# 一种改进的迭代最近点算法①

## An Improved Iterative Closest Points Algorithm

蒋成成 胡同森 周 维 (浙江工业大学 信息工程学院 浙江 杭州 310014)

**摘 要：** 不管在三维人脸识别中或点云配准中，ICP(迭代最近点)算法都是当前使用最多的一种匹配算法。然而对于大数据集，ICP 的时间效率很低，这限制了其在各方面的使用。本文提出了一种新的 ICP 算法，因为它基于 Delaunay 剖分，我们称它为 Delaunay-ICP。所有的实验是在 P4 2.0，1G 内存，操作系统为 Windows XP 的 PC 机上完成，结果表明，Delaunay-ICP 的时间效率比 ICP 优秀。

**关键词：** 三维点集 ICP 算法 单位四元数 Delaunay 剖分 Delaunay-ICP

ICP 算法[1]是三维人脸(或点云等等)配准中使用最多的一种匹配算法，它通过迭代优化矩阵，在每次迭代过程中，对目标点集上的每个点，在参考点集中寻找最近点，并利用这样的对应点，计算相应的旋转矩阵和平移向量，将其用于目标点集上，得到新的目标点集并进入下次迭代过程，最终得到优秀的转换矩阵，实现两点集的精确配准[2]。

## 1 引言

最初提出的 ICP 算法，它的计算效率不高，问题主要在于寻找最近点的过程，因为它需要目标点集的每个点都与参考点集的所有点进行比较来确定相应的最近点，基于此，国内外的研究者对此算法做出来很多改进，如:①用点的切平面来逼近点云[3]，最后归结为求点到切平面的最小二乘距离的方法，但这种方法速度仍然很慢。②结合逆向定标法和随机搜寻法提高速度，但会对配准精度产生一定的影响[4]。③使用 ICL 算法[5]，通过直接对两个点云集中的点连线并寻找对应线段的配准。

本文提出了一种基于 n 维 Delaunay 剖分(这里取 n=3)[6]的 ICP，它在原始的 ICP 基础上，在寻找最近点的时候使用 n 维 Delaunay 剖分，它把对应点集剖分成一组单纯形，每个单纯形由四个点构成，在寻找最近点的时候，从这些单纯形入手，大大提高了时间效率。因为使用 Delaunay 剖分，我们称此算法为 Delaunay-ICP。

## 2 ICP算法

ICP(Iterative Closest Points)算法是一种三维匹配算法，其中的一个关键性算法，称为对应点集配准算法，它的目标就是寻找两个点集的坐标变换矩阵，在此可以使用两种方法，SVD(singular value decomposition)或单位四元数法，而实验表明在 2 维和 3 维中[1]，单位四元数法要优于 SVD，因此采用单位四元数法[1,7]。

### 2.1 单位四元数法

Horn[8]提出的单位四元数法最主要的目的就是找到两点集的坐标变换矩阵，通过这个矩阵，使得两点集可以尽可能的重合在一起，因此有两个条件必须满足：两点集点数目必须相同、且两点集中各点必须对应。整个过程可以概述如下：旋转向量为单位四元数 $q_R=[q_0q_1q_2q_3]^T$,其中 $q_0>=0$,且 $q_0^2+q_1^2+q_2^2+q_3^2=1$，这样我们可以得到一个 3×3 的旋转矩阵 $R(q_R)$。再取平移变换矩阵为 $q_T=[q_4q_5q_6]^T$，则完全的变换矩阵 $q=[q_R\,|\,q_T]^T$。经过坐标变换，我们的目的是要两个点集尽量的匹配，也即各对应点间的距离最小，因此我们可以把坐标变换问题转化为求 q 使得

$$f(q)=\frac{1}{N_p}\sum_{i=1}^{N_p}\|x_i-R(q_R)p_i-q_T\|^2 \tag{1}$$

① 收稿时间:2008-11-03

最小。其中

$$R(q_R)=\begin{pmatrix} (q_0^2+q_1^2-q_2^2-q_3^2) & 2(q_1q_2-q_0q_3) & 2(q_1q_3+q_0q_2) \\ 2(q_1q_2+q_0q_3) & (q_0^2+q_2^2-q_1^2-q_3^2) & 2(q_2q_3-q_0q_1) \\ 2(q_1q_3-q_0q_2) & 2(q_2q_3+q_0q_1) & q_0^2+q_3^2-q_1^2-q_2^2 \end{pmatrix} \quad (2)$$

计算出这样一个完全变换矩阵后，目标点集通过这个矩阵进行旋转平移，以尽可能的与参考点集重合，这是 ICP 算法必不可少的一步。

### 2.2 ICP 算法

作为一个三维配准算法，ICP 算法可以归结为，它在每次迭代过程中，在参考点集中寻找目标点集每个点的最近点，然后利用这两组对应点集，配合对应点集配准算法，计算出坐标变换向量和误差，并作用于目标点集中，得到新点集代入下次迭代过程，直到误差收敛，得到最终变换向量及变换后的目标点集坐标。虽然 ICP 算法正在被广泛的使用，但原始的 ICP 算法的时间效率很低，特别是对于大规模的数据集，它在寻找目标点各最近点的时候，花费的时间是不可接受的，因此我们需要对它进行改进，正如下面提出的 Delaunay-ICP。

## 3 Delaunay-ICP算法

正是原始 ICP 时间效率的低下促使研究者对算法进行不断的改进，而本文也提出了一个基于 ICP 的新算法：Delaunay-ICP。通过对点集进行 Delaunay 剖分，可以大大提高点集间的配准效率。

### 3.1 Delaunay 三角剖分

设 P 为二维实数域上的有限点集，边 e 是以 P 中点作为端点构成的线段，E 为 e 的集合，那么 P 的一个三角剖分 T 是一个满足如下条件的平面图：

①除了端点，平面图中的边不含 P 中的任何点

②没有相交边

③ 平面图中所有的面都是三角面，且所有三角面的合集就是 P 的凸包。

E 中的一条边 e(端点为 a,b)，如果存在一个圆经过 a,b 两点，且圆内不含 P 中任何点，则称 e 为 Delaunay 边，如果 P 的一个三角剖分 T 只包含 Delaunay 边，那么该三角剖分称为 Delaunay 三角剖分[6]。

### 3.2 Delaunay-ICP 算法

Delaunay 三角剖分适用于二维，而我们要配准的是三维点集，因此在这里我们使用 n 维(此处取 n=3)Delaunay 剖分[9]，它的原理同 Delaunay 三角剖分大体一样。n 维 Delaunay 剖分把对应点集 X 分成一组单纯形(如四边形),每个单纯形都由四个点构成，这四个点对应的是 X 点集中的坐标。我们假设点集 X 为 m×n,它代表 n 维空间中的 m 个点,经过 n 维 Delaunay 剖分，如果我们用一个数组 T 来代表这些单纯形，则 T 是一个 num×(n+1)的数组，每行保存的是对应单纯形的各顶点坐标。这样要寻找某一点的最近点就不需要同参考点集的所有点进行比较，只需要和相关单纯形中的参考点进行比较就可以得出相应的最近点。以图 1 为例解释这个算法的原理，假设目标点集中的一点 p，寻找参考点集中的最近点 $x_i$，我们设 p 的初始最近点为 x1,(x1,x2,x3,x4)为包含 x1 的一个单纯形，从前面介绍中知道 p 的最近点可能在这个单纯形中，因此求出 p 到(x1,x2,x3,x4)各点的距离，取最短距离的那点，假设 p-x2 的距离小于 p-x1 距离，则继续在包含 x2 的单纯形中用同样的方法找到最近点，直到找到一点 $x_i$，p 到 $x_i$ 的距离最短，$x_i$ 就是最终所要找的最近点。

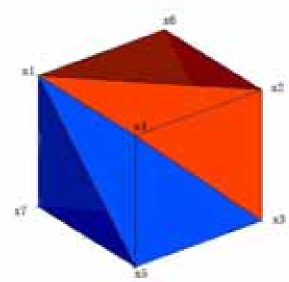

图 1 例图

Delaunay-ICP 算法流程图如图 2 所示

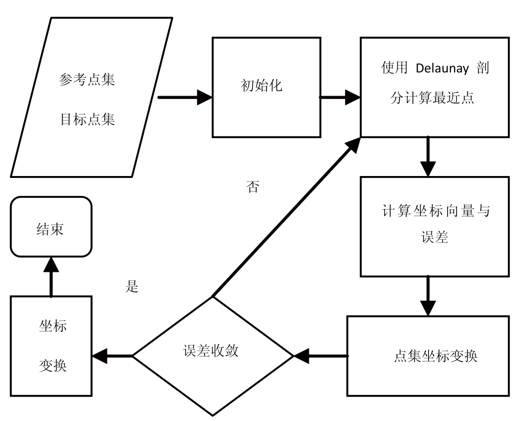


图 2 Delaunay-ICP 算法流程图

整个算法过程如下：

①获得目标点集 P(含有 $N_p$ 个点)，参考点集 X(含有 $N_x$ 个点)

②初始化 $P_0=P$，$q_0=[1,0,0,0,0,0,0]^T$，$k=0$。

③利用 n 维(这里取 3)Delaunay 剖分寻找 P 在 X 中的最近点 Y：$Y_k=Closest(P_k,X)$。

④应用对应点集配准算法计算坐标变换向量以及误差$(q_k, d_k)=Q(P_0, Y_k)$。

⑤应用坐标变换：$P_{k+1}=q_k(P_0)$。

⑥判断误差是否收敛，如果 $d_k-d_{k-1}<\tau\ (\tau>0)$，则收敛，否则跳到步骤③。

⑦最终坐标变换 $P'= q_k(p)$。

⑧ 结束。

ICP 算法的时间效率主要体现在寻找最近点上面，在 ICP 中，求目标点集中每点在参考点集中的最近点，需要同参考点集中的每个点都进行比较，如果两个点集的规模很大，那么寻找所有对应最近点所花费的时间是惊人的，Delaunay-ICP 算法弥补了 ICP 的这个缺点。通过这种方法寻找所有最近点的时间必然比 ICP 优秀，因为它不需要和参考点的所有点都进行比较。

## 4 仿真实验

在这里我们做两个实验，一个是比较参考点集和目标点集在规模相同的情况下所花费的时间，第二个实验是比较参考点集和目标点集在规模不同的情况下所花费的时间。

实验 1：随机产生 N 个点的参考点集 X，为了了解 ICP 算法匹配的效果，所以目标点集 P(包含 N 个点)是经过参考点集的随机旋转平移产生的。我们使用 MATLAB 分别实现上述两种算法，并且在几个不同的数据集上做了实验，所有的实验是在 P4 2.0，1G 内存，操作系统为 Windows XP 的 PC 机上完成，实验结果如表 1,图 3 所示：

表 1　两种 ICP 算法在点集规模相同的情况下的配准速度比较

| 数据点数 N | 平均花费时间（S） | |
|---|---|---|
| | ICP | Delaunay-ICP |
| 500 | 3.1441 | 3.0125 |
| 1000 | 14.5076 | 8.5718 |
| 2000 | 75.2433 | 22.0892 |
| 5000 | 522.7468 | 79.6351 |
| 10000 | 2496.7882 | 255.4165 |

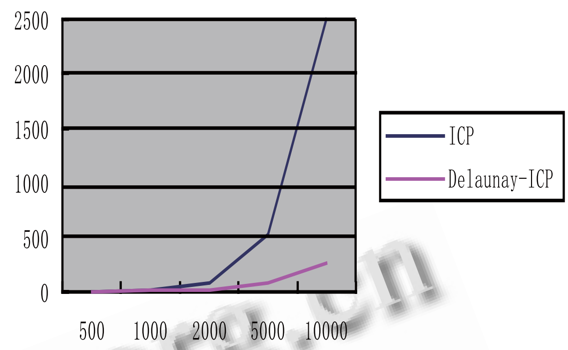


图 3　两算法的配准速度时间图

从表 1 中我们看到，当目标点集和参考点集规模一样的时候，ICP 算法在小规模数据集（500，1000）的时间效率还可以接受，但一旦规模变大后，如 10000 点，就让人无法忍受了，相对的，Delaunay-ICP 的算法的效率还是让人可以接受的，规模越大，两算法的差距也越来越大。

实验 2：随机产生 N 个点的参考点集 X,取 X 中的前 1000 个点作为目标点集 P，并对它做随机旋转平移，实验环境与前一个实验相同，实验结果如表 2 所示。

表 2　两种算法在点集规模不同的情况下的配准速度比较

| 参考点集规模 | 平均花费时间（S） | |
|---|---|---|
| | ICP | Delaunay-ICP |
| 5000 | 90.3041 | 13.6623 |
| 10000 | 247.4399 | 20.9637 |
| 50000 | 2807.7015 | 44.9550 |
| 100000 | >5000 | 65.3476 |
| 150000 | >5000 | 113.4942 |

表 2 中看到，当参考点集规模不同，目标点集规模固定的时候，Delaunay-ICP 的时间效率相对于 ICP 更为明显。

通过这两个实验，我们可以知道 Delaunay-ICP 比 ICP 算法优秀。

取参考点集和数据点数 N=1000 时，原始数据点集与经过 Delaunay- ICP 配准后的数据点集效果图分别如图 4，图 5

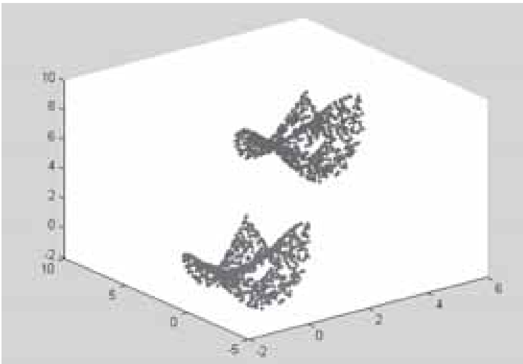

图 4　原始数据点集

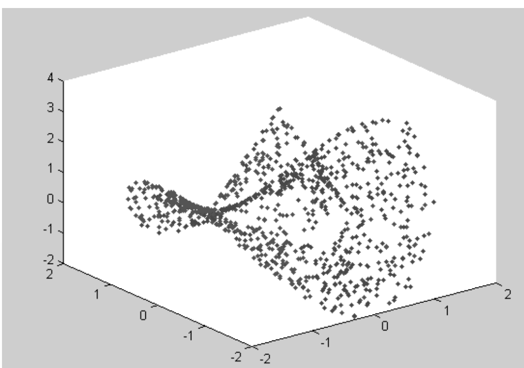

图 5　Delaunay-ICP 配准后的数据点集

## 5　总结

ICP 算法是三维模型(点云等等)配准中常用的一种方法，但对与大规模数据集，ICP 的时间效率太低。本文提出了一种新的算法 Delaunay-ICP，它能够在一定程度上解决 ICP 时间效率低的问题。通过实验结果我们可以看出，Delaunay-ICP 远远优于 ICP 算法，特别是在大数据规模的时候，ICP 所用的时间在实际应用中是不能忍受的，因此在一般的应用中，我们可以采用 Delaunay-ICP 来进行配准。

## 参考文献

1 Besl PJ, McKay ND. A Method for Registration of 3-D Shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence(PAMI),1992,14(2):239－256.

2 张广鹏,张艳宁,郭哲.基于精确主轴分析及 ICP 的三维人脸配准.计算机工程与应用, 2006,29:62－64.

3 Chen Y, Medioni G. Object modeling by registration of multiple range images. Proceedings of the 1991 IEEE International Conference on Robotice and Automation, Sacramento,CA,USA, 1991:2724－2729.

4 Blais G, Levine MD. Registering multiview range data to create 3D computer graphics. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1995, 17(8):820－824.

5 Li Q, Griffiths JG. Iterative closest geometric objects registration. Computers and Mathematic with Application, 2000,40(10):1171－1188.

6 德贝尔赫等.邓俊辉译.计算几何—算法与应用(第二版).北京:清华大学出版社, 2005:206－233.

7 戴静兰,陈志杨,叶修梓.ICP 算法在点云配准中的应用.中国图像图形学报, 2007,12(3):517－521.

8 Berthold K.P.Horn. Closed-form solution of absolute orientation using unitquaternions.Opt. Soc. Amer. A, 1987,4(4):629－642.

9 徐东艳,孟晓刚.MATLAB 函数库查询词典.北京:中国铁道出版, 2006:139－140.