文章编号：1005-6734(2017)01-0057-06 doi: 10.13695/j.cnki.12-1222/o3.2017.01.012

基于三角形周长的暗星全天球自主快速识别

张同双[1,2]，周海渊[1]，钟德安[1,2]，郭敬明[1,2,3]

（1. 中国卫星海上测控部，江阴 214431；2. 飞行器海上测量与控制联合实验室，江阴 214431；
3. 中国科学院 长春光学精密机械与物理研究所，长春 130033）

摘要：随着星敏感器极限探测星等能力的提高，导航星暗星数目及识别特征库急剧增加，造成星图识别耗时长，误匹配率高。通过介绍传统的三角形星图识别算法原理及不足，结合主星识别算法思想，以主星对星角距和周长为识别特征，提出了一种基于三角形周长的暗星全天球自主快速识别算法。由于 9.0Mv 全天球导航特征库的构建过于繁琐，为了提高运算速度，采用 NVIDIA 图像处理器（GPU）并行计算架构，通过 CUDA 计算比 CPU 获得了 20 倍以上的加速。对特征库按周长构造了散列函数，分段存储成若干个子块。识别过程中，对观测三角形周长采用哈希查找法实现子块的快速定位，星角距的识别只在相应的子块内进行，提高了识别速度，增加了识别成功率。用实拍星图分别对基于星角距和周长特征的识别算法进行了验证，实验表明，两种算法均能实现 9Mv 星的识别，平均识别时间分别为 37.7123s 和 2.0422s，后者具有明显的优势，能够大大提高星敏感器的数据更新率。

关 键 词：星敏感器；星图识别；导航星；三角形周长；并行计算

中图分类号：U666.1 **文献标志码：**A

Fast all-sky autonomous dark star identification algorithm based on triangle perimeter

ZHANG Tong-shuang[1,2], ZHOU Hai-yuan[1], ZHONG De-an[1,2], GUO Jing-ming[1,2,3]

(1. China Satellite Maritime Tracking and Controlling Department, Jiangyin 214431, China;
2. Joint Laboratory of Ocean-based Flight Vehicle Measurement and Control, Jiangyin 214431, China;
3. Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Science, Changchun 130033, China)

Abstract: With the improvement of limiting magnitude detectivity of star sensor, the number of dark stars in guide stars and the recognition feature library are dramatically increased, which lead to more time-consuming and higher mismatching rate in star-map recognition. By understanding the principle and deficiencies of traditional triangle star map recognition algorithm, and combined with idea of the main star recognition algorithm, a fast all-sky autonomous dark star identification algorithm is proposed, which takes the main star-pair angular distance and triangle perimeter as the recognition features. Since 9.0Mv all-sky guide feature library is too cumbersome to develop, a NVIDIA Graphics Processor Unit (GPU) parallel computing architecture is adopted to improve the processing speed, which shows that the computation by CUDA is more than 20 times faster than that by CPU. A hash function is constructed for the feature library according to the triangle perimeter and is segmented into sub-blocks. In the process of star identification, the hash look-up method is applied for the triangle perimeter to realize the fast positioning of sub-blocks, while the recognition of star angular distance is only implemented within the corresponding sub-block, which improve the recognition speed and the recognition success rate. Finally, based on a series of real star maps, the experiments are made to verify the star recognition algorithm based on star angular distance and the one based on triangle perimeter, which show that two algorithms can both achieve 9.0Mv star recognition, while the average recognition time is 37.7123 s and 2.0422 s, respectively, which show that the algorithm based on triangle perimeter has obvious advantages, and can greatly improve the data update rate of the star sensor.

Key words: star sensor; star identification; guide star catalog; triangle perimeter; parallel computing

收稿日期：2016-10-30；**修回日期：**2017-01-20

作者简介：张同双（1968—），男，高级工程师，从事船姿船位测量技术。E-mail: zts_123@163.com

星敏感器是目前已知精度最高的载体姿态敏感器。星敏感器通过测量恒星指向来解算载体的姿态。星图识别算法是星敏感器的关键技术，快速性和鲁棒性一直是对其评价的重要指标[1-2]。典型的星图识别算法主要有三角形及其改进算法（主星算法、金字塔算法等）[3-5]和栅格算法[6]。星表尺寸较小时，这些算法具有很好的鲁棒性，而当星敏感器视场（Filed of View，FOV）较小、极限探测星等较高时，星表尺寸将急剧增加，相关算法的局限性将会显现，如：三角形算法特征维数低，导航星数量的增加将导致特征库的容量急剧增加，从而容易出现误匹配或冗余匹配；采用矢量内积作为特征值的金字塔算法，在小角距范围内区分度欠佳；栅格算法在星体数目较少而伪星较多时存在稳健性下降的问题；主星识别算法在视场中出现许多星等相近的亮星时，识别成功率将会严重降低[7]。

工程上有时为了追求更高的姿态测量精度，需要星敏感器具有较小的视场和更暗的星等探测能力，此时需要解决暗星情况下的星图识别问题。解决暗星星图识别算法有两种技术途径：一是局部天区辅助识别[8-9]；二是对现有算法的改进。

局部天区辅助识别利用辅助系统（如惯性导航系统）提供的相关测量信息确定星敏感器的粗指向，构建局部导航特征三角形，在此基础上采用现有星图识别算法进行识别。局部天区辅助识别的主要不足是识别过程严重依赖外部信息，丧失星敏感器的自主性。

算法改进方面，小视场暗星的星图识别方面文献不多。文献[10]对金字塔算法进行了改进，采用矢量外积作为匹配特征值，实现了 8Mv 极限星等的星图识别（视场 2.32°×1.73°）。该算法的不足是以两星的矢量外积作为匹配特征，每个三角形的识别仍需进行三次搜索。另外，为了减少特征库的容量，设定了一个最小临界星角距，在这一角距范围内只保留一颗最亮星体，因而可能会出现观测星不能识别的问题。

本文提出一种基于主星对的星角距和三角形周长为特征的星图识别算法：

首先，采用重叠球矩法将全天球划分成若干局部重叠天区，找出每个重叠天区中最亮的两颗星组成主星对，其余星作为辅星构建三角形周长特征库。本文选取的 9.0Mv 导航星库约有 12 万颗，建立三角形特征库异常繁琐耗时，为此采用了基于 NVIDIA 图像处理器和统一设备计算架构（Compute Unified Device Architecture, CUDA）的并行算法，大大提高了特征库的生成速度。

然后，对生成的特征库按三角形周长构造散列函数，进行分段存储。识别过程中根据观测三角形周长实现子块的快速定位，星角距的匹配只在相应的子块内进行，从而提高匹配速度。

最后对算法进行了验证，实现了 9.0Mv 星的快速星图识别。

1 基于星角距的三角形识别及其改进

三角形识别算法的基本特征量为三颗星两两间的星角距。假设恒星 i 、j 的赤经赤纬分别为(α_i,δ_i)和(α_j,δ_j)，对应的星敏感器像平面坐标分别为(x_i,y_i)和(x_j,y_j)，则恒星 i 、j 的参考星角距 d_{ij} 与观测星角距 l_{ij} 的计算公式分别为

$$d_{ij}=\arccos\left(\boldsymbol{V}_i\cdot\boldsymbol{V}_j\Big/\left(\|\boldsymbol{V}_i\|\times\|\boldsymbol{V}_j\|\right)\right) \tag{1}$$

$$l_{ij}=\arccos\left(\boldsymbol{W}_i\cdot\boldsymbol{W}_j\Big/\left(\|\boldsymbol{W}_i\|\times\|\boldsymbol{W}_j\|\right)\right) \tag{2}$$

式中：“$\cdot$”为矢量点积运算符；“$\|x\|$”为 x 的范数。

$\boldsymbol{V}_i$ 、$\boldsymbol{V}_j$ 分别为恒星 i 、j 的参考矢量：

$$\begin{cases}\boldsymbol{V}_i=\left[\cos\alpha_i\cos\delta_i\quad\sin\alpha_i\cos\delta_i\quad\sin\delta_i\right]^{\mathrm{T}}\\ \boldsymbol{V}_j=\left[\cos\alpha_j\cos\delta_j\quad\sin\alpha_j\cos\delta_j\quad\sin\delta_j\right]^{\mathrm{T}}\end{cases} \tag{3}$$

$\boldsymbol{W}_i$ 、$\boldsymbol{W}_j$ 分别为观测矢量，计算公式分别为

$$\begin{cases}\boldsymbol{W}_i=\left[-x_i\quad-y_i\quad f\right]^{\mathrm{T}}\Big/\sqrt{x_i^2+y_i^2+f^2}\\ \boldsymbol{W}_j=\left[-x_j\quad-y_j\quad f\right]^{\mathrm{T}}\Big/\sqrt{x_j^2+y_j^2+f^2}\end{cases} \tag{4}$$

其中，f 为星敏感器焦距。

基于星角距匹配的三角形识别算法就是根据下式判断观测星角距与参考星角距是否匹配：

$$|l_{ij}-d_{ij}|\leqslant\varepsilon \tag{5}$$

式中，ε 为星角距观测误差门限，$|x|$ 表示 x 的绝对值。

传统的三角形识别算法至少需要三次匹配才能确定参考三角形与观测三角形是否匹配，最大搜索量是特征库总数的三次方。而理论上，n 颗星可以组成 C_n^2 个星角距特征值，加上视场限制条件，特征值的数目也是相当可观的。三角形改进算法基本上均围绕减少特征值数量、引入新的特征值以及缩短搜索时间等方面展开。

在深入研究传统三角形算法的基础上，针对暗星识别难题，改进了基于星角距的三角形识别算法，并进行了试算。导航星库选用某天文台提供的基本星表，剔除大于 9Mv 的恒星、双星及变星后，星表容量为 119528 颗。目前常用的查找方法有散列查找法、线性查找法和二分查找法，最大搜索次数分别为 1 次、N 次及 $\log_2^N$ 次（N 为特征或特征子块总数），因而散列查找法具有明显的优势。基于散列查找的思想，将导航特征库按星角距大小分块存储和读取[2]，假设分块总数为 N ，则允许的星角距匹配误差为 $\sqrt{2}\,\mathrm{FOV}/N$ 。

读取时，子块编号 $H(d)$ 可采用哈希查找法进行定位：

$$H(d)=\mathrm{fix}\left(\frac{N\times d}{\sqrt{2}\,\mathrm{FOV}}\right)+1 \tag{6}$$

式中：d 为星角距（d_{ij} 或 l_{ij}）；$\mathrm{fix}(x)$ 表示 x 向零最近方向取整。

星图识别时，根据式(2)计算观测星角距，由于星点提取误差和恒星相对运动带来的位置不确定性，将 $d_{ij}\pm\varepsilon$ 代入散列函数(6)定位搜索相应的子块，可以大幅缩短搜索时间。$\varepsilon=3\sigma_d$，其中 σ_d 为星点位置误差均方差。具体算法可借鉴文献[11]的公共星号查找法。对每个观测三角形来说，改进的星图识别算法仍需进行三次子块定位和三次子块搜索，计算量为三次搜索次数的乘积，因而匹配速度较慢的问题未能得到根本改善。

2　基于三角形周长的星图识别算法

2.1　导航星特征库的构建

周长相等是三角形全等的必要条件，因此三角形识别时，可以基于散列查找的思想，首先搜索周长相等的三角形集合，然后在该集合中搜索两边匹配的三角形，从而可以减少匹配次数。为此可以结合主星算法[3]构建特征三角形：以视场中的最亮星与次亮星作为主星对（Main Star-pair），其余星作为辅星，分别与主星对构成特征三角形，具体数据结构为：

Struct Triangle
{Short　ID1；//主星星号
Short　ID2；//次星星号
Short　ID3；//辅星星号
float　Angledis12；//主星与次星间星角距
float　Angledis13；//主星与辅星间星角距
float　Perimeter；//三角形周长}；　　(7)

1）分区星表的构建

理论上 n 颗星可以构建 C_n^3 个特征三角形，即使加上双星和视场等约束条件，特征三角形的数量也是很庞大的，且存在很多冗余的特征三角形，此时构建三角形周长特征库将面临海量的计算。为减小构建特征三角形的计算量，可采用重叠球矩法将整个天球分割成若干个局部重叠天区，以减少视场跨天区对识别正确率的影响，如图 1 所示。

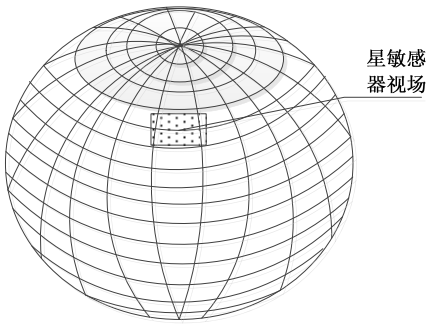

图 1　重叠球矩法对天球划分

Fig.1　Corresponding sub sky area of FOV of star sensor

球矩大小应满足如下条件：

$$\begin{cases}C_\alpha\in\left(\alpha_0-R/\cos\delta_0,\alpha_0+R/\cos\delta_0\right)\\ C_\delta\in\left(\delta_0-R,\delta_0+R\right)\end{cases} \tag{8}$$

式中，C_α、C_δ 分别为赤经范围与赤纬范围，α_0、δ_0 为星敏感器视轴指向（赤经、赤纬），R 为视场半径。考虑到赤经、赤纬的角度范围，计算球矩大小时需对 C_α 过零处理，赤纬最大值、最小值分别设为–90°和 90°。假设分区间隔取 4°，视场半径 R 取 4°，则建立分区星表时，以星敏感器视轴为中心（赤经从 0°起，赤纬从 –90°起），计算球矩大小，将球矩内的所有恒星按星等升序排列，并存储到以分区号为文件名的文本文件中。分区索引号的计算公式如式(9)所示，导航星表数据结构如式(10)所示：

$$\mathrm{Index}=\left(\mathrm{fix}\left(\frac{\alpha_0}{4}\right)+1\right)\times\left(\mathrm{fix}\left(\frac{\delta_0}{4}\right)+23\right) \tag{9}$$

Struct Sub Catalogue
{Short　ID；//星表号
float　Ra；//赤经，单位：度
float　Dec；//赤纬，单位：度
float　Magnitude；//星等，单位：Mv}；　　(10)

由此可知，分区星表中的第一颗星和第二颗星分别为主星和次星，其它恒星作为辅星，分别与主星、次星构建特征三角形，可极大减少特征三角形的数量。为了提高星图识别算法的适应性，还可将分区星表中较亮的恒星均作为主星对，以适当增加特征库的数量。

2）特征库的构建

读取分区星表，以选定的主星对分别与其它恒星构建特征三角形。读取相关星表号、赤经及赤纬等信息，则根据式(1)计算相关星角距 d_{ij}。

星敏感器光学系统一般采用离焦设计以提高星点细分定位精度，弥散斑大小一般控制在 3×3~5×5 像元之间，因此构建特征三角形时，星角距 d_{ij} 应满足如下条件：

$$\xi<d_{ij}\leqslant\sqrt{2}\,\mathrm{FOV} \tag{11}$$

式中，ξ 为弥散斑所对应的角度。若三角形的三条边均满足如上星角距限制条件，则按照导航特征数据结构存储该三角形。

3）周长散列特征库的构建

判断所有主星是否计算完毕，否则重复步骤 **2**），是则对所有特征三角形按照周长大小分块存储，子块编号按下式构造散列函数 $K(P)$：

$$K(P)=\mathrm{fix}\left(P\times N/(6\mathrm{FOV})+1\right) \tag{12}$$

式中，P 为三角形周长，N 为分块总数。

2.2 特征库的并行化计算

上述计算过程异常繁琐耗时，可以利用 CUDA 并行算法实现。CUDA 是 NVIDIA 推出的通用并行计算架构，可以使用标准 C 语言编写[12]。采用 CUDA 编写并行程序时，应用程序分为设备端（device）和主机端（host）代码两部分，主机端执行串行代码，并行部分代码以 kernel 形式在设备端运行。

如图 2 所示，每个核函数（kernel）对应一个网格 Grid（可以为三维），每个网格分为若干个线程块（Block），每个线程块对应若干个线程（Thread）。这样可以同时运行上万个线程，大大减少了运算时间。主机代码中读入导航星表信息，将数据通过 cudaMemcpy 复制到 GPU 设备内存中，指定参数 cudaMemcpyHostToDevice，在计算完成后，将结果通过 cudaMemcpyDeviceToHost 复制回主机，写入文件。

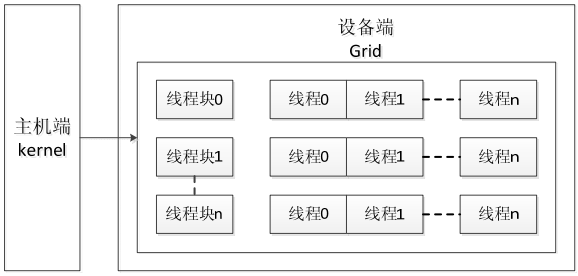

图 2 GPU 并行计算模型

Fig.2 The parallel computation model of GPU

导航特征库的构建可分成三个核函数完成：

1）划分小天区：由于导航星为 n 颗，创建 n 个线程，分成(n+127)/128 个线程块，每个线程块 128 个线程，每颗星对应一个线程，判断处于哪个小天区内，此时 n 为 119 528。

2）生成分区星表文件：对于 4°×4°的星敏感器，采用上述分区星表构建方法，共可构建 4050 个分区。创建个 115 478 个线程，分别为每个分区构建符合球矩要求的分区星表。

3）生成特征文件：根据每个分区星表，采用并行算法，构建满足星角距要求的分区导航特征库，具体如图 3 所示，依照式(7)存入核函数输出缓存，最后将结果复制回主机，写入文件。

以第 1 颗主星为例，主机程序读取次星表信息，次星数目为 n，分成(n+127)/128 个线程块，每个线程块 128 个线程，开辟 $n\times n$ 个式(7)结构体数组作为输出，建立 n 个线程，对应每颗次星：

tid=threadIdx.x+blockIdx.x*blockDim.x;

其中，threadIdx.x 为线程块中线程索引，blockIdx.x 为线程块的索引，blockDim.x 为线程块中线程数目，tid 为总的线程索引。每个线程计算主星、次星（tid）及其他辅助星（tid+1~n），计算两两星角距，判断是否符合限制条件，并按星等重新排序后，依照式(4)存入核函数输出缓存，最后将结果复制回主机，写入文件。

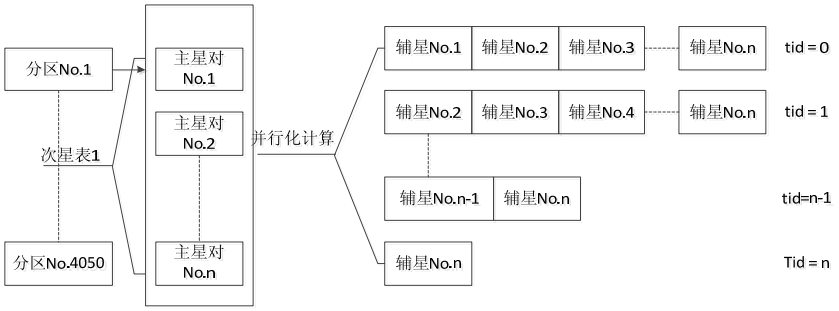

图 3 导航特征库建立

Fig.3 Established navigation feature database

2.3 星图识别

以上建立的三角形特征库为减少搜索量奠定了基础，一个三角形只需进行一次定位和一个分块搜索即可完成，具体算法流程如下：

1）初始化：读入导航星表、导航三角形特征库，设置星敏感器参数、星角距观测误差门限 ε 等参数；

2）对星图提取数据进行编号（星点号 i），并按灰度值降序排列；

3）判断是否存在 3 颗以上观测星，小于 3 颗则不能识别。

4）若观测星数为 3，则按灰度值确定主星对与辅助星，构建观测三角形，计算观测三角形的对应星角距和周长，根据式(12)定位子块编号，在该子块中搜索以主星为顶点的两个星角距是否存在唯一满足下式的三角形，是则将匹配结果存入识别矩阵 Identify 中，其中第 i 个元素表示星点 i 的星表号，识别失败时则该元素置 0；

$$\begin{cases}|l_{AB}-d_{AB}|\leqslant\varepsilon\\ |l_{AC}-d_{AC}|\leqslant\varepsilon\end{cases} \tag{13}$$

式中，A、B、C 分别为主星、次星及辅星。

5）若观测星大于 3 颗，则分别以星点 1、2、3 和星点 1、2、4 构建两个观测三角形，并按步骤 4）分别进行识别。若两个观测三角形存在唯一相同的主星对匹配结果，则匹配成功，将前 4 颗星的匹配结果存入识别矩阵 Identify 中，然后根据已匹配观测星对星图中其它恒星进行投影识别，并利用观测星角距对识别结果进行验证；若两个观测三角形没有唯一匹配结果，则剔除星点 3 或星点 4，读入下一颗星继续识别。

6）判断该帧星图是否识别完毕，否则转 5），是则根据识别矩阵 Identify 和导航星表确定观测星的星表号、赤经、赤纬及视星等等参数，转入下一帧星图识别，直至整个识别任务完毕。

3　试验结果分析

为分析星图识别算法性能，编制了相应的程序包，包括导航星库制定，三角形星图识别，基于三角形周长的星图识别算法等。试验设备：星敏感器，时统终端及 GPS，串口通信卡，NVIDIA GeForce GTX980，CPU Intel i3 处理器（主频 3.40GHz）及内存 4GB 的工控机；软件开发工具：Windows XP 操作系统，Matlab 2011a，Visual Studio 2012 及 CUDA 6.5。

选用某星敏感器样机拍摄了 2100 帧星图，先采用局部辅星图识别算法进行识别，作为识别结果的比对基准。所用星敏感器参数如表 1 所示。

表 1　星敏感器参数

Tab.1　Parameters of ship-borne star sensor

Parameter	Value
FOV/(°)	4×4
Pixel number	1024×1024
Focus/mm	198.9432
Star detectivity/Mv	9.0

选用某天文台的基本星表，剔除双星及高于 9Mv 的恒星后，星表尺寸为 119528 个，容量为 3.93MB，在此基础上，构建的三角形周长特征库总数为 6054793 条，文件大小 267MB。将其按周长大小等分为 512 个子块文件，子块特征数最多为 45936 条，最少为 1 条。

针对导航特征库建立，在 GPU 上测试其运行速度，并且与在 CPU 上运行速度进行比较。表 2 说明 3.2 中三个核函数均明显提高了运算速度，加速比均达 20 以上。

3.1　识别速度分析

识别速度方面，基于三角形周长的全天球自主识别算法计算量与观测星数、子块特征数等有关，假设参与计算恒星数量为 M 颗，则可以构建$(M–2)$个主星对观测三角形，每个三角形特征库子块有 n 个特征值，则需进行（M–2）次子块定位和 $n(M–2)$次子块搜索，而基于星角距的三角形全天球自主识别算法总的计算量为与观测星数、特征总数等有关，最大计算量为特征库总数的三次方，效率明显低于前者。图 4 为某帧星图二值化后的三维视图，表 3 是某帧识别结果，28 颗恒星全部识别。两种算法对所选星图序列所用平均计算时间分别 2.0422 s 和 37.7123 s，前者速度明显高于后者。

表 2　导航特征库建立时间对比

Tab.2　Contrast of guide feature library building time

Parameter	CPU	GPU	Acceleration
Sub-block segmentation	321.028 s	13.547 s	23.70
Main star-pair set up	545.385 s	17.868 s	30.52
Feature lib built	15130.01 s	365.824 s	41.35

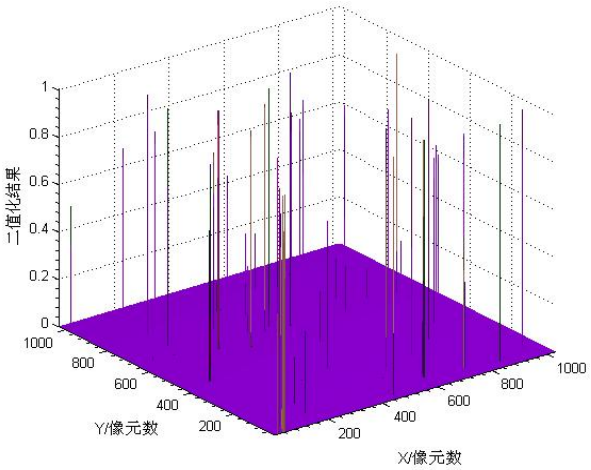

图 4　恒星提取结果

Fig.4　Result of star extraction

3.2　视星等对识别结果的影响

识别率方面，当主星对星等相差较大，且气象条件较好时，观测星均能较好地识别。由于隐含了视星等这个辅助识别特征，当主星对间星等差异较小时，由于气象因素的干扰，可能会出现主星、次星灰度值大小次序发生变化，从而导致部分恒星甚至整幅星图识别失败，为此对识别算法进行了优化了：假设该帧星图未识别，则将第 2 颗恒星作为主星，第 1 颗恒星作为次星，重复星图识别过程。这样可以提高星图识别成功率，但仍然有部分恒星识别不了，举例如表 3 所示，星点 1 的星等高于星点 2，但由于气象原因，星点 1 的灰度值反而高于星点 2。由表 3 可知，算法优化后，已提取的 20 颗恒星中，仍有 2 颗未识别（括号中的数字为该星点的正确星表号，0 表示未识别），本次试验数据处理中，有 184 帧星图中的观测星未能完全识别，占比 8.76%。极端情况下，当因气象原因及恒星光谱波段而导致主星不可探测时，整幅星图将不能识别。本次试验处理的 2100 帧星图数据中，这样的数据出现了 76 帧，占比达 3.62%。解决措施是，适当增加主星对的数量，可以一定程度上减小发生概率。

表 3 算法识别结果

Tab.3 Identification result of the algorithm proposed

Point Id	*X*/pixels	*Y*/pixels	Grey Value	Reference Star ID	Identification Star ID	Right Ascension/(°)	Declination/(°)	Magnitude/Mv
1	797.9158	411.3926	4590	117995	117995	192.306938	83.412904	5.288
2	459.3819	942.4562	1754	117901	117901	181.117138	85.587134	6.329
3	887.7991	191.0007	1503	118005	118005	193.555602	82.517724	7.246
4	323.2683	957.0255	1368	117854	117854	174.287347	85.617320	7.359
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
27	495.31580	810.0263	909	117912	117912	182.655292	85.078797	8.717
28	183.0000	961.53260	908	117812	117812	167.394565	85.544232	8.397

表 4 主星对星等对识别结果的影响

Tab.4 Influence of photometric magnitude on identification

Point Id	*X*/pixels	*Y*/pixels	Grey Value	Identification Star ID	Right Ascension/(°)	Declination/(°)	Magnitude/Mv
1	380.3725	548.0470	1381	64129	229.149170	9.712993	6.770
2	371.4902	192.8434	1324	80284	228.196548	10.702310	6.646
3	882.3679	226.3453	1418	64051	226.887283	9.226050	7.352
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
10	670.4881	115.9371	1106	（64088）	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
17	117.0000	849.7177	920	（80277）	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
20	800.0000	670.7857	881	64103	228.330841	8.245168	8.788

4 结束语

提出了基于三角形周长特征的暗星自主全天球识别算法，通过采用 NVIDIA GPU 架构编写 CUDA 并行算法，构建三角形周长和以主星为顶点的星角距特征库，大大减少了特征三角形的数量，对生成的特征库按三角形周长构造散列函数，进行分段存储，识别时采用哈希查找法对周长特征进行定位，相对传统基于星角距的三角形识别算法，减少了星角距比较次数，提高了识别速度。试验数据处理结果表明，针对 9.0Mv 恒星，基于周长和星角距的三角形识别算法平均时间分别为 2.0422s 和 37.7123s，识别速度得到了显著提升，识别率达 96.38%。该方法在小视场、暗星自主识别及提高星敏感器的数据更新率方面具有较高的工程应用价值。下一步将在优化特征库与算法适应性等方面进一步深入研究。

参考文献（References）：

[1] 张广军. 星图识别[M]. 北京: 国防工业出版社, 2011: 22-29.

[2] Kaplan L M. New robust star identification algorithm[J]. IEEE Transactions on Aerospace and Electronic Systems, 2006, 42: 1126-1131.

[3] Padgett C. Evaluation of star identification techniques[J]. Journal of Guidance, Control, and Dynamics, 1997, 20(2): 259-267.

[4] Accardo D, Rufino G. Brightness-independent start-up routine for star tracker[J]. IEEE Transactions on Aerospace and Electronic Systems, 2002, 38(3): 813-823.

[5] Lamy R G, Bostel J, Mazari B. Star recognition algorithm for APS star tracker: Oriented triangles[J]. IEEE Aerospace and Electronic Systems Magazine, 2005, 20(2): 27-31.

[6] Padgett C, Kreutz-Delgado K. A grid algorithm for star identification[J]. IEEE Transactions on Aerospace and Electronic Systems, 1997, 33(1): 202-213.

[7] 陆敬辉, 王宏力, 孙渊, 等. 基于 P 向量与三角形内切圆的星图识别算法[J]. 光学技术, 2011,37(1): 101-106.
Lu Jing-hui, Wang Hong-li, Sun Yuan, et al. Star identification algorithm based on P vector and triangle incircle[J]. Optical Technique, 2011, 37(1): 101-106.

[8] 江浩, 樊晓宁. 基于 FPGA 的局部天区星图识别技术[J]. 导弹与航天运载技术, 2013(1): 68-70.
Jiang Hao, Fan Xiao-ning. Local area star identification by FPGA[J]. Missiles and Space Vehicles, 2013(1): 68-70.

[9] 王宏力, 崔祥祥, 陆敬辉. 基于惯导姿态信息的高鲁棒性星图识别算法[J]. 中国惯性技术学报, 2010, 18(6): 729-732.
Wang Hong-li, Cui Xiang-xiang, Lu Jing-hui. Highly robust star identification algorithm based on attitude information of INS[J]. Journal of Chinese Inertial Technology, 2010, 18(6): 729-732.

[10] 李辉, 王安国, 张磊. 改进金字塔算法用于小视场星图识别[J]. 应用光学, 2013, 34(2): 267-272.
Li Hui, Wang An-guo, Zhang Lei. Modified pyramid algorithm for small FOV star image recognition[J]. Journal of Applied Optics, 2013, 34(2): 267-272.

[11] 李欣璐, 杨进华, 张刘, 等. 一种快速全天自主星图识别算法[J]. 长春理工大学学报(自然科学版), 2014, 37(3): 43-47.
Li Xin-lu, Yang Jin-hua, Zhang Liu et al. A fast all-sky autonomous star pattern identification algorithm[J]. Journal of Changchun University of Science and Techno- logy (Natural Science Edition), 2014, 37(3): 43-47.

[12] Sanders J, Kandrot E. CUDA by example: An introduction to general-purpose GPU programming[M]. Addison-Wesley Educational Publishers Inc, 2010.