Neil Patrick Del Gallego
Edward Patrick Tighe
Oliver Bob Urag

CSC781M Final Project Report
Superresolution Techniques and their Fundamental Limitations

Introduction

This paper tackles the problem of image resolution enhancement. Given a downsampled image, we attempt to upscale the image by a factor of 4 and observe if the enhanced image has achieved better overall composition and natural structures of the images are properly recovered. This paper covers two major techniques of superresolution, the multiple-image SR and the single-image SR. An experimental approach to single-image SR using an Android device is also discussed on this paper.

The multiple-image SR is a technique wherein several low resolution images are combined to form a higher resolution image. The first step is to align these images such as computing the offset of pixels from one another, which is called the image registration. Once the image registration is complete, the fusion step attempts to recover the high resolution image by factoring the constraints and modelling the image formation process.

The single-image SR technique only attempts to consider as little as one image for attempting to create a higher resolution image. Techniques involve slowly resizing the image through iterations while imposing sharpening or deconvolution operators per iteration in order to bring back the natural structures of the image that is otherwise lost through distortion via interpolation. Another technique is to identify common pixel patches (5x5) that should also occur on its higher image resolution counterpart.

However, while numerous work has already been done, the results of the algorithms are mixed. There are certain parts of an image that is difficult to recover [1]. We will attempt to discuss these limitations alongside the techniques mentioned in detail.

Supperresolution Techniques

Related work on superresolution(SR) techniques are discussed in detail here. There are many techniques that have been proposed by numerous researchers that aims to solve the SR problem. We can group this into two major categories, the multiple-image SR and the single-image SR technique.

Multiple-Image SR

Multiple-Image SR technique involves having a set of low resolution images as input for the higher resolution image to be processed. An image resolution is mainly affected by the physical camera sensor. The goal of multiple-image SR is that increasing the sampling rate could potentially yield a better higher resolution image. These means that the camera should obtain more samples of a scene representing them as sequences of displaced images [2].

These image sequences are then combined and aligned using transformations that considers the geometric displacements of the images as well as its photometric components (such as lighting, changes in color, contrast, etc.) to yield a common reference frame. Thus, the first step is to properly “register” the images to create a common base frame. Once this process has been performed, a formulation of an SR estimator is considered that attempts to model the SR image by reversing the steps caused by image degradation caused by the image formation process. Important details that are present in the set of low resolution images are estimated and propagated to the higher resolution image [3]. We shall discuss the first step, **image registration**, followed by the **SR estimation** in detail.

- **Image Registration**

  The first key component of a multi-image SR technique is the image registration process. Given a set of low resolution images, attempt to combine them to form a common reference frame using transformations. The overall accuracy of this process directly affects the outcome of a multi-image SR algorithm. This problem, according to [3], can be stated as follows: given N different images of a scene, for each image point in k view ($k \in N$), find the image point in $k + 1$ view which has the same properties of k (meaning it corresponds to the same point in the scene).

There are three kinds of registration process, the **geometric** registration type, **feature-based** registration type and the **photometric** registration type. The **geometric** registration considers displacements of images in its coordinates found and defined by the planar homography. The planar homography is dicussed in [3]. We shall briefly describe the geometric registration as follows: for each x in image k where $k \in N$ (N is the number of images of a scene), x' in $k+1$, x'' in $k+2$, and so on should correspond to point $\boldsymbol{X}$ found in the world when we attempt to map it into a plane.

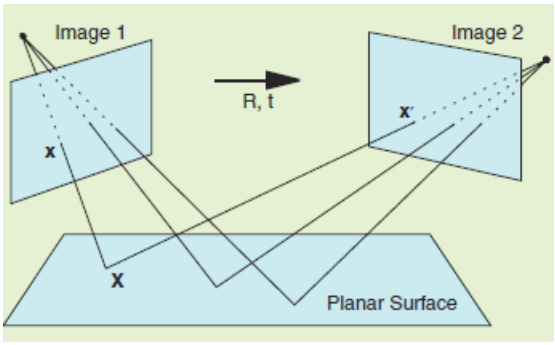

Figure 1: How images should correspond to a certain point in a planar surface as described in [3]

Feature-based registration appears to be most accurate according to [3] as it has the ability to cope to widely disparate views and has excellent robustness to illumation changes. Furthermore, feature-based registration allows easier derivation of an SR estimator (using maximum likelihood or ML). In feature-based registration, interest points (or corners) are detected using an algorithm such as the Harris feature detector

Photometric registration refers to the procedure of considering photometric transformations between images. Such transformation include changes in illumination, intensity variations and changes in color composition. In this case, using a photometric transformation uses affine transformation under the constraints imposed by the contrast and brightness of the image, or the RGB channel [3]

- **SR Estimation**

Once a set of low-resolution images are properly aligned using the common registration techniques discussed in the previous set, the challenge would be to properly invert the image formation process to generate the SR image. The details present in the SR image should also be present across all the low-resolution images. This means that each low-resolution image corresponds directly to the SR image modeled.

We shall model the super resolution process mathematically as discussed in [4]. Given N low resolution images, $\{I_L^k\}_{k=1}^N$, of size L_1 X L_2. Find a high resolution image I_H of size H_1 x H_2 with

$H_1 > L_1$ and $H_2 > L_2$ which minimizes the cost function:

$$E(I_H) = \sum_{k=1}^{N} \left\| P_k(I_H) - I_L^{(k)} \right\| \qquad (1)$$

where $P_k(I_H)$ is the projection of I_H onto the coordinate system. I_L represents a low resolution image. P_K is modeled by four linear transformations that affects the high resolution image, namely the motion, camera blur, down sampling and additive noises. We can use a matrix-vector notation for the following:

$$I_L^k = D_k B_k W_k I_H + e_k \qquad (2)$$

where D_K is a down-sampling matrix, B_k is a blurring matrix, W_k is a warping matrix and e_k is a noise vector. This simply means that a low-resolution image is affected by four factors caused by generating these images from the camera sensor. We should **inverse** this process to yield an HR image. The inversion of this image process is referred to as superresolution.

Figure 2: Inversion of the image formation process is called superresolution

For simplicity of this discussion, we shall only discuss one method for SR estimation. We refer to this as the ML or **maximum likelihood** estimation. Given such estimate, the resulting SR image when projected into the LR images via the image formation process, the difference between the actual and the "predicted" observations should be minimal.

This technique (although not specifically mentioned) is also similar in the technique discussed in [2]. This estimation technique is performed in an iterative manner. Given an initial SR image (which can be formed using bicubic interpolation or other interpolation methods), a set of low resolution images are determined. These set of LR images are compared to the original LR images entered. If these LR images are sufficiently equivalent (by measuring an error function), then we can say that the SR image is the "best" SR image to be returned as an output. Otherwise, we continue to refine the SR image until we minimize the error function.

For most multiple-image SR techniques, the error function is computed as follows [2]:

$$e^{(n)} = \sqrt{\sum_{k} \sum_{(x,y)} (g_k(x,y) - g_k^{(n)}(x,y))^2}. \qquad (3)$$

The difference in images ($g_k - g_k(n)$) are computed and used to improve the "guess" of the SR image. Succeeding iterations should minimize the error function. Once a sufficient amount of the error function has been reached, or the error function has converged, then the iteration stops and outputs the final SR image. Variations for minimizing the error function exists on various works, but the goal is similar.

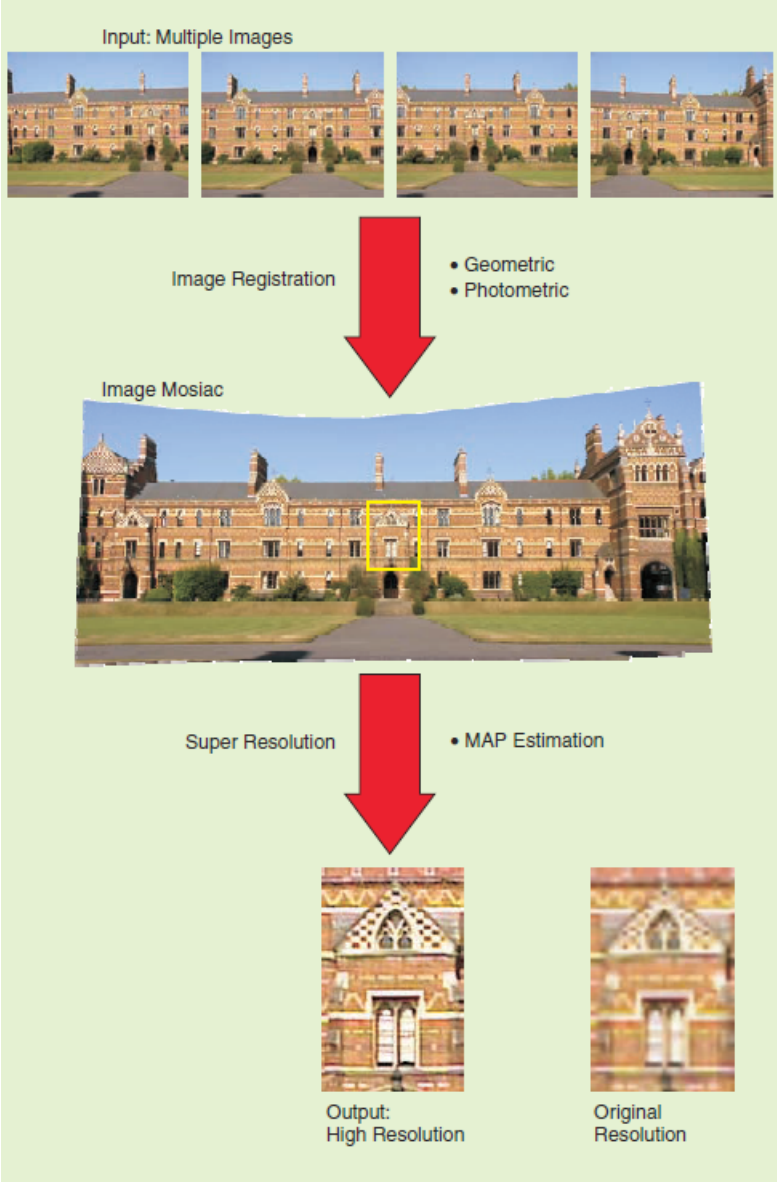

Figure 3: The general multi-image SR process as described by [3]

Figure 3 shows the general multi-image SR process as described in detail in the previous sections. An image mosaic is generated from a set of low input resolution images accurately aligned using one of the three common registration types (or a mixture of these), geometric registration, feature-based registration and photometric registration. The image mosaic generates an initial SR image which is further refined by considering its error function as discussed in the previous section.

Figure 4 shows the general error minimization process for multiple-image SR algorithms. Given an initial SR image (we call it the *SR_candidate)*, the set of potential LR images are generated (*LR_set)*. The *LR_set* and the original LR images (*LR_initial_set)* are compared using the error function formula mentioned in the previous section to get a function *e(n)*. If a certain threshold has been reached, *SR_candidate* becomes the final SR image. Otherwise, *SR_candidate* should be refined to minimize the error function *e(n)* until it reaches a certain threshold. A threshold of 0 theoretically means that the LR images back-projected by the *SR_candidate* are exactly similar to the input LR images, but this is generally impossible to be performed.

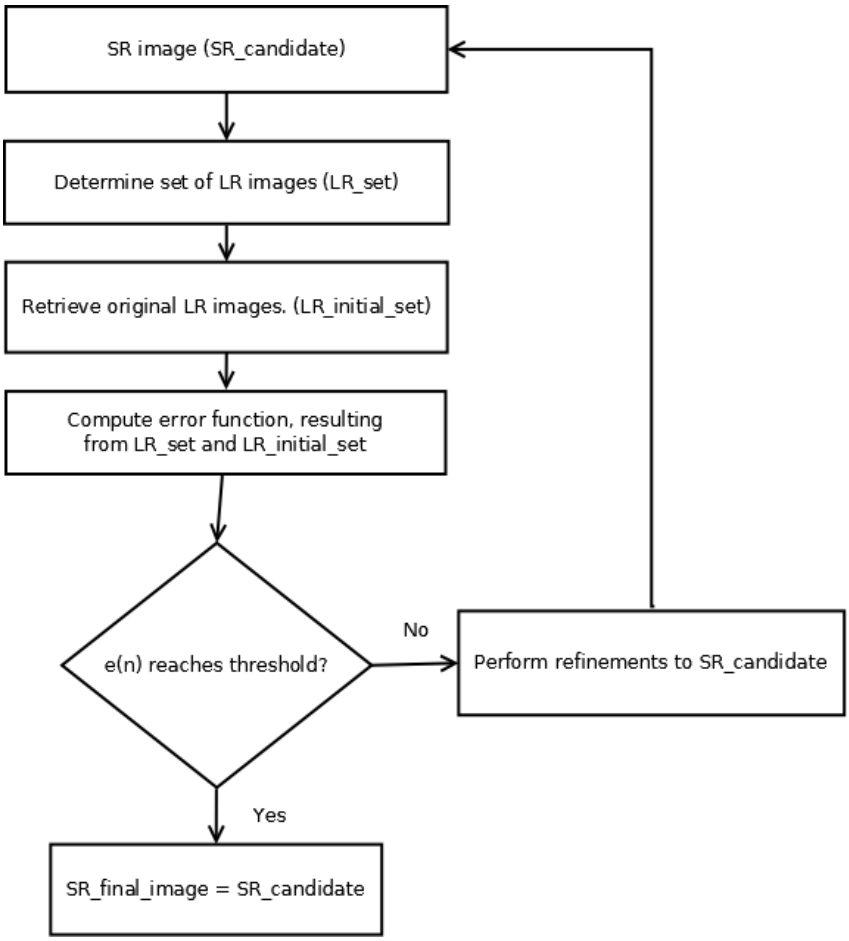

Figure 4: The general error function minimization process

Multiple-Image SR Examples and Limitations

To show proof of concept and idea to readers, this section contains sample images taken from various literature that uses a multi-image SR approach to generate a high-resolution image. We will also discuss our findings and observations regarding these images and how multiple-image SR approaches contain various limitations.

Oppo, a smartphone manufacturer, managed to release a commercial software product that implements a multiple-image SR technique to create 50MP photos from the phone's camera (Oppo Find 7 and Oppo Find 7a). The LR images entered for processing only takes four of the initial input images of 10. The four best shots of image sequences are only considered for the ML estimation. Figure 5 shows a sample image, given a 13MP initial LR image, and the output of 50MP image. Edges of the images are smoother on the output image which proves to be fulfilling. Noises present in the original 13MP are removed from the output image as well. The technique employed by Oppo (which is confidential as of this writing) also has certain limitations. It fails to emphasize the finer details in an image that seems to be affected by the overall luminance of the input images. Let us consider the image in Figure 6.

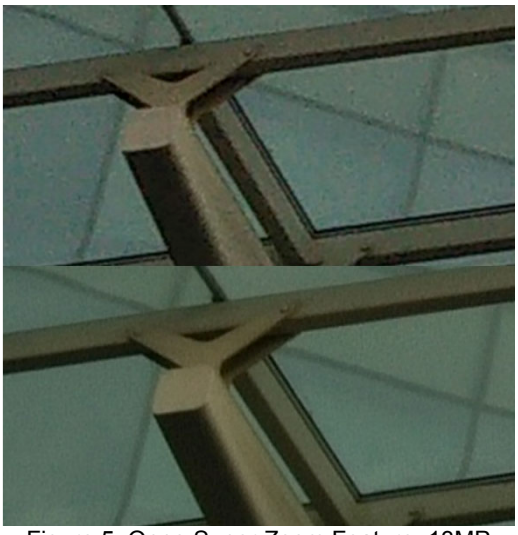

Figure 5: Oppo Super Zoom Feature. 13MP above. 50MP below.

Figure 6: Oppo Super Zoom Feature. 13MP above. 50MP below. (zoomed to a certain area)

Notice the lack of details when we attempt to zoom in on a particular area in Figure 6. Ideally, a high-resolution image should contain finer details but it was not really present in the output SR image. We also compare an output SR image to an SR image generated by a simple bicubic operation in Figure 7. Notice that while the multiple-image SR technique by Oppo is fairly better than an interpolation technique, it also fails to recover the finer details present in the texts of the image.

Figure 7: Oppo Super Zoom feature. 50MP left vs 50MP right. Left photo uses the multiple-image SR technique. Right photo uses bicubic interpolation.

Consider the image in Figure 8 taken from [3]. Notice that as the magnification factor increases, details of the image are not really recovered. It has been explained in [1] that the performance of multiple-image SR techniques get worse as the magnification factor is increased. It shows that some of the reconstruction constraints imposed by the four transformations mentioned in the previous section become ireversible. According to [1], a set of low resolution images to reconstruct an SR image is no longer applicable beyond a point wherein the magnification factor is great (greater than 4 on most related work) as it is now affected by a wide range of factors which is discussed in their paper.

So how can an SR image be improved further? Clearly, the multiple-image SR technique poses a limitation (let us say a scaling factor of 4) for it to be actually effective. Other works regarding SR has shown that it is quite possible to recover an SR image given only one input LR image by attempting to “guess” the missing pixels or to repeatedly deconvolve a given image.

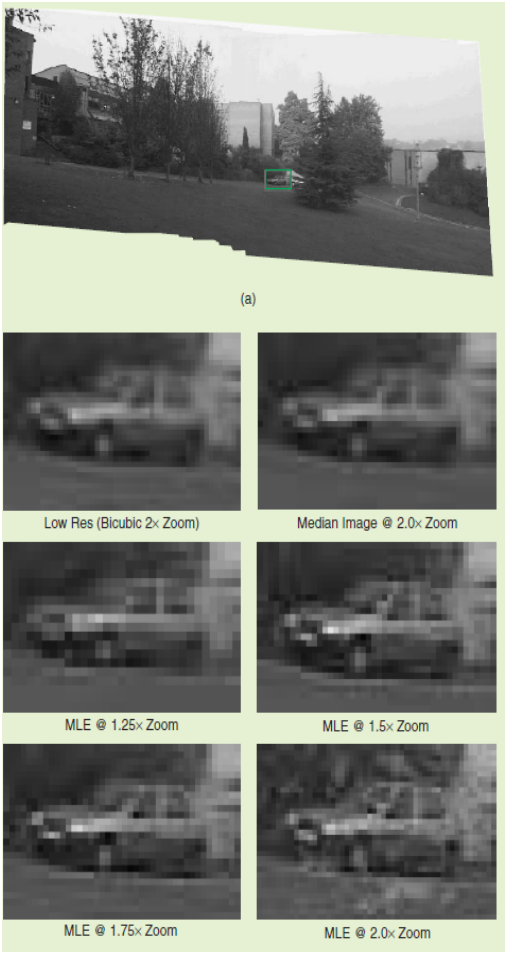

Figure 8: Given an image mosaic, an increasing magnification factor makes it more difficult to recover structures.

Single-Image SR

Other works that attempt to solve the superresolution problem attempts to reconstruct the high-resolution image by only considering one input LR image. The philosophy behind this is that only using one input LR image makes it computationally faster and more resource efficient than a multiple-image SR technique. This concept has been proven to be fast in [5]. It can also be easily expanded into processing videos, as videos can be dissected into several frame sequences, each frame can be processed using a single-image SR technique as demonstrated in [5].

Related work has exploited the approach of redundant patches that occur as an image is scaled further. These patches of pixels are reused and propagated to the higher resolution image iteratively. This concept is used in [6] where source patches are identified in an LR image, and then “copied” to an increased scale of the LR image.

Another method is to emphasize the edges of an SR image. It has been observed that in a multiple-image SR approach, the edges are inevitably unrecovered from the LR images. These result in lost of detail. The approach discussed in [7] attempts to mitigate this problem. We will discuss each of the techniques in detail under the single-image SR approach.

- **LR image to SR image through repeated deconvolution and convolution**

  The work in [5] proposed a method wherein an initial input image *L* is initially upsampled and continuously improved using repeated deconvolution and convolution until the image process model is satisfied. The upsampling loop is feedback-controlled wherein they attempt to minimize the reconstruction error which is quite similar to the error minimization process discussed in the multiple-image SR section. The input image *L* is initially upsampled (let's call it H'_0), wherein it is iteratively improved in the feedback-controlled loops, which progressively reduces image bluriness and recovers natural structures.

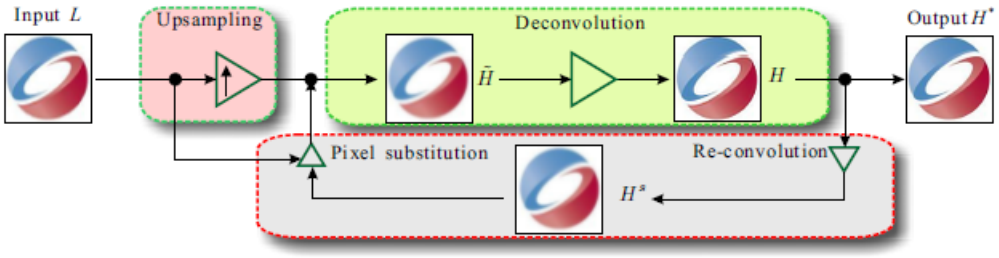

Figure 9: The iterative feedback-controlled framework in [5]

Figure 9 shows the system framework implemented in [5]. The input image *L* is initially upsampled to H'0 using a bicubic interpolation method. H'_0 is iteratively improved over the feedback-control loops represented by the green and grey boxes. This consists of three steps: deconvolution, re-convolution and pixel substitution.

The important step in their approach is to compute if *H'* satisfies the properties of *L*. Once this has been proven, the iteration stops and outputs *H'* as the final high-resolution image. These steps are detailed as follows.

Deconvolution

They describe H' as a linearly filtered high-resolution image that follows this equation:

$$\tilde{H} = f \otimes H, \quad L = \tilde{H} \downarrow^d \tag{4}$$

where *f* denote the discreted PSF (point spread function), operated by a convolution matrix from *H* and *d* is a decimating (subsampling) operator. If this process can be inverted, that is given *L*, we attempt to uncover the possible causes of subsampling and remove those effects through deconvolution, we therefore yield *H* that is further refined using a PSF to output *H'* as the high-resolution image. However, attempting to recover H' from L is a severely ill-posed problem as only one pixel is kept for every *n* = *d x d* pixels in the high-resolution image .

As recovering an image caused by an unknown downsampling operator is difficult, they introduced a non-blind deconvolution method described in [8] to estimate for the deconvolved image. As the problem is ill-posed, they add an additional factor to make the problem well-posed. They have discovered that natural image gradients generally follow a heavy-detailed distribution. They use this factor to approximate the gradient density distribution which is symbolized as $\Phi(x)$. By using such term into the deconvolution process, a formula has been proposed to minimize the energy function:

$$E(H) \propto \|f \otimes H - \tilde{H}\|_2^2 + \lambda_1(\|\Phi(\partial_x H)\|_1 + \|\Phi(\partial_y H)\|_1) \tag{5}$$

where @xH and @yH respectively denote the values of the x- and y-direction gradients, and λ_1 is a weight. Figure 10 shows an example of their deconvolution operator.

Figure 10: The result of the deconvolution process. Left image is the input image, middle image blurred using a gaussian kernel, rightmost image is the result after the deconvolution operation.

Reconvolution

As stated in [5], if we attempt to reconstruct a filtered H'k from the deconvolution operator, should H'k be similar to the initial input image prior to deconvolution, then the convolution model is satisfied. Otherwise, H'k needs to be further refined by reinforcing information obtained from L. Reobtaining information from L is

performed using pixel substitution method.

Pixel substitution

In the pixel decimation process, a high-resolution image *H* is subsampled by a factor of *d* to get *L*, meaning that only one pixel is kept in *L* for every *d x d* samples. Based from this principle, the pixels in image *H* are replaced by those from *L* in order to reinforce and preserve image information from the given LR image. The process is simple. If the upsampling factor is *n,* therefore, replace pixels (n x i + 1, n x j + 1) in the high resolution image H with those in image *L* whose pixels are in (i, j) for each i and j. For example, if n = 3 (image *L* is rescaled to 3x its size), about one-ninth of the image pixels are replaced. Figure 11 shows an illustration of the pixel substitution process.

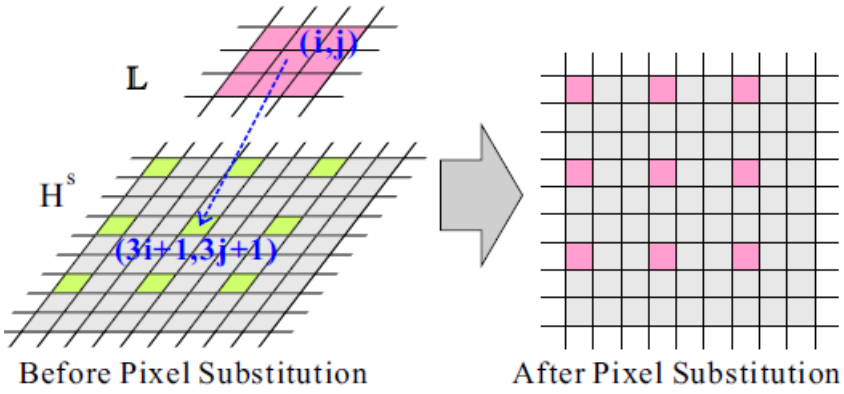

Figure 11: Pixel substitution process

The reconstruction error from the iterative feedback-controlled framework discussed shows that it is significantly reduced on its fourth iteration. More details of it are discussed in their work [5].

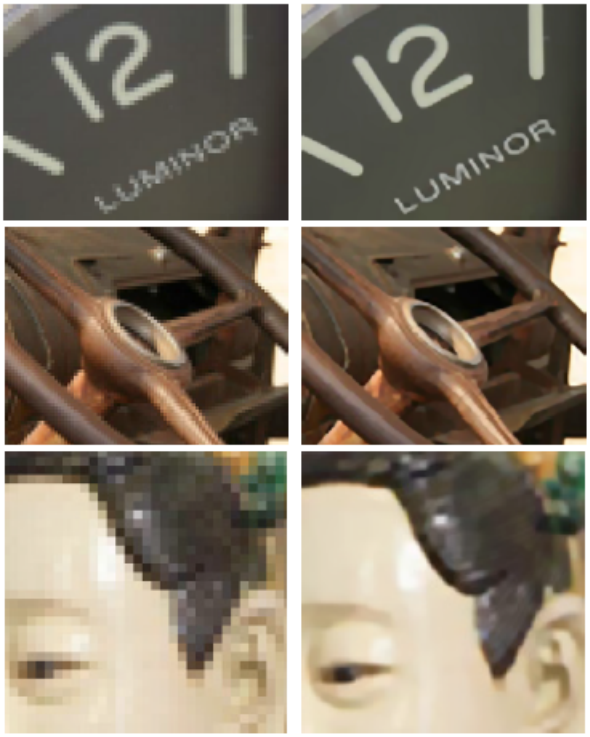

Figure 12: Results from [5] upsampled by 4 .Left image is the input image *L*. Right image is the output image.

- **LR to SR image through patch observation**

A recent work has emerged wherein given a low-resolution image, attempt to recover a high-resolution image by observing redundant patches occuring in the low resolution image and apply them to a high-resolution image. Natural images tend to have reduntant patches across different positions. Factor in the factors caused by diminishing details toward the horizon, these patches should simply occur similarly on a high-resolution image. Let us consider the image in Figure 13.

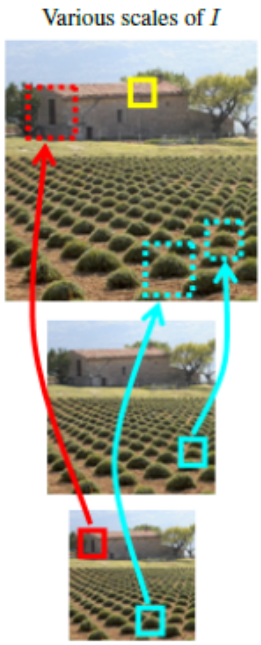

Figure 13: Source patches in I are found in different locationsand in other image scales of I (solid-marked squares).The high-res corresponding arent patches (dashed-markedsquares) provide an indication of what the (unknown) high-resparents of the source patches might look like [6].

According to [6], natural images tend to contain repetitive visual content, usually small (5 x 5) image patches redundantly occur many times inside the image within the same scale, as well as across different scales. They have studies and quantified that such observations always occur on natural images (taken from a scenery). Any patch that recurs in another scale should appear **as is** and is unaffected by blurring or downsampling. Finding similar patches on an LR image, we can predict its high resolution parent from LR. These patches serve as a database of information to construct the HR image.

They have deduced the observation as follows: “Let *p* be a pixel in *L*, and *P* be its surrounding patch (e.g., 55), then there exist multiple similar patches *P1, …, Pk* in *L* (inevitably, at subpixel shifts). These patches can be treated as if taken from *k* different low-resolution images of the same high resolution “scene”, thus inducing k times more linear constraints.”

Considering this observation, they have proposed a simple algorithm: “For each pixel in *L* find its *k* nearest patch neighbors in the same image *L* (e.g., using an Approximate Nearest Neighbor algorithm and use k=9) and compute their subpixel alignment (at *1/s* pixel shifts, where s is the scale factor.) Assuming sufficient neighbors are found, this process results in a determined set of linear equations on the unknown pixel values in *H*. Globally scale each equation by its reliability (determined by its patch similarity score), and solve the linear set of equations to obtain *H*.”

This algorithm is illustrated in Figure 14.

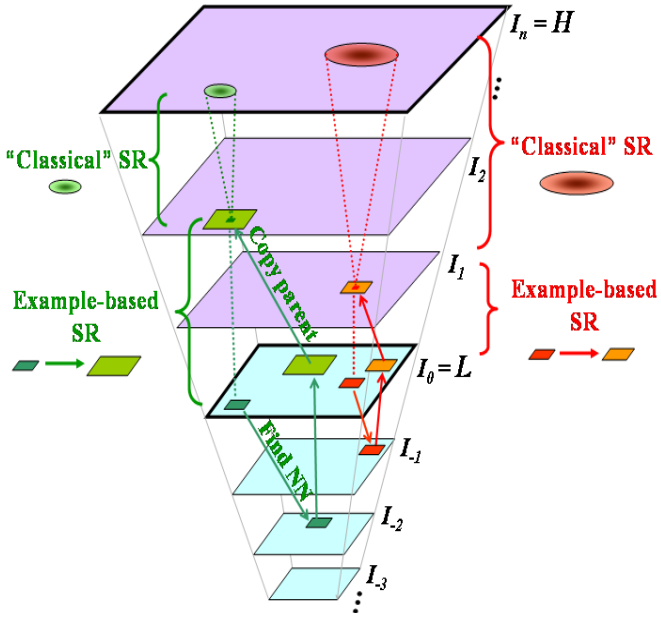

Figure 14: The algorithm proposed wherein patches are propagated to its higher scaled images.

Notice the iterative process described in [6]. An input LR image *L* is "slowly" scaled by a constant scale factor (in their work, they have set this to 1.25) until the target scale factor has been reached. For each scaling, attempt to predict patches that occur in the previous image and apply them to the higher scaled image. Figure 15 shows the result of this technique.

Figure 15: The result of the proposed SR technique. Using scale of 4. From left to right. Input image. Bicubic interpolation. Proposed SR technique. Ground truth image.

However, Figure 16 shows that finer details, specifically the edges are not properly recovered in the HR image. Only the color composition are maintained. This is a good observable limitation by this proposed SR technique. How can we attempt to recover edges from an input image *L*? This has been the principle behind the next technique discussed in the next section.

Figure 16: Result from the proposed SR technique. Finer details are not recovered, only the color composition are maintained which is the characteristic imposed by patch prediction.

- **LR to SR image via Edge Statistics**

  A new method for upsampling images that is capable of generating sharp edges with reduced input-resolution grid-related artifacts are discussed in [7]. They have observed that real-world images exhibit pixel differences that depend on their distance from an edge, such as images from indoor and outdoor scenery.

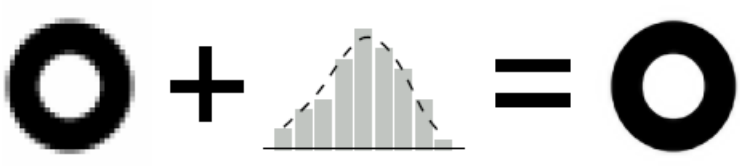

Figure 17: Imposing edge statistics on the LR image results an HR image with good edges

The derivation of their features based on their prediction is discussed in their paper where spatial derivatives and gradient norm are evaluated for every pixel. They have also provided a function that calculates the ray in the gradient direction. We won't cover the derivation (they refer to it as Edge-Frame Continuity Moduli) here. The features they attempt to extract for each pixel *x* describe the shape of the nearest edges passing by *x*. The features are stored in a table.

Using the Edge-Frame Continuity Moduli (EFCM), input images are upsampled using prior knowledge stored on the EFCM table. The steps are as follows:

1. Let *H_initial* be an upsampled input image *L* using bicubic interpolation (or other interpolation method).
2. Extract edge-related features *r,d,m* and *s* from *H_initial*. These features are discussed in detail in their work [7].
3. Compute for the resulting distribution *P(I)* also known as the Gibbs distribution. Details in [7].
4. Extract image *L* intensity values and compare to *P(I)* to reduce false predictions of edges.

5. Use the modified *P(I)* to preserve edges (contouring or highlighting using Laplacian operator) from *H_initial.*
6. Let *H_final* be the output HR image. *H_final* gets *H_initial*.

Figure 18: Results from [7] . While edges are perfectly preserved and recovered, notice the lack of texture.

In Figure 18, artifacts which are inevitably introduced from other SR techniques are more minimal here in this method. However, lack of texture and fine details are present from the output high resolution image. Cartoon images make good candidates for this method as seen in the dog image. The outlines remain sharp while also enhancing some of the pixels.

In the next section, we will discuss an experimental approach by combining some of the important steps discussed previously. We will focus on the single-image SR paradigm and discuss how it was implemented as an application (specifically on an Android platform). The results are shown and future work for this application.

Applied SR in an Android Platform

In this section, a single-image SR approach is discussed and how it was implemented in an Android platform. An application (APK) package is created that utilizes the device camera (front or back) for capturing images and using OpenCV, an open-source computer vision library to process the images. The results are preliminary, with improvements mentioned as we discuss this issue. Recommendations are provided and design issues are tackled.

The approach was to take some of the steps mentioned (in the single-image SR technique) previously and apply it as part of the SR operation. We considered the “gradual scaling” method introduced by [6] wherein an LR image is slowly upsampled to the desired result. For each upsample step, we attempted to preserve information from the previous scaled image. This idea was based on [5] where according to their work, pixel substitution would preserve the information from the lower scaled image. Thus, we imposed a pixel substitution method after every upsample step. Highlighting the edges also becomes an importance as stated in [7]. Thus, we use a Laplacian operator (after the pixel substitution step) to sharpen the edges.

Technical Specifications

This section mentions technical details in order to understand how the SR approach is implemented on an Android device. The minimum Android version for the application is 4.0 (Ice Cream Sandwich). It is capable of working with any Android device as long as the minimum version is met and the device has at least one camera. The application utilizes OpenCV, an open-source computer vision library. It is required for the user to have this installed on their Android device. Otherwise, it is still safe to run the SR application but the user will be prompted to install the openCV for them to proceed.

The input resolution size of the LR image depends on the device camera default size. It has been set that the resolution range should be somewhere between 1024x768 to 1600x1200. But for the sake of clarity, samples included in this document have an input size of **1600x1200**. The HR image is a factor of 4, thus having 6400x4800 as final size.

Implementation Details

This section discusses the implementation per step. The general framework is as follows: given an input image *L*, generate a higher-resolution image *H* through iterative upsampling. For each step, we performed pixel substitution in order to propagate information from its lower resolution image. Once the desired upsample factor has been reached, we performed a weighted laplacian sharpening filter on the image in order to attempt to emphasize the edges. In this section, we will also define some variables used and our recommended values for them based from our experiments.

Figure 19 shows the general system framework of the application. The system is pipelined into three stages: preprocessing, processing and postprocessing stage. Each stage has one or more processing steps performed as seen in the figure. We shall go to each stage and each step in detail.

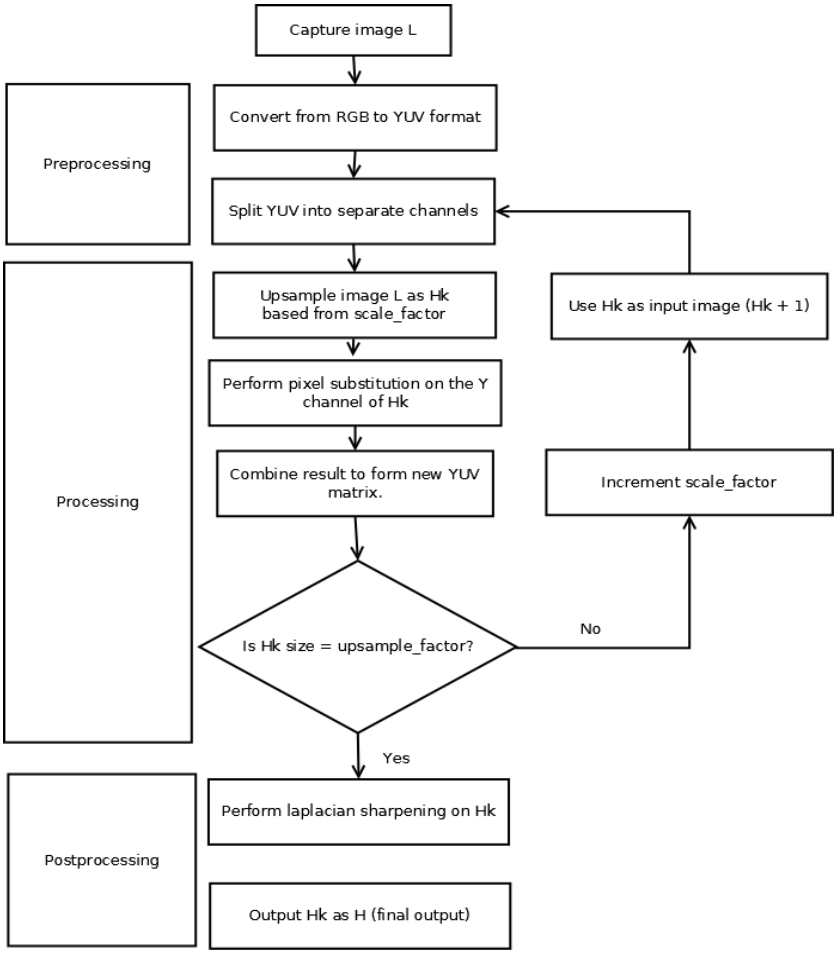

Figure 19: The general system framework of the proposed SR application.

Converting Input Image to YUV format

As observed from the works in [5], [6] and [7], they all implemented their techniques by only considering the intensity values of the image (Y channel) and perform simple bicubic interpolation on the other channels. This proves to be faster than traditionally processing the image under the RGB format. Because of this, the input image is first converted to YUV format using the function provided by openCV.

Splitting YUV into separate channels

The resulting YUV values are separated and stored on a matrix. This makes it possible for the pixel substitution method to only process the Y channel.

Upsampling image iteratively

For all our test cases, our *HR* image has an upsample factor of 4. An input image *L* is upsampled gradually until the upsample factor is reached. We introduce a variable *scale_factor* that slowly increases until the upsample factor is reached. For all our test cases, the *scale_factor* is set to 0.25. That means that an image is scaled to 1.25x its size, to 1.50x, 1.75 and so on, until the upsample factor (which is 4) is reached. This method is similar to the method proposed in [6]. The upsampling technique uses a bicubic interpolation method that outputs an image of specified size.

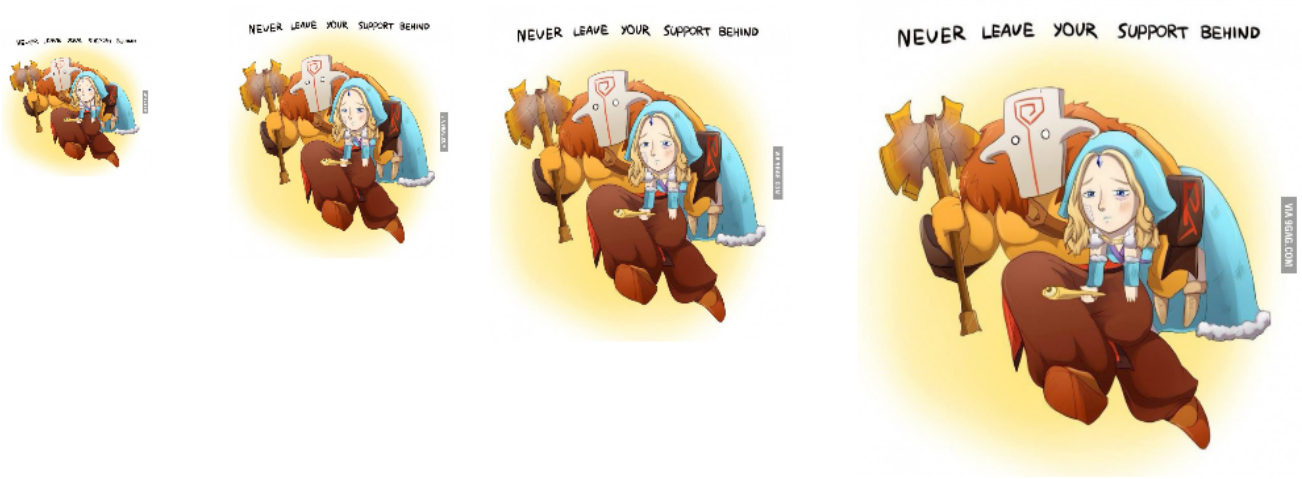

Figure 20: How iterative upsampling is being done on a cartoon image.

Performing pixel substitution

After the upsampling operation, the initial image *Hk* has pixels substituted from its previous scaled image (from *Hk – 1 or L* if *Hk = 0)* in order to reinforce and preserve information that would otherwise be lost through repeated upsampling. This method is very similar to [5]. Only the Y channel is processed and entered for the pixel substitution operation, meaning that only the intensity values are reinforced to the image.

Combining result into YUV matrix

The processed Y channel matrix from the pixel substitution operator is merged with the UV matrix. This represents the YUV values of the modified image *Hk* that is used as a candidate for making it a final output of image *H.* If the desired upsample factor is reached, then the system stops and outputs the final image. Otherwise, the *scale_factor* is incremented and the new image is fetched by the system to be processed again.

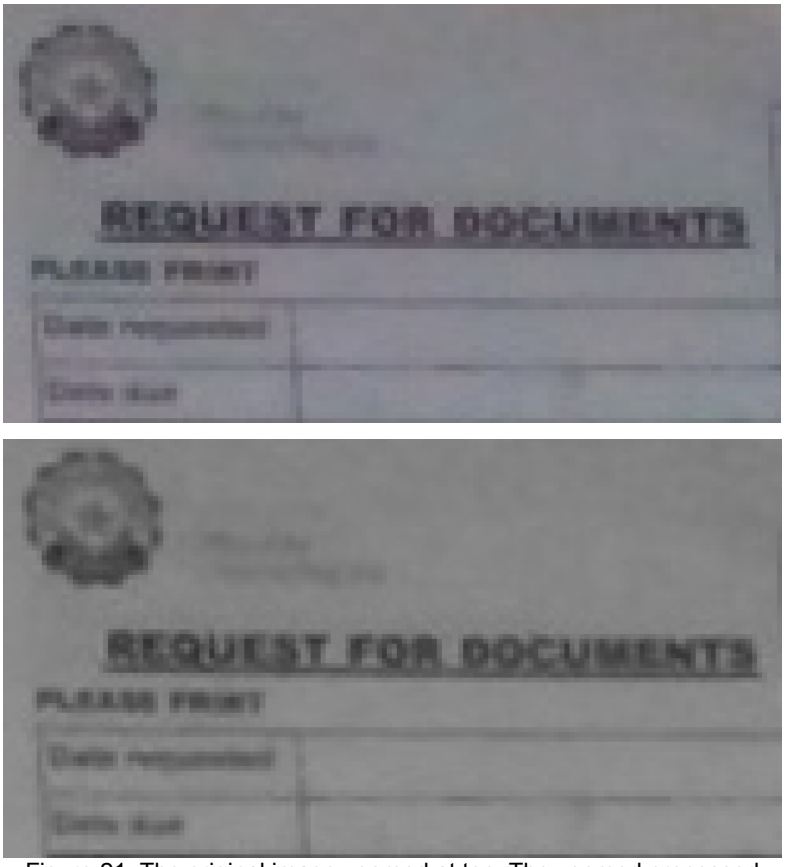

Figure 21: The original image zoomed at top. The zoomed processed image upsampled (by 1.25x) and substituted pixels. The processed image only shows the intensity value.

Sharpening the Image using Laplacian Operator

Once the desired upsample factor has been reached, we attempt to preserve the edges using a laplacian sharpening filter. Given an image *Hk*, it is assumed that it is still in its YUV format. We convert *Hk* into RGB before being entered into the sharpening operator. The sharpening operator needs to convert a given image into grayscale from its RGB format. A laplacian operator is performed between the grayscale image and the RGB image in order to outline the edges.

The result is in its grayscale form, meaning only 1 channel is available. The values are then converted to RGB format for it to be compatible with the original image *Hk* if we attempt to add them together. We introduce weighted values such that the *Hk* has a higher weight while the laplacian filtered image has a negative weight. In our test, the value for the *Hk* is 1.0 while the value for the laplacian filtered image is *-0.25.* This makes it acceptable for edges to be preserved without over-emphasizing it. We utilize the *addWeighted* function that combines two matrices provided by OpenCV. Figure 22 shows the original image zoomed and its laplacian filtered counterpart.

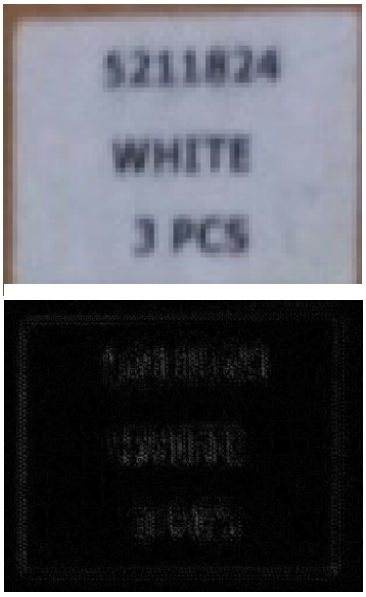

Figure 22: Original zoomed image at top. The laplacian filtered image at bottom.

Results and Observations

This section discusses the results of the proposed SR framework. Using only a single LR image, we attempted to reconstruct its HR image counterpart based from principles and techniques discussed in the previous sections.

Let us consider an image wherein the weighted value from merging with the laplacian filtered image is completely removed. This means that the sum of both matrices are computed evenly without bias. Based from our findings, this caused an image to have very sharp edges and over-emphasis of corners.

Figure 23 and Figure 24 shows an example of an unbiased addition of the *Hk* candidate image and its laplacian-filtered counterpart. Notice that while the barcode appears readable, there's too much edge definition around it which we considered unneeded for the image. Thus, we have introduced a weighted addition as discussed previously.

Using a weighted addition scheme, notice how everything appears balanced in Figure 25 and Figure 26.

Figure 23: The processed image downscaled for illustration only.

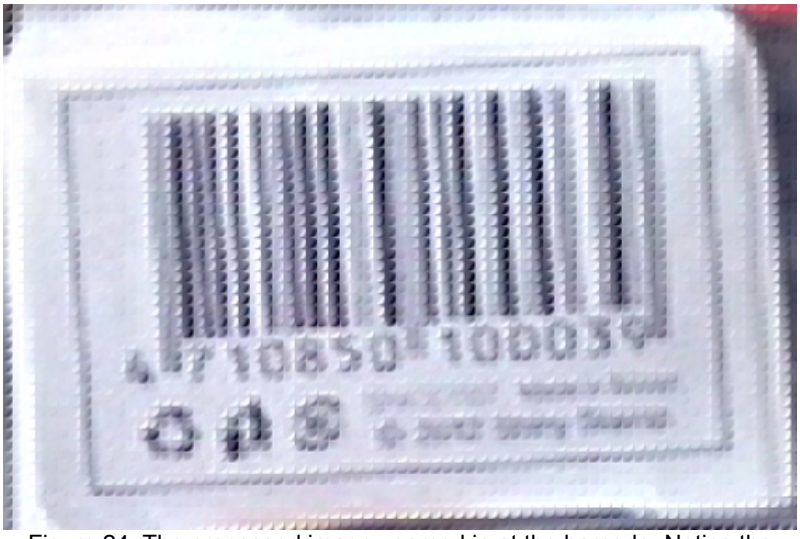

Figure 24: The processed image zoomed in at the barcode. Notice the very sharp edge contours.

Figure 25: Original image using weighted addition.

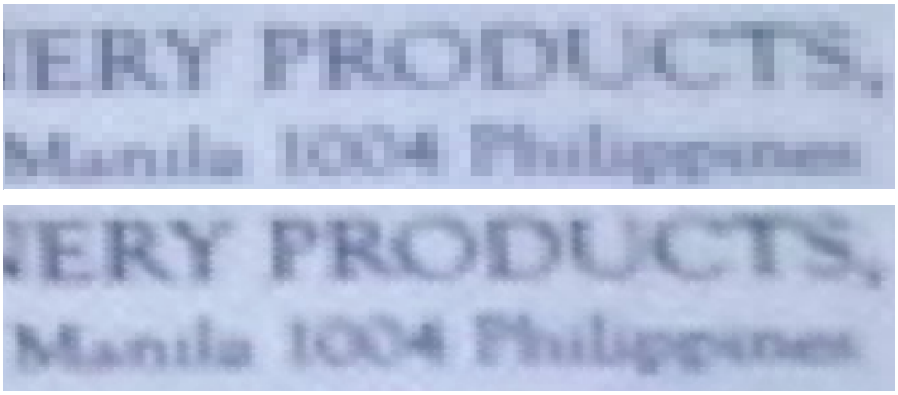

Figure 26: Zoomed in at text using weighted addition. Original image at top. Processed image at bottom. Although still blurry, we can already read what is written in the image with a naked eye.

Consider another example in Figure 27 and Figure 28. Our SR approach tends to capture some details of a certain text about 2 meters to 3 meters away from the camera. However, only the headers are readable and the intricate details on the smaller texts are barely readable. These are inevitably lost and cannot properly recover it by our single-image SR technique.

Figure 27: Processed image scaled down for illustration.

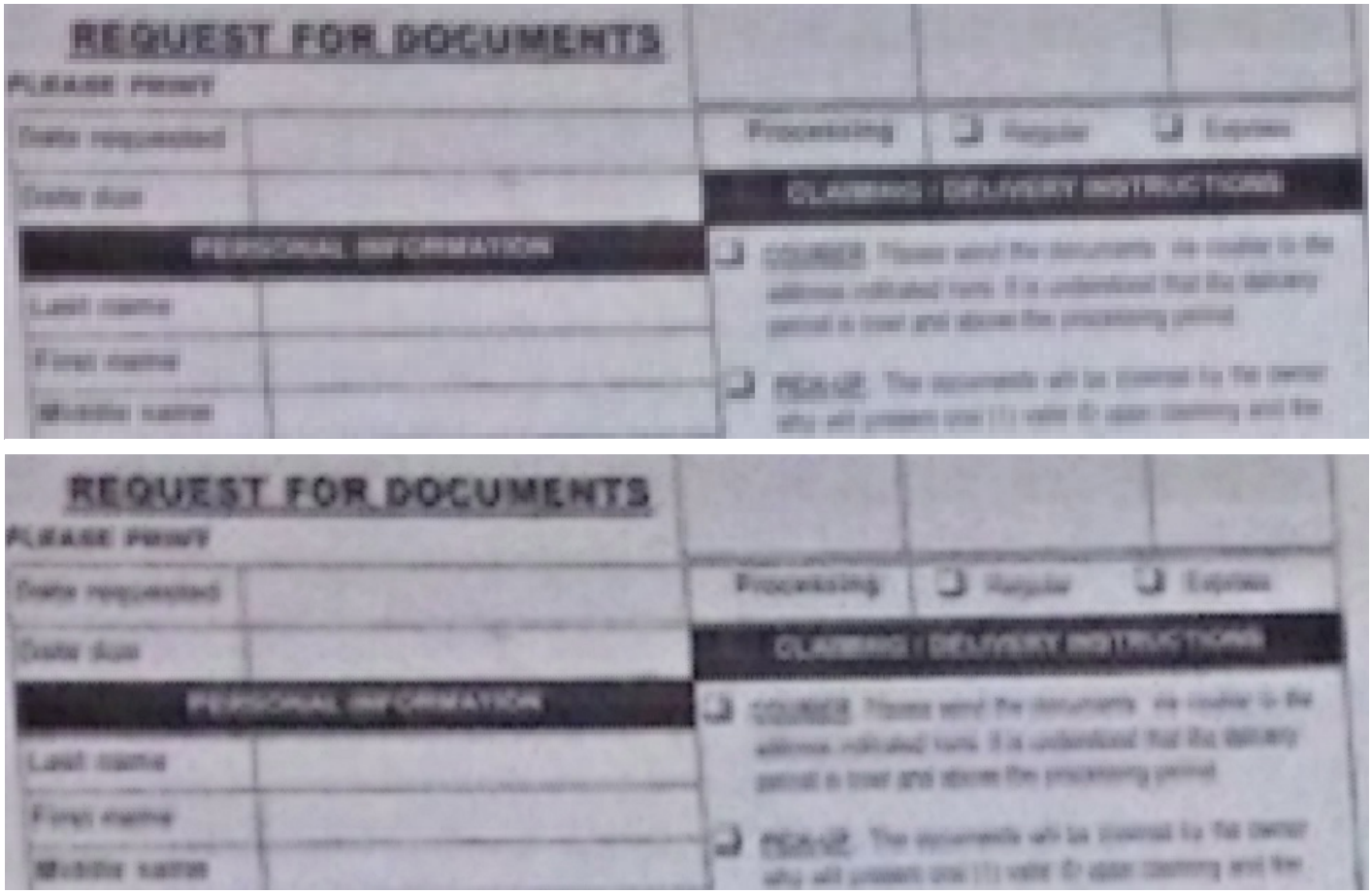

Figure 28: Original image zoomed in at text. Processed image zoomed in at text. Notice that the header has been recovered. Smaller font styles can no longer be retrieved.


Figure 29: Sample outdoor image processed using the proposed SR approach.

Let us consider an outdoor scenery in Figure 29. The proposed SR approach managed to process it properly but zooming in to the vehicles in Figure 30 and Figure 31, we can notice that finer details were no longer recovered but the edges and contours of the vehicle are smoother than the original.

Figure 30: Zoomed in at vehicle. The right image is finer than its original counterpart in left.

Figure 31: The jeepney appears finer as well but specific details were no longer recovered.

Conclusion and Future Work

In this paper, we have discussed and identified two common SR techniques currently being tackled by many researchers. SR techniques are divided into two, the multiple-image SR approach and the single-image SR approach. We have also demonstrated an experimental procedure based from the principles employed by single-image SR techniques discussed in this paper. It is observed that while we manage to smoothen out the image, finer details are not recovered. This is also evident in the examples discussed as limitations on other works provided. Most SR techniques discussed here in this paper struggle to recover the lost details when we attempt to scale an image by 4.

According to [1], most SR algorithms proposed have mixed results but commonly they struggle at a scale factor of 4. Going beyond that factor introduces a lot of distortion to images. Using a set of low resolution images for creating an HR image no longer helps beyond that factor. A single-image SR technique makes it more difficult to “guess” the missing pixels as the downsampling operator and blurring factors are severely ill-posed problems discussed in [5] and [8]. Only assumptions were made to make it well-posed.

How can we go beyond this limit? As stated in [1], “Suppose, however, that the input images contain text. Moreover, suppose that it is possible to perform optical character recognition (OCR) and recognize the text. If the font can also be determined, it would then be easy to be perform super-resolution. The text could be reproduced at any resolution by simply rendering it from the recognized text and the definition of the font.” Based from this principle, they have proposed the **hallucination** algorithm that attempts to recognize these features through learning from a database. This method is also called **recognition-based** superresolution technique. We shall leave this for the readers to explore.

The SR techniques discussed here shows that while it manage to recover natural structures, finer details from most images are not really recovered.

References

[1] Baker, Kanade (2000). Limits on Superresolution and How to Break Them. IEEE Transactions on Pattern Analysis and Machine Intelligence. IEEE Conference Proceedings. Carnegie Mellon University
[2] Irani and Peleg (1990). Super Resolution From Image Sequences. IEEE Transactions 1990. IEEE. The Hebrew University of Jerusalem
[3] Capel and Zisserman (2003). Computer Vision Applied to Super Resolution. IEEE Signal Processing Magazine. .
[4] Mitzel, Pock, Schoenemann, and Cremers (2008). Video Super Resolution usingDuality Based TV-L1 Optical Flow. IEEE. IEEE Conference Proceedings. Institute for Computer Graphics and Vision, University of Bonn, Germany
[5] Shan, Li, Jia and Tang (2008). Fast Image/Video Upsampling. ACM Transactions on Graphics. ACM. The Chinese University of Hong Kong, The Hong Kong University of Science and Technology
[6] Bagon, Glasner and Irani (2009). Super-Resolution from a Single Image. IEEE Conference Proceedings. IEEE. The Weizmann Institute of Science, Dept. of Computer Science and Applied Mathematics
[7] Fattal (2007). Image Upsampling via Imposed Edge Statistics. ACM Transactions on Graphics. . University of California, Berkeley
[8] Yuan, Sun, Quan and Shum (2008). Progressive Inter-scale and Intra-scale Non-blind Image Deconvolution. ACM Transactions on Graphics. ACM. The Hong Kong University of Science and Technology, Microsoft Research Asia