normxcorr2

Normalized 2-D cross-correlation

Syntax

```
C = normxcorr2(template, A)
gpuarrayC = normxcorr2(gpuarrayTemplate, gpuarrayA)
```

Description

`C = normxcorr2(template, A)` computes the normalized cross-correlation of the matrices `template` and `A`. The matrix `A` must be larger than the matrix `template` for the normalization to be meaningful. The values of `template` cannot all be the same. The resulting matrix `C` contains the correlation coefficients, which can range in value from -1.0 to 1.0.

`gpuarrayC = normxcorr2(gpuarrayTemplate, gpuarrayA)` performs the normalized cross-correlation operation on a GPU.

Class Support

The input matrices `template` and `A` can be numeric. The output matrix `C` is `double`.

The input matrices `gpuarrayTemplate` and `gpuarrayA` are gpuArrays whose underlying type must be numeric. The output matrix `gpuarrayC` is a gpuArray whose underlying class must be `double`.

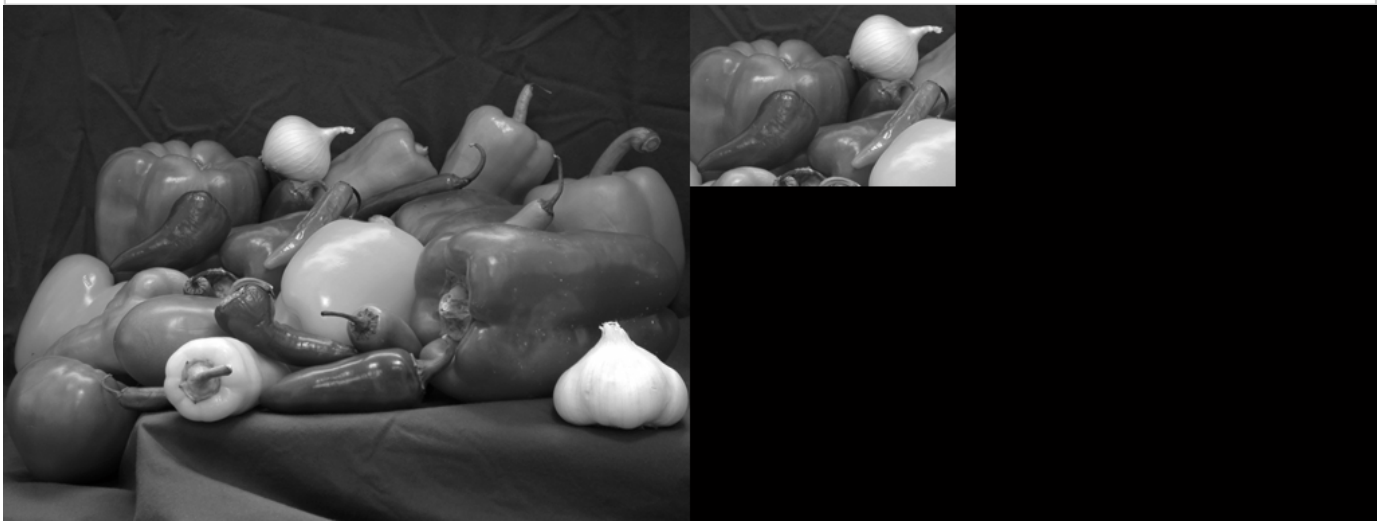
Examples

collapse all

Use cross-correlation to find template in image

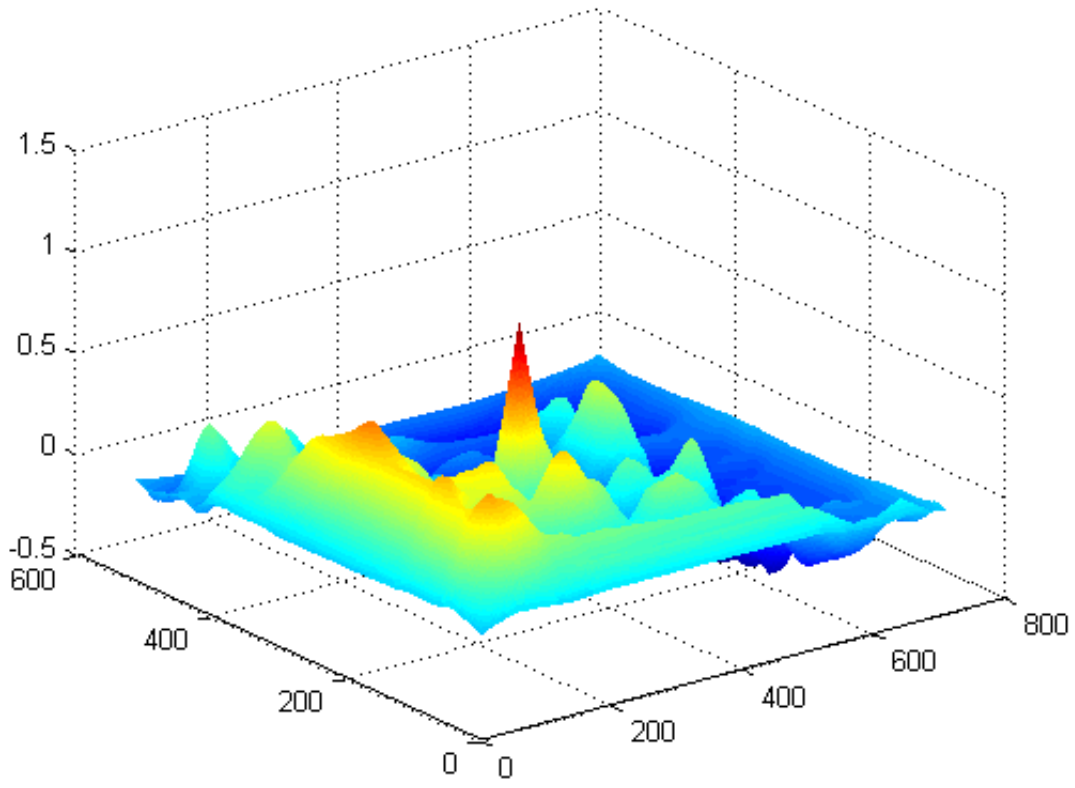
Read images and display them side-by-side.

```
onion   = rgb2gray(imread('onion.png'));
peppers = rgb2gray(imread('peppers.png'));
imshowpair(peppers,onion,'montage')
```



Perform cross-correlation and display result as surface.

```
c = normxcorr2(onion,peppers);
figure, surf(c), shading flat
```



Find peak in cross-correlation.

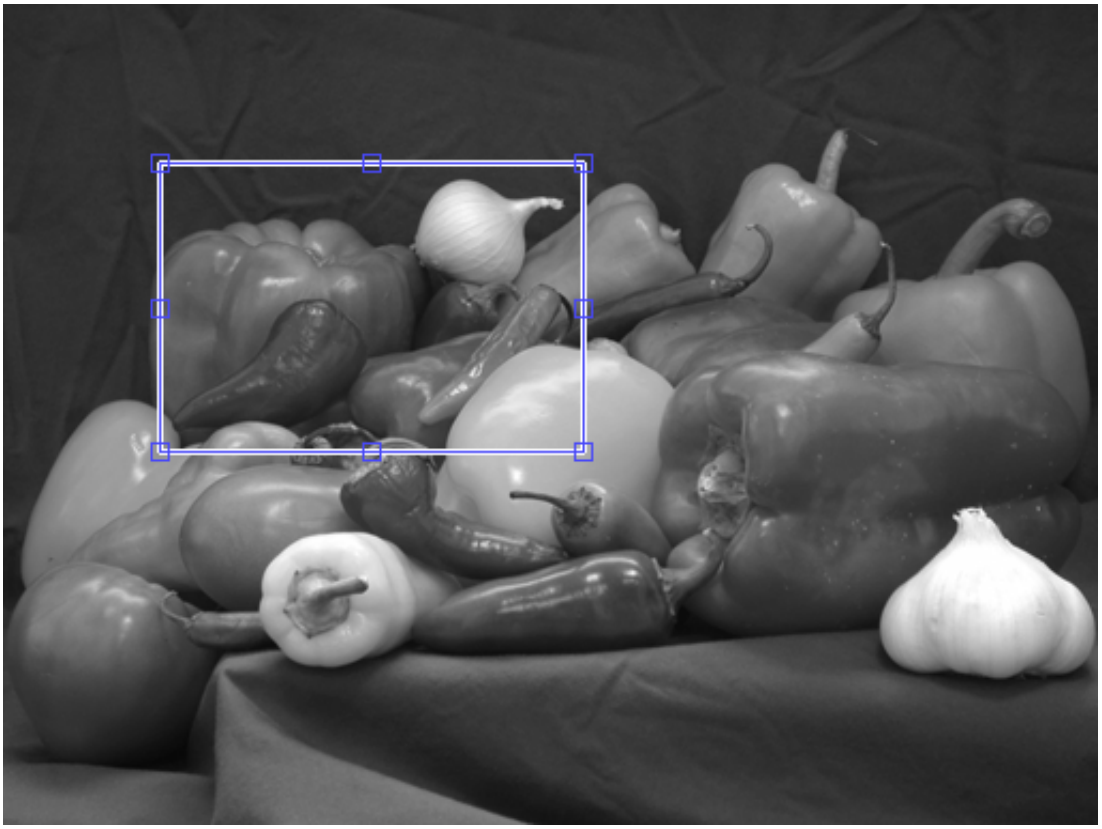
```
[ypeak, xpeak] = find(c==max(c(:)));
```

Account for the padding that `normxcorr2` adds.

```
yoffSet = ypeak-size(onion,1);
xoffSet = xpeak-size(onion,2);
```

Display matched area.

```
hFig = figure;
hAx  = axes;
imshow(peppers,'Parent', hAx);
imrect(hAx, [xoffSet, yoffSet, size(onion,2), size(onion,1)]);
```



Use cross-correlation to find template in image on a GPU

Read images into gpuArrays.

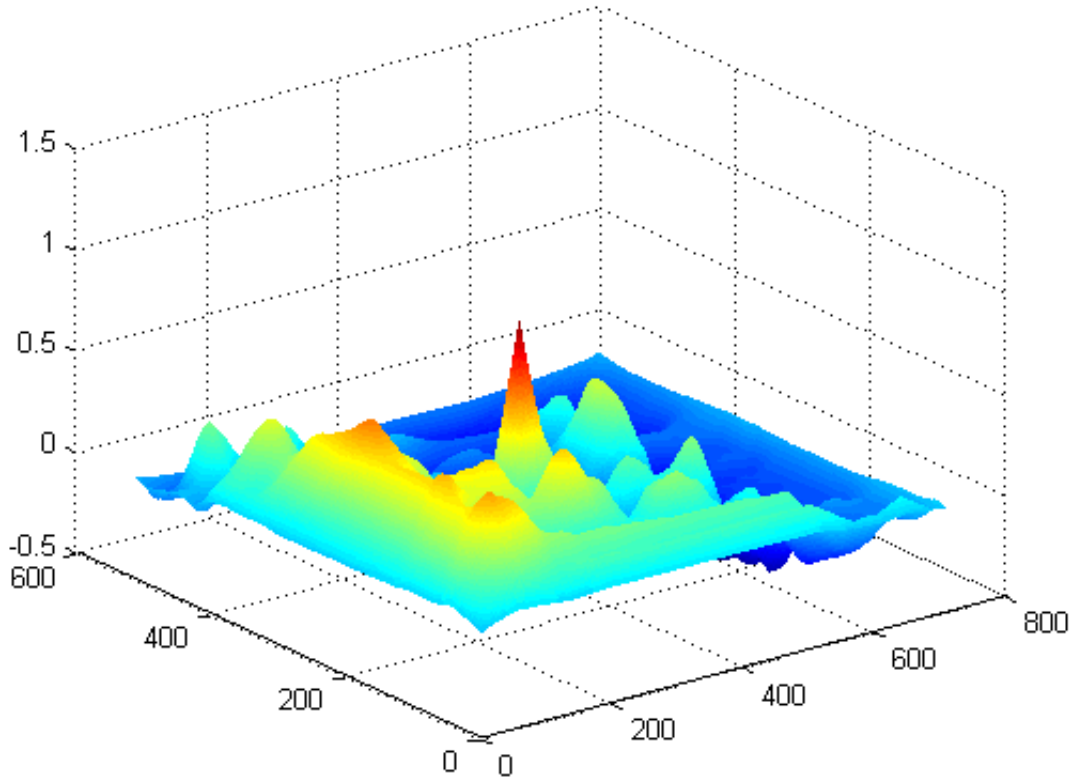
```
onion   = gpuArray(imread('onion.png'));
peppers = gpuArray(imread('peppers.png'));
```

Convert the color images to 2-D. The `rgb2gray` function accepts gpuArrays.

```
onion   = rgb2gray(onion);
peppers = rgb2gray(peppers);
```

Perform cross-correlation and display result as surface.

```
c = normxcorr2(onion,peppers);
figure, surf(c), shading flat
```



Find peak in cross-correlation.

```
[ypeak, xpeak] = find(c==max(c(:)));
```

Account for the padding that `normxcorr2` adds.

```
yoffSet = ypeak-size(onion,1);
xoffSet = xpeak-size(onion,2);
```

Move data back to CPU for display.

```
yoffSet = gather(ypeak-size(onion,1));
xoffSet = gather(xpeak-size(onion,2));
```

Display matched area.

```
hFig = figure;
hAx  = axes;
imshow(peppers,'Parent', hAx);
imrect(hAx, [xoffSet, yoffSet, size(onion,2), size(onion,1)]);
```

More About

expand all

Tips

Normalized cross-correlation is an undefined operation in regions where `A` has zero variance over the full extent of the template. In these regions, we assign correlation coefficients of zero to the output `C`.

Algorithms

`normxcorr2` uses the following general procedure [1], [2]:

1. Calculate cross-correlation in the spatial or the frequency domain, depending on size of images.
2. Calculate local sums by precomputing running sums. [1]
3. Use local sums to normalize the cross-correlation to get correlation coefficients.

The implementation closely follows following formula from [1]:

$$\gamma(u,v) = \frac{\sum_{x,y}\left[f(x,y) - \overline{f}_{u,v}\right]\left[t(x-u,y-v) - \overline{t}\right]}{\left\{\sum_{x,y}\left[f(x,y) - \overline{f}_{u,v}\right]^2 \sum_{x,y}\left[t(x-u,y-v) - \overline{t}\right]^2\right\}^{0.5}}$$

where

- f is the image.
- $\overline{t}$ is the mean of the template
- $\overline{f}_{u,v}$ is the mean of $f(x, y)$ in the region under the template.

References

[1] Lewis, J. P., "Fast Normalized Cross-Correlation," Industrial Light & Magic

[2] Haralick, Robert M., and Linda G. Shapiro, *Computer and Robot Vision*, Volume II, Addison-Wesley, 1992, pp. 316-317.

See Also

`corrcoef`

Introduced before R2006a