing to select the best method.

- Spatial Filtering
- Homomorphic Filtering
- Skeleton & Universe of Discourse

1. *Spatial Filtering*

By using spatial filtering, threshold image is smoothed or blurred. Blurring is used in pre-processing such as removal of small details from an image tier to object extraction and bridging of small gaps in line or curves. Smoothing is to remove unnecessary noise present in the image. The filtering operations were repeated for every pixel in the original image to generate the smoothed image.

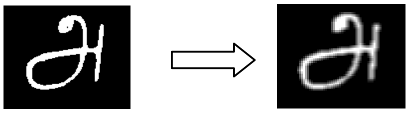


Fig. 2. Binary image and image after spatial filtering

2. *Homomorphic Filtering*

Homomorphic filtering is used for image enhancement. It simultaneously normalizes the brightness across an image and increases contrast. Here homomorphic filtering is used to remove multiplicative noise.

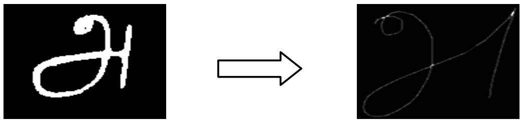


Fig. 3. Binary image and image after Homomorphic filtering

3. *Skeleton & Universe of Discourse*

Skeletonization

The skeletonization process estimates a skeleton for a character and removes pixels that do not belong to the skeleton. The skeletonized thin line character will be input to the feature extraction step.

Universe of Discourse

Universe of discourse is defined as the smallest rectangular region which the entire character skeleton fits into. The purpose of this process is to minimize the effect of character size on classification.

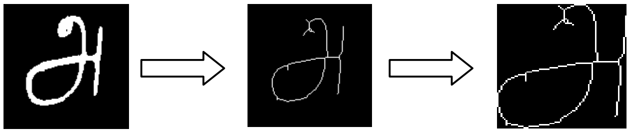


Fig. 4. Binary image, skeleton image and universe of discourse image

B. *Segmentation*

After pre-processing, the noise free image is passed to the segmentation phase, where the image is decomposed into lines, words and individual characters. A vertical histogram is used to segment the image into lines. A horizontal histogram is used to segment lines into words and words into characters.

C. *Classification*

A classifier assigns the characters to one of the several classes. Statistical techniques [5] are considered for classification, which determine the probability of the observed pattern belonging to a certain class. We have considered the following six algorithms to choose the best classifier for our system.

- Nearest Neighbour (IB1)
- Nearest Neighbour Generalized Classifier (NNGE)
- Decision Tree (J48) Classifier
- Bagging Classifier (RFPTree)
- NaiveBayes

V. EXPERIMENTAL RESULTS

Table I shows the performance of the six classifiers when trained with 2080 instances and tested for 1040 new tablet data which belong to 156 distinct classes. Classification accuracy is taken as the evaluation measure.

TABLE I. PERFORMANCE OF SIX CLASSIFIERS ON TABLET DATA

Classifiers Methods	IB1	NNGE	J48	NaiveBayes	Bagging (REPTree)
156 distinct characters	58	55	28	54	36
Skeleton, Universe of Discourse	52	49	41	47	64
Spatial Filter	**78**	63	41	53	53
Homomorphic Filter	76	63	41	61	53

Table II shows the performance of the classifiers when trained with 3120 instances and prediction accuracy calculated for 50 fresh handwritten data. Classification accuracy is taken as the evaluation measure.

TABLE II. PREDICTION ACCURACY ON FRESH HANDWRITTEN DATA

Classifiers Methods	IB1	NNGE	J48	NaiveBayes	Bagging (REPTree)
156 distinct characters	0	19	0	0	0
Skeleton, Universe of Discourse	16	34	16	24	28
Spatial Filter	**45**	38	13	25	4
Homomorphic filter	35	35	6	21	19

According to the experimental results, Nearest Neighbour Classifier (IB1) outperforms the other methods with spatial filtering.

We have also tested the proposed system with handwritten characters of different individuals. Fig. 5 shows few examples for original images and corresponding recognized image.

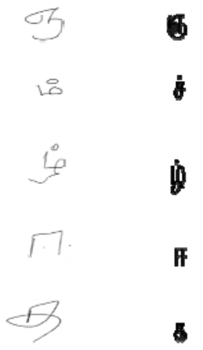


Fig. 5. Original image and corresponding recognized image

VI. CONCLUSION

In this paper, handwritten text recognition system for Tamil language is described. The input text document is pre-processed, segmented and recognized and the recognized text is displayed in a text file.

As evident from the experimental results, spatial filtering increases the accuracy compared to other methods. To increase the accuracy further, we can train large datasets and use scanned images for training the dataset.

REFERENCES

[1] N. Joshi, G. Sita and A. G. R. S. Madhavanath, “Comparison of Elastic Matching Algorithms for Online Tamil Handwritten Character Recognition”, Proc. Intl Workshop Frontiers Handwriting Recog., 2004, pp. 444- 449.

[2] Suresh Kumar C and Ravichandran T, “Handwritten Tamil Character Recognition using RCS algorithms”, Int. J. of Computer Applications, (0975 – 8887) Volume 8– No.8, October 2010.

[3] Seethalakshmi R., Sreeranjani T.R, Balachandar T., Abnikant singh, Markandey Singh, Ritwaj Ratan, SarveshKumar, "Optical Character Recognition for printed Tamil text using Unicode", Journal of Zhejiang University SCIENCE, Vol. 6A No.11, 2005.

[4] Kannan, R. Jagadeesh, and R. Prabhakar. "An Improved Handwritten Tamil Character Recognition System using Octal Graph." 2008.

[5] Bortolozzi, Flávio, Alceu de Souza Britto Jr, Luiz S. Oliveira, and Marisa Morita. "Recent advances in handwriting recognition." Proceedings of the IWDA 5, 2005.

[6] HP Labs India, “Intuitive Multimodal and Gestural Interaction project”, http://www.hpl.hp.com/india/research/penhw-resources/tamil-iwfhr06-train.html, 2006.

Object Storage for Work and Play

A. Nissanka, T. Dassanayake and Z. Maraikar

Department of Computer Engineering, Faculty of Engineering,
University of Peradeniya, Sri Lanka

***Abstract*— Data is getting bigger and complex, and they are being consumed via various devices and applications, creating a need for a reliable storage solution. As a solution, cloud based storage solutions and storage servers were introduced. But, there are certain cases in which purchasing and maintaining a high-performance servers or using cloud based solutions is too costly and not worth the investment. Typically, the internal system of companies and institutes contain desktop workstations with large free and unutilised storage. Such storages can be used as a storage system to fulfil storage requirement within the organization eliminating the need for costly solutions.**

This paper proposes a storage solution which can be deployed on top of commodity hardware with average performance. And, the solution consists of a distributed storage system which can be deployed on a local network and a client application which is to be deployed on a small scale web server.

I. Introduction

With rapid evolution of the technology, the data is getting more complex and growing at a higher rate. As more digital devices connect to the Internet and new applications are deployed, massive amount of new data is being created. This rapid growing behaviour of data generates an emerging requirement of reliable storage solutions.

Over past few years, many solutions have been introduced to cater the storage requirements, and they were either cloud based solutions or data servers with high I/O throughput. But within some contexts, these systems are lacking some factors as explained below.

- Servers with high I/O throughput are very expensive and they should be maintained by experienced personals.
- Transferring large amount data to/from cloud based solutions results in a high bandwidth usage which is expensive.

As a comprehensive solution for the above mentioned factors, this paper suggests a distributed storage system which can be deployed on commodity hardware similar to desktop workstations.

The solution suggested in this paper has been deployed and tested on an environment similar to the target environment which is a local network with nodes with average performance, using a local network built using Raspberry Pis as the nodes.

A cluster of Raspberry Pis have been used to simulate the production environment and to test the system performance while processing requests. Main component, which is the storage system, has been installed on the cluster and the client application has been installed on a web server.

The system has been evaluated using test data for certain parameters and it has the potential of running in a production environment.

II. Literature Survey

Many researchers have carried out many researches related to storage systems explained above, and among those, the following have a significant value since most of other solutions have been influenced from these basic systems.

A. Amazon Dynamo [1]

Amazon's Dynamo was a large scale research project which was carried out in order to address some of the critical issues Amazon's e-commerce operations faced. They introduced a distributed storage system in which the information (data) is not broken into tables as in relational databases. Instead, it stores them as a key/value pair that is all objects are stored and looked up via a key.

B. Google Bigtable [2]

Google BigTable is a distributed storage for managing structured data and it is designed to reliably scale to petabytes of data and thousands of machines. Bigtable maintains data in lexicographic order by row key. The row range for a table is dynamically partitioned. Each row range is called a tablet, which is the unit of distribution and load balancing.

X. Apache Hadoop [3]

Apache Hadoop introduced a way of storing and processing data. Instead of relying on expensive, proprietary hardware and different systems to store and process data, Hadoop enables distributed parallel processing of huge amounts of data across inexpensive, industry-standard servers that both store and process the data, and can scale without limits.

III. Proposed Solution

Considering the contexts and facts described above, a system was designed to be deployed on an environment with limited resources. There are two major components involved in the system described below.

- Distributed Storage System
- Client Application

Distributed Storage System is deployed in the internal network of desktop workstations. It consists of workstations with commodity hardware in which a distributed database has been installed. The database consumes a predefined amount of capacity from each workstation's local storage leaving its resources to be used for general purposes. Within the system, these workstations are considered as data nodes.

Client application has been implemented as a web application to provide access to the distributed database, showing it as a single storage to the end user. It is deployed in a small centralised server to access by all the users within the network. End user can perform file operations similar to operating cloud storage.

In the architecture of the system, the availability of facts, fault-tolerance, operational simplicity and scalability have been addressed. And, for the purpose of balancing the load/traffic generated by the client application, a load balancer has been integrated ensuring the system does not interrupt the other operations of the workstations.

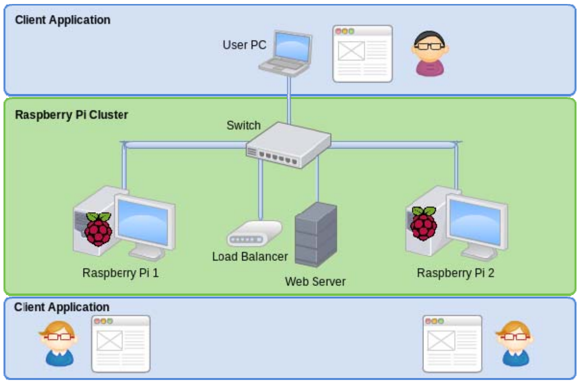


Figure 3.1 Architecture of the Solution

A. Tools and Technologies

There were lot of distributed systems developed and they could be narrowed down to small list, containing MongoDB, Riak and Couch DB. Out of these, Riak was selected as the suitable one by considering architectural and feature comparison.

TABLE I
COMPARISON OF DISTRIBUTED STORAGE SYSTEMS

Distributed Storage System		
Mongo DB	**Riak**	**Couch DB**
Written in C++	Witten in Erlang, C and JavaScript	Written in Erlang
Retains some friendly properties of SQL	Higher Fault tolerence	Easily usable consistent database
Protocols : Custom, BSON	Protocols : HTTP/REST	Protocols : HTTP/REST
Master/Slave replication	Masterless multi-site replication	Bi-directional replication
JavaScript expression queries	Map/reduce in JavaScript or Erlang	Embedded map/reduce views
Best used for dynamic queries and perform better on large databases	Best used for dynamo-like data storage with less bloat and complexity when single site scalability is required	Best used for accumulating occasionally changing data

Riak provides good single site scalable, fault tolerant distributed storage. Apart from these features, Riak is designed for commodity hardware and each node of a Riak cluster plays the same role.

The second component of the system is the client application to access the distributed storage. In this component, the goal was to implement a web interface which is lightweight and works on a web server with limited hardware. And, reliable data transportation between client and the core system was also expected. Considering those factors, HTML5, Twitter Bootstrap, KnockoutJS. PHP, JQuery and HAProxy were selected to implement the client side application.

IV. IMPLEMENTATION

Since the system has to be deployed on a multinode environment, it was required to have a similar environment for the developments. During the initial development stage, this was catered by Riak itself. Riak has the built in support for simulating a four node development environment in the single machine itself.

The core functionalities of the Riak distributed storage system were tested using the Riak development environment. Riak HTTP API was used to communicate with the storage facility.

After the initial development stage, it was required to deploy the system on an environment where limited resources are available, and test the system performance. This requirement could be catered using a cluster of Raspberry Pis in which limited hardware resources are available. This provided a suitable production environment for testing the system. So, Riak was deployed on the two Raspberry Pis and they were interconnected via an Ethernet switch.

Then the client to access the storage facility was implemented on top of the twitter bootstrap boilerplate code. The web interface was built using the UI components provided with bootstrap, and the communication between the client and the storage facility was implemented using the functions provided with jQuery. Client application was implemented with the functions to store, view and delete key/value pairs.

Each Riak node has inbuilt RESTful api to access its storage. But specifying a single destination id in the HTTP calls generated by the client was identified as a bad practice due to the fact that there is no guarantee of the availability of a single node and directing all the traffic to a single node introduces an additional overhead to the node resulting in a breakdown. Hence, it was required to keep track of all the nodes and route the traffic among the nodes as the load is balanced. As a solution for this problem, a load balancer was implemented in between the cluster (storage system) and the client application. The client is directly connecting to the load balancer and the load balancer route the requests to an available nodes as the load is balanced. This was achieved using HAProxy which is a lightweight, powerful load balancer.

V. RESULTS AND ANALYSIS

The system was evaluated by varying the factors of network throughput, processing power of nodes, node failures, size of files etc. The system responded consuming considerably low resources like processing power and memory usage from the nodes, resulting in no impact on the existing processes of the nodes. And, since the transactions happen only within the local network, internet bandwidth usage is zero, and the bandwidth usage of the local network was at a considerably low level.

The system was tested for handling files larger than 50MB. But due to the limitations of the Riak community edition, it couldn’t handle larger files. As a solution to this limitation, Luwak which is a third party service layer for Riak can be integrated. Luwak can handle the large files by breaking into multiple blocks.

The load balancer helps to distribute the traffic among the nodes within the distributed system. Therefore, it does not make an overhead traffic to a single node, leaving the system suitable to be used for general purposes.

VI. CONCLUSIONS

This paper described a storage solution which can be deployed on top of commodity hardware where the mutual trust is not assumed. The suggested system has been tested against selected key factors which can be identified as capable of handling large files, resource usage and node failures. Under these conditions, the system performed and operated ensuring average level of availability and fault tolerance leaving the node to be used for general purposes. Moreover the system requires only commodity hardware making it possible to provide a low cost solution. This solution reduces the disk space wastages of workstations and there is no requirement for skilled personnel for maintenance.

REFERENCES

[1] Dynamo: amazon’s highly available key-value store - Giuseppe DeCandia , Deniz Hastorun , Madan Jampani , Gunavardhan Kakulapati , Avinash Lakshman , Alex Pilchin , Swaminathan Sivasubramanian , Peter Vosshall , Werner Vogels - 2007

[2] Bigtable: A Distributed Storage System for Structured Data - Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach Mike Burrows, Tushar Chandra, Andrew Fikes, Robert E. Gruber – 2007

[3] Apache Hadoop : http://hadoop.apache.org/

[4] CAP Theorem - Wikipedia article : http://en.wikipedia.org/wiki/CAP_theorem

[5] Using Gossip Protocols For Failure Detection, Monitoring, Messaging And Other Good Things: http://highscalability.com/blog/2011/11/14/using-gossip-protocols-for-failure-detection-monitoring-mess.html

[6] Riak Documentation : http://docs.basho.com/riak/

[7] Ceph, A distributed object store and file system - http://ceph.com/

[8] Andrew S. Tanenbaum and Maarten Van Steen. 2001. Distributed Systems: Principles and Paradigms (1st ed.). Prentice Hall PTR, Upper Saddle River, NJ, USA.

Conversion of an Ordinary Computer Screen to Touch Screen Using a Webcamera

D.J.E. Davidson, B. Gunenthiran and K.Samarakoon
Department of Computer Engineering, Faculty of Engineering,
University of Peradeniya, Sri Lanka

***Abstract*—this paper presents a low cost conversion system of an ordinary computer screen to touch screen. A user with an ordinary computer screen may like to experience a touch compatible operating system, software or game, but to experience these, the user needs to have costly touch screens. This paper presents a low cost solution to make a virtual touch screen using a webcam and image processing techniques. In addition to a webcam, the user needs to have an infrared led pen to signal contact of touch. The webcam is equipped with an infrared-only pass filer to see infrared signals which are analyzed to make virtual touch screen.**

Keywords—touch screen, webcam, image processing, virtual touch screen, light blob, infrared

I. Introduction

Since the arrival of the iPhones and smart phones, touch screen technology became extremely popular and it gives a greater interaction between user and device. Therefore the software developers tend to make their products compatible to give better touch experience. Popular companies like Microsoft and Google are actively working in this area. Microsoft recently released an operating system which is specially designed for touch screens.

The touch experience has been achieved by combinations of deferent technologies and most of the technologies are expensive. Converting the existing non-touch technologies such as LCD, LED displays to touch compatible devices would also be expensive. An average user without any touch screens can have the same experience with an ordinary screen. This can be achieved with aid of conversion systems. Since the average user cannot offer a high cost for the conversion, this project introduces a low cost solution using a web camera and image processing techniques. Since web cameras are cheap and easy to find, they were chosen for the low cost solution.

The primary use of this system is to work with an ordinary laptop or PC screen, but the system could able to interact with a projected computer screen, so that the projected screen works as a touch screen.

II. Literature Survey

As touch technology became popular, the PC users had a desire to experience touch technology and they gradually moved towards touch screens. But the transition was too costly for an average user. Therefore some of them wanted a low cost solution to convert an existing screen to touch capable screen. Due to innovation of people, several cheap technologies were immerged. The following are some existing solutions which can transform any old regular monitor into a touch screen monitor of sorts.

NAVIsis—pair of sensors that are attached to the upper corners of the monitor, a special stylus pen and their own proprietary software makes the screen as a touch screen one which costs around $150 [1].

Touch screen overlay kit—a touch sensor is fixed over the existing screen and it is connected via a USB port. This setup makes an old screen a touch screen. This costs around $100 - $200 [2].

FTIR Screen—screen uses Frustrated Total Internal Reflection (FTIR) to detect the touching points with a web camera. Then the system analyses coordinates and handle the mouse or touch gestures. The system is not a commercial product and it may costs $50 - $100 [3].

Wiimote Whiteboard—Wii Remote is the primary controller for Nintendo's Wii console [4]. A main feature of the Wii Remote is its motion sensing capability, Since the Wiimote can track sources of infrared (IR) light, system can track pens that have an IR led in the tip. By pointing a wiimote at a projection screen or LCD display, interactive whiteboards can be made [5].

Webcam Whiteboard (WW)—WW makes use of the ability of a webcam to see infrared (IR) light. The tracking engine can detect blobs. Using a pen with an infrared LED in the tip, the position can be tracked [6].

NAVIsis [1] and Touch screen overlay kit [2] have more accuracy and quick response but they are costly. FTIR screen [3] is not practicable since it needs more space to setup. Wiimote Whiteboard [5] is moderately fast and costs moderate too, but it needs a manual calibration. Webcam Whiteboard [6] is similar to this project. It has to be calibrated manualy and don't support different lighting conditions. This paper proposes some advanced techniques such as *autothresold* and *autocalibration* to overcome the above mentioned problems while keeping the cost low as possible.

III. Methodology

In this project, several approaches were tried, such as using two vertical cameras, detecting mirror images and using infrared LED pen mode. The primary task of all of the approaches are finding touching coordinate, at the moment when user touches the screen surface. Then touching coordinate is passed to the operating system and then it moves the mouse cursor to the touching coordinate.

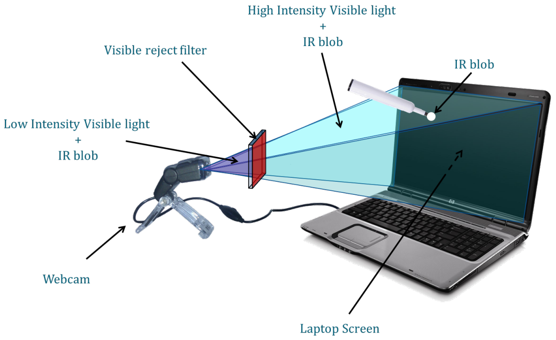


Fig. 1. The Solution

Initially the idea was to use two web cameras simultaneously to find horizontal and vertical coordinates. When touching the surface of the screen, one camera calculates horizontal distance from top left corner of the screen while other camera calculates vertical distance. These distances are the coordinate with respect to upper left corner.

Since the field of view of web cameras are around 45 degrees, some part of the screen are not covered by both the cameras and some by one camera. Therefore the second idea immerged.

This second idea was to detect mirror image of the pointing finger. At the touching point both the finger's tip and mirror image's tip coincide, at that moment touching coordinate is calculated. In order to recognize mirror image, the screen surface should be reflective, but most of the latest laptop screens are not reflective (shiny). LED screens are good examples for non-reflective screens.

Instead of fingers, LED pen which has a small LED on the tip is used. This small light could be seen on the camera when reflected or even on a dump surface. But the problem was to distinguish that small light on a bright background. Both the light and reflection could not be identified on bright background.

When we use a visible light rejecting filter (which blocks visible light and passes infrared light) and an infrared LED instead of the normal LED, we could distinguish the small blob of light even on a bright background. Because the bright background is filtered by the filter and, only IR light is passed, which can be seen on the camera. Film negatives and colour cellophane are simple materials that can be used as a filter.

A. The Solution

Among the tested methods visible light filtering was the best and it was chosen as the solution. A visible light rejecting filter in between camera and the screen was placed to give the ability for the camera to see IR blob. see Fig. 1. Then the coordinate of light blob is calculated and passed to the operating system with appropriate mouse messages. The following sections elaborate on the implementation of the proposed solution.

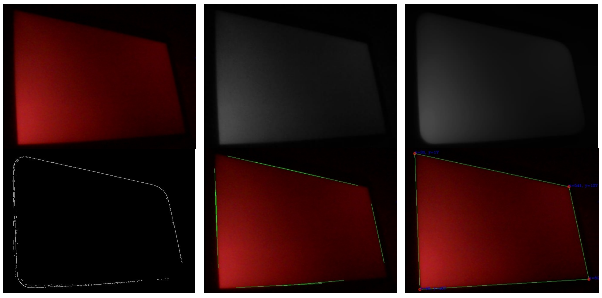


Fig. 2. (i) Filtered image. (ii) Gray image of (i). (iii) Median filter applied on (ii). (iv) Canny edge detection applied on (iii). (v) Line segments are detected on (iv). (vi) Solved corner points using line segments on (iv).

IV. Implementation

A. Get image

In order to get image input a USB 2.0 web camera is used. This web camera should be placed in such a way that whole area of the screen is visible in order to get full view.

B. Find Corners

The necessity of finding corners is described in section IV-D. This section explains how to find the corners. The process of finding corners is also called *Calibration*.

In this project two type of calibration is implemented, auto and manual. Before finding corners, a calibration process called *Autothreshold* (see IV-C1) is run. When the user starts execute auto or manual calibration the autothresold is run.

1) Auto Calibration: Auto calibration is done automatically by the software system at the startup. It will find four corners of the screen automatically. The following describes the procedure.

First, a colour image is received from web camera. Since an IR filter is used in front of the web camera lens, a low quality image results. Therefore a white screen is displayed to identify the screen area. Then the colour image is converted into gray image and then median filter is applied with several mask sizes starting from 3 to 35 until all salt and pepper noises are removed. After that Canny edge detection is performed to the resulted image. Then Hough line detection algorithm is applied to get line segments with maximum and minimum line size and gap specifications. Since the edges of the screen are around 90 degrees to each other, lines parallel to each other are discarded. Intersection points are found by solving remaining line segments and then the points are sorted into groups (upper-left, upper-right, lower-right and lower-left) and averaged to get four corners' coordinates. see fig. 2.

2) Manual Calibration: Whereas manual calibration waits for the user to specify corner of the screen until it detects all four corners. The procedure to detect the point when user specifies a corner is explained in section IV-C.

C. Find Touching Point

The colour image received from web camera is converted into gray scale before applying smoothing filter. After smoothing, threshold function is applied to the resulting images with

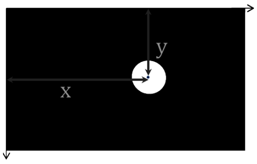


Fig. 3. The touching point

a specific threshold value. The threshold value depends on lighting condition of the surrounding environment and it is found using Auto threshold function (section IV-C1). Result is a binary image. Next step is to find the centroid of the white dot or blob. Equation 1 calculates the centroid of white blob in fig 3.

$$(x, y) = \frac{\sum(coordinatesofwhitepixels)}{numberofwhitepixels} \quad (1)$$

1) Auto Threshold: The lighting condition of surrounding environment of a laptop or screen may vary. Bright lights, reflections, doors and windows in the background may cause the inaccurate blob detection, i.e. centroid of white blob is inaccurate. Therefore a dynamic threshold value is needed depending on the surrounding lighting conditions.

Initially threshold value is set to zero and then it is increased one by one and applied to image until the resulting image just becomes black, which means all the gray level values of the image are lying under the current threshold level including the background. The IR blob's gray levels always stay above this current threshold level while backgrounds gray level values lie under. This is done to ensure accurate blob detection without background lighting interference.

D. Perspective Transformation

When an image is received from web camera, the shape of the screen in image is not rectangular because the camera sees the screen at an angle rather perpendicular. The shape is more like distorted rectangle, a quadrilateral shape. Therefore it is necessary to wrap quadrilateral shape into rectangular shape. To do this transformation, four corners of the screen are needed.

To get actual touching coordinate, a transform matrix is found using the corner points which were found when calibrating and the original corners of the rectangular screen. Touching coordinate is found by multiplying the matrix is with the touching coordinate of the distorted rectangle (section IV-C). Only the touching points is transformed instead of transforming whole image. This reduces lot of computations. Then transformed point is used to generate mouse events.

E. Generate Mouse Events

Generating mouse events and setting current cursor position are executed separately. One function sets cursor to current touching coordinate when a blob is detected. This function does not generate mouse events but enhances the cursor movement. Another function generates mouse events when a state change occurs.

Event generating function observes the blob and updates a Boolean variable. If this variable is *TRUE* indicates a blob is detected on the screen, *FALSE* indicates no blob is detected. A Blob can be detected only when user touches the screen.

V. Result and Analysis

When comparing with other existing solutions, this project has several advantages as well as some disadvantages. This solution is low cost and easy to use. User just needs to plug-in webcam and start the software. The software calibrates automatically, which means no manual calibration procedure is needed. But in all other solutions a manual calibration procedure is done. In addition to the auto calibration, an auto threshold function is applied to work with different lighting conditions.

Since a filter on top of the webcam is used, the amount of light going into the webcam is very low. This makes the camera operate in low frame per second (fps). Hence there is a delay of 140ms. The other concern is the usage of infrared pen which is not comfortable as using a finger to touch and can damage the screen surface.

VI. Conclusion

This report presents a method to convert an ordinary laptop screen or a projected screen into a touch screen using a web camera and image processing techniques. Moreover this solution is a low cost and can be used by any average users. The system gives the user to experience touch compatible software and operating systems with existing technologies such as LCD or LED displays which do not support touch.

References

[1] Gizloco. (2006, May) Navisis turns regular monitors into touch screen monitors. [Online]. Available: http://gizmodo.com/173203/navisis-turns-regular-monitors-into-touch-screen-monitors

[2] Cyclotouch.com.au. Turn any lcd into touch screen. touch screen panel convert kit. [Online]. Available: http://www.cyclotouch.com/

[3] T. M. Brand. (2007, March) Ftir multitouch and display device - experiments with processing, osc - thomas m. brand. [Online]. Available: http://lowres.ch/ftir/

[4] nintendo.com. (2013) Wii official site - what is wii? [Online]. Available: http://www.nintendo.com/wii/what-is-wii/

[5] J. C. Lee. (2007, Dec) Johnny chung lee - projects - wii. [Online]. Available: http://johnnylee.net/projects/wii/

[6] (2010, March) Download webcam whiteboard. [Online]. Available: http://webcam-whiteboard.apponic.com/