非均匀三维扫描仪融合点云以快速更新地盘模型

在土方工程中，随着工作的进行，地形的形状也在不断变化。如果有人想要获取这些地点的三维几何数据，目前可用的技术（如3D激光扫描仪）不能准确地实时（即每秒1-10次扫描）收集合理精确的数据。 为了解决这个问题，作者采用了“双扫描仪”的方法，其中低速（但高精度）的激光扫描仪用于工作现场的初始扫描，然后使用从挖掘机收集的三维几何数据来增加初始数据， 安装立体视觉。对于这种方法，作者使用ICP（Iterative Closest Point）迭代算法来合并立体视觉数据和激光扫描数据，两个数据的相似性由于变化的位置形状而受到限制。

关键词：激光扫描仪，立体视觉，形状配准，智能挖掘机，土方工程

研究范围和方法

我们研究的范围是开发一个合并的两个三维点云，这些点云是从不同来源获得的，一个来自大范围的激光扫描仪，另一个来自立体视觉相机。 详细的研究方法如下。

1）分析最新的技术和相关研究，以获取3-D信息获取的最新趋势

2）回顾ICP算法（及其衍生物）以寻求算法的改进

3）开发基于 ICP

4）准备测试数据集作为真实世界的数据不可用

5）应用导出的算法对测试数据：“原始”ICP算法用于与我们的算法进行比较，分析两个算法的结果。

合并两个三维点云的方法

对于从多个点扫描的物理对象合并点云以避免阴影，迭代最近点算法（Besl 1992）及其派生被广泛使用。该算法（及其派生）也基本上假定一个点云中的一个点将匹配另一个云中的最近点;点距离的平均值被用作“误差”函数 - 通过计算使函数最小化的变换矩阵，两个点云可以更接近，并且通过迭代地应用计算（因此名称来），可以使它们足够靠近。该算法主要用于注册同一对象的扫描数据，例如机器部件，建筑立面等。为了合并地形数据，由于其形状不规则和形状梯度比算法所期望的更少（例如，合并两个平面形状站点的扫描数据是非常困难的，因为它会匹配局部极小值，而不是正确的注册点。对于我们的系统，ICP算法也有一个问题需要解决：本地模型包含一个更新后的形状，与具有较旧形状数据的相应世界模型显着不同。为了合并这些数据，我们做了两个假设：

首先，向本地数据提供近似坐标信息（误差小于几米，这是在[2]中利用车载GPS和方位传感器完成的）。 用近似信息，收敛到局部最小值的注册结果可以被拒绝。 其次，为了计算最近的距离，我们只选择一定误差范围内的点（即两个最近点之间的距离）。 这个方法是基于Zhang（1992）中使用的方法，但是当迭代计算以抵消由更新点引起的效应时，我们修剪了更多点; 相比之下，传统的ICP使用更多的点进行匹配，因为两个点云应该几乎相同。

来自非均匀源的合并点云

获取土方工地形状

作者参与的研究联盟正在开发一个智能挖掘系统，该系统依靠工作场地的三维形状信息。 为了在更短的时间内获得工地的精确形状模型，使用车载激光扫描仪。 当挖掘机运行时，其安装在挡风玻璃上的立体视觉摄像头捕获工作进度并将其转换成3D点形状，并利用机载GPS和方位传感器捕获近似坐标信息（详见[2]）。 你等。 （2009）描述了帮助正确配准整个场地模型（即世界模型）的硬件和工具的更多细节

准备测试数据

由于我们的智能挖掘机系统正在开发中，我们准备了一套类似土方形状的仿真数据。 包含点云的三维坐标的一组文本文件被创建。 为了表示工作进展的延时表示，创建了三个类似但不完全相同的数据集。 名为“测试图1”的数据集表示初始条件，接着是“测试图2”和“测试图3”。 他们是用Microsoft Excel制作的。 所有这三张地图每个都包含1581个点，除了它们的深度之外，它们位于相同的位置。

用现有的ICP实施进行测试

我们首先使用MATLAB公开的ICP实现（Bergstrom 2006）。该算法使用奇异值分解来计算最近距离的最小二乘。它还使用Delaunay三角剖分从点云生成网格，以生成精确结果的曲面向量。图1显示了MATLAB实现的测试结果。由于操作的输出是一个变换矩阵（用于旋转）和一个平移矢量，在真实世界的情况下，世界模型数据将被替换为正确位置的局部数据，而不是合并。

我们制定了自己的ICP实施来解决这些问题。我们的实现是基于Zhang（1992）提出的算法。该算法使用统计分析来确定要合并的“正确”点，这符合我们需要从计算中消除来自更新区域的点云。此外，该算法提供了合并形状数据，无需表面法线，减轻了计算云网格的需要（降低精度）。为了计算误差函数的变换（确定的最近点之间的平均距离），Zhang的算法使用Walker等人引入的双四元数法。 （1991年）。

我们的实现是用C / C ++编写的，并且用Microsoft Visual Studio 2008编译。为了可视化结果，使用了Meshlab软件（Cignoni et al。2008）。图3显示合并Test Map1和2的结果，其中局部模型相对于参考模型的中心倾斜45度。由于我们仅使用点云（即，表面及其法线不计算更快的计算），所以我们修改了算法以包含法线，从而导致更合理的结果，如图3所示。两个计算都使用相同的系统不到一秒用于运行Matlab实现。对于使用分形算法生成的更逼真的形状，我们的算法表现良好。图4中的右侧图片显示了使用我们的实现的十次迭代的结果，其花费很少时间（即使与我们使用的相同的PC也立即运行），即使没有表面法线也具有合理的精度;另一方面，相同的数据在Matlab中在5-10秒内合并，比使用我们的“平面”测试图的情况更慢但速度更快。