Int J Comput Vis
DOI 10.1007/s11263-008-0200-2

Analysis of Rain and Snow in Frequency Space

Peter C. Barnum · Srinivasa Narasimhan · Takeo Kanade

Received: 10 February 2008 / Accepted: 10 December 2008

Abstract Dynamic weather such as rain and snow causes complex spatio-temporal intensity fluctuations in videos. Such fluctuations can adversely impact vision systems that rely on small image features for tracking, object detection and recognition. While these effects appear to be chaotic in space and time, we show that dynamic weather has a predictable global effect in frequency space. For this, we first develop a model of the shape and appearance of a single rain or snow streak in image space. Detecting individual streaks is difficult even with an accurate appearance model, so we combine the streak model with the statistical characteristics of rain and snow to create a model of the overall effect of dynamic weather in frequency space. Our model is then fit to a video and is used to detect rain or snow streaks first in frequency space, and the detection result is then transferred to image space. Once detected, the amount of rain or snow can be reduced or increased. We demonstrate that our frequency analysis allows for greater accuracy in the removal of dynamic weather and in the performance of feature extraction than previous pixel-based or patch-based methods. We also show that unlike previous techniques, our approach is effective for videos with both scene and camera motions.

Keywords De-weathering · Image enhancement · Noise removal

P.C. Barnum (✉) · S. Narasimhan · T. Kanade
Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA, USA
e-mail: pbarnum@cs.cmu.edu

S. Narasimhan
e-mail: srinivas@cs.cmu.edu

T. Kanade
e-mail: tk@cs.cmu.edu

Published online: 28 January 2009

1 Introduction

Rain and snow are often imaged as bright streaks. Not only can these streaks annoy or confuse a human viewer, but they degrade the effectiveness of any computer vision algorithm that depends on small features. For example, feature point trackers can fail if even small parts of an image are occluded. If these streaks are removed, then the tracker can work with greater accuracy. Alternately, rain may need to be added to a scene. For example, after a shot is taken, a movie director may decide that there should be more rain. The scene could be filmed again, but this would be costly and time consuming. Rather than requiring people to wait for the weather to be perfect, we develop techniques to digitally control the amount of rain and snow in a video.

Rain and snow are specific examples of bad weather. Although one good day is much like another, the properties of bad weather vary depending on the size of the constituent particles. Static bad weather, such as fog and mist, are caused by microscopic particles. Due to the small particle size, fog and mist are usually spatially and temporally consistent. Since their effect does not vary significantly over space and time, it is sufficient to only analyze their effect locally, on individual pixels (Nayar and Narasimhan 1999; Narasimhan and Nayar 2002; Cozman and Krotkov 1997). For large particles such as raindrops and snowflakes, analysis is more difficult. Spatially and temporally neighboring areas are affected by rain and snow differently, so must be handled differently.

Several methods have been developed to remove rain and snow from videos. The earliest use a temporal median filter for each pixel (Hase et al. 1999; Starik and Werman 2003). Temporal median filtering exploits the fact that in all but the heaviest storms, each pixel is clear more often than corrupted. The problem is that anything that moves will become

Fig. 1 (**a**) The snow has been detected by finding its global spatio-temporal frequencies. (**b**) The brightness and amount of snow is then manipulated to increase from its left to right

blurred. Zhang et al. (2006) extended the idea of per-pixel removal by correcting for camera motion via planar image alignment and detecting rain with k-means clustering. This method is an improvement over simple median filtering in cases where the scene is static and the video frames can be accurately aligned.

Garg and Nayar (2004) suggested that streaks can be segmented by finding pixels in individual streaks that change over space and time in the same way as rain. False matches can then be reduced via a photometric constraint that models the appearances of streaks. Searching for individual streaks this way can theoretically work for dynamic scenes with a moving camera. But this method is most effective when the streaks are against a relatively textureless background.

Garg and Nayar (2005) also demonstrated how to prevent rain from being imaged in the first place, by modifying camera parameters during acquisition. They suggest using temporal and spatial blurring, either by increasing the exposure time or reducing the depth of field. This removes rain for the same reasons as the per-pixel median filtering, and will not cause blurring when all objects are at the same depth or the scene is static.

In this work, we combine realistic streak modeling with the knowledge of the statistics of dynamic weather. Unlike previous works that detect rain by only looking at individual pixels or patches, we treat rain and snow as image-global phenomena. In order to determine the influence of rain and snow on a video, we develop a global model in frequency space. In image space, single rain and snow streaks appear similar to any type of vertical stripe. Likewise, in frequency space, a single streak is difficult to distinguish in the clutter. But as the number of streaks increases, the pattern they cause in frequency space becomes distinct. Although spotting an individual tree might be hard, finding the forest is easy.

We begin with a physical model of a single raindrop or snowflake. The dynamics of falling particles are well understood (Foote and duToit 1969; Magono and Nakamura 1965; Böhm 1989), and it is simple to determine the general shape of the streak that a given raindrop or snowflake will create. Based on the shape, the streak's appearance is then approximated as a motion-blurred Gaussian. The statistical characteristics of rain and snow have been studied in the atmospheric sciences (Marshall and Palmer 1948; Ulbrich 1983; Feingold and Levin 1986; Gunn and Marshall 1958; Ohtake 1965), and it is possible to predict the expected number and sizes of the streaks as well. The information of how one streak appears, combined with a prediction of the range of streak sizes, allows us to predict the appearance of rain and snow in an image.

The problem with the image space model is that it is difficult to apply to real scenes. However, even such complex and chaotic phenomena as rain and snow can be well behaved in frequency space (Heeger 1987; Langer and Mann 2003). Therefore, rather than trying to find every rain and snow pixel in an image, we instead model their effect in frequency space. Although it is not possible to predict the exact streak sizes and locations in a video a priori, we can create a frequency-space model by sampling in particle size, depth from the camera, and streak orientation. The frequency model is then fit to an image sequence by matching streak orientation and rain/snow intensity. The inverse Fourier transform of the ratio between the model's predictions and the actual frequencies highlights the rain and snow in image space.

We perform a comprehensive comparison of this work with other methods of rain and snow detection and removal. We compare each algorithm both on the amount of rain/snow removed versus background corrupted and on how much the removal increases the accuracy of a feature point tracker. Six sequences are tested, half from real storms and half with realistically rendered rain added. The results demonstrate the advantage of image-global rain and snow analysis.

Once detected, we are then able to either decrease the amount of rain and snow by subtraction, or increase it by sampling and cloning. Other researchers have also developed methods for rain and snow synthesis. Rain can be generated via various approximate methods (Langer and Zhang 2003; Langer et al. 2004; Reeves 1983; Starik and Werman 2003; Tatarchuk and Isidoro 2006), but physically accurate rain synthesis (Garg and Nayar 2006) involves accurately modeling how a raindrops deforms and refracts light as it falls. Well-designed rain textures combined with a particle system can be used to create realistic scenes (Tariq 2007).

The advantage of combining detection, removal, and synthesis is that when the scene is uniformly illuminated, no additional scene analysis is required; the streaks are already correctly formed and illuminated. Instead of using separate tools to remove and to render rain and snow, we present a framework that does both.

2 Image-Space Analysis

A frame from a movie m acquired during a storm can be decomposed into two components: a clear image c and a rain/snow image r. Generally, a background scene point is occluded by raindrops or snowflakes for only a short time, therefore we can approximate their effect as being purely additive. For location (x, y) at time t, we have:

$$m(x, y, t) \approx c(x, y, t) + r(x, y, t) \tag{1}$$

In this paper, we develop an algorithm to find r, based on the overall appearance and statistical properties of rain and snow. Although it is sometimes possible to create clear videos by increasing the camera aperture and exposure time (Garg and Nayar 2005), this paper focuses on cases where this is not possible, such as when the entire scene needs to be in focus or when there are fast-moving objects that should not be blurred. When in focus and not blurred by a long exposure time, rain and snow appear in images as bright streaks. We begin the analysis in image-space by creating an appearance model of a streak.

2.1 The Shape of a Rain or Snow Streak

Raindrops and snowflakes can have complex shapes. However in a typical video sequence, their shapes are not prominently visible, therefore we ignore any variation in their shape and consider them to be symmetric particles. At a given instant in time, a camera with focal length f images an in-focus particle of diameter a at a distance from the camera z as an image with breadth b:

$$b(a, z) = a\frac{f}{z} \tag{2}$$

If a particle is not in focus, then its image will be broader. For the purposes of image analysis, out of focus raindrops or snowflakes are less important than in-focus ones, because they have a milder effect on images. (The appearance model in Sect. 2.2 implicitly handles slightly out-of-focus particles.)

The lengths of streaks depends on how fast the particles are falling and how far they are from the camera. Because they are so small, wind resistance is a major factor, and their terminal velocities depend on their sizes. For common altitudes and temperatures, a raindrop's speed s can be approximated by a polynomial in its diameter a (Foote and duToit 1969):

$$s(a) = -0.2 + 5.0a - 0.9a^2 + 0.1a^3 \tag{3}$$

Finding the speed of snowflakes is more difficult (Magono and Nakamura 1965; Böhm 1989), because they have more complex shapes. But since our detection algorithm uses a range of streak sizes, it is not necessary to obtain exact bounds on individual snowflakes. As a result, snowflakes can be assumed to fall half as fast as raindrops of similar size. If the ratio between size and speed is approximately correct, then the streaks can still be detected.

If the camera and particle are moving at constant velocities and the particle stays at a uniform distance, then it will be imaged as a straight streak. A falling particle imaged over a camera's exposure time e creates a streak of length l:

$$l(a, z) = (a + s(a)e)\frac{f}{z} \tag{4}$$

2.2 The Appearance of a Rain or Snow Streak

The appearance of a raindrop or snowflake depends on the particle's shape and reflectance, and the lighting in the environment. As shown in Fig. 2(a), under common lighting conditions, a falling drop will produce a horizontally symmetric streak.

A completely accurate prediction of a streak's coloring would require extensive physical modeling, as was done to render rain in Garg and Nayar (2006), and is shown in Fig. 2(c) and (d). But in most cases, the breadth is only a few pixels, and it is not necessary to form an exact model of light reflecting off a snowflake or determine the exact distorted image that will appear in a tiny drop (Garg and Nayar 2004; Van de Hulst 1957). Instead, we use a simple, analytical model that is fast to compute and well-behaved in frequency space.

To begin, the image of a raindrop or snowflake is approximated as a Gaussian, which appears similar to a slightly out-of-focus sphere. As the particle moves in space, the image it creates is a linear motion blurred version of the original Gaussian. If the sphere is larger or closer to the camera, the Gaussian will have a higher variance. If it is falling faster, then it will be blurred into a longer streak. The equation of a blurred Gaussian g, centered at image loca-

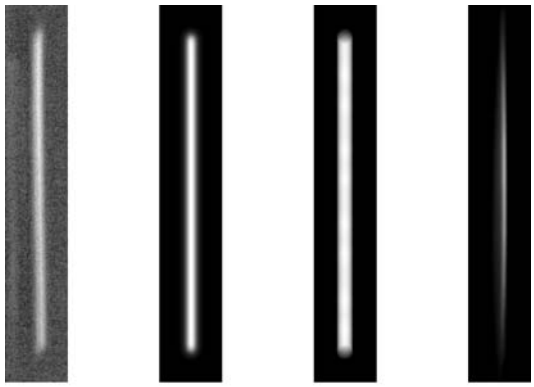

Fig. 2 Raindrops and snowflakes create streaks of different appearances, depending on factors such as the environmental illumination, their depth from the camera, and how much they are in focus. (**a**) A streak from a real water drop under illumination from a broad source. (**b**) The streak's appearance can be modeled by a blurred Gaussian (5). (**c**) A rendered streak from Garg and Nayar (2006) with broad environmental lighting. If the lighting is from a point source, then the streak would appear as in the point lighting example (**d**), which is also from Garg and Nayar (2006). In this paper, we use the blurred Gaussian, because it has approximately the correct appearance and is efficient to compute

tion $\boldsymbol{\mu} = [\mu_x, \mu_y]$, with orientation θ, variance given by the breadth b of the streak, and motion blurred over the length l of the streak, is given by:

$$g(x, y; a, z, \theta, \boldsymbol{\mu}) = \int_0^{l(a,z)} \exp\left(-\frac{(x - \cos(\theta)\gamma - \mu_x)^2 + (y - \sin(\theta)\gamma - \mu_y)^2}{b(a,z)^2}\right) d\gamma \tag{5}$$

The values for diameter a and depth z are combined with (2), (3), and (4) to compute the correct values of breadth b and length l. In the notation, a semicolon is used to differentiate between the parameters of image location versus all others. For example, $(x, y; a, z, \theta, \boldsymbol{\mu})$ means at location (x, y), with parameters $a, z, \theta, \boldsymbol{\mu}$.

An example of this appearance model is shown in Fig. 2(b). With broad environmental lighting from the sky, the variations due to the drop oscillations discussed in Garg and Nayar (2006), Tokay and Beard (1996), Kubesh and Beard (1993) are subtle, so Fig. 2(a), (b), and (c) appear almost identical. Even though blurred Gaussians are an inaccurate approximation of raindrops illuminated with a point light source, most outdoor scenes are not lit with point sources during the day, so the effects of oscillation can be ignored.

2.3 The Appearance of Multiple Streaks

The pixel intensity due to rain or snow in one movie frame at a given location (x, y) should be the sum of the streaks created by all N visible drops:

$$\sum_{d=1}^{N} g(x, y; a_d, z_d, \theta_d, \boldsymbol{\mu}_d) \tag{6}$$

For a given time t, each of a_d, z_d, θ_d, and $\boldsymbol{\mu}_d$ are drawn from different distributions. Drops are equally likely to appear at any location in space, so the x and y positions in $\boldsymbol{\mu}_d$ are drawn from uniform distributions. Because a greater volume is imaged further from the camera, more drops are liable to be imaged at greater depths, so z_d is drawn from a simple quadratic distribution. Streak orientation has a mean orientation θ_d, with a slight variance. The most problematic parameter is the drop size a_d. Fortunately, many researchers in the atmospheric sciences have studied the expected number of each size of raindrop or snowflake, and we draw upon their conclusions.

It is well known that in a single storm, there will be particles of various sizes. Size distributions are commonly used for raindrops (Marshall and Palmer 1948; Ulbrich 1983; Feingold and Levin 1986), snowflakes (Gunn and Marshall 1958; Ohtake 1965), and various other hydrometeors, such as graupel and hail (Auer 1970; Auer 1972). Previous works on rain removal (Garg and Nayar 2004, 2005) have used the Marshall-Palmer (1948) distribution. For more information, Microphysics of Clouds and Precipitation by Pruppacher and Klett (1997) is a good general resource for the physics of precipitation. Unfortunately, as discussed by several authors (Jameson and Kostinski 2001, 2002; Desaulniers-Soucy et al. 2001; Desaulniers-Soucy 1999), size distributions can be inaccurate. Nevertheless, they give useful general bounds. Both size distributions and observational studies show that drops rarely grow larger than 3 mm. In addition, drops smaller than .1 mm cannot be seen individually. Although not accurate for every storm, we find that using a uniform distribution between .1 mm and 3 mm is sufficiently accurate.

With all of the variables sampled from their distributions, generating images with rain or snow is straightforward with this model. But determining if part of an image is rain or snow would require a search across all (x, y), with each possible N, $\boldsymbol{\mu}_d$, a_d, and z_d. Performing this search in image space would be prohibitive, so we instead perform a simplified search in frequency space.

3 Frequency-Space Analysis

Since rain and snow streaks create repeated patterns, it is natural to examine them in frequency space. Rather than attempting to find each pixel of each streak, we can instead find their general effect on the Fourier transforms of the images. But applying the Fourier transform to (6) does

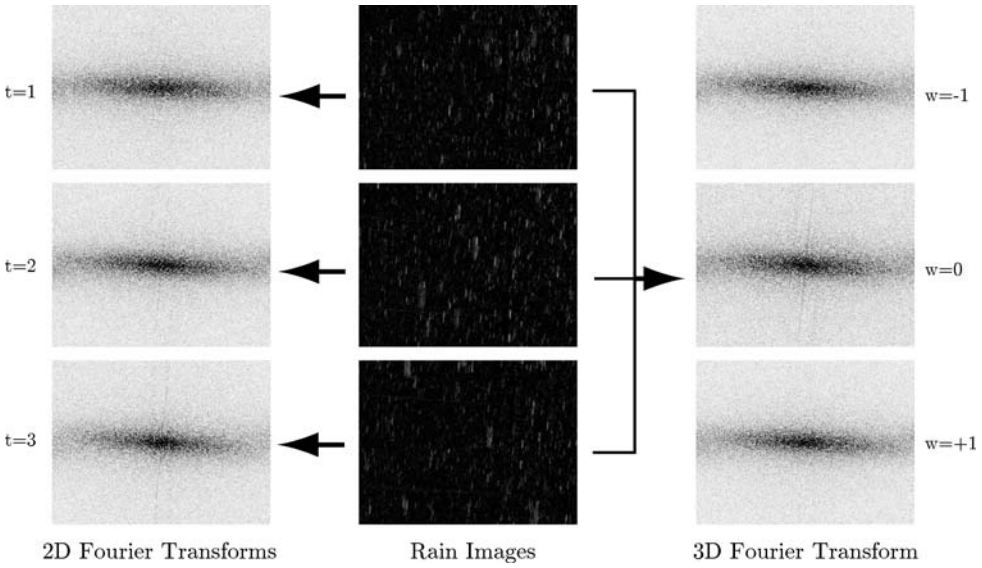

Fig. 3 The center column is three consecutive frames of rain acquired at times $t = 1, 2, 3$. The left column is three two-dimensional Fourier transforms, one for each of the images. The right column is a single three-dimensional transform of all three frames, with temporal frequency $w = -1, 0, 1$. As expected, the $w = 1$ and $w = -1$ frequencies are mirror images. But what is interesting is that *all* of the Fourier transform images appear similar, due to the statistical properties of rain

not make it easier to analyze images. For this, we make three key observations of the magnitude of the Fourier transform of rain and snow.

Observation 1 *The shape of the magnitude does not depend strongly on streaks' locations in an image*. Figure 3 shows an example of the Fourier transform of a sequence with real rain. The middle column is a sequence of three consecutive frames. They were generated from a sequence of heavy rain with a stationary scene and with an almost stationary camera, by finding the difference of each pixel with the median of itself and its two temporal neighbors:

$$|M(x, y, t) - \text{median}(M(x, y, t-1), M(x, y, t), M(x, y, t+1))| \tag{7}$$

The left column of Fig. 3 is three separate two-dimensional Fourier transforms, one for each image. Notice that even though the streaks are in different locations in different frames, the magnitudes appear similar. Appendix A contains a derivation and simulation that shows the magnitude is only weakly dependent on the number and positions of streaks. We find that although the expansion of the magnitude of the Fourier transform of rain and snow can be arbitrarily complex, it can still be well behaved, which explains the phenomenon seen in the left column.

Observation 2 *The shape of the magnitude is similar for different numbers of streaks*. Although the exact Fourier transforms of images with different numbers of streaks are different from each other, changing the number of streaks has a similar effect to multiplying all frequencies by a scalar. This pattern is shown in Fig. 4. Appendix A also contains a validation of this observation, for the special case where there are the same number of streaks of each length at each location.

Observation 3 *The magnitude is approximately constant across the temporal frequencies*. This rightmost column of Fig. 3 shows the three-dimensional Fourier transform of all three frames. Interestingly, apart from a few artifacts, the magnitude appears similar across temporal frequency w. This observation allows us to predict that the three-dimensional Fourier transform will be constant in temporal frequency w.

These observations will allow us to create a simplified model of the frequencies of rain and snow. Instead of finding every streak individually, we can fit this model by only estimating a few parameters.

3.1 A Frequency-Space Model of Rain and Snow

As shown in (6), an image full of rain or snow is the sum effect of a group of streaks. The same is true in frequency space, where the magnitude of the Fourier transform of (6) is:

$$\left\| \mathcal{F}\left\{ \sum_{d=1}^{N} g(x, y; a_d, z_d, \theta_d, \boldsymbol{\mu}_d) \right\} \right\| \tag{8}$$

which is equivalent to the sum of the Fourier transforms of each streak g:

$$\left\| \sum_{d=1}^{N} G(u, v; a_d, z_d, \theta_d, \boldsymbol{\mu}_d) \right\| \tag{9}$$

(Note that in this work, we use only the main lobe of the blurred Gaussian G, which has a similar appearance to a standard oriented Gaussian.)

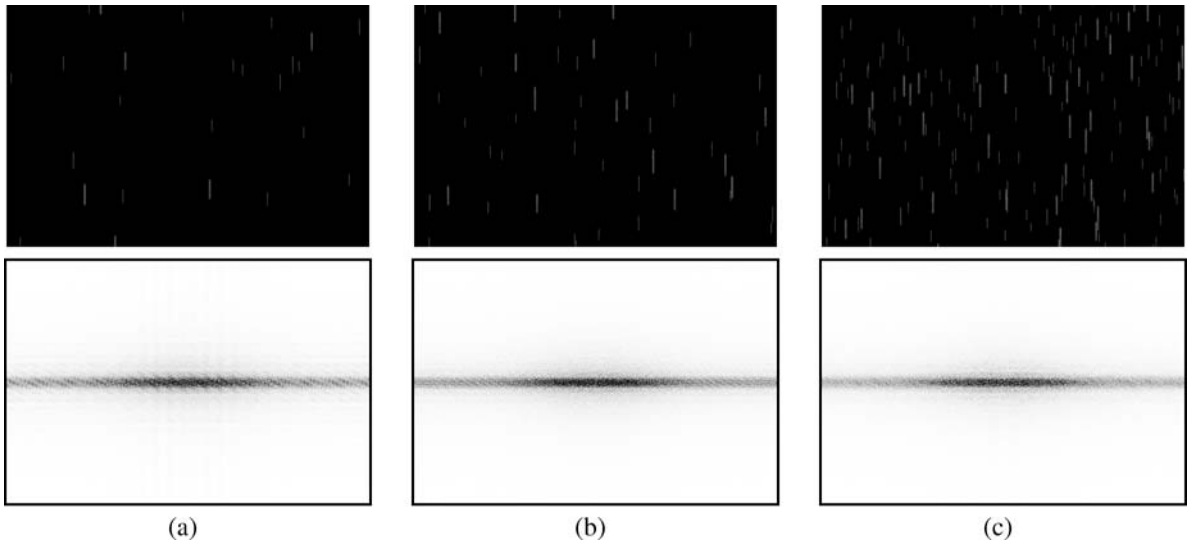

Fig. 4 Three examples of images with streaks rendered by Garg and Nayar (2006) and their corresponding two dimensional Fourier transforms. The images have approximately (**a**) 50, (**b**) 100, and (**c**) 300 streaks. To make them appear similar, each Fourier transform is multiplied by a scalar. Apart from being scaled differently, their magnitude appear similar

Based on the Observations 1 and 2 in the previous section, (9) can be simplified as:

$$\sum_{d=1}^{N} \|G(u, v; a_d, z_d, \theta_d)\| \tag{10}$$

Equation (10) is simpler, but still depends on the number of streaks N in the image. This is where the statistical properties of rain and snow discussed in Sect. 2.3 become helpful. Since determining the exact value of each frequency is not vital, we can simplify (10) further, based on three assumptions. First, each spatial location $[z_{min}, z_{max}]$ is equally likely to have a raindrop or snowflake. In a perspective camera, the volume imaged at a given depth is relative to the depth squared. This means that in a perspective camera, the number of drops imaged at a given depth will also be relative to the depth squared. Second, the resulting streaks are equally likely to have any orientation within the range $[\theta_{min}, \theta_{max}]$. Third, a given particle is equally likely to be any size between $a_{min} = .1$ mm and $a_{max} = 3$ mm.

Instead of trying to determine the properties of each of the N streaks, we use a model R^* that has frequencies proportional to the mean streak and scaled by overall brightness Λ:

$$R^*(u, v; \Lambda, \theta_{max}, \theta_{min}) = \Lambda \int_{\theta_{min}}^{\theta_{max}} \int_{a_{min}}^{a_{max}} \int_{z_{min}}^{z_{max}} z^2 \|G(u, v; a, z, \theta)\| dz\, da\, d\theta \tag{11}$$

These integrals can be approximated by sampling across θ, a, and z, yielding an estimate of the frequencies of rain and snow.

From Observation 3, we can predict that the magnitude will be constant in temporal frequency w:

$$R^*(u, v, w; \Lambda, \theta_{max}, \theta_{min}) = R^*(u, v; \Lambda, \theta_{max}, \theta_{min}) \tag{12}$$

The scalars for rotation θ and brightness Λ are based on the specific movie. In the next section, we show how to fit θ and Λ.

3.2 Fitting the Frequency-Space Model to a Video

Only a single intensity Λ needs to be estimated per frame, and often only one orientation θ per sequence. To estimate these parameters, we can use the fact that rain and snow cover a broad part of the frequency space. Most objects are clustered around the lowest frequencies, while rain and snow are spread out much more evenly. This means that even if the total energy of the rain or snow is low, a frequency chosen at random is fairly likely to contain a strong rain or snow component. This is especially true if we only examine the non-zero temporal frequencies, which are those that correspond to changes between frames.

The model parameters can be estimated with two heuristics. The scalar multiplier Λ should be such that the rain/snow model is approximately the same magnitude as the rain or snow in the movie. For the Fourier transform of a small block of frames, Λ can be estimated by taking a ratio of the median of all frequencies, except for the constant temporal frequencies $w = 0$:

$$\Lambda \approx \frac{\text{median}(\|M(u, v, w)\|)}{\text{median}(R^*(u, v, w; \Lambda = 1, \theta_{max}, \theta_{min}))} \tag{13}$$

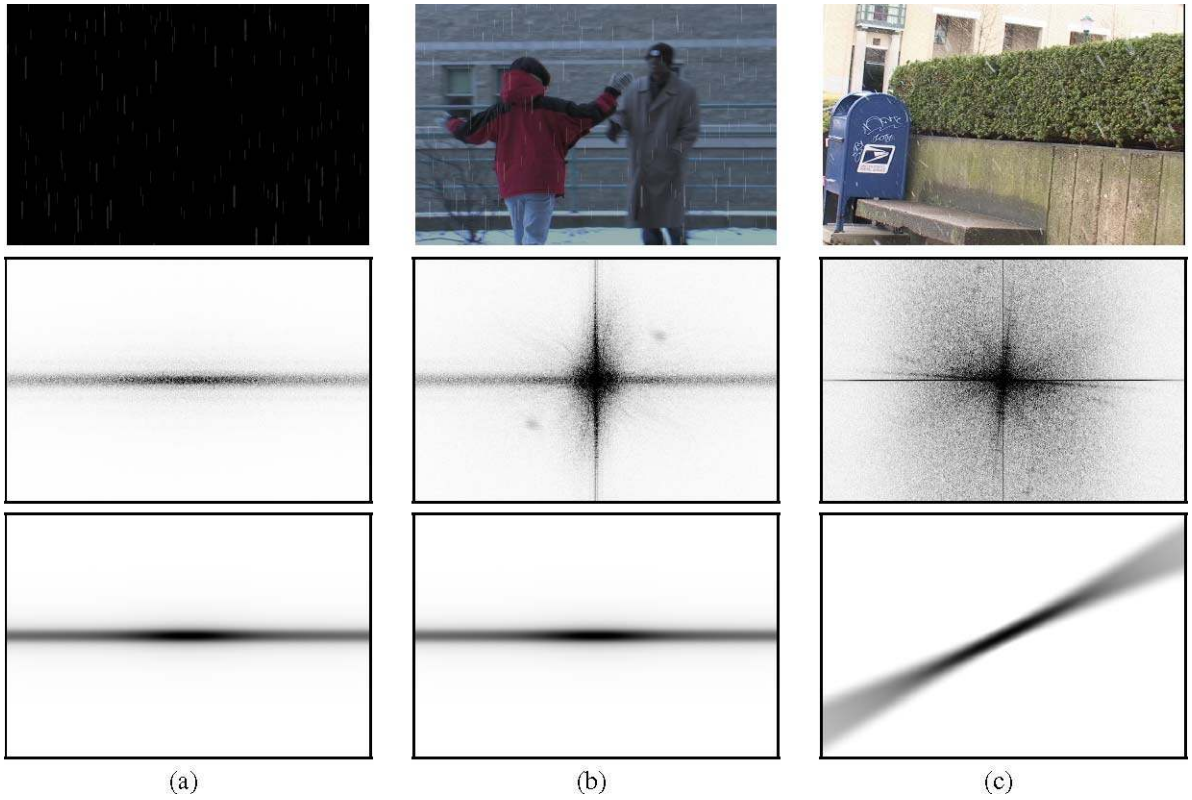

Fig. 5 Examples of the model for three video sequences. From *top* to *bottom*, we have the original image, its two dimensional Fourier transform, and the corresponding rain/snow model. (**a**) A sequence of a black background, plus 300 rendered streaks per image. (**b**) The same 300 streaks, but now against a moving background with a moving camera. (**c**) Real snow and a moving camera

Taking the median is effective, because as discussed in Observation 3 in the beginning of the section, rain and snow are strong in non-zero temporal frequencies, while most of the scene is concentrated in the zero temporal frequencies.

The streak orientation can be automatically computed if there is a short subsequence where only rain and snow are moving. Again using Observation 3, we expect that individual rain and snow frequencies will change greatly, even though their overall effect stays the same. To find orientation, we do not need to find the correct values for each frequency, we only need to determine which are due to rain and snow. Therefore, rather than using the median of the frequencies as in (13), we use the standard deviation across time as a more robust estimator. An estimate $\tilde{R}$ of the important frequencies can be obtained by computing the standard deviation over time for each spatial frequency, for T frames:

$$\tilde{R}(u,v) = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(\|M(u,v,t)\| - \|\overline{M}(u,v)\|)^2} \tag{14}$$

The correct θ is found by minimizing the difference between the model and the estimate:

$$\underset{\theta}{\operatorname{argmin}} \iint (\|R^*(u,v;\Lambda,\theta_{max},\theta_{min})\| - \tilde{R}(u,v))^2 dv\,du \tag{15}$$

Because the search space is one dimensional and bounded, an exhaustive search can be performed. Different raindrops generally fall in almost the same direction, so $\theta_{min} = \theta_{max}$ for rain. But since snow has a less consistent pattern, a range of orientations is needed. Using $\theta_{min} = \theta_{max} - .2$ radians is effective for most videos with snow.

Figure 5 shows the models obtained by fitting to three videos. The frequencies corresponding to the rain are easy to see in (a) and (b), but it easier to see the snow frequencies in the movie for (c) available at http://www.cs.cmu.edu/~pbarnum/rain/barnum08frequency.html.

4 Applications

The model that we developed in the previous sections can be used to either decrease or increase the amount of rain and snow. For both cases, the first step is detection, which requires an analysis across entire images, and is performed in frequency space. Once detected, the rain or snow can either be directly removed by subtraction, or else the detected

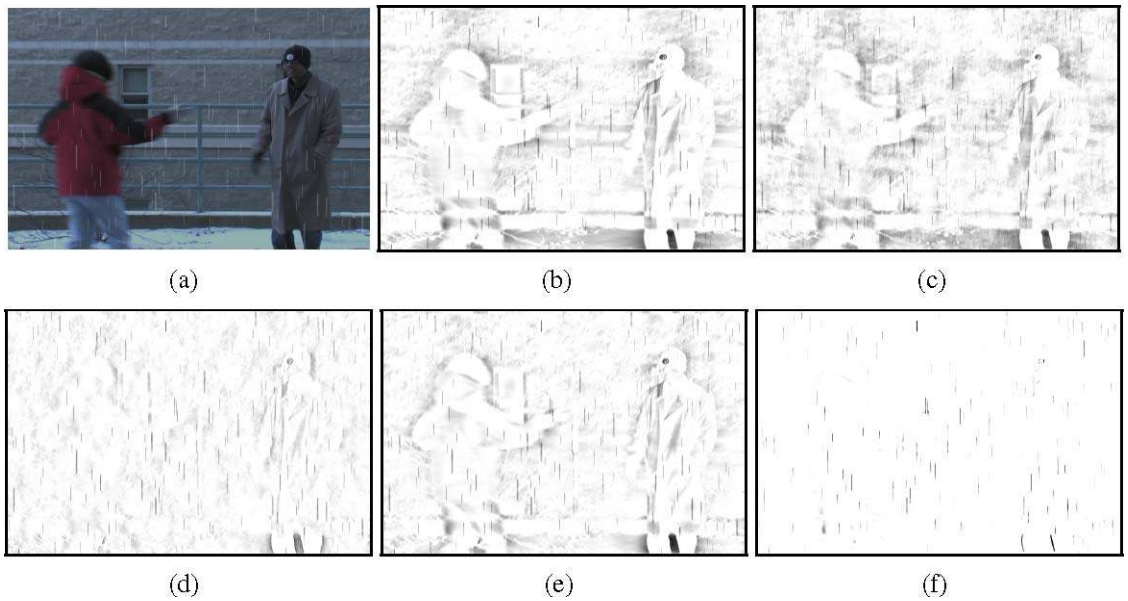

Fig. 6 Rain can be detected in several ways, with the same frequency model. Subfigure (**a**) shows a frame from the original sequence, which has rendered rain streaks. (**b**) With detection based on a single frame, the rain is segmented fairly accurately, but there are many false detections. Even the fairly textureless ground is mistakenly detected, because it shares many of the low frequencies of the model. In (**c**), detection still uses a single frame, but a random 50% of the model's frequencies are set to zero. (The effect of setting some frequencies to zero is more evident in the videos on the website.) The true magnitude of the rain is used instead of our model in (**d**). The rain is still detected accurately, although there are fewer erroneous detections. The reason the ground truth magnitude has any errors is because our method of computing the rain/snow component does not generally allow a complete separation of rain/snow and the clear image. This example shows the theoretical limit of using a ratio of magnitudes for a single frame. (**e**) Shows detection based on three consecutive frames, with similar accuracy to one frame. The best results are in (**f**) when detection is performed on a single frame, then refined over three frames. The exact frequencies of rain and snow change from frame to frame, and using the two step estimation finds only those frequencies that are both rain-like and rapidly changing

pixels can be blended with their temporal neighbors. Alternately, to increase the amount of rain or snow, individual streaks can be found by matching with blurred Gaussian in image space and then copied onto another image.

4.1 Detecting Rain and Snow Using Frequency Space Analysis

The frequency model can be used to detect rain and snow in a similar way to notch filtering (Gonzalez and Woods 2002). Intuitively, we want to highlight those frequencies corresponding to rain and snow while ignoring those corresponding to objects in the scene. This can be done with a simple ratio. For example, suppose that the model predicts a low value for a given frequency, but the actual value is high. Something besides rain or snow is likely causing the high value. The frequencies that are mostly due to rain and snow should be found first, as estimated by the ratio of the predicted value to the true value.

Detecting streaks in a single frame is done by taking the inverse transform of the estimate of the proportion of energy due to rain or snow. Where $M(u, v)$ is the two dimensional Fourier transform of one movie frame and ϕ is the phase of $M(u, v)$, p_2 is the estimate based on a single image at time t:

$$p_2(x, y, t) = \mathcal{F}^{-1}\left\{ \frac{R^*(u, v; \Lambda, \theta_{max}, \theta_{min})}{\|M(u, v)\|} \times \exp(i\phi\{M(u, v)\}) \right\} \tag{16}$$

The output is an image that is bright only where rain or snow is detected. When $R^*(u, v)$ is less than $M(u, v)$, then the ratio is the estimated percentage of rain and snow at that frequency. For example, for a given (u, v), if $R^*(u, v) = 3$ and $M(u, v) = 10$, then $R/M = .3$. This means that we believe that thirty percent of the energy at (u, v) is due to rain and snow. If $R^*(u, v) > M(u, v)$, then the ratio is greater than one, which is not meaningful. The ratio of R^* over M is therefore capped at one. This capping is both semantically valid as well as practical, in that it prevents frequencies with a very high value for R^*/M from dominating the result.

Figure 6(b) shows the one-frame estimation. For visual comparison, Fig. 6(c) shows the result if 50% of the frequencies in the model are set to zero before using (16). Figure 6(d) shows the result if the ground truth of the rain magnitude is used in place of the rain model.

By performing a three-dimensional transform of the images and using (u, v, w) instead of (u, v), (16) can be used for a three dimensional transform of multiple consecutive frames, shown in Fig. 6(e). Using multiple frames improves the accuracy, but not significantly.

We found through experimentation that the best approach is to perform a three dimensional analysis on a series of consecutive two dimensional estimates. A three dimensional Fourier transform is applied to $p_2(x, y, t)$ to obtain $P_2(u, v, w)$, and the resulting rain/snow estimation is then:

$$p_3(x, y, t) = \mathcal{F}^{-1}\left\{ \frac{R^*(u, v, w; \Lambda, \theta_{max}, \theta_{min})}{\|P_2(u, v, w)\|} \times \exp(i\phi\{M(u, v, w)\}) \right\} \quad (17)$$

Figure 6(f) shows the results from this method. At first glance, it appears even better than the single frame ground truth in Fig. 6(d). But the ground truth magnitude actually correctly identifies streaks more precisely, even if it has more false detections. But since the ground truth is not generally known, we use p_3 as our final estimate of the location of the rain and snow.

4.2 Reducing Rain and Snow Using the Frequency Space Model

Once detected, the rain or snow pixels can be removed by replacing them with their temporal neighbors. The detected rain and snow $p_3(x, y, t)$ is used as a mixing weight between the original image m and an initial estimate $\tilde{c}$ of the clear image c. We find that a per-pixel temporal median filter works well for $\tilde{c}$, although it could be the output of any rain/snow removal algorithm. The detection $p_3(x, y, t)$ is multiplied by the removal rate α, where the product of $\alpha p_3(x, y, t)$ is capped at one:

$$c(x, y, t) = (1 - \alpha p_3(x, y, t))m(x, y, t) + \alpha p_3(x, y, t)\tilde{c}(x, y, t) \quad (18)$$

Since rain and snow are brighter than their background, $c(x, y, t)$ is required to be less than or equal to $m(x, y, t)$. For a large α, $\alpha p_3(x, y, t)$ equals 1 for all (x, y, t), therefore c will approach $\tilde{c}$.

Images created with this equation will be temporally blurred only where the rain and snow is present. But the disadvantage is that it can never remove more rain and snow than the initial estimate $\tilde{c}$. If removal is more important than smoothness, then we can iterate the detection and removal.

The first iteration c^1 is the result from (18) on the original sequence. Subsequent iterations are based on the last clear estimate c^{n-1} and the last initial estimate $\tilde{c}^{n-1}$. In this case, $\tilde{c}^{n-1}$ is the per-pixel temporal median of c^{n-1}. Where p_3^n is the detection from (17) as applied to c^{n-1}, the next iteration is:

$$c^n(x, y, t) = (1 - \alpha p_3^n(x, y, t))c^{n-1}(x, y, t) + \alpha p_3^n(x, y, t)\tilde{c}^{n-1}(x, y, t) \quad (19)$$

Selecting a good value for α is not difficult. We use a fixed $\alpha = 3$ for all the results in this paper. And as with (18), each iteration is required to be less than or equal to the original.

Although the result from each subsequent iteration is more clear than the previous, the amount of rain and snow removed decreases per iteration. This means that it may be necessary to iterate many times to remove most of the streaks. Since this is time consuming, the process can be iterated only a few times, and subsequent c^ns can be linearly extrapolated from the final two iterations. Figure 7 shows the results from the iterative removal method on four example sequences with moving cameras and scenes.

4.3 Increasing Rain and Snow in Image Space

Although the rain and snow can be detected using only the frequency magnitude, creating new streaks requires manipulation of phase as well. The main advantage of working in frequency space was that the locations of the streaks could be ignored. But since we need them for rendering, it is simpler to work in image space. Our approach is to use the blurred Gaussian model to sample real rain and snow.

We start with the rain/snow estimate $m(x, t, y) - c^n(x, y, t)$. Large streaks are detected by filtering the rain/snow estimate with a bank of size derivatives of blurred Gaussians, similar to scale detection in (Mikolajczyk and Schmid 2001). For an average a and z with orientation θ, the size derivative is given by:

$$f(x, y; \gamma_1, \gamma_2, a, z, \theta) = g(\gamma_1 x, \gamma_1 y; a, z, \theta) - g(\gamma_2 x, \gamma_2 y; a, z, \theta) \quad (20)$$

where γ is a scalar, $\gamma_1 < \gamma_2$, and both of the blurred Gaussian terms are normalized to sum to one. Each image is filtered with a set of different γs. The filter with the maximum response corresponds to the size of the streak at that location.

This detection method will find many strong streaks, but will have a few false matches as well. Since we do not need to find every streak, several steps are taken to cull the selection. First, locations that appear to have very large and very small scales are eliminated, and non-maxima suppression is performed in both location and scale. Next, to ensure that only bright streaks are used, only the streak candidates with the most total energy are kept. (The total energy is the

(a) The mailbox sequence: There are objects at various ranges, between approximately 1 to 30 meters from the camera. The writing on the mailbox looks similar to snow. Most of the snow can be removed, although there are some errors on the edges of the mailbox and on the bushes

(b) Walkers in the snow: This is a very difficult sequence with a lot of high frequency textures, very heavy snow, and multiple moving objects. Much of the snow is removed, but the edges of the umbrella and parts of the people's legs are misclassified

(c) Sitting man sequence: This scene is from the movie Forrest Gump. The rain streaks are fairly large, as is common in films. The rain can be completely removed, although the letters and windows in the upper portion of the images are misclassified

(d) A windowed building: The rain is not very heavy, but this sequence is difficult, because there are a large number of straight, bright lines from the window frames and the branches. Almost all of the rain is removed, but parts of the window frames and the bushes are erroneously detected

Fig. 7 Several examples of rain and snow removal based on spatio-temporal frequency detection. Some of the sequences have several moving objects, others have a cluttered foreground with high frequency textures, and all of them are taken with a moving camera

Fig. 8 (**a**) An image from the original mailbox sequence and one with added snow. The snow is removed and then sampled, yielding a streak database. (**b**) The streaks are then added to the new image, increasing the amount of snow

sum of the values of neighboring pixels within a small window.) Finally, to prevent multiple off-center copies of the same streak, only the streak candidates that have the greatest percentage of their energy near their centers are kept. For a window of size (s_x, s_y), the energy near the center of a streak at (x, y, t) is given by:

$$\frac{\sum_{d_x=-s_x/2}^{s_x/2} \sum_{d_y=-s_y/2}^{s_y/2} Q(d_x, d_y)\, m(x+d_x, y+d_y, t)}{s_x s_y \sum_{d_x=-s_x/2}^{s_x/2} \sum_{d_y=-s_y/2}^{s_y/2} m(x+d_x, y+d_y, t)} \tag{21}$$

where

$$Q(d_x, d_y) = \sqrt{\left(\frac{s_x}{2}\right)^2 + \left(\frac{s_y}{2}\right)^2} - \sqrt{d_x^2 + d_y^2}$$

The images of streaks of various sizes are then combined into a database. Optionally, artifacts can be reduced by projecting the magnitude of the sampled streak onto the magnitude of the blurred Gaussian streak model.

Once the database is created, it can be used in the same way as the database of streaks rendered with area-source environmental lighting from Garg and Nayar (2006). The advantage of our method is that the sampled streaks already have natural variation in size and defocus blur. The disadvantage is that our method has no concept of lighting direction, so will not be accurate for night scenes where drop oscillations create complex specular effects. For scenes with area-source illumination from the sky, streaks from both our work and Garg and Nayar (2006) can be added in the same way.

Since the main focus of this work is on rain and snow detection during the day, we show only examples where the camera exposure is short and the scene is well-illuminated by the sky. Given an approximate depth map of the scene, the streaks can be rendered with the appropriate sizes and densities. However, most of the visible streaks are within the depth of field of the camera, we sample uniformly in this volume to create both the spatially varying example in Fig. 1 and the full-frame example in Fig. 8.

5 Comparison of Rain and Snow Detection and Removal Methods

Various methods have been proposed to remove rain and snow from images (Hase et al. 1999; Starik and Werman 2003; Garg and Nayar 2004; Zhang et al. 2006). Ideally, a removal algorithm should output images of the scene as it would appear with no bad weather effects. No algorithm to date can completely clear an image without corrupting the background, but some are more effective than others. In this section, we quantitatively compare and qualitatively discuss the accuracy of each method on several sequences with real and rendered rain and snow.

5.1 Evaluation Methodology

Each algorithm is compared quantitatively in two ways. First, we compare the amount of rain and snow removed versus the amount of the images incorrectly modified. Second, we run a feature point tracker on both the original sequence and the output of each algorithm, and compare the number of feature points correctly tracked.

We test each algorithm on three sequences with real snow (Fig. 12) and three with rendered rain (Fig. 11).

(a) (b)

Fig. 9 Two example frames and detection results for Zhang et al. (2006). (**a**) k-means can often correctly segment streaks, although there are errors around strong gradients and very bright parts of the images. (**b**) In this other frame, the grayvalue intensity of the mailbox increased by approximately 10, which causes additional false positives. Only the false detections around strong gradients cause noticeable errors in the snow-removed images, because areas with weak gradients do not change when median filtered

Each sequence is either 720×480 or 640×480 pixels, and all are 60 frames long. Since the various algorithms require between three to thirty frames to initialize, only the accuracy on frames 30–59 is evaluated.

In the real sequences, the camera had a small aperture and a short exposure time, which causes bright, well-defined streaks. The streaks in the rendered sequences are generated with the photorealistic process of Garg and Nayar (2006). We use the streaks rendered with large area source environmental-lighting instead of point-lighting, because our scenes are illuminated by the sun.

5.1.1 *Quantifying Removal Accuracy*

The first metric is a per-pixel comparison of the amount of rain/snow removed compared to the amount of the background erroneously changed. (All equations are given for grayscale, although algorithms are evaluated by separately computing the error for each channel and averaging.)

Each algorithm outputs an estimate of the true brightness of the rain at each pixel $\tilde{r}$:

$$\tilde{r}(x, y, t) = m(x, y, t) - c(x, y, t) \tag{22}$$

Since adding rendered rain to an image only increases the image brightness, a removal algorithm should ideally either decrease the image brightness or leave it constant. If the image is darkened, then the difference D between the true rain component r and the estimate $\tilde{r}$ is the arithmetic difference. And since any increase is an error, if the image is brightened, the difference D is how much the removal algorithm increased the image brightness:

$$D(x, y, t) = \begin{cases} r(x, y, t) - \tilde{r}(x, y, t) & \tilde{r}(x, y, t) > 0 \\ \tilde{r}(x, y, t) & \tilde{r}(x, y, t) < 0 \end{cases} \tag{23}$$

Once the difference D has been computed, each algorithm's accuracy is determined. H is the ratio between the amount of rain not removed and the total rain present. E is the ratio between the amount of the background incorrectly changed and the background's total energy:

$$H = \frac{\sum_{x,y,t}\{D(x, y, t) : D(x, y, t) > 0\}}{\sum_{x,y,t} r(x, y, t)} \tag{24}$$

$$E = -\frac{\sum_{x,y,t}\{D(x, y, t) : D(x, y, t) < 0\}}{\sum_{x,y,t} c(x, y, t)} \tag{25}$$

Figure 11 shows the removal accuracy for each method in the following sections. Methods that use a threshold can have varying values for H and E, and are plotted as lines. Specifics of the results are discussed in Sect. 5.2.

5.1.2 *Quantifying Feature-Point Tracking Accuracy*

The second metric is the increase or decrease in the number of feature points that can be tracked. Feature point tracking accuracy is selected as a comparison for several reasons. First, it does not require ground truth like the per-pixel accuracy evaluation, so sequences with real rain and snow can be used. Second, quantitative evaluation is simple; accuracy is the number of points correctly tracked. Third, tracking accuracy should be correlated with an algorithm's accuracy in preserving and revealing high frequencies in the scene.

Fig. 10 One frame each from two sequences, and its corresponding correlation magnitude (Garg and Nayar 2004) (scaled linearly for display). (**a**) For the stationary sequence of the reflective ball, all of the current streaks are visible in the magnitude image, plus ghosts from streaks in the previous frames. (**b**) The general characteristics of the scene can be seen in the magnitude image, but because the camera is moving, the scene appears ghosted

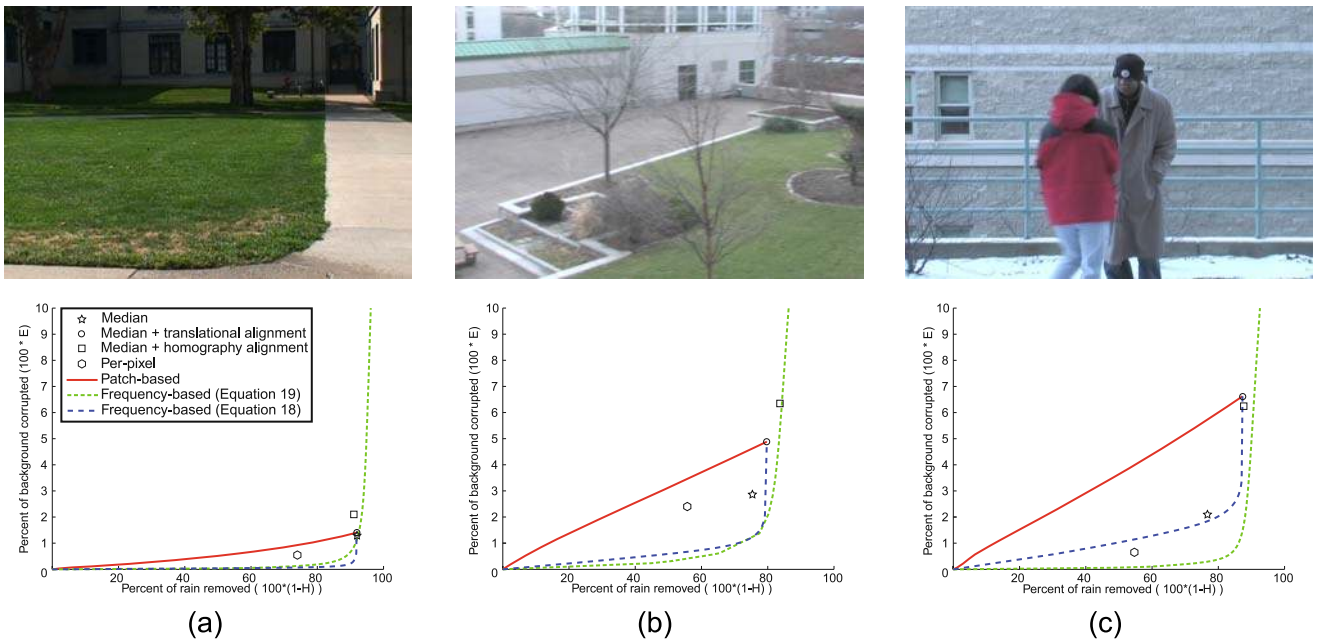

Fig. 11 Three sequences are used for tests with rendered rain. The origin corresponds to unmodified images, and the lower right corresponds to perfect removal accuracy. As the thresholds for the patch-based and frequency-based (18) are changed, they become more and more similar to the median. The per-pixel accuracy will always lie somewhere between the original image and the aligned median, depending on how many pixels are detected as rain. (**a**) Grass lawn: A stationary camera views a grass lawn and a few buildings. All algorithms are able to remove most of the rain with little background corruption. (**b**) Park and patio: A rotating camera was used to acquire this sequence of a small park and patio. The scene is mostly stationary, except for the trees waving slightly in the breeze. Results are similar to the stationary grass lawn, except all algorithms have larger error. (**c**) Two friends: This video of two people greeting each other was acquired with a moving camera. Because the foreground motion causes errors in the automatic alignment, the median actually performs better without alignment. But even if the background was fully aligned, the foreground motion would cause low scores for the median filter methods. And because there are large blocks of uniform color, per-pixel detection is able to correctly classify most of the pixels, yielding a low corruption score. For all sequences, the frequency-based removal performs the best

We compare the results of tracking feature points in all six of the sequences mentioned in Sect. 5.1 and their corresponding de-weathered versions. The features are selected using the method of Shi and Tomasi (1994), and tracked using a Lucas-Kanade tracker (Bouguet 2000). The strongest features are selected independently for the original sequences and the outputs of the removal algorithms.

We use the same evaluation method as Sand and Teller (2006), which is to track points while the sequence is played forward then backwards. Since the sequence starts and ends on the same frame, each point should be in the same location at the beginning and the end. Tracking accuracy is defined as the distance between each point at the beginning and end of the loop. For algorithms that use a threshold, results are reported at the threshold where the highest number of points are tracked within one pixel of accuracy.

Results are reported for the number of points tracked to within one pixel and five pixels of accuracy. We report both

(a) (b) (c)

Fig. 12 Real sequences of rain and snow. (**a**) Reflective sphere: This scene is of a reflective sphere, acquired with a stationary camera. The entire scene changes very little, except for a light amount of snow. (**b**) Snowy mailbox: This sequence of a mailbox, bushes, and building is acquired with a moving camera. The camera motion is mostly rotational, and the images can be aligned reasonably well with a planar homography. (**c**) Pedestrians in the snow: With multiple people walking, heavy snow, and a moving camera, this sequence is the most complex.

numbers, because rain and snow more often cause point tracks to be slightly dislodged than completely lost. The necessary accuracy depends on the application. Structure from motion requires points tracked within one pixel of accuracy, but five pixels of accuracy is sufficient for object tracking. Figure 13 has point tracking results for all methods, with details discussed in the next section.

5.2 Explanation and Evaluation of Algorithms Used in the Comparison

Each detection and removal algorithm has three steps. First, using some combination of image processing, machine learning, and physical models, pixels are clustered into two categories: rain and non-rain. Second, an initial estimate of the true background is obtained as a temporal average or median. Third, pixels detected as rain are either partially or completely replaced by a pixel from the initial estimate. The first step is the main difference between methods, but each computes the initial estimate in a slightly different way. We use the temporal median filter with image alignment as the initial estimate for all algorithms, which allows for an indirect quantitative evaluation of detection accuracy.

5.2.1 No Explicit Detection

In some cases, it is not necessary to explicitly detect rain and snow in order to remove them. Temporal median filtering is the simplest method for cleaning videos (Hase et al. 1999; Starik and Werman 2003). For this method, each pixel is replaced with the median of its values over time. If the scene and camera are completely static, then this is often the most accurate and visually pleasing way of removing rain and snow.

The main advantage of this method is that it is extremely fast, but if the camera is not stationary or there are moving objects, then median filtering performs poorly. The problems are most noticeable when tracking feature points.

Image alignment can increase the accuracy significantly. To align frames, we perform RANSAC (Fishler and Bolles 1981) on SIFT (Lowe 2004) features, computing either the image translation or a full homography between images. Interestingly, in some difficult cases, translational alignment can give more accurate results. This is likely because correcting for translation only requires the values of the x and y offsets to be found, which is less prone to errors than computing a full homography. In both cases, the aligned image is made the same size as the original. Pixels with unknown values are set to their corresponding value in the unaligned image.

The point tracking results in Fig. 13 are evidence of the usefulness of this simple image alignment. But unless the alignment is accurate, errors are visible in areas with strong gradients. These errors result in low scores in comparisons between the rain-removed and background-corrupted comparisons. However, the strongest features are kept and enhanced, allowing for superior tracking results. It is interesting that aligning can increase performance even in stationary scenes. This appears to be because strong, repeated features are aligned first and then median filtered, enhancing them rather than blending them with noise. On simple stationary scenes, translation-only alignment has better 5-pixel accuracy, while homography alignment has better 1-pixel accuracy. But in general, translation-only alignment is more accurate, and should be used for applications requiring either high accuracy or large numbers of point tracks.

It would be interesting to test other types of image alignment. Computing layers of motion with methods such as Torr et al. (1999), Ke and Kanade (2002), Zelnik-Manor et al. (2006), then aligning and filtering each layer separately could increase performance. In addition to layer extraction, techniques such as Gaussian mixture models (Stauffer and Grimson 1998), kernel density estimation (Elgammal et al. 2000), or robust PCA (de la Torre and Black 2001) could model the variation caused by rain and snow more effectively.

Method	Maximum tracking error 1 pixel	5 pixels
With no rain	838	953
With rain added	241	600
Median	750	927
Median+trans	766	924
Median+homog	817	920
Per-pixel	623	906

(a) Grass lawn

Method	Maximum tracking error 1 pixel	5 pixels
With no rain	729	814
With rain added	95	424
Median	202	698
Median+trans	618	789
Median+homog	597	802
Per-pixel	292	684

(b) Park and patio

Method	Maximum tracking error 1 pixel	5 pixels
With no rain	304	343
With rain added	151	273
Median	192	346
Median+trans	249	367
Median+homog	215	363
Per-pixel	70	281

(c) Driving car

Method	Maximum tracking error 1 pixel	5 pixels
With rain	267	680
Median	684	980
Median+trans	676	982
Median+homog	720	977
Per-pixel	528	914

(d) Reflective sphere

Method	Maximum tracking error 1 pixel	5 pixels
With rain	349	659
Median	129	687
Median+trans	700	793
Median+homog	690	808
Per-pixel	636	795

(e) Snowy mailbox

Method	Maximum tracking error 1 pixel	5 pixels
With rain	131	394
Median	14	240
Median+trans	438	642
Median+homog	289	515
Per-pixel	183	453

(f) Pedestrians in the snow

Fig. 13 Results for feature point tracking. For each method, the columns signify the number of points that are tracked within 1 and 5 pixels of accuracy. Points tracked within 1 pixel are completely correct, while those within 5 pixels have drifted a small amount. Rain and snow tend to slightly disrupt point tracks more often then causing them to be completely lost, therefore we show both 1-pixel and 5-pixel accuracy. Results are reported for median filtering with no alignment, with translational alignment, and with homography alignment. For sequences where ground truth is available, accuracies with no rain are also displayed. In the limit, both the patch-based and frequency-based techniques are identical to the median and have identical tracking accuracy, so are not included in the charts. Results for each method are explained in more detail in their respective sections, from Sects. 5.2.1 to 5.2.4

5.2.2 Per-Pixel Detection

Instead of applying the same approach on all pixels, most techniques first determine which pixels are rain. This is similar to the idea of background subtraction, except the foreground layer is just the rain and snow.

If the only difference between aligned frames is rain or snow, then each pixel should have one of two values. In Zhang et al. (2006), k-means with two clusters is used on the grayscale intensity of each pixel over all frames. Since rain and snow are bright, the pixels corresponding to the cluster with the higher grayvalue intensity are tagged for removal.

Zhang et al. (2006) also discuss reducing false matches by using the fact that rain is normally has a neutral hue. Colorful pixels are unlikely to be rain. In practice, this generally requires hand tuning for each sequence, so it is not included in the comparison.

Two examples of applying this technique are shown in Fig. 9. In such scenes, the rain and non-rain clusters are not always well separated, and some pixels are misclassified. Although many pixels are correctly labeled, individual pixels sometimes flicker, causing the low numbers of tracked feature points shown in Fig. 13. The flickering causes points to be lost more often than slightly mis-tracked, causing low scores for both 1-pixel and 5-pixel accuracies. A tracker that uses more than two frames might not have as much difficulty with single frame impulses, but this would be true of unmodified rain and snow videos as well.

Proper application of morphological operators and blurring have potential to improve results, although we did not find an effective combination for the sequences tested. It might be possible to extend this method, so that instead of doing a hard assignment between two clusters, a distance metric is used to determine how far a pixel is from the rain cluster.

5.2.3 Patch-Based Detection

Instead of looking only at individual pixels, Garg and Nayar (2004) suggest that detection should involve examining small patches over a long sequence of frames. The algorithm has three steps. First, all pixels that flicker from dark to light then back to dark are labeled as "candidate" pixels. Second, these candidate images are thresholded and segmented with connected components. Components that are not linearly related within a threshold are eliminated, resulting in a binary image where each pixel is either 0 for non-rain or 1 for rain. The correlation of individual pixels within small patches is computed to find the magnitude and angle of the rainfall.

We found that for the second step, it is difficult to set a threshold that allows individual streaks to be segmented. Therefore, we skip the second step, and set all candidate pixels to 1 to create the binary images. Figure 10 shows examples of the magnitude of the correlation.

In the original paper, a single threshold was set to differentiate rain from non-rain. Rather than trying to find the optimal threshold by hand, an ROC curve is computed for the magnitude of the correlation. If the magnitude is above a given threshold, the pixel is replaced by the three-frame temporal median. As more and more pixels are classified as

rain, the rain-removed images become increasingly closer to the median images.

There is a clear trend on the removal accuracy curve, but the point tracking accuracy is hard to quantify for this algorithm. As increasingly lower thresholds are chosen, this method converges to the median filter result, which usually has the best tracking accuracy.

As with other removal methods, converting from hard to soft constraints would likely improve accuracy. In addition, no image alignment is advocated in this method, but aligning via one of the methods discussed earlier could also improve the results.

5.2.4 *Frequency-Based Detection*

Two versions of removal for the spatio-temporal frequency method presented in this work are compared. For the method of (19), we use a fixed removal rate $\alpha = 3$ with four iterations, and we linearly interpolate to predict the pixel values at different levels of removal. As the level of removal is increased, all pixels are forced to decrease monotonically. For the method of (18) the rain/snow detection is only computed once, and the value of α is changed to generate the ROC curves.

Both of these frequency-based methods are usually accurate in terms of amount of rain removed versus background corrupted, but do not increase the number of tracked points more than the aligned median, shown in Fig. 13. This is because this method is able to reduce the brightness of most streaks with few errors, but it rarely completely eliminates all streaks. This means that features still become occluded by flickering streaks.

6 Conclusion

We have demonstrated a method for globally detecting rain and snow, by using a physical and statistical model to highlight their spatio-temporal frequencies. Previous works have shown that examining only pixels or patches can be used to enhance videos in some cases, but the best results come from treating rain and snow as global phenomena.

Even a human observer can have difficulty in finding individual streaks in an image, although groups are easy to see. By generating a rain and snow model based on the expected properties of groups of streaks, we are able to achieve accuracy beyond what can be expected from local image analysis. On several challenging sequences, we show that rain and snow can be reduced or enhanced by studying their global properties.

The advantage of working in the frequency domain is that it allows for fast analysis of repeated patterns. The disadvantage is that changes made in frequency space do not always cause visual pleasing effects in image space. Although the segmentation accuracy surpasses any image space method developed so far, the removal results do not always have the best appearance.

Local image-space detection has certain types of errors, such as blurring across temporal edges. Frequency-based detection has errors when the frequencies corresponding to rain and snow are too cluttered. One possible direction for future work will be to combine global cues, as developed in this work, with local information. The synthesis of the global and local approaches will allow for accurate detection even in cases where either method fails alone.

Acknowledgements This work is supported in part by the National Science Foundation under Grant No. EEEC-0540865, NSF CAREER Award No. IIS-0643628, NSF Award No. CCF-0541307, ONR Award No. N00014-05-1-0188, and Denso Corporation. Parts of this research have appeared in Barnum et al. (2007).

Appendix A: Accuracy of the Rain/Snow Model

In this appendix, we derive an expression highlighting the simple structure of (9). Rain and snow are fundamentally random, and there is no guarantee if the magnitude of (9) will have a simple closed form. However, we can analyze it in the case where each image has the same number of each size of streak, given that each streak is equally likely to appear in any location.

To begin, we show the closed form solution of the Fourier transform of a blurred Gaussian. We show only the case where the streak is completely vertical, and include a normalizing constant for ease of reading. The notation in this section is similar to the rest of the paper, but some symbols are redefined.

A.1 The Fourier Transform of a Rain Streak

To begin, a Gaussian, with width b, blurred over length l is given by:

$$g(x, y; b, l) = \int_{c=0}^{l} \exp\left(-\pi \frac{(x^2 + (y-c)^2)}{b^2}\right) dc$$

The Fourier transform is given by:

$$G(u, v; b, l) = \mathcal{F}\left\{\int_{c=0}^{l} \exp\left(-\pi \frac{(x^2 + (y-c)^2)}{b^2}\right) dc\right\}$$

The length integral can be moved outside of the transform:

$$\int_{c=0}^{l} \mathcal{F}\left\{\exp\left(-\pi \frac{(x^2 + (y-c)^2)}{b^2}\right)\right\} dc$$

Since a shift in space is a multiplication in frequency, the equation becomes:

$$\int_{c=0}^{l} \mathcal{F}\left\{\exp\left(-\pi \frac{(x^2+y^2)}{b^2}\right)\right\} \exp(2\pi i v c) dc$$
$$= \int_{c=0}^{l} b^2 \exp(-\pi b^2(u^2+v^2)) \exp(2\pi i v c) dc$$

Only the rightmost exponential depends on n, so we can take out the left part and solve the integral:

$$b^2 \exp(-\pi b^2(u^2+v^2)) \int_{c=0}^{l} \exp(2\pi i v c) dc$$
$$= G(u, v; b, l)$$
$$= b^2 \exp(-\pi b^2(u^2+v^2)) i \frac{1-\exp(2\pi i v l)}{2\pi v}$$

The magnitude of G is simple itself multiplied by its complex conjugate:

$$\|G(u, v; b, l)\|$$
$$= \left(b^2 \exp(-\pi b^2(u^2+v^2)) i \frac{1-\exp(2\pi i v l)}{2\pi v}\right)$$
$$\times \left(-b^2 \exp(-\pi b^2(u^2+v^2)) i \frac{1-\exp(-2\pi i v l)}{2\pi v}\right)$$
$$= \frac{b^4 \exp(-2\pi b^2(u^2+v^2))}{4\pi^2 v^2}$$
$$\times (1-\exp(2\pi i v l))(1-\exp(-2\pi i v l))$$
$$= \frac{b^4 \exp(-2\pi b^2(u^2+v^2))}{4\pi^2 v^2}(2-2\cos(2\pi v l))$$
$$= \frac{b^4 \exp(-2\pi b^2(u^2+v^2))}{4\pi^2 v^2}(4\sin^2(\pi v l))$$
$$= \|G(u, v; b, l)\|$$
$$= \frac{b^4 \sin^2(\pi v l) \exp(-2\pi b^2(u^2+v^2))}{\pi^2 v^2} \quad (A.1)$$

A.2 The Fourier Transform of Multiple Identical Streaks

The next step is to determine the magnitude of multiple streaks. To begin, we assume that all streaks are identical, but in different locations, $\boldsymbol{\mu} = [\mu_x, \mu_y]$. The image of all streak is then:

$$\sum_n g(x, y; b, l, \boldsymbol{\mu}_n)$$

The Fourier transform is:

$$\mathcal{F}\left\{\sum_n g(x, y; b, l, \boldsymbol{\mu}_n)\right\}$$
$$= \sum_n \mathcal{F}\{g(x, y; b, l, \boldsymbol{\mu}_n)\}$$
$$= \sum_n G(u, v; b, l) \exp(2\pi i(u\mu_{xn} + v\mu_{yn}))$$
$$= G(u, v; b, l) \sum_n \exp(2\pi i(u\mu_{xn} + v\mu_{yn})) \quad (A.2)$$

The magnitude is given by:

$$\left(G(u, v; b, l) \sum_n \exp(2\pi i(u\mu_{xn} + v\mu_{yn}))\right)$$
$$\times \left(G^*(u, v; b, l) \sum_n \exp(-2\pi i(u\mu_{xn} + v\mu_{yn}))\right)$$
$$= G(u, v; b, l) G^*(u, v; b, l)$$
$$\times \left(\sum_n \exp(2\pi i(u\mu_{xn} + v\mu_{yn}))\right)$$
$$\times \left(\sum_n \exp(-2\pi i(u\mu_{xn} + v\mu_{yn}))\right) \quad (A.3)$$

The two sums will multiply into pairs of exponentials, which can be converted into cosines:

$$G(u, v; b, l) G^*(u, v; b, l)$$
$$\times \left(N + \sum_{a=1}^{N-1} \sum_{b=a+1}^{N} \cos(2\pi(u(\mu_{xa} - \mu_{xb}) + v(\mu_{ya} - \mu_{yb})))\right) \quad (A.4)$$

The first part with G multiplied by its conjugate is the magnitude from (A.1):

$$\frac{b^4 \sin^2(\pi v l) \exp(-2\pi b^2(u^2+v^2))}{\pi^2 v^2}$$
$$\times \left(N + \sum_{a=1}^{N-1} \sum_{b=a+1}^{N} \cos(2\pi(u(\mu_{xa} - \mu_{xb}) + v(\mu_{ya} - \mu_{yb})))\right) \quad (A.5)$$

The resulting equation is the magnitude of the blurred Gaussian, multiplied by the addition of N and a sum of cosines term.

A.3 The Fourier Transform of Multiple Streaks of Different Sizes

In the general case of streaks of many different sizes, the resulting expression does not reduce as cleanly as in (A.5). But it does in the special case where at a given location, there are the same number of each size. Starting from (A.2), but with multiple bs and ls:

$$\sum_n \left(\sum_l \sum_b G(u, v; b, l)\right) \exp(2\pi i(u\mu_{xn} + v\mu_{yn}))$$

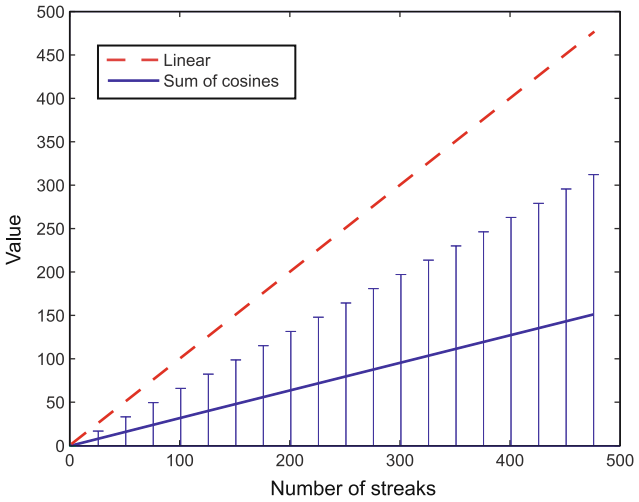

Fig. 14 To determine the value of the sum of cosines in (A.5) compared to the number of streaks N, we ran simulations with different numbers of streaks. The *dotted line* is the value of the linear component. The *solid line* is the mean across trials of the median of the absolute value of the sum of cosines

Since it does not depend on n, the double sum of blurred Gaussians can be pulled out in the same way, yielding:

$$\left(\sum_l \sum_b G(u, v; b, l)\right) \sum_n \exp(2\pi i (u\mu_{xn} + v\mu_{yn})) \quad \text{(A.6)}$$

And the remaining steps are the same as (A.3) to (A.5).

A.4 Approximating the Fourier Transform of Multiple Streaks

The value of the sum of cosines in (A.5) varies depending on the frequency (u, v) and the distribution of streak locations μ. It has a minimum of zero and a very large maximum at $(u, v) = (0, 0)$, but streaks at different locations tend to cancel each other out, so it is generally low. We ran simulations to find the kind of values that the sum of cosines tends to have.

In our experiments, we computed the value by sampling different numbers of streaks with uniformly distribution locations. Figure 14 shows the value for the median of all frequencies, for one to five hundred streaks, with one thousand trials each. For each number of streaks N from one to five hundred, we ran one thousand trials of randomly sampled streak locations. The dotted line is the value of N from the first term in the addition in (A.5). The solid line is for the sum of cosines term. For all frequencies u and v, we compute the absolute value of the sum of cosines. For each trial, we then compute the median of the value for all frequencies except $(u, v) = (0, 0)$. This median is computed for each of the one thousand trials, and the solid line is the mean across trials of the medians. The error bars represent the mean across trials of the standard deviation for all frequencies. The results show that although the value for some frequencies is large, the median value is low enough that $N + \sum_{a=1}^{N-1} \sum_{b=a+1}^{N} \cos(2\pi(u(\mu_{xa} - \mu_{xb}) + v(\mu_{ya} - \mu_{yb})))$ can be approximated as being linear in N, which validates our use of (10).

References

Auer, A. H. Jr. (1972). Distribution of graupel and hail with size. *Monthly Weather Review*, *100*(5), 325–328.

Auer, A. H. Jr., & Veal, D. L. (1970). The dimension of ice crystals in natural clouds. *Journal of the Atmospheric Sciences*, *27*(6), 919–926.

Barnum, P., Kanade, T., & Narasimhan, S. (2007). Spatio-temporal frequency analysis for removing rain and snow from videos. In *Workshop on photometric analysis for computer vision, in conjunction with international conference on computer vision*.

Böhm, H. P. (1989). A general equation for the terminal fall speed of solid hydrometeors. *Journal of the Atmospheric Sciences*, *46*, 2419–27.

Bouguet, J.-Y. (2000). *Pyramidal implementation of the Lucas Kanade feature tracker.* Intel Corporation, Microprocessor Research Labs.

Cozman, F., & Krotkov, E. (1997). Depth from scattering. In *International conference on computer vision*.

de la Torre, F., & Black, M. L. (2001). Robust principal component analysis for computer vision. In *International conference on computer vision*.

Desaulniers-Soucy, N. (1999). *Empirical test of the multifractal continuum limit in rain*. PhD thesis, Mcgill.

Desaulniers-Soucy, N., Lovejoy, S., & Schertzer, D. (2001). The HYDROP experiment: an empirical method for the determination of the continuum limit in rain. *Atmospheric Research*, *59–60*, 163–197.

Elgammal, A., Harwood, D., & Davis, L. (2000). Non-parametric model for background subtraction. In *European conference on computer vision*.

Feingold, G., & Levin, Z. (1986). The lognormal fit to raindrop spectra from frontal convective clouds in Israel. *Journal of Climate and Applied Meteorology*, *25*, 1346–63.

Fishler, M., & Bolles, R. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. In *Communications of the ACM*.

Foote, G. B., & duToit, P. S. (1969). Terminal velocity of raindrops aloft. *Journal of Applied Meteorology*, *8*(2), 249–53.

Garg, K., & Nayar, S. K. (2004). Detection and removal of rain from videos. In *Computer vision and pattern recognition*.

Garg, K., & Nayar, S. K. (2005). When does a camera see rain? In *International conference on computer vision*.

Garg, K., & Nayar, S. K. (2006). Photorealistic rendering of rain streaks. In *SIGGRAPH*.

Gonzalez, R. C., & Woods, R. E. (2002). *Digital image processing* (2nd ed.). New York: Prentice Hall.

Gunn, K., & Marshall, J. (1958). The distribution with size of aggregate snowflakes. *Journal of Meteorology*, *15*, 452–461.

Hase, H., Miyake, K., & Yoneda, M. (1999). *Real-time snowfall noise elimination*.

Heeger, D. (1987). Optical flow from spatiotemporal filters. In *International conference on computer vision*.

Jameson, A. R., & Kostinski, A. B. (2001). What is a raindrop size distribution? *Bulletin of the American Meteorological Society*, *8*(6), 1169–1177.

Jameson, A., & Kostinski, A. (2002). When is rain steady? *Journal of Applied Meteorology*, *41*(1), 83–90.

Ke, Q., & Kanade, T. (2002). A robust subspace approach to layer extraction. In *IEEE workshop on motion and video computing*.

Kubesh, R. J., & Beard, K. (1993). Laboratory measurements of spontaneous oscillations of moderate-size raindrops. *Journal of the Atmospheric Sciences*, *50*, 1089–1098.

Langer, M., & Mann, R. (2003). Optical snow. *International Journal of Computer Vision*, *55*(1), 55–71.

Langer, M., & Zhang, Q. (2003). Rendering falling snow using and inverse Fourier transform. In *ACM SIGGRAPH technical sketches program*.

Langer, M. S., Zhang, L., Klein, A., Bhatia, A., Pereira, J., & Rekhi, D. (2004). A spectral-particle hybrid method for rendering falling snow. In *Eurographics symposium on rendering*.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, *20*, 91–110.

Magono, C., & Nakamura, T. (1965). Aerodynamic studies of falling snowflakes. *Journal of the Meteorological Society of Japan*, *43*, 139–147.

Marshall, J., & Palmer, W. (1948). The distribution of raindrops with size. *Journal of Meteorology*, *5*, 165–166.

Mikolajczyk, K., & Schmid, C. (2001). Indexing based on scale invariant interest points. In *International conference on computer vision*.

Narasimhan, S. G., & Nayar, S. K. (2002). Vision and the atmosphere. *International Journal of Computer Vision*, *48*(3), 233–254.

Nayar, S. K., & Narasimhan, S. G. (1999). Vision in bad weather. In *International conference on computer vision*.

Ohtake, T. (1965). Preliminary observations on size distribution of snowflakes and raindrops at just above and below the melting layer. In *International conference on cloud physics*.

Pruppacher, H. R., & Klett, J. D. (1997). *Microphysics of clouds and precipitation*. Amsterdam: Kluwer Academic. Second revised and enlarged edition.

Reeves, W. T. (1983). Particle systems—a technique for modeling a class of fuzzy objects. *ACM Transactions on Graphics*, *2*(2), 91–108.

Sand, P., & Teller, S. (2006). Particle video: long-range motion estimation using point trajectories. In *Computer vision and pattern recognition*.

Shi, J., & Tomasi, C. (1994). Good features to track. In *Computer vision and pattern recognition*.

Starik, S., & Werman, M. (2003). Simulation of rain in videos. In *International workshop on texture analysis and synthesis*.

Stauffer, C., & Grimson, W. (1998). Adaptive background mixture models for real-time tracking. In *Computer vision and pattern recognition*.

Tariq, S. (2007). *Rain* (Technical report). NVIDIA.

Tatarchuk, N., & Isidoro, J. (2006). Artist-directable real-time rain rendering in city environments. In *Eurographics workshop on natural phenomena*.

Tokay, A., & Beard, K. (1996). A field study of raindrop oscillations. Part I: Observation of size spectra and evaluation of oscillation causes. *Journal of Applied Meteorology*, *35*, 1671–1687.

Torr, P. H. S., Szeliski, R., & Anandan, P. (1999). An integrated Bayesian approach to layer extraction from image sequences. In *International conference on computer vision*.

Ulbrich, C. W. (1983). Natural variations in the analytical form of the raindrop size distribution. *Journal of Applied Meteorology*, *22*(10), 1764–1775.

Van de Hulst, H. (1957). *Light scattering by small particles*. New York: Wiley.

Zelnik-Manor, L., Machline, M., & Irani, M. (2006). Multi-body factorization with uncertainty: Revisiting motion consistency. *International Journal of Computer Vision*, *68*(1), 27–41.

Zhang, X., Li, H., Qi, Y., Kheng, W., & Ng, T. K. (2006). Rain removal in video by combining temporal and chromatic properties. In *International conference on multimedia and expo*.