CNN-SLAM: Real-time dense monocular SLAM with learned depth prediction

CNN-SLAM:实时稠密单目 SLAM 及深度估计

【摘要】

鉴于深度卷积神经网络(CNN)在深度估计方面的最新进展，本文研究了如何用 CNN 和单目相机重建出准确且稠密的三维场景。我们提出了一个方法，它可让 CNN 预测的稠密深度图与单目 SLAM 估计的深度图自然融合。该融合方案在图像定位中由于传统的单目 SLAM 方法，在单目 SLAM 失效的时候（如低纹理区域）仍然能够很好地工作。我们展示了用深度预测来估计重建场景的绝对尺度，克服了单目 SLAM 无法得到绝对尺度这一主要缺陷。最后，我们提出了一个框架，它可将单幅图像中提取的语义标签和稠密 SLAM 高效地融合在一起，从单视角产生语义一致性(coherence，也可译作“语义相干”)的场景重构。两个基准数据集的评估结果显示了我们方法的鲁棒性和准确性。

【1 引言（部分内容）】

传统 SLAM 有几点缺陷：

1.工作范围小。

2.基于主动感知的 SLAM 方法在日光场景下表现很差。

3.单目 SLAM 无法获得绝对尺度，存在尺度漂移现象。若无法获得绝对尺度，则在 VR/AR 和机器人应用中都会遇到很大问题。

4.单目 SLAM 无法在纯旋转下工作，因为此时的 stereo-baseline 丢失，无法进行立体估计。

近来，通过机器学习方法从单幅图像估计场景深度的研究取得了新进展。其中，深度卷积神经网络(CNN)表现出色，其优点是可以得到高分辨、含绝对尺度的深度估计图，甚至在特征点稀疏或重复性纹理(repetitive patterns)的情况下也表现出色。由于场景的绝对深度可用机器学习通过样本训练得到，因此该方法不需要针对场景做出模型假设，也不需要几何上的约束。然而，此方法的缺陷是：虽然深度预测在全局上是准确的，但深度图中物体的边界处是局部模糊的，这样会丧失物体深度、形状的细节信息。

CNN-SLAM 的主要思想是：将深度估计和单目 SLAM 结合。为了解决深度图中边缘模糊的问题，我们将 CNN 预测的深度图作为稠密重建的初始猜测，并连续地通过直接法 SLAM 来优化（依靠 small-baseline stereo matching，与文献【4】的 LSD-SLAM 相似。LSD-SLAM 是典型的直接法)。非常重要的一点是，small-baseline stereo matching 拥有改善边缘区域深度估计的潜力。与此同时，CNN 预测的深度图含有绝对尺度，可为位姿估计提供更多约束条件，显著提高了位姿估计、路径和重建场景的精度。由于该方法可在纯旋转等恶劣情况下工作，因此 tracking 的稳定性大大增强。在实践方面，该框架可在 PC 上实时运行，CPU 和 GPU 同时运算，用 GPU 进行 CNN 深度估计，用 CPU 进行深度优化。

除可用于深度估计之外，CNN 还可成功地用于其他高维回归(high-dimension regression)的任务，其中一个典型的例子就是语义分割(semantic segmentation)。我们基于语义分割提出了 CNN-SLAM 的扩展版本：用像素级的(pixel-wise) labels 将语义 labels 和稠密 SLAM 准确、高效率地融合。

【2 相关研究工作】

SLAM 按传感器分，可分为基于深度相机的和基于单目相机的，按方法分可分为直接法和特征点法。

对于单目 SLAM（或者说纯几何的 SLAM），ORB 可以说是 state-of-the-art 的。

由于深度学习的发展，从单视角进行深度估计的研究得到了越来越多的重视。

【3 单目语义 SLAM】

我们在该部分详细地介绍 CNN-SLAM，图 2 显示了该算法的框架和流程。

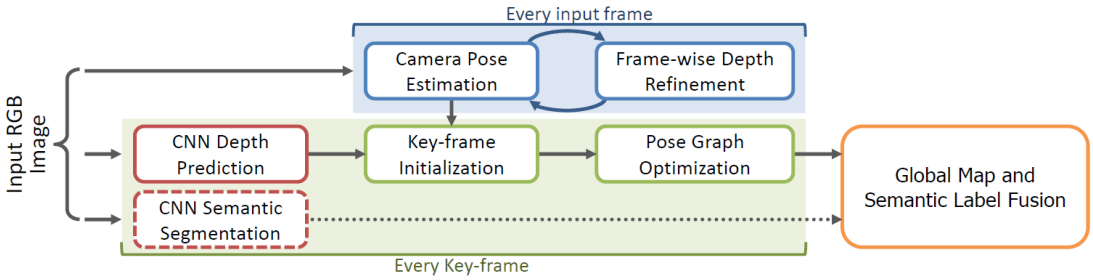

Figure 2. CNN-SLAM Overview.

我们使用基于关键帧的 SLAM 框架，并用文献【4】的方法（直接法、半稠密）作为 baseline，从视觉上显著的(visually distinct)帧集合中选出关键帧。这些关键帧的位姿将参与全局优化。与此同时，每输入一帧图像，程序就通过该帧与其最近关键帧的变换(transformation)来估计该帧的相机位姿。

为保持算法能以高帧率运行，我们只预测关键帧的深度图（用 CNN 生成）。如果当前的位姿估计和已有的关键帧位姿相差很多，则算法基于当前帧生成一个新的关键帧，并用 CNN 生成其深度图。另外，我们还通过对每个深度预测图进行像素级的置信衡量(confidence measure)，生成一个不确定性图(uncertainty map)。由于一般情况下，SLAM 用的相机和拍摄 CNN 训练图像的相机并不同，因此我们提出了将深度图规范化（normalized）的方法。这样，对不同内参的相机拍摄的视频流都有较好的鲁棒性。当进行语义 label fusion 时，我们用另一个卷积网络对输入帧进行语义分割的预测。最后，生成关键帧的位姿图，来全局优化它们的相对位姿。

该框架一个很重要的环节（也是我们的主要贡献），就是通过 small-baseline stereo matching 来改善 CNN 预测的关键帧深度图。我们通过在关键帧和输入帧之间最小化 color consistency 来实现。深度图中的边缘区域将会主要通过该环节得到优化，epipolar matching 可提供更高的准确性（这部分内容将在 3.3 和 3.4 中介绍）。优化后的深度的传播方式受各深度值的不确定性的影响，这种不确定性由我们专门提出的置信衡量而得出（在 3.3 中定义）。该框架的每一个步骤都将在以下小节中详细阐述。

【3.1 相机位姿估计】

CNN-SLAM 的位姿估计受 LSD-SLAM 的关键帧法的启发而得。系统保存了一系列关键帧 $k_1, k_2, \ldots k_n$ 并将它们作为结构元素参与 SLAM 重建。每个关键帧 k_i 都与相应的关键帧位姿T_{k_i}、深度图D_{k_i}和深度不确定图U_{k_i}对应。与【4】不同的是，我们的深度图D_{k_i}是稠密的，它由 CNN 进行深度预测而得（详见 3.2 节）。D_{k_i}估计了每个深度值的 confidence。【4】将D_{k_i}初始化为一个大的、恒定值。与之不同的是，我们根据深度预测的 measured confidence 初始化D_{k_i}。我们将深度图的元素标记为 $u=(x, y)$，并用$\dot{u}$作为 u 的齐次坐标表示(homogeneous representation)。

我们对每一帧 t 估计它的相机位姿$T_t^{k_i} = [R_t, t_t]$，即离 t 最近的关键帧k_i到 t 的变换。它由 3*3 的旋转矩阵R_t和 3D 平移向量t_t组成。该变换由最小化当前帧图像I_t最近关键帧图像I_{k_i}的光度残差(photometric residual)，并通过高斯-牛顿优化目标函数得到：

$$E\left(T_t^{k_i}\right)=\sum_{\tilde{u}\in\Omega}\rho\left(\frac{r\left(\tilde{u},T_t^{k_i}\right)}{\sigma\left(r\left(\tilde{u},T_t^{k_i}\right)\right)}\right)$$

ρ 是 Huber 范数(Huber norm)，σ 是衡量残差不确定度的函数（详见【4】）。r 是光度残差，它的定义如下：

$$r\left(\tilde{u},T_t^{k_i}\right)=I_{k_i}(\tilde{u})-I_t\left(\pi\left(KT_t^{k_i}\tilde{V}_{k_i}(\tilde{u})\right)\right)$$

由于深度图是稠密的，因此从效率上考虑，我们仅对颜色梯度高的区域中的像素子集计算光度残差，由$\tilde{u}$表示，$\tilde{u}\subset u\in\Omega$。$\pi$是将 3D 点投影到 2D 图像上的投影函数：

$$\pi([xyz]^T)=\left(\frac{x}{z},\frac{y}{z}\right)^T$$

$V_{k_i}(u)$表示顶点贴图(vertex map)中的 3D 元素，该顶点贴图由关键帧深度图计算生成：

$$V_{k_i}(u)=K^{-1}\dot{u}D_{k_i}(u)$$

K 是相机的内参矩阵。

一旦得到了$T_t^{k_i}$，当前帧在世界坐标系下的的相机位姿即可用$T_t=T_t^{k_i}T_{k_i}$计算。

【3.2 基于 CNN 的深度估计和语义分割】

每当生成一个新的关键帧时，CNN 便产生一幅对应的深度预测图。我们使用的深度预测方法属于目前的最高水平（文献【16】的方法），它基于 Residual Network (ResNet)延伸出一个全卷积网络。该架构的第一部分基于 ResNet-50（文献【9】），由 ImageNet 训练生成的权重对该部分进行初始化。框架的第二部分将 last pooling 和 fully connected layers (由 ResNet-50 提出)替换为一系列上采样的 blocks，这些 blocks 由 unpooling 和卷积层组成。在上采样之后，drop-out is applied before a final convolutional layer which outputs a 1-channel output map representing the predicted depth map. The loss function is based on the reverse Huber function（文献【16】）.

我们参考其他利用相同的架构进行深度预测、语义分割的成功范例（文献【3，29】），也训练了该网络，用于从 RGB 图像中预测像素级的语义 label。我们修改了网络，使它的输出通道数和类别数相同，并用了 soft-max 层和 cross-entropy loss function，通过反向传播和 Stochastic Gradient Descent(SGD)最小化此 loss function。需要指明的是，虽然原则上可以使用任何语义分割算法，但我们此工作的目的是展现帧级的分割图是如何成功地和单目 SLAM 框架融合在一起的（详见 3.5 节）。

【3.3 生成关键帧和位姿图优化】

使用预先训练的 CNN 来预测深度的局限之一是：如果 SLAM 所用传感器的内参和拍摄训练集图像的相机的内参不同，那么在 3D 重建时绝对尺度的估计必然不准确。为解决该问题，我们用当前相机的焦距f_{cur}和训练集所用相机的焦距f_{tr}的比值，通过 CNN 调整深度：

$$D_{k_i}(u)=\frac{f_{cur}}{f_{tr}}\widetilde{D}_{k_i}(u)$$

$\widetilde{D}_{k_i}$是通过关键帧图像I_{k_i}直接由 CNN 退化得到的深度图。

图 3 显示了用上式校正的有效性，我们使用的是 *ICL-NUIM* 数据集。如图所示，经过校正后，深度图和轨迹的准确性都有明显的提升。

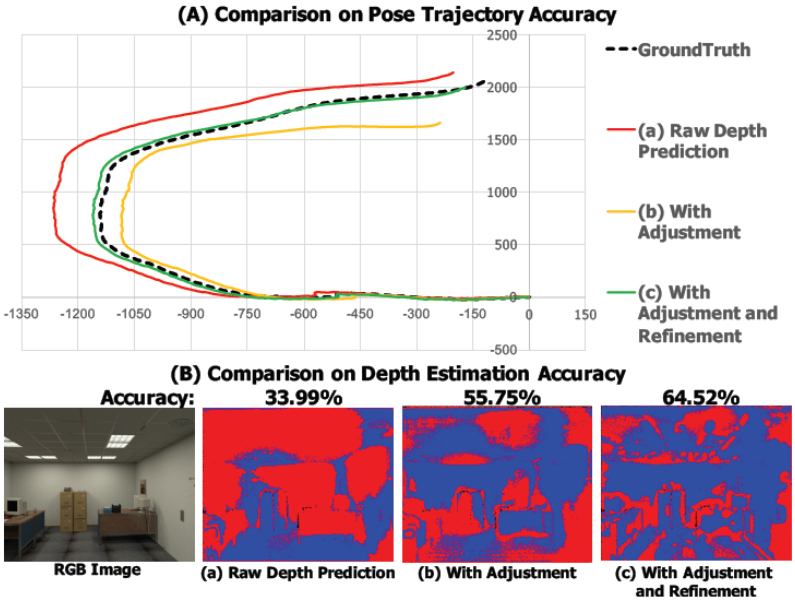

Figure 3. Comparison among (a) direct CNN-depth prediction, (b) after depth adjustment and (c) after depth adjustment and refinement, in terms of (A) pose trajectory accuracy and (B) depth estimation accuracy. Blue pixels depict correctly estimated depths, i.e. within 10 % of ground-truth. The comparison is done on one sequence of the *ICL-NUIM* dataset [8].

另外，我们将每个深度图D_{k_i}和不确定图U_{k_i}联系在一起。文献【4】将U_{k_i}的每个元素初始化为一个大的、恒定值。由于 CNN 可在不依赖 temporal regulation 的情况下产生深度图，因此我们根据当前深度图D_{k_i}和最近关键帧中对应的场景点(respective scene point)的差别，计算出置信值，从而将U_{k_i}初始化。这样以来，置信值衡量了在不同帧中，每个深度估计值的一致性程度(this confidence measures now coherent each predicted depth value is across different frames)。高置信值的元素在连续优化中将比文献【4】中的更快、更有效。

具体而言，我们将U_{k_i}定义为当前关键帧k_i的深度图D_{k_i}和最近关键帧k_j的深度图对应元素差的平方，k_j的深度图由估计的$T_{k_j}^{k_i}$(k_i到k_j的位姿变换矩阵)得到：

$$U_{k_i}(u) = \left(D_{k_i}(u) - D_{k_j}\left(\pi\left(KT_{k_j}^{k_i}V_{k_i}(u)\right)\right)\right)^2$$

为进一步提高每个新初始化关键帧的准确性，在经过它们和输入帧的优化后，我们将其D_{k_i}、U_{k_i}和由最近关键帧产生的 D、U 融合(we propose to fuse its depth map and uncertainty map with those propagated from the nearest key-frame after they have been refined with new input frames)（显然，我们不会对第一个关键帧这样操作），深度优化的详细流程详见 3.4 节。为此，我们首先定义了从最近关键帧k_j传播的不确定地图(propagated uncertainty map)：

$$\widetilde{U}_{k_j}(v) = \frac{D_{k_j}(v)}{D_{k_i}(u)} U_{k_j}(v) + \sigma_p^2$$

$$v = \pi\left(KT_{k_j}^{k_i}V_{k_i}(u)\right)$$