

# Correlation Algorithm for the NIR TTS project (aka TRICK)

Chris Neyman

February 15, 2011

## 1. Correlation Algorithm

The suggested algorithm for the NIR TTS is a correlation algorithm of the type described by Poyneer in “Scene-based Shack–Hartmann wave-front sensing: analysis and simulation”, Applied Optics, Vol. 42, 2003. See attached copy of the paper as a PDF file. The relevant algorithm is described on pages 5809-5810 of that article. The reference image  $r[i,j]$  is defined as the intensity measured on an array of  $N$  pixels on each side, with the indices spanning the range, 1- $N$ . The measured image is denoted  $s[i,j]$  over the same set of pixels.

The algorithm makes the following assumptions:

- 1) The pixel intensities which are only available for a finite length can be treated as a single period of an infinite periodic signal (similar to the discrete Fourier transform)
- 2) Owing to the periodicity of the signals, as one end of the signal moves away, it wraps around from the other side. The image and reference never leave the array and therefore no zero padding of signals is used. Negative image shifts,  $m$ , are represented by correlation function value at  $N-|m|$ .

Using the above assumptions, the correlation between  $r$  and  $s$  is defined as

$$C[m,n] = \sum_{i=1}^N \sum_{j=1}^N r[i-m, j-n] s[i, j]. \quad (1)$$

This is equivalent to Equation 11 in Poyneer. Note that the correlation can be calculated using pixel values directly as in (1) or using the FFT version of the correlation theorem:

$$C[m,n] = FFT^{-1} \left( FFT(r)^* FFT(s) \right). \quad (2)$$

The superscript -1 denotes the inverse FFT and the asterisk (\*) denotes the conjugation operation.

The location of the image will be given to the nearest pixel as the value of  $m$  and  $n$  that maximizes  $C[m,n]$  in either equation 1 or 2 above. In order to estimate the image location to sub pixel accuracy the discrete values around the peak are used to fit a continuous parabolic function to the discrete values around the peak separately in each axis using only the nearest pixels. Let the index values that correspond to the peak pixel of  $C$  be  $m_0$  and  $n_0$ , the resulting sub pixel estimates are given by

$$\hat{m} = m_0 + \frac{(C(m_0 - 1, n_0) + C(m_0 + 1, n_0))}{2(C(m_0 - 1, n_0) + C(m_0 + 1, n_0) - 2C(m_0, n_0))} \quad (3)$$

and

$$\hat{n} = n_0 + \frac{(C(m_0, n_0 - 1) + C(m_0, n_0 + 1))}{2(C(m_0, n_0 - 1) + C(m_0, n_0 + 1) - 2C(m_0, n_0))}. \quad (4)$$

These equations are written in Poyneer and the NIRTSS/TRICK statement of work using linear coordinates  $x$  and  $y$ . Given a conversion factor between pixels and coordinated these expressions (3 and 4) above are equivalent to equation 13 in the Poyneer's paper. The notion  $C_{-1}$  and  $C_{+1}$  are used by her to denote values of the correlation function that are one pixel above and one pixel below the pixel nearest the peak denoted  $C_0$ .

## 2. MATLAB Example Code

The correlation algorithm was provided by Marcos van Dam as a Yorick source file and converted to MATLAB. Both codes use equation (2) above for simplicity, but equation (1) may be more effective with real-time hardware and small pixels arrays used in NIR TTS.

The flowing MATLAB source files (m-files) are included with this note:

correlation.m: calculates the correlation and sub-pixel parabolic fits  
gauss.m: used to produce reference images and test image for correlation.m  
test.m: test files

The 2 dimensional arrays in MATLAB are denoted: a[rows, columns] to be consistent with mathematical convention used for matrices. This is counter to most array languages like IDL or Yorick that use the convention a[x,y] or a[columns, rows]. The above MATLAB codes use the MATLAB convention. Images of an array, a, display correctly using the following MATLAB statement:

```
imagesc(a), axis xy
```

The correlation code can be tested using the file test.m which contains the following code

```
%  
%  
npix=16;  
refim=gauss(0,0,3,3,16);  
%  
% test image center at [-0.5,0.5], FWHM=3  
%  
[subim2]=gauss(-0.5,0.5,3,3,npix);  
[cc,dispx,dispy]=correlation(subim2,refim,1);
```

The pixel array is 16 by 16 and the reference image is a Gaussian function with FWHM of 3 pixels. The test image is displaced by -0.5 and 0.5 pixels in x(columns) and y(rows) respectively. The full correlation function  $C[m,n]$  is returned as the variable cc.