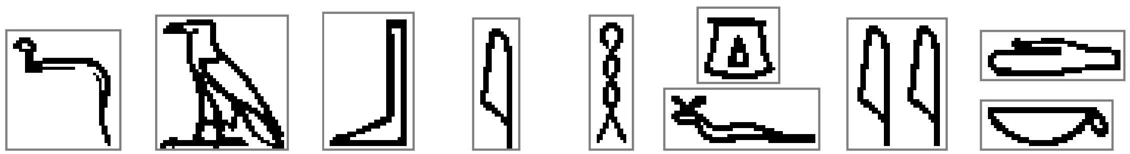
The gradient operator ∇

Outline

- Compute (x,y) and (M,θ) components of the gradient of a smoothed image
- Discretize the orientation values and compute a histogram of gradient magnitude (HoGM)
- To avoid complete loss of spatial information the HoGM should be extracted from different regions of the image, separately
- Use the HoGM extracted from a grid of rectangual image regions to represent and match the Hieroglyphs dataset



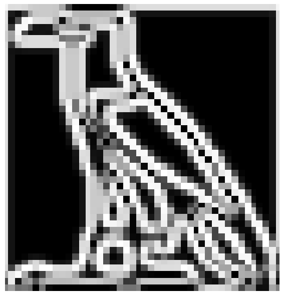
Basic software modules

- A procedure to compute the (M,θ) representation of the gradient from its (x,y) representation
- A procedure that, given the number of orientation bins and one (M,θ) gradient value, returns the code of the corresponding orientation bin
- A procedure to compute the histogram of gradient orientation values from one rectangular region in the image (rowMin, colMin, rowMax, colMax)
- A procedure to partition the image into a regular grid of [n m] rectangular regions and compute (rowMin, colMin, rowMax, colMax) for a generic region

Gaussgradient

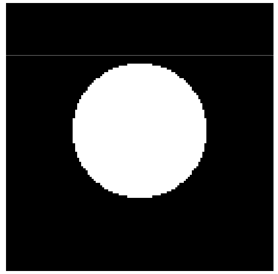
- The procedure **gaussgradient**(img, sigma) returns the x and y components of the image img convolved with a gradient of Gaussian (along x and y direction) with variance sigma

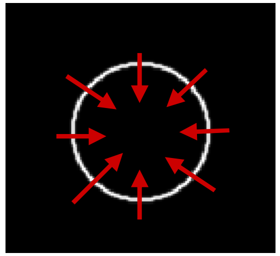
```
img = imread('hiero_01/01.png');

[imgDx imgDy] = gaussgradient(img, 0.7);

imgMag = sqrt( imgDx .* imgDx + imgDy .* imgDy );
imgTheta = atan2( imgDy, imgDx );

figure; imshow( double(imgMag) ./ max(imgMag(:)) );
```



Quantize the orientation

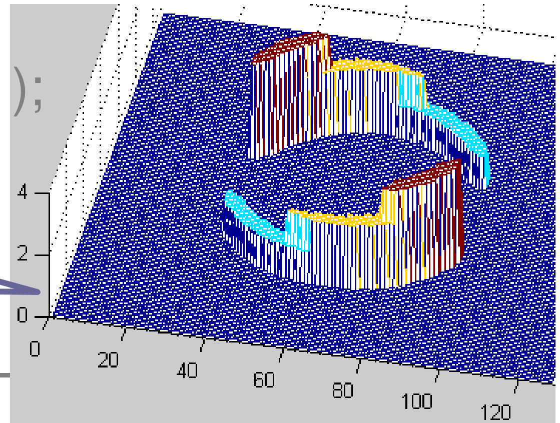
- We need a procedure to quantize values of the gradient orientation (-180, 180] into N bins, starting from the angle 0°





```
img = imread('hiero_01/01.png');

[imgDx imgDy] = gaussgradient(img, 0.7);

imgMag = sqrt( imgDx .* imgDx + imgDy .* imgDy );
imgTheta = atan2( imgDy, imgDx );

figure; imshow( double(imgMag) ./ max(imgMag(:)) );

imgQTheta = quantizeOrientation(imgTheta, 5);
```

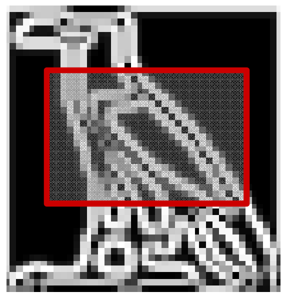
Consider only positive orientation values

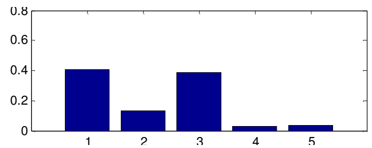


Compute the histogram

- We need a procedure to compute, for each quantized orientation value θ the sum of gradient magnitude values alligned to θ
- We want this procedure to compute the histogram over a rectangular region of the image

```
img = imread('hiero_01/01.png');

[imgDx imgDy] = gaussgradient(img, 0.7);
imgMag = sqrt( imgDx .* imgDx + imgDy .* imgDy );
imgTheta = atan2( imgDy, imgDx );

imgQTheta = quantizeOrientation(imgTheta, 5);

gradHisto = gradientHistogram(imgMag, imgQTheta, ...
                     rMin, cMin, rMax, cMax);
bar(gradHisto);
```

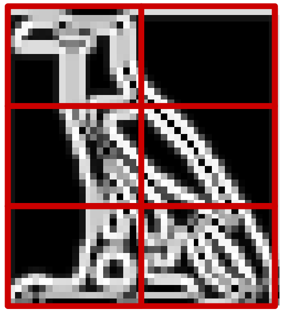




Compute the histogram

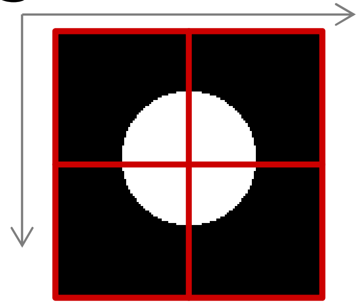
- We need a procedure to decompose the image into a regular [n m] grid and compute extrema of row and col coordinates for the k-th region

```
…
imgQTheta = quantizeOrientation(imgTheta, 5);
numGridRows=3; numGridCols=2;
for k=0:(numGridRows*numGridCols-1)
  [rMin, rMax, cMin, cMax] = gridBounds(size(imgMag),...
    numGridRows, numGridCols, k);
  gradHisto(:,k) = gradientHistogram(imgMag, imgQTheta, ...
                   rMin, cMin, rMax, cMax);
end
```

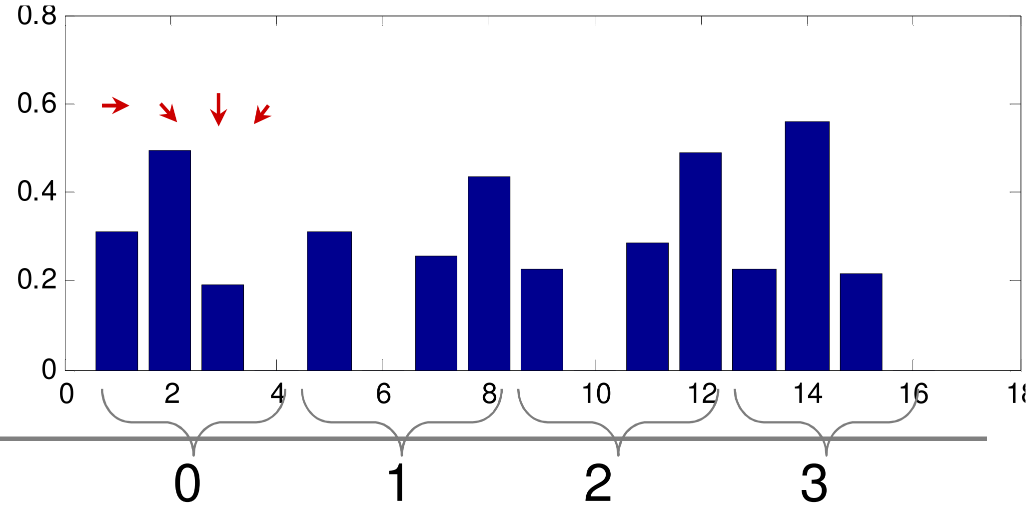


Compute the histogram

- Embed all the designed procedures into a new one (computeGradientHistogram) that given the image name, gridRows, gridCols and the number of orientation values returns the histogram of gradient orientation values



```
globalHisto = computeGradientHistogram('test.png', 2, 2, 4);
bar(globalHisto);
```

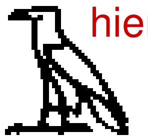


Comparing two histograms

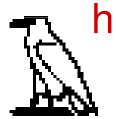
- The procedure **computeGradientHistogram** extracts a feature vector (the histogram of gradient orientation values, HoGO) that can be used to characterize the content of the original image
- Images with similar content should have similar HoGO descriptors
  - The dissimilarity between two images can be estimated through the distance of their histograms
- Several distance measures can be used to compare two histograms: Minkowski, Chi^2, Kullback-Leibler, …

HoGO descriptor accuracy

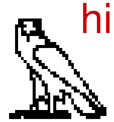
- Check the accuracy of the HoGO descriptor and the Euclidean distance to match corresponding hieroplyphs



hiero_01/01.png



hiero_02/01.png



hiero_02/02.png

```
templateHisto = computeGradientHistogram('hiero_01/01.png', 2, 2, 6);
queryHisto = computeGradientHistogram('hiero_02/01.png', 2, 2, 6);

norm( templateHisto-queryHisto ) =  0.4013

queryHisto = computeGradientHistogram('hiero_02/02.png', 2, 2, 6);

norm( templateHisto-queryHisto ) =  0.5552
```

HoGO descriptor accuracy

- Check the extent to which the accuracy of the HoGO descriptor changes depending on the number of grid columns/rows and on the number of quantized orientation values