Steps for implementing VLocNet++

Use Keras to import ResNet50

# Focus on VLocNet Global Pose Network

Add three fully connected layers to the end of Global Pose Network’s ResNet, with the dimensions of fc1 – 1024, fc2 – 3 (xyz absolution position), fc3 – 4 (absolute quaternion rotation).

Initialise these weights with a Gaussian distribution

Load in 7scenes images, scale them to 341x256, subtract the pixel mean from a scene and subtract it from input images, then do a random crop of 224x244.

Replace all ReLUs with Exponential Linear Units (ELUs)

Apply feedback loop Pt-1

Copy across pre-trained weights to manually recreated Res5 and maybe ELUs

# Expand Model to include Odometry Network

Add additional two ResNet50 networks in a Siamese setup, concatenating results after the Res4 stage, then add the same three fully connected layer addition as in the Global Pose Network.

Share Res1, Res2, and Res3 weights between the Global Pose Network and the Lower Odometry part of the Siamese network.

Set up Model training with the Adam solver, a batch size of 32, β1 =0.9, β2 = 0.999 and ε = 10-10, initial learning rate λ0 = 10-4, dropout probability of 0.2, and understand and use the Geometric Consistency Loss Function with and .