Automated Red-Eye Detection & Correction.

Hua Yuen Hui (99436949)

B.A.(Mod.) Computer Science

Final Year Project, April 2004

Supervisor: **Dr. Kenneth Dawson-Howe**.

Abstract

The occurrence of the phenomenon known as red-eye in photos is a large and widespread problem. It has been and is still one area that camera and imaging specialists are having difficulties in dealing with. Solutions are available, but not to the extent that will be of benefit to people.

The goal of this project was to create a program that would automatically solve the problem of red-eye for the user.

Contents

- 1. Introduction……..5
  - 1.1 The Red-Eye Problem……5
  - 1.2 Existing Solutions……8
  - 1.3 The Aim……9
  - 1.4 Overview……..9
    - 1.4.1 Sequence of Events……10

- 2. Automating the Process……10
  - 2.1 Regions……..11
  - 2.2 The Queuing Mechanism……14
  - 2.3 Segmentation Rules……15
    - 2.4 Colour Models……..15
  - 2.4 Region Identification……17
    - 2.5.1 The Design……18
    - 2.5.2 The Process……..21

- 3. Red-Eye Identification……22
  - 3.1 Probable Locations……...22
  - 3.2 Skin-Based Segmentation……..24
  - 3.3 Mapping the Skin Boundaries……26
  - 3.4 Quantifying Permissible Red Eye colours……26
    - 3.4.1 Factors……...27
    - 3.4.2 Colour Range……27
  - 3.5 Glint in the Eye……29
  - 3.6 Constraining the bounding box……..30

- 4. Statistical Analysis……...32
  - 4.1 Motivation……..32
  - 4.2 Description……..33
  - 4.3 The Process……..36
    - 4.3.1 Colour Check……37
    - 4.3.2 Size Check……37
    - 4.3.3 Edges……37
    - 4.3.4 Colour Transitions……..37
    - 4.3.5 The Cornea Detector……...39

- 5. Red-Eye Correction……39
  - 5.1 Technique 1……...39
  - 5.2 Technique2……39
  - 5.3 Blending the Edges……..40
  - 5.4 The Shrinking Problem……41

- 6. Evaluation……..42
  - 6.1 Results……42
  - 6.2 Development……43
  - 6.3 Difficulties……..43
  - 6.4 Successes……..44
  - 6.5 Future Work……..44

- 7. References ……46

1. Introduction.

Humans can use vision to identify objects quickly and accurately. Computer Vision seeks to emulate human vision by analysing digital image inputs. That fact that the world is a three-dimensional environment and a digital image is just a two-dimensional representation, is just one of the many problems that complicate Computer Vision. Indeed to fully solve the problem of Computer Vision, we must first have a very deep understanding of the inner workings of the human brain. Only then can Computer Vision algorithms be designed to truly mimic the intricate complexities of the human visual interpretive processes. This, however, does not prevent Computer Vision techniques to be applied successfully to specific problem sets.

1.1 The Red-Eye Problem.

Everyday millions of photographs are taken from all around the world. Photography is a profession. It is a hobby. However, to most people it is a means to capture and relive memories. Great advances have taken place in the world of photography, with the transition to colour being one of the most prominent. In spite of all the advancements, one problem is still very much evident. The problem is the reddish colouration of the pupils of the human eyes that can occur in photographs. This phenomenon is known as the Red-Eye effect

Red-eye is a result of using flash photography. The red colour of the pupil is due to the flash reflecting from blood vessels at the back of the eye. The effect is worsened as the angle of reflection between the flash and the lens decreases.

Figure 1. Example of Red Eye

Red-eye has become increasingly more prevalent in recent years particularly with the advent of digital cameras. Digital cameras do not require any film, as a result the cameras are easier to maintain and can be built to very compact dimensions. Also the facility to transfer images directly to a computer has made digital cameras very popular. It is not surprising that there has been a huge growth in digital camera sales in recent years.

Digital cameras do not suffer the problem of running out of film. The cameras do not incur the cost of having to replace film or the cost to develop the pictures in order to see them, as images can simply be viewed on the computer. People as a result are less hesitant in taking lots of photos and also less inclined to take the necessary precautions that would otherwise lessen the effect of red-eye. These precautions are

as simple as taking the photo in an area that has appropriate lighting, or making use of the camera's pre-flash.

The diminishing size of the camera has contributed significantly to the problem. As the size of a camera shrinks, so to does the distance between the flash and the lens, decreasing the angle of reflection. In short red-eye is a large and widespread problem.

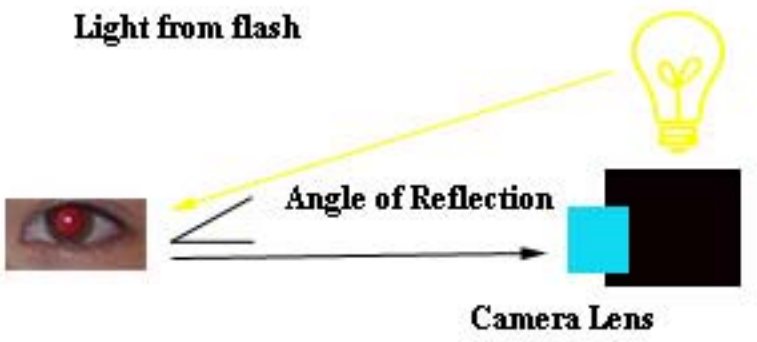


Figure 2. Angle of Reflection

1.2 Existing Solutions.

- The most common approach to the red-eye problem is a provision of a means to reduce the occurrence rather than actually correcting the problem itself. This approach is known as ‘red-eye reduction’. Most modern cameras provide this functionality. Red-eye reduction is performed by the usage of a pre-flash. The pre-flash attempts to shrink the diameter of the pupil of the eye. As a result, less light from the second flash can be reflected from the back of the eye, when the picture is eventually taken.

- The earliest type of solution to this problem was to examine the red-eye artefact and to manually redraw it pixel by pixel using some sort of image editing software. This is a very time consuming process and people would ultimately ignore this option.

- The next type of approach was the design of simple algorithms, which could be incorporated in some of the more popular image editing software. For these plug-ins, the user only had to highlight the red-eye in a photo. The algorithm would then attempt to re-colour the eye. The problem in this case was that the software for which these plug-ins were designed were often very expensive, and people would be reluctant to pay for something that only does half the task.

- Hardware has become available. This hardware is purpose built and incorporates very sophisticated algorithms. The devices are simply too expensive and not aimed at the home market. The hardware is available for use mostly at digital print kiosks. An example is the Automatic Photo Machine (APM) from Lucidiom Inc.

There is obviously a huge market potential for the sale of software that can automatically correct red-eye but at the same time be affordable and easy to use. As a result algorithms pertaining to the red-eye identification and correction process are patented and kept secretive.

1.3 The Aim

- The aim of this project is to use computer vision techniques to automatically detect and remove the effects of red-eye, in a manner that is realistic to the observer of the photo.

- To develop a system that is easy to use, can easily be adapted, modified, reproduced and even improved upon.

- The ultimate goal was to create a widely accessible piece of software that will work and benefit many users. The functionality could be provided by means of a web page.

1.4 Overview

This document is intended to provide an overview of the project as a whole, as well as to delve deeper into the design of the techniques used. All source code is fully commented and available on the accompanying CD.

The software is designed to minimise user interaction. The user runs the software, opens the picture to be corrected and then presses the button to begin the process. The process is fully automatic.

The image will be systematically broken down and analysed by a series of algorithms to determine the pixels that are representative of red eyes. After this time another algorithm will be applied that will attempt to modify the pixels of the eye to appear more natural in colour.

1.4.1 Sequence of Events.

1. Image opened.
2. Skin segmenting algorithm applied.
3. All skin-like areas will be noted.
4. Each of the skin like areas will be searched. The boundaries of the skin regions will be marked.
5. Only the areas that are fully bounded by skin areas will be retained.
6. Evidence of a reflections is sought for within theses areas.
7. A series of metrics is then applied to each of the remaining areas to determine a high probability of that artefact to indeed being a red eye.
8. The correctional algorithm is applied.
9. An altered image is returned, if it contained red eyes.

2. Automating the Process

In order to implement a system that can automatically correct incidences of red eye, a series of logical steps would have to be developed. Two approaches were immediately apparent. The first approach was to simply scan through the image and analyse, with a series of algorithms, all pixels suspected to be incidences of red eye. This would mean every part of the image would have to be thoroughly examined in a recursive manner. The concept was immediately abandoned in lieu of a more elegant and systematic approach. The idea was to develop a queuing system that could divide an image into smaller parts (regions), and then analyse these regions in turn. This meant that algorithms could be applied to specific areas only. These algorithms would analyse each region, determine the regions that represent red eyes and finally correct the red-eye artefacts. This design made the system more efficient and easier to adapt.

The queuing mechanism and governing system for the Red-Eye system is analogous to a conveyor belt in a manufacturing plant. At different points along the belt, different operations are performed.

The governing system consists of all the routines performed on the queue at various different stages.

Components of the Queuing System.

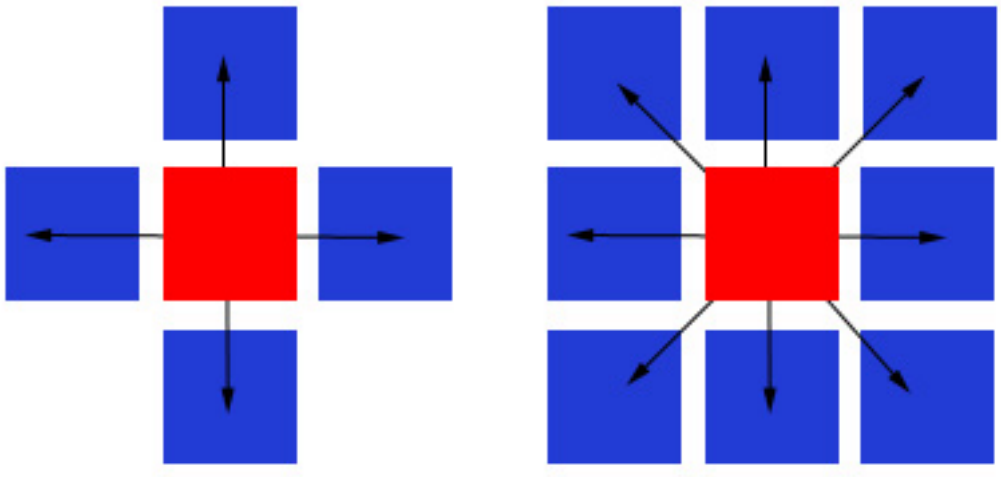
- Regions
- Queuing Mechanism

Governing System

- Segmentation Rules
- Region Identification Process
- Red-Eye identification algorithms
- Red-Eye correction algorithm

2.1 Regions

In Computer Vision, a region by definition is a grouping of connected pixels. Regions can be described according to the connectivity of the adjacent pixels. The connectivity can be either 4-way or 8-way connectivity. The Red-Eye program uses the 8-way connectivity model.



.

Figure 3. Connectivity of Pixels (four and eight way connectivity).

For the purpose of this project it was necessary to describe a region as the set of pixels encapsulated by a bounding rectangular box. The bounding sides of these rectangular boxes are determined by the uppermost, lowermost, leftmost and rightmost pixel points of the region, by the basic Computer Vision definition of a region. All operations performed by the Red-Eye system are confined to operating within regions. The original image supplied to the program to be corrected is defined as the first region.

As processing takes place, each region can potentially be further divided to yield several smaller regions. Every region contains coordinates, or offsets, into the original image. In other words, there is a direct mapping from each pixel outlined by a region to the corresponding pixels in the original image. This was necessary for the correction process, whereby alterations to specific parts of various regions could be applied correctly to the original image.

```
//pseudo code Class Definition
Class CSubRegion
{
        int RegionType;
        int Area;
        int Height;
        int Length;
        //pixel coordinates;
        int highx_coordinate, lowx_coordinate,
            highy_coordinate, lowy_coordinate;
        CsubRegion Pointer_Next_Region;
};
```

Extracting the bounds of the box from the pixel extremities.

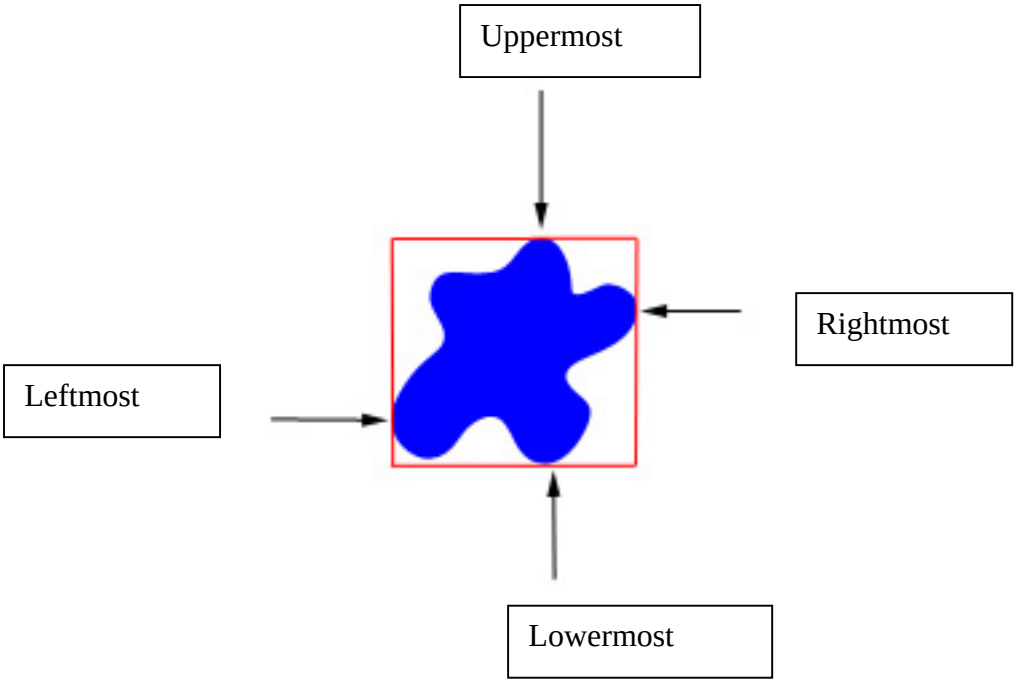


Figure 4. Determining the Bounding Box

The region class also contains information about the length, height and area of the region, as well as details regarding the type of pixels the region encapsulates. In addition the region must also know the location of other regions.

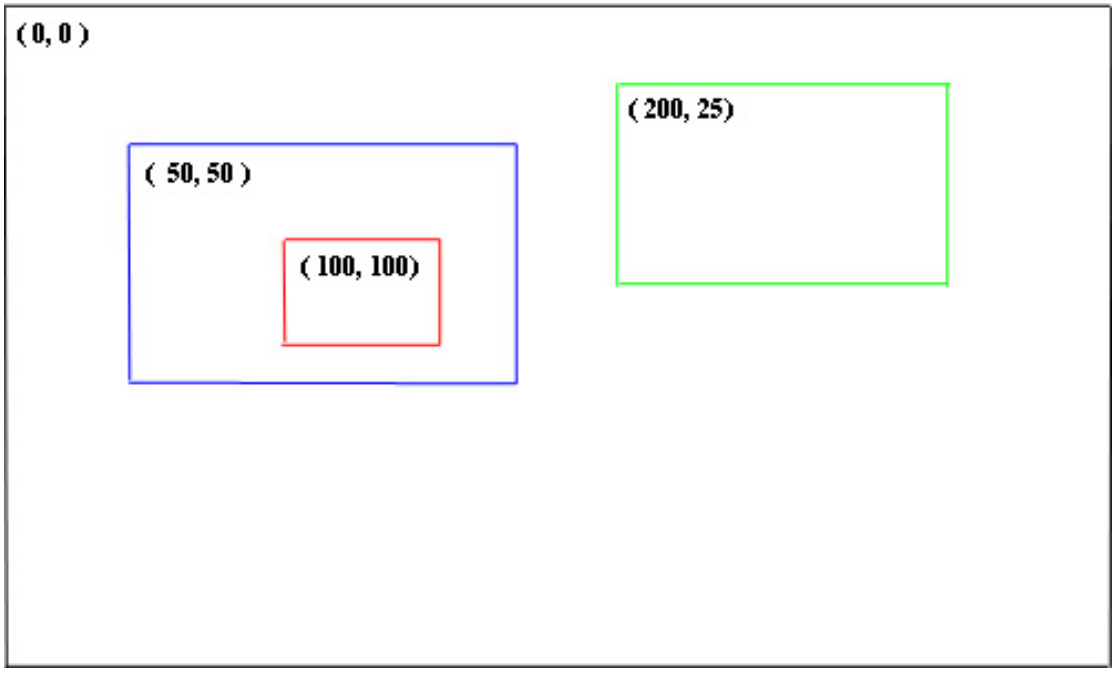


Figure 5. Origin of sub Regions relative to origin of original image region.

2.2 Queuing Mechanism

The Queuing mechanism is the core mechanism in automating the red-eye removal process. The task that the system performs is to label the various regions and to ensure that regions are operated on in a specified order. This order is simply a first-in first-out order. The queue consists of three parts:

1. The region at the front of the queue.
2. The region at the end of the queue
3. All the regions between.

The queue structure takes the form of a single-link linked list. When the first region is inserted into the queue it will become the head, and at the same time the tail, of the queue. The queue at this point will be just one element long. As each region is analysed, it can be removed completely from the queue, attached to the end of the queue and relabelled, or it can spawn several smaller regions that will be labelled and attached to the end of the queue. All operations are performed on the region at the head of the queue.

In order to do this the class representing the queue must at all times know the location of the head region. A pointer to the tail region is maintained to make the queuing process as efficient as possible. This saves the computational overhead of traversing the queue each time to find the last element it contains, in order to append more regions.

```
//pseudo code Class Definition
class CRegionQueue
{
        CRegion Pointer_headRegion;
        CRegion Pointer_tailRegion;
        int mQueueLength;
        void AppendRegion(int x_val, int y_val);
        void AppendRegion(CSubRegion *region);
        void DeleteRegion();

};
```

As regions are added, the queue sets the pointer variable in the tail region to point to the newly added region. This newly added region is then set as the tail region.

2.3 Segmentation Rules

Segmenting an image is the process by which a computer attempts to separate objects within an image from the background as well as from other objects. The segmentation rules are the rules that will determine the formation of regions. The segmentation rules are based on analysing the colour and edge properties of a region. These will be examined more thoroughly in a later chapter.

2.3.1 Colour Models

RGB

The RBG colour model is the most widely recognised colour model. It comprises of three components namely the red, green and blue colour channels. The RGB model is used to a great extent in solving computer vision problems, but is better known for colour representation in the displays of television sets and monitors. The value of a colour by this model is best described as being a vector in three-space where red, green and blue represent the axis. Colour is thus a result of the combination of the red, green and blue components. The origin of the cube (0,0,0) represents pure black while its polar opposite (255, 255, 255) represents pure white.

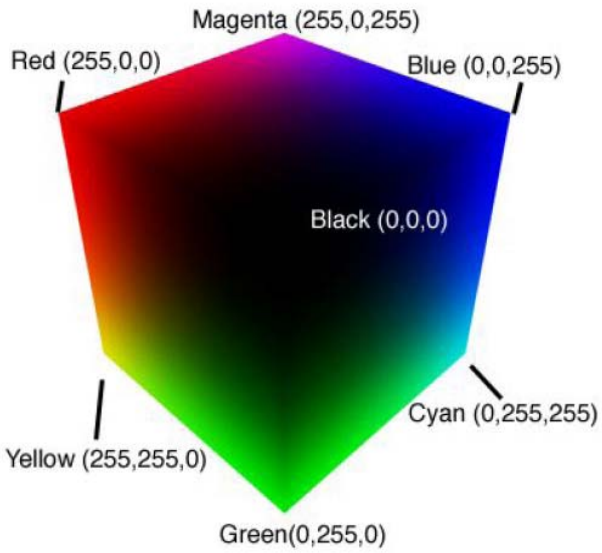


Figure 6 a. RGB cube from perspective of the black corner

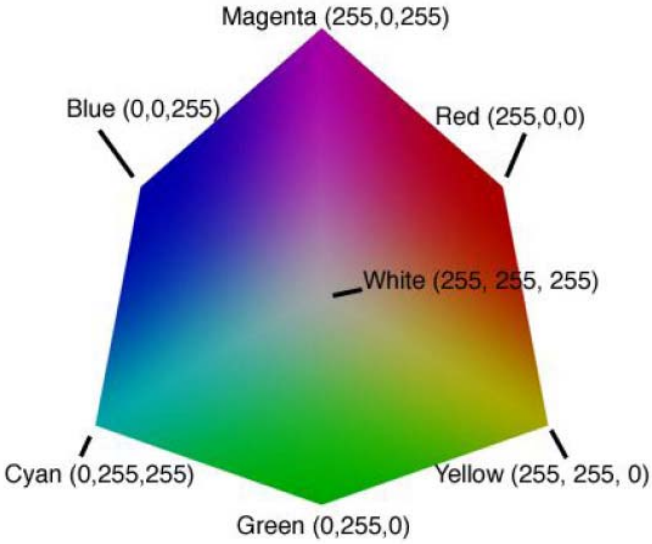


Figure 6 b. RGB cube from perspective of the white corner

HSI/HSV

The HSI model comprises of three components. These are the hue, the saturation, and the intensity/value [Cardini, 01]. This model is a more intuitive representation of colour information. The model is best presented as a colour cone or cylinder.

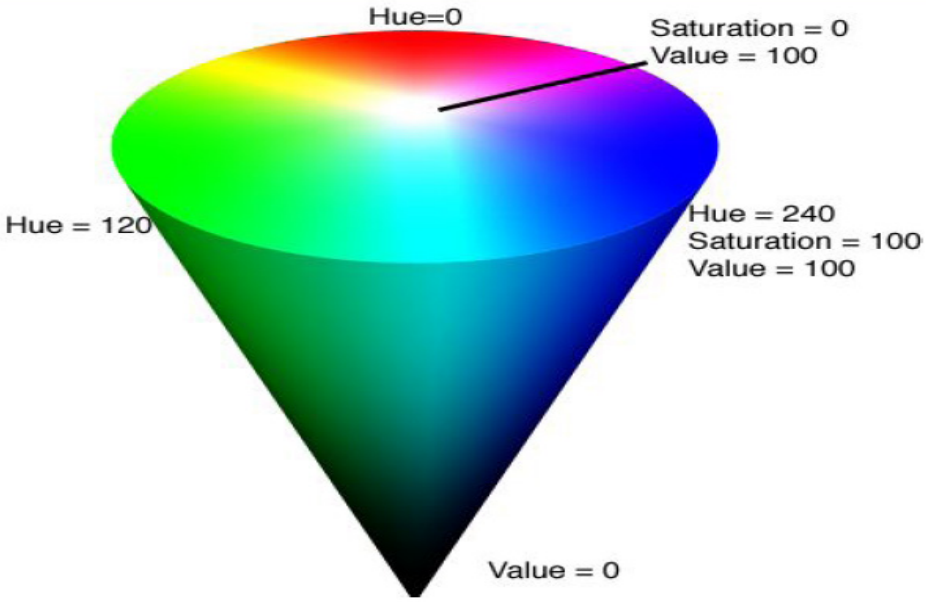


Figure 7. The HSV colour cone

The hue represents a person's perception of a colour, for example green or orange. Hue changes as one moves around the cone. Saturation is a measure of a colour's dilution by white light. This provides us with light or dark shades of a colour. Saturation increases from the centre of the cone to the outside. Finally, the intensity is a measure of the brightness of the colour. Intensity increases along the cone's vertical axis. The great advantage of using this colour model is the fact that it separates intensity from the colour components unlike the RGB model that couples intensity with colour information.

2.4 REGION IDENTIFICATION

The TIPS (TRINITY IMAGE PROCESSING SOFTWARE) provided as an environment to develop, execute and test the Red-Eye system provides functionality to identify and extract regions within a segmented image. Unfortunately there were several issues resulting in the need to use an alternate version.

The red-eye system largely performs segmentation of an image based on a simultaneous analysis of both an image's RGB and HSI representations. This was a

minor detail and the solution to this was to create a new image, either a RGB or HSI representation, then segment this new image based on the analysis of both colour representations before applying the region identification process. However, the real problem did not become apparent until it was necessary to further sub divide regions and at the same time retain these regions coordinates within the original image. The region identification used in TIPS was designed primarily to extract regions from the entire image provided. Although it was possible to create images from each of the smaller regions, the coordinates within the region then became relative to the origin of the new smaller image as opposed to the origin of the original image.

It was possible to use an image that had the same dimensions as the original image and segment according to the smaller regions offsets and mark every pixel falling outside of this region as being part of the background. This approach, given the large potential number of regions proved to be computationally expensive and unnecessary. The TIPS version also had the undesired effect of outputting a new image showing the extracted regions with each use of the function. It was necessary to develop a way to restrict the region identification process to regions within regions in the original image.

The decision to program a new region extraction algorithm was further influenced by the fact that this version could more easily be adapted or exported to other platforms without all the extra facilities provided by the TIPS region extraction process. In addition to this, TIPS is a GUI based application. In order to build a server version of the program it was necessary to write the program as a console application, namely a program that uses a command line interface. The existing region identification and extraction class was simply incompatible.

2.4.1 The Design.

The solution devised was to create a map of the image, where each pixel location is represented by a single integer. The design is a slight variation on the algorithm outlined in [Sonka, Hlavac and Boyle, 99]. The map construct is a two-dimensional integer array structure. After segmentation, the value of zero represents a pixel belonging to the background and the value of one belongs to a region. The extraction process attempts to identify and extract regions from this map. The function parameters are the map and a region of the map to be analysed.

The map is scanned one element at a time, in a row-by-row series of sweeps, within the boundaries specified by the region input. If the value is zero, no more processing takes place and the pointer is moved to the next element in the row. If the value is greater then zero, it is part of some region and the value is replaced with another integer value corresponding to a label. The label values start at one. Once an element has been identified as to belong to some region, its neighbouring elements are checked. The neighbourhood consists of a 2x2 square.

There are three possibilities:

1. All the other values in the neighbourhood belong to the background so a new label value is created, and this value applied to the reference element.

2. If the neighbourhood contains valid pixels, the label from that valid pixel is used for the labelling of the reference pixel.

3. In the case where there is more than one label value in the neighbourhood, a label collision has occurred. The reference pixel is assigned a label value equal to any of the other label values in the neighbourhood.

When a label collision occurs it is necessary to mark the two labels as being equivalent, as the two different labels represent the same region.

Figure 8 a shows the region map immediately after a segmentation process.

Figure 8 b shows the labelling of the regions with collisions.

Figure 8 c shows the extracted regions.

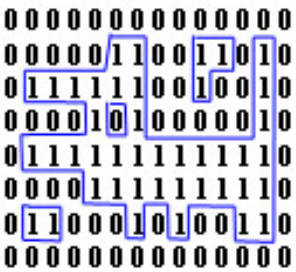


Figure 8 a

Figure 8 b

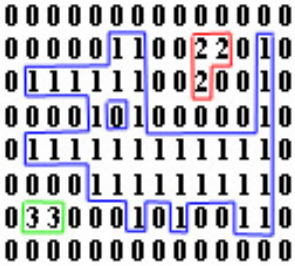


Figure 8 c

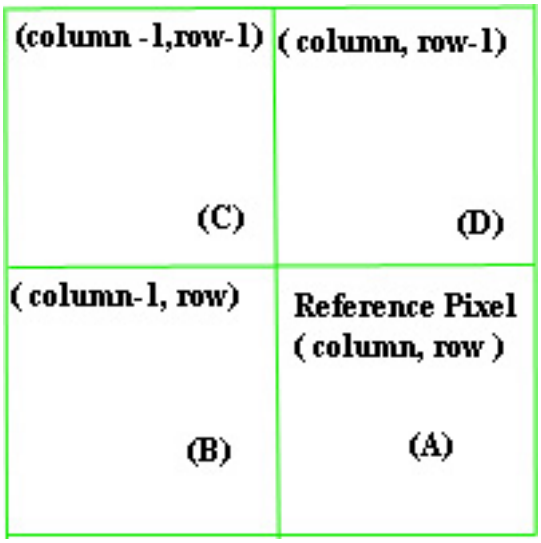


Figure 9. The region identification mask.

Equivalences occur across (B) and (D) in the neighbourhood mask.

The equivalences are handled by two class structures. The first of these classes is very simple. It is a node construct, whose functionality is to hold the label value and also a pointer that can point to other nodes that contain an equivalent label. Essentially equivalences are stored along a linked list structure.

```
class CEquivalenceNode
{
        long int region_tag;
        CEquivalenceNode *mNextNode;
};
```

The second class maintains the list. At the end of the region extraction process, each different label will represent a different region. Since it is not possible to guess how many regions there will be before hand, a dynamic structure is used. The standard C++ vector template class is used for this purpose, taking the CEquivalenceNode as its base type. The structure is essentially an array of equivalent nodes.

For each new label created, a new slot in the vector class is added. The vector class acts as a look-up table that is indexed by the label value. The use of the Vector Class greatly simplified memory allocation.

2.4.2 The Process:

For each new label, a new element is added to the vector class. For a label collision, the two labels are examined. The concept is to use the lowest label to identify a region. Search along the two lists, A and B. If the two labels in the nodes at the end of the lists are the same, the two labels have already been marked as being equivalent. Otherwise append the list with the lowest value in the tail to the tail of the other list. This results in the two labels being made equivalent, in addition to all the other labels that the lists A and B might already contain.

Every label will have an associated region. Regions grow in size as connected pixels are identified. The final step is to extract only the regions that are associated with unique labels from the tails of each equivalence list.

3. Red Eye Indentification.

3.1 Probable location of Red-Eyes

Eyes are located on a persons face. An obvious solution to locating eyes or indeed red-eyes would be to first locate faces within an image. There has been extensive research put into developing ever quicker and more robust face detection and tracking software. This is however out of the scope of this project. Several algorithms exist that could have been implemented in this project to locate these faces. One such algorithm that analyses a series of features [Voila and Jones, 2002] can be used with a ‘high degree’ of success and speed to locate the upper part of the face. The problem however is that the algorithm will dismiss faces that are occluded or facing in such a manner that the head is more profile than portrait. Red eye tends to occur more often in pictures taken without precautions. Frequently these could be spontaneous pictures in which all subjects are not fully in frame.

This is the primary reason for the Red-Eye system to adopt a more general approach. This approach is based on the assumption that an eye is surrounded by skin. The advantages of skin-based segmentation is the loss of the sizable overhead of first determining faces within an image coupled with the fact that even faces that are occluded, profiled, partially out of frame or simply not recognised, will all be further processed.

Figure 10 a: An example of a partially occluded face.

Figure 10 b: An example of a face partially out of frame.

Finding possible red-eyes is a five-step process.

1. Segmentation of the skin.
2. Extract the regions.
3. Map the skin boundary in the region.
4. Search for high red energy within skin boundaries of the region.
5. Extract the candidate red-eye regions.

3.2 Skin-based segmentation

There has been much research entailing skin- based segmentation, which incidentally is the basis for some face-detecting algorithms. Most of the algorithms use a range of colour values to define skin colour. There seems to be a preference, amongst the developers of skin-segmenting algorithms, to using the RGB colour model.

Figure 11: A skin-segmented image.

This project bases its analysis of skin colour using both the RGB and HSI colour models. The values are arbitrary, unique to this project and obtained by extensive colour sampling from flesh tone colours from digital photos.

Algorithms using the RGB colour model used very specific values, which gave the skin detector a limited scope. They were designed primarily to segment Caucasian skin tones with little variance.

Using the HSI model allowed for a greater range of skin tones. The problem however was the HSI model made it difficult to constrain the colours to specifically flesh-like tones.

The solution adopted was to use the HSI model to allow for a large range and the RGB values to constrain over dominance of specific colours e.g. too ‘red’.

Skin colour seems to occupy a small range of hue values. Experiments put this in the range of >=350º and <= 30º. This is further constrained by limiting the range of saturation to being greater than 50. Pixels that had an intensity value less then 25 were considered too dark and simply ignored. Further constraints are imposed by examining the relative values of the RGB component colours. The range is left as broad as possible to minimize the probability of excluding actual skin pixels.

Skin- like pixels:

HUE >= 350º && <= 30º.	Large range of colours.
Saturation >50	
Intensity >25	Otherwise too dark
If R > 100 && R>1.2*(G+B)	Too red - ignore
If R+G+B >600	Overly bright white – ignore

Similarly more complicated RGB ratios are used to exclude pixels that exhibit pale orange characteristics.

A connected component analysis followed by a region extraction process is then performed on the segmented image map. The regions are marked as SKINSEGMENTED.

3.3 Mapping the skin boundaries.

Regions are bounded by a rectangular box, and the algorithms are applied to the entire area of these boxes. The patches of skin are not regular in shape or size, therefore, regions are not fully composed of skin pixels. The possibility exists that regions could be created in the gaps between the outside of the skin regions and the bounding box. This is an unacceptable possibility, as eyes cannot occur outside of a skin area. It is necessary to mark the boundaries of the skin region such that the search for red eyes is confined to within skin regions. It is not possible to simply confine the search to skin pixels, as an eye area will be marked by a hole (not skin pixels) within a skin area.

The boundary mapping is a two-pass procedure. Horizontal and vertical sweeps are performed one line of pixels at a time across the region. For each pass along a vertical sweep the uppermost and lowermost skin pixel is mapped. Similarly for each horizontal sweep the leftmost and rightmost skin pixel for each line is mapped.

Storage of the skin bounding pixels is provided by a two-dimensional dynamic array structure.

```
INT SKIN_MAP[LENGTH][4];
```

LENGTH is set to the greatest dimension of the region. The second index in the skin map is fixed at four. The four values represent uppermost, lowermost, leftmost and rightmost.

3.4 Quantifying Permissible Red Eye colours.

Every eye is unique to the individual. Several factors influence the severity of the red-eye effect. Examination of many photos in which red-eye was forcibly induced revealed several factors that influence the severity of the effect.

3.4.1 Factors.

1. The smaller the angle of reflection between the flash and the lens, the greater the reflected energy from the back of the eye captured by the camera lens.

2. Lighter blue eyes, for example, appear to have a greater tendency to exhibit a more severe red eye effect, than darker eyes such as those with a brown coloured iris.

3. The refractive properties of the lens of the eye also seem to be an issue. Eyes that require glasses do not exhibit the red eye effect to the same extent as eyes that do not require lenses. This is the case when the person is not wearing lenses. The assumption being that as light from the flash enters the eye, it does not converge properly on the blood vessels at the back of the retina. Under this assumption it stands to reason that the light from the flash that is reflected is not as focused as it would otherwise be in an eye that does not require lenses.

4. The ambient lighting is a factor that determines how dilated the pupil will be. The more dilated or wider the pupil the more light that can reflect from the back of the eye. This is the principle by which the red-eye reduction system, built into most modern cameras work. The pre-flash attempts to contract the pupil thus reducing the amount of light reflected when the picture is actually taken.

3.4.2 Colour Ranges

The red-eye effect is a general term referring to the undesired reflection from the back of the eye. Depending on the factors involved red-eye eye can range in colour from shades of pink, to glowing bright red to shades of purple. It is dark coloured eyes that exhibit the purple colour.

As with the algorithm for selecting skin-like pixels, both the RGB and HSI colour models are used for selecting the red-eye pixels.

The range of values used to segment the red eye pixels in the regions were selected arbitrarily based on empirical measurements on sampled photos. The values allow for as large a range of values as possible.

Figure 12: Examples of varying degrees of red-eye.

The HSI model simplified the pixel selection. The transition from purple to red occupies a continuous arc around the hue component. The sector falls between 280 degrees and 7 degrees.

As with the skin segmentation procedure, the RGB model is used to constrain values. Only the colours with a comparatively high ratio of red energy to the green and blue energies are allowed. This ratio is reduced for colours that are purpler in appearance. In other words, there are different ratios depending on the value of the hue component.

The pixel selection algorithm also allows for pixels that are bright white. The reason for this is that the white pixels might otherwise divide potential red-eye regions into two regions, which is an unacceptable possibility.

Once the potential red-eye pixels have been segmented. The region identification process is used to extract the red-eye regions. Each new potential red-eye region is compared with the skin-boundary map for the parent region. Only the regions that are fully encapsulated by bounding skin pixels are added to the queue. The regions are

labelled as red-eye regions. Regions not meeting this requirement are simply deleted.

3.5 Glint Detection

The next few algorithms applied to each of the regions will attempt to remove regions that have a low probability of representing red-eyes. The first of these algorithms was designed to check for a glint in the pupil of the eye. The premise behind using this algorithm first, was the simple fact that all eyes that exhibit the red-eye problem must contain a glint. The assumption underlying this was that in order for the red–eye to occur there must have been a flash. Regions that do not contain a glint can simply be removed from the queue with no further processing required.

The glint detector is very simple. It searches for a relatively bright, high contrast area within the potential eye region. It should be made clear at this point, that the regions actually encapsulate just the pupil part of the eye, as this is the part of the eye that actually appears red. The reason for finding the pupil region will be made clearer in the next chapter.

A high contrast within the pupil region will generally denote a sharp change in colour. An edge can be defined loosely as locations in an image where there is a sudden variation in the grey level or colour [Efford, 2000]. In order to define the glint it is therefore necessary to find evidence of pixels that reside on edges and that are relatively bright when compared to other pixels from the same region.

TIPS provides the functionality for creating edge images. An edge image of the entire image is created and stored internally in the program. The edge image is generated using the Prewitt operator and then thresholded to the value of 80.

The HSI colour model is used to detect the glint. The standard deviation is first calculated across the region for the Intensity values of the pixels. In order to work as generally as possible, all pixels that reside in the set of pixels where that pixel has a intensity value greater than the sum of the mean and the standard deviation. In other

words only the top 2.5% of the most intense pixels are considered eligible, which are then checked to see if they coincide with an edge. One last criteria to this, the intensity must be greater than 140 else the pixel is too dark to be considered a glint. All regions not satisfying these criteria are removed from the queue.

3.6 Constraining the Bounding Box.

The red detector allows for a large range of reds and indeed purples. The idea is to lock the region border onto the pupil region exclusively. This is necessary for a statistical description of the eye to be applied at the next stage.

The reason that there exists a need to remove the stray pixels has to do with the fact the red colour detector applied earlier is very general. There is a high probability that part of the eye-lid or indeed blood vessels visible in the cornea (the white of the eye) of the eye will be included as part of the pupil, giving a distorted shape and size to the supposed pupil region. In order to rectify this, it is necessary to perform a standard deviation across the pixels that had been previously identified as being red. This is achieved by checking the pixel-map for valid pixels for the corresponding pixel offsets in the region. It is the red energy of the pixel that is of importance. The standard deviation is calculated in accordance to the ratio of the red to green energy from the red and green channels of the RGB colour model. Based on values obtained from examining the red colour in red-eyes, the red to green ratio is very important. The green colour acts in a manner that lessons the intensity of the red colour. Although the red to blue colour ratio is also significant, the algorithm does not factor it in. This is due to the allowance of a closer ratio between the red and blue channel to allow for shades of purple colouration.

All pixels that have a red to green ratio less than the difference of the mean and the standard deviation are marked on the pixel map as being invalid. The region identification is once again used and applied to the region. In theory the result should simply be a slight shrinking effect to the region. The region should now outline just the dominant reds. The only precaution necessary when programming this aspect is to check for a zero division operation. This is when the green channel has a value of zero. It should be noted that as the green channel value gets smaller, the ratio

between red and green channels could become significantly large. This ratio needs to be limited else the standard deviation value will become too large to be of use. All ratios greater than four are capped at four.

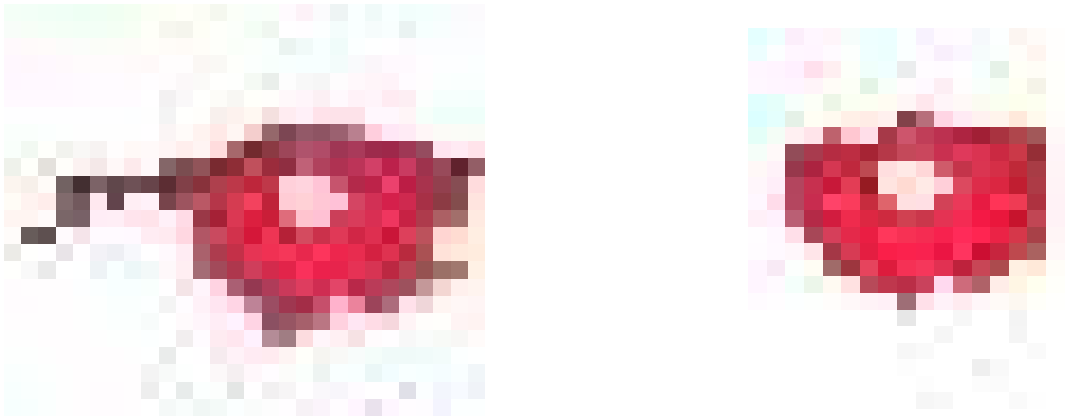


Figure 13: The pupil region of an eye before and after constraining the reds.

4. Statistical Pattern Recognition

Statistical pattern recognition is a very large area in the field of Computer Vision. Essentially it is an approach, in the broadest possible sense, which allows computers to differentiate between objects. A simple statistical analysis bases identification of objects by a classification of its features. The greater the number of matches to a set of features the higher the probability that the object being analysed belongs to that particular class. A statistical approach uses probability to determine the likelihood that an object is indeed the object being described.

It should be pointed out that the Red–Eye system primarily uses feature analysis. Up until now it has been removing the regions that do not adhere to the most basic features of a red-eye:

1- Location, must be in a skin region.
2- Colour, must be purple to red in colour
3- Must contain evidence of a glint from a flash.

4.1 Motivation

The algorithms for the Red-Eye system were designed to be very general. The idea behind this was to allow the system to be sensitive enough to detect instances of red-eye, which can occur in the background of an image or in subjects that are simply at a relatively large distance from the camera. In cases such as these, the reds can be pale in colour or even off red. As a result there is a chance that there will be several false detections of red-eye instances at this point. These false detects could represent anything from a person's lips to red skin blemishes.

4.2 Description

An eye, on a superficial level comprises of the pupil, the iris, the cornea and the eye-lid. Compared to other parts of the body, there is a surprisingly large amount of detail and indeed colour variation over such a small area. This fact is used as the basis for building the description of the eye.

The description of the eye uses the pupil as the foundation. A grid type structure is placed over the potential eye region with the pupil at the centre. This was the reason that it was necessary to constrain the region to fit the pupil as tightly as possible. The pupil is small at the centre of the eye and relatively round or symmetrical in all directions. The pupil is nearly always completely visible as opposed to the iris, which is frequently partially occluded by either the upper or lower eyelid. This coupled with the fact that the pupil exhibits the red colour made it the ideal basis to forming the description of the eye.

The pupil region is divided into nine cell shaped areas. The division is not always equal. All the cells along the centre row or the centre column can be scaled slightly to accommodate any remainders from the division. Following this the grid is expanded to two further cells in all directions. The result is a mask overlaying the eye that is 7x7 in size.

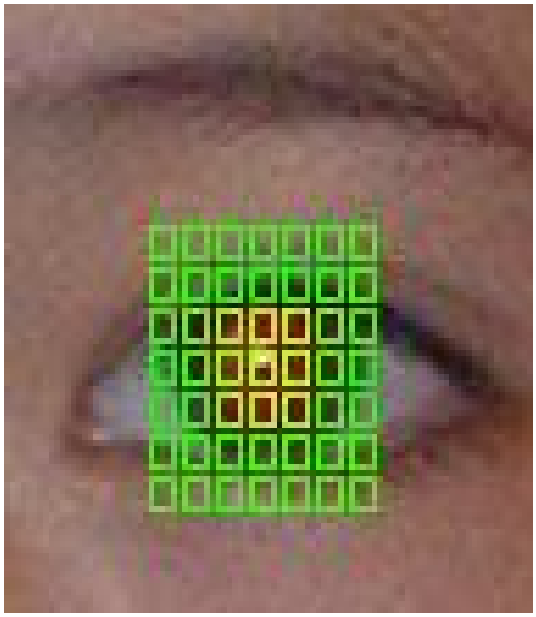


Figure 14 a: A visual interpretation of the grid over an eye region.

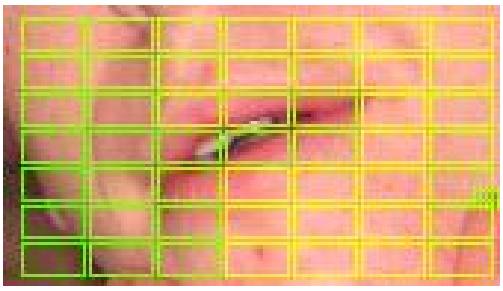


Figure 14 b: The grid over a non-eye region.

Each cell is further divided into quadrants to allow greater weight to be associated with each quadrant.

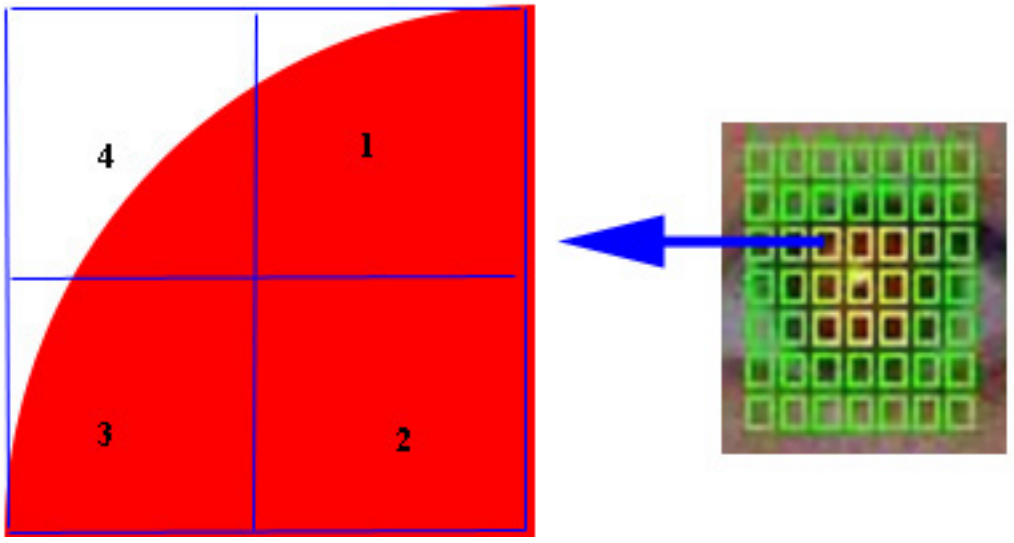


Figure 15: Cell divided into quadrants.

The cell belonging to the top-left part of the pupil (Figure 17.) is analysed for the abundance of red it contains. The quadrant division allows greater emphasis to be focused on quadrants 1-3. This applies equally to each cell where emphasis can be based on any of other quadrants. Obviously it not possible to divide each cell perfectly. In cases such as this the odd pixel will form an overlap and will then constitute parts of the two adjacent quadrants, which it is being divided between. Each of the cells are individually numbered from 0 to 48 inclusive. Access to the cells is provided by a large switch statement structure. The ***analyse()*** function performs the operation on the specified cell. The parameters needed are the location and size of the cells in addition to the type of analysis to be performed and the quadrants to be operated on.

```
switch(i)
{
        case(0):
        Analyse(……..,SKIN,ALLQUADRANT);
        break;
        case(1):
        Analyse(……..,SKIN,ALLQUADRANT);
        break;
        case(2):
        .
        .
        .
        Analyse(……..,SKIN,ALLQUADRANT);
        break;
        case(48):


}
```

The switch value takes the form of a counter that loops from 0 to 48. A series of offsets and translations takes the counter value and transforms it to a point of origin that represents the cell that is correspondingly numbered.

```
xoffset=i%MASKSIZE;
yoffset=(i-xoffset)/MASKSIZE;
x=xorigin+xoffset*section_width;
y=yorigin+yoffset*section_height;
```

(xorigin, yorigin) is the point representing the origin of the grid structure, or mask, in the image. The top left point.

(x, y) is the point of origin of the cell to be analysed.

xoffset and yoffset represent the row and column that the cell in the mask is located. section_width and section_height represent the dimensions of the cell.

The procedure for the statistical analysis was intended to be complex and provides functionality to allow for a complex analysis. This level of detail was not required.

The model assumes an ideal eye as a model. The centre nine cells represent the pupil of the eye. Moving outwards, the ring of 16 cells represents the iris. The ring of cells outside of this then represents part of the eyelid and the cornea.

The orientation of the eye is of little consequence due to the round symmetry of the pupil.

4.3 The Process

4.3.1 Colour Check

A series of checks is performed on the inner nine cells of the grid.

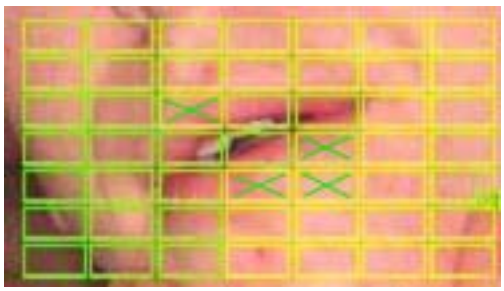


Figure 16: Invalid pupil cells.

These checks are to ensure the likelihood that the centre does indeed contain a red pupil. In the figure above it can be seen that four cells have been marked as invalid. This is too many for an allowance. The check is for a reasonable of amount red energy. It is clear from the figure that this is simply not the case. An allowance is made for cells that are primary white in colour, which could constitute a flash reflection. However too much white and the region is declared a non eye region.

In the event that the red pupil candidate is in fact some sort of marking on the skin, a check is also performed for colours that should not occur in the pupil area, namely checks for yellows and greens and blues. The region will be immediately marked as invalid.

4.3.2 Size Checks

The eye is essentially symmetric in shape. The pupil is inherently round. The check in this case is to determine the ratio of the width of the pupil to the height. An allowance is made for eyes that are partially shut. This allowance however is more directed to the fact that, as the pupil regions become very small, even a few pixels in the difference between the width and height can result in a large ratio. The allowance allowed is that the ratio of the width to the height is no greater than two to one. Regions are otherwise dismissed.

4.3.3 Edges

The eye is a very complicated region. An edge analysis over the entire grid will reveal that edges are distributed over the majority of cells in the grid. This is due to the edges belonging to the pupil and iris border, likewise edges due to the cornea, eyelids, the glint and eye corners.

This is opposed to a region that could otherwise be a part of the lips, surrounded immediately by skin. The border between the lips and skin would form the majority of the edges in this case.

4.3.4 Colour Transitions

A check is performed to analyse the colour transitions moving outwards from the pupil of the eye. The colour transition should be visually noticeable. The transition is from the red colour of the pupil to the colour of the iris, or cornea or eyelid.

The saturation is the chosen channel to check for changes. Although intensity levels very greatly, there is a large possibility that despite the different colours of the pupil and iris the intensity value could be very similar due to exposure from the lighting of the flash. Similarly the RGB model is not used, as intensity information is included in all channels.

The hue would seem to be the obvious choice but is not used as distorted values can be obtained, primarily regarding colours that are a very pure red. The hue values exist on the edge of the colour cone between 355 degrees and 5 degrees. The

average would then become 180, which will then imply that the colour is turquoise (residing between green and blue). This is not acceptably.

The average value of saturation is first calculated for each cell of the grid. Next the standard deviation is calculated across the 49 cells based on the average value of the saturation for each cell.

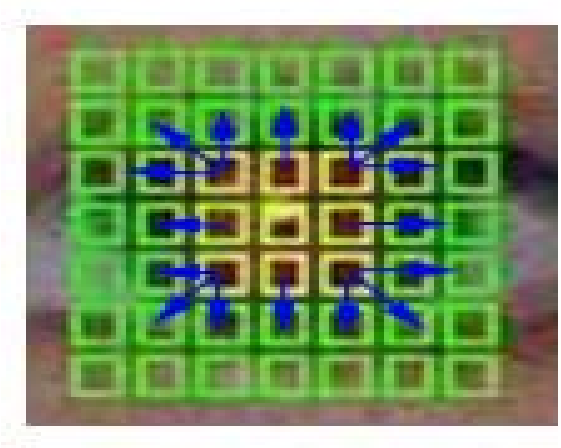


Figure 17: Saturation test.

Tests conducted on images indicate that saturation levels for the nine cells comprising the pupil greatly contrast with the saturation levels of the surrounding cells. Saturation comparisons are performed in the directions specified by the arrows in figure 17. The difference necessary to be considered a valid transition is :

abs(cell1.saturation – cell2.saturation)> alpha*standard_deviation.

Alpha is arbitrary but values greater than .75 work well.
The greater the number of valid differences raises the probability.

The last test is to check that there is a significant variation in colour between the cells surrounding the pupils cells i.e. colour variations from the iris, cornea and skin. The possibility exists that the red region could be a red skin discolouration. The check is basically to ensure that the majority of the surrounding pixels are not simply all skin pixels.

4.3.5 The Cornea Detector

An obvious check would be for the cornea. The concept is simple but execution is not very effective. The cornea has a whitish appearance. It was thus necessary to develop an algorithm that could segment whitish pixels in the vicinity of the pupil. The RGB model was used for this purpose. Pure white on the RGB model is the (255,255,255) vector. The algorithm considers pixels in close proximity to this vector to be whitish in colour. The design worked well for eyes in which the cornea was very evident. There were however to many problems. The cornea is often only visible on one side of the eye, if the eye is turned. The amount of white to be considered part of the cornea was also in question. In distant eyes, the cornea is often accompanied by a red or blue tint in its appearance. These were the main reasons for the abandonment of this option.

5. The Correction Process.

This is the final operation to be performed. There should be relatively few regions, if any, remaining in the queue at this stage. The correction process must correct the colouration of the eyes in a realistic fashion. Any type of redrawing has the potential to appear more unnatural than the already unnatural looking red-eye.

There were two algorithms designed for this purpose. Each algorithm examines each pixel in the region that is overly red.

5.1 Technique 1.

The intensity channel from the HSI model is examined. The premise behind this approach is simple, to rebuild the pupil based on the light intensity at that point on the surface of the pupil. A pupil is fundamentally dark, very dark in colour. It is a hole in the eye. It is the light reflections from it that make it look real in a photo. The light intensity is measured. All pixels that are inherently white and intense in colour are left unchanged. This is the first aspect of redrawing the pupils with a sense of realism.

The eye is spherical in shape. This means that light reflecting of the surface of the eye will be more intense at the point directly facing the source of light. Light

intensities will have different values across the pupil. The value will decrease; the further the point is from the source of light.

The approach to the re-colouring of the eye involves assigning coefficients to each of the three channels of the RGB model. These coefficients are then multiplied by the intensity value at that point on the eye.

5.2 Technique 2.

This solution attempts to decrease the level of the red colour by placing greater emphasis on the green and blue channels of the RGB model. The technique tries to reduce the energy of the red component by replacing the value with an average of the other two components. Since the pupil is dark and blackish in colour, the idea was to perform several changes to the pixel values to ensure that there was no particularly dominant value in a channel. The technique is essentially darkening the pupil colour. The deviation of the red, green, and blue channels provides for a nice variation of colour over the pupil.

5.3 Edge Blending

Once either technique has been used an averaging mask is applied to the pixels on the boarder of the pupil region. The idea is to soften the transition between the iris and the pupil region. The mask simply calculates an average value for a reference pixel based on other pixels in the neighbourhood of the mask.

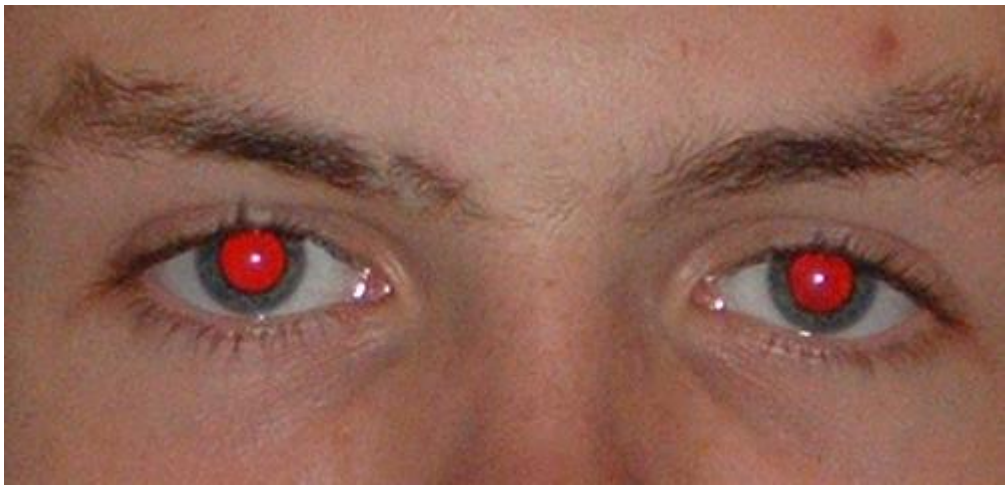


Figure 18 a: A pair of red eyes.

Figure 18 b: The corrected eyes.

5.4 The Shrinking Problem.

In some of the corrected photos it seemed that the pupil seemed unnaturally oversized. This was clearly not the case. A decision was made to try and shrink the pupil in a realistic manner that would make the photo look more appealing. Intel's image processing library, which is incorporated into TIPPS, provided functionality to reduce the size of the image. An image was created representing the pixels in the pupil region. This pupil image was then scaled down by a few percent. The resultant pixels were then placed at the centre of the region that represented the original pupil. The result is that there was now an empty gap around the new pupil area. The idea was to sample surrounding pixels randomly which were not skin coloured and fill the gap with them. Although the concept was good, and did not seem to represent the too many problems, the opposite was of course true. Several factors were the cause for the eventual abandonment of the inclusion of this algorithm.

1. It was a difficult task to determine which pupils should be shrunk. Pupils in the foreground of an image could be largish in size. Pupils in the background, where faces are smaller had pupils that were slightly more dilated and appeared to be the same size as the pupils in the foreground. Clearly some sort of ratio between the iris size and pupil had to be used. Iris measurement proved to be unreliable. Factors such as partially closed eyes, and eyes not facing forward made designing a general algorithm tricky at best.
2. In order to find the edge of the iris between the iris and the cornea, it was necessary to develop an algorithm that could detect the whitish colouration of the cornea. (see statistical analysis)
3. The redrawing of the iris proved to be a difficult task. The iris is multi-tonal and it is not unusual for the iris colour to change from one colour bordering the cornea side to a very different colour bordering the pupil side. Although sampling and colouring was possible, the result was often very noticeable. As the pupil dilates, the iris retreats making part of the iris colour unpronounced. Sampling the colour at random points could lead to a drastic colour difference. A re-colouration of the pupil followed by the shrinking of the pupil region was easier to notice in corrected images as opposed to just the correction of the colour.

6. Evaluation

6.1 Results

The Red-Eye System successfully identified, and performed a correction on the majority of images supplied. The degree to how well corrected pupils appeared varied from picture to picture. The general consensus, however, was that all corrections looked better than the initial red colour. No false detects were found or wrongly re-coloured. The only problem was that several red-eyes were misclassified as being non-eyes over a large number of tests. These eyes were either in the background of the picture or a large distance away from the camera resulting in a very low level of detail. Images of corrected photos have been included on the accompanying CD as is the program for testing.

6.2 Design

The development of the software can be broken into three parts:

1.The automation

2. The detection.

3. The correction.

The design of the software was such that testing and modifications could be easily implemented. The queuing system was designed and tested. This was the first part of the program to be completed, as it is the backbone of the automatic process.

A switch statement is used to apply the different algorithms on the regions. The decision to design the structure in this fashion was the simple fact that this enabled the software to be very easily adapted. Adding a new algorithm is as simple as adding a new case to the switch statement.

Correction took place at an early stage. Although at this stage several non-eye regions were being corrected. Notice was only taken of the changes made to the eyes. It was assumed that false detects would eventually be removed.

6.3 Difficulties.

There were two most notably difficult aspects.

The first was in designing a metric that would identify red-eye regions. The difficulty with this aspect was that eyes in an image could be of any size. The algorithm had to be very general.

Although there is very little code, in the algorithm for the colour correction, the problem was to do it in a realistic manner. Although large eyes were relatively easy to correct, smaller eyes i.e. those in the background of a photo were more difficult to deal with. The problems ranged from there being very little detail, to the fact that the red colouration of the pupil sometimes completely washed out the colour of the iris.

Obtaining good quality pictures containing red-eye was also a problem. Though the principle behind why red eye occurs was known, it did not help when it came to taking photos where the red was purposely induced and conversely where it was not wanted.

6.4 Successes

The procedure has a high success rate. A test was conducted on several individuals based on corrected images. Though it was possible to spot the images that were corrected when attention was specifically drawn to the parts that had been corrected. These tell-tails were in some cases a visible but faded pink rim around the pupil that was corrected or a very washed out residue of the pink colour visible as specks in the corrected pupil region. All corrected images were considered more favourable than the original images.

A computer with a PentiumIV processor running at 1.4Ghz can correct images in approximately 3 seconds.

The program was successfully converted to a console based application and set up to operate via the Internet. A simple web page was set up to allow images to be uploaded to the server hosting the program and images were successfully corrected and returned. The computer acting as a server was used only on a test basis; a more permanent hosting needs to be found.

6.5 Future Work

There are several aspects of the code that could be modified. The statistic description provided for the eye is general and aimed primarily at eyes that detail is clearly visible in. A second model should be designed to target eyes that are relatively small in the image and as such would contain less detail.

An alternative to the second model could be the inclusion of a manual selection tool. This would be ideal for identifying red-eye in very small eyes that are a faded red in colour. This could be an alternative to designing a new metric.

A means to change the sensitivity could also be included. The change in sensitivity would simply be a lowering of the probability that would allow candidate regions to be considered eye regions.

The efficiency could also be adapted. On aspect that could be changed to make the coding more efficient would be the edge image. The edge detection should ideally be constrained to the region that is being analysed as oppose to an edge image of the entire image being created.

Presently the red-eye system operates on a single image at any one time. The provision of a batch facility would allow the user to select several pictures at one time and the software would then operate on the images in turn. This would safe the user the time spent waiting to use the software on each picture at a time.

More tests need to be conducted on the algorithms and indeed some tweaks. Following on from this, a series of fully working algorithms could have the potential to be incorporated into the firmware of some digital camera models.

7. References

[Cardini, 01] D. Cardini. Adventures in HSV Space . April 2001 http://www.beuna.com/articles/hsvspace.pdf

[Sonka, Hlavac and Boyle, 99] Milan Sonka, Vaclav Hlavac, Roger Boyle. Image Processing, Analysis, and Machine Vision, Second Edition. 1999.

[Voila and Jones, 01] Paul Voila, Michael Jones. Robust Real-time Object Detection. 2001. url=citeseer.ist.psu.edu/viola01robust.html

[Efford, 2002] Nick Efford. Digital Image Processing a practical introduction using Java . Addison-Wesley 2000