Steps for implementing VLocNet++

Use Keras to import ResNet50

# Focus on 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).

Load in 7scenes - chess 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 previous position insertion between Res4 and Res5

Copy across pre-trained weights to manually recreated Res5

Expand loading function to load in all the other scenes in preparation for training

Train network with added fc layers and ELUs (no previous position insertion) to get a benchmark for the additional improvements (in progress: currently not working right. See graph on next page.)

Access NUbots synthetic dataset and train on rectilinear images

Capture our own dataset of images from igus

Train on our dataset to see some preliminary results

# 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.

Train the 2 network model with the multi-task learning technique from paper



