Modifications to FMR (Feature-Metric Registration) Module

Orignial repo: <https://github.com/XiaoshuiHuang/fmr>

Paper: <https://arxiv.org/pdf/2005.01014.pdf>

The default model is still being used but the files have been modified to allow for the use of additional datasets.

1. train.py: In this file, three new choices were added for the argument for dataset type in the argument parser. Now, apart from 'modelnet' and '7scene', it can also take 'eth', 'sun3d' and ‘both’ (to train on eth and sun3d together) as inputs.
2. se\_math/mesh.py: A function named csvread() was added. This function reads csv point cloud and returns a mesh object representing the point cloud.
3. data/dataset.py: Two new classes were added, namely ETHDataset and SUN3D. These classes inherit from PointCloudDataset and are responsible for loading the corresponding datasets. In the constructor for each of these classes, they take the dataset path, transform function, and classinfo as arguments. They set the loader to the read the point cloud as implemented in se\_math/mesh.py, and specify the file pattern to match the files to load. They then call the constructor of the superclass PointCloudDataset with these parameters.
4. data/dataset.py: In the get\_datasets() function, it now includes handling for the newly introduced ETH and SUN3D datasets, based on the dataset type argument in train.py. There is also an option for handling both datasets we want to train on. The function takes an argument object that contains dataset information and details on what kind of dataset should be prepared testing or training.

The code first checks what the args.dataset\_type is, and then follows the code for the particular dataset. It then check on the args.mode to determine if we need data for training or testing. Depending on the mode, it sets the dataset path and the category file path. The split between training and testing is in the categories file. When mode is in training, the code returns training and testing (validation) sets with an 80% split.

Similar to functionality on the existing datasets, a set of transformations is applied. The function calls TransformedDataset which takes a point cloud and transformation, it returns the original point set, the point set with transformation applied along with the transformation matrix that was applied. This applies random transformations to augment the data for training. By randomly rotating and translating the input data, the model learns to be robust to these transformations, which improves its performance on unseen data.

Lastly these transformed datasets are then returned. In testing mode, the code does much of the same things, but it doesn't split the dataset into a training dataset and a testing dataset.

1. Categories files: These are text files used to select parts of the data for the ETH and SUN3D datasets. They define the categories of data that the datasets should be split into for training and testing. They are in data/categories.

FMR Example Usage For Training:

Python train.py -data eth –o ./result/eth​

Python train.py -data sun3d –o ./result/sun3d

Python train.py -data both