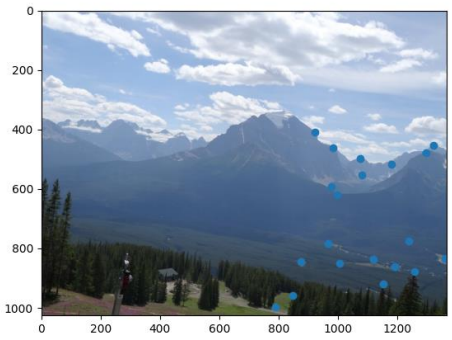
Computer Vision - EX1

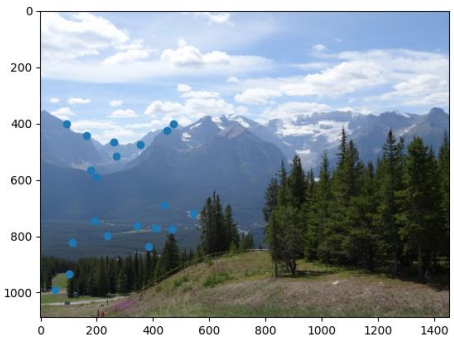
Lior Soffer 203135058

Idan Daniel 308088624

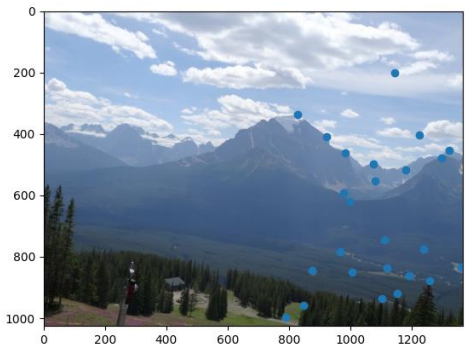
A: Homography computation

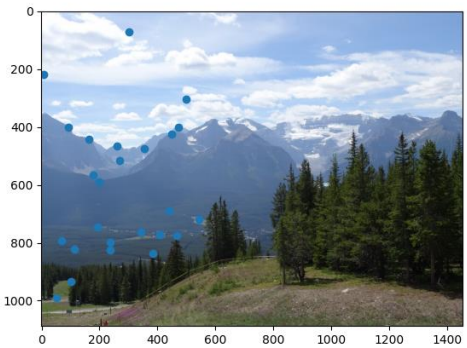
Without inliers: src | dst





With inliers: src | dst





A.1

projective transformation is a linear transformation on homogeneous 3-vector represented by a **non-singular** 3x3 matrix.

$$\begin{pmatrix} x'_1 \\ x'_2 \\ x'_3 \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$$

Or in short,

$$x' = Hx$$

Both x and x′ are homogeneous coordinate so x can have a scaling factor derived from matrix H. As a result, we can divide H by h_{33} and have this scalar position equal to 1 than.

Thus, we say that H is a **homogeneous matrix**. There're **8** independent ratios among the **9** elements of H, and it follows that a projective transformation has **8 degrees of freedom (dof)**.

Get the conversion matrix is as follow:

Selecting a section of image corresponding to a planar section of the world. Let the inhomogeneous coordinate of a pair of matching points in the world and image place be (x,y) and (x',y') respectively.

The projective transformation can be written in inhomogeneous form as:

$$\begin{pmatrix} x'_1 \\ x'_2 \\ x'_3 \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{bmatrix} xh_{11} + yh_{12} + h_{13} \\ xh_{21} + yh_{22} + h_{23} \\ xh_{31} + yh_{32} + h_{33} \end{bmatrix}$$

$$x' = \frac{x'_1}{x'_3} = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}}$$

$$y' = \frac{x'_2}{x'_3} = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}}$$

Each point correspondence (x,y) <-> (x',y') generates two equations for the elements of H as follow:

$$x'(h_{31}x + h_{32}y + h_{33}) = h_{11}x + h_{12}y + h_{13}$$
$$y'(h_{31}x + h_{32}y + h_{33}) = h_{21}x + h_{22}y + h_{23}$$

Notice that these are linear equations for elemnts in H. **4** point of correspondence lead to **8** of such linear equations in the entries of H, which are sufficient to solve H (since H has 8 dof) up to an insignificant multiplicative factor.
The only restriction is that the 4 points must be in "general position", which means that not 3 points are collinear.

if we use more points than this is an overdetermined problem and we except noise so we will use the least square method as describe:

$$|Ax|^2 - |Ae_1|^2 = (Ax)^T(Ax) - (Ae_1)^T(Ae_1) = x^T A^T Ax - e_1^T A^T Ae_1 \geq \lambda_1(\sum_i \mu_i^2 - 1) = 0$$

$$x^T A^T Ax = \sum_i \mu_i e_i^T A^T A \sum_j \mu_j e_j = \sum_i \mu_i e_i^T \sum_j \mu_j A^T Ae_j = \sum_i \lambda_i \mu_i^2$$

$$where\ \mu\ defenition\ is: x = \sum_i \mu_i e_i$$

Then we will choose the eigen vectors with the minimum eigenvalues.

2 (In code)

3

Naive Homography 0.0000 sec

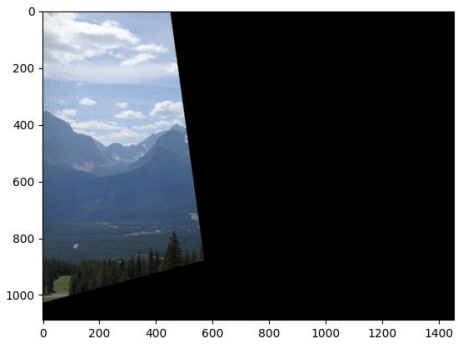
1.43457214]]e+00 2.10443232e-01 -1.27718679e+03[

1.34265153] e-02 1.34706123e+00 -1.60455872e+01[

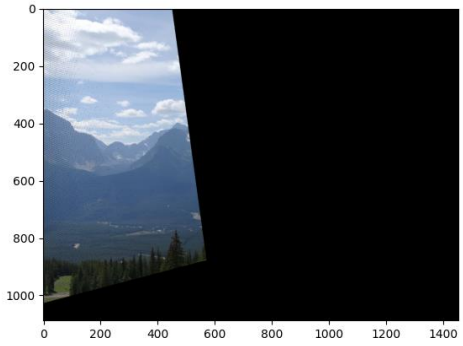
[3.79279298e-04 5.56523146e-05 1.00000000e+00]]

A.2

4 (In code)



5 (In code)

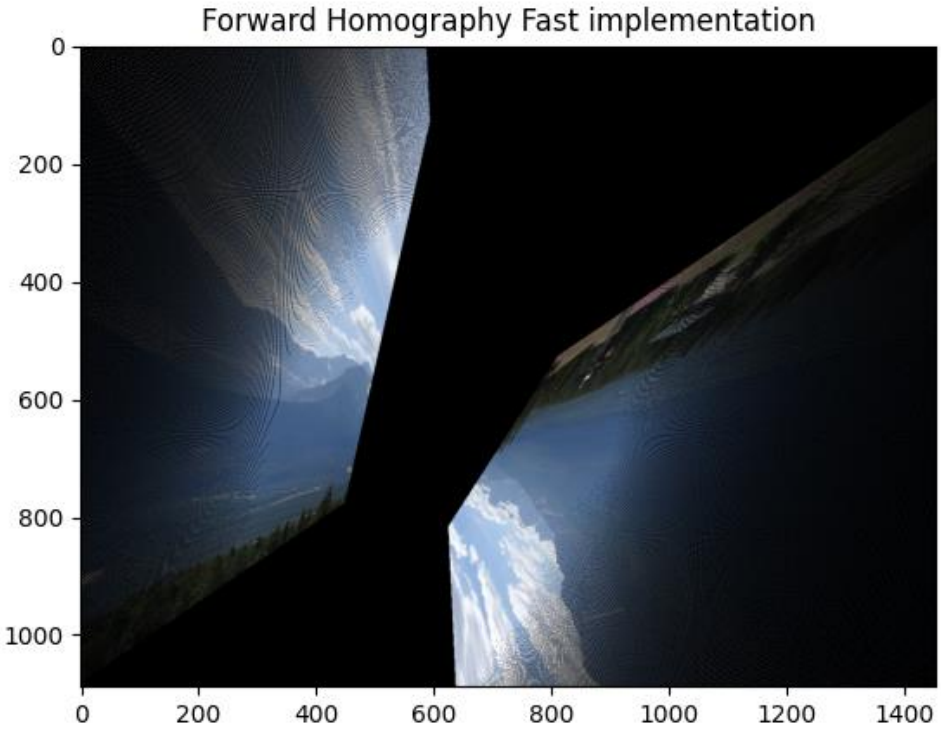


6

There are several problems with forward mapping:

a. It is possible that Hx is a float number. We solved it by rounding the results using the int() function.
b. There are pixels that got 2 values or more and each pixel can have only one value at the end – conflict.
c. We can see that the result has many "black areas" where the are pixels with no value after the transformation.

7



Yes, the results are different. By using all the points not the good ones we got in one image what looks like the source image separated to 2 images with different orientation.

8 (In code)

9 (In code)

10

The number of iterations N that is necessary to guarantee that a correct solution is found can be computed by

$$N = \frac{\log(1-p)}{\log(1-(1-\varepsilon)^s)}$$

s is the number of points from which the model can be instantiated
ε is the percentage of outliers in the data
p is the requested probability of success

Ref : Visual_Odometry_Tutorial

Suppose there are 30 match points and it is known that 80% of them are correct.

s = 4 (points needed for general homography)

ε = 20% = 0.2

for p = 0.9 we will need 5 iterations

for p = 0.99 we will need 5 iterations

11 (In code)

12

Naive Homography Test 9.1989 sec

[0.16, 21.329346060984314]

RANSAC Homography 2.0566 sec

[[1.43457214e+00 2.10443232e-01 -1.27718679e+03]

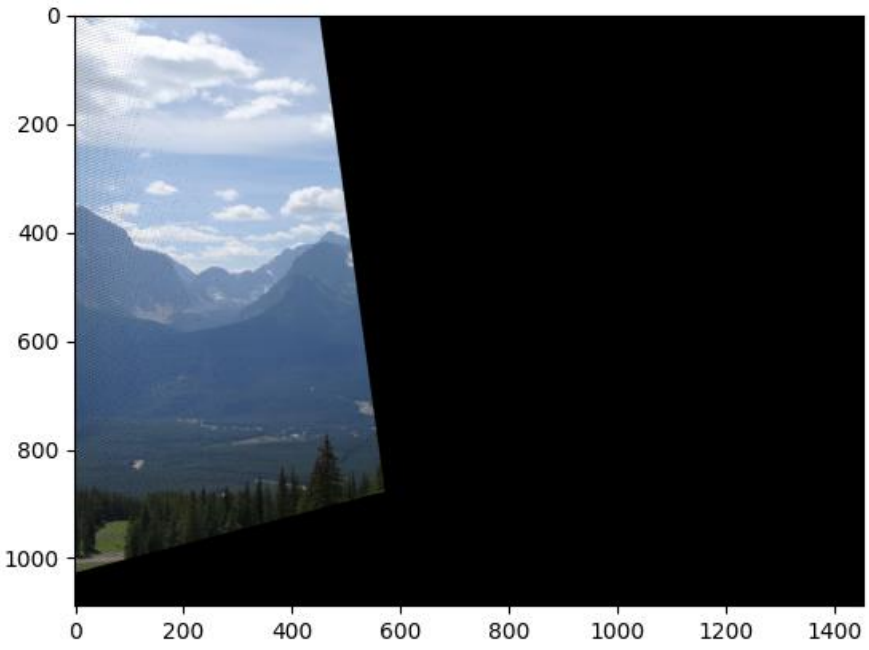
 [1.34265156e-02 1.34706123e+00 -1.60455874e+01]

 [3.79279298e-04 5.56523147e-05 1.00000000e+00]]

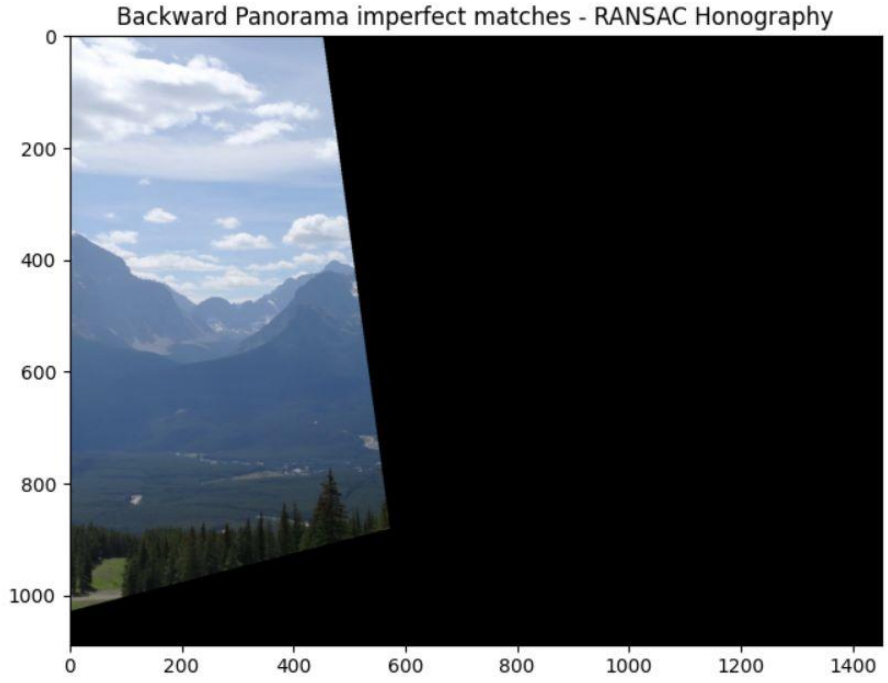
RANSAC Homography Test 2.0777 sec

[0.8, 1.9520284459063002]

Although we are using the bad matching points file, we can see that our forward mapping is much better than before and the result is similar to the perfect matching points forward mapping. RANSAC Algorithm were cleaning the inliers from the data.



13

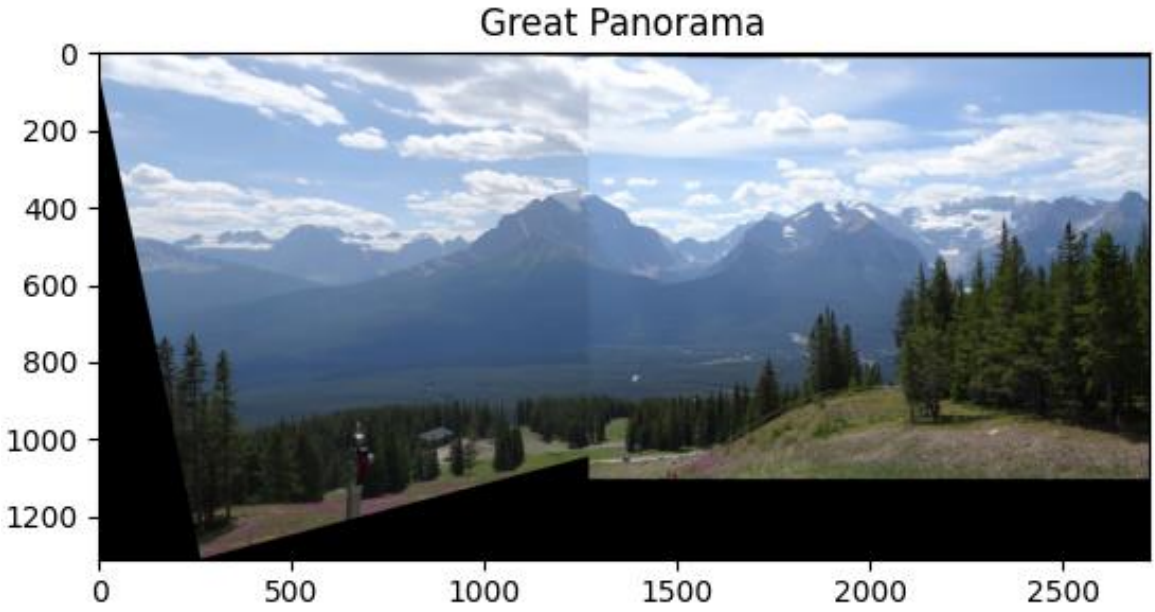


The image quality was improved, the image is much smoother and there are no black pixels as we saw at section 12. This is due to the backward mapping and the bilinear interpolation.

14 (In code)

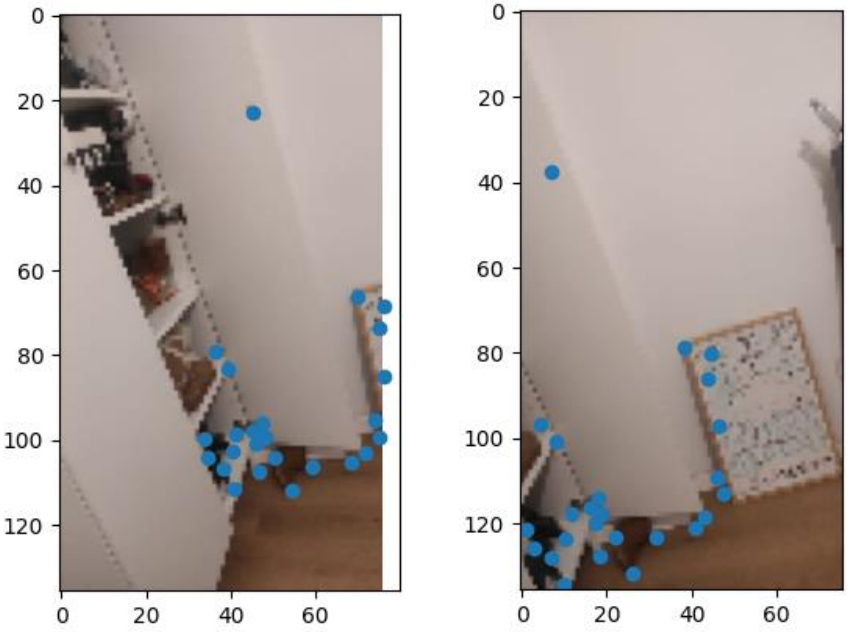
15 (In code)

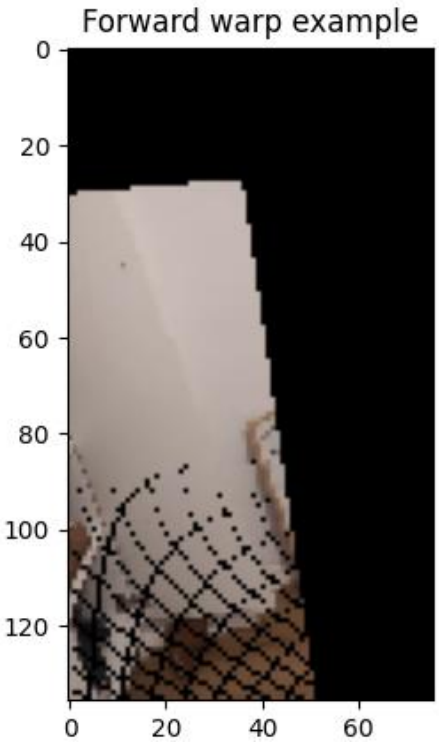
16

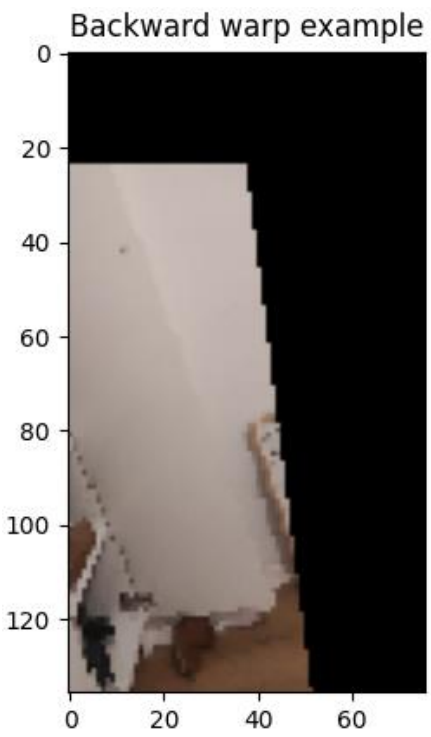


17

FIRST TEST (NO OUTLIERS)

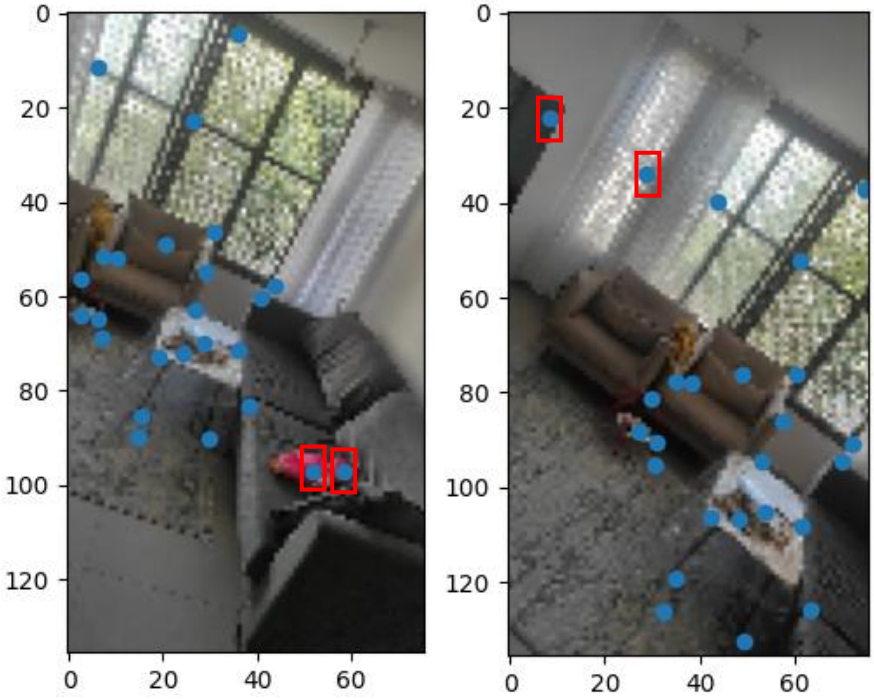


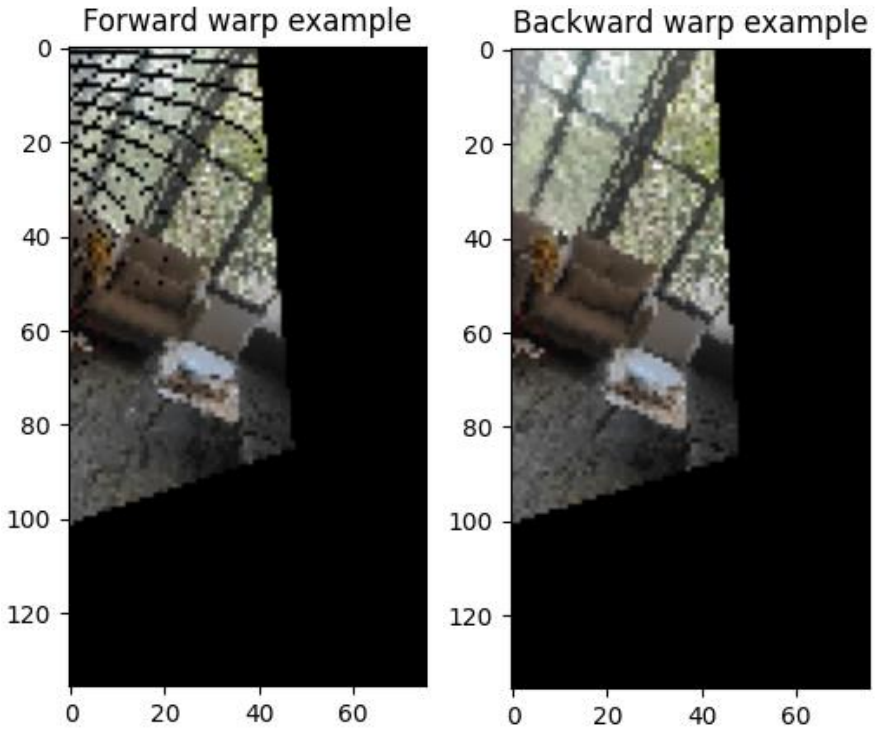

Forward warp example
0
20
40
60
80
100
120
0
20
40
60

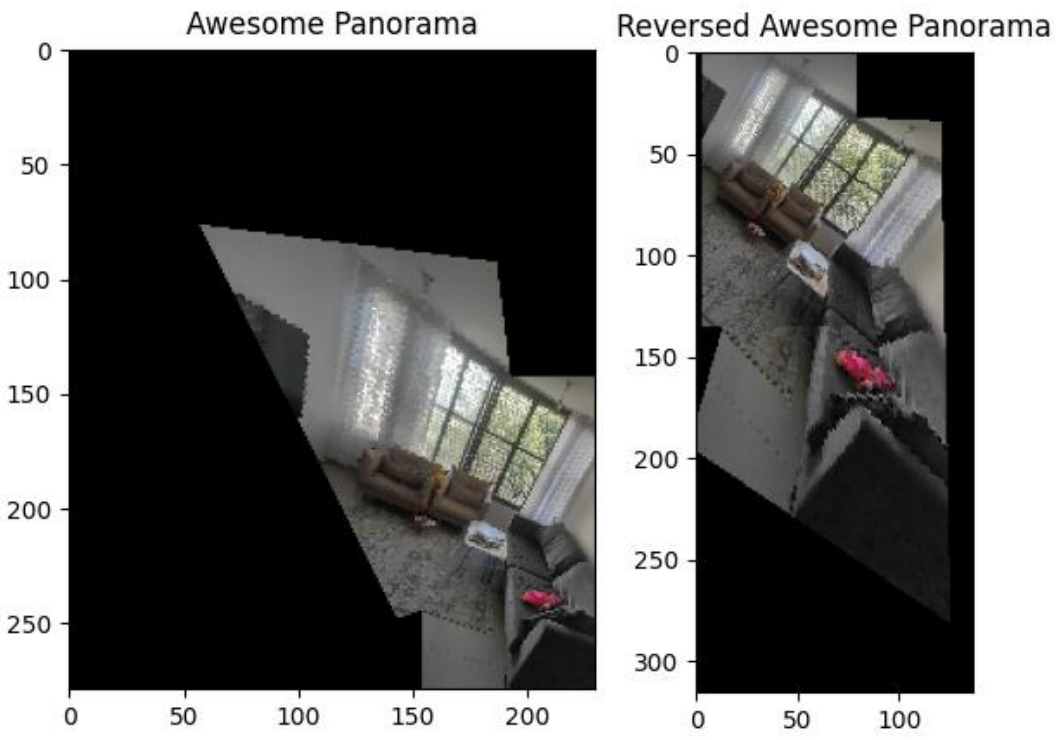

Backward warp example
0
20
40
60
80
100
120
0
20
40
60


Awesome Panorama
0
20
40
60
80
100
120
140
160
0
25
50
75
100

SECOND TEST (WITH OUTLIERS)






Awesome Panorama
Reversed Awesome Panorama
0
50
100
150
200
250
300