第 1 章 问题描述

1.1 多目标检测与追踪问题描述

给定图像序列 $I_1, I_2, ..., I_t$，每帧图像中有 M_t 个目标，其中 t 是当前帧号，每个目标的状态为 $s_1(t)$，其中状态一般包括位置，速度，加速度，朝向等。

$$s_i(t) = \{x, y, z, h, w, l, v_x, v_y, v_z, \theta\} \tag{1-1}$$

当前帧的所有目标的状态就能表示成

$$S(t) = \{s_1(t), s_2(t), s_3(t), ..., s_{M_t}(t)\} \tag{1-2}$$

而每个目标的的轨迹则可以描述成

$$s_i(1:t) = \{s_i(1), s_i(2), s_i(3), ..., s_i(t)\} \tag{1-3}$$

则所有目标的状态集合就能表示成

$$S(1:t) = \{S_1.S_2, ...S_t\} \tag{1-4}$$

同理，我们类似的得到观测结果的定义，记作 $o_i(t), o_i(1:t), O(1:t)$。

而多目标跟踪任务就是通过观测结果找出所有目标的状态，我们用后验估计来进行描述。

$$S(1:t) = argmax_{S(1:t)} P(S(1:t)|O(1:t)) \tag{1-5}$$

1.2 问题的求解

多目标检测与追踪一般的求解都基于 TBD 架构，如图1-1所示。

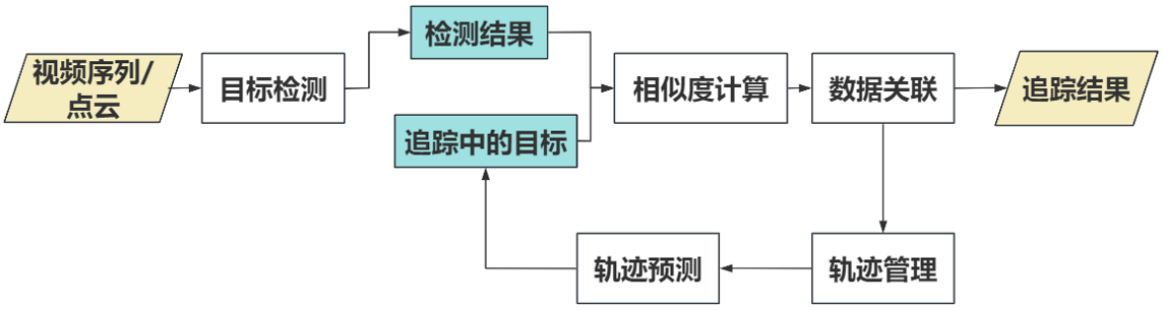

图 1-1 基于检测的追踪框架

1.3 滤波的作用

主要概括，滤波的作用主要包括两个部分，融合多个传感器的追踪结果以及对目标进行预测，最终的目的都是为了得到更好的估计值。它主要应用在图1-1中轨迹预测部分。

首先我们构建线性系统的状态空间描述:

$$\mathbf{x}_t = \mathbf{A}\mathbf{x}_{t-1} + \mathbf{B}\mathbf{u}_t + \boldsymbol{\epsilon}_t$$

$$\mathbf{z}_t = \mathbf{H}\mathbf{x}_t + \boldsymbol{\delta}_t$$

接着我们利用卡尔曼滤波器进行最优状态估计：

$$\hat{\boldsymbol{x}}_t^- = \boldsymbol{A}\hat{\boldsymbol{x}}_t + \boldsymbol{B}\boldsymbol{u}_t \tag{1-6}$$

$$\boldsymbol{\Sigma}_t^- = \boldsymbol{A}\boldsymbol{P}_{t-1}\boldsymbol{A}^T + \boldsymbol{Q} \tag{1-7}$$

$$\boldsymbol{K}_t = \frac{\boldsymbol{\Sigma}_t^- \boldsymbol{H}^T}{\boldsymbol{H}\boldsymbol{P}_t^- \boldsymbol{H}^T + \boldsymbol{R}} \tag{1-8}$$

$$\hat{\boldsymbol{x}}_t = \hat{\boldsymbol{x}}_{t-1} + \boldsymbol{K}_t(\boldsymbol{z}_t - \boldsymbol{H}\hat{\boldsymbol{x}}^-) \tag{1-9}$$

$$\boldsymbol{P}_t = (\boldsymbol{I} - \boldsymbol{K}_t\boldsymbol{H})\boldsymbol{P}_t^- \tag{1-10}$$

第 2 章 DeepVO: Towards End-to-End Visual Odometry with Deep Recurrent Convolutional Neural Networks"[1]

2.1 问题描述

DeepVO 首次提出了一种端到端的视觉里程计算法，整个流程只需要给视频就能够得到状态。其将状态估计问题描述成一个最大似然估计问题。

对于给定的单目相机图片序列，计算姿态的条件概率。

$$P(Y_t|X_t) = (y_1, y_2, ...y_t|x_1, x_2, ...x_t)$$

利用神经网络进行建模之后，问题就成为求解网络最优超参数 θ^* 的问题：

$$\theta^* = \arg\max_{\theta} p(Y_t|X_t;\theta)$$

2.2 解决方法

第一次提出了用端到端的方法求解状态估计问题，其具体网络结构如图2-1所示。

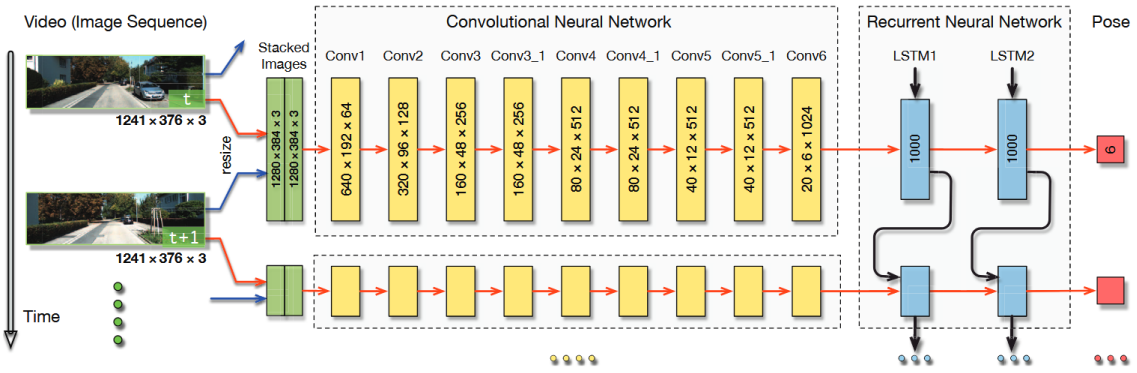

图 2-1 DeepVO 整体网络结构

其中的 RNN 结构部分，考虑到了物体的运动方程，在潜状态空间进行了建模，其节点结果如图2-2所示。

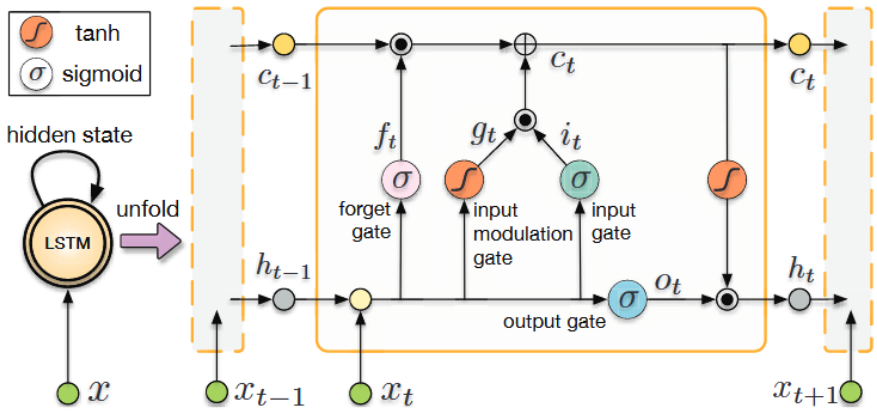

图 2-2 RNN 节点结构

其构建的模型：

$$\mathbf{h}_k = \mathcal{H}(\mathbf{W}_{xh}\mathbf{x}_k + \mathbf{W}_{hh}\mathbf{h}_{k-1} + \mathbf{b}_h)$$
$$\mathbf{y}_k = \mathbf{W}_{hy}\mathbf{h}_k + \mathbf{b}_y$$

其具体内部的参数计算则是:

$$\mathbf{i}_k = \sigma(\mathbf{W}_{xi}\mathbf{x}_k + \mathbf{W}_{hi}\mathbf{h}_{k-1} + \mathbf{b}_i)$$
$$\mathbf{f}_k = \sigma(\mathbf{W}_{xf}\mathbf{x}_k + \mathbf{W}_{hf}\mathbf{h}_{k-1} + \mathbf{b}_f)$$
$$\mathbf{g}_k = \tanh(\mathbf{W}_{xg}\mathbf{x}_k + \mathbf{W}_{hg}\mathbf{h}_{k-1} + \mathbf{b}_g)$$
$$\mathbf{c}_k = \mathbf{f}_k \odot \mathbf{c}_{k-1} + \mathbf{i}_k \odot \mathbf{g}_k$$
$$\mathbf{o}_k = \sigma(\mathbf{W}_{xo}\mathbf{x}_k + \mathbf{W}_{ho}\mathbf{h}_{k-1} + \mathbf{b}_o)$$
$$\mathbf{h}_k = \mathbf{o}_k \odot \tanh(\mathbf{c}_k)$$

其中主要的参数说明：h_k 是潜状态，x_k 是输入的图像，y_k 是输出的姿态，c_k 是记忆存储单元（记录了之前的运算结果），W_k 是网络权重。

2.3 仿真结果

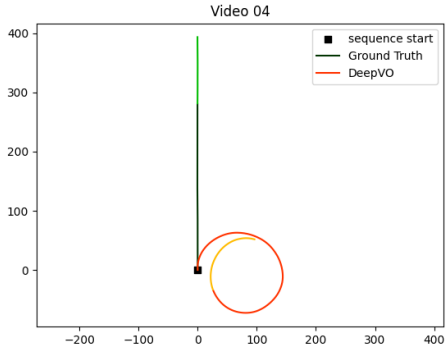

图 2-3 序列 4 测试结果

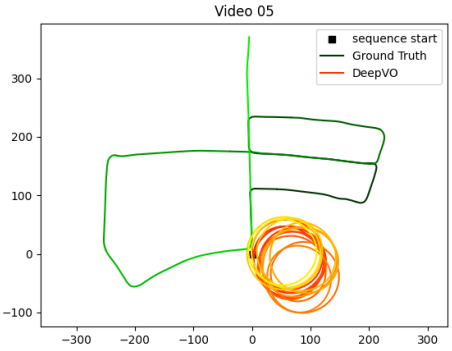

图 2-4 序列 5 测试结果

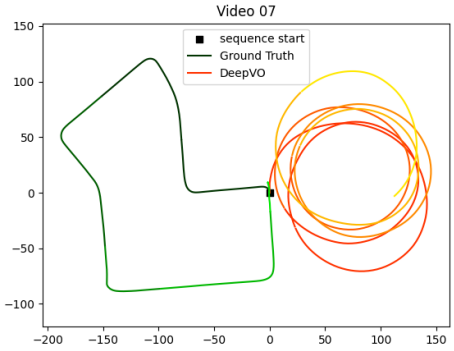

图 2-5 序列 7 测试结果

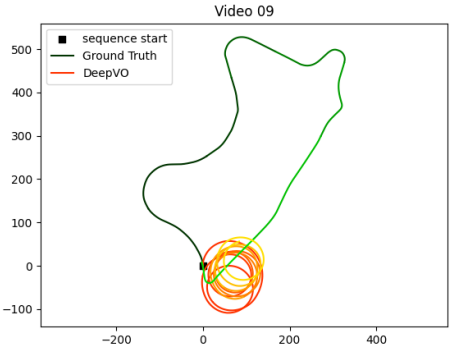

图 2-6 序列 9 测试结果

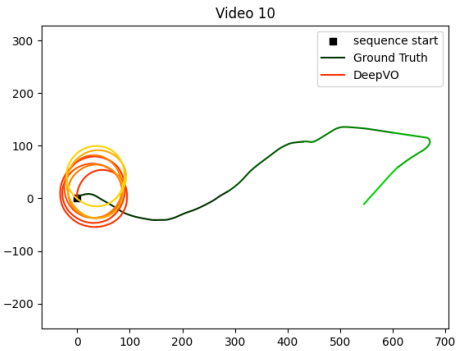

图 2-7 序列 10 测试结果

2.4 学习体会

DeepVO 首次提出了一种端到端的视觉里程计算法，整个流程只需要给视频就能够得到状态。最终的效果还算不错（并没有完全取代传统方法）。试着运行了端到端算法的效果，作为以后对比的内容。

可以发现纯神经网络算法的效果并没有达到最优，而且存在数据集不足，应对场景泛化能力不足，原理黑盒等问题。但是他们也展现出一些优势：不需要进行复杂的建模，整体的结构简单容易实现。

第 3 章 论文“SWformer-VO: A Monocular Visual Odometry Model Based on Swin Transformer”[2]学习

3.1 问题描述

本文是对之前基于学习的方法的改进，问题描述同 DeepVO。基于学习的方法距离效果最好的视觉里程计还有一定的差距。此外，一些新的学习方法的计算复杂度比较高。

3.2 解决方法

1. 将 swim-transformer 结构应用到视觉里程计的任务上，据此计算复杂度由二次方降低为线性，大大提高了计算效率。提出的网络结构如图3-1所示。

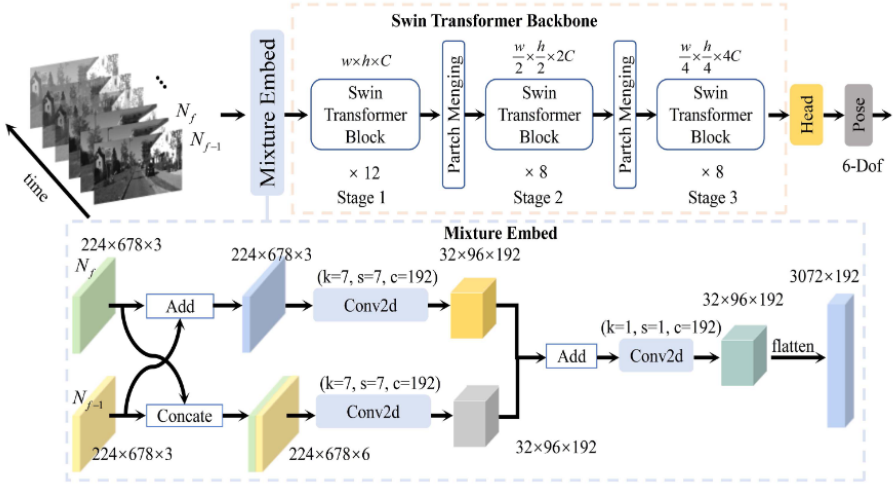

图 3-1 swformer 网络结构

新网络的计算复杂度和过去基于 Transformer 网络的方法相比显著下降。

$$\begin{aligned}\Omega(T\text{Sforn er } -VO) &= 4\times 2NC^2 + 2(2\,\mathrm{N})^2\,C,\\ &= 8NC^2 + 8\,N^2C \quad ,\\ \Omega(SW\text{former } -VO) &= 4NC^2 + 2\,\mathrm{M}^2\mathrm{N}C.\end{aligned}$$

图 3-2 计算复杂度的降低

2. 提出了一种“Mixture Embed Module”模块对图像进行预处理，由此构成网络。

文受启发于“视觉残留”现象，提出了一种帧间插值的方法，如图3-1所示。这种方法顺利的减小了计算量，由原来需要处理连续的两帧图片，变成了处理一个经过预处理的合成图片。

3.3 代码复现

1. 训练

训练了 50 轮次

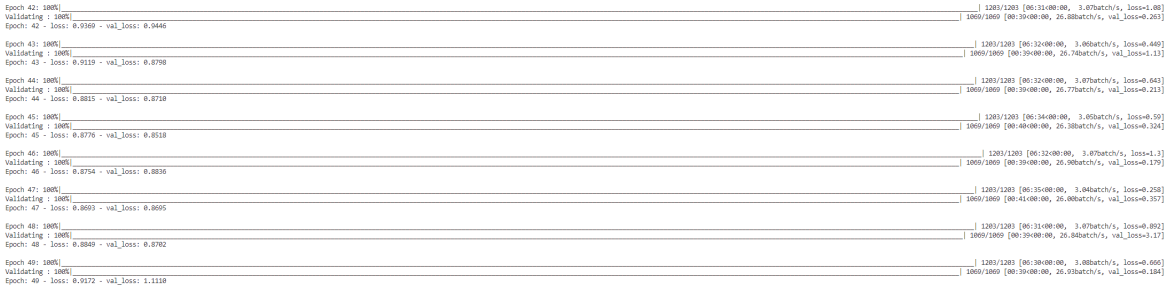

图 3-3 网络训练结果

2. 测试如图3-4、3-5、3-6所示。

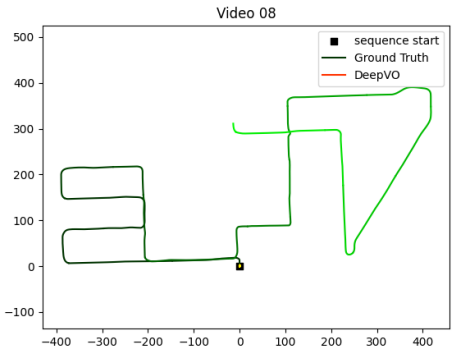

图 3-4 序列 8 测试结果

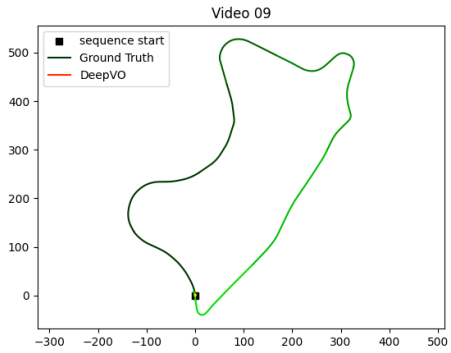

图 3-5 序列 9 测试结果

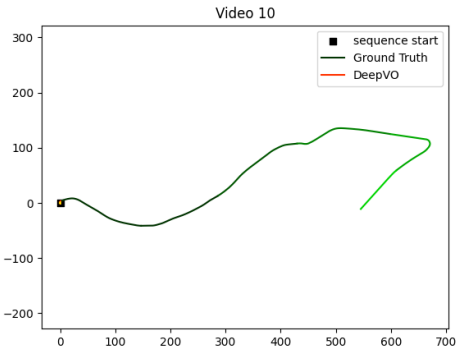

图 3-6 序列 10 测试结果

3.4 总结与思考

虽然该论文也是端到端的学习算法，不过依旧可以从中学到一些内容。

1. 数据集的补充

根据阅读的两篇论文，我发现它们都提到了数据集的不足。学习类的算法是非常依赖数据集的数量和质量的，本文再次映证了这一点。

2. swim-transformer 结构的学习

transformer 的实际应用需要进行一定的本地化，因此学界提出了 swim-transformer 的结构。后面需要仔细了解这以结构的原理和实现。

3. 图像预处理技术

由于需要处理运动学的问题，那么我们就需要在一个时刻内处理多帧图像。本文就提供了一种处理方法。

参考文献

[1] Wang S, Clark R, Wen H, et al. DeepVO: Towards end-to-end visual odometry with deep Recurrent Convolutional Neural Networks[J].

[2] Wu Z, Zhu Y. SWformer-VO: A Monocular Visual Odometry Model Based on Swin Transformer[J]. IEEE Robotics and Automation Letters, 2024, 9(5): 4766-4773.

[3] Sui Q, Wang Y, Chen Z, et al. Deep Koopman Kalman Filter for Nonlinear Model-free Industrial Process Data Denoising and Its Soft Sensing Application[J]. IEEE Sensors Journal, 2024.

[4] Cohen N, Klein I. A-KIT: Adaptive Kalman-informed transformer[J]. arXiv preprint arXiv:2401.09987, 2024.

[5] Chen C, Lu C X, Wang B, et al. DynaNet: Neural Kalman dynamical model for motion estimation and prediction[J]. IEEE Transactions on Neural Networks and Learning Systems, 2021, 32(12): 5479-5491.

[6] Vaswani A. Attention is all you need[J]. Advances in Neural Information Processing Systems, 2017.