Image Blurring and Image Motion

Justin Domke and Yiannis Aloimonos

***Abstract*—Since cameras blur the incoming light before sampling, different images of the same surface contain information about that surface. We first consider how to synthesize one view of a surface from another; If the transformation between the two views is affine, we show that this is possible if and only if the singular values of the affine matrix are positive definite at all points. Next, we consider how to combine the information in several views of a surface into a single output image. By developing a new tool called "frequency segmentation" we show how this can be done despite not knowing the blurring kernel.**

***Index Terms*—Reconstruction, Restoration, Sharpening and deblurring, Smoothing**

I. Introduction

This paper concerns a very basic question: What are the relationships between multiple views of a surface? This question is only partially answered by the geometry of the situation. Consider two points in different images that project to the same 3-D surface point. Even supposing the images are perfectly Lambertian, and disregarding issues like like lighting or noise, the image intensities will still in general be different. This is because cameras do not merely *measure* the incoming light. Before measurement, the signal is optically filtered or *blurred*. Given the finite resolution of cameras, this is necessary to avoid aliasing effects. However, because this blurring is fixed to the camera's coordinate system, different views of a surface result in different measured signals. This is true even if the geometrical transformation between the views is corrected for.

To understand this situation, this paper introduces a very simple formalism. We contrast between what we call the "ideal image", corresponding to the unblurred incoming light, and the "real image", corresponding to the measured signal. The ideal image can never actually be measured, but it allows us to separate the geometric and filtering effects in our analysis. If the ideal image is some function i and the real image is some function j, we simply write $j = i * e$, where e is the blurring kernel. (Here, i, j, and e are all continuous functions. Assuming that the signal is low-pass filtered appropriately, j can be reconstructed from the discretely sampled pixels.) The advantage of this is that given the transformation between two views, one view's ideal image exactly specifies the other's. So we can ignore the 3-D structure of the scene, and consider only the transformation between the views.

Given this setup, it is easy to derive a number of theorems about the relationships between different views. After a few basic results, we will ask the following question: Suppose we have the real image j for one view, the blurring kernel e, and a transformation to a second view. When will it be possible to construct the (real) image for the second view? We will see this this is only sometimes possible. We will characterize the class of transformations for which it is possible, and how to construct this second image in each case.

Finally, as an example of the practical use of this framework, we consider the problem of multiple view image reconstruction. Suppose that we have several views, and the transformations between them. Each input image here will have been subjected to different filtering. Now, we want to construct an output image containing the best content from all input images. Here, we consider this problem in the context that the blurring kernel e is *unknown*. (Though we will make some weak assumptions about its form.) We will see that despite this, it is still possible to reconstruct an output image that is visibly less blurred than any input image. Intuitively, this is done by taking each frequency from the image in which it is least filtered. To do this we develop a tool we call "frequency segmentation". This divides up the space of frequencies into a few regions. In each region, one input image has been least filtered. We first apply this to the case of affine transformations, and then more general transformations through a strategy of local linearization.

A. Previous Work

As regards our application of multiple view image reconstruction, we are aware of only of the paper by Wang et. al [10]. Their method uses the assumption the the image blurring is an ideal low-pass filter, so it is not applicable here, where the filtering is unknown.

There is also work less closely related to this particular application that touches on similar issues. In 1990, Stone was interested in the problem of shape from texture [8]. He pointed out that in general, a circular filter on the image will project to a non-circular region on the surface in view. As such, he proposed an iterative scheme for computing both the surface shape, and a set of elliptical filters, one for each image point. These filters would each project to an identical area on the object surface. Each iteration consisted of an update to the shape, and then the filters, each using the current best estimates. In later work [9], however, Stone and Isard suggested only updating the size of circularly symmetric filters. They argued that, in practice, filters projecting to different shapes on the object surface will yield similar measurements as long as the areas of projection are the same.

From a more theoretical perspective, Lindeberg [3] considered how to extend linear (symmetric) scale-space, arriving at the idea of Affine Gaussian scale-space. The principal observation is that Affine Gaussian scale-space is closed under affine warpings, unlike the space where the image is filtered only with a symmetric Gaussian. These ideas were later used by Lindeberg and Gårding [3] for 3-D shape estimation from texture.

Several authors have used Affine Gaussian scale space for different applications. Ravela [7] uses an optimization procedure to select local affine parameters at all points. This is shown to improve performance on recognition problems.

Another area of work that has touched similar issues is affine invariant region detection [5]. Specifically, the Harris-Affine & Hessian Affine detectors [6] consider the affine Gaussian scale space of an image, and search for extrema, both with respect to position and the covariances of the Gaussian.

II. Setup

Though all our proofs are fairly simple, for clarity, we postpone them whenever they are tedious, or distract from the main ideas of the paper.

A. Preliminaries

Given some function f, and a matrix A, we will use the notation f^A to represent f, warped under the motion A. That is, f^A is the function such that,

$$\forall \mathbf{x}, f^A(\mathbf{x}) = f(A\mathbf{x}). \tag{1}$$

We will use lowercase letters (i, or j) to represent functions in the spatial domain, and uppercase letters (I, or J) for the frequency domain. Boldface letters ($\mathbf{x}$) represent vectors.

Finally, we will use of the following theorem, which is a special case of the affine Fourier theorem [1].

Theorem 1 (Translation-Free Affine Fourier Theorem):

$$\text{If } \mathcal{F}\{f(\mathbf{x})\} = F(\mathbf{u}) \text{ then } \mathcal{F}\{f^A(\mathbf{x})\} = |A^{-1}|F^{A^{-T}}(\mathbf{u})$$

Proof: (Postponed) ■

B. Basic Results

Assume that we have two images of the same surface, taken under affine motion. With out loss of generality, assume that the transformation sends the origin to the origin. Then if i_1 and i_2 are the ideal images, there exists some A such that

$$\mathbf{x}_2 = A\mathbf{x}_1 \rightarrow i_2(\mathbf{x}_2) = i_1(\mathbf{x}_1) \tag{2}$$

Or equivalently,

$$i_2^A(\mathbf{x}) = i_1(\mathbf{x}) \tag{3}$$

Let j_1 and j_2 be the real, observed images. Let e be the low-pass filter applied by the optics of the camera.

$$j_1(\mathbf{x}) = [i_1 * e](\mathbf{x}) \tag{4}$$

$$j_2(\mathbf{x}) = [i_2 * e](\mathbf{x}) \tag{5}$$

The results in this paper are independent of the particular form of e. However, we make two assumptions:

1) e is circularly symmetric. Formally, $e(\mathbf{x})$ is a function of $|\mathbf{x}|$ only. Notice that if $E(\mathbf{u})$ is the Fourier Transform of $e(\mathbf{x})$, this also implies that E is circularly symmetric.
2) e is a monotonically decreasing low-pass filter. Formally, if $|\mathbf{u}_2| \geq |\mathbf{u}_1|$, then $E(\mathbf{u}_2) \leq E(\mathbf{u}_1)$.

Notice that unlike the ideal images, the real images need not match at corresponding points.

$$\mathbf{x}_2 = A\mathbf{x}_1 \not\rightarrow j_2(\mathbf{x}_2) = j_1(\mathbf{x}_1) \tag{6}$$

The key result driving this paper is the following simple Theorem. Contrast this with Eqn. 3 for the ideal images.

Theorem 2 (Affine Blurring Theorem):

$$\boxed{j_2^A(\mathbf{x}) = |A|[i_1 * e^A](\mathbf{x})} \tag{7}$$

Proof: (Postponed) ■

This shows that when warped into the coordinate system of the first image, the second image is equivalent to the first ideal image, convolved with a warped version of the low-pass filter. $|A|$ acts as a normalization factor so that e^A integrates to the same value as e.

Another useful result is given by taking the Fourier Transform of both sides of Eqn. 7.

Theorem 3 (Fourier Affine Blurring Theorem):

$$\boxed{\mathcal{F}\{j_2^A(\mathbf{x})\} = [I_1 \cdot E^{A^{-T}}](\mathbf{u})} \tag{8}$$

Proof: (Postponed) ■

C. Affine Gaussian scale-space

If the blurring kernel is a Gaussian, we can use Eqn. 7 to get the relationship between j_1 and j_2^A more explicitly. Let g_Σ denote an origin-centered Gaussian with covariance matrix Σ. Then, $e = g_{\sigma I}$.

$$j_1(\mathbf{x}) = [i_1 * g_{\sigma I}](\mathbf{x}) \tag{9}$$

Then, we can understand how the Gaussian behaves under warping through the following theorem.

Theorem 4:

$$g_\Sigma^A(\mathbf{x}) = \frac{1}{|A|} g_{A^{-1}\Sigma A^{-T}}(\mathbf{x}) \tag{10}$$

Proof: (Postponed) ■

From this, the following relationship follows immediately.

$$j_2^A(\mathbf{x}) = [i_1 * g_{\sigma A^{-1}A^{-T}}](\mathbf{x}) \tag{11}$$

Notice that the constant of $|A|$ was absorbed into the Gaussian. Note that essentially this same relationship is known in the literature on Affine Gaussian scale-space. [4]

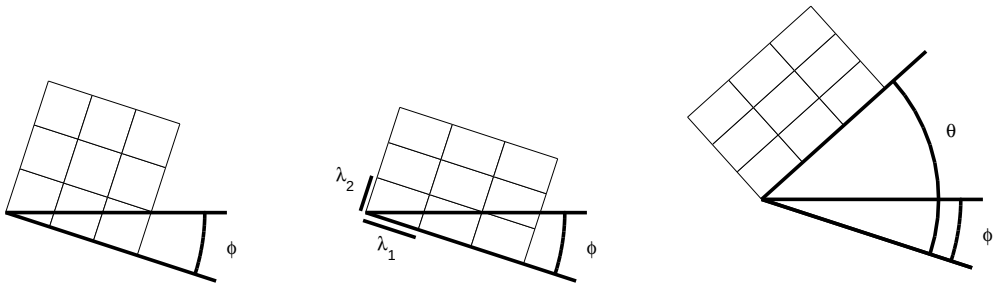
D. Parametrization of A

Though A has four parameters, it turns out that there are only three parameters of interest to us here. Consider the following decomposition of A, which follows from the Singular Value Decomposition [2].

$$A = R(\theta)R(-\phi)DR(\phi) \tag{12}$$

$$D = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \tag{13}$$

Figure 1. Visualization of the four parameters of A. We can see that the effect of $R(-\phi)DR(\phi)$ is to stretch by factors of λ_1, and λ_2 along axes an angle of ϕ away from the starting axes.



(a) Initial Points. (b) After $R(-\phi)DR(\phi)$. (c) After rotation by θ.

This decomposition is visualized in Fig. 1. It will be convenient later to make reference to the parameters θ, ϕ, λ_1, and λ_2 with out explicitly specifying that they correspond to some matrix A.

The key result here is that the parameter θ has no role in the blurring behavior of the images. Intuitively, θ is just a rotation, which has no effect on the filtering since the low-pass filter has no preferred direction. Warping j_2 into j_2^A removes any effects due to θ. This is formalized in the following theorem.

Theorem 5:

$$j_2^A \text{ is independent of } \theta.$$

Proof: Notice that the expression for j_2 in Eqn. 7 depends on A only through $|A|$, and e^A. By our assumption that e is circularly symmetric, $e^A(\mathbf{x}) = e(A\mathbf{x}) = e(|A\mathbf{x}|)$ does not depend on θ. It is also easy to see that $|A|$ is independent of θ. ∎

III. Two-View Accessibility

Suppose we are given only A and j_2. A question arises: is it possible to filter j_2^A so as to synthesize j_1? As we will show below, this is only sometimes possible.

We must first specify what operations are allowed. In this paper, we want to avoid the problem of deconvolution, or deblurring. In theory, if the low-pass filter E were nonzero for all frequencies, it would be possible to increase the magnitude of all frequencies to completely cancel out the effects of the filtering. With the ideal image in hand, any other view could be synthesized. However, in practice, this can be done only for a limited range of frequencies, because for high frequencies the noise in the observed image will usually be much larger than the signal remaining after filtering. [cite deconvolution review] If deconvolution is practical, we can imagine that it has already been applied, and the problem restated with an appropriately changed blurring kernel and input images. Hence, we will limit ourselves to strategies that do not attempt to invert the blurring kernel.

Suppose that we apply some filter d to one of the images. If D is the Fourier Transform of d, we require that D does not increase the magnitude of any frequency. Formally, for all $\mathbf{u}$, we must have $D(\mathbf{u}) \leq 1$.

We will say that a view "is accessible" from another view taken under motion A if there exists a filter d such that $e = |A|e^A * d$. Notice that if this is the case, then

$$\begin{aligned}
[j_2^A * d](\mathbf{x}) &= |A|[(i_1 * e^A) * d](\mathbf{x}) && (14)\\
&= |A|[i_1 * (e^A * d)](\mathbf{x}) && (15)\\
&= [i_1 * e](\mathbf{x}) && (16)\\
&= j_1(\mathbf{x}). && (17)
\end{aligned}$$

The following theorem formalizes exactly when one view is accessible from another. It is worth sketching the proof in some detail.

Theorem 6 (Two-View Accessibility Theorem): j_1 is accessible from j_2 if and only if $\lambda_1 > 1$ and $\lambda_2 > 1$.

Proof: We will develop several conditions, each of which is equivalent to the assertion "j_1 is accessible from j_2". By definition, this means there is a valid d such that

$$\forall \mathbf{x}, [|A|e^A * d](\mathbf{x}) = e(\mathbf{x}). \quad (18)$$

Apply the Translation-Free Affine Fourier Theorem to e^A.

$$\mathcal{F}\{e^A(\mathbf{x})\} = |A^{-1}|E^{A^{-T}}(\mathbf{u}) \quad (19)$$

Now, if we also take the Fourier Transform of d and e, the condition is, by the convolution theorem,

$$\forall \mathbf{u}, E(A^{-T}\mathbf{u}) \cdot D(\mathbf{u}) = E(\mathbf{u}). \quad (20)$$

Since we need that $D(\mathbf{u}) \leq 1$ for all $\mathbf{u}$, we require that $E(A^{-T}\mathbf{u}) \geq E(\mathbf{u})$. This will be the case when

$$\forall \mathbf{u}, |A^{-T}\mathbf{u}| \leq |\mathbf{u}|. \quad (21)$$

It is not hard to show that this is equivalent to the singular values of A (λ_1 and λ_2) being greater than zero. ∎

So when j_1 is in fact accessible from j_2, constructing the filter is trivial. Simply set $D(\mathbf{u}) = E(\mathbf{u})/E(A^{-T}\mathbf{u})$ for all $\mathbf{u}$, and obtain d by the inverse Fourier Transform. The exact form of d will of course depend on the form of e.

For example, suppose e is a Gaussian filter. Let g_Σ denote a Gaussian function with covariance matrix Σ. We need that $g_{\sigma I} = g_{\sigma A^{-1}A^{-T}} * d$. (Recall Eqn. 11.) Since $g_{\Sigma_1} * g_{\Sigma_2} =$

$g_{\Sigma_1+\Sigma_2}$, we can see that $d = g_{\sigma(I-A^{-1}A^{-T})}$. For this to be a valid Gaussian, the covariance matrix $I - A^{-1}A^{-T}$ must be positive definite. This is true if and only if the singular values of A are greater than zero, confirming the above theorem.

For another example, suppose e is an ideal low pass filter. That is, suppose $E(\mathbf{u}) = 1$ when $|\mathbf{u}| < r$ for some threshold r, and zero otherwise. In this case, we can use $d = e$. The effect is to filter out those frequencies in j_2^A that are not in j_1. However, if the singular values of A are not both greater than one, there will be some frequencies in j_1 that are not in j_2^A, and again j_1 would be inaccessible.

A. Accessibility and Matching

The accessibility theorem has some possible applications to image matching that we briefly discuss here. This section can be skipped with out loss of continuity.

Suppose that we have both j_1 and j_2, and we hypothesize that they vary by some motion A. How can we check if this is true? Of course, we can not just compare pixel intensities, because j_1 and j_2^A will usually be different, despite corresponding to the same surface.

A better idea is to do filtering as in the previous section to create a "possible copy" of j_1 from j_2^A, under the assumption that the image regions do correspond. However, this may or may not be possible, depending on A. There are three situations.

1) $\lambda_1 > 1$, $\lambda_2 > 1$. Here we could do the filtering exactly as described above.
2) $\lambda_1 < 1$, $\lambda_2 < 1$. Now, j_1 is not accessible from j_2. However, by symmetry, we can instead try to synthesize j_2 from j_1, under a hypothesized motion of A^{-1}. (The singular values of A^{-1} will both be greater than one.)
3) $\lambda_1 > 1$, $\lambda_2 < 1$, or $\lambda_1 < 1$, $\lambda_2 > 1$. In this situation, neither image is accessible from the other. Intuitively, this means that both images contain some frequencies that are reduced (or eliminated) in the other image.

The first two situations can be dealt with by the methods above, but the third needs more discussion. The problem is how to filter the images to remove any extra information in one image that is not present in the other. At the same time, we would like to remove as little information as possible. The goal is to create filters d_1 and d_2 such that

$$\forall \mathbf{x}, [j_1 * d_1](\mathbf{x}) = [j_2^A * d_2](\mathbf{x}). \tag{22}$$

Consider this in the frequency domain. For the left hand side, we have

$$\mathcal{F}\{j_1 * d_1\} = J_1(\mathbf{u}) \cdot D_1(\mathbf{u}) = I_1(\mathbf{u}) \cdot E(\mathbf{u}) \cdot D_1(\mathbf{u}). \tag{23}$$

For the right hand side, recall the Fourier Transform of j_2^A from Eqn. 8. Then

$$\mathcal{F}\{j_2^A * d_2\} = I_1(\mathbf{u}) \cdot E^{A^{-T}}(\mathbf{u}) \cdot D_2(\mathbf{u}). \tag{24}$$

So we will need to construct the filters such that

$$\forall \mathbf{u}, E(\mathbf{u}) \cdot D_1(\mathbf{u}) = E^{A^{-T}}(\mathbf{u}) \cdot D_2(\mathbf{u}). \tag{25}$$

The optimal solution is to set the filters so as to minimize the amount of information removed.

$$D_1(\mathbf{u}) = \begin{cases} \frac{E^{A^{-T}}(\mathbf{u})}{E(\mathbf{u})} & \text{if } E(\mathbf{u}) \geq E^{A^{-T}}(\mathbf{u}) \\ 1 & \text{if } E(\mathbf{u}) \leq E^{A^{-T}}(\mathbf{u}) \end{cases} \tag{26}$$

$$D_2(\mathbf{u}) = \begin{cases} 1 & \text{if } E(\mathbf{u}) \geq E^{A^{-T}}(\mathbf{u}) \\ \frac{E(\mathbf{u})}{E^{A^{-T}}(\mathbf{u})} & \text{if } E(\mathbf{u}) \leq E^{A^{-T}}(\mathbf{u}) \end{cases} \tag{27}$$

The problem remains how to divide up the space of $\mathbf{u}$ into those where E or $E^{A^{-T}}$ is greater. This problem will be addressed in a more general context in Section IV.

B. Example

In this example, we use synthetically generated images consisting of white noise, because the broad frequency content makes filtering easy to see. We use a symmetric Gaussian for the low pass filter. Hence, $e = g_{\sigma I}$.

In Fig. 2, the first image is observed with a motion

$$A = \begin{bmatrix} 1.1 & 0 \\ 0 & 2.2 \end{bmatrix}$$

relative to the first image. (Notice the large amount of vertical stretching and the mild horizontal stretching.) Since here, both λ_1 and λ_2 are greater than one, the first image is in fact accessible from the second. Hence, if we filter the second image appropriately, we obtain the first, as in the example.

IV. Frequency Segmentation

Before we can move on to multiple view image reconstruction, we must develop a tool we call "frequency segmentation". Suppose we have a set of images warped to a common coordinate system, $\{j_i^{A_i}(\mathbf{x})\}$, as well as the set of motions that produced those images, $\{A_i\}$. The question we are interested in is: for a given frequency $\mathbf{u}$, in what image is it least filtered? In this section, we will develop a method to segment frequency space into labeled regions. In each region, the label gives the index of the image with the least filtering for that region's frequencies.

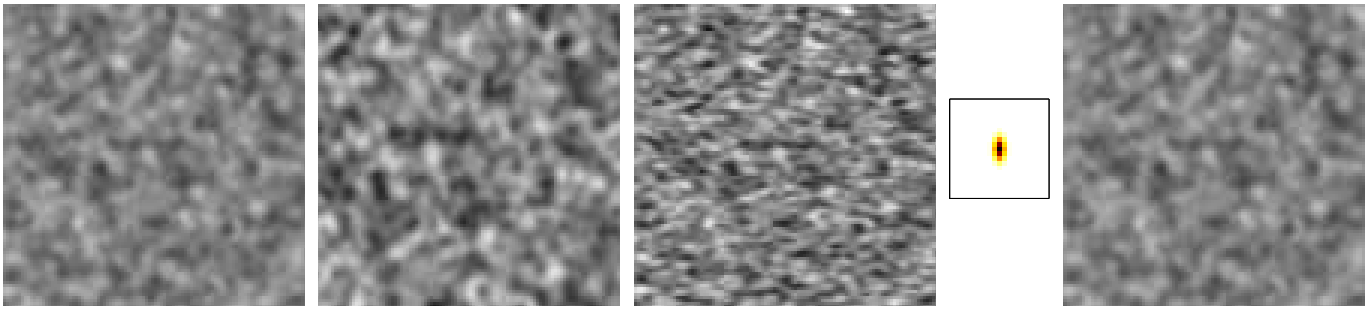
We can write the Fourier Transform of each of the images as the ideal image with a warped version of the low-pass filter. Recall Eqn. 8.

$$\mathcal{F}\{j_i^{A_i}(\mathbf{x})\} = [I \cdot E^{A_i^{-T}}](\mathbf{u})$$

The problem is to determine for each $\mathbf{u}$, for which A_i is $E(A_i^{-T}\mathbf{u})$ maximum, or equivalently, for which A_i is $|A_i^{-T}\mathbf{u}|$ minimum. Notice that this will depend only on the angle of $\mathbf{u}$, and not on its magnitude– if A_i is minimum for some $\mathbf{u}$, it will also be minimum for $a\mathbf{u}$, for any a.

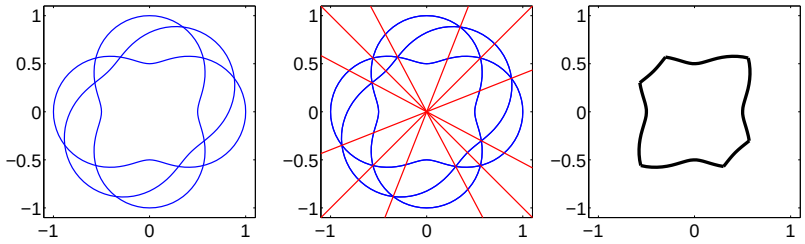
So, for each angle θ, we would like to choose the motion such that $E(A_i^{-T}[\cos\theta, \sin\theta]^T)$ is maximum. This is equivalent to $|A_i^{-T}[\cos\theta, \sin\theta]^T|$ being minimum. We can picture the situation by drawing a curve, where for each angle θ, we use the length $|A_i^{-T}[\cos\theta, \sin\theta]^T|$. (Fig 3(a)). To 'carve up' the space of frequencies, we need two steps.

Figure 2. A demonstration of 2-view accessibility, with an image of white noise, an image from the declaration of independence. a) j_1. b) j_2. c) j_2^A. Notice that there is visual content clearly visible in j_1, but not in j_2^A d) the filter d. e) the result of convolving j_2^A with d. Notice that this is almost identical to j_1.



(a) (b) (c) (d) (e)

Figure 3. Frequency Segmentation.



(a) (b) (c)

1) For each pair of motions A_i, and A_j, find points for which the curves meet. That is, find $\mathbf{u}$ such that

$$|A_i^{-T}\mathbf{u}| = |A_j^{-T}\mathbf{u}|. \tag{28}$$

This is equivalent to finding $\mathbf{u}$ such that

$$\mathbf{u}^T(A_i^{-1}A_i^{-T} - A_jA_j^{-T})\mathbf{u} = 0. \tag{29}$$

Notice that this does not depend on the magnitude of $\mathbf{u}$. If we assume that either the first or second component of $\mathbf{u}$ is nonzero, this can be solved by setting $\mathbf{u} = [u_x, 1]^T$ or $\mathbf{u} = [1, u_y]^T$, and solving a quadratic equation. Complex values as a solution indicate that the two curves do not intersect.

2) Find the angles of the points $\mathbf{u}$ found in the previous step and sort them. Now, form pairs from all adjacent angles. This results in a sequence of pairs of angles, $< \theta_1, \theta_2 >, < \theta_2, \theta_3 >, ... < \theta_{m-1}, \theta_m >$. It remains to find the motion that has the smallest value in each region. The simplest procedure would simply be to test all motions, and explicitly find

$$\arg\min_i |A_i^{-T} \begin{bmatrix} \cos(.5(\theta_j + \theta_{j+1})) \\ \sin(.5(\theta_j + \theta_{j+1})) \end{bmatrix}|. \tag{30}$$

This works, but has a worse-case time complexity of $O(n^3)$, where n is the number of input images. However, an optimization can reduce this to $O(n^2 \log n)$, the complexity of the sorting step: For each angle θ_j, keep track of the motions whose intersection produced that angle. Then, if some motion A_i is minimum in the region $< \theta_{j-1}, \theta_j >$, A_i will also be minimum in the region $< \theta_j, \theta_{j+1} >$, unless θ_j was produced by the intersection of A_i with some other motion A_k, and A_k is "smaller" than A_i in the sense of Eqn. 30. This means that we only need to test at most two motions in each region.

The entire frequency segmentation process is illustrated in Fig. 3. Part (a) shows the magnitude $|A_i^{-T}[\cos\theta, \sin\theta]^T|$ plotted for each angle θ. In part (b), the angles are found where the motions "intersect". In part (c), the shape is shown, where for each angle, the magnitude is taken from the best motion.

V. Reconstruction with an Unknown Kernel

At this point, one might proceed to show how to extend the results of Section III-A to the case of many views. This is certainly possible. One could define alternate conditions to those in the two-view accessibility theorem– rather than insisting that each frequency in the desired view is less filtered in the other, the condition would be that for each frequency in the desired view there is at least one image in which it is less filtered. Given this, one could use frequency segmentation to build appropriate filters and synthesize new views.

However, the above will only be possible if the blurring kernel, e, is known. Given the difficulty of measuring the blurring kernel in real cameras, this might be of mostly theoretical interest. In this paper, we will instead focus on what to do if the blurring kernel is unknown.

It might initially seem that it is not possible to do multiple view reconstruction with out knowing the blurring kernel. However, notice that our frequency segmentation method does not make use of the particular form of e– only the assumption that it is circularly symmetric and monotonically decreasing.

The problem of how to define appropriate filters remains. Clearly, it is impossible to synthesize a view with out knowing the blurring kernel that view is the result of. Hence, we must redefine the output of our algorithm. Rather than trying to create an image that is the result of some motion, we will simply combine the frequencies in the input images, *at their observed levels*, into one output image. To do this, it is not necessary to know the blurring kernel. After frequency segmentation is done, one needs to only design a filter that will take a certain range of frequencies from each image. This problem is addressed in the Section VI.

The intuition behind the method is given by the following algorithm. However, we emphasize that this is only for the sake of explanation– the results in this paper do not use this algorithm.

1) Input a set of images, $j_1, j_2, ...j_n$, and a corresponding set of motions, $A_1, A_2, ...A_n$.
2) Warp each image to the central coordinate system, to obtain $j_i^{A_i}(\mathbf{x})$.
3) For each image, j_i compute the Fourier Transform of the warped image, $\mathcal{F}\{j_i^{A_i}\}(\mathbf{u}) = [I \cdot E^{A_i^{-T}}](\mathbf{u})$.
4) Create the reconstructed image in the Fourier domain. For all $\mathbf{u}$, set $K(\mathbf{u}) = \mathcal{F}\{j_l^{A_l}\}(\mathbf{u})$, where $l = \arg\min_i |A_i^{-T}\mathbf{u}|$.
5) Output the image in the spatial domain. $k(\mathbf{x}) = \mathcal{F}^{-1}\{K\}$.

Steps 1-3 simply create the frequency domain representations of the input images in a common coordinate system. To understand step 4, notice that for each frequency, we would like to take it from each image in which it has been filtered least. Now, since we assume that $E(\mathbf{u})$ is a monotonically decreasing function of $|\mathbf{u}|$ only, we would like to select the motion for which $|A_i^{-T}\mathbf{u}|$ is least.

However, this simple algorithm can only work when the global motion is affine. Here, we will present a different algorithm which operates completely in the spatial domain. It will later turn out that this allows us to do reconstruction for transformations that are only locally affine- for example full projective transformations. Nevertheless, the algorithm operates on the same principles as the simple one above.

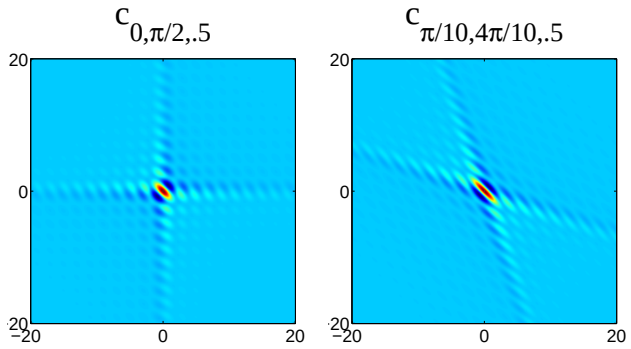
It is also important to consider what will happen if our assumptions on e violated. Suppose e is not exactly circularly symmetric, or suppose that it is spatially variant. (In real cameras these are both likely to be true to some degree.) As long as these conditions are not dramatically violated, the frequency segmentation boundaries will still be found nearby the optimal ones, and hence the reconstruction will still improve the results.

VI. The Frequency Slice Filter

The overall goal here is to create a filter which will pass a certain range of frequencies. More specifically, given $\theta_1$and θ_2, we would like a filter c such that, in the frequency domain (Fig. 4(a)),

$$C_{\theta_1,\theta_2}(\mathbf{u}) = \begin{cases} 1 & \theta_1 \leq \arg u \leq \theta_2 \\ 0 & \text{else} \end{cases} \tag{31}$$

Figure 5. The frequency slice filter in the spatial domain.



(a) (b)

First, suppose we would like to create a filter that passes exactly those frequencies in the first and third quadrants. (That is, $\theta_1 = 0$, $\theta_2 = \pi/2$.) Naively, we would just plug these values into the above equation. However, if we do this, the inverse Fourier Transform will not converge. A convenient fact is useful here- the images have already been low-pass filtered. Hence, it does not matter what the filter does to very high frequencies. So, instead, we will define the following filter (Fig. 4(b))

$$C_{0,\pi/2,r}(\mathbf{u}) = \begin{cases} 1 & 0 \leq \arg u \leq \pi/2, \text{ and } |u| \leq r \\ 0 & \text{else} \end{cases} \tag{32}$$

Before extending this to the case of other θ_1, θ_2, we will find the inverse Fourier transform.

$$c_{0,\pi/2,r}(\mathbf{x}) = \mathcal{F}^{-1}\{C_{0,\pi/2,r}(\mathbf{u})\} \tag{33}$$

$$= \int_{\mathbf{u}} C_{0,\pi/2,r}(\mathbf{u}) \exp(2\pi i \mathbf{u}^T\mathbf{x}) d\mathbf{u} \tag{34}$$

$$= \int_{\mathbf{0}\leq\mathbf{u}\leq\mathbf{r}} [\exp(2\pi i \mathbf{u}^T\mathbf{x}) + \exp(-2\pi i \mathbf{u}^T\mathbf{x})] d\mathbf{u} \tag{35}$$

$$= \int_{\mathbf{0}\leq\mathbf{u}\leq\mathbf{r}} 2\cos(2\pi \mathbf{u}^T\mathbf{x}) d\mathbf{u} \tag{36}$$

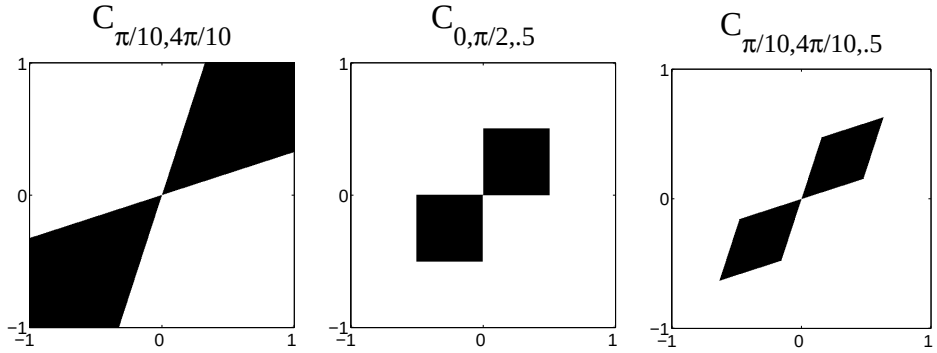
$$= \frac{1}{2\pi^2 xy}[\cos(2\pi rx) + \cos(2\pi ry) - \cos(2\pi r(x+y)) - 1] \tag{37}$$

Examples of c in the spatial domain are shown in Fig. 5. Notice specifically that this function decreases by the inverse of xy. Hence, it has relatively small extent in the spatial domain. This will be important in extending the approach to non-affine transformations. Now, we need to define the filter for arbitrary angles. Given θ_1, and θ_2, define the following matrix:

$$V = \begin{bmatrix} \cos\theta_1 & \cos\theta_2 \\ \sin\theta_1 & \sin\theta_2 \end{bmatrix} \tag{38}$$

Now, we can define the frequency slice filter for arbitrary angles.

Figure 4. The frequency slice filter in the Fourier domain.



(a) (b) (c)

Algorithm 1 Affine Reconstruction

1) Input a set of images, j_1, j_2, ... j_n, and a corresponding set of motions, A_1, A_2, ... A_n
2) Warp each image to the central coordinate system, to obtain $j_i^{A_i}(\mathbf{x})$.
3) Use the method described in Section IV to segment frequency space. Obtain a set of pairs of angles, along with the best motion in that region, $< \theta_{i1}, \theta_{i2}, A_i >$.
4) Output $k = \sum_i [j_i^{A_i} * c_{\theta_{i1},\theta_{i2},r}]$.

$$c_{\theta_1,\theta_2,r}(\mathbf{x}) = |V| c^{V^T}_{0,\pi/2,r}(\mathbf{x}) \tag{39}$$

To see why this works, apply the translation-free affine Fourier theorem to the right hand side.

$$\mathcal{F}\{|V| c^{V^T}_{0,\pi/2,r}(\mathbf{x})\} = |V| \cdot |V^{-T}| C^{(V^T)^{-T}}_{0,\pi/2,r}(\mathbf{u}) \tag{40}$$

$$\mathcal{F}\{|V| c^{V^T}_{0,\pi/2,r}(\mathbf{x})\} = C^{V^{-1}}_{0,\pi/2,r}(\mathbf{u}) \tag{41}$$

To see why V^{-1} works, notice that it will send $[cos\theta_1, \sin\theta_1]^T$ to $[1,0]^T$, and $[cos\theta_2, \sin\theta_2]^T$ to $[0,1]^T$. Hence only those frequencies in the correct range of angles will be passed. Fig. 4 (c) shows an example in the frequency domain with $\theta_1 = \pi/10$, $\theta_2 = 4\pi/10$.

VII. Affine Reconstruction

Given the tools that have been developed, we can now explicitly give the algorithm for affine reconstruction. The final method is quite simple. The images are warped to a common coordinate system, and then convolved with a filter calculated for each image. This filter depends on the results of frequency segmentation. Finally, the results of the convolutions are simply added together to produce the output image.

A. Example

VIII. General Reconstruction

The theory developed thus far has all been for the case of affine motion. We can observe, however, that it is essentially a local process- the filters have a small area of support. It turns out that we can extend the method to essentially arbitrary differentiable transformations. This is because any differentiable transformation can be locally approximated as affine. We will give examples here for projective transformations, but it is simplest to first show how to approximate a general transformation. Suppose that some function $\mathbf{t}(\mathbf{x})$ gives the transformation, so

$$\text{if } \mathbf{x}_2 = \mathbf{t}(\mathbf{x}_1) \text{ , then } i_2(\mathbf{x}_2) = i_1(\mathbf{x}_1). \tag{42}$$

It follows that

$$\forall \mathbf{x},\ i_2(\mathbf{x}) = i_1(\mathbf{t}^{-1}(\mathbf{x})). \tag{43}$$

Now, write j_2 in the usual way.

$$j_2(\mathbf{x}) = [i_2 * e](\mathbf{x}) \tag{44}$$

$$j_2(\mathbf{x}) = \int_{\mathbf{x}'} i_1(\mathbf{t}^{-1}(\mathbf{x} - \mathbf{x}')) e(\mathbf{x}') d\mathbf{x}' \tag{45}$$

Notice here that $e(\mathbf{x}')$ will be zero unless $\mathbf{x}'$ is small. So, we will use a local approximation for the transformation.

$$\mathbf{t}(\mathbf{x} - \mathbf{x}') \approx \mathbf{t}(\mathbf{x}) - J_{\mathbf{t}}(\mathbf{x})\mathbf{x}' \tag{46}$$

Where $J_{\mathbf{t}}(\mathbf{x})$ denotes the Jacobian of $\mathbf{t}$, evaluated at the point $\mathbf{x}$. Now, take the inverse.

$$\mathbf{t}^{-1}(\mathbf{x} - \mathbf{x}') \approx \mathbf{t}^{-1}(\mathbf{x}) - J^{-1}_{\mathbf{t}}(\mathbf{x})\mathbf{x}' \tag{47}$$

Substitute this in the above expression for j_2.

$$j_2(\mathbf{x}) = \int_{\mathbf{x}'} i_1(\mathbf{t}^{-1}(\mathbf{x}) - J^{-1}_{\mathbf{t}}(\mathbf{x})\mathbf{x}') e(\mathbf{x}') d\mathbf{x}' \tag{48}$$

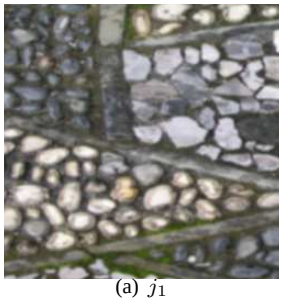
Now, change variables. Set $\mathbf{y} = J^{-1}_{\mathbf{t}}(\mathbf{x})\mathbf{x}'$.

$$j_2(\mathbf{x}) = \int_{\mathbf{y}} i_1(\mathbf{t}^{-1}(\mathbf{x}) - \mathbf{y}) e(J_{\mathbf{t}}(\mathbf{x})\mathbf{x}') |J_{\mathbf{t}}(\mathbf{x})| d\mathbf{y} \tag{49}$$

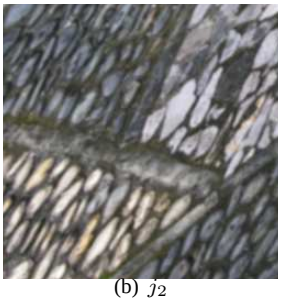
$$j_2(\mathbf{x}) = |J_{\mathbf{t}}(\mathbf{x})| [i_1 * e^{J_{\mathbf{t}}(\mathbf{x})}](\mathbf{t}^{-1}(\mathbf{x})) \tag{50}$$

So finally, we have a simple local approximation.

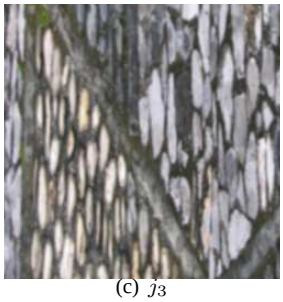
Figure 6. Affine Reconstruction



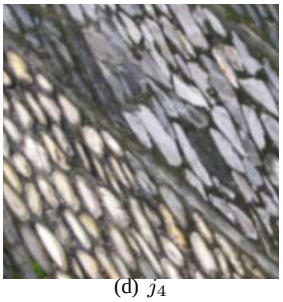
(a) j_1



(b) j_2



(c) j_3



(d) j_4

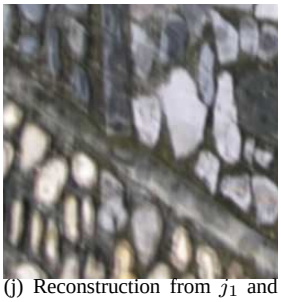
(e) $j_1^{A_1}$

(f) $j_2^{A_2}$

(g) $j_3^{A_3}$

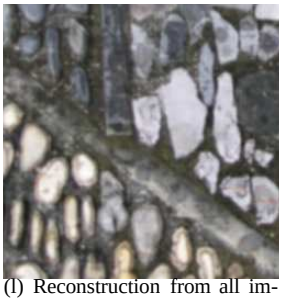
(h) $j_4^{A_4}$

(i) Average of $j_1^{A_1}$, $j_2^{A_2}$, $j_3^{A_3}$ and $j_4^{A_4}$

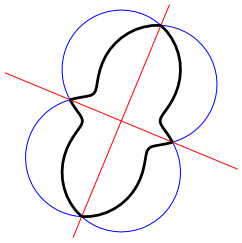


(j) Reconstruction from j_1 and j_2.

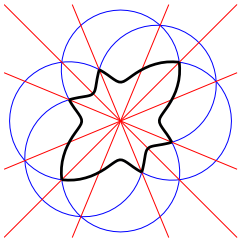
(k) Reconstruction from j_1, j_2 and j_3.



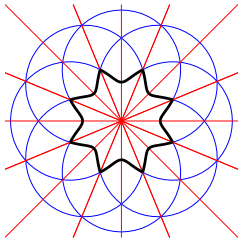
(l) Reconstruction from all images.



(m) Frequency segmentation for j_1 and j_2.



(n) Frequency segmentation for j_1, j_2, and j_3.



(o) Frequency segmentation for all images.

$$j_2(\mathbf{t}(\mathbf{x})) = |J_{\mathbf{t}}(\mathbf{x})|[i_1 * e^{J_{\mathbf{t}}(\mathbf{x})}](x) \tag{51}$$

The method for general reconstruction is given as Algorithm 2. Conceptually, the only difference with affine reconstruction is that the final image k is the sum of *spatially varying* filters convolved with the input images.

A. Projective Reconstruction

In the experiments of this paper, we will focus on reconstruction from images that were taken under perspective projection. It is most common to write such a transformation in homogeneous coordinates, in which case it is just an arbitrary linear transformation. Here, however, we use non-homogeneous coordinates. In this case, let $\mathbf{x}_1 = [x_1, y_1]^T$ and $\mathbf{x}_2 = [x_2, y_2]^T$ be corresponding points in two images. Then, for some parameters $h_1, h_2, ..., h_9$, $\mathbf{x}_2 = \mathbf{t}(\mathbf{x}_1)$ is equivalent to

$$x_2 = \frac{h_1 x_1 + h_2 y_1 + h_3}{h_7 x_1 + h_8 y_1 + h_9}, \tag{52}$$

$$y_2 = \frac{h_4 x_1 + h_5 y_1 + h_6}{h_7 x_1 + h_8 y_1 + h_9}. \tag{53}$$

The Jacobian is simply a matrix containing four partial derivatives.

Algorithm 2 General Reconstruction

1) Input a set of images, j_1, j_2, ... j_n, and a corresponding set of transformations, $\mathbf{t}_1$, $\mathbf{t}_2$, ... $\mathbf{t}_n$
2) Warp each image to the central coordinate system, to obtain $j_i^{\mathbf{t}_i}(\mathbf{x})$.
3) For each point $\mathbf{x}$,
   a) For each transformation $\mathbf{t}_i$, compute the Jacobian at $\mathbf{x}$, $J_{\mathbf{t}_i}(\mathbf{x})$.
   b) Use the method described in Section IV to segment frequency space. Obtain a set of pairs of angles, along with the best motion in that region, $< \theta_{i1}, \theta_{i2}, J_{\mathbf{t}_i}(\mathbf{x}) >$.
   c) Set $k(\mathbf{x}) = \sum_i [j_i^{J_{\mathbf{t}_i}(\mathbf{x})} * c_{\theta_{i1},\theta_{i2},r}](\mathbf{x})$
4) Output k.

$$J_{\mathbf{t}}(\mathbf{x}) = \begin{bmatrix} \frac{\partial x_2}{\partial x_1} & \frac{\partial x_2}{\partial y_1} \\ \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial y_1} \end{bmatrix} \tag{54}$$

These are easily evaluated.

$$\frac{\partial x_2}{\partial x_1} = \frac{y_1(h_1h_8 - h_7h_2) + h_1h_9 - h_7h_3}{(h_7x_1 + h_8y_1 + h_9)^2} \tag{55}$$

$$\frac{\partial x_2}{\partial y_1} = \frac{x_1(h_2h_7 - h_8h_1) + h_2h_9 - h_8h_3}{(h_7x_1 + h_8y_1 + h_9)^2} \tag{56}$$

$$\frac{\partial y_2}{\partial x_1} = \frac{y_1(h_4h_8 - h_7h_5) + h_4h_9 - h_7h_6}{(h_7x_1 + h_8y_1 + h_9)^2} \tag{57}$$

$$\frac{\partial y_2}{\partial y_1} = \frac{x_1(h_5h_7 - h_8h_4) + h_5h_9 - h_8h_6}{(h_7x_1 + h_8y_1 + h_9)^2} \tag{58}$$

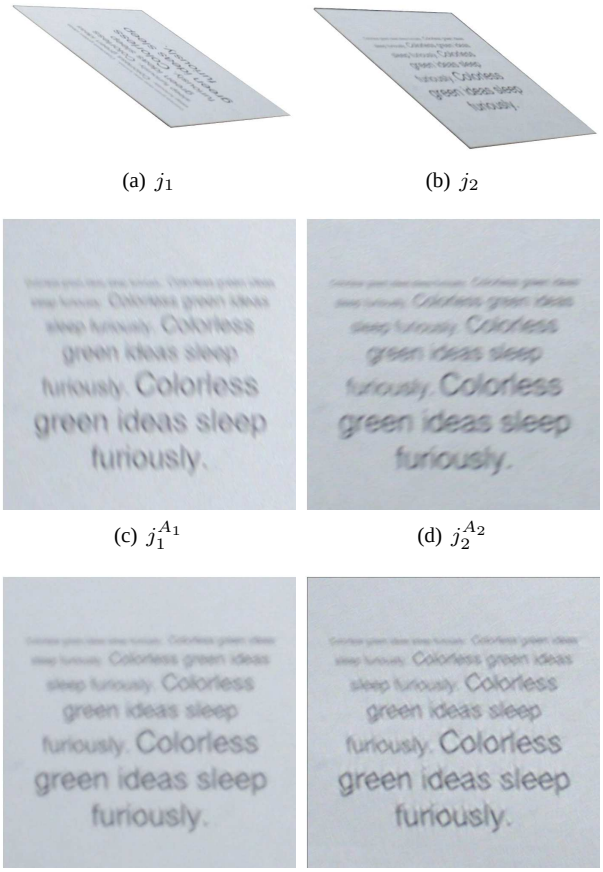
B. Making Reconstruction Faster

For affine reconstruction, the algorithm is extremely fast- the computation is dominated by the convolutions in step 4. So, if there are n input images, the algorithm will just take the time to run n convolutions. In the algorithm for general reconstruction, however, the filters are spatially varying, and need to be recalculated at each pixel. This makes it much slower.

A simple trick can speed this up. Instead of calculating the filters for each pixel, calculate them for some small patch of pixels. If the affine approximation is slowly changing, this will result in no minimal change to the output, while hugely reducing the overhead for recomputing filters. For example, if we use the same filter for each 20 by 20 patch of pixels, we reduce the overhead by a factor of 400.

As above, this strategy could introduce artifacts on some images, at the boundary where the filters change. To reduce this, we allow the patches to overlap and use a small amount of blending. In these experiments, a one-pixel overlap was sufficient. Suppose we are using 4 by 4 patches. Then, when adding the result of the convolution to the output image, it is first multiplied by a mask of

Figure 7. Projective Reconstruction



(a) j_1 (b) j_2

(c) $j_1^{A_1}$ (d) $j_2^{A_2}$

(e) Average of $j_1^{A_1}$ and $j_2^{A_2}$ (f) Reconstuction.

$$\begin{bmatrix} .25 & .5 & .5 & .25 \\ .5 & 1 & 1 & .5 \\ .5 & 1 & 1 & .5 \\ .25 & .5 & .5 & .25 \end{bmatrix}. \tag{59}$$

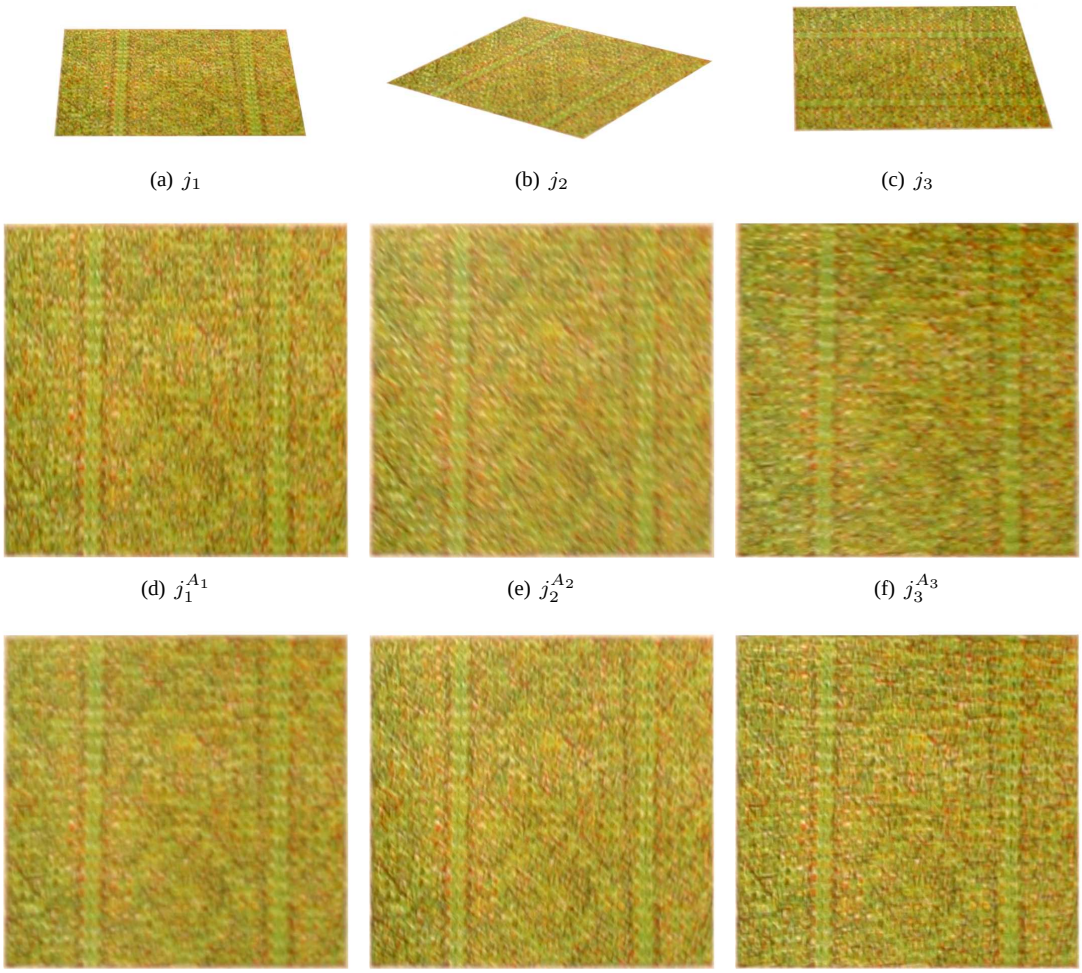
The convolutions are computed for every third pixel both dimensions. Notice that the corner pixels will eventually be the sum of four convolutions, while the sides will be the sum of two.

C. Experiments

Figure 7 shows a reconstruction from two real views of a surface with printed text. After reconstruction, for presentation, we segment out the background. We compare our results to a simple averaging process. Notice that reconstruction succeeds despite the changes in lighting between the two images. Since the output image is just the sum of the input images convolved with different filters, lighting changes, noise, etc. naturally average out.

Figures 8 and 9 show the reconstruction process for images taken of two different textures. To better understand how using more images leads to better reconstruction, we also include the results of reconstruction using only the first two of the three images

Figure 8. Projective Reconstruction



(a) j_1 (b) j_2 (c) j_3

(d) $j_1^{A_1}$ (e) $j_2^{A_2}$ (f) $j_3^{A_3}$

(g) Average of $j_1^{A_1}$, $j_2^{A_2}$, and $j_3^{A_3}$ (h) Reconstuction from j_1 and j_2. (i) Reconstuction from j_1, j_2, and j_3.

IX. Conclusions

This paper introduced the formalism of the "ideal image", consisting of the unblurred incoming light, and the "real image" consisting of the measured image, after blurring. We showed that because this framework separates the filtering and geometrical aspects, it makes it easy to derive several results. As an example of this framework, we showed that it is possible to perform multiple view image reconstruction, even with an unknown blurring kernel.

X. Appendix: Proofs

Theorem 7 (Translation-Free Affine Fourier Theorem):

If $\mathcal{F}\{f(\mathbf{x})\} = F(\mathbf{u})$ then $\mathcal{F}\{f^A(\mathbf{x})\} = |A^{-1}|F^{A^{-T}}(\mathbf{u})$.

Proof:

$$\mathcal{F}\{f^A(\mathbf{x})\}(\mathbf{u}) = \int f(A\mathbf{x})e^{-2\pi\mathbf{u}^T\mathbf{x}}d\mathbf{x} \tag{60}$$

Now, define $\mathbf{x}' = A\mathbf{x}$, and change variables.

$$\mathcal{F}\{f^A(\mathbf{x})\}(\mathbf{u}) = |A^{-1}|\int f(\mathbf{x}')e^{-2\pi(A^{-T}\mathbf{u})^T\mathbf{x}'}d\mathbf{x}' \tag{61}$$

But this is just the Fourier transform of f, evaluated at the point $A^{-T}\mathbf{u}$.

$$\mathcal{F}\{f^A(\mathbf{x})\}(\mathbf{u}) = |A^{-1}|F(A^{-T}\mathbf{u}) \tag{62}$$

■

Lemma 1 (Affine Convolution Lemma):

$$[f^A * g](\mathbf{x}) = \frac{1}{|A|}[f * g^{A^{-1}}](A\mathbf{x})$$

Proof:

By definition,

$$[f^A * g](\mathbf{x}) = \int f(A\mathbf{x} - A\mathbf{x}')g(\mathbf{x}')d\mathbf{x}'. \tag{63}$$

Now, define $\mathbf{y} = A\mathbf{x}'$. Then, we can re-write the integral.

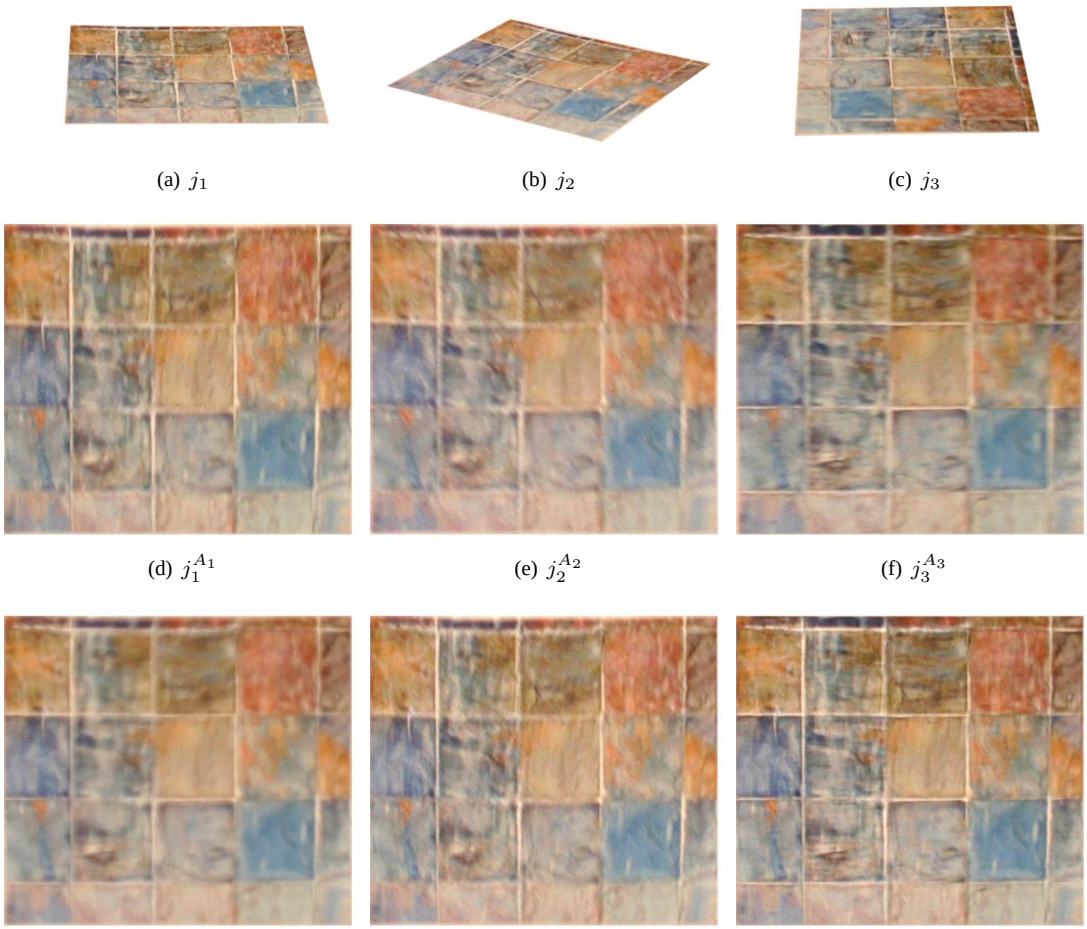
$$[f^A * g](\mathbf{x}) = \frac{1}{|A|}\int f(A\mathbf{x} - \mathbf{y})g(A^{-1}\mathbf{y})d\mathbf{y} \tag{64}$$

$$[f^A * g](\mathbf{x}) = \frac{1}{|A|}[f * g^{A^{-1}}](A\mathbf{x}) \tag{65}$$

■

Theorem 8 (Affine Blurring Theorem):

Figure 9. Projective Reconstruction



(a) j_1 (b) j_2 (c) j_3

(d) $j_1^{A_1}$ (e) $j_2^{A_2}$ (f) $j_3^{A_3}$

(g) Average of $j_1^{A_1}$, $j_2^{A_2}$, and $j_3^{A_3}$ (h) Reconstuction from j_1 and j_2. (i) Reconstuction from j_1, j_2, and j_3.

$$j_2^A(\mathbf{x}) = |A|[i_1 * e^A](\mathbf{x})$$

Proof:
By definition, $i_2(\mathbf{x}) = i_1^{A^{-1}}(\mathbf{x})$. Hence,

$$j_2^A(\mathbf{x}) = [i_2 * e](A\mathbf{x}) = [i_1^{A^{-1}} * e](A\mathbf{x}). \quad (66)$$

Now, apply the affine convolution lemma to the right hand side.

$$[i_1^{A^{-1}} * e](A\mathbf{x}) = |A|[i_1 * e^A](A^{-1}A\mathbf{x}) \quad (67)$$

■

Theorem 9 (Fourier Affine Blurring Theorem):
$$\mathcal{F}\{j_2^A(\mathbf{x})\} = [I_1 \cdot E^{A^{-T}}](\mathbf{u})$$

Proof:
Start with the result of the affine blurring theorem.

$$j_2^A(\mathbf{x}) = |A|[i_1 * e^A](\mathbf{x}) \quad (68)$$

Now, apply the affine Fourier theorem to e^A.

$$\mathcal{F}\{e^A(\mathbf{x})\} = |A^{-1}|E^{A^{-T}}(\mathbf{u}) \quad (69)$$

Now, if we apply the convolution theorem to both sides of the affine blurring theorem, the result follows. ■

REFERENCES

[1] R.N. Bracewell, K.-Y. Chang, A.K. Jha, and Y.-H. Wang. Affine theorem for two-dimensional fourier transform. *Electronics Letters*, 29(3):304, 1993.

[2] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.

[3] Tony Lindeberg. On the axiomatic foundations of linear scale-space: Combining semi-group structure with causality vs. scale invariance. Technical report, Department of Numerical Analysis and Computing Science, Royal Institute of Technology, 1994.

[4] Tony Lindeberg and Jonas Gårding. Shape-adapted smoothing in estimation of 3-d shape cues from affine deformations of local 2-d brightness structure. *Image Vision Comput.*, 15(6):415–434, 1997.

[5] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *Int. J. Comput. Vision*, 65(1-2):43–72, 2005.

[6] Krystian Mikolajczyk and Cordelia Schmid. Scale & affine invariant interest point detectors. *Int. J. Comput. Vision*, 60(1):63–86, 2004.

[7] S. Ravela. Shaping receptive fields for affine invariance. In *CVPR*, volume 02, pages 725–730, 2004.

[8] J. V. Stone. Shape from texture: Textural invariance and the problem of scale in perspective images of textures surface. In *BMVC*, pages 181–187, 1990.

[9] J. V. Stone and S. D. Isard. Adaptive scale filtering: A general method for obtaining shape from texture. *IEEE Trans. Pattern Anal. Mach. Intell.*, 17(7):713–718, 1995.

[10] Lifeng Wang, Sing Bing Kang, Richard Szeliski, and Heung-Yeung Shum. Optimal texture map reconstruction from multiple views. In *CVPR (1)*, pages 347–354, 2001.