M2 Computer Science
Lab Classes

« Computer Vision »

Laboratory classes

M2 Computer Science
Lab Classes

Lab 1

Depth estimation using stereo images.

Goal.

The goal of this lab class is

a) implement and evaluate the algorithm of depth estimation using a couple of stereo images ;
b) find the limits of the investigated approach.

1.Depth map.

The images which will be used are already rectified ; the epipolar lines are horizontal and aligned between the two images.

1.1 Design an algorithm which estimates the depth map using SSD or ZNCC (cf. class handouts & image 1.

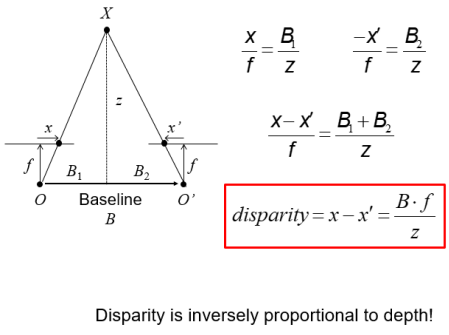

Figure 1. Principle of depth estimation.

Algorithm could be :

For each pixel of the left image in x_1, a pixel x_2 research is performed on the same line in image of the right picture ; the SSD/ZNCC value is minimal in x_2 .

The difference (disparity) $d = x_1 - x_2$ identified is inversly proportional to the depth $p \approx 1/d$.

In a new image you memorize the disparity of each pixel.

Comment.
The size of the correlation window is a parameter which can be modified.

1.2. Test and validate your algorithm on data of the « synthetic » file.

2. Size of the correlation window.

2.1. Test your algorithm with data included in files « cones » and « teddy ».

2.2. Change the size of the correlation window. What changes on your depth map ?

2.3. Compare your results with true data provided. What are, according to you, limits of the (local) correlations ?

2.4. Filter your depth map using the median filter of size 7 x 7 (several application of this filter). How changes your depth map with respect to true data ?

3. Constraint of unicity (non mandatory part).

Estimate the depth map of the left image to right image and vice versa.
Apply the following constraint :

- for each pixel I_g de left image associated with a pixel of right image check that its corresponding point of the right image I_d matches effectively the point I_g of the left image ; if there is no case, put (-1) in the depth map.

Compute once again the depth maps of « cones » and « teddy » while implementing the above unicity constraint.