Reverse-Projection Method for Measuring Camera MTF

Stan Birchfield, Microsoft Corporation, Redmond, Washington, USA

Abstract

This paper proposes several adjustments to the ISO 12233 slanted edge algorithm for estimating camera MTF. First, the Ridler-Calvard binary image segmentation method is used to find the line. Secondly, total least squares, rather than ordinary least squares, is used to compute the line parameters. Finally, the pixel values are projected in the reverse direction from the 1D array to the 2D image, rather than from the 2D image to the 1D array. Together, these changes yield an algorithm that exhibits significantly less variation than existing techniques when applied to real images. In particular, the proposed algorithm is largely invariant to the rotation angle of the edge as well as to the size of the image crop.

Introduction

The modulation transfer function (MTF) is a widely used measure for the sharpness of an imaging system. High values of the MTF indicate a system that generates sharp images, whereas low values of the MTF indicate a system that generates blurry images. In digital imaging systems, the MTF is usually estimated by the spatial frequency response (SFR), which is the normalized modulus of the Fourier transform of a line spread function (LSF). For the purpose of this paper, the terms MTF and SFR will be used interchangeably.[1]

A standard technique for measuring MTF is known as the "slanted edge" approach, for which the algorithm was standardized with the publication of ISO 12233 more than a decade ago. Since then, numerous open-source and proprietary implementations of the algorithm have appeared, which are widely used to measure the sharpness of digital cameras. Such algorithms are used, for example, to declare a camera "Skype certified," which requires as one of its tests that the MTF30 value (the frequency at which the MTF reaches 30% of its zero-frequency value) exceed a certain threshold.

Calculating MTF

The slanted-edge algorithm standardized in ISO 12233 involves the following ten steps [4]:

1. Determine a region of interest (ROI) in the image containing a single step edge. (This step, unlike the others, can be performed either manually or automatically.)
2. Linearize the pixel data by undoing the opto-electronic conversion function (OECF), also known as gamma compression. (Undoing gamma compression is called gamma expansion.)
3. In the case of a color camera, calculate a weighted sum of the red, green, and blue values to yield a luminance value for each pixel; alternatively, perform the remaining steps separately for each color channel. (Skip this step in the case of a monochrome camera.)
4. Find the coordinates of points along the intensity step edge.
5. Fit the parameters of a line to the coordinates.
6. Project the 2D array of pixel values onto a 1D array known as the edge spread function (ESF).
7. Differentiate the ESF by convolving with an FIR filter to yield the line spread function (LSF).
8. Apply a Hamming window function to reduce the effects of noise far from the edge.
9. Compute the discrete Fourier transform (DFT) of the LSF.
10. The magnitude of the DFT yields an estimate of the MTF.

In this paper, a new method for calculating MTF is proposed. This method follows these same ten steps but differs in how several of the steps are implemented. More specifically, the proposed technique is identical to the standard technique regarding Steps 1–3 and 7–10. The middle steps, however, differ as follows:

- **Step 4, Find coordinates of points.**
  Standard technique: Compute the centroid of the derivative along the rows of the image.
  Proposed technique: Apply the Ridler-Calvard binary image segmentation algorithm [1] to the pixels in the ROI to segment the image into the light and dark regions. This yields the coordinates of points that straddle the light and dark regions.
- **Step 5, Fit line parameters.**
  Standard technique: Calculate the slope of the line using ordinary least squares.
  Proposed technique: Perform total least squares to calculate the line that best fits the coordinates.
- **Step 6, Project from 2D to 1D.**
  Standard technique: Forward project the 2D pixel values onto the 1D oversampled array.
  Proposed technique: For each cell in a 1D oversampled array perpendicular to the line, calculate the average linearized pixel value by reverse projecting into the interpolated 2D array.

In contrast to the standard technique, which is sensitive to both the slant of the edge and the number of pixels in the ROI, the proposed technique is largely insensitive to both the edge orientation and the number of pixels in the ROI. It also requires simpler calculations, since there is no possibility of empty bins, and no need to use a certain number of image rows (see below). Of the three changes listed above, the last one (Step 6) is the most crucial, as it allows more reliable construction of the 1D array from the 2D array by eliminating the high-frequency noise introduced

[1] If the input edge cannot be considered an ideal edge, then the SFR must first be divided—frequency by frequency—by the corresponding input edge modulation to produce the MTF [5, 7]; this detail, however, is generally not as important with cameras as it is with scanners.

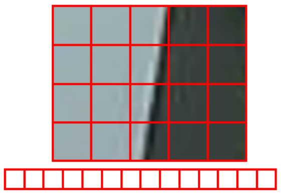

Figure 1. The goal of projection (Step 6) is to construct a 1D array (bottom) from a 2D array of pixels (top). (Note that the overlaid squares are shown here large for illustrative purposes; in reality each is the size of a single pixel.)

by forward projection. This results in a more stable and reliable estimate of the MTF.

For completeness, it should be mentioned that the ISO 12233 standard imposes additional details on the performing of several of the ten steps above: 1) the edge must be approximately vertical (as opposed to approximately horizontal—if not, the image is first rotated by 90 degrees); 2) The edge must be slanted, typically about 5 degrees; 3) the coordinates in Step 4 are found by computing the centroid of the derivative of each row, using a convolution kernel of $\begin{bmatrix}0.5 - 0.5\end{bmatrix}$; 4) in Step 5, ordinary least squares is used to calculate the line parameters;[2] 5) in Step 6, the number of rows used is truncated to ensure that each phase (of the edge position relative to the horizontal center of the pixel) occurs in the same number of rows; 6) in Step 6, the pixel values of each row are shifted according to the line parameters and placed into an oversampled LSF using four bins per pixel (4X oversampling).

The proposed method differs regarding these six details: 1) the edge can be any orientation; 2) Although it is best for the edge to be slanted, it is not strictly speaking necessary; 3) the coordinates in Step 4 are found using binary image segmentation; 4) in Step 5, total least squares is used instead of ordinary least squares; 5) truncation is no longer needed; 6) in Step 6, the oversampled LSF is created by reverse projection from the 1D array to the 2D array, as opposed to forward projection from the 2D array to the 1D array. However, in this last step, 4X oversampling is still used.

Forward Versus Reverse Projection

To better understand the difference between forward and reverse projection in Step 6 of the procedure, consider the slanted edge shown in Figure 1. The superimposed grid represents the pixels in the 2D image, while the array below the image represents the 1D oversampled signal obtained by projecting the pixels. To avoid clutter, this particular illustration shows 2X oversampling (which can be seen from the fact that there are two samples in the 1D array for every column in the image), rather than the more common 4X oversampling used in practice. The goal in Step 6 is to produce this 1D array from the 2D array.

The two approaches to projection are illustrated in Figure 2. Forward projection loops through the pixels in the 2D image and, for each pixel, updates the value in the corresponding cell in the 1D array by adding the value of the pixel. A separate 1D array keeps track of how many pixels have contributed to each cell, so that, when all pixels have been considered, the average value of each cell can be computed. In forward projection, the 1D array is aligned with the rows of the image. In contrast, reverse projection loops through the cells in the 1D array and, for each cell, computes its value as the average of the interpolated values at equally-spaced locations in the 2D image along a line perpendicular to the 1D array, which itself is oriented perpendicular to the intensity edge. Bicubic interpolation is used in this paper, although bilinear interpolation is an alternative.

An example of the two approaches applied to an actual slanted edge is presented in Figure 3. Both implementations are identical except for the projection method. The steps are as follows: gamma expansion using the Rec. 709 nonlinear transfer function, Ridler-Calvard binary image segmentation, total least squares to compute the line parameters, projection (either forward or reverse) with 4X oversampling, differentiation using a convolution kernel of $\begin{bmatrix}1 & -1\end{bmatrix}$, Discrete Fourier Transform (DFT), and retaining only the magnitude of the DFT of the positive frequencies to yield the MTF. Forward projection truncates the number of rows (in this case, to 56) to ensure an integral number of phase shifts, whereas reverse projection uses all the rows (in this case, 62). Note that the MTF30 (the frequency that yields the MTF value of 0.3) is significantly different between the two approaches: 0.71 cycles per pixel for forward projection, versus 0.53 cycles per pixel for reverse projection. Similarly, the MTF50 (the frequency that yields the MTF value of 0.5) is different: 0.56 versus 0.47.

The reason the MTF values are higher for forward projection than they are for reverse projection appears to be that the former introduces extraneous high frequencies into the computation. To see this phenomenon, Figure 4 shows zoomed-in portions of the projections and windowed derivatives, where noise with a frequency of 0.25 cycles per sample[3] (or, equivalently, a period of 4 samples) is evident. The cause of this noise is the inherent periodicity in the 4X oversampling procedure of forward projection. Indeed, when 2X oversampling is used instead, the noise from forward projection has a frequency of 0.5 cycles per sample (period of 2 samples), as shown in Figure 5.

The MTF30 and MTF50 values for this edge for different amounts of oversampling are as follows:

	forward		reverse	
	MTF30	MTF50	MTF30	MTF50
1X	0.46	0.42	0.45	0.42
2X	0.59	0.53	0.51	0.46
4X	0.71	0.56	0.53	0.47
8X	0.69	0.57	0.54	0.48

From this table, note that reverse projection yields fairly stable values for any amount of oversampling. In contrast, the noise in forward projection increases as the amount of oversampling increases, which in turns raises the MTF curve. The lowest noise level is achieved with 1X oversampling, but the benefits of oversampling are lost as well. With 8X oversampling, the values from forward projection are particularly unreliable since some of the bins in the 1D array have zero values.

The high-frequency noise appears to be the result of periodicity in the mapping from the 2D pixel coordinates to the 1D array

[2]While not specifically stated by the standard, the reference implementation listed in the appendix of the standard does use ordinary least squares. To the author's knowledge, every existing implementation follows this precedent.

[3]"Sample" here refers to an element in the 1D array.

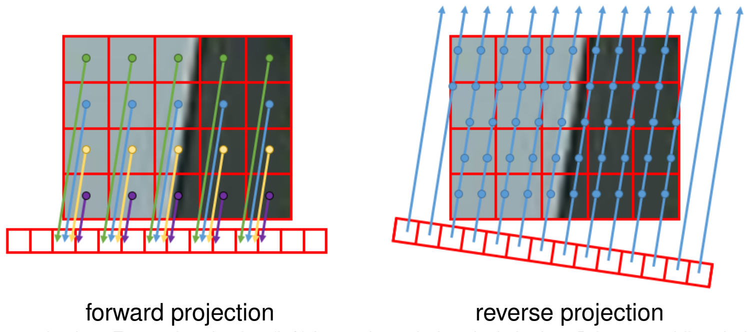

Figure 2. *Forward versus reverse projection. Forward projection (left) loops through the pixels in the 2D image, adding their values to the corresponding cells in the 1D array in order to average them. Reverse projection (right) loops through the cells in the 1D array, computing an average of interpolated values at equally-spaced locations in the 2D image.*

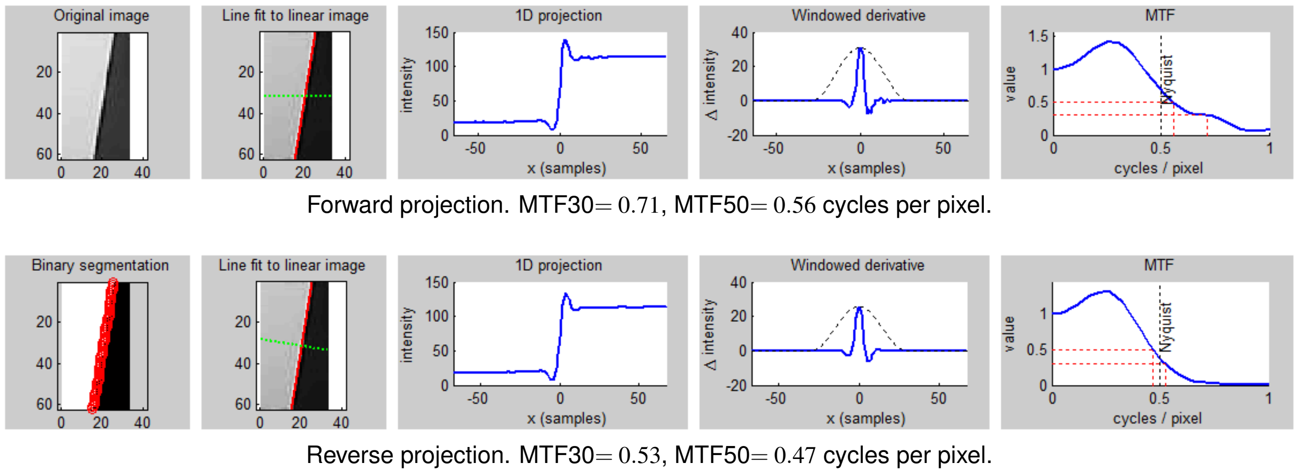

Figure 3. *Forward projection (top) compared with reverse projection (bottom) with 4X oversampling on an actual* 33×62 *JPEG-compressed slanted edge. From left to right: Original image and result of Ridler-Calvard segmentation with boundary points overlaid (red circles), image after gamma expansion with fitted line overlaid (solid red) and projection line (dashed green), 1D projection (either forward or reverse), windowed derivative (solid) with Hamming window (dashed), and MTF curve with MTF30 and MTF50 (dashed red lines).*

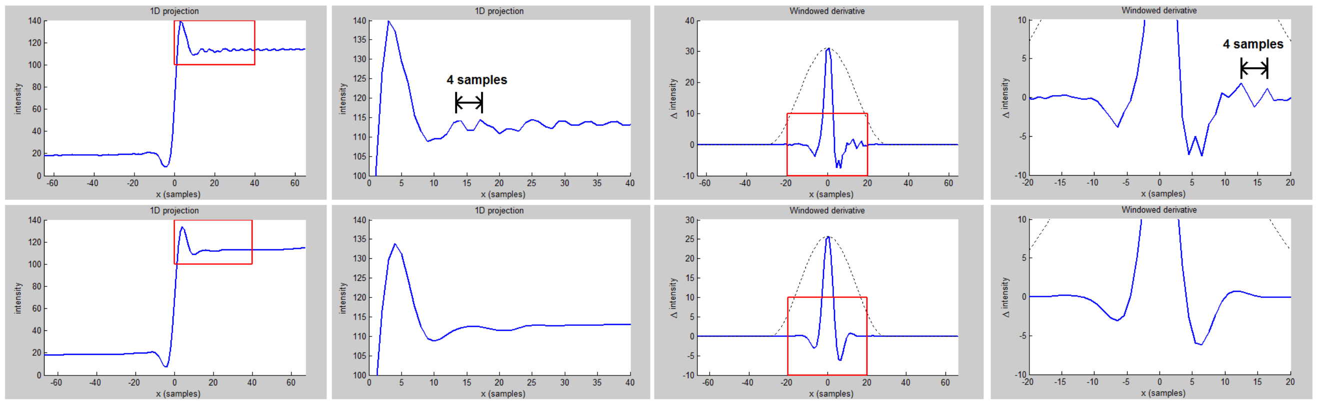

Figure 4. *Zoomed-in plots of the 1D projection (left two columns) and the windowed derivative (right two columns) of the previous figure for both forward (top) and reverse (bottom) projections. Notice the high-frequency noise present in the former signals that is absent from the latter signals. This noise has a frequency of 0.25 cycles per sample, or equivalently, a period of 4 samples. This high-frequency noise artificially inflates the MTF curve computed by forward projection.*

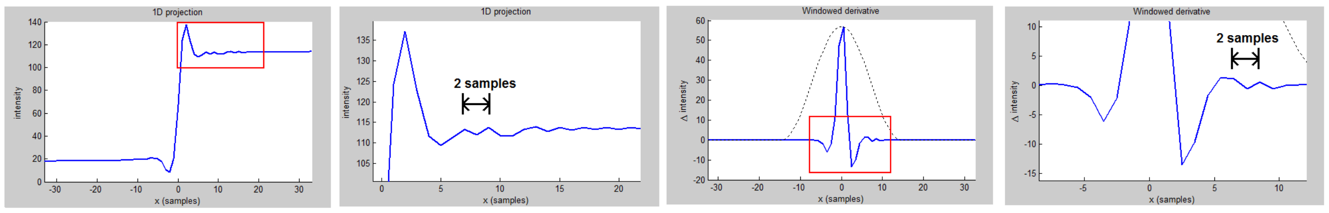

Figure 5. *Zoomed-in plots of forward projection using 2X oversampling. The noise from forward projection has a frequency of 0.5 cycles per sample, or a period of 2 samples.*

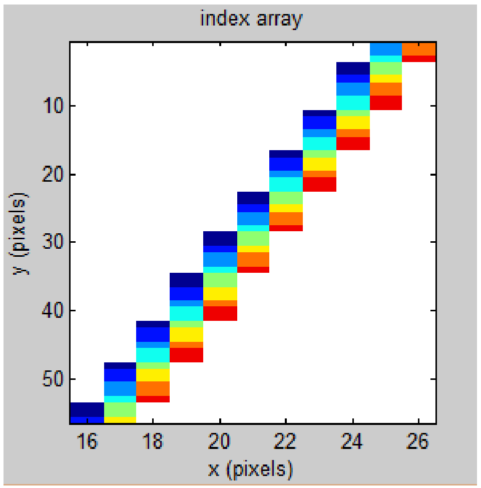

Figure 6. *Depiction showing which pixels in the 2D image contribute to which samples in the 1D array for 8 samples near the center of the 1D array (4X oversampling). Each color represents a different sample index. The repetition between indices spaced 4 samples apart is evident: Navy blue (1st sample) always appears on the same row as green (5th sample), and so on. See text for details.*

indices. Figure 6 shows the pixels associated with eight consecutive samples near the center of the 1D array, using 4X oversampling. Each sample is assigned a color, and the pixels that contribute to that sample are displayed with that color. (White pixels do not contribute to these eight samples.) Note that the 1st (navy blue) and 5th (green) samples receive contributions from the same rows, and similarly for the 2nd (bright blue) and 6th (yellow) samples, as well as the 3rd (royal blue) and 7th (orange) samples, and the 4th (teal) and 8th (red) samples. In other words, as one traverses the 1D array, one discovers that the contributing rows are repeated every 4th sample, which corresponds to the 4-sample period of the high-frequency noise. Similarly, with 2X oversampling, the rows are repeated every 2nd sample. These repetitions appear to resonate with the periodicity of the JPEG compression to introduce the high-frequency noise. As an aside, note that the edge in these experiments is at an angle of 9.1°, so the slope is 6.2, which can be seen in the figure as the step size between the multicolored foreground and white background.

Implementation Details

Now that we have analyzed the characteristics of forward and reverse projections, in this section we describe several other details of the implementation. Step 4 applies the Ridler-Calvard algorithm to segment the pixels in the ROI into two categories (light and dark). Ridler-Calvard is a classic binary segmentation algorithm that is easy to implement. As a type of k-means segmentation, the algorithm alternates between two steps: first, all the pixels are assigned to one of the two categories depending upon whether they are above or below the threshold, then, a new threshold is computed as the average of the mean value of the light pixels and the mean value of the dark pixels. The threshold is initialized with the average value of all the pixels, and the algorithm usually converges in just a few iterations. Note that this approach makes no assumption on the orientation of the edge, unlike the traditional approach that scans the image one row at a time.

Once the image has been segmented, the edge is found by creating a point between each pair of adjacent pixels with different classifications (one light and one dark, according to the threshold found by Ridler-Calvard). Then, given a set of n such points $\{(x_i, y_i)\}_{i=1}^n$, Step 5 uses the method of total least squares to find the parameters of the line, which is represented using homogeneous coordinates as $ax + by + c = 0$ (which again has the advantage of representing lines at any orientation, unlike $y = mx + b$). First, the covariance matrix is constructed:

$$C = \frac{1}{n}\sum_{i=1}^{n}\begin{bmatrix} (x_i - \bar{x})^2 & (x_i - \bar{x})(y_i - \bar{y}) \\ (x_i - \bar{x})(y_i - \bar{y}) & (y_i - \bar{y})^2 \end{bmatrix}, \tag{1}$$

where $(\bar{x}, \bar{y})$ is the centroid of the points. Then the eigenvector associated with the smallest eigenvalue of C yields the normal (a, b) to the line, and the signed distance to the origin is given by $c = -a\bar{x} - b\bar{y}$. The advantage of total least squares is that it minimizes the perpendicular distances from the points to the line, as opposed to the vertical distances (as in ordinary least squares), thus again making the computation insensitive to the orientation of the edge.

Another point to be mentioned is that the ESF should be obtained by convolving the LSF with the convolution kernel $\begin{bmatrix}1 & -1\end{bmatrix}$, which is the so-called forward (or backward) difference kernel.[4] In contrast, some existing implementations use the central difference kernel, $\begin{bmatrix}0.5 & 0 & -0.5\end{bmatrix}$. It is important to recognize that the latter will always yield a lower MTF curve than the former, because it introduces additional smoothing into the computation [5]. That is, convolving the 1D projected signal with

[4]Note that the kernel (FIR filter) $\begin{bmatrix}-0.5 & 0.5\end{bmatrix}$ mentioned in ISO 12233 [4, 9] is, technically speaking, not correct. The definition of derivative is $\lim_{h\to\infty}(f(x+h) - f(x))/h$. For adjacent pixels, $h = 1$, so the discrete approximation is $f(x+1) - f(x)$, which leads to $\begin{bmatrix}1 & -1\end{bmatrix}$, as in [5]. However, this distinction does not affect the MTF estimate, since the normalization factor of 0.5 is removed anyway when the DFT is normalized so the DC component is 1; and the sign disappears when the magnitude of the DFT is taken.

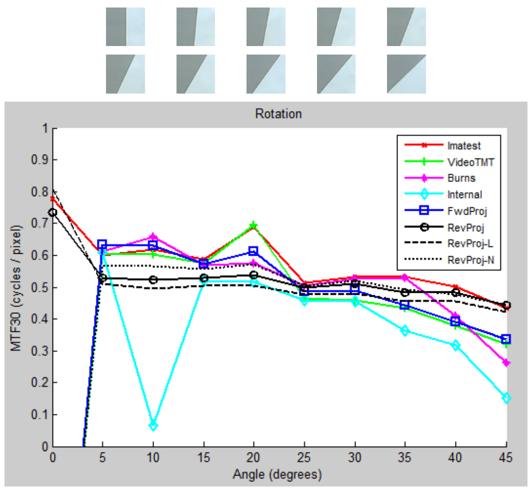

Figure 7. *TOP: Ten 50×50 crops of a slanted edge at angles of 0 to 45 degrees in steps of 5. BOTTOM: MTF30 values for the different techniques. The least variation over angle was achieved by reverse projection. Note that only two implementations were able to compute a value at 0°; invalid values are shown as -1.*

the central difference kernel is equivalent to first smoothing the signal using the kernel $\frac{1}{2}\begin{bmatrix}1 & 1\end{bmatrix}$ then convolving with the forward/backward difference kernel, which is easily seen from the associative property of convolution and the fact that

$$\begin{bmatrix}0.5 & 0 & -0.5\end{bmatrix} = \frac{1}{2}\begin{bmatrix}1 & 1\end{bmatrix} \circledast \begin{bmatrix}1 & -1\end{bmatrix}, \quad (2)$$

where $\circledast$ represents convolution. As a result, to avoid such unnecessary smoothing, it is recommended to use the smallest differentiating kernel possible, namely, $\begin{bmatrix}1 & -1\end{bmatrix}$.

In Step 8, the width of the Hamming window is set to the minimum of the length of the ESF and a maximum width. This maximum width is set to 15 pixels total (7 pixels on either side of the edge). Therefore, if a large ROI is used, pixels far from the edge (greater than 7 pixels away) will have no effect on the result.

Experimental Results

To evaluate the proposed technique, the following implementations were compared:

- Proprietary commercial implementation of ISO 12233 by Imatest (version 3.10),[5]
- Video TMT for Skype implementation of ISO 12233 (version 2.3.0.18).[6]
- Peter Burns' implementation of ISO 12233 (version 2.0).[7]
- Internal implementation of ISO 12233.
- Forward projection, implemented by the author.
- Reverse projection, implemented by the author. All steps are identical to the previous one except for the projection method (and the truncation of rows, which is not necessary in the case of reverse projection).

Note that all implementations except the last one use forward projection. Default parameters were used for all implementations.

[5]http://www.imatest.com
[6]https://www.microsoft.com/en-us/download/details.aspx?id=43372
[7]http://losburns.com/imaging/software/SFRedge/index.htm

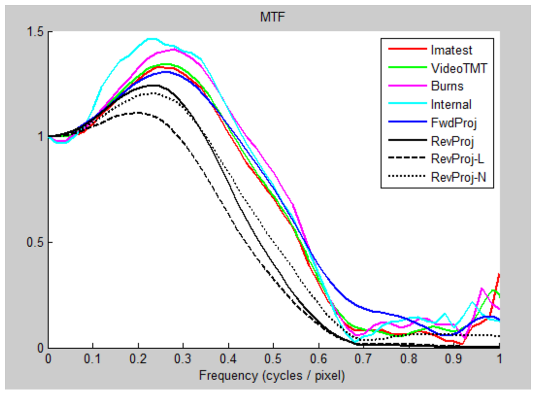

Figure 8. *MTF curves of the techniques of the 50×50 crop of the 5° edge.*

Experiment 1: Rotation. A slanted edge target with 4:1 contrast ratio was captured on-axis at 580 lux with 5540 K temperature (all corners within 10% of center) by a consumer-grade camera. Automatic gain control (AGC) was turned off, and the image was JPEG compressed. Beginning with a vertical orientation (0°), the target was rotated in increments of 5° to 45°. From each image, a 50×50 region was cropped around the edge.

Results of the different techniques are shown in Fig. 7, along with two variations of reverse projection by replacing bicubic interpolation with either bilinear (RevProj-L) or no (RevProj-N) interpolation. These results are summarized in the following table, ignoring 0° (at which most techniques failed to produce a result at all):[8]

	MTF30		
Method	μ	σ	outliers
Imatest	0.56	0.08	0
VideoTMT	0.50	0.12	0
Burns	0.52	0.12	0
Internal	0.42	0.14	1
FwdProj	0.51	0.11	0
RevProj	0.51	**0.03**	**0**
RevProj-L	0.48	**0.03**	**0**
RevProj-N	0.52	0.05	0

Note that reverse projection yielded the tightest result, with a standard deviation of only 0.03 cycles per pixel. Also note that the MTF30 value was only slightly higher without interpolation, with significantly higher variance. Not shown in the table is the fact that, even if 0° is included, reverse projection yields just $\sigma = 0.08$ cycles per pixel, which is the same as the second-best technique *without* considering 0°. Thus, the proposed technique degrades somewhat gracefully in extreme conditions.

The MTF curves of the techniques at 5 degrees are shown in Fig. 8. The MTF curve of reverse projection is smooth, whereas the other curves exhibit significant roughness, along with noticeable instability after 0.7 cycles per pixel. Fig. 9 shows the MTF curves of reverse projection for different angles; except for 0 and 45 degrees, there is a large degree of consistency across angle.

Experiment 2: Vertical crop. The 50×50 image for 5° slant (from the previous experiment) was successively cropped to

[8]Outliers were ignored in computing μ and σ. The best values are indicated by bold.

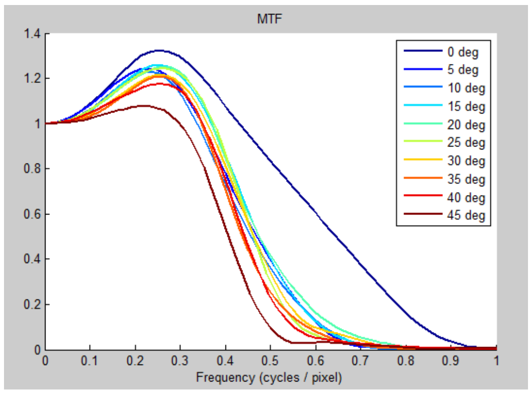

Figure 9. *MTF curves of reverse projection for the* 50×50 *crops at different angles.*

a different number of rows, from 3 to 50. Results of the different techniques are shown in Fig. 10. Note that, because all the crops were from the same image, the only difference between the inputs was the number of rows used, and therefore all the MTF30 values should have been identical. These results are summarized in the following table:

Method	MTF30 μ	σ	outliers
Imatest	0.66	**0.04**	**0**
VideoTMT	0.66	0.04	3
Burns	0.63	0.02	8
Internal	0.30	0.72	9
FwdProj	0.68	0.05	2
RevProj	0.56	**0.02**	**2**
RevProj-L	0.54	**0.02**	**2**
RevProj-N	0.62	0.05	2

Without ignoring outliers, the smallest variation was achieved by Imatest ($\sigma = 0.04$), with reverse projection following closely behind ($\sigma = 0.05$). But if the two smallest crops are ignored, the variation for reverse projection drops to $\sigma = 0.02$ cycles per pixel, which is the lowest among all techniques. By comparison, Imatest does not drop to $\sigma = 0.02$ until all crops less than 25 rows are ignored. If all outliers are included, then among crops from 25 to 50 rows, the variations are $\sigma = 0.01$ (reverse projection), $\sigma = 0.02$ (Imatest and VideoTMT), and $\sigma = 0.04$ (forward projection). Burns and internal do not stabilize until at least 40 rows of pixels are used.

Experiment 3: Horizontal crop. The same 50×50 5° image was then successively cropped to a different number of columns, from 5 to 50 in increments of 5. Results of the different techniques are shown in Fig. 11. Again, because all the crops were from the same image, all the MTF30 values should have been identical. These results are summarized in the following table:

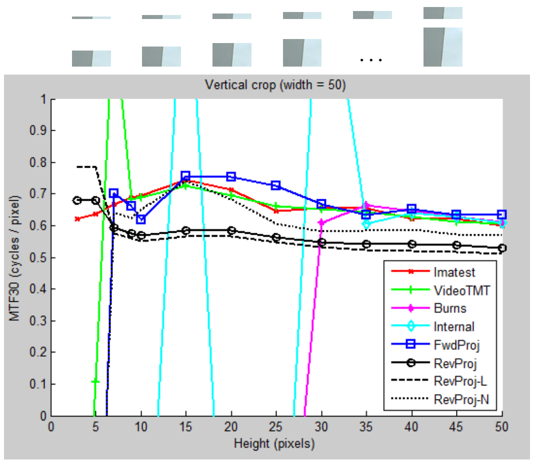

Figure 10. Top: *Ten of 13 crops from the same* 5°*-slant image, each with a different number of rows.* Bottom: *MTF30 values at vertical crops of 3, 5, 7, 9, and 10 through 50 pixels, the latter in steps of 5. Of all the techniques, Imatest and reverse projection produced the least variation.*

Method	MTF30 μ	σ	outliers
Imatest	0.60	0.01	1
VideoTMT	0.61	0.00	1
Burns	0.62	0.01	1
Internal	0.64	0.01	1
FwdProj	0.64	0.01	0
RevProj	0.53	**0.00**	**0**
RevProj-L	0.51	0.01	0
RevProj-N	0.57	**0.00**	**0**

Among the techniques with zero outliers, the smallest variation was achieved by reverse projection, with $\sigma = 0.00$ cycles per pixel. In other words, reverse projection yielded almost the exact same measurement for all crops, even with as few as 5 columns of pixels.

Experiment 4: Square crop. The same 50×50 5° image was then successively cropped in concentric squares, with side lengths from 5 to 50 pixels in steps of 5. Results of the different techniques are shown in Fig. 12, summarized in the following table:

Method	MTF30 μ	σ	outliers
Imatest	0.65	0.04	2
VideoTMT	0.66	0.04	1
Burns	0.65	0.04	3
Internal	0.63	0.02	4
FwdProj	0.68	0.06	1
RevProj	0.56	**0.02**	**1**
RevProj-L	0.54	**0.02**	**1**
RevProj-N	0.62	0.06	1

Again, reverse projection yielded the least variation.

Summary. These four experiments are together summarized in the boxplot of Fig. 13. Only angles between 5 and 20 degrees, inclusive, were considered, and only crops with smallest dimension at least 25 pixels. Even so, there was 1 outlier with Burns and 3 with internal. These results are summarized in the following table:

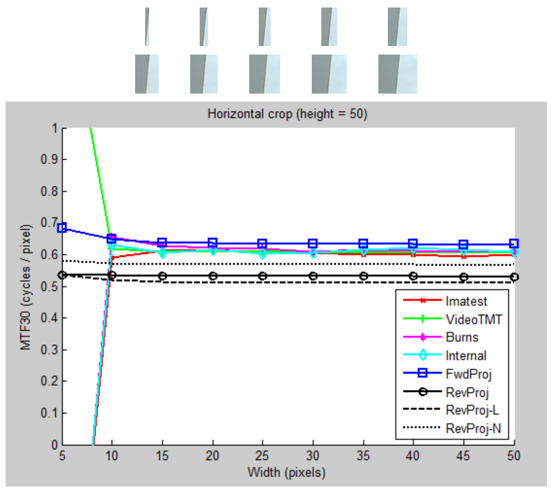

Figure 11. TOP: *Ten crops from the same 5°-slant image, each with a different number of columns.* BOTTOM: *MTF30 values at horizontal crops of 5 through 50 pixels in steps of 5. Reverse projection produced the least variation.*

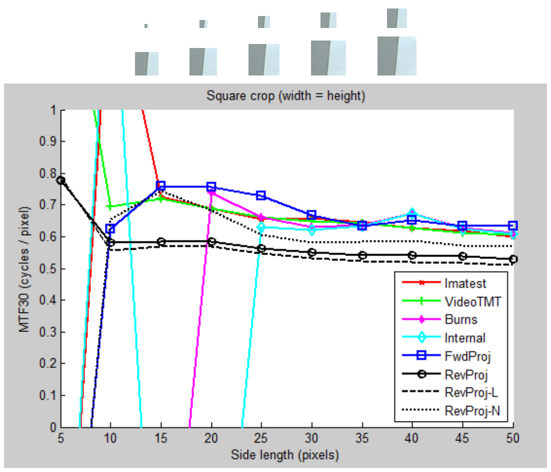

Figure 12. TOP: *Ten crops of concentric squares from the same 5°-slant image. The length of the side of the square ranges from 5 to 50 pixels in steps of 5.* BOTTOM: *MTF30 values at these square crops. Reverse projection produced the least variation, but all techniques failed to find a reasonable value for the smallest 5×5 square.*

Method	μ	σ	MTF30 outliers	min	max	range
Imatest	0.62	0.03	**0**	0.59	0.69	0.10
VideoTMT	0.62	0.03	**0**	0.58	0.69	0.11
Burns	0.62	0.03	1	0.57	0.67	0.10
Internal	0.61	0.04	3	0.52	0.67	0.15
FwdProj	0.64	0.03	**0**	0.57	0.73	0.16
RevProj	0.54	**0.01**	**0**	0.52	0.56	**0.04**
RevProj-L	0.52	**0.01**	**0**	0.50	0.55	0.05
RevProj-N	0.58	**0.01**	**0**	0.56	0.61	0.05

In other words, if someone desires to measure the MTF30 of a particular slanted edge, the result can vary within a substantial range (0.52 to 0.69 in this case) depending upon the method used. To remove the variation due to different angles, the following table shows the results limited to the single angle of 5 degrees. In other words, the variation listed here arises simply from different crops within a single image of a single edge:

Method	μ	σ	MTF30 outliers	min	max	range
Imatest	0.62	0.02	**0**	0.60	0.66	0.06
VideoTMT	0.62	0.02	**0**	0.61	0.66	0.05
Burns	0.63	0.02	1	0.61	0.67	0.06
Internal	0.62	0.02	2	0.60	0.67	0.07
FwdProj	0.65	0.03	**0**	0.63	0.73	0.10
RevProj	0.54	**0.01**	**0**	0.53	0.56	**0.03**
RevProj-L	0.52	**0.01**	**0**	0.51	0.55	0.04
RevProj-N	0.58	**0.01**	**0**	0.57	0.61	0.04

As before, reverse projection yields the lowest variation ($\sigma = 0.01$) and the tightest range (0.03).

Relationship to Previous Work

In this section let us briefly consider some of the relevant literature. Williams [2] analyzed the ISO 12233 SFR plug-in accompanying the standard for accuracy, precision, and field robustness. The tool was found to yield results insensitive to the angle of the edge when presented with noiseless 1D Gaussian edges shifted horizontally to create rows of a 2D image. At 5 degrees, and with a blurry edge (MTF30 approximately 0.25) the tool yielded accurate results for a range of SNRs, with horizontal cropping necessary at lower SNRs. For a less blurry edge, more subsampling was found to be necessary (8X for MTF30 = 0.5, 16X for MTF30 = 0.75). The final conclusion was that the tool is accurate, precise, and robust.

Several authors have analyzed the ISO 12233 algorithm in various ways. Burns and Williams [3] applied several steps from the algorithm to the different color channels in order to analyze color misregistration error in an image. Burns [5] analyzed several sources of error in MTF calculation, such as errors in the estimate of the edge angle, reduced MTF due to using a central difference convolution kernel for differentiation, and the effect of additive white Gaussian noise. Williams and Burns [6] used the algorithm to analyze characteristics of various cameras and scanners by measuring MTF; they recommend MTF10 as the limit of resolution, based on the Rayleigh criterion. An additional analysis of sources of error is provided by Burns and Williams [7], who found that large crops across the edge lead to noise, and errors can be caused by curved lines from lens distortion; they also recommend low contrast in the target design to reduce clipping.

More recently, Williams and Burns [8] describe deviations from the ISO 12233 standard that practitioners have devised, such as Gaussian blur filtering away from the edge, and fitting a higher-order polynomial rather than a line to handle lens distortion. Note that the potential benefit of such a higher-order model is reduced substantially by the technique proposed in this paper due to its ability to handle small crops. Roland [10] suggests a 4:1 contrast ratio and 5-degree edge from the ISO 12233:2014 standard, and he proposes a correction function to overcome the dependency of MTF on angle, in a study that highlights that even a well-controlled environment can have significant variability in es-

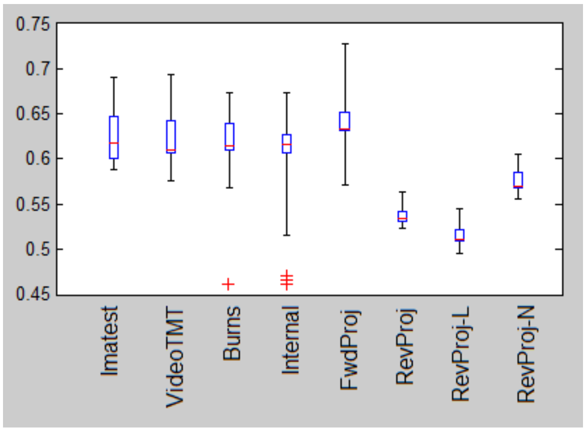

Figure 13. *Boxplot of MTF30 results on the set of images with angle between 5 and 20 degrees and crop at least 25 pixels. Each blue box delineates 25-75% percentiles, each red line shows the median, the black bars show the range of values for inliers, and each red plus (+) indicates an outlier (value of -1, displayed arbitrarily near* 0.45*).*

timated MTF. The rotation invariance of the technique proposed in this paper obviates the need for such a correction.

Conclusion

This paper has presented a novel algorithm for measuring MTF using a slanted edge. Rather than projecting values forward from the 2D image to a 1D array, as in the ISO 12233 standard, the algorithm projects in the reverse direction from the array to the image. To increase the algorithm's robustness, and to reduce its dependence upon the angle of the edge, two additional novelties are introduced: using the Ridler-Calvard binary image segmentation method for finding the edge, and using total least squares to find the line parameters. Together, these changes yield an algorithm that is largely rotation invariant and exhibits less variation than existing solutions.

Because the technique is stable, it requires fewer pixels to yield an estimate for MTF. As a result, smaller crops are allowed than are usually considered sufficient, thus justifying the use of line fitting rather than fitting higher-order models. Nevertheless, an important question that remains unanswered is that of accuracy: The proposed approach yields MTF values that are noticeably less than those of existing methods, even when interpolation is not used. Further investigation is needed to determine whether these lower values are more or less accurate than those found by existing techniques.

Acknowledgments

The author appreciates the help of Andrew Juenger, Erik Garcia, and Kanzah Qasim in conducting this investigation.

References

[1] T. W. Ridler and S. Calvard, Picture thresholding using an iterative selection method, *IEEE Transactions on Systems, Man, and Cybernetics*, 8(8):630–632, Aug. 1978.

[2] D. Williams, Benchmarking of the ISO 12233 slanted-edge spatial frequency response plug-in, Proc. of the IS&T PICS Conference, pp. 133–136, May 1998.

[3] P. D. Burns and D. Williams, Using slanted edge analysis for color registration measurement, Proc. of the IS&T PICS Conference, pp. 51-53, Apr. 1999.

[4] ISO 12233: Photography — Electronic still-picture cameras – Resolution measurements, 2000.

[5] P. D. Burns, Slanted-edge MTF for digital camera and scanner analysis, Proc. of the IS&T PICS Conference, pp. 135-138, March 2000.

[6] D. Williams and P. D. Burns, Diagnostics for digital capture using MTF, Proc. of the IS&T PICS Conference, pp. 227–232, May 2001.

[7] P. D. Burns and D. Williams, Refined slanted-edge measurement for practical camera and scanner testing, Proc. of the IS&T PICS Conference, pp. 191–195, Apr. 2002.

[8] D. Williams and P. D. Burns, Evolution of slanted edge gradient SFR measurement, Proc. of SPIE 9016: Image Quality and System Performance XI, Aug. 2014

[9] ISO 12233, 2nd edition: Photography – Electronic still-picture imaging – Resolution and spatial frequency responses, 2014.

[10] J. K. M. Roland, A study of slanted-edge MTF stability and repeatability, Proc. of SPIE 9396: Image Quality and System Performance XII, Feb. 2015.

Author Biography

Stan Birchfield conducts research in computer vision and robotics at NVIDIA. This work was performed while he was at Microsoft. Previously, he was a tenured professor of electrical and computer engineering at Clemson University, where he remains an adjunct faculty. He also worked as a research engineer at a startup company in Silicon Valley and has consulted for various companies. He received his Ph.D. from Stanford University.