Image Analysis And Computer Vision
Exercises

Christian Rossi

Academic Year 2023-2024

Abstract

The topics of the course are:

- Introduction.
- Camera sensors: transduction, optics, geometry, distortion
- Basics on Projective geometry: modelling basic primitives (points, lines, planes, conic sections, quadric surfaces) and projective spatial transformations and projections.
- Camera geometry, and single view analysis: calibration, image rectification, localization of 3D models.
- Multi-view analysis: 3D shape reconstruction, self-calibration, 3D scene understanding.
- Linear filters and convolutions, space-invariant filters, Fourier Transform, sampling and aliasing.
- Nonlinear filters: image morphology and morphology operators (dilate, erode, open, close), median filters.
- Edge detection and feature detection techniques. Feature matching and feature tracking along image sequences.
- Inferring parametric models from noisy data (including outliers), contour segmentation, clustering, Hough Transform, Ransac (random sample consensus).
- Applications: object tracking, object recognition, classification.

Contents

Chapter 1

Introduction to MATLAB

1.1 Main MATLAB operators

To print some string it is possible to use;

```
% Print the string
disp('string');
% Print the string with C-like syntax
fprintf('string\n%s %d', 'string', number);
```

The variables are created as follows:

```
% Variables are created by assignements
v = 3
c = 'k'
size(v)
% Data types are automatically defined
whos v
% Casting to 8-bit integers
v = uint8(v)
```

The main data types used on MATLAB: double, uint8, and logical. The arrays are defined in the following ways:

```
% A row vector
r=[1, 2, 3, 4]
% A column vector
c=[1; 2; 3; 4]
% Vectors by regular increment operator
% [start : step : end]
a = [1 : 2 : 10];
```

```
% A matrix
v=[ 1 2;
    3 4 ]
```

It is possible to concatenate arrays:

```
B = [v', v']
C = [v ; v]
```

The arrays can be divided in subarrays:

```
% First row and second column
v(1,2)
% The second column of v
v(:,2)
% The first row of v
v(1,:)
% Some of the columns from 2 to 4
B(:,2:4)
```

Some useful mathematical operations are:

```
% . means elementwise operation
[1 2 3].*[4 5 6]
[1 2 3] + 5
[1 2 3] * 2
[1 2 3] .* 2
[1 2 3] / 2
[1 2 3] ./ 2
[1 2 3] .^ 2
% Inner product
[1 2 3] * [4 5 6]'
% This is the matrix product, returns a matrix
[1 2 3]' * [4 5 6]
% Functions for rounding functions
ceil(10.56)
floor(10.56)
round(10.56)
% Arithmetic functions
sum([1 2 3 4])
sum([1:4;5:8])
sum([1:4;5:8],2)
```

1.2 Commands for images

The images in MATLAB are treated as matrices:

```
im=imread('photo.png');
% Show the image
imshow(im);
% Show two concatenated images orizontally
imshow([im im]);
% Show two concatenated images vertically
imshow([im; im]);
```

To plot the histogram of the various pixels it is possible to write:

```
h = hist(im(:), [0: im_length]);
figure(2), stairs([0: im_length], h), title('
   Intensity histogram')
axis tight
```

It is possible to modify the brightness and contrast with a simple operation:

```
figure(1), imshow(im + 50), title('50 graylevels')
figure(1), imshow(im + 100), title('100 graylevels')
% contrast modify
eq = double(im - min(im(:)))/double(max(im(:)) - min
   (im(:))) * 255;
```

Image ranges (in the visualization) can be also controlled:

```
imshow(im,[-100 156]); title('more brighness');
imshow(im,[0 156]); title('more brighness and
   contrast');
imshow(im,[ 100 256]); title('less brighness, more
   contrast');
imshow(im,[ 100 356]); title('less brighness');
```

The gamma correction is done in this way:

```
for gamma = [.04 .1 .2 .4 .7 1 1.5 2.5 5 10 25]
    y = x.^gamma;
    plot(x,y,'DisplayName',sprintf('\\gamma = %.2f',
       gamma));
    % display the text
    text(x(round(end/2)),y(round(end / 2)), sprintf(
       '\\gamma = %.2f',gamma));
```

```
end
```

Color is represented by 3 channels (RGB). We can read each channel:

```
% Red channel
imr=im(:,:,1);
% Green channel
imr=im(:,:,2);
% Blue channel
imr=im(:,:,3);
```