COC202 Computer Vision
Lab 8 – Image segmentation – Solutions

1.

```
im = imread('rice.png');
imhist(im)
imb = im>150; % select 150 as threshold
imshow(imb)
```

2.

```
imb = im2bw(im, graythresh(im));
imshow(imb)
```

Matlab implements Otsu's thresholding algorithm, which is the most widely used one in computer vision. Otsu's method selects a threshold so as to minimise the intra-class variance of foreground and background pixels (or, equivalently, maximise the inter-class variance).

3.

```
function [cb, cb_ind] = kmeans(data, k)
% k-means clustering algorithm

epsilon = 0.01; % used for convergence testing
[N M]= size(data); % N samples, each of length M

% generate initial clusters
cb = 255*rand(k, M); % for simplicity, we assume we know that we are
dealing with data in [0;255]

iter = 0;
err = Inf;
newerr = Inf;
while (iter<2 | ((err-newerr)/newerr)>epsilon)
    err = newerr;
    % map to nearest cluster centre
    [cb_ind newerr] = nn(data, cb);
    % re-calculate cluster centres
    for j=1:size(cb,1)
        d_ind = find(cb_ind==j);
        if (length(d_ind)>0) % only for non-empty clusters
            cb(j,:) = mean(data(d_ind,:));
        end
    end
    iter = iter + 1;
    % display previous/current error and change
    [err newerr (err-newerr)/newerr]
end
cb_new = cb;

% map to cluster centres once more
cb_ind = nn(data, cb);
```

```
function [cb_ind, meanerr] = nn(data, cb)
% Nearest neighbour mapping: maps each of the (row) vectors in data to
the nearest cluster in cb
% Returns the indices as well as the average mapping error.

nSamples = size(data,1);
cb_ind = zeros(1, nSamples); err = cb_ind;
for i=1:nSamples
    [err(i) cb_ind(i)] = min(sum((ones(size(cb, 1), 1)*data(i,:) -
cb).^2, 2), [], 1);
end
meanerr = mean(sqrt(err));
```

4.

```
% perform segmentation

im = imread('peppers.jpg');

k = 3; % number of clusters - play with this

% reshape data
[h w d] = size(im);
data = reshape(double(im), h*w, 3);

% do clustering
[cb ind] = kmeans(data, k);

ind = reshape(ind, h, w);  % reshape back

% generate "segmented" image (replacing indices with cluster colour for
display)
segmim = uint8(ind2rgb(ind, cb));

% show original and segmented image
figure
subplot(1,2,1);
imshow(im);
subplot(1,2,2);
imshow(segmim);

% show segments
figure
for i=1:k
    subplot(1,k,i)
    imshow(ind==i);
end
```

5.

```
function [cb, cb_ind] = kmeans(data, k)
% k-means clustering algorithm

epsilon = 0.01; % used for convergence testing
[N M]= size(data); % N samples, each of length M

% generate initial clusters by selecting k (unique) pixels from the image
pixvals = unique(data, 'rows'); % unique pixel values in image
cb = pixvals([randperm(length(pixvals), k)],:);

iter = 0;
err = Inf;
newerr = Inf;
while (iter<2 | ((err-newerr)/newerr)>epsilon)
    err = newerr;
    % map to nearest cluster centre
    [cb_ind newerr] = nn(data, cb);
    % re-calculate cluster centres
    for j=1:size(cb,1)
        d_ind = find(cb_ind==j);
        if (length(d_ind)>0)
            cb(j,:) = mean(data(cb_ind==j,:));
        end
    end
    iter = iter + 1;
    % display previous/current error and change
    [err newerr (err-newerr)/newerr]
end
cb_new = cb;

% map to cluster centres once more
cb_ind = nn(data, cb);
```