Efficient Image Generation for EO System Simulation

D. L. Webb, D. F. Elliott, L. L. Hung
Rockwell International Corporation
Anaheim, CA 92803

Abstract

Efficient image generation techniques for electro–optical (EO) system simulation are presented. The techniques reduce computation time by orders of magnitude so Monte Carlo simulations become practical. Ray tracing to project signals onto a focal plane and convolution to blur projected images are time consuming so methods that minimize this time are most important. Methods for efficient simulation of detector characteristics, including noise, are also described.

1 Introduction

Electro–optical sensors are playing an increasingly important role in surveillance and defense systems, and there is a growing emphasis on simulation to verify performance before expensive hardware fabrication. Generating images that represent an EO sensor's output is time consuming so minimizing this time is important for engineering and computer resources. This paper describes how to compute the images and how to speed up this computation while maintaining the necessary accuracy.

A baseline for EO sensor image generation is presented first to establish terminology and concepts and give a general purpose method. Techniques are then described which give orders of magnitude speed up over the baseline.

2 Baseline method of image generation

Image generation means producing an output value for each detector and doing so at each readout time. The output value, out_{ij} for the detector in row i and column j depends on 1) input photons from background, b_{ij}, and signal, s_{ij}, 2) photon noise, $pnoise_{ij}$, with zero mean and variance $b_{ij} + s_{ij}$, 3) the detector's gain and offset, 4) electronics noise, $enoise_{ij}$, and 5) the detector's digital gain and offset corrections, if any:

$$out_{ij} = \min\left[\left(\begin{matrix}saturation\\value\end{matrix}\right), \left(\left(b_{ij} + s_{ij} + pnoise_{ij}\right) \cdot gain_{ij} + offset_{ij} + enoise_{ij}\right)\right] \cdot gaincorr_{ij} + offsetcorr_{ij} \quad (1)$$

The detector gains, offsets, and gain and offset corrections are assumed constant and obtaining values for electronics noise is straightforward so the problem is to determine the input photons. A detector receives most of the photons that are directed toward it by the optical system and some that are directed toward nearby locations. Let $g(x,y)$ be the instantaneous rate of photons directed toward focal plane location x,y and let S_{ij} be the instantaneous arrival rate for detector i,j. Then [2] shows that S_{ij} is given by

$$S_{ij} = \iint_{\substack{region\ of\ focal\\plane\ around\\detector\ i,j}} \left[g(x,y) f_{sr}(x-i, y-j)\right] dx\,dy \quad (2)$$

where $f_{sr}(u, v)$ is called a sensor response function and is effectively zero except near the u,v origin.

The set of $g(x,y)$ values is referred to as a geometric image. Equation (2) is used when objects can be represented as point sources. Otherwise, and for the remainder of the baseline method, the geometric image is approximated by a two–dimensional (2D) array of values, $g(m,n)$, so that

$$S_{ij} = \sum_{m=1}^{M} \sum_{n=1}^{N} \left[g(m,n) \cdot f_{sr}(m-i, n-j)\right] \Delta x \Delta y \quad (3)$$

where $\Delta x \Delta y$ is the spacing in the 2D array. The set of S_{ij} values is referred to as a blurred image. Equation (3) indicates that neighboring elements of a blurred image share values from a geometric image. Rather than calculate a shared value more than once, an entire geometric image is obtained prior to calculating any of the blurred image.

Each geometric image is computed from the objects in a field of view at one instant in time to form a "snapshot" which can be used to produce a blurred image. Time integration of a blurred image over the frame integration period is an image of detector input signals, s_{ij}, as required by (1) for that frame. This is approximated by

$$s_{ij} = \sum_{k=1}^{K} S_{ij,k} \Delta t \quad (4)$$

where Δt = frame integration period/K and $S_{ij,k}$ is the i, j^{th} sample of the blurred image at the center of the k^{th} time interval. If motion is negligible then K can be equal to 1.

2.1 Geometric image formation

A geometric image is formed by projecting objects from a field of view onto the plane of the focal plane array. This can be done by 1) representing an object as a collection of point sources and projecting each point onto the focal plane, or 2) assuming constant radiance over an object's surface, determining the boundary of its projection on the focal plane, and assigning a constant intensity throughout the

1058-6393/95 $4.00

interior of the bounded region, or 3) using ray tracing to determine the intensity at each point of an $M \times N$ grid on the focal plane. The baseline method uses ray tracing.

Ray tracing: Ray tracing produces an instantaneous snapshot by projecting rays from the sensor outward into the field of view. A ray's first intersection, if any, with an opaque object determines the intensity at the ray's position in the focal plane. The rays are spaced apart in a grid covering a region of interest in the focal plane and each ray is defined by its grid position and the viewing direction of the sensor at that point.

Each ray, defined as a point and a vector in the sensor coordinate system, is transformed into the object's coordinate system using a 3D rotation and translation, as described in [1]. The intersection, if any, of the ray and the object is then determined, as shown in the following sections where $\vec{a},\hat{b}$ is a ray in object coordinates, $\vec{a}$ is the ray origin, $\hat{b}$ is a unit vector along the ray, $\vec{a} = (\alpha_1, \alpha_2, \alpha_3)$, and $\hat{b} = (\beta_1, \beta_2, \beta_3)$.

Ray intersection with a cylinder: A cylindrical solid is defined with radius r about its own z–axis from z_{base} to z_{top}. The ray intersects the cylinder if it is distance r from the axis when its z–coordinate is between z_{base} and z_{top}. This requires a scalar g such that

$$(\alpha_1 + g\beta_1)^2 + (\alpha_2 + g\beta_2)^2 = r^2 \tag{5}$$

$$z_{base} \le \alpha_3 + g\beta_3 \le z_{top} \tag{6}$$

One of two real solutions, if any, of the quadratic equation (5) determines g. If $\alpha_1^2 + \alpha_2^2 < r^2$ or $\beta_3 < 0$, but not both, then the larger solution is used and, otherwise, the smaller is used. g is then checked using (6) and if the solution is valid then the radiance at that point on the cylinder is used to assign an intensity at the ray's position in the focal plane.

Ray intersection with a cone: A solid half–cone is defined with radius r about its own z–axis at its base at z_{base}, and tapering to a point at z_{tip}. The ray intersects the cone if it is a distance $\frac{z - z_{tip}}{z_{base} - z_{tip}} \cdot r$ from the axis when its z–coordinate is z and $z_{base} \le z \le z_{tip}$. This requires a scalar g such that

$$(\alpha_1 + g\beta_1)^2 + (\alpha_2 + g\beta_2)^2 = \left[\frac{\alpha_3 + g\beta_3 - z_{tip}}{z_{base} - z_{tip}} r\right]^2 \tag{7}$$

$$z_{base} \le \alpha_3 + g\beta_3 \le z_{tip} \tag{8}$$

One of two real solutions, if any, of the quadratic equation (7) determines g. If $\vec{a}$ is inside the full–cone or $\beta_3 < 0$, but not both, then the larger solution is used and, otherwise, the smaller is used. g is then checked using (8) and, if the solution is valid, the radiance at that point on the cone is used to an assign an intensity at the ray's position in the focal plane.

Ray intersection with convex polygon: A planar convex polygon is defined as a counterclockwise ordered list of vertices in its own x, y plane. The ray intersects the polygon if its 2D intersection point in the plane is inside the polygon. Let $\vec{p}$ be the position of the ray in the plane, $\vec{p} = (p_1, p_2) = (\alpha_1, \alpha_2) - \frac{\alpha_3}{\beta_3} \cdot (\beta_1, \beta_2)$. $\vec{p}$ is inside the polygon if it is to the left of each boundary segment. This requires, for each two successive vertices, $\vec{u}$ and $\vec{v}$, with $\vec{u} = (u_1, u_2)$ and $\vec{v} = (v_1, v_2)$, that $(\vec{v}-\vec{u}) \times (\vec{p}-\vec{u}) > 0$ or, equivalently, that

$$(v_1 - u_1) \cdot (p_2 - u_2) > (v_2 - u_2) \cdot (p_1 - u_1) \tag{9}$$

If $\vec{p}$ is inside the polygon then the radiance at $\vec{p}$ is used to assign an intensity at the ray's position in the focal plane.

2.2 Blurred image computation

A blurred image is obtained by convolving a geometric image with a sensor response function and evaluating the result at each detector position.

Sensor response function: The sensor response function is a surface relating power received from a point source by a detector, to distance of the detector center from the geometrical image of the point source. It is the convolution of a point spread function and a detector response function.

Point spread function: A point spread function (PSF) is a surface which represents a point object diffraction pattern resulting from the telescope optics over the EO wave band to which the detector responds. When power from a distant point source reaches the optics it spreads out as represented by the PSF.

Detector response function: The detector response function is a surface which represents the sensitivity of the detector to photons in an underlying region of the focal plane. This surface is a function of the material and physical area of a detector but blurred by the effect of diffusion across the focal plane array substrate [3].

2.3 Readout image calculation

Each blurred image can be obtained from an instantaneous snapshot of the field of view. The effect of sensor motion can be produced by summing a number of blurred images during the frame integration period. The result is an image of detector input signals, s_{ij} for each i,j, as required by (1) to produce the final readout.

2.4 Summary of baseline method

1. Get geometric image (snapshot) using ray tracing.
2. Blur the snapshot using convolution.
3. If more snapshots are required, then go to step 1.
4. Get readout image from the sum of blurred snapshots.

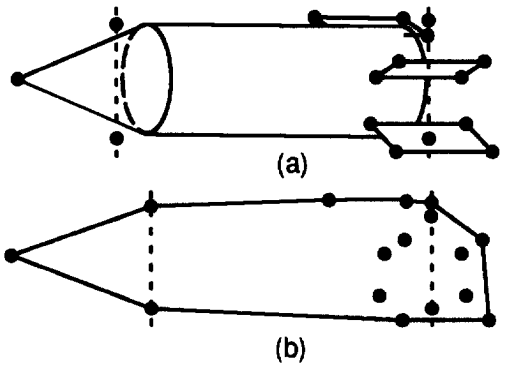

Figure 1. Possible vertices on object's convex hull.

3 Efficient image generation

There are many ways to speed up the baseline method. Techniques for reducing convolution time are most important, followed by table–driven noise modeling, and then optimization of the ray tracing procedure. Convex hulls are described first because they are used to limit calculations to regions of interest.

3.1 Convex hulls

The convex hull of a set of 2D points is the smallest subset that forms a closed polygon around the entire set. An efficient method for finding the hull is

1. Initialize a hull with top, leftmost, bottom, and/or rightmost points, counterclockwise.
2. Let $\vec{p}$ be the first (or next) remaining point, if any, else END.
3. Let $\vec{u},\vec{v}$ be the first (or next) segment, if any, else goto step 2.
4. If $\vec{p}$ is to the left of $(\vec{v}–\vec{u})$, (using (9)), then goto step 3.
5. Let $\vec{v},\vec{w}$ now be the next segment of the hull.
6. If $\vec{p}$ is not to the left of $(\vec{w}–\vec{v})$, (using (9)), then goto step 5.
7. Let $\vec{p}$ replace all the vertices between $\vec{u}$ and $\vec{v}$ and goto step 2.

This method is implemented using a linked list for the hull vertices and it keeps them in counterclockwise order. If the hull consists of M points out of a total of N then this algorithm is of order $M \cdot N$ worst case with $M \cdot N/2$ expected. Due to the low cost of each step, this is much more efficient for reasonable M and N than even the $N \cdot logN$ algorithm shown in [4].

3.2 Optimization of ray tracing

Ray tracing is optimized by 1) doing ray tracing only within the convex hull of an object's projection on the focal plane, 2) using a grid of rays fine enough for accuracy, but no finer, 3) avoiding calculations on hidden object components, and 4) exploiting the common origin of the rays.

3.2.1 Convex hull: Determining the convex hull of an object's projection on the focal plane and doing ray tracing only within this hull avoids examining rays that cannot intersect the object. Figure 1 illustrates the convex hull of a missile object's projection on the focal plane. All possible vertices of the convex hull are determined as extreme points of the projection of the object's components. Note in Figure 1 (a) that vertex points (dots in the figure) are placed at the cone tip and fin corners, and outside the ellipses formed by projecting the ends of the cylinder. Figure 1 (b) shows the vertices selected by the algorithm of Section 3.1 to form the convex hull.

Rays are defined on a 2D grid in the focal plane, covering the object's convex hull. The rays within the hull are accessed by scanning across the grid from one edge of the hull to another. Efficient implementation of the scanning procedure relies on the hull's convexity, the ordering of its edges, and, for horizontal scanning, having the top–left vertex be the first one in the list.

3.2.2 Fineness of the grid: The spacing between rays on a grid influences the number of rays that intersect an object. A more accurate representation requires more rays and, hence, more computation time. The grid is made finer for distant objects so they are intersected by a sufficient number of rays, e.g. at least five across the object width. The grid is made coarser for nearer objects to get only a necessary number of rays, e.g. as few as one per detector.

3.2.3 Hidden objects: The object's aspect angle allows some components to be completely ignored since they are not visible to the sensor. For the missile object in Figure 2, this includes the cylinder's base in any head–on view and the entire nose cone in many tail–on views. When the sensor views the missile between head–on and broadside, two fins, the nose cone, body, and possibly part of the other two fins can be seen. If ray intersections are checked in that order, then the first component that a ray intersects is the only component seen by that ray, and checking ceases.

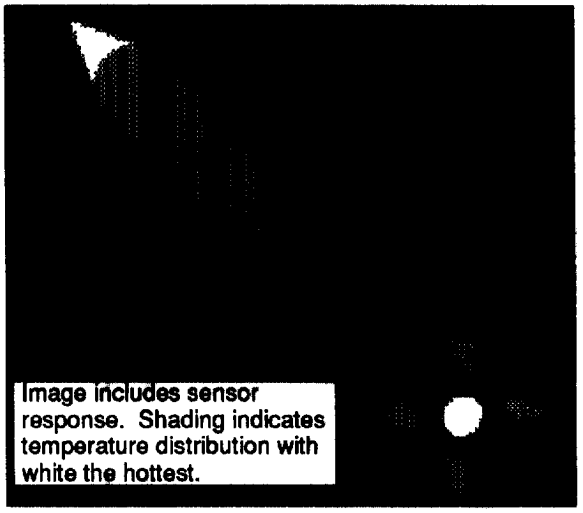

Figure 2. Image determined by ray tracing.

When the object aspect angle is between broadside and end view, the order of checking is somewhat reversed. In this case, the order is two fins, cylinder base (including exhaust

exit hole) cylinder, cone, and the other two fins. Knowledge of the missile shape makes possible the definition of the aspect angles at which a component can be seen, as well as the order in which to check the components.

3.2.4 Common ray origin: The rays' origin, $\vec{a}$, is the same for all object components defined in the same coordinate system. This allows many of the intermediate values in the intersection calculations to be computed only once for the entire set of rays. For example, a solution to (5) or (7) can use intermediate values which, if independent of β_1,β_2,β_3, are calculated once for all rays toward the cylinder or cone.

3.3 Reducing convolution time

Convolution time can be reduced by exploiting 1) point source objects, 2) constant PSF, 3) linearity of convolution, 4) alignment of the ray tracing grid, 5) fineness of the grid, 6) matrix dot product, 7) eight–way symmetry of the sensor response function, and 8) convex hull.

3.3.1 Point source objects: When an object is represented as a collection of point sources, then its signal can be processed one point at a time to obtain a blurred image without convolution. For each point source, the sensor response function is centered at its projected position on the focal plane, scaled by the amount of signal, and evaluated at each nearby detector position.

As an example, the projection of a missile at long range can be represented as points on a line. This requires assigning an appropriate intensity to each point on the line. An array of values representing the distribution of signal along the line is precomputed for each possible orientation of the missile with respect to the sensor. The values are stored in a table, indexed by orientation angles, so the object's point source representation is always available.

3.3.2 Constant PSF: The detector response function is constant, but the PSF depends on the distance from the sensor to the object. Only when this distance is small and rapidly changing must the PSF be updated. Thus, the sensor response function need be recomputed only once for many images. For example, it can be recomputed whenever an object's distance changes by a factor of two.

3.3.3 Linearity of convolution: When the sensor response function is constant for the duration of a frame integration period, the sum of the blurred images can be obtained as the convolution of a sum of geometric images, i.e. snapshots. The convolution is done once for the sum of snapshots rather than once for each snapshot, reducing convolution arithmetic by a factor equal to the number of snapshots per frame.

3.3.4 Alignment of the ray tracing grid: The ray tracing grid is positioned so that detector center positions are coincident with grid positions. The grid spacing is constrained to be an integer number of samples per detector. This approach avoids the arithmetic associated with interpolation of a 2D surface because the convolution's output surface will have a defining point at each detector center position.

3.3.5 Fineness of the grid: Convolution time is affected by grid fineness in a manner similar to the discussion in Section 3.2.2. The sensor response function is recomputed, when necessary, to have the same spacing as the grid.

3.3.6 Matrix dot product: In the baseline method the sensor response is convolved with a snapshot and the result is evaluated at detector center positions to obtain an image of instantaneous detector inputs. Efficient processing avoids the convolution by aligning the grid with the detector center positions (see Section 3.3.4) so the instantaneous input value for each detector is the matrix dot product of 1) the sensor response function centered at the detector position, and 2) a region of the grid centered at the detector's position in the geometric image.

3.3.7 Eight–way symmetry of sensor response function: The sensor response function is the convolution of the PSF of the optics and the detector response function (see [2]), both of which may be assumed to have 8–way symmetry. In general, the PSF is not perfectly symmetrical. However, simulations have demonstrated that the exact shape of a PSF is not critical, but that the maximum fractional power from a point object on a detector is significant in modeling a sensor. Typically, the percent power of the PSF on a detector is matched with a gaussian surface which has eight–way symmetry and is used for the PSF.

Many detector response functions also have eight–way symmetry as can be shown by solving the diffusion equations for electrons resulting from photon impacts [3]. Due to the symmetry of of the PSF and the detector response function, the sensor response function, which is their convolution, is symmetric. The coordinate system is chosen so that points on the function have equal value if they are opposite each other with respect to any line through the center with an orientation of 0°, ±45°, or 90°, as shown by the eight points in Figure 3 having equal value.

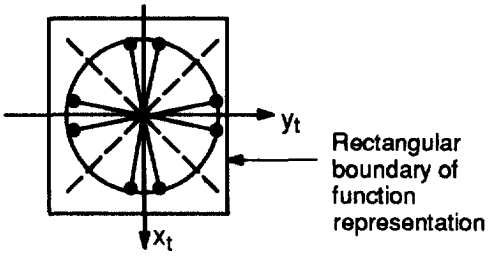

Figure 3. Eight–way sensor response symmetry.

The matrix dot products impact simulation speed, so they are optimized using the inherent eight–way symmetry of the sensor response function. When forming the dot product, the object signal on the grid under the dots is added and

then multiplied by the common sensor response for the eight points. For integer ADC outputs, eight fixed–point additions and one floating–point multiplication are required vs. eight fixed–point additions and eight floating–point multiplications for the baseline method. This yields a savings factor of between two and eight, depending on the computer's arithmetic.

3.3.8 Convex hull: An image to be blurred is a grid of signal values accumulated from one or more snapshots. The region in which blurring is required must be determined. First, the accumulated list of convex hull vertices from the ray tracing optimization of Section 3.2.1 is reduced to a single hull over the entire region using the algorithm of Section 3.1. This hull is then expanded in size by the effective radius of the sensor response function, as shown in Figure 4. Any detector that lies outside this expanded hull is not affected by the signal values so it can be ignored.

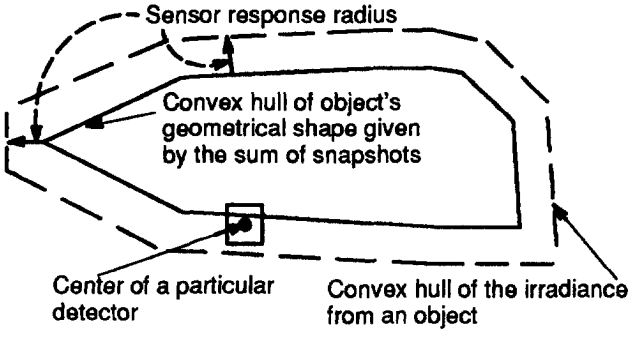

Figure 4. Convex hull including sensor response.

3.3.9 Comparison with FFT convolution: To compare FFT convolution with the methods of Sections 3.3.4–3.3.8, consider a region of $n \times n$ square detectors with $s \times s$ samples per detector, an effective sensor response function diameter of d detectors, and p percent of the n^2 detectors having their centers inside the expanded convex hull. The FFT method requires two 2D FFTs of size $N \times N$ where $N = 2^{\lceil \log_2 ns \rceil}$ at 6 operations per butterfly or $24N^2 logN$ ops. The method of Sections 3.3.4–3.3.8 requires $pn^2d^2s^2$ ops. Table 1 compares the number of operations for FFT and dot product methods.

3.4 Table driven readout calculations

Assume the detector gains and offsets are constant. If the background, b_{ij}, is also constant and if $b_{ij} \gg s_{ij}$ then (1) reduces to

$$out_{ij} = \min\left[\left(\begin{matrix}saturation\\value\end{matrix}\right), \left(s_{ij} \cdot gain_{ij} + constant_{ij} + noise_{ij}\right)\right] \cdot gaincorr_{ij} + offsetcorr_{ij} \quad (10)$$

In this case the following steps can be used to produce the readout values.

1) For each i,j: $u_{ij} = v_{ij} = constant_{ij} + noise_{ij}$.
2) For each $s_{ij} > 0$: $v_{ij} = \min[(saturation\ value), (u_{ij} + s_{ij} \cdot gain_{ij})]$.
3) For each i,j: $out_{ij} = v_{ij} \cdot gaincorr_{ij} + offsetcorr_{ij}$.

This reduces computation time because gain factors and saturation checks are only required for those detectors which receive an input signal $s_{ij} > 0$ from an object.

The $noise_{ij}$ term includes 1/f noise and white gaussian noise in amounts that depend on the individual detector and are known a priori. A number of long noise sequences are generated off line and each detector is assigned a random start location in an appropriate sequence. Each noise value can then be obtained by a table lookup saving approximately a factor of 50 over the gaussian generator described in [5].

4 Summary

Efficient methods of EO sensor image generation have been described to speed up simulations by orders of magnitudes. Ray tracing yields an image on a grid that is in general finer than the detector size. This image is effectively convolved with a sensor response function to determine the amount of signal on each detector. Convex hulls are used to limit ray tracing to regions of interest. Matrix dot products replace convolutions in these regions.

Table 1. Cost of FFT and dot product arithmetic.

p (%)	s	d	n	N	$24N^2logN$ (FFT)	$pn^2d^2s^2$ (dot prod)	Ratio
50	2	2	4	8	4608	128	36
		3	6	16	24,576	648	38
	3	2	8	32	122,880	1,152	110
		3	10	32	122,880	4,050	30
			20	64	589,824	16,200	36
25	4	2	4	16	24,576	128	190
		3	6	32	122,880	1,296	95
	5	2	8	64	589,824	1,600	370
		3	10	64	589,824	5,625	100
			20	128	2,752,512	22,500	122

References

[1] D. L. Webb, L. L. Hung, R. A. Hubbs, and D. F. Elliott, "Passive seeker/KKV performance evaluation," reprints available from DTIC, *3rd Annual AIAA/BMDO Interceptor Technology Conf., July 11–14, 1994*.

[2] D. F. Elliott and J. L. Jenkins, "Convolutional relationships in EO sensors," Conf. Record *Twenty–fourth Asilomar Conf. Signals, Systems & Computers*, pp. 88–92, Nov., 1990.

[3] D. T. Cheung, "MTF Modelling of Backside–Illuminated PV Detector Arrays," pp. 301–316, *Infrared Physics*, Vol. 21, 1981.

[4] R. Sedgewick, "Algorithms," Addison–Wesley, 1983.

[5] W. H. Press, et. al., "Numerical Recipes," Cambridge University Press, 1988.