Restoration of Faded Slides II

Geoff Daniell
gjd@lionhouse.plus.com

April 18, 2016

1 Overview

Some years ago I created a `gimp` plug-in for the automatic improvement of colour slides or prints that had deteriorated with age. While this worked quite well it was obviously not the best that could be done. Its strength was that it usually got close to a good restoration; this is actually the difficult part of doing the operation by hand since there is a tendency to go in circles. But in many cases a further correction to the colour balance produced a better result. It was also possible, with experience, to recognise pictures that had been through the process, which proves that the result is different to the original. This document describes a new attack on the problem.

The previous document 'Restoration of Faded Slides' discusses the process of colour photography and possible models for the deterioration. The conclusion there was that it should be possible to reverse the deterioration using just the `gimp 'levels'` command if the parameters are suitably chosen and ways of determining these values were suggested. I feel that this discussion is still valid but more robust ways of fixing the parameters are needed.

The new approach uses three stages: first a crude restoration that removes any major imbalance between the overall colours. The image is 'restored' using the parameters for this stage and a second colour adjustment is done so that the 'average' colour is some shade of grey. Finally there is an optional stage to increase the saturation of the colours which sometimes appear very 'washed out' when the previous algorithm is used. The original method included correction for the 'side absorptions' which are an inherent defect in colour film. Experience showed that in practice although this correction made a difference it was not necessarily an improvement and it has been removed.

There has also been an important change of strategy in the numerical methods for estimating the parameters. The iterative methods used earlier are replaced by a systematic search for the best values using a minimisation algorithm. This is much more powerful and flexible but had been rejected before because it was anticipated that it would be too slow. This has proved not to be the case.

2 The main restoration calculation

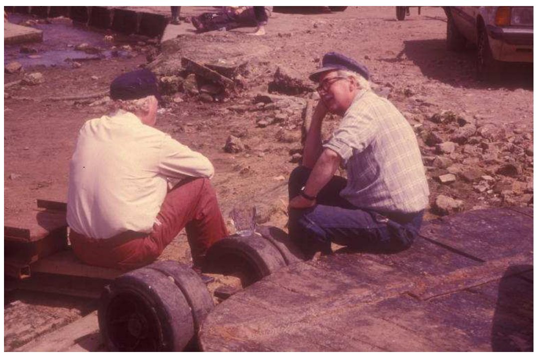
A digital image may contain several million pixels. A program such as `gimp` is designed so that common operations are built into the C source code and are all reasonably fast, however a special calculation required in a plug-in may be so slow that operating on all the pixels is unrealistic. Our plug-in uses two tricks to circumvent this limitation, firstly estimating the restoration parameters is done on a small copy of the image and secondly this copy is converted to `colour indexed` mode using 256 colours. For the purpose of the restoration we do not care *how many* pixels have a particular colour but care only about the colours that occur and we can summarise the colour information by 256 triplets of numbers.

A photograph that has deteriorated may have the quantities of colour dyes reduced, probably by differing amounts. The previous document showed that the fading could be reasonably predicted by the equation

$$C' = 255\left(\frac{C}{255}\right)^{\alpha_c}$$

in which C denotes the original colour and C' the faded colour and α_c is a measure of the loss of coloured dye in channel c. Specifically $\alpha_c < 1$.

Figure 1 (left) is a typical Agfacolor slide scanned after being stored for several years[1]. On the right are the colours making up the image plotted against index number. The low values in the green channel correspond to the overall magenta colour.



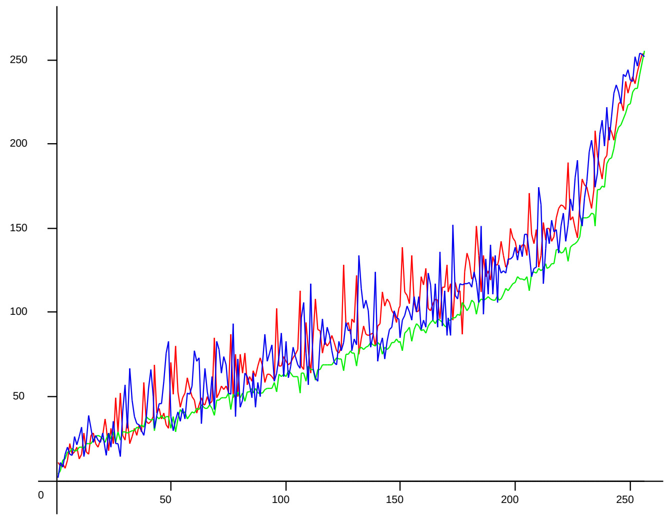


Figure 1: left: An old slide, right: The colours in the image

The earlier algorithm fixed the restoration parameters by moving the *maximum* intensity for each colour to 255 and the moving the *lower end* of

[1]I am grateful to Tim Gossling for supplying this

the curve to an appropriate point. This led to two relatively simple equations but only used information at the ends of the intensity range. The new method uses all the colours to fix the parameters; this is more robust but requires a systematic minimisation in place of simple equations.

It must be remembered that there is no certain way of recognising an image which has deteriorated (attempts to process sunsets will always fail!) however the fact that *all* the green values are low is very suggestive. Reversing this reasoning implies that in a 'good' photograph the numbers for each channel should all lie close to the same curve.

We can also imagine an 'ideal' photograph in which the colours in channel c (red, green or blue) are each of the values 0 to 255 exactly once. After deterioration colour intensity C becomes $C' = 255(C/255)^{\alpha_c}$ and the assignment of optimum colours in the faded image will spread the faded colours uniformly over the values of C' so colour C'_i will be assigned an index $i \simeq C'_i$. This means that the original colours will lie along the curve $C_i \simeq 255(i/255)^{1/\alpha_c}$. This exactly what is seen in Figure 1, the curves resemble power laws with exponent $1/\alpha_c > 1$.

The above concept of an 'ideal' photograph is unnecessarily restrictive, we do not require all intensities of a colour to occur just once but the colours in a 'good' photograph can be expected to span the range 0 to 255 along a smooth curve which will not be too far from a straight line.

We therefore *define* an 'ideal photograph' as one with a plot of optimum colours, similar to figure 1, as a power law with the 'ideal colour' numbered i given by the equation

$$\mathscr{C}_i = 255\left(\frac{i}{255}\right)^{\gamma}.$$

If the plot in figure 1 were derived from an ideal photograph it would be described by this equation. A straight line corresponds to $\gamma = 1$ but there is nothing fundamental about this shape and it is convenient to make γ adjustable.

We can now choose our restoration parameters to make the restored image as close as possible to an 'ideal image'. It must be remembered that the numbers representing the colours have no perceptual significance and all we are imposing is that red, green and blue are regarded as equally important in the picture and have the same distribution of values in the restored image.

The value of γ is a simple way of giving the user a range of possible restorations; a larger value of γ produces a darker restored image. A value about 1.0 usually produces pleasing results.

The first step in the restoration is to choose parameters λ_c and σ_c for each channel so as to minimise

$$\sum_{i=0}^{255}\left[255\,\sigma_c\left(\frac{C_i}{255}\right)^{\lambda_c} - \mathscr{C}_i\right]^2$$

The values of λ and σ are found for each colour separately. An alternative would be to minimise the quantity obtained by adding together the above sums of squares for the three colour channels. Experiments showed that the restorations obtained using these numbers are less good.

This minimisation leads to six numbers λ_r, σ_r, λ_g, σ_g, λ_b, σ_b which permit us to calculate the intensities in the image that best fits the 'ideal' one as

$$C_i' = 255\,\sigma_c \left(\frac{C_i}{255}\right)^{\lambda_c}$$

which maps the values [0, 255] onto [0, 255 σ]. The values of σ produced by the minimisation may be greater than unity, implying colour intensities greater than 255, so some scaling is necessary. This needs some thought.

Scaling the result

In figure 1 the curve of the colours clearly has two sections with the portion for the higher intensity colours having a larger slope. Although the *maximum* intensity is more or less the same in all three channels at lower intensities the green channel is clearly lower and if we *extrapolate* these values we would obtain a much lower intercept with the right hand edge of the graph. This is a quite common situation.

Figure 2 illustrates the issue; on the left are shown some colour index curves, these are purely hypothetical and are not derived from a photograph. The lower set of red, green and blue curves are supposed to be correspond to the unrestored picture. The green and blue intensities are low and approximately equal while the red intensities are higher, but still well below 255 (the magenta line). If this were a real picture it would be dark and with a red cast. The red curve has been deliberately designed with a higher slope at the higher intensities. The graphs on the right are simply an enlarged version of the top right corner of the ones on the left. The upper set of red, green and blue curves corresponds to the raw restored image and some of the values exceed 255 (the magenta line).

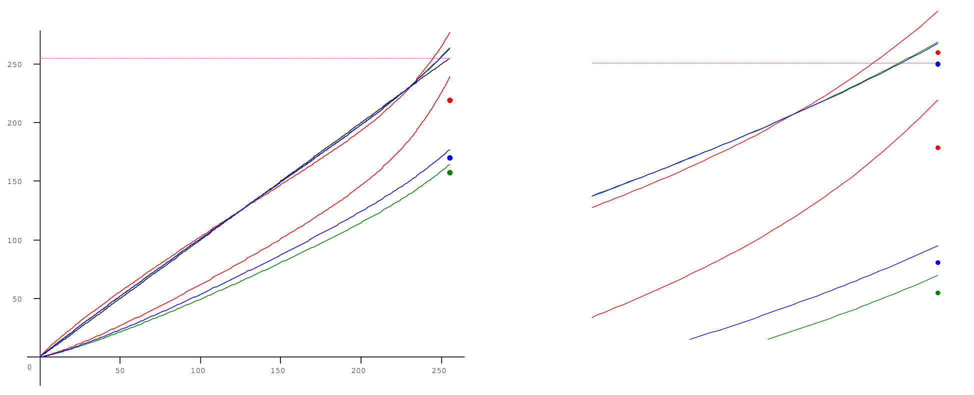


Figure 2: left: Hypothetical Colour Curves, right: Enlarged portion of left figure

We have to decide on an overall scaling factor for the final restored image. The simplest solution is to scale all the channels so that the colour channel with the largest restored value (in this case red) has its largest value reduced to 255. This proves to be not entirely satisfactory.

The region of high slope includes relatively few distinct colours and these are therefore poorly differentiated. Furthermore this part of the curve may correspond to pixels where there has been a severe loss of dye and the intensities should be regarded as uncertain. To prevent this part of the curve causing undue emphasis it may be better to adopt parameters that result in some pixels being 'burnt out', and having their restored intensities capped at 255.

We adopt the following rule: in the raw colour curves let C_{max} denote the maximum intensity in channel C and C_{ext} the result of linearly extrapolating the numbers in channel C. Now define

$$C_{\mathrm{hi}} = \tfrac{1}{2}[C_{\mathrm{max}} + C_{\mathrm{ext}}]$$

These points are shown as dots on the left part of figure 2 and for the green and blue channels are fairly close to the actual maximum values because the green and blue curves are almost straight. The point for the red curve is significantly below the red maximum because of the section of higher slope in the red curve.

The corresponding values in the restored image will be

$$s_c = \sigma_c (C_{\mathrm{hi}}/255)^{\lambda_c}$$

and these points are additionally marked in the right hand part of figure 2. The green and blue points are more-or-less superimposed and close to 255 and the red point is somewhat higher. We adopt the red point (or in general the highest) to scale all the colours.

Let $s_{\mathrm{max}} = \max(s_r,\, s_g,\, s_b)$ and scale the numbers s_r, s_g and s_b by $255/s_{\mathrm{max}}$. We can now construct a `gimp levels` command that computes

$$C' = 255 \frac{s_c}{s_{\mathrm{max}}} \left(\frac{C}{C_{\mathrm{hi}}} \right)^{\lambda_c} \quad = \quad \sigma_c \frac{255}{s_{\mathrm{max}}} \left(\frac{C}{255} \right)^{\lambda_c}$$

Note that C_{hi} only affects the result through a scaling of the overall intensity and not the individual colours.

3 The colour balance stage

The crude restoration described above produces an image close to an 'ideal' one but some overall colour cast may remain. The issue is similar to that of setting the 'white balance' in digital camera images and there has been a lot of work on algorithms for achieving this. The situation in cameras is slightly different to that needed here in that the purpose of 'white balance' is to allow for differing illumination of the scene being photographed. In some cameras it is possible to photograph a white test object under specified lighting, such as bright sunlight, and set the parameters that the camera uses subsequently. The 'colour' of different lighting conditions, such as artificial light, can then be selected on the camera and the colours of the scene 'corrected' for this. In simpler cameras, or more sophisticated ones used in 'automatic' mode, the 'white balance' is achieved by assuming that the 'average colour' of the scene is grey. Since in our case all we have is the image we have to adopt some variant of this approach. Various algorithms are in use with differing degrees of sophistication.

The initial restoration simply imposed equal status to the red, green and blue intensities and did not depend on how colours were perceived by the eye. Now colour perception is important and a defect of the previous algorithm was that it worked directly with the numbers in the red, green and blue channels.

There has been a lot of work on describing colours quantitatively in a way that is close to how they are perceived. We use what is called the (L^*, U^*, V^*) colour space which is a set of three numbers derived from the (R, G, B) values by non-linear transformations. L^* is designed so that is represents 'lightness' and U^* and V^* contain colour information. The important point is that the (L^*, U^*, V^*) colour space has an approximate metric, that is the space can be divided into cells of equal size which correspond to differences of lightness and colour that are just distinguishable to the human eye.

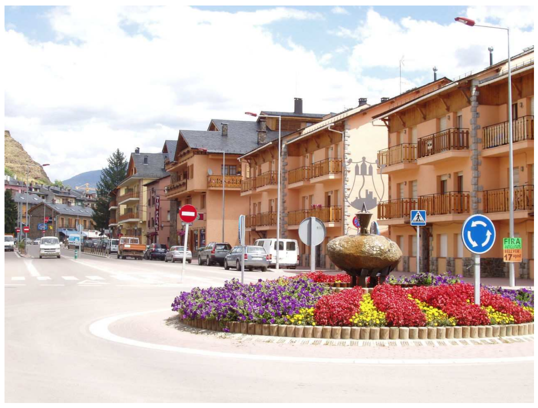
As an illustration figure 3 shows an image, chosen for its bright colours. The right hand picture is the L^* U^* V^* space looking along the L^* axis, so the two dimensions correspond to U^* (horizontal) and V^* (vertical). The axes $U^* = 0$ and $V^* = 0$ are shown and the origin corresponds to a 'colourless' point, that is white or a shade of grey. Each small square is positioned at the correct point in the plane for one of the 256 colours in the index form of the image and is coloured with the corresponding colour. Because one is looking at all the L^* values squares that are close together have the same colour but may differ in intensity.

Because this space approximates to a metric space it makes sense to talk about the 'distance' between two colours as the distance between the corresponding points in the (U^*, V^*) plane.

Our rule for colour balancing is now that the *weighted average* value of U^* and V^*, taken over all the colours, is close to the origin $U^* = 0$, $V^* = 0$. In practice the results look better if the average point is fixed but not exactly at the origin. The weights are chosen so that only colours close to grey are used and saturated colours are ignored.

Again the calculation is done on the colours in the `colour indexed` version of the partially restored image. The adjustments to the image are again made with the `levels` command and a systematic search for the best values is made. Mathematically we minimise

$$\left[\frac{\sum wU^*}{\sum w} - U_{\text{offset}}\right]^2 + \left[\frac{\sum wV^*}{\sum w} - V_{\text{offset}}\right]^2$$



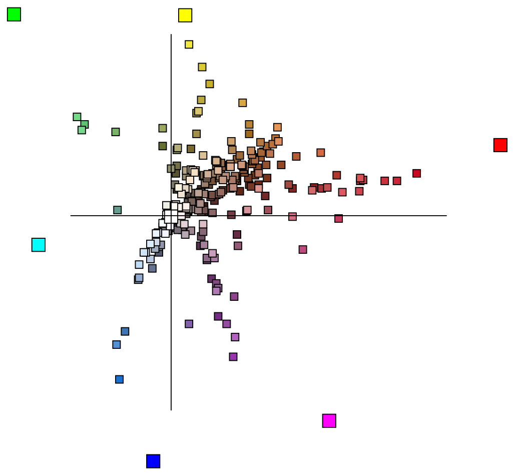


Figure 3: left: An colourful picture, right: The colours in U-V space

where the sums are taken over the colours of the image and the weighting factor w is given by

$$w = \frac{g}{g + U^{*2} + V^{*2}}$$

The constant g fixes the departure from pure grey that is allowed. It is not critical but a value about 50 seems to work well.

The `gimp decompose` command offers conversion to HSV, HSL and LAB colour spaces but not LUV. LAB was designed as an improvement of LUV but it is not a metric space, as is sometimes claimed, and is less suitable for our application. It is possible to code the calculation to LUV using a rather complicated sequence of `gimp` commands but this is not sensible since the dynamic range of the output values is much larger than could be represented using 8 bit integers of `gimp` images. Presumably this is the reason that such a conversion is not built-in, although the conversion to LAB *is* available and suffers from the same problem.

We are forced to do the calculations in python but as they work with the 256 colours of the indexed image they are fast enough.

4 The saturation adjustment

The above stages usually give good restorations but because they are designed to remove colour casts they sometimes result in a rather dull colours, particularly if the original was badly faded. This was seen as a defect in the earlier restoration algorithm and we now include a option to make the colours more saturated.

Several definitions of 'saturation' are in use, derived from the R, G, B values; two common ones are described as HSV and HSL colour spaces and these are provided by the `gimp decompose` command. It is worth emphasising that the values for the saturation S in these two cases may be completely different, for example colours that are close to white are are regarded as saturated in the HSL space but completely unsaturated in the HSV space. Since the aim of the saturation enhancement is to push the colours away from white or grey the HSV option is the one to be used here.

The best definition of saturation uses the (L^*, U^*, V^*) colour space and is $S = \sqrt{((U^*)^2 + (V^*)^2}/L^*$; for equal values of L^* the saturation is measured by the distance from the origin in $U^* - V^*$ space. If we want to increase the saturation we convert the R, G, B to L^*, U^*, V^* value, scale U^* and V^* and convert back to R, G, B. However, as explained above, it is impossible to do the conversion to LUV space in `gimp` and we would be forced therefore to do the whole calculation in python. It also needs to be done on all the pixels of the whole image and will therefore be very slow. Some experiments showed that this calculation would be possible on a fast machine but also that the results were not visibly better than those using HSV space, which is possible with `gimp` procedures.

The plug-in normally produces the more saturated result but it is possible to turn off this calculation if the result appears to gaudy.

5 Technical Details of Coding

The colours of an ideal image $\mathscr{C}$ are pre-tabulated for the chosen value of γ. The systematic search for parameters of the `gimp levels` command that makes the image closest to ideal is done using the simplex minimisation algorithm which seems to work well.

The computation of (L^*, U^*, V^*) is used for the colour balancing phase. Since the input value in any channel is an integer less than 256 the initial γ decompression is done using a look-up table and then the values of L^*, U^* and V^* calculated. The reference values are taken for an observer with 2° field of view and D65 Illuminant. The look-up table needs to allow that during the minimisation search colour values greater then 255 may be encountered.

6 Testing

Before looking at the good results it is instructive to examine a case where the results are bad and to understand why this is so. Figure 4 shows on the left a photograph that has deteriorated badly with age and below it the colours. The fact that the blue line is well above the green and red is consistent with the overall blue colour. What is more important is the very low value in the red channel for the darker colours. On the right is the restored image and its colour map. The algorithm has raised the low dark values in the red channel but this has pushed the bright red values above the green and blue curves and this is obvious in the pink snow in the restored picture. The concept of the 'ideal' image fails to be useful in this case. In fact a reasonable restoration of this picture is possible using the `gimp levels` command although the darker colours will not be accurate because of the loss of information in the red channel. The failure here of our algorithm is a result of the compromise between the robustness necessary to handle the majority of images and the poor results in some extreme cases.

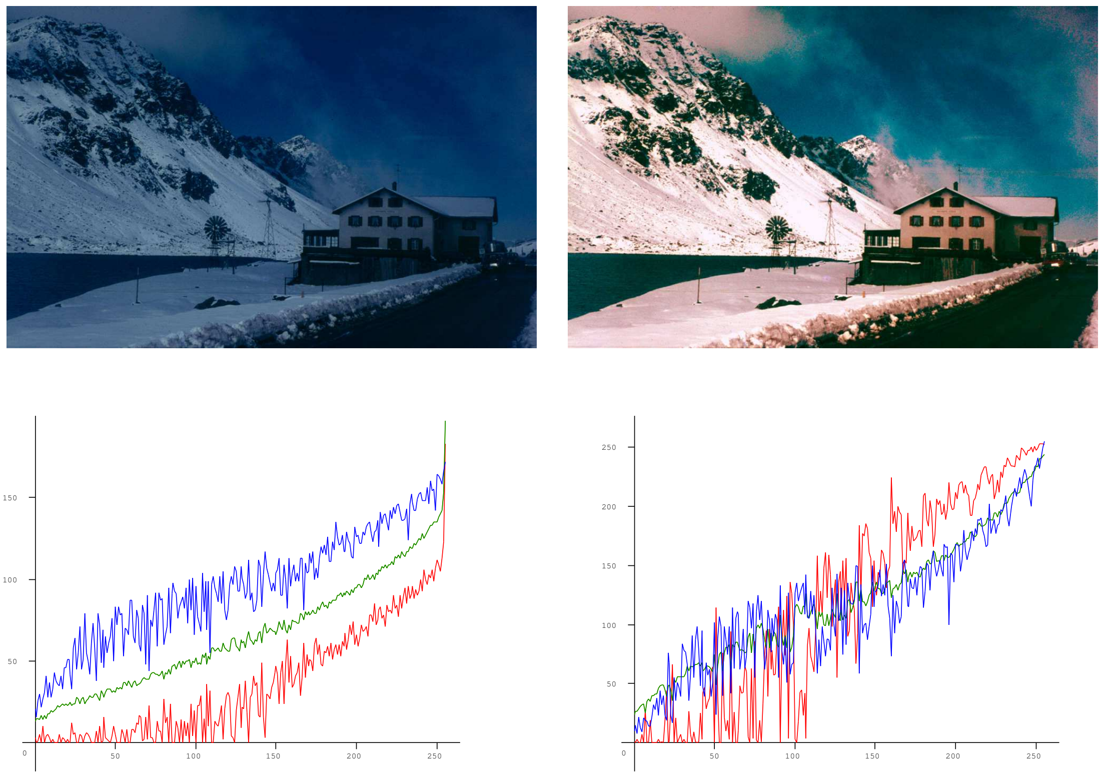


Figure 4: left: Original, right: Restored

Another test is shown in figure 5. This is not a photograph that has deteriorated with age but a digital picture that has been artificially degraded. The reason for including this is that an exact restoration is possible although the parameters used by the plug-in will not the the ones used in the degradation. The result is, if anything, an even better picture than the original!

The plug-in was tested on a collection of 28 miscellaneous pictures with different types of deterioration and although some restored images were deemed to be 'acceptable' rather than 'good' all were improved to the point where they were worth looking at. In general the method works best for pictures where the colour had faded as opposed to those that have gone very dark. This is to be expected since these require scaling up very low intensities with attendant amplification of noise. In the following section I have, of course, chosen examples where the the plug-in works fairly well.

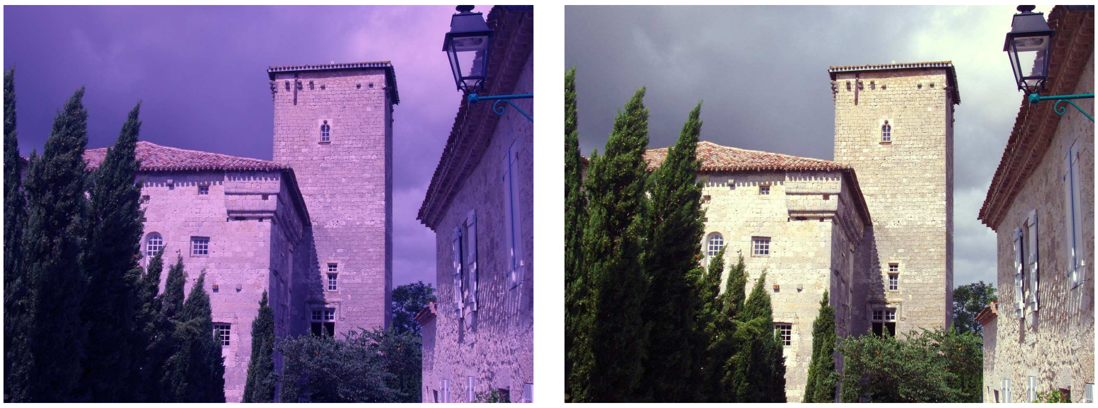


Figure 5: An artificial example. left: Original, right: Restored

7 Examples

I include a selection of results to illustrate the effect of the plug-in on a variety of pictures that have deteriorated in different ways. Figure 6 shows the restoration of the magenta tinted Agfacolor picture shown at the start of this document. The restoration of this using the earlier algorithm was included in the previous account of restoration. In general I have chosen not to compare the new method with the old; there are bound to be cases where one works better than the other but the justification for the new method lies in the theoretical reasoning about the choice of restoration parameters.

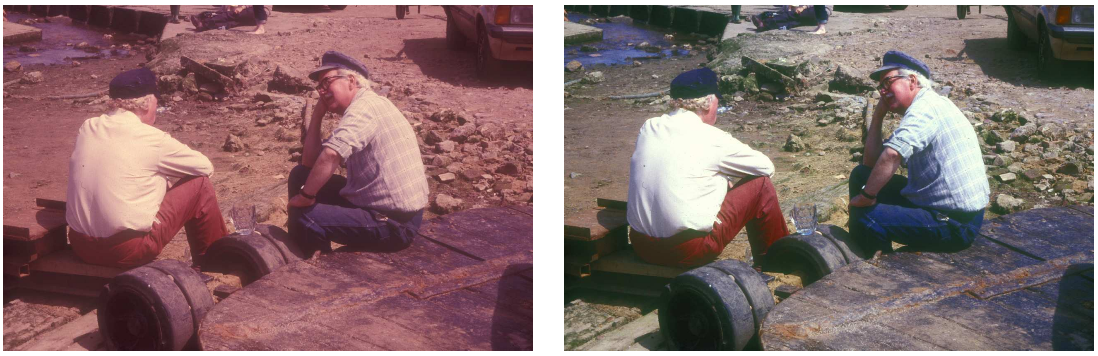


Figure 6: left: Original, right: Restored

As explained in the earlier document this image presents a very challenging problem, there is severe dye loss, the contrast range is very high with the white shirt in the foreground and deep shadows, and the range of colours present is unusual. The picture contains no sky, almost no green objects, and large regions of ground and the table whose greyish colour will be fixed by the colour-balancing phase.

In my opinion the new algorithm performs slightly better than the old one, the colour balance is better and the saturation enhancement improves the result.

Figure 7 shows two Kodachome slides from about 1970 that have faded and in the case of the upper one gone blue with age. Although the colours in the restored pictures are not perfect both are improvements.

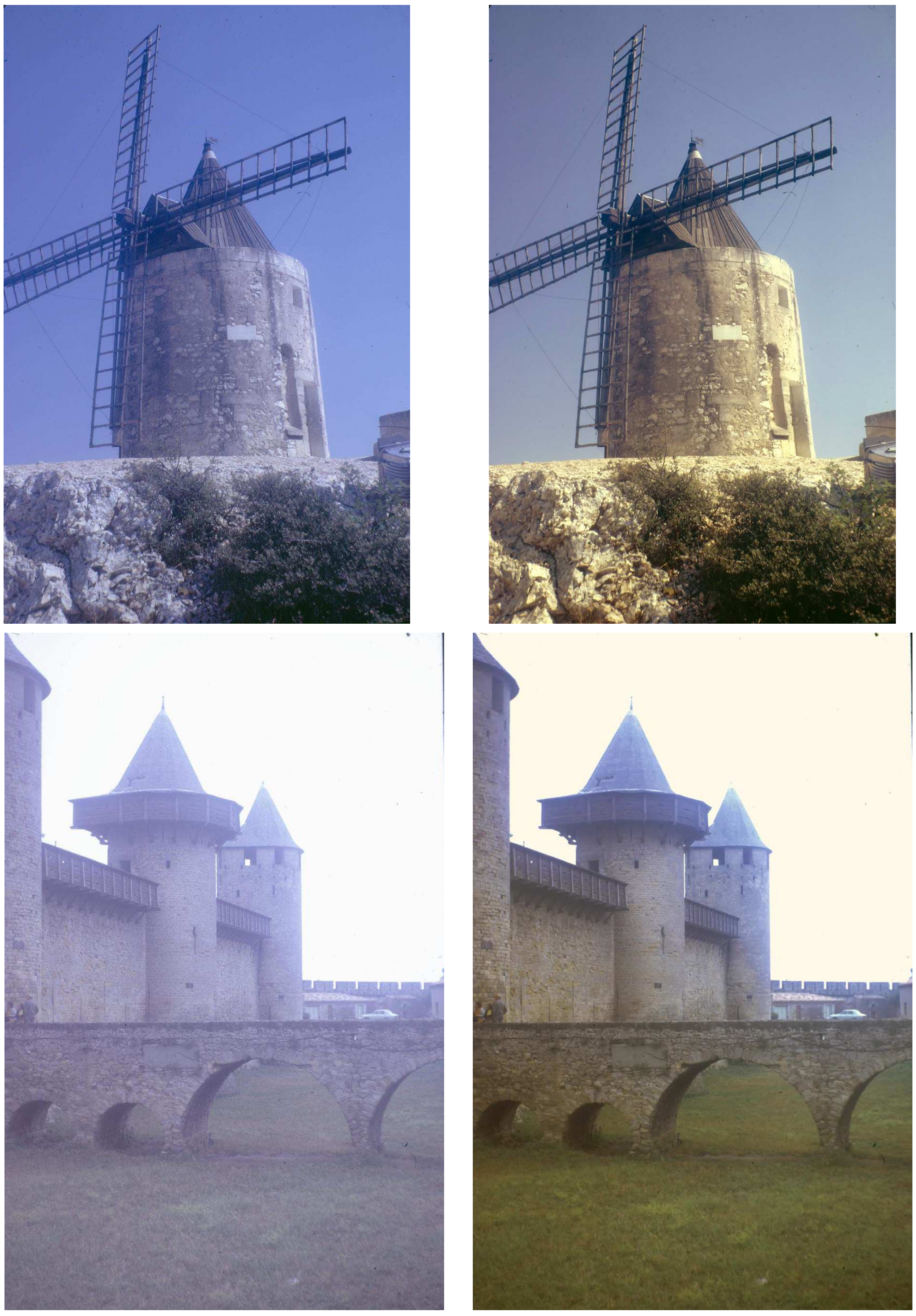


Figure 7: left: Original, right: Restored

Figure 8 is two examples of Ektachrome pictures that have gone very grey with age. I am not sure of the mechanism for this, possibly the grey is silver that has not been properly removed in the processing. The good white highlights are a check on the scaling calculation.

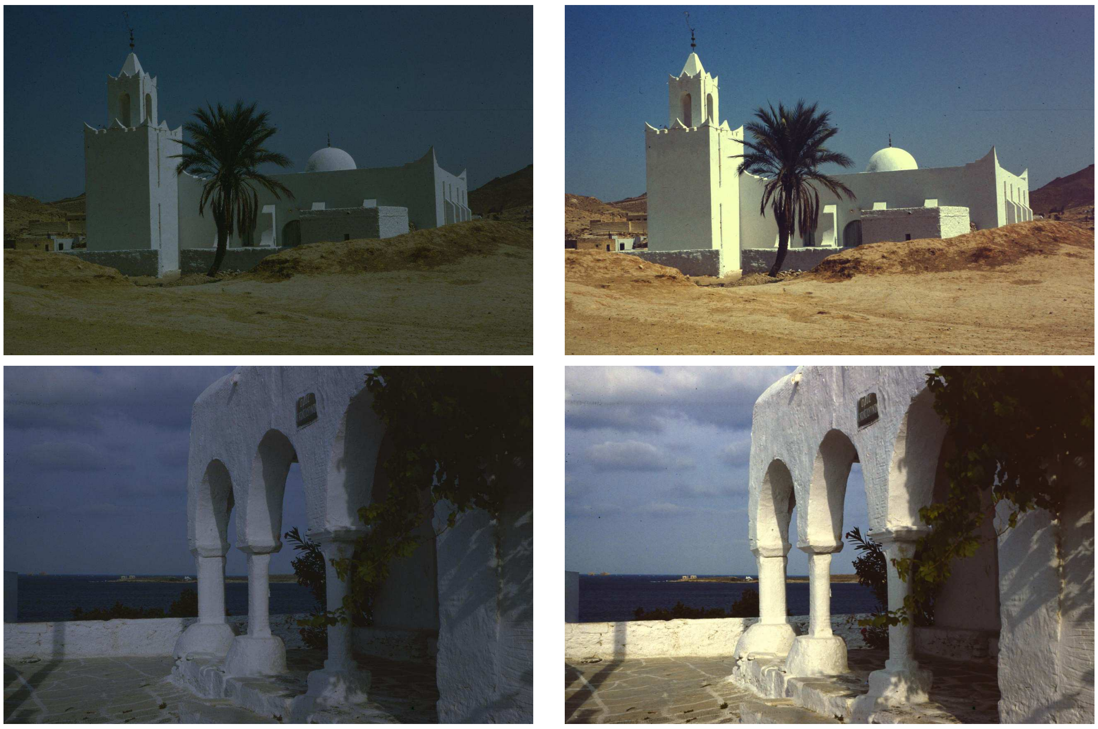


Figure 8: left: Original, right: Restored

Figure 9 is an example of a faded picture that has developed a greenish cast. The restored version looks a bit artificial but is a definite improvement.

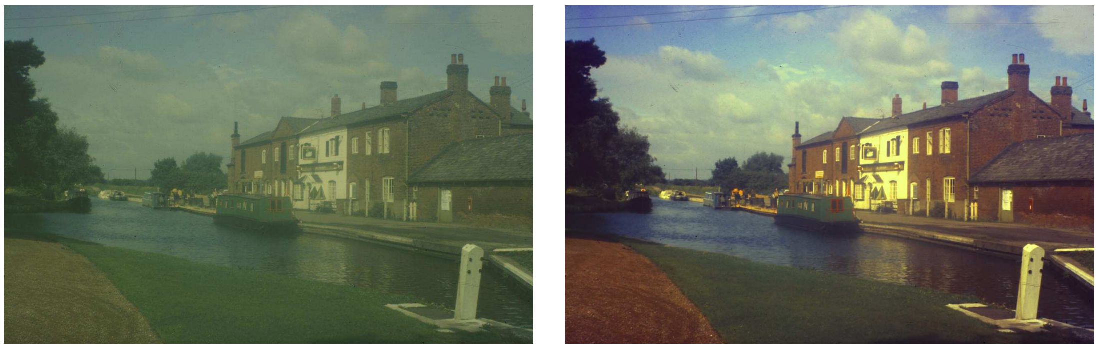


Figure 9: left: Original, right: Restored

Figure 10 is an old Kodachrome slide from the 1960s which shows the characteristic blue fading; it has also developed some fungal infection!.

As a final example we show another Agfacolor slide[2] which has gone pink. The colours in the restored version could be a bit more accurate, particularly the distant trees, and river but the general effect is surprisingly good.

[2]This has also been kindly provided by Tim Gossling

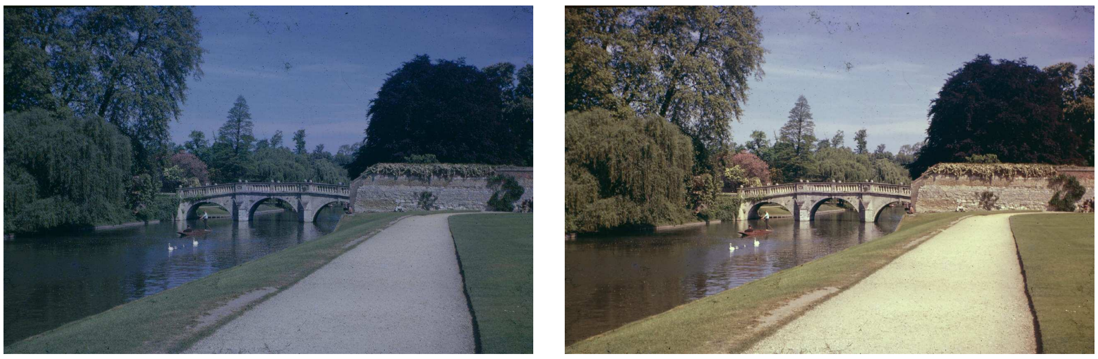


Figure 10: left: Original, right: Restored

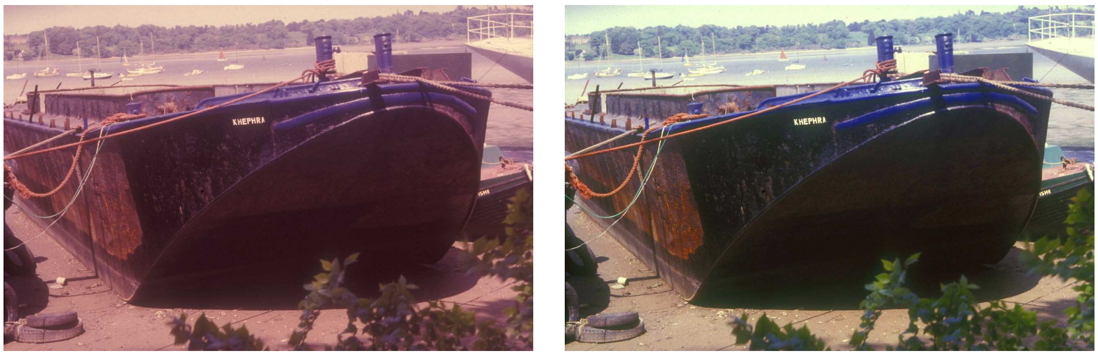


Figure 11: left: Original, right: Restored

8 Conclusions

A previous document described a `gimp` plug-in for restoring colour photographs that had deteriorated with age. We have now developed what promises to be a better version. It still uses the `gimp levels` command to do the restoration but uses a more robust method of estimating the parameters. It also includes an step to boost the saturation of the final image to counteract the tendency of the restoration to push the colours towards grey.

It is important to appreciate that although there are options for adjusting the overall brightness of the restoration and for turning the saturation enhancement on or off the process is essentially automatic and does not need manual intervention. All the examples shown here used the default brightness value and saturation adjustment.

There is also a version operating in batch mode. It processes all the `.jpg` or `.JPG` files in a given directory and puts the restored versions in a subdirectory called `restored` (which must exist).

Appendix – Conversion to `colour indexed` mode

A procedure for conversion of a image to colour indexed mode is built into `gimp`; the main use of this historically was to reduce the size of image files and there has been some discussion of possibly removing this operation at some point in the future. The algorithm described here depends critically on this procedure as a way of estimating the deterioration and in particular depends on the ordering of the colours. It therefore also depends on the particular method used to assign the optimum colours and the results would change if a different method were used in the future. For these reasons it seems best to write a plug-in that includes its own code for colour indexing.

Several methods have been suggested of generating optimum colours which go under the name 'colo(u)r quantisation', some more complicated than others. Since the computation will be done in python we need a simple, fast method and it is worth noting that we do not need to generate the indexed image but simply get a list of colours. The methods based on `octrees` satisfy these requirements. An `octree` is a tree for data storage in which each node has exactly eight branches, and which is particularly suitable for coloured images since the branches can correspond to successive bits in the R,G,B colour values. There is published computer code for the 'Leptonica' implementation of this idea but the rules for the assignment of colours seem rather arbitrary and tests showed that it could assign a colour even if there were no pixels in the image with this colour. This is not important for colour compression but upsets the restoration calculation. We have therefore designed a new algorithm for colour quantisation.

Each node in the tree consist of: a colour value, the number of pixels assigned that colour and a list of branches that lead to other coloured pixels. Think of the tree as covered with wiring for coloured lights each of a different optimum colour. The wiring may pass through a node to further branches even if there is no lamp at that node. At the top level (computer trees grow downwards) any pixels are coloured (127, 127, 127), at level 1 there maybe be pixels of the eight colours with R,G,B values formed from all the combinations of 63 and 191. At level ℓ the colour is specified with an precision proportional to $2^{-\ell}$; and if the there are n pixels with a colour the overall error in colour assignment is the sum over all nodes: $\sum n/2^{\ell}$ and we attempt to minimise this.

Initially colours are assigned to all the 64 nodes at level 2 that have any pixels. We now search the tree for the largest value of $n/2^{\ell}$ and move some of these pixels down a level, keeping a count of the new colours assigned. This continues until ≥ 256 colours have been assigned. We now search for the smallest value of $n/2^{\ell}$ and move these pixels up to a higher level. This decreases the number of colours by one or two. When the numbers of colours equals 256 we check the overall error and stop if this starts to increase.

The set of optimum colours produced by this algorithm is not identical to the set produced by the built-in `gimp` procedure but is similar. Restoration of faded images using both methods are very similar and one would not expect one method to be systematically better than the other. Both algorithms produce reasonable optimum colours and systematic comparison of the restored images does not serve any useful purpose.