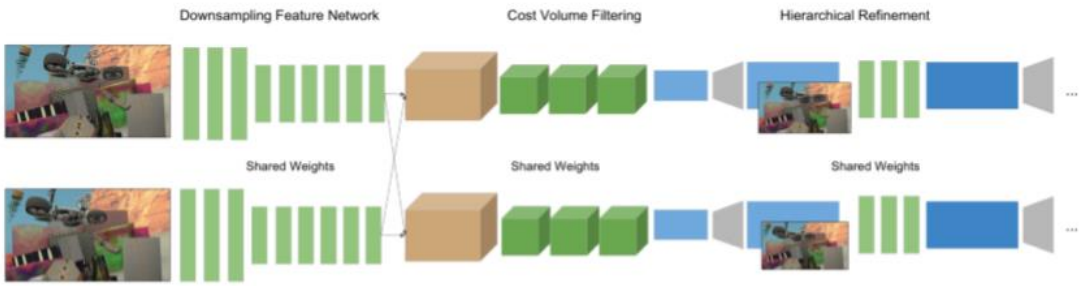
实时端到端双目系统深度学习网络 StereoNet 学习报告

背景介绍

立体匹配是一个重要的计算机视觉问题。它是关于如何通过双目看到的不同图片得到三维深度信息并生成深度图。谷歌研究员 Sameh Kkamis 提出的 StereoNet 是一种实时端到端双目系统深度学习网络。通过使用低分辨率的代价立方体（cost volume），StereoNet 比之前的网络例如 Alex Kendall 的 GC-Net 和 Jiahao Pang 的 CRL 的运算量要少，从而提高了运行速度，最高运行速度为 60FPS。为了仍然保持较高的空间精度，StereoNet 使用了边缘感知过滤层来实现高质量的输出。由于该项目代码未开源，我学习的代码来源是 Yili Xuan 在 Github 上完整复现的，代码链接：https://github.com/meteorshowers/StereoNet。该复现代码在终点误差(end point error)上略优于原论文中的指标，速度上能够以 30-50FPS 的速度推断。

算法分析



StereoNet 算法的第一步是特征提取也称为下采样（downsampling）。特征网络(feature network)使用的是孪生网络，特点是两图用相同的权值进行卷积。卷积的目的是缩小采样并保留主要特征。这个孪生网络的具体实现在下面的代码分析里，最后的输出是在每个像素都生成了一个 32 维度的特征矢量。这个特征矢量有一个大的感受野（receptive field），用于处理弱纹理区域，因为弱纹理区域特征不明显，需要大的面积输入来判断。再者这个特征矢量非常紧凑，因为它代表了经过卷积后的低分辨率图。

第二步是生成代价立方体通过计算两图对应像素的特征矢量的差值，即视差（disparity）。StereoNet 使用了过滤层操作，最终得到了每个像素与可能对应

的像素的视差代价立方体。因为立体匹配的特性，大部分效果提升是在低分辨率图的计算，而大部分运行时间是消耗在了高分辨率图的计算，所以 StereoNet 选择放弃高分辨率图，实验证明微小的效果牺牲能换来大量的运行速度的提升，且效果牺牲的精度被算法本身相较于传统算法的优越性弥补了。具体是因为传统算法使用的是赢家通吃策略，即选择两个特征矢量的最低欧几里得距离作为视差，但 StereoNet 选择通过多层卷积让网络自动学习调整计算视差的方法。所以 StereoNet 的精度是超过传统算法一个量级的。

第三步是从代价立方体中选择代价最小的视差作为该像素的结果来生成深度图。选择最小值的算法用的是 Softmax 函数因为该函数可导：

$$d_i = \sum_{d=1}^{D} d \cdot \frac{\exp(-C_i(d))}{\sum_{d'} \exp(-C_i(d'))}$$

最后一步是优化细节，因为使用了特征提取后的低分辨率图，输出的深度图缺乏细节。StereoNet 使用了边缘感知来优化深度图。为了避免棋盘效应（checkerboard effect），具体操作用的是双线性的上采样和卷积而不是转置卷积（deconvolution）。然后使用扩张卷积来扩大样本量但不扩大网络。最后得到一个单维度的深度图加到原图中得到预测深度图。

代码运行条件及所需资源

代码是用 python3.6 实现的，要求安装 PyTorch 作为机器学习库。运行机器的 GPU 需要支持 CUDA 运算平台，我把所有代码和数据都传到了谷歌云平台上运行，谷歌云使用的 GPU 是 NVIDIA Tesla P100,安装了 CUDA10 和 CUDNN7。下图是运行代码时的 GPU 数据：

在运行中显存占用基本不变，保持在 2950MB 左右。

代码及运行分析

代码中首先载入了训练和测试网络用的 Sceneflow 数据集，把输入的左图和

右图以及作为预期输出的视差图导入并排序。接下来创建迭代器用于遍历并创建训练日志。

然后建立 StereoNet 模型，模型包括特征提取，过滤生成代价立方体和边缘感知。特征提取使用了 3 个 5*5 的卷积层，6 个残差块（residual blocks）和 1 个 3*3 的卷积层，输出通道数保持为 32。对残差块的操作中使用了带有批正则化和矫正线性单元（Leaky ReLU）的卷积层（CBL）。卷积的作用是提取特征，批正则化的作用是提高训练速度，Leaky ReLU 是当作神经网络的激活函数。Leaky ReLU 相较于其他激活函数的优势是减少计算成本和更有效率的梯度下降，符合 StereoNet 降低计算量的目的。过滤层同样使用了 CBL，再用一个独立的卷积层输出。边缘感知则是使用了一次 CBL,6 次含扩张卷积的残缺块，和一次独立的卷积层输出。

值得注意的是算法的第三步是在 forward 函数中而不是在建立模型，这是因为在每一步传播的过程中都要选择代价最小的视差。另一点是虽然建立模型的三步各不同，但每一步都使用了 CBL 来提取特征和缩小数据量，也都使用了一个不含批正则化和激活层的卷积操作来得到符合输出规格的数据。

建立模型后就赋予初始权重，初始权重值是使用预训练模型得到的。再把优化器设为 RMSProp。RMSProp 是通过用 Gamma 衰减调整学习率来优化模型中梯度下降的过程。

正式训练开始。训练目标是最小化损失函数

$$L = \sum_{k} \rho(d_i^k - \hat{d}_i)$$

，损失函数代表了每个像素预测与真实视差的差值的总和。

训练分为 15 个周期，在每个周期内(epoch),先把模式设为训练模式，然后传入数据，执行包含了完整四步算法的 forward 函数，计算损失函数。然后由于 StereoNet 是端到端(end to end)的网络，所以直接把损失函数在网络中反向传播，再使用 RMSProp 优化的梯度下降更新权重。处理一定量的数据后输出这个周期中各个阶段（stage）的损失函数下降量，最后保存这个周期得到的深度图。15 个周期完成后输出每个阶段损失函数下降量的平均值。下图为我训练完毕后的输出值：

```
[2019-07-01 07:01:17 main8Xmulti.py:191] INFO     Average train loss = Stage 0 = 2.37   Stage 1 = 1.48  Stage 2 = 1.28
Stage 3 = 1.15
```

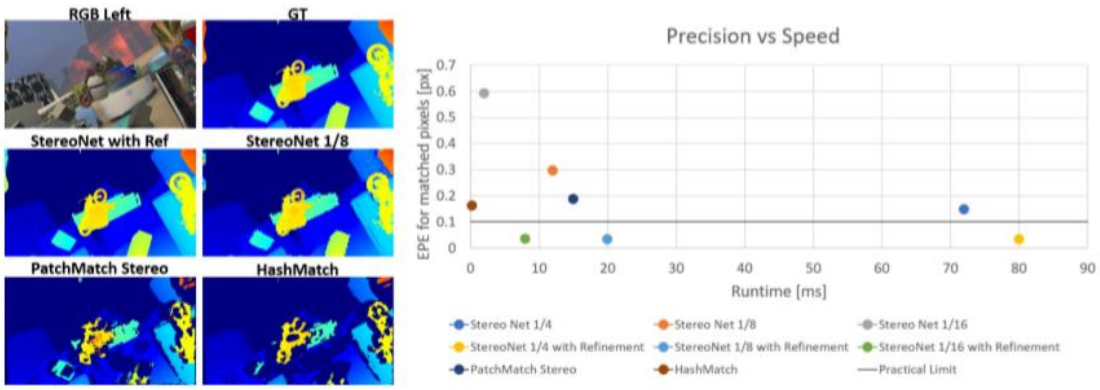
Github 上作者并未给出该值所以无法进行对比，但因为平均训练下降量为正，所以损失函数经过训练后有所下降，从侧面说明预测的深度图是在逐渐清晰并不断靠近真实深度图的。下图给出在多层次优化视差图过程中的每一个中间结果，从左到右可以看出从低分辨率图经过边缘感知过滤后得到了具有锐利边缘的精细视差图。

训练阶段之后就是测试阶段。测试阶段使用训练后的网络执行包含了完整 4 步算法的 forward 函数，把预测深度图和真实深度图对比，求出终点误差为预测视差与真实视差的差值的平均值。下图为我测试完毕后的输出值：

```
[2019-07-01 08:07:36 main8Xmulti.py:259] INFO     Average test EPE = Stage 0=2.72, Stage 1=2.29, Stage 2=1.97, Stage 3=1
.76
[2019-07-01 08:07:36 main8Xmulti.py:127] INFO     full training time =  6.185844 Hours
```

总训练时长约为 6 小时，各阶段终值误差的平均值为 2.185，原作者输出的终值误差为 1.48，所以此次测试的终点误差比理想的误差要高，推测是由于运行环境不同所导致的，但仍然得到了较精细的视差图。

算法精度比较



原论文中测试时将不同分辨率图下和有无优化的 StereoNet 和其他传统算法比较了精度和速度，结果如图所示。上图中灰线是实际情况下传统算法的能达到的最低精度约为 0.1px,而经过优化的 StereoNet 可以达到 0.03px，比传统算法例如 PatchMatch Stereo, HashMatch, UltraStereo 等低了一个数量级。原因

是因为 HatchMatch 和 UltraStereo 等算法使用的是基于条件随机场的方法，要解的方程是一个 NP 困难问题，所以得到精确解的时间较长（以秒为单位），所以 HatchMatch 和 UltraStereo 都追求得到近似解，损失了精度。而 UltraStereo 更缺少下采样，没有卷积操作，所以运算量较大，比 StereoNet 的运算速度慢。

与双目 SLAM 结合

SLAM（Simultaneous localization and mapping）即同时定位与地图构建：希望机器人从未知环境的未知地点出发，在运动过程中通过重复观测到的地图特征（比如，墙角，柱子等）定位自身位置和姿态，再根据自身位置增量式的构建地图，从而达到同时定位和地图构建的目的。双目 SLAM 即使用具有固定基线的双目摄像头获取环境信息，因为 StereoNet 是基于双目的图片来生成深度图，所以可以将 StereoNet 运用到双目 SLAM 的算法中帮助地图构建。【4】文中提出了一种快速双目 SLAM 算法（FBSLAM），可以获得较完整的环境信息和机器人轨迹信息，下图为 FBSLAM 的线程框图：

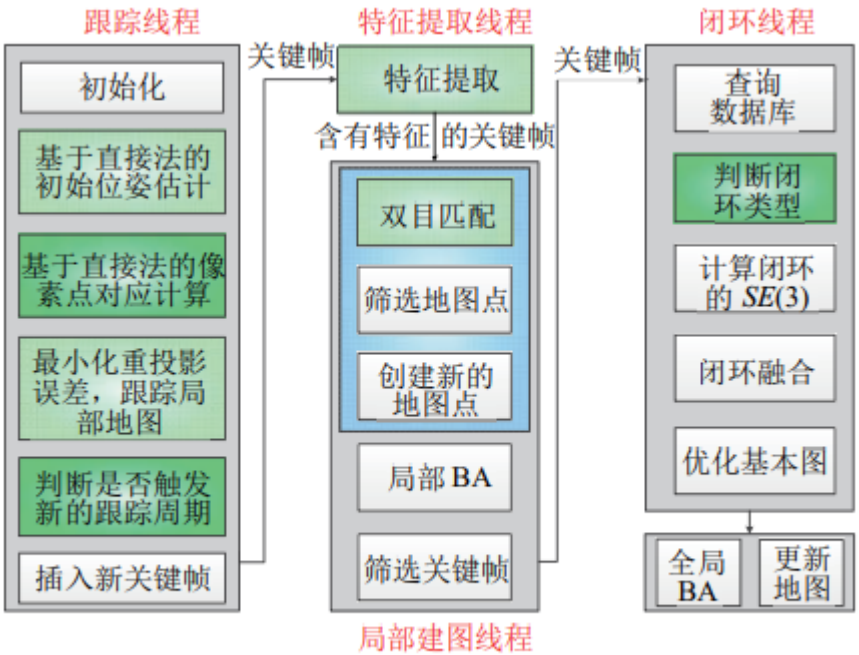


图 2　FBSLAM 线程

具体流程是在跟踪线程中获得初始的相机位姿估计和像素点的对应关系，在特征提取线程中获取图像特征，并与局部建图线程交互，采用特征法进一步优化

相机位姿和环境地图，对机器人被绑架后回到已探索环境的情况进行处理，在机器人被绑架后重新获得可靠的图像信息时，重新开启新的 SLAM 过程，若与被绑架前的环境发生闭环，则进行被绑架前后的闭环融合，得到更完整的环境信息和机器人轨道信息。

在跟踪线程中，StereoNet 可用在初始化，使用 StereoNet 进行立体匹配得到地图点的深度和 3 维坐标。之后 FBSLAM 是通过最小化光度误差得到初始的位姿估计和像素点对应关系，通过最小化重投影误差来跟踪局部地图，并判断图像帧是否为关键帧。可把光度误差和重投影误差导入 StereoNet,把视差换为这两个参数来计算损失函数，达到原算法的目标。在局部建图线程中，StereoNet 可以使用特征提取后的关键帧来进行双目匹配，筛选地图点并创建新的地图点。现在我还未复现 FBSLAM 的代码所以这些只是猜想，需运行实验后验证结果并与原算法精度及效率比较。

工作总结

在这段时间学习代码原理与运行 demo 过程中，遇到了一些困难，主要是在配置运行环境与理解论文原理，但在解决这些问题中学到了很多，不仅仅是深度学习相关的知识。在配置运行环境的过程中，由于本地机器没有独立显卡不支持 CUDA,学会了如何在谷歌云上搭建深度学习平台，其中困扰最久的问题是如何上传数据集。由于数据集较大的问题，尝试了从谷歌云下载因为数据是种子文件而失败，本地上传因为不稳定的连接而失败，用 ftp 服务器因为公司 ftp 服务器是内网而失败。最后在王宁同事的建议下采用了先上传至 Onedrive 网盘再传至谷歌云的方法，解决了数据传输不稳定的问题，使我对大批次数据传输与远程服务器运维有了更深的理解。在理解论文原理的过程中，不断积累背景知识，学完了 Coursera 上吴恩达的机器学习课程的 1-4 周，也学了 Coursera 上神经网络与深度学习课程的关于卷积神经网络的内容，为理解论文与实际代码打下了基础。

参考文献

[1] Sameh Khamis, Sean Fanello, Christoph Rhemann, Adarsh Kowdle, Julien Valentin, Shahram Izadi; The European Conference on Computer Vision (ECCV), 2018, pp. 573-590

[2] Kendall, A., Martirosyan, H., Dasgupta, S., Henry, P., Kennedy, R., Bachrach, A., Bry, A.: End-to-end learning of geometry and context for deep stereo regression. CoRR, vol. abs/1703.04309 (2017)

[3] Pang, J., Sun, W., Ren, J., Yang, C., Yan, Q.: Cascade residual learning: A twostage convolutional neural network for stereo matching. In: International Conf. on Computer Vision-Workshop on Geometry Meets Deep Learning (ICCVW 2017). vol. 3 (2017)

[4] 张国良，等。融合直接法与特征法的快速双目 SLAM 算法。机器人，2017，39(6): 879-888。