



Keck Adaptive Optics Note #364

Implementation and Computational Cost of the Correlation Algorithm

Marcos van Dam and Erik Johansson

Version 1: October 18, 2005

1 Introduction

The computational and memory requirements for the 8x8 correlation algorithm for the NGWFC have not yet been evaluated. In this KAON, we calculate the computation involved and determine the feasibility of the algorithm. The amount of data required to be input to the algorithm is the pixel data for each subaperture, which tensor containing 304x8x8 elements. For the FFT-based correlation algorithm, the data would be complex but only half the plane is required since the reference values are all real.

2 Algorithm and implementation

The correlation of image $i(x,y)$ and reference $r(x,y)$ is taken to give the correlation at an integer number of pixels. Then the value of the correlation is interpolated between pixels to find the sub-pixel shift.

2.1 Brute force approach

Here, the correlation, $c(x,y)$, is evaluated directly.

$$c(x,y) = \sum r(x',y')i(x+x',y+y')$$

This computational approach requires 64 multiplications and 63 additions for each position in the correlation. The full solution space allows a shift range of $[-7, +7]$ for both x and y , giving a result size of 15x15 values. This yields a total of 14,400 multiplies and 14,175 additions. Depending on the compiler optimizations available for the underlying CPU, the multiples and adds may be overlapped to save additional clock cycles.

If the spot size and position are known to be within certain limits, the range of computations may be reduced, resulting in a dramatic drop in the total number of calculations required. For example, if the correlation space is limited to $[+2, -2]$ in x and y , the number of computations drops to 1600 multiplies and 1575 adds.

2.2 Fourier transform approach

The correlation can also be calculated using the discrete Fourier transform. The correlation is given by

$$c(x, y) = F^{-1}[F^*[r(x, y)]F[i(x, y)]],$$

where F denotes the Fourier transform and $*$ denotes the complex conjugate. Since $F^*[r(x, y)]$ can be pre-computed, two Fourier transforms and a dot multiplication (element-by-element) are needed.

The discrete Fourier transform of a 2D array requires $O(2N^3)$ computations, where N is the size of the array. To prevent aliasing in the correlation result, it may be necessary to zero pad the array of intensities. However, there is no evidence that this is needed for the images that we expect to use. For an 8x8 subaperture, a 16x16 zero-padded array must be used. This results in $O(8192)$ complex operations. The full correlation requires two Fourier transforms and one dot multiplication (256 operations). Therefore, the full correlation requires $O(16384) + 256$ complex operations. The $O()$ notation implies “on the order of”, or proportional to within a small scaling constant (<10). The precise value of the proportionality constant depends on the particular low-level CPU implementation of the discrete Fourier transform.

As before, if the spot size and position are known to be within certain limits, it may be possible to skip the zero padding and use an aliased correlation. In this case, the number of operations drops to $O(2048) + 256$ complex operations. The use of the aliased correlation is acceptable if the amount of spot shift relative to the reference lies outside of the aliased region.

2.3 Fast Fourier transform approach

The Fast Fourier transform (FFT) is a very efficient form of the discrete Fourier transform. Instead of $O(2N^3)$ computations as in the case of the discrete Fourier transform, the number of computations drops to $O(2N^2 \log_2 N)$ for a 2D array. Therefore, the total number of computations for an 8x8 subaperture correlation drops to $O(4096) + 256$ complex operations when zero-padding, and $O(1024) + 256$ complex operations without zero-padding. Again, the proportionality constant for the “order of” approximation depends on the particular implementation of the FFT being used and the CPU target.

It should also be noted that some optimized FFT routines have minimum array sizes (i.e., $N > 8$), and that must be taken into consideration as well.

2.4 Interpolation

Having obtained the correlation values, we now need to interpolate to find the best fit with sub-pixel accuracy. The exact algorithm to be used is not yet certain, but the current algorithm implemented for the LBWFS is shown below:

- 1) Find the position to the nearest pixel that results in the maximum correlation value, (x_{\max}, y_{\max}) .
- 2) Find the sub-pixel value, $(\tilde{x}_{\max}, \tilde{y}_{\max})$ using quadratic interpolation of the correlation at integer pixel values:

$$\tilde{x}_{\max} = x_{\max} + \frac{0.5(c(x_{\max} + 1, y_{\max}) - c(x_{\max} - 1, y_{\max}))}{c(x_{\max} + 1, y_{\max}) + c(x_{\max} - 1, y_{\max}) - 2c(x_{\max}, y_{\max})}$$

$$\tilde{y}_{\max} = y_{\max} + \frac{0.5(c(x_{\max}, y_{\max} + 1) - c(x_{\max}, y_{\max} - 1))}{c(x_{\max}, y_{\max} + 1) + c(x_{\max}, y_{\max} - 1) - 2c(x_{\max}, y_{\max})}$$

The total computation for this part of the algorithm has not been calculated, but is significantly less than for the correlation calculation discussed above.

3 Discussion

The algorithms discussed above must be repeated for each subaperture in the WFS pixel image.

The reduced-range brute force method and the FFT-based method (both with and without aliasing) seem to be possible algorithm candidates. If the images need to be zero padded, then the background must be subtracted. There is, however, no evidence that zero padding is needed. If it is not, then a uniform background does not need to be subtracted. The reference images will be computed theoretically initially, and then from the long-term average of the subimages in closed loop. The images need to be updated whenever with approximately the same frequency that a new reconstructor is triggered (not more often than once every 10 seconds).

The effects of aliasing should be evaluated using sample spot patterns on an 8x8 subaperture. An open question is the best interpolation algorithm, both from the point of view of performance and speed.