Robot Vision

RV ex1 Image representation and simple thresholding methods

Mateusz Warmuz

1. Propose an algorithm for calculating the number of gray level in grayscale images coded with 8 bits per pixel (bpp).

```
RGB = imread('kotek.jpg');
I = rgb2gray(RGB);
imwrite(I, 'kotek_mono.jpg');
count1 = 0;
zero = zeros();

for i = 1:size(I,1)   % size of rows
   for j = 1:size(I,2)     %size of columns
      if(ismember(I(i,j), zero) == 0)
         zero = [zero I(i,j)];
         count1 = count1 + 1;
      end
   end
end
```

The algorithm checks if the element of the matrix (which represents a gray level) is the same as previous (zero). If it is false then the value is overwritten to variable zero and hold it after finding a different value. From calculation of the unique number grey level is equal 256.

2. Propose an algorithm for calculating the number of unique colours in true colour RGB images (24 bpp) — the usage of unique instruction and counting of color number for indexed representation of an image is forbidden.

```
count2 = 0;
A = [];
B = [];

for i=1:size(RGB,1)
   for j=1:size(RGB,2)
      A = [A [RGB(i,j,1); RGB(i,j,2); RGB(i,j,3)] ];
   end
end

for i = 1:size(A,2)
   if (sum(ismember(A(:,i), B)) ~=3)
       B = [B A(:,i)];
       count2 = count2 + 1;
   end
end
```

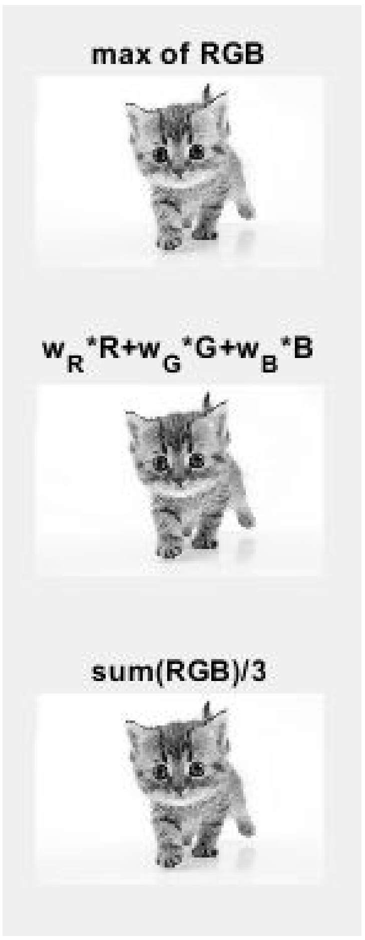
The RGB is a 3 dimensional array in which one pixel is represented by three components: red, green, blue. I convert this in a 2 dimensional array in which one element is represented by a three component vector of RGB. The second part of the algorithm is the same as in the previous task. The only difference is that it has to check 3 elements instead of 2 and sum it. If the sum of these elements is different than 3 (sum of three true values 1+1+1=3), the algorithm has found a new value. Value of the unique RGB color is 150. As we can observe the algorithm finds slightly less amount of unique RGB colors than in achromatic images.

3. Implement your algorithms in MATLAB. Test your computer program, comparing the obtained results with results obtained from e.g. IrfanView, PaintShopPro.

Comparing the result of the greyscale image to the IrfanView program confirms the correctness of the algorithm. The situation with calculation of unique colors in true RGB images is not the same.

4. Use different conversion methods of colour image to grayscale image (use 8-bit coding for grayscale images). Compare numbers of gray levels obtained for those images.

```
RGB = imread('kotek.jpg');
A = uint8(size(RGB));
B = uint8(size(RGB));
C = uint8(size(RGB));

zero1 = [];
zero2 = [];
zero3 = [];
count1 = 0;
count2 = 0;
count3 = 0;
for i=1:size(RGB,1)
   for j=1:size(RGB,2)
         A(i,j) = max(RGB(i,j));
         if(ismember(A(i,j), zero1) == 0)
            zero1 = [zero1 A(i,j)];
            count1 = count1 + 1;
         end
   end
end
subplot(3,1,1);
imshow(A);
title('max of RGB');
for i=1:size(RGB,1)
   for j=1:size(RGB,2)
         B(i,j) = (0.3*RGB(i,j,1)+0.6*RGB(i,j,2)+0.1*RGB(i,j,3));

         if(ismember(B(i,j), zero2) == 0)
            zero2 = [zero2 B(i,j)];
            count2 = count2 + 1;
         end
   end
end
subplot(3,1,2);
imshow(B);
title('w_R*R+w_G*G+w_B*B');
for i=1:size(RGB,1)
   for j=1:size(RGB,2)
         C(i,j) = ((double(RGB(i,j,1))+double(RGB(i,j,2))+double(RGB(i,j,3))))/3;
         if(ismember(C(i,j), zero3) == 0)
            zero3 = [zero3 C(i,j)];
            count3 = count3 + 1;
         end
   end
end
subplot(3,1,3);
imshow(C);
title('sum(RGB)/3');
```



We can observe that the first and second method convert image very well. The number of different colors in an image are the same 256.

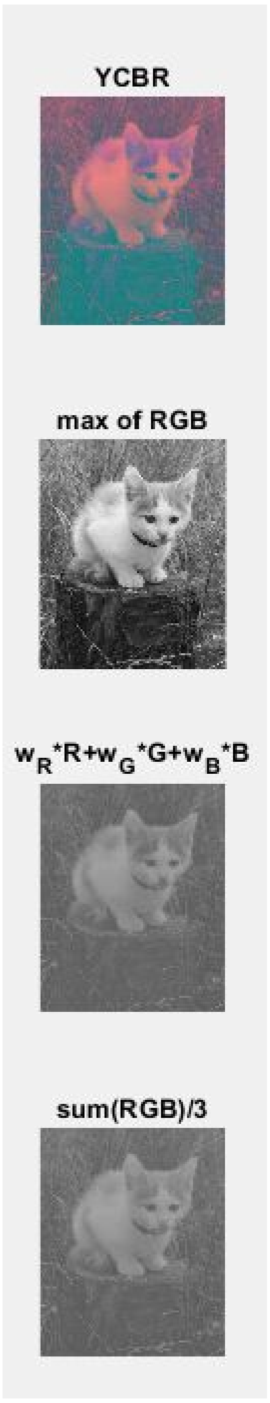
5. Compare the results of colours counting obtained for the same image saved using different file formats e.g. BMP, GIF, JPEG, PNG.

bmp 40950
png 40950
jpg 38835
gif 80

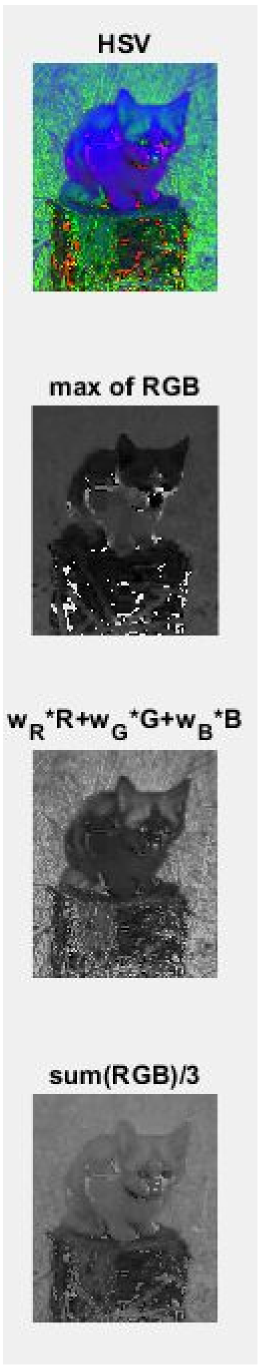
Image formats can be separated into three categories: lossy compression, lossless compression and uncompressed.
Uncompressed formats take up the most amount of data, but they are exact representations of the image. Bitmap formats such as BMP generally are uncompressed.
Lossy compression in practice .jpeg is for photographs, and .png for graphics and screenshots. To be honest I'm not able to see the difference between JPG and BMP. If you really need high quality images of course you will use BMP format.
Lossless compression formats are suited for illustrations, drawings, text and other material that would not look good when compressed with lossy compression. As the name implies, lossless compression will encode all the information from the original, so when the image is decompressed, it will be an exact representation of the original.

6. Check if your program works for images transformed to YCRCB and HSV color spaces. If not, propose a corrected algorithm and check it.

```
RGB = imread('kotek.jpg');
YCBR = rgb2ycbcr(RGB);

A = uint8(size(RGB));
B = uint8(size(RGB));
C = uint8(size(RGB));

zero1 = [];
zero2 = [];
zero3 = [];
count1 = 0;
count2 = 0;
count3 = 0;

subplot(4,1,1)
imshow(YCBR);
title('YCBR');
for i=1:size(YCBR,1)
   for j=1:size(YCBR,2)
         A(i,j) = max(YCBR(i,j));
         if(ismember(A(i,j), zero1) == 0)
            zero1 = [zero1 A(i,j)];
            count1 = count1 + 1;
         end
   end
end
subplot(4,1,2);
imshow(A);
title('max of RGB');
for i=1:size(YCBR,1)
   for j=1:size(YCBR,2)
         B(i,j) = (0.3*YCBR(i,j,1)+0.6*YCBR(i,j,2)+0.1*YCBR(i,j,3));

         if(ismember(B(i,j), zero2) == 0)
            zero2 = [zero2 B(i,j)];
            count2 = count2 + 1;
         end


   end
end
subplot(4,1,3);
imshow(B);
title('w_R*R+w_G*G+w_B*B');
for i=1:size(YCBR,1)
   for j=1:size(YCBR,2)
         C(i,j) = ((double(YCBR(i,j,1))+double(YCBR(i,j,2))+double(YCBR(i,j,3))))/3;
         if(ismember(C(i,j), zero3) == 0)
            zero3 = [zero3 C(i,j)];
            count3 = count3 + 1;
         end
   end
end
subplot(4,1,4);
imshow(C);
title('sum(RGB)/3');
```



```
RGB = imread('kotek.jpg');
HSV = rgb2hsv(RGB);

A = uint8(size(RGB));
B = uint8(size(RGB));
C = uint8(size(RGB));

zero1 = [];
zero2 = [];
zero3 = [];
count1 = 0;
count2 = 0;
count3 = 0;

subplot(4,1,1)
imshow(HSV);
title('HSV');
for i=1:size(HSV,1)
   for j=1:size(HSV,2)
         A(i,j) = max(HSV(i,j))*255;
         if(ismember(A(i,j), zero1) == 0)
            zero1 = [zero1 A(i,j)];
            count1 = count1 + 1;
         end
   end
end
subplot(4,1,2);
imshow(A);
title('max of RGB');
for i=1:size(HSV,1)
   for j=1:size(HSV,2)
         B(i,j) = (0.3*HSV(i,j,1)+0.6*HSV(i,j,2)+0.1*HSV(i,j,3))*255;

         if(ismember(B(i,j), zero2) == 0)
            zero2 = [zero2 B(i,j)];
            count2 = count2 + 1;
         end


   end
end
subplot(4,1,3);
imshow(B);
title('w_R*R+w_G*G+w_B*B');
for i=1:size(HSV,1)
   for j=1:size(HSV,2)
         C(i,j) =
(((double(HSV(i,j,1))+double(HSV(i,j,2))+double(HSV(i,j,3))))/3)*255;
         if(ismember(C(i,j), zero3) == 0)
            zero3 = [zero3 C(i,j)];
            count3 = count3 + 1;
         end
   end
end
subplot(4,1,4);
imshow(C);
title('sum(RGB)/3');
```

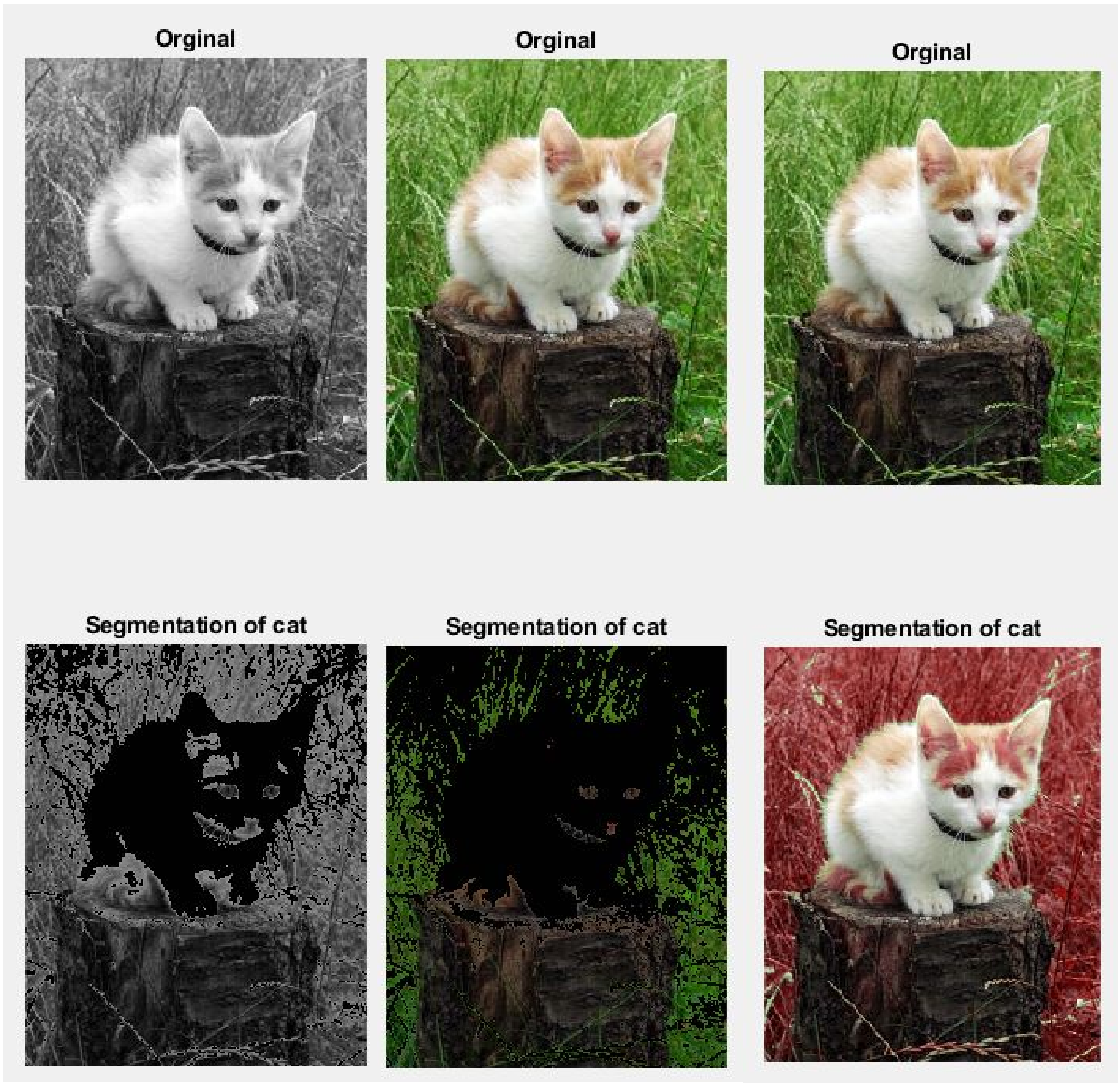


The problem appears in the sum method when the value exceeds 256 (8 bit value). It has to be implemented as a double to achieve higher values and then divided into 3.

7. Propose algorithms for segmentation of grayscale and colour (coded in RGB and HSV colour spaces) images; implement it using MATLAB.

```
RGB = imread('kotek.jpg');
grey = rgb2gray(RGB);
HSV = rgb2hsv(RGB);


A =grey;
B = RGB;
C = HSV;
% subplot(2,1,1)
% imshow(RGB);
% title('Orginal');

%grey
for i = 1:size(A,1)   % size of rows
   for j = 1:size(A,2)     %size of columns
      if(A(i,j)>140)
         A(i,j) = 1;
      end
   end
end



%RGB
for i = 1:size(B,1)   % size of rows
   for j = 1:size(B,2)     %size of columns
      if((B(i,j,1)+B(i,j,2)+B(i,j,3))>245)
         B(i,j,1) = 1;
         B(i,j,2) = 1;
         B(i,j,3) = 1;
      end
   end
end



%HSV
for i = 1:size(C,1)   % size of rows
   for j = 1:size(C,2)     %size of columns
      if(C(i,j,2)>0.35)
         C(i,j,1) = 1/360; %red background
      end
   end
end
D = hsv2rgb(C);
% subplot(2,1,2)
% imshow(D);
% title('Segmentation of cat');
```

8. Compare result obtained for colour images and its gray level versions.



We can notice that grey and hsv methods highlight a cat better than in RGB color. It is due to the fact that Greyscale images have only two dominant colors black and white, so it is easier to select a white image from a grey background. The same is for HSV method where we have distinctive colors.

Questions

1. Does the use of different methods of converting a color image to grayscale image influence on the resulting gray level number?

For example in the weighted method, I use weights 0.3, 0.6 and 0.1 respectively for wr, wg and wb. These values are the best for our vision system, especially weight responsible for green color, we are the most sensitive. If we change above values the greyscale will be not sufficiently good. Also if we take into account only one component of color space in an RGB image will be not sufficiently converted to greyscale.

2. Does the change of the color space (RGB, HSV, YCrCb) change the number of colors in the image?

Of course, if we for example change one component of color space. It will have significantly different colors on the output image.

3. Which version of colour space gives better results in the segmentation process?

HSV method is the best for segmentation, such transformation will allow to take advantage of color separation in the chrominance space without interference of the luminance (a.k.a. brightness).

4. Which colour objects can be correctly segmented from test images?

Image segmentation is the process used to find in objects the boundaries (e.g., lines or curves) in images. As the segmentation results depend on the used color space, there is no single color space that can provide acceptable results for all kinds of images. So in my opinion the best colors have to be distinguishable colors [0, 0, 0] white and [255, 255, 255] black. For this color the condition will be much easier to detect the edges.

5. Discuss the influence of shadows and highlights located on the surface of objects on the results of segmentation.

It is a robust problem for the algorithm to neglect a shadow and highlights on the image. Single physical reflectance can have many different image values. These variations are caused by shadows, shading and highlights and due to varying object geometry.