BIOMED - TME 1

Marie Diez

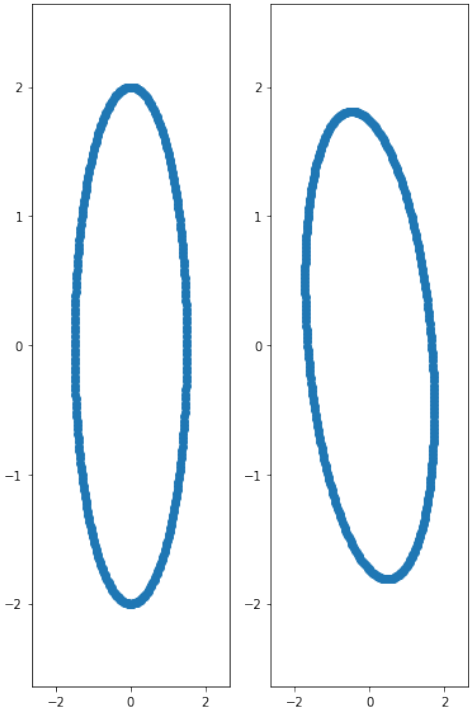
1 Introduction

The goal of this notebook is to implement the algorithms seen in courses for pixel-based image registration. The exercise consists more precisely in setting up a transformation matrix which will allow to transform an image. The final goal is to transform a source image to a target image so that the difference between these 2 images is minimal (Lucas-Kanade Algorithm)

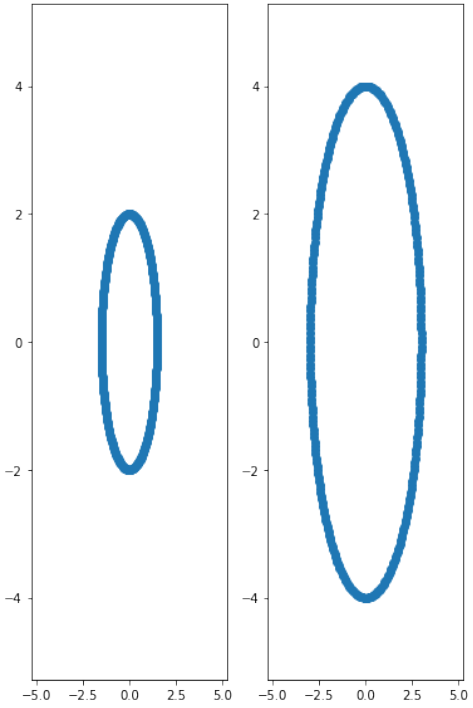
2 Implementation Results

2.1 applyTransformation function - Ellipse

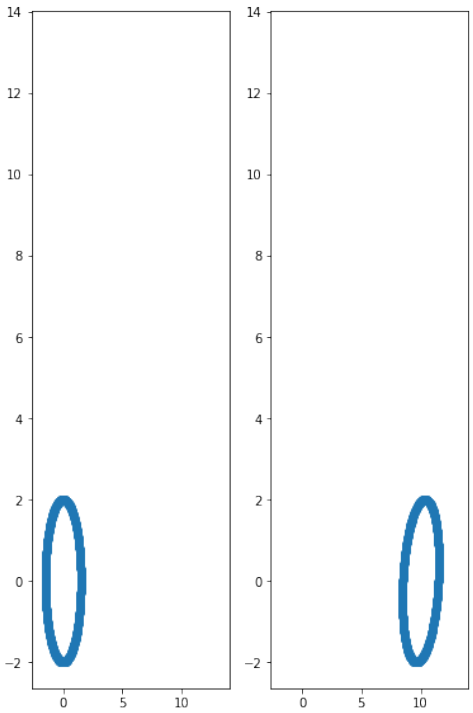
```
T=Tmatrix(scale=(1,1), rotationAngle=40, shearAngles=(0,0),translation=(0,0), elation
    =(0,0),reflection=(False,False))
```



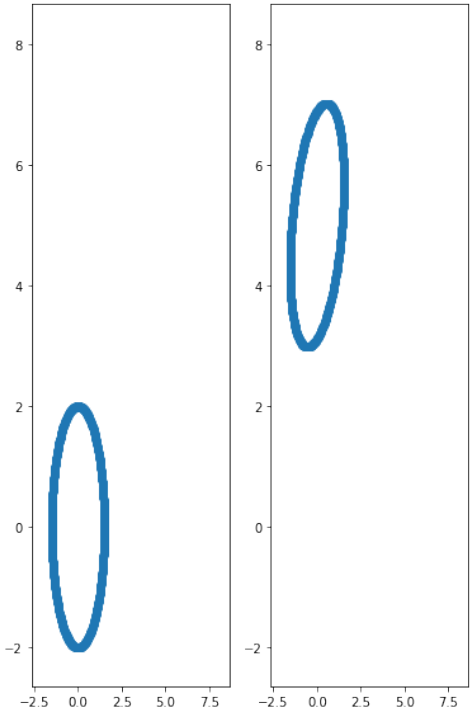
```
T=Tmatrix(scale=(2,2), rotationAngle=0, shearAngles=(0,0),translation=(0,0), elation
    =(0,0),reflection=(False,False))
```



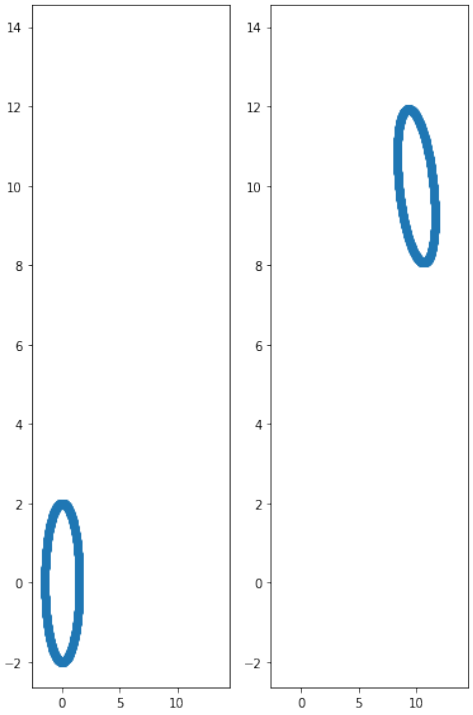

```
T=Tmatrix(scale=(1,1), rotationAngle=0, shearAngles=(10,0),translation=(10,0), elation
    =(0,0),reflection=(False,False))
```



```
T=Tmatrix(scale=(1,1), rotationAngle=0, shearAngles=(10,10),translation=(0,5), elation
    =(0,0),reflection=(False,False))
```

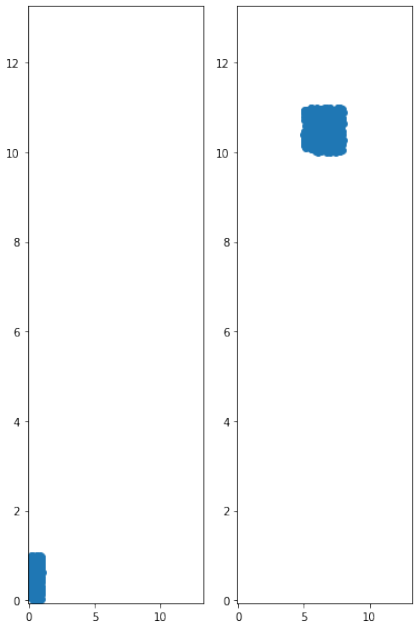



```
T=Tmatrix(scale=(1,1), rotationAngle=60, shearAngles=(10,10),translation=(10,10),
    elation=(0,0),reflection=(False,False))
```

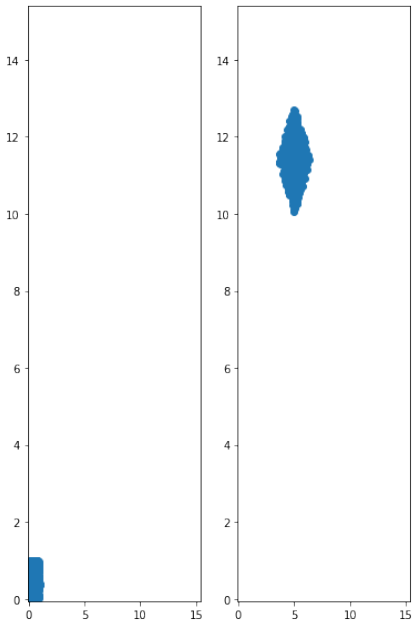


2.2 applyTransformation function - Square

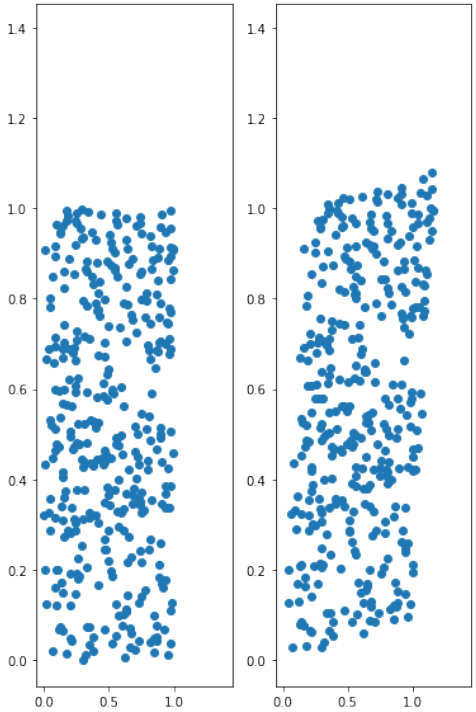
```
T=Tmatrix(scale=(3,1), rotationAngle=0, shearAngles=(0,0),translation=(5,10),elation
    =(0,0),reflection=(False,False))
```



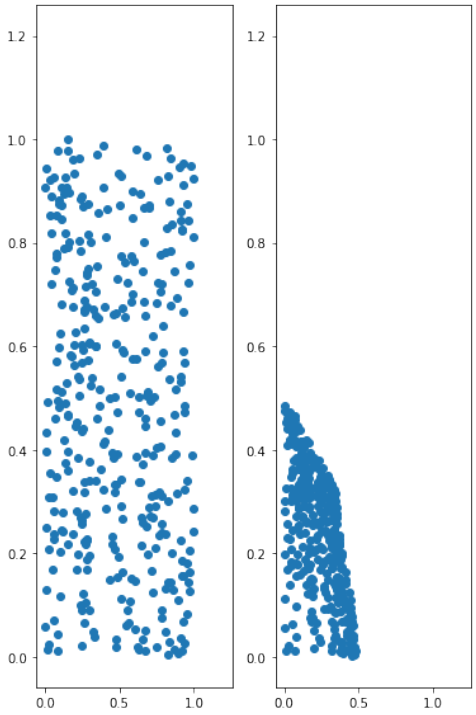

```
T=Tmatrix(scale=(2,2), rotationAngle=45, shearAngles=(0,0),translation=(5,10),elation
    =(0,0),reflection=(False,False))
```



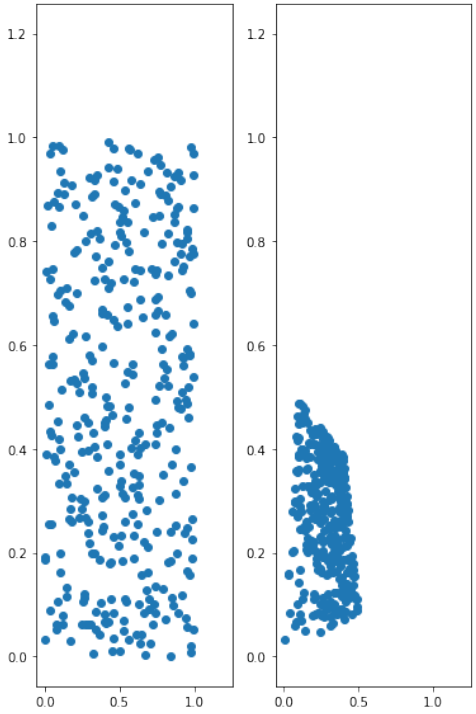
```
T=Tmatrix(scale=(1,1), rotationAngle=0, shearAngles=(10,5),translation=(0,0),elation
    =(0,0),reflection=(False,False))
```




```
T=Tmatrix(scale=(1,1), rotationAngle=0, shearAngles=(0,0),translation=(0,0),elation
    =(1,1),reflection=(False,False))
```



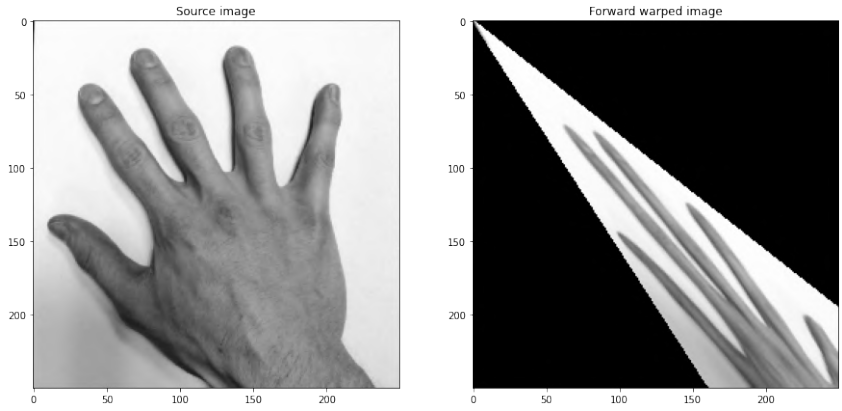
```
T=Tmatrix(scale=(1,1), rotationAngle=0, shearAngles=(10,10),translation=(0,0),elation
    =(1,1),reflection=(False,False))
```



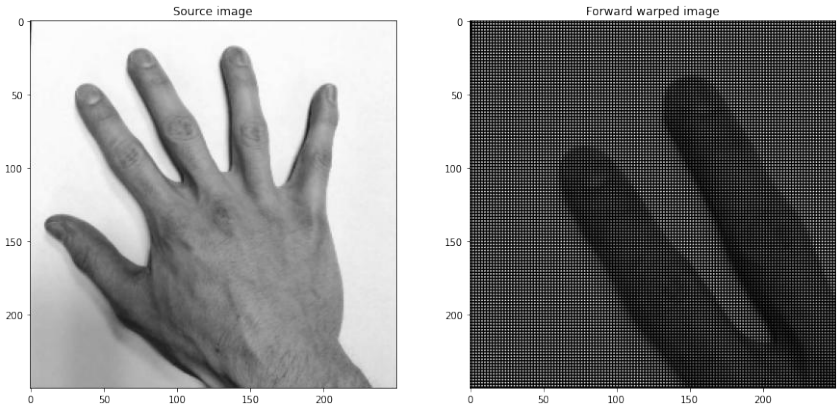
2.3 ForwardWarping function

Now that we have a transformation matrix T, we will apply it in a first step with the method Forward-Warping the transformation to an image.

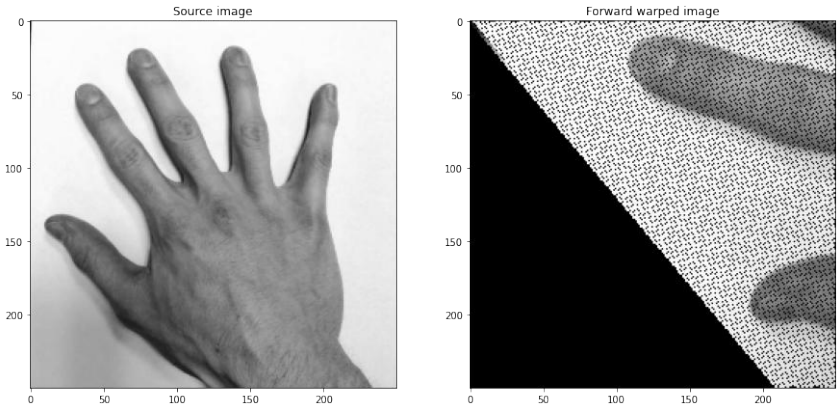
```
T=Tmatrix(scale=(1,1), rotationAngle=0, shearAngles=(45,37),translation=(0,0),elation
    =(0,0),reflection=(False,False))
```



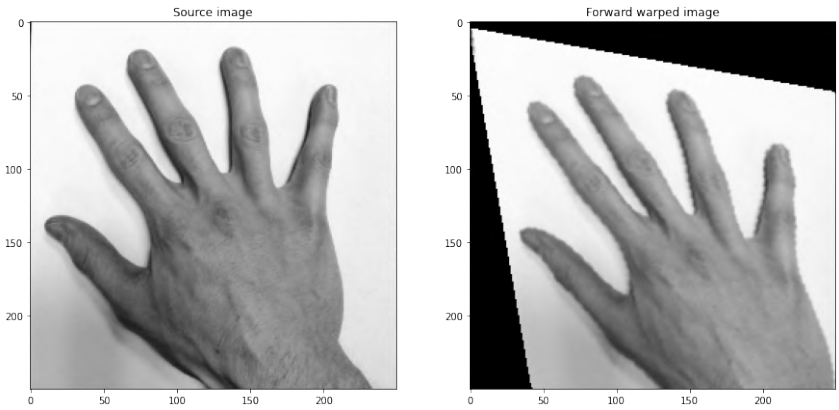
```
T=Tmatrix(scale=(2,2), rotationAngle=0, shearAngles=(0,0),translation=(0,0),elation
    =(0,0),reflection=(False,False))
```



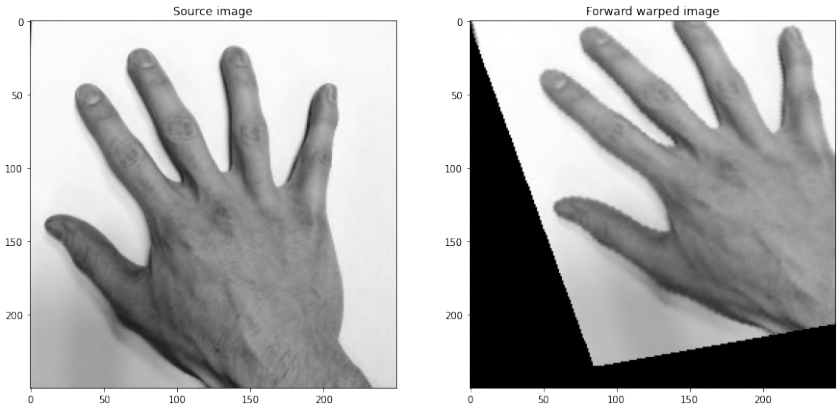

```
T=Tmatrix(scale=(2,2), rotationAngle=40, shearAngles=(0,0),translation=(0,0),elation
    =(0,0),reflection=(False,False))
```




```
T=Tmatrix(scale=(1,1), rotationAngle=0, shearAngles=(10,10),translation=(5,0),elation
    =(0,0),reflection=(False,False))
```



```
T=Tmatrix(scale=(1,1), rotationAngle=20, shearAngles=(10,0),translation=(0,0),elation
    =(0,0),reflection=(False,False))
```

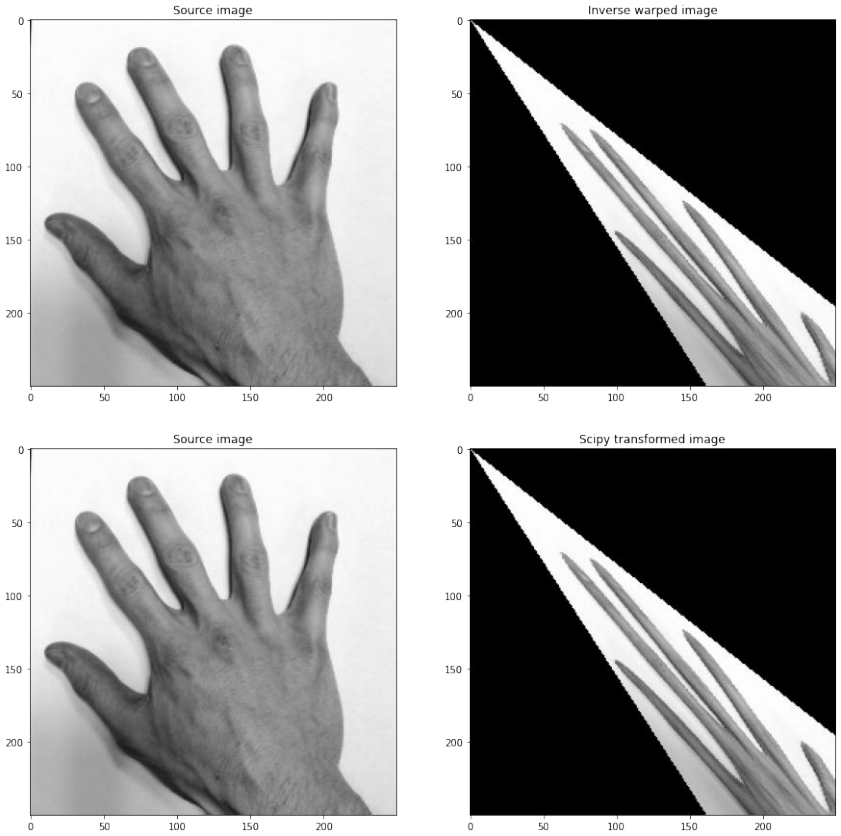


We can see on the results above, mainly on the zoomed images, that the forward mapping is not perfect and leaves some pixels without values on the transformed image. This is why we will use the inverse mapping method.

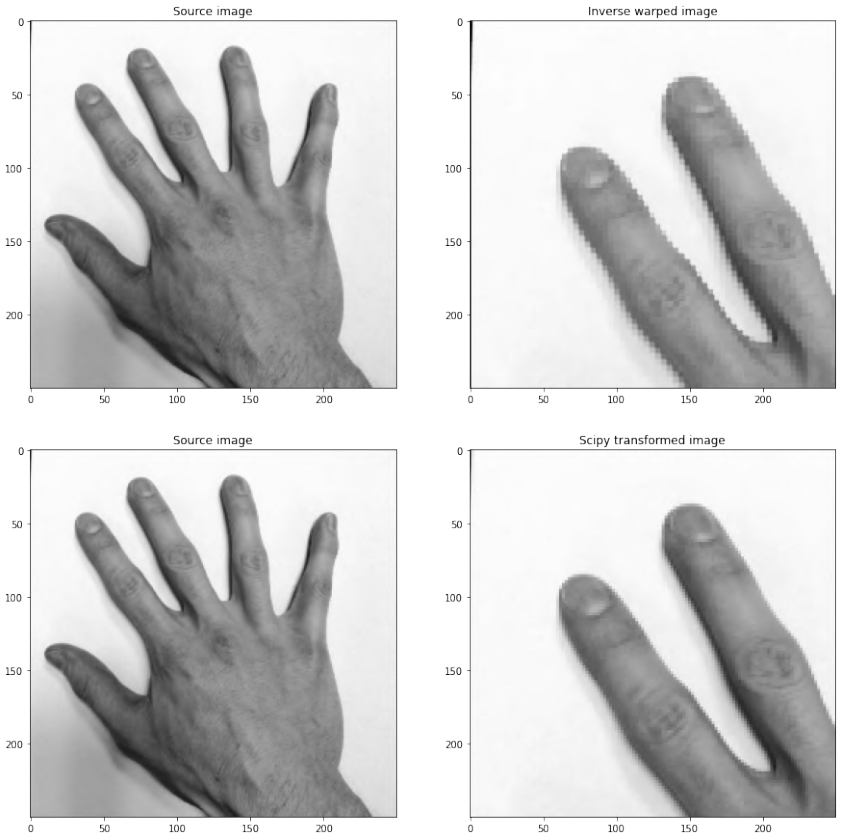
2.4 InverseWarping function

Here are the results for the same transformations as above, with the associated scipy result (with nearest neighbors interpolation) :

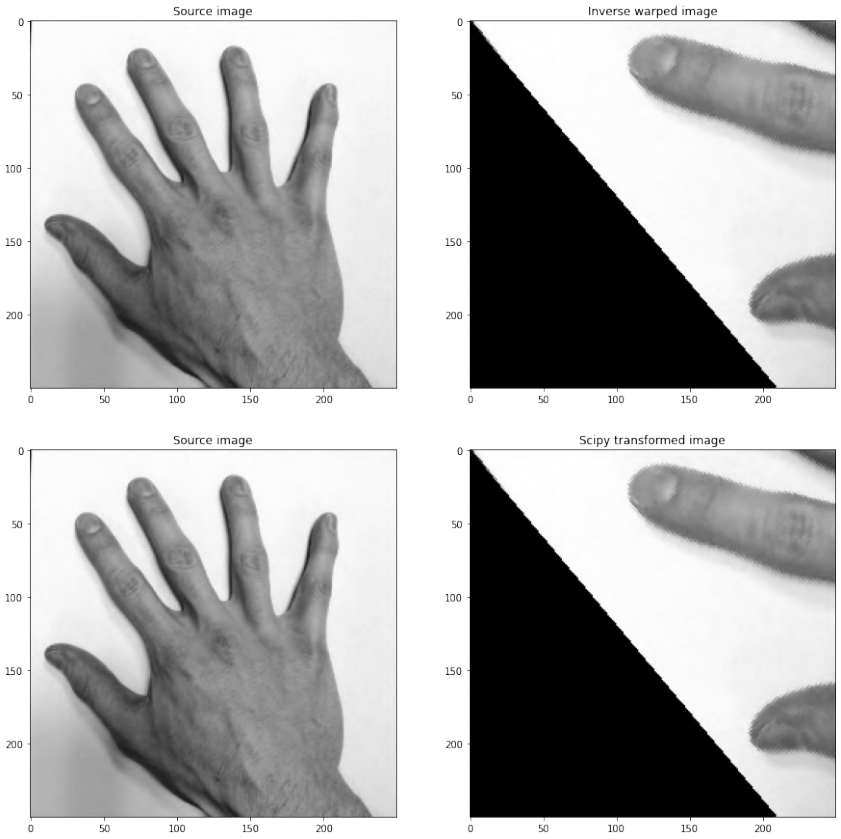
```
T=Tmatrix(scale=(1,1), rotationAngle=0, shearAngles=(45,37),translation=(0,0),elation
    =(0,0),reflection=(False,False))
```



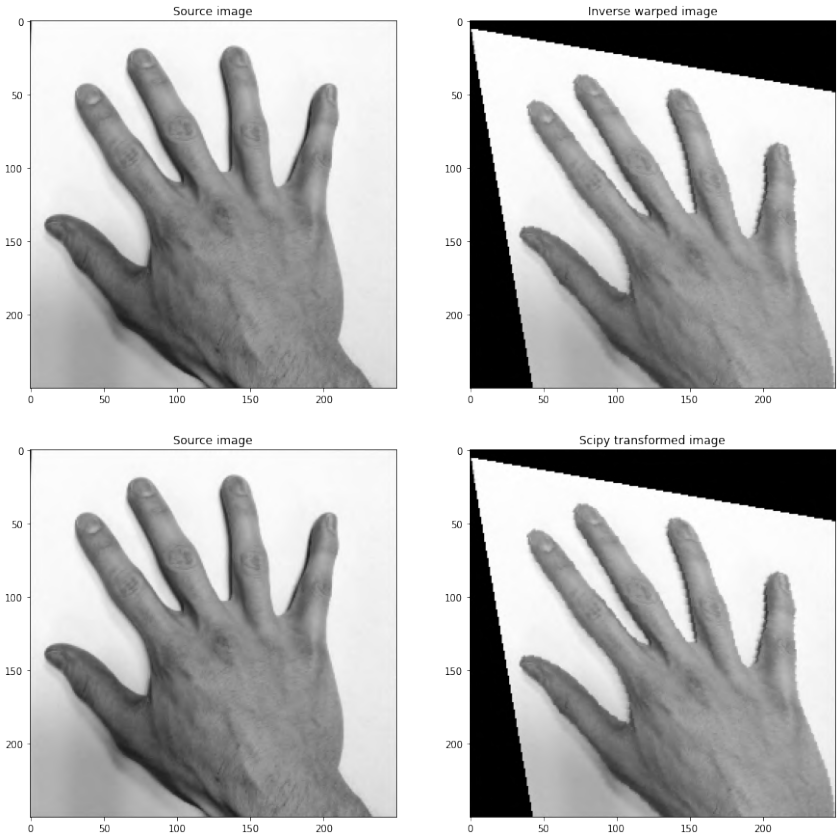
```
T=Tmatrix(scale=(2,2), rotationAngle=0, shearAngles=(0,0),translation=(0,0),elation
    =(0,0),reflection=(False,False))
```



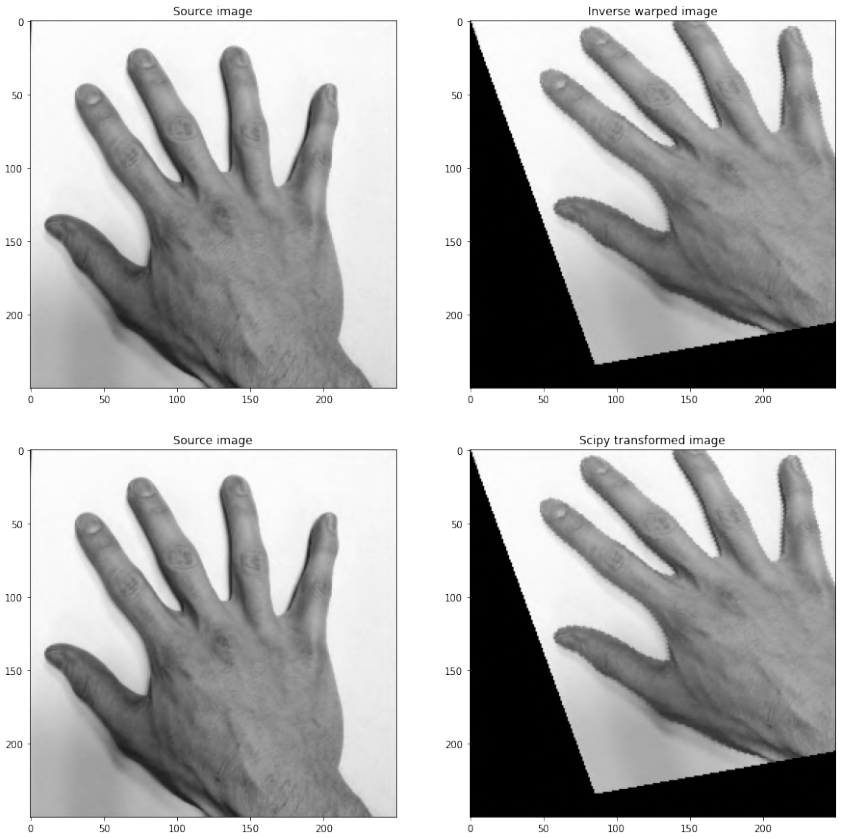
```
T=Tmatrix(scale=(2,2), rotationAngle=40, shearAngles=(0,0),translation=(0,0),elation
    =(0,0),reflection=(False,False))
```



```
T=Tmatrix(scale=(1,1), rotationAngle=0, shearAngles=(10,10),translation=(5,0),elation
    =(0,0),reflection=(False,False))
```



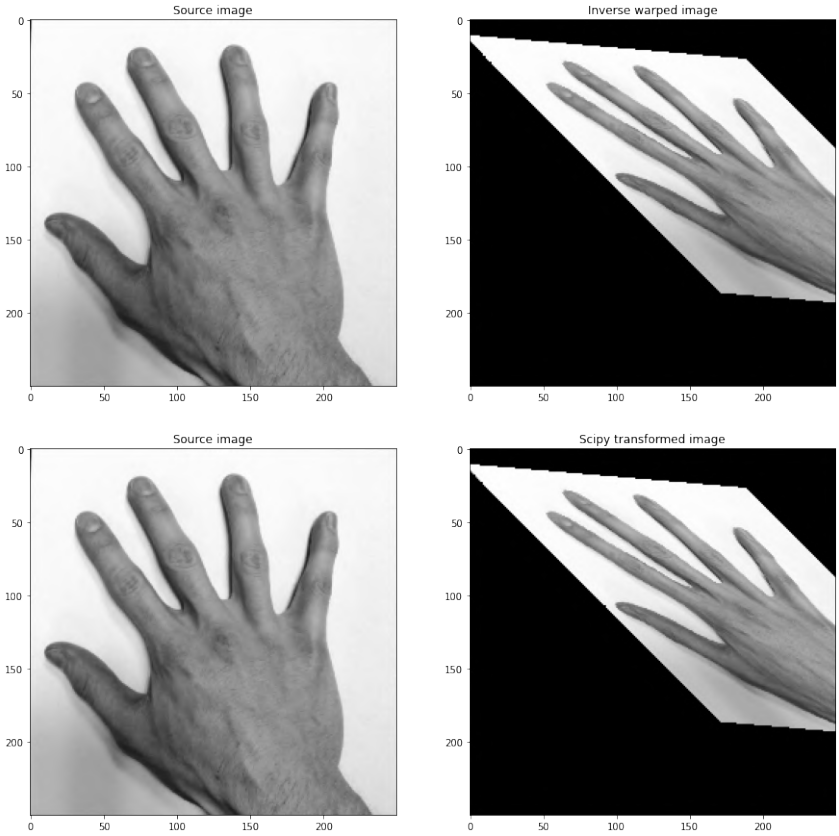
```
T=Tmatrix(scale=(1,1), rotationAngle=20, shearAngles=(10,0),translation=(0,0),elation
    =(0,0),reflection=(False,False))
```



We can see that there is indeed no more problem of pixel without value.

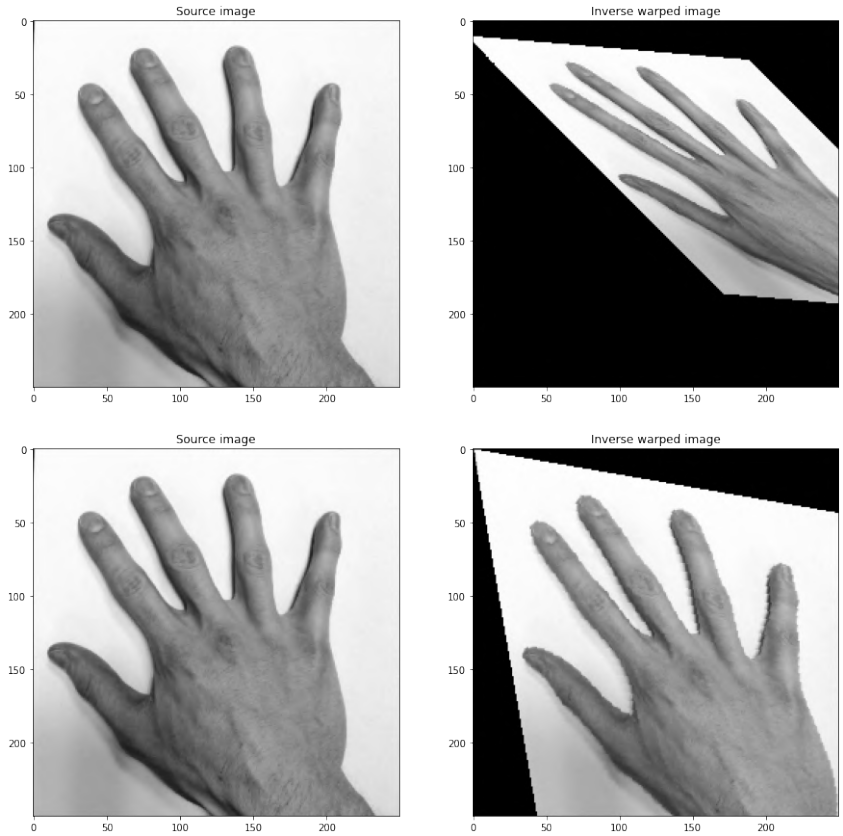
Here Here is the result on the transformation initially requested :

```
T=Tmatrix(scale=(1,0.5), rotationAngle=45, shearAngles=(34,0),translation=(10,-5),
    elation=(0,0),reflection=(False,False))
```

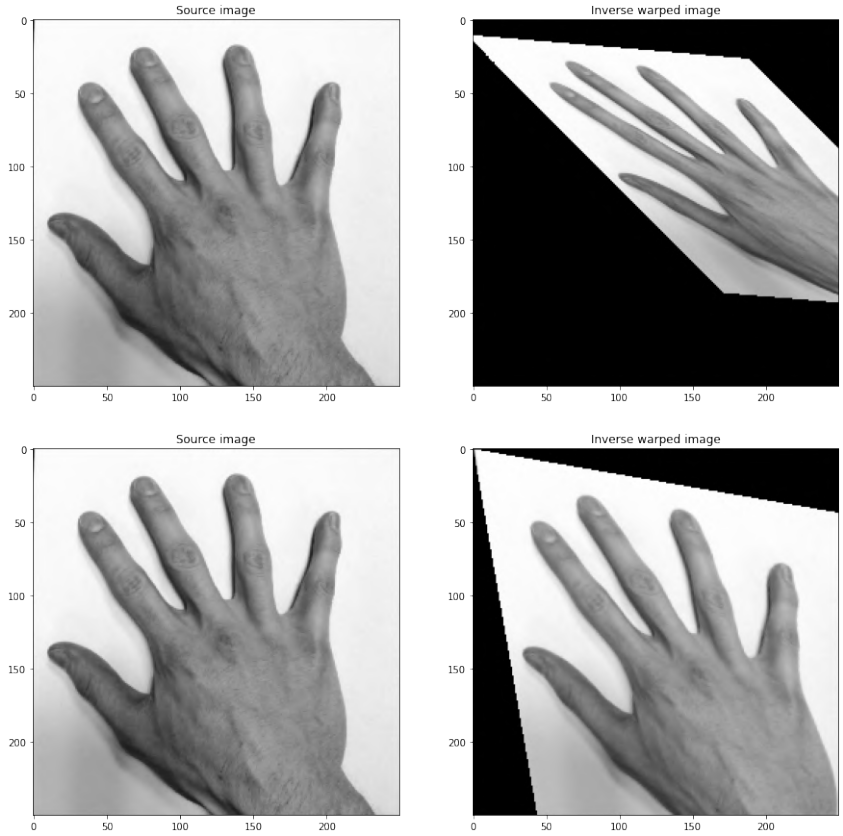


Comparaison of differents interpolations methods

- Nearest neighbors



- Bilinear



We can see that bilinear interpolation provide better results.

3 Optional part : Lucas-Kanade Algorithm - image registration

Unfortunately I did not have the time to develop Lucas-Kanade algorithm completely, i didn't understand how to compute the jacobian matrix.