NICP: Dense normal based point cloud registration

Jacopo Serafin, Giorgio Grisetti

2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)

智能网络与优化实验室

http://jacoposerafin.com/nicp/

1 问题提出

问题提出

- 不同于经典ICP（point to point ICP），NICP 在匹配两组点云时，将点云的局部特征（法向量，曲率）考虑在内，即在迭代求解过程中，误差函数不仅包含点云之间的法向量的投影距离（point-to-plane ICP），还包含了法向量方向误差。

- NICP 算法的特点在于，其在匹配两组点云时并非考虑匹配点云之间的欧氏距离，而是将点云曲面的局部特征作为点对匹配以及计算变换的准则。具体来说主要可以分为以下几部分：

（1）计算点云中每个点的特征，即其领域的法向量（normal）和曲面曲率(curvature)，以标记每个点；

（2）根据点的距离和特征找两组点云中的匹配点对；

（3）利用最小二乘法最小化，最小化目标函数，以求解点云变换矩阵。此处目标函数包括点面投影和法向量旋转误差。

参考：
- https://zhuanlan.zhihu.com/p/110428934
- https://blog.csdn.net/weixin_41469272/article/details/105522443

源码：
- https://github.com/yorsh87/nicp

2 论文主体

特征提取

为了求解法向量，我们提取目标点周围半径为 R （超参数）的球体范围内的所有点的高斯分布的协方差矩阵。如果目标点领域内的点云表面是良定义的话，它们就可以近似为一个平面，并且协方差的特征值中仅有一个特征值远小于另外两个。那么这个最小的特征值对应的特征向量则被认为是这个点领域点云平面的法向量。上述文字描述通过数学公式表达如下：

$$\mu_i^s = \frac{1}{|\nu_i|}\sum_{p_j \in \nu_i} p_j\,, \qquad \Sigma_i^s = \frac{1}{|\nu_i|}\sum_{p_j \in \nu_i} (p_j - \mu_i^s)(p_j - \mu_i^s)^T$$

式中，ν_i 是指以点 p_i 为球心半径为 R 的球体中的点云簇。

特征提取

因为要找领域内的点云，如果用 KD-Tree 搜索，时间开销会很大。为了提高计算速度，NICP 基于 integral images 的方法，计算高斯协方差。一旦计算出协方差矩阵，就可以用一下的方式对协方差矩阵进行分解（SVD分解）

$$\Sigma_i^S = R \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} R^T$$

其中，$\lambda_{1:3}$ 是特征向量并且升序排列，R 则是近似点云分布的椭球的特征向量 n_1, n_2, n_3 所构成的矩阵。$\lambda_1 < \lambda_2 < \lambda_3$ 说明在 n_1 方向上的点较集中，而在另外两个方向较发散。使用

$$\sigma_i = \frac{\lambda_1}{\lambda_1 + \lambda_2 + \lambda_3}$$

表示点云表面的的平坦程度（曲率），σ 越小，说明表面越平坦（PCA原理）。

特征提取

由于现实中噪声的影响，即使是真正的表面也无法使 $\lambda_1 = 0$，那么则通过调整椭球体的一个轴，使得另外两个较大的轴相等，即

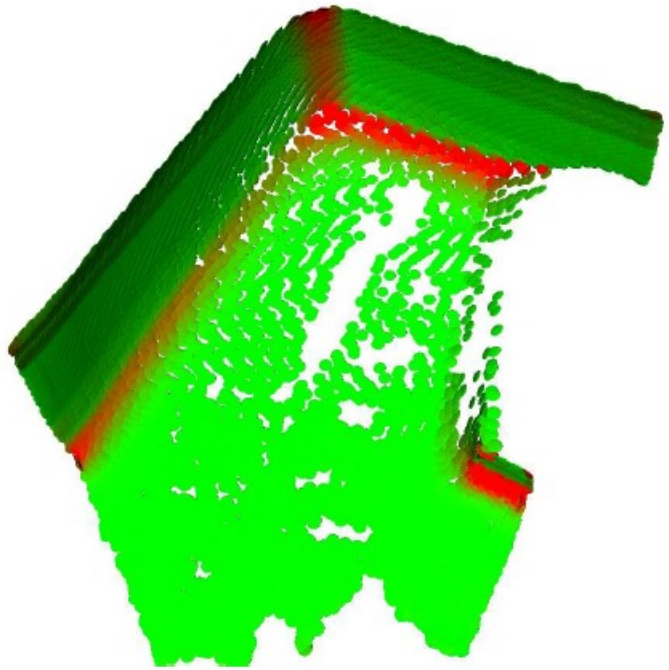
$$\Sigma_i^S = R \begin{bmatrix} \epsilon & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} R^T$$

回到GICP，实验中 $\epsilon = 0.001$

最终每一个点 p_i 都会被分到一个标签 $< \mu_i^s, \Sigma_i^s, \sigma_i >$

图中是一个 σ 的分布图，红色表示值较大点云处于曲面，

绿色则表示值较小处于点云较平滑的面。



寻找匹配点对

采用投影的方式，将两组点云投影到同一个深度图当中，那些落在同一个像素并且有一致的法向量和曲率的点就被认为是匹配点对。如果有点多个点落在同一个像素，就选择距离最近的且法向量指向投影中心的点对。这些备选匹配点对还会经过如下条件的筛选，如果满足以下条件中的至少一个就去掉该组点对：

- 点对中的两个点至少有一个没有良定义的法向量，即选择比较结构化的点，如果对应点周围过于杂乱就丢弃
- 两个点的距离超过阈值，即 $\|p_i^c - \boldsymbol{T} \cdot p_i^r\| > \epsilon_d$，$p_i^c$ 和 p_i^r 分别指目标点云和源点云中的点
- 两个点曲率对数之差超过阈值，即 $\left\|\log \sigma_i - \log \sigma_j\right\| > \epsilon_\sigma$
- 两点的法向量之间的角度超过阈值，即 $n_i^c \cdot \boldsymbol{T} n_i^r = |n_i^c| \cdot |\boldsymbol{T} n_i^r| \cdot \cos\theta < \epsilon_n$

相比ICP的距离最小匹配，增加了对点所在曲面的曲率及法向量的约束，筛除更多的错误匹配点。

根据匹配点计算两组点云之间的位姿变换

在得到了两组点云之间的匹配点对之后，通过迭代最小二乘法求取它们之间的变换。通过前面介绍的步骤，给定一个点，会得到其坐标，表面曲率，法向量以及协方差。记 $\tilde{p}$ 是带法向量的点坐标，即

$$\hat{p} = (p^T \ n^T)^T = \begin{pmatrix} p \\ n \end{pmatrix}$$

定义 $\boldsymbol{T}$ 为位姿变换矩阵，定义运算 $\oplus$ 如下

$$\hat{p}' = \boldsymbol{T} \oplus \hat{p} = \begin{pmatrix} Rp + t \\ Rn \end{pmatrix}$$

式中，R 表示旋转矩阵， t 表示平移向量。

根据匹配点计算两组点云之间的位姿变换

当给定一个点对和当前的位姿变换矩阵后，可以重新定义误差函数为

$$e_{ij}(\boldsymbol{T}) = (\hat{p}_i - \boldsymbol{T} \oplus \hat{p}_j)$$

从而所有点对所构成的目标函数变为

$$\sum_{C} e_{ij}(\boldsymbol{T})^T \hat{\Omega}_{ij} e_{ij}(\boldsymbol{T})$$

其中，$\hat{\Omega}_{ij}$ 是一个 6×6 的信息矩阵。理想情况下，我们想要一个信息矩阵来旋转匹配的两组点云，使得点对的法向量对齐，而且惩罚法向量方向上的投影距离忽略切向的投影误差。

根据匹配点计算两组点云之间的位姿变换

$\hat{\Omega}_{ij}$ 构造如下：

$$\hat{\Omega}_{ij} = \begin{pmatrix} \Omega_i^s & O \\ O & \Omega_i^n \end{pmatrix}$$

其中，Ω_i^s 是点 p_i^c 周围的信息矩阵

$$\Omega_i^s = (\Sigma_i^s)^{-1}$$

而 Ω_i^n 是点 p_i^c法向量的信息矩阵，若平面足够平滑（表面曲率足够小），则可以取下矩阵，否则给其预定义

$$\Omega_i^n = R \begin{pmatrix} \frac{1}{\epsilon} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} R^T$$

实验中，Ω_i^n 取值可以是 0（GICP），0.33，0.66，1（NICP）

根据匹配点计算两组点云之间的位姿变换

NICP算法使用增量扰动的局部参数化来最小化目标函数，增量

$$\Delta \boldsymbol{T} = (\Delta t_x, \Delta t_y, \Delta t_z, \Delta q_x, \Delta q_y, \Delta q_z)$$

包含平移向量和旋转四元数的虚部，迭代中使用阻尼高斯牛顿法

$$(H + \lambda I)\Delta \boldsymbol{T} = b$$

$$H = \sum J_{ij}^T \hat{\Omega}_{ij} J_{ij}, \qquad b = \sum J_{ij}^T \hat{\Omega}_{ij} e_{ij}(\boldsymbol{T})$$

式中 H 为近似的海森堡矩阵，通过上式更新位姿变换矩阵

$$\boldsymbol{T} \leftarrow \Delta \boldsymbol{T} \oplus \boldsymbol{T}$$

根据匹配点计算两组点云之间的位姿变换

雅可比矩阵 J_{ij} 的求法如下

$$J_{ij} = \frac{\partial e_{ij}(\Delta \boldsymbol{T} \oplus \boldsymbol{T})}{\partial \Delta \boldsymbol{T}} = \begin{pmatrix} -I & 2[\boldsymbol{T} \oplus p_j^r]^\wedge \\ 0 & 2[\boldsymbol{T} \oplus n_j^r]^\wedge \end{pmatrix}$$

其中

$$[a]^\wedge = \begin{pmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{pmatrix}$$

3 结果

3 结果

实验结果

1. 在深度相机的数据集[1]中：

- 环境变化的复杂度：low，medium，high
- 运动变化状态：rotation，translation，fly
- 变化速度：slow，medium，fast

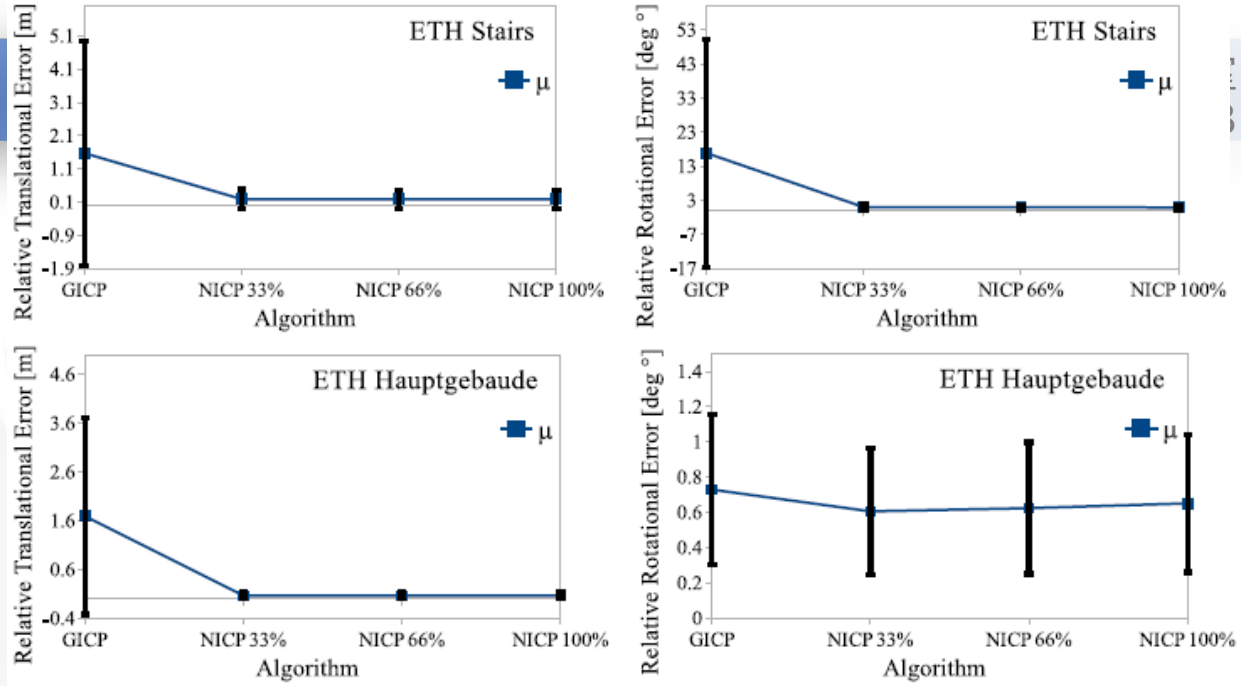
[1] “Tracking a depth camera: Parameter exploration for fast icp,” in Intelligent Robots and Systems (IROS), 2011

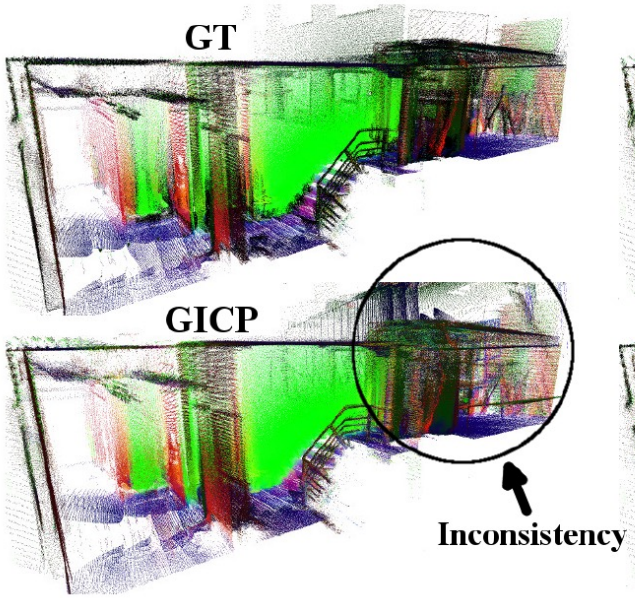
	Relative Translational Error [m]				Relative Rotational Error [deg °]			
Dataset	GICP	DVO	NDT	NICP	GICP	DVO	NDT	NICP
high-fast-fly	**0.284**	1.150	0.452	0.291	20.203	41.831	**17.552**	24.119
high-fast-rot	0.892	0.824	1.142	**0.357**	35.386	24.718	29.081	**14.260**
high-fast-tr	0.115	0.174	0.132	**0.096**	5.933	5.393	6.438	**5.308**
high-medium-fly	0.076	0.249	0.225	**0.069**	**4.760**	9.372	9.833	5.886
high-medium-rot	0.123	0.276	0.217	**0.103**	8.403	8.636	9.960	**8.369**
high-medium-tr	0.046	0.147	0.112	**0.043**	2.538	4.614	5.602	**2.524**
high-slow-fly	0.055	0.128	0.150	**0.053**	**3.189**	4.713	4.857	3.300
high-slow-rot	**0.090**	0.226	0.213	**0.090**	3.956	6.091	5.854	**3.785**
high-slow-tr	**0.038**	0.150	0.064	0.042	1.706	4.531	2.911	**1.433**
low-fast-fly	**0.282**	1.204	0.704	0.294	**12.783**	36.736	19.885	14.643
low-fast-rot	0.329	1.094	0.614	**0.316**	20.775	29.819	26.936	**15.481**
low-fast-tr	0.073	0.519	0.175	**0.072**	**4.938**	14.499	6.465	4.961
low-medium-fly	**0.069**	0.521	0.280	0.077	**3.750**	13.908	6.937	4.301
low-medium-rot	0.075	0.488	0.185	**0.054**	6.979	12.686	8.703	**6.750**
low-medium-tr	**0.063**	0.346	0.094	**0.063**	3.828	8.872	4.899	**3.741**
low-slow-fly	0.059	0.407	0.162	**0.052**	2.842	14.647	3.968	**2.659**
low-slow-rot	0.100	0.466	0.231	**0.089**	4.321	11.435	5.915	**3.741**
low-slow-tr	0.048	0.418	0.114	**0.039**	1.971	9.059	3.013	**1.632**
medium-fast-fly	0.460	1.282	0.652	**0.343**	33.917	38.484	35.010	**29.471**
medium-fast-rot	0.744	0.614	0.330	**0.285**	27.339	18.163	15.617	**13.920**
medium-fast-tr	0.090	0.217	0.128	**0.086**	5.167	5.716	6.278	**5.012**
medium-medium-fly	0.070	0.422	0.236	**0.058**	4.275	12.702	11.081	**4.132**
medium-medium-rot	0.114	0.280	0.160	**0.076**	8.572	**6.209**	8.479	7.693
medium-medium-tr	**0.036**	0.229	0.091	0.038	1.978	5.998	3.937	**1.953**
medium-slow-fly	0.050	0.204	0.113	**0.040**	2.742	6.106	3.564	**2.560**
medium-slow-rot	0.060	0.212	0.111	**0.055**	2.993	5.245	4.532	**2.433**
medium-slow-tr	**0.032**	0.215	0.074	0.034	1.610	6.240	3.068	**1.432**

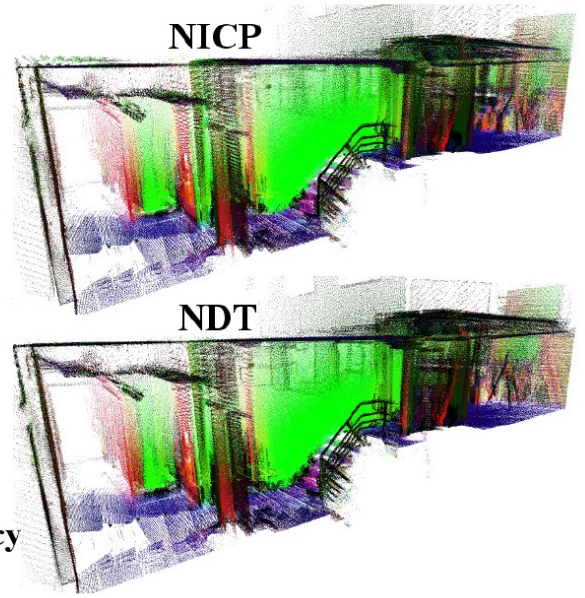
实验结果

2. 在3D激光的数据集[1]中：

- ETH Hauptgebaude：描绘了建筑的主要结构，激光器在走廊中沿直线运动
- ETH Stairs：激光器延走廊经过两次门口从室内走到室外，两次门口之间存在一个5级楼梯







[1] “Challenging data sets for point cloud registration algorithms,” The International Journal of Robotics Research

谢谢

Thank You

THANKS