End-to-End, Sequence-to-Sequence Probabilistic Visual Odometry through Deep Neural Networks

传统VO，当训练数据（有准确的轨迹信息）存在的时候，无法从中获益，训练数据只是用在debug阶段。

有些人使用机器学习的方法(GP, SVM)去学习，但效果不好，传统的机器学习能力有限，而且不是从原始图像学习，而是从光流或一些从图像中提取的特征。

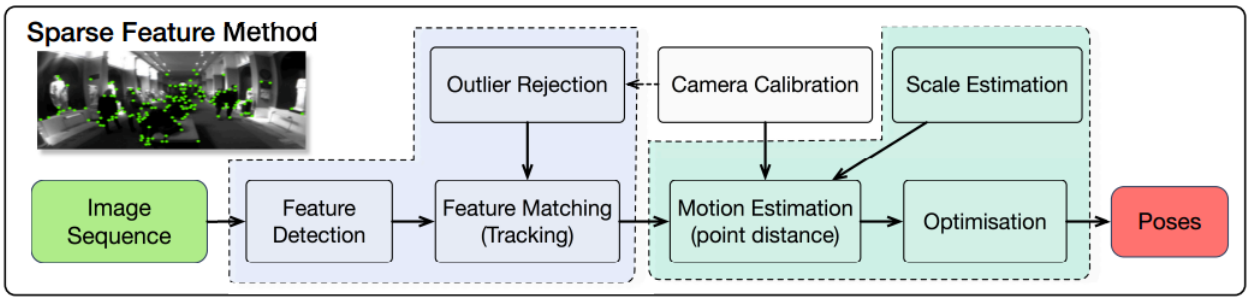
与分类和检测不同，DNN用于VO需要学到新的特征表示，这个特征中封装了几何信息.

两种VO类型： geometry based(multiple-view geometry and photometric consistency) methods and learning based methods.

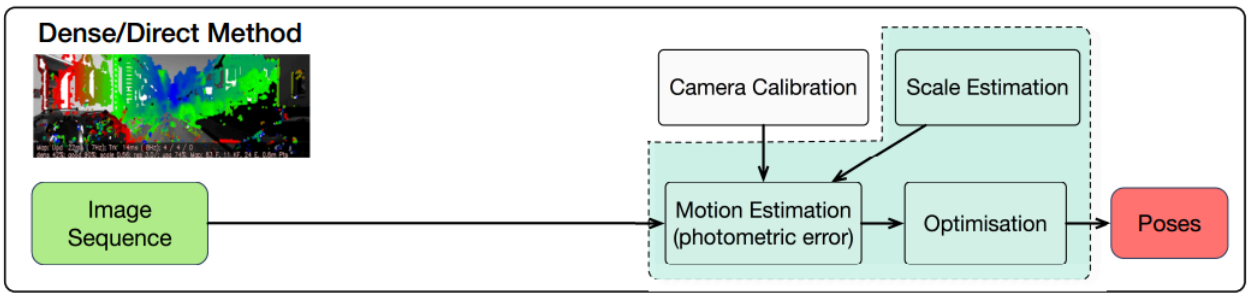
geometry:

geometry based methods relying on geometric constraints extracted from imagery to estimate motion.

can be divided into sparse feature based method and dense/direct methods according to whether image are pre-processed with sparse feature extraction and matching.



(a) Sparse feature based method.

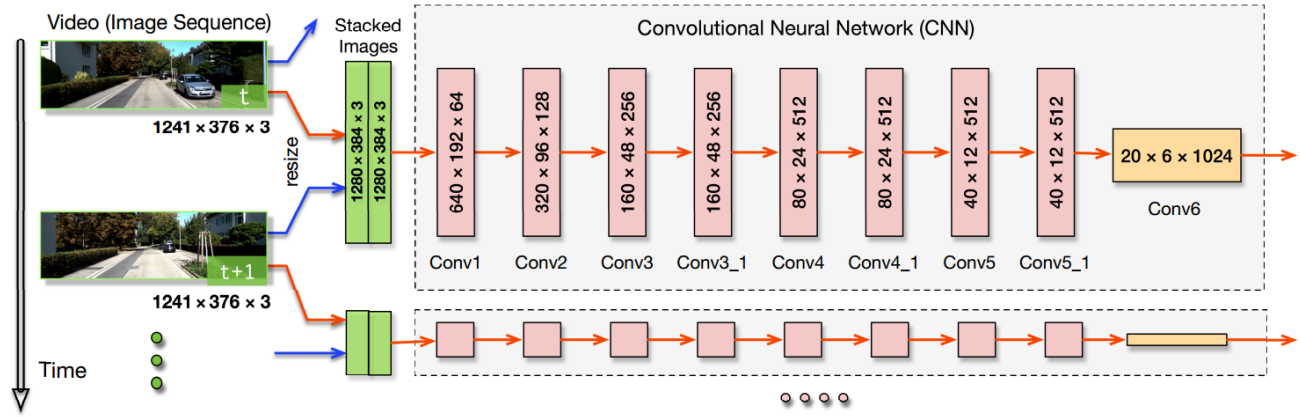


(b) Dense/Direct method.

For sparse feature based methods, first, salient feature points are extracted from raw images and matched to find feature point correspondences among images.Then poses are estimated based on geometric relationship of feature point.To mitigate the effects from noise and outliers, outlier rejection are necessary.

缺点：只用了静态物体特征，没用使用整张图像。对于outliers很敏感。

CNN:



Conv6的输出作为RNN的输入。

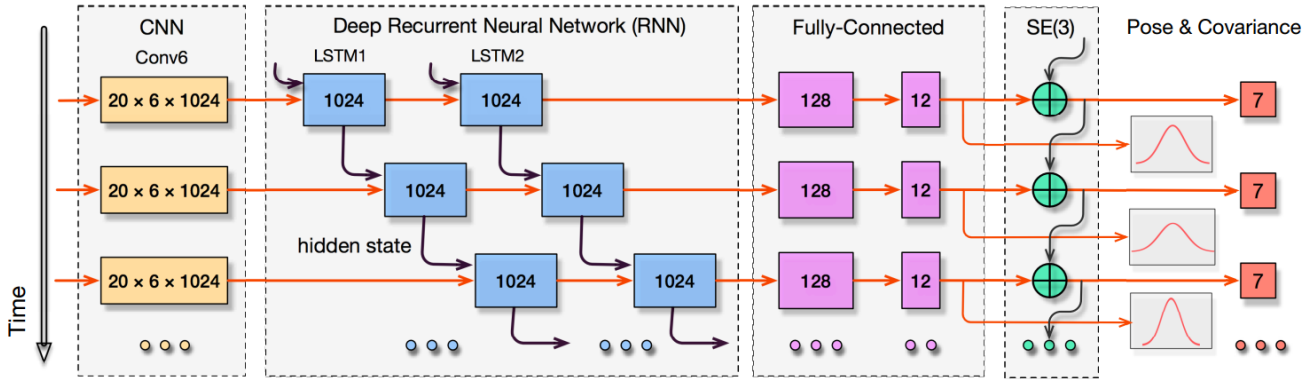


Fig. 3. Architecture of RNN, fully-connected (FC) layer and **SE**(3) composition layer of proposed monocular VO system (continued from Figure 2).

SE(3)层不理解，需要看论文
fc中12 代表估计的x,y,z,roll,pitch,yaw, 和 6个量的variance,是两帧图像的相对运动.

Uncertainty estimation of VO

assume a robot system is described by a no-linear discrete-time process model:

$\mathbf{y}_k = f\left(\mathbf{y}_{k-1}, \mathbf{u}_k, \mathbf{v}_k\right)$

$\mathbf{u}_k = \left[\Delta x_k, \Delta y_k, \Delta z_k, \Delta\phi_k, \Delta\theta_k, \Delta\varphi_k\right]^T$

yk is pose at time k

$\mathbf{v}_k \sim \mathcal{N}\left(0, \mathbf{Q}_k\right)$ is gaussian process noise on motion input.

In order to obtain uncertainty for VO, we need to model feature extraction and camera geometry in a probabilistic perspective.

$$\begin{aligned}\boldsymbol{\theta}^* &= \operatorname{argmax}_{\boldsymbol{\theta}} p\left(\mathbf{u}_k|\mathbf{x}_k;\boldsymbol{\theta}\right)\\ &= \operatorname{argmin}_{\boldsymbol{\theta}} -\log p\left(\mathbf{u}_k|\mathbf{x}_k;\boldsymbol{\theta}\right)\\ &= \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \log|\mathbf{Q}_k|\\ &\quad + \left(F_k\left(\mathbf{x}_k;\boldsymbol{\theta}\right) - \mathbf{u}_k\right)^T \mathbf{Q}_k^{-1}\left(F_k\left(\mathbf{x}_k;\boldsymbol{\theta}\right) - \mathbf{u}_k\right)\end{aligned}$$

uncertainty的label怎么弄？

不需要，求theta的时候，Q类似一个权重，自动学习。

covariance matrix需要是半正定的，在优化的时候需要保留。

Loss:

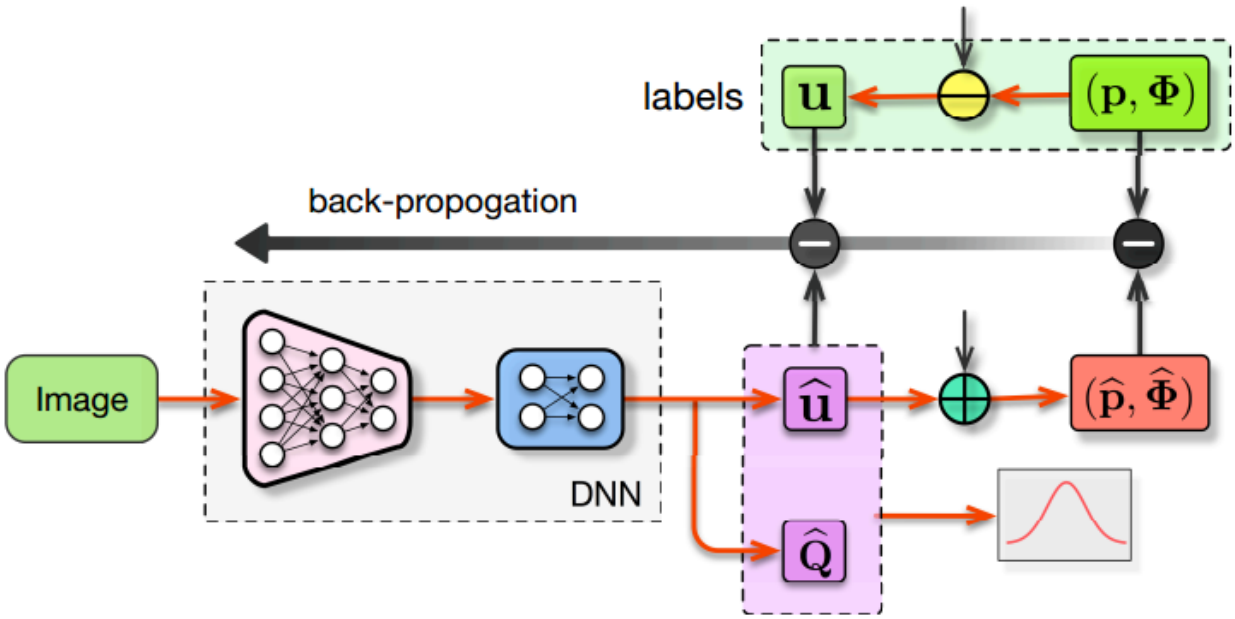


Fig. 6. Network training on pose and uncertainty with joint cost functions. ⊖ (Black) represents errors in cost function, while ⊕ (Green) and ⊖ (Yellow) denote pose compounding and reverse relationship, respectively.

the cost function for training is hybrid and composed of two parts, each of which deals with one type of prediction.

$\mathbf{Y}_t = (\mathbf{y}_1, \ldots, \mathbf{y}_t)$

$\mathbf{X}_t = (\mathbf{x}_1, \ldots, \mathbf{x}_t)$

x代表输入的图像，Y代表预测的pose.

$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, p\left(\mathbf{Y}_t | \mathbf{X}_t; \boldsymbol{\theta}\right)$

最大化神经网络参数

based on MSE, the Euclidean distance between the ground truth pose $\mathbf{y}_k = \left(\mathbf{p}_k^T, \boldsymbol{\Phi}_k^T\right)^T$ and it's estimated :$\widehat{\mathbf{y}}_k = \left(\widehat{\mathbf{p}}_k^T, \widehat{\boldsymbol{\Phi}}_k^T\right)^T$ at time k can be minimised by:

$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \frac{1}{t} \sum_{k=1}^{t} \|\widehat{\mathbf{p}}_k - \mathbf{p}_k\|_2^2 + \kappa \left\|\widehat{\boldsymbol{\Phi}}_k - \boldsymbol{\Phi}_k\right\|_2^2$

$\mathbf{p}$ and $\mathbf{\Phi}$ 代表平移和旋转
k是scale factor
$\mathbf{\Phi}$ 是四元数的形式， 避免了欧拉角在全局坐标系中的问题？

In terms of the motion and uncertainty estimation, the RCNN is trained by maximising the conditional probability of the motion Ut = $(U_1, U_2, U_3, \ldots U_t)$ given a sequence of monocular image $X_t = (x_1, x_2, \ldots, x_t)$

$p\left(\mathbf{U}_t|\mathbf{X}_t\right) = p\left(\mathbf{u}_1, \ldots, \mathbf{u}_t|\mathbf{x}_1, \ldots, \mathbf{x}_t\right)$
$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \frac{1}{t}\sum_{k=1}^{t} \log\left|\widehat{\mathbf{Q}}_k\right| + \left(\widehat{\mathbf{u}}_k - \mathbf{u}_k\right)^T \widehat{\mathbf{Q}}_k^{-1} \left(\widehat{\mathbf{u}}_k - \mathbf{u}_k\right)$

$\widehat{\mathbf{u}}$ and $\widehat{\mathbf{Q}}$ 来自最后一个fc layer. $\hat{u}$ 的label来自相邻帧之间的运动, u的旋转分量使用欧拉角表示而不是四元数，因为旋转在局部坐标系中比较小？， 通过实验证明在这一层使用四元数会是旋转的估计变差。

augument

随机选取一个sequence中不同长度和不同起始点的序列

Adam lr 0.001, p1=0.9, p2=0.999, 200 epochs, dropout, early stopping and incremental training techniques.

CNN的网络是基于预训练的flownet模型。

当测试的时候，一整个测试图像sequence被送入了网络。

overfitting affect VO

理想的是loss最后重合到了一块，如果差太大就是过拟合了。
在作者实验中发现，旋转相比于平移更容易过拟合。
这可能是因为旋转的变化通常很小。

总结：

deepvo只是对相邻两帧图像的相对运动做了监督，而这篇论文中，不仅监督了相对运动，还监督了绝对运动（后序n帧相对于第一帧的位姿）.
相对运动使用欧拉角，绝对运动使用四元数。
全局运动估计使用SE(3) layer完成