1. Introduction

1.1 Overview

This project deploys the aspects of image processing and pattern recognition techniques used in Computer Vision. The interpretation of an image offers useful information to analyze complicated visual events (Huang, 1996). Especially, implementation of algorithms which may reproduce natural processes to differentiate significant features for the spatial characterization of surroundings, environment, or objects (Kanellakis & Nikolakopoulos, 2017). Such an approach is called Computer Vision and helps to copy the process by which human eyes assess the spatial depth and unique spatial features from a pixel level. Furthermore, significant information may be resolved using spectral decomposition techniques which exceed the processing capability of natural eyesight.

1.2 Experiment: Cross-Correlation in 1D and 2D

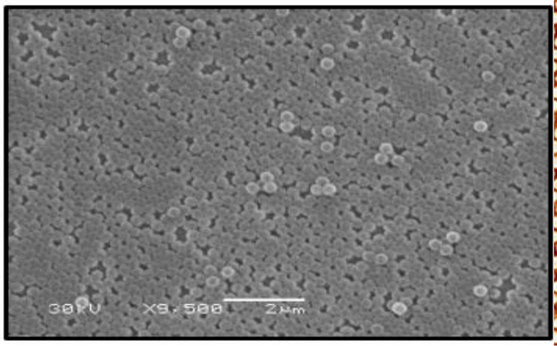
The first set of experiments are executed to investigate and evaluate standard pattern matching approaches using cross-correlation. Cross-correlation (Rao et al., 2014) is widely used in statistical image processing as a method for image analysis. This method is used to evaluate the degree of similarity of multidimensional signals, image patches, features extraction, and pattern matching. Based on simple convolution, calculation of the correlation coefficient is a computationally direct method that benefits from Fourier transformations that can reduce the time complexity of operations. Further, it may be implemented across separate data sets to find an unbiased measure of similarity. Cross-correlation is often combined with other image manipulation methods such as filtering to improve the resolution of the correlation coefficients.

1.3 Experiment: Patterns in Stereovision

The second set of experiments are to investigate the parallax nature of recorded real-world images when digitized. A depth map is a 3D reconstruction of an image from the binocular disparity map of a set of images, or in some advanced implementations; a monocular disparity map. Real-world intensity and position are approximated from the perspective projection using distance calculations to pixel coordinates.

1.4 Application: Moth-eye SEM image analysis

Eventually, this project aims to extend the investigated cross-correlation and Computer Vision methods to analyze experimental data from a Scanning Electron Microscope (SEM) images. Moth-eye surfaces are nano-scale surfaces, because of the arrangement of the particles the surface takes on antireflective properties. Thus, the main interest is to investigate a rapid way to estimate and characterize features.



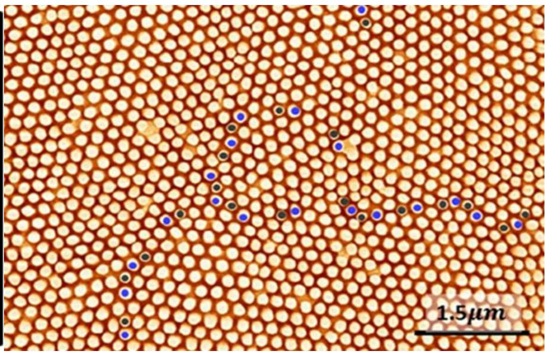


Figure 1: *Moth eye*

Figure 2: *Textured Diamond*

Images are processed and then analyzed using Fourier Transform methods and filtering. The analysis will implement correlation to determine regions where there are grain non-uniformities and defects in the ordering and to extract information relating to irregularities of the surfaces. (Varija Raghu & Thamankar, 2020).

2. Theory

A simple way of determining the relationship between two separate datasets is by comparison of patterns with template data. Particularly, by a method of single or multi-element template matching via correlation calculation. Correlation, also called covariance, is a measure of the strength of a linear relationship between two or more quantitative variables. Computationally, two classical approaches to find the cross-correlation, spatial, and spectral are implemented in this project.

2.1 Pattern matching

2.1.1 Cross-correlation

Statistically, cross-correlation provides the degree of strength of a linear relationship between two or more quantitative variables (Derrick & Thomas, 2004). The correlation value may be determined from a normalized weighting of the data set. In general, the value of the cross-correlation is determined by eq1 and can be generalized for higher dimensions.

$$c(s) = \sum_{n=-\infty}^{\infty} f1 * [n] + f2[n+s] \qquad [1]$$

Where f1 and f2 are the two different data sets, n is a discrete element within the data, is the shifted position, and c(s) is the correlation coefficient. The effect of the offset position is visualized below (figure 3). The position of greatest correlation appears as a peak in the graph of correlation coefficient plotted against shift position (figure 4).

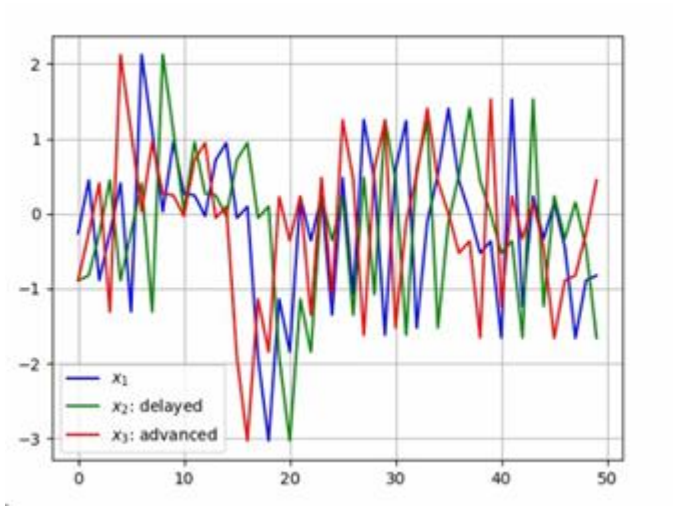


Figure 3: Effect of the offset position

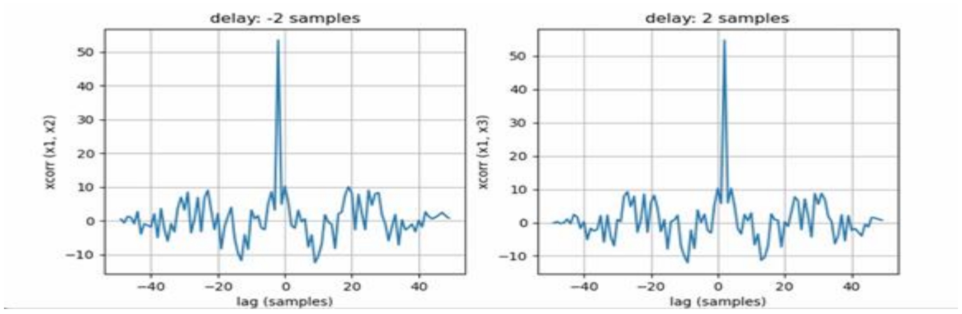


Figure 4: Correlation coefficient vs. Shift position

2.1.2 Normalized cross-correlation

The dependence of direct correlation was removed by normalizing the correlation coefficient, referred to as the normalized cross-correlation. Normalized cross-correlation is used to evaluate the degree of similarity between any two data sets. It has the advantage of less sensitivity to linear changes in the amplitude of the quantitative variables under comparison (Varija Raghu & Thamankar, 2020). The normalized cross-correlation is divided by the product of the autocorrelation magnitude as this is the largest correlation value in the data we have.

$$NCC(h,k) = \frac{\sum_{i,j}\left(S(i,j) - M_S\right)\left(\left(L(i+h,j+k) - M_L\right)\right)}{\left\{\sum_{i,j}\left(S(i,j) - M_S\right)^2 \sum_{i,j}\left(L(i+h,j+k) - M_L\right)^2\right\}^{0.5}}.$$

[2]

The Normalized Cross-Correlation (NCC) is confined to a range from the addition of the denominator so that the resulting correlation ranges from -1 to 1. A perfect match has a value of 1. ML(h,k) is the mean of the subsection of the large image at offset (h,k), N is the number of pixels in the small image and NCC(h,k) is the normalized cross-correlation metric at offset (h,k).

2.1.3 Spatial

It is a measure of the strength of the relationship between two temporal data sets. This method is useful for finding the converging pattern over a periodic signal covered by noise. Specifically, this is implemented for two signal files that are analyzed in this section. In spatial cross-correlation, a weighting is needed to be derived at each alignment. This is computationally intensive as the time complexity for this process is N^2 where N is the input size of the larger array. This process of cross-correlation can be performed more efficiently for large inputs by transforming the inputs into the frequency domain.

2.1.4 Spectral

This approach does not use normalization and directly computes the simple cross-correlation, C(i,j), using Forward and Inverse Fourier transforms. A basic principle of Fourier Transforms is that convolution in the spatial domain is equivalent to multiplication in the frequency domain. Likewise, correlation in the spatial domain is equivalent to multiplication in the frequency domain using the complex conjugate of one of the transformed arrays. Thus, either convolution or cross-correlation can be used to perform spectral cross-correlation. The cross-correlation of f(t) and g(t) will have the same output as the convolution of f(t) and g(t) but reversed. Since time series data is sample discretely, a summation analog of the continuous Fourier transform is used:

The Discrete Fourier Transform can be expressed as

$$\mathbf{F}(n) = \sum_{k=0}^{N-1} f(k)\, e^{-j2\pi nk/N} \qquad (n = 0..1\ :N-1)$$

[3]

In practice, the discrete Fourier transform (DFT) is implemented typically using the Fast Fourier Transform (FFT).

In contrast to spatial correlation, the normalized cross-correlation does not have a minimum frequency domain expression. It cannot be directly computed using the more efficient FFT in the spectral domain. Its computation time increases dramatically as the window size of the template gets corpulent.

2.2 Stereo Vision

2.2.1 Depth mapping

Stereovision aims to reproduce the depth determination capability of human vision. A depth map records the distance information concerning a viewpoint in a 2D image of digitized real-world coordinates (Zhang et al., 2009). The proposed algorithm uses two images of the same scene. Pixel to pixel matching is used between two images, left and right, to find the corresponding point in the same scene from different views. Given the geometric description of the location of the camera, separations, and pixel location, a depth map may be created using a method binocular disparity.

The disparity value of a point is often interpreted as the inverse distances to the observed objects. Thus, the disparity is inversely proportional to Depth. Darker regions in the Depth-Map are created for signifying that an object is far away, and this darker color gradually decreases to brighter with a decrease in depth and finally becomes white for closer objects.

2.3 Practical Considerations

2.3.1 Data set Size

The efficiency of a cross-correlation algorithm will vary with the size of the data being compared. In the case of image data, which is the primary focus of the project, the degree of computational complexity will depend upon image size and resolution. The nature of image data is such that, a digital image a[m, n], described in a 2D discrete space is derived from an analog image a(x, y), in a 2D continuous space through a sampling process. The process is often referred to as digitization. The digitization relates image data to the pixel location. The data sets used in this study consider one- and two-dimensional analog information that is already pre-digitized in text or image-based formats.

2.3.2 Time complexity

Direct calculation of NCC is computationally intensive and the time cost is proportional to the region size (Yoo et al., 2010). Assuming the average size of the matching regions is R, the computational complexity to match two images with an image size of M and disparity range of D is O (M ×R× D). Since M, R and D are usually large. Since stereo matching usually needs to match each pixel across a disparity space for a whole image, the computational efficiency of the pixel-wise similarity is necessary for a fast depth determination.

2.3.2.1 Spatial

The spatial cross-correlation method considered in this project utilizes convolution to relate the elements of the two compared data sets. The computational complexity for an N×M convolution kernel implemented in the spatial domain on an image of N × M is O (N2) where the complexity is measured per pixel based on the number of multiplies-and-additions

2.3.2.2 Spectral

The time upper bound time complexity and operation counts are O (N log(N)). Thus the FFT is an optimized algorithm that reduces the time complexity of the DFT from N^4 to NlogN. [6] The computational complexity per pixel of the Direct Fourier approach for an image of $N^2 \times N^2$ and a convolution kernel of N × N is O (log N) complex Multiplication and Additions independent of N.

3. Experiments

3.1 Cross-correlation in 1D and 2D

3.1.1 1D Signal Offset

Time Series

In time series analysis, cross-correlation produces a measure of the temporal similarity of the data sets. To remove noisy signals and synchronize signals, a correlogram plot of the cross-correlation values at each lag is useful. Cross-correlations between the signals will be on the peak at the lag in which the signals converge. The delay is determined through the cross-correlation of the signals. Cross-correlations between the signals will peak at the lag in which the signals converge. The peak cross-correlation value occurs at the lag where the signals have the greatest similarity. Figure (5) shows the correlogram between the two signals with what is likely random noise introduced, possibly from the environment or recording equipment. The noise obscures the pattern of the signal, but at zero lag, the correlogram demonstrates a distinct peak cross-correlation. This means that the two signals are synchronized.

1D Signal Application /data

Discrete-time series data is used in this experiment (Proakis et al., 2004). state that 'n' represents the index of the discrete-time instants as the independent variable. This allows the signal to become a function of an integer variable. The signals used in the 1D case are analog signals, with a constant sample time 't', the time between successive samples. The reciprocal of the sampling period is known as the sampling rate Fs, which represents the samples per second. The signal data is stored in a .txt file, (UTF-8 encoding) with a data point separated by a carriage return operator.

Analysis – temporal

The theory outline in Chapter 2 is applied to temporal 1D cross-correlation. The selected pattern and template signals are stored separately as lists and passed into the *find_offset* function. *find_offset* examines the shift by calculating the index of the maximum value determined from a list of cross-correlation coefficients. To determine the cross-correlation coefficients, the template array is zero-padded with the length of the pattern, [array -1] at both ends.

Then the result of the convergence between the pattern array and non-padding elements of template array for an iterated offset position of the index. For the two-signal data, the offset of the maximum cross-correlation value specifies the number of sample periods between the beginning of each signal file and the maximum convergence. The corresponding cross-correlation function will show a maximum value at the index where the convergence is most congruent.

$$r = \frac{1}{N}\sum_{i=1}^{i=N}(f(i)-\overline{f})(g(i)-\overline{g}) \quad [4]$$

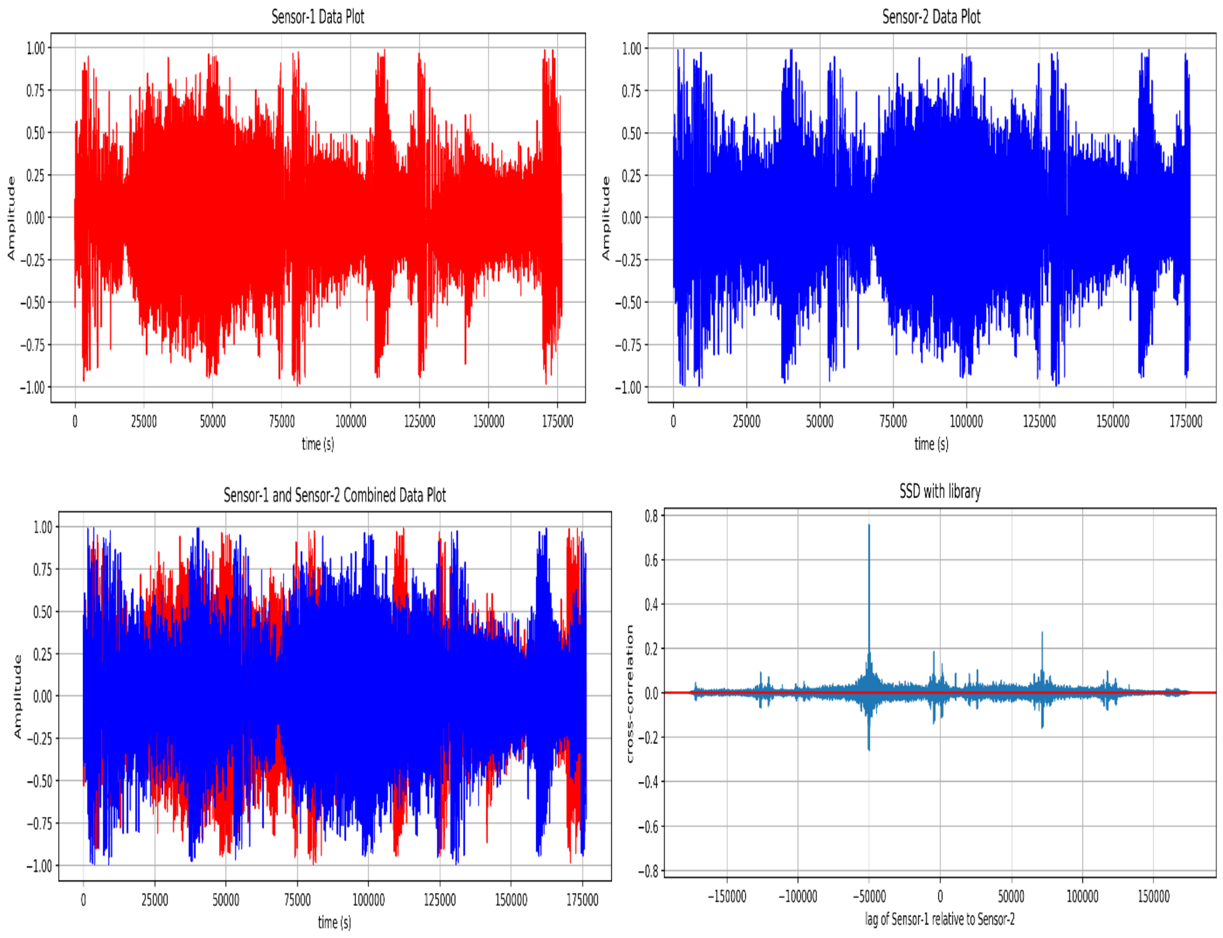


Figure 5: Cross correlation between sensor-1 & sensor-2 data

Analysis - Spectral

To improve the performance, spectral cross-correlation is employed to reduce the computational time. The code implements the spectral decomposition of the two signal files as described above, to determine the cross-correlation coefficient and associated lag. The low pass filter removes the high amplitude low-frequency noise and sensor-related noise. The related matrices are Hermitian because signal data has real values. So, the frequency spectrum is consequently symmetric and may be mirrored or removed above the Nyquist frequency of Fsample/2 and filtered using a low pass filter.

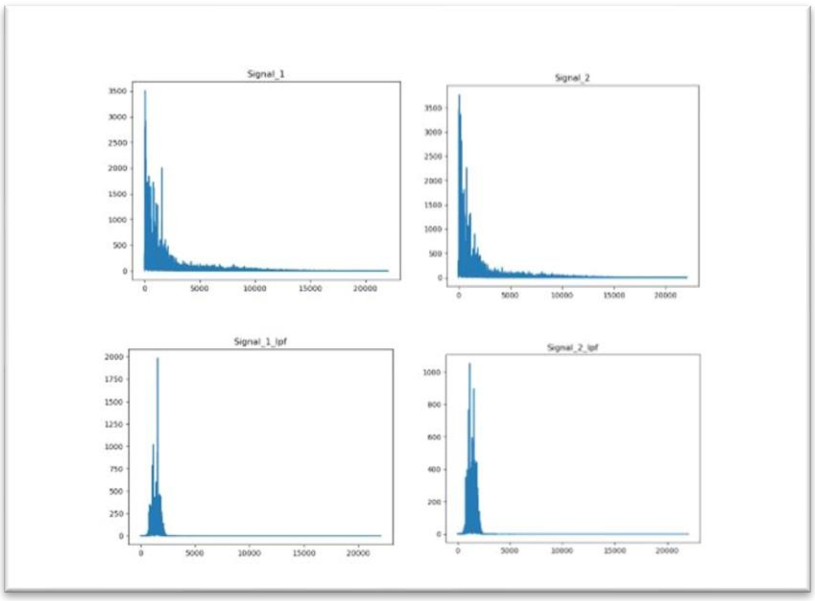


Figure 6

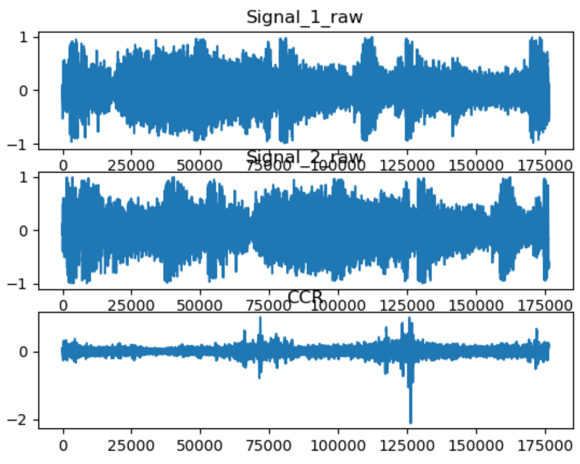


Figure 7

Evaluation – Temporal

For fast implementation, the pattern and template lists are converted into NumPy arrays inside the *n_corr* function. This implementation is very slow due to the use of for loops in the *n_corr* and *calculate_energy* functions. These enforce a minimum time complexity of N^2 or more than this. To reduce the calculation time, cached values are used for the determination of the sum of the squared elements of the *pattern* array. Initially, the calculation was performed for every time, the *calculate_energy* function was called *i* many times, it means iterate over the size of the correlation array, i.e. “template_padded – pattern”. Caching the value reduced the loop time for *n_corr* by half. However, the calculation time was still .5 sec per loop in *n_corr*. The computation time was significantly reduced due to the zeros that are multiplied each time the pattern and template are passed.

The calculation was compared to the library implementation of the cross-correlation using *NumPy.correlate.* These changes resulted in a 64% decrease in the run time of the *n_corr* loop but at the cost of proper normalization techniques. The offset was still calculated to be 389m but given a significant speed decrease, and the loss of proper normalization, this method was abandoned.

Evaluation - Spectral

To improve the performance, spectral cross-correlation was then applied to decrease the run time. Spectral cross-correlation initially took 123 sec to perform. This technique is much faster than temporal cross-correlation because NumPy.fft uses the Cooley-Tukey algorithm to calculate the Fourier transform. As it is a divide-and-conquer algorithm, it runs most optimally on inputs of a power of two in size and least with sizes or prime numbers. The signal lengths are prime numbers, and so by padding these inputs with zeros to the closest power of two, the run time was remarkably faster. The computational time of spectral cross-correlation compared to the Temporal cross-correlation was noticeably faster. This is due to the; Temporal cross-correlation must repeat the same operations for each slice

Spectral Cross-correlation	Temporal Cross-correlation
• Off-Set = -50081 • Off-Set Time = -1.136 • Distance between two sensors = 378.16 meters	• Off-Set = -50082 • Off-Set Time = -1.136 • Distance between two sensors = 378.17 meters

3.1.2 2D Pattern Finding

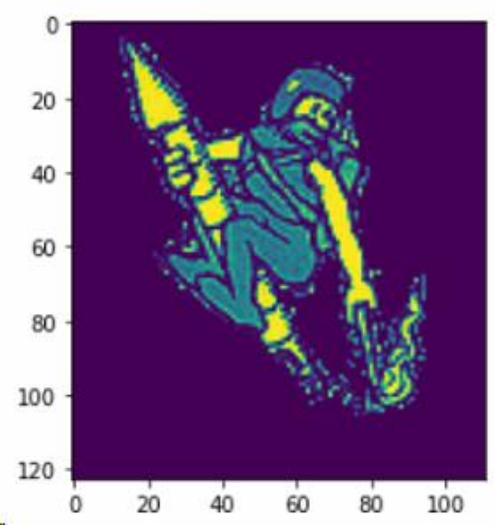
Vision functions, such as stereo and motion estimation, involve the finding of matching features through two or more views. Is a "search" problem, a correspondence issue in a pattern matching technique. Given an element in the left image, search for the corresponding element in the right image, we must need geometric constraints to reduce the size of the search space.

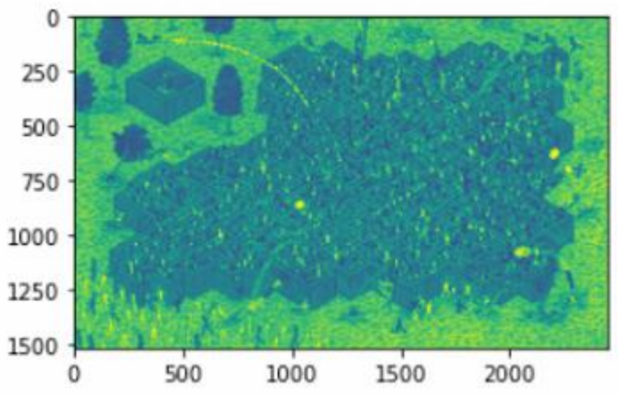
To solve this issue we must choose a similarity measure to compare elements. A simple way of determining the relationships between images is by comparison of patterns with template data. In this experiment, a method of pixel-based template matching over correlation calculation is used.

Test/Mean shift

Shifting the value of the pixel intensity spectrum affects the improvements of the cross-correlation algorithm accuracy and efficiency. Related to mean-shifted and non-mean shifted, the mean shifted speed and accuracy were found to perform better. This was apparent in the case of both the spatial cross-correlation and spectral convolution approaches. In the case of the spatial correlation, the speed up was 40 seconds, at the same time slightly more efficient 0.022s for the spectral approach. (figure). The effect of the zero-mean is to redistribute the intensity over a positive and negative range. In the case of a greyscale image, ranging from black to white, the dark regions of the images will now have a much weaker correlation to brighter regions of equal amplitude, but with opposite sign.

In this experiment, a "rocketman' was searched in a larger template image to investigate 2D cross-correlation techniques.





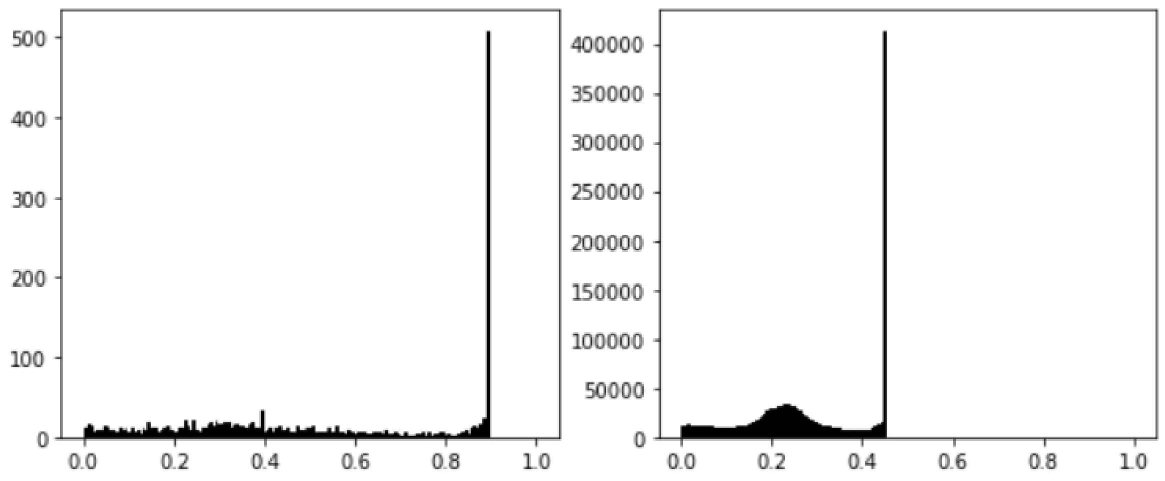


Figure 8: rocket man spatial and spectral cross-correlation

Application/Data

The pattern and template image data are stored in a tensor-like array of intensities over three color channels, (RGB) and one transparency channel (A). Each channel list represents a pixel with three components of color, stored in 8 bits as the Human eye purportedly can only see 8 bits of color, ranging from 0 to 255. To calculate the scalar cross-correlation value, the images are converted into greyscale by locating the correlative intensity of grey from a color table . The functions m*atplotlib.image* and convert_grey*()* rescale the UTF-8 data from each channel to float32 values between 0.0 and 1.0 as the Human eye purportedly can only see 8 bits of color.

Spatial - Analysis

The method to discover the position of a pattern image inside a larger target image can be performed by extending the method of the 1D spatial correlation. In 1D spatial convolution, the template image is padded at either end of its coordinated axes with the length of the pattern -1. The cross-correlation is then found by *n_corr2d* by sliding the pattern across the padded template and finding the sum product of every instance of the converged 2D arrays over the non-padded region of the template image.

The *find_offset* function will then return the coordinates which correspond to the largest correlation value given by the position of the top left corner of the pattern image array. The pattern image is found within the template image with the top left corner coordinate of (x,y) = (529,983).

Spectrum - Analysis

In this simple cross-correlation, the Fourier transform procedure is as follows. The images are read and converted into greyscale as in the spatial cross-correlation script. Within *ccr_2d* the smaller pattern image is padded with zeros at the bottom and right sides to reshape it to the size of the larger template image. The FFT is then applied to both images, along with rows and columns. In this way, the information about the neighbor elements as well as the frequency is transformed twice. Thus, a full spatial description in the 2D array of the image is found for any given channel. The phase information is removed by taking the complex conjugate of one of the transformed matrices before evaluating their product.

The transformed arrays are multiplied element-wise. At last, the inverse Fourier transform is applied to determine the convolution of the two images in the spatial domain. This process is much faster than doing the un-normalized correlation in the spatial domain. In most implementations, the Fourier image is shifted in such a way that the DC-value (i.e. the image mean) F(0,0) is displayed in the center of the image. The further away from the center an image point is, the higher is its corresponding frequency.

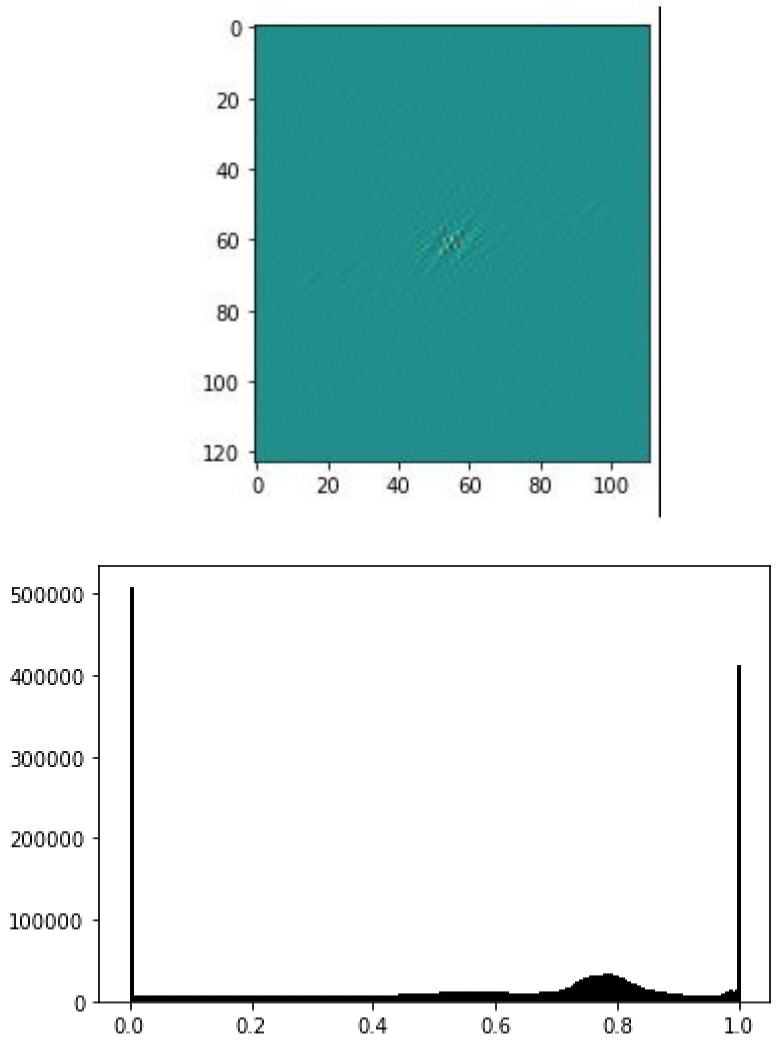


Figure 9

Spatial – Evaluation

- Mostly, the Normalized cross-correlation of image searching is computational slow. The calculation time for this implementation was 40.2 sec.
- Normalized cross-correlation, as described by equation (2), is computationally exhaustive and slow. The difficult task is to evaluate the numerator correlation in the spatial domain when the template image is large. The other aspect that adds to the complexity is the computation of the mean and standard deviation of each subsection of the larger image.
- Improvements can be made by using more robust algorithms that take statistical samples of the template image region to reduce the size of the search region (WEINHAUS, 2014).

Spectral – Evaluation

The method is much faster than spatial correlation since the number of operations is N^2logN not N^4. This is because of the implementation of the FFT. If the DFT were used, it would require a double sum over the indices for each offset position. This is for (RGB) color images, so will take FFT three times.

3.1.3 Conclusion

A 'rocket man' was located within a complex image to demonstrate the accuracy of these techniques and the speed of spectral cross-correlation. This can be seen in figure 8. Both spatial and spectral cross-correlation correctly located the rocket man within the search area, but there was a substantial difference in runtime. The spectral cross-correlation took less than one second. As both techniques produced identical results, spectral cross-correlation was chosen for all remaining alignment tasks in this document.

References

Proakis, J. G., & Manolakis, D. G. (2004). Digital signal processing. *PHI Publication: New Delhi, India.*

Derrick, T. R., & Thomas, J. M. (2004). Time series analysis: the cross-correlation function.

Yoo, J.-C., Choi, B. D., & Choi, H.-K. (2010). 1-D fast normalized cross-correlation using additions. *Digital Signal Processing, 20*(5), 1482-1493.

Rao, Y. R., Prathapani, N., & Nagabhooshanam, E. (2014). Application of normalized cross correlation to image registration. *International Journal of Research in Engineering and Technology, 3*(5), 12-16.

Zhang, K., Lu, J., Lafruit, G., Lauwereins, R., & Van Gool, L. (2009). *Robust stereo matching with fast normalized cross-correlation over shape-adaptive regions.* Paper presented at the 2009 16th IEEE International Conference on Image Processing (ICIP).

Huang, T. (1996). Computer vision: Evolution and promise.

Kanellakis, C., & Nikolakopoulos, G. (2017). Survey on computer vision for UAVs: Current developments and trends. *Journal of Intelligent & Robotic Systems, 87*(1), 141-168.

Varija Raghu, S., & Thamankar, R. (2020). A Comparative Study of Crystallography and Defect Structure of Corneal Nipple Array in Daphnis nerii Moth and Papilio polytes Butterfly Eye. *ACS omega*, *5*(37), 23662-23671.

WEINHAUS, F. (2014). ACCELERATED TEMPLATE MATCHING USING LOCAL STATISTICS AND FOURIER TRANSFORMS.