COC202 Computer Vision
Lab 7 – PCA for feature reduction/object recognition – Solutions

1.

```
% QBE with colour histograms + Euclidean distances

% create image datastore
imds = imageDatastore('*.bmp');
imgs = readall(imds); % read in all images

% create colour histograms
for i=1:length(imgs)
    disp(sprintf('%2d - %s', i, imds.Files{i}));
    ch = colourhist(imgs{i});
    allhists(i,:) = ch(:); % reshape into vector and add to data matrix
end

sel = input('Select query image by number: ');

qhist = allhists(sel,:);
for i=1:length(imgs)
    mhist = allhists(i,:);
    dist(i) = norm(qhist-mhist, 2); % Euclidean distance
end

[d, ind] = sort(dist, 'ascend');
figure
for i=1:length(ind)
    subplot(10,10,i);
    imshow(imgs{ind(i)});
end
```

2.

```
% QBE with colour histograms+PCA

% create image datastore
imds = imageDatastore('*.bmp');
imgs = readall(imds); % read in all images

% create colour histograms
for i=1:length(imgs)
    disp(sprintf('%2d - %s', i, imds.Files{i}));
    ch = colourhist(imgs{i});
    allhists(i,:) = ch(:); % reshape into vector and add to data matrix
end

% do PCA on data
[pcs evals projdata] = mypca(allhists);
projimgs = projdata; % projected (full) data

sel = input('Select query image by number: ');
qhist = projimgs(sel,:);
for i=1:length(imgs)
    mhist = projimgs(i,:);
    dist(i) = norm(qhist-mhist,2); % Euclidean distance
end

[d, ind] = sort(dist, 'ascend'); % distances -> sort ascending
figure
for i=1:length(ind)
    subplot(10,10,i);
    imshow(imgs{ind(i)});
end
```

3.

```
% QBE with colour histograms+PCA

k = 25; % number of PCs to use

% create image datastore
imds = imageDatastore('*.bmp');
imgs = readall(imds); % read in all images

% create colour histograms
for i=1:length(imgs)
    disp(sprintf('%2d - %s', i, imds.Files{i}));
    ch = colourhist(imgs{i});
    allhists(i,:) = ch(:); % reshape into vector and add to data matrix
end

% do PCA on data
[pcs evals projdata] = mypca(allhists);
v = sum(evals(1:k))/sum(evals) % print variance captured by the k PCs
projimgs = projdata(:,1:k);

sel = input('Select query image by number: ');

qhist = projimgs(sel,:);
for i=1:length(imgs)
    mhist = projimgs(i,:);
    dist(i) = norm(qhist-mhist,2); % Euclidean distance
end

[d, ind] = sort(dist, 'ascend'); % distances -> sort ascending
figure
for i=1:length(ind)
    subplot(10,10,i);
    imshow(imgs{ind(i)});
end
```

4. +
5.

```
% PCA-based face recognition

k = 25; % number of PCs to use

querygroup = 'glasses'; % facial expression group used for testing
modelimds = imageDatastore('*.tif');
queryimds = imageDatastore(strcat('*.',querygroup,'.tif'));
% remove query images from model DB
modelimds.Files = setdiff(modelimds.Files, queryimds.Files);

% read in database images and store (as vectors) in data matrix
modelimgs = readall(modelimds); % read in all model images
for i=1:length(modelimgs)
    disp(sprintf('M:%2d - %s', i, modelimds.Files{i}));
    im = im2double(modelimgs{i});
    im = imresize(im,0.25);
    modelfaces(i, :) = im(:);
end
[dimx dimy] = size(im);

% do PCA on data
[eigenfaces evals projdata] = mypca(modelfaces);
projmodels = projdata(:,1:k); % project database images onto PCA space

% display eigenfaces
figure
for i=1:k
    subplot(5,10,i);
    eface = reshape(eigenfaces(:,i),dimx,dimy); % reshape into image
    eface    =    (eface    -    min(min(eface)))    /    (max(max(eface))-
min(min(eface))); % normalise for display
    imshow(eface);
end

% perform face recognition based on selected group
queryimgs = readall(queryimds); % read in all query images
figure
for i=1:length(queryimgs)
    disp(sprintf('Q:%2d - %s', i, queryimds.Files{i}));
    qim = im2double(queryimgs{i});
    qim = imresize(qim,0.25);
    projface = qim(:)' * eigenfaces(:,1:k); % project onto k-dim PCA space
    for j=1:length(modelimgs)
        dist(j) = norm(projface-projmodels(j,:),2); % Euclidean distance
in PCA space
    end
    [d, ind] = sort(dist, 'ascend'); % distances -> sort ascending
    subplot(5,6,2*i-1);
    imshow(queryimgs{i});
    subplot(5,6,2*i);
    imshow(modelimgs{ind(1)});
end
```