DSP 平台的图像特征点快速索引结构

廖　超，王贵锦，林行刚
（清华大学 电子工程系，北京 100084）

摘　要：针对 DSP（digital signal processor，数字信号处理器）平台上的图像特征点匹配问题，提出了一种高效的基于自聚类二分查找树的快速索引结构，并设计了适合于 DSP 结构特点的索引存储布局。通过在离线情况下将特征点参考数据集逐级地二分聚类，生成多级索引结构。以顺序数组的方式将树状索引结构存储到连续的内存空间中，便于导出为数据文件存储及进一步加载到 DSP 内存中使用。实验表明，该索引结构能够快速有效地在 DSP 平台完成特征点匹配工作。

关键词：特征点匹配；索引结构；二分查找；K-均值聚类；数字信号处理器
中图分类号：TP391　　**文献标志码**：A　　**文章编号**：1001-3695(2012)11-4398-03
doi:10.3969/j.issn.1001-3695.2012.11.103

Fast indexing method for image feature matching on DSP platform

LIAO Chao, WANG Gui-jin, LIN Xing-gang
(*Dept. of Electronic Engineering, Tsinghua University, Beijing* 100084, *China*)

Abstract: This paper proposed a fast indexing method to match image features efficiently on DSP platform. It designed a proper layout scheme to load the indexing structure into DSP memories. It used K-means clustering to split the dataset and generate hierarchical indexing data. The index was saved into sequential array to utilize efficient serialization on disk and reconstruction in DSP memories. Experiments show that this method is fast and robust in matching image features on DSP platform compared with other searching method.

Key words: feature matching; indexing structure; binary search; K-means clustering; DSP

0　引言

近年来提出的图像局部不变性特征如 SIFT/SURF[1,2] 等为图像配准问题提供了非常鲁棒的解决方案。通过检索两幅图像的特征点高维描述，找到两两最相似的特征点并建立对应关系，就可以使用 RANSAC 等随机采样一致性算法[3] 实现准确的配准。由于特征点的稳定性和鉴别性，这种配准方法能有效处理光照视角变化，因此很适合应用于地形匹配、目标定位等领域。然而，提取和检索特征点都是高计算复杂度的运算，在基于移动平台的应用中存在实时性问题，如移动机器人或者飞行器平台上配备的是运算能力较弱的 DSP 系统，相比于性能强劲的计算机，其不仅主频低，而且存储资源非常有限。如何在这类运算平台上快速地检索匹配图像特征点是亟待解决的一个实际应用问题。

输入图像中检测到的特征点集合为 $Q=\{q_i\in \mathrm{R}^n\}$，对其中每一个特征点 q，发起对参考特征点数据库 $P=\{p_j\in \mathrm{R}^n\}$ 的查询，以找到匹配的特征点对，进而实现输入图像到参考图像的配准和定位。匹配特征点 p 应该满足式(1)中的相似性条件，其中 $d(p,q)$ 表示两个特征点在欧式空间中的距离。

$$p=\arg\min_{p\in P} d(p,q) \tag{1}$$

对于上述相似性查询问题，最基本的检索方法为穷尽搜索，即计算查询点与参考数据库中每个特征点 $p_i\in P$ 的距离 $d(q,p_i)$，取其中最小值即为匹配点 p。在不具备参考数据集的任何结构信息时，穷尽搜索法可以很好地解决问题。但如能通过计算了解参考数据集内部特征点的分布情况，并有针对性地建立索引结构，检索时就能够有的放矢，快速地剔除搜索空间中不可能包含有匹配点的部分，加快检索进程。

建立索引一般采用树状结构[4]，按照分割树节点的策略可以简单地分为基于空间划分的方法和基于数据划分的方法。前者用超方体或者超球体将高维数据空间划分为具有包含关系的层次结构，如 Lowe 进行 SIFT 特征点匹配时采用的 KD 树[5] 就是在单个维度方向上划分空间。后者根据数据分布情况划分高维空间，通过比较数据点的远近关系，自底向上以聚类的方式构建索引层次。这些索引结构都受到维数灾难现象（curse of dimensionality）的影响，在数据点维数较低时检索效率正常，在维度上升到几十以上时，检索性能将退化到顺序搜索的水平。维数灾难现象出现的根本原因是索引结构不能准确地刻画高维空间中数据点的真实分布特性。以超方体或者超球体等固定形状切分出来的空间与数据实际分布的区域总存在偏差。基于高维度下数据的稀疏性，索引结构切分的空间几乎都会与查询点邻域相交，导致检索时需要反复回溯防止漏掉候选数据点，这样就违背了建立索引以缩小搜索范围的初衷。为了减少回溯，Liu 等人[6] 提出在划分空间时引入一定的冗余量，这种称为 SP 树的结构在一定程度上提升了检索速度，但冗余度也意味着更多的存储空间。此外，还有学者提出采用

收稿日期：2012-03-15；**修回日期**：2012-05-10

作者简介：廖超(1984-)，男，湖北咸宁人，博士研究生，主要研究方向为并行图像配准(liaoc02@mails.tsinghua.edu.cn)；王贵锦(1976-)，男，副教授，博士，主要研究方向为图像处理、无线网络；林行刚(1947-)，男，教授，博导，博士，主要研究方向为模式识别、视频分析.

降维的方式来解决维数灾难问题,如 Indyk 等人[7]提出的局部 hash 算法 LSH 将高维数据点映射到低维空间中再作匹配。

在 DSP 平台上进行特征点检索的一个优势是数据集规模有限,参考数据库一般不超过几万个特征点,而输入图像中需要查询的特征点数目也在几百的量级。但设备的内存资源非常有限,所以不适宜采用 SP 树等存储需求高的索引结构。而 KD 树检索需要 Best-Bin-First[8]回溯,效率还不够高。Anan 等人[9]通过随机化切分维度的方式构建多棵 KD 树,优化了检索精度,但存储空间也倍增。自底向上的数据聚类索引方法由于划分过细,对存储资源利用效率不够高。Trzcinski 等人[10]指出将特征描述子二值化后再检索可提高性能,但二值化模块可能对系统复杂度造成影响,不一定适合 DSP 平台。

本文提出一种自聚类的二分索引结构及适应 DSP 平台结构特点的存储方式,通过采用 K 均值聚类方法自顶向下逐层划分数据集,并使叶节点包含有一定的数据规模,在索引深度和存储空间之间取得较好的平衡。实验表明,这种结构可以达到良好的检索速度,适合在 DSP 平台上使用。

1　索引结构

参考数据库的索引可以离线建立,所以可充分计算并利用数据分布特性,在检索时尽量避免回溯,以达到实时匹配的目的。此外,设计合理的内存布局和存储加载方式,可以提高索引的硬件执行效率和易用性。

1.1　建立索引

构建索引本身是一个对高维数据空间进行划分的过程。其关键部分在于确定按照何种准则将数据集划分到左右两个孩子节点中去。KD 树是选择具有最大方差的一个维度切分。SP 树则复制分割面附近的数据点给左右两个孩子节点各分配一份,相当于每次切分都增加了少量数据。本文在每个层次都采用 K 均值两类聚类的方法来划分数据集,并将两类中心记录在该层节点中,作为查询时的判定依据。构建索引的过程可以离线完成,因此基本不需要考虑这部分的运算复杂度。采用 K 均值聚类能够充分地描述数据集的簇状结构,可以避免自底向上的聚类方法陷入局部最优的问题。自聚类二分索引树的构建是一个逐级切分数据集的递归过程,建立索引步骤如下:

a)初始化。建立根节点 Root,对应参考数据集$\{p_i\}$,记为$\{\text{Root},\{p_i\}\}$。

b)展开。用 K 均值两类聚类算法将$\{p_i\}$划分为左右两类$\{p_i^l\}$和$\{p_i^r\}$,得到左子节点$\{N_l,\{p_i^l\}\}$和右子节点$\{N_r,\{p_i^r\}\}$。两类中心分别为 l_c 和 r_c,将根节点 Root 重记为$\{\text{Root},l_c,r_c\}$。

c)递归。依次进一步展开左右孩子节点。若当前节点对应的数据集大小不超过 η,则该节点标记为叶子节点;否则按步骤 b)展开当前节点。

重复过程,直到所有叶节点数据集都小于 η。

数据集划分的过程是原地工作的,在索引节点中记录其中管辖的数据点在存储数组中的起始位置和终止位置就完成了数据切分,有时需要交换一下数据存储的位置,但不需要额外的存储空间或者数据拷贝。

1.2　查询索引

查询索引是对二分查找树的一个深度优先检索的过程,其查询步骤如下:

a)深度优先检索。对查询特征点 q,从根节点$\{\text{Root},l_c,r_c\}$开始检索。

(a)计算 q 与左右子树的类中心点的距离 $d(q,l_c)$、$d(q,r_c)$;

(b)如果 $d(q,l_c)>d(q,r_c)$,则检索右子树,否则继续检索左子树;

直到到达某个叶节点$\{N_y,\{p_i^y\}\}$。

b)穷尽搜索。逐一计算 q 与 N_y 所辖数据点的距离 $d(q,p_i^y)$,设 $d_1=\min\limits_i d(q,p_i^y)$ 为 q 到最近邻数据点 $p_{i_1}^y$ 的距离,$d_2=\min\limits_{i\neq i_1} d(q,p_i^y)$ 为 q 到次近邻数据点 $p_{i_2}^y$ 的距离。

c)匹配判断。设定门限 thr,如果$\dfrac{d_1}{d_2}<\text{thr}$,则 $p_{i_1}^y$ 即为查询到的匹配点,否则无匹配。

在上述检索过程中,每检索一级索引节点,都将搜索范围减少近一半,而剔除的另外一半就是相比穷尽法得到的性能提升。越接近叶子节点时,索引结构带来的性能提升越少。相应地,检索树节点需要付出一定的代价,即比较查询点到左右两个子树类中心的距离,这相当于在穷尽搜索中比较了两个数据点。因此,索引树不宜将数据集划分过细,叶子节点中最好包含有一定的数据点。如果索引树的叶子节点中只包含一个数据点,那么最后一步的检索将得不偿失。为此本文使用参数 η 来控制数据划分的深度,如建立索引时设置叶节点可以包含 $\eta=12$ 个数据点。在检索过程到达叶子节点以后,直接枚举比较其中包含的数据点来判定匹配。

上述检索过程并没有类似 KD 树采用的回溯操作,这是假设在分割聚类的过程中已经将数据集作了很好的划分,而查询点的邻域与分割面没有太大的交集。在实际检索过程中,这种假设基本成立,体现在检索结果与采用回溯操作的结果差别不大,绝大部分特征点匹配都能够在无回溯的情况下检索到。因此本文的检索方法相比 KD 树等需要回溯的方法具有更低的时间复杂度,查询效率较高。

1.3　存储加载

最终建立的索引结构是为了在 DSP 平台上进行快速、有效的查询,因此需要设计一套有效的存储结构,使 PC 端建立的树状索引能够导出成数据文件形式存储,需要时能够方便地导入 DSP 内存里再次重建出来。同时也需要针对 DSP 的体系结构特点设计出一种合适的内存布局,充分发挥 DSP 平台具备多套数据总线的优势,提高检索速度。

如图 1 所示,本文采用顺序的结构体数组来表示有层次的树状结构。这种方式主要有两个优点:

a)以相对寻址代替绝对寻址,方便索引结构的存储和重建。树节点的位置都用数组下标的方式来标注。在 PC 端父节点原本是通过指针记录的内存地址直接寻找孩子节点,存储为数组结构以后,父节点通过记录孩子节点的数组下标来间接定位左右孩子节点。这种方式实际上是通过与根节点的相对位置来定位各个索引节点。在存储和重建时,只要提供新环境中树根节点或者数组首元素的内存位置,则索引节点的位置都可以通过相对位置方便地计算出来,不需要修改索引结构的内部内容。

b)由于顺序数组的存储空间是连续的,所以内存的利用效率很高,可以直接将索引结构从 PC 内存中以二进制比特流

的形式导出到数据文件中存储,进而导入到 DSP 内存中重建。

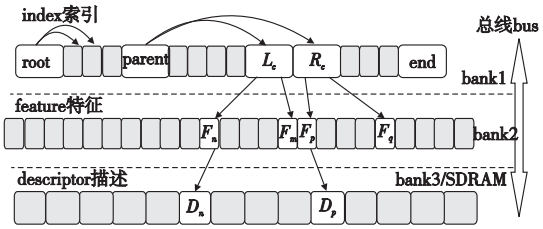

图1 索引存储的数组结构

针对 DSP 结构特点设计的内存布局也将有利于快速检索。以 TigerSHARC201 系列芯片为例,片内有限的存储空间由六块内存 bank 组成,不同的 bank 配置有独立的读写缓存。针对该特点,本文将索引、特征点和描述子数据分为三个部分放置在不同的存储空间中,如图 1 所示。索引与特征点的基本信息如位置、方向等分别存储在两个内存 bank 中,描述子的数据量比较大,单独放在一个 bank 或者外部存储器中,相互之间都是通过相对指针来寻找对应关系。这种设计在检索时可以同时利用多个 bank 的读写缓存,减少了缓存换页次数,提高了缓存数据命中率,有利于提高检索效率。此外,由于检索过程只与输入的查询数据点有关,通过把查询数据集分配到多片 DSP 内核上同时执行检索,可以并行地完成查询并获得匹配的特征点数据。

2 实验结果

本文首先在 Intel Core 3.0 GHz 的 PC 机上比较了自聚类索引方法与前述的常见索引在不同数据集规模下的性能表现。比较实验没有选择直接在 DSP 上进行,一方面是因为 DSP 内存上装载不了太多特征点,另一方面有些比较对象不适合在 DSP 上实现。数据集从独立的图像库中提取。图 2 是各索引的构建时间曲线。从图中容易看出,自聚类的二分搜索树方法在构建索引时需要的时间最多,Lowe 使用的 KD 树以及 Liu 等人 SP 树的方法在建树过程中需要的时间较少。这个时间差异是可以预料的,因为自聚类方法在二分查找树的每一层都对数据分布情况作了充分计算,而 KD 树只是在一个维度上对数据分布作了估计,SP 树虽然引入了冗余度,但是也没有充分地计算数据分布情况。由于这一步是在离线状态下计算,其运算复杂度不影响检索的实时性。实际上,这一步开销越大说明索引对数据集的刻画越准确。

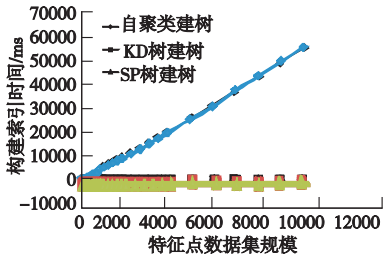

图2 构建索引的时间

图 3 是各索引的查询时间曲线。实验中统计的是数据集自身到自身的查询时间。线性扫描方法即穷尽搜索,图中显示检索时间随数据集规模以平方的速度增长。KD 树和 SP 树的检索时间也随数据集规模明显增加。KD 树采用回溯法求取最近邻,因此其搜索时间比较长。SP 树由于引入了冗余,一定程度上减少了需要回溯的空间范围,因此搜索时间相对于 KD 树明显减少。本文的自聚类方法由于构建时花费了大量的代价计算数据集的分布情况,所以在检索时能够有效地界定最近邻点出现的空间范围。其检索复杂度在 $O(\log N)$ 量级,实际检索时间基本保持在几十毫秒以内。

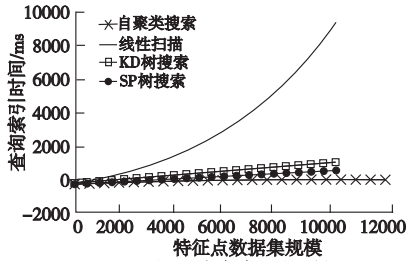

图3 检索索引的时间

在双片 TigerSHARC201 系列 DSP 芯片组成的平台上的测试也验证了本文方法的有效性。320 × 256 的输入图像提取出的上百特征点与包含上千特征点的参考数据库匹配可以在 1.2 ms 完成。结合并行化特征点(SURF)提取模块[2],可以在 120 ms 以内接近实时地完成图像配准。

3 结束语

本文针对 DSP 平台上的实时图像配准应用提出了一种快速的特征点索引及对应的存储结构。实际平台上的测试证明了该索引能快速地完成特征点匹配,存储结构也便于从 PC 端制备数据文件和导入 DSP 内存重建。今后将进一步研究多核并行处理对检索的加速作用。

参考文献:

[1] LOWE D G. Distinctive image features from scale-invariant key points [J]. International Journal of Computer Vision, 2004, 60(2): 91-110.

[2] BAY H, ESS A, TUYTELAARS T, *et al*. Speeded-up robust features (SURF) [J]. Computer Vision and Image Understanding, 2008, 110(3): 346-359.

[3] FISCHLER M A, BOLLES R C. Random sample consensus: a paradigm for model fitting with application to image analysis and automated cartography [J]. Communications of the ACM, 1981, 24(6): 381-395.

[4] 刘芳洁,董道国,薛向阳. 度量空间中高维索引结构回顾[J]. 计算机科学,2003,30(7):64-68.

[5] BENTLEY J L. Multidimensional binary search trees used for associative searching [J]. Communications of the ACM, 1975, 18(9): 509-517.

[6] LIU T, MOORE A, GRAY A, *et al*. An investigation of practical approximate nearest neighbor algorithms [C]//Proc of Conference of Neural Information Processing Systems. 2004: 825-832.

[7] INDYK P, MOTWANI R. Approximate nearest neighbors: towards removing the curse of dimensionality [C]//Proc of the 30th Symposium on Theory of Computing. Now York: ACM Press, 1998: 604-610.

[8] BEIS J, LOWE D G. Shape indexing using approximate nearest-neighbour search in high-dimensional spaces [C]// Proc of IEEE Conference on Computer Vision and Pattern Recognition. [S. l.]: IEEE Press, 1997: 1000-1006.

[9] ANAN C S, HARTLEY R. Optimised kd-trees for fast image descriptor matching [C]//Proc of IEEE Conference on Computer Vision and Pattern Recognition. [S. l.]: IEEE Press, 2008: 1-8.

[10] TRZCINSKI T, LEPETIT V, FUA P. Thick boundaries in binary space and their influence on nearest-neighbor search [R]. Lausanne: EPFL, 2011.