

CREATING HIGH DYNAMIC RANGE MANIFOLD MOSAICS USING A VIDEO CAMERA WITH AUTOMATIC GAIN CONTROL

Chris Aimone

University of Toronto
Dept. of Electrical and Computer Engineering
10 King's College Rd.
Toronto, Canada

ABSTRACT

This paper presents a system that automatically produces high dynamic range manifold mosaics from a stream of images captured by a handheld camcorder. The method gathers multiple images of the same subject matter at different exposure levels, by simply allowing the camera's automatic gain control feature (AGC) to operate while the camera is in motion. The image registration algorithm then simultaneously estimates the spatial and tonal alignment between pairs of images from the video sequence. The 8 parameter projective transformation is used for the spatial alignment between pairs of images. Tonal alignment is computed by using the inverse camera response function to linearize the pixel response with respect to light quantity, and then finding the scalar gain between the images. The alignment model thus has 9 parameters, which are jointly estimated in an iterative multiscale least-squares method. Since it is not usually possible to accurately adjust the exposure of a hand held video camera, conventional methods for recovering the camera response function cannot be used. To solve this problem a new method of obtaining camera response function is presented that only requires that the camera be equipped with an exposure lock feature. The method solves for the response function directly using superposition constraints imposed by different combinations of two (or more) lights to illuminate the same subject matter.

1. INTRODUCTION

The creation of panoramic images from images obtained from a camera that is free to pan, tilt and zoom, is well understood and has been explored by many in the computer vision and graphics fields. It seems however, that the tonal registration between images is usually neglected. Many methods as a result, require that the exposure is not changed during the image acquisition process. This restriction seems rather unpleasant, since panoramic scenes often contain large variations in lighting. It is also the case that many inexpensive video devices, such as USB webcams are not equipped with any manual exposure features.

The method described in this paper is a natural combination of an interactive image space registration technique with the photogrammetric methods of high dynamic range imaging. The technique allows panoramic scenes to be captured with an AGC enabled video camera, and it facilitates the construction of high dynamic range images, simply by centering the camera's view on different portions of the scene (with varying brightness). For example, if when using AGC, we point the camera at bright objects,

the gain decreases so that darker objects in the periphery may be underexposed. Similarly, when we point the camera at dark objects, the periphery will be over exposed. Since, the AGC in video cameras, tend to respond quite slowly, as we pan from a bright object to a dark object, the camera will transition smoothly through a range of exposures, giving us a nice range of exposure settings for the common subject matter.

To achieve a good spatial alignment between images, it is important to use a model that describes the transformation between images accurately. The model most commonly used, is the eight parameter projective coordinate transformation:

$$\mathbf{x}' = \begin{bmatrix} x' \\ y' \end{bmatrix} = \frac{\mathbf{A}[x, y]^T + \mathbf{b}}{\mathbf{c}^T[x, y]^T + 1} = \frac{\mathbf{A}\mathbf{x} + \mathbf{b}}{\mathbf{c}^T\mathbf{x} + 1} \quad (1)$$

where \mathbf{A} , \mathbf{b} , \mathbf{c} contain the eight parameters of the projective coordinate transformation (see equation (6)).

This model is ideal since when applied across the entire image, it can describe exactly the relation between two images where all the objects in the scene are static and the camera is only free to rotate and zoom. It is also exact for a planar scene imaged from arbitrary locations. This latter point is not so important for the common cylindrical and spherical panoramic mosaics, however for the more general mosaicing framework presented by Peleg et. al. [1] This extended capability is useful.

Since the handheld panning motion is primarily rotational, the presented system makes use of a simple rotational model presented by Peleg et. al. in [2] to coarsely estimate the transformation between images. This model describes the relation between images as:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \sim \mathbf{T}\mathbf{V}\mathbf{R}\mathbf{V}^{-1}\mathbf{T}^{-1} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (2)$$

where

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & c_x \\ 0 & 1 & c_y \\ 0 & 0 & 1 \end{bmatrix}, \mathbf{V} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3)$$

, and \mathbf{R} is the rotation matrix describing the camera's 3D rotation

about its optical centre:

$$\mathbf{R} = \begin{bmatrix} r_{00} & r_{01} & r_{02} \\ r_{10} & r_{11} & r_{12} \\ r_{20} & r_{21} & r_{22} \end{bmatrix} \quad (4)$$

f is the camera focal length in pixels. (c_x, c_y) is camera's principle point (where the optical axis of the camera intersects the image plane).¹

The rotational model (2) requires that we know the focal length of the camera, however the reduced number of free parameters (three in total) makes the matching algorithm more stable and thus more able cope with large camera motions. For the fine adjustments in image registration, the presented methods makes use of the full eight parameter model (1) with a ninth parameter added to describe the change in camera exposure between images. Both of these methods will be described in detail in the next section.

To use a single gain parameter to describe the change in pixel values due to a change in exposure, the camera must be linear. For example, if in image one, pixel (x, y) has value $\mathbf{I}_1(\mathbf{x}, \mathbf{y})$ and we double the exposure for image two, then corresponding pixels must have twice the value. i.e. $\mathbf{I}_2(\mathbf{x}, \mathbf{y}) = 2\mathbf{I}_1(\mathbf{x}, \mathbf{y})$. It is known that most cameras can be quite non-linear especially in the extents of their range. It is therefore necessary to find a function that relates the pixel values to photoquantities. This function is known as the camera response function. Once this function is known, the camera can be linearized by applying the inverse camera response function to the pixel values.

In this paper, a method is presented that determines the inverse response function of the camera by using the superposition property of light. The method solves for the inverse response function directly using superposition constraints imposed by different combinations of two (or more) lights to illuminate the same subject matter. The method overcomes the problem of needing to accurately set the exposure of the camera, and requires only that the camera be equipped with an exposure lock feature.

2. SPATIAL ALIGNMENT

The basic spatial alignment technique is very similar to the method presented by Shum and Szeliski in [2]. Alignment between pairs of image is done in a coarse to fine framework, where first, an image pyramid is computed, then starting at the coarsest level, the motion model of choice is fit between the images using an iterative method.

2.1. 8-Parameter Projective Transformation

At a given scale, we wish to estimate the motion between two images $\mathbf{I}_0(\mathbf{x})$ and $\mathbf{I}_1(\mathbf{x})$, in terms of a parametric motion model $\mathbf{x}' = \mathbf{f}(\mathbf{x}, \mathbf{m})$ where the components of \mathbf{m} are the motion parameters. We would thus like to minimize:

$$\epsilon_{\mathbf{m}} = \sum_{\forall \mathbf{x}} [\mathbf{I}_1(\mathbf{f}(\mathbf{x}, \mathbf{m})) - \mathbf{I}_0(\mathbf{x})]^2 \quad (5)$$

Here we are assuming that the exposure has not changed between images. This case will be considered in a later section. For the full

¹ c_x and c_y can be approximated by using the pixel coordinates for the centre of the image.

eight parameter projective model:

$$f(\mathbf{x}, \mathbf{m}) = \begin{bmatrix} x' \\ y' \end{bmatrix} = \frac{\begin{bmatrix} m_{00} & m_{01} \\ m_{10} & m_{11} \end{bmatrix} [\mathbf{x}, \mathbf{y}]^T + [\mathbf{m}_{02}, \mathbf{m}_{12}]^T}{[\mathbf{m}_{20}, \mathbf{m}_{21}] [\mathbf{x}, \mathbf{y}]^T + 1} \quad (6)$$

To iteratively solve for \mathbf{m} , we compute the first order Taylor series approximation of $\mathbf{I}_1(\mathbf{f}(\mathbf{x}, \mathbf{m}))$ with respect to \mathbf{m} .

$$\mathbf{I}_1(\mathbf{f}(\mathbf{x}, \mathbf{m})) \approx \mathbf{I}_1(\mathbf{x}) + \nabla \mathbf{I}_1(\mathbf{x}) \frac{\partial \mathbf{f}}{\partial \mathbf{m}} \mathbf{m} \quad (7)$$

By using an update vector $\mathbf{d} = [d_0, d_1, d_2, d_3, d_4, d_5, d_6, d_7]^T$ such that:

$$\begin{aligned} & [m_{00}, m_{01}, m_{02}, m_{10}, m_{11}, m_{12}, m_{30}, m_{31}, m_{32}]^T \\ & = [1+d_0, 1+d_1, 1+d_2, 1+d_3, 1+d_4, 1+d_5, 1+d_6, 1+d_7, 1]^T \end{aligned} \quad (8)$$

We can simplify (9) to:

$$\mathbf{I}_1(\mathbf{f}(\mathbf{x}, \mathbf{m})) \approx \mathbf{I}_1(\mathbf{x}) + \nabla \mathbf{I}_1(\mathbf{x}) \mathbf{J}(\mathbf{x}) \mathbf{d} \quad (9)$$

where, $\nabla \mathbf{I}_1(\mathbf{x})$ is the image gradient of \mathbf{I}_1 at location \mathbf{x} , and

$$\mathbf{J}(\mathbf{x}) = \frac{\partial \mathbf{f}}{\partial \mathbf{m}} = \begin{bmatrix} x & y & 1 & 0 & 0 & 0 & -x^2 & -xy \\ 0 & 0 & 0 & x & y & 1 & -xy & -y^2 \end{bmatrix}^T \quad (10)$$

is the Jacobian of the projective transformation with respect \mathbf{m} .

By combining this result with equation (5), we obtain:

$$\epsilon_{\mathbf{m}} \approx \sum_{\forall \mathbf{x}} [\mathbf{I}_1(\mathbf{x}) + \nabla \mathbf{I}_1(\mathbf{x}) \mathbf{J}(\mathbf{x}) \mathbf{d} - \mathbf{I}_0(\mathbf{x})]^2 \quad (11)$$

To minimize (11), we use the standard least-squares solution by *normal equations*.

$$\mathbf{A} \mathbf{d} = -\mathbf{b} \quad (12)$$

where

$$\mathbf{A} = \sum_{\forall \mathbf{x}} [\mathbf{J}(\mathbf{x}) \nabla \mathbf{I}_1(\mathbf{x}) \nabla \mathbf{I}_1(\mathbf{x})^T \mathbf{J}(\mathbf{x})^T]^2 \quad (13)$$

is the *Hessian*, and

$$\mathbf{b} = \sum_{\forall \mathbf{x}} [(\mathbf{I}_1(\mathbf{x}) - \mathbf{I}_0(\mathbf{x})) \mathbf{J}(\mathbf{x}) \nabla \mathbf{I}_1(\mathbf{x})] \quad (14)$$

is the *residual*. Since \mathbf{A} is symmetric and non-singular, it is easily inverted to solve for the motion parameters \mathbf{d} .

Since equation (9) is only a first order approximation of $\mathbf{I}_1(\mathbf{f}(\mathbf{x}, \mathbf{m}))$, the solution for \mathbf{m} by (8) will only be accurate for small values of \mathbf{d} . This means that in general the estimated projective transformation between pairs of images will only be accurate for small camera motions. In order to increase the range of the technique (larger camera motions), we make use of the fact that the set of projective transformations forms a group and is thus closed under composition, to iteratively solve for desired motion parameters.

The iterative method has the following steps:

- (i) Estimate the motion parameters \mathbf{m} of equation (6) by minimizing equation (11).
- (ii) Use the inverse transformation $f^{-1}(\mathbf{x}, \mathbf{m})$ to warp the original image I_0 to create \tilde{I}_0 .

$$\tilde{I}_0(\mathbf{x}) = I_0(f^{-1}(\mathbf{x}, \mathbf{m})) \quad (15)$$

- (iii) Estimate the motion parameters \mathbf{m}' between $I_1(\mathbf{x})$ and $\tilde{I}_0(\mathbf{x})$ by using the updated *residual* vector $\tilde{\mathbf{b}}$:

$$\tilde{\mathbf{b}} = \sum_{\forall \mathbf{x}} \left[(I_1(\mathbf{x}) - \tilde{I}_0(\mathbf{x})) \mathbf{J}(\mathbf{x}) \nabla I_1(\mathbf{x}) \right] \quad (16)$$

- (iv) Generate an improved motion estimate of $f(\mathbf{x}, \mathbf{m})$ by composing $f(\mathbf{x}, \mathbf{m}')$ and $f(\mathbf{x}, \mathbf{m})$.
- (v) goto (ii) until the desired accuracy has been achieved

To make use of the group properties of the projective motion model (6), it is natural to use the matrix formulation in homogeneous coordinates.

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \sim \begin{bmatrix} m_{00} & m_{01} & m_{02} \\ m_{10} & m_{11} & m_{12} \\ m_{20} & m_{21} & m_{22} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (17)$$

or in vector notation:

$$\mathbf{X}' \sim \mathbf{M}\mathbf{X} \quad (18)$$

The inverse transformation can now be simply written as:

$$\mathbf{X} \sim \mathbf{M}^{-1}\mathbf{X}' \quad (19)$$

Incremental motions can then be composed into one transformation by using this representation. I.e.

$$\mathbf{x}' = \mathbf{f}(\mathbf{f}(\mathbf{x}, \mathbf{m}_1), \mathbf{m}_2) = \mathbf{f}(\mathbf{x}, \mathbf{m}') \quad (20)$$

as

$$\mathbf{X}' \sim \mathbf{M}_2\mathbf{M}_1\mathbf{X} = \mathbf{M}'\mathbf{X} \quad (21)$$

In the actual implementation of this algorithm, a coarse to fine strategy is employed with the following steps:

- (a) Construct an image pyramid for images I_0 and I_0 , by blurring with a Gaussian kernel and decimating.
- (b) Low pass filter each image with a Gaussian kernel to smooth the spatial image gradients.
- (c) Compute ∇I_1 using differences between adjacent pixels.
- (d) Starting at the coarsest level, estimate the projective transformation between the images using the iterative method described above. Compute a predetermined number of iterations before proceeding to the next level.

To improve the numerical stability of the method, all the images are rescaled such that $x, y \in (-0.5, 0.5]$. The actual transformation between images given by \mathbf{M} is found from \mathbf{M}_s the transformation estimated from the scaled images:

$$\mathbf{M} = \mathbf{S}\mathbf{M}_s\mathbf{S}^{-1} \quad (22)$$

where

$$\mathbf{S} = \begin{bmatrix} s_x & 0 & c_x \\ 0 & s_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \quad (23)$$

Here s_x and s_y are the x and y dimensions of the images.

2.2. 3-Parameter Projective Transformation (Rotational Model)

In the restricted scenerio where the camera motion is restricted to rotations (in three dimensions) about the camera's optic center. The projective transformation relating pairs of overlapping images can be described completely by the three Euler angles, describing the 3D-rotation of the camera. This relationship is given by:

$$\mathbf{M} \sim \mathbf{T}\mathbf{V}\mathbf{R}\mathbf{V}^{-1}\mathbf{T}^{-1} \quad (24)$$

where \mathbf{T} , \mathbf{V} and \mathbf{R} are defined in equations (3) and (4).

To solve for \mathbf{M} in an iterative scheme as in the previous section, we use a linear approximation of the projective transformation for small rotations $\Omega = [\Delta\theta_x, \Delta\theta_y, \Delta\theta_z]^T$ about the camera's optical center in the camera coordinate system.

$$\mathbf{M} \approx \mathbf{T}\mathbf{V}[\mathbf{I} + \mathbf{X}(\Omega)]\mathbf{V}^{-1}\mathbf{T}^{-1} = \mathbf{T}(\mathbf{I} + \mathbf{D}_\Omega)\mathbf{T}^{-1} \quad (25)$$

where

$$\mathbf{X}(\Omega) = \begin{bmatrix} 0 & -\Delta\theta_z & \Delta\theta_y \\ \Delta\theta_z & 0 & -\Delta\theta_x \\ -\Delta\theta_y & \Delta\theta_x & 0 \end{bmatrix} \quad (26)$$

is the cross product operator, and

$$\mathbf{D}_\Omega = \mathbf{V}\mathbf{X}(\Omega)\mathbf{V}^{-1} = \begin{bmatrix} 0 & -\Delta\theta_z & f\Delta\theta_y \\ \Delta\theta_z & 0 & -f\Delta\theta_x \\ -\Delta\theta_y/f & \Delta\theta_x/f & 0 \end{bmatrix} \quad (27)$$

To solve for \mathbf{D}_Ω , we minimize:

$$\varepsilon_\Omega \approx \sum_{\forall \mathbf{x}} [I_1(\mathbf{x}) + \nabla I_1(\mathbf{x})\mathbf{J}_\Omega(\mathbf{x})\Omega - I_0(\mathbf{x})]^2 \quad (28)$$

with the origin of each image shifted using transformation \mathbf{T} . The least-squares solution by *normal equations* is again:

$$\mathbf{A}_r\Omega = -\mathbf{b}_r \quad (29)$$

where

$$\mathbf{A}_r = \sum_{\forall \mathbf{x}} \left[\mathbf{J}_\Omega(\mathbf{x})\nabla I_1(\mathbf{x})\nabla I_1(\mathbf{x})^T \mathbf{J}_\Omega(\mathbf{x})^T \right]^2 \quad (30)$$

and

$$\mathbf{b}_r = \sum_{\forall \mathbf{x}} [(I_1(\mathbf{x}) - I_0(\mathbf{x})) \mathbf{J}_\Omega(\mathbf{x})\nabla I_1(\mathbf{x})] \quad (31)$$

and, the Jacobian of the projective transformation with respect to $[\Delta\theta_x, \Delta\theta_y, \Delta\theta_z]^T$ is:

$$\mathbf{J}_\Omega(\mathbf{x}) = \frac{\partial \mathbf{f}}{\partial \Omega} = \begin{bmatrix} -xy/f & f + x^2/f & -y \\ -f - y^2/f & xy/f & x \end{bmatrix}^T \quad (32)$$

We can use the iterative method presented in the last section by using this model for estimating the motion parameters at steps (i) and (iii).

2.3. Estimating the Camera's Focal Length

To use the 3-parameter motion model for estimating the projective transformation, we need to know the camera's focal length. To estimate the focal length from pairs of images that are well registered using a projective transformation, the method presented by Szeliski and Shum in [2] was implemented. It was found however that estimates provided by this method were very inconsistent when applied to typical motion estimates from a handheld camera. The lack of an error measure also makes it difficult to compare the estimates that result from a large set of registered images. In [2] the median value is used as the best focal length estimate. This however, performs poorly in the presence of outlier points, which will frequently occur with any non-rotational camera motion. The observed inconsistency in the method is however most likely due to the implicit assumption that recovered projective transformations obey equation (24). To address these problems, a new method was developed.

For general camera motions, the recovered projective transformation \mathbf{M} when decomposed as in equation (24), will result in a value for \mathbf{R} that is not orthonormal. We should thus rewrite (24) as:

$$\mathbf{M} \sim \mathbf{T}\mathbf{V}\hat{\mathbf{R}}\mathbf{V}^{-1}\mathbf{T}^{-1} \quad (33)$$

By rearranging we get:

$$\hat{\mathbf{R}} \sim \mathbf{V}^{-1}\mathbf{T}^{-1}\mathbf{M}\mathbf{T}\mathbf{V} \quad (34)$$

Since \mathbf{M} and \mathbf{T} are known, we can ease the notation by writing:

$$\mathbf{T}^{-1}\mathbf{M}\mathbf{T} = \mathbf{Q} = \begin{bmatrix} q_{00} & q_{01} & q_{02} \\ q_{10} & q_{11} & q_{12} \\ q_{20} & q_{21} & q_{22} \end{bmatrix} \quad (35)$$

Therefore,

$$\hat{\mathbf{R}} \sim \begin{bmatrix} q_{00} & q_{01} & q_{02}/f \\ q_{10} & q_{11} & q_{12}/f \\ q_{20}f & q_{21}f & q_{22} \end{bmatrix} \quad (36)$$

The strategy is then to find the optimal value of f that make $\hat{\mathbf{R}}$ as similar as possible to a true rotation matrix. In other words, we would like to solve for the "best" rotation matrix \mathbf{R} to approximate $\hat{\mathbf{R}}$. Here, "best" is quantified by the minimum Frobenius norm of the difference $\mathbf{R} - \alpha\hat{\mathbf{R}}$, where α is a scalar multiplier to account for the similarity relationship. Stated formally, the problem is:

$$\mathbf{R} = \arg \min_{\mathbf{R}, \alpha} \|\mathbf{R} - \alpha\hat{\mathbf{R}}\|_F^2 \quad (37)$$

with the constraint that:

$$\mathbf{R}^T\mathbf{R} = \mathbf{I} \quad (38)$$

Since

$$\|\mathbf{R} - \alpha\hat{\mathbf{R}}\|_F^2 = \mathfrak{h}((\mathbf{R} - \alpha\hat{\mathbf{R}})^T(\mathbf{R} - \alpha\hat{\mathbf{R}})) \quad (39)$$

$$= 3 + \alpha^2\mathfrak{h}(\hat{\mathbf{R}}^T\hat{\mathbf{R}}) - 2\alpha\mathfrak{h}(\mathbf{R}^T\hat{\mathbf{R}}) \quad (40)$$

where \mathfrak{h} denotes the *trace* operator.

By computing the singular value decomposition (SVD) of $\hat{\mathbf{R}}$, we have $\hat{\mathbf{R}} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T$, where $\mathbf{\Lambda} = \text{diag}(\sigma_1, \sigma_2, \sigma_3)$. We would thus like to minimize (40):

$$\vartheta = 3 + \alpha^2\mathfrak{h}(\mathbf{V}\mathbf{\Lambda}\mathbf{U}^T\mathbf{U}\mathbf{\Lambda}\mathbf{V}^T) - 2\alpha\mathfrak{h}(\mathbf{R}^T\mathbf{U}\mathbf{\Lambda}\mathbf{V}^T) \quad (41)$$

Using the properties of \mathfrak{h} , this becomes:

$$= 3 + \alpha^2\mathfrak{h}(\mathbf{\Lambda}^2) - 2\alpha\mathfrak{h}(\mathbf{V}^T\mathbf{R}^T\mathbf{U}\mathbf{\Lambda}) \quad (42)$$

$$= 3 + \alpha^2 \sum_i \sigma_i^2 - 2\alpha \sum_i z_{ii}\sigma_i \quad (43)$$

where z_{ii} are the diagonal elements of $\mathbf{Z} = \mathbf{V}^T\mathbf{R}^T\mathbf{U}$. Since \mathbf{Z} is an orthogonal matrix:

$$\vartheta \geq 3 + \alpha^2 \sum_i \sigma_i^2 - 2\alpha \sum_i \sigma_i \quad (44)$$

ϑ is clearly minimized by setting $\mathbf{R} = \mathbf{U}\mathbf{V}^T$.

We can then solve for the minimizing α by differentiating (41) by α and setting the result to 0.

$$\frac{\partial \vartheta}{\partial \alpha} = 2\alpha \sum_i \sigma_i^2 - 2 \sum_i \sigma_i \quad (45)$$

Therefore,

$$\alpha = \frac{\sum_i \sigma_i}{\sum_i \sigma_i^2} \quad (46)$$

The error in the rotation matrix $\hat{\mathbf{R}}$ with respect to its distance from the nearest orthonormal matrix, can be written simply as:

$$\varepsilon_{\hat{\mathbf{R}}} = \min_{\mathbf{R}, \alpha} \|\mathbf{R} - \alpha\hat{\mathbf{R}}\|_F^2 = 3 - \frac{(\mathfrak{h}\mathbf{\Lambda})^2}{\mathfrak{h}(\mathbf{\Lambda}^2)} \quad (47)$$

The focal length f can then be estimated by minimizing $\varepsilon_{\hat{\mathbf{R}}}$ with respect to f using (47):

$$f = \arg \min_f \{ \min_{\mathbf{R}, \alpha} \|\mathbf{R} - \alpha\hat{\mathbf{R}}(\mathbf{Q}, f)\|_F^2 \} \quad (48)$$

In the current implementation of the system the minimization over f is carried out using the *fminsearch* function in Matlab. Estimates can then be characterized by their error defined by (47) and curvature in the error function at the minimum (computed through a simple finite difference method). Two methods were compared for determining the best estimate of the focal length. One method was to use the estimate with the lowest error, The the other was to take the median focal length as in [2].

The results from the two methods were tested by using the different focal length estimates for the 3-Parameter rotation method described in the previous section on a sequence of image, and comparing the resulting error as defined by equation (47). Three sequences were used: a smoothly panning video sequence, the same sequence but using every 5th frame, and a carefully collected sequence with the camera panning about its optical center in large steps. As expected, it was found that the carefully collected sequence produced the best results. The median method performed slightly better than lowest error method with a large number of input images, and much worse with a small number. Figure 1 shows a histogram of the estimated focal lengths for a Sony DCR-TRV110 Handycam estimated from a 280 image smooth panning video sequence.

In all of the methods and sequences, the focal length estimates generally differed by no more than 15 percent. This amount of deviation produced a negligible difference in performance of the overall system, since the 3-parameter rotation method was used only for coarse alignment.

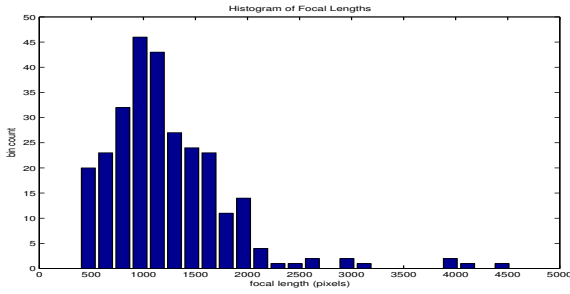


Fig. 1: Histogram of the estimated focal lengths for a Sony DCR-TRV110 Handycam estimated from a 280 image smooth panning video sequence. Here the median focal length is 1115 and the minimum error focal length 1385.

3. TONAL ALIGNMENT

While the geometric calibration of cameras is widely practiced and understood [3][4], often much less attention is given to the camera response function (how the camera responds to light). In digital cameras, the camera response function maps the actual quantity of light impinging on each element of the sensor array to the pixel values that the camera outputs.

Linearity (which is typically not exhibited by most camera response functions) implies the following two conditions:

1. Homogeneity: A function is said to exhibit homogeneity if and only if $f(ax) = af(x)$, for all scalar a .
2. Superposition: A function is said to exhibit superposition if and only if $f(x + y) = f(x) + f(y)$.

The two are often written together, as: $f(ax + by) = af(x) + bf(y)$.

In image processing, homogeneity arises when we compare differently exposed pictures of the same subject matter. Superposition arises when we superimpose (superpose) pictures taken from differently illuminated instances of the same subject matter, using a simple *law of composition* such as addition (i.e. using the property that light is additive).

A variety of techniques have been proposed to recover camera response functions, such as using charts of known reflectance, and using different exposures of the same subject matter [5][6][7][8]. The method proposed here differs from other methods in that it does not require the use of charts, nor a camera that is capable of adjusting its exposure. The method produces very accurate results and requires only that the camera has an exposure lock feature.

The comparagram, as defined in [5][9][6][10] has been widely used as a tool for the comparison of multiple differently exposed pictures of the same subject matter. With enough data, a direct nonparametric solution for the camera response function can be obtained, otherwise, a semi-parametric method such as Candocia's piecewise linear comparagram method will often provide better results[5]. A drawback of completely nonparametric methods is that periodicity in the amplitude domain, i.e. amplitude "ripples", also known as fractal ambiguity [11] and comperiodicity [12], plague the result unless more than two input images are used with exposure differences that are inharmonic (in the amplitude domain).

The method propose in this paper uses the notion of superposition rather than homogeneity to solve for the camera response function. In this method the linear constraint eliminates the fractal ambiguity. The following technique is used: in a dark envi-

ronment, set up two distinct light sources. Take three pictures, one with each light on individually (p_a, p_b), and one with the two lights on together (p_c). From this data we solve for the camera response function f by using the following constraints: For the i^{th} pixel position in each of the three images: $p_a[i] = f(q_a)$, $p_b[i] = f(q_b)$, and $p_c = f(q_a + q_b)$. Where the quantity q is known as the *photographic quantity* or *photoquantity* [5][10]. Note that the photoquantity is neither radiance, irradiance, luminance, nor illuminance, rather, it is a unit of light, unique to the spectral response of a particular camera.

3.1. The Camera Response Function

The camera response function f may in general be modeled by cascading two non-linear functions as shown in figure 2. In this diagram, the *photographic quantity* (*photoquantity*) of light measured by the sensor, is mapped into pixel space by a non-linear dynamic range compression function and a uniform quantizer. We call the first function the Range Compression Function because most camera response functions, such as the familiar gamma mapping, are convex.

In this method it is assumed only that this function is monotonic and convex[13]. The quantizer in turn maps the range compressed photoquantities into discrete pixel values.

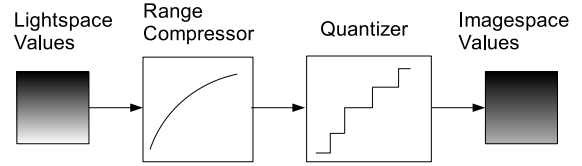


Fig. 2: The simple camera model used for the method. Lightspace values collected by the camera sensor elements are compressed with a non-linear function, and then quantized to yield pixel values.

By assumption the Range Compression Function is monotonic. Thus: photoquantities in the range $[q_1, q_2]$, $q_1 < q_2$ will result in some pixel value p_1 ; photoquantities in the range $[q_2, q_3]$, $q_2 < q_3$, will result in pixel value p_2 ; with $p_1 < p_2$. We then simplify our analysis by approximating the Range Compression Function as being linear between quantization points, and by assuming that the probability distribution of the measured photoquantities is uniform in this range. Therefore, given pixel value p_x , the maximum likelihood estimate of the original photoquantity \bar{q}_x is given by:

$$f^{-1}(p_x) = q_x = \frac{q_x + q_{x+1}}{2}, \quad (49)$$

where q_x and q_{x+1} are lightspace quantization points for pixel value p_x .

3.2. Solving for the Inverse Camera Response Function

Since the property of superposition holds with photoquantities, we can form the following equation:

$$f^{-1}(f(q_a)) + f^{-1}(f(q_b)) = \hat{q}_c. \quad (50)$$



Fig. 3: One of the image sets used. Leftmost: Picture with light A turned on. Middle: Picture with light B turned on. Rightmost: Picture with light A and B turned on together.

By equation (49), $\hat{q}_c = \bar{q}_a + \bar{q}_b$ and by our assumptions, \hat{q}_c is the maximum likelihood estimate of $q_a + q_b$ given our prior knowledge of $f(q_a)$ and $f(q_b)$. We can now form the superposition equation:

$$f^{-1}(f(\bar{q}_a)) + f^{-1}(f(\bar{q}_b)) = f^{-1}(f(\bar{q}_a + \bar{q}_b)) + \bar{\epsilon}_Q, \quad (51)$$

where $\bar{\epsilon}_Q$ is the mean error due to quantization of \hat{q}_c . Under the assumptions stated, we can solve for f^{-1} , i.e. the mapping from pixel value p_x to maximum likelihood (ML) photoquantities \bar{q}_x , by minimizing the following equation:

$$e = \sum_{\forall n} (f^{-1}(p_a[n]) + f^{-1}(p_b[n]) - f^{-1}(p_c[n]))^2, \quad (52)$$

where $p_a[n], p_b[n], p_c[n]$ are the n^{th} pixels of three images taken of a scene with constant exposure and three illumination permutations of two light sources in an otherwise dark environment. Pixel values p_a and p_b are from images of the scene with each of the two light sources turned on independently. Pixel value p_c is from the image of the scene with both light sources turned on together, as shown in Fig 3.

Since digital cameras output a finite range of discrete pixel values, care must be taken when applying the assumptions made at the ends of the camera's range where clipping occurs. In the remainder of the paper, we will assume that the camera outputs pixel values in the range $[0, 255]$ with clipping occurring at 0 and 255. This is not always the case, but the modification to the analysis under other conditions is very simple.

Using the proposed model, pixel values 0 and 255 are produced by the range of photoquantities $[0, q_1)$ and $[q_{255}, \infty)$ respectively. Since the range $[q_{255}, \infty)$ is infinite in size, the assumption that the distribution of photoquantities that produce a pixel value of 255 will approach a uniform distribution over a finite number of images will obviously not hold. A similar argument applies for pixel value 0. We therefore do not try to solve for $f^{-1}(0)$ or $f^{-1}(255)$, or equivalently, to solve for \bar{q}_0 and \bar{q}_{255} . Instead we can solve for the quantization points q_1 and q_{255} . This allows us to conclude that if we measure a pixel value of 0 with the camera, a quantity of light below q_1 was measured. Similarly, a pixel value of 255 represents a photoquantity greater than q_{255} . The method of solving for these thresholds is presented later in this section.

In accordance with this development, we define f^{-1} as the mapping from pixel values $(1, 2, 3...254)$ to the maximum likelihood photoquantities $(\bar{q}_1, \bar{q}_2, \bar{q}_3... \bar{q}_{254})$. We can now write equation 52 more simply as:

$$e = \sum_{\substack{\forall n, P_a, P_b \\ P_c \neq 0, 255}} (\bar{q}_{p_a[n]} + \bar{q}_{p_b[n]} - \bar{q}_{p_c[n]})^2 \quad (53)$$

Equation (5) can be efficiently minimized using a singular value decomposition (SVD). To do this, we represent f^{-1} as a

vector $\vec{f}^{-1} = [\bar{q}_1, \bar{q}_2, \bar{q}_3... \bar{q}_{254}]^T$ and we form a constraint matrix A such that the n^{th} row of the matrix corresponds to the n^{th} pixel in images p_a, p_b and p_c . Each row has a 1 in columns a and b , -1 in column c and zeros in all other columns. In the n^{th} row, a , b and c correspond to pixel values $p_a[n]$, $p_b[n]$ and $p_c[n]$ respectively. The least squares solution of the homogeneous equation: $A\vec{f}^{-1} = 0$ is then obtained by obtaining the SVD of $A = U\Sigma V^T$ and using the column of V corresponding to the smallest singular value in Σ .

Solving for f^{-1} by this method assumes that the error: $\epsilon = q_a + \bar{q}_b - \bar{q}_c$ has zero mean. Without noise, clipping at 255 can create a problem by biasing the distribution of the measured pixel values. With camera noise, this bias becomes very significant in pixel ranges near both clipping points: 0 and 255. Also, as with all least squares methods, outlier points can significantly perturb the solution.

With these considerations, the method is improved by robustly estimating $f(\bar{q}_c)$ by generating a histogram of the measured pixel values of c for each additive combination of a and b . By assuming that the normalized histogram is a reasonable approximation of the actual probability distribution of c , we can use the peak of this histogram \hat{c}_{a+b} as our best estimate of $f^{-1}(f(\bar{q}_a + \bar{q}_b))$. Our minimization problem thus becomes:

$$e = \sum_{\forall pairs \{x,y\}} N_{\{x,y\}} (\bar{q}_x + \bar{q}_y - \bar{q}_{\hat{c}_{x+y}})^2 \quad (54)$$

Where $N_{\{x,y\}}$ is the number of instances of $f^{-1}(a) + f^{-1}(b) = f^{-1}(c)$ in the dataset.

For a digital camera with 256 pixel levels, this collection of histograms can be expressed in a $256 \times 256 \times 256$ array, with the first two dimensions being the pixel values in image P_a and P_b respectively, and the third dimension containing the number of occurrences of each pixel value for each $\{a, b\}$ combination. This representation is effective since we can easily compile information from multiple image sets by simply adding the collection of histograms produced by each set, thereby increasing the accuracy of our estimate of $f(\bar{q}_c)$. In order to improve the estimate of the peak location, the histograms are each smoothed with a Gaussian kernel. This procedure is especially important when $\{a, b\}$ combinations are poorly represented.

To show that the method performs reliably, random synthetic lightspace data was generated. To this data, a C^1 continuous function was applied to the lightspace data, and the result quantized into 256 imagespace/pixel values. Following this procedure, Gaussian noise of standard deviation 10 was added to the pixel values. The singular value decomposition method was then used to recover the function using the described constraint matrix associated with equation (54). The results of this procedure on the synthetic data are shown in figure 4. As can be seen in the figure, the high quantity of noise added has significantly affected the recovered response functions, however it demonstrates the stability of the algorithm. When more reasonable noise levels are used, such as a standard deviation of 3, the results are very accurate.

In situations where only a small number of image sets are available, or the camera exhibits a large amount of noise, it is necessary to add smoothness term to the objective function (54) to achieve reliable results:

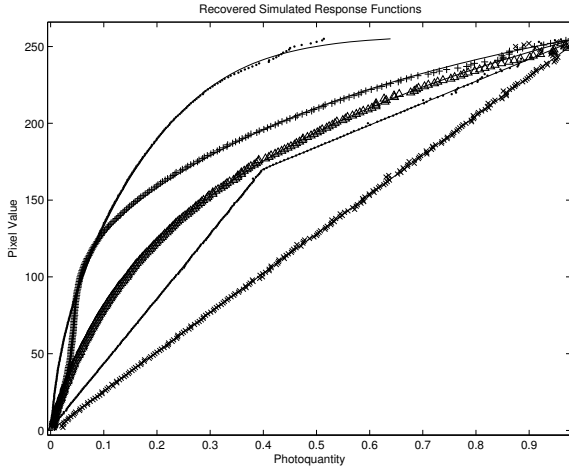


Fig. 4: Plots of five different recovered response functions, together with ground truth, using synthetically generated data. The proposed algorithm was then used to recover 256 discrete points (plotted as various shaped points). Note that the recovered points fall virtually on top of the original functions (plotted as solid lines).

$$e = \sum_{\forall \text{ pairs } \{x,y\}} N_{\{x,y\}} (\bar{q}_x + \bar{q}_y - \bar{q}_{\bar{c}_{x+y}})^2 + \lambda \sum_{n=3:252} \ddot{f}^{-1}(n)^2 \quad (55)$$

where λ weights the smoothness term and $\ddot{f}^{-1}(n)$ is computed by:

$$\ddot{f}^{-1}(n) = -\ddot{f}^{-1}[n-2] + 16\ddot{f}^{-1}[n-1] - 30\ddot{f}^{-1}[n] + 16\ddot{f}^{-1}[n+1] - \ddot{f}^{-1}[n+2] \quad (56)$$

It was found that for high quality digital cameras (such as a Nikon D1) the response function could accurately determined without the smoothness term. However, it was necessary to find the response function for the Sony DCR-TRV110 Handycam. The results of applying the smoothness term can be seen in figure 5.

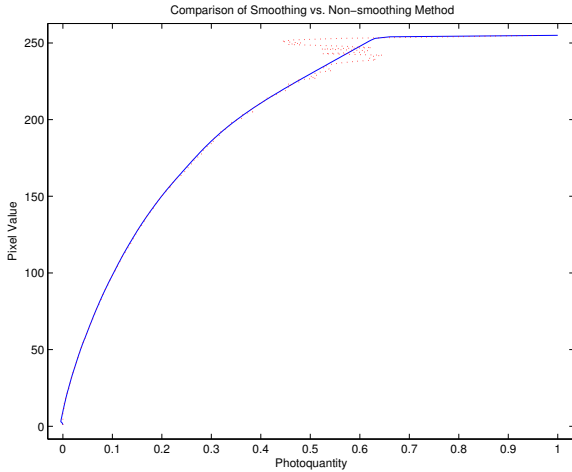


Fig. 5: A comparison of the camera response functions solved using the method presented, with and without the smoothness term. The dotted line represents the solution without the smoothness constraint.

Once a good estimate for \ddot{f}^{-1} has been determined, we can solve for the quantization point q_1 by examining elements in the collection of histograms that involve pixel value 0. We then apply the inverse response function to the histogram and look for a step in the distribution to infer the value of q_1 . The same method is then used to determine q_{255} . In practice, q_1 has little value since at this low level, the noise inherent in the imaging system dominates. In most cases, it is reasonable to simple assign $f^{-1}(0) = 0$ and $f^{-1}(255) = q_{255}$.

To test the accuracy of the recovered camera response functions, a similar set of image triples are used: $p_a = f(q_a)$, $p_b = f(q_b)$ and $p_c = f(q_{a+b})$. To make the test fair, we use a different set of images then those used to generate the response function. The inverse response function is applied to p_a and p_b and the resulting photoquantities q_a and q_b are added. We now compare this sum (in either imagespace or lightspace) with p_c (or q_c). The resulting mean squared difference is then used to rate the response function *superposition error*. This method can then be repeated for sets of images with different lighting configurations to test the full camera range.

3.3. Estimating the Exposure Difference Between Images of Identical Subject Matter

If we have two images of identical subject matter varying only exposure, we can use the inverse camera response function to allow us to estimate the exposure difference.

Assuming the system is perfect and the lighting doesn't change between exposures, for a given pixel location \mathbf{x} : image A has pixel value $P_A(\mathbf{x}) = f(q(\mathbf{x}))$ and image B has pixel value $P_B(\mathbf{x}) = f(\kappa q(\mathbf{x}))$. Where κ represents the multiplicative change in exposure. For example, using a shutter speed of 1/250 s in image A, and a shutter speed of 1/500 s in image B would yield $\kappa = 2$. To solve for the κ we use the inverse response function \ddot{f}^{-1} and minimize the following squared error objective function across the entire image:

$$\kappa = \arg \min_{\kappa} \sum_{\forall \mathbf{x}} (\ddot{f}^{-1}[P_A(\mathbf{x})] - \kappa \ddot{f}^{-1}[P_B(\mathbf{x})])^2 \quad (57)$$

Which is easily solved by:

$$\kappa = \frac{\sum \ddot{f}^{-1}[P_A(\mathbf{x})] \ddot{f}^{-1}[P_B(\mathbf{x})]}{\sum (\ddot{f}^{-1}[P_B(\mathbf{x})])^2} \quad (58)$$

This solution assumes that we have equal certainty in the values of $\ddot{f}^{-1}[n]$ for all values of n . This is of course not true for several reasons:

1. The variation in the slope of the inverse response function causes the quantization noise in the estimated photoquantities to vary.
2. The recovered inverse response function can vary in accuracy for different pixel values.
3. The noise characteristic of the camera varies for different pixel values.

To account for these effects, a certainty function $c[n] \in [eps, 1]$ was created that describes our confidence in the photoquantities estimated for each pixel value n . We expect $c[0]$ and $c[255]$ to be

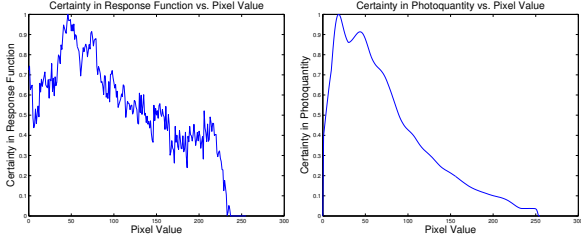


Fig. 6: On the Left we have a plot of $c_f[n]$ for the Sony Handycam. On the right we have a plot of the combined certainty function $c[n] = c_f[n]c_Q[n]$, representing our confidence in the photoquantity obtained using the inverse camera response function.

very small due to clipping. We include this function into our estimate of κ by:

$$\kappa = \underset{\kappa}{\operatorname{arg\,min}} \sum_{\forall \mathbf{x}} c[P_A(\mathbf{x})]c[P_B(\mathbf{x})] \{(\tilde{f}^{-1}[P_A(\mathbf{x})] - \kappa \tilde{f}^{-1}[P_B(\mathbf{x})])^2\} \quad (59)$$

$$\kappa = \frac{\sum c[P_A(\mathbf{x})]c[P_B(\mathbf{x})]\tilde{f}^{-1}[P_A(\mathbf{x})]\tilde{f}^{-1}[P_B(\mathbf{x})]}{\sum c[P_A(\mathbf{x})]c[P_B(\mathbf{x})]\tilde{f}^{-1}[P_B(\mathbf{x})]^2} \quad (60)$$

$c[n]$ is determined in two steps: first the certainty in the response function $c_f[n]$ is estimated experimentally, then the certainty due to quantization $c_Q[n]$ is estimated from the slope of the response function. The combined certainty is then computed as: $c[n] = c_f[n]c_Q[n]$.

$c_f[n]$ is determined during the testing of the recovered inverse response function as described at the end of last section. For each common pixel in each triple of images: $p_a = f(q_a)$, $p_b = f(q_b)$ and $p_c = f(q_a + q_b)$, the inverse response function is applied to p_a and p_b and the resulting photoquantities \bar{q}_a and \bar{q}_b are added. We then compare $f(\bar{q}_a + \bar{q}_b)$ with p_c . The resulting mean squared difference is then normalized by the number of pixels of value p_c in image C, and then are accumulated in an error histogram $e_{hist}[n]$ at position p_c . This process is applied to all the testing images, and the resulting error histogram is normalized so that its range is $[0, 1)$. $c_f[n]$ is then computed as:

$$c_f[n] = 1 - e_f[n] \quad (61)$$

For the results of this process applied to the Sony Handycam, see figure 6. We expect the certainty function to be smooth, however, since we are creating it with a finite amount of testing data, we usually have to smooth the result.

The certainty due to quantization is calculated by taking the derivative of the response function:

$$c_Q[n] = \frac{df(\tilde{f}^{-1}[n])}{dq} \quad (62)$$

In practice f is approximated by inverting \tilde{f}^{-1} . The derivative is approximated using a finite difference method. c_Q is also normalized so that its range is limited to $(0, 1]$.

$c[n]$ is then obtained by composing c_Q and c_f . The resulting certainty function for the Sony Handycam can be seen in figure 6.

4. SIMULTANEOUS SPATIAL AND TONAL ALIGNMENT

In order register images from camera that varies its exposure (whether through AGC or through manual adjustment), it is necessary to simultaneously estimate the motion parameters \mathbf{m} and the exposure parameter κ . The method presented is for the full 8-parameter projective model, since this is what is used in the current implementation, however the results are easily extended to other models. The development is very similar to that presented in section 2.1, so only the major points are covered.

Input images are first linearized (i.e. converted to photoquantities) using the inverse response function:

$$\bar{\mathbf{I}}_0(\mathbf{x}) = \tilde{f}^{-1}[\mathbf{I}_0(\mathbf{x})] \quad (63)$$

$$\bar{\mathbf{I}}_1(\mathbf{x}) = \tilde{f}^{-1}[\mathbf{I}_1(\mathbf{x})] \quad (64)$$

Using certainty function $c[n]$ as an element-wise matrix operator we can also define certainty images to simplify our notation:

$$\mathbf{C}_0(\mathbf{x}) = c[\mathbf{I}_0(\mathbf{x})] \quad (65)$$

$$\mathbf{C}_1(\mathbf{x}) = c[\mathbf{I}_1(\mathbf{x})] \quad (66)$$

Now, if we add the exposure parameter κ and certainty to equation (5), we get:

$$\varepsilon_{\mathbf{m}} = \sum_{\forall \mathbf{x}} \mathbf{C}_0(\mathbf{x})\mathbf{C}_1(\mathbf{x}) [\bar{\mathbf{I}}_1(f(\mathbf{x}, \mathbf{m})) - \kappa \bar{\mathbf{I}}_0(\mathbf{x})]^2 \quad (67)$$

Equation (11) now becomes:

$$\varepsilon_{\mathbf{m}} \approx \sum_{\forall \mathbf{x}} \mathbf{C}_0(\mathbf{x})\mathbf{C}_1(\mathbf{x}) [\bar{\mathbf{I}}_1(\mathbf{x}) + \nabla \bar{\mathbf{I}}_1(\mathbf{x})\mathbf{J}(\mathbf{x})\mathbf{d} - \bar{\mathbf{I}}_0(\mathbf{x})]^2 \quad (68)$$

With the least squares solution:

$$\bar{\mathbf{A}} \begin{bmatrix} \mathbf{d} \\ \kappa \end{bmatrix} = -\bar{\mathbf{b}} \quad (69)$$

where

$$\bar{\mathbf{A}} = \sum_{\forall \mathbf{x}} \mathbf{C}_0(\mathbf{x})\mathbf{C}_1(\mathbf{x}) \begin{bmatrix} \mathbf{J}(\mathbf{x})\nabla \bar{\mathbf{I}}_1(\mathbf{x}) \\ \bar{\mathbf{I}}_0(\mathbf{x}) \\ [\nabla \bar{\mathbf{I}}_1(\mathbf{x})^T \mathbf{J}(\mathbf{x})^T \quad \bar{\mathbf{I}}_0(\mathbf{x})] \end{bmatrix} \quad (70)$$

and

$$\bar{\mathbf{b}} = \sum_{\forall \mathbf{x}} \mathbf{C}_0(\mathbf{x})\mathbf{C}_1(\mathbf{x})\bar{\mathbf{I}}_1(\mathbf{x}) \begin{bmatrix} \mathbf{J}(\mathbf{x})\nabla \bar{\mathbf{I}}_1(\mathbf{x}) \\ \bar{\mathbf{I}}_0(\mathbf{x}) \end{bmatrix} \quad (71)$$

The motion parameters \mathbf{m} can now be iteratively estimated using the steps outlined in section 2.1, with the simple difference of warping both the photoquantity image $\bar{\mathbf{I}}_0(\mathbf{x})$ and the certainty image $\mathbf{C}_0(\mathbf{x})$ at step (ii), and recomputing $\bar{\mathbf{A}}$ and $\bar{\mathbf{b}}$ using the warped images to estimate the motion improvement \mathbf{m}' .

5. CREATING HIGH DYNAMIC RANGE MANIFOLD MOSAICS

The system was implemented using a Sony DCR-TRV110 Digital Handycam. The response function of the camera was solved using the method presented in this paper. The results of the can be seen

in figure 5. The focal length of the camera was estimated from a sequence of images spatially registered using the 8-parameter projective method. For details see section 2.3.

Pairwise estimates from a video sequence are estimated in a two step process. First the multiscale 3-parameter rotation method is used to create an initial estimate of the interframe transformation (section 2.2). Then the estimate is refined by using the combined spatial and tonal alignment technique presented in section 4.

It was found that this procedure produced the quickest and most stable transformation estimates. While not described in this paper, an affine motion model was also tested for roughly estimating the transformation. It was however, drastically outperformed by the rotation method for all situations except for pure translatory camera motion and zoom. It was also found that the performance in the rotation method was rather insensitive to moderate errors in the focal length. Note that when a very large focal length is used in the model, the resulting projective transformations degenerate to similarity transformations, which also perform very well for a rough motion estimate in many cases. Thus, when the focal length is not known, an overestimate can be used in this two step process, to aid in finding the true focal length.

Since the rotation model is only used for coarse estimates, the input images are blurred with a gaussian kernel with $\sigma = 2$, whereas for the final refinements $\sigma = 1$ is used. See figures 7 and 8 to see the method in action. Figure 7 shows the input images. Figure 8 shows the estimate from the rotation method (note the greater blurring), and the refinement from the joint spatial/tonal alignment.



Fig. 7: Images to be spatially and tonally aligned

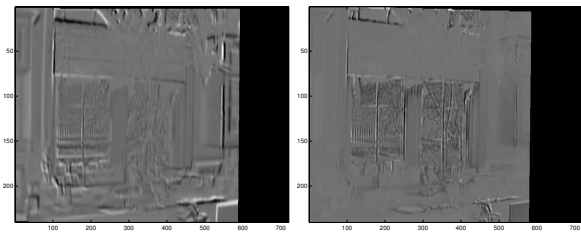


Fig. 8: Results (in the form of difference images) from the two stage alignment technique. On the left: the results from the rotation model estimate. On the right: the refinement from the spatial/tonal projective method.

The system implemented produces mosaics from a sequence of input images using a similar method to the *Rectified Mosaicing With Asymmetrical Strips* presented by Peleg et. al. [1]. It was found that both the *Asymmetric* and *Symmetric* strip algorithms created visually undesirable results for images with very little motion between them.² This system uses a simplified approach that

²It is possible that a imperfection in my implementation caused the poor

produces very good results even with very large camera motions. The technique uses rectangular vertical strips with a width determined by the horizontal offset of the origin between time adjacent images. More precisely, the vertical strip used from image I_k begins at the centre line of the image and ends at the horizontal position of the origin of image I_{k+1} projected into I_k (using the projective transformation relating the spatial coordinates relating the two images). This strip is then cemented into the mosaic using the vertical offset determined by the vertical position of the origin of I_k projected into I_{k-1} (using the transformation relating I_k and I_{k-1}).

Before cementing the image strips into the final mosaic, they are first converted to photoquantities using the inverse camera response function. The resulting panorama thus tonally aligned and is linear in light. Using this method, these panoramas can have very high dynamic range (depending on the input sequence used) and must be viewed accordingly. A simple method to see everything is to take the square root of the light quantities in the panorama and scale the result to $[0, 255]$ for displaying. Figure 9 displays the results of this method for a kitchen scene with a window. Another method is to use the camera response function to produce a picture taken by a virtual camera. To do this we multiply the panoramic photoquantities by a scaling factor to achieve the desired exposure, and then use the camera response function to generate a picture as the camera would have seen it. We can see the results using this display method, for the same kitchen scene in figures 10, 11 and 12.

To illustrate the robustness of the motion estimation method and the effect of spatial/tonal alignment, the same kitchen scene was processed using every 15th video frame. Figure 13 shows the spatially and tonally registered mosaic and figure 14, shows the same composite without the tonal alignment.

6. ACKNOWLEDGEMENTS

Thanks to Corey Manders and Steve Mann who worked with me to develop the superposition based method for recovering the camera response function.

7. REFERENCES

- [1] Rav-Acha A Zomet A Peleg S, Rousso B, "Mosaicing on adaptive manifolds," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22(10), October 2000.
- [2] Szeliski R Shum H-Y, "Creating full view panoramic image mosaics and environment maps," *Computer Graphics Proceedings, Annual Conference Series*, pp. 251–258, August 1997.
- [3] E. Trucco and A. Verri, *Introductory Techniques for 3-D Computer Vision*, Prentice Hall, NJ, 1998.
- [4] O. Faugeras, *Three-Dimensional Computer Vision - A Geometric Viewpoint*, M.I.T. Press, Cambridge, MA, 1993.
- [5] F. M. Candocia, "A least squares approach for the joint domain and range registration of images," *IEEE ICASSP*, vol. IV, pp. 3237–3240, May 13-17 2002, avail. at <http://iul.eng.fiu.edu/candocia/Publications/Publications.htm>.
- [6] A. Barros and F. M. Candocia, "Image registration in range using a constrained piecewise linear model," *IEEE ICASSP*, vol. IV, pp. 3345–3348, May 13-17 2002, avail. at <http://iul.eng.fiu.edu/candocia/Publications/Publications.htm>.

results, however, both methods, produced very slight scaling differences that caused the resulting panoramas to look rippled.



Fig. 9: Spatially and tonally aligned strip mosaic for a high dynamic range kitchen sequence. This greyscale image displays the square root of the photoquantity panorama



Fig. 10: Kitchen sequence displayed using a virtual camera using 1/4th the exposure of the leftmost image strip.



Fig. 11: Kitchen sequence displayed using a virtual camera using 1/16th the exposure of the leftmost image strip.



Fig. 12: Kitchen sequence displayed using a virtual camera using $1/32$ nd the exposure of the leftmost image strip.



Fig. 13: Kitchen sequence mosaic created from every 15th image. Displayed by a virtual camera at $1/20$ th exposure.



Fig. 14: Kitchen sequence mosaic created without tonally adjusting the strips.



Fig. 15: Every 30th frame of the kitchen sequence

- [7] S. Mann and R. Mann, "Quantigraphic imaging: Estimating the camera response and exposures from differently exposed images," *CVPR*, pp. 842–849, December 11–13 2001.
- [8] Martin Bichsel and Krystyna W. Ohnesorge, "How to measure a camera's response curve from scratch," Tech. Rep. ifi-93.19, 1, 1993.
- [9] F. M. Candocia, "Synthesizing a panoramic scene with a common exposure via the simultaneous registration of images," *FCRAR*, May 23–24 2002, avail. at <http://iul.eng.fiu.edu/candocia/Publications/Publications.htm>.
- [10] S. Mann, "Comparametric equations with practical applications in quantigraphic image processing," *IEEE Trans. Image Proc.*, vol. 9, no. 8, pp. 1389–1406, August 2000, ISSN 1057-7149.
- [11] Michael D. Grossberg and Shree K. Nayar, "What can be known about the radiometric response function from images?," *Proc. of European Conference on Computer Vision (ECCV) Copenhagen, May, 2002*.
- [12] C. Manders S. Mann and J. Fung, "Painting with looks: Photographic images from video using quantimetric processing," *ACM Multimedia 2002*, pp. 117–126, 2002.
- [13] Michael D. Grossberg and Shree K. Nayar, "What is the space of camera response functions," *In Proc. IEEE Computer Vision and Pattern Recognition (CVPR), Wisconsin, June, 2003*.