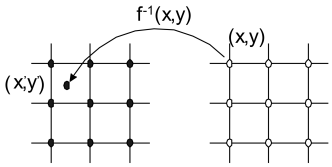
Computer Vision: Lab Session n.1
Image warping and bi-linear interpolation

Francesca Canale 4113133
Filippo Gandolfi 4112879
Marco Giordano 4034043

20/03/2019

1 Introduction

Given a coordinate transform $(x, y) = h(x', y')$ and a source image $f(x', y')$, the goal of this first lab is to compute a transformed image $g(x, y) = f(h(x', y'))$ using the technique of backward warping with a bi-linear interpolation. In backward method, to calculate the value of a destination image pixel, reversed transfer function is mapped to the coordinates of that pixel and the coordinates of the corresponding point in the source image is calculated. Pixel of the destination image with its coordinates (x, y) is mapped to the backward function f to the point with coordinates (x', y') in the source image. Then we need an interpolation to evaluate the pixels that have no values after the backward warping in order to avoid the holes.



First we imported the images in Matlab using function *imread()* and we converted them from RGB to gray scale using function *rgb2gray()*. Starting from these gray scale images we have performed two different transformation: translation and rotation.

Figure 1: Boccadasse gray scale

Figure 2: Flower gray scale

2 Translation

To perform translation we implemented one function *translation()* that:

- With function $[X, Y] = meshgrid(x, y)$ transforms the domain specified by vectors x and y into arrays X and Y, which can be used to evaluate functions of two variables and three-dimensional mesh/surface plots or images. The rows of the output array X are copies of the vector x; columns of the output array Y are copies of the vector y.
- Changes the position of the image by the selection of two coefficient $(xTranslation, yTranslation)$ for the translation along X and Y axes.
- Performs an interpolation with function $griddata()$.

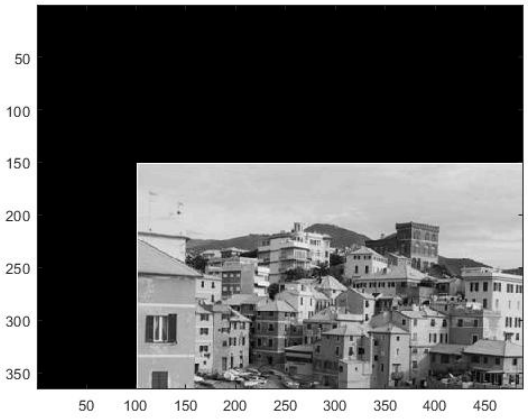


Figure 3: Boccadasse translated ($xTranslation = 100, yTranslation = 150$)

Figure 4: Flower translated ($xTranslation = 70, yTranslation = 50$)

3 Rotation

To perform rotation we implemented one function *rotation()* that:

- With function *meshgrid(x,y)* performs the same procedure as in function *translation()*.
- Computes the center of the image to permit a centered rotation of the selected angle theta expressed in radiant.
- Performs an interpolation with function *griddata()*.

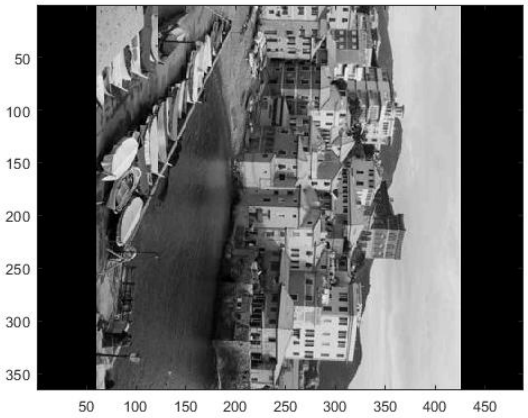


Figure 5: Bocadasse rotation ($\theta = \pi/2$)

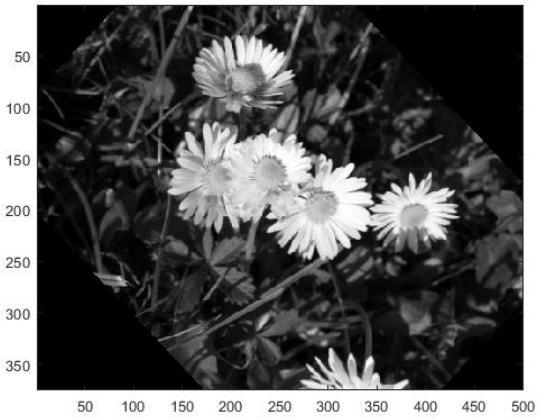


Figure 6: Flower rotation ($\theta = \pi/4$))