Video Based Fog Removal

Yuri Parijs
Thesis Number: ICA-3369234
University of Utrecht, Faculty of Science
Utrecht, The Netherlands

Supervisor: dr. Robby T. Tan

May 8, 2012

Abstract

Visibility enhancement in bad weather is important in many applications, including in decreasing road accidents. Current single image visibility enhancement or specifically fog removal methods are capable of increasing the visibility of a fog plagued image. In this master thesis project we attempt to improve a single image method by using tracking information obtained from a video using SIFT flow.

Before starting on enhancing visibility in video we analyse and compare two often cited papers in the field of single image visibility enhancement in bad weather. From the comparison of the two methods we conclude that Tan's method works best for foggy images and Tarel *et al.*'s method is better at images containing haze.

Our method of choice for visibility enhancement in video is Tarel *et al.*'s method, this choice is based on the analysis of the single image methods. The method is fast and should benefit more from additional data obtained from video as it has difficulties with correctly estimating the atmospheric veil for white objects. The atmospheric veil is based on the whiteness of the image, where the whiter an object is the further it is estimated to be. Using the tracking data from SIFT flow we try to detect wrongly estimated objects and correct the atmospheric veil for these objects. We focus on finding white objects that are close to the observer.

Contents

Chapter 1

Introduction

The presence of fog can reduce visibility dramatically. This is dangerous for any kind of traffic, whether it is road, sea or air. The low visibility combined with the velocity of the traffic can lead to serious accidents, that could be prevented by improving the visibility somehow. A possible solution is to have a camera to record the scene and an algorithm to remove the fog real-time and show the fog free scene on a screen. However, currently there are no such algorithms that specifically use a video as the input, although there are multiple methods that can remove fog from a single image.

The goal of this master thesis is to search for information that can be extracted from video that improves visibility enhancement in bad weather. We begin the project by analysing and comparing two often cited papers in visibility enhancement in bad weather. The first method we investigate is Tan's[1] method, the second method is by Tarel *et al.*[2]. As we grow more familiar with these methods and learn their advantages and disadvantages we move on to video-based fog removal. This should yield better results than single image fog removal as video gives more information than a single image. From the analysis of the single image methods we know their flaws and under which circumstances they can fail, allowing us to focus on these flaws and correct them.

1.1 Single image fog removal

In our approach for visibility enhancement in bad weather for video we start by analysing the two aforementioned methods and the fog model. In contrast single image visibility enhancement is a more difficult topic than its video counterpart. This is mainly because the limited information that comes from an image and the difficulty of estimating depth with this little information. There exist multiple methods that are capable of enhancing the visibility with a single image, i.e., the work of Fattal[3]. Our choice of methods to analyse and compare is based on how often they are cited and their appearance in the comparison of other methods. Another reason for our choice are the differences between the two methods. Tan's work seems to be more capable of improving the visibility under thick fog but can take several minutes to compute. Tarel *et al.*'s work has more difficulties with images with thicker fog but can restore an

image within seconds. These methods are well capable of creating a good estimation of the atmospheric veil, however these methods only focus on a single image.

1.2 Video based fog removal

The main advantage of working with video instead of single images is the extra information that video can provide, but that is also the first problem. How can we extract information from video that can help improve the results of single image methods? After exploring two different ways to obtain information from video we move on to applying the found information to improve the results of single image fog removal. Because we are using extra information our result should be better than simply applying a single image fog removal method to each frame. Extracting the video information is time consuming, meaning that the required computational time of our method will be slower than a single image method. We compare the airlight of a single image method and our improved airlight to ground truths manually created with a graphics editing program. We choose to compare the airlight instead of the restored images as the airlight is only dependent on depth. The airlight that we correct is created by applying the information from video and using the intensity information of a single image. The method that we create to enhance visibility in bad weather for video is not fast and is therefore not useful in applications that require fast visibility enhancement, e.g., road sign detection in bad weather for cars. Our method is applicable for applications that process material from security cameras.

Chapter 2

Background knowledge

In this chapter we provide knowledge that is required to fully understand the main chapters of this master thesis. A small part of this knowledge is weather terminology explaining fog and haze. This clears any confusion when a method is said to work well on images containing haze but has problems with foggy images for example. The digital images that are enhanced are all represented by numbers, there are multiple colour models that can be used to represent an image. We explain which colour models are used and what the benefits are of using these models. For the restoration of images plagued by bad weather the atmospheric veil is estimated by the methods we discuss later. As a way to improve the estimation it can be smoothed while maintaining strong edges. The smoothing of the estimation can be done in various ways, we briefly describe several methods in this chapter that are used for this goal. In Chapter 4 we are dealing with video, in order to observe how objects change over time they need to be tracked. SIFT flow is used for tracking objects over time and is compared to Optical Flow from which it is inspired.

2.1 Bilateral filter

A Bilateral Filter is a filter based on Gaussian blur but preserves edges. The Gaussian blur uses the same Gaussian kernel for every pixel of the image as can be seen in Figure 2.2. The kernel is used for averaging the pixels on weight. The Bilateral Filter changes the kernel to conserve edges. The kernels used in a Bilateral Filter are shown in Figure 2.3. The results of applying a bilateral filter to an image of airlight is shown in Figure 2.1

The edges are preserved by taking the change of intensity into account and penalizing large changes. The change in intensity is taken into account because high changes in intensity are observed as edges. By penalizing the pixels with high difference in intensity they affect the smoothing process less leading to maintaining strong edges within an image. Intuitively this can be seen as smoothing over pixels while excluding pixels whose intensity differs significantly. Figure 2.4 shows a one dimensional image containing two edges, the edges are located between pixel 5 and 6 and between 14 and 15. By adding penalty for changes in intensity results like Figure 2.3 are obtained.

Figure 2.1: Before and after results of applying a bilateral filter to an airlight image. On the left is the before image and on the right the after image.

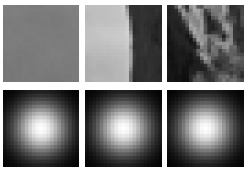


Figure 2.2: On the top are image patches and the corresponding Gaussian kernels used for Gaussian blur are on the bottom.

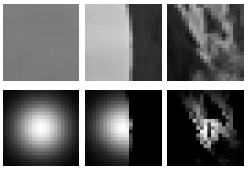


Figure 2.3: On the top are image patches and the corresponding kernels used for Bilateral Filter are on the bottom.

Figure 2.4: One-dimensional image containing two strong edges.

For more information about Bilateral Filters see `http://people.csail.mit.edu/sparis/bf_course/`

2.1.1 Median filter

The Median Filter is another type of filter for smoothening and reducing noise from an image while preserving strong edges. It is a simple filter that uses the median to obtain its result. What the filter does is replacing the value of a pixel by the median of the values of surrounding pixels up to a defined range. In the case of noise this works because the median is unaffected by the outlying values from the noise as long as there is not too much noise. The same goes for preserving edges, however thin lines on a plain background will not be preserved. A Median Filter only requires a window size as input whereas a Bilateral Filter also requires parameters for the change in intensity.

2.2 Markov random fields

Also known as a Markov Network a Markov Random Field (MRF) is an undirected graphical model that specifies factorization and a set of conditional independence relations in probabilistics. In this masters thesis the use of Markov Random Fields for smoothing of an image is replaced by a bilateral filter.

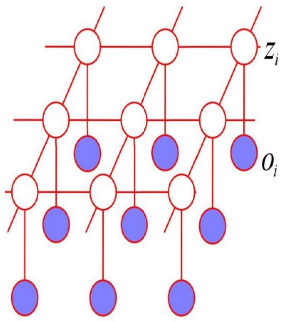


Figure 2.5: Representation of a Markov Random Field. The blue nodes are observed, the white nodes are latent variables.

The model can be imagined as existing out of two layers, a latent layer and an observed layer. In Figure 2.5 the blue nodes represent the observed nodes and the white nodes are latent variables. Each node in the observed layer is only connected to its corresponding latent variable, while the latent variables are also connected to their neighbouring latent variables. The model is factorized based on cliques which are sets of mutual connected nodes. Due to the factorization on cliques the complexity only increases slightly when a set of nodes is added to the model.

In the model two types of cliques are encountered, the first clique is in the form of $\{z_i, o_i\}$, where z_i is a latent variable and o_i an observed node. This clique describes a correlation between the two variables and can be controlled by an energy function that should give a small value for correct values of z_i

with respect to o_i. The collection of these cliques is known as the data term. The other type of clique is in the form of $\{z_i, z_j\}$, where z_j is a neighbouring node of z_i. Similar to the previous clique a smaller energy value is assigned when the two nodes are more similar. This constrains the smoothness and is therefore known as the smoothness term or prior term.

2.2.1 Graph cuts

A solution with high probability of a MRF model can be found efficiently by applying a graph cuts algorithm on the model by interpreting it as an energy minimization problem. This minimization problem can then be changed to a maximum flow problem that is solved with a maximum-flow/min-cut optimization. Such an optimization aims to maximize the flow which results in a minimum cut according to the max-flow min-cut theorem. The term graph cuts is derived from the cuts the optimization makes in the graph but applies specifically to models that apply the max-flow/min-cut algorithm.

2.3 Colour model

Representing images on a computer requires a colour to be converted to a number or a set of numbers because a computer can only interpret numbers. The purpose of colour models is to convert colours to numbers. In this master thesis we work with two colour models. The preferred model is the RGB model, this model represents colours by the colour channels red, green and blue. The RGB colour model is easy to understand. Colours can be created by adding or subtracting from a colour channel. The red channel controls the amount of red. When the red channel is at its highest possible value the colour has the maximum amount of red in it. When it is at its lowest possible value the colour has no red in it at all. In this colour model the colour black is created by having all three channels at their lowest possible value while having all three channels at their maximum value gives the colour white. The opposite of this additive colour model is the CMY model which creates colours by subtracting the colours cyan, magenta and yellow from black. Because the CMY model is the opposite of the RGB model we can derive how to create the colours cyan, magenta and yellow using the RGB colour model. The colour yellow for example is created in the RGB model by setting the colour channels red and green to their maximum while keeping the blue channel at its minimum. The colour model can be depicted as a cube as shown in Figure 2.6.

A different colour model is the HSV model, this model also uses three channels to describe a colour. In this colour model hue, saturation and brightness each have their own channel. This colour model allows us to easily read the intensity of a pixel which is our reason to use this colour model. Figure 2.7 shows the colour model represented as a cone.

2.4 SIFT flow

The SIFT flow algorithm proposed in the paper 'SIFT flow: Dense Correspondence across Scenes and Its Applications'[5] can generate the flow field of two

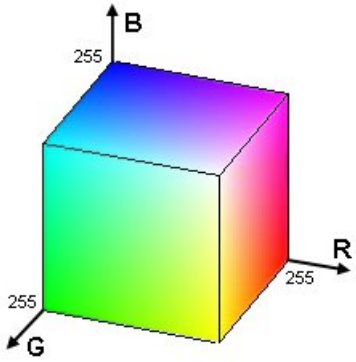


Figure 2.6: A cube representing RGB colour space.

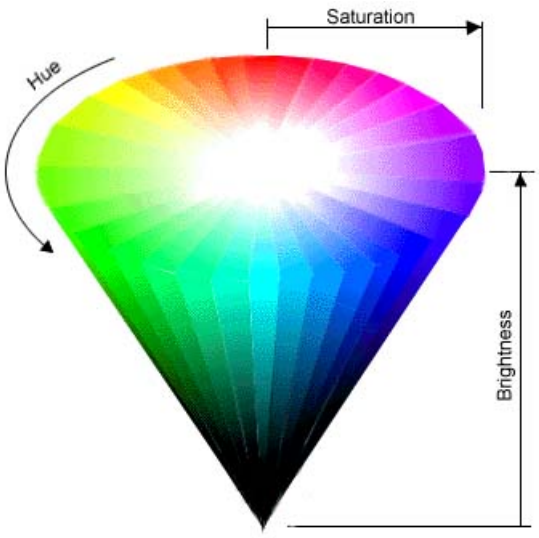


Figure 2.7: A cone representing HSV colour space.

images that describes the movement of the objects between the images. As mentioned in their paper SIFT flow can be used for multiple types of applications. Some examples are motion field prediction from a single image, motion synthesis via object transfer, satelite image registration and face recognition. In this master thesis SIFT flow is used for tracking objects in a video, in Section 4.1.2 an experiment is conducted to evaluate the tracking capabilities of SIFT flow.

Inspired by Optical Flow methods SIFT flow differs from these methods by matching SIFT descriptors instead of brightness patterns. The algorithm starts off by creating a dense SIFT descriptor, this is done by creating a SIFT descriptor for every pixel in the image. The dense SIFT descriptors can be visualized by mapping the top three principal components of SIFT descriptors from a set of images to the principal components of the RGB space. The dense SIFT descriptors, or SIFT images, are then used for matching. The matching technique is based on Optical Flow and is therefore very similar. A major difference between the two methods is the search window size which is much larger for SIFT flow increasing the matching complexity. This problem is solved by using a coarse-to-find matching scheme. In the paper they mention that the basic idea behind this scheme is to roughly estimate the flow at a coarse level of image grid and then gradually propagate and refine the flow from coarse to fine. An example of the input images and the resulting SIFT image describing the flow between the input images is shown in Figure 2.8, the colours in the SIFT image describe the velocity and direction of the flow. The direction and the velocity, which is normalized, can be read from the colour wheel pictured in Figure 2.9. An object that does not or barely moves would be represented with white or a very bright colour.

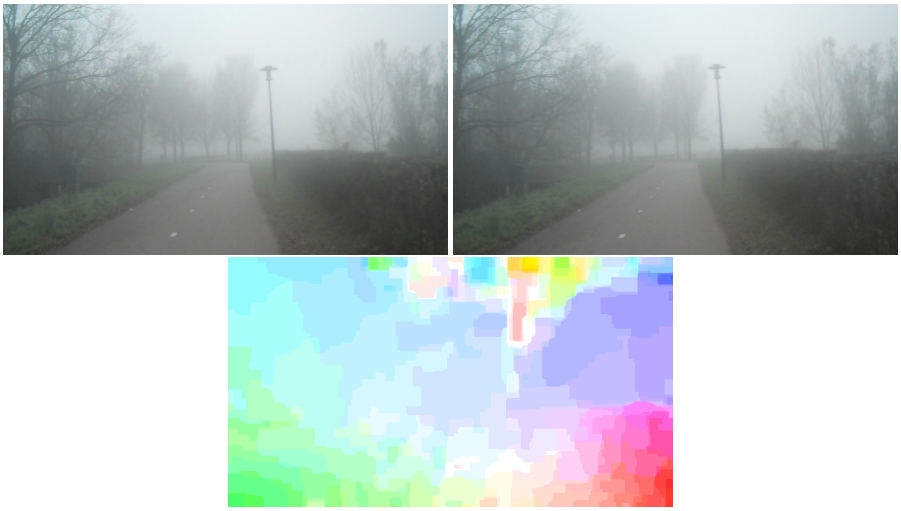


Figure 2.8: The two top images are the required input images, the bottom image is the flow image created using SIFT flow.

The images that we provide as input contain fog or haze. When the visibility of an object is too low to visually see we expect that SIFT flow has an ad-

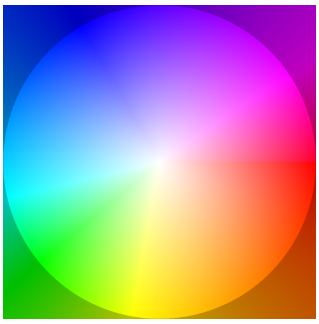


Figure 2.9: Colour wheel representing direction and velocity. A white pixel, which is found in the center, translates to no movement.

vantage over Optical Flow. This is because it SIFT flow uses SIFT descriptors which should be more distinctive than the brightness patterns Optical Flow uses. This is the main reason for choosing SIFT flow over Optical Flow. Even though the analogy of the authors of the SIFT flow papers states that Optical Flow should be preferred when there is dense sampling in time and SIFT flow when there is dense sampling in the space of all images. In Section 4.1.2 we experiment with the tracking capabilities of SIFT flow and find that the algorithm works well for images containing fog. We have no results of Optical Flow on foggy images. If Optical Flow would be successful at tracking objects under bad weather like fog and haze it is possible to replace SIFT flow with Optical Flow as we are merely interested in the object displacement between frames.

2.5 Bad weather: fog and haze

Bad weather like fog and haze lower visibility, this is a direct problem for people partaking in traffic. The lowered visibility can be the cause of accidents in traffic. Bad weather can also cause problems for surveillance cameras as makes the cameras view less without enhancing software.

In this thesis we intend to improve the visibility in videos that are affected by fog. The international definition of fog declares that fog is a hydrometeor consisting of a visible aggregate of minute water droplets (or ice crystals) that reduces the visibility below one kilometre. A hydrometeor is any product of condensation or sublimation of atmospheric vapor. In contrast haze reduces the visibility between two to five kilometres and is the product of fine dust or salt particles dispersed through a portion of the atmosphere. These particles are so small that they cannot be felt or individually seen with the naked eye.

In our comparison of single image methods we include images that are affected by fog and haze. We do not detect fog or haze automatically in this project. The paper "Fog Detection System Based on Computer Vision Techniques" by Bronte *et al.*[8] describes a *foggy detector* that can detect at least 3 states of fog: no fog, moderate fog and high fog. This detector is based on the observation that foggy images are blurrier and have lower contrast than sunny ones.

Definitions according to National Snow and Ice Data Center (nsidc.org).

Chapter 3

Single image fog removal

Before starting on video-based fog removal we take a look at single-image fog removal. The general idea is that if we can remove the fog from one image, we can also do it for a sequence of images. If we are to use or improve a methods we first require a better understanding of the method, this knowledge is later used in this thesis for restoring weather degraded video. We will experiment with two papers, analyse them and try to improve them. The papers we discuss are often cited and well known in the field of single image fog removal. The first paper is "Visibility in Bad Weather from a Single Image" by Tan[1], which tries out values for the airlight to maximise the contrast. The other paper is "Fast Visibility Restoration from a Single Colour or Gray Level Image" by Hautire and Tarel *et al.*[2] who use the whiteness of the image to estimate the airlight.

3.1 Fog model

Fog is a cloud near the surface of the Earth, it is a substance that exists out of water droplets. As light travels through fog part of the light is absorbed, the amount of light absorbed depends on the substance through which the light travels and the absorption coefficient. According to the Beer-Lambert[7] law there is a logarithmic dependence between the transmission of light and the product of distance and the absorption coefficient.

$$\mathbf{I}(x) = \mathbf{L}_\infty \boldsymbol{\rho}(x) e^{-\beta d(x)} + \mathbf{L}_\infty (1 - e^{-\beta d(x)}) \tag{3.1}$$

The fog model used in this thesis is a common used model when dealing with the removal of fog. The model is shown in Equation (3.1). The logarithmic dependence of the Beer-Lambert law is in both the direct attenuation, $\mathbf{L}_\infty \boldsymbol{\rho}(x) e^{-\beta d(x)}$, and the airlight, $\mathbf{L}_\infty (1 - e^{-\beta d(x)})$. The parameter $\mathbf{I}(x)$ in the equation is the intensity of the image that suffers of fog. The direct attenuation exists out of airlight colour, $\mathbf{L}_\infty$, and $\boldsymbol{\rho}(x)$, which is the colour reflectance of the object in the image. $e^{-\beta d(x)}$ reduces the direct attenuation as the distance increases, at distance zero it is cancelled out having no effect, but as the distance increases it darkens the direct attenuation until it becomes zero. The same goes for the airlight, but because it is one minus $e^{-\beta d(x)}$ it has no effect when the

distance is zero, as the distance increases its effect also increases going up to a maximum value of $\mathbf{L}_\infty$. This model allows the observed image ($\mathbf{I}(x)$) to range from a fog free image to an image only containing fog while staying true to the Beer-Lambert law that accurately describes the absorption of light. The fog removal methods we discuss later in this chapter both use this model as their basis. They adapt it based on their observations because in its current state solving the model for $\mathbf{L}_\infty \boldsymbol{\rho}(x) e^{-\beta d(x)}$ with only $\mathbf{I}(x)$ is an ill-posed problem. It should be noted that the model applies for objects by day and that sky has to be visible.

In this model we start off knowing only the input image, $\mathbf{I}(x)$, if we want to obtain $\mathbf{L}_\infty \boldsymbol{\rho}(x)$ we need to find $\mathbf{L}_\infty$ and $\beta d(x)$. There is no possibility to disciminate between β and $d(x)$, therefore we will focus on finding only the depth and consider β as a scalar. The combination of β and $d(x)$ is referred to as the airlight. In Section 3.3.1 and Section 3.3.2 two different methods to obtain the airlight are explained and in Section 3.2.1 two methods to compute the airlight colour are explained.

3.2 Whitebalance

The lightning conditions under which a photograph is taken cause a colour cast. The tint of the colour casts depends on the colour temperature of the light source. Light sources with a low colour temperature, like an incandescent light bulb, cause a red/amber colour cast, as the colour temperature rises to a higher colour temperature, like the sun, the colour cast is blueish. The colour cast in a photograph can be removed by performing white balance, also known as colour balance or neutral balance. By removing the colour cast the original colours are restored making the image look more natural and the colours more distinctive. A higher distinction between colours can improve visibility, but these improvements are not significant. The main motivation for performing white balance is the restoration of the colours.

$$\alpha_c = \frac{L_{\infty c}}{L_{\infty r} + L_{\infty g} + L_{\infty b}} \tag{3.2}$$

$$I'_c = I_c(x)/\alpha_c \tag{3.3}$$

To perform whitebalance the illuminant must be determined, for now we assume that the illuminant is known, in Section 3.2.1 we explain how to obtain the illuminant, otherwise known as the airlight colour. From the airlight colour, $L_{\infty c}$, light chromaticity is computed following Equation (3.2). The light chromaticity, α_c, is then used to correct the colour cast of the image by dividing the original colours of the image, $I_c(x)$, by the light chromaticity as shown in Equation (3.3), this results in the colour balanced image denoted as I'_c.

sources: wikipedia: Colour balance, Colour temperature and Colour Cast.

3.2.1 Airlight colour

A significant part of white balance is knowing the airlight colour, there are several ways to compute the airlight colour and we will look at two ways to find it.

Figure 3.1: An outdoors taken photograph, due to the high colour temperature there is a bluish colour cast. After performing whitebalance the colour cast is removed.

The exponential function in the fog model gives us a clue for finding the airlight colour. The distance to the sky is very large and can be said to be infinite, resulting in an exponential function of minus infinity which equals to zero. Filling the equation out for an airlight of ∞ we find that the airlight colour is the same as the observed intensity as can be seen in Equations (3.4) to (3.6). In this set of equations we first show the fog model in Equation (3.4). Because the exponential function of minus infinity equals zero, we replace the function with zero in Equation (3.5). The multiplication with zero in the direct attenuation cancels out the other variables. This leaves us with only the airlight, which is $\mathbf{L}_\infty$ multiplied by one. This turns Equation (3.5) into Equation (3.6). Now we can claim that the sky has the highest intensity, as long as there are no saturated pixels in the image. We can claim this because the direct attenuation can only equal $\mathbf{L}_\infty$ when $\boldsymbol{\rho}(x)$ is pure white and the distance to the object is zero. Nor can it be larger than $\mathbf{L}_\infty$ because $\boldsymbol{\rho}(x)e^{-\beta d(x)}$ can not be larger than one.

$$\mathbf{I}(x) = \mathbf{L}_\infty \boldsymbol{\rho}(x) e^{-\infty} + \mathbf{L}_\infty (1 - e^{-\infty}) \tag{3.4}$$

$$\mathbf{I}(x) = \mathbf{L}_\infty \boldsymbol{\rho}(x) * 0 + \mathbf{L}_\infty (1 - 0) \tag{3.5}$$

$$\mathbf{I}(x) = \mathbf{L}_\infty \tag{3.6}$$

Assuming that there are no saturated pixels and no light sources in the image we can find the airlight colour by searching for pixels with the highest intensity. Simply finding the pixel with the highest intensity is rather error prone, to make it more robust we have two options: we can take the average of the pixels with the highest intensity; or we can take the colour of a patch with

Figure 3.2: White balance by highest intensity with different p. Starting from left, original, p=1, p=5 and p=25.

the highest intensity. Both options require a parameter to be set which is dependent on the image size. The option that looks at the pixels with the highest intensity will take the colour at percentile p, in Figure 3.2 are the results when changing p. Increasing the value p will give an airlight colour with lower intensity. The benefits of using a window is that it reduces the effect of noise in the image, this is achieved by using the mean of the window. In Figure 3.3 are results showing the effect of changing the window size.

3.3 Airlight and restoration

The fog model describes the airlight as $\mathbf{L}_\infty(1 - e^{-\beta d(x)})$. In the previous section is explained how to obtain the airlight colour, with the airlight colour known there are still two unknown variables. As stated in Section 3.1 it is not possible to discriminate between these two variables, therefore we will consider them as one variable. Even with this single unknown variable, besides $\boldsymbol{\rho}(x)$, solving for $\boldsymbol{\rho}(x)$ is an ill-posed problem.

In the following subsections two methods for finding the airlight from a single image are studied. When the airlight and the airlight colour of the image are found the restored image is obtained by plugging the values into the fog model.

3.3.1 Tan's implementation

Tan's approach starts off by rewriting the fog model in terms of chromaticity to Equation (3.7) by using Equations (3.8) to (3.10) and Equation (3.2).

$$\mathbf{I}(x) = D(x)e^{-\beta d(x)}\gamma(x) + A(x)\alpha \tag{3.7}$$

$$\gamma_c = \frac{L_{\infty c}\rho_c}{L_{\infty r}\rho_r + L_{\infty g}\rho_g + L_{\infty b}\rho_b} \tag{3.8}$$

Figure 3.3: White balance by patches with different window sizes. Starting from left, original, window=7, window=9, window=25.

$$D(x) = L_{\infty r}\rho_r(x) + L_{\infty g}\rho_g(x) + L_{\infty b}\rho_b(x) \tag{3.9}$$

$$A(x) = (L_{\infty r} + L_{\infty g} + L_{\infty b})(1 - e^{-\beta d(x)}) \tag{3.10}$$

Next white balance is incorporated into the model making it invariant to the airlight colour. In Section 3.2 is explained that white balance is performed by Equation (3.11). We can rewrite Equation (3.11) to Equation (3.12) by writing out I'_c. Equation (3.13) is used to divide $D(x)e^{-\beta d(x)}$ by α in Equation (3.12).

$$I'_c = I_c(x)/\alpha_c \tag{3.11}$$

$$I'_c = D(x)e^{-\beta d(x)}\gamma'_c(x) + A(x) \tag{3.12}$$

$$\gamma'_c(x) = \frac{\gamma_c(x)}{\alpha_c} \tag{3.13}$$

Solving for $D(x)\gamma'$ results in Equation (3.14).

$$D(x)\gamma'(x) = (\mathbf{I}'(x) - A(x)\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix})e^{\beta d(x)} \tag{3.14}$$

$$e^{-\beta d(x)} = \frac{\sum_c L_{\infty c} - A(x)}{\sum_c L_{\infty c}} \tag{3.15}$$

However, the model still contains the unknown variables $\beta d(x)$. These variables can be isolated from the airlight turning Equation (3.10) into (3.15). In this form $\beta d(x)$ is dependent on the known variables L_∞ and the unknown variable $A(x)$, meaning $D(x)\gamma'$ can be computed by estimating $A(x)$.

Based on the observations that the output image should have better contrast than the input image and that the airlight is smooth except along depth

discontinuities Tan maximises $A(x)$ for contrast. This can be done by quantitatively defining image contrast. The gradient of the image can be used for defining image contrast. Maximizing the contrast is done by trying out multiple values for $A(x)$ and comparing the results of the gradient locally in order to find the value for $A(x)$ that maximizes the contrast of the current patch. By doing this all changes in contrast are exaggerated making minor changes more obvious and better visible. In the case of a foggy skyline the RGB values of far away buildings are almost identical to those of the sky, however the minor changes are exaggerated by the algorithm and becomes better visible.

The different values for $A(x)$ are placed in a Markov Random Field with the corresponding gradient value as cost. This is done as a smoothness constraint on the airlight, because the airlight is smooth except along depth discontinuities. After the MRF is built graph cuts is used to optimize $A(x)$. Once $A(x)$ is known Equations (3.14) and (3.15) are used to restore the image.

Adaptation

Building the Markov Random Fields and using graph cut is computationally rather expensive. It is used to smoothen out the airlight when neighbouring values are similar while keeping strong edges where there is a large difference in airlight. This description of the reason to use MRF is the same as the description of what a bilateral filter does. As a way to speed up this smoothing process we replaced MRF with a bilateral filter. We compared the airlight images created with a bilater filter with those we created with MRF, these images are shown in Figure 3.4. All of our results in Appendix B are created using the bilateral filter. Our results are similar to Tan's results which implies that our airlight images are similar as well to which we can conclude that replacing MRF with a bilateral filter gives similar results while speeding up the process. One of the parameters of the bilateral filter is the window size, this variable depends on the size of the image and the type of image. An aerial image might require a larger window size than a non-aerial image. Requiring different window size seems a drawback of using a bilateral filter over MRF, however it does allow some control over halo. Halo is a side effect of the smoothing process where a smaller window size means less room for halo.

The implementation of this method had some difficulties. As a result our implementation still slightly differ from Tan's results. We required our own implementation of the method because the original code is not released, in Appendix B we compare our results with released results of Tan's implementation. Visually there are minor differences between the results but overall the results are very similar. We have already stated that the replacement of MRF for a bilateral filter would not affect the result much but would increase the speed of the algorithm. When we compare our results with actual results of Tan's method we can see that our implementation with a bilateral filter obtains similar results.

Conclusion

The two observations on which Tan's implementation relies give good results both for foggy images and images containing haze. The drawbacks of the implementation are the speed of the algorithm and the restored images tend to

Figure 3.4: Top left: input. Bottom left: input after white balance. Top middle: airlight using MRF. Bottom middle: results using MRF. Top right: airlight using a bilateral filter. Bottom right: result using a bilateral filter.

be over-saturated. The speed of the algorithm is slowed down by trying out a large set of possible values of airlight for each pixel and building a Markov Random Field for this. Even when replacing the MRF by a bilateral filter the algorithm is still slower than Tarel *et al.*'s implementation. The restored images tend to be over-saturated because we are maximizing the contrast, in some cases an incorrect value for $A(x)$ is chosen because it results in a higher contrast.

3.3.2 Tarel *et al.*'s implementation

One of the disadvantages of the algorithm discussed in the previous section is its speed. In this section we will discuss an algorithm which is faster and claims to have similar or in some cases better results. The computational time is less than a second for a 598x396 image on a 3 GHz dual core processor, the time required is linear in the amount of pixels.

$$\mathbf{I}(x) = R(x)(1 - \frac{V(x)}{L_\infty}) + V(x) \tag{3.16}$$

$$R(x) = \frac{\mathbf{I}(x) - V(x)}{1 - V(x)} \tag{3.17}$$

The fog model Tarel *et al.* uses is based on the same model as Tan uses. As it is not possible to separate the depth and extinction coefficient the atmospheric veil is introduced $V(x) = L_\infty(1 - e^{-\beta d(x)})$, where L_∞ is the airlight colour. The rewritten fog model is shown in Equation (3.16), $\mathbf{I}(x)$ is the input image and $R(x)$ is the restored output image. Due to whitebalance in the pre-processing stage $L_\infty = [1, 1, 1]^T$, allowing the equation to be rewritten to Equation (3.17) which isolates $R(x)$.

Inferring the atmospheric veil

The fog model has been rewritten to an equation containing three variables of which only one is known, $\mathbf{I}(x)$. The variable $V(x)$ is the missing link for knowing $R(x)$. In this section $V(x)$ will be inferred based on two constraints: the values are positive, so $0 \leq V(x)$ and $V(x)$ being pure white, can not be larger than a component of $\mathbf{I}(x)$. The whiteness of the image, $W(x)$, is introduced for the latter constraint and is defined as $\min(\mathbf{I}(x))$. When the airlight is pure white the atmospheric veil adds whiteness to the image, the amount of whiteness added depends on the depth of the object. Because the whiteness gives information about the depth and the depth is proportional to the atmospheric veil it is possible to base the atmospheric veil on the whiteness of the image. The mentioned constraints can be formulated as Equation (3.18).

$$0 \leq V(x) \leq W(x) \tag{3.18}$$

$$A(x) = \text{median}_{s_v}(W)(x) \tag{3.19}$$

$$B(x) = A(x) - \text{median}_{s_v}(|W - A|)(x) \tag{3.20}$$

$$V(x) = \text{max}(\text{min}(pB(x), W(x)), 0) \tag{3.21}$$

Besides the constraints as stated in Equation (3.18), the atmospheric veil is smooth except along depth discontinuities. Using a median filter, $W(x)$ can be smoothened while keeping edges. The smoothened version of $W(x)$ is stored in matrix $A(x)$. Equation (3.19) shows this first step. The next step is preventing contrasted texture to be affected too much, this is done by subtracting the local standard deviation of $A(x)$ from $W(x)$. This operation is stored in matrix $B(x)$ as shown in Equation (3.20). Finally $V(x)$ is inferred using Equation (3.21). In this equation p is used to control the strength of the restoration. The parameter s_v in Equations (3.19) and (3.20) is the median filters window size.

Experimentation

The effects of p and s_v are best understand by example and can be viewed in Figure 3.5. The variable p is only used in Equation (3.21) and that equation is only used to satisfy the constraints stated in Equation (3.18). p is a scalar for matrix $B(x)$ which can be seen as the atmospheric veil before it satisfies the constraints. Lowering p consequently reduces the atmospheric veil, in Equation (3.17) the restored image is stated as the input image minus the atmospheric veil and the denominator is used for normalization of the restored image. The atmospheric veil is based on the whiteness of the image, therefore less whiteness is removed when p is lowered as can be seen from the top images of Figure 3.5. Choosing a value for p is an iterative process, the resulting image depends on personal taste, however a good starting value lies between 0.85 and 0.95 depending on the type of image that is to be restored.

The bottom two images of Figure 3.5 shows the results when s_v is changed. s_v controls the window size of the median filter used in Equations (3.19) and (3.20). The median filter is used to satisfy the observation that the atmospheric veil is smooth except along depth discontinuities. The downside of using a median filter is that it can smooth the atmospheric veil too much resulting in halo in the restored image. In Figure 3.5 halo can be seen in the bottom right image, for this result a large value for s_v is used while for the bottom left, without halo, a small value is used. Increasing s_v will smoothen over a larger area, also at depth discontinuities, resulting in a smoothened change in depth. The pixels at the border of the fore- and background will have incorrect values for the depth and are restored incorrect. Using a too low value for s_v has the downside of not smoothing enough. Finding the correct value for s_v depends on the image, the simple assumption that a larger image requires a larger value for s_v is safe, but it is not a guarantee.

Different types of images require different type of parameters, the image in Figure 3.5 has better results when a small value for s_v is chosen. In Figure 3.6 the result gets better when using a larger value for s_v, this is the case because it is an aerial photograph and the distances between the buildings is relative small compared to the distance of the camera to the buildings. Besides the relative distances between the objects in the image and the camera the thickness of the fog affects the outcome. Figure 3.7 is another aerial photo but with a much thicker fog than in the previous figure, the algorithm is still able to improve contrast and visibility.

Figure 3.5: Results of Tarel *et al.*'s method with different parameter values. Top left: Original image. Top middle: $p = 0.75$, $s_v = 3$. Top right: $p = 0.95$, $s_v = 3$. Bottom left: $p = 0.85$, $s_v = 3$. Bottom right: $p = 0.85$, $s_v = 15$

Figure 3.6: Results of Tarel *et al.*'s method with different values for the window size. Left: $p = 0.95$, $s_v = 5$. Right: $p = 0.95$, $s_v = 15$.

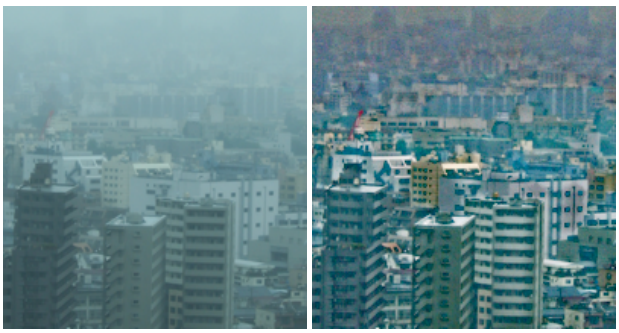


Figure 3.7: Left: Original image. Right: Restored image.

Conclusion

The algorithm is entirely based on simple matrix operations, this makes it a fast method depending on image size only. The downside of this method is the requirement of the input parameters, in our experimentations we showed that a same sized image can require a different value for s_v depending on the image. The scalar variable p is slightly dependent on personal taste, making it difficult to find a good value for it. Basing the atmospheric veil on the whiteness of the image has two drawbacks. The whiteness of an object is assumed to be caused by the atmospheric veil, not taking into account objects that are white from themselves, this can cause these objects to appear darker in the result than they should be. In Figure C.13 of Appendix C this effect can be seen near the windows, in the original image the concrete above the windows is light-grey, in the restored image the concrete has become dark-grey. The other drawback is the lack of contrast enhancement. This results in objects that are visually lost in the fog will still be difficult to detect with the naked eye. This can be observed in Figure C.1 of Appendix C in which the results of Tan and Tarel are compared.

3.4 Comparing both methods

We have discussed two algorithms for the removal of fog from a single image. The difference between the algorithms lies at their cores, Tan's work is based on the gradient while Tarel *et al.*'s work is based on the whiteness of the image. In images where objects are visually lost in the fog Tan's method can improve the visibility of the objects by maximizing the contrast while Tarel *et al.*'s method can not. When dealing with images containing haze the tides turn. Tan's method still visually enhances the image but tends to oversaturate the colours. Tarel *et al.*'s method on the other hand restores these images while keeping the colours in the image realistic. The difference in speed between the two methods also lies at the core of the methods. Computing the whiteness of the image is a simple operation that can be done very fast, the other operations for inferring the atmospheric veil are not difficult either allowing to quickly infer the atmospheric veil and restore the image. Maximizing the contrast is a much heavier operation and causes the method to be significantly slower than Tarel *et al.*'s method.

The core differences lead to different results, in Figure 3.8 for example the airlight of the rock in front is wrongly estimated in Tan's method due to the texture of the rock. In Tarel *et al.*'s result the airlight of the rock is better estimated as it does not try to maximise the contrast locally. However, when working on images with thicker fog Tan's method seems to outperform Tarel *et al.*'s method. Figure 3.9 shows an image of Tokyo in thick fog/smog, both methods enhance the skyline, with the difference that Tan's method shows the sky as white whereas Tarel *et al.* shows it as dark grey. Also note that in Figure 3.8 Tarel *et al.*'s result is more natural while in Figure 3.9 it is Tan's result that is more natural.

The major downside of Tan's method is that it takes several minutes to restore a high resolution image. Resizing an image to a lower resolution reduces the computational time significantly but the same can be said for Tarel *et al.*'s

Figure 3.8: Left: Original image. Middle: Result using Tan. Right: Result using Tarel *et al.*. (*These images were taken from Tarel et al.'s comparative study page*)

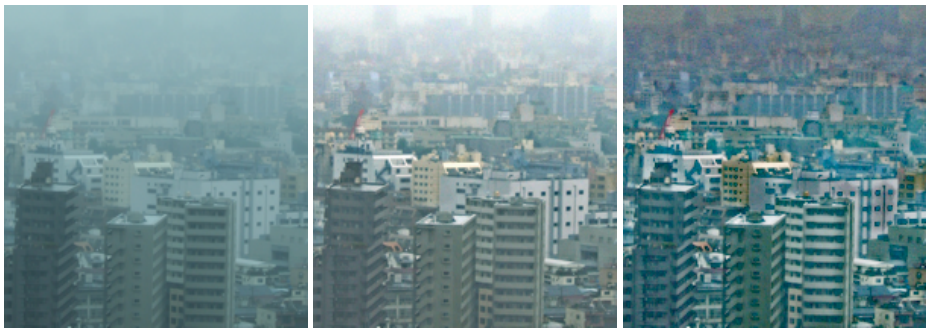


Figure 3.9: Left: Original image. Middle: Result using Tan. Right: Result using Tarel *et al.*.

		Tan	Tarel *et al.*
fog1	mean	3.8	3.2
	std	2.25	1.62
fog2	mean	3.6	0.6
	std	1.89	0.96
fog3	mean	2.8	0.4
	std	1.62	0.51
fog4	mean	3	3.2
	std	2.31	2.3
fog5	mean	2	1
	std	2.4	1.15
fog6	mean	3	0
	std	2.7	0
fog7	mean	3.4	1.4
	std	2.4	1.35

Table 3.1: Results showing the mean error and the standard deviation for five different images.

Figure 3.10: Left: Original image. Middle: Result using our implementation of Tan. Right: Result using Tarel *et al.*.

method. When dealing with video which has a framerate of twentyfour images per second the speed of the restoration algorithm highly affects the speed of the whole algorithm. For this reason we consider Tarel *et al.*'s method when dealing with video's.

The restoration of the images depends on the estimation of the airlight, which is dependend on the depth. In a small experiment we have chosen ten points with increasing depth and compared it to the estimation of the airlight from both methods. The images used in this experiment are displayed in Appendix A and the results are displayed in Table 3.1. From these results it seems Tarel *et al.*'s method outperforms Tan's method significantly, however these results can be misleading. Taking a look at the airlight images of image $fog2$, in Figure 3.10, which favors Tarel *et al.* by a large margin, we can see that the airlight images are actually quite similar, which goes for all airlight images in our experiment. According to the experiment Tarel *et al.* outperforms Tan's method, but not by as much as it seems.

Comparison by survey

Judging the methods on their depth estimation is important as the methods heavily rely on the airlight and the airlight is highly dependable on depth. However, correct airlight estimation does not imply that the visualisation of the image is optimized. To further evaluate the results of the methods we held a survey for a small group of people with a high variety of background knowledge. We expect that the subjects will prefer Tan's method under thicker fog conditions as his method seems to improve visualisation of the image better than Tarel *et al.*'s method when objects are almost completely hidden by the fog. In our set of images we have several pictures with a large view that are hazy. Our expectations are that Tarel *et al.*'s method will be favoured for these images because the method stays close to the original colours while Tan's method tends to oversaturate.

We base these expectations on the differences between the implementations. An image with thick fog that reduces visibility significant will have pixels of objects at different depth with almost the same colour and same intensity due to the strong airlight. Restoring such an image with Tarel *et al.*'s method which uses the intensity of pixels to estimate the depth will have less effect. Whereas Tan's method which tries to locally maximise contrast will be able to improve visibility more. In our dataset we have multiple images with an aerial like view that are hazy. We expect Tarel *et al.*'s method to perform better on these images because the contrast does not have to be maximised to increase the visibility in these images. By correcting these images based on the whiteness the result will stay true to the original images making the result more natural while maximising the contrast can lead to an artificial look and feel.

The images that were shown in the survey are displayed in Appendix C. In the survey the subjects were shown the original images first followed by the results of the two methods in random order. The subjects were not informed which image belonged to which method. After having seen the original image together with the results of the two methods they had to choose which image gave the best result. In the final part of Appendix C the images are grouped together based on towards which method the images are favoured according to the survey.

The group that favours Tan's method is shown in Figure C.14, this group seems to contain most of the images that are more heavily hindered by fog. In the group that favours Tarel *et al.*'s method, shown in Figure C.15, one image stands out as it is more heavily hindered by fog than the others. From the pie chart we can see that only around two third of the subjects prefers Tarel *et al.*'s method for this image. Whereas for the other images three fourth or more of the subjects prefers Tarel *et al.*'s method. Excluding the image with thicker fog the group mainly contains images that are hazy and do not suffer much under the effect of the haze as we had expected. The last group contains the images that gave no clear preference to a single method and is shown in Figure C.16. The first image in this group depicts an aerial view of an urban environment, we would expect from the previous choices the subjects have made that they would prefer Tarel *et al.*'s method for this type of images. In both images the whole scene is clearly visible and the subjects based their choice based on their preference of staying true to the original image or more saturated colours. The other image of this group depicts a black and white image of a road as seen

from a drivers perspective. Once again both methods do a good job at improving the visibility in the image, however there are clear differences between the two images. In Tan's method there is a higher contrast between the road and the markings on the road allowing to better follow the road, whereas Tarel *et al.*'s method better reveals that the road is wet. The subjects that were surveyed personally noted that they would prefer a combination of the two images. The results of the survey seem to hold to our expectations, allowing us to say that the best suited method for an image depends on the scene depicted in the image.

During the survey several subjects gave notes on their thought process for their judgement. Common notes were that the images from Tan's method are sharper but that Tarel *et al.*'s images were better looking. This sometimes led to the choice that Tarel *et al.*'s method was preferred while they could see sharper in Tan's image. Two subjects mentioned on a few of Tan's images that it looked like a painting due to its colourfulness. Another subject thought that the colourfulness of Tan's images would always improve the visibility of an image, but noticed after several images that to be false. One subject had difficulties believing Tan's result to be correct in the first image as he could not see the building in the back in the original image. After closer inspection he noticed that it indeed seem to be correct, however the colours are not correct for the buildings in the back but still chose Tan's result. This same subject seemed to have a profound liking to Tarel *et al.*'s method and managed to choose Tarel *et al.*'s result eleven times out of the thirteen choices. The subjects noted that Tarel *et al.*'s result were more realistic and that Tan's results tended to be over the top.

Correlation between quantitative and qualitative comparison

We have conducted two different experiments in which we compare the airlight of Tan's and Tarel *et al.*'s method. The first experiment focused on depth, with the idea that a better depth estimation gives a better restored image. The follow-up experiment was a survey in which the test subjects had to choose which result they preferred. We will use these results to determine if the method with a better depth estimation restores the image better according to our test subjects. According to our first experiment Tarel *et al.*'s method outperformed Tan's method every time with a single exception. Depending on the image the difference in error ranged from minor to major although the resulting airlight images are very similar in all cases. From the results of the second experiment we can conclude that the better airlight image does not always give the best restored image. However we already mentioned that the airlight images are very similar while the difference in error varies a lot. Therefore we will not determine the outcome based on how often a methods depth estimation is preferred instead we will look at what kind of image yields better depth estimations for each method.

From Table 3.1 we can read which images give the best result for each method. For Tan's method the images that were estimated with the smallest error were in ascending order *fog5*, *fog3*, *fog4* and *fog6* while *fog6*, *fog3* and *fog2* gave the smallest error for Tarel *et al.*'s method. What we are now interested in is what scene these images depict and if these results match those of the survey. As a reminder the images of Table 3.1 are shown in Appendix A. The

images that give a better depth estimation for Tan's method seem to be affected by thicker fog in most cases and they match the results of the survey except for one image, which is equally preferred by both methods in the survey. In Tarel *et al.*'s case the results are more confusing, only one image in the set preferred Tarel *et al.*'s method in the survey and this image does not do well in the depth estimation experiment. The image that gave the best depth estimation for Tarel *et al.*'s method is an image that according to the survey prefers both methods. Following the expectations mentioned in the previous section images *fog6* and *fog7* should be amongst the better depth estimation. Although this holds true for image *fog6* this is not the case for *fog7* which only outperforms two results but it does so with a large margin. The difference between these two results is interesting as there two images are quite similar, they both are aerial views and are affected by haze still allowing descent visibility. The major difference between the images is the actual scene, a suburban view and the view of a landscape. There are also similarities like the small differences in depth among the smaller buildings and trees and bigger depth differences between the top of the taller buildings and the depth jumps between parts of the mountains.

In conclusion the survey supports our expectations while the experiment merely leans towards supporting our expectation. From the comparison of both results we may conclude that a better depth estimation does not guarantee a better restored image assuming both depth estimations are similar. An example would be the presence of halo which is caused by incorrect depth estimation but can improve visibility within the image. Both methods have their benefits and drawbacks, choosing the right method depends on the image that is to be restored. Our expectations can be used as a guide for choosing the correct method.

3.5 Contributions

The restoration of visibility in video is the main goal of this master thesis. To achieve our goal we start by analysing and comparing two often cited papers about single image visibility restoration. Analysing the two papers gives us insight on how visibility can be restored or improved.

In an effort to speed up the implementation of Tan we decided to replace the combination of MRF and graph cuts for smoothing the airlight while containing strong edges with a bilateral filter. In our findings the bilateral filter obtains similar results as would be obtained with MRF as mentioned in Section 3.3.1.

From the analysis of Tarel *et al.*'s method we concluded that this method can have difficulties with estimating the depth of white objects as it bases the estimation on the whiteness of an object. This flaw can cause white objects to be estimated further away, which results these objects to appear darker in the restored image. The depth estimation of a white object could be corrected using information from video.

Both methods improve the visibility in bad weather, but the weather conditions affect the results. The input images of our comparison ranges from weather conditions with thick fog to conditions with haze. At the start of Section 3.4, where we compare the methods using a survey, we state that we expect the subjects will prefer Tan's method when the image contains thick fog while the subjects will prefer Tarel *et al.*'s method when the image contains haze. At

the end of Section 3.4 we conclude that our expectations are correct and that the method to use for a certain image can be based on the weather condition that plagues the image.

Chapter 4

Video based fog removal

In the previous chapter two approaches for fog removal from a single image are discussed. This chapter will focus on the removal of fog from video. Because it is possible to apply single image fog removal methods to video we strife to obtain better results by using information that can only be obtained from video and not from an image.

In this chapter we start by exploring methods to obtain extra information. The usefulness of the new information depends on how it is used, therefore we move on to finding ways to use the obtained information.

4.1 Information from video

A video can be seen as a sequence of images in chronological order with small differences of time between them. The small differences in time between the images assures us that there is a high correlation between sequential images. This correlation is information that can only be obtained from video, and not from a single image. The information might be obvious, but transforming it into a useful form of information is less straight forward.

In this section we discuss two methods to obtain information from video. In the first approach a set of frames is cut to show the change of a row or column in time, how this cut is made and what its result are is explained in Subsection 4.1.1. The second method uses SIFT flow to track all objects in the image, this gives information on the movement of the objects. Because we do not know how well SIFT flow tracking works under the presence of fog we first experiment with it.

4.1.1 Time slice

An image is a two-dimensional representation of the three-dimensional world. As objects move the new position of each objects within the image is likely to be close to its previous location depending on the movement speed of each object. A video contains a large set of images wherein objects move, if we are to place all these two-dimensional images after each other we create a three-dimensional object representing the video. This three-dimensional object can be looked upon from different angles, the default angle would be from the

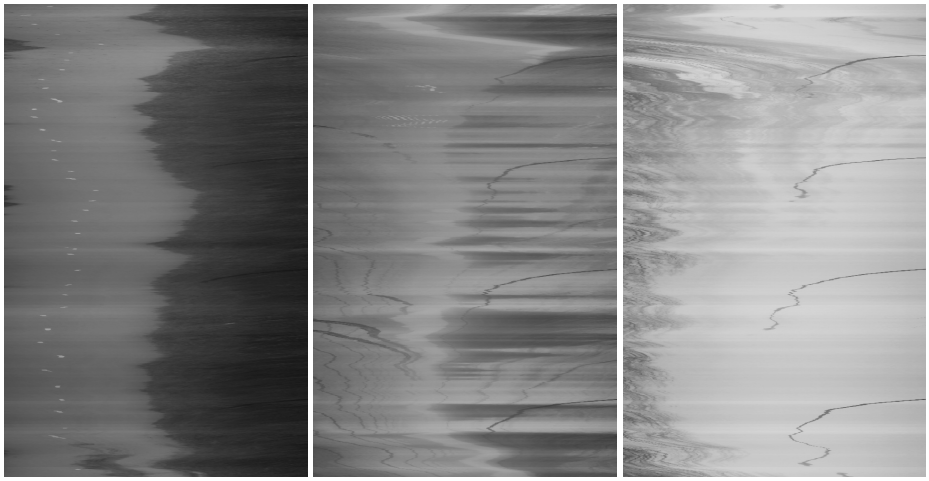


Figure 4.1: Three horizontal slices of airlight at different heights of the same scene. The slices are chronologically ordered from bottom to top.

front showing the input images as layers. Looked upon from the top or sides we can witness the change in intensity of a row or column layer over time. Creating such a three-dimensional object is not a difficult task, but can these views provide information to improve image restoration?

Figure 4.1 shows three horizontal slices of the three-dimensional objects created from a set of airlight images. These slices show the exact same scene at different heights, from left to right the images show the lower part, middle and upper part of the scene. From these images alone it is difficult to understand the scene, however when one knows the scene objects can be recognised. The left image shows the road in light-grey with white dots on it which is the road marking. The image on the right mainly shows air with treetops in light-grey and the curved lines are lampposts that appear as the camera moves closer and goes to the right as the camera passes the lamppost. The last image is difficult to understand without understanding the previous two images, but it is a combination of the two images as the camera bobs alternating between road and sky.

The right-most image in Figure 4.1 shows a good example of objects coming closer to the camera, aside from the lampposts there are also trees visible in the image. According to the fog model, discussed in Chapter 3.1, the intensity of an object reduces as its distance to the camera is reduced. We can observe that this is correct by looking at the trees in the right-most image, in the upper-half of the image the image slowly turns darker as the trees come closer. Measuring these values is prone to errors because it is not possible to tell the path of an object and objects can pop in and out of the slice. These problems make it difficult to use the information from the time slice. In the next section SIFT flow will be discussed for tracking objects, tracking eliminates the problems we have with the time slices but is significantly slower than creating and reading time slices.

The time slices give a good insight of the changes of objects over time, however it is not useful as a tool to improve the restoration of a video sequence.

Figure 4.2: SIFT flow tracking results when iterating on the previous frame. Top left: input. Top Right: frame 52. Bottom Left: frame 80. Bottom Right: frame 160.

4.1.2 SIFT flow tracking

When dealing with video data it is usefull to be able to track objects. The same object at different distances gives information that is not possible to have when dealing with a single image. Tracking objects close to the camera should pose little troubles, however the objects further away can pose a problem due to the fog. Being interested in objects that are far away we experiment with SIFT flow tracking to see how well objects can be tracked in foggy videos.

The experiments have been conducted with the Matlab code from the authors of the paper "SIFT flow: Dense Correspondence across Scenes and Its Applications"[5]. For the first frame four positions have been manually chosen that will be tracked. In the top left image of Figure 4.2 the input positions are colour marked and highlighted with a circle, the other images are the results for tracking the marked positions. The SIFT flow algorithm requires two input images for which it will compute the flow. Figure 4.3 shows the first and the second frame of the video sequence and the flow in the middle. The flow image shows the movement of each pixel by representing each direction and distance with a colour. The colour not only depends on the direction but also on the length of the vector, Figure 4.4 shows the colour wheel from which the colours are derived. Bright colours mean little movement while a more saturated colour implies more movement. The colours do not translate directly to distances but are normalized.

The algorithm does a good job tracking in the fog, however there are problems with occlusion. In Figure 4.5 the object is lost after it is occluded. The cause of this problem is the use of only two images for computing the flow. The information of the previous iteration only exists of the position, there is no information on what kind of object it was. By occluding the branch we were

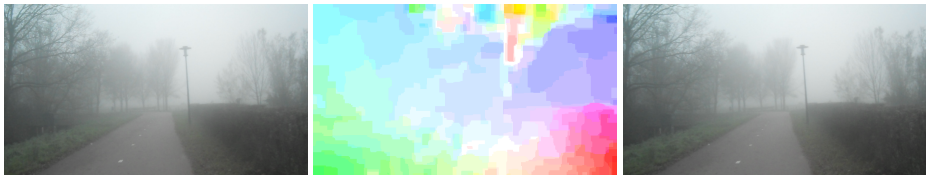


Figure 4.3: In the middle the flow image is shown which is computed from the input images on the left and the right.

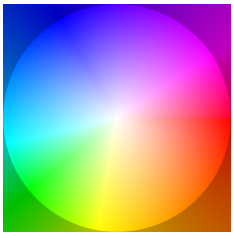


Figure 4.4: SIFT flow colorwheel. Represents direction and relative velocity for the flow images computed by SIFT flow.

tracking the lamppost will have the same position as the branch and will be tracked in the following iterations. A way to deal with this problem is by taking larger steps in time instead of taking two sequential frames and using that information to correct the tracking.

Why does SIFT flow work for Foggy images?

As the name implies, SIFT flow uses SIFT descriptors the compute the flow of an image. SIFT descriptors describe local features in images, these descriptors are invariant to scale and rotation invariant. Being invariant to scale and rotation are not the reasons why SIFT works well in foggy images. The gradient calculation in one of the later phases of SIFT used to make the descriptor rotation invariant takes minor changes in intensity into account, which describes distant objects in fog.

Figure 4.5: SIFT flow occlusion error. The lamppost occludes the part of the tree that is tracked and the object is lost.

4.2 Appliance of video information

The first section of this chapter is dedicated to finding information that can be extracted from video that is beneficial to a fog removal algorithm. We find that the tracking information that can be obtained through SIFT flow as it allows us to observe how an object changes over time which is information that is not possible to have for a single image method. Now that we have this information the question remains how to use it. In this section we search for ways to use the tracking information from SIFT flow.

4.2.1 Narasimhan *et al.*'s contrast restoration method on video

In the paper "Contrast Restoration of Weather Degraded Images" by Narasimhan *et al.*[4] a method is described to restore weather degraded images using two pictures taken from the same position but under different weather conditions. In this section we will try to use this method to improve the visibility of video. It will not be possible to extract frames from a video and use these as input directly, because the input images are required to be of the same position but under different weather conditions. As a camera moves forward the effect of fog or haze is lessened as the distance to the objects is decreased, however is this change in fog or haze density enough for the method to work?

With the use of SIFT flow we can track all objects within the image and observe the change in intensity of the pixel. By replacing the pixels of the input image with the pixels that have been tracked for a certain amount of time we can create an image that has all objects at the same place but with less fog or haze, assuming that the camera in the video is moving forward. Figure 4.6 shows the results of replacing the pixels of the input image with the corresponding pixels of the tracked objects. When more frames are used the difference in contrast is increased, however it also deforms the original shape of the objects more. The results of Figure 4.6 are part of a larger image and are taken from the center of the image allowing us to track all objects up to ninety frames without any object moving out of view. Objects moving out of the visual scope becomes a problem when performing this method on full screen images, Figure 4.7 shows the original input image and the results after tracking objects for thirty frames. After thirty frames deformities are visible. Overall the contrast has not changed significantly. Increasing the amount of frames for computing the result would improve the contrast for further away objects but would also increase the deformities and the amount of objects that can not be tracked until the chosen end frame. When we encounter an object that moves out of the visual scope, we use the values read from the last frame containing the object.

In the first paragraph of this section we ask ourselves if we can create an image with a large enough change in fog or haze density to be useful as input for Narasimhan *et al.*'s method. Based on the input images from Figure 4.7 we expect that the result will not improve the visibility much if any at all. When we provide our implementation of the method with the input images the visibility is not increased in the output. An obvious error are the black artefacts in the image, this is likely the side effect of using two images that are too similar although this effect does not occur when the input consists of two identical

Figure 4.6: Results of replacing pixels of foggy trees with pixels of the same objects several frames later. The top left image is the original image, on the top right the pixels have been replaced after having the objects tracked for 30 frames. At the bottom the objects were tracked for 60 and 90 frames.

images. The slight change in colour is a bug in our implementation as can be seen in Figure 4.9 which shows correct input images together with the output.

From our experiments we conclude that we can not create a usable input image for Narasimhan *et al.*'s method by tracking all objects in the image using SIFT flow. The created image is not different enough from the original to produce an output image with increased visibility. Our created input image does show that the tracking information from SIFT flow is accurate because the image is very similar to the input image, this means that the tracking information can still be useful when used in a different way.

4.2.2 Colour correction using multiple frames

Most of the current existing dehazing techniques compute the airlight from a in single image. The resulting airlight images are relatively reliable to enhance visibility but still contain errors, like halo and in Tarel *et al.*'s case problems with white objects. Dealing with video we have airlight images with a strong positional correlation for neighbouring frames, this correlation might be used to improve the airlight and thus improving the resulting image.

From multiple frames the average object colour can be computed. For a single image the object colour is denoted as $\boldsymbol{\rho}(x)$, the average object colour is denoted as $\boldsymbol{\rho'}(x)$. This averaged object colour makes the transition of an object from one frame to the next more consistent as there is less variation in colour. The change in object colour can have altered the airlight. To correct this possible change the airlight is recomputed from the input image ($\mathbf{I}(x)$) and the average object colour ($\boldsymbol{\rho'}(x)$). This will give a different airlight (α') as it is computed using the averaged object colour instead of the normal object colour. Finally α' can be used together with $\mathbf{I}(x)$ to compute the restored image which will have a high correlation with its neighbouring frames.

Figure 4.7: The bottom image is a re-creation of the top image which has tracked all objects for up to thirty frames and replaced the colour data of the objects with the colour data it found after tracking the objects.

Figure 4.8: Output of Narasimhan *et al.*'s method with the images of Figure 4.7 as input using our implementation.

Figure 4.9: Input and output of Narasimhan *et al.*'s method for correct input using our implementation. The right most image is the output.

Computing ρ'

Considering Equation (4.1) which is a simplified version of Equation (3.1). The airlight colour is not taken into account as it is constant and can easily be computed. The absorption coefficient and depth are merged together into the variable α as it is not possible to distinguish between these two variables.

$$\mathbf{I}(x) = \boldsymbol{\rho}(x)e^{-\alpha} + (1 - e^{-\alpha}) \tag{4.1}$$

Using the input images we only know $\mathbf{I}(x)$, we can obtain $\boldsymbol{\rho}(x)$ and α from an existing single image dehazing algorithm. Knowing all variables we can try to improve the result using other frames which should have the same value for $\boldsymbol{\rho}(x)$. The movement of the camera and/or movement of objects a pixel does not represent the same object in two different frames, this problem can be overcome by tracking the objects. In Section 4.1.2 we experimented with the tracking capabilities of SIFT flow in a fog plagued video and found it working correctly. With the use of SIFT flow we track all objects and average the colour of the objects over a set amount of frames, in our experiments we used five to ten frames. The chose to use five to ten frames because we need enough frames to get a good average, but not too many as there are some tracking errors, like occlusions, that could greatly effect the average object colour.

Computing α'

The average object colour we just computed changed our object colour, this means that our α is no longer correct as the object colour is a product of the input image, which is constant, and α. To make sure the average object colour does not conflict with α we approach α from the average object colour and the input image using the Newton-Raphson method and denote it as α'. The curvature of the function for $\alpha'(x)$ guarantees that there is always only one result. It is possible that a result for $\alpha'(x)$ turns out negative, if this is the case, the value is to be corrected to zero, which is the nearest possible value for $\alpha'(x)$. We mentioned earlier that α is based on the absorption coefficient and depth, the same holds for α' as well, this means that α' should be smooth except along depth discontinuities. We force the smoothness by applying a median filter. We now have α' which is approached from the input image and the more robust average object colour that contains no conflicts like negative values and is smooth. The changes we made to α' affect the average object

Figure 4.10: Top: Dehazing result using Tarel *et al.*. Middle: Result of averaging object colour over several frames. Bottom: Final result.

colour we used, we now compute the object colour using the input image and α' which will be our final result.

Comparison of the different types of object colour.

In Figure 4.10 the same scene is shown three times, the top image is the object colour as computed by Tarel *et al.*. The middle image shows the average object colour created by tracking all objects for several frames using SIFT flow. Finally the bottom image is shown which is our final result created using α' which is approached from the input image and the average object colour and then smoothed using a median filter.

Our goal is to improve the object colour computed by Tarel *et al.*, Figure 4.10 shows that Tarel *et al.*'s result contains some noise, best seen in the sky and on the road. In the preferred case the air and road should remain the same colour or change gradually in colour, which is the case in our final result and even our temporary result (averaged object colour).

The differences between our temporary result and final result are only minor. The final result only corrects possible conflicts like negative depth and smoothens the α'. The object colour of Tarel *et al.* is computed from smoothened airlight, meaning that α' is likely to be smooth already making it redundant to smoothen it again. Instead of smoothing α' and recomputing the object colour from it, it is safe to use the averaged object colour as the final result.

De-noising effect of averaging the object colour

From Figure 4.10 we can observe that improvement of the object colour comes mainly from averaging the object colour over several frames. It seems that by averaging the object colour noise that slightly changes the object colour is removed. To verify if this is indeed the case we go over both images and compute the standard deviation in a small window. We will assume that there is less noise when the standard deviation is smaller. We can reason that this assumption is correct by assuming a small patch of sky, there should be little change in colour values in such a patch. Outlying values like noise would increase the standard deviation. To make sure that the difference in the standard deviation is significant enough to consider that the averaged object colour image is noise free we also compare the results with an image de-noised using the image de-noising tool Neat Image. Figure 4.11 shows an image where the object colour is averaged over several frames and the same scene but de-noised using Neat Image.

In Table 4.1 the average standard deviation using a patch of 5x5 is shown for six images together with the average standard deviation of sixty sequential frames. In all cases the standard deviation of our averaged object colour image ($\boldsymbol{\rho'}(x)$) is significantly lower and similar to the results of the de-noising tool. This means that there is less noise when the object colour is averaged over several frames. To make sure these results are not depending on the patch size Table 4.2 shows the average standard deviation of sixty frames using patches of different size.

According to the results the de-noising tool slightly outperforms our method, which would imply it would be better to use a de-noising tool instead of averaging the object colour over several frames to improve the restored image.

	Input	$\rho'(x)$	Neat Image
Frame 2175	22.72	17.55	18.75
Frame 2185	22.81	17.73	18.87
Frame 2195	24.23	19.09	17.3
Frame 2205	23.64	18.95	16.72
Frame 2215	25.37	19.91	21.14
Frame 2225	25.31	20.97	21.23
Frames 2175 to 2234	24.24	19.47	19.39

Table 4.1: Average standard deviation of frames using a patch of 5x5.

	Input	$\rho'(x)$	Neat Image
3x3	16.83	13.33	13.34
5x5	24.24	19.47	19.39
7x7	28.52	23.48	23.41
10x10	32.26	27.03	27.14
15x15	35.51	30.21	30.57
20x20	37.44	32.20	32.71

Table 4.2: Average standard deviation of sixty frames using patches of different size.

However, our way of comparison is flawed in our favour as it only takes the average standard deviation into account, meaning that a blurring tool would also score well in this comparison. To determine if the de-noising tool is visually better than our method frame 2205 is shown in Figure 4.11. In our results for frame 2205 our method is outperformed by a reasonable margin. When we look at Figure 4.11 the image de-noised with the tool looks more blurred and has lost sharpness at several points, like the branches of the trees. Having obtained similar results in our experiment while having better sharpness in the image our method outperforms the de-noising tool for this type of noise. We have not tested the de-noising effect on other types of noise as it is not our goal to create a de-noising method. We do prefer our method over the de-noising tool as we already intend to use SIFT flow for improving single image fog removal methods. The small overhead of averaging the object colour over several frames is a small cost for de-noising an image and faster than using a de-noising tool.

4.2.3 Rough depth estimation.

The Moon orbits around the barycentre of the Earth and the Moon with a mean orbital velocity of 1.023 km/s, in spite of this incredible speed it seems to barely move if we look upon it from Earth. The average distance of the Moon to the centre of the Earth is 384.403 km, because of this large distance the Moon seems to move a lot slower. Any object that we see up close will seem to move faster than when looked upon from far. Can this observation be used to estimate depth and used to improve the estimation of airlight?

Figure 4.11: Top: De-noised by averaging the object colour over several frames. Bottom: De-noised using tool.

The SIFT flow algorithm allows us to track objects, it gives us the flow of each object which can be seen as a travelled distance. If we have the flow of a sequence of images taken at a regular interval, which is the case when dealing with video, the images can be used as a unit of time. From the distance obtained from SIFT flow together with the time derived from the images the velocity of each object can be computed, this velocity could be expressed as pixels per frame.

In the first paragraph of this section we made the observation that there is a correlation between the observed velocity and the distance of the object. In the previous paragraph we explained how to obtain a velocity map using SIFT flow, this velocity map is our key to a rough depth map. The unit in which the velocity is measured does not allow us to determine how far an object is, it can only roughly estimate differences in distance. Aside from the velocity map we can also create a time map. The time map is closely related to the velocity map, the difference is that it shows how long an object is tracked, up to a certain limit of frames. The longer an object can be tracked the higher the likelihood that it is further away.

Dealing with sky

The depth of the objects can be derived from their movement, the object furthest away should be mainly stationary. In a small experimentation we place two markers in the sky and tracked them for multiple frames. The markers moved around too much to make a significant difference between objects that are far away and the sky. This movement could be explained by the movement of the camera. Not being able to distinct between far away objects and sky this method seems to have a major flaw. As a way to get rid of this flaw we can remove it from the equation. By identifying the sky based on its intensity value it can be set as the furthest away object, while all non-sky objects are ordered by their movement.

Obtaining a distinctive depth map

If the depth of the objects is derived from their current movement the velocity must be computed with a small time frame, preferably as small as possible. The smallest time frame that can be used is the flow image of two sequential frames. According to a small experiment on two images of 640x360 pixels the velocity map is not distinctive enough. The cause of this problem is the movement range at which the pixels move. Over a sequence of 90 frames the maximum movement of objects lies within -14 and 18 pixels on a 640x360 image with most of the movement lying between -4 and 5 pixels as displayed in the histogram of Figure 4.12. To obtain a higher distinction between the movement of the objects a longer time frame can be used. The downside of using a longer time frame is that the velocity gets averaged over that period of time, no longer resulting in the current velocity. To see if multiple frames allow us to distinct better between objects a small experiment was set up to measure how many frames are required to obtain a distinctive velocity map. Because we want the current velocity we prefer to use as little frames as possible. In our small experiment objects became distinctive around 5 to 7 frames as can be seen in Figure 4.13.

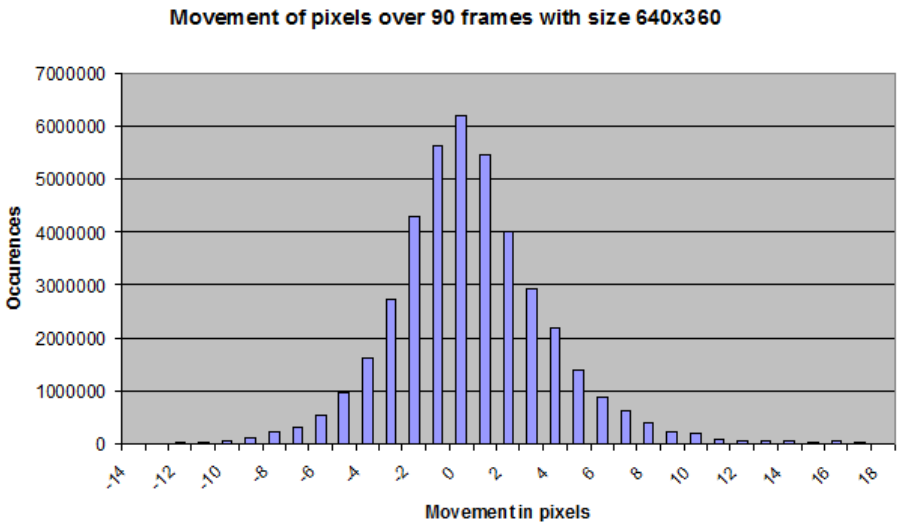


Figure 4.12: Histogram that shows how much pixels move between two frames in X or Y direction over 90 frames in a video sequence of size 640 by 360.

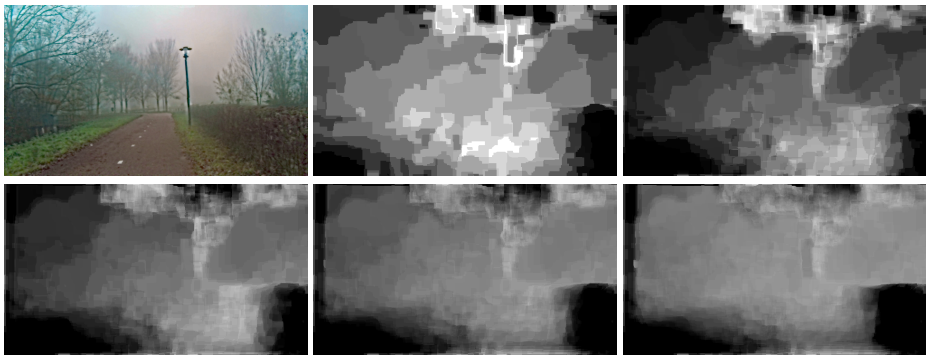


Figure 4.13: Velocity of pixels computed using 1 to multiple frames visualized. From left to right; Input, 1, 2, 5, 7, 10 frames

Computing the velocity of objects using multiple frames introduces a new problem. Not all objects remain for multiple frames within view. Objects lying at the border of the image can move out of the image after the first frame. A possible solution would be to use the velocity the object had until it moved out of view. However due to the change in velocity of the objects over time their value will be either higher or lower because they have been averaged over a smaller amount of frames. Instead we say that objects move out of view are close, as mentioned earlier, pixels with a high intensity are ranked as far away, therefore pixels of far away objects at the border of the image do not get ranked as close.

Applying the rough depth map.

In fog removal obtaining the depth map plays a key role to restoring the image. This depth map should be as accurate as possible to improve the final result. In our case we have estimated a rough depth map, applying this depth map directly would not yield good results. Our depth map can be used to improve the results of a different method. We have chosen to apply our method to the final result of Tarel *et al.*'s single image method. Reasons for using this method are its speed and our expectation of this method to benefit from a depth estimation. We expect this method to benefit from our depth map because it is based on the whiteness of pixels. This means that objects that are white or very bright will always be estimated as far away. Our depth map could correct this as it takes the velocity of the object into account.

It is possible to subtract our depth map directly from Tarel *et al.*'s result. The depth map should be multiplied with a scalar so that the furthest away object will not have its intensity reduced too much. When applying the depth map the estimated depth discontinuities can become obvious in the final result. A blurring function can reduce this unwanted effect, as shown in Figure 4.14.

Manual annotated comparison.

Determining the depth of objects based on their velocity has some flaws, most of these flaws are overcome by small adaptations. Our goal is to obtain a better depth estimation using video information. To determine if we have achieved this goal by using the velocity of objects to estimate the depth we have set up an experiment to compare the depth estimation of our method to the airlight of Tarel *et al.*'s method.

In our experiment we place twenty-three markers in a frame and assign a depth to it manually. As it is not possible to know the actual depth of the markers they are placed into groups of similar depth ranges. Each group is numbered from one to the amount of groups there are. Our ground truth is the set of groups ordered from close to far, within the groups we are not interested in the order. The estimation of the depth of the markers from both methods are also ordered from close to far and compared to the ground truth. For the error we are only interested in how many groups a marker is away from the group it should be in. If the marker is in the correct group, the error is zero, when the marker is one group off, the error is one, when the marker is two groups off, the error is two, etc.

Figure 4.14: Top: Restored image. Left: Depth map directly applied, this introduces unnatural edges to appear because the depth estimation is not entirely correct. Right: Blurred depth map is applied, the unnatural edges are less obvious.

Figure 4.15: Example of points of which the depth is estimated for comparison. When using 8 groups the markers 1 and 11 are in the same group, when using 4 groups the markers 1, 2, and 10 to 12 are in the same group.

		4 groups		8 groups	
		Tarel *et al.*	Ours	Tarel *et al.*	Ours
frame 2175	mean	0.43	0.47	0.87	1.30
	std	0.59	0.59	1.05	1.22
frame 2185	mean	0.43	0.43	0.87	1.13
	std	0.59	0.66	1.10	1.32
frame 2195	mean	0.43	0.43	0.95	1.13
	std	0.59	0.59	1.14	1.32
frame 2205	mean	0.34	0.52	0.78	1.13
	std	0.57	0.66	1.12	1.25
frame 2215	mean	0.43	0.52	1.21	1.39
	std	0.59	0.59	1.20	1.30
Average	mean	0.41	0.47	0.94	1.21
	std	0.59	0.62	1.12	1.28

Table 4.3: Results showing the mean error and the standard deviation for five frames of size 640x360 using Tarel *et al.*'s and our method.

We have measured the mean error and standard deviation for five frames. For each method the mean and standard deviation were computed when using four groups and a second time using eight groups. From our results shown in Table 4.3 we can say that Tarel *et al.*'s results outperforms our method. When using eight groups the difference in performance is more obvious as the accuracy must be better to obtain better results, even with only four depth groups among the markers our method is outperformed.

We mentioned earlier that most of the pixels in our method move within the same small range of -4 to 5 pixels. This problem made it more difficult to distinct between depth ranges, especially because the computed distance uses absolute integer values. This problem was attended to by using multiple frames to compute the velocity. Another method to counter this problem is by using a larger copy of the image, making the measured distance between far away objects more distinct. In a second histogram we made using the same frames but with a higher resolution, shown in Figure 4.16, we can verify that the range in which the pixels move has increased.

To determine if this larger range of movement of pixels also increases the accuracy of the velocity map we perform the exact same experiment on the larger images. From the results, which are shown in Table 4.4, we can notice that the results of the Tarel *et al.*'s airlight remain almost the same, while our method seems to have improved on average. The results of Tarel *et al.*'s airlight are similar to the previous results because there is no extra information, in both experiments the whiteness of the pixels would lie between 0 and 255 as integer values. Another thing to notice is that in a single case our method outperforms Tarel *et al.*'s airlight in the 4 depth groups scenario. In the more accurate 8 groups scenario the tables have turned and our method is outperformed with a reasonable margin. Lastly, in this experiment a result in the 8 groups scenario approaches similar results as Tarel *et al.*'s airlight with the only distinction in the standard deviation between them.

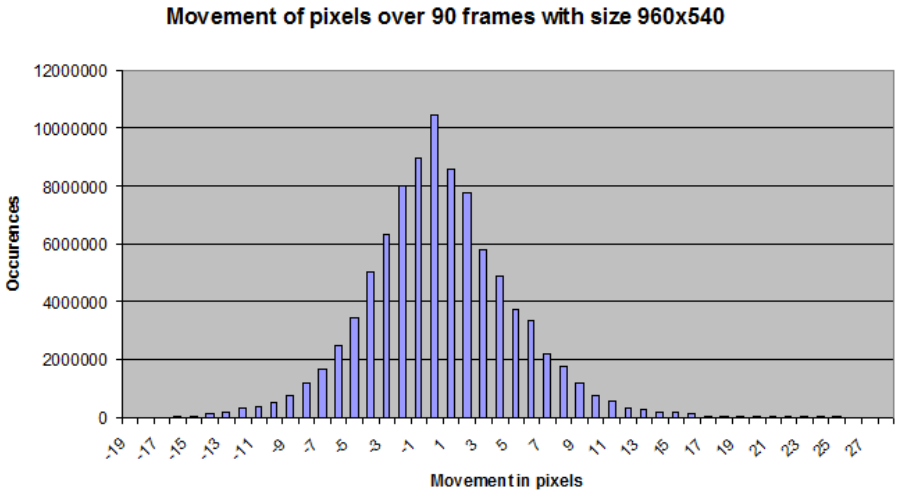


Figure 4.16: Histogram that shows how much pixels move between two frames in X or Y direction over 90 frames in a video sequence of size 960 by 540.

		4 groups		8 groups	
		Tarel *et al.*	Ours	Tarel *et al.*	Ours
frame 2175	mean	0.43	0.43	0.95	1.04
	std	0.59	0.59	1.10	1.18
frame 2185	mean	0.43	0.34	0.87	1.21
	std	0.59	0.57	1.10	1.20
frame 2195	mean	0.43	0.43	0.95	1.13
	std	0.59	0.59	1.14	1.39
frame 2205	mean	0.34	0.34	0.78	0.78
	std	0.57	0.57	1.08	1.18
frame 2215	mean	0.43	0.69	1.08	1.74
	std	0.59	0.70	1.16	1.32
Average	mean	0.41	0.45	0.93	1.18
	std	0.59	0.60	1.12	1.25

Table 4.4: Results showing the mean error and the standard deviation for five frames of size 960x540 using Tarel *et al.*'s and our method.

The movement of objects and pixels over time

According to the manual annotated comparison Tarel *et al.*'s method outperforms our method and gives a better depth estimation. The human perception is not perfect and can therefore be more forgiving, for this reason we wanted to set up an evaluation in which people had to choose which result is better, a video restored using Tarel *et al.*'s method or ours. After viewing the video sequences prepared for the evaluation it was obvious that Tarel *et al.*'s method is better. In our method the differences between the depth estimation between two sequential frames can change significantly resulting in objects changing in intensity over just a few frames. This would indicate a poor correlation between the depth estimations, which is strange as the depth estimations are based on multiple frames. To force a higher correlation between the depth estimations we merged up to five depth estimations together in a similar way as $\boldsymbol{\rho'}(x)$, described in Section 4.2.2. Even after having applied this modification the results were not stable.

Trying to better understand the sudden changes in estimated depth the movement of ten random tracked objects are plotted in Figure 4.17. From this plot it is clear to see that the pixels travelled approximately the same distance for the first thirty frames. This means that if we were to base depth on travelled distance we would require a large amount of frames to make clear distinctions between distances, if at all possible. The plot also clearly shows that the slope of the lines varies a lot, these variations in the slope affect our depth estimation based on velocity. The changes in slope can be caused by multiple sources, perhaps the most obvious cause would be the movement of the camera itself, as the camera moves the objects seem to move. Another cause could be the small integer values we are dealing with. An object moving at one and a half pixel per frame for a certain amount of time will move 2 pixels one frame and only one the other, as it is not possible to use non integer values. Add to this that most objects only move zero to five pixels per frame the effect of rounding the movement will have more significance.

According to our observation that closer objects move faster the lines should be more steep before moving out of view. When an object moves out of view its line will no longer rise and can therefore be easily detected in the plot. In a few cases the lines are steeper a few frames before the object moves out of frame.

Correcting airlight of white objects

In previous attempts we tried to correct images by applying the depth map to the whole image. During the creation of the depth map we took pixels that have a high airlight value and labelled them as far away. Instead of applying the depth map to the whole image we now apply it only to objects with a high airlight value. White objects in the image tend to have a high airlight value labelling them as far away, even when the white object is close. By applying our depth map obtained from movement we try to correct the airlight of white objects.

The first step in this approach is to find the white objects which will be subject to airlight correction. In our experiment we used a naive way to detect white objects, all pixels with an airlight value higher than a certain per-

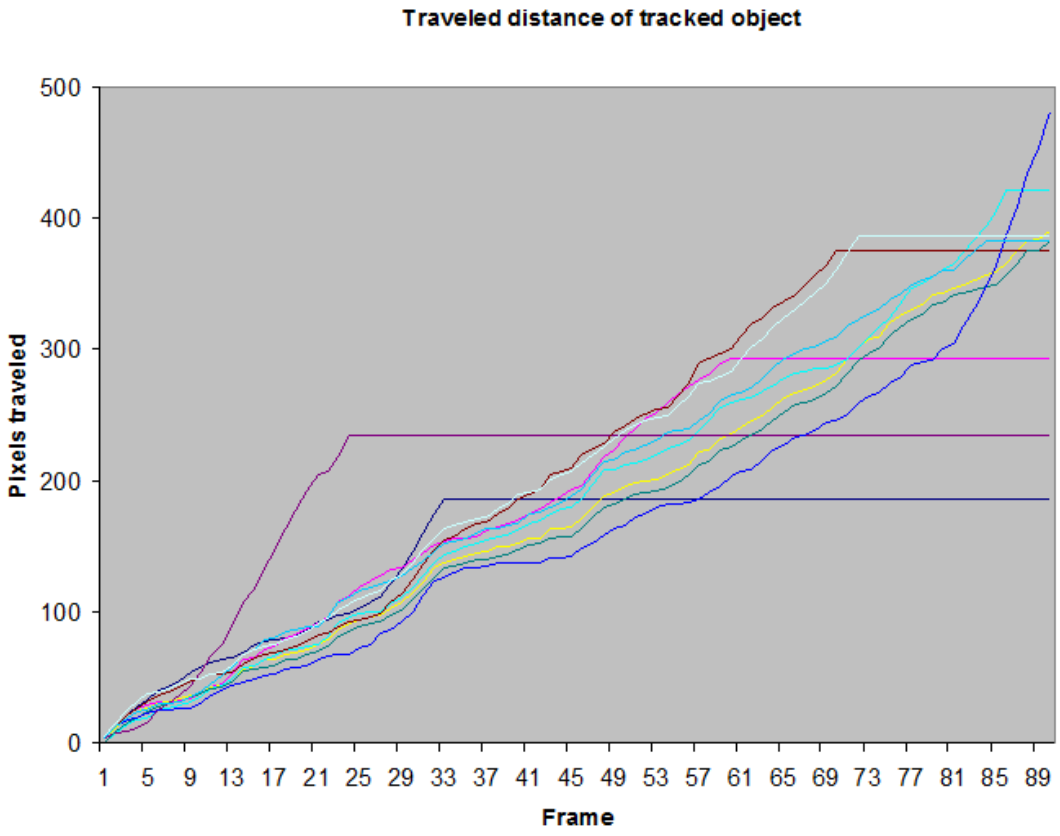


Figure 4.17: Plot depicting the movement of ten random objects.

centage of the highest airlight value within the image are considered to be white objects. For our experimentation this naive method satisfies our purpose, however for an implementation of this technique a better and more consistent method for detecting white objects should be used.

Having determined the white objects they can be corrected based on their depth according to the depth map. In our experimentation we deducted the depth map from the restored image and the airlight. Figure 4.18 shows the airlight of a frame in which a sign with white numbers is seen. The top image has a high airlight value for the white parts of the sign, meaning it is far away. In the bottom image we tried to correct this error by deducting the depth map from the airlight. In this example it is obvious that our naive method for detecting white objects does not suffice. In the bottom image the white parts of the sign are not completely corrected, and the parts that are corrected do not have the same airlight values as the other parts of the sign. In the perfect case the whole sign should have the same airlight value. Instead of correcting the found white objects based on their depth as estimated by our depth map they should be corrected according to the airlight around the white objects. This assures that the white parts of the sign will have the same airlight as the non-white parts of the sign. The depth map could be used to detect the white objects together with their airlight value, for example, an object with a high airlight value that is near according to the depth map would be considered a white object. So, instead of using the depth map to correct objects directly, it should be used to determine which objects are to be corrected. The basic idea of detecting white objects and then correct them remains the same.

Detecing white objects

In the previous section we used a naive method to detect white objects, we will now focus on creating a better technique to detect these objects. Our goal is to find white objects that are near, our tools are the intensity of all pixels and the velocity map. Because we are only interested in white objects we can ignore objects with low intensity, we only take the p^{th} percentile of highest intensity pixels into consideration. The percentile of objects to ignore depends on the image, an image in which most objects are far away requires a lower value for p because the intensity of objects increases with distance. A correct value for p will exclude near objects that are not white, leaving us with the task to exclude far away objects. Before moving on to that task we will take a closer look at the variable p. Figure 4.19 shows the effects of changing p, it is obvious that there are less pixels excluded as p lowers as it is a percentile, however all excluded pixels are in the lower part of the image where objects that are near are likely to be. In the airlight image belonging to these images it seems that the white objects that are near are brighter than most objects that are further away. By reading the intensity values it is revealed that this is not the case and merely an optical illusion due to the graduate change in intensity of the far away objects and the sudden change of intensity of the closer white object. Due to this optical illusion less pixels can be excluded than expected. In our experiments we used a value of forty percent for p which never excluded the white objects that are near in a series of one hundred frames. This means that we have already excluded 40 percent of pixels, which is quite significant, with a simple operation.

Figure 4.18: Top: Airlight of an image with a white object nearby. Bottom: Corrected airlight by deducted the depth map.

Figure 4.19: White pixels are possible white objects. p describes which percentage will be considered not white enough. Top: p=70%. Middle: p=55%. Bottom: p=40%.

Excluding objects based on their intensity allows us to remove darker objects leaving us with bright objects that are either close or near. To determine whether an object is near or not we use the velocity map. The velocity map is segmented in several layers by applying K-means, which should segment the map into depth layers. The segmented velocity map can assign pixels of the same object to different layers. These errors are corrected by creating a new map that takes object colour into account. Pixels with the same colour are likely to belong to the same object and move at the same velocity, we enforce this observation by averaging the velocity of pixels with the same colour within a certain range. By doing so, small wrongly estimated patches are corrected. Figure 4.20 shows the results of averaging the velocity layer only when the pixels within the window are within 10 percent of the maximum Euclidean distance possible. From the results we can see that the blurring effect increases with window size and objects in the image become better recognizable. As the window size increases the chance that two objects at different depth with the same colour are within the same window increases. Therefore the window size should be large enough to make objects better recognizable based on velocity but small enough to not take other objects into account. In our experiments we used a window size of 80x80 for all images in the sequence. By combining the segmented velocity map and the newly created map that takes object colour into account we obtain a better segmented map. This improved segmented map is then used to determine which bright objects are near.

The implementation of this method requires several parameters to be chosen manually. An important parameter is the percentile that controls the intensity mask, as can be seen in Figure 4.19. Aside from this parameter three other parameters play an important role. These parameters are: The window size for blurring objects with similar colours; A weight for the velocity map and the blurred velocity map; A parameter determining which layer is considered nearby. In our experiments we searched for values for all parameters for one single image, then used the same parameters for a sequence of 100 images in which we try to detect the white parts of the sign in the image. The image we used for finding the parameter values was in the middle of the sequence. In the first thirty-five to forty images the results are disappointing as can be seen in Figure 4.21. Reasons for this can include the distance at which the sign is. Another reason can be the movement of the sign, the camera is focussing on the sign and keeps it in the same position of the camera making the observed velocity of the sign lower. In later frames the sign moves closer and is better detected by our method, there are still quite a few false positives, however we have detected all true positives and there are no false negatives. This raises the question if the true positives are a significant problem, this question will be answered when the detected pixels have had their airlight values corrected.

Now that we have marked a set of pixels we will try to correct the airlight of these pixels with two different methods. In the first method we use interpolation to obtain the airlight values. We expect this method to work reasonably well because in our experimental setup the parts we want to correct are surrounded by pixels with correct airlight. From the false-positives we will be able to tell whether or not interpolation gives correct airlight values under less favoured conditions. In the other method we apply an image inpainting algorithm to correct the airlight of the marked pixels. The algorithm that is used is a Matlab implementation of the paper "Object Removal by Exemplar-based

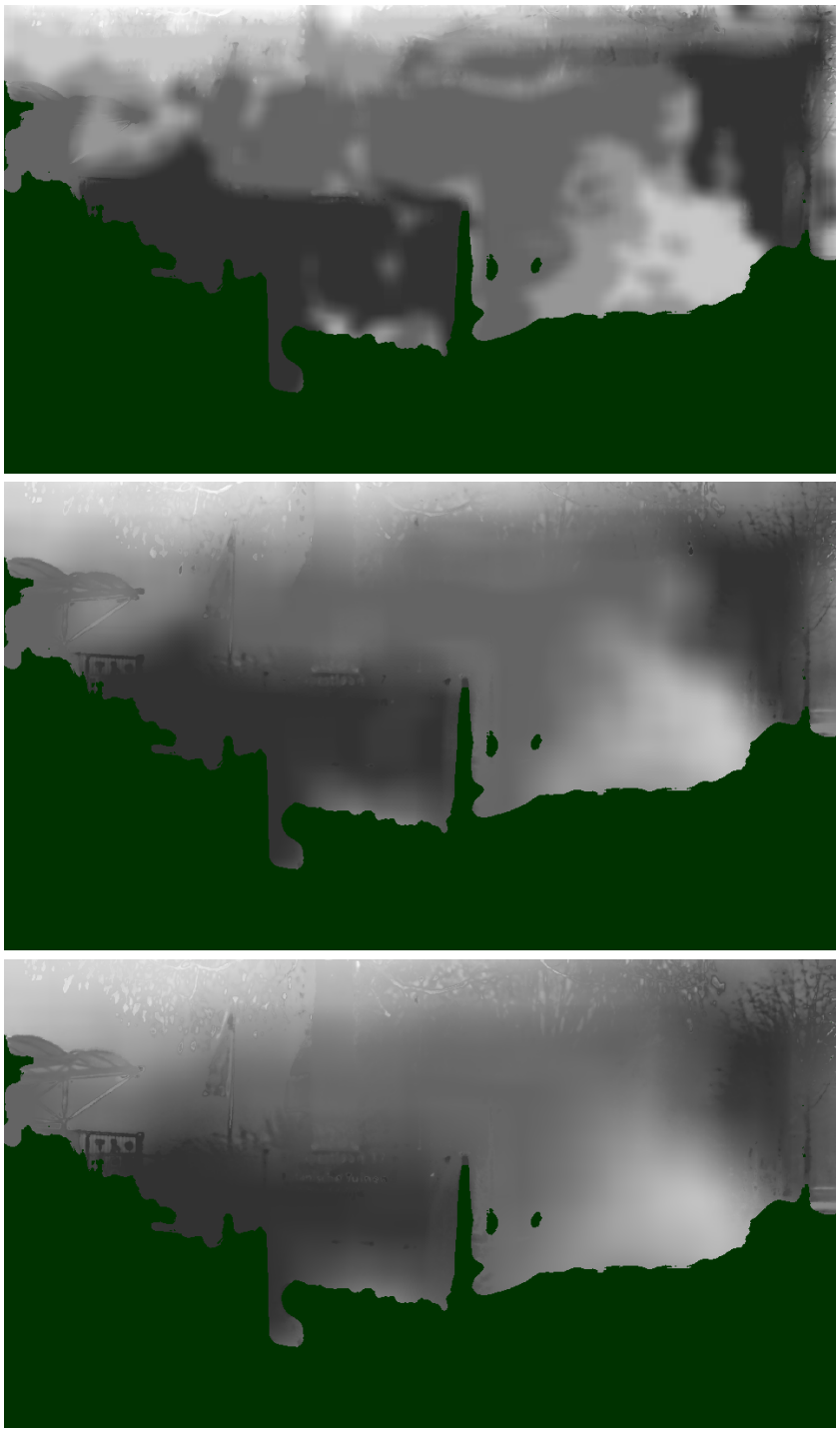


Figure 4.20: Velocity blurring that takes colour into account with different windows sizes. The intensity mask is shown as dark green in the image. Top: 20x20. Middle: 60x60. Bottom: 100x100.

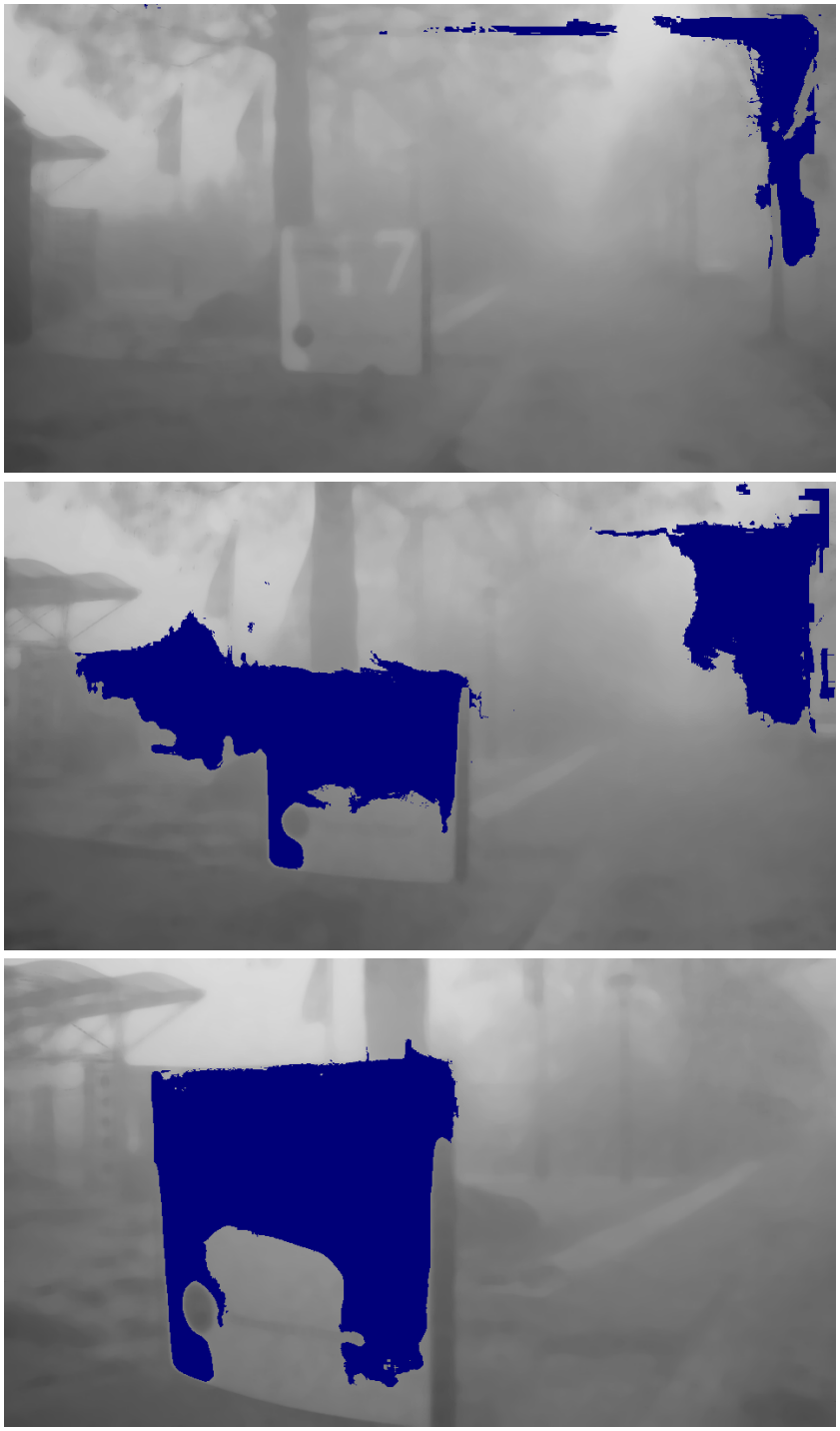


Figure 4.21: Three images from the sequence. The goal is to detect the white parts of the sign. Objects that are considered white and nearby are shown as blue in the images.

Inpainting"[6]. Using an inpainting algorithm might seem like a logical approach as that's basically what we wan to achieve, however our expectations our lower than the first method. The inpainting algorithm uses surrounding patches to fill the marked region and tries to maintain strong edges. We expect that the algorithm will extend the tree behind the sign through the sign, which is not what we want. In Figure 4.22 the results of the two methods are shown, the image created by the inpainting algorithm is not entirely what we expected, but it is indeed worse than the other method. It seems that the inpainting algorithm is placing sky at the top part of the sign making the airlight too high. The left part of the sign is darker, this is likely due to the tree behind the sign. The result created by interpolation managed to correct the airlight of the sign where it was too high and outperforms the inpainting method significantly. We had expected that interpolation would yield good results for the true-positive pixels, our concerns lied with the false-positives. Figure 4.23 shows the before and after airlight images of the middle image of Figure 4.21 which contained a large patch of false-positives and another large patch existing out of true-positives and false-positives. In the top right image there is a large patch of false-positive white objects, after correction the airlight has slightly changed for these pixels. The difference is only minor because the surrounding pixels had similar airlight values, however these values were slightly lower and were therefore not considered to be a possible white object. The false-positives would have their airlight lowered more significantly if they were surrounded by pixels with low airlight values, the opposite can be said for the true-positives. For this method to work correctly it is imperative that the objects that are to be corrected are surrounded by values that have the same airlight value as corrected objects should have.

To determine the improvement in a quantitative way we compare Tarel *et al.*'s and our airlight to a ground truth. This ground truth has its airlight manually corrected for white objects that are near using a graphics editing program. Because our method uses Tarel *et al.*'s airlight as input we only focus on parts of the image that have been marked for correcting. The mean-error and the standard deviation are based on the intensity changes between the ground truth and the airlight images, the higher the difference between the airlight image and the ground truth the worse the result is. Our first comparison is shown in Figure 4.24, this result contains all true-positives and is surrounded by a lot of false-positives, the top image of Figure 4.25 shows the pixels that have been altered by our algorithm. We computed the mean-error and standard deviation for all pixels that we altered for both images and found that the difference in mean-error and standard deviation is small. The small difference is caused by the large amount of falsely positive marked pixels which were very close to the correct airlight. In this case the mean-error of our improved result is even higher than the original as can be seen in Table 4.5, however our standard deviation is lower. The reason that the standard deviation of the original image is higher is due to the fact that the airlight values are either very close to being correct or are not correct at all. The amount of false-positives makes it difficult to measure how well the true-positives have been improved, therefore we manually marked the pixels of which we wish to see the improvement. The first manually marked image is the middle image of Figure 4.25 which contains mainly true-positives with a few false-positives. The results belonging to this image are found under *Manual 1* in Table 4.5 while the results of the bot-

Figure 4.22: Images of which the airlight is corrected, the top image uses an inpainting algorithm while the bottom image is obtained by interpolation.

Figure 4.23: Images of airlight before and after correction. Top: before. Bottom: after.

		Mean	Std
Computed	original	4	10
	restored	8	6
Manual 1	original	14	28
	restored	9	6
Manual 2	original	17	34
	restored	8	6

Table 4.5: The measured mean-error and standard deviation of the airlight images to the ground truth.

tom image are under *Manual 2*. The bottom image is different from the middle image as it does not contain any false-positives. As expected the difference in mean-error grows when there are fewer false-positives. So far the standard deviation has not been mentioned, but it does play an important role. The white objects we wish to correct should be close to the ground truth, therefore the mean-error should be low, however a low mean error with a relative high standard deviation is not good. We assume that all pixels of a white object are at the same depth, meaning their airlight value is the same or only varies little, therefore the standard deviation should be low. In Table 4.5 we can see that the standard deviation of the restored airlight is in all cases lower. From these results we can say that the depth of the restored airlight is constant and close to the expected depth.

The first comparison focuses on how well the true-positives are corrected, but how does the algorithm behave when there are many false-postives and barely any true-positives? Before answering this question we need to know whether it is possible or not to have a large amount of pixels marked for correction, this is easily answered by a few frames from the video we corrected with our algorithm. It is possible to have a many pixels marked for correction, but why? Well, the algorithm for marking pixels for correction makes its decision based on only two parts. The first part is the brightness of the pixels, the algorithm sorts all pixels on intensity and uses a chosen percentile as threshold for pixels that are considered as white objects. The second part is based on the relative velocity in the image, where faster moving objects are considered to be near. Both parts of this decision making process are controlled by a single parameter each. The parameter for detecting the white objects is as earlier mentioned a percentile of intensity within the image. In our experiment we set this percentile to 40%, this means that atleast forty percent of all pixels are considered as a white object. This value has to be high because the sky within the image usually is the brightest part of the image. The velocity based part is more strict, it clusters the pixels based on their velocity using K-means. In the less likely scenario that the slower clusters are small and the fastest cluster contains a large amount of pixels that also happen to be considered as white objects it is possible to have a large amount of pixels marked for correction. Now that we know that it is indeed possible to have a large amount of false-positives in the image we want to know how these pixels are corrected. To answer this question we use the same comparison as we used for defining how well the true-positives are corrected. The method is the same but instead of an image

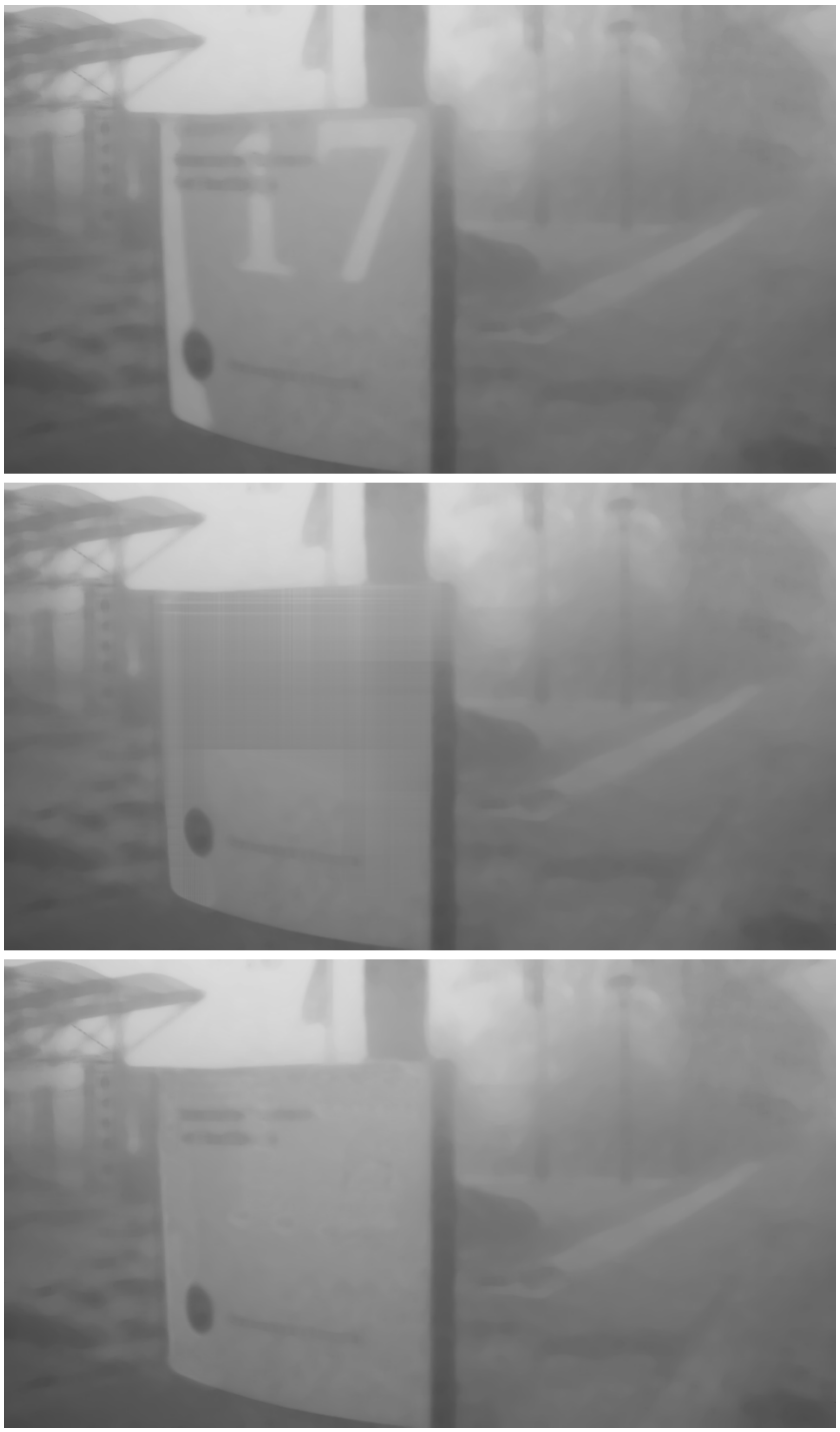


Figure 4.24: The top image is the airlight created by Tarel *et al.*'s method. The middle airlight image is our result, which is based on Tarel *et al.*'s airlight. The bottom image is the ground truth manually created using a graphics editing program. In our comparison we compare the top and middle image with the ground truth and determine the mean-error and the standard deviation.

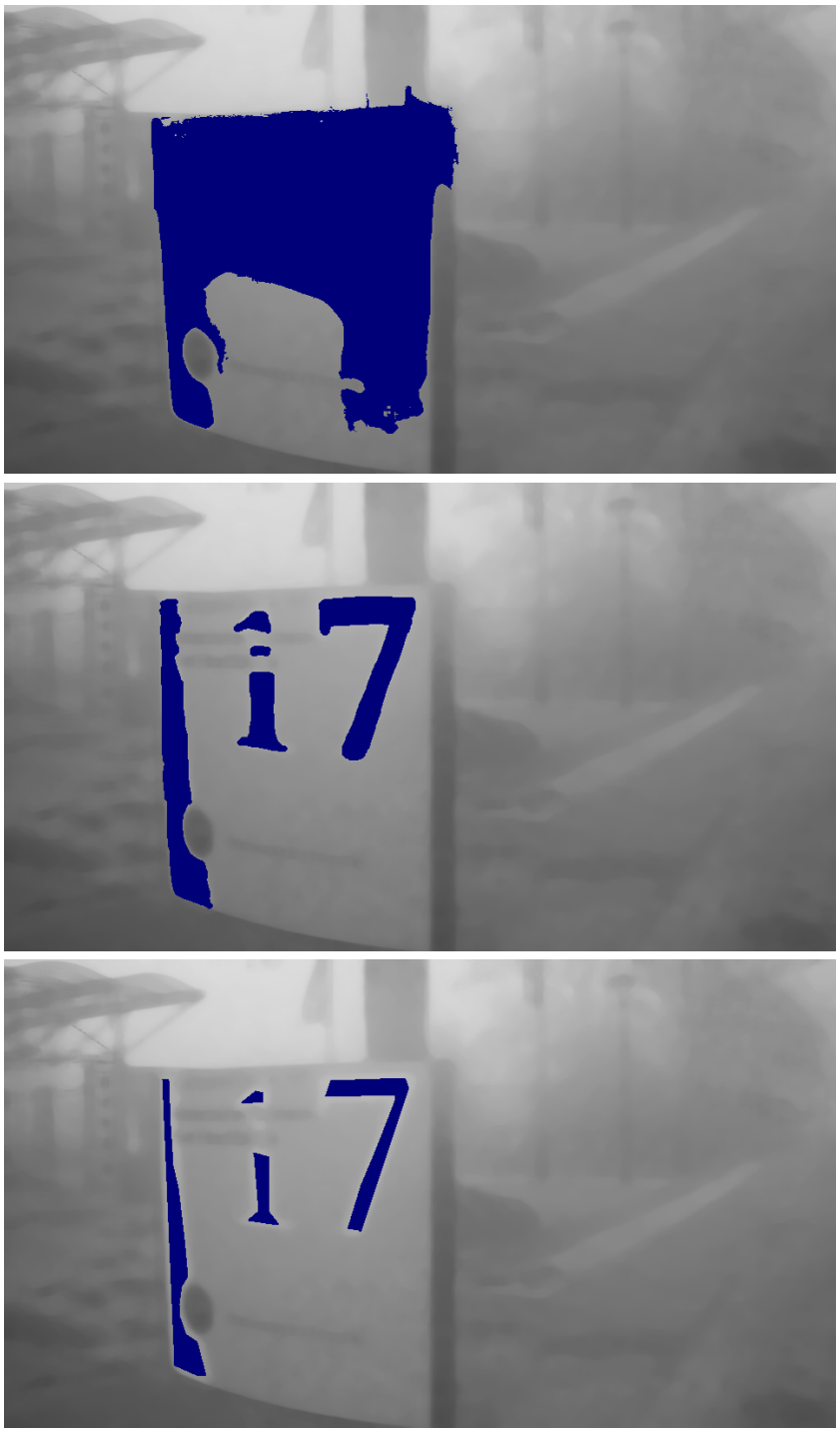


Figure 4.25: Top: *Computed*, the blue pixels are marked by our algorithm for correction, it contains all true-positives and a lot of false-positives. Middle: *Manual 1*, the blue pixels are manually marked and contain all true-positives and few false-positives that lie on the border of the true-positives. Bottom: *Manual 2*, another manually marked image, but this one only contains true-positives.

	Mean	Std
original	0.4	4
restored	17	16

Table 4.6: The measured mean-error and standard deviation of the airlight images to the ground truth belonging to Figure 4.26 and 4.27.

with many true-positives we use an image with a lot of false-positives and as few true-positives as possible. We once again create a ground truth, but in case of the image we use the ground truth is very similar to Tarel *et al.*'s airlight so the original image should score very well on this comparison. After creating the ground truth we proceed by computing the mean-error and standard deviation and compare and analyse these results.

For our next comparison we use the images shown in Figure 4.26 and 4.27. The objects to be corrected in this image are the brighter *dots* in the image that are part of a sign that indicate a bend in the road. In the top image of Figure 4.27 we can see that a large part of the sky has been marked for correction and that there is a lack of true-positives. Due to the high intensity of the sky it is halfway to being marked for being corrected but as the sky should not be observed to moving it seems that this could be an error caused by the tracking algorithm. The large patch of marked pixels is partially corrected properly, the major problems are the vertical darker lines created by the restoration, these vertical lines are the result of the interpolation. These errors could possibly be averted by taking the gradient of the original airlight into account. The same image also shows that larger patches can be corrected neatly in the top left. We have computed the mean-error and standard deviation for the original and corrected airlight, the results are in Table 4.6. The original image is very similar to the ground truth and that can be seen in the results, the few true positives in comparison with the false-positives manages to increase the standard deviation to four. The bad results of the restored image are reflected in the mean-error and standard deviation. The results of the original image can be used to reflect back on the previous comparison. This near-perfect result has a mean-error of 0.4 and a standard deviation of 4. The results of our restored image in the previous comparison had a mean-error of 8 and a standard deviation of 6. The mean-error is slightly off but the standard deviation is almost the same as the near-perfect original airlight. We have already mentioned earlier that the correction by interpolation requires the correct value at the borders of the patch that is to be corrected, the vertical darker lines are the result of incorrect values at the border of the patch. So how does correction hold for the falsely positive marked pixels? In most cases the patches are smaller and not patches with only gradual change in intensity. For these case the correction goes by unnoticed. In the case shown in Figure 4.27 the wrong correction of the airlight goes by noticed. The pixels with lower airlight values become significantly brighter than their neighbours with correct airlight values.

The final step of the algorithm is using the improved airlight to compute the dehazed image. Our method uses the airlight of Tarel *et al.*'s method as its base, for this reason we restore the image by substituting the airlight Tarel *et al.* computes by our improved airlight image. An example of an improved

Figure 4.26: The airlight computed by Tarel *et al.* is shown on the top and on the bottom is the manually created ground truth.

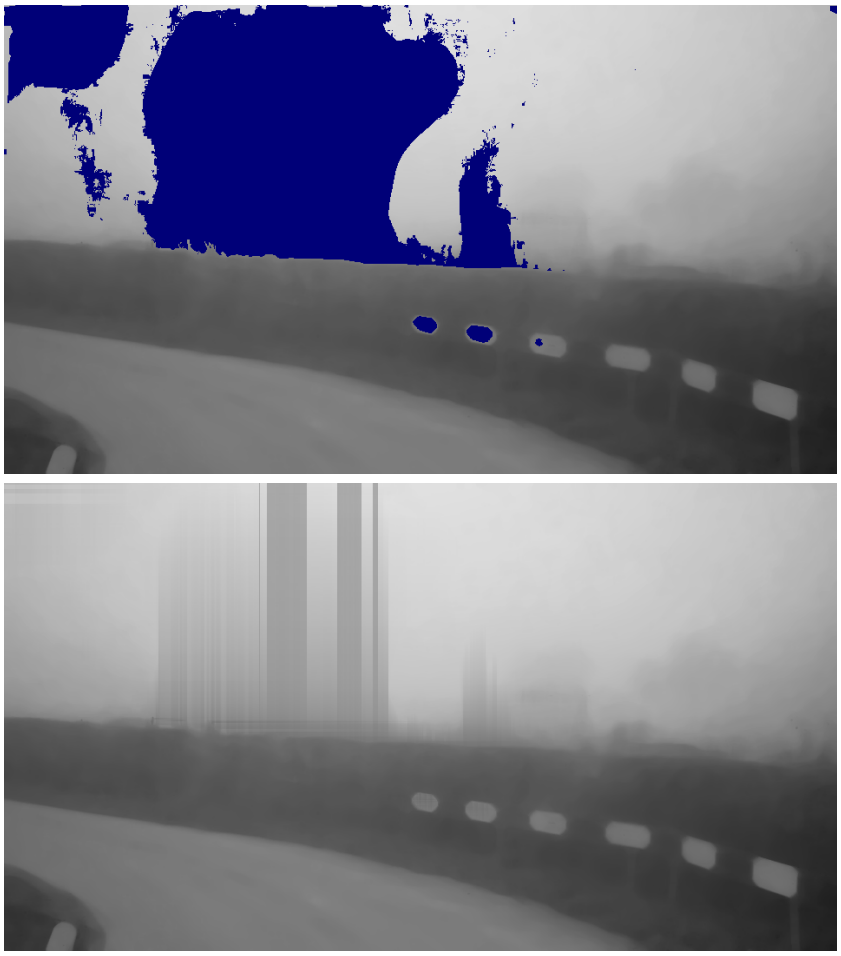


Figure 4.27: The top image has the pixels marked that are to be corrected and the bottom image is the result after correction. The ground truth for the bottom image is shown in Figure 4.26.

Figure 4.28: Images showing the improvement of restoration, the top image is the restored image by Tarel *et al.*, the bottom image is our improved restored image. The corresponding airlight images are shown in Figure 4.24

restored image is shown in Figure 4.28, in this image we can see the effect of the corrected airlight. Because the corrected airlight is lower than the originally estimated airlight those pixels have a higher intensity and are closer to the correct object colour, which is white.

4.3 Contributions

In this chapter we have been exploring ways to extract information from video and to apply that information to improve the results of a single image visibility restoration method. The possible useful information from video were the slices we patched together by pasting rows or columns of a sequence of frames together and the tracking information from SIFT flow. The tracking information appeared useful and was used for colour correction, noise removal and rough

depth estimation. These techniques were all used to improve the result of Tarel *et al.*'s method by improving the estimation of nearby white objects.

As an object clouded in fog moves closer or further away from the camera the observed intensity of the objects changes. When the camera in a video moves forward the observed intensity of the objects changes. Our so-called time slices, which are created by stacking frames to create a three-dimensional objects and cutting it so that it shows the change of a row or column in time, give an insight on how the intensity changes. From the fog model we know that the intensity of objects decreases as the distance to an objects becomes less, this is exactly what can be seen from the time slices. However, using this information is difficult to use because it only focuses on one row or column at a time. Seeing how an object changes over time we decided that actually tracking an object would provide better and more useful information.

The SIFT flow algorithm computes the flow between two images. The flow can be visualised as an image representing how each object moves from the first image to the second image. When the flow is computed for a sequence of images from a video objects can be tracked by following the movement according to the flow images. In Section 4.1.2 we ran some experiments in which we computed the flow for a sequence of images taken from a video and used it to track objects like trees and road markings in foggy weather. The only drawback in our implementation of this concept is that it does not handle occlusions well. When an occlusion occurs the implementation will track the object that occludes the object that was being tracked.

In the output of Tarel *et al.*'s single image method there tends to be noise in the whole image. As an experiment for SIFT flow we used the tracking information to average the colour of an object over a few frames. By averaging the colour over several frames the colour of the object should be closer to its true colour, because this is done for all objects within the image the image will look smoother. The transition between two frames will also be smoother because the colour of an object will not change drastically due to noise any more.

Correct depth estimation is important when improving the visibility of a weather degraded image. Based on the observation that a fast moving object that is far away stays longer in view than a slow moving object that passes by we try to estimate depth. SIFT flow can tell us how many pixels each object moves each frame. For a single frame these values are not very distinctive. When too many frames are used there can appear errors in the information (an object can move out of view or be occluded). By taking the amount of frames as a time unit and the movement in pixels as a distance unit it is possible to compute the velocity of an object. Displaying the velocity as an image resembles a rough depth estimation of the scene. The velocity map is too rough to directly use for restoring the visibility in an image, but it can be used to improve the estimation of other depth estimations.

The depth estimation of Tarel *et al.*'s method is based on the whiteness of the image. The method assumes that an object that has high values for all three colour channels is far away. However, this results in white objects that are nearby to be estimated as far away, which is incorrect. In Section 4.2.3 we propose a method which detects nearby white objects. Because we are only interested in white objects we filter out objects based on their intensity. From the group that remains we filter out objects based on their velocity using our

velocity map. After filtering out the non-white and far away objects we are left with the objects we wish to find. Having found the objects we wish to correct we interpolate their values from surrounding objects, assuming that the nearby white objects are surrounded by objects which are at the same depth. The corrected depth map is then used to restore the image.

Chapter 5

Conclusion

The goal of this master thesis project is to remove fog from a video. We have done so by first analysing two methods that can remove fog from a single image. In the first method we analysed, which was Tan's implementation, we replaced Markov Random Fields and graph cuts for a bilateral filter and obtained similar results but faster. This change to the method is possible because the MRF is used for smoothing the airlight while maintaining strong edges, which can also be done by a bilateral filter. By analysing the single image methods we learned the benefits and drawbacks of the two methods, allowing us to better chose a candidate to use as a base for removing fog from a video.

After having analysed the two single image methods we moved on to video based fog removal. We focus our attention on the change of pixels over time and start by investigating our so called 'time slices'. From this investigation we conclude that we require the ability to track pixels over time. We then move on to a method that allows us to track objects, namely SIFT flow. We experiment with SIFT flow on video's that contain fog to see if the algorithm works under foggy conditions. The experiments show that it is possible to track pixels using the method making us ask the question (and answer) why it works under these conditions.

Having found a method that allows us to track pixels over time we proceed by using the tracking information to improve the results of single image fog removal. Our main focus is on the improvement of the airlight, in our first attempt we combine the airlight of several sequential frames as a way of smoothing the airlight transitions. The results of this attempt were not successful, however when applied to the result of Tarel *et al.*'s single image method, instead of airlight, the result is smoother and less noisy. We then compare our created de-noise effect with the results from a professional de-noise tool and find our de-noising effect competitive with the de-noise tool for this type of noise.

In our second and final attempt to improve the single image fog removal results we start by estimating depth based on velocity of pixels. Based on the observation that objects that are near move faster than objects that are far away we aim to estimate the depth from velocity. We create a velocity map and try to use it to improve the airlight unsuccessfully. Instead of using the velocity map to improve the airlight we decide to use it to detect parts of the airlight that have to be improved. Having already analysed the single image methods we

know that a drawback of Tarel *et al.*'s method is its failure to correctly handle white objects that are near, as it estimates depth based on the whiteness of an object. Noticing that we need to focus on nearby pixels and knowing that our velocity map can detect nearby objects we combine our velocity map together with an altered intensity mask to detect white objects that are nearby.

Now that we have a method that can tell which parts of an airlight image have to be corrected we need a way to actually do so. We observed that the pixels that are to be corrected are surrounded by pixels with correct airlight allowing us to interpolate to find the correct airlight values. We compare the results of interpolation with the results of an inpainting method and a manually created ground truth, and find that interpolation gives the best results as we expected. Finally we use our corrected airlight to restore the fog plagued video.

In short the contributions of this master thesis project are, in order of importance, the improvement of Tarel *et al.*'s airlight by detecting and correcting white objects that are near using a velocity map created with SIFT flow. We also created a method to de-noise the results created by Tarel *et al.*'s method using SIFT flow. Furthermore we lowered the computational time required for Tan's single image method by replacing the Markov Random Fields with a bilateral filter.

5.1 Future work

In its current state our algorithm is slow, which can be addressed in future work. The use of SIFT flow is one of the main reasons for the method to be slow. Replacing SIFT flow for a faster method that yields similar result is a possibility. Another possibility would be to adapt SIFT flow to work faster on video. A different option for future work is further improving the computational time of Tan's method. We have already improved the computational time of this method by replacing MRF with a bilateral filter. Further time gain can be obtained in the step before applying the bilateral filter. An intelligent design that finds airlight values that result in high contrast fast would improve the computational time. Currently the airlight values are found brute force.

Bibliography

[1] R. Tan, *Visibility in Bad Weather from a Single Image*.

[2] N. Hautire and J.-P. Tarel *Fast Visibility Restoration from a Single Colour or Gray Level Image*

[3] R. Fattal. Single image dehazing. In *ACM SIGGRAPH'08*, pages 1-9, New York, NY, USA, 2008. ACM.

[4] Srinivasa G. Narasimhan, Shree K. Nayar. *Contrast Restoration of Weather Degraded Images.* IEEE Trans. Pattern Anal. Mach. Intell., 2003: 713-724

[5] Ce Liu, J. Yuen and A. Torralba *SIFT flow: Dense Correspondence across Scenes and Its Applications*

[6] A. Criminisi, P. Perez, and K. Toyama. *Object removal by exemplar-based inpainting.* In Proc. Conf. Comp. Vision Pattern Rec., Madison, WI, Jun 2003.

[7] Lambert-Beer law, `en.wikipedia.org/wiki/Beer-Lambert_law`

[8] S. Bronte, L. M. Bergasa, P.F. Alcantarilla *Fog Detection System Based on Computer Vision Techniques*

Appendix A

Depth estimation comparison

In order to conclude which method creates the best depth estimation we created an experiment in which we have five images that each have ten marked points and ordered those ten points manually on depth. We compared the airlight images of Tan and Tarel *et al.* with our manually marked images. We do not look to the exact airlight values but only to the order in which the points are marked, which should be similar to our manually ordered point set.

On the following pages we show the input images and the resulting airlight images. The top image is the input image with the black marks shown, the middle image is the airlight of our implementation of Tan's method and the bottom image is the airlight of our implementation of Tarel *et al.*'s method. In the case that the images are shown sideways the left image is the input, the middle Tan's result and the right image is Tarel *et al.*'s airlight. Note that the airlight images were created using our implementation of the methods. In our airlight results of Tan's method several artefacts appear that are either caused by the compressed input image we used or a possible error in the computation of the airlight. None of the artefacts are on our marked points and should therefore not affect the results.

Figure A.1: *Fog 1*.

Figure A.2: *Fog 2*.

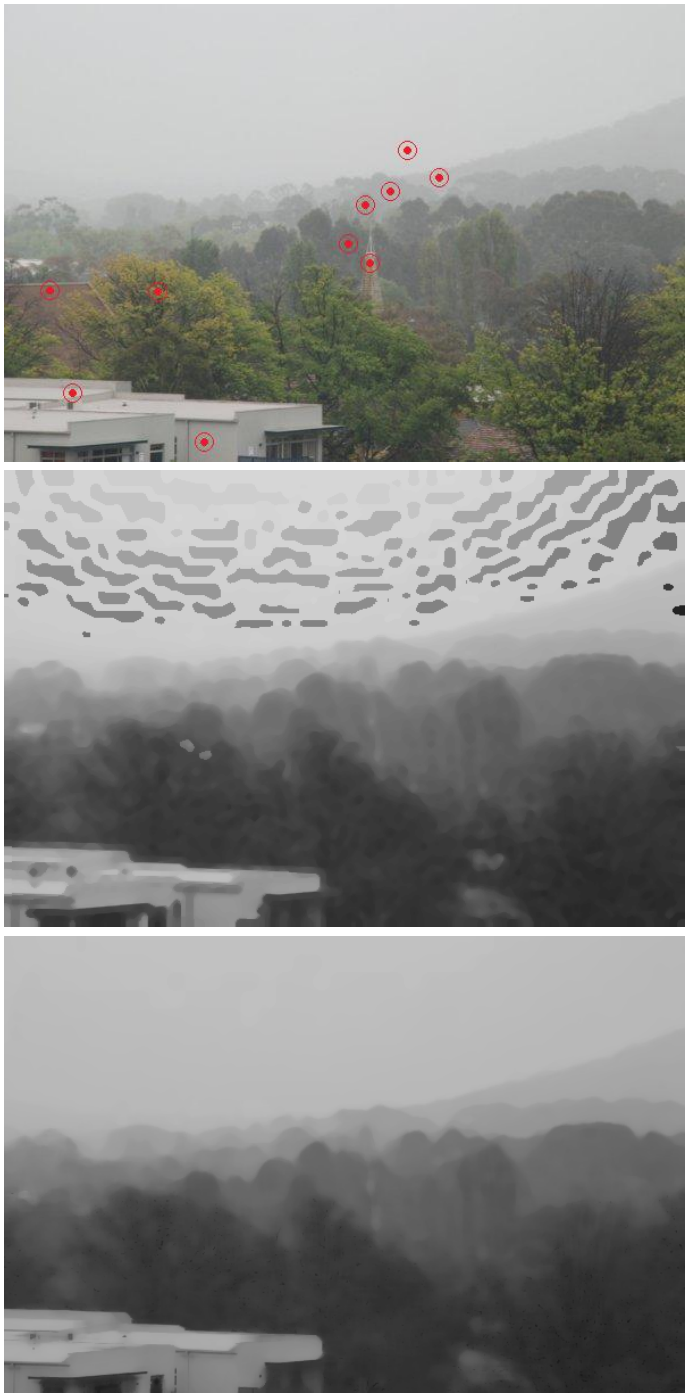


Figure A.3: *Fog 3*.

Figure A.4: *Fog 4*.

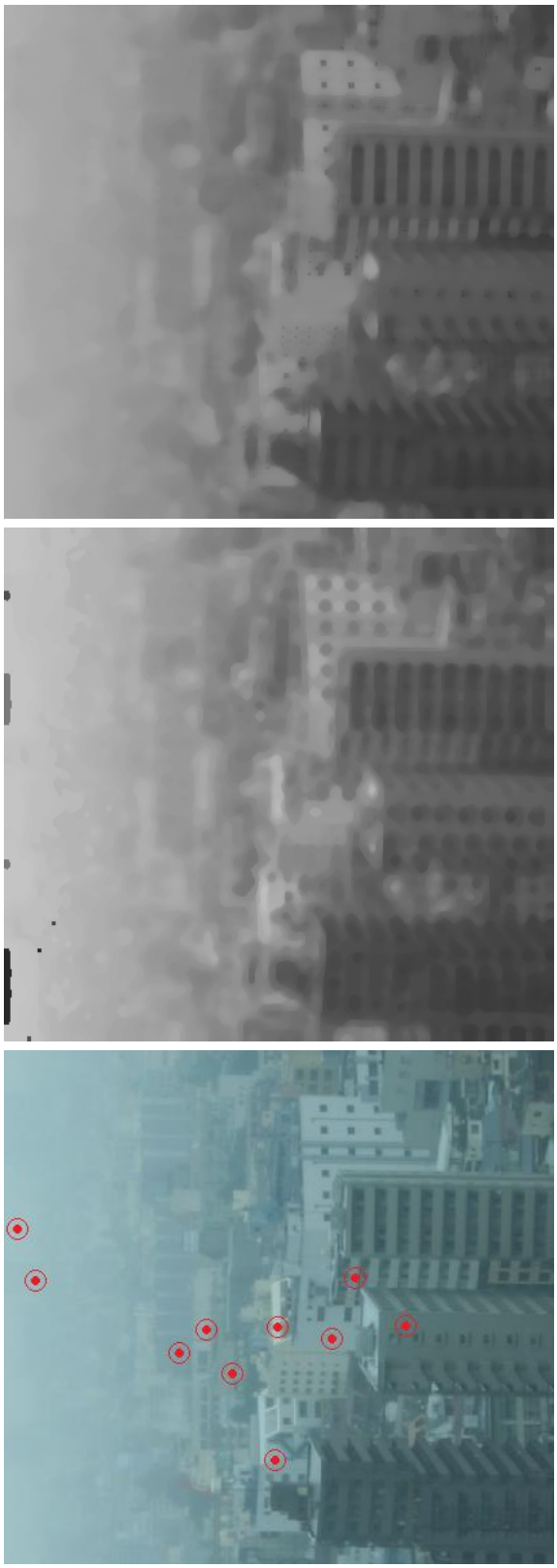


Figure A.5: *Fog 5*.

Figure A.6: *Fog 6*.

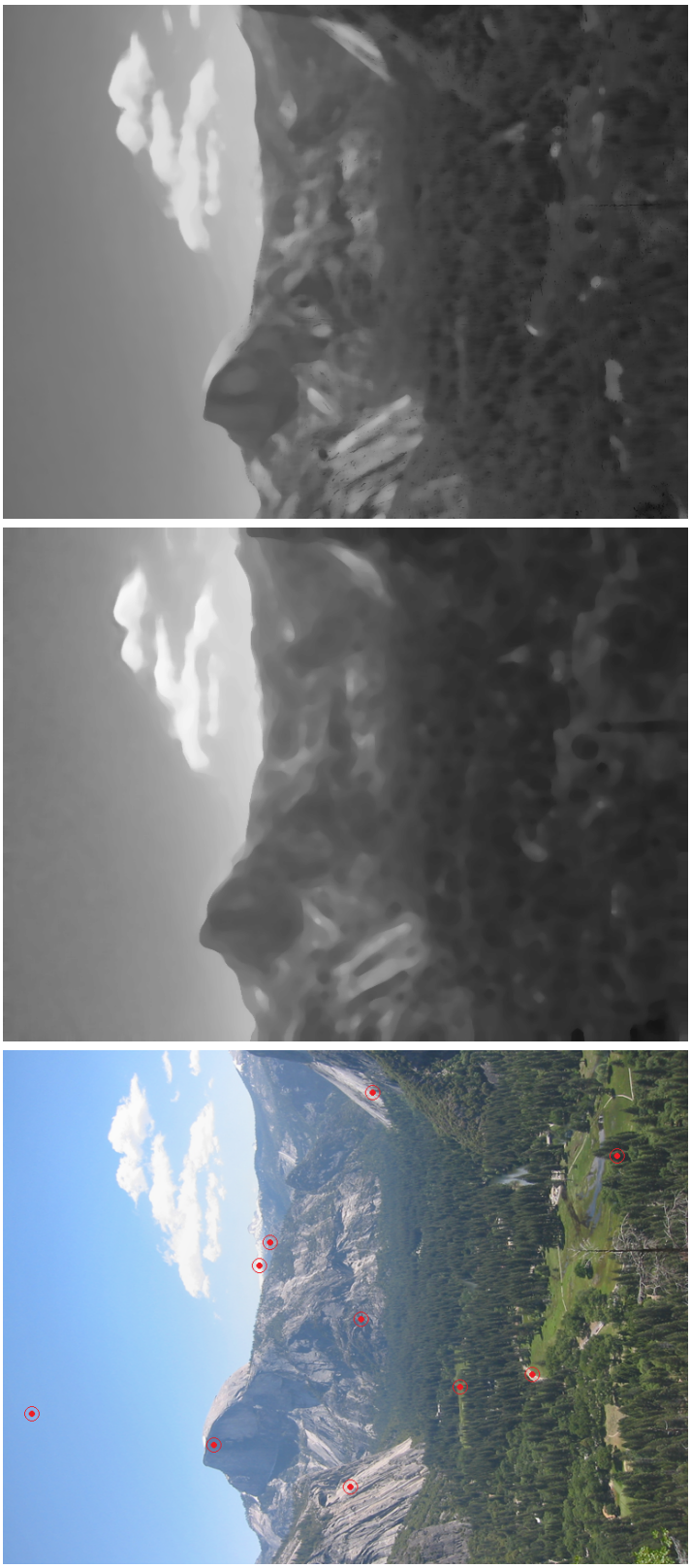
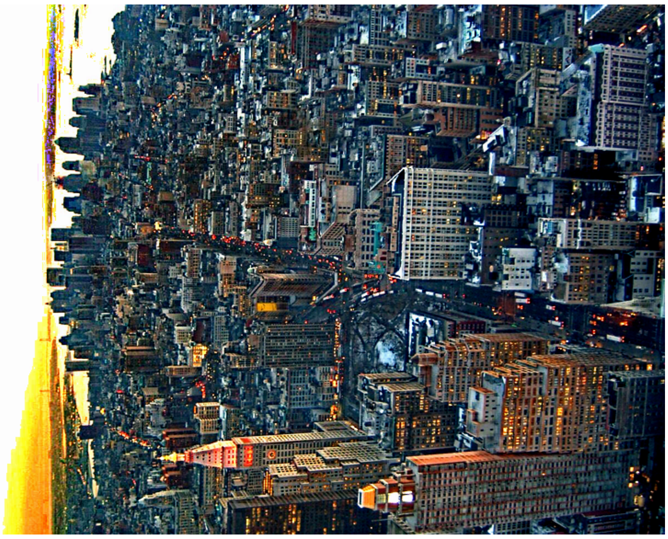
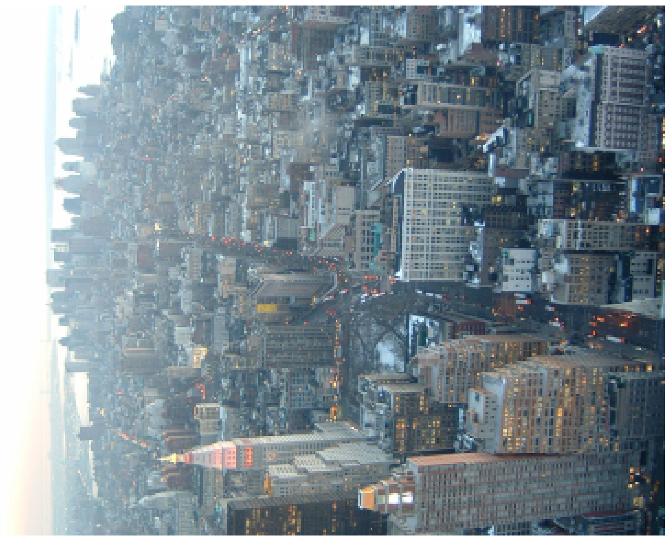


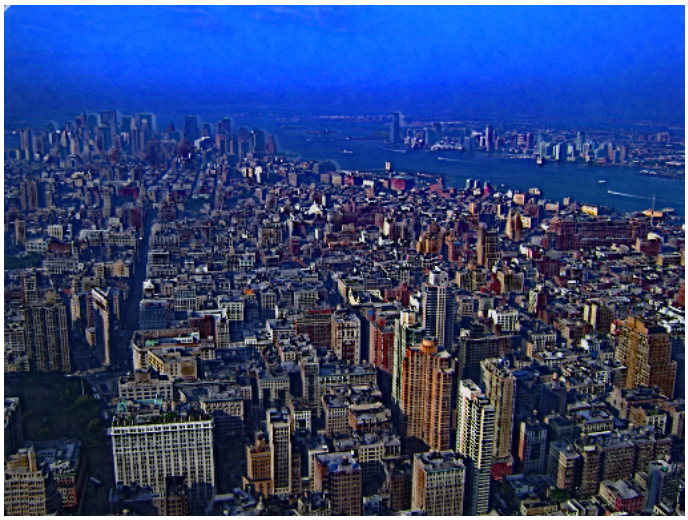
Figure A.7: *Fog 7*.

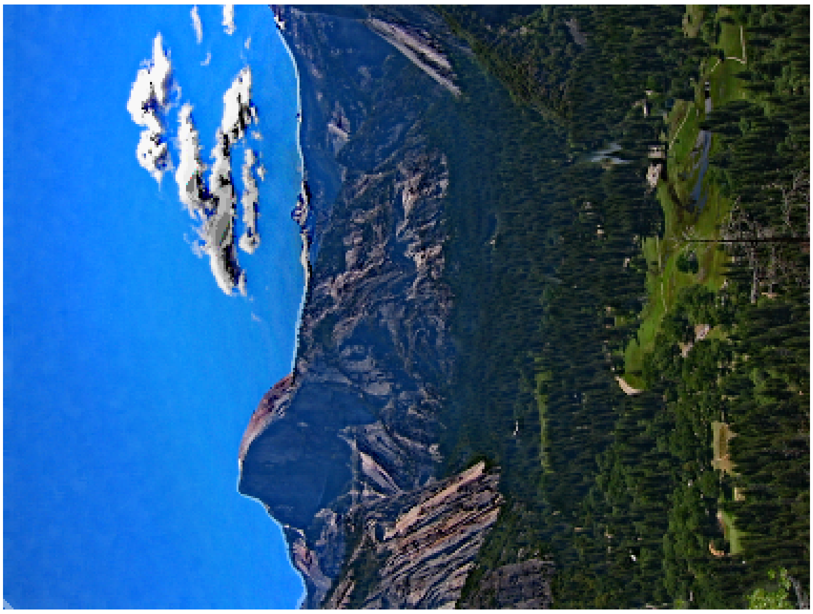
Appendix B

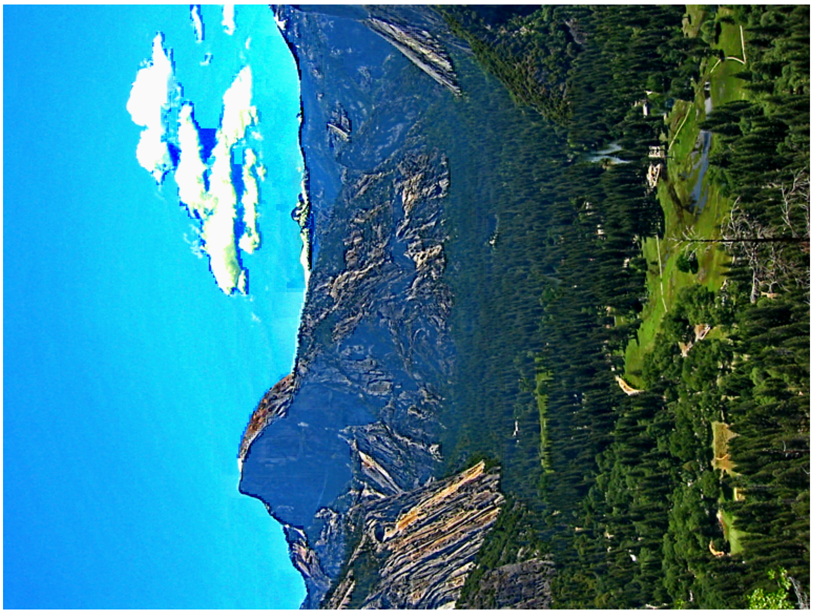
Comparing Tan's results with our results of Tan's implementation

The code of Tan's implementation has not been released. We created our own implementation based on the paper[1] and compare it with released results of his implementation. Several images on the web are compressed and this seemed to affect the results of our implementation, therefore we only compare our implementation to images of which we have the non-compressed original image. The results of source images that are compressed contain artefacts in the airlight which is not the case when the source image is of a non-compressed data type, as of writing we still have not precisely determined the cause of these artefacts. On the following pages each page will contain three images which are usually rotated to better fit the page. When rotated the images are from left to right: the original, tan's result and our result. When not rotated the top image is the original the middle image is tan's result and the bottom image is our result.







Appendix C

Survey: Comparing the results of Tan and Tarel *et al.*

To determine which method gives the best defogging results according to human observers we have created a small survey. Each person was shown 13 sets containing three images. First they were shown the original image then the results of the defogging methods, the order of the defogged images was randomized and the subjects did not know to which method each result belonged. The thirteen set of images are shown on the following pages. The top image is always the original image followed by tan's result and finally tarel's result, when the images are shown sideways the left image is the original with tan's result in the middle and tarel's result on the right.

After the sets the images are grouped by favoured method, displaying the original image, result of the method and a pie chart showing the results of the survey for the image. This allows to easily see what kind of images are best suited for each method.

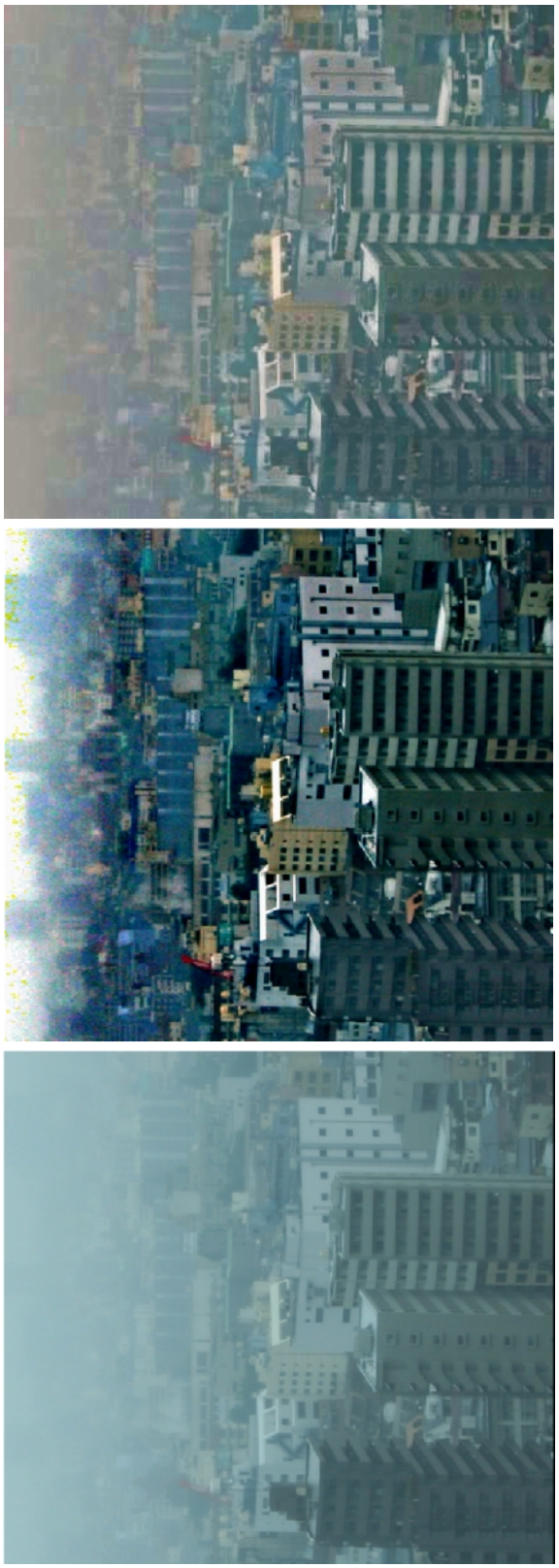


Figure C.1: Set 1.

Figure C.2: Set 2.

Figure C.3: Set 3.

Figure C.4: Set 4.

Figure C.5: Set 5.

Figure C.6: Set 6.

Figure C.7: Set 7.

Figure C.8: Set 8.

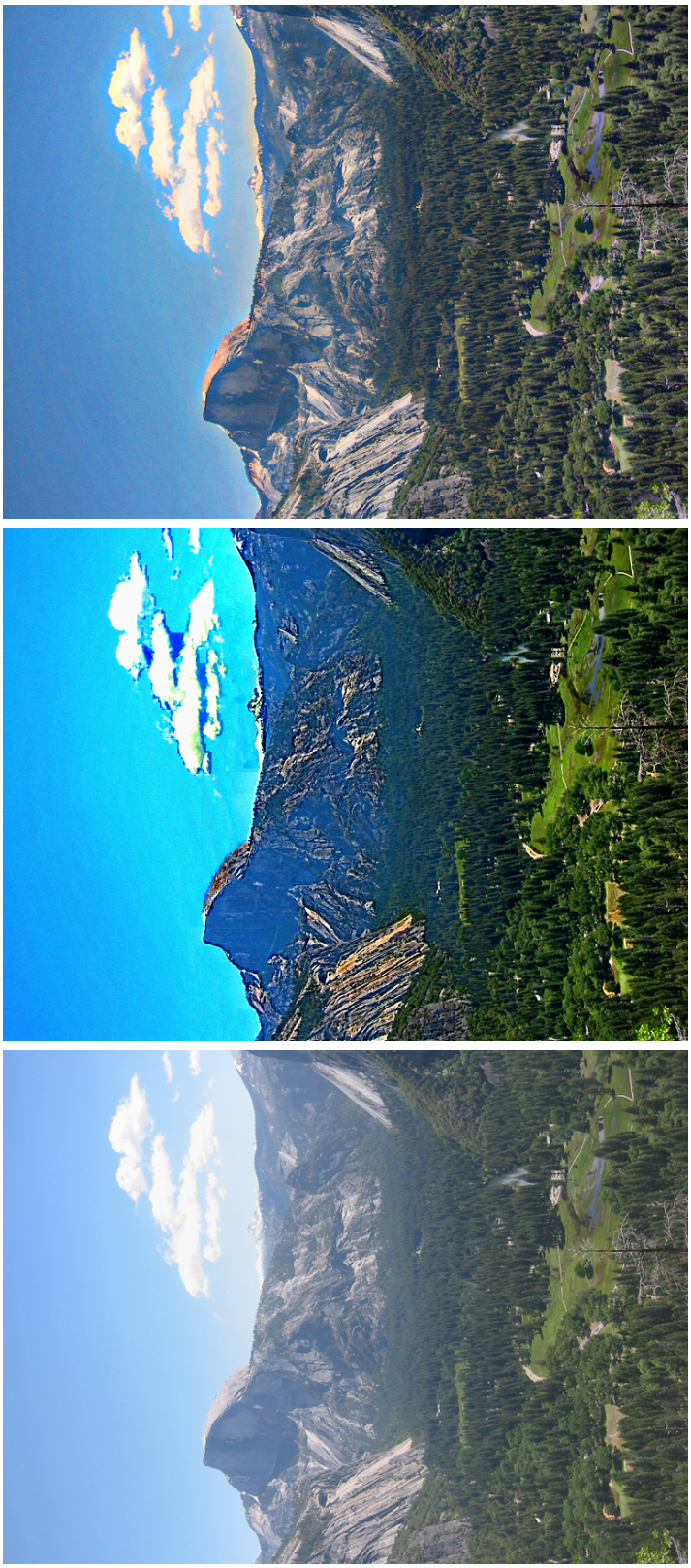


Figure C.9: Set 9.

Figure C.10: Set 10.

Figure C.11: Set 11.

Figure C.12: Set 12.

Figure C.13: Set 13.

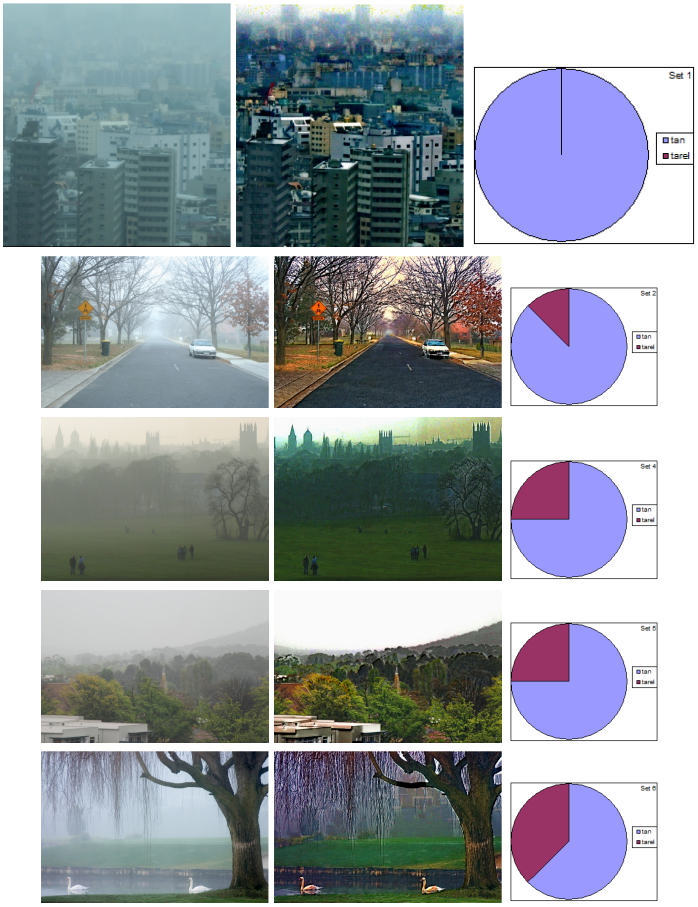


Figure C.14: Images favoring tan

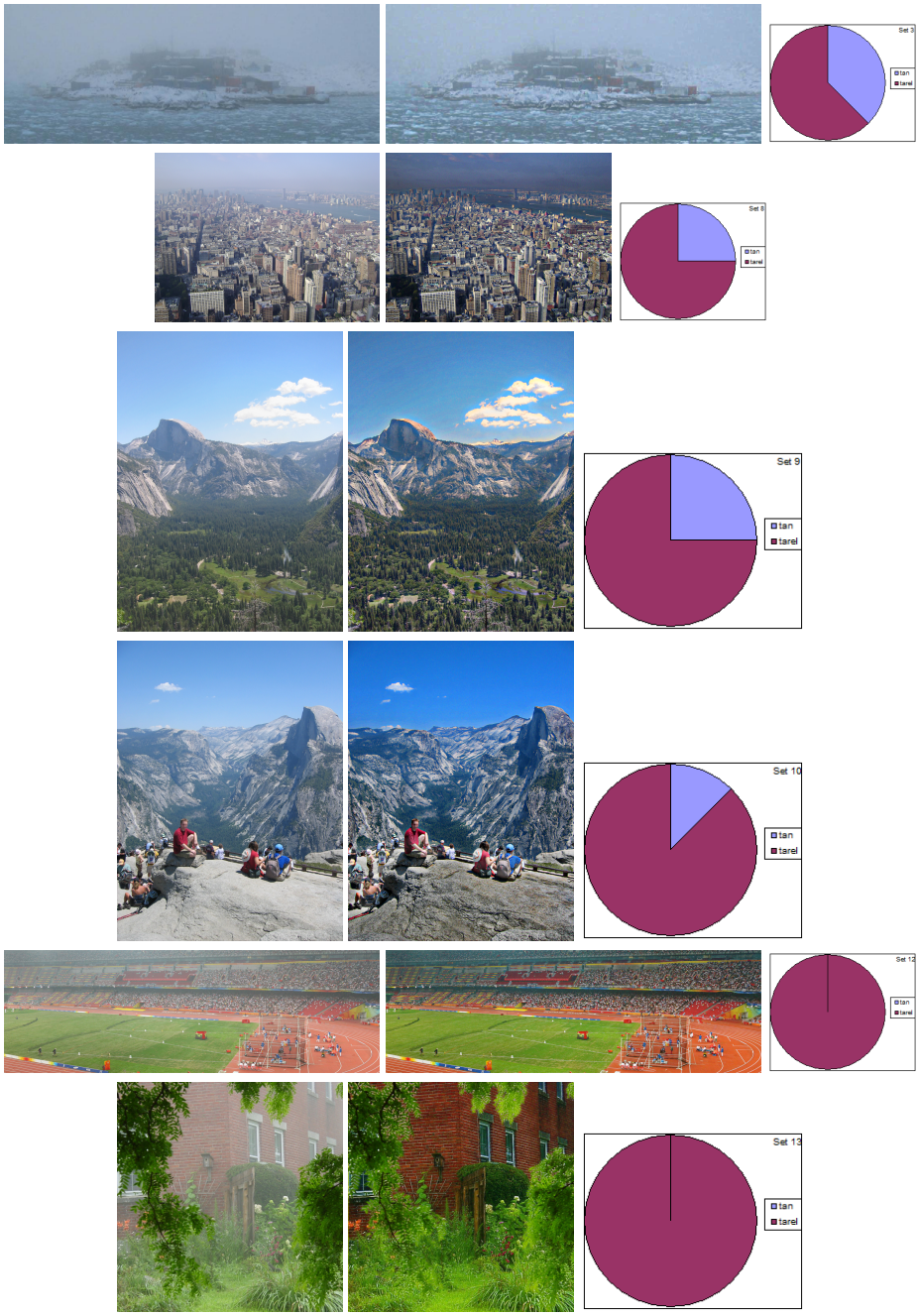


Figure C.15: Images favoring tarel

Figure C.16: Images that are equally favored.

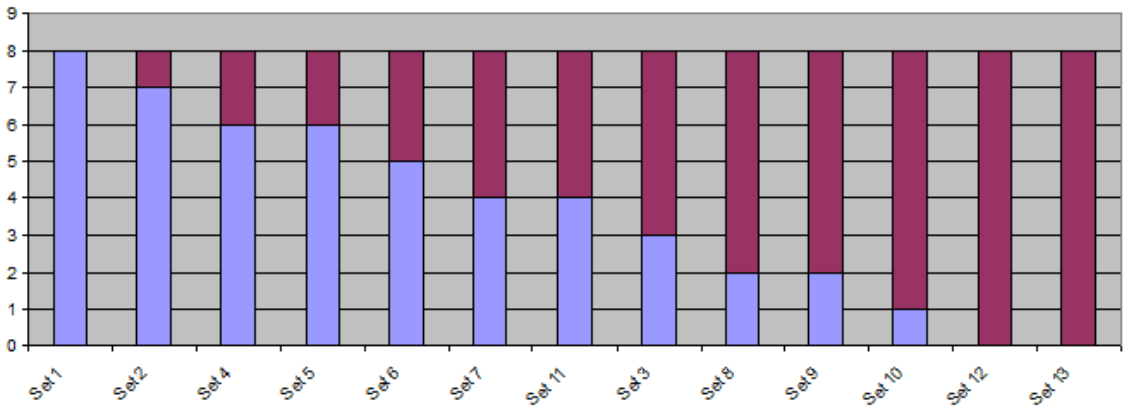


Figure C.17: Graph depicting the distribution of the votes for each image set, the blue line shows Tan's votes and the purple line shows Tarel *et al.*'s results.