Design Process for NIF Laser Alignment and Beam Diagnostics

Andrew Grey

June 9, 1998

This is an informal report intended primarily for internal or limited external distribution. The opinions and conclusions stated are those of the author and may or may not be those of the Laboratory.

Work performed under the auspices of the Department of Energy by the Lawrence Livermore National Laboratory under Contract W-7405-Eng-48.

DISCLAIMER

This document was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or the University of California, and shall not be used for advertising or product endorsement purposes.

This report has been reproduced
directly from the best available copy

Available to DOE and DOE contractors from the
Office of Scientific and Technical Information
P O Box 62, Oak Ridge, TN 37831
Prices available from (615) 576-8401, FTS 626-8401

Available to the public from the
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Rd.,
Springfield, VA 22161

Hartmann Spot Centroid Calculation Using Frequency Domain Analysis

1. Abstract

In a controller for an adaptive optic system designed to correct phase aberrations in a high power laser, the wavefront sensor is a discrete Hartmann-Shack design. It uses an array of lenslets (like a fly's eye) to focus the laser into 77 spots on a CCD camera. Average local tilt of the wavefront across each lenslet changes the position of its focal spot.

The system requires 0.1 pixel accuracy in determining the focal spot location. We determine a small area around each spot's previous location. Within this area, we calculate the centroid of the light intensity in x and y. This calculation fails if the spot regions overlap. Especially during initial acquisition of a highly distorted beam, distinguishing overlapping spots is difficult.

However, low resolution analysis of the overlapping spots allows the system to estimate their positions. With this estimate, it can use the deformable mirror to correct the beam enough so we can detect the spots using conventional image processing.

Figure 1 -- wavefront control system

2. Background

Adaptive optics are required to achieve the National Ignition Facility focal spot requirement of 600TW inside a 200-250 micron diameter circle [J. Hunt]. The wavefront control system anticipates distortions induced by the laser amplifiers, long wavelength optical finishing errors, optical mounting errors, whole beam self focusing errors¹ and polishing errors during a high power shot. During laser alignment, the system corrects for thermal and optical effects in a closed control loop.

¹ A distortion present in intense lasers. A convex intensity profile produces a phase distortion similar to a positive lens. [J.M. Auerbach]

Wavefront control consists of three major components: a deformable mirror, wavefront sensor and digital controller (Figure 1). The mirror corrects phase errors. It uses 39 actuators to adjust zones across the face of the mirror. Sensing the wavefront, a Hartmann sensor uses 77 small lenslets focused on a CCD. Phase error across the wavefront causes the focus spot of each lenslet to move. The microprocessors analyze the Hartmann image and move the deformable mirror actuators to correct the phase aberrations.

Image processing initially acquires the spots using their expected locations given a beam free of distortions. In closed loop operation it tracks the spots by defining a small area around each spot's previous location. Within this area, it calculates the centroid of the light intensity in x and y.

Overlapping regions around each spot produce errors in the centroid calculation. Worse, the image processor may lose track or be unable to acquire one or more spots with large distortions.

3. Solution

Downsampling the Hartmann image using a linear phase filter with four zeros at π produces a low resolution image which reflects the concentration of spots across the image. The algorithm estimates each spot location's offset from its reference location based on the intensity of light surrounding it. Spots move towards brighter areas and away from darker areas. Using the estimated spot locations, the mirror controller calculates the actuator movement required to compensate for the spot distortion.

3.1. Low Resolution Image

I tried various trees and filters to create the low resolution image. A linear phase filter retains the relative locations of light and dark areas. However, a simple Haar filter applied in each dimension produced poor results. Resolution low enough to produce a smooth reference image did not retain enough detail to correct problems. Requiring zeros at π helped smooth the low resolution image, creating an image which, qualitatively, seemed to better reflect the original. However, the light and dark areas failed to stay registered with various length Daubechies' filters[M. Vetterli, table 4.3], since these filters are not linear phase. Since I use only one branch of the wavelet tree, orthogonality of the filters is not important. Additionally, perfect reconstruction of the original image is not necessary, since the algorithm works completely in the wavelet domain. Therefore, I calculated a linear phase filter with zeros at π . I used roots at $2 \pm \sqrt{3}$ in addition to the four roots at π :

$$[-0.0442 \quad 0 \quad 0.3977 \quad 0.7071 \quad 0.3977 \quad 0 \quad -0.0442]$$

With four zeros at π , the filter smoothed the reference image after five iterations and retained enough detail to estimate distortions in the high resolution image. Figure 2 shows the frequency response of the filter.

Since my intuitive picture of the algorithm involves a blurry version of the original Hartmann image, the best algorithm should use all low pass filters. I attempted a few variations using high pass filters, but none produced a good image.

I chose five iterations of the low pass filter. More iterations would have mapped many of the reference focal spots to the same pixel. With two focal spots mapped to the same pixel, their estimated movement would always be identical, so the system would be unable to correct for them separately. Fewer iterations would require longer low pass filters to smooth the reference image, complicating downsampling. The spot estimation algorithm would also require a larger (in pixels) weighted average to cover the same area on the CCD with fewer downsampling iterations.

3.2. Spot Estimation

After calculating the low resolution image, I map the reference spot locations to it and subtract the reference low resolution image. Since bright areas positively indicate a concentration of spots I reduce the weight given to darker pixels by five. This significantly improves performance in large corner distortions, since otherwise the spot tends to move in the wrong direction. For each reference spot in both x and y, I weight the surrounding pixels using the following vector in the chosen direction (for the shift in x, this vector multiplies the columns):

FIG 2

and the following vector in the opposite direction (for x, it multiplies the rows):

$$\begin{bmatrix} \frac{1}{8} & \frac{1}{2} & \frac{1}{2} & \frac{1}{\sqrt{2}} & 1 & \frac{1}{\sqrt{2}} & \frac{1}{2} & \frac{1}{2} & \frac{1}{8} \end{bmatrix}$$

The sum of all the weighted surrounding pixels produces the estimated offset in x and y for the focal spot.

With the estimated focal spot locations, the controller can calculate the phase distortion and move the actuators to correct it.

4. Performance

I simulated the algorithm's performance using Matlab. The algorithm processes distorted images simulated by calculating spot offsets produced by large actuator displacements. After processing the image and estimating the spot locations, it uses these offsets to determine corrective changes in the actuator positions. Using these new actuator positions, the simulator can determine the new focal spot positions and produce a new high resolution, simulated Hartmann image.

Initially, I evaluated the low resolution images qualitatively by thresholding the resulting image and comparing it to the original, high resolution image (Figure 3 - the images were created by moving the selected actuators to the given positions). Evaluating convergence of the maximum spot offset and mean square error confirmed my qualitative assessment. Many of the nonlinear filter simply didn't converge.

Once I produced a satisfactory low resolution image for a variety of distorted images, I developed and tested the spot estimation algorithm. Figures 4 and 5 show the system performance with the distortions shown in figure 3. [Note: the origin of Figures 4 and 5 is at the bottom left.] Figure 6 shows convergence of the spot's mean square error and maximum spot offset. The maximum spot offset measures the ability of the normal image processing algorithm to acquire the spots. Mean square error gives a rough estimate of the beam's quality. With actuators 1, 22 and 24, a sample corner, edge and center actuator, respectively, the algorithm converges to about five pixels maximum spot offset in seven iterations. With an average reference spot distance of about fifty, about ten pixels of error on each spot may be small enough to allow spot acquisition. Certainly it's a significant improvement over the original image.

In figure 7, further iterations reduce the quality of the image. However, examination of the resulting images reveals that the system starts spreading the edge spots, aiding spot acquisition. Since the dark areas are underweighted, the simulated spots in a noisy system will receive more influence from slightly light areas than slightly dark areas. Therefore, the estimated spots will tend to move toward the center. The weighted vectors described above are a compromise between correcting edge distortions and limiting this effect.

The mirror controller attempts to correct the wavefront while ignoring first order distortions. Another system controls tip and tilt so the mirror actuators save their dynamic range for higher order distortions. Figure 8 removes tip and tilt errors from the curves. Calculated with tip and tilt removed, mean square error starts larger but improves rapidly. The maximum spot offsets change only slightly, since the mean offset (the first order distortion) is only a small fraction of the maximum.

I compared results and computation speed using several techniques. With the Haar scaling function I performed convolution and downsampling five times. I tried one cascaded filter and a 32x downsample. I also computed the cascaded filter in the Fourier domain. The results were similar, but not identical. Iteration required less than 1/45th the floating point operations required by the Fourier method. The large convolution required four times the calculations of the Fourier method.

5. Refinements

I used relatively simple methods to construct the simulated Hartmann images. As a focal spot moves away from the reference location, it stretches along the offset axis. Test images did not simulate this effect. In addition, I constructed all the images by displacing actuators. This provided a problem with a perfect solution. Distortions produced in the optical path might have higher frequency variations than could be corrected by the mirror. Phase variation across the lenslets also blurs the focal spots. Other effects may produce intensity variations across the image. Both variations in spot shape and intensity will affect this algorithm's ability to estimate spot locations. Perhaps a lower resolution image will help estimate the

FIGURE 3

FIGURE 4

FIGURE 5

FIGURE 6

FIGURE 7

FIGURE 8

intensity variations across the image. Both the temporal and spatial frequencies of these variations will determine the best method to correct for them.

Actuator failures cause many of the highly distorted images. Correction of this image may be impossible since the controller biases the actuators at high voltage and a failed actuator rests outside its normal control range. The algorithm might automatically diagnose this problem and provide a candidate list of failed actuators.

Further revisions of this procedure might shift to a higher resolution image as the spots converge to their reference positions. As the working image progresses towards the initial, high resolution image, accuracy of locating the spot would increase to the 0.1 pixel precision required by the National Ignition Facility.

6. Bibliography

1. Auerbach, J.M. 1997. "The effect of full beam self-focusing on focal spot size." Presentation to staff of the National Ignition Facility on 24 April.
2. Hunt, J. 1997. "Review of the focal spot requirements and flowdown of requirements from the UV focal spot to the one micron system." Presentation to the staff of the National Ignition Facility on 24 April.
3. Salmon, J.Thaddeus, et al. 1991. "Real-time wavefront correction system using a zonal deformable mirror and a Hartmann sensor." *SPIE Vol. 1542 Active and Adaptive Optical Systems*: 459-467.
4. Vetterli, Martin and Jelena Kovacevic. 1995. *Wavelets and Subband Coding*. New Jersey: Prentice-Hall.

7. Appendices

7.1. Close

```
%
% FUNCTION - close
%
% USAGE: [new_actuators,new_image] =
%         close (actuators,filter,Offsets,Hmat,Gmat,blob,ref_image,image)
%
% PURPOSE: plot the original, estimated and corrected spot patterns
%           for two iterations of the low resolution spot estimateion
%           algorithm
%

function [actuators,high] = close
(actuators,fil,offsets,H,G,blob,low_ref,high);

%
% calculate the estimated spot positions based
% on either the given image (high) or actuator positions
%
if nargin < 8
    [new_offsets,low] = try(actuators,fil,offsets,H,blob,low_ref);
else
    [new_offsets,low] = estimate(fil,offsets,low_ref,high);
end;
%
% show the original, distorted image
%
figure(1);
```

```
clf;
subplot (2,2,1);
hold on;
distorted_offsets = ceil(offsets+H*actuators);
plot (distorted_offsets(1:77),distorted_offsets(78:154),'o');
plot (offsets(1:77),offsets(78:154),'.');
axis ([1 618 1 494]);
hold off;
title ('actual');
%
% show the estimated version
%
subplot (2,2,2);
hold on;
plot (new_offsets(1:77),new_offsets(78:154),'o');
plot (offsets(1:77),offsets(78:154),'.');
axis ([1 618 1 494]);
hold off;
title ('estimated');
%
% calculate the actuator movement required to correct
% the estimated image
% find the resulting image
%
actuators = actuators - 0.5*(G'*(new_offsets-offsets));
subplot (2,2,3);
if nargin > 1
    high = mk_image (actuators,offsets,H,blob);
    imagesc(high);
else
    new_offsets = ceil (offsets+H*actuators);
    plot (new_offsets(1:77),new_offsets(78:154),'o');
    hold on;
    plot (offsets(1:77),offsets(78:154),'.');
    axis ([1 618 1 494]);
    hold off;
end;
title ('first iteration');
%
% Do a second iteration
%
[new_offsets,low] = try(actuators,fil,offsets,H,blob,low_ref);
actuators = actuators - 0.5*(G'*(new_offsets-offsets));
subplot (2,2,4);
new_offsets = ceil (offsets+H*actuators);
plot (new_offsets(1:77),new_offsets(78:154),'o');
hold on;
plot (offsets(1:77),offsets(78:154),'.');
axis ([1 618 1 494]);
hold off;
title ('second iteration');
```

7.2. *Iter*

```

%
% FUNCTION - iter
%
% USAGE: [pv,mse,new_actuators,new_image] =
%         iter (actuators,filter,Offsets,Hmat,Gmat,blob,ref_image,image)
%
% PURPOSE: iterate the low resolution spot estimation algorithm and
%           compute both the mean square error and peak to valley
%           errors in the spots
%
function [pv,mse,acts,high] = iter
(acts,fil,offsets,H,G,blob,low_ref,high);

new_offsets = offsets + H*acts;
xtilt = mean (new_offsets(1:77) - offsets(1:77)) * ones(1,77);
ytilt = mean (new_offsets(78:154) - offsets(78:154)) * ones(1,77);
mse(1) = mean( (new_offsets-offsets - [xtilt ytilt]').^2 );
pv(1) = max( abs(new_offsets-offsets - [xtilt ytilt]') );
%
% calculate the estimated spot positions based
% on either the given image (high) or actuator positions
%
if nargin < 8
    [new_offsets,low] = try(acts,fil,offsets,H,blob,low_ref);
else
    [new_offsets,low] = estimate(fil,offsets,low_ref,high);
end;
%
% calculate the actuator movement required to correct
% the estimated image
% find the resulting image
%
acts = acts - 0.5*(G'*(new_offsets-offsets));
high = mk_image (acts,offsets,H,blob);
new_offsets = offsets + H*acts;
xtilt = mean (new_offsets(1:77) - offsets(1:77)) * ones(1,77);
ytilt = mean (new_offsets(78:154) - offsets(78:154)) * ones(1,77);
mse(2) = mean( (new_offsets-offsets - [xtilt ytilt]').^2 );
pv(2) = max( abs(new_offsets-offsets - [xtilt ytilt]') );
%
% repeat
%
for i = 1:6
    [new_offsets,low] = try(acts,fil,offsets,H,blob,low_ref);
    acts = acts - 0.5*(G'*(new_offsets-offsets));
    high = mk_image (acts,offsets,H,blob);
    new_offsets = offsets + H*acts;
    xtilt = mean (new_offsets(1:77) - offsets(1:77)) * ones(1,77);
    ytilt = mean (new_offsets(78:154) - offsets(78:154)) * ones(1,77);
    mse(2+i) = mean( (new_offsets-offsets - [xtilt ytilt]').^2 );
    pv(2+i) = max( abs(new_offsets-offsets - [xtilt ytilt]') );
end;

```

7.3. *Estimate*

```
%  
% FUNCTION - estimate  
%  
% USAGE: [new_offsets,low_image] =  
%         estimate (filter,offsets,reference,image)  
%  
% PURPOSE: estimate spot locations and produce a low resolution  
%           Hartmann image, give a high resolution image  
%  
  
function [new_offsets,low] = estimate (fil,offsets,low_ref,high);  
  
%  
% filter and downsample to create the low resolution image  
% subtract the reference image (this compensates for  
% darker areas around the edges).  
%  
low = high;  
for j = 1:5  
    low = conv2 (fil,fil,low,'full');  
    low = down(low);  
end;  
low = low - low_ref;  
%  
% estimate the spot positions from the low  
% resolution image  
% specifically, weight the surrounding pixels and  
% move the reference spots away from dark areas  
% and toward light areas - note: the algorithm gives  
% less weight to darker areas since they are, essentially,  
% an absence of information  
%  
offsets_low = 4 + ceil (offsets/2^5);  
s = size(low);  
direction_pattern = [-1/200 -1/70 -1/70 -1/50 0 1/50 1/70 1/70 1/200];  
distance_pattern = [1/8 1/2 1/2 1/sqrt(2) 1 1/sqrt(2) 1/2 1/2 1/8];  
for j = 1:s(1)  
    for k = 1:s(2)  
        if (low(j,k) < 0)  
            low(j,k) = low(j,k)/5;  
        end;  
    end;  
end;  
%  
% set up vectors and perform the linear  
% algebra to move the spots  
%  
new_offsets = zeros(size(offsets));  
for j = 1:77
```



```

    x = offsets_low(j);
    y = offsets_low(j+77);
    xrow = zeros(1,s(1));
    xcol = zeros(s(2),1);
    yrow = zeros(1,s(1));
    ycol = zeros(s(2),1);
    xrow(x-4:x+4) = direction_pattern;
    xcol(y-4:y+4) = distance_pattern;
    new_offsets(j) = offsets(j) + (xrow * low * xcol);
    yrow(x-4:x+4) = distance_pattern;
    ycol(y-4:y+4) = direction_pattern;
    new_offsets(j+77) = offsets(j+77) + (yrow * low * ycol);
end;
new_offsets = ceil(new_offsets);

```

7.4. Try

```

%
% FUNCTION - try
%
% USAGE: [new_offsets,new_image] =
%         try (actuators,filter,Offsets,Hmat,blob,reference)
%
% PURPOSE: compute estimated spot locations and a low resolution
%           image, given current actuator positions
%
function [offs,low] = try (actuators,fil,offs,H,blob,low_ref);

if nargin < 3
    %
    % read the reference spot locations
    % 77 x coordinates, followed by 77 y's in "o"
    %
    offsets;
    offs = o;
    clear o;
end;
if nargin < 4
    %
    % read the influence function and convert it to a
    % matrix 154 x 39
    %
    hMat;
    H = convertH(h);
    clear h;
end;
if nargin < 5
    %
    % get a sample focal spot from a real Hartmann image
    %
    i = convImag('image1.dat');
    blob = i(86:120,91:125);

```

```

    clear i;
end;
if nargin < 6
    %
    % create the reference low resolution image from an
    % artificial, reference Hartmann image
    % filter and downsample the image
    %
    low_ref = mk_image(zeros(39,1),offs,H,blob);
    for j = 1:5
        low_ref = conv2 (fil,fil,low_ref,'full');
        low_ref = down(low_ref);
    end;
end;

high = mk_image(actuators,offs,H,blob);
[offs,low] = estimate (fil,offs,low_ref,high);

```

7.5. *Mk_Image*

```

%
% FUNCTION - mk_image
%
% USAGE: image = mk_image (actuators,Offsets,Hmat,blob)
%
% PURPOSE: Simulate a high resolution Hartmann image give
%           current actuator positions
%           This procedure assumes the mirror is the only
%           source of distortion
%
function img = mk_image (actuators,offs,H,blob);

if nargin < 1
    actuators = zeros(39,1);
end;
if nargin < 2
    offsets;
    offs = 0;
    clear o;
end;
if nargin < 3
    hMat;
    H = convertH(h);
    clear h;
end;
if nargin < 4
    i = convImag('image1.dat');
    blob = i(86:120,91:125);
    clear i;
end;

img = zeros([618 494]);

```

Final Project

6/9/98

```
offs = ceil(offs + H*actuators);
for j = 1:77
    tmp = img(offs(j):offs(j)+34,offs(j+77):offs(j+77)+34);
    img(offs(j):offs(j)+34,offs(j+77):offs(j+77)+34) = tmp + blob;
end;
```

7.6. Down

```
%
% FUNCTION - down
%
% USAGE: new = down (old)
%
% PURPOSE: downsample an image
%
function new = down (old);

sz = size(old);
new = old(sz(1):-2:1,sz(2):-2:1);
sz = size(new);
new = new(sz(1):-1:1,sz(2):-1:1);
```

Technical Information Department • Lawrence Livermore National Laboratory
University of California • Livermore, California 94551