

**Video Super-Resolution From  
Scene-Specific Images**

Sam Johnson

BSc Computer Science

2007/2008

The candidate confirms that the work submitted is their own and the appropriate credit has been given where reference has been made to the work of others.

I understand that failure to attribute material which is obtained from another source may be considered as plagiarism.

(Signature of student) \_\_\_\_\_

## Summary

The aim of this project is to design and implement a super-resolution algorithm to improve the visual quality of videos. Super-resolution is the process of improving the resolution of an image or video and inserting high-frequency detail which was not previously present. This detail will be taken from higher resolution images of the same scene. It is hoped that any object present in the video will also be present in the still images, where they will be captured in much higher detail. Even if an exact match for every object in the video is not present it is hoped that the algorithm will still be able to create plausible high-frequency detail to improve the input video.

In this report I will review previous work done in this area and describe how my super-resolution algorithm was implemented. I will also evaluate my algorithm and suggest possible future extensions.

## Acknowledgements

First of all I would like to thank my project supervisor Dr. Mark Everingham for fitting our weekly meetings into his busy schedule, and for all the sound knowledge and guidance he gave me throughout the year. I would also like to thank my assessor Dr. David Hogg for the valuable feedback he gave me on my mid project report, and the questions and guidance he gave me during the project demonstration at the end of the year.

I would also like to thank my family, for supporting me throughout my degree and being there for me when times were hard. My friends both in university and outside also deserve thanks, for not turning their backs on me when I disappeared into the School of Computing labs for weeks on end, and for giving me ongoing evaluation of how my output videos were looking.

Finally I would like to thank my girlfriend who has kept me motivated to work to my fullest potential throughout this project and also kept me fed and watered!

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Aim . . . . .	1
1.2	Objectives . . . . .	1
1.3	Minimum Requirements . . . . .	2
1.4	Relevance to Degree . . . . .	2
<b>2</b>	<b>Background Reading</b>	<b>4</b>
2.1	An Introduction to Super-resolution . . . . .	4
2.2	Methods of Super-resolution . . . . .	6
2.2.1	Example Based Super-resolution . . . . .	8
2.2.1.1	Training . . . . .	9
2.2.1.2	Super-resolving the Input Image . . . . .	10
2.2.1.3	Reducing Artifacts in Static Areas of Video . . . . .	12
2.2.1.4	Reducing Artifacts in Areas of Video Containing Motion . . . . .	13
2.3	Evaluating Super-resolution Results . . . . .	14
2.4	Methodologies . . . . .	16
2.5	Appropriate Technologies . . . . .	18
<b>3</b>	<b>Design and Implementation</b>	<b>20</b>
3.1	Super-resolution of a Single Image . . . . .	20
3.1.1	Overview . . . . .	20
3.1.2	Training . . . . .	21
3.1.2.1	Training Image Preparation . . . . .	21
3.1.2.2	Patch Extraction . . . . .	23
3.1.2.3	Patch Database Construction . . . . .	25
3.1.3	Search . . . . .	26

3.1.3.1	Patch Extraction . . . . .	26
3.1.3.2	Patch Lookup . . . . .	27
3.1.3.3	Output Construction . . . . .	27
3.2	Improving Search Efficiency . . . . .	28
3.2.1	Patch Database Storage Method . . . . .	28
3.2.2	Use of a kd-tree for efficient search . . . . .	28
3.3	Improving Spatial Consistency . . . . .	30
3.3.1	Previous Solutions . . . . .	31
3.3.2	Changes to the Algorithm . . . . .	35
3.4	Super-resolution of Video . . . . .	35
3.4.1	Overview . . . . .	35
3.4.2	Training . . . . .	36
3.4.3	Search . . . . .	37
<b>4</b>	<b>Results and Evaluation</b>	<b>38</b>
4.1	Evaluation of Perceived Visual Improvements . . . . .	38
4.1.1	Bridge Video . . . . .	38
4.1.2	Pit Stop Video . . . . .	42
4.1.3	Climb Video . . . . .	42
4.1.4	Ski Video . . . . .	43
4.2	Quantitative Evaluation . . . . .	44
4.3	Qualitative Evaluation . . . . .	47
<b>5</b>	<b>Conclusion</b>	<b>48</b>
5.1	Evaluation Criteria . . . . .	48
5.2	Objectives and Requirements . . . . .	48
5.3	Possible Future Developments . . . . .	49
	<b>Bibliography</b>	<b>51</b>

# 1. Introduction

## 1.1 Aim

A camera capable of recording VGA quality video ( $640 \times 480$  pixels) can now be found built into affordable mobile phones. The same camera device is often also capable of taking high resolution still images (e.g.  $2592 \times 1944$  pixels, or 5 megapixels). The system described will improve the resolution and quality of the captured videos by using higher quality and resolution still images captured from the same device. The high-resolution images will contain information which can be copied into the video to improve it. The process of enlarging an image (or a single video frame) to a higher resolution and inserting new high frequency, fine detail is known as super-resolution. Uses for a system such as this include:

- An application used on camera phone video to increase its size and quality. The videos captured on modern mobile phones lack fine, high resolution detail due to their small resolution and the compression used to decrease their storage requirements. This application would enable a video captured on a phone to be viewed at higher resolutions and on larger display devices in higher detail than previously possible.
- Improving the quality of videos available on internet sites such as YouTube and AOL Video. High resolution images of the same, or similar scenes as these videos are likely to be found on sites such as Flickr and Google Image search. The detail in these images could be used to remove compression artifacts from online videos, and improve their resolution and quality.

## 1.2 Objectives

The objectives of the project are to:

- Capture a number of videos and images on a Nokia N95 mobile phone. This phone contains a 5 megapixel camera which can be used to capture videos and high quality still images of the same scene. This high resolution detail will be inserted into the videos so that their spatial resolution can be improved. A good set of training images can be created

by taking some frames from a video containing little movement, and then down-sampling the video. This leaves a low quality video and images of higher quality and resolution.

- Implement an algorithm to improve the quality of low resolution video using a database trained from the high resolution images. These high resolution images will be copied and filtered so that they closely resemble the size and quality of the input video. These pairs of high quality and filtered images will then be used to create a database linking low frequency, blurred detail with its corresponding high frequency, fine detail. The high frequency detail for each frame of the input video can then be generated, producing a super-resolved video.

### **1.3 Minimum Requirements**

The minimum requirements are:

- To improve an image using the method outlined above given a similar image in a higher resolution. The results of the system will be simple to evaluate when the high resolution image is known.

The possible extensions are:

- Making the system work on multiple frames of a video, incorporating some temporal model into the selection of high resolution detail. This will use information from previous frames to help predict which detail to choose for the current frame. Doing this should reduce any noticeable inconsistencies in the processed video, such as areas flickering over successive frames.
- Reduce the computational time needed to process one frame of a video by optimisation of the code and storage methods. This will involve writing parts of the system in lower level code and reducing the computational complexity of the algorithms used.

### **1.4 Relevance to Degree**

Throughout my degree I have chosen to study the majority of Artificial Intelligence (AI) modules available. I have found image processing and computer vision AI modules particularly interesting and enjoyable. This project will extend the skills I have developed throughout my

degree in image analysis and machine learning by applying them to a project of a size and scale which I have never before attempted. It should also prepare me well for the experience of research-based post-graduate study which I wish to pursue later.

## 2. Background Reading

Super-resolution is an area of computer vision which has seen a large amount of research activity in recent years. The reasons for this, and the methods which have been developed will be explained in the following chapter.

### 2.1 An Introduction to Super-resolution

Due to the improved visual quality and level of detail present in high-resolution images it is desirable for photographs and videos to be captured in the highest resolution possible. The resolution of an image refers to the number of pixels present, a larger number of pixels meaning that finer, higher-frequency detail is visible. The technology used in digital camera devices to capture images is usually a charged-coupled device (CCD) or CMOS image sensor. These sensor technologies are now reaching the point where it becomes physically difficult to increase the number of pixels which can be captured [19]. There are two main ways to increase the number of pixels contained on an image sensor, either physically shrink the size of each pixel on the sensor allowing an increase in pixel density, or increase the size of sensor itself, allowing room for more pixels. Unfortunately both of these methods of increasing image sensor resolution are now approaching physical limits.

Reducing the size of each pixel reduces the amount of light available to capture. Once this light decreases to a certain level a physical phenomenon called shot noise can occur due to irregularities in capturing the low number of photons (small “packets” of light) available [22]. Alternatively the method of enlarging the image sensor itself leads to an increase of capacitance across it, which increases the time taken to transfer the captured image to memory [15]. In a device used to capture video the time taken to transfer each frame from the sensor to memory, including processing to decrease the storage requirements, severely limits the frame rate. To reduce the processing time and storage requirements further the resolution of captured video is also kept relatively small. This has led to most multiple function devices, such as camera phones, being able to capture video at low resolution whilst capturing photographs at much higher resolutions. Another factor which limits increasing the resolution of the image sensor itself is that of cost, both to the manufacturer and to the consumer who has now come to expect

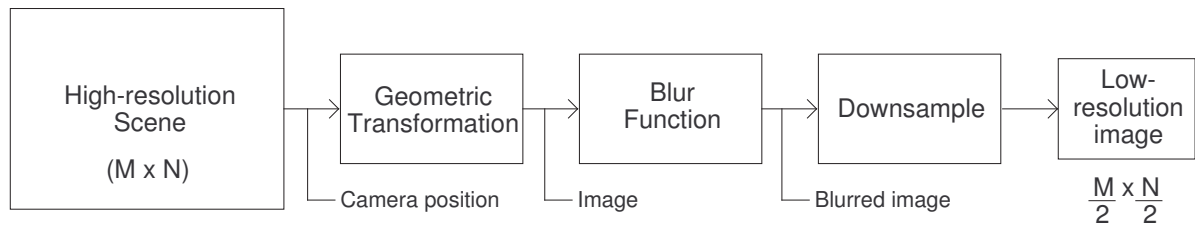


Figure 2.1: The process used to capture a low-resolution image from a real world scene. This process is what super-resolution attempts to reverse.

advances in technology to become affordable in a short space of time. Super-resolution offers a method of bypassing both the cost, and physical limitations of obtaining high-resolution, high frame rate video on relatively cheap image capturing devices.

As well as increasing the resolution of images or video super-resolution can also be used to increase the quality. With the widespread adoption of the internet, the rise of websites driven by user submitted media, and a surge in the number of people owning digital image capture devices, there has been a huge increase in the number of people recording amateur video and photographs. This increase in the abundance and availability of digital cameras is largely due to the presence of such devices built into most modern mobile phones. For transmitting this captured content over the internet as fast as possible the size of the media files needs to be reduced. Compression is commonly used on both video and images which whilst successfully reducing the file size, and hence transfer time, also reduces the visual quality. Online video, such as that found on YouTube, AOL Video and similar video sharing sites has usually been compressed quite aggressively and sub-sampled to a relatively small resolution. This combination of sub-sampling to reduce the resolution of the video, and heavy compression used to further decrease the file size results in video of very poor quality when compared to the original. The video compression methods used also commonly introduce distracting artifacts into the video especially when the video contains high levels of motion or fine detail. Images on the other hand tend to be compressed much less aggressively and can be found online at sites such as Flickr in much higher resolutions than online video, and with little or no artifacts or blurring present. Super-resolution can be used to increase the visual quality of online video, removing compression artifacts and increasing the resolution, using higher resolution images from online photo collections such as Flickr [16].

When a camera is pointed at a scene the light is transformed into a captured image. This

transformation process is shown in figure 2.1 and converts the high-resolution scene into a relatively lower resolution image. Super-resolution is a computer vision method which tries to reverse this process as much as possible by increasing the level of fine, high-frequency detail in an enlarged low-resolution image. As well as increasing the level of detail in the spatial domain, super-resolution can also refer to increasing the level of detail in the temporal domain, where it is used to increase the frame rate of video [24]. For the rest of this paper, whenever the term super-resolution is used I will be referring to super-resolution in the spatial domain. Super-resolution is a difficult problem to solve as it is under-constrained if only a single low-resolution input image is available. From the reduction in size in figure 2.1 we are left with  $\frac{M \times N}{4}$  measurements with which to calculate the  $M \times N$  measurements of the original high-resolution image. Therefore to solve this problem requires either multiple low-resolution images of the same scene, or prior information to bias the solution towards a particular high-resolution image.

## 2.2 Methods of Super-resolution

There has been a large amount of research done into super-resolution and from this a number of different methods of tackling the problem have emerged:

- Multiple Source Images

Using multiple low-resolution images for super-resolution is only effective if there are sub-pixel differences between the images, for examples see work by Capel *et al.* [5] and Park *et al.* [19]. The multiple images required can be gathered from a selection of frames of a video sequence [1], from multiple captures from the same camera, or even from different cameras at different locations but still capturing the same scene. If the collection of low-resolution images are shifted relative to each other by integer quantities of pixels then no new information can be gathered. However, if the images are shifted by sub-pixel amounts then different information about the same scene exists in each image. This different information can be combined to reverse the process shown in figure 2.1 and output a single higher-resolution, super-resolved image.

- Implicit Prior

If only a single low-resolution input image is available then to enlarge it requires prior information about how to interpolate between known samples, or pixels. The most commonly used methods of doing this use an implicit prior. Such methods include bilinear and

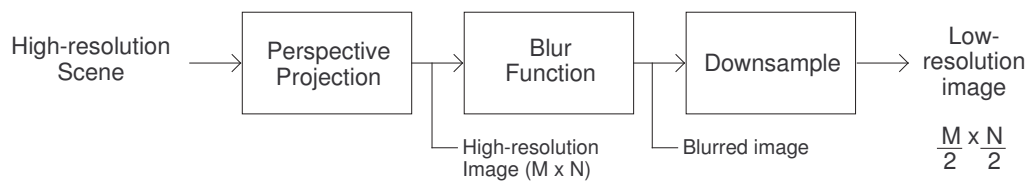


Figure 2.2: The process by which example based super-resolution algorithms work. A high-resolution training image is copied and filtered, and then the relationship between the medium-frequency detail of the filtered image, and the high-frequency detail of the original image is learnt. A low-resolution input image is then analysed and the most plausible corresponding high-frequency detail is inserted.

bicubic interpolation [14], where missing pixel values are assumed to lie along a functional line between known pixel values. Methods of increasing an image size using implicit priors are rarely used for super-resolution as they fail to insert fine, high-frequency detail into the enlarged image.

- Parametric Prior

Methods of interpolation which have performed better when applied to super-resolution use parametric priors. Such methods include Huber [20] and Gaussian interpolator functions [23].

- Non-Parametric or Learnt Prior

Implicit and parametric priors both fail to insert plausible high-frequency detail into an enlarged image. Although it may seem like the generation of previously missing high-frequency detail is impossible, images of natural scenes are actually much less random than one may think. Research into the early stages of mammalian visual systems has shown that there are a lot of structural regularities in real world scenes which would not be present in images formed from truly random noise [25, 9]. These regularities can be taken advantage of through a learning-based approach to super-resolution, where the relationship between medium- and high-frequency detail is learnt as a non-parametric prior. Once this relationship has been learnt it should be possible to analyse lower resolution images containing medium-frequency detail and insert the corresponding high-frequency detail which is missing.

### 2.2.1 Example Based Super-resolution

The aim of this project is to improve the quality of video captured on a device which can also capture high-resolution still images. The higher resolution still images can be used to learn the relationship between medium- and high-frequency detail. This relationship can then be used to improve low-resolution input images. This method of increasing the resolution of images using a learnt relationship is known as example-based super-resolution. Using scene-specific training images is likely to give the best results, as it is likely that objects contained in the input image or video are also present in the high-resolution training images. As well as using higher resolution images of the same scene as a prior [17], algorithms have been implemented which use a collection of unrelated high-resolution images [10, 16, 4, 11]. Although using unrelated training images generally gives poorer results, there still exists enough information to learn the general relationship between medium- and high-frequency detail.

Using higher resolution images containing a relatively large variety of objects is not the only method of learning a super-resolution prior — one could instead focus on a particular class of images. Zhuang *et al.* implemented a super-resolution method which focuses solely on faces and generates a prior using pairs of low and high-resolution images of a large number of faces [28]. Another method which has been investigated before, by Pickup *et al.* [21], is using prior knowledge of the textures which may be present in a scene. Once again the prior is constructed using pairs of images containing low- and high-resolution copies of the textures and the relationship between medium- and high-frequencies is learnt.

The key areas of an example-based super-resolution algorithm were originally covered by Freeman *et al.* [11, 10] and then expanded on by Bishop *et al.* [4]. Most other example based super-resolution algorithms are based on the original work by Freeman *et al.* and so this, along with the extensions for video super-resolution by Bishop *et al.* will be discussed in detail. The second paper by Freeman *et al.* covers two different methods of implementing the same algorithm, one based on the use of Markov networks using a belief propagation algorithm which requires multiple passes, and a more efficient single-pass algorithm [10].

The basic process of the super-resolution algorithm designed by Freeman *et al.* is shown in figure 2.2. The example-based super-resolution algorithm can be split into two distinct stages, a training stage and a search stage. The training stage is where the relationship between medium- and high-frequency information is learnt. This learnt information forms a fully non-parametric prior and is used during the second stage of the super-resolution algorithm to increase the

resolution of an image.

### 2.2.1.1 Training

The goal of the training stage of an example-based super-resolution algorithm is to create a searchable database representing the relationship between medium- and high-frequency detail. Freeman *et al.* [10] choose to take a generalised, non-scene-specific approach and use a collection of unrelated training images. The high-resolution training images are degraded to match as closely as possible the look and quality of the input images which are to be super-resolved later. The relationship between the degraded images and the original high-resolution images needs to be learnt and stored in the searchable training set [10]. Freeman *et al.* believe that to predict the high-frequency detail to add, only the highest frequencies of the enlarged low-resolution input image are needed [10]. The information contained in these highest frequencies should be very similar to the medium-frequency details of their original high-resolution copies.

The high-frequency components of the original high-resolution training images are extracted as these contain the fine detail which will be added to the lower resolution input image during the super-resolution search stage. The other component images needed are the highest frequency components of the degraded copies of the training images. These contain the medium-frequency detail corresponding to the high-frequency detail extracted previously and will closely match the highest frequencies found in the enlarged input images mentioned previously [10]. Figure 3.2 contains small examples of these component images. The lowest frequency detail in both degraded, and original training images can be discarded as it has little effect on the high-frequency detail present. This reduces the amount of information to be stored and increases the generality of the searchable part of the training set, which will be constructed from the medium-frequencies extracted from the degraded training images [10].

From the component images Freeman *et al.* [10] extract spatially corresponding square patches of pixels. Working on small patches of an image makes the relationship between different frequency levels much easier to model than working on the image as a whole. The pairs of corresponding medium- and high-frequency extracted patches form patch pairs. The set of all patch pairs extracted from all training images is called the patch database and stores the relationship between medium- and high-frequency detail in the training images. It is this learnt relationship which will be used to insert new high-frequency detail into the low-resolution input images later. These input images should contain medium-frequency detail which is similar to the medium-frequency information stored in the patch database.

Freeman *et al.* found that without enforcing spatial consistency in the selection of high-frequency patches during the super-resolution search stage that the high-resolution output of their algorithm ended up looking “like oatmeal” [10], with obvious edges between inserted patches of detail. The solution to this problem is to overlap the high-frequency patches of the output image during the search stage. These overlapping edges can then be compared to ensure that neighbouring patches are spatially consistent.

The method which Freeman *et al.* use forms the basis for the algorithm implemented by Bishop *et al.* [4] for the super-resolution of video. There are however some slight differences between the two and I believe that by using ideas from both I can create an efficient, and successful video super-resolution algorithm for this project. Bishop *et al.* perform training image preparation in a slightly different way by repeatedly filtering and subtracting combinations of component images. The resulting component images match what Freeman *et al.* generated, a high-frequency component image from the original high-resolution training image, and a medium-frequency component image from a degraded copy. The component images are normalised using a local measure of the energy in the image and then spatially corresponding patches of pixels are extracted to construct a patch database in the same way as Freeman *et al.*

### 2.2.1.2 Super-resolving the Input Image

With a patch database constructed the super-resolution of an input image can now be performed. Both Freeman *et al.* [10] and Bishop *et al.* [4] super-resolve images and video respectively which are unrelated to the training images used. The search process for the super-resolution of a single image will be discussed first, as this forms the basis for the work by Bishop *et al.* in super-resolving video. The first step in super-resolving an image is to enlarge it to the target output size, which should be the same as the high-resolution training image size. Due to the interpolation used when enlarging the input image, the highest frequency component of it will not include any high-frequency detail. This detail will be inserted from the high-frequency patches contained in the patch database. The highest frequency component available in the enlarged input image will be similar to the medium-frequency components extracted from the training images.

Small patches of pixels are extracted from the medium-frequency component of the input image and used to search for corresponding high-frequency detail in the patch database. Freeman *et al.* found that without enforcing spatial consistency between neighbouring patches the output looked “like oatmeal”. The reasons for this will be discussed in detail later, but arise due

to the under-constrained nature of super-resolution. To improve on these results Freeman *et al.* tried modelling the search problem as a Markov network to represent the spatial relationship between patches [10]. The set of high-frequency patches which maximise the probability of the Markov random field should be the most visually pleasing [10]. Freeman *et al.* employ an approximation algorithm called Belief Propagation to iteratively improve the probability of the network and find the most optimal set of high frequency patches. However, it was found that this algorithm converged to a satisfactory result after “typically 3 or 4 iterations” [10]. Therefore it was decided that a single pass algorithm could probably be implemented to give similar results. It is this single pass algorithm which I will base my project on and so I have not described in detail the compatibility matrices between nodes of the Markov network, however if the reader is interested a full description can be found in the original paper [10].

With the closest matching patch alone not being sufficient to generate plausible high-frequency detail, Freeman *et al.* use previously selected high-frequency patches to add further constraints. Due to the raster scan order in which patches are extracted from the input image generally a high-frequency patch already has neighbours selected above and to the left. This enables a greedy, single-pass, spatially-consistent algorithm to be developed which maximises the spatial consistency of the selected high-frequency patch in the local neighbourhood. Using this one pass algorithm instead of the Markov network, Freeman *et al.* found that the reduction in super-resolved output quality was negligible [10].

The work by Bishop *et al.* [4] in super-resolving video is similar to the Markov network approach first used by Freeman *et al.* However, instead of explicitly modelling the problem as a Markov network, they use a cost function to select the most appropriate high-frequency patch from a set of 100 candidate patches. This set of candidate high frequency patches are the patches corresponding to the 100 nearest neighbours of the current medium frequency patch from the input image. Similarly to Freeman *et al.*, Bishop *et al.* [4] found the high-frequency component image generated by their super-resolution algorithm was not spatially consistent. They added an extra term to their cost function to measure the differences between the overlapping regions of the high-frequency patches. The cost functions and further discussion about them will be contained in the design and implementation chapter of this report.

Once the most plausible high frequency patch has been selected for every extracted medium frequency patch of the input image any normalisation performed when the patch database was constructed is reversed. The high frequency patches are merged to create a high frequency component image which can be simply added to the previously enlarged input image. The

enlarged input image, which previously only contained low- and medium-frequency detail now contains a full spectrum of frequencies and thus contains both broad and fine detail. The super-resolution process is now complete and has added new high-frequency fine detail to the input image at an increased spatial resolution.

### 2.2.1.3 Reducing Artifacts in Static Areas of Video

Bishop *et al.* found that although their algorithm successfully added new high-resolution detail to their test video, there were obvious artifacts caused by the lack of temporal consistency in the algorithm. These artifacts were visible when the super-resolved video frames were reconstructed into a video sequence, and appeared as shimmering areas over successive frames [4]. This shimmering is caused by different patches being chosen for the same location over a series of frames. These artifacts are partly due to the fact each frame is super-resolved independently of the others, using no information about the previously super-resolved frames. Although the medium frequency components of successive frames may not differ greatly, the high-resolution patches selected might be highly different.

Once again this problem arises due to the ill-posed nature of super-resolution, and temporal consistency needs to be enforced similarly to spatial consistency. The cost function used by Bishop *et al.* to select the best patch from a set of 100 candidates contains no reference to previously selected high-resolution patches in previous frames. This problem is addressed in two stages, first an attempt is made to tackle the shimmering artifacts occurring static areas of the video, and then in areas where there is motion [4].

The cost function is improved by a new term whose purpose is to enforce temporal consistency in the super-resolved video output. This term, which Bishop *et al.* call the stasis prior, favours using high-resolution patches which were used at the same location in the previous frame using a weighted binary indicator function [4]. It is demonstrated through graphs showing the amount of temporal artifacts in a certain region, which Bishop *et al.* call "Flicker traces"[4], that this term successfully reduces artifacts where the video is completely static. In fact when the flicker trace for static regions of the video before the temporal term was added is compared to the flicker trace with the temporal term added it shows that artifacts in these regions have been virtually eliminated [4]. They also go on to state that the stasis prior gives a "limited improvement, if any, in regions where there is motion" [4]. It would appear however, that the improvement in regions of motion only occurs when these regions are relatively static. The factor by which the patch in the previous frame is favoured is calculated by comparing

differences across all frames and patch locations in the video [4].

#### 2.2.1.4 Reducing Artifacts in Areas of Video Containing Motion

Information about the high-resolution patches selected for previous frames was used to reduce shimmering temporal artifacts in static areas of the scene, however artifacts still occur where the video contains movement. Bishop *et al.* found that where the motion is fast then although artifacts due to temporal inconsistencies may appear, the impact on perceived visual quality is low, so much so that the artifacts are not noticed by an observer who is not looking for them [4]. In regions of relatively slow motion however the shimmering artifacts are much more noticeable. Bishop *et al.* argue that a simple solution of favouring patch pairs which were used in nearby locations in previous frames will do little to reduce the artifacts. Instead they opt to use information from the previously super-resolved frame of video to create a new smaller, and more scene specific patch database [4].

For each medium frequency patch extracted from the input image, a new patch database is constructed from a spatially corresponding rectangular window in the previous frame. This second patch database is constructed in the same way as the original patch database, learning the relationship between medium and high frequency detail and storing this as patch pairs. The medium frequency patches in these new patch pairs are much more likely to match the medium frequencies present in the current frame of the video because they have been taken from a super-resolved, higher resolution copy of almost the exact same scene present in the previous frame. If the scene changes between the previous and current frames then there is no problem, as there is no possibility of any temporal artifacts. The subsequent frame will be temporally consistent with the current frame containing the new scene. This second patch database ensures that a similar set of patches is used for successive frames, and should greatly reduce the amount of shimmering artifacts. Bishop *et al* call this new patch database the motion prior [4].

The flicker traces produced after the motion prior was added to the cost function show that a reduction in the number of temporal artifacts has been achieved in some regions . However, compared to the reduction in artifacts achieved using the stasis prior the motion prior is not as beneficial. Bishop *et al.* acknowledge this and comment on the “modest reduction in flicker over the moving region”, but highlight the fact that in frames of the video where the movement is slow the motion prior has decreased the amount of temporal artifacts successfully [4]. I would argue that the negligible improvements gained using the motion prior are not worth the decrease in computational efficiency which comes from having to construct a second, albeit smaller, patch

database from the previous frame. In fact, it is clear from the resulting flicker graphs that the motion prior has not only increased the amount of shimmering artifacts in regions of motion in a number of frames, but has also increased the number of artifacts in static regions of the video over all frames. Based on this I do not see a benefit in including the same motion prior in the super-resolution algorithm implemented for this project. Due to the scene-specific training images my algorithm will use there should not be as many artifacts anyway, as it is likely that exact, distinct matches for each medium frequency patch are present in the patch database.

## 2.3 Evaluating Super-resolution Results

Evaluation of an algorithm which improves the level of detail in a video through super-resolution is challenging as the improvements may not be obvious to someone who does not know what to look for. To evaluate the results of my project I will perform a qualitative evaluation and a quantitative evaluation. The qualitative evaluation will involve a number of human participants, who will each be asked to compare a number of videos and rate the quality of the super-resolved videos on a scale of 1 to 5. I will ask each participant to rate the quality of the super-resolved videos compared to both a true high-resolution copy and the low resolution copy which was used as input for the algorithm. A low rating would be given if the participant feels the super-resolved output video is more like the low resolution input video, a high rating would be given if it were more like the original, true high-resolution video. If the majority of participants rate the super-resolved videos highly, this would indicate that the algorithm successfully improves the perceivable quality of the input videos.

To quantitatively evaluate my videos I will use two methods of image quality assessment and apply them to each frame of my super-resolved videos. In previous works the majority of image quality assessment methods tend to be based on the pixel-to-pixel differences, the most common of these being the root mean square error (RMSE) and the peak signal to noise ratio (PSNR) which is based on the RMSE. Most previous work on super-resolution which has contained a quantitative measure of results tends to use the PSNR [3, 28]. However, it is known that the PSNR, and in fact most image quality assessment methods which are based on pixel-to-pixel differences, fail to perceive image quality in the same way as the human visual system (HVS) [27, 26, 8, 12]. Eskicioglu and Fisher[8] performed an analysis of image quality using both the HVS of a number of human volunteers and a number of pixel-to-pixel difference based image quality assessment methods. They found that there was very little correlation between the

perceived quality of images by the volunteers and the results of the image quality assessment methods. This led to the conclusion that pixel-to-pixel difference based assessment methods are not reliable for the evaluation of perceivable image quality [8].

Wang *et al.*[27] have devised a new method of image quality comparison which they have based around what is known of the HVS. They base their work on the assumption that the HVS has evolved to understand images based on the structural information contained in them. This correlates with what was discussed earlier about the regularity, or structure, which is present in all natural images as opposed to them being completely random [9]. Pixels which are spatially close in an image tend to be influenced by their nearest neighbours at least, to form edges and other structural detail. Wang *et al.* argue that error based image quality comparisons quantify the differences in images based on perceived errors between spatially corresponding pixels. However to match the HVS approach more effectively, differences should be quantified by distortions in the structural information of an image [27]. Based on this Wang *et al.* develop an image quality assessment method called the Structural SIMilarity (SSIM) index which measures these structural distortions between two images. Assessing image differences using visual structure also offers immunity to changes in lighting and contrast, as although changes in these areas will make an image differ from its original, it is likely that the structural content of the images remains the same and the original content could be recovered easily [27].

Bégin and Ferrie performed a comparison of two learning based super-resolution algorithms [11, 13] using three different image quality assessment methods. The PSNR, the correlation coefficient (CC) and the mean structural similarity measure (MSSIM) which is a mean of the SSIM index taken from multiple smaller regions of an image [2]. Their conclusion was that CC, which measures the correlation of images based on the least squares method, does not handle the comparison of super-resolution results well at all, failing to find large differences between two images which were clearly different when compared using the HVS. The PSNR, although a quality assessment method that is used widely, was found to incorrectly rate some super-resolution output as worse than output produced by a much simpler interpolation. In their conclusion, Bégin and Ferrie found that MSSIM gave statistical results which closely matched the qualitative assessments made using the HVS of several participants [2]. Based on this I will use two image quality assessments to evaluate my super-resolution algorithm, the PSNR and the SSIM index. The PSNR will be used because it has seen wide use before, and so is useful for comparing results of the algorithm in this project to other previous works. The SSIM index will also be used because as discussed it more closely matches the qualitative evaluation of image

quality as performed by the HVS. The results of the SSIM index and PSNR assessments when compared to the ground truth high-resolution videos should correlate well with the results of my qualitative testing using human participants.

Another criteria which will be used to evaluate the quality of the super-resolution algorithm will be the quality of its output in comparison to a sharpened input video. Using a sharpen filter is a much simpler, more computationally efficient way of improving the perceived quality of an image, and it should not produce higher quality results when compared to the output of the super-resolution algorithm. This criteria can again be evaluated using the image quality assessment methods described above, PSNR and SSIM. The differences between the ground truth, high-resolution videos and the sharpened videos, as quantified by PSNR and SSIM, should be greater than the differences between true high-resolution and the super-resolved output videos. This will be measured for each video using a mean PSNR and SSIM index of all frames, which should give a larger result for the super-resolved and high-resolution comparison. PSNR is measured in decibels (dB) with a higher value representing a larger amount of signal compared to noise, and hence more similarities. The SSIM index is an assessment in the range of 0 to 1, with 1 meaning the images are exactly the same, and 0 meaning there are no similarities. A SSIM index and/or PSNR which is higher for the super-resolved and high-resolution comparison than the sharpened and high-resolution comparison would indicate the super-resolved output was of a perceived higher quality.

## **2.4 Methodologies**

There are a large number of project methodologies which can be used to plan the process of designing and implementing a complex software project. During a number of university modules which have been studied during my degree I have been exposed to a number of these methodologies, the waterfall model, extreme programming and the spiral model, to name but a few. Each of these methodologies tend to be designed for software projects which are much larger than mine, and which have requirements put upon them by people other than the core developers, such as end users and company stake-holders. With a system such as the one implemented for this project, where there is only one main functional algorithm, there is little need to spend large amounts of time reassessing the high level requirements or get feedback from end users at each iteration. Instead, each rapid iteration of the system will require testing and analysis before being improved upon for the next iteration. Therefore the methodology I

will be using to create the system is evolutionary prototyping. This methodology, and other prototyping methodologies, work in four key stages:

- **Identification of Requirements** The requirements of the proposed system are analysed and identified in as much detail as possible.
- **Development of Prototype System** A prototype of the system is developed which meets as many of these requirements as possible .
- **Review of Prototype System** This prototype system is tested and reviewed in much the same way as the final system would be.
- **Revise and Enhancement of Prototype System** Finally the prototype system is revised and enhanced to add new features which were not initially identified as requirements, and improve features or fix bugs found in the review stage.

The last two stages of the prototyping methodology can be repeated as many times as necessary. There are two other types of prototyping methodology as well as the evolutionary methodology which I have chosen to use, these are throwaway prototyping and incremental prototyping. With throwaway prototyping anything developed for a prototype system is "thrown away" and not used in the final release. With incremental prototyping parts of the system are created as separate prototypes and tested individually before being combined into the final system. Each of these methodologies does not match how I plan to implement the super-resolution system. I do not plan on creating whole prototypes which are not used in the final version and I do not have distinct areas of the system which can be built as separate prototypes and tested individually.

Evolutionary prototyping differs from throwaway and incremental prototyping in that each time a prototype is produced it is fully functional and attempts to meet the overall requirements of the system. Each prototype goes on to be refined, with any bugs being fixed, any improvements to efficiency made, and new features be added to create the next prototype. These new features may be required after more research is done on a particular area for example which requires an approach not realised in the initial requirements analysis. A quote from Davis sums up the key benefit of evolutionary prototyping well: "evolutionary prototyping acknowledges that we do not understand all the requirements and build only those that are well understood." [7]

The benefits of the chosen methodology are as follows:

- There are no restrictions on exactly what needs to be produced for each iteration of the prototype, something I feel is very important in a project involving computer vision or other areas of artificial intelligence.
- The requirements of the system can change over time, and any new functionality needed can be built into the next prototype.
- It allows any changes to be made to the system quickly and effortlessly, without the need for detailed analysis and testing beforehand.
- Time is not wasted prototyping areas of the system which are never actually used in the final version.

The drawbacks of evolutionary prototyping are:

- The requirements of the system are allowed to constantly change, meaning there is the possibility of feature creep, adding more and more functionality and never ending up with a fully tested and finished system.
- Unlike other methodologies it is hard to set timescales for each iteration, because it is not known what will need to be added at each stage. This could lead the project to become organised and deadlines in the schedule to be missed.
- Previous prototype code which is reused in later versions may contain bugs due to the unrestricted nature of any rapid changes which are made.

Overall I feel that the evolutionary prototyping methodology has more valuable benefits than drawbacks. It also agrees with how I intend to tackle the research based project, implementing algorithms in as much detail as possible whilst carrying out research on other areas. Evolutionary prototyping also matches the style of software development I have used in previously successful software development projects both inside and outside of university.

## 2.5 Appropriate Technologies

For the system I propose to implement there are a number of programming languages that I could choose to use. These include:

- C

- C++
- MATLAB

Due to the nature of the system, in that it is expected to work on videos, it should be able to apply the super-resolution process to an image rapidly. The ultimate goal of real time processing would require the system be able to insert high-resolution detail into a lower resolution image in about  $1/25^{th}$  of a second. Although this is probably not possible on current computer processors it would still be good to have the system perform as efficiently as possible.

C would probably be the most efficient language to use for the implementation but I would probably have to write a lot of low level code myself, which would be time consuming. C++ would be a slight improvement on the amount of time needed to implement the system, and yet still be fast. The other advantage to implementing in C or C++ is that I would be able to produce a stand alone piece of software at the end of the project. This could include features such as a GUI for displaying input and output and components to keep the user informed about the progress.

MATLAB is a highly suitable candidate for programming language and environment. It has a number of very useful image processing features and handles matrix mathematics with great speed. Another benefit of MATLAB is that there are tools available for viewing input and output images, as well as saving the results of tests. The best method for my system may be to use a hybrid approach, using MATLAB for all input and output tasks, but writing parts of the code in C. MATLAB supports this option through MEX files. MEX files are pre-compiled functions which are stored in the form of Dynamic Link Libraries (DLLs), which can be loaded by a program to add extra functionality. By using the MATLAB command *mex* C or C++ can be compiled to a DLL ready to work with MATLAB. This hybrid approach combines a good computational speed, with the ease of use of MATLAB and should give good results.

## 3. Design and Implementation

### 3.1 Super-resolution of a Single Image

#### 3.1.1 Overview

The minimum requirement of this project is to implement an algorithm to improve the spatial resolution of an image given a similar image in a higher resolution.

The approach I have taken to solve this problem is based on previous work by Freeman *et al.* [10]. The method of applying super-resolution can be broken down into two key stages, a training stage and a search stage. These stages can be further broken down into a number of key tasks, which will be outlined below, and then explained in detail:

- **Training**

The first stage of my example-based super-resolution algorithm is the training stage, where the relationship between medium and high frequencies is learnt from scene-specific high-resolution images. Once this relationship is known it can be used to super-resolve an image of the same scene.

- **Training image preparation**

The training image from which the relationship between medium- and high-frequency detail is learnt must first be prepared for the extraction of patches. This involves filtering the image and creating two new images from the medium- and high-frequency components of the original.

- **Patch extraction**

Small square patches of pixels will be extracted from corresponding locations in these two prepared images. These patches make modelling the relationship between medium- and high-frequency detail much easier.

- **Building the patch database**

The collection of extracted patch pairs is grouped into the final training set, called the patch database.

- **Search**



Figure 3.1: The above figures show how an image is represented in the YCbCr colour space. The original image is shown on the left along with each of its three channels, a luminance channel and two chrominance channels containing blue and red levels.

Once the training stage is complete and a patch database has been constructed we can attempt to super-resolve images of the same scene. This stage is essentially a search over all regions of the medium frequencies of the input image to find matching patches in the patch database.

- **Patch extraction**

Patches are extracted in a grid pattern from the highest frequency component of an enlarged input image.

- **Patch lookup**

For each of these extracted patches the patch database is searched to find the approximate nearest neighbour (ANN). The corresponding high-frequency detail of this nearest neighbouring patch will be inserted into the enlarged input image.

- **Output construction**

Once all high-frequency patches have been found the high-frequency component image can be constructed and added to the enlarged input image to increase its level of detail.

### 3.1.2 Training

#### 3.1.2.1 Training Image Preparation

Freeman *et al.* chose to perform super-resolution on the red, green and blue (RGB) channels of the input image together [10]. For the super-resolution algorithm implemented in this project it

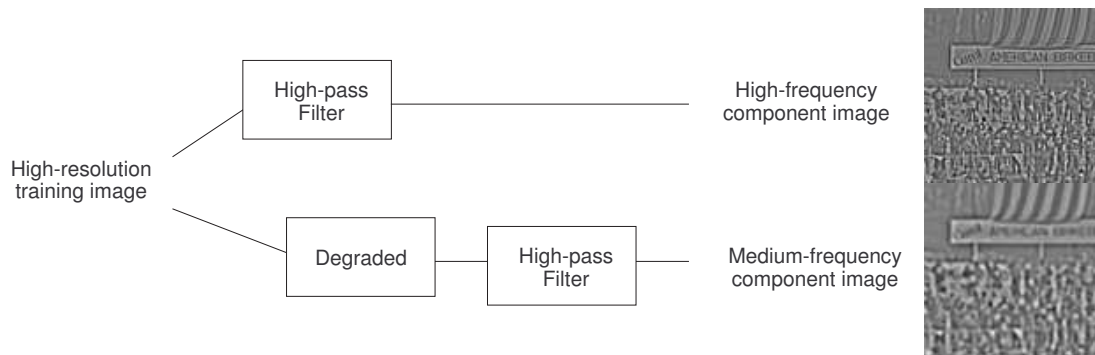


Figure 3.2: This figure shows the process used to generate the two training images used to construct the patch database. The high-resolution training image has its high frequencies extracted, as well as being copied and degraded for extraction of the medium frequencies.

was decided to only super-resolve the luminance channel (Y channel) of the image in a YCbCr colour space. The other channels, Cb and Cr, contain the colour information of the image. For an idea of what each of these channels contain see figure 3.1. The basis for choosing to only super-resolve this channel is that the luminance of an image can affect the perceived quality of an image much more than either of the other two channels. The HVS is much more sensitive to high-frequency changes in intensity than changes in colour. This property of the HVS is exploited in many compression algorithms such as that used in television transmission. This means that only adding higher frequency information to the luminance channel of an image will increase its perceived quality and also decrease the dimensionality of the patches threefold, giving a large increase in computational efficiency. For the rest of the training and search stages only the luminance channel of training and input images is used, unless otherwise stated. To create the patch database we required the medium- and high-frequency component images of each high-resolution training image. The medium-frequency component image is taken from a copy of the original training image which has been degraded to match the properties of the future input image. This involves synthesising the theoretical filtering process used to capture the input images from real world scenes (see figure 2.1). To ease the implementation and testing process needed to iteratively develop the super-resolution algorithm it was decided that the input images should be taken from known high-resolution images and simply blurred. This allows the results of the super-resolution algorithm to be compared to known ground-truth images for statistical evaluation. This means that the filtering function, which in a real world video would be hard to calculate, can be explicitly defined and used to also degrade the training

images. The input images used also lacked any compression artifacts, which mean that the degraded training images did not need to have artifacts synthetically inserted. To degrade the copies of each training image a Gaussian blur kernel is used. The Gaussian kernel used is generated using the following function:

$$G(x) = e^{-\frac{x^2}{2\sigma^2}} \quad (3.1)$$

with  $x = -4, \dots, 4$  and  $\sigma = 1$ .

Freeman *et al.* [11] assume that the high-frequency band of a signal,  $H$ , is conditionally independent of the lower frequency band,  $L$ , given the medium-frequency band,  $M$ :

$$P(H|M, L) = P(H|M) \quad (3.2)$$

This means that only the statistical relationship between the highest-frequency band of the degraded training images (medium-frequencies) and the high-frequency band of the original high-resolution training images needs to be learnt. Discarding the low-frequency details reduces the amount of information to be stored and increases the generality of the searchable part of the patch database. To obtain the high- and medium-frequency component images needed to construct the patch database a filtering process similar to that used by Bishop *et al.* is employed [4]. This process is summarised in figure 3.2. First the original high-resolution training image is run through a high-pass filter. The resulting high-frequency component contains the detail which will be added to any input images during the super-resolution search stage. The degraded copy of the high-resolution training image is run through another high-pass filter. The result of this filter is a component image composed of medium-frequency detail in comparison to the previous high-pass filter run on the original training image. We now have the two component images needed to construct the patch database.

### 3.1.2.2 Patch Extraction

To effectively store the relationship between medium- and high-frequency detail the training images are split into small patches of pixels as in previous works on example-based super-resolution [11, 10, 4]. This makes the relationship between frequencies much easier to model than using the image in its entirety. These small patches of pixels are taken from every possible location in the medium-frequency component image, and from corresponding locations in the

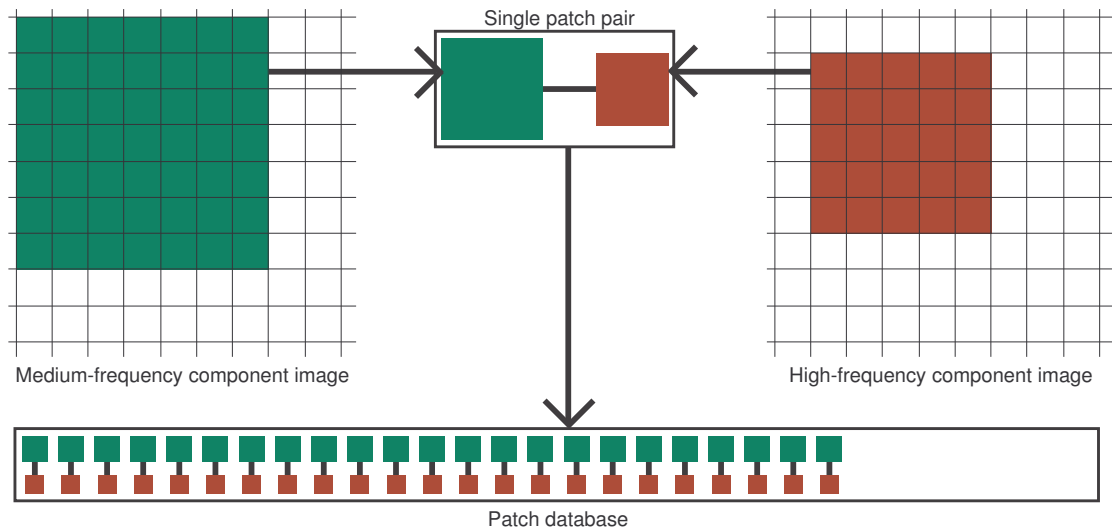


Figure 3.3: The patch database is constructed from pairs of corresponding medium- and high-frequency patches taken from the medium- and high-frequency components of the training images.

high-frequency component image. These corresponding patches form a single patch pair, which can be inserted into the patch database as shown in figure 3.3. Each patch is normalised in a similar way to Freeman *et al.*, by dividing with the mean of its absolute values. The size of the patches extracted from the medium-frequency component image is  $7 \times 7$  pixels, with corresponding  $5 \times 5$  pixel patches taken from the high-frequency component image. The patch sizes selected are the same as those used by Freeman *et al.* [11].

The size of the patches used has a large effect on the quality of the super-resolved output, and depends both on the images used for training and the images which will be super-resolved. If the training and input images were to match almost perfectly, that is they are allowed to differ in size but have exactly the same scene contained in them, then a larger patch size could be used. This would improve the quality of the results as it would mean the patch database contained a highly specific match for each patch. Although a large patch size may seem attractive due to this reason, if the scene in the input image differs from the training image by anything other than a small shift in the image plane the patch database is unlikely to contain any close matches. With small patch sizes, such as the  $7 \times 7$  pixel patches used in this project and by Freeman *et al.* [10], the contents of the patches will be much more generalised. This means that if the training and input images are very similar then a good match in the patch database for any particular input patch will still be found. If the input and training scenes differ, then due to

the generality of the patches, the patch database is still likely to contain a near match for every medium-frequency input patch. This means that although the exact high-frequencies may not be found, the super-resolution algorithm should still find plausible high-frequency detail. The plausibility of the high-frequency patch returned can be further improved by the enforcement of spatial consistency between neighbouring patches. For discussion of methods used to do this see section 3.3 later in this chapter.

### 3.1.2.3 Patch Database Construction

The patch database is constructed from all pairs of medium- and high-frequency patches extracted from a single scene-specific training image as shown in figure 3.3. It therefore contains information about the high frequencies that should be present for any particular medium-frequency patch. To allow the patch database to be searched efficiently it is stored in a single matrix, with each column containing a medium-frequency patch. To enable this each medium-frequency patch is transformed into a single column vector as shown below:

$$\begin{bmatrix} 1 & 8 & 15 & 22 & 29 & 36 & 43 \\ 2 & 9 & 16 & 23 & 30 & 37 & 44 \\ 3 & 10 & 17 & 24 & 31 & 38 & 45 \\ 4 & 11 & 18 & 25 & 32 & 39 & 46 \\ 5 & 12 & 19 & 26 & 33 & 40 & 47 \\ 6 & 13 & 20 & 27 & 34 & 41 & 48 \\ 7 & 14 & 21 & 28 & 35 & 42 & 49 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 \\ 2 \\ 3 \\ \dots \\ 47 \\ 48 \\ 49 \end{bmatrix}$$

The medium-frequency patch matrix is constructed using each of the medium-frequency patch vectors as a single column. This leads to a large  $49 \times N$  element matrix where  $N$  is the number of patch pairs extracted from the training images, in the majority of tests this number was around 30,000. The completed patch database consists of a set of patch pairs, where each pair contains a medium-frequency patch and its corresponding high-frequency patch. The medium-frequency patches are structured into a single matrix which can be searched using a patch extracted from an input image. The corresponding high-frequency patch for the closest matching medium-frequency patch can be inserted into an output high-frequency component image during the search stage.

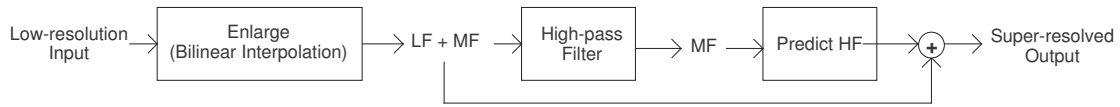


Figure 3.4: This diagram shows the process by which the super-resolution search stage operates. The low-resolution input image is enlarged using a bilinear filter. It then contains low-frequency (LF) plus medium-frequency (MF) detail. The medium-frequency detail is extracted using a high-pass filter and used to predict the high-frequency (HF) details missing. The summation of this HF detail and the LF and MF detail gives a super-resolved output image

### 3.1.3 Search

The search process is used to predict the high-frequency detail missing from an enlarged input image. A diagram and brief explanation of this process can be found in figure 3.4.

#### 3.1.3.1 Patch Extraction

To find plausible high-frequency detail to add to the low-resolution input image we must first enlarge the input image to the same size as the training images. This enlargement is done using a bilinear interpolation which as mentioned previously will not insert any high-frequency data. This enlargement ensures that any high-frequency detail that is added is of the correct scale, and also increases the chances of the medium-frequency patches of the input image matching those in the patch database. Similarly to the preparation of the training images we convert the input image into the YCbCr colour space and extract just the luminance (Y) channel to work on. From this enlarged image, which contains only low- and medium-frequency data, the medium-frequency component is extracted using a high-pass filter.

The previously enlarged input image, containing luminance and two chrominance channels, is kept so that we can add the generated high-frequency detail to it later. From the medium-frequency component of the input image we extract patches in a similar process to the patch extraction in the training stage. However in the search stage extracting all possible patches is not necessary and so we extract patches in a raster scan order on a  $5 \times 5$  pixel grid. Predicting all  $5 \times 5$  high-frequency patches on this grid allows a full high-frequency component to be constructed for the input image.

### 3.1.3.2 Patch Lookup

For each patch extracted from the medium-frequency component of the input image we find the nearest neighbour (NN) in the patch database. This is done by calculating the  $L_2$  norm between the input patch and each column of the medium-frequency patch matrix using the following function:

$$L_2(\mathbf{x}, \mathbf{y}) = \sum_i |\mathbf{x}_i - \mathbf{y}_i|^2 \quad (3.3)$$

where  $\mathbf{x}$  is the medium-frequency patch from the input image,  $\mathbf{y}$  is a patch from the patch database and  $i$  refers to a particular pixel of both patches. This must be performed for all medium-frequency patches in the patch database to find the NN patch. Once the index of the closest matching medium-frequency patch is known, its corresponding high-frequency patch can be added to the output high-frequency component image.

One early improvement made to the algorithm was to perform the  $L_2$  norm calculations for each input patch on the whole medium-frequency patch matrix in one operation, instead of for each column individually. After expanding out equation 3.3 we get the following:

$$L_2(\mathbf{x}, \mathbf{y}) = \sum_i \mathbf{x}_i^2 - 2 \sum_i \mathbf{x}_i \mathbf{y}_i + \sum_i \mathbf{y}_i^2 \quad (3.4)$$

Applying this expanded equation can be performed in MATLAB using a much more efficient sequence of matrix and vector calculations for which MATLAB is highly optimised.

### 3.1.3.3 Output Construction

Once the most plausible high-frequency patches have been selected for every location on the  $5 \times 5$  pixel grid of the input image they can be rebuilt into a high-frequency component image. This component image contains the detail which is missing from the enlarged input image. As seen in figure 3.4, the addition of this high-frequency component to the low and medium frequencies in the input image creates a super-resolved output image.

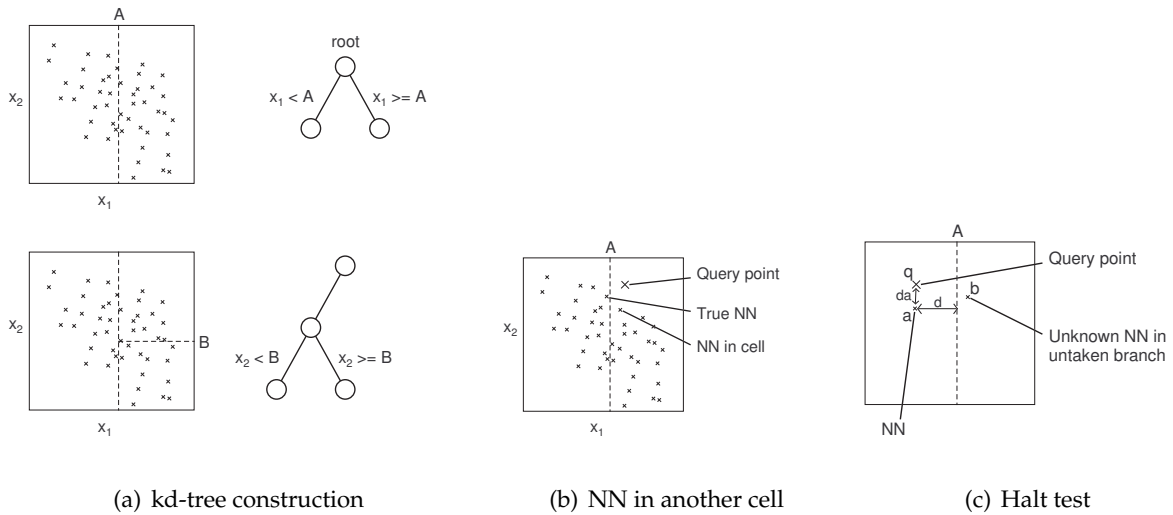


Figure 3.5: The above images show examples of particular kd-tree properties in 2-dimensions. Figure (a) shows the construction of a kd-tree by recursively partitioning the search space. Figure (b) shows that sometimes the true NN to a given query point can lie in a cell which has not been searched. Figure (c) shows how the backtracking process of the kd-tree search can check whether to search other cells of tree or to whether to halt.

## 3.2 Improving Search Efficiency

### 3.2.1 Patch Database Storage Method

In the first prototype of the algorithm the medium-frequency patches of the patch database were stored as a  $49 \times N$  element matrix. This meant that an exhaustive search needed to be performed to find the closest matching patch for each medium-frequency input patch. This led to the super-resolution process taking up to 30 minutes to super-resolve a  $640 \times 480$  pixel image to twice its original size during testing. This is due to a full patch database search being performed for every medium-frequency patch from the input image of which there are in the order of 10,000. Freeman *et al.* [10] and Bishop *et al.* [4] chose to use a highly efficient search structure called a kd-tree instead of a single large matrix.

### 3.2.2 Use of a kd-tree for efficient search

A kd-tree works on the principle of “branch and bound”. The search space defined by the medium-frequency patches is recursively partitioned into smaller cells to form a binary tree. At each level of the tree the data is split along the dimension of maximum variance at the median.

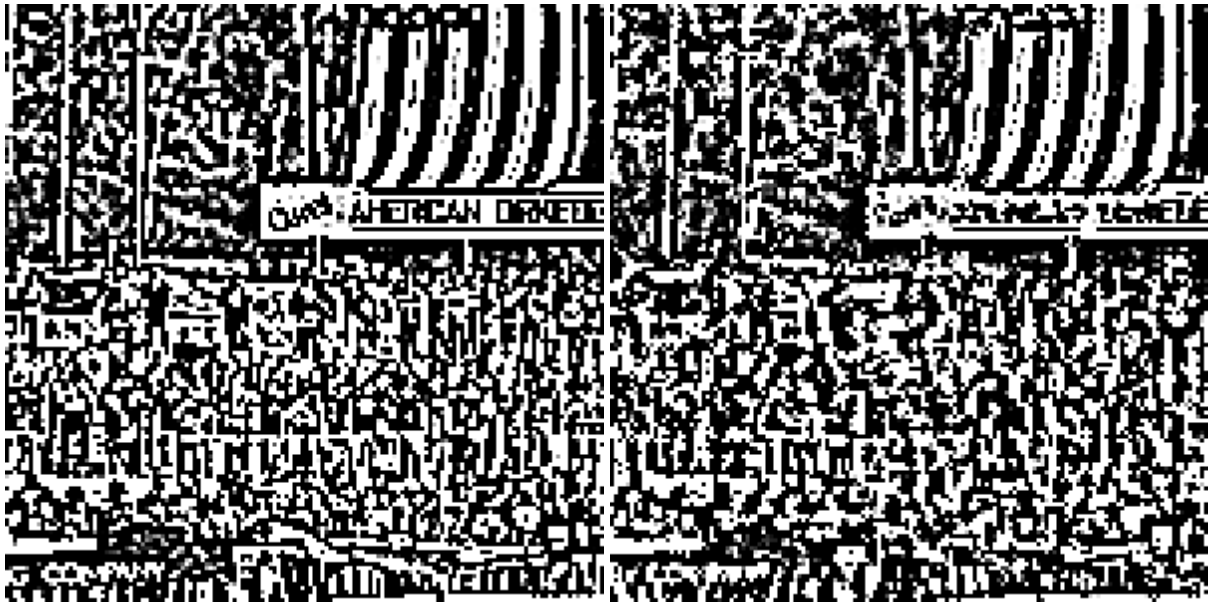
The set of data on each side of this median value forms the next branches of the tree as shown in figure 3.5(a).

To find the NN for a given query point in the  $k$ -dimensional space the tree is traversed according to the tests performed at each node until a leaf node is reached. This testing procedure compares the median value at each node to the corresponding element of the query vector. Once a leaf node is reached the distance between the point(s) at that leaf node and the query point is calculated. Due to the sub-optimal nature of this binary search problems can arise. Sometimes the NN may not lie in the same cell as that query point as shown in figure 3.5(b).

The solution to this is to maintain an upper bound of the distance between the query point and its NN i.e. the minimum distance between the query point and any points tested at the leaf nodes. The path taken through the tree is then tracked backwards and each branch not taken is explored. If the distance of any point in that branch exceeds the upper bound then the search is stopped and the NN is returned. Figure 3.5(c) shows how the search routine decides whether a particular branch is worth exploring further. Referencing the points shown in the figure the search does not continue down into the right hand cell due to all points in that cell being at least a distance of  $d$  away from the query point. This distance  $d$  is greater than the distance  $da$  and so no point in the right hand cell is closer than point  $a$ . These measurements are performed in the single dimension tested held in the current node of the tree.

Another problem with using kd-trees to perform a search is that it only works for moderately low dimensional search spaces. In high dimensions the upper bound on the distance changes very slowly so the whole tree ends up being searched. Therefore no improvement over an exhaustive search is gained. The solution to this is use an ANN search. Here we find a point which is close to the query point but not guaranteed to be the closest. For further details of this algorithm, which is out of scope of this project, see Lowe [18]. The basic idea is to maintain a queue of “promising” nodes based on an approximation of the upper bound of distances to points in that node (similar to the exact NN search). These nodes are tested according to this queue and the search is aborted when a user-specified number of points has been detected.

Use of the kd-tree for the algorithm developed in this project dramatically increased the computational efficiency of the search. Super-resolution time for a single frame was reduce from around 30 minutes to around 15 seconds using an ANN search and aborting after testing 10 “promising” nodes.



(a) High-frequency component of training image

(b) High-frequency output

Figure 3.6: This figure shows the differences in spatial consistency between the current algorithm output and the original image. The output high-frequencies in (b) lack the spatial consistency that is apparent in (a) and hence the output looks “like oatmeal” as Freeman *et al.* described.

### 3.3 Improving Spatial Consistency

The super-resolved output generated by the algorithm so far lacks spatial consistency. This is most obvious when the high-frequency component which is added to the enlarged input image is compared to the high-frequency component of the training images. Figure 3.6 shows a section of these two images for comparison. This shows that relying on the approximate nearest neighbouring medium-frequency patch alone is not enough to generate visually pleasing output. Freeman *et al.* also found that using just the nearest neighbour medium-frequency patch to construct plausible high-frequency detail did not create visually pleasing results and commented that their results looked “like oatmeal” [10]. The reason for this stems from the problem of super-resolution being under-constrained. For every medium-frequency patch there are a large number of possible high-frequency patches which correspond. For an example see figure 3.7. The example patch in 3.7(a) has been used to search for corresponding patches in an example patch database. The 16 nearest matches are shown in figure 3.7(b) and all look relatively similar. However, the high frequency patches corresponding to each of these medium frequency patches, shown in figure 3.7(c) look a lot less similar to each other. This indicates that although

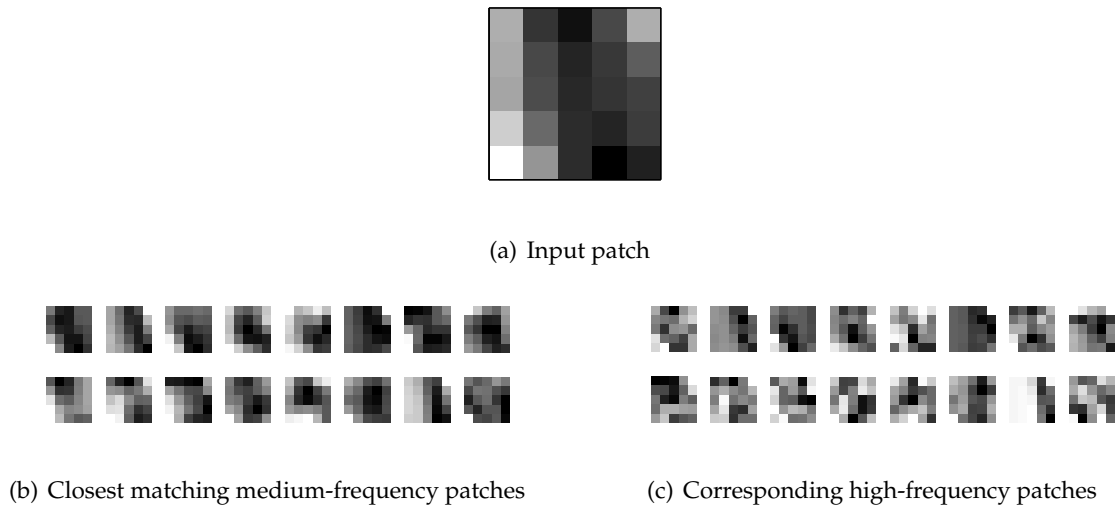


Figure 3.7: This figure shows a selection of  $5 \times 5$  pixel medium- and high-frequency patches resulting from a search of the patch database kd-tree using the  $5 \times 5$  pixel patch also displayed. Clearly the medium-frequency patches are much more similar to each other than the corresponding high frequency patches.

the algorithm may correctly find a full set of similar medium-frequency patches in the patch database, the corresponding high-frequency output can lack consistency (see figure 3.6). This leads Freeman *et al.* to the conclusion that simply searching for the nearest match for the patch alone, without using any other constraints, is not sufficient to generate plausible high-frequency detail [10].

### 3.3.1 Previous Solutions

To improve on these results Freeman *et al.* first tried modelling the search problem as a Markov network representing the spatial relationship between patches. To measure the spatial consistency between neighbouring high-frequency patches they used a 1-pixel overlap in which pixels from multiple patches should be similar, if not equal [10]. The Markov network is constructed with the known medium-frequency patches from the input image as the observation nodes. The states of the hidden nodes are the “16 or so” closest matching medium-frequency patches found in the patch database. The set of medium-frequency patches which maximise the probability of the Markov random field should give the most visually pleasing high-frequency patches [10]. Freeman *et al.* employ an efficient approximation algorithm called Belief Propagation to iteratively improve the probability of the network and find the most optimal set of high-frequency

patches. However, it was found that this algorithm converged to a satisfactory result after “typically 3 or 4 iterations” [10]. Therefore it was decided that a single-pass algorithm could probably be implemented to give similar results. It is this single-pass algorithm which I will use to enhance the spatial consistency of the super-resolution algorithm for this project and so I have not described in detail the compatibility matrices between nodes of the Markov network, however if the reader is interested a full description can be found in the original paper [10].

With the closest matching patch alone not being sufficient to generate plausible high-resolution detail, Freeman *et al.* use previously selected high-resolution patches to add further constraints. Due to the raster scan order in which patches are extracted from the input image generally a high-frequency patch already has neighbours selected above and to the left. Due to the high-frequency patches being selected on a  $4 \times 4$  pixel grid there is an overlap of 1-pixel on the top and left edge of the high-frequency patches. When the patch database is being constructed Freeman *et al.* concatenate the overlapping edges of high-frequency patches onto their corresponding medium-frequency patch vector, as shown in figure 3.8. This allows the patch database to be searched with a medium-frequency patch which also includes the overlapping edges of previously selected high-frequency patches.

When an input image is super-resolved there is no previous information for the first patch so the algorithm must fall back to using just the closest medium-frequency patch found in the patch database. This means that the one-pass algorithm is a greedy method, with the choice of this first patch influencing the selection of every other patch as it must be spatially consistent with the first one selected. Once the first row of high-frequency patches has been selected, and as long as the current patch is not on the left-hand edge of the input image, there will be overlapping high-frequency pixels. Freeman *et al.* concatenate the bottom row of pixels from the previously selected high-frequency patch above patch above, and the right column of the high-frequency patch selected to the left onto the medium-frequency input patch as shown in figure 3.9. The full vector is then normalised by dividing by the mean of the absolute value of the medium frequency patch. The high-frequency patch edges are modified with a weighting variable,  $\alpha$ , to specify the importance of enforcing spatial consistency over selecting the closest matching medium-frequency patch. This weighting variable is calculated using the following function, where  $M$  is the size of the medium-frequency patches and  $N$  is the size of the high frequency patches:

$$\alpha = 0.1 \frac{M^2}{2N - 1} \quad (3.5)$$

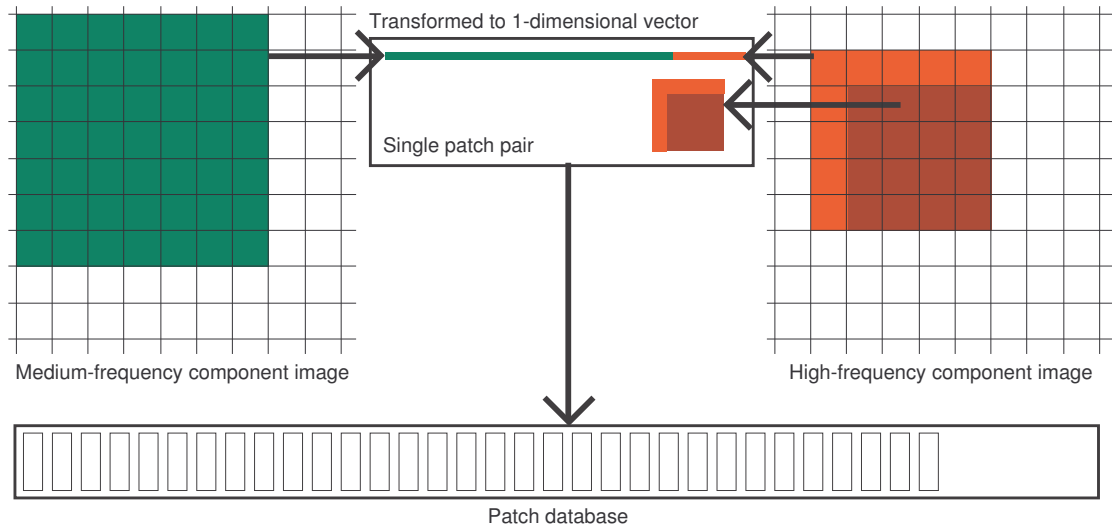


Figure 3.8: Freeman *et al.* enforce spatial consistency using an overlap in high-frequency patches. This means that the overlapping edges of high-frequency patches must be included with their corresponding medium-frequency patch in the patch database.

With values of  $M = 7$  and  $N = 5$  the weighting term  $\alpha$  is roughly 0.6. The resulting search vector matches the construction of the vectors used to build the patch database, which included the top and leftmost pixels of each high frequency patch (see figure 3.8). Compared to the Markov network method, it is now much easier to perform a search of the patch database kd-tree to find the approximate nearest matching vector in a single pass. From this nearest matching patch database vector the most corresponding high-frequency patch can be used to build the output. Using this one pass algorithm, Freeman *et al.* found that the reduction in super-resolved output quality was negligible [10].

The work by Bishop *et al.* [4] in super-resolving video is similar to the Markov network approach first used by Freeman *et al.* However, instead of explicitly modelling the problem as a Markov network, they use a cost function to select the most appropriate medium-frequency patch from a set of 100 candidate patches. This set of candidate patches are the 100 approximate nearest neighbours of the current medium-frequency patch from the input image. The cost function devised by Bishop *et al.* is based on the  $L_2$  norm difference between the medium-frequency patch from the input images, and each candidate patch from the set of 100 [4]. Let the current patch from the medium-frequency component of the input image be  $x$  and the medium-frequency candidate patches from the set of 100 nearest neighbours be  $x_k$  where  $k = 1, \dots, 100$ . The cost function [4] which determines the closest matching candidate patch is:

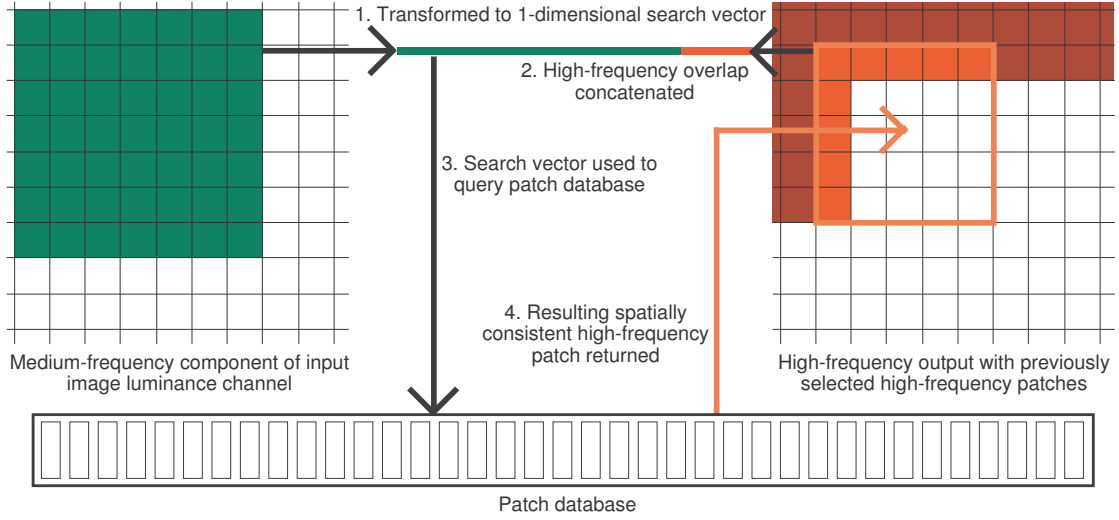


Figure 3.9: The improved search process of the one-pass spatial consistency enforcing super-resolution algorithm by Freeman *et al.* Overlapping edges from previously selected high-frequency patches are used to ensure the selected high-frequency patch is spatially consistent with those previously chosen.

$$\mathcal{L}_k(x) = \|x - x_k\|^2 \quad (3.6)$$

Again, Bishop *et al.* found that finding the closest matching patch without enforcing spatial consistency did not produce visually pleasing results. They improved their cost function by including a term to measure the difference between the overlapping edges of the candidate patches corresponding high-frequency patches, and the previously selected high-frequency patches. This overlap comes from the  $4 \times 4$  grid layout used to extract the patches, as used by Freeman *et al.* Let the high-frequency patches from the 100 candidate patch pairs be  $y$ . The improved cost function devised by Bishop *et al.* to enforce spatial consistency is:

$$\varepsilon_k^{(a)}(x) = \mathcal{L}_k(x) + \alpha \mathcal{V}(y_k, \{y_{k'} : k' \in \mathcal{N}_x\}) \quad (3.7)$$

where  $\mathcal{N}_x$  is the set of previously chosen high frequency patches which are neighbours of  $x$ , and  $\mathcal{V}()$  measures the  $L_2$  norm of all three colour channels in the overlapping regions [4]. The  $\alpha$  parameter of the cost function is used to control the importance of enforcing spatial consistency over using the patch pair with the closest matching medium frequency patch. The value used by Bishop *et al.* is calculated using the function derived by Freeman *et al.* which is shown in equation 3.5. The patch with index  $k$  which minimises  $\varepsilon_k$  is the patch selected as giving the

most plausible high-resolution detail. Once again, due to the patches being extracted from the input image in a raster like order,  $\mathcal{N}_x$  usually contains just the patch to the left and above the current [4].

### 3.3.2 Changes to the Algorithm

Based on the improvements made by Freeman *et al.* the patch extraction during the training stage was modified to include the appropriate high-frequency patch edges as shown in figure 3.8. The search stage was also modified by first changing the layout of patch extraction from a  $5 \times 5$  pixel grid to a  $4 \times 4$  pixel grid. This induces a 1-pixel overlap in the high-frequency patches selected which can then be used to influence the selection of future high-frequency patches. The overlapping edges of previously selected high-frequency patches are concatenated onto the medium-frequency patch to form a search vector similar to those in the patch database. As shown in figure 3.9 this search vector is then used to query the patch database. The returned high-frequency patch is much more likely to be spatially consistent with previously chosen high-frequency patches.

These changes successfully increased the spatial consistency of the high-frequency output of the super-resolution search stage. The perceived visual quality of the super-resolved output image was also increased, with a visible increase in the amount of plausible high-frequency detail added as shown in figure 3.10. The results of the algorithm before and after these changes were made will be evaluated in detail later.

## 3.4 Super-resolution of Video

### 3.4.1 Overview

Essentially a video clip is a sequence of images shown in rapid succession, usually at a frame rate of around 25 frames per second. At this speed the HVS no longer senses a series of changing still images and instead believes it is seeing true motion. Applying the super-resolution algorithm developed for single images to each frame of a video independently should create visually pleasing output. Unfortunately super-resolving video is not as simple as this, and super-resolving each frame with no information about previously processed frames leads to visibly distracting artifacts. The steps involved to extend the single image super-resolution process to videos, and how I created examples for testing and evaluation purposes is detailed below:



(a) Region of super-resolved image before enforcing spatial consistency (b) Same region of image super-resolved with spatial consistency enforced

Figure 3.10: The enforcement of spatial consistency in the selection of high-frequency patches has led to an increase in the level of visually pleasing high-frequency detail in the resulting super-resolved image. This improvement is most noticeable in areas of text in the super-resolved images.

### 3.4.2 Training

To develop the single image super-resolution algorithm the majority of testing was done on examples where the training images and input images contained precisely the same scene, but with the input images being of lower resolution. To develop a video super-resolution algorithm similar examples for video are required, where the ground truth high-resolution detail is known. This will allow evaluation of the algorithm output both qualitatively and quantitatively against the ground-truth high-resolution originals. The videos I used for testing were collected from the Microsoft WMV HD Content Showcase [6], and were at a resolution of either  $1280 \times 720$  pixels (also known as 720p) or  $1980 \times 1080$  pixels (also known as 1080p). The four video clips chosen for testing and evaluation can be seen in the evaluation section of this report, where the reasons for choosing these particular videos are also explained. The collection of video clips found at this location are particularly appropriate for the creation of super-resolution examples as they are in high-definition (HD) formats, and used to showcase the detail present in high definition video. From these videos I took short 200 frame clips which would become the test and evaluation

videos for the algorithm.

To generate examples for testing and evaluation a number of frames were extracted from the original HD videos. These high-resolution frames, which would be used to train the super-resolution algorithm, were extracted at a number of different intervals for testing purposes. This matches the work done previously by Kong *et al.* [17] to test their scene-specific video super-resolution algorithm. Once the high-resolution training frames have been extracted from the video clips the videos can be filtered in preparation for being super-resolved. This involves blurring and then sub sampling each frame to end up with a video half the resolution of the original. The blurring was performed using a simple Gaussian blur similar to that used in training.

### **3.4.3 Search**

Each frame of the low-resolution, degraded input videos is super-resolved independently. Frames where a high-resolution training image was taken tend to super-resolve almost perfectly, however between these frames there exists content which may not be present in the training images. This scenario was discussed earlier and led to the selection of a relatively small medium-frequency patch size. It is hoped that when an area of the input image which was not present in the training image is split into small patches, these patches will be fairly general. This increases the chance that a closely matching patch will be contained in the large number of small patches in the patch database. So, although the correct corresponding high frequency detail will not be found, it is hoped that the high frequency detail inserted will at least be plausible, and look visually pleasing. Once each frame has been super-resolved the output images can be reconstructed into a video file and viewed as a super-resolved video.

## 4. Results and Evaluation

The super-resolution algorithm created for this project was tested on four videos for which the high resolution ground-truth was available. These high-resolution videos were sub-sampled by a factor of 2 to create low resolution videos to super resolve. For the majority of experiments high-resolution images were taken every 15 frames from the ground-truth video and used as training images for the super-resolution algorithm. The resulting super-resolved videos were then compared to the ground-truth high-resolution versions using two image quality assessment methods, the PSNR and SSIM. The mean PSNR and SSIM and the standard deviations over all frames of each of the four test videos are discussed. Full PSNR and SSIM graphs showing comparisons between a number of different versions of each frame, along with a selection of SSIM error images for each video will also be shown and discussed. A larger collection of SSIM error images and examples of individual low resolution input frames, high resolution training frames, super-resolved frames, and high-frequency components added during the super-resolution process can be found in Appendix B. Before discussing the quantitative assessment of each video I will discuss the perceived enhancement in quality using enlarged regions of interest.

### 4.1 Evaluation of Perceived Visual Improvements

The four videos chosen to perform the testing and evaluation of the super-resolution algorithm each have interesting properties. These properties which make them useful in evaluating the super-resolution algorithm will be described in detail below, and in-depth analysis of the visual quality of resulting videos will be discussed.

#### 4.1.1 Bridge Video

The bridge video (figure 4.1(a)) is an interesting video to super-resolve as it contains both moving and static content in fine detail. The video shows a large number of marathon runners travelling across a suspension bridge, taken from a camera which appears to be mounted on top of one of the bridge towers looking down. This video contains a slow moving background, in the form of the water moving beneath the bridge, as well as the static parts of the bridge itself. These static parts of the bridge include the suspension cables which appear to consist of 4 smaller individual cables each in the original high-resolution version of the video. After the video has been prepared to be used as input for the super-resolution algorithm each cluster of cables



(a) Bridge video



(b) Pit stop video



(c) Climb video



(d) Ski video

Figure 4.1: The four videos chosen for testing and evaluation of the super-resolution algorithm. Each video was specifically chosen due to their availability in high-resolution, and their interesting properties.

appears as one poorly defined blurry line (see figure 4.2(d)). The marathon runners moving across the bridge are highly distinct in the original video, yet after being degraded to use as input it becomes nearly impossible to distinguish between two runners. Instead of appearing as a large number of individual runners they instead look like constant stream of moving coloured texture.

Figure 4.2 shows a number of images taken from a region of the bridge video which includes a number of different types of object as discussed in the previous paragraph. Looking at the true high-resolution image (figure 4.2(a)) one can see that this image region contains static content in the form of the fine suspension cables, the bridge deck and the rocks at the base of the bridge tower. This particular image region also contains the slowly moving background, including the shadow of the bridge to the left. Finally the marathon runners are visible on the bridge deck with each one being distinguishable in the true high-resolution image. The input image (figure 4.2(d)) is generated by performing a Gaussian blur on the true high-resolution original

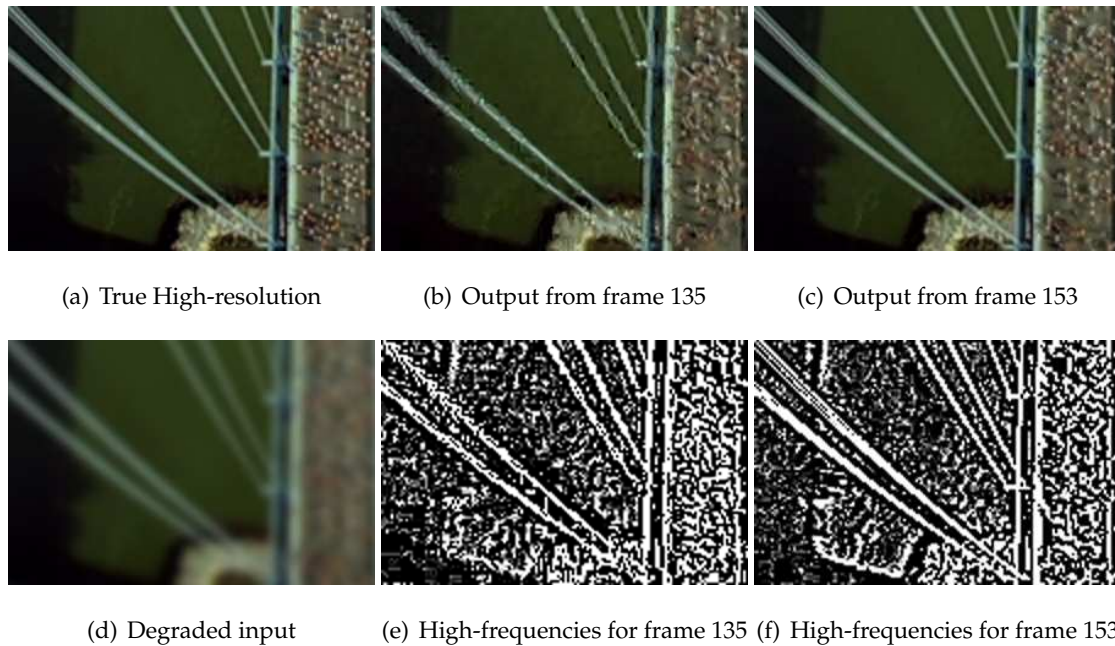


Figure 4.2: Each image shows an enlarged (2x) region from the bridge video. The high-frequency images shown are added to the degraded input to get the output for each frame. Frame 135 is after the scene has changed and before a new training image is loaded.

image. It is clear that this input image lacks high-frequency fine detail due to the obvious blurring of the suspension cables and marathon runners. The super-resolved output of two frames of the bridge video are shown in figures 4.2(b) and 4.2(c). This super-resolved output was created using training images taken every 15 frames, the image quality assessment graphs are shown in figure 4.3 and the mean and standard deviations across all frames in table 4.2. The frames that these images were taken from are numbers 135 and 153. Frame 135 (figure 4.2(b)) is of particular interest because it occurs just after the scene in the video has changed. From the image quality assessment graphs (figure 4.3) it is clear that the quality drops sharply at this frame to the lowest statistical quality level in the whole video. The scene-specific training image which was last loaded into the patch database is from the previous scene in the video. Due to this the super-resolution algorithm has performed relatively poorly and failed to insert visually pleasing high-frequency detail, in fact the sharpened input video is statistically higher quality when measured using the SSIM index. In comparison to this, frame 153 4.2(c) has been super-resolved after a new training image has been loaded. There are obvious visual differences between the quality of the two images. For instance, the suspension cables in frame 153 are clearly visible, and towards the top left of the zoomed region it is clear that they are composed

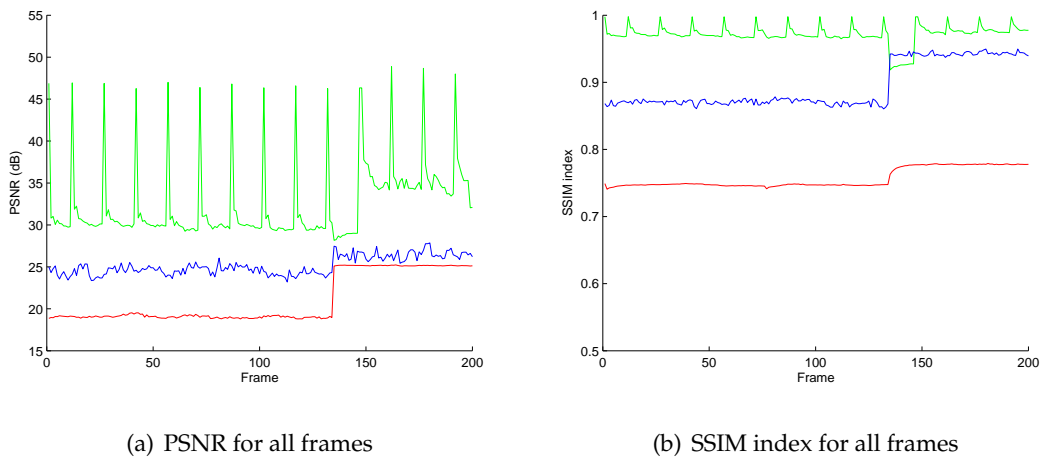


Figure 4.3: This figure shows the PSNR and SSIM index for three different comparisons over all frames of the bridge video. The green line shows the output quality, the blue shows sharpened input quality and the red shows low-resolution input quality, all in comparison to the ground-truth high-resolution video. Spikes can clearly be seen in the output quality where new training images are loaded every 15 frames.

of a number of smaller cables. This is an obvious improvement over the input image displayed in figure 4.2(d).

However in frame 135, where the patch database does not contain exact matches for the static content in the image, the algorithm has failed to find perfect matches for the input patches. This has led to the cables appearing less clearly defined than those in frame 153 as the high-frequency detail inserted does not precisely match the angles of the cable edges. The differences in the plausibility of the high frequency detail added is clarified by comparing the two high-frequency component images shown in figures 4.2(e) and 4.2(f). These high-frequency component images are the result of the search stage of the super-resolution algorithm and are added to the enlarged input frames from the original video. Figure 4.2(f) contains the high frequency detail which when added to figure 4.2(d) gives the super-resolved output in figure 4.2(c).

The marathon runners have been improved somewhat by the super-resolution process, although they do not appear as high-quality as in the original high-resolution video. In frame 135 (figure 4.2(e)), just after the scene has changed, the marathon runners are much more clearly defined than in the input video (figure 4.2(d)). However the scene-specific image used to super-resolve this particular frame did not include the marathon runners at this particular scale or angle and so they are not as well defined as in frame 201 (figure 4.2(c)). Overall the video

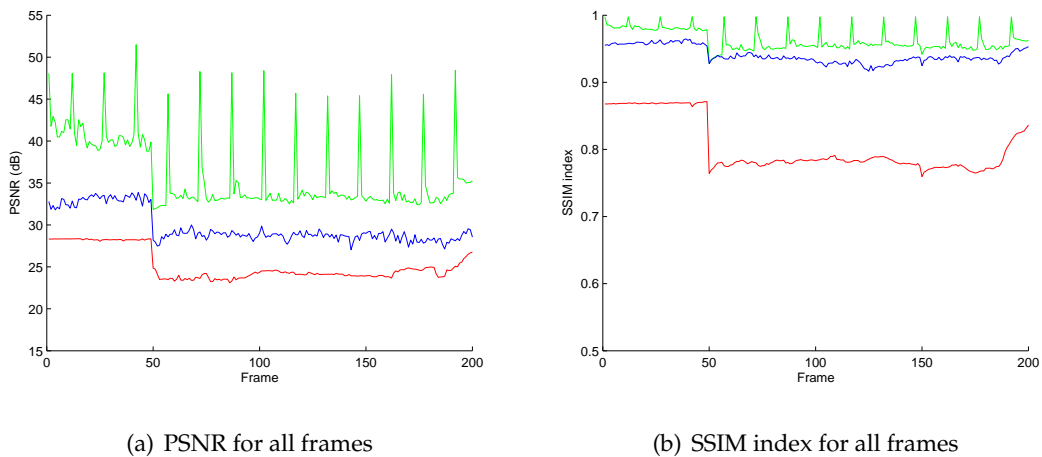


Figure 4.4: This figure shows the PSNR and SSIM index for three different comparisons over all frames of the pit video. The green line shows the output quality, the blue shows sharpened input quality and the red shows low-resolution input quality, all in comparison to the ground-truth high-resolution video. Spikes can clearly be seen in the output quality where new training images are loaded every 15 frames.

quality has been vastly improved from the input video quality and so I would argue that the super-resolution algorithm is successful for this particular video.

### 4.1.2 Pit Stop Video

The pit stop video (figure 4.1(b)) shows a close up of a race car driver sat in his car, then changes camera view to see the car surrounded by mechanics and then accelerate away. Due to the relatively static nature of the first scene in the video the super-resolution algorithm does fairly well. In all experiments (see figures 4.4 and 5.2 in appendix B) both the PSNR and SSIM index values are much higher during this static scene than in the latter part of the video. However, throughout the scene the super-resolved output has been rated as having a much higher level of similarity with the ground-truth high-resolution than the sharpened input video has.

### 4.1.3 Climb Video

The video showing a man hanging off an overhanging rock face (figure 4.1(c)) does not super-resolve particularly well. Although the PSNR graphs for the output are generally higher than for the sharpened input the SSIM graphs rate it lower in every experiment (see figures 4.5 and 5.3). This video is of particular interest because the camera is constantly rotating and the scene

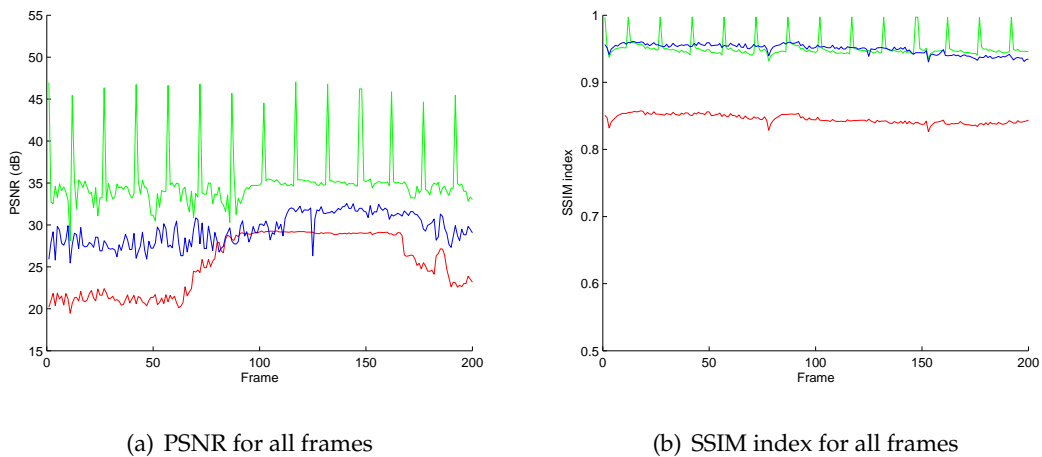


Figure 4.5: This figure shows the PSNR and SSIM index for three different comparisons over all frames of the climb video. The green line shows the output quality, the blue shows sharpened input quality and the red shows low-resolution input quality, all in comparison to the ground-truth high-resolution video. Spikes can clearly be seen in the output quality where new training images are loaded every 15 frames.

content is composed of two main parts, the highly textured rock face, and the block colour plain blue sky. The jump in the PSNR of the input quality shown in the graphs is due to the video becoming mostly blue sky at around frame 80. However the super-resolution algorithm does perform well when high-resolution training frames are taken every 15th frame, as would be expected. But even then, when measured using the SSIM index the peaks above the sharpened input video quality only last 1 or 2 frames.

#### 4.1.4 Ski Video

The ski video (figure 4.1(d)) shows a large crowd and a large number of cross-country skiers beginning a race. Behind them are a number of banners and trees. This video contains the highest quantity of high-frequency content. Due to this the PSNR results are much lower in comparison to the other videos (see figures 4.6 and 5.4). However the super-resolution algorithm has managed to improve the video in all test cases beyond the quality which was present in the sharpened input videos. This is surprising given that the camera used to capture the scene is constantly panning to the right, meaning that new content is always entering the frame. I would image that due to this new content being similar to the content already in the training images, i.e. crowds of people and trees, there exists relatively good matches for every medium-

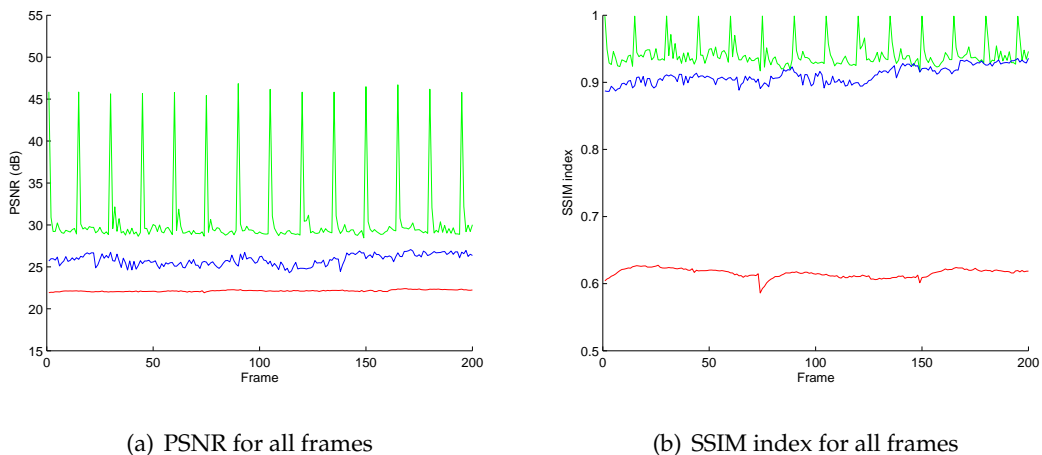


Figure 4.6: This figure shows the PSNR and SSIM index for three different comparisons over all frames of the ski video. The green line shows the output quality, the blue shows sharpened input quality and the red shows low-resolution input quality, all in comparison to the ground-truth high-resolution video. Spikes can clearly be seen in the output quality where new training images are loaded every 15 frames.

frequency input patch. This means that although the high-frequency output is not exact, it is highly plausible.

## 4.2 Quantitative Evaluation

A number of experiments were run on the four test videos shown in 4.1 with high-resolution training images taken at different intervals. The mean PSNR and SSIM index results are shown in tables 4.2 and 4.2.

The results of the quantitative evaluation show that in most cases the super-resolved output of my algorithm is closer in statistical quality to the ground-truth high-resolution videos than both the input video and sharpened video. However, the SSIM index of the sharpened climb video is very similar to the SSIM index of the super-resolved climb video, differing by only a SSIM index of 0.001. The camera used to capture this video is generally rotating throughout, meaning that the super-resolution algorithm has a hard time finding perfect matches in the patch database. This could perhaps be improved by storing rotated medium- and high-frequency patches in the patch database, but this would decrease the efficiency of the algorithm greatly. I believe that the sharpening effect has almost matched the output quality due to the nature of the scene rotation,

<b>Metric</b>	<b>Video</b>	<b>Input</b>	<b>Output</b>	<b>Sharpened</b>
<b>PSNR (dB)</b>	Bridge	21.1 (2.89)	31.4 (2.63)	25.1 (1.08)
	Pit stop	25.2 (1.84)	35.1 (3.15)	29.7 (1.94)
	Climb	25.4 (3.48)	34.3 (1.47)	29.5 (1.83)
	Ski	22.1 (0.988)	29.4 (0.624)	25.8 (0.657)
<b>SSIM index</b>	Bridge	0.757 (0.014)	0.969 (0.013)	0.894 (0.034)
	Pit stop	0.803 (0.039)	0.960 (0.013)	0.940 (0.012)
	Climb	0.845 (0.0065)	0.948 (0.0056)	0.950 (0.0077)
	Ski	0.616 (0.0063)	0.937 (0.0098)	0.911 (0.013)

Table 4.1: This table shows the mean and standard deviation (in brackets) for the PSNR and SSIM index across each video. The PSNR and SSIM results were calculated by comparing the ground-truth high-resolution video frames to the low-resolution input, super-resolved output and sharpened input video frames. For this set of results high-resolution training frames were loaded at 15 frame intervals. The corresponding, perfectly super-resolved frames were not included in the results.

and the majority of the scene content being a cliff face. The texture of this cliff face is likely to see a greater improvement from sharpening than from super-resolution due to the rotation mentioned.

The PSNR and SSIM results for the super-resolved videos all differ by at least two standard deviations from the input videos in comparison to the ground-truth high-resolution. This shows that the algorithm has, without doubt, increased the statistical quality of the videos. The worse resulting video is the climb video again, which although sees a slightly better result using just a sharpen still shows an improvement of over 10 standard deviations between the input and output.

The video which shows the least similarities with the ground-truth high-resolution video when measured using both the PSNR and SSIM is the ski video. I believe that this is due to the content of this particular video. Where as the other three videos contain at least some large areas of colour, or static areas, the ski video contains none. The camera is also constantly panning meaning that there are never any static areas and there is a constant stream of new content appearing on the right hand edge of the video. In this particular set of tests, where a new training image is loaded every 15 frames and used to super-resolve the following 14, this new

content will never be present in the patch database. The super-resolved output is still a vast improvement over the input with both measurements, being over 7 standard deviations better when measured using the PSNR and over 30 standard deviations better when measured using the SSIM index.

<b>Metric</b>	<b>Video</b>	<b>First frame</b>	<b>First and last</b>	<b>Every 50<sup>th</sup></b>	<b>(No Spatial)</b>
<b>PSNR (dB)</b>	Bridge	29.4 (0.44)	31.1 (2.01)	30.8 (2.13)	30.0 (1.60)
	Pit stop	34.0 (3.56)	34.3 (3.26)	34.6 (3.02)	33.6 (2.60)
	Climb	32.5 (1.03)	33.2 (0.94)	33.2 (1.39)	32.9 (0.92)
	Ski	28.75 (0.26)	29.1 (0.27)	29.0 (0.48)	28.4 (0.17)
<b>SSIM index</b>	Bridge	0.955 (0.018)	0.969 (0.033)	0.966 (0.012)	0.960 (0.011)
	Pit stop	0.952 (0.016)	0.955 (0.015)	0.957 (0.013)	0.951 (0.014)
	Climb	0.932 (0.0078)	0.938 (0.0056)	0.940 (0.0057)	0.934 (0.004)
	Ski	0.921 (0.0053)	0.931 (0.0049)	0.931 (0.012)	0.916 (0.005)

Table 4.2: This table shows the mean and standard deviation (in brackets) for the PSNR and SSIM index across each video using a different number of high-resolution training images. For the first column only a single high-resolution image of the first frame was used. The second column shows the results of using the first and last frame. The third column shows the results of using every 50th frame. The final column shows the results of using every 15th frame, but not enforcing spatial consistency. In each case the frames for which the true high-resolution was known were not included in the calculations.

As one would expect the results tend to get much better as training images are used more frequently. This is due to the increased likelihood that content in a particular frame is also present in the current loaded training image. For the videos super-resolved using the first and last high-resolution frame both were loaded into a patch database which was twice the usual size. Hence in some cases the results for this particular experiment are better than using every 50th frame. Graphs of PSNR and SSIM results across all frames can be found in Appendix B at the end of this report.

### 4.3 Qualitative Evaluation

To qualitatively evaluate the results of the super-resolution algorithm developed for this program a small pilot study was carried out. This involved questioning 15 people about the quality of my output videos in comparison to the input videos and ground-truth high-resolution videos. Participants were asked to rate the videos on a scale of 1 to 5 where 1 indicated that they believed there was no difference between the input video and super-resolved output, and 5 indicated that they could see no difference between the true high-resolution and super-resolved output. The results of this survey are shown in table 4.3.

Video	1	2	3	4	5
Bridge	0	0	0	12	3
Pit stop	0	0	0	7	8
Climb	0	0	3	11	1
Ski	0	0	2	9	4
Totals	0	0	5	39	16

Table 4.3: The table shows the results of the pilot study performed with 15 participants. Each column contains the number of people who gave that particular rating to each video, with the numbers totalled in the bottom row.

The results of the pilot study show that the super-resolution algorithm developed for this project has successfully increased the perceptual quality of the four test videos when measured using the HVS of multiple participants. The climb and ski videos were the only two videos to receive ratings of 3 in low numbers. This correlates well with the results of the SSIM assessment which rated the super-resolved output of these two videos as poorer than the other two. The PSNR however indicated that the bridge video was one of the two poorest videos, however this was not really reflected in the pilot study.

The evaluation of perceptual increases in video quality is not easy to gauge. I feel that the methods I have used, in particular the SSIM and small pilot study, have given good results. As hoped before the project development began the results of the SSIM index, PSNR and pilot study all correlated fairly well indicating that the quantitative measures match to some degree the evaluation done using the HVS.

## 5. Conclusion

### 5.1 Evaluation Criteria

The evaluation criteria to measure the success of the super-resolution algorithm were that the image quality assessment methods should give a higher similarity rating for the super-resolved output than the input and sharpened input videos when compared with the ground-truth high-resolution. The majority of test cases succeeded in this. The only video which failed to achieve this was the climb video which I believe is due the constant rotation of the scene. The other, qualitative evaluation criteria was that when a survey was undertaken the majority of people rated the super-resolved videos as high quality. This would mean that the participants scored the videos with ratings of 3 and above. No rating was received less than 3 and so I would rate this qualitative evaluation as successful. One thing to note about the survey however is that the sample group was very small, for this reason I described the survey as a “pilot study” and further research would need to be carried out to verify these results. Also the results were likely to be biased towards high quality as all of the participants were friends of mine and most probably realised the significance of giving poor results. Ultimately all of the evaluation criteria were met and so I feel the super-resolution algorithm is of high quality.

### 5.2 Objectives and Requirements

The objectives and requirements set out before undertaking the project have been completed to some degree, the minimum requirement, which was to super-resolve a single image, has also been completed. The following list details how objectives were met, and any changes to objectives that were made:

- Capture a number of low-resolution videos and high-resolution images of the same scene

This objective was met, although the quality of images and videos captured on the Nokia N95 camera phone was poor. Due to the poor focus of the camera the quality of high-resolution photos was not as high as I imagined and did not sufficiently exceed the quality of the video, therefore the images and videos captured from this device could not be used as input for super-resolution. However I did also use a number of videos downloaded from the internet from which high-resolution, scene-specific images and low-resolution

videos were created.

- Train a database from the high-resolution images and use it to super-resolve an input video

This task was successfully completed and evaluation has shown that the video super-resolution algorithm successfully improves the perceived and statistical quality of low-resolution video.

- Minimum requirement: Super-resolve a single image

This minimum requirement of the project was completed early on and the algorithm developed formed the foundation upon which to build a video super-resolution algorithm.

- Extension: Super-resolve video and incorporate temporal enhancements

A system was developed to apply the super-resolution algorithm to multiple frames of a video using scene-specific high-resolution training images. Due to the scene-specific nature of these images the quantity of temporal artifacts in the resulting videos were minimal, and barely noticeable to a third party observer. It was decided not to implement any temporal enhancements due to the improvement in quality gained being negligible and the amount of time required being large.

- Extension: Increase computational efficiency

The slowest part of the first iteration of the video super-resolution algorithm was the exhaustive search stage. This was improved by re-structuring the searchable medium-frequency patches into a kd-tree. The kd-tree code itself was implemented in C and called from MATLAB, giving a significant boost in computational efficiency and allowing the algorithm to process multiple video frames in a reasonable time.

### 5.3 Possible Future Developments

There are number of possible future developments that could be implemented to increase the quality of the video super-resolution algorithm developed for this project. These include the following:

- Implement temporal enhancements

Due to a shortage in time towards the end of the implementation phase of the project the temporal enhancements devised by Bishop *et al.* were not implemented. Implementing

this should improve the quality of the results, especially if the scene content of the training images does not closely match the content of the videos.

- Use online sources for training images

Websites such as Flickr could be used to find images related to the scene content contained in input videos. Images on such sites tend to be of very high resolution and scene-specific examples could be found by searching using carefully selected keywords.

- Experiment with a variety of patch sizes and shapes

The patch sizes used were based on previous work and no experimentation was done using different patch sizes. Due to the scene-specific nature of the training images larger patch sizes may improve results. Super-resolution results for the climb video were relatively poor due to the rotating nature of the scene content. Patch shapes other than squares could be used, or the patch database could include examples of rotated patches.

- Distance metrics other than the  $L_2$  norm

Whilst the  $L_2$  norm does well to find NN matches in the patch database it may disregard patches which are perceptually similar to the input patch. Other distance metrics could be experimented with to increase the generality of the patch database.

# Bibliography

- [1] Y. Altunbasak, A. J. Patti, and R. M. Mersereau. Super-resolution still and video reconstruction from mpeg-coded video. *Circuits and Systems for Video Technology, IEEE Transactions on*, 12(4):217–226, 2002.
- [2] I. Bégin and F. P. Ferrie. Comparison of super-resolution algorithms using image quality measures. In *Computer and Robot Vision, 2006. The 3rd Canadian Conference on*, pages 72–72, 2006.
- [3] S. Baker and T. Kanade. Limits on super-resolution and how to break them. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(9):1167–1183, 2002.
- [4] C. Bishop, A. Blake, and B. Marthi. Super-resolution enhancement of video, 2003.
- [5] D. Capel and A. Zisserman. Computer vision applied to super resolution, 2003.
- [6] Microsoft Corporation. Wmv hd content showcase. <http://www.microsoft.com/windows/windowsmedia/musicandvideo/hdvideo/contentshowcase.aspx>, 2008.
- [7] A. M. Davis. Operational prototyping: a new development approach. *Software, IEEE*, 9(5):70–78, 1992.
- [8] A. M. Eskicioglu and P. S. Fisher. Image quality measures and their performance. *Communications, IEEE Transactions on*, 43(12):2959–2965, 1995.
- [9] David J. Field. What is the goal of sensory coding? *Neural Comput.*, 6(4):559–601, July 1994.
- [10] William T. Freeman, Thouis R. Jones, and Egon C. Pasztor. Example-based super-resolution. *IEEE Comput. Graph. Appl.*, 22(2):56–65, March 2002.
- [11] William T. Freeman, Egon C. Pasztor, and Owen T. Carmichael. Learning low-level vision. *International Journal of Computer Vision*, 40(1):25–47, 2000.

- [12] Bernd Girod. What's wrong with mean-squared error? pages 207–220, 1993.
- [13] Aaron Hertzmann, Charles E. Jacobs, Nuria Oliver, Brian Curless, and David H. Salesin. Image analogies. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 327–340, New York, NY, USA, 2001. ACM Press.
- [14] R. Keys. Cubic convolution interpolation for digital image processing. *Acoustics, Speech, and Signal Processing [see also IEEE Transactions on Signal Processing]*, *IEEE Transactions on*, 29(6):1153–1160, 1981.
- [15] T. Komatsu, K. Aizawa, T. Igarashi, and T. Saito. Signal-processing based method for acquiring very high resolution images with multiple cameras and its theoretical analysis. *Communications, Speech and Vision, IEE Proceedings I*, 140(1):19–24, 1993.
- [16] S. Kondo, H. Amirshahi, T. Toma, and T. Aoki. Example-based super-resolution using internet photo collection. 2007.
- [17] D. Kong, M. Han, W. Xu, H. Tao, and Y. H. Gong. Video super-resolution with scene-specific priors. In *British Machine Vision Conference*, pages 549–558, 2006.
- [18] David G. Lowe. Object recognition from local scale-invariant features. In *Proc. of the International Conference on Computer Vision ICCV, Corfu*, pages 1150–1157, 1999.
- [19] Sung C. Park, Min K. Park, and Moon G. Kang. Super-resolution image reconstruction: a technical overview. *Signal Processing Magazine, IEEE*, 20(3):21–36, 2003.
- [20] L C Pickup, S J Roberts, and A Zisserman. Optimizing and learning for super-resolution. In *In Proceedings of the British Machine Vision Conference*, pages 439–448, 2006.
- [21] Lyndsey Pickup, Stephen Roberts, and Andrew Zisserman. A sampled texture prior for image super-resolution.
- [22] R. Sarpeshkar, T. Delbruck, and C. A. Mead. White noise in mos transistors and resistors. *Circuits and Devices Magazine, IEEE*, 9(6):23–29, 1993.
- [23] W. F. Schreiber. *Fundamentals of electronic imaging systems*. 1986.
- [24] E. Shechtman, Y. Caspi, and M. Irani. Space-time super-resolution. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(4):531–545, 2005.

- [25] E. P. Simoncelli. Statistical models for images: compression, restoration and synthesis. In *Signals, Systems & Computers, 1997. Conference Record of the Thirty-First Asilomar Conference on*, volume 1, pages 673–678 vol.1, 1997.
- [26] Zhou Wang, A. C. Bovik, and Ligang Lu. Why is image quality assessment so difficult? volume 4, pages IV–3313–IV–3316 vol.4, 2002.
- [27] Zhou Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *Image Processing, IEEE Transactions on*, 13(4):600–612, 2004.
- [28] Yueting Zhuang, Jian Zhang, and Fei Wu. Hallucinating faces: Lph super-resolution and neighbor reconstruction for residue compensation. *Pattern Recogn.*, 40(11):3178–3194, November 2007.

# Appendix A

---

Overall I feel that the experience of carrying out my final year project has been very valuable. I made sure that I picked a project in an area of computer science that interested me which I think is probably the most important factor to bear in mind when making a choice on which project to take up. Although initially I was worried about finding a project which I was passionate about, it became clear once the project list was published that this would not be a problem. There is such a wide selection of possible titles that there is usually something there for everyone. Once the list of available titles was out I had the opposite problem in that I found too many of them to be of high interest.

To solve this issue I made a shortlist of the most interesting project titles over a number of different fields with a number of different supervisors. I then booked an appointment and visited each supervisor to discuss each project in detail. It became much clearer from these meetings which combination of supervisor and project would be best. One of the most important and helpful people throughout the final year project is your project supervisor. They have a wealth of experience in the particular area of your project and can also offer highly critical, but highly useful advice. I feel it is therefore highly important to build a good relationship with your supervisor and attend every one of your scheduled meetings. Some of my fellow students did not value their supervisors input as highly and spent days or even weeks struggling with problems that their supervisors would have known the answer to instantly.

Although it is mentioned in almost every project reflection from previous years it is well worth stating again that the most important aspect of the project is time management and starting early. Thankfully due to the interest I had in my project I spent a lot of time throughout the year

developing and refining the code that I had created. However probably the most important part of the project is the write-up which I believe I should have started much earlier. Once I began writing up the project it led to a much deeper understanding of previous work, but due to it being relatively late in the project schedule I was unable to put this deeper understanding back into my system. I feel that if I had done the write-up as I was developing the project code I could have achieved much greater things.

Throughout the project I used a website called CiteULike (<http://www.citeulike.org>) to manage my library of research papers. This website made recording which papers I had read, and papers I needed to read very simple. It is a one-click process to add papers to my personal library from catalogues such as IEEE Xplore and the ACM Digital Library. Once added the papers can be rated to indicate the priority with which they need to be read. It also has a tagging system which makes similar papers easy to find, for example I had tags to separate papers into image only super-resolution and video super-resolution. Possibly the biggest benefit of using a site such as CiteULike comes when writing the final report. The site offers a function to export your entire library to a BibTeX file, which can then be easily included in the LaTeX write-up.

Although at first I found the use of LaTeX frustrating at times, especially when it came to graphical tasks such as positioning figures or drawing tables, overall I found it worth-while to use. Using a LaTeX editor such as LEd made the write-up a lot easier as it includes toolbar buttons for many commonly used symbols, along with buttons to compile to PDF and then open the output in a reader. I would highly recommend that anyone who writes their final report up in LaTeX uses a custom editor such as this, rather than a simple text editor.

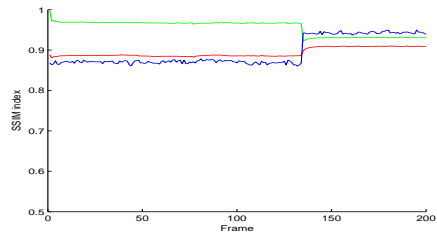
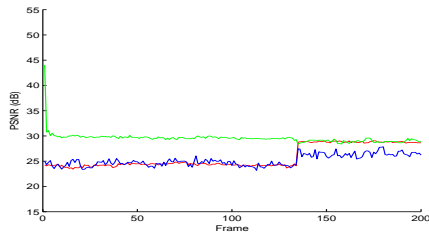
Setting the time aside to undertake the final year project was a challenge. As well as final year course work and exams I also have a part time technical support job which I work for 20 hours a week. This has meant I have had very little personal time during the final year where I can truly relax and so I would recommend that future students schedule in a break or two when planning their final year projects. In my project schedule I failed to allocate time for preparation of the project demonstration. Putting together a presentation for this took around two weeks, during which time I was supposed to be doing my project write-up.

Ultimately however the project has enabled me to apply the skills and knowledge which I have developed through university to a challenging problem. I have learnt the importance of beginning work early and working as hard as possible for an extended period of time. I would recommend that all future students read previous projects and pay particular attention to the project reflections of both good, and not so good, reports. As long as the authors have been

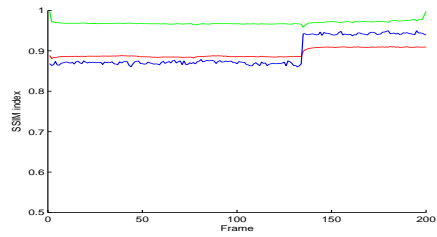
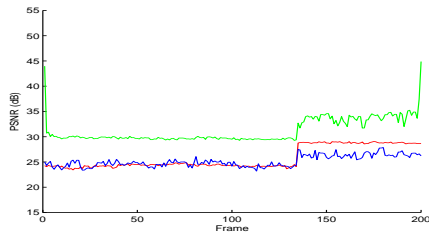
honest it will become clear that although it is possible to get through university leaving a lot of work till the last minute, this is not possible with the final year project. The project, and my experience of the final year, has matured my attitude to academic work, an attitude which I intend to carry on into post-graduate research.

# Appendix B - Results

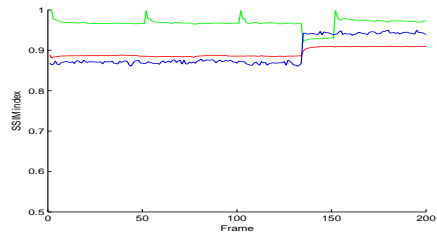
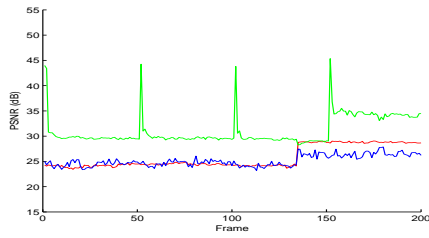
---



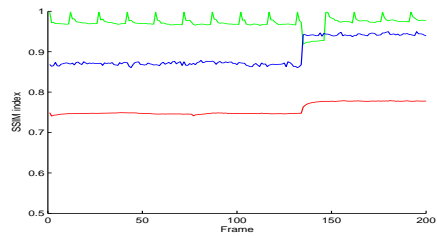
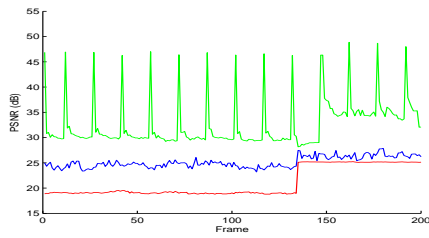
(a) PSNR using training image at first frame only (b) SSIM using training image at first frame only



(c) PSNR using training image at first and last frame (d) SSIM using training image at first and last frame

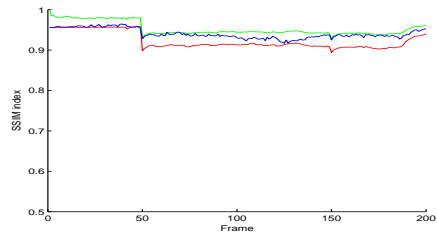
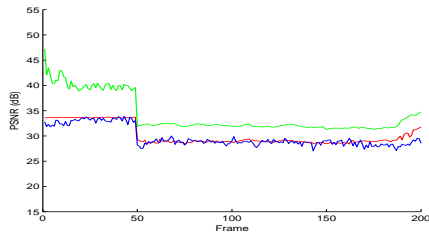


(e) PSNR using training image at every 50<sup>th</sup> frame (f) SSIM using training image at every 50<sup>th</sup> frame

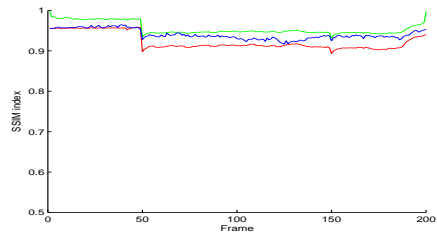
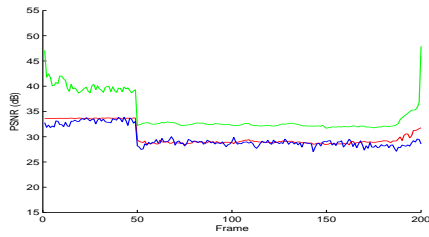


(g) PSNR using training image at every 15<sup>th</sup> frame (h) SSIM using training image at every 15<sup>th</sup> frame

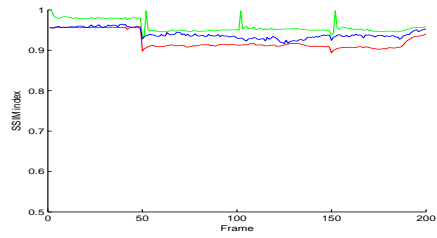
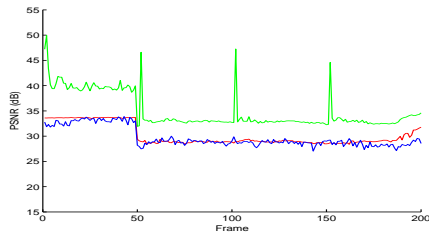
Figure 5.1: PSNR and SSIM graphs for the bridge video with high-resolution training frames used at different intervals. The green line shows the output quality, the blue shows sharpened input quality and the red shows low-resolution input quality, all in comparison to the ground-truth high-resolution video.



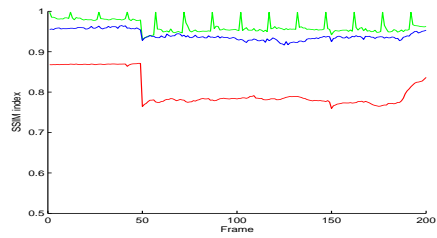
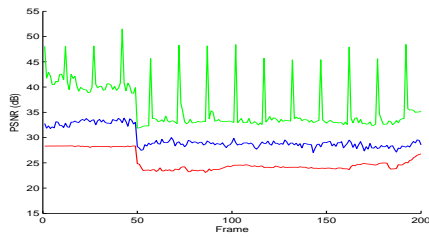
(a) PSNR using training image at first frame only (b) SSIM using training image at first frame only



(c) PSNR using training image at first and last frame (d) SSIM using training image at first and last frame

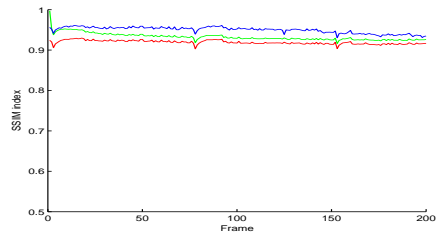
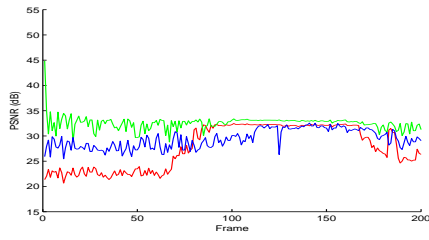


(e) PSNR using training image at every 50<sup>th</sup> frame (f) SSIM using training image at every 50<sup>th</sup> frame

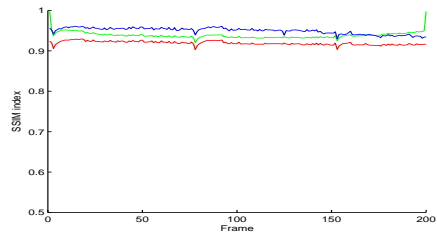
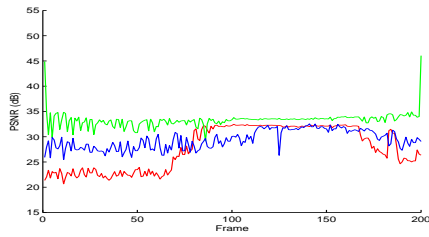


(g) PSNR using training image at every 15<sup>th</sup> frame (h) SSIM using training image at every 15<sup>th</sup> frame

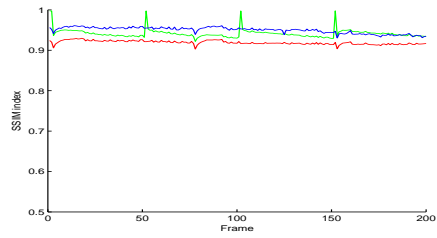
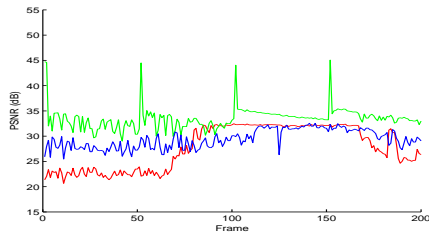
Figure 5.2: PSNR and SSIM graphs for the pit video with high-resolution training frames used at different intervals. The green line shows the output quality, the blue shows sharpened input quality and the red shows low-resolution input quality, all in comparison to the ground-truth high-resolution video.



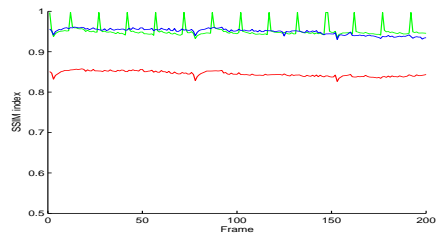
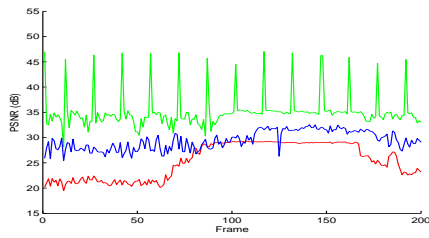
(a) PSNR using training image at first frame only (b) SSIM using training image at first frame only



(c) PSNR using training image at first and last frame (d) SSIM using training image at first and last frame

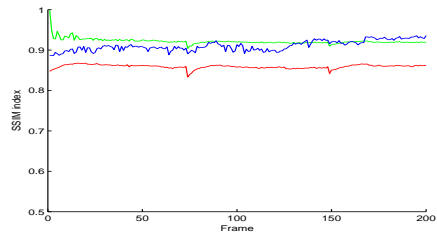
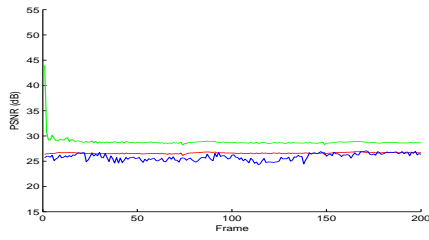


(e) PSNR using training image at every 50<sup>th</sup> frame (f) SSIM using training image at every 50<sup>th</sup> frame

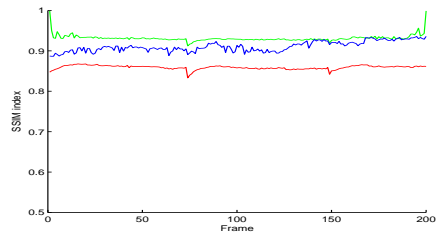
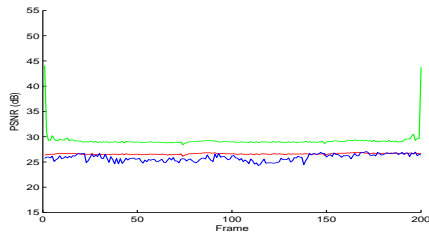


(g) PSNR using training image at every 15<sup>th</sup> frame (h) SSIM using training image at every 15<sup>th</sup> frame

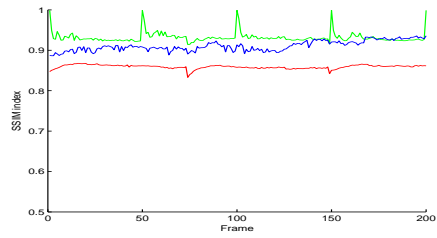
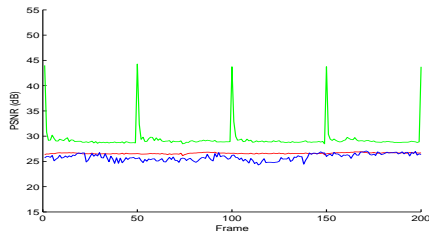
Figure 5.3: PSNR and SSIM graphs for the climb video with high-resolution training frames used at different intervals. The green line shows the output quality, the blue shows sharpened input quality and the red shows low-resolution input quality, all in comparison to the ground-truth high-resolution video.



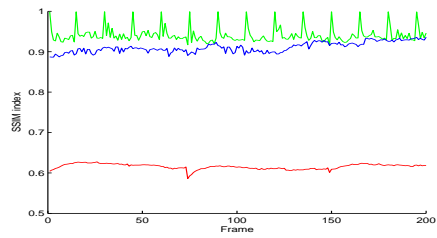
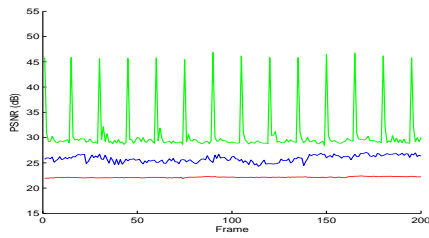
(a) PSNR using training image at first frame only (b) SSIM using training image at first frame only



(c) PSNR using training image at first and last frame (d) SSIM using training image at first and last frame



(e) PSNR using training image at every 50<sup>th</sup> frame (f) SSIM using training image at every 50<sup>th</sup> frame



(g) PSNR using training image at every 15<sup>th</sup> frame (h) SSIM using training image at every 15<sup>th</sup> frame

Figure 5.4: PSNR and SSIM graphs for the ski video with high-resolution training frames used at different intervals. The green line shows the output quality, the blue shows sharpened input quality and the red shows low-resolution input quality, all in comparison to the ground-truth high-resolution video.

# Appendix C - Project Schedule

---

Task	Oct 07				Nov 07				Dec 07				Jan 08				Feb 08				Mar 08				Apr 08					
Week Commencing:	1	8	15	22	29	5	12	19	26	3	10	17	24	31	7	14	21	28	4	11	18	25	3	10	17	24	31	7	14	21
Background Reading	█				█																									
Implement System to Improve Image					█				█																					
Write Mid-Project Report									█																					
Implement Extensions													█				█													
Prepare for Project Report													█				█				█									
Write Project Report																	█				█									
Evaluation																					█									

Figure 5.5: The project schedule from the mid-project report. No changes were required.