Computer Vision 2
Assignment 1

Martín de la Riva, Nuno Neto de Carvalho Mota
11403799, 11413344

April 2018

Introduction

The Iterative Closest Point (ICP) is an algorithm used to compute the rigid transformation that describes going from one point cloud to another. In this case point clouds are obtained from images and are composed of points corresponding to the positions on the surface of an object.
We will first implement a Vanilla ICP to correspond two point clouds obtained from artificial data, to test the efficiency of it. Then, using the point clouds corresponding to consecutive images of a human rotating, we will attempt to sequentially merge them, so as to obtain a unique point cloud that represents a 3D reconstruction of the person.

2 Iterative Closest Point - ICP

Our implementation of the ICP algorithm attempts to follow the one described in the assignment and the literature.
Out of the sampling methods described we did not implement the informative one. As for the others, they were implemented, with the only restriction being that the number of final points in the sampled source had to match the final number of points in the target (which meant that, if necessary, we would sample the target in order to obtain the correct number of points).
When applying the algorithm on the artificial data, it seems to give correct results converging after 20 iterations, Root Mean Squared error going to 0.02891 (To get results run *DemoICP.m*).

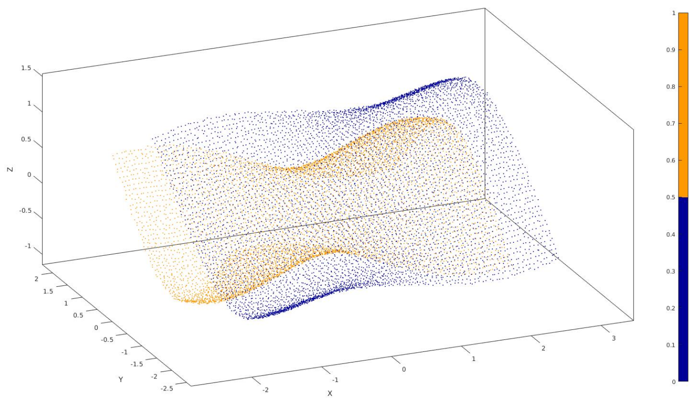

Figure 1: Original artificial data. Source in blue and Target in orange.

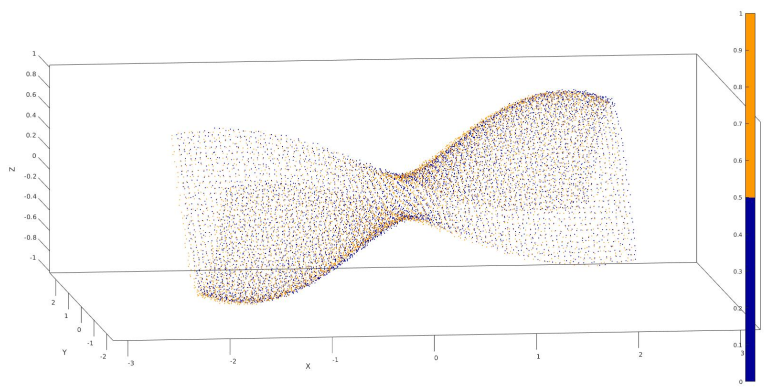

Figure 2: Source and Target after converging ICP algorithm.

3 Merging Scenes

Unfortunately we weren't able to produce any good results in any of the methods (when computing the merging for ALL images). We noticed that if we compute the merging for only one side of the person, let's say the left side (that is, a view with the camera pointing at the left arm of the person), we were able to get significantly better looking results. We aren't sure if this is due to a bug in our implementation of the ICP algorithm (though the tests performed on source.mat and target.mat seem to indicate otherwise) or if we are merging the

different frames in an incorrect fashion. One last alternative is that, due to the complexity of the task, it would be necessary to have implemented the informative sampling in order to obtain acceptable results. It seems that the problem is mostly focused when transitioning from either side (so, camera facing one of the arms) to either the back or the front. In order to reduce that problem in visualization, we compute the aforementioned merging of only one of the sides of the person (that is, without merging both sides into one cloud point).
We did not find a big difference when changing parameters such as subsampling method, number of points to keep on the subsampling, nor frame sampling rate. Also, we removed outliers from the cloud points, as it is necessary for ICP to work.
Below follow the results more detailed on each specific part.

3.1.a Does the merging produce sufficient result? Discuss why.

The following images show our results given all the images. When using all images the views show bad results when rotating the sides. We computed only some left and right sides, showing good results on them, and proving that the main difficulty occurs when rotating. We think this might be due to a sharp transition when the camera suddenly stops seeing either the front or the back of the body, making it very hard for ICP to properly match the 2 different consecutive clouds.

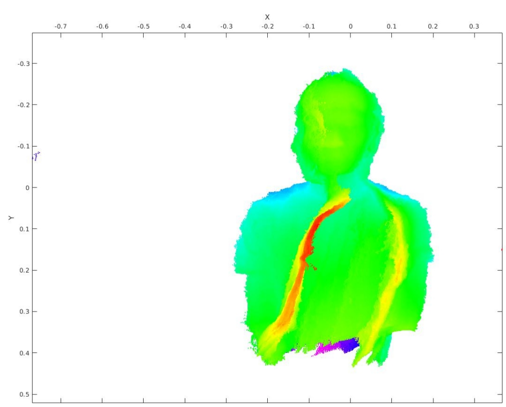

Figure 3: Frontal view of 3D model using ALL frames

On the other hand, if we compute the side view separately, it becomes easier to visualise the 3D reconstruction and the deformities when transitioning from on of the sides to either the back or the front (the top view provides the best visualisation, in this case):

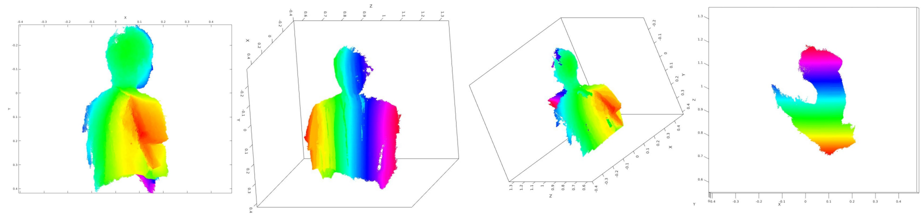

Figure 4: View of 3D model using images from the right side (camera pointing at right arm - frames 5-45).

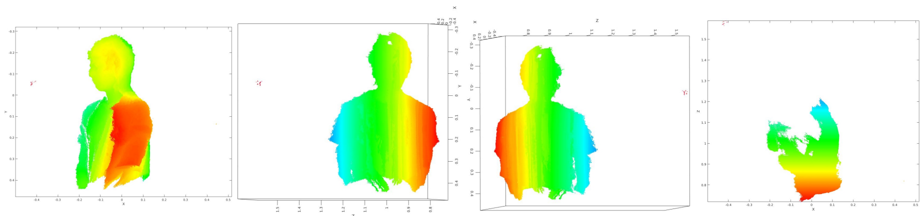

Figure 5: View of 3D model using images from the left side (camera pointing at left arm - frames 55-95).

3.1.b Does the camera pose estimation change?

We achieved similar results as in 3.1.a, as can be seen in the images below. Note, in the 3rd image, the clear deformation mentioned in 3.1.a becomes more evident, we think due to the frame skipping, which accentuates the problem described.

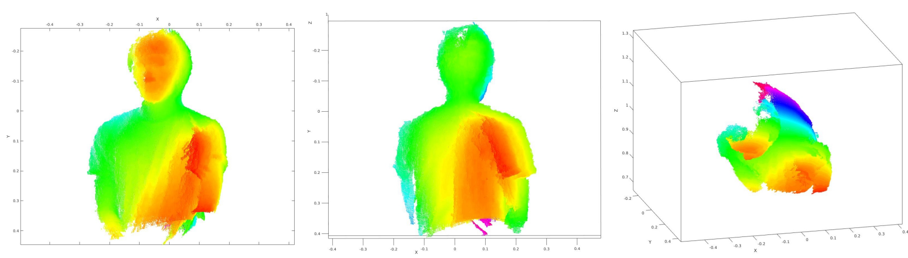

Figure 6: View of 3D model focusing on one of the sides. Front (computed left side), back (computed right side) and top-lateral view (computed left side).

3.2 Do the estimated camera poses change in comparison with the previous estimates (Section 3.1)? Does this estimation produce better results?

When using all previous frames the results get much better when using all images, although still showing some clear misalignments, specially on the sides as it can be seen in the top view.

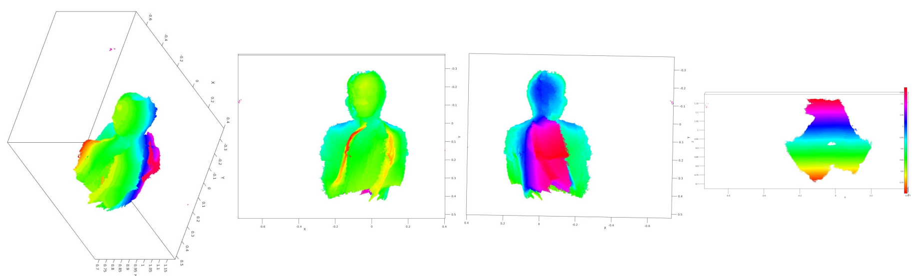

Figure 7: View of the 3D model using all the frames. Side, front, back and top views, respectively.

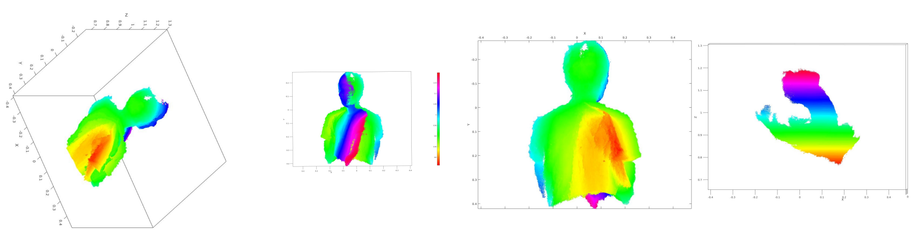

Figure 8: View of 3D model using images from the right side (camera pointing at right arm - frames 5-45).

4 Questions

4.1 What are the drawbacks of the ICP algorithm?

Depending on RMS as a measure does not ensure the best results, as although the points may be close the pose of the transformed data may be different to the target data.
The convergence to the algorithm to a global optimum is extremely dependent on the initial relative positions of the two different point clouds. As such, it can generally happen that the estimated transformation is not the best one.
Additionally, ICP can be very sensitive to noise, since it bases a lot of its

algorithm in finding the closest point (for each point) in the other cloud point. As such, it is important that outliers are removed (as previously mentioned).

4.2 How do you think the ICP algorithm can be improved, beside the techniques mentioned in [2], in terms of efficiency and accuracy?

Creating new measures that take into account the pose of the transformed data and target data.
Ignoring the "lateral parts" of a source (and target) image that are not expected to be present in the (correct) intersection of the two cloud points (for example, in the Case of the merging task here, after understanding the general movement of the rotation, we could try to sample more from the side that will be closer to the next target).

Running experiments

File *DemoICP.m* contains the code to compute the sample ICP (source.mat vs target.mat). The other experiments are contained in file *main.m.* The rest of the files include functions to compute this experiments, with self-explanatory names (such as *RefineRT.m*, *ICP.m* or *getPointCloud.m*).

Self-evaluation

Both students divided the work evenly both in documenting, implementing and writing the report.

Conclusion

To conclude, we saw the potential of ICP. Unfortunately we might have committed an error in the algorithm or the implementation, failing to get as good as results as we could have achieved. We expect that when using better forms of sampling (regardless of our current implementation) better results can be achieved.