基于GPU的实时图像拼接

郭一汉 史美萍 吴 涛
(国防科技大学机电工程与自动化学院 长沙 410073)

摘 要 大视野、高质量的图像信息对地面移动机器人的遥控操作具有非常重要的意义。提出了一种基于先验信息的自适应图像拼接方法。该方法在图像大致重叠区域中均匀选取待匹配点,利用改进的具有旋转不变性的NCC(Normalized Cross Correlation,归一化互相关)匹配方法进行区域相似性度量,通过RANSAC(Random Sample Consensus,随机采样一致性)算法估计图像射影变换模型,采用线性淡入淡出法进行图像融合。利用GPU强大的并行处理能力对算法进行了并行化实现,使图像拼接效率比单独采用CPU提高了60倍以上,稳定的拼接速度可达21.3fps。

关键词 图像拼接,并行图像处理,RANSAC,GPU,Computer Unified Device Architecture
中图法分类号 TP391 **文献标识码** A

Real Time Image Mosaic Based on GPU

GUO Yi-han SHI Mei-ping WU Tao
(College of Mechatronics and Automation, National University of Defense Technology, Changsha 410073, China)

Abstract Telepresence with wide field of vision is one of the key technologies for teleoperated unmanned vehicle. To meet the real-time requirement, a fast image mosaic method was proposed, which selects evenly distributed points and calculates their similarity by rotation-invariant NCC (Normalized Cross Correlation) operator, calculates the projective transformation model by RANSAC (Random Sample Consensus) method, and fuses images by linear fade in and fade out. Finally, the algorithm was accelerated with a parallelized self-adaptive image matching method using GPU (Graphic Process Unit). Experiment results show that the efficiency of the parallelized image mosaicing method, compared with that of the serial scheme on CPU, is dramatically improved with more than 60 times on frame rate.

Keywords Image mosaic, Parallel image mosaic, RANSAC, GPU, CUDA

1 绪论

要遥控地面移动机器人,首先必须使操作人员获得足够的、实时的远端场景信息,使其操作就像在远端现场一样[1]。视觉是人类获取信息、观察世界和认识世界的最重要方式之一,实时的、大视野的图像信息对于地面移动机器人的遥控操作具有非常重要的意义。然而,由于图像拼接过程中涉及的数据量大、计算密集,因此许多串行处理方法在实际应用中都难以满足实时性要求。随着并行计算软硬件技术的发展,图像处理的并行化已成为图像技术领域和计算机科学领域一个重要的学科方向[2]。GPU在并行计算、分布计算和浮点运算方面,拥有数十倍乃至上百倍于CPU的运算能力。用GPU进行通用计算(General Purpose Graphic Process Unit,GPGPU),减少程序运算过程的时间代价,大幅提升程序的执行效率,是近年来GPU发展的一大特色和趋势,受到了研究人员的广泛关注。国内外研究人员对此展开了一系列研究,在基于GPU的数据库操作、图像处理、代数计算以及其它方面取得了许多进展[3]。为此,研究基于GPU的并行图像拼接技术,并将之运用到地面移动机器人的遥控技术中,不仅对于提高遥控系统的性能有着重要的现实意义,而且对所有涉及到计算机视觉、图像处理工程和系统的设计实践也有着十分重要的理论和指导意义。

图像拼接过程主要分为图像获取、图像预处理、图像匹配和图像融合4个步骤,其中最核心的技术就是图像匹配。图像匹配的关键是精确确定两幅图像中重叠部分的位置,从而得到两张图像的变换关系。该过程计算复杂度较大,可采用GPU并行化加速。通常,用在图像拼接和立体视觉技术中的匹配方法主要有基于特征的图像匹配算法和基于区域的图像匹配算法两类。基于特征的图像匹配通过查找图像的边缘、轮廓、角点、线段等几何特征,对图像进行匹配,计算量相对较小,但相对区域匹配而言,其算法实现中有大量的随机数据访问以及条件分支,不利于用GPU进行高度并行化实现。

本文给出了一种基于区域相似性的、改进的归一化互相关(Normalized Cross Correlation,NCC)算法,它检测图像中的匹配点,并通过随机采样一致性(Random Sample Consensus,RANSAC)算法进行参数估计,得到图像之间的射影变换模型。经过射影变换、最佳拼接缝选择与图像融合后,得到最终的拼接结果。其中,具有旋转不变性的NCC匹配、基于

到稿日期:2011-08-23 返修日期:2012-03-19 本文受高速公路车辆智能驾驶中的关键科学问题研究(90820302)资助。
郭一汉(1987-),男,硕士生,主要研究方向为模式识别与智能系统,E-mail:guo_yihan@126.com;史美萍(1969-),女,博士,副教授,主要研究方向为虚拟现实与仿真可视化、模式识别;吴 涛(1975-),男,博士,副教授,主要研究方向为机器学习、模式识别、机器视觉。

RANSAC的图像变换模型求解以及图像射影变换，都是在GPU上并行完成的，具有非常高的实时性。

2 CUDA编程模型

CUDA(Compute Unified Device Architecture，统一计算设备架构)是一种将GPU作为并行计算设备的软硬件架构。在CUDA编程环境中，CPU作为主机(Host)，负责进行逻辑性强的事务处理和串行计算，以及GPU上线程的创建、显存的申请与数据存取等工作；GPU作为设备(Device)，专用于执行高度线程化的并行运算。

在GPU上运行的并行计算函数称为Kernel。Kernel是以线程块(Block)为单位执行的。Block之间并行、无序执行，且各Block之间无法进行通信。每个Block可以由1～512个线程(Thread)组成，一般GPU发挥最佳性能的线程数是64～256个，但是具体分配多少个线程要根据实际情况来定。合理的线程划分对程序的执行效率具有很大影响。

CUDA中的线程在执行时会访问到处于多个不同存储空间中的数据。在算法设计过程中，必须根据各种存储器的大小、访问速度、可读写性以及算法特点选择合适的存储器，才能使CUDA的运行效率最高。表1给出了CUDA中各类存储器的属性[4]。

表1 CUDA编程模型中各类存储器的基本属性

存储器类型	访问权限	空间大小	访问速度	是否只读
寄存器	单个线程	非常有限	快	否
本地存储器	单个线程	有限	慢	否
共享存储器	块中所有线程	非常有限	快	否
全局存储器	所有线程	大	慢	否
固定存储器	所有线程	有限	慢	是
纹理存储器	所有线程	大	慢	是

3 结合先验信息的自适应图像匹配方法

图像拼接的核心问题是图像匹配，即寻找场景中的同一目标在两幅或多幅图像中的像点之间的对应关系。通常，待拼接的相邻两幅图像(分别称为左图像 $\boldsymbol{I}_L$ 和右图像 $\boldsymbol{I}_R$)的重叠区域的大致范围总是可以事先获得的。因此，本文针对实时图像拼接的应用需求，提出了一种基于先验信息的自适应图像匹配方法。其基本思想为：首先根据事先得知的待拼接图像的重叠区域范围，在左图像 $\boldsymbol{I}_L$ 的重叠区域内均匀选取 $k_m\times k_n$ 个待匹配点 $P_i(i=1,2,\cdots,k_m\times k_n)$；然后通过区域相似性度量和自适应匹配搜索区域(Search Region，SR)在右图像 $\boldsymbol{I}_R$ 中进行最佳匹配搜索，得到 $k_m\times k_n$ 个最佳匹配点 P_i^* $(i=1,2,\cdots,k_m\times k_n)$。

3.1 自适应图像匹配算法的计算复杂度

对于两幅待匹配图像 $\boldsymbol{I}_L$ 和 $\boldsymbol{I}_R$，假设在 $\boldsymbol{I}_L$ 中有 k 个待匹配点，匹配窗口 W 的大小为 $W_d\times W_d$，所选择的相似性度量算子的计算复杂度为 f，则在 $\boldsymbol{I}_R$ 中进行最佳匹配点搜索的计算量为

$$T=k\times SR_m\times SR_n\times W_d^2\times f \tag{1}$$

式中，$SR_m\times SR_n$ 表示自适应匹配搜索区域的大小。由此可见，自适应图像匹配算法的计算复杂度主要取决于待匹配点的个数、匹配窗口的尺寸、相似性度量算子和搜索区域大小4个方面。

3.2 待匹配点的选取

由于图像拼接的关键是找到两幅图像之间的变换模型，因此它并不需要太多的匹配点。对于射影变换来讲，理论上只需要4对匹配点即可得到其变换模型，但过少的匹配点会使匹配精度和稳定性无法保证。因此，在本文的算法中，根据事先得知的待拼接图像的重叠区域范围，在左图像 $\boldsymbol{I}_L$ 的重叠区域内均匀选取 $k_m\times k_n$ 个像素点 $P_i(i=1,2,\cdots,k_m\times k_n)$ 作为待匹配点。

在预计的重叠区域内均匀选取 $k_m\times k_n$ 个待匹配点，具有如下优点：

1) 避免了角点检测，减少了算法的计算复杂度；

2) 待匹配点分布均匀，数量稳定；

3) 待匹配点之间已知的结构信息是判断匹配结果是否正确的有效依据；

4) 在边缘特征不明显的图像(如越野环境)中具有较强的适应性。

3.3 区域相似性度量

区域相似性度量是以待匹配图像 $\boldsymbol{I}_L$ 中待匹配点 P_i 为中心像素来创建一个匹配窗口 W，用窗口内图像的灰度信息来表征该像素的特征。在图像 $\boldsymbol{I}_R$ 搜索区域 SR 中取出一个与 W 同样大小的像素邻域，根据相似性度量准则计算两个窗口之间的相似程度。

在实际应用中，匹配窗口 W 的大小和方向的选取至关重要。其中匹配窗口的大小不仅直接影响计算量，对匹配的效果也有很大影响。一方面，窗口的尺寸要尽量大，以便为可靠的匹配包容足够多的灰度变化；另一方面，窗口太大则包含更大的视差变化，由于左、右图像上不同的投影畸变，使得匹配的准确度下降。

对于匹配窗口的方向选取，传统的矩形窗口下的匹配算法主要是针对经过极线校正的立体图像进行匹配，不具有旋转不变性。而对于图像拼接而言，待拼接的两幅图像一般都未经过畸变校正和极线校正，两幅图像中对应点之间可能存在着旋转变换。为了解决这一问题，本文提出了一种考虑待匹配点主方向的匹配窗口选取方法，使其具有旋转不变性。

对于一个待匹配点 $p(x,y)$，可以根据该点的方向导数：

$$\boldsymbol{u}(x,y)=\nabla p(x,y)=\begin{bmatrix}\dfrac{\partial p}{\partial x}\\ \dfrac{\partial p}{\partial y}\end{bmatrix} \tag{2}$$

来确定匹配窗口的方向。为了使得待匹配点主方向的估计对点位置的误差具有鲁棒性，要求图像的局部梯度场 $\boldsymbol{u}(x,y)$ 平滑变化：

$$\boldsymbol{u}_l(x,y)=\nabla_{\sigma_0} p(x,y) \tag{3}$$

式中，σ_0 是一个平滑滤波的因子。在计算主方向之前对原图像进行高斯平滑滤波，以得到平滑的梯度值。然后计算每个待匹配点计算的方向向量：

$$[\cos\theta,\sin\theta]=\frac{\boldsymbol{u}}{|\boldsymbol{u}|} \tag{4}$$

式中，θ 为点 $p(x,y)$ 处匹配窗口的主方向。

除了匹配窗口大小和方向，选择恰当的相似性度量算子也对匹配结果和匹配效率有至关重要的作用。常用的相似性度量算子有SSD(Sum of squard difference)、ZSSD(zero mean SSD)、SAD(sum of absolute difference)、ZSAD(zero mean

SAD)、NCC(normalized cross correlation)、ZNCC(zero mean NCC)等。其中最经典、最常用的就是归一化互相关法(NCC),该算法利用两幅图像中对应的匹配窗口内像素灰度值的互相关大小来寻找匹配点对,准确度高,鲁棒性好。为此本文采用了式(5)给出的NCC算子进行区域相似性度量:

$$c_{NCC}(p,p')=\frac{\sum_{(x,y)\in W,(x',y')\in W'} I_L(x,y)I_R(x',y')}{\sqrt{\sum_{(x,y)\in W} I_L^2(x,y)\sum_{(x',y')\in W'} I_R^2(x',y')}} \tag{5}$$

式中,W、W'分别为$\boldsymbol{I}_L$和$\boldsymbol{I}_R$中以p、p'为中心、以θ为主方向的匹配窗口。在与待匹配点主方向相同的矩形区域中进行相似性计算,可以使算法具有旋转不变性。

3.4 自适应匹配搜索区域

由式(1)可知,搜索区域SR的大小对算法效率的影响不可忽视。过大的搜索范围不仅会严重影响算法的实时性,而且会使误匹配的概率增大。为了解决这一问题,本文利用待匹配图像$\boldsymbol{I}_L$和$\boldsymbol{I}_R$之间已知的大致重叠范围,自动计算得到左右图像间的偏移量,并利用待匹配点之间的结构信息来缩小搜索范围,使搜索区域能够自动适应待匹配点的位置。具体实现方法为:在算法启动时,先令搜索区域SR为$\boldsymbol{I}_L$和$\boldsymbol{I}_R$的大致重叠区域,从$P_i(i=1,2,\cdots,k_m\times k_n)$中选取位于重叠区域中部的3个待匹配点,在$SR$中搜索得到3个最佳匹配点,根据这3个匹配点的位置计算得到$\boldsymbol{I}_L$和$\boldsymbol{I}_R$之间的大致偏移量$Disp_{xy}$;之后,在对其它的待匹配点进行搜索时,就可以根据$Disp_{xy}$和待匹配点在$\boldsymbol{I}_L$中的位置,初步估计出它在$\boldsymbol{I}_R$中的对应点位置,并以此作为搜索区域SR的中心。在确保SR能够包括真实匹配点的前提下,使其尽可能地小,以减小搜索范围。

4 基于RANSAC的图像变换模型求解

常用的二维平面图像变换模型有欧拉变换、仿射变换、射影变换等。本文中采用自由度最高的射影变换作为图像变换模型。射影变换是在齐次坐标系下的单应性变换,有8个自由度,可以表示成

$$\begin{bmatrix}x\\y\\1\end{bmatrix}=\begin{bmatrix}h_{00}&h_{01}&h_{02}\\h_{10}&h_{11}&h_{12}\\h_{20}&h_{21}&1\end{bmatrix}\begin{bmatrix}x^*\\y^*\\1\end{bmatrix} \tag{6}$$

式中,$(x^*,y^*,1)^{\mathrm{T}}$为待匹配点P_i在$\boldsymbol{I}_R$中的最佳匹配点P_i^*的齐次坐标,$(x,y,1)^{\mathrm{T}}$为P_i^*经过射影变换后在$\boldsymbol{I}_L$中的齐次坐标,射影变换矩阵记为$\boldsymbol{H}$。

由式(6)可知,对射影变换模型的求解,可根据$\boldsymbol{I}_L$中的待匹配点P_i及其在$\boldsymbol{I}_R$中的最佳匹配点P_i^*来估计变换矩阵$\boldsymbol{H}$中各元素h_{ij}的值。考虑到在图像拼接的实际应用中,匹配点总是有限的,而且其中可能包含无法得到补偿的严重错误数据,本文采用RANSAC算法进行射影变换模型参数估计,其基本思想如下:

1)假设样本集为P,初始化模型参数所需的最小样本个数为n,集合P所包含的样本数大于n。从P中随机抽取包含n个样本的子集S,初始化模型,记为$\overline{\boldsymbol{H}}$;

2)余集$SC=P/S$中与模型$\overline{\boldsymbol{H}}$的误差小于某一给定阈值$\varepsilon$的样本以及$S$构成$S'$,称为内点集;

3)若S'中的样本个数大于某一给定的阈值N,则认为得到正确的模型参数;

4)重新抽取新的S,重复以上步骤。在完成一定的抽样次数后,若未找到S',则算法失败;否则选取抽样后得到的样本数最大的S',记为S^*,并根据S^*内所有的样本,采用最小二乘法重新计算模型参数,得到$\boldsymbol{H}$,算法结束。

RANSAC算法包括了3个输入参数:判断样本是否满足模型的误差容忍度ε,该参数对算法性能和正确性有很大的影响;随机抽样的次数T,采样次数影响着算法所得到模型的正确性;表征得到正确模型时内点集S'的大小N,一般要求N足够大,以有足够多的一致样本,使得到的模型更精确。

5 基于CUDA的实时图像拼接算法

5.1 实时图像拼接算法的基本模块与流程

如图1所示,基于CUDA的实时图像拼接算法分为Host端和Device端两部分。其中,Host端主要完成图像读取、待匹配点$P_i(i=1,2,\cdots,k_m\times k_n)$的选取、GPU资源分配、自适应搜索区域$SR$计算和图像融合等功能;Device端则负责完成最佳匹配点的并行化搜索、射影变换模型并行求解以及对图像进行射影变换等功能。

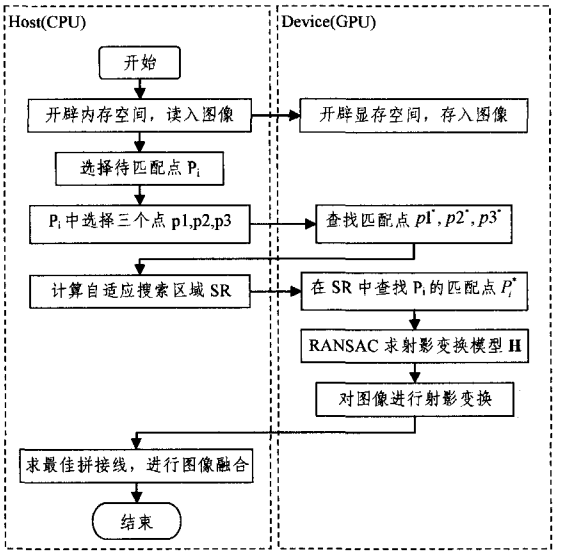


图1 基于CUDA的实时图像拼接算法的基本模块与流程

下面将重点针对GPU中实现的最佳匹配点并行搜索算法和射影变换模型并行求解算法进行论述。

5.2 最佳匹配点并行搜索算法

对于$\boldsymbol{I}_L$中的待匹配点$P_i(i=1,2,\cdots,k_m\times k_n)$,本文给出了基于GPU的最佳匹配点并行搜索算法,具体描述如下:

1)创建一个Kernel函数,其所对应的Grid中包含SR_m个Block,每个Block中包含SR_n个Thread,如图2所示。每个Thread对应于$\boldsymbol{I}_R$中的一个像素p_j',($p_j'\in SR$),在Thread内计算得到p_j'与待匹配点p_i的区域相似性$c_{NCC}(p_i,p_j')$。Grid中所有的Thread计算得到的结果组成一个关于p_i的$SR_m\times SR_n$的相似性度量矩阵$\boldsymbol{C}_{NCC}$。

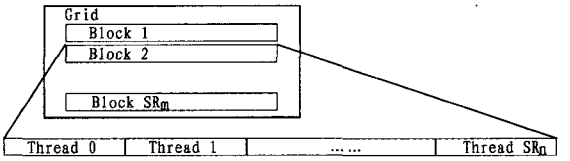


图2 GPU中Thread与Block的划分示意图

2)创建第二个Kernel函数,其所对应的Grid中仅包含1个Block,Block中包含SR_n个Thread。每个Thread实现

C_{NCC}中每一列极大值的查找，得到 SR_n 个局部极大值 $C_{NCC}(\max,j),(j=1,2,\cdots,SR_n)$。

3)创建第三个 Kernel 函数，其所对应的 Grid 中包含 1 个只有 1 个 Thread 的 Block，用以实现从 $C_{NCC}(\max,j)(j=1,2,\cdots,SR_n)$中找出最大值 $\max(C_{NCC})$，$\max(C_{NCC})$所对应的像素点即为待匹配点 p_i 的最佳匹配点 p_i^*。

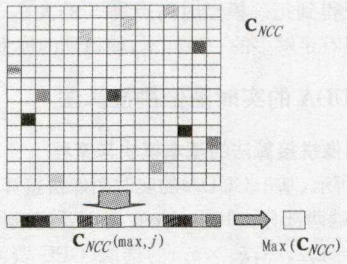


图 3　相似性度量矩阵 C_{NCC}中的最大值搜索示意图

本文中，匹配窗口之间的相似性度量是在单个 Kernel 函数中实现的，不受线程划分以及 GPU 存储器分配的影响。因此，不用改变程序结构，只要改变 Kernel 函数中的相似性度量公式，即可实现不同准则下的相似性计算。

5.3　基于 RANSAC 的射影变换模型并行求解算法

RANSAC 算法存在大量的条件分支和逻辑判断，为了提高程序并行化运算效率，本文给出了基于 RANSAC 的射影变换模型并行求解算法。具体实现分为以下 4 个步骤：

步骤 1　初始化算法参数。

如第 3 节所述，算法包含了采样次数 T、误差容忍度 ε、判断模型是否正确的阈值 N。为了充分利用 GPU 的资源，提高效率，T 设置为 64 的整数倍；ε 等于待匹配点间距离的1/5；$N=\frac{k_m\times k_n}{2}$，为待匹配点个数的一半。

步骤 2　每次随机采样相应的射影变换模型求解。

1)令样本集为 $P=\{p_i^* \mid p_i^*=(x_i^*,y_i^*),i=1,2,\cdots,k_m\times k_n\}$，其中 $p_i^*=(x_i^*,y_i^*)$表示左图像 $\boldsymbol{I}_L$ 中第 i 个待匹配点 p_i 在右图像 $\boldsymbol{I}_R$ 中的最佳匹配点；

2)创建一个 Kernel 函数，其所对应的 Grid 中包含 $T/64$ 个 Block，每个 Block 中包含 64 个 Thread。

3)每个 Thread 负责在 P 中进行随机采样，得到 4 个不共线的样本点，然后根据这 4 个样本点及其在 $\boldsymbol{I}_L$ 中的对应点，利用式(6)计算 $\overline{\boldsymbol{H}}$。其中 $i=1,2,\cdots,T$，是 Thread 的下标，对应于不同次的采样。

步骤 3　计算 $\overline{\boldsymbol{H}}$ 对应的内点集 S'。

1)创建一个 Kernel 函数，其所对应的 Grid 中包括 T 个 Block，每个 Block 中包括 $k_m\times k_n$ 个 Thread，其中第 i 个 Block 以 $\overline{\boldsymbol{H}}_i$ 为输入参数。

2)每个 Thread 利用 $\overline{\boldsymbol{H}}_i$ 和式(6)将 p_i^* $(i=1,2,\cdots,k_m\times k_n)$映射到 $\boldsymbol{I}_L$ 中，并计算其与对应的待匹配点 $p_i(i=1,2,\cdots,k_m\times k_n)$的误差 ε_i。如果 $\varepsilon_i<\varepsilon$，则认为其属于集合 S'，S'的元素个数加 1。

步骤 4　查找内点数最多的模型 S^*，并利用最小二乘法求解 $\boldsymbol{H}$。

1)查找 $S_i'(i=1,2,\cdots,T)$中内点数最多的变换模型，得到 S^*；

2)用 S^* 内的所有点进行最小二乘拟合，重新计算变换模型，得到最终的射影变换矩阵 $\boldsymbol{H}$。

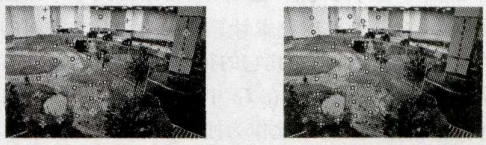
6　实验结果与分析

本文算法实现的硬件平台为 Hp Z800 工作站，CPU 为 Intel(R) X5550 2.67GHz，内存 3.48GB，显卡为 NVIDIA Quadro FX4800。

采用 NCC 算子进行匹配窗口之间的相似性度量，对两幅 600×400 的图像进行匹配。在不同的匹配窗口下，用 GPU 实现的未采用自适应匹配搜索区域算法在整个右图像中进行搜索匹配的结果，以及采用自适应匹配搜索区域算法进行图像匹配的结果对比如表 2 所列。图 4 给出了在匹配窗口大小为 11×11 时两种算法的匹配结果，其中红色的点为用 RANSAC 算法计算得到的一致点。可以看出，自适应匹配搜索区域算法能够显著地提高图像匹配的计算效率和正确率。

表 2　自适应搜索区域匹配和全图搜索匹配结果对比

匹配窗口大小	全图搜索匹配		自适应搜索区域匹配	
	实现时间(ms)	匹配正确率	实现时间(ms)	匹配正确率
5×5	63	20.0%	31	58.3%
11×11	156	37.5%	47	90.0%
15×15	265	65.0%	64	85.0%
21×21	484	62.5%	72	88.3%
31×31	984	52.5%	125	86.7%



(a) 全图搜索匹配结果　　(b) 自适应匹配搜索区域匹配结果

图 4　自适应搜索区域匹配和全图搜索匹配结果对比

使用和未使用 GPU 进行并行化加速的图像拼接计算效率对比如表 3 所列。其中，加速比＝CPU 实现时间/GPU 实现时间。

表 3　不同匹配窗口下 CPU 和 GPU 的实现效率对比

匹配窗口大小	CPU 实现时间(ms)	GPU 实现时间(ms)	加速比	匹配正确率
5×5	491	31	15.8	58.3%
11×11	2851	47	60.7	90.0%
15×15	5554	64	86.8	85.0%
21×21	11173	72	155.2	88.3%
31×31	24210	125	193.7	86.7%

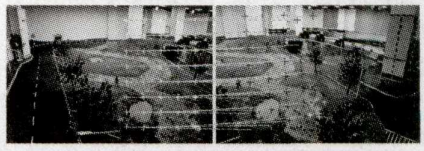


图 5　图像匹配结果

图 6　最终的拼接效果

从表 3 可以看出，无论是采用 CPU 还是 GPU，匹配窗口的大小对算法效率的影响都不可忽视，而且过大或过小的匹

配窗口都会使匹配正确率下降。在本文中，11×11 的匹配窗口匹配正确率最高，而且算法的实现效率也较优。

采用本文的方法所实现的待匹配点提取和匹配效果如图 5 所示，其中右图像中红色的点为用 RANSAC 算法计算得到的一致点。对 IR 进行射影变换得到的最终拼接效果如图 6 所示。

结束语 本文针对实时图像拼接的应用需求，提出了一种结合先验信息的自适应实时图像拼接算法。算法采用具有旋转不变性的 NCC 匹配方法，并利用 CUDA 对图像特征点匹配以及 RANSAC 图像射影变换模型求解算法进行了并行化。实验结果表明，自适应匹配搜索区域算法能够有效地提高图像匹配的效率和正确率；在采用 11×11 的匹配窗口时，图像匹配精度和最终的拼接效率都比较好；相比单独采用 CPU 而言，利用 GPU 进行并行化图像匹配的算法能够将图像匹配效率提高 60 倍以上，拼接速度可达 21.3fps。

本文所研究的算法主要是针对一般的图像拼接应用，即无需知道待拼接图像之间的相互关系。事实上，在实际应用中，有很多图像拼接应用都是可以事先得到更多的先验信息的。例如在地面移动机器人的遥控操作中，车载摄像机都是固定在机器人上，可以进行事先标定。因此，针对实际应用需求引入和利用已知的摄像机标定信息，增加图像匹配过程中的约束条件，进一步提高图像拼接的效率和精度，将是下一步的研究重点。

参 考 文 献

[1] 费树岷，陈启宏，宋爱国. 力觉临场遥感操作系统的研究进展[J]. 控制工程，2003，10(1)：11-14

[2] 李俊山，李新社，焦康. 并行图像处理[M]. 西安：西安交通大学出版社，2003：3-4

[3] Purcell T J，Buck I，Mark W R，et al. Ray Tracing on Programmable Graphics Hardware[J]. ACM Transactions on Graphics (Proceedings of SIGGRAPH)，2002，21(3)：703-712

[4] Zitova B，Flusser J. Image registration method：a survey[M]. Image and Vision Computing，2003：977-1000

[5] 张舒，褚艳利. GPU 高性能计算之 CUDA[M]. 北京：中国水利水电出版社，2009：9-20

[6] 王恺. 基于 GPU 的多摄像机全景视场拼接[D]. 长春：长春理工大学，2010

[7] 刘婷. 基于 GPU 的图像隐写分析实现[D]. 上海：华东理工大学，2011

[8] Manavski S A. CUDA compatible GPU as an efficent hardware accelerate for AES cryptography[C]// IEEE International Conference on Signal Processing and Communications. 2007：65-68

[9] 袁修国，彭国华，王琳. 基于 GPU 的变型 SIFT 算子实时图像配准[J]. 计算机科学，2011，38(3)：300-303

[10] 宋宝森，付永庆，宋海亮. 一种消除图像拼接痕迹的新方法[J]. 计算机科学，2011，38(2)：260-263，292

[11] Fischler M A，Bolles R C. Random Sample Consensus：A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography [J]. Communications of the ACM，1981，24(6)：381-395

(上接第 256 页)

[2] Kittler J，Illingworth J. Minimum Error Thresholding [J]. Pattern Recognition，1986，19(1)：41-47

[3] Pal N R，Pal S K. Image Model，Poisson Distribution and Object Extraction[J]. International Journal of Pattern Recognition and Artificial Intelligence，1991，5(3)：459-483

[4] Jiulun F，Weixin X. Minimum error thresholding：a note[J]. Pattern Recognition Letters，1997：18：705-709

[5] Jiulun F. Notes on Poisson distribution-based minimum error thresholding[J]. Pattern Recognition Letters，1998，19：425-431

[6] 雷博，范九伦. 灰度图像的二维交叉熵阈值分割法[J]. 光子学报，2009，36(6)：1572-1576

[7] 范九伦，雷博. 灰度图像的二维交叉熵直线型阈值分割法[J]. 电子学报，2009，37(3)：476-480

[8] 范九伦，雷博. 灰度图像最小误差阈值分割法的二维推广[J]. 自动化学报，2009，35(4)：386-393

[9] Li C H，Lee C K. Minimum Cross Entropy Thresholding[J]. Pattern Recognition，1993，26(4)：617-625

[10] Brink A D，Pendock N E. Minimum cross-entropy threshold selection[J]. Pattern Recognition，1996，29(1)：179-189

[11] Pal N R. On minimum cross-entropy thresholding[J]. Pattern Recognition，1996，29(4)：575-580

[12] Chang C I，Chen K，Wang J，et al. A Relative Entropy-based Approach to Image Thresholding[J]. Pattern Recognition，1994，27(9)：1275-1289

[13] Lee S S，Horng S-J，Tsai H-R. Entropy Thresholding and Its Parallel Algorithm on the Reconfigurable Array of Processors with Wide Bus Networks[J]. IEEE Transactions on Image Processing，1999，8(9)：1229-1242

[14] Ramac L C，Varshney P K. Image Thresholding Based on Ali-Silvey Distance Measures[J]. Pattern Recognition，1997，30(7)：1161-1174

[15] Lee S K，Lo C S，Wang C M，et al. A Computer-aided Design Mammography Screening System for Detection and Classification of Microcalcifications[J]. International Journal of Medical Informatics，2000，60(1)：29-57

[16] Wang Jian-wei，Du Ying-zi，Chang C I. Relative Entropy-based Methods for Image Thresholding [J]. Circuits and Systems，2002，2：II-265-II-268

[17] Zhu Hui，Fu Z，Li Z. A New Image Thresholding Method Based on Relative Entropy[C]// Proc. of International Conf. on Communications，Circuits and Systems. 2002，1：634-637

[18] Chang C I，Du Y，Wang J，et al. Survey and Comparative Analysis of Entropy and Relative Entropy Thresholding Techniques [J]. Vision，Image and Signal Processing，IEEE Proceedings，2006，153(6)：837-850

[19] El-Feghi I，Adem N. Improved Co-occurrence Matrix as a Feature Space for Relative Entropy-based Image Thresholding [J]. Proceedings of the Computer Graphics，Imaging and Visualization. 10.1109/CGIV.2007，49：314-320

[20] 胡勇，赵春霞，郭志波，等. 一种基于相对熵阈值分割的改进算法[J]. 系统仿真学报，2009，21(12)：3731-3733

[21] 张弘，范九伦. 应用最小方差滤波的图像分割方法[J]. 计算机工程与应用，2009，45(13)：182-185

[22] Sahoo P K，Wilkins C，Yeage J. Threshold selection using Renyi's Entropy[J]. Pattern Recognition，1997，30(1)：71-84

[23] Mehmet Sezgin：blt_image_references[OL]. http://mehmetsezgin.net，2009