Nucleus - A P2P Image Sharing App Employing Compression

CS4089 Project

End Semester Report

Anand.M.P (B120361CS)
Ronn George Jacob (B120189CS)
Sreeraag Mohan Thekkepurakal (B120469CS)
Guided By: Ms. Pournami P.N

November 10, 2015

Abstract

In a world where media is the most important form of data, we aim to create a peer-to-peer image sharing application that implements a lossy image compression algorithm for minimum loss in the quality of the application. This would also have an added advantage of saving up storage space on a users device where the device would not need to have a higher resolution image due to its limitations. To reduce the time between transfers of files between peers, WiFi transfers have proven to be 50 times faster than Bluetooth as well as other short-wave radio communication techniques. Reliability being a key factor for any software in today's world could be ensured by the above transfer technique over other methods for transfer. The range for the transfer to take place and limitations are worries of the past with the emergence of WiFi. Applications have a tendency to drain batteries thus making it unproductive and hence techniques to minimize power consumption would be employed.

1 Introduction

We aim to implement Nucleus, an application that would enable users to share images over an ad-hoc WiFi network which would allow for faster transfer times and minimal power consumption. This would be a Peer to Peer application which would disregard issues involving central dependency and configuration. A lossy compression algorithm based on human visual perception measurements would be chosen with the help of a study comparing existing lossy compression algorithms. The resulting algorithm would be used in the application. The users would be able to create groups allowing them to interact with other users of Nucleus for transfer of compressed images. Users would be able to join existing Peer to Peer groups and send compressed images over a network. Sending compressed images helps in overcoming a key issue faced by mobile devices which would be the depletion of storage space.

The end semester report aims to provide an insight into the development of Nucleus which has presently completed its design stage. It would also provide the reader with a brief idea about what the team intends to do in the implementation phase of the project.

2 Problem Statement

To implement a peer-to-peer image sharing application that uses an optimal compression algorithm to send compressed image files over an ad-hoc Wi-Fi network.

3 Literature Survey

To learn the rudimentary concepts of Android and the essentials required to develop in the Android Studio environment, a few resources were utilized. Neil Smyth's Android Studio Development Essentials [1] and Google's official developer site and public forums [2] provided us with valuable insight on the process. WiFi direct and the WiFi P2P functionalities were explored using documentation provided at [3].

Three image compression standards were selected to find the best one that suited the objectives of the project. The fundamentals of the JPEG Image Compression Standard were provided by Wallace in [4]. JPEG2000, an evolutionary upgrade over the JPEG standard, and it's comparative benefits were learnt from Singh and Srivastava's paper [5]. However, we soon learnt that JPEG2000 is inherently more complex and it's usage on a device with limited processing, storage and energy capabilities would lead to a significant decrease in the performance of the application, and was thus eliminated from the comparative study. Google's WebP, a format described as one made for the web, was also analyzed in our study.

To compare the JPEG and WebP compression algorithms, multiple techniques were explored before finally settling on the SSIM index. SSIM, as opposed to Mean Squared Error (MSE) and Peak Signal Noise Ratio (PSNR) techniques, is a full reference metric and thus measures the quality of the image with the initial, uncompressed and distortion-free image as a reference. The working and advantages of the metric were studied from Wang, Bovik et.al's paper [6].

Power optimization was also keenly studied by the team and the best practices to reduce the load on the processor were explored. Yereaztian and Luthiger's paper [7] that provided us with practices such as write amplification and necessary algorithms and data structures to use for the task. Udacity's online course [8] also proved helpful to achieve the same purpose.

To complete the design phase of the project and to prepare the use case diagrams and the Software Requirements Specification (SRS), Lethbridge and Laganiere's classic textbook on software engineering practices [9] proved very helpful.

4 Work Done

4.1 Identifying Compression Algorithms

To determine the best algorithm to use in the application, we initially considered three compression algorithms - JPEG, JPEG 2000, it's evolutionary upgrade, and WebP, an image format developed by Google for the web. However, as part of our literature survey, we were able to deduce that JPEG 2000, due to its inherently complex code and large storage footprint, would be taxing on the Android device which has severe performance, storage and energy limitations. As a result, the algorithm was not considered for our comparative study using the SSIM metric.

4.2 Comparative Study of Lossy Compression Algorithms Using SSIM Metric

Aim:

The aim of the study was to choose a lossy compression algorithm that would suffice the requirements of our project.

Methodology:

For the purposes of the study, we considered two popular compression algorithms, namely JPEG and Googles WebP. An implementation of the Structural Similarity index (SSIM) was developed on JavaScript for the same.

The SSIM was run on three types of images: a greyscale image, a colour landscape, and a portrait image. The images were compressed at various bitrates, and the SSIM index was calculated for each of these images to the original, uncompressed image. The results so obtained were graphically represented and the algorithm which provides the best SSIM score over the entire range was chosen.

The SSIM index is calculated on various windows of an image. The measure between two windows x and y of common size N x N is

$$\text{SSIM}(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}$$

- μ_x the average of x ;
- μ_y the average of y ;
- σ_x^2 the variance of x ;
- σ_y^2 the variance of y;
- $\sigma_x y^2$ the covariance of x and y;
- $c_1 = (k_1L)^2, c_2 = (k_2L)^2$ two variable to stabilize the division with weak denominator;
- L is the dynamic range of the pixel-values (typically this is $2^n - 1$), where n is the number of bits per pixel;
- $k_1 = 0.01$ and $k_2 = 0.03$ by default.

Observations:

The three distortion-free images were compressed at a range of widths, starting from 320px to 1024px, keeping the original aspect ratio the same. The compressed images were then paired up with the original, and the SSIM index was calculated for the same. This process was repeated for all the three images that were sampled during the course of the study. The SSIM index ranges from a value of -1 to 1. For the ease of its graphical representation, the SSIM_DB was calculated.

$$SSIM\_DB = -10.0 * log_{10}(1.0 - SSIM)$$

Conclusion:

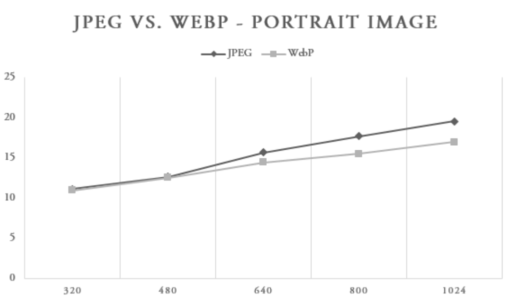

Figure 1: *Results obtained for the comparison on a portrait image*

From the comparitive study, it was evident that JPEG outperforms WebP. The algorithm also provides advantages such as being computationally less intensive, thereby placing minimal strain on the limited processing capabilities of the device. The SSIM index was found to be higher over the range of the study in the case of the JPEG compression algorithm, as opposed to WebP, which brought us to the conclusion that the JPEG algorithm is the better choice of algorithm for our application.

4.3 Software Requirement Specification Documentation

The team then proceeded to develop the use case diagrams and model the various interactions that a potential user would have with the application. These were then documented as a Software Requirement Specification document, based on existing IEEE standards. The SRS document would help the team in their implementation of the application and would also serve as a guide in the Software Development Life Cycle. (SDLC).

4.4 Power Optimization

We also wanted Nucleus to be battery friendly and use the resources available to it wisely. A wide range of battery optimization techniques were studied and their benefits and performance tradeoffs were evaluated. Nucleus would actively pursue a policy where the network signal strength, the battery levels of the device, and the CPU usage history would be taken into account before making a transfer.

Themes were also found to be helpful in reducing battery usage, as darker colours are found to use lesser battery than brighter colours. This conclusion was a major motivation behind our team deciding to make Nucleus's UI available in two flavours, a dark theme and a light theme. The developers of the application will also make sure that the application runs on a minimum number of threads at any given time, thereby minimizing the impact on the CPU usage.

5 Future Work and Conclusions

From the study, it is evident that JPEG outperforms WebP. The algorithm also provides advantages such as being computationally less intensive, thereby placing minimal strain on the limited processing capabilities of the device. The SSIM index was found to be higher over the range of the study in the case of the JPEG compression algorithm, as opposed to WebP, which brought us to the conclusion that the JPEG algorithm is the better choice of algorithm for our application. Battery Historian, a tool built into Android Studio to monitor battery statistics, will also be used to monitor battery usage statistics for the code.

Nucleus will enter its implementation phase as soon as the basic mockups for the interface have been designed. The team aims to actively use a Version Control System (VCS), Git, so that the team can actively collaborate on the project, even while working remotely. Modularity, polymorphism and other OOP paradigms would also be implemented in the project, making code reuse and maintanence easy.

References

[1] Neil Smyth, *Android Studio Development Essentials*, Second Edition

[2] Android Developers, developer.android.com, Google Inc., 2015

[3] Android Wi-Fi P2P, http://developer.android.com/reference /android/net/wifi/p2p/package-summary.html

[4] Wallace, G.K.,*The JPEG still picture compression standard* , in Consumer Electronics, IEEE Transactions on , vol.38, no.1, pp.xviii-xxxiv, Feb 1992

[5] Singh, R.; Srivastava, V.K.,*JPEG2000: A review and its performance comparison with JPEG* , in Power, Control and Embedded Systems (ICPCES), 2012 2nd

International Conference on , vol., no., pp.1-7, 17-19 Dec. 2012

[6] Zhou Wang; Bovik, A.C.; Sheikh, H.R.; Simoncelli, E.P.,*Image quality assessment: from error visibility to structural similarity* , in Image Processing, IEEE Transactions on , vol.13, no.4, pp.600-612, April 2004

[7] Chris Yereaztian, Jurg Luthiger, *Android Best Practices to Improve Battery Efficiency*

[8] Colt McAnlis, Chris Lei, *Android Performance - Optimizing Apps for Speed and Usability*, udacity.com/course/android-performance–ud825

[9] Lethbridge, Timothy and Laganiere, Robert, *Object-oriented software engineering: Practical Software Engineering Using UML and Java*, Mcgraw-Hill College, 2001