(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 103856727 A
(43)申请公布日 2014.06.11

(21)申请号 201410111457.X

(22)申请日 2014.03.24

(71)申请人 北京工业大学
地址 100124 北京市朝阳区平乐园100号

(72)发明人 刘李纬 张银钱 肖创柏

(74)专利代理机构 北京思海天达知识产权代理有限公司 11203
代理人 楼艮基

(51)Int.Cl.
H04N 5/262(2006.01)
G06T 7/00(2006.01)

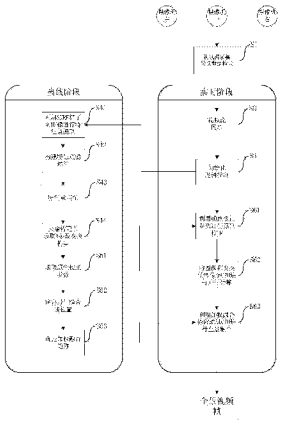
权利要求书3页 说明书18页 附图9页

(54)发明名称

一种多路实时视频拼接处理系统

(57)摘要

一种多路实时视频拼接处理系统属于实时视频图像处理领域，其特征在于在离线阶段求取当前场景下的多路视频图像之间的模型变换参数、颜色亮度伽马校正系数、最佳缝合线以及对应的加权融合矩阵，使得最终融合的全景视频图像在重叠区域平滑过渡；在实时阶段，直接利用离线阶段求解出来的伽马校正系数对图像的亮度值进行调整，在服务器中利用CUDA对多路实时视频流进行多线程并行投影变换计算和加权融合，生成实时全景视频图像。本发明的优点是在实时阶段直接利用离线阶段求得的相应参数，以及通过CUDA并行编程的实现，使得计算速度提高了至少5倍，而且相邻两两图像间的缝合线过渡带比传统方法更加平滑。



CN 103856727 A

1. 一种多路实时视频拼接处理系统，其特征在于，是一个带有支持 CUDA 的 NVIDIA 显卡的服务器，设有：视频采集单元（U1）、视频流同步单元（U2）、视频初始化逻辑控制单元（U3）、视频初始化配准单元（U4）、视频初始化融合单元（U5）、实时视频拼接单元（U6），其中：

视频采集单元（U1），是三路具有相同镜头、焦距的同款摄像机依次从左到右水平排开且固定位置，使采集的多路视频图像在水平方向有 30% 的重合度，并将原始数据从 YUV 格式转换为图像处理所需的 RGB 格式，向所述视频流同步单元（U2）传入三路视频图像：$I_1(x,y)$、$I_2(x,y)$ 和 $I_3(x,y)$，其中 $0 \leqslant y \leqslant H-1$，$0 \leqslant x \leqslant W-1$，H 为高度、W 为宽度，x、y、H、W 均为非负整数；

视频流同步单元（U2），设有：大小为 5 帧连续图像的三个缓冲区队列，依次缓存所述的三路视频图像流 $I_1(x,y)$、$I_2(x,y)$ 和 $I_3(x,y)$，采用先进先出 FIFO 的队列置换方式，利用操作系统生产者 - 消费者机制对所述三路实时视频流进行同步互斥控制，确保视频流全景图像的正确拼接；

视频初始化逻辑控制单元（U3），判断从所述视频流同步单元（U2）传入的三路同步视频图像的实时性：

若当前为离线阶段，则将三路图像送到视频初始化配准单元（U4），

若当前为实时阶段，则将三路图像送到实时视频拼接单元（U6）；

视频初始化配准单元（U4），利用 SURF 算子对当前的三路同步视频图像提取特征点，找到图像之间特征点的对应关系，再利用 RANSAC 算法求解图像间的空间变换模型，设有：特征点提取子单元（U41）、特征点匹配子单元（U43）和模型变换子单元（U44），其中：

特征点提取子单元（U41）以及特征点描述子单元（U42），利用 Herbert Bay 在“SURF:Speeded Up Robust Features”中提出的 SURF 算子进行特征点提取和描述，其中：

特征点提取子单元（U41），对所述的三幅同步视频图像 $I_1(x,y)$、$I_2(x,y)$ 和 $I_3(x,y)$ 利用 SURF 算法提取特征点：用不同尺寸的盒子滤波模板近似高斯二阶微分，构造尺度空间，并利用积分图像加速所述盒子滤波模板的卷积操作，在所述尺度空间进行非极大值抑制，得到特征点的位置 (x,y) 和尺度信息 s；

特征点描述子单元（U42），首先在以特征点为中心，以 6s 为半径的区域内，分别计算出 x 方向和 y 方向的 Haar 小波响应，其中 Haar 小波模板的大小为 4s；将一个 60 度的扇形作为滑动窗口，对窗口内的 Haar 响应值利用高斯权重进行累加；以 36 度为步长，旋转一圈，当 Haar 响应累加值最大时，对应的方向即为所求特征点的主方向；

以特征点为中心，沿着特征点的主方向，在 20s×20s 大小区域内，划分 4×4 个子区域，在每一个子区域中，计算 5*5=25 次 Haar 响应值，生成 4 维的 SURF 特征描述符 $v=(\sum dx, \sum |dx|, \sum dy, \sum |dy|)$，其中 dx 和 dy 分别为每个像素点经过 Haar 小波后得到在 x 方向和 y 方向的响应值；$\sum dx$ 和 $\sum dy$ 分别为对子区域内所有像素点在 x 方向上和 y 方向上，以特征点为中心进行高斯加权的累加响应值，最终得到 16×4=64 维的 SURF 特征点描述符；

特征点匹配子单元（U43），对相邻两幅图像 $[I_1(x,y), I_2(x,y)]$ 中检测到的特征点集合 P_1 和特征点集合 P_2 进行匹配，步骤如下：

a. 先建立带优先级的 KD 树索引，

b. 从特征点集合 P_1 中任意选取一点 p_1^i，在特征点集合 P_2 中找出其的最近邻点 p_2^j 和次近邻点 p_2^{j+1}，其中 $i \in \{1,2,\ldots,N_1\}$、$j \in \{1,2,\ldots,N_2\}$，N_1 和 N_2 分别为特征点集合 P_1 和 P_2 中特征点的个数，

c. 计算 p_1^i 到所述最近邻距点 p_2^j、p_1^i 到所述次邻距点 p_2^{j+1} 的距离的比值 $Ratio=\frac{dis(P_1^i,P_2^j)}{dis(P_1^i,P_2^{j+1})}$，若比值 Ratio 小于 0.8，则判断 p_1^i 与 p_2^j 是一对特征匹配点，分别记录其在特征点集合 P_1 和特征点集合 P_2 中的索引，

d. 重复步骤 b 和步骤 c，直到遍历完特征点集合 P_1 为止；

模型变换子单元(U44)，利用单应矩阵计算一个三维平面上的点在不同二维图像中的投影位置，通过 RANSAC 算法精确求出所述单应矩阵的 8 个参数，使得两组相邻两幅图像 $[I_1(x,y),I_2(x,y)]$ 和 $[I_2(x,y),I_3(x,y)]$ 分别得以配准；

视频初始化融合单元(U5)，包括颜色亮度校正子单元(U51)，最佳缝合线子单元(U52)以及加权融合子单元(U53)，以便对重叠区域中图像的颜色亮度和结构差异进行调整，其中：

颜色亮度校正子单元(U51)，步骤如下：

把所述相邻的两幅图像 $I_1(x,y)$、$I_2(x,y)$ 从 RBG 颜色空间转换到 l α β 颜色空间，分离亮度通道 l 和颜色通道 α β，对所述相邻两幅图像的重叠区域，

求出图像 $I_1(x,y)$ 在所述重叠区域部分的归一化亮度通道均值 $\bar{Y}_1$，以及图像 $I_2(x,y)$ 在所述重叠区域部分的归一化亮度均值 $\bar{Y}_2$，最终对应的伽马校正参数 $\gamma_1 \in (0,10)$ 和 $\gamma_2 \in (0,10)$ 通过以下最优化方程求解得到：

$$\min_{\gamma_1,\gamma_2} E=\frac{1}{2}\left(\frac{(\gamma_1 L_{1,2}-\gamma_2 L_{2,1})^2}{\sigma_N^2}+\frac{(1-\gamma_1)^2}{\sigma_g^2}+\frac{(1-\gamma_2)^2}{\sigma_g^2}\right)$$

其中 σ_N 为图像归一化灰度误差标准差和 σ_g 为伽马增益标准差，取值 σ_N=2.0/255，σ_g=0.5/255，$L_{1,2}=\ln\bar{Y}_1$，$L_{2,1}=\ln\bar{Y}_2$，$\bar{Y}_1=\frac{1}{N}\sum_{n=1}^{N}Y_{1,2}(p_n)$，$\bar{Y}_2=\frac{1}{N}\sum_{n=1}^{N}Y_{2,1}(p_n)$；

求解得到最终的颜色亮度伽马校正参数 γ'_1 和 γ'_2，在对原来的图像 $I_1(x,y)$、$I_2(x,y)$ 的亮度通道进行伽马变换，得到校正后的图像；

最佳缝合线子单元(U52)，在所述两幅图像的重叠区域，寻找一条缝合线，使得缝合线的两侧图像之间的颜色和结构差异最小，颜色差异用对应像素值之差进行度量、结构差异用梯度差进行度量，综合颜色和结构差异，用一个二维矩阵表示，从第一行随机选取 10 个像素点作为缝合线生长起始点，在最后一行选取值最小的那个像素点为缝合线终点；利用人工智能中的启发式 A* 搜索算法，分别计算出每个生长点对应的一条缝合线的平均累计误差值，选取平均累计误差值最小的线作为最佳缝合线，再在所述最佳缝合线的两侧分别选择一幅图像的重叠部分，进行全景图像的合成，

加权融合子单元(U53)，传入待合成全景的两幅图像 $I'_1(x,y)$ 和 $I'_2(x,y)$，分别建立一个二值图像表示初始化权重矩阵 $R_1(x,y)$ 和 $R_2(x,y)$，以所述最佳缝合线为边界，在其

两侧，$R_1(x,y)$ 的值分别为 1 和 0，$R_2(x,y)$ 的值分别为 0 和 1，分别对每个初始化权重矩阵用距离变换函数计算出对应初始化权重矩阵中所有非零像素点到与其相邻的最近的零像素点的街区距离，再通过一个设定的平滑过渡带区域大小参数 $\varepsilon \in (0,1]$ 和阈值 T=1，得到对应全景图像的所述两个相邻图像的归一化加权融合矩阵 $\alpha_1(x,y)$ 和 $\alpha_2(x,y)$；

实时视频拼接单元(U6)，对传入的两幅相邻的实时视频图像进行以下步骤来求出最终融合的图像：

利用所述颜色校正子单元(U51)在离线阶段计算出的最终伽马校正参数 γ'_1 和 γ'_2，直接对采集的实时视频图像在亮度通道进行颜色校正；

调用预置所述服务器内的基于并行编程模型指令集架构，直接利用离线阶段计算出的单应矩阵，通过实现 CUDA 的核函数 mapFunc<<<grid,block>>>(src,mapMatrix,dst)，在图像处理器 GPU 上实现多线程并发的图像变换计算，对相邻图像进行配准，确定图像的重叠区域；

用离线阶段求出的加权融合矩阵 $\alpha_1(x,y)$ 和 $\alpha_2(x,y)$ 对所述服务器得到的投影变换图像通过 CUDA 实现进行加权融合，从而得到在缝合线处更为平滑过渡的实时全景视频图像；

最后通过 cudaMemcpy2D 接口的 cudaMemcpyDeviceToHost 参数，将在 GPU 中计算得到的全景图像数据返回给 CPU，供界面显示。

一种多路实时视频拼接处理系统

技术领域

[0001] 本发明涉及图像处理领域，具体涉及一种多路实时视频拼接处理系统。

背景技术

[0002] 随着电子计算机技术的进步，计算机图像处理近年来得到飞跃的发展，已经成功的应用于几乎所有与成像有关的领域，并正发挥着相当重要的作用。人类传递的信息有70%是视觉信息，图像信息是传递信息的重要媒体和手段。单个摄像机所呈现的画面范围有限，不能较好的体现出全景动态范围。因此，为了更好地展现场景信息，在保证画面质量和实时性的前提下，提高视频的视野范围，是极其必要的。视频拼接技术具有广阔的应用前景，在城市交通、视频监控、智能车辆等计算机视觉领域都有着广泛的应用。

[0003] 视频拼接的本质仍然是图像的拼接。图像拼接的目的是形成一个视野更广的全景图像，即要求全景图像在拼接缝处颜色和结构上都能自然的过渡。视频拼接主要有两大挑战，一是视频图像的质量要求，这就需要有较好的图像拼接算法；其二是需要保证视频的实时性，这就需要并行计算架构，来提高算法的运行效率。针对第一个挑战，图像拼接主要由图像配准和图像融合两大部分组成，图像配准主要包括基于变换域的配准和基于特征的配准两大类，图像融合主要从颜色亮度和结构两方面来消除图像之间的差异，使得过度更为自然。对于第二个实时性的挑战，可以由FPGA嵌入式编程、英特尔公司的IPP、英伟达的CUDA并行计算架构等技术来解决。

[0004] 从图像采集角度，图像配准应用可以大致可以分为三类。1）多相机在不同视角对同一场景进行图像采集，对同一场景不同视角下的图像进行配准，得到更大的场景图像。2）不同时间获取同一场景的图像，利用配准找出场景的变化。3）利用不同传的感器获得同一场景的图像，例如同一场景的红外图像和自然光图像，目的是将不同数据源进行综合，得到更多的场景信息。根据本发明研究的实际问题，我们主要关注于第一类情况，即利用多路摄像机在不同视角下，对同一场景进行视频采集、拼接。

[0005] 图像配准方法主要有基于变换域和基于特征的两大类。基于变换域的方法主要有Kuglin在1975年提出的相位相关法，该方法利用了傅里叶变换的平移性质对图像进行配准，但该方法只适合于存在纯平移关系的两幅图像之间的像素级别配准，后人Sarvaiya等在其基础之上进行改进，通过对数极坐标变换，使得旋转和缩放转化为平移，从而使基本相位相关法扩展到具有平移、旋转和缩放关系的图像配准。由于傅里叶变换的基本性质，决定了该模型只适合于存在纯平移的配准，在仿射和透视变换模型中，该方法就不能成功配准图像。而实际过程中还很难做到相机位置以及其成像平面的绝对平行，一般成像平面都有一定的夹角，故需要采取新的办法。基于特征的匹配方法主要有Harris、SIFT和SURF等。Harris主要是通过微分算子计算窗口在各方向上的灰度变化，具有亮度不变性和旋转不变性，但对尺度变化比较敏感；SIFT特征算子具有尺度、旋转、光照不变性，同时对遮挡也具有较好的鲁棒性，准确率高但它的计算量较大；SURF算法是在SIFT思想的基础上，利用了盒子滤波和图像积分简化了计算复杂度，同时将特征描述子维度从SIFT的128维减少到64

维，这都在一定程度上加快了算法的执行速度。

[0006] 图像融合主要从颜色亮度和结构上两方面消除图像间的拼接缝。消除颜色亮度差异有 Reinhard 提出的颜色匹配模型，即利用颜色空间变换分离颜色和亮度通道，利用两幅图像间在不同通道的均值和标准差，进行尺度变换和平移变换，使得两幅图像具有相似的像素分布，从而使得图像相似；也有考虑像素的全局信息，利用重叠区域的直方图进行匹配，计算出一个颜色变换矩阵 M，从而对另外一个图像进行校正，使两副图像相似；或者在全局颜色变换基础上，利用高斯混合模型 GMM 对图像区域进行软分割，不同的区域对应不同的颜色变换，使得效果得到显著的提升，但是由于复杂的分割模型，使得算法在速度方面不适用于实时的视频拼接处理中。

[0007] 在颜色处理完毕后，仍然可能会有一定的结构上过度差异问题。Szeliski 提出了用羽化 (feathering) 的方法，根据距离对权重进行平均，虽然能够降低对比度但是也存在一定的问题，尤其是在配准阶段存在误匹配的话，即投影矩阵有误差，则羽化融合将会造成图像的模糊，即模糊效应和“鬼影”的问题仍然存在。对于图像拼接来说，在由于运动的物体出现的重影模糊，那么可以通过中值滤波来消除，但是在实时视频拼接中，这样的方法就不适用了，因为会导致视频中我们关心的一些运动的物体将被滤波器过滤掉。多频带融合法的主要思想是利用 Laplacian 金字塔分别构造图像的高频部分和低频部分，不同部分采用不同的融合策略。低频部分采用加权求和，起到模糊的效果；高频部分则利用最大权值的信息，保留边缘等变化的信息，最后将两部分组合起来，得到的融合效果令人满意。但是对于实时的视频流融合来说，这个算法的处理速度尚不能满足实时性要求。

[0008] 为了加速程序的运行速度，英伟达公司于 2006 年 11 月推出的一种基于并行编程模型和指令集架构的通用计算架构——CUDA。它可以让 GPU 与 CPU 协同工作，把一部分复杂的计算任务交给 GPU 进行并行处理。图像处理的本质是大规模矩阵运算，特别适合并行处理，GPU 在并行数据运算上具有强大的计算能力，具有很高的并发度，当执行具有高密度运算的多数据元素时，内存访问的延迟可以被忽略。在现有的视频实时拼接专利中，如张春雨的“一种基于多路摄像机的视频实时拼接方法”中，存在 3 个问题，一是对多路视频的同步没有详细的介绍；二是只是简单的通过投影映射进行配准，并没有对重叠处做过多的处理，导致视频融合质量不高；第三没有运用新型的并发编程架构，故实时性有一定的限制，有待于进一步提高。正是由于硬件的不断发展，高性能、新型的 CUDA 并行计算架构的出现，使本发明的实时视频拼接成为可能。

[0009] 关于图像拼接原理和方法的相关研究已有较多年的历史，也有不少论文发表，如卞春晓的“一种图像拼接处理系统”，虽然在拼接质量上能达到比较好的效果，但是运用在视频拼接上，其实时性就远远达不到要求。现在还没有通用的拼接效果较好的且能够达到实时处理要求的视频拼接系统。

发明内容

[0010] 有鉴于此，本发明提供了一种多路实时视频拼接处理系统，以解决现有视频拼接技术在保证拼接效果的条件下，不能达到视频拼接实时性要求的问题。

[0011] 一种多路实时视频拼接处理系统，其特征在于，是一个带有支持 CUDA 的 NVIDIA 显卡的服务器，设有：视频采集单元(U1)、视频流同步单元(U2)、视频初始化逻辑控制单

元(U3)、视频初始化配准单元(U4)、视频初始化融合单元(U5)、实时视频拼接单元(U6),其中:

[0012] 视频采集单元(U1),是三路具有相同镜头、焦距的同款摄像机依次从左到右水平排开且固定位置,使采集的多路视频图像在水平方向有30%的重合度,并将原始数据从YUV格式转换为图像处理所需的RGB格式,向所述视频流同步单元(U2)传入三路视频图像:$I_1(x,y)$、$I_2(x,y)$和$I_3(x,y)$,其中$0 \leqslant y \leqslant H-1$,$0 \leqslant x \leqslant W-1$,H为高度、W为宽度,x、y、H、W均为非负整数;

[0013] 视频流同步单元(U2),设有:大小为5帧连续图像的三个缓冲区队列,依次缓存所述的三路视频图像流$I_1(x,y)$、$I_2(x,y)$和$I_3(x,y)$,采用先进先出FIFO的队列置换方式,利用操作系统生产者-消费者机制对所述三路实时视频流进行同步互斥控制,确保视频流全景图像的正确拼接;

[0014] 视频初始化逻辑控制单元(U3),判断从所述视频流同步单元(U2)传入的三路同步视频图像的实时性:

[0015] 若当前为离线阶段,则将三路图像送到视频初始化配准单元(U4),

[0016] 若当前为实时阶段,则将三路图像送到实时视频拼接单元(U6);

[0017] 视频初始化配准单元(U4),利用SURF算子对当前的三路同步视频图像提取特征点,找到图像之间特征点的对应关系,再利用RANSAC算法求解图像间的空间变换模型,设有:特征点提取子单元(U41)、特征点匹配子单元(U43)和模型变换子单元(U44),其中:

[0018] 特征点提取子单元(U41)以及特征点描述子单元(U42),利用Herbert Bay在“SURF:Speeded Up Robust Features”中提出的SURF算子进行特征点提取和描述,其中:

[0019] 特征点提取子单元(U41),对所述的三幅同步视频图像$I_1(x,y)$、$I_2(x,y)$和$I_3(x,y)$利用SURF算法提取特征点:用不同尺寸的盒子滤波模板近似高斯二阶微分,构造尺度空间,并利用积分图像加速所述盒子滤波模板的卷积操作,在所述尺度空间进行非极大值抑制,得到特征点的位置(x,y)和尺度信息s;

[0020] 特征点描述子单元(U42),首先在以特征点为中心,以6s为半径的区域内,分别计算出x方向和y方向的Haar小波响应,其中Haar小波模板的大小为4s;将一个60度的扇形作为滑动窗口,对窗口内的Haar响应值利用高斯权重w=2.5s进行累加;以36度为步长,旋转一圈,当Haar响应累加值最大时,对应的方向即为所求特征点的主方向;

[0021] 以特征点为中心,沿着特征点的主方向,在20s×20s大小区域内,划分4×4个子区域,在每一个子区域中,计算5×5=25次Haar响应值,生成4维的SURF特征描述符$v=(\sum dx, \sum |dx|, \sum dy, \sum |dy|)$,其中dx和dy分别为每个像素点经过Haar小波后得到在x方向和y方向的响应值;$\sum dx$和$\sum dy$分别为对子区域内所有像素点在x方向上和y方向上,以特征点为中心进行高斯加权($\sigma=3.3s$)的累加响应值,最终得到16×4=64维的SURF特征点描述符;

[0022] 特征点匹配子单元(U43),对相邻两幅图像$[I_1(x,y), I_2(x,y)]$中检测到的特征点集合P_1和特征点集合P_2进行匹配,步骤如下:

[0023] a.先建立带优先级的KD树索引,

[0024] b.从特征点集合P_1中任意选取一点p_1^i,在特征点集合P_2中找出其的最近邻点p_2^j

和次近邻点p_2^{j+1}，其中 i ∈ {1,2,...,N_1}、j ∈ {1,2,...,N_2}，N_1 和 N_2 分别为特征点集合 P_1 和 P_2 中特征点的个数，

[0025] c. 计算 p_1^i 到所述最近邻距点 p_2^j、p_1^i 到所述次邻距点 p_2^{j+1} 的距离的比值 $Ratio=\frac{dis(P_1^i,P_2^j)}{dis(P_1^i,P_2^{j+1})}$，若比值 Ratio 小于 0.8，则判断 p_1^i 与 p_2^j 是一对特征匹配点，分别记录其在特征点集合 P_1 和特征点集合 P_2 中的索引，

[0026] d. 重复步骤 b 和步骤 c，直到遍历完特征点集合 P_1 为止；

[0027] 模型变换子单元(U44)，利用单应矩阵计算一个三维平面上的点在不同二维图像中的投影位置，通过 RANSAC 算法精确求出所述单应矩阵的 8 个参数，使得两组相邻两幅图像 [$I_1(x,y)$, $I_2(x,y)$] 和 [$I_2(x,y)$, $I_3(x,y)$] 分别得以配准；

[0028] 视频初始化融合单元(U5)，包括颜色亮度校正子单元(U51)，最佳缝合线子单元(U52)以及加权融合子单元(U53)，以便对重叠区域中图像的颜色亮度和结构差异进行调整，其中：

[0029] 颜色亮度校正子单元(U51)，步骤如下：

[0030] 把所述相邻的两幅图像 $I_1(x,y)$、$I_2(x,y)$ 从 RBG 颜色空间转换到 lαβ 颜色空间，分离亮度通道 l 和颜色通道 αβ，对所述相邻两幅图像的重叠区域，

[0031] 求出图像 $I_1(x,y)$ 在所述重叠区域部分的归一化亮度通道均值 $\bar{Y}_1$，以及图像 $I_2(x,y)$ 在所述重叠区域部分的归一化亮度均值 $\bar{Y}_2$，最终对应的伽马校正参数 $\gamma_1 \in (0,10)$ 和 $\gamma_2 \in (0,10)$ 通过以下最优化方程求解得到：

[0032] $$\min_{\gamma_1,\gamma_2} E=\frac{1}{2}\left(\frac{(\gamma_1 L_{1,2}-\gamma_2 L_{2,1})^2}{\sigma_N^2}+\frac{(1-\gamma_1)^2}{\sigma_g^2}+\frac{(1-\gamma_2)^2}{\sigma_g^2}\right)$$

[0033] 其中 σ_N 为图像归一化灰度误差标准差和 σ_g 为伽马增益标准差，取值 σ_N=2.0/255，σ_g=0.5/255，$L_{1,2}=\ln\bar{Y}_1$，$L_{2,1}=\ln\bar{Y}_2$，$\bar{Y}_1=\frac{1}{N}\sum_{n=1}^{N}Y_{1,2}(p_n)$，$\bar{Y}_2=\frac{1}{N}\sum_{n=1}^{N}Y_{2,1}(p_n)$；

[0034] 求解得到最终的颜色亮度伽马校正参数 γ'_1 和 γ'_2，在对原来的图像 $I_1(x,y)$、$I_2(x,y)$ 的亮度通道进行伽马变换，得到校正后的图像；

[0035] 最佳缝合线子单元(U52)，在所述两幅图像的重叠区域，寻找一条缝合线，使得缝合线的两侧图像之间的颜色和结构差异最小，颜色差异用对应像素值之差进行度量、结构差异用梯度差进行度量；综合颜色和结构差异，用一个二维矩阵表示，从第一行随机选取 10 个像素点作为缝合线生长起始点，在最后一行选取值最小的那个像素点为缝合线终点；利用启发式 A* 搜索算法，分别计算出每个生长点对应的一条缝合线的平均累计误差值，选取平均累计误差值最小的线作为最佳缝合线，在重叠区域内的所述最佳缝合线的两侧分别选择一幅图像的像素，进行全景图像的合成，

[0036] 加权融合子单元(U53)，传入待合成全景的两幅图像 $I'_1(x,y)$ 和 $I'_2(x,y)$，分别建立一个二值图像表示初始化权重矩阵 $R_1(x,y)$ 和 $R_2(x,y)$，以所述最佳缝合线为边界，

在其两侧，$R_1(x,y)$ 的值分别为 1 和 0，$R_2(x,y)$ 的值分别为 0 和 1，分别对每个初始化权重矩阵用距离变换函数计算出对应初始化权重矩阵中所有非零像素点到与其相邻的最近的零像素点的街区距离，再通过一个设定的平滑过渡带区域大小参数 $\varepsilon \in (0,1]$ 和阈值 T=1，得到对应全景图像的所述两个相邻图像的归一化加权融合矩阵 $\alpha_1(x,y)$ 和 $\alpha_2(x,y)$；

[0037] 实时视频拼接单元(U6)，在实时阶段对传入的两幅相邻的实时视频图像进行以下步骤来求出最终融合的图像：

[0038] 利用所述颜色校正子单元(U51)在离线阶段计算出的最终伽马校正参数 γ'_1 和 γ'_2，直接对采集的实时视频图像在亮度通道进行颜色校正；

[0039] 调用预置所述服务器内的基于并行编程模型指令集架构，直接利用离线阶段计算出的单应矩阵，通过实现 CUDA 的核函数 mapFunc<<<grid,block>>>(src,mapMatrix,dst)，在图像处理器 GPU 上实现多线程并发的投影变换计算，对相邻图像进行配准，确定图像的重叠区域；

[0040] 用离线阶段求出的加权融合矩阵 $\alpha_1(x,y)$ 和 $\alpha_2(x,y)$ 对所述服务器得到的投影变换图像进行加权融合，从而得到在缝合线处更为平滑过渡的实时全景视频图像；

[0041] 最后通过 cudaMemcpy2D 函数的 cudaMemcpyDeviceToHost 参数，将在 GPU 中计算得到的全景图像数据返回给 CPU，供界面显示。

[0042] 将变换模型算法用 CUDA 的并发机制实现，大大的加快了算法速度。通过对两路摄像机采集的 704*576 像素的图像进行实时配准，在 Win764 位操作系统、Intel Xeon3.60GHz、8G RAM、NVIDIA Quadro K600 显卡配置下，得到各阶段的实验数据，如下表。

[0043]

阶段	不使用 CUDA	利用 CUDA	备注
特征提取	2816ms	无	平均找到 1400 个特征点
特征匹配	2560ms	无	
投影变换	105ms	16ms	实时拼接阶段需对每个像素进行投影变换，需要全部时间
颜色校正	10ms	无	实时拼接阶段只需直接利用该参数对图像进行颜色校正
最佳融合线	23ms	无	该时间为离线阶段求取参数花费的时间，实时拼接阶段只需直接利用该参数，在加权融合矩阵中体现
加权融合	339ms	36ms	实时拼接阶段需对每个像素进行融合，需要全部时间

[0044] 其中实时拼接阶段只是简单的利用离线阶段计算出的空间投影变换单应矩阵 H、颜色亮度伽马校正参数 γ'_1、γ'_2 和加权融合矩阵 $\alpha_1(x,y)$ 和 $\alpha_2(x,y)$，由于投影变换和加权融合阶段需要对全景图像的每一个像素做相对复杂的运算，故每一帧需要投影变换阶段、加权融合阶段的全部时间和直接利用颜色亮度伽马校正参数 γ_1、γ_2 做颜色校正

的时间，即平均花费 16+10+36=62ms，达到了 16 帧 / 秒；而不利用 CUDA 并行架构的话，需要 105+10+339=454ms，只有不到 3 帧 /s，实验表明利用 CUDA 比单纯只用 CPU 计算要加速 16/3=5 倍以上。

[0045] 需要指出的是，本发明为了确保拼接的效果，利用了最佳融合线和加权融合的方法，而不是像现有视频拼接系统直接求取投影变换后就进行融合，虽然可以减少一定的时间，但是效果不太好，如图 12d 所示，黑色框所选区域内有明显的过渡带，如图 12e 所示本发明方法图像间的过渡效果要更好，而且帧率也保证在 15 ～ 20 帧之间。

[0046] 本发明将图像拼接算法作为实时视频拼接的重要基础，在此基础之上，利用操作系统多线程调度机制相关原理和 CUDA 并行计算架构，让实时的视频拼接成为可能。通过操作系统生产者 - 消费者模型和多线程机制，实现了实时视频流的采集和同步；通过经典的 SURF 特征提取算法，结合一定监控场合下的特定情况，对特定区域进行特征查找和利用带优先级的 KD 树索引机制，加速了特征提取匹配算法的速度；在颜色校正阶段，利用了伽马变换使得相邻图像间的颜色亮度整体一样；在寻找最佳缝合线时，构造了一个度量误差矩阵，使缝合线尽可能地穿越图像的平滑区域，同时利用了启发式搜索算法，加速了最优路径的寻找速度；最后在实时视频拼接中利用了 CUDA 并行计算架构来加速计算，保证了实时性的要求。

[0047] 与现有技术相比，本发明的有益效果是：本方法充分结合了 CPU 和 GPU 各自的优点，利用 CUDA 并行计算架构，构建两者协同工作的编程模型，并利用基于特征的图像配准、透视投影变换模型和基于最佳缝合线的加权融合法，最终实现又好又快的多路视频实时拼接。

附图说明

[0048] 为了更清楚地说明本发明的实施例和现有技术中的技术方案，下面将对实施例和现有技术描述中所需要使用的附图做简单地介绍，显而易见地，下面描述中的附图仅仅是本发明的一些实施例，对于本领域普通技术人员来讲，在不付出创造性劳动的前提下，还可以根据这些附图获得其他的附图。

[0049] 图 1 为本发明公开的多路实时视频拼接处理系统结构示意图；

[0050] 图 2 为本发明公开的视频采集单元的结构示意图；

[0051] 图 3 为本发明公开的视频流同步单元的结构示意图；

[0052] 图 4 为本发明公开的视频初始化配准单元的结构示意图；

[0053] 图 5 为本发明公开的视频初始化融合单元结构示意图；

[0054] 图 6 为本发明公开的实时视频拼接单元结构示意图；

[0055] 图 7 为本发明实施例一公开的多路实时视频拼接处理系统具体流程图；

[0056] 图 8 为本发明实施例一公开的 SURF 算子盒子滤波模板的示意图；(a)、(b)、(c) 分别为 9*9 大小的高斯二阶微分模板 $\frac{\partial^2}{\partial x^2}g(x,y,\sigma)$、$\frac{\partial^2}{\partial y^2}g(x,y,\sigma)$ 和 $\frac{\partial^2}{\partial x\partial y}g(x,y,\sigma)$；(d)、(e)、(f) 分别为与 (a)、(b)、(c) 对应的 9*9 大小的近似盒子滤波模板；

[0057] 图 9 为本发明实施例一公开的 x 方向 (a) 和 y 方向 (b) 的 Haar 模板示意图；

[0058] 图 10 为本发明实施例一公开的距离变换的示意图；(a) 为原二值图像，(b) 为距离

变换后的图像，(c) 为取 ε =0.2 的过渡权重图像，(d) 为取阈值 T=1 的最终归一化权重图像；

[0059] 图 11 为本发明公开的获取室内全景视频图像示意图，(a) 为摄像机左采集的视频图像，(b) 为摄像机右采集的视频图像，(c) 为基于最佳缝合线 (a) 的权重模板，(d) 为基于最佳缝合线 (b) 的权重模板，(e) 为最终融合的视频图像；

[0060] 图 12 为本发明实施例一获取的三路室外拼接全景视频图像示意图；(a) 为左路摄像机采集的视频图像，(b) 为中路摄像机采集的视频图像，(c) 为右路摄像机采集的视频图像，(d) 为简单方法前 2 路视频的融合图像，(e) 为本发明方法前 2 路视频的融合图像，(f) 为本发明方法 3 路视频的融合图像。

具体实施方式

[0061] 一种多路实时视频拼接处理系统，包括：

[0062] 视频采集单元，用于实时采集多路视频流，并将原始 YUV 数据格式转换为图像处理所需要的 RGB 格式；

[0063] 视频流同步单元，对实时采集的多路视频流，利用操作系统生产者 - 消费者机制对多路视频流进行同步控制，确保拼接的视频流全景图像不会出现错乱和断层；

[0064] 初始化逻辑控制单元，对视频拼接的逻辑进行控制，分为两个阶段：离线阶段和实时阶段。若当前为离线阶段，则将采集到的同步视频图像送至视频初始化配准单元（U4）和视频初始化融合单元（U5）进行处理；若当前为实时阶段，则直接将采集到的实时视频流送至实时视频拼接单元（U6）进行处理。

[0065] 视频初始化配准单元，利用 SURF 算子对当前的三路同步视频图像提取特征点，再利用匹配的特征点，在 RANSAC 算法的基础上，求解空间变换参数；

[0066] 视频初始化融合单元，对上述三路同步视频图像的重叠区域进行预处理，使重叠区域的颜色亮度尽可能地相似，从而使拼接后的图像在过渡处更完美；先在重叠区域内求出颜色结构差异度量矩阵，最后通过启发式路径搜索算法，选取累计误差最小的路径作为最佳缝合线；并在最佳缝合线的基础上，利用距离变换函数，建立加权融合矩阵，供实时拼接阶段加权融合。

[0067] 实时视频拼接单元，获得之前预处理阶段计算得到的图像投影变换模型、最佳缝合线加权融合矩阵和颜色亮度伽马校正系数，首先对图像进行颜色校正处理，再将待处理的同步实时视频帧图像从 CPU 传入 GPU，让 CUDA 并行架构进行图像变换，以及通过加权融合重矩阵对图像进行融合，生成全景视频帧图像，计算完毕后传回 CPU，供界面显示；

[0068] 优选的，所述视频初始化配准单元包括：

[0069] 特征点提取子单元，分别对多路同步实时视频图像利用 SURF 算子进行特征提取，得到感兴趣的特征点；

[0070] 特征点描述子单元，对上一步提取到的特征点进行描述，定义特征描述符，使得特征具有鲁棒性；

[0071] 特征点匹配子单元，通过最近邻算法对特征点进行匹配，计算两幅对应图像特征点之间的欧氏距离，利用最近邻距离与次近邻的比来确定匹配的特征点对；

[0072] 模型变换子单元，通过提取匹配得到对应的匹配点对，并利用 RANSAC 算法对其中

的错配点进行剔除，从而加强配准参数的准确性。

[0073] 优选的，所述视频初始化融合单元包括：

[0074] 颜色亮度校正子单元，分别计算两幅相邻图像重叠区域的平均亮度值，利用最优化方法求出伽马变换校正系数，使得两幅相邻图像的颜色亮度接近；

[0075] 最佳缝合线子单元，计算两幅相邻图像重叠区域的颜色结构差异度量，构造一个邻接矩阵，利用启发式搜索算法，求得累计误差最小的路径作为最佳缝合线；

[0076] 加权融合子单元，在所求得的最佳缝合线基础上，通过距离变换函数，分别为相邻的两个图像建立一个权重矩阵，最终通过该矩阵进行加权融合。

[0077] 优选的，所述实时视频拼接单元包括：

[0078] 实时颜色校正子单元，该单元主要是利用离线阶段所求的颜色亮度伽马校正参数对图像进行颜色亮度校正。

[0079] GPU 模型变换子单元，该单元主要是在离线阶段所求得的单应矩阵的基础上，对实时采集的视频图像利用 CUDA 并行计算架构进行投影变换，实现图像的快速配准；

[0080] GPU 加权融合子单元，该单元主要是在离线阶段求得的加权融合矩阵的基础上，对经过模型变换后的图像进行加权融合，得到最终的全景视频图像。

[0081] 下面将结合本发明实施例中的附图，对本发明实施例中的技术方案进行清晰、完整地描述，显然，所描述的实施例仅仅是本发明的一部分实施例，而不是全部的实施例。基于本发明中的实施例，本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其他实施例，都属于本发明保护的范围。

[0082] 本发明公开了一种多路实时视频拼接处理系统，以解决现有技术的算法存在的不能同时保证拼接效果和视频拼接实时性的问题，本发明能够保证较好拼接效果的同时，即在颜色亮度和结构上在重叠区域有较好的过渡，并且保证拼接视频的实时性。其结构图如图 1 所示，包括：视频采集单元 U1、视频流同步单元 U2、初始化逻辑控制单元 U3、视频初始化配准单元 U4、视频初始化融合单元 U5、实时视频拼接单元 U6，其中：

[0083] 视频采集单元 U1，如图 2 所示，三路摄像机左中右分别用于实时采集具有重叠区域的视频图像，由于原始采集到的视频图像数据是 YUV 格式，需要对其进行格式转换，转换为传统的 RGB 格式，为稍后的图像配准融合做准备，并标记为 $I_1(x,y)$、$I_2(x,y)$ 和 $I_3(x,y)$。其详细内容可参见下面所对应的实施例。

[0084] 视频流同步单元 U2，如图 3 所示，三路摄像机左中右将实时采集并转换后的 RGB 图像分别放入 3 个与之对一一应的缓冲区队列 1、缓冲区队列 2 和缓冲区队列 3，缓冲区队列的大小均设为 5 帧连续视频图像，为了使得采集的多路视频流同步，一旦缓冲队列满后，则将队首图像帧丢弃，使得实时采集的最新图像能够补充至缓冲队列。其详细内容可参见下面所对应的实施例。

[0085] 初始化逻辑控制单元 U3，对视频拼接的逻辑进行控制，分为两个阶段：离线阶段和实时阶段。若当前为离线阶段，则将采集到的同步视频图像送至视频初始化配准单元（U4）和视频初始化融合单元（U5）进行处理；若当前为实时阶段，则直接将采集到的实时视频流送至实时视频拼接单元（U6）进行处理。

[0086] 视频初始化配准单元 U4，如图 4 所示，通过利用 SURF 算子对视频图像进行特征点的提取，将检测出来的特征点构造对应的特征向量描述符，计算特征向量之间的欧氏距离，

利用最近邻距离与次近邻距离的比值作为判断是否匹配的标准，从而对两幅相邻图像进行配准；并利用 RANSAC 算法去除一些错误匹配的特征点对对变换参数的影响，最后求解出空间变换参数，从而确定两幅图像的重叠区域。其详细内容可参见下面所对应的实施例。

[0087] 视频初始化融合单元 U5，如图 5 所示，利用相邻图像的重叠区域求出颜色亮度伽马校正参数，以尽可能地消除图像拼接中颜色亮度差异造成的拼接缝；最佳缝合线子单元通过对重叠区域进行结构颜色差异度量，得到一个矩阵表示，在第一行随机选取 10 个生长点，利用启发式搜索算法，分别计算出每个生长点对应的一条拼接线的平均累计误差值，选取平均累计误差值最小线作为最佳缝合线。在最佳缝合线求出来之后，利用距离变换函数，求得最佳缝合线位置加权融合矩阵，在实时视频融合中，利用该权重矩阵进行加权融合，从而使图像之间过渡更为平缓。其详细内容可参见下面所对应的实施例。

[0088] 实时视频拼接单元 U6，如图 6 所示，对实时同步视频流进行颜色校正，这一阶段主要任务是将实时变换的图像送至 GPU，编写实现自己的核函数，让其在 CUDA 并行计算架构上进行并行运算，从而加速图像变换的速度；以及利用之前得到的加权融合矩阵，进行全景图像融合，最终得到实时的全景视频流。其详细内容可参见下面所对应的实施例。

[0089] 其具体实施方式如下所示：

[0090] 实施例一

[0091] 本实施例公开的实时视频拼接处理的流程如图 7 所示，包括：

[0092] 步骤 S1、将具有相同镜头和焦距的 3 路同款摄像机依次从左到右水平排开，固定位置，分别读入实时采集的 YUV 格式原始图像数据，其中摄像机左和摄像机中为两个相邻摄像机，其采集的图像中在水平方向大约有 30% 的重合度，摄像机中和摄像机右采集的图像在水平方向也有 30% 左右的重叠区域，同理，可以扩充至更多摄像机；

[0093] 采集的图像为 YUV 原始数据格式，为了后期图像处理需要将其转变为 RGB 数据格式。其转换公式为：

[0094] $$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & -0.00093 & 1.401687 \\ 1 & -0.3437 & -0.71417 \\ 1 & 1.77216 & 0.00099 \end{bmatrix} \begin{bmatrix} Y \\ U-128 \\ V-128 \end{bmatrix}$$

[0095] 其中 Y ∈ [0,255]、U ∈ [0,255]、V ∈ [0,255]，由于每一个 YUV 离散分量都有与之对应的 RGB 数值，故可以利用查表找对浮点型运算进行加速。设转换后的 RGB 图像为 $I_1(x,y)$、$I_2(x,y)$ 和 $I_3(x,y)$，其高为 H 个像素，宽为 W 个像素，(x,y) 表示二维空间的坐标，分别代表图像的行和列，其中，$0 \leqslant y \leqslant H-1$，$0 \leqslant x \leqslant W-1$，x、y、H、W 均为非负整数。

[0096] 步骤 S2、对采集的多路视频流进行同步，每一路视频流对应一个视频帧缓冲队列，其本质是利用操作系统的生产者 - 消费者模型，其中帧缓冲队列为临界资源，共有生产者线程和消费者两个线程，其读写同步通过互斥锁来实现。主要步骤如下：

[0097] 若某路摄像机采集转换后的视频图像到达计算机内存时，生产者线程获得互斥锁，生产者将其加入到对应的视频图像缓冲队列中，生产者线程释放互斥锁；若所有的视频图像缓冲队列中都含有至少一帧视频帧的时候，即所有缓冲队列都不为空，则首先消费者线程获得互斥锁，消费者将视频图像缓冲队列的队首视频图像取出来，用于下一步拼接成全景视频；若某一视频图像缓冲队列中的视频图像到达上限，即某一队列满了，那么则将队

首的那帧图像丢弃掉，以便让后来最新的图像能够及时进入缓冲队列。

[0098] 我们归纳出，这种采用视频图像缓冲队列的同步方案的异步时间差 T_{diff} 可以表示为：

[0099] $$T_{diff} = P_{scene} \frac{B_{size}}{F_c} N_c D$$

[0100] 其中：

[0101] B_{size} 表示缓冲区队列的大小，缓存的图像帧数越多，其不同步的现象就越明显。

[0102] F_c 代表摄像机的采集频率，即帧率，摄像机的帧率越低，其不同步的现象就越明显。

[0103] N_c 代表摄像的数量，同时采集的摄像机的路数越多，其不同步的现象就越明显。

[0104] D 代表网络延时，由于图像数据比较大，在视频采集传输线路和服务器接收视频流的时候，也可能会因为带宽受限和系统总线等原因产生时延，网络延时越大，其不同步的现象就越明显。

[0105] P_{scene} 代表视频流之间发生不同步的概率，它取决于多路摄像机所拍摄的场景的复杂度差异，如果两个摄像机所拍摄场景的复杂度有较明显的差异，视频流之间发生不同步的概率就越高，不同步的现象就越明显。

[0106] 在一般实验中，摄像机的帧率是采集初始化时确定的，采集端网络延时可以在局域网中控制或者通过模拟信号采集卡进行采集，而场景的相对复杂度往往是不可控制的，因此从理论上说，在摄像机数量固定的条件下，缓冲队列越小，采集的图像就表现得越同步。但是，缓冲队列设置得过小，比如 1，就意味着当场景复杂度突然发生较大变化时，比如复杂运动的物体突然进入场景时，采集帧率突然下降，缓冲区供应的图像比消费的更慢，即缓冲区队列有较大概率为空，这时采集的视频就会发生较明显的丢帧和滞后现象。这里我们通过多次实验，将视频帧缓冲队列的大小设为 5 较为合适。

[0107] 步骤 S3、对实时采集的视频流进行逻辑控制，分为两个阶段：离线阶段和实时阶段。若当前为离线阶段，则将采集到的同步视频图像通过步骤 S41-S44 和步骤 S51-S53 进行处理；若当前为实时阶段，则将采集到的实时视频流直接通过步骤 S61-S63 进行处理。

[0108] 为了说明简便，下面步骤均以两路图像 $I_1(x,y)$ 和 $I_2(x,y)$ 的配准融合进行说明，容易扩展到多路图像中去。

[0109] 在本实施例中，参见图 7，步骤 3 获取两幅相邻图像间的 8 参数变换模型具体以步骤 S41-S44 加以实现，包括：

[0110] 步骤 S41、拿到一幅图像后，需要提取出我们感兴趣的特征，从而能够用这些特征表示一幅图像，这就是对图像进行特征点提取。本发明采用 Herbert Bay 于 2006 年在“SURF:Speeded Up Robust Features”中提出的 SURF 算子对图像进行特征提取，由于系统的特定结构，已经知道图像的大概重合部分，故不需要对整幅图像进行特征提取和匹配，只需对图像的部分地方进行操作，节省了算法的运行时间，同时也在一定程度上提高了匹配的精确度。

[0111] SURF 算子由尺度空间表示，其特征点是由 Hessian 矩阵的行列式极值来检测的。在图像 $I(x,y)$ 中的一个点 $p(x,y)$，在尺度为 σ 的 Hessian 矩阵定义如下：

[0112] $$H(x,y,\sigma)=\begin{bmatrix} L_{xx}(x,y,\sigma) & L_{xy}(x,y,\sigma) \\ L_{xy}(x,y,\sigma) & L_{yy}(x,y,\sigma) \end{bmatrix}$$

[0113] 其中：$L_{xx}(x,y,\sigma)$、$L_{xy}(x,y,\sigma)$和$L_{yy}(x,y,\sigma)$分别是高斯滤波二阶偏导数$\frac{\partial^2}{\partial x^2}g(x,y,\sigma)$、$\frac{\partial^2}{\partial x\partial y}g(x,y,\sigma)$和$\frac{\partial^2}{\partial y^2}g(x,y,\sigma)$在点p(x,y)处与图像I(x,y)卷积的结果，其中二维高斯滤波函数为$g(x,y,\sigma)=\frac{1}{2\pi\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}}$。

[0114] 为了加快算法的速度，SURF算子采用盒子滤波模板，来近似高斯二阶微分，如图8所示，并使用积分图像来加速模板与图像卷积操作。

[0115] 从而Hessian矩阵的行列式可以得到：

[0116] $Det(H)=L_{xx}(x,y,\sigma)*L_{yy}(x,y,\sigma)-L_{xy}(x,y,\sigma)L_{xy}(x,y,\sigma)\approx D_{xx}D_{yy}-(0.9D_{xy})^2$

[0117] 其中D_{xx}、D_{yy}和D_{xy}分别是由图8中9*9大小的盒子滤波模板(d)、(e)和(f)与图像I(x,y)进行卷积而得，可以由积分图像将卷积运算转化为加减运算，并且计算时间的复杂度与模板尺寸无关。

[0118] SURF通过不断改变盒子滤波模板的尺寸，如9*9、15*15、21*21、27*27等，求取Hessian矩阵的行列式响应值，从而构建尺度金字塔。初始尺度空间层对应的模板尺寸大小为9*9，此时尺度值s=1.2，对应高斯滤波函数中σ=1.2；模板尺寸大小N与尺度值s成比例，$s=1.2\times\frac{N}{9}$，如27*27的盒子滤波模板其尺度s=1.2*27/9=3.6=σ；同样，SURF尺度空间划分了若干组(Octave)，每一组包括若干层(Layer)，由逐步变大的盒子滤波器模板，如9*9、15*15、21*21、27*27等，与同一图像I(x,y)卷积得到的响应图组成；不同组的之间的尺度有相互重叠、模板尺寸的间隔增量也在不断的翻倍，如第一组间隔为6，第二组间隔为12，第三组间隔为24等，故第二组模板尺度变化为15*15、27*27、39*39、51*51等，第三组尺度变化为27*27、51*51、75*75、99*99等，依次类推。一般情况下为3组，每组4层。

[0119] 随着同一组中的模板尺寸间隔增量不断变大，特征点的采样间隔也在变大，例如第一组(Octave)，每个像素点都计算Hessian矩阵的行列式响应值，到了第二组，隔一个点计算一次，第三组则隔2个点计算一次，成倍递增，依此类推。

[0120] 对于每一组(Octave)，尺度空间中的每一个盒子滤波器，与图像卷积，对计算出的Hessian矩阵行列式响应值设一个阈值T=300，大于该阈值的点为候选兴趣点。对候选兴趣点进行非极大值抑制：对于该层(Layer)的周围8个点以及上下相邻层对应位置的9*2个点，一共26个点比较行列式响应值的大小，若该点是周围26个点中行列式响应值最大的，则该点为所求的特征点。需要注意的是每一组的头尾两层是没法计算的。

[0121] 步骤S42、对提取的特征构造特征描述符，首先需要求得特征点的主方向，以特征点为中心，以6s为半径的区域内(其中s为当前特征点的尺度)，分别计算出x方向和y方向的Haar小波响应，其中Haar小波模板的大小为4s，如图9所示。然后以特征点为中心，将一个60度的扇形作为滑动窗口，用以下公式对窗口内的Haar响应值利用高斯权重(w=2.5s)进行累加，即离特征点近的Haar响应值权重大，离特征点远的Haar响应值权重小。

[0122] $$m_w = \sum_w dx + \sum_w dy$$

[0123] $$\theta_w = \arctan\left(\frac{\sum_w dx}{\sum_w dy}\right)$$

[0124] 其中 w 为高斯权重，dx、dy 分别为 x 方向和 y 方向的 Haar 小波响应值。

[0125] 以 36 度为步长，旋转一圈，当 Haar 响应累加值 m_w 最大时，对应的方向 θ_w 即为所求特征点的主方向。

[0126] 以特征点为中心，将坐标轴移至特征点主方向，在 20s×20s 大小区域内，划分 4×4 共 16 个子区域，在每一个子区域中计算 5*5=25 次 Haar 响应值，生成 4 维的 SURF 特征描述符 v=(Σ dx, Σ |dx|, Σ dy, Σ |dy|)，其中 dx 和 dy 分别为每个像素点经过 Haar 小波后得到在 x 方向和 y 方向的响应值；Σ dx 和Σ dy 分别为对子区域内所有像素点在 x 方向上和 y 方向上，以特征点为中心进行高斯加权（σ=3.3s）的累加响应值，将 16 个子区域的向量分别加入特征向量中形成 16*4=64 维的 SURF 特征向量描述符；

[0127] 步骤 S43、相邻两路摄像机采集的图像进行提取特征后，确定各自特征点的对应匹配关系。P_1 为在图像 $I_1(x,y)$ 中检测出的特征点的集合，P_2 为在图像 $I_2(x,y)$ 中检测到的特征点的集合，计算对应匹配点的步骤为：

[0128] 第一步，先建立带优先级的 KD 树索引，加快匹配点的搜索。

[0129] KD 树是一个二叉树，通常用于高维数据的索引。能够在每一维度将数据分为左右两部分，搜寻路径即从其中一条路径进行前进，直到叶子节点。但在求最近邻的时候，当查询点的领域与分割超平面两侧都有交集，则需要回溯检测两侧的特征，导致回溯过程过多，效率下降。故可以利用带优先级的 KD 树，其本质为在 KD 树的基础上利用了一个优先级队列，记录各自分割超平面与查询点的距离排序，距离越近，优先级越高，回溯检测总是从优先级高的节点开始。

[0130] 第二步，遍历集合 P_1，其中 p_1^i 为集合 P_1 中的任意一点，从另一集合 P_2 中找出 p_1^i 的最近邻 p_2^j 和次近邻点 p_2^{j+1}，其中 $i \in \{1,2,\ldots,N_1\}$、$j \in \{1,2,\ldots,N_2\}$，N_1 和 N_2 分别为集合 P_1 和 P_2 中特征点的个数，

[0131] 第三步，计算 p_1^i 的最近邻距离与次近邻距离比值 $Ratio = \frac{dis(P_1^i, P_2^j)}{dis(P_1^i, P_2^{j+1})}$，当比值 Ratio 小于 0.8 时，则认为 p_1^i 与 p_2^j 的特征点是一对匹配点，并分别记录其在集合 P_1 和 P_2 中的索引

[0132] 第四步，重复以上两步直到遍历完集合 P_1 为止；

[0133] 为了保证匹配的准确性，我们进行交叉验证。同理，

[0134] 第一步，遍历集合 P_2，其中 p_2^j 为集合 P_2 中的任意一点，依次从另一集合 P_1 中找出距 p_2^j 的最近邻点 P_1^i 和次近邻点 P_1^{i+1}，

[0135] 第二步，计算 p_2^j 的最近邻距离与次近邻距离比值 $Ratio=\frac{dis(P_2^j,P_1^i)}{dis(P_2^j,P_1^{i+1})}$，当比值 Ratio 小于 0.8 时，则认为 p_2^j 与 P_1^i 的特征点是一对匹配点，并分别记录其在集合 P_2 和 P_1 中的索引

[0136] 第三步，重复以上两步直到遍历完集合 P_2 为止；

[0137] 第四步，对比前后两次验证的索引对，找出公共部分的索引对，即为最终的特征点匹配对。

[0138] 步骤S44、设在步骤S43中得到的一对特征点匹配对为 $p_1^i=(x,y)$ 和 $p_2^j=(x',y')$。根据小孔成像原理，我们知道一个三维空间坐标点分别对应两个图像 $I_1(x,y)$ 和 $I_2(x,y)$ 中不同位置的像素点，那么它们存在一一对应关系。可以通过透视投影映射函数，利用一个 3*3 的单应(homography)矩阵 H，使得图像配准。单应矩阵用来计算同一个三维平面上的点在不同的二维图像中的投影位置的，是一个一对一的映射。其 8 参数矩阵表现形式为：

[0139] $$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \approx \begin{bmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \\ c_1 & c_2 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

[0140] 化简得到：

[0141] $$x'=\frac{a_1x+a_2y+a_3}{c_1x+c_2y+1}$$

[0142] $$y'=\frac{b_1x+b_2y+b_3}{c_1x+c_2y+1}$$

[0143] 其中 a_1、a_2、a_3、b_1、b_2、b_3、c_1、c_2 为所求的参数，共 8 个自由度，理论上至少需要 4 个对应的特征点匹配对，即可计算出两幅图像间的透视投影变换关系。

[0144] 由于我们得到的特征点匹配对比未知数的个数要多，这是一个超定方程组。但是经过交叉验证后的匹配点对中仍然可能有部分错误匹配的点对。下面利用 RANSAC 算法，求出精确的 8 参数变换模型。具体步骤为：

[0145] 第一步，从特征匹配点对集合中随机选取 4 组匹配点对，带入到上式映射函数中，求出变换参数，其中一幅图像的 4 点中不能有任意三点在一条直线上的情况，

[0146] 第二步，将特征匹配点对集合中剩余的匹配点对，利用第一步求出来的变换参数矩阵进行验证，若误差在一定阈值之内，则计为正确的匹配点对，个数加 1，

[0147] 第三步，直到遍历完特征匹配点对集合中的所有点对，统计出最终正确的匹配点对个数，记录下来，

[0148] 第四步，重复第一至第三步 30 次，选取正确的匹配点对个数最多所对应的变换参数矩阵为最终所求的 8 参数透视变换模型。

[0149] 在本实施例中，参见图 7，步骤 5 获取两幅相邻图像间的颜色校正参数、最佳缝合线及融合加权矩阵求取的具体以步骤 S51-S53 加以实现，包括：

[0150] 步骤 S51、求取颜色校正参数，多路摄像机采集的图像由于受不同角度光照和镜头工艺等因素的影响，会使得图像的颜色亮度产生一些差异，对重叠区域直接拼接将对全景

图像视觉效果有着严重的影响，使得拼接出来的全景不自然。

[0151] 在对图像进行加权融合之前，先对相邻图像的颜色进行预处理，尽可能的消除图像拼接中的颜色亮度缝隙。由于人对亮度的变化的敏感程度比对颜色变化的敏感度要高，本系统利用颜色空间变换，将 RBG 颜色空间转换到 l α β 颜色空间，分离了亮度通道和颜色通道，利用重叠区域的像素信息，对亮度通道做伽马变换使得两幅图像尽可能的相似。具体算法步骤为：

[0152] 第一步，获得左边图像 $I_1(x,y)$ 与中间图像 $I_2(x,y)$ 的重叠区域像素，并根据以下公式，进行颜色空间变换，

[0153] $$\begin{bmatrix} L \\ M \\ S \end{bmatrix} = \begin{bmatrix} 0.3811 & 0.5783 & 0.0405 \\ 0.1969 & 0.7244 & 0.0782 \\ 0.0241 & 0.1288 & 0.8444 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

[0154] $$\begin{bmatrix} l \\ \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{3}} & 0 & 0 \\ 0 & \frac{1}{\sqrt{6}} & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & -2 \\ 1 & -1 & 0 \end{bmatrix} \begin{bmatrix} \log L \\ \log M \\ \log S \end{bmatrix}$$

[0155] 第二步，初始操作，在左边图像 $I_1(x,y)$ 的重叠区域求出亮度通道均值 $\bar{Y}_1$，并用参数 γ_1 做伽马校正：

[0156] $$\bar{Y}_1 = \frac{1}{N}\sum_{n=1}^{N} Y_{1,2}(p_n)$$

[0157] $$U_{1,2} = \left(\bar{Y}_1\right)^{\gamma_1}$$

[0158] 同理，在中间图像 $I_2(x,y)$ 的重叠区域求出亮度通道均值 $\bar{Y}_2$，用参数 γ_2 做伽马校正：

[0159] $$\bar{Y}_2 = \frac{1}{N}\sum_{n=1}^{N} Y_{2,1}(p_n)$$

[0160] $$U_{2,1} = \left(\bar{Y}_2\right)^{\gamma_2}$$

[0161] 使得校正后的两幅图像尽可能的相似，

[0162] $$\left(\bar{Y}_1\right)^{\gamma_1} = \left(\bar{Y}_2\right)^{\gamma_2}$$

[0163] 其中 N 为重叠区域的像素个数，$Y_{1,2}(p_n)$、$Y_{2,1}(p_n)$ 分别为左边图像 $I_1(x,y)$ 和右边

图像 $I_2(x,y)$ 在重叠区域中第 n 个像素点的亮度通道 l 分量值，γ_1 和 γ_2 分别为要求解的左边图像 $I_1(x,y)$ 和中间图像 $I_2(x,y)$ 的伽马校正参数，。

[0164] 为了计算方便，对上式两边取对数

[0165] $$\gamma_1 \ln\left(\bar{Y}_1\right) = \gamma_2 \ln\left(\bar{Y}_2\right)$$

[0166] 简记，$L_{1,2} = \ln \bar{Y}_1$，$L_{2,1} = \ln \bar{Y}_2$

[0167] 第三步，根据前一步的亮度变换，转化为求以下最优化问题

[0168] $$\min_{\gamma_1,\gamma_2} E = \frac{1}{2}\left(\frac{\left(\gamma_1 L_{1,2} - \gamma_2 L_{2,1}\right)^2}{\sigma_N^2} + \frac{\left(1-\gamma_1\right)^2}{\sigma_g^2} + \frac{\left(1-\gamma_2\right)^2}{\sigma_g^2}\right)$$

[0169] 其中 σ_N^2、σ_g^2 分别表示图像的归一化灰度误差标准差和图像伽马增益标准差，这里我们分别取 σ_N=2.0/255，σ_g=0.5/255。

[0170] 第四步，将求出的伽马校正参数 γ_1 和 γ_2 分别对图像进行变换

[0171] $$I_1(x,y) \leftarrow I_1(x,y)^{\gamma_1}$$

[0172] $$I_2(x,y) \leftarrow I_2(x,y)^{\gamma_2}$$

[0173] 第五步，输出颜色亮度校正后的图像。

[0174] 步骤 S52、该方法的思想是在两幅图像的重叠部分，寻找一条缝合线，使得缝合线的两边图像之间的颜色差异和结构差异同时最小，从而在缝合线的两边只选一幅图像的像素进行合成全景图像。利用人工智能中的启发式 A* 算法搜索最优路径，得到最佳缝合线。

[0175] 第一步，从颜色差异来看，对所述两幅图像 $I_1(x,y)$ 和 $I_2(x,y)$ 的重叠区域做差，得到差图像 $D_c(x)$，即有

[0176] $D_c(x) = |I_1(x,y) - I_2(x,y)|$

[0177] 第二步，从结构差异来看，对相邻两幅图像 $I_1(x,y)$ 和 $I_2(x,y)$ 的重叠区域分别在 x 和 y 方向上求梯度，并构造梯度差异算子 $D_g(x)$，即有

[0178] $$D_g(x) = \left|\nabla_x I_1(x,y) - \nabla_x I_2(x,y)\right| * \left|\nabla_y I_1(x,y) - \nabla_y I_2(x,y)\right|$$

[0179] 其中，梯度可以通过分别对图像 $I_1(x,y)$ 和 $I_2(x,y)$ 的重叠区域利用 Sobel 算子 S_h 和 S_v 分别求取水平和垂直方向的梯度值，并记为 $\nabla_x I_1(x,y)$，$\nabla_y I_1(x,y)$ 和 $\nabla_x I_2(x,y)$，$\nabla_y I_2(x,y)$。

[0180] $$S_h = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \qquad S_v = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

[0181] 第三步，综合颜色和结构差异，得到 $D_t(x)=D_c(x)+D_g(x)$，该结果构成一个邻接矩阵。我们随机从第一行选取 10 像素作为个生长起始点，选择最后一行值最小的那个像素为终点；利用启发式 A* 搜索算法，分别计算出每个生长点对应的一条拼接线的平均累计误差

值，选取平均累计误差值最小线作为最佳缝合线，从而保证最佳缝合线要尽可能的穿越重叠区域平滑部分的原则。

[0182] 对于每一个生长起始点，最佳缝合线 A* 算法步骤如下：

[0183] 第一步，构造一个估计函数 = 从生长起始点到当前位置的实际值 + 当前位置到终点的估计值，即 f(n)=g(n)+h(n)，其中 g(n) 为从生长起始点到当前点的误差累加值，h(n) 为从当前节点到终点的估计误差值，这里定义为街区权重距离；规定扩展方向从右开始，顺时针顺序。

[0184] 第二步，创建一个 OPEN 表来存放未扩展节点，初始化时将生长起始点放入该表；创建一个 CLOSED 表来存放已经扩展的节点，初始化时该表为空

[0185] 第三步，若 OPEN 表为空，则查找失败，算法结束；

[0186] 在表 OPEN 表中找到使估计函数 f(n) 最小的节点 n，将它从 OPEN 表中移出，放入 CLOSED 表中。

[0187] 若节点 n 是终点，则最佳缝合线找到，通过父节点指针得到从生长起始点到终点的路径，算法结束；

[0188] 若节点 n 不是终点，则根据扩展规则产生它周围的相邻节点 ni，作为它的子节点，并将每个子节点 ni 的父节点指针指向 n，用于回溯。然后对于每一个子节点 ni，计算估计函数 f(ni)=g(ni)+h(ni)=g(n)+c(n,ni)+h(ni)，其中 c(n,ni) 为从节点 n 到 ni 的代价。有以下三种情况：

[0189] 1 若 ni 已经在 CLOSED 表中，则忽略此节点，

[0190] 2 若 ni 为新的节点，则将 ni 放入 OPEN 表中，

[0191] 3 若 ni 已经在 OPEN 表中，则比较其新老估计函数值，若 f(ni)<f(old)，说明从起始生长点经过 n 到 ni 的路径要比之前搜索得到的路径更短，用 ni 代替原来 OPEN 表中的节点，

[0192] 第四步，返回第三步。

[0193] 最终从 10 条缝合线中选择一条平均累计误差值最小的作为最佳缝合线。

[0194] 步骤 S53、在相邻图像 $I_1(x,y)$ 和 $I_2(x,y)$ 最佳缝合线的基础上，通过加权融合使得其在接缝处过渡得更平滑，主要有四步。

[0195] 第一步，对相邻图像 $I_1(x,y)$ 和 $I_2(x,y)$，分别建立一个二值图像表示初始化权重矩阵 $R_1(x,y)$ 和 $R_2(x,y)$，对于 $R_1(x,y)$ 在缝合线的两侧分别为 1 和 0，对于 $R_2(x,y)$ 在缝合线的两侧分别为 0 和 1，如图 11c、11d 所示，

[0196] 第二步，定义一个距离变换函数 D(p(x,y))，对 $R_1(x,y)$ 和 $R_2(x,y)$ 进行变换，

[0197] D(p(x,y))=min(dis(p,q))p ∈非零像素集、q ∈零像素集

[0198] 其中距离函数定义为街区距离 $dis(p(x_1,y_1),q(x_2,y_2))=|x_1-x_2|+|y_1-y_2|$。

[0199] 该距离变换函数 D(p(x,y)) 的本质就是计算初始化权重矩阵中所有非零像素点到与其相邻的最近的零像素点的距离，如图 10a、10b 所示。

[0200] 第三步，通过一个阈值 $\varepsilon \in (0,1]$ 来设定平滑过渡带的大小，分别计算出对应图像 $I_1(x,y)$ 和 $I_2(x,y)$ 新的过渡融合权重 $\alpha_1(x,y)$ 和 $\alpha_2(x,y)$，归一化 0 ～ 1 之间，如图 10c、10d 所示

[0201] $\alpha_1(x,y)=\varepsilon * R_1(x,y)$ if $\varepsilon * R_1(x_0,y_0)>1$，则 $\alpha_1(x_0,y_0)=1$

[0202] $\alpha_2(x,y)=\varepsilon*R_2(x,y)$ if $\varepsilon*R_2(x_0,y_0)>1$, 则 $\alpha_2(x_0,y_0)=1$

[0203] 第四步，由以下公式计算最终融合的图像，

[0204] $$I_{res}=\frac{\alpha_1(x,y)*I_1(x,y)+\alpha_2(x,y)*I_2(x,y)}{\alpha_1(x,y)+\alpha_2(x,y)}$$

[0205] 在本实施例中，参见图 7，步骤 5 对图像进行实时视频拼接的具体以步骤 S61-S63 加以实现，包括：

[0206] 步骤 S61、利用步骤 S51 求得的伽马校正参数 γ_1 和 γ_2，对图像进行颜色变换。

[0207] 步骤 S62、将图像传送至 GPU，调用自己实现的核函数，实现多线性并发计算，实时计算出投影变换后的图像。

[0208] 该步骤是本系统最为重要的一个环节，关系到能否成功实现实时视频拼接。考虑到本系统的特定应用场合，用于监控的摄像机相对位置基本固定不变，主要从以下两个方面，对算法速度进行加速，以实现实时拼接。

[0209] 第一，利用操作系统多线程调度原理，将本系统的工作分为两个线程，一个是离线拼接初始化线程，该线程主要负责特征点的提取、匹配、求取变换模型和颜色校正参数，由于这个过程需要的时间比较长，故不是对每一帧采集的图像组进行操作，而是过一定时间或等到用户发送指令后再进行。另外一个线程就是实时拼接线程，考虑到前期初始化的离线线程已经算出了配准阶段所需的图像之间的位置变换关系和融合阶段所需的颜色亮度校正系数，而且图像间的相对位置保存不变，故可以一次运算，多次利用。在实时拼接阶段只需要根据相应的空间变换模型、颜色亮度伽马校正参数和加权融合矩阵对图像进行处理计算即可，大大节省了运算时间。

[0210] 第二，在图像的配准阶段，本系统利用 S44 阶段求得的 8 参数投影变换模型，对相邻摄像机采集的图像进行配准。由于图像的变换主要是涉及到矩阵元素的加减乘除运算，这是 CUDA 并行计算架构的优势，实现算法主要由以下步骤构成：

[0211] 第一步，通过 CUDA 并行编程的 cudaMemcpy2D 接口和 cudaMemcpyHostToDevice 参数，将内存中的待配准的图像数据拷贝至 GPU 中，

[0212] 第二步，通过调用自己实现的基于 CUDA 架构的核函数 mapFunc<<<grid,block>>>(src,mapMatrix,dst)，在图形处理器 GPU 上实现多线程并发的图像变换计算。一个核函数是 CUDA 程序中的一个可被并行执行的步骤，其中 grid 为线程块的集合，表示 CUDA 在执行核函数时使用的并行线程块的数量；block 为线程的集合，表示一个线程块中包含线程的数量，故总的线程数量为 grid*block 个。src 为源图像，mapMatrix 为 S44 阶段求解的 8 参数投影变换模型矩阵，dst 为变换后的目标图像。

[0213] 步骤 S63、利用步骤 S53 计算出来的融合矩阵 $\alpha_1(x,y)$ 和 $\alpha_2(x,y)$，用以下公式通过 CUDA 实现，对投影变换后的图像进行加权融合，得到全景图像，

[0214] $$I_{res}=\frac{\alpha_1(x,y)*I_1(x,y)+\alpha_2(x,y)*I_2(x,y)}{\alpha_1(x,y)+\alpha_2(x,y)}$$

[0215] 最后，通过 cudaMemcpy2D 接口的 cudaMemcpyDeviceToHost 参数，实现将在 GPU 中的全景图像计算结果数据返回给 CPU，供界面显示。如图 11e 所示。

[0216] 本实施例利用了编程模型的多线程机制，离线线程处理复杂度较高的图像算法运算，在线线程负责实时拼接；利用多路摄像机对实时视频流进行采集；在操作系统消费

者-生产者和缓冲区队列临界资源互斥控制的基础上实现了多路视频流的同步;利用 SURF 算子对图像进行特征提取、描述符定义,进行交叉验证特征点匹配,结合 RANSAC 算法精确求解相邻图像间的 8 参数单应矩阵变换模型;通过空间变换将图像亮度和颜色通道分离,在人类更敏感的亮度通道做伽马校正,求得校正系数供实时拼接阶段使用;从结构和颜色两方面考虑,利用启发式搜寻算法,求出最佳缝合线以及对缝合线位置邻近进行距离函数变换,求得加权融合矩阵;通过 CUDA 并行计算架构对相邻图像进行模型变换和加权融合,较单纯利用 CPU 计算速度提高 5 倍以上,最终生成平滑过渡、无缝拼接的实时全景视频流,视频帧率达到 15 ～ 20 帧。

[0217] 上述实施例是在理论方面对本发明公开的一种多路实时视频拼接处理系统的详细地描述,同时在理论上也对其有益效果进行了描述。本发明也通过实验证明在拼接全景视频流效果上能够达到较好的结果,同时本发明公布的方法在实时性上也能取得满意的效果。

[0218] 本说明书中各个实施例采用递进的方式描述。专业人员还可以进一步意识到,结合本文中所公开的实施例描述的各个实例的单元及算法步骤,能够以电子硬件、计算机软件或者二者的结合来实现,为了清楚的说明硬件和软件的可互换性,在上述说明中已经按照功能一般性的描述了各示例的组成及步骤。这些功能究竟以硬件还是软件的方式来执行,取决于技术方案的特定应用和设计约束条件。专业人员可以对每个特定的应用来使用不同方法来实现所描述的功能,但是这种实现不应认为超出本发明的范围。

[0219] 结合本发明中所公开的实施例描述的方法或算法的步骤可以直接用硬件、处理器执行的软件模块,或者二者的结合来实施。软件模块可以置于随机存储器(RAM)、内存、制只读存储器(ROM)、电可编程 ROM、电可擦除可编程 ROM、寄存器、图形处理器 GPU、硬盘、可移动磁盘、CD-ROM、或技术领域内所公知的任意其他形式的存储介质中。

[0220] 对所公开的实施例的上述说明,使本领域专业技术人员能够实现或使用本发明。对这些实施例的多种修改对本领域的专业技术人员来说将是显而易见的,本文中所定义的一般原理可以在不脱离本发明的精神或范围的情况下,在其他实施例中实现。因此,本发明将不会被限制于本文所示的这些实施例,而是要符合于本文所公开的原理和新颖特点相一致的最宽的范围。

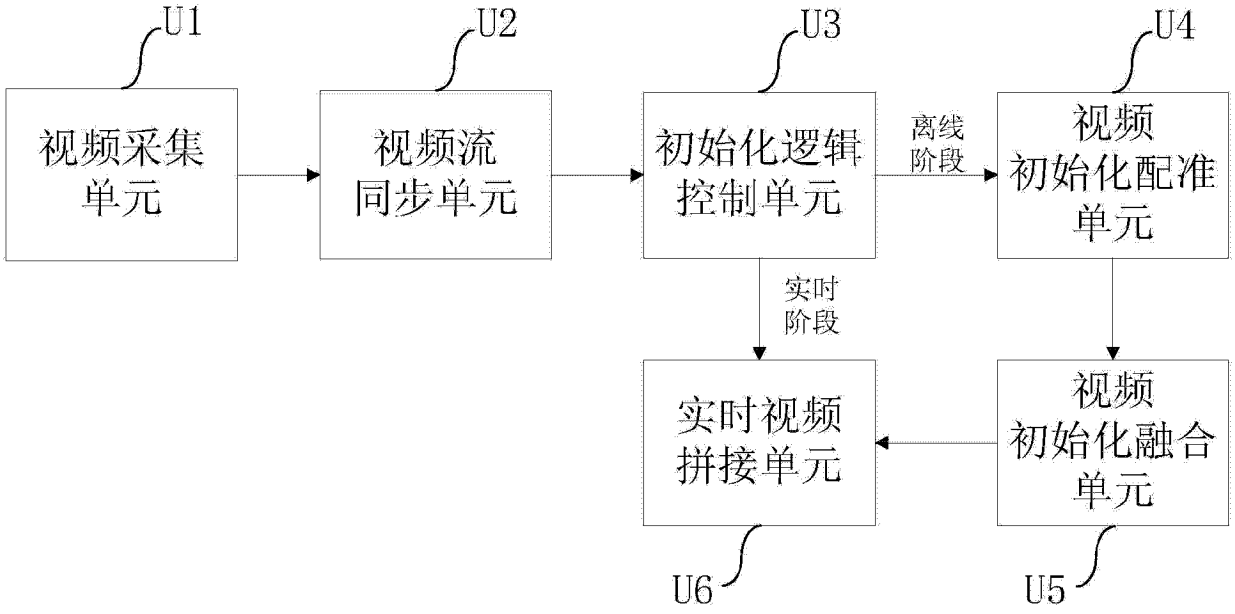

U1
视频采集单元
U2
视频流同步单元
U3
初始化逻辑控制单元
离线阶段
U4
视频初始化配准单元
实时阶段
实时视频拼接单元
U6
视频初始化融合单元
U5

图 1

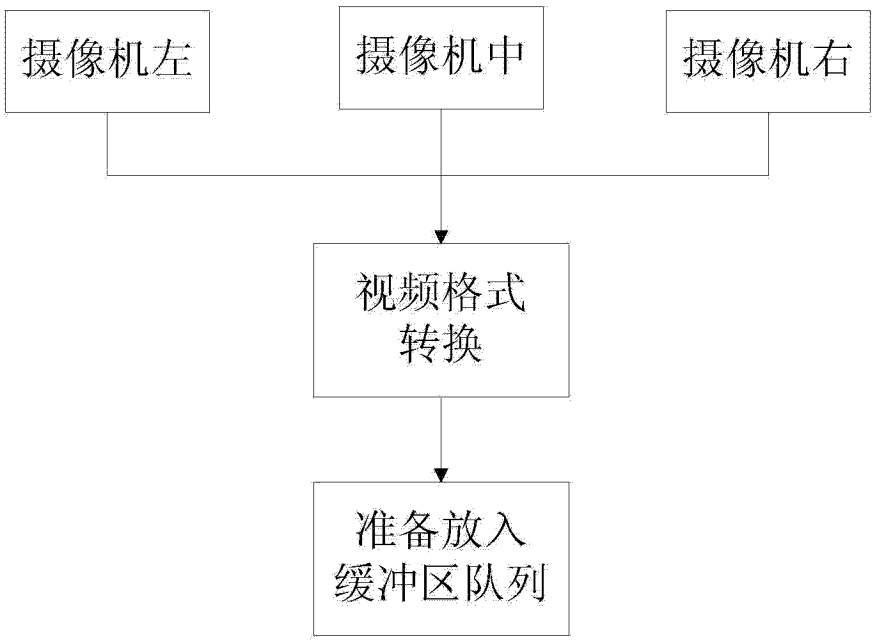

摄像机左
摄像机中
摄像机右
视频格式转换
准备放入缓冲区队列

图 2

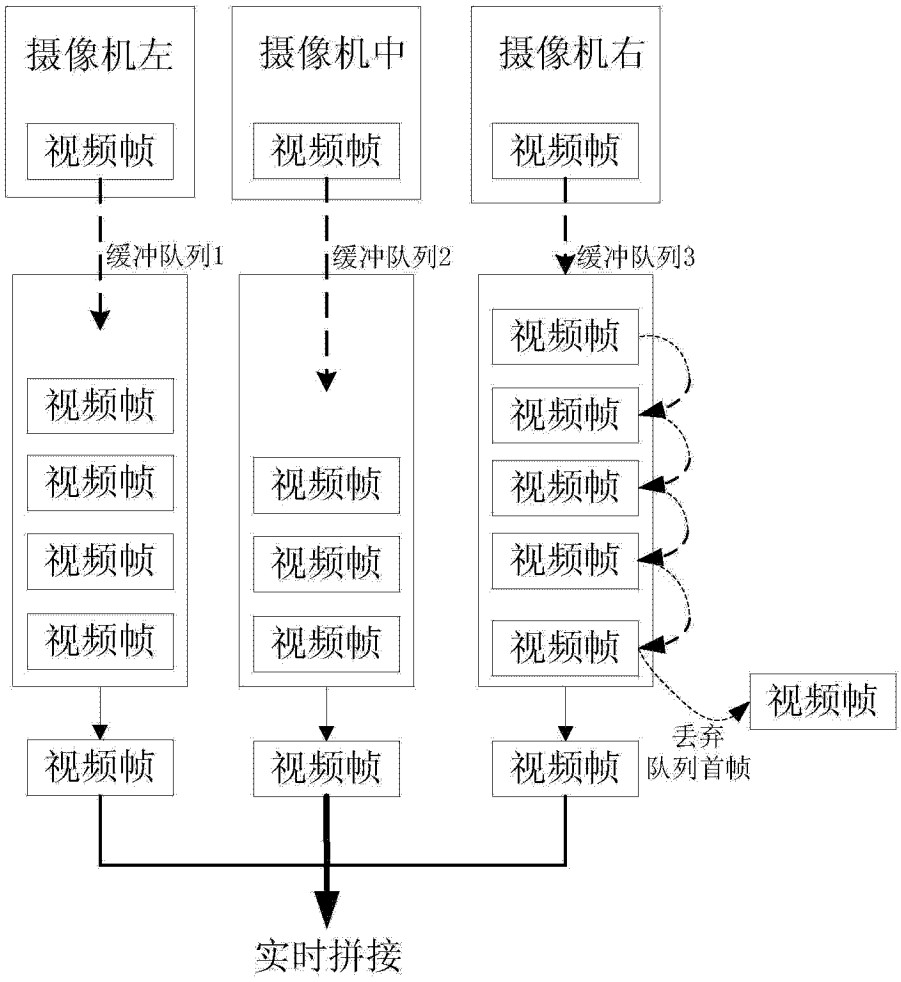

摄像机左
视频帧
摄像机中
视频帧
摄像机右
视频帧
缓冲队列1
缓冲队列2
缓冲队列3
视频帧
视频帧
视频帧
视频帧
视频帧
视频帧
视频帧
视频帧
视频帧
视频帧
视频帧
视频帧
视频帧
视频帧
丢弃
队列首帧
实时拼接

图 3

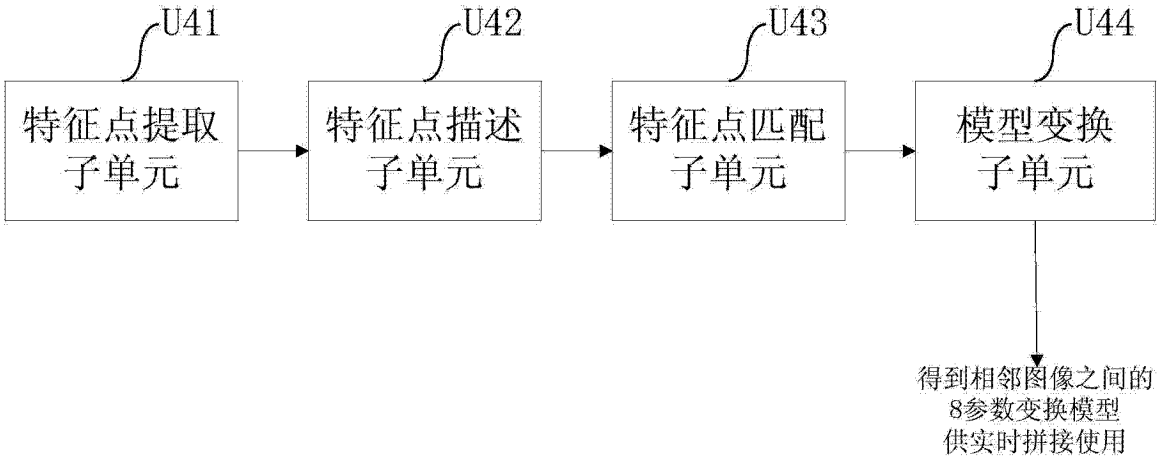

U41
特征点提取
子单元
U42
特征点描述
子单元
U43
特征点匹配
子单元
U44
模型变换
子单元
得到相邻图像之间的
8参数变换模型
供实时拼接使用

图 4

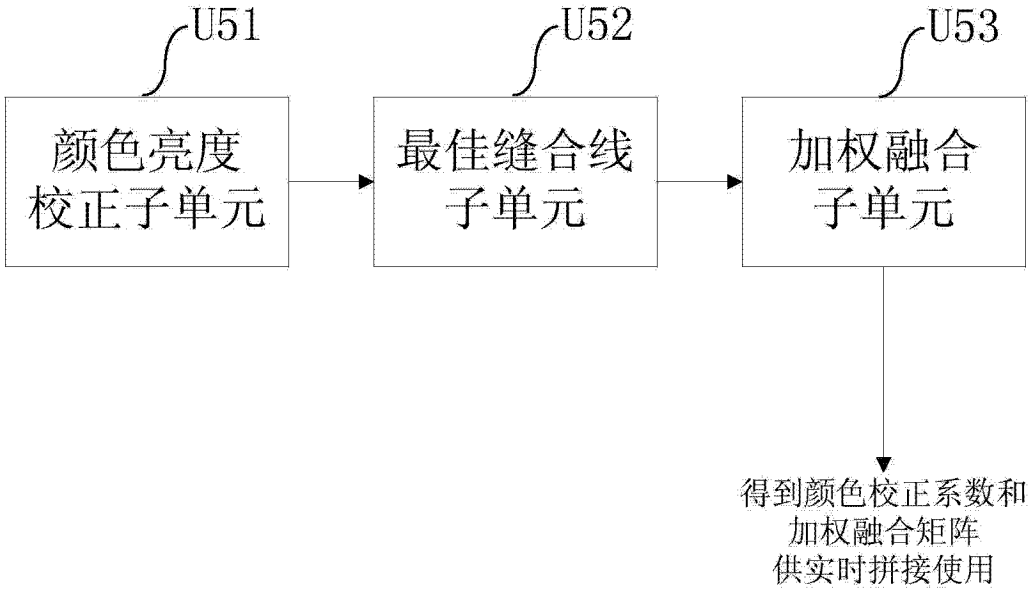

U51
U52
U53
颜色亮度
校正子单元
最佳缝合线
子单元
加权融合
子单元
得到颜色校正系数和
加权融合矩阵
供实时拼接使用

图 5

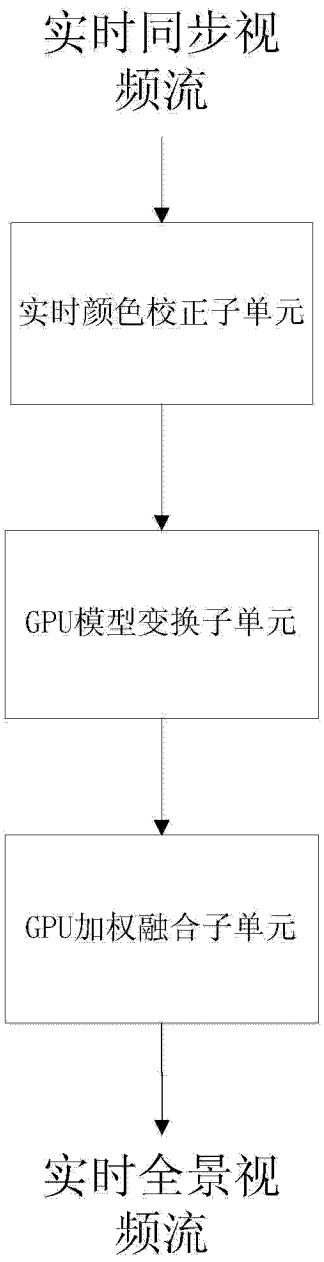


图 6

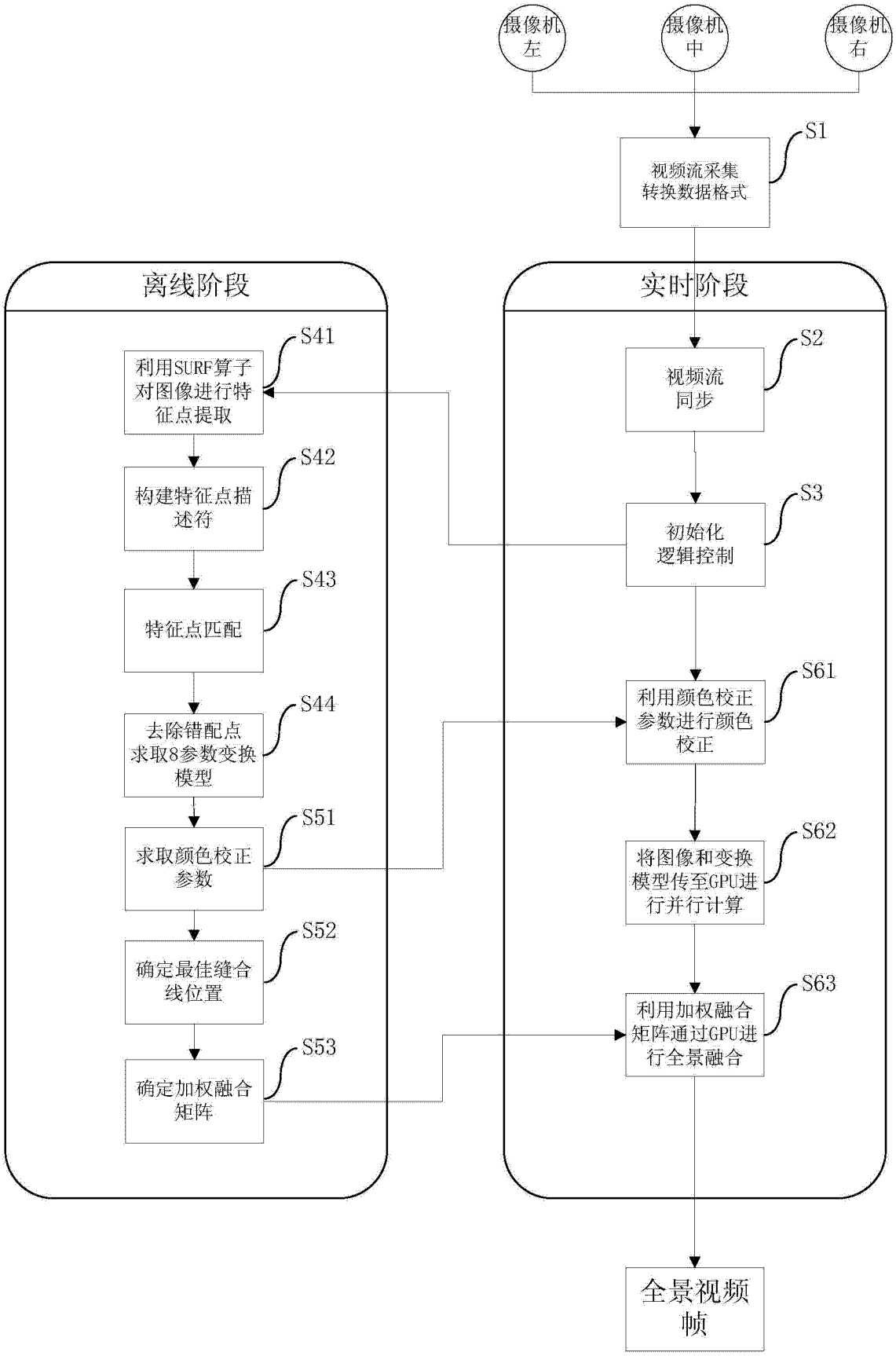

摄像机左
摄像机中
摄像机右
S1
视频流采集转换数据格式
离线阶段
实时阶段
S41
利用SURF算子对图像进行特征点提取
S42
构建特征点描述符
S43
特征点匹配
S44
去除错配点求取8参数变换模型
S51
求取颜色校正参数
S52
确定最佳缝合线位置
S53
确定加权融合矩阵
S2
视频流同步
S3
初始化逻辑控制
S61
利用颜色校正参数进行颜色校正
S62
将图像和变换模型传至GPU进行并行计算
S63
利用加权融合矩阵通过GPU进行全景融合
全景视频帧

图 7

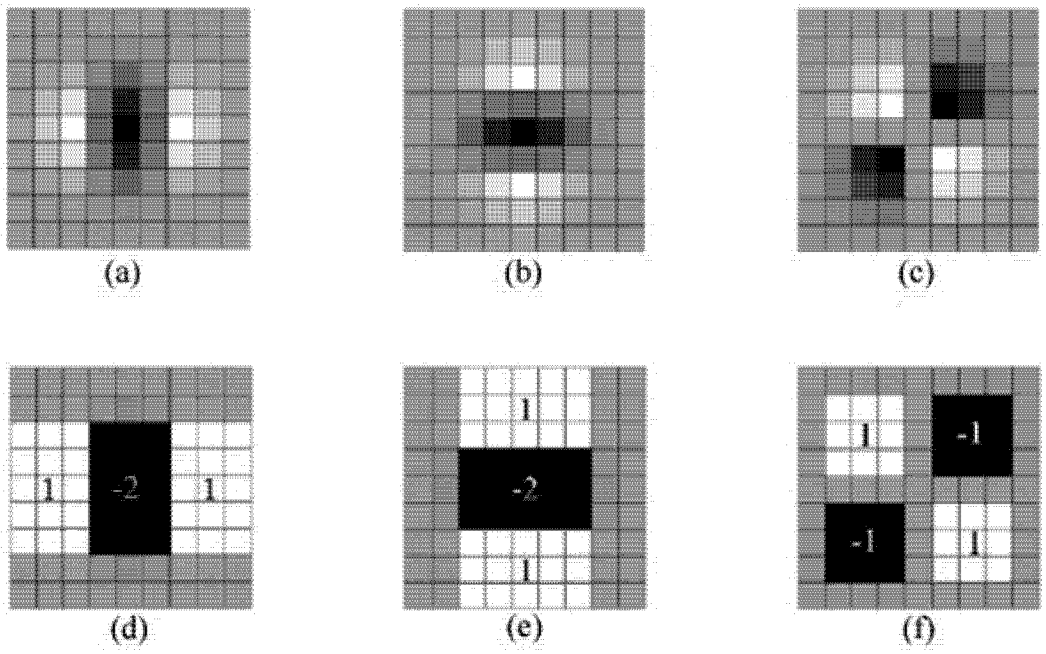


图 8

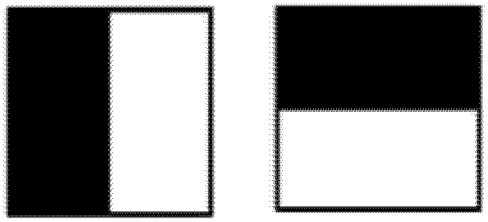


图 9

1	1	1	1	1	1	0	0	0	0
1	1	1	1	1	1	0	0	0	0
1	1	1	1	1	1	0	0	0	0
1	1	1	1	1	1	0	0	0	0
1	1	1	1	1	1	0	0	0	0
1	1	1	1	1	1	0	0	0	0
1	1	1	1	1	1	0	0	0	0
1	1	1	1	1	1	0	0	0	0
1	1	1	1	1	1	0	0	0	0

(a)

6	5	4	3	2	1	0	0	0	0
6	5	4	3	2	1	0	0	0	0
6	5	4	3	2	1	0	0	0	0
6	5	4	3	2	1	0	0	0	0
6	5	4	3	2	1	0	0	0	0
6	5	4	3	2	1	0	0	0	0
6	5	4	3	2	1	0	0	0	0
6	5	4	3	2	1	0	0	0	0
6	5	4	3	2	1	0	0	0	0

(b)

1.2	1.0	0.8	0.6	0.4	0.2	0	0	0	0
1.2	1.0	0.8	0.6	0.4	0.2	0	0	0	0
1.2	1.0	0.8	0.6	0.4	0.2	0	0	0	0
1.2	1.0	0.8	0.6	0.4	0.2	0	0	0	0
1.2	1.0	0.8	0.6	0.4	0.2	0	0	0	0
1.2	1.0	0.8	0.6	0.4	0.2	0	0	0	0
1.2	1.0	0.8	0.6	0.4	0.2	0	0	0	0
1.2	1.0	0.8	0.6	0.4	0.2	0	0	0	0
1.2	1.0	0.8	0.6	0.4	0.2	0	0	0	0

(c)

1.0	1.0	0.8	0.6	0.4	0.2	0	0	0	0
1.0	1.0	0.8	0.6	0.4	0.2	0	0	0	0
1.0	1.0	0.8	0.6	0.4	0.2	0	0	0	0
1.0	1.0	0.8	0.6	0.4	0.2	0	0	0	0
1.0	1.0	0.8	0.6	0.4	0.2	0	0	0	0
1.0	1.0	0.8	0.6	0.4	0.2	0	0	0	0
1.0	1.0	0.8	0.6	0.4	0.2	0	0	0	0
1.0	1.0	0.8	0.6	0.4	0.2	0	0	0	0
1.0	1.0	0.8	0.6	0.4	0.2	0	0	0	0

(d)

图 10

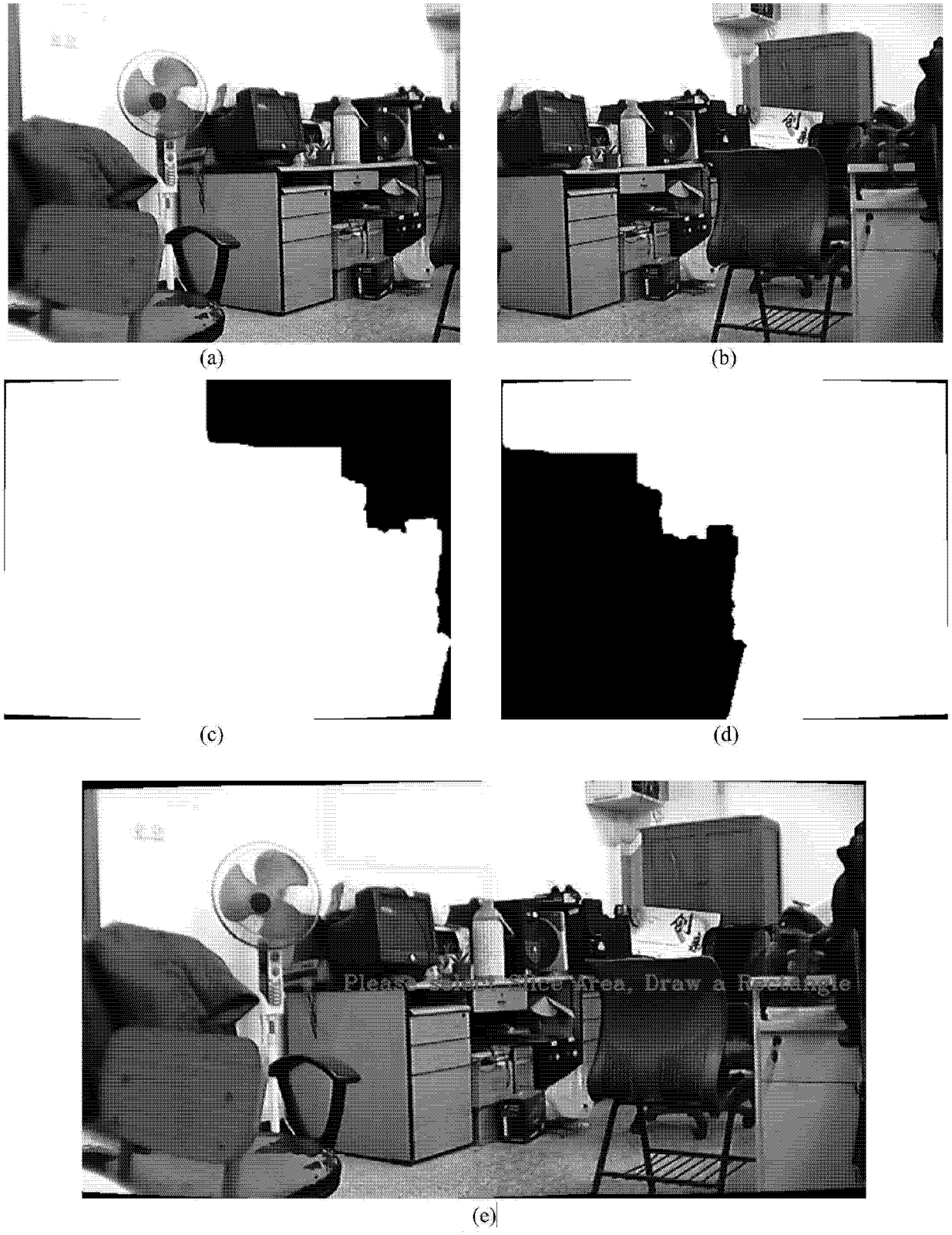
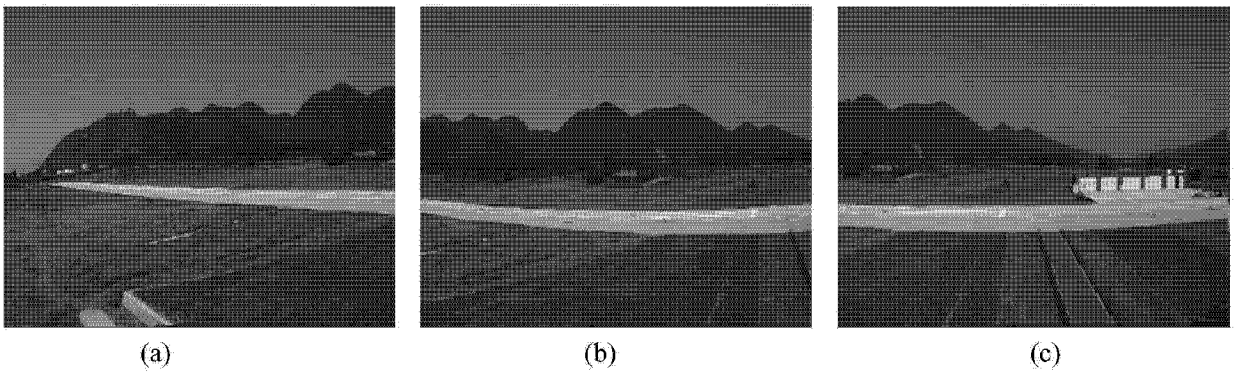
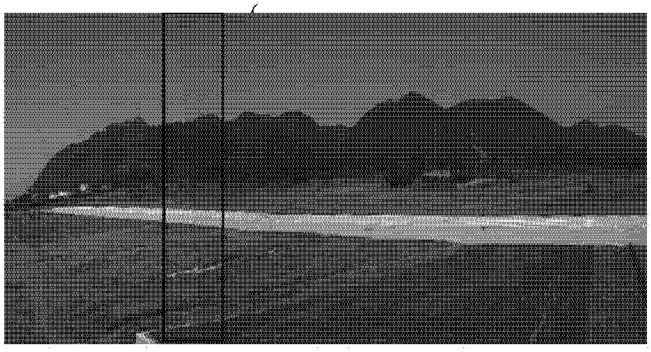


图 11



(a) (b) (c)



(d)

(e)

(f)

图 12