Development of a Super Resolution Image Enhancement plug-in for The GIMP

Submitted in partial fulfilment
of the requirements of the degree
Bachelor of Science (Honours)
of Rhodes University

Stephen Hawkridge

November 6, 2006

Abstract

This paper outlines the current models of super resolution and their merits, and illustrates the development of the most promising method as a plug-in for The GIMP. The process of Super Resolution uses multiple low resolution images to construct a single high resolution image which is clearer. The limitations of this implementation are illustrated by the output from the filter being less clear than is expected.

Acknowledgements

I would like to thank my supervisor for his continual help throughout the year. Also, special thanks to the sponsers of the Rhodes computer science honours program, Telkom Business Connexion, Thrip, Verso, Stortech and Tellabs, for their contribution of hardware and facilities.

Contents

List of Figures

Chapter 1

Introduction

Using a sequence of low quality, low resolution images as an input, the Super Resolution filter will combine information from these and produce an output of a single high resolution image which is clearer. The problems associated with low quality, under sampled images have been evident since the first digital image capture devices were introduced. With the introduction of cheap image capture hardware to devices such as cellular phones and low cost digital cameras, the discovery of a solution to this problem has become increasingly important. Simple interpolative, blur or sharpen filters do not offer the sort of quality improvement expected by most users and the limits of their effectiveness are now being reached. The problem of decreased clarity in these methods is due to the lack of data obtained by a single digital image. This problem can be overcome by the introduction of multi-frame image restoration techniques. Where single frame restoration must introduce outside data to compensate for the lack of high resolution information, the multi frame restoration systems can obtain data from the successive frames to improve the restoration technique. This allows for high resolution information, which is lost in a single image, to be obtained using the combination of multiple images. Although this is still a fairly new field, the results obtained are promising and illustrate that Super Resolution can overcome the limits previously imposed on interpolative filters.

1.1 Background

Captured images are stored digitally using pixels, allowing each point to contain information about colour data and other attributes. This causes the data to be quantized at a fixed resolution, which limits the amount of information that is stored. A low resolution image will not contain the high resolution components needed to create the clarity seen in an analogue photograph. Various methods of overcoming this problem have been introduced, starting with simple single image interpolative filters.

1.1.1 Single Frame Image Enhancement

Many techniques have been proposed for the field of image enhancement, such as those outlined by [Farsiu et al., 2006],[Baker and Kanade, 2000] and [Szu and Kopriva, 2001], each with its own merits with regard to computational efficiency and level of enhancement. The most common of these deals purely with the application of interpolation, blur and sharpen filters, but the area of single image enhancement extends to the idea of texture replication with the use of neural networks. Both offer a superior level of computational efficiency to Super Resolution, but neither produce the enhancement seen in most Super Resolution applications.

1.1.1.1 Standard Filters

Early attempts at image enhancement were limited mainly by the computer hardware available, and so offered little improvement. This requirement caused the filters to work quickly on low speed computers, a factor that has been included for all interpolative filters since then. These filters can be implemented in either the spatial or the frequency domain, each method offering advantages and disadvantages.

Spatial Domain - Histogram Equalization

This is a common filter offering acceptable quality improvement with good computational efficiency, and relies on the histogram distribution of the input image. Using equalization of this data, a dark image can be lightened, a light image darkened, or a contrast balance corrected. Most unclear images produce histograms which are compressed into a particular area. By correcting this using histogram equalization, the grey levels are increased to balance the compressed data.

Spatial Domain - Image Smoothing

Images are often blemished by information which is incorrectly captured or recorded by cameras. Image smoothing filters provide a mechanism for overcoming this problem. The simplest form is produced in a blur filter combining information from a 3x3 neighbourhood to produce the resultant pixel information. It is evident that information in a particular pixel is more closely related to those in close proximity than those further away, thus the introduction of a Gaussian blur improves performance with regard to edge preserving. Although Gaussian filters are an improvement over simpler methods, they still tend to blur edges and are not an optimum solution. A median filter produces a further improvement with regard to blurred edges. High frequency components are often lost in image manipulation and sharpening tools are used to amplify available data to give the appearance of an increase in clarity. The formulation of this is based in high pass filtering, and the application essentially works as a converse to the simple blur function.

Frequency Domain - Filtering

By using data in the frequency domain, filtering techniques can be implemented in an even more efficient manner. Furthermore, a wider variety of filters is available for calculations made in this domain. The major disadvantage of using the frequency domain is the introduction of blurring and ringing.

1.1.1.2 Texture Replication

The greatest problem facing single image restoration is the lack of high resolution data. One method of overcoming this, shown in [Pickup et al., 2003], uses the idea of texture replication to introduce the missing information. Sometimes referred to as Super Resolution, this method does not deal with multi frame restoration, but rather relies on a filter trained on a chosen set of data to introduce data obtained from a training set.

Using high resolution and matching low resolution data, the image is broken into pieces, known as patches. Each patch in the low resolution image corresponds to one in the high resolution frame. The high resolution image contains the data which is used to introduce the high resolution information into the target image. The first step of the process is to use an interpolative filter so that the low resolution frame is expanded to its new size. The low resolution frame is then divided in the same way as the training data, and each square is compared to the training set until the closest match is established. The equivalent high resolution data can then be added to the current high resolution estimation. At each step the edges of the high resolution patch must be matched, as far as possible, to the previous edges so that the steps can flow from one section of the image to the next. Once the whole image has been constructed from the high resolution patches the result is blurred with the high resolution estimation obtained from the interpolated low resolution frame.

The results obtained from this form of restoration show promise, but do not have

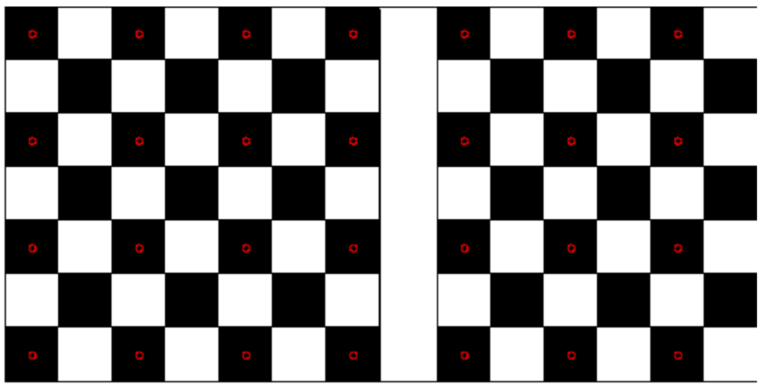


Figure 1.1: Shifted Images

the same accuracy as other methods. The main problem associated with this method of restoration is the introduction of foreign data from the training set. The information contained in the training set can influence the data required for the high resolution output negatively causing it to be unusable for identification requirements.

1.1.2 Advancing to Multi-Frame Enhancement

The use of multiple images for the process of resolution enhancement poses problems when the images are shifted with reference to each other. This is usually overcome by using a motion estimation algorithm [Farsiu et al., 2005], [Milanfar, 1999]. The idea of optical flow, relating the transformation from one frame to the next, offers a good assimilation of motion within specific parts of the image [Baker and Kanade, 2000].

Multi frame super resolution consists of interpolation and blending. After the position of each image is matched the images are interpolated to the higher resolution by an inverse decimation and then combined to form the output. Using the information from multiple images the missing high resolution components are placed in the right position. For this process to be successful information in each low resolution needs to differ, but it needs to be correlated to information in the other images. This requires the calculation of the position of each image relative to the others or the motion from one to the next. Motion estimation is essential to the correct matching of each pixel.

Due to the limitations posed by single image enhancement, such as the inherent lack of high resolution data, the introduction of multi-frame enhancement allows for the reintroduction of these elements. In the process of image capture, the single capture process is replaced by a multiple capture process where each frame is shifted from the previous frame's position, as in figure 1.1. The contents of the corresponding shifted images are subtly different, due to the nature of image capture devices and the noise that is caused by the capture procedure. This change provides more information than the single image and the possibility of reintroducing this to a higher resolution representation. This shift does however introduce the requirement of difference estimation before processing can be done on the image, seen in section 2.2.

The basic idea behind the multi frame restoration process is to integrate the data obtained from the sequence of low resolution images. This process is represented as an inverse problem. It is a simple calculation to obtain a series of low resolution images from a high resolution equivalent, but it is far more complicated to reverse this process and obtain the high resolution image.

One of the most important components of the calculation is that of error minimization as it affects the rate at which the optimal image is reached and affects the overall efficiency of the algorithm. Various methods are outlined in chapter 2. Currently one of the most optimal methods is that seen in [Farsiu et al., 2003] which is outlined in section 3.2.

1.2 Document Structure

The structure of the remainder of this document is as follows:

Chapter 2 illustrates the current and prior work related to image enhancement in general and specifically super resolution.

Chapter 3 outlines the workings of the Super Resolution with regard to colour and greyscale and the algorithm used in the method.

Chapter 4 deals with the implementation as an entity and the collaboration with the GIMP.

Chapter 5 presents the results obtained from implementing the filter with emphasis on the input variables and their influence on the outcome.

Chapter 6 discusses the conclusions obtained from the work in this thesis.

Chapter 2

Related Work

A vast amount of theoretical research has been done in the field of super resolution, but until now the methods used were not feasible as filters for home users because they needed expensive computers for them to complete calculations in a short amount of time. This chapter outlines the development of super resolution from its inception to its current point and gives a brief overview of some literature available on the topic of motion estimation. The Super Resolution section is divided chronologically begining with the first paper published on Super Resolution,[Lucas and Kanade, 1981], and ending with the paper used for the development of the plug-in implemented in this project.

2.1 Super Resolution

2.1.1 Initial Ideas

Early super-resolution methods by [Lucas and Kanade, 1981] and [Tsai and Huang, 1984] were limited in their capabilities, but have had a strong influence on the implementations since then. New ideas of super-resolution as an iterative inverse-problem are introduced in [Irani and Peleg, 1991], which further outlines the problem as ill-posed and demonstrates the requirement of regularity constraints. These ideas are developed in [Lorette et al., 1997] which introduces the possibility of a contradiction between super-resolution and regularization. By making the process a calculation of local correlations, [Candocia and Principe, 1999] propose a relation between the pixel in question and those around it. This process has the potential to be suboptimal as it does not rely directly on the error calculations for the super-resolution.

Image restoration methods are commonly classified into three main categories namely Maximum Likelihood Estimator (ML), Maximum A Posteriori Estimator (MAP) and Projection onto Convex Sets (POCS).

[Elad and Feuer, 1997] propose a combination of these algorithms and revises the promising algorithms of [Irani and Peleg, 1991] with the use of space invariant blur functions and illustrates the importance of motion estimation within the super-resolution calculation. This method also relies on a process of back projection to obtain a progressively improved image. Furthermore the paper illustrates the possibility of motion-free super-resolution whereby a series of images is captured from a fixed viewpoint, observing motionless objects.

2.1.2 Progressive Development

Turning to the POCS algorithm, [Patti and Altunbasak, 1998] introduce some improvements to the algorithm, specifically allowing for higher order interpolation methods by improving the discretization process. This does not deal with the relation between Super Resolution and motion estimation which is more clearly defined by [Baker and Kanade, 2000] which illustrates the negative effect producing artefacts around objects with incorrect translation of the source images. Due to the computational power required for previous Super Resolution techniques, [Nguyan and Golub, 1999] address the issue of performance and uses relation between consecutive frames to minimize complexity.

Following on from previous implementations, [Bhattacharya, 2000] analyses the performance of the POCS method, but relates the requirement of accurate point spread function and aperture knowledge for successful filtering.

Outlining the inefficiency of previous attempts, [Zomet et al., 2001] provide support for the argument that outliers will have a substantial accuracy cost on non-robust motion estimation engines and show how MAP solutions often employ a smoothness function to eliminate unwanted artefacts. These methods, although offering improved outputs, still lose accuracy. Further work is seen on motion segmentation, but this still does not prove to be robust to outliers. Thus the main section of this work deals with the introduction of a robust method which is less susceptible to inconsistencies due to outliers and offers improved accuracy for inliers.

2.1.3 Disputed Ideas

In opposition to this idea, [Sroubek and Flusser, 2003] explain the shortcomings of the assumptions of accurate spatial alignment and correct estimation of the blur radius. They then propose a return to the MAP algorithm and its ability to overcome the problems related to images which are severely corrupted by noise. The previous accuracy loss is attributed to the incorrect calculation of the blur radius and is corrected herein.

Due to the promising results of super-resolution application, the possibility is sometimes deemed limitless, but, due to the nature of the data being used, limits are reached in the ability of the algorithms to construct any feasible improvement. [Boccacci and Bertero, 2003] relate the idea of computer Super Resolution to the instruments used in the image capture and illustrate the inefficiency to overcome the clear boundaries involved in restoration. Although these limits can be approached with newer mathematical algorithms, it is clear that there are limits that cannot be overcome. When an image has been degraded to a certain point only a finite number of incremental processes can be performed on the information before excess artefacts will block the relevant information.

2.1.4 Video and Zoom

Although it is sometimes thought of as limited to translational or rotational motion, applications of super-resolution can include variation in zoom factor as seen in [Joshi et al., 2004]. By modelling the function using a Markov random field (MRF) and a MAP function, the idea is proposed that an entire scene can be interpolated using images captured at various known and unknown optical zoom increments. In this case the motion estimation will be reduced to a case of alignment estimation. Ultimately, the resolution of the entire scene is obtained at the resolution of the most zoomed image.

The inevitably variant nature of video sequences poses greater problems when used as an input for the algorithms due to anomalies such as occlusion and scene changes. The optical flow method is reintroduced in [Jiang et al., 2003] which focuses on the extrapolation from dynamic video.

Computational inefficiency is highlighted when the problem of super-resolution is brought into the domain of real time video reconstruction. This problem is aggravated by the requirement of most video streams to contain some form of compression creating a greater computational overhead. Gunturk et al. [2004] suggest a Bayesian framework to overcome the related problems, and creates additional constraints using prior image models. Again the issues related to motion estimation and the requirement for accurate estimation prior to the application of the algorithm are emphasized .

2.1.5 Current Ideas

The increase in graphics hardware performance has provided a promising platform for image registration and [Vasa et al., 2005] illustrate methods of integrating the processing power of modern workstation graphics processing units (GPU). This improves computation times, but the implementation is limited in its scope and only covers the registration phase of the algorithm.

Graph-cut methods have been implemented widely for the processing of images and offer application in areas of super-resolution, [Mudenagudi et al., 2006] show the implementation in the process of approximate optimization. This follows an application of an MRF-MAP function for the interpolation. Problems are associated with this method due to its inefficiency and work is being done to improve it.

As seen previously, the super-resolution problem can be divided into two processes including a registration phase where low resolution images are aligned, and a reconstruction phase from the aligned low resolution images. Dealing with the first phase, [Zitova and Flusser, 2003] utilize a frequency domain method to perform the registration. This works by performing registration on aliasing-free parts of the image spectrum to register aliased parts.

Dealing with colour images illustrates inefficiencies in image capture architecture and the arguments put forward by [Farsiu et al., 2006] present a well rounded implementation. Ultimately this algorithm deals with the problem of processing colour images by separating the image into its respective colour domains, offering a vastly improved outcome from a group based model.

2.2 Motion Estimation

This process is essential for efficient and accurate image enhancement, as an incorrect alignment in source images has the potential to reduce the quality of the output image rather than improving it. Motion estimation can be categorized into an anchored mode, which deals with each frame in reference to the first, or a progressive mode, which calculates each frame according to the previous. Initial ideas proposed in [Lucas and Kanade, 1981] look at the flow of information from one frame to the next and use a series of previous and future images to calculate the transformation. This offers promising results for localised motion, but proves to be less robust than the methods of [Farsiu et al., 2005] when more varied input frames are used due to the influence of outliers on the calculated output. The initial ideas seen in [Lucas and Kanade, 1981] have been improved in [Baker and Kanade, 2000] and offer better estimation providing a more successful prior.

A different view, seen in [Bergen et al., 1992] proposes a hierarchical model, also seen in [Lee and Chen, 1997] which deals with the inputs one block at a time. The sum pyramid is used to eliminate blocks that do not match and shows a vast improvement in performance over previous methods.

By limiting motion to pure translation and rotation equation, [Elad and Or, 2001] allow for a more robust and simpler calculation, while still maintaintaining an acceptable error rate. A new idea presented in [Govindu, 2004] deals with lie-algebra and offers further improved performance, but again deals with the calculation on a global scale. Ultimately, [Farsiu et al., 2005] offer the most promising method with regard to robust calculation and performance.

2.3 Summary

Although the majority of work related to Super Resolution is focused on creating better theoretical processes, little work has been done on the process of implementation. Most theoretical outlines are not feasible for implementation, but those outlined by [Farsiu et al., 2003] are simple enough for implementation. This gap in available implementation is the basis of the work outlined in this report.

Chapter 3

Design

Although much work has been done on the theoretical side of the Super Resolution problem, little has been done for the implementation of methods available for common image manipulation products. Of the many available theoretical formulas only a few are feasible for implementation in the form of a filter plug-in due to their complexity or their need for fast computer hardware. This chapter outlines a few of the theoretical methods and presents a detailed explanation of the algorithm used in the development of the plug-in for this project.

3.1 Application of Super Resolution

The theoretical outline of Super Resolution has been illustrated in various previous works [Baker and Kanade, 2000; Farsiu et al., 2003; Vasa et al., 2005], each providing solutions to specific areas. The issue of robustness for motion between frames is best described by [Farsiu et al., 2005], although other papers do address this problem by introducing prior requirements. The theoretical work done in many of the papers up to this point show promise in their current form, but for the process of implementation as a GIMP plug-in, they are not applicable. The algorithms seen in the early work of [Tsai and Huang, 1984] are simpler in their implementation, but do not offer the requirements for a successful implementation of the plug-in. Other suggested methods, such as those by [Elad and Feuer, 1997] are more advanced in their theoretical application, but prove to be far too complex to implement efficiently. For a theoretical model to be successful in its implementation, it must be efficient with regard to both simplicity and speed. As will be seen in the next section, the equations outlined by [Farsiu et al., 2003] can be simplified to a series of filters or matrix convolutions. This simplicity proves to be useful in the implementation, seen in chapter 4 due to the fact that it can be modularized and offer an abstract interface using convolution. The performance enhancement seen in [Staelin et al., 2003] which introduces neural networks as a means for solving the Super Resolution problem is substantial, but is not as applicable for the development of a GIMP plugin.

3.2 The algorithm

The theoretical equations developed for the process of Super resolution prove mostly to be very complicated in their implementation. The method outlined in [Farsiu et al., 2003], although complicated in its theoretical form, simplifies well into the form of an algorithm.

$$X_{n+1} = X_n - \beta[HH^T \sum F_k^T D^T sign(DF_k X_n - Y_k)] \tag{3.2.1}$$

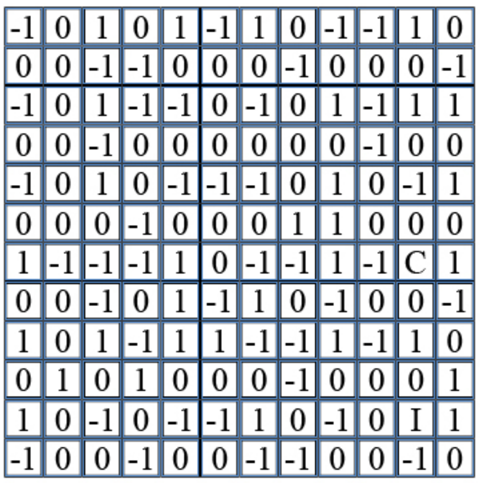
The Super Resolution equation by [Farsiu et al., 2003], seen in equation 6.1 represents the algorithm required for the process, with the components as follows:

- D_k: The decimation or down-sample/up-sample matrix
- F_k: The warp or transformation matrix.
- H_k: The blur matrix.
- X_n: The current high resolution estimation.
- Y_k: The k^{th} low resolution image.
- β: The amount that each iteration affects the current estimation.

3.2.1 The Super Resolution Algorithm Explained

The Super Resolution expression can be broken down into sections, each dealing with a particular aspect of the process. To explain the process it will be broken into three main sections. Firstly, the subtraction of the low resolution images from the current high resolution estimation.

$$(DF_k X_n - Y_k) \qquad 3.2.1.1$$



-1	0	1	0	1	-1	1	0	-1	-1	1	0
0	0	-1	-1	0	0	0	-1	0	0	0	-1
-1	0	1	-1	-1	0	-1	0	1	-1	1	1
0	0	-1	0	0	0	0	0	0	-1	0	0
-1	0	1	0	-1	-1	-1	0	1	0	-1	1
0	0	0	-1	0	0	0	1	1	0	0	0
1	-1	-1	-1	1	0	-1	-1	1	-1	C	1
0	0	-1	0	1	-1	1	0	-1	0	0	-1
1	0	1	-1	1	1	-1	-1	1	-1	1	0
0	1	0	1	0	0	0	-1	0	0	0	1
1	0	-1	0	-1	-1	1	0	-1	0	I	1
-1	0	0	-1	0	0	-1	-1	0	0	-1	0

[h]

Figure 3.1: Example of sign difference matrix

1. k is the number of the current low resolution image.

2. The current high resolution estimation Xn is used.

3. The image, Xn, is shifted using the warp matrix, then down-sampled.

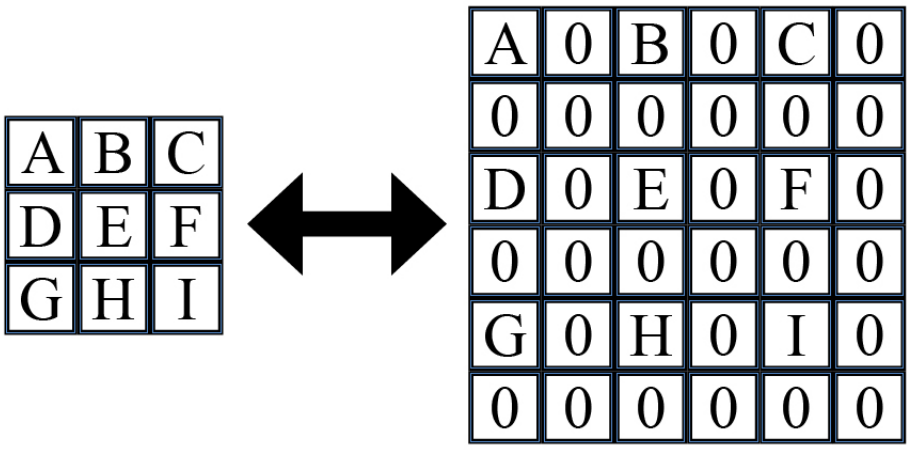
4. The equivalent low resolution image is subtracted from the output.

This process produces a difference image which is the error at each pixel within the image. A sign function can produce either a -1 or a 1. When the sign function is passed over the difference image, any pixel which has a positive value will become 1 and any with a negative value will become -1. This process allows for each low resolution image to affect the iteration by the same amount.

The current position, seen in figure 3.1 is now contains a low resolution description of the errors in the image. For this to be used to correct the high resolution image it needs to be up-sampled.

$$HH^T \sum F_k^T D^T sign(DF_k X_n - Y_k)$$

This low resolution image must now be up-sampled and shifted so that it is aligned with the high resolution image. After the image is at the right position and size it can be added to a sum matrix, along with every other low resolution image. This sum matrix



[h]

Figure 3.2: Representation of up-sample and down-sample process

is then blurred, as explained in section 3.2.4. The current image now contains the errors from all low resolution images at the right size and position to affect the high resolution image.

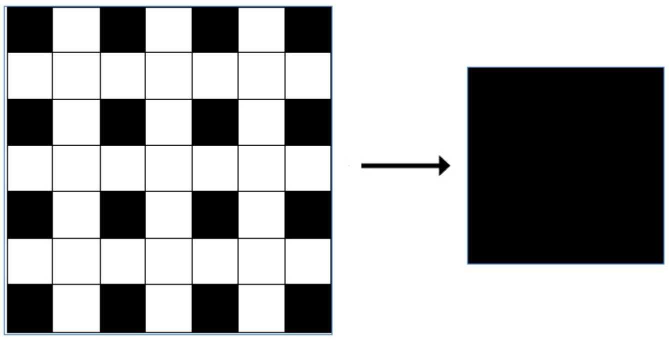
$$X_{n+t} = X_n - \beta[HH^T \sum F_k^T D^T sign(DF_k X_n - Y_k)]$$

The last part of the process contains a subtraction of the sum of errors from the high resolution estimation. β shows the amount that each iteration affects the high resolution image and, depending on its value, can accelerate or decelerate the process, but can cause problems if too high or too low as seen in section 4.3.2.

3.2.2 The Decimation Matrix

For a comparison to be made of pixels in the high and the low resolution images, an up-sample or down-sample filter needs to be used. The decimation matrix produces both up-sampled and down-sampled images.

The process seen in figure 3.2 is referred to as a shift and add method in [Farsiu et al., 2006] and produces the required low or high resolution representation by removing or zero filling the positions needed.



[H]

Figure 3.3: Example of Down-Sample

3.2.2.1 The Down-sample Process

This process requires removal of intermediate pixels to produce a lower resolution image.

As seen in figure 3.3, the first pixel is transferred to the new image, and the second is skipped. Similarly, the second row is skipped. This process continues until the low resolution image is constructed.

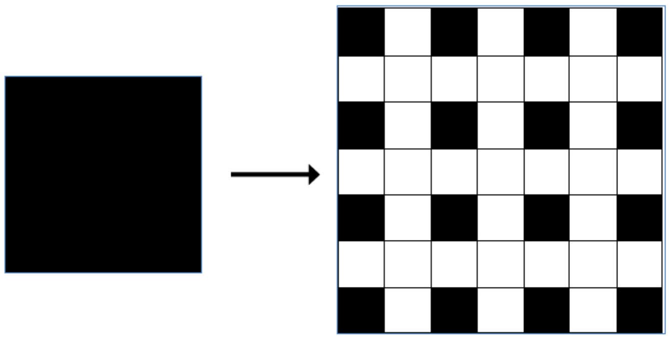
3.2.2.2 The Up-sample Process

Using a reverse of the down-sample process produces gaps within the high resolution image where the data was removed. Some methods of up-sampling will fill these with data obtained from adjoining pixels, but for this work, the gaps are zero filled.

A simple process of interpolation would up-sample using a blurring filter to insert missing data. For the Super Resolution process, the area is zero filled and the blurring, seen in section 3.2.4, is processed in the blur section. Figure 3.4 shows the up-sample process where the first pixel is taken and the adjoining ones are zero filled.

3.2.3 The Warp Matrix

Along with the downsample and blur processes, each low resolution image is shifted in relation to the high resolution estimation and so the warp matrix is introduced to shift the information to the correct position for processing. In this implementation the difference information is known as the test data is limited to translation by a single pixel in one



[H]

Figure 3.4: Example of Up-Sample

0	0	0
1	0	0
0	0	0

Figure 3.5: Warp Matrix

direction. This allows for a simple warp of each low resolution image. For real world applications, however, the difference information must be obtained from a motion estimation system which will allow for a more varied movement between frames. The simplest form of motion between frames is a unidirectional translation where the frame is shifted a single pixel in one direction. This can however be expanded to shift in multiple directions without too much change to the matrix. This filter is implemented by convolving the warp matrix with the image.

The direction of movement is indicated by a 1.0 on the opposite side of the matrix, seen in figure 3.5. Therefore, for the example, the image will be shifted 1 pixel to the right which is the movement chosen for the demonstration in this report. Due to the nature of convolution the edges need to be treated differently as the information must be obtained

[h]

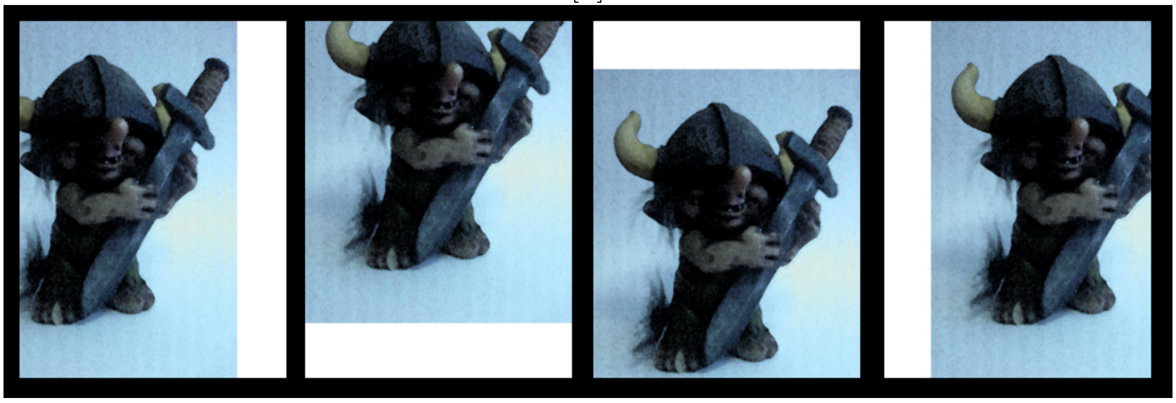


Figure 3.6: Examples of image shifted in each direction

either from padding, or from wrapping around to the other side of the image. This is explained in section 4.2.2. For the warp process, the unknown row for the shift to the left is merely padded with zeros, as the extra information will not be used in the rest of the calculations.

The images can be shifted in any of 8 directions so that the information can be matched to the correct pixels. Incorrect matches will result in poor results, as seen in figure5.10.

3.2.4 The Blur Matrix

After an image has been up-sampled, the intermediate data does not contain any relevant information. By passing a blur filter over the image, the information contained in the known pixels is spread to its adjoining pixels at a level defined by the type of blur filter used. Using a convolution of a blur matrix produces the required result efficiently and allows for different blur filters to be used.

The results of filtering using the simplest form of blur filter are shown in figure 3.7. This filter uses equal amounts of information from all pixels within the boundaries of the filter matrix. This can be expanded to any size matrix, but for this implementation is limited to a 3x3 or 5x5 matrix. Another form of blur filter uses the Gaussian form which places more emphasis on the centre and closest pixels than those further away. This produces better results when the change between consecutive pixels is larger.

Figure 3.8 shows the result of filtering using a simple 5x5 Gaussian matrix. The centre pixel has a higher weighting than the surrounding ones.

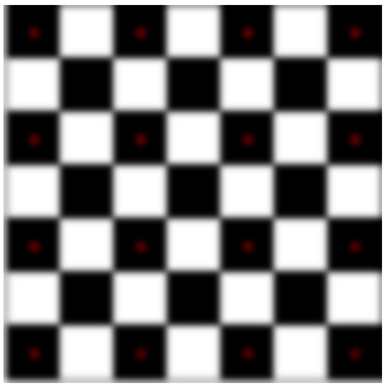
[h] 

Figure 3.7: Image after simple blurred filter is applied

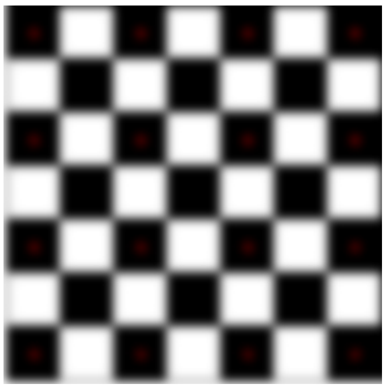
[h] 

Figure 3.8: Image after Gaussian blur is applied

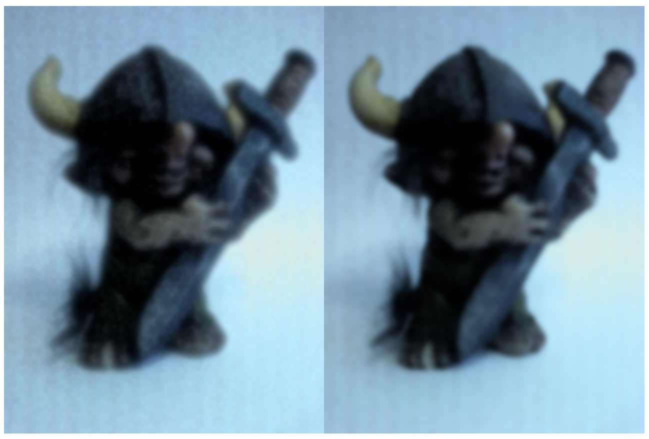
[h] 

Figure 3.9: Comparison of Simple and Gaussian Filters Respectively

A comparison of the blur filters, seen in figure 3.9, shows the effect of the weighting in the Gaussian matrix where the edges are better defined.. This produces some positive results when applied in the Super Resolution process.

3.2.4.1 Iteration

The basis of this Super Resolution filter is the iteration as it produces a more optimal image with each pass. The subtraction of errors will bring the estimation successively closer to the correct image and the correct outcome depends considerably on the calculation of the error rate.

3.3 Summary

This chapter has outlined the processes involved in the implementation of the Super Resolution algorithm by [Farsiu et al., 2003]. These steps must be implented in the plug-in to produce the required high resolution output. The Gaussian blur filter is chosen as the more accurate method for the process.

Chapter 4

Implementation

This chapter deals with the development of the plug-in, and outlines the problems associated with the interface between parts of the application. An outline of the basic components is introduced to illustrate the interaction between the parts. This is expanded in the discussion of the methods and classes involved as part of the plug-in. The GIMP is an opensource image manipulation package which provides an interface for the development of user defined plug-ins. The functionality of the GIMP is expanded for the implementation of this plug-in using an opensource matrix library seen in section 4.2.

4.1 Development of a Gimp plug-in

The GIMP development libraries provide a generic interface for easy development of plug-ins. Furthermore, a plug-in template is provided to allow for rapid development. This gives areas for development of the GIMP interface as well as the filter itself and even allows for the implementation of a graphical interface if required.

4.1.1 File Loading

The GIMP platform was chosen as the best means for the plug-in implementation due to its easy to use API interface and the ability to use both built in and exterior plug-ins. Although a limited number of methods are provided for image loading and storing within the matrix library, the interface presented by the GIMP API offers a far more general solution with regard to file types as the matrix library used only offers the methods for loading a few file types where the GIMP allows for a far greater number. Furthermore, the use of channels in the GIMP drawable class provides further abstraction of the process and allows for easy transference of the data into the matrix objects.

4.1.2 Interface

Initial designs focused on the backend of the Super Resolution implementation, but a graphical interface makes loading and ordering of input files simpler. The interface requires methods for adding and removing files and an area for viewing the files chosen. The CList class provides an interface for storing and accessing the names of files, and allows for inserting and deleting specific files.

Requirements for interface:

- Load and Remove file buttons
- List containing the names of files to be used

Possible Extensions of interface:

- Area to preview the selected image
- Selection for maximum number of iterations
- Selection of acceptable error rate
- File/Folder selection window instead of load/remove buttons

Figure 4.1 shows a screenshot of the plug-in interface. The main focus was to present a simple and efficient design with the required capabilities. Appendix C explains the method for using the plug-in with reference to the interface provided.

4.2 The Matrix Library

Due to the nature of the filters used in the Super Resolution algorithm, information is lost if the results are stored in simple integer arrays as is standard in the GIMP. These limitations are overcome by the inclusion of a matrix library which allows for the information to be stored in float or double arrays, offering more precision.

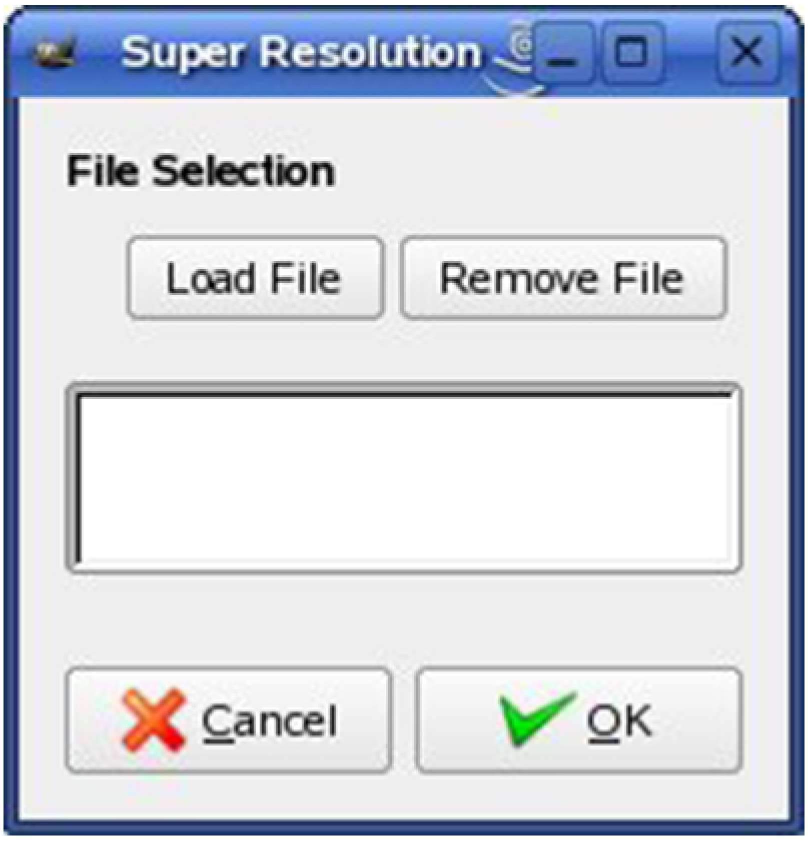
[h] 

Figure 4.1: Interface of Super Resolution Plugin

Algorithm 1 Code for accessing image data

```
    for(i=0;i<drawable->height;i++){
        for(j=0;j<drawable->width;j++){
            for(k=0;k<channels;k++){
                Mat2D_getDataFloat(mat)[i][channels * j + k] =
            row[channels * j + k];
            }
        }
    }
```

4.2.1 Storing Image data

Data loaded from images into the GIMP environment is stored in array structures known as drawable objects which have the ability to store data from multiple channels. The data from each channel is stored to the right of each prior channel and can be accessed using a simple nested iteration.

As seen in algorithm 1, the channels are dealt with through the drawable object and are stored in the matrices defined beforehand. In the matrices, each channel is stored to the right of the previous one and represents that of the drawable to make the data easily interchangeable. When files are loaded into the filter they are created as drawables and then transferred to matrix objects where the processing is done. Due to the limited integer type of the drawable objects, and the nature of the blur creating floating point values, the matrices are introduced to increase accuracy. Furthermore the interface provided by the matrix library simplifies functions such as convolution.

4.2.2 Convolution using channels

Convolution filters are problematic at image edges due to the fact that the edge pixels do not have sufficient surrounding data to perform the convolution. A few methods have been suggested to overcome this problem such as zero filling, edge replication and wrap around.

4.2.2.1 Zero Filling

The convolution process requires data around the pixel being processed. For pixels on the edge of an image, there is insufficient. In the case of zero filling, the data at the edges is padded with zero data, and in the case of an image, a black colour value. This method

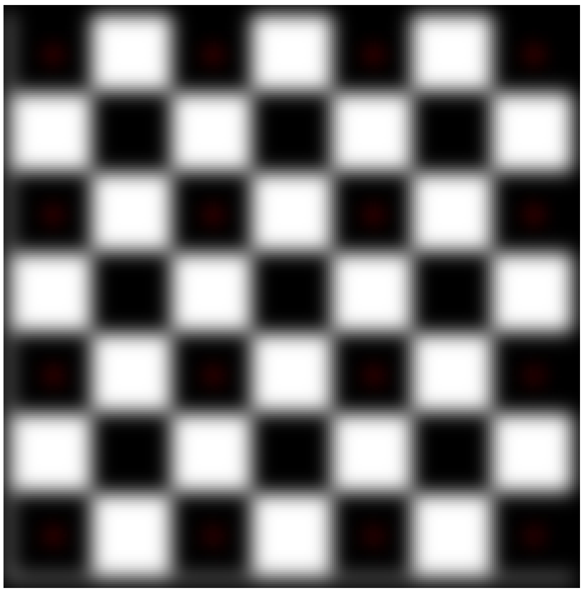
[h] 

Figure 4.2: Blur using zero filling

requires the least calculation, but has errors associated with it as the edge pixels are then dependant on that data.

The black data filled to surround the edge pixels has an influence on these pixels, as seen by the blur used in figure 4.2. The edge data is then falsely represented and reduces the clarity. In some instances this minor difference does not prove to be problematic, but in the case where the required data is at the edge of the image this method does not produce good results.

4.2.2.2 Edge Replication

Due to the problems experienced with zero filling, the idea of edge replication is introduced. Instead of the pixels surrounding the edges being padded with zeros, each pixel is replicated outwards to the same padding level. This allows for the edge data to remain correct, even in the case of a blur filter.

Figure 4.3 shows the result of using this method for convolution. The disadvantage of this is the need for calculation of the padding pixels from those existent on the edge of the image. Furthermore, corner pixels still remain inconsistent due to the lack of data for replication.

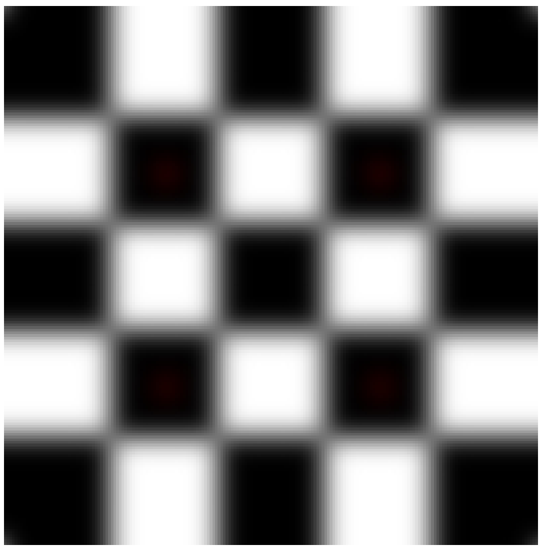
[h] 

Figure 4.3: Blur using edge replication

4.2.2.3 Wrap Around

The process of wrap around removes the need for extra information to be added to the image, and relies purely on the data on the opposing side of the image. This, like the edge replication method requires further computation, but does not introduce data that was previously not part of the image.

4.2.2.4 Chosen Method

The edge replication method provides better results, as seen in 4.2.2.1 and 4.2.2.2 where edge data is blurred with the zero filling. As the focus of this plug-in is restoration and clarity enhancement, it is better to keep the data and not introduce extra, the edge replication process is chosen.

4.2.3 Memory management

Memory management and garbage collection are dealt with in the matrix library used in the implementation. The interface provided for matrix creation and deletion deal with the allocation and freeing of memory used for the matrices. This does pose problems due to inconsistencies in The GIMP, and could be improved by manual allocation. Although inefficiencies do exist, they prove to be insignificant and do not affect the running of the application harshly. Memory considerations must be made due to the large amount of data used, caused mainly by the number of iterations which must be made to accomplish accurate enhancement. Furthermore, due to the nature of image data, space must be provided for data stored in each pixel. This causes a requirement of sufficient memory due to the introduction of channel data. Garbage collection within the plug-in is handled through interfaces provided by the GIMP.

4.3 Iterative Back Projection

The process of iteration improves the quality of the image with each iteration. This section outlines the considerations made with regard to the variables affecting efficiency and accuracy of the process.

4.3.1 Error comparison

To obtain the clearest high resolution image the error between the down-sampled high resolution estimation and each low resolution frame must be minimized. In the process of error calculation there are a few points during the iteration at which the data can be obtained. Each presents different results when used in the implementation. These changes do not affect the structure of the algorithm, but do affect the performance. The simplest point at which to calculate the error is at the calculation of a single low resolution difference. This allows for the minimization of this value without the requirement of calculation at each level. The disadvantage of this is that it does not take all error data into account and although it may offer some improvement in efficiency, it may not calculate the most optimal output. This can be expanded to the summed error of all low resolution differences, which does take all images into account, but causes a calculation to be made at each point in the summation. A further improvement can be seen if the calculation is made after the up-sample and blur process is completed, but again this has the disadvantage of decreased efficiency do to the increase in the size of the calculations which need to be made. The error calculation must then be a tradeoff between the efficiency and the optimality of the result, and due to the length of time which the plug-in inherently takes to process an image, the extra overhead may be acceptable in the quest to obtain an optimal image.

As seen in algorithm 2, the three positions for the error calculation have a large influence on the calculation overhead due to the number of times that it must be calculated.

Algorithm 2 Position of Error comparison data collection

```
while ( max itterations have not been reached)
{
clear current error matrix
for each file
{
apply downsample and sign function
[1] calculate error rate
add to error from all files
}
[2] caclulate error rate
blur error matrix
[3] calculate error rate
subtract errors from image estimation
}
```

4.3.2 Rate of change

Due to the iterative nature of the algorithm, the influence of each iteration must be balanced with the need to reach a specific point. For the procedure to be successful, the amount of influence of each iteration must be such that the steps do not affect the current array negatively. If the influence is too large, then the optimum solution will be skipped numerous times, and may not be achieved at all. Furthermore, if the influence is small, then the number of iterations must be increased for the optimum solution to be reached. Various methods have been proposed for similar situations. One solution is to introduce a changing rate which is influenced by the amount that the iteration affected the previous output. This produces the advantage of a faster acquisition of the point close to the solution, and the slow rate of change for values closer to the optimum. The problems associated with this method are based on the way in which the error is defined. The chosen method, seen in section 4.3.1, observes the global difference between each succession, and so does not accurately map the small changes needed for the point close to the solution. Experimentally optimal values prove to be more efficient when used on a variety of test data, seen in chapter 5.

4.4 Summary

The implementation considerations have been outlined in this chapter. The method chosen for convolution of edges is edge replication.

Chapter 5

Results

This chapter illustrates the outcome of the application of the filter to various test sets. A comparison is made of grey scale and colour inputs, followed by an analysis of the optimal settings and inputs. Although this implementation allows for both colour and greyscale images to be processed, there is a vast performance difference due to the increased processing time required for the extra channels. The outcome from both greyscale and colour images does not present the most promising result, but this is due to the lack of motion estimationin this implementation. These tests outline the results obtained and their relevance to the experiment. All experiments are conducted in on a computer using a Pentium 4 processor with 1Gigabyte of memory. The operating system used for testing was SUSE 10.1 with the latest stable GIMP release.

5.1 Super Resolution of grey scale images

Due to the fact that the intermediate pixels are interpolated iteratively, when the functions used to blur the input high resolution estimation, seen in section 3.2.4, have a small radius, then these pixels do not contain the expected values. For the testing of greyscale images, the images were taken using a Samsung D600 cellular phone at a resolution of 240x180. The image contained part of a peak cap.

Figures 5.1 and 5.2 show the effect of the blur being run too few times per iteration with intermediate pixels being lighter than those surrounding.

Figures 5.4 to 5.6 show the outcome of a greyscaled image processed to 200, 500 and 1000 iterations with a beta value of 2 for the first two and 0.5 for the third. As can be seen from the first and second image, an incorrect number of iterations will decrease the clarity

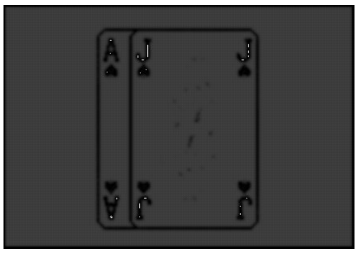


Figure 5.1: 50 iterations with single blur per iteration

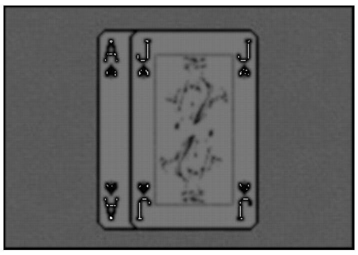


Figure 5.2: 100 iterations with single blur filter per iteration

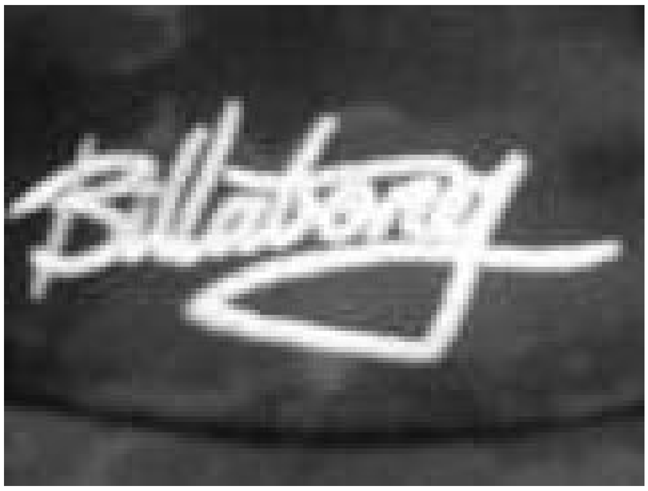


Figure 5.3: Image scalled using interpolation

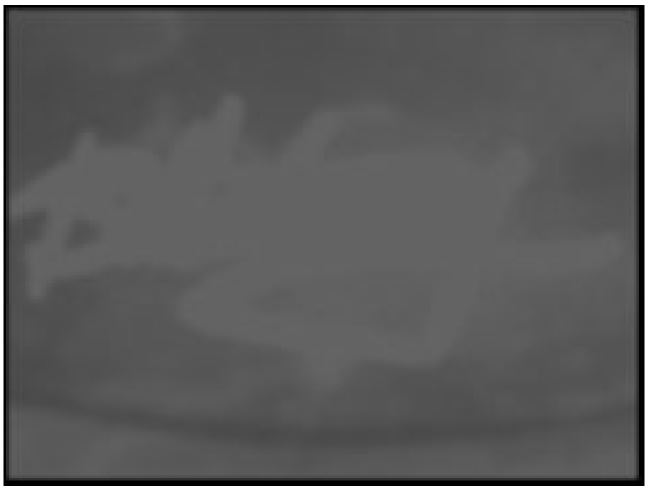


Figure 5.4: Test using 6 images with a beta of 2 for 200 iterations

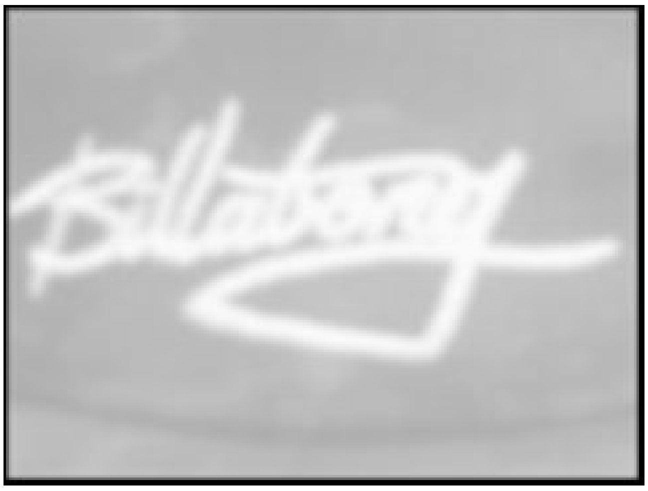


Figure 5.5: Test using 6 images with a beta of 2 for 500 iterations

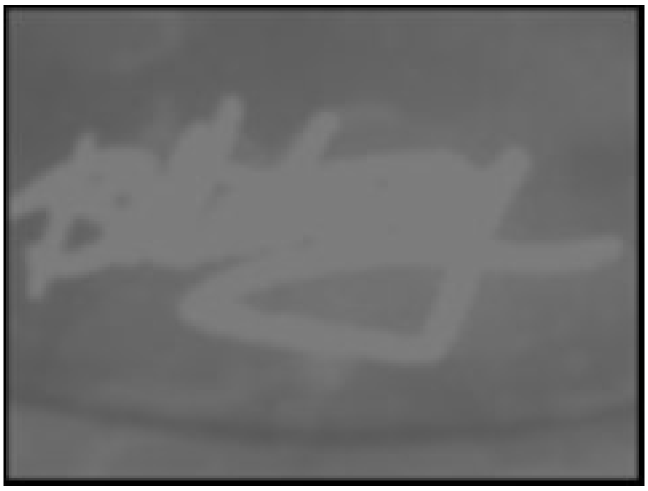


Figure 5.6: Test using 6 images with a beta of 0.5 for 1000 iterations

of the image, but results do show promise when compared with the image scaled through a standard filter in 5.3. This error could also be caused by an incorrect beta value as seen in section 4.3.2. The output produced is a blurred version of the collection of images as expected, but results are not as clear as expected which could be due to the lack of motion compesation or due to the incorrect warp matrix being constructed.

5.2 Super Resolution of colour images

As with the testing performed on the greyscale images, the pictures used were taken with a Samsung D600 cellular phone at a resolution of 240x180.

Figure 5.7 shows the output from the image being scaled using simple interpolation. As can be seen from the result, artefacts are created around the objects within the image due to the additive method of interpolation.

Figures 5.8 to 5.9 show the results of various values for iteration, with a fixed beta value of 2. As can be seen from these results, having a number of iterations which is too small will adversely affect the output, but with a value that is too high, the output will have a higher brightness than required. The colour examples reiterate what was said in section 5.2 where an incorrect value of beta could cause the incorrect result.

If input images are shifted, the ouput will obtain values from incorrect data and will not accurately represent the inputs, seen in figure 5.10. With the introduction of a motion

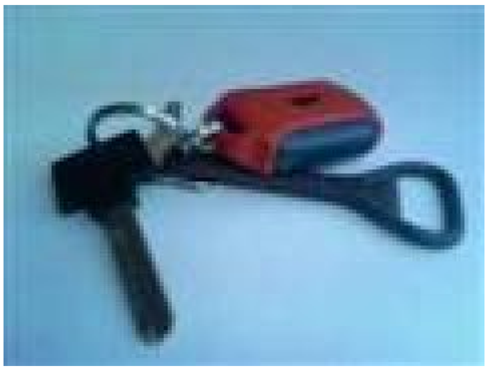


Figure 5.7: Image scaled with standard filter

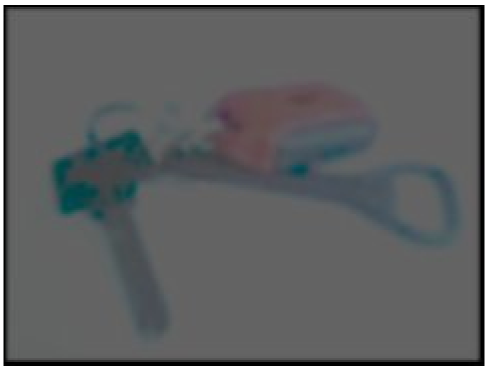


Figure 5.8: Test image using 6 images a beta of 2 and 200 iterations

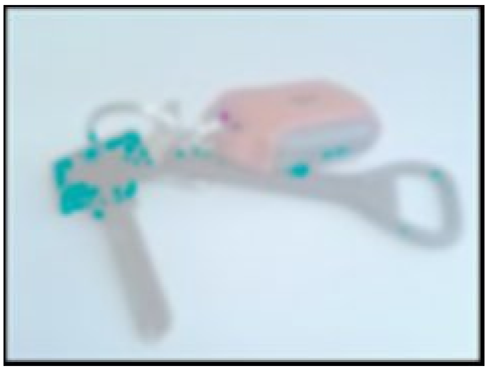


Figure 5.9: Test image using 6 images a beta of 2 and 500 iterations

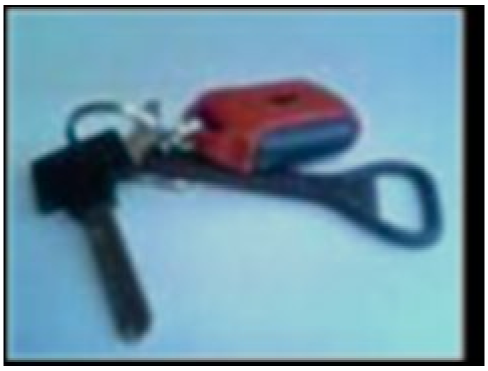


Figure 5.10: Test image with shifted inputs

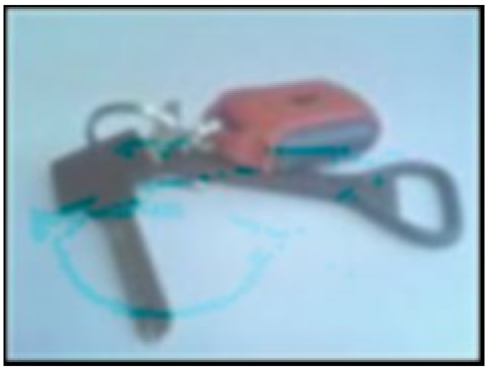


Figure 5.11: Test image for completely different input images

detection system, this problem will not occur as each image is matched to the correct position in the high resolution data. Furthermore, with more testing, figure 5.11 shows the effect of combining multiple low resolution images as the outline of the second image is seen in the example.

5.3 Optimizations

The rate of influence and the number of iterations has a adverse effect on the ouput when incorrect values are chosen. Through experimentation, the results seen in figure 5.13 show the optimum testing values and present an image which is more accurate than previous examples.

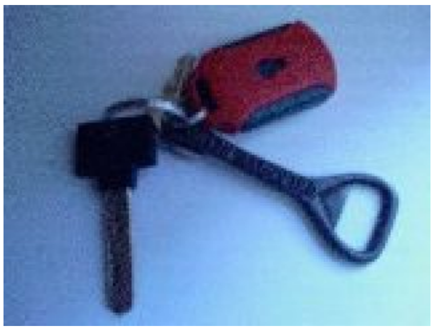


Figure 5.12: Image scaled through standard filter

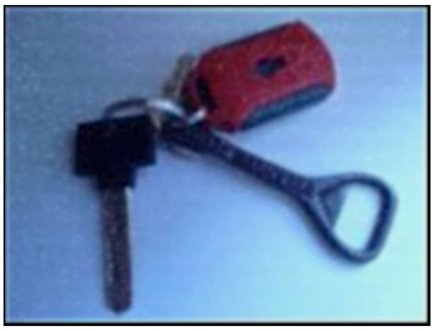


Figure 5.13: Test using 2 images a beta of 2 with 500 iterations

Results for this section were obtained using 2 input images which could account for the increased accuracy. With a greater number of images, the incorrect values obtained from the lack of correct motion estimation could account for the increased error.

5.4 Summary

The results obtained in this section are not as clear as was expected, but this is probably due to the lack of a motion detection interface. Futhermore, the difference between using greyscale and colour images as input is mostly seen in the time taken for processing. For this plug-in to work correctly a motion estimation method would have to be added to calculate the difference between the position of each low resolution image and the high resolution output.

Chapter 6

Conclusion

This chapter outlines the conclusions drawn from the work done during this project. The possible future work and application follows with a brief outline of their requirements.

6.1 Outcomes

6.1.1 Current Application

Although the hardware requirements for an efficient Super Resolution filtering system are higher than those of previous interpolative methods, the Super Resolution filtering seen in works by [Baker and Kanade, 2000; Boccacci and Bertero, 2003; Candocia and Principe, 1999] prove to be more successful in the restoration of the required images. This requirement of faster computer hardware may force the filter system to remain as a tool for those with faster hardware, the work on both the algorithm and entry level computer systems show promise for its future. The time for a single image calculation is far greater than that of previous filters, as seen in section 5, but for applications such as security or identification, this performance of between two and five minutes for completion is acceptable. The Super Resolution algorithm has reached a point where it is robust enough to use on real world objects provided that an efficient motion estimation system is used in conjunction with it which presents an opportunity for it to be integrated into the image restoration field. Much work is still required before the filters will be viable for a released commercial application, but with the rate of development of methods and hardware, these filters will be accessible in the near future.

6.1.2 Limitations imposed

Although the field of motion estimation is expanding rapidly, this area still poses an increasing problem for Super Resolution application as correct identification of successive frames is essential for an effective output. Furthermore, motion estimation models focused only in the global estimation framework do not allow for local motion and hence limit the application in real world implementation.

The greatest shortcoming of the current implementation is that of motion calculation and correct alignment of the warp matrix allowing for real world application. The limitations brought about due to the lack of the motion estimation platform prevent the plug-in from being used in simple real world applications and motion between images must be calculated beforehand. The impact of this design decision is evident in chapter 5, but it does offer performance improvement due to the fact that the processing for image difference is not required for each usage.

6.2 Future Work

With regard to the expansion of this plug-in, the integration of an accurate motion detection system would introduce the ability to process captured real world data including that obtained from video feeds. This integration would take the form of the construction of a warp matrix from the data obtained and would fit neatly into the construction of the algorithm, providing the missing functionality. Further development must be done on the process of image loading and output, where currently an image of correct size must be loaded into the drawable region before the Super Resolution processing can be done. This limitation is imposed by the API, although with further research, methods could be found to overcome this and allow for the plug-in to be converted to an extension of the GIMP system rather than requiring an image to be loaded.

Bibliography

S. Baker and T. Kanade. Limits on Super-Resolution and How to Break Them. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 372-379, Los Alamitos, June 2000. IEEE.

J. Bergen, P. Anandan, and K. Hanna. Hierachical model-based motion estimation. In *Proceedings of European Conference on Computer Vision*, pages 237-252, 1992.

C. Bhattacharya. Performance Evaluation of A Superresolution Algorithm for Image Restoration - . In *Proceedings of Indian conference on Computer Vision, Graphics and Image Processing*, September 2000.

P. Boccacci and M. Bertero. Super-resolution in computational imaging. *Micron 34*, 34(6):265-273, October 2003.

F. M. Candocia and J. C. Principe. Super-Resolution of Images Based on Local Correlations. *IEEE Transactions on neural networks*, 10(2):372, March 1999.

M. Elad and Y. Hel Or. A fast super-resolution reconstruction algorithm for pure translational motion and common space-invariant blur'. *IEEE Trans. Image Processing*, 10(8): 1187-1193, August 2001.

Michael Elad and Arie Feuer. Restoration of a single superresolution image from several blurred, noisy, and undersampled measured images. *IEEE Transactions on Image Processing*, 6(12):1646-1658, February 1997.

S. Farsiu, M. Elad, and P. Milanfar. Constrained, Globally Optimal, Multi-Frame Motion Estimation. In *Proceedings of IEEE Workshop on Statistical Signal Processing*, pages 1396 - 1401, Bordeaux, France, July 2005.

Sina Farsiu, M. Dirk Robinson, Michael Elad, and Peyman Milanfar. Fast and robust super-resolution. In *Proceedings of International Conference on image Processing*, pages 291-294, December 2003.

Sina Farsiu, Michael Elad, and Peyman Milanfar. Multiframe demosaicing and super-resolution of color images. *IEEE Transactions on Image Processing*, 15(1):141-159, August 2006.

Venu Madhav Govindu. Lie-Algebraic Averaging for Globally Consistent Motion Estimation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 684-691, October 2004. URL http://csdl.computer.org/comp/proceedings/cvpr/2004/2158/01/215810684abs.htm,.

Bahadir K. Gunturk, Yucel Altunbasak, and Russell M. Mersereau. Super-resolution reconstruction of compressed video using transform-domain statistics. *IEEE Transactions on Image Processing*, 13(1):33-43, December 2004.

M. Irani and S. Peleg. Improving Resolution by Image Registration. *Graphical Models and Image Processing*, 53: 231-239, 1991.

Zhongding Jiang, Tien-Tsin Wong, and Hujun Bao. Practical Super-Resolution from Dynamic Video Sequences. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 549-554'. IEEE Computer Society, August 2003. ISBN 0-7695-1900-8'.

Manjunath V. Joshi, Subhasis Chaudhuri, and Rajkiran Panuganti. Super-resolution imaging: use of zoom as a cue. *Image Vision Computing*, 22(14):1185-1196, April 2004.

Chang-Hsing Lee and Ling-Hwei Chen. A fast motion estimation algorithm based on the block sum pyramid. *IEEE Transactions on Image Processing*, 6(11):1587-1591, February 1997.

A. Lorette, H. Shekarforoush, and J. Zerubia. Super-Resolution with Adaptive Regularization - . In *Proceedings of IEEE International Conference on Image Processing*, pages 169-172, 1997.

B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of DARPA Image Understanding Workshop*, pages 121-130, 1981.

Peyman Milanfar. Two-dimensional matched filtering for motion estimation. *IEEE Transactions on Image Processing*, 8(3): 438-444, January 1999.

U. Mudenagudi, R. Singla, P. Kalra, and S. Banerjee. Super Resolution Using Graph-Cut. In *Proceedings of Asian Conference on Computer Vision*, pages 385-394, 2006.

N. Nguyan and G. Golub. Preconditioners for regularized image superresolution. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 6, pages 3249-3252, Phoenix, AZ, 1999.

A. Patti and Y. Altunbasak. Artifact Reduction for POCS-based Super Resolution with Edge Adaptive Regularization and Higher-Order Interpolants'. In *Proceedings of IEEE International Conference on Image Processing*, pages 217-221, 1998.

Lyndsey C. Pickup, Stephen J. Roberts, and Andrew Zisserman. A Sampled Texture Prior for Image Super-Resolution. In Sebastian Thrun, Lawrence K. Saul, and Bernhard Schölkopf, editors, *Proceedings of NIPS*. MIT Press, October 2003. ISBN 0-262-20152-6'.

Filip Sroubek and Jan Flusser. Multichannel blind iterative image restoration. *IEEE Transactions on Image Processing*, 12 (9):1094-1106, December 2003.

Carl Staelin, Darryl Greig, Mani Fischer, and Ron Maurer. Neural Network Image Scaling Using Spatial Errors. November 2003.

H. Szu and I. Kopriva. Artifcial neural networks for noisy image super-resolution. In 198, editor, *Proceedings of Optics Communications*, volume 198, pages 71-81. Elsevier Science, October 2001.

R. Tsai and T. Huang. Multiframe image restoration and registration. *Advances in Computer Vision and Image Processsing*, pages 317-339, 1984.

L. Vasa, I. Hanak, and V. Skala. Improved super-resolution method and its acceleration. In *Proceedings of European Signal Processing Conference*, page 121, 2005.

B. Zitova and J. Flusser. Image registration methods: a survey. *Image and Vision Computing*, 21(11):977-1000, October 2003.

Assaf Zomet, Alex Rav-Acha, and Shmuel Peleg. Robust Super-Resolution. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 645-650. IEEE Computer Society, August 2001. ISBN 0-7695-1272-0’.

Appendix A

Installation of The GIMP

The current implementation of the Super Resolution plug-in requires a linux environment with the gimp development libraries installed. The implementation has been tested on various distributions to ensure its ability to be transferred to client machines. The cross-platform nature of the gimp allows for simple transference with a dependence on the correct libraries and binaries being installed. This section will outline the installation of the gimp onto a number of distributions with emphasis on generality so as to expand to unlisted distributions.

Due to the different installation methods used in various linux distributions the next few sections will outline the install methods for some of the popular distributions.

A.1 Ubuntu

This distribution uses the aptitude installation system developed for use in the debian environment and so will be applicable in that as well. For successful installation and use of the plug-in there are two components that need to be installed. The first contains the binaries of The GIMP application and the second contains the development libraries and binaries used for compilation and installation. Furthermore the gcc and make applications must be installed for compilation of the plug-in.

```
apt-cache search gimp
```

This will output the available packages which contain the word gimp. The list will contain the available version of the “gimp” and “gimp-devel” binaries.

```
apt-cache install {name}
```

This command will install the applications where {name} is the name of the package to be installed. These binaries can also be installed using the graphic installation application synaptic package manager.

A.2 Suse

Suse, another commonly used linux distribution uses an installation system known as yast2. As with the ubuntu installation there are two packages which must be installed. The package manager has a search feature which makes finding the required applications simple. Once both have been selected and dependencies are calculated they can be installed.

Appendix B

Installation of the Plug-in

For the plug-in to be successfully installed, the development libraries of the gimp need to be installed, this is dealt with in appendix A and will not be reiterated. There are two main methods available for the installation, one corresponding to the installation as a single user and one to a platform-wide installation. Both methods will be outlined, but the correct one must be chosen at the discretion of the user and the extent of their access rights. A tool is provided with the installed gimp-development binaries called gimptool, or gimptool-2.0 for the second version of the gimp application.

> GIMPTOOL-2.0
> Section: GIMP Manual Pages (1)
> Updated: March 23 2004
> Index
> NAME
> gimptool-2.0 - script to perform various GIMPy functions
> DESCRIPTION
> gimptool-2.0 is a tool that can, among other things, build plug-ins or scripts and install them if they are distributed in one source file.
> gimptool-2.0 can also be used by programs that need to know what libraries and include-paths GIMP was compiled with. This is especially
> useful in Makefiles.

The gimptool application will compile and install the plug-in to the correct directory. As was said previously, two options exist for this installation. One installs the plug-in for global plug-in directory and one installs to a local plug-in directory located in the user's home directory. The options used for each installation are similar except for a single command.

```
gimptool-2.0 –install-bin super
```

This shows the command used for single user installation.

```
gimptool-2.0 –install-admin-bin super
```

This shows the command used for multi user installation. After the plug-in is installed it will appear in the gimp plug-in library under Filter->Enhance->Super Resolution. The use of the plug-in will be outlined in the next appendix.

Appendix C

Using the Plug-in

After successful installation of the plug-in to the GIMP environment, there are a few things to note before beginning use of the filter.
Intitial Requirements:

- An initial image must be loaded which is twice the size of the low resolution images. This can either be a new blank image, or the initial low resolution image scaled to the size required. This is so that the output drawable is sized correctly before the Super Resolution process begins.
- For this version of the Super Resolution filter requires that each image is shifted one pixel to the right of the previous one. This is implemented as one of the design considerations. Due to the lack of an efficient motion estimation component, the image difference is unknown and so must be defined.
- The image data contained in each image of the sequence must be the same with regard to channel information and resolution.

The procedure for using the filter is as follows:

Starting the interface:

1. Load The GIMP image manipulation software.
2. Open or create an image double the size of those to be filtered.
3. Select Filters->Enhance->Super Resolution from the menu.

4. The Super Resolution interface will appear.

Loading Files:

1. Click the "Load File" Button and this will bring up a file open dialog.
2. Multiple files can be selected, or simply double click each file in order.
3. When the files have been selected press the "Ok" button and the files will be added to the file selection list.
4. Files can then be removed from the list or positioned in the correct order.
5. If more files need to be added the "Load File" button can be pressed again.
6. When the list contains the images in the correct order then press the "Ok" button on the Super Resolution dialog.

Output:

1. The filter will then perform the required operations to produce the Super Resolution image.
2. This process does take a while and will output the results to the image frame on completion.
3. The resultant image will be the Super Resolution representation of the low resolution frames.

If problems occur during the Super Resolution process, close The GIMP and restart the process. If errors continue to occur then the image data used as the input probably contains errors and will not work in the filtering process.