

projective manifold gradient layer for deep rotation regression

摘要

利用深度神经网络回归SO(3)流形上的旋转是一个还未被解决的重要问题。欧式网络空间与非欧空间SO(3)流形之间的差距对神经网络的前向和后向学习有一定的不同。虽然有几项工作提出了不同的回归旋转表示，但是很少有工作致力于改进后向过程中的梯度反向传播。

本文提出了一种流形感知梯度，该梯度可以直接反向传播到网络权重中，使用黎曼优化构造一种新的投影梯度，本文提出了一种正则化投影流形梯度（RPMG）方法有助于网络在各种旋转估计任务重获得最新的性能。提出的梯度层可以应用于其他的光滑流形，例如单位球体。

一些名词

- SO(3) manifold流形
- 黎曼优化
- 正则化投影流形梯度（RPMG）

这篇论文大概要干啥

三维平移和三维旋转一同构成了六维位姿。然而与三维平移不同的是，三维旋转属于非欧几里得变量，其非欧的性质对神经网络直接进行回归带来了重大的挑战。具体来说，全部三维旋转构成了一个连续群SO(3)，这一连续群是一种三维黎曼流形（Riemannian manifold），而并非传统的欧几里得空间。传统的神经网络是部署在欧氏空间上的，所以往往对旋转的预测不太好，我们希望在黎曼流形中构建一个神经网络，用于对旋转预测进行改进。

1. 引言

三维旋转（朝向）估计是为了预测三维空间中三自由度的旋转。这是一个计算机视觉领域的典型问题，如物体/人体/手的位姿估计、相机重定位等任务。人们开始尝试用神经网络来解决三维旋转的预测任务，其中，直接回归(regression)三位旋转具有速度快、端到端的优势，吸引了很多关注

现在的工作主要围绕三位旋转的表示，常用的包括三维的欧拉角表示、三维轴角表示、四元数表示和九维的旋转矩阵表示。比较主流的是四元数，在这个维度上神经网络的预测比较准。

在本项工作中，本文第一次关注到了这样常常是多对一的流形映射在梯度回传中构成的特有问題，并针对这样一类从欧式空间到黎曼流形的映射设计了一个新的梯度反传层——**针对流形的正则投影梯度层**（regularized projective manifold gradient layer），来取代简单的基于链式法则的梯度回传，以促进网络的优化。我们的方法不仅能广泛适用于之前工作提出的各种非欧表示（四元数/六维/九维/十维），还能推广到其他的黎曼流形。


通过大量的实验，本文展示了本文的方法能够在不带来额外时空开销的情况下，在旋转矩阵的各种非欧表示（四元数/六维/九维/十维）及多项任务（如基于点云的三维物体位姿估计、基于图像的三维物体位姿估计、相机重定位等）中取得一致且显著的提升。

- Main Contribution
 - 提出了一种新的流形感知梯度层----RPMG，用于旋转回归的后向传递，/端到端/应用于不同的旋转表示和损失
 - RPMG很SOTA
 - 可以被推广到其他流形

2. 相关工作

一些姿态估计的旋转表示方法

- 方向余弦矩阵 Direction Cosine Matrix (DCM)
- 轴角 Axis-Angle
- 四元数 Quaternion
- 欧拉角 Euler Angle

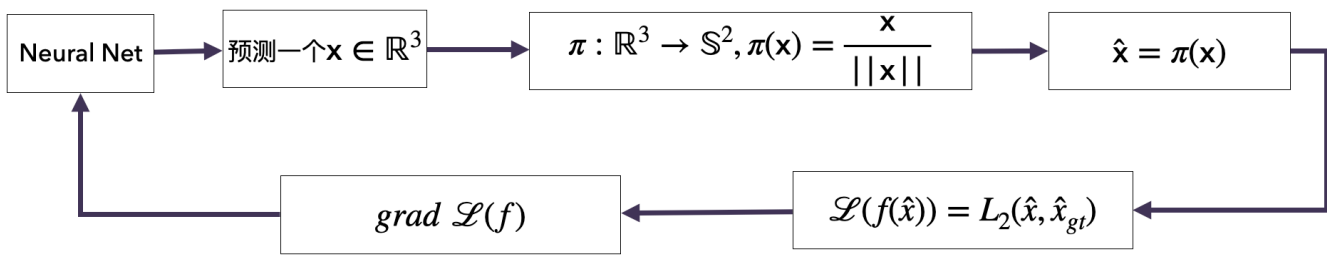
早期方法

现在的一些更好的表示方式

- 与SO(3)同构--homeomorphic, 是更好的表示方式, 能够提升回归的准确性
 - SVD orthogonalization
 - Riemannian optimization
 - SO(3), SE(3), Sim(3)

3. 准备工作

3.0 为什么要用黎曼几何?



传统预测三维空间中的单位球面 S^2 （也是一个黎曼流形）上的一个元素 \hat{x} 的流程如上图所示，如果我们要用神经网络去回归 上的一个元素（满足 $\|x\| = 1$ ），那么最常用的做法是首先用神经网络去预测一个 R^3 中的元素，然后再通过一个流形映射来得到 \hat{x} 。这就是神经网络正向传播的过程。在训练的时候，人们会对 \hat{x} 施加一个损失函数（常见的选择是 L2 loss），然后用基于链式法则来计算梯度并进一步反向传播给网络，用于更新网络的参数。

问题是，这样的优化方法没有球面 S^2 的约束，可能通过梯度优化求得的元素偏离球面本身。

3.1 黎曼几何

$$SO(3) = \{ \mathbf{R} \in \mathbb{R}^{3 \times 3} : \mathbf{R}^T \mathbf{R} = \mathbf{I}, \det(\mathbf{R}) = 1 \}$$
$$\exp_{\mathbf{I}}(\phi^\wedge) = \mathbf{I} + \phi^\wedge + \frac{1}{2!}(\phi^\wedge)^2 + \dots$$
$$\exp_{\mathbf{R}}(\phi^\wedge) = \mathbf{R} \left(\sum_{n=0}^{\infty} \left(\frac{1}{n!} (\phi^\wedge)^n \right) \right)$$

通过上面的李群-李代数的变换可以得到在流形面中的旋转矩阵表达，通过Rodrigues Formula展开，得到

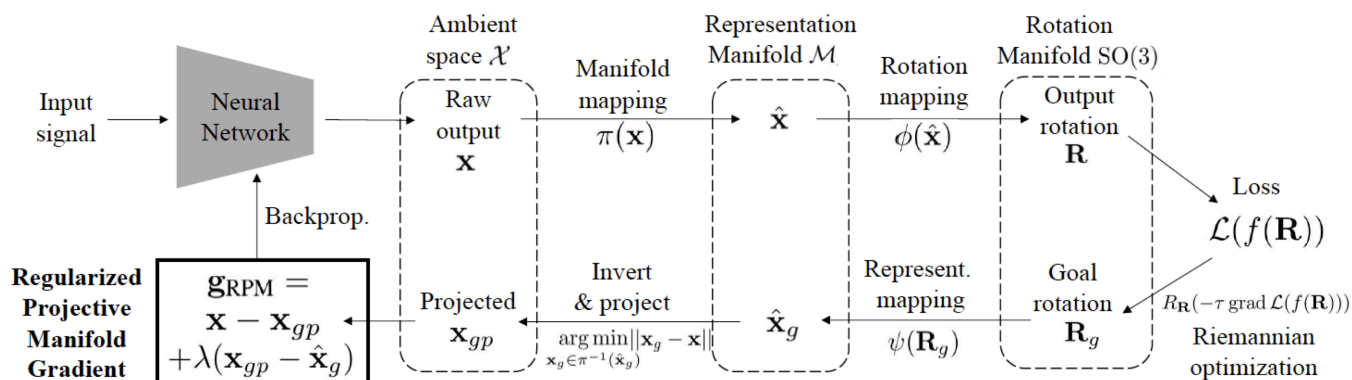
$$Exp_{\mathbf{R}}(\phi) = \mathbf{R}(\mathbf{I} + \sin\theta\omega^\wedge + (1 - \cos\theta)(\omega^\wedge)^2)$$

这样，我们就可以在李代数空间（流形）上求出旋转矩阵的导数，进而求得梯度

$$\frac{\partial}{\partial \phi_x} Exp_{\mathbf{R}}(\phi)|_{\phi=0} = \mathbf{R} \left(\cos\theta \frac{\partial}{\partial \phi_x} \omega^\wedge \right) \Big|_{\phi=0}$$

$$grad \mathcal{L}(f(\mathbf{R})) = \left(\frac{\partial f}{\partial \mathbf{R}} \frac{\partial}{\partial \phi} Exp_{\mathbf{R}}(\phi)|_{\phi=0} \right)^\wedge$$

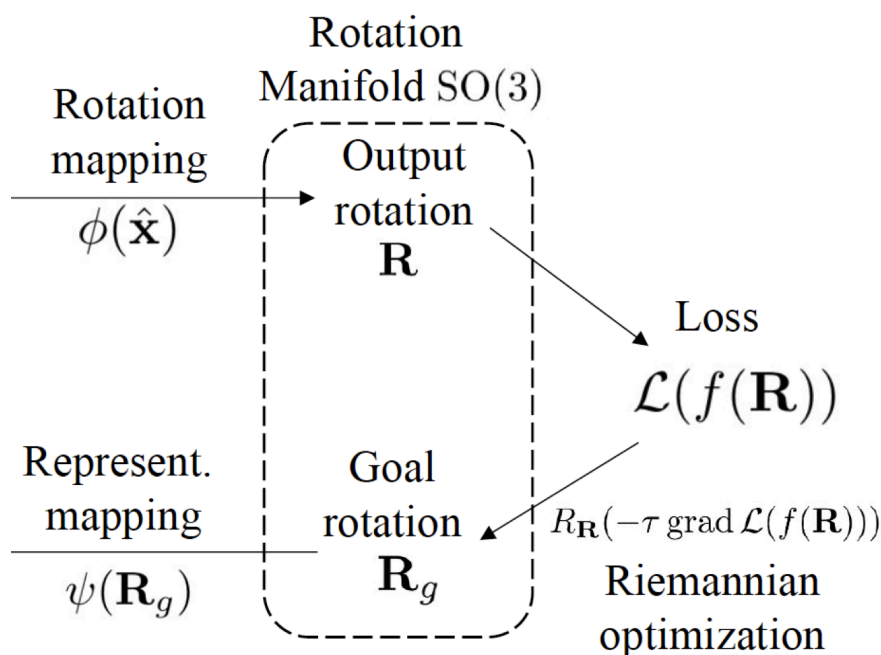
3.2 深度旋转回归



首先采用流形上的优化技术——黎曼优化（Riemannian optimization）来得到一个在黎曼流形 S^2 上的更新 \hat{x}_g ，之后进一步利用它来计算 x 的梯度：希望达到的目的是让 x 沿梯度方向做一次更新后能满足 $\pi(x - grad_x) = \hat{x}_g$ 。最直接的解决办法是令 $grad_x = x - \pi^{-1}(\hat{x}_g)$ ，但是由于映射 π 是多对一的映射， π^{-1} 得到的往往是一个集合而非单一元素（对二维球面来说，这个元素集合就是一条从原点指出通过 \hat{x}_g 的射线），因此我们需要从这个集合中选一个最优的 x_{gp} 来计算梯度。

4.1 黎曼梯度&&黎曼优化

- Step1: 【损失函数】 $\mathcal{L}(f(\mathbf{R}))$ ，计算 \mathbf{R} 与 \mathbf{R}_{gt} 的 loss
- Step2: 【黎曼优化】 $\mathbf{R}_g \leftarrow R_{\mathbf{R}}(-\tau grad \mathcal{L}(f(\mathbf{R})))$
- Step3: 【Mapping】 通过 \mathbf{R}_g 计算出 \hat{x}_g

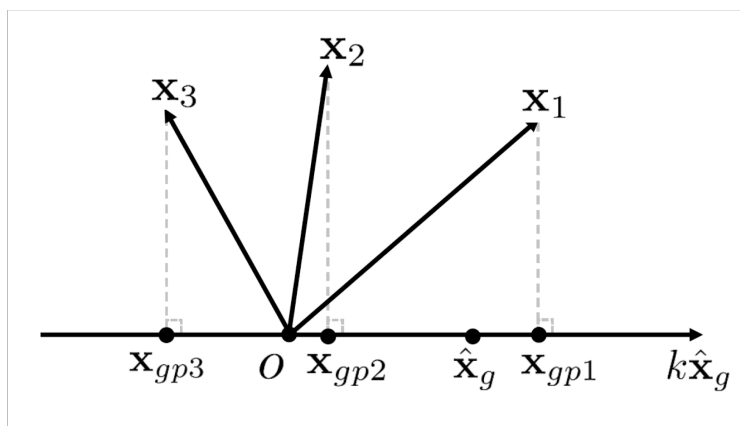
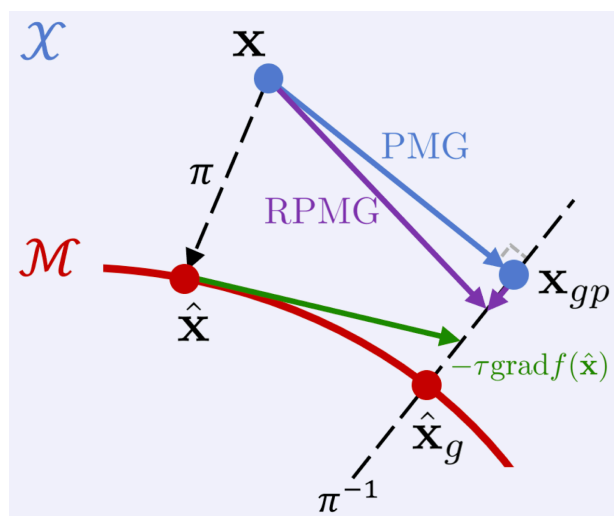


给定 \mathbf{R}_g ，我们可以使用表示映射 ψ 在流形上找到对应的 $\hat{\mathbf{x}}_g = \psi(\mathbf{R}_g)$ 。然而，由于 π 的投影性质，可能会得到很多不同的 \mathbf{x}_g ，似乎我们可以使用满足 $\pi(\mathbf{x}_g) = \hat{\mathbf{x}}_g$ 的任意 \mathbf{x}_g 构造一个梯度： $(\mathbf{x} - \mathbf{x}_g)$ 。无论我们选择哪一个，如果这个梯度被更新，它将导致相同的旋转矩阵。但是，当反向传播到网络时，这些梯度会以不同的方式更新网络权值，可能会导致不同的学习效率和网络性能。

本文提出的方法是建议在所有可能的梯度中选择最小冗余的梯度，也就是选择距离神经网络输出的 \mathbf{x} 最近的 \mathbf{x}_g 作为最终结果。

由于四元数 q 和 $-q$ 都代表了同一个旋转矩阵，相当于存在两个正确答案，但是网络的输出并不是一个分布而是一个特定向量，它只能往 q 或者 $-q$ 其中一个靠近，因此在计算损失函数时必须从 q 和 $-q$ 这两个答案中选取其一。考虑到希望网络能以尽可能小的变动去实现所需要的更新，他们提出的解决办法是选取离当前网络输出值最近的那个答案。

4.2 射影流形梯度



- 在forward中：

我们投影到 $\hat{\mathbf{x}} = \pi(\mathbf{x})$

- 在backward中，首先计算一个黎曼梯度（绿色箭头）得到下一个目标 $\hat{\mathbf{x}}_g$ 然后映射回去。以上述方式计算出来的梯度 PMG 会持续地给 \mathbf{x} 的模长 $\|\mathbf{x}\|$ 一个变小的趋势（如图右中），在不同流形、不同表示上均有此现象。这会使得网络的输出变得特别的小，相当于等效地增加了相对学习率（learning rate），造成了训练的不稳定和收敛的困难。

4.3 正则化射影流形梯度

为了解决这一问题，我们需要引入一项额外的正则项来稳定输出的模长，最终得到RPMG（regularized projective manifold gradient，图左紫色箭头）。

$$\|\mathbf{x}\| < \|\mathbf{x}_{gp}\|$$

$$\mathbf{g}_{RPM} = \mathbf{x} - \mathbf{x}_{gp} + \lambda(\mathbf{x}_{gp} - \hat{\mathbf{x}}_g)$$

（ λ 看作超参数，取值的原则就是比较小（本文取0.01））

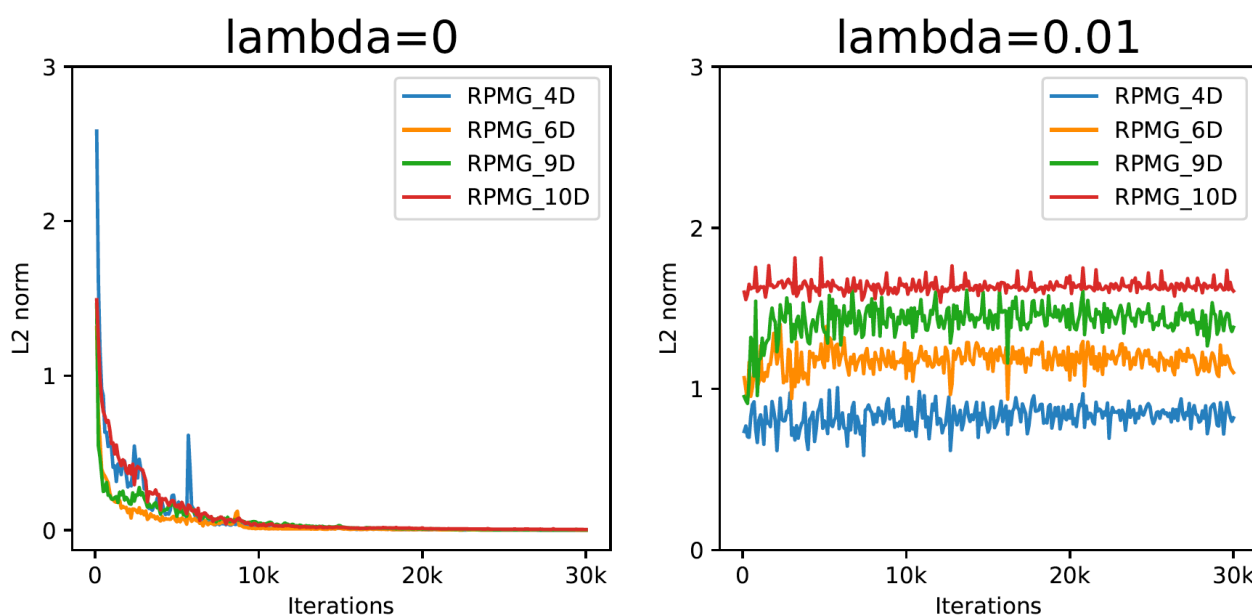


Figure 3. **Average L2 norm of the network raw output \mathbf{x}** during training. Left: PMG-4D/6D/9D/10D (w/o reg. $\lambda = 0$). Right: RPMG-4D/6D/9D/10D (w/ reg. $\lambda = 0.01$)

5. 实验

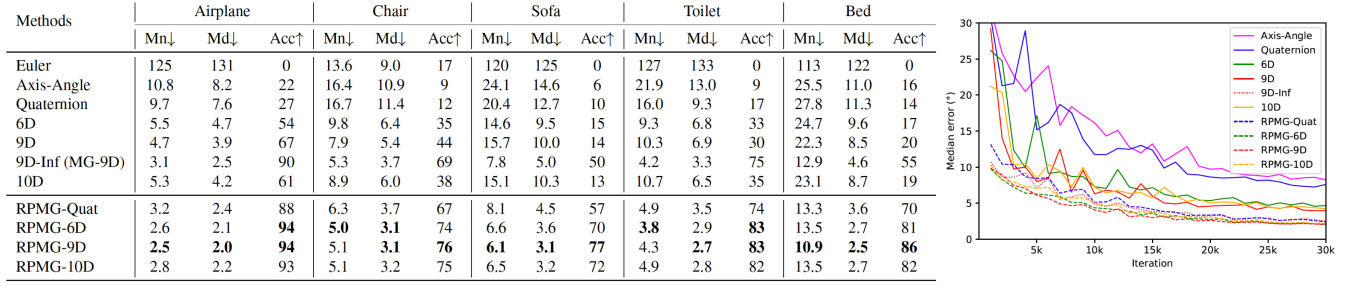


Table 1. **Pose estimation from ModelNet40 point clouds.** Left: a comparison of methods by mean, median, and 5° accuracy of (geodesic) errors after 30k training steps. Mn, Md and Acc are abbreviations of mean, median and 5° accuracy. Right: median test error of *airplane* in different iterations during training.

我们的主实验是基于三维点云的类别级物体位姿估计。我们用一个物体的三维点云作为输入，希望网络去预测输入的点云 相对于该类物体预定义的物体坐标系的旋转。我们采用和 ground truth 旋转矩阵的 L2 loss 来作为损失函数。实验结果如图3、图4所示，可以看出我们的梯度层能在不同类别的物体、不同的旋转矩阵表示下（table 1），在各个误差区间都能获得显著的提升（图4右）。另外我们的方法能收敛得更快更好（图4左）。

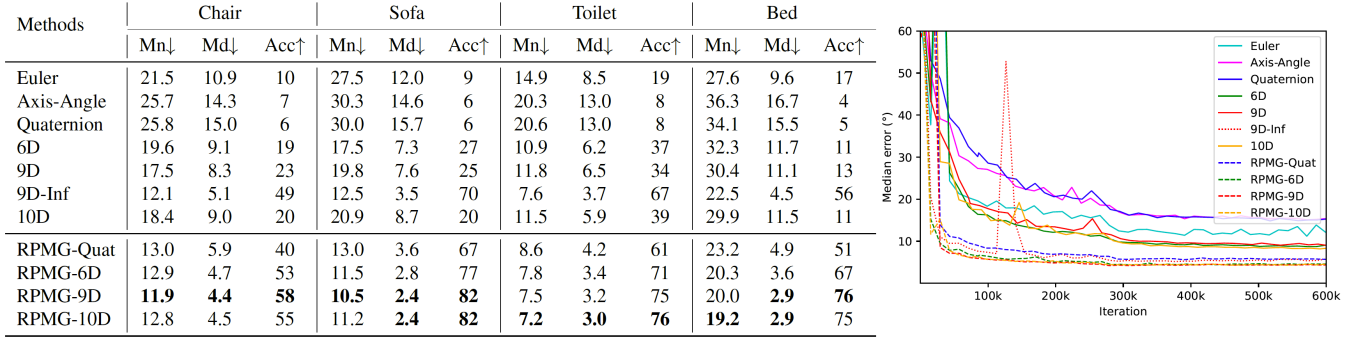


Table 3. **Pose estimation from ModelNet10 images.** Left: a comparison of methods by mean($^\circ$), median($^\circ$), and 5° accuracy(%) of (geodesic) errors after 600k training steps. Mn, Md and Acc are abbreviations of mean, median and 5° accuracy. Right: median test error of *chair* in different iterations during training.

我们的方法在基于图片的类别级物体位姿估计的实验中（图5），也能有显著、一致的提升。实验的设置和第一个实验完全一致，唯一的区别是给网络的输入变成了一张包含特定类别物体的二维图片。

Methods	Mean ($^{\circ}$) \downarrow	Med ($^{\circ}$) \downarrow	3 $^{\circ}$ Acc (%) \uparrow
Euler	131.9	139.1	0.0
Axis-Angle	4.5	3.8	34.5
Quaternion	4.3	3.5	37.5
6D	55.1	6.7	20.0
9D	1.8	1.6	88.0
9D-Inf	118.2	119.5	0.0
10D	1.6	1.5	91.0
RPMG-Quat	3.5	2.4	70.0
RPMG-6D	15.0	2.9	55.0
RPMG-9D	1.3	1.2	97.5
RPMG-10D	1.5	1.4	97.0

Table 4. Self-supervised Instance-Level Rotation Estimation from Point Clouds. We report mean, median and 3 $^{\circ}$ accuracy of (geodesic) errors after 30K iterations.

本文的方法在自监督问题中，能够很好的将 $\hat{\mathbf{x}}_{gp}$ 约束在流形上，通过chamfer distance的方法求出伪ground truth，但是如果在欧氏空间做这样的问题，效果会很差。

局限性

- 引入了两个超参数 τ 和 λ
- 只能应用于具有一定约束条件的流形