Precise completion of deformable 3d shapes using SIREN

Project report

Name: Sahar Proter

Id: 206392433

Supervisor: Oshri Halimi

Semester: winter 2020-2021

Date: 18/4/2022

Introduction:

With the invention of VR headsets available to the public at a reasonable price, the possibility of VR chat has become a source of interest for many people, allowing people to virtually meet, char, travel and experience this limitless world in a more immersive way.

Making this world more immersive and life-like feeling can be done by scanning the user's body and uploading it in real time to the VR application, allowing the user to show and see others as if they were right in front f them.

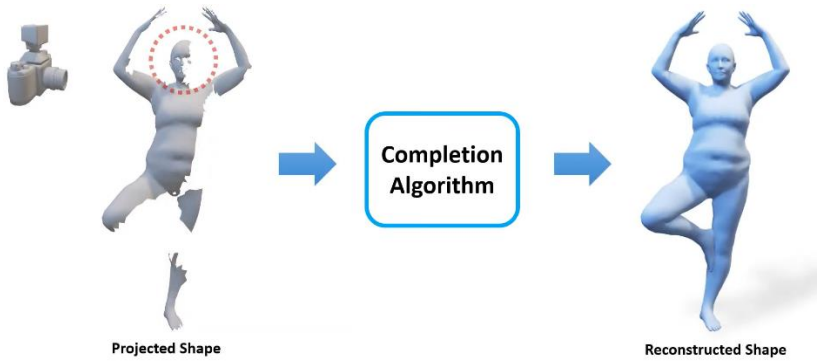
There are many ways to address this issue, such as:

- scanning the full body of the person from any direction, which requires an expensive and complex scanner.
- using image processing to recognize the different moving parts of the user body and moving an avatar in the 3D space the same way by using a known movement model for humans, which could be problematic for representing things like hair and cloths that have no known movement model.

In this project I chose a different approach to address this issue, by using precise completion of deformable 3D shapes.

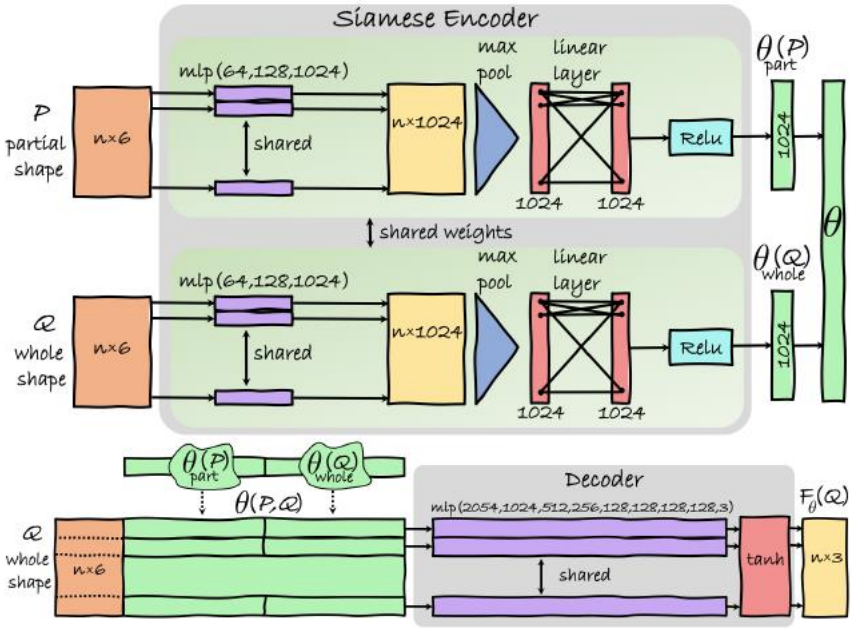
Precise completion of 3D shapes:

In the shape processing field, 3D data acquisition is usually achieved with depth sensors, capturing scans done from only one point of view, resulting in a deformed and incomplete point cloud, and a completion algorithm is needed to reconstruct the 3D shape in its new position, as was analyzed from the partial scanned shape.



Background:

This project was built from an existing project named "Towards Precise Completion of Deformable Shapes" which offers a solution to this problem using a Siamese PointNet network on a partial scan from a single view of a person in some pose P and a full shape of this person from a different pose Q, producing features, representing features of function F that transforms the point cloud Q to the full shape R which is the full scan of the person in seen in the point cloud Q. after that the features output are sent with point cloud Q to a new network, representing the function F seen above.

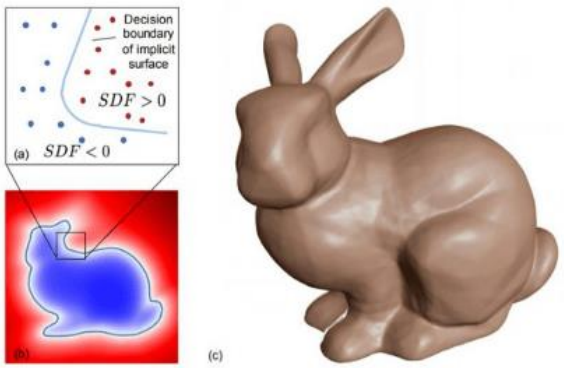


Different shape representations:

There are many ways of representing a 3D shape, with a very popular way known as point cloud.

This method represents the shape in the closest way to the raw scan and has the advantage of being order invariant and can be converted easily to other representations.

Another interesting way to represent 3D shapes is Signed Distance Function (SDF). An SDF is a function that gets a point in space and returns the shortest distance from the represented shape, with the sign of the value being whether the point is inside the shape or outside, having value 0 only on the surface of the shape.



In this project I chose to use the SDF representation for 3D shapes.

Why SDF?

SDF has a natural smoothing effect, which might come in helpful in shape reconstruction.

Another advantage of using SDF is how light it can be compared to big point clouds since it is a function representation of a shape and not a set of many points. This attribute might help speed up the training process of the net.

Another reason is the good results of SIREN on fitting a point cloud to an SDF using periodic activation functions, which was a big inspiration for me

to fuse this network into my implementation.

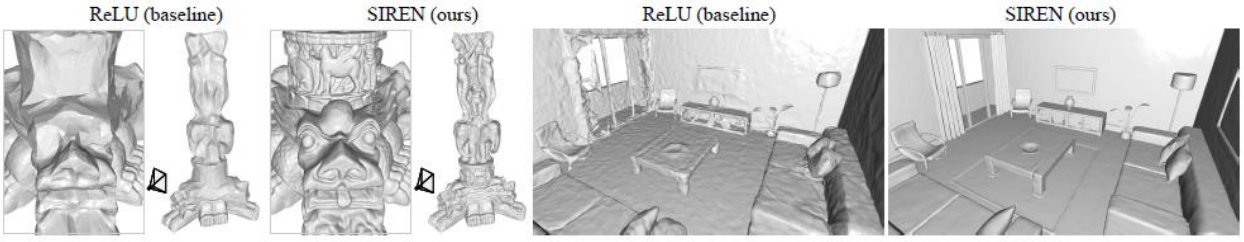


Figure 4: Shape representation. We fit signed distance functions parameterized by implicit neural representations directly on point clouds. Compared to ReLU implicit representations, our periodic activations significantly improve detail of objects (left) and complexity of entire scenes (right).

My solution:

My model is based on an encoder-decoder model. The encoder receives as an input a partial shape which is the shape we hope to complete.

The encoder is a PointNet network, configured the same way as in “Towards Precise Completion of Deformable shapes” as would be discussed later.

The output of the encoder is a 1024X1 latent, which is then sent as an input to the decoder.

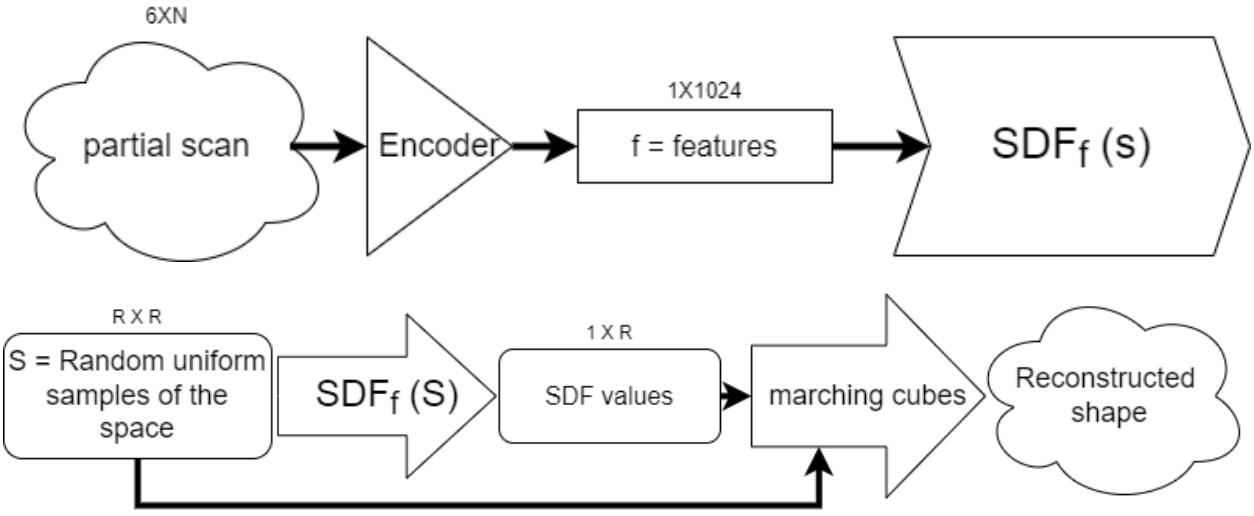
The Decoder is a SIREN network configured in a slightly different way.

The decoder receives as an input the encoder's output along with a random sampling of the 3D space, resulting in an input of size 1027X1.

The random sampling is a batch of random samples of the [-1:1, -1:1, -1:1] space with some, with some of the samples being randomly sampled from the ground truth point cloud (represents the shape’s surface and are used for the Loss calculation).

All the samples are than forwarded to the decoder which return’s the matching SDF value for each sample.

The chosen Loss function is the same used by SIREN, as would be specified later.



Chosen Configuration:

For the Encoder part I am using a batch size of 1 shape, and for the Decoder part I am using a batch of 3000 samples containing both on-surface samples from the ground truth and random samples from the space.

I am running 40 epochs; each is running 1000 iterations.

After each training epoch I am running a validation epoch with 50 iterations.

The samples consist of 50% on-surface points from the ground truth up to a max of 80% of the points on the point cloud, the rest being random samples.

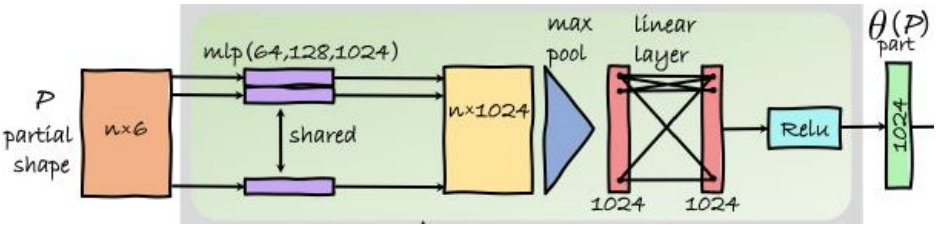
For the viewing part I am using a resolution of 128^3 for the 3D cube of SDF results transferred to the Marching Cubes Algorithm.

I am using Adam algorithm for optimization with a weight decay of 0.001 and a learning rate of 0.0001.

Encoder's Configurations:

The encoder is the same as used for "Towards Precise Completion of Deformable Shapes" , a PointNet network with 5 Convolutional layers of

sizes (6, 64, 128, 256, 512, 1024), having Batch normalization and RELU activation function in between except for the last one having max pool, a Linear layer and RELU.



Decoder's Configurations:

The decoder is using a F(φ)=Sin (α· φ) activation function, and after a few tests with SIREN I chose to keep the factor in the same value as used for SIREN implementation on git as α =30.

For the Decoder's initial weights, I am using the default normal distribution initialization.

The decoder is built from 5 Linear layers of sizes (1027,512,256,128,64,1) with the Sine function in between except for the last layer.

Data Preprocessing:

The network's input is a point cloud of the full shape Q and a point cloud of a partial scan P, both are represented as a set of XYZ and normals respectively.

The way SIREN works, it expects to be trained on samples of a const range of values, and it could not be trained on the same shape with different ratios (for example, one shape is twice the size of the other). This condition is derived from the way SDF is only defined for a specific range of the 3D space as the shortest signed distance from the surface of the shape.

In our case I chose to normalize the shapes to the range of [-1:1,-1:1;-1:1].

Dataset:

I am using FAUST dataset as used in the implementation of “Towards Precise Completion of Deformable shapes”.

Loss:

The used Loss function is:

$$\mathcal{L}_{\text{sdf}} = \int_{\Omega} \left\| \left|\nabla_{\mathbf{x}} \Phi(\mathbf{x})\right| - 1 \right\| d\mathbf{x} + \int_{\Omega_0} \|\Phi(\mathbf{x})\| + \left(1 - \langle \nabla_{\mathbf{x}} \Phi(\mathbf{x}), \mathbf{n}(\mathbf{x}) \rangle\right) d\mathbf{x} + \int_{\Omega \setminus \Omega_0} \psi\left(\Phi(\mathbf{x})\right) d\mathbf{x}$$

The above function is the same used for the SIREN model.

It consists of several losses:

1. The distance of the network’s gradient absolute value from 1 for all samples.
2. The distance from the decoder’s output on all on-surface points to zero.
3. The distance of the network’s output on all on-surface point’s gradient from the ground-truth normals.
4. $\psi(x) = \exp\left(-\alpha \cdot |\phi(x)|\right)$ for $\alpha \gg 0$, the chosen one is $\alpha = 100$, for the decoder’s output on all points not on the ground truth’s surface, used to panelize the distance of the SDF output of points not on the surface from zero.

The overall Loss is calculated as the weighted sum of those 4 losses with factors of 50, 3000, 100, 100 respectively.

Viewing Result

In order to convert from the SDF result on a specific scan to a point cloud we can see and understand, I uniformly sample the space [-1:1,-1:1;-1:1] and send it as input to the decoder, and send it to the decoder as input while using several batches of 32^3 samples each time.

I then we use the Marching Cubes algorithm on the full 3D cube of SDF values from the decoder to get the verts and faces, using them to create a ply file.

Results:

For this part there are several ways to test the quality of our reconstruction:

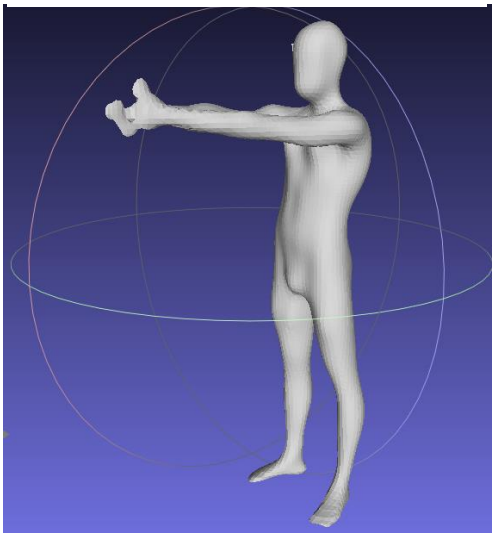
1. Compering SIREN's lowest loss value on the reconstructed shape in the same normalized space, since our network uses SIREN as the decoder and uses the same loss function, therefore the best loss result we can get is also SIREN's loss best result.
2. After transforming the final SDF to a point cloud we can use a simple L2 loss function from the ground truth using nearest neighbor.

I chose to compare the loss to SIREN's loss and in addition have a visual comparison between the transformed SDF and the ground truth point cloud.

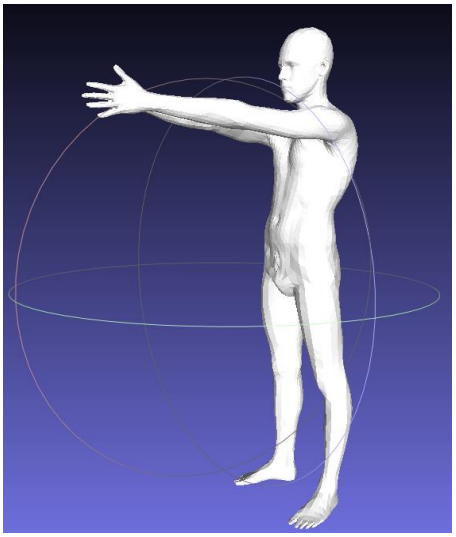
Experiments:

1. Ignoring the encoded partial scan (passing zeros as input to the decoder) and using a single shape as the training and testing set. The results of this experiment are as expected from the SIREN based Decoder with about the same Loss as the SIREN net on its own with loss of around 15-20.

Our output

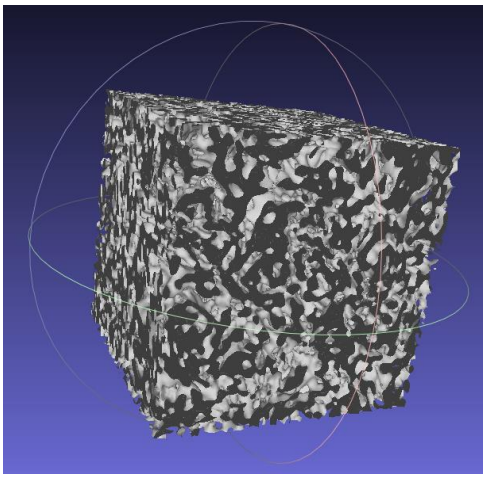


Ground Truth



2. Ignoring the encoded partial scan as before, but now using a single shape for a few iterations (300) and then passing to a new shape, each epoch having 4000 iterations in total.
   This experiment was showing fast fitting to the shape.
   When switching to a new shape, it showed a rise in the loss at the start but then it lowered again and fitted to the new shape with the expected loss of around 15-20, showing that once the network was fitted to a single shape, it was easier for it to fit to a new shape of similar size and over all shape but with a different pose.
   The results of this experiment are interesting because it shows the potential of this net to represent a new shape from a previous fitted shape.

3. using the encoder's output and having a single scan in the data set resulted in a meaningless shape that has no resemblance to any human form:



   A possible conclusion from the above results is that the network does not have the ability to distinguish between the first part of the input (xyz) and the rest (encoded shape), otherwise we would assume the network would ignore the encoder’s output by zeroing the weights of the layers related to those features and only look at the random samples like SIREN does, but it did not happen.

4. Trying the same above experiment again but setting the encoders output to be const nonzero (tensor of ones) resulted in similar result.

Conclusion:

Although the network failed to use the data from the encoded partial scan, I do see potential in this kind of representation for shapes as SDF and especially the use of SIREN's loss function for it's agility in moving from representing one shape to the other, the smoothing affect to the fitted shape that is derived from representing the picture as a continues function and how well and fast the shape is fitted using the sine activation function.

Future work

Looking at the network from the paper “Towards Precise completion of Deformable Shapes” an interesting idea for the future could be learning only the transition function F that transforms the full shape template Q using the feature from the encoder on P and Q.

A simple first experiment can be using a constant Q for the training and representing it as an SDF (a map from xyz to the SDF value) for the transition function F.

Using this method the network only needs to learn the transition function which might be easier to learn than a full shape for every partial scan.

Bibliography:

- “Towards Precise Completion of Deformable Shapes” – Technion, Stanford University, Sapienza University
- “Implicit Neural Representations with Periodic Activation Functions” – Stanford University
- “PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation” – Stanford University
- “DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation” – University of Washington, MIT, Facebook Reality Labs