参赛密码 _____

（由组委会填写）

第六届“星云股份杯”

福州大学研究生数学建模竞赛

学院专业	物信学院　集成电路工程
参赛题号	A 视频抄袭检测方法研究
队员信息	**1.杨坚伟　181127127**
	2.陈炜炜　181127069
	3.陈宗航　181127074

参赛密码 _____

（由组委会填写）

第六届星云股份杯福州大学研究生数学建模竞赛

题 目 **视频抄袭检测方法研究**

摘 要：

本文针对视频抄袭与盗用的检测方法进行研究。首先对源视频各帧进行读取，然后，分析第 i 帧与 i+ξ帧的 RGB 特征，并计算这些特征的直方图差异。最后针对各个问题建立模型并求解。

针对问题一，我们利用第 i 帧和第 i + ξ 帧的 RGB 特征的直方图差异来定义关键帧，当第 i 帧与第 i + ξ 帧的 RGB 直方图差异大于某个阈值时，我们判定第 i 帧为其中一个关键帧。我们的数学模型如下：

$$fi = diff_{(th)}\left\{\sum_{c=1}^{3} ABS[hist(f_{1c}) - hist(f_{2c})]\right\}, i = 1,2,3...$$

其中，f_{1c}、f_{2c} 为以 ξ 为步长相邻的两张图片。$c \in (1,2,3)$是 RGB 图片的 3 个通道。$hist(\bullet)$用来计算直方图。$ABS(\bullet)$用来计算绝对值。$diff_{(th)}$ 为在阈值的约束下运用 RGB 直方图差异筛选出关键帧的函数。

针对问题二，我们首先对问题一对应模型所提取的关键帧随机提取 2 帧以上，作为匹配对象的关键帧，并用 SURF 算法对该关键帧进行特征点检测。然后根据视频帧与关键帧的粗匹配，我们建立了预仿射 Max 模型模型：D(x,y)=max{MF(i,H)}。其中 H 为筛选条件汉明距离，i 为视频第 i 帧，MF()函数为视频第 i 帧与关键帧的匹配对数，D(x,y)中的 x 记录帧的帧数，y 为对应帧的点数值。通过模型，推断编辑方法后视频中是否有相似的关键帧。

针对问题三，我们使用了均值哈希算法。通过缩小图片尺寸、简化色彩、计算平均值、比较像素的灰度，分别生成源视频关键帧与噪声帧的哈希值，最后算出两组哈希值的汉明距离来衡量两帧图片的相似度。基于此，我们的关键帧与噪声帧的匹配算法模型为：$\eta = \sum \Phi_{(\text{th})}[\mathrm{H}(\mathrm{fi}), H(yk)]$

针对问题四，我们在问题一的基础上融合了感知哈希算法。与均值哈希算法不同的是，该算法运用了二维离散余弦变换（DCT）；而与均值哈希算法相同的是，最后也是通过生成源视频关键帧与待匹配帧的哈希值并计算汉明距离来衡量两帧图片的相似度。我们的关键帧检测的改进方法是通过感知哈希值的汉明距离衡量两帧图片的相异度，从而在问题一的基础上提高了关键帧选取的置信度。只有当第 i 帧、第 $i + \xi$ 帧的 RGB 差异大于平均值，且第 i 帧、第 $i + \xi$ 帧的感知哈希值的汉明距离大于某个阈值的时候，才可以判定为关键帧。

在本题求解过程中，我们所建立的模型与实际紧密联系，有很好的通用性和推广性。但在关键帧提取时，我们的方法对于某些视频来说，关键帧偶尔会聚集在间隙不大的某几帧，从而使得误差增大。而实际上如果间隙越大的几帧，相关信息越少，越能具有代表性。

关键词：关键帧，RGB 特征，SURF，特征匹配，哈希算法，余弦变换

目 录

视频抄袭检测方法研究

1 问题重述

随着计算机网络和视频处理技术的发展，许多视频被人随意直接盗用或者经过编辑后盗用，造成了侵权和商业纠纷。当今各类商用视频相关软件编辑工作易学易用，视频编辑技术门槛较低，这大大增大了视频抄袭的可能性。因此，随着数字视频技术的迅猛发展，当今社会对视频版权保护产生了迫切的需求。

视频抄袭或盗用是指给定若干查询的视频片段，在视频数据库中进行查找，检测在数据库中是否存在相应的视频片段与查询视频片段内容相同；如果存在，查询视频片段就被称为视频抄袭或盗用片段。多数情况下，视频会被编辑后盗用，编辑的方法有多种，基本包括：缩放、剪切、镜像、画中画、改变亮度、增加字幕、增加噪声等。

1、定义一种关键帧，从源视频（每组中的视频1）中提取该关键帧，并解释该关键帧图像在后续检测视频是否被抄袭的算法中所起的作用。

2、基于问题1中的关键帧图像，建立相应的数学模型，检测经过亮度处理(每组中的视频2)、增加字幕（每组中的视频3）的编辑方法后视频中是否有相似的关键帧，并以此为依据推断源视频是否被盗用。

3、基于问题1中的关键帧图像，建立数学模型，检测增加噪声后的视频（每组中的视频4）中是否有相似的关键帧，并以此为依据推断源视频是否被盗用。

4、综合运用并改进上述基于关键帧检测的模型，从所给三组数据集中将被盗用的视频片段检测出来，分析并评估所使用模型的性能。

2 问题分析

2.1 问题一分析

问题一要求从源视频中定义一种关键帧并提取，解释该关键帧在后续检测视频中所起的作用。针对从原视频中提取关键帧的问题，我们先对原视频逐帧读取并分析，得出了第 i 帧和第 $i + \xi$ 帧的 RGB 特征的直方图差异[1]，再采用阈值法建立模型，进行关键帧的提取。

2.2 问题二分析

问题二中，要求基于问题 1 中的关键帧图像，建立相应的数学模型，检测经

过亮度处理(每组中的视频 2)、增加字幕（每组中的视频 3）的编辑方法后视频中是否有相似的关键帧。我们采用 SURF 算法[2]对问题一提取的关键帧和待检测视频逐帧进行特征点检测并匹配，发现粗匹配结果，存在许多误匹配。基于此我们建立了预仿射 Max 模型模型，来对初步的粗匹配结果进行筛选，剔除掉错误匹配。最终得到待检测视频中与关键帧正确匹配对数最多的帧。

2.3 问题三分析

问题三要求基于问题 1 中的关键帧图像，建立数学模型，检测增加噪声后的视频（每组中的视频 4）中是否有相似的关键帧。起先，我们准备将噪声视频逐帧读取，并进行噪声帧预处理，进而处理掉噪声点，采用许多去噪方法都不尽人意。最终，我们采取了均值哈希法，利用图片所包含的特征用来生成一组“指纹”并进行不同“指纹”的比较。具体的做法是缩小图片尺寸、简化色彩、计算平均值、比较像素的灰度，进一步地，分别计算源视频关键帧与噪声帧的哈希值并进行汉明距离的比较，最终得到关键帧与噪声视频的匹配结果。

2.4 问题四分析

问题四要求综合运用并改进上述基于关键帧检测的模型，从所给三组数据集中将被盗用的视频片段检测出来，分析并评估所使用模型的性能。我们在关键帧检测的改进问题中，融合了感知哈希算法。该算法运用了二维离散余弦变换对图像块进行处理，取图像块的低频系数构造特征矩阵。我们的关键帧检测的改进方法是添加了两帧图片哈希值的汉明距离差进行判断，扩大关键帧的检测范围，同时也能更准确地将待测视频与关键帧相似的片段检测出来，判断视频是否被盗用。

3 模型假设

⑴假设每组编辑过的视频具有实验通用性。缩放、剪切、镜像、画中画等视频暂不考虑。

⑵假设三组源视频的视频数据样本足够大，出现偶然数据概率低。

⑶假设实验样本有足够的代表性。

4 符号说明

ξ :遍历图片的步长

f_{1c}、f_{2c}:以 ξ 为步长相邻的两张图片

$c \in (1,2,3)$: RGB 图片的 3 个通道

$hist(\bullet)$:计算直方图的函数

$ABS(\bullet)$:计算绝对值的函数

$diff_{(th)}$:在阈值的约束下利用 RGB 直方图差异筛选出关键帧的函数

f_i:关键帧

y_k:带噪声视频的第 k 帧

$\Phi(\bullet)$:用于计算汉明权重与汉明距离的函数

η:带噪声视频中与关键帧 fi 相似的帧数 η

$\Phi_{th}(\bullet)$:在 Φ 的基础上结合阈值 th 计算带噪声视频中与关键帧 f_i 相似的帧数 η

$H(\bullet)$:均值哈希算法

5 模型建立与求解

5.1 问题一:关键帧检测

5.1.1 颜色直方图差异模型

我们利用第 i 帧和第 i + ξ 帧的 RGB 特征的直方图差异来定义关键帧，当第 i 帧与第 i + ξ 帧的 RGB 特征直方图差异大于某个阈值时，我们判定第 i 帧为其中一个关键帧；特殊地，假设待检测视频其有 λ 帧，当遍历到第 λ - ξ 帧的时候 ，取第 i + 1 帧为其中一个关键帧。

我们的数学模型如下：

$$fi = diff_{(th)}\left\{\sum_{c=1}^{3} ABS[\,hist(f_{1c}) - hist(f_{2c})]\right\}, i = 1,2,3...$$

其中，f_{1c}、f_{2c} 为以 ξ 为步长相邻的两张图片。$c \in (1,2,3)$是 RGB 图片的 3 个通道。$hist(\bullet)$用来计算直方图。$ABS(\bullet)$用来计算绝对值。$diff_{(th)}$ 为在阈值的约束

下差别 RGB 直方图差异筛选出关键帧的函数。

我们以 ξ 为步长遍历源视频的第 i 帧和第 i + ξ 帧，实验证明，以 3 帧为 ξ 进行遍历能使我们的模型在不失精准度的情况使运行的速度得到极大的提升。

5.1.2 实验结果与分析

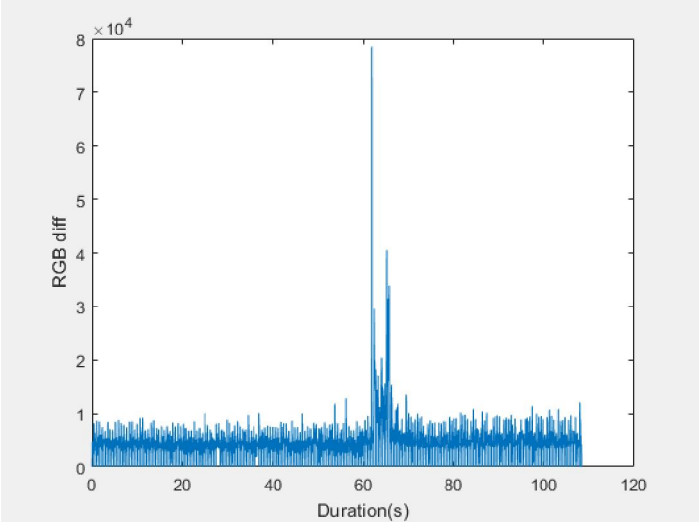


图 1 第一组视频中两帧间的 RGB 直方图差异

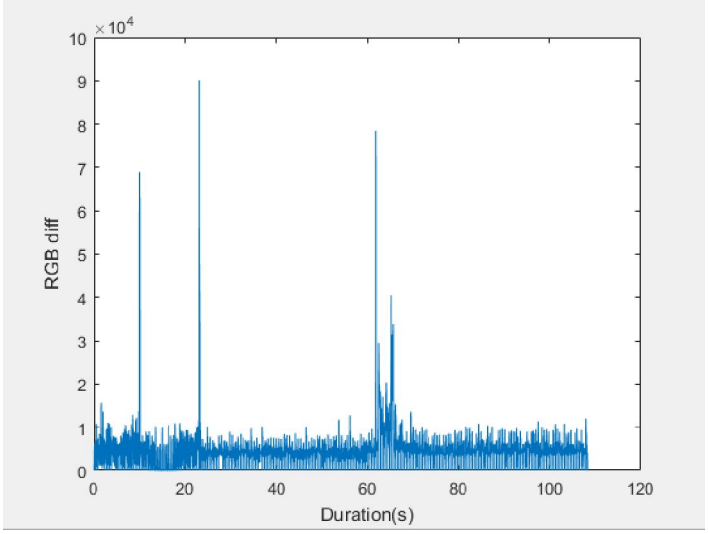


图 2 第二组视频中两帧间的 RGB 直方图差异

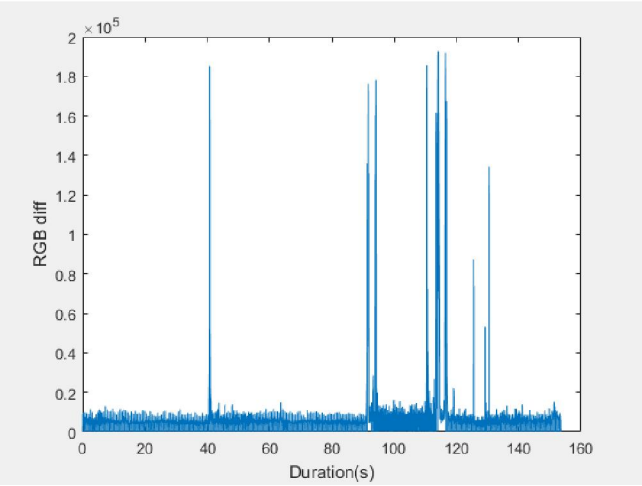


图 3 第三组视频中两帧间的 RGB 直方图差异

在上述三个图表中，X 轴表示视频的时间，Y 轴表示两帧图片间 RGB 直方图的差异。图形的最大点是从两帧中提取的 RGB 特征的最大变化，因此我们可以将 RGB 特征变化最大的相应帧视为关键帧。

从图 1 可以看出，第一组视频的关键帧有 2 至 3 帧；第二组视频的关键帧有 4 至 5 帧；第三组视频的关键帧约有 7 至 11 帧。

5.2 问题二:关键帧与处理视频匹配

5.2.1 预仿射 Max 模型

针对问题二，检测经过亮度处理(每组中的视频 2)、增加字幕（每组中的视频 3）的编辑方法后视频中是否有相似的关键帧，我们小组决定采用 SURF 方法进行关键帧与待测视频逐帧特征检测。接着使用我们自己提出的：

$$D(x,y)=\max\{MF（i,H）\}$$

预仿射 Max 模型对 SURF 算法[3]的匹配对（含野值）进行预测仿射变化，去除不满足变化的野值，进而输出匹对数最多的相似帧。（其中 H 为筛选条件汉明距离，i 为视频第 i 帧，MF()函数为视频第 i 帧与关键帧的匹配对数，D(x,y)中的 x 记录帧的帧数，y 为对应帧的点数值）图 4 为处理问题二的流程图。

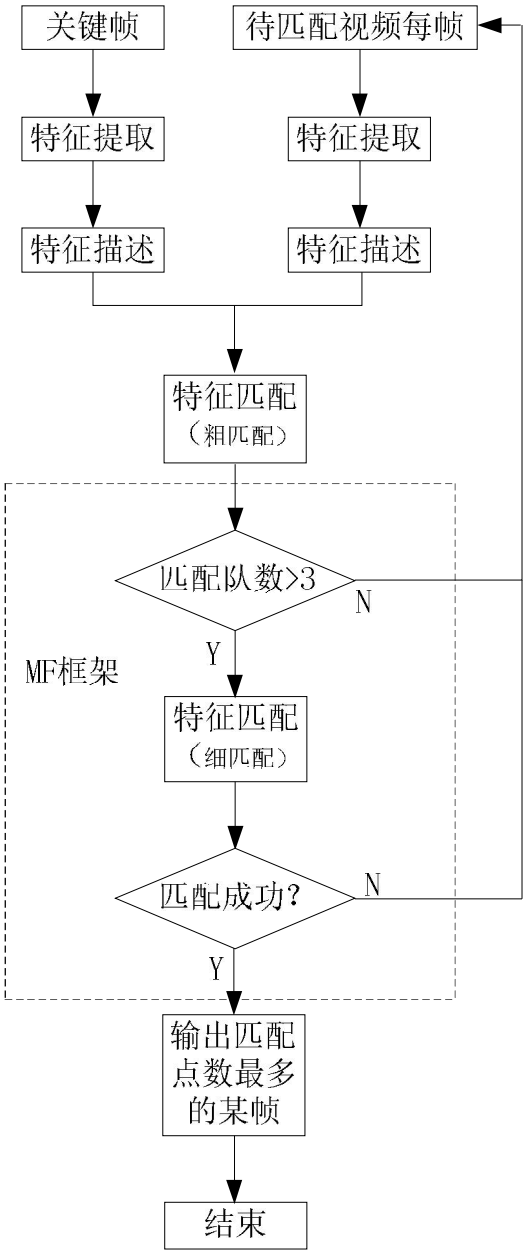


图 4 关键帧与待测视频匹配流程图

5.2.2 实验结果与分析

最终在 Matlab 上基于 SUFR 实现图像特征点检测，结果如下。Figure1、Figure2、Figure3 分别为基于问题一对第一组、第二组、第三组视频提取的关键帧，进行的特征点检测，并拟画出 100 个特征点可视图。

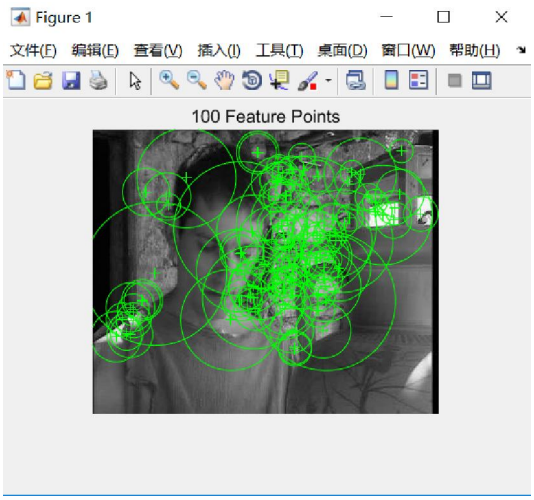


图 5 视频 1 关键帧特征点图

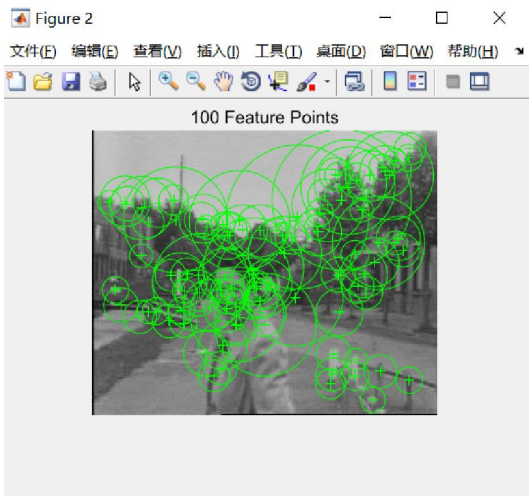


图 6 视频 2 关键帧特征点图

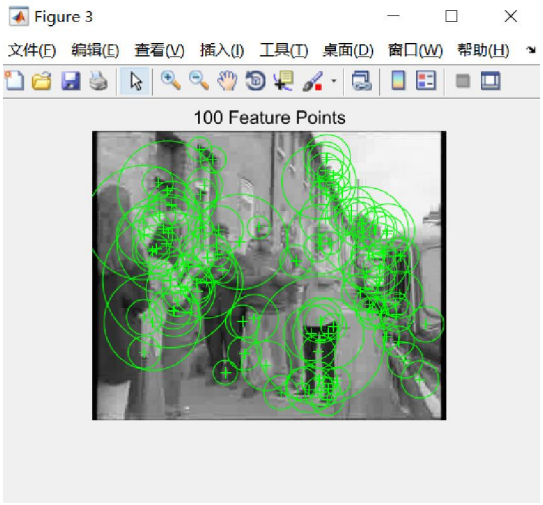


图 7 视频 3 关键帧特征点图

由于简单基于欧氏距离测度的 SURF 特征匹配中存在许多误匹配的点，这些误匹配点将严重影响后面变换模型参数的解算，因此需要利用汉明距离的方法将这些误匹配的点剔除，进而比较输出正确匹配的图像。

图 8 为问题一对应模型针对视频 1 所提取的关键帧，与经过亮度处理的待检测视频的图像配对情况。其中 figure1 为粗匹配，figure2 为细匹配。图一中的左下角为该视频中某帧与关键帧的匹配点数情况。可知当待检测视频的第 1547 帧时，该与关键帧匹配的点数为 115 对，为最多并输出图像配对情况。回到问题一所提关键帧为源视频的第 1547 帧，因而证明该视频存在与关键帧相似帧。

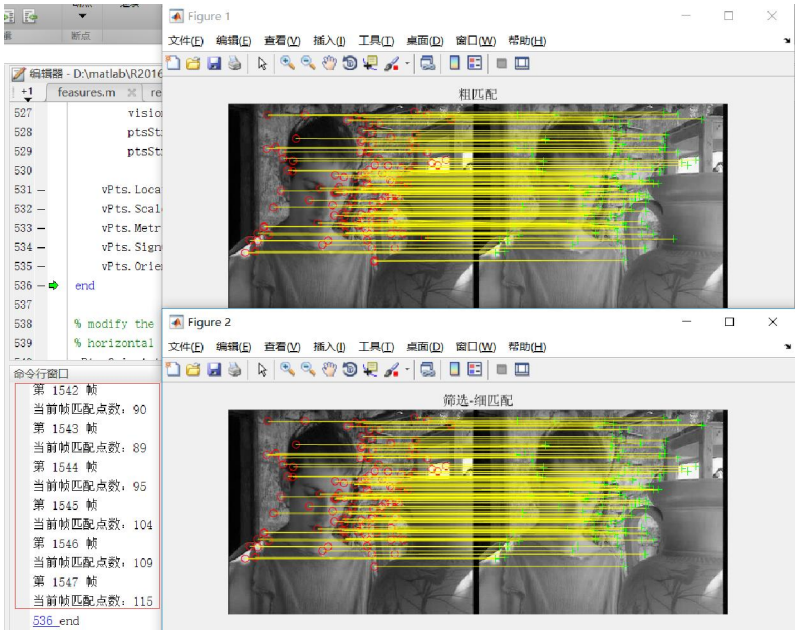


图 8 亮度视频检测结果

图 9 为问题一对应模型针对视频 1 所提取的关键帧，与经过添加字幕处理的待检测视频的图像配对情况。其中 figure1 为粗匹配，figure2 为细匹配。图一中的左下角为该视频中某帧与关键帧的匹配点数情况。可知当待检测视频的第 1547 帧时，该与关键帧匹配的点数为 141 对，为最多并输出图像配对情况。回到问题一所提关键帧为源视频的第 1547 帧，因而证明该视频存在与关键帧相似帧。

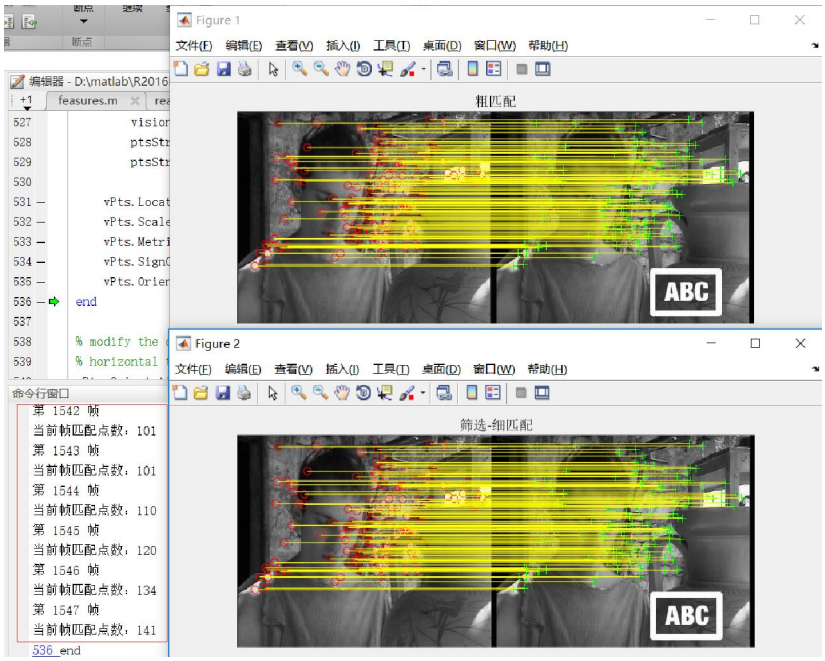


图 9 字幕视频检测结果

图 10 为问题一对应模型针对视频 2 所提取的关键帧，与经过亮度处理的待检测视频的图像配对情况。其中 figure1 为粗匹配，figure2 为细匹配。图一中的左下角为该视频中某帧与关键帧的匹配点数情况。可知当待检测视频的第 580 帧时，该与关键帧匹配的点数为 85 对，为最多并输出图像配对情况。回到问题一所提关键帧为源视频的第 580 帧，因而证明该视频存在与关键帧相似帧。

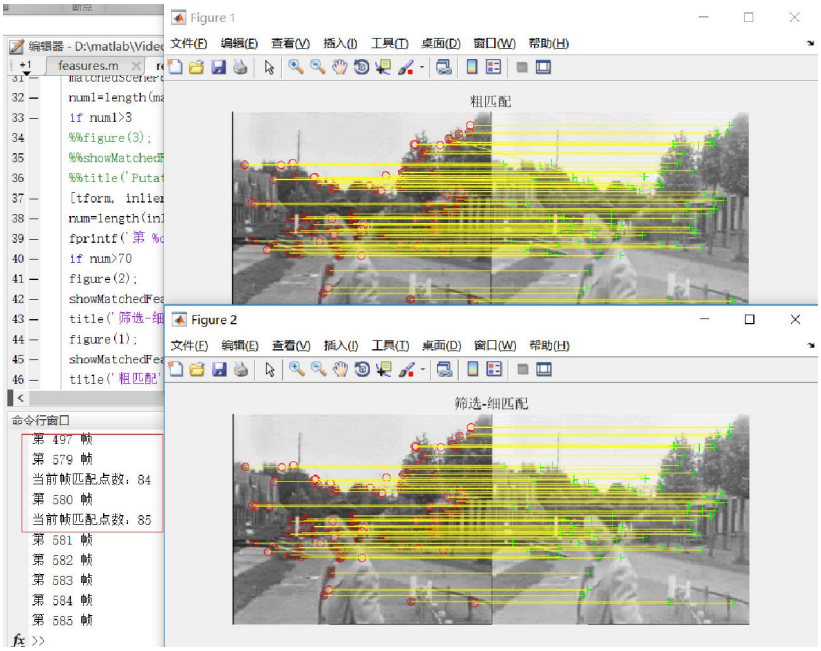


图 10 亮度视频检测结果

图 11 为问题一对应模型针对视频 2 所提取的关键帧，与经过添加字幕处理的待检测视频的图像配对情况。其中 figure1 为粗匹配，figure2 为细匹配。图一中的左下角为该视频中某帧与关键帧的匹配点数情况。可知当待检测视频的第 580 帧时，该与关键帧匹配的点数为 41 对，为最多并输出图像配对情况。回到问题一所提关键帧为源视频的第 580 帧，因而证明该视频存在与关键帧相似帧。

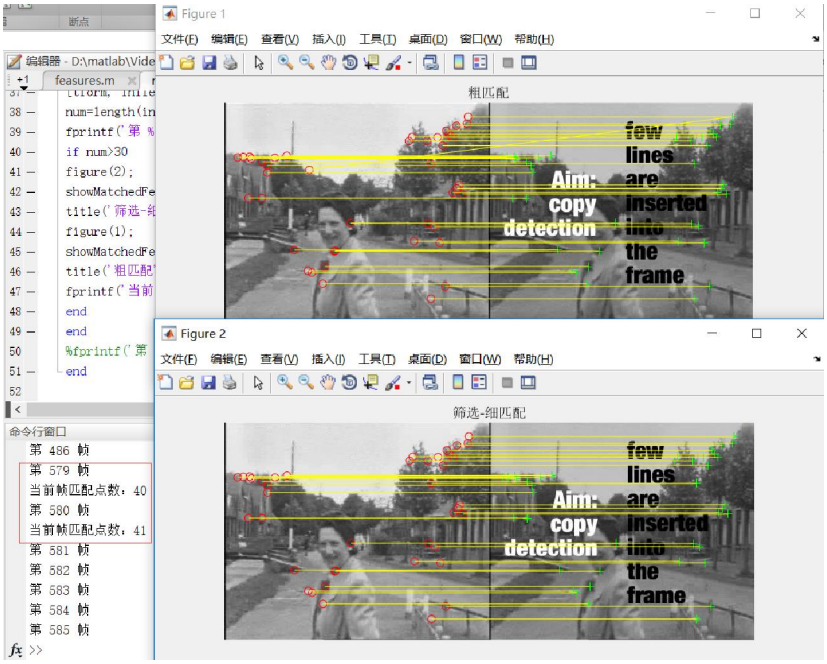


图 11 字幕视频检测结果

图 12 为问题一对应模型针对视频 3 所提取的关键帧，与经过亮度处理的待检测视频的图像配对情况。其中 figure1 为粗匹配，figure2 为细匹配。图一中的左下角为该视频中某帧与关键帧的匹配点数情况。可知当待检测视频的第 2283 帧时，该与关键帧匹配的点数为 192 对，为最多并输出图像配对情况。回到问题一所提关键帧为源视频的第 2283 帧，因而证明该视频存在与关键帧相似帧。

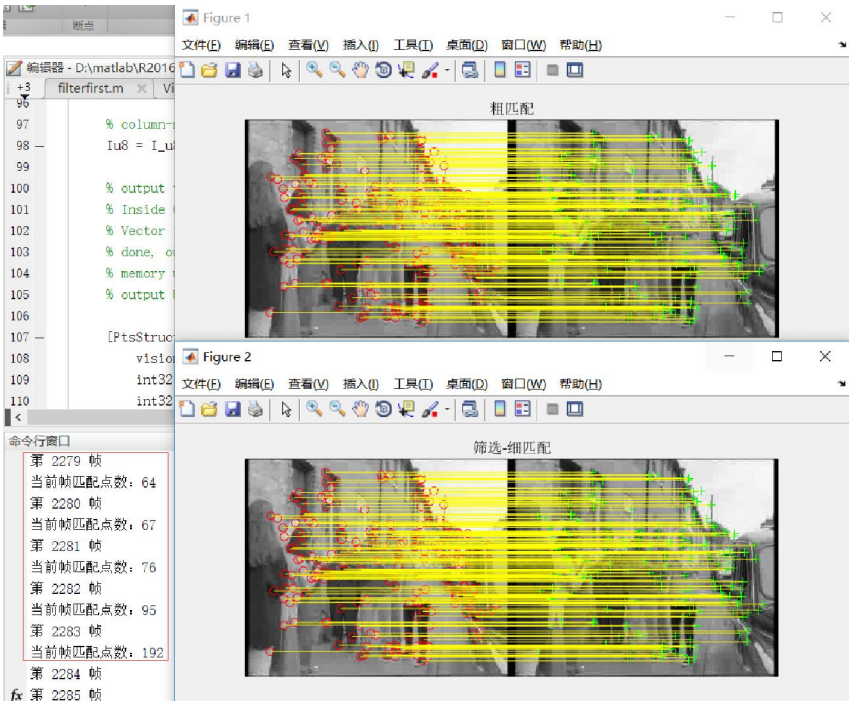


图 12 亮度视频检测结果

图 13 为问题一对应模型针对视频 3 所提取的关键帧，与经过添加字幕处理的待检测视频的图像配对情况。其中 figure1 为粗匹配，figure2 为细匹配。图一中的左下角为该视频中某帧与关键帧的匹配点数情况。可知当待检测视频的第 2283 帧时，该与关键帧匹配的点数为 80 对，为最多并输出图像配对情况。回到问题一所提关键帧为源视频的第 2283 帧，因而证明该视频存在与关键帧相似帧。

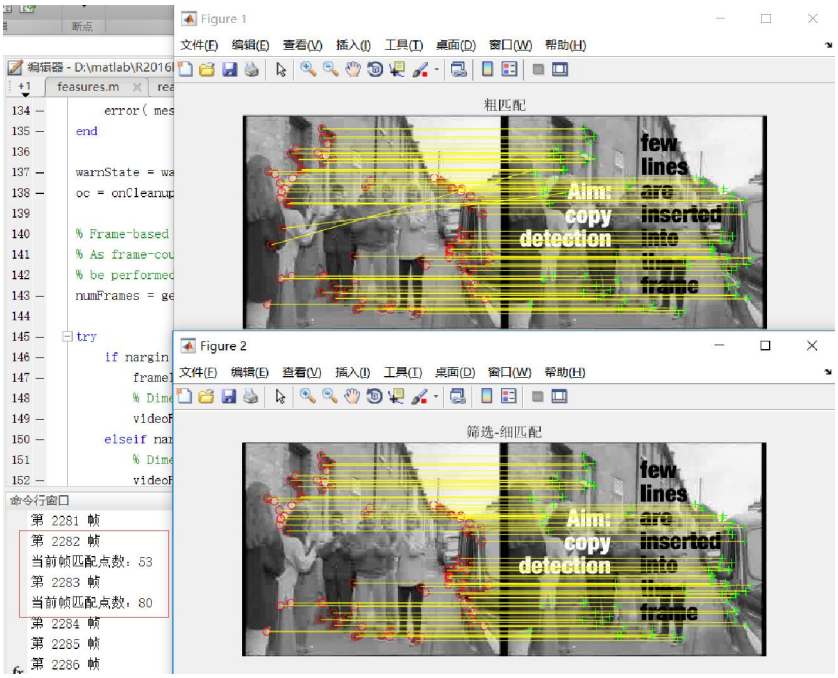


图 13 字幕视频检测结果

视频组\帧数\待匹配视频组	汉明距	粗匹配点数	细匹配点数 D	能否检测相似帧
1\1646\亮度	25	50	30	能
1\1646\字幕	50	86	53	能
2\252\亮度	30	66	35	能
2\252\字幕	40	73	44	能
3\3140\亮度	30	77	43	能
3\3140\字幕	60	96	78	能

表一

应用该所提出的模型，对问题一所提出的更多关键帧进行与待测视频逐一进行帧匹配，结果得表一，其中汉明距为 SURF 算法计算测得相应坐标点最小距离的

两倍。通过实验分析，模型 D(x,y)=max{MF（i,H）}能够更精准的检测出视频之中的相似帧。

5.3 问题三：关键帧与噪声视频的匹配

5.3.1 哈希模型简述

感知哈希技术起源于数字水印技术，借鉴于传统密码学哈希和多媒体认证等相关领域的概念和理论。图像感知哈希技术是感知哈希最重要的分支之一。基于感知哈希图像认证为图像内容认证提供了安全可靠的技术支撑，认证模型如图 14 所示。

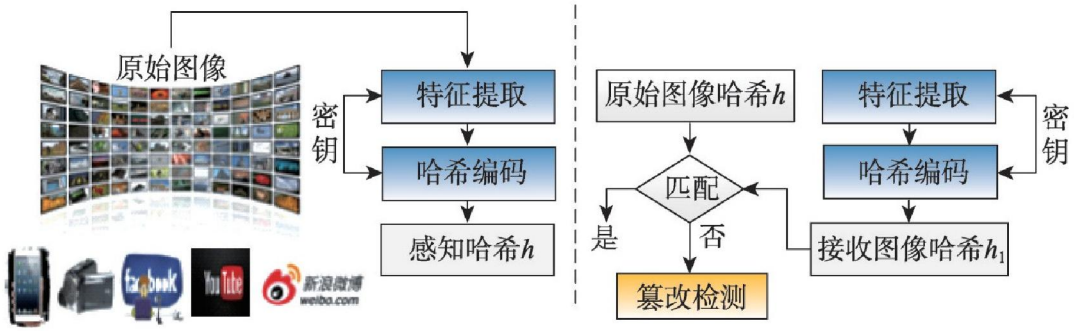


图 14 感知哈希认证应用模型

目前，针对基于感知哈希的图像认证，大多数哈希算法致力于基于鲁棒的特征提取产生图像哈希。主要包括：（1）基于不变特征变换的方法，如傅里叶-梅林变换（Fourier-Mellin transform，FMT）[4]，离散余弦变换（discrete cosine transform，DCT）[5]等变换域方案。（2）基于局部特征点的方案，如基于 end-stopped 小波[6]和基于 SIFT（scale-invariant feature transform，SIFT）[7]，主要利用局部特征在几何变换等图像处理攻击下的不变性产生哈希值。（3）基于降维的方案，如奇异值分解（singular value decomposition，SVD）[8]、非负矩阵分解（non-negative matrix factorization，NMF）[9]和快速约翰逊-林登斯特拉斯变换（fast Johnson-Lindenstrauss transform，FJLT）[10]

由哈希模型得到的哈希编码一般用汉明码距离计算公式 Φ 来进行匹配，h_1 和 h_2 为两个图片的哈希序列：

$$\text{Distance} = \Phi(h_1, h_2)$$

其中 $\Phi(h_1, h_2) = |h_1 - h_2|$ 。

5.3.2 均值哈希模型简述

均值哈希算法（Average hash algorithm）是哈希算法的一类。图片所包含的特征被用来生成一组指纹（不过它不是唯一的）,而这些指纹之间是可以相互进行比较的。下面是均值哈希算法的步骤 ：

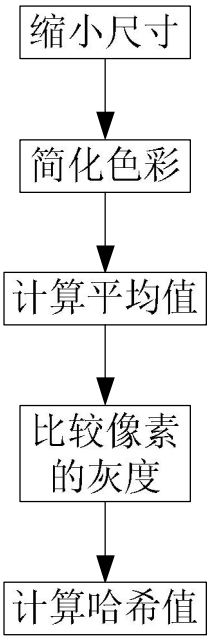


图 15 均值哈希算法步骤

第一步，缩小尺寸。最快速的去除高频和细节，只保留结构明暗的方法就是缩小尺寸。将图片缩小到 8x8 的尺寸，总共 64 个像素。摒弃不同尺寸、比例带来的图片差异。

第二步，简化色彩。将缩小后的图片，转为 64 级灰度。也就是说，所有像素点总共只有 64 种颜色。

第三步，计算平均值。计算所有 64 个像素的灰度平均值。

第四步，比较像素的灰度。将每个像素的灰度，与平均值进行比较。大于或等于平均值，记为 1；小于平均值，记为 0。

第五步，计算哈希值。将上一步的比较结果，组合在一起，就构成了一个 64

位的整数，这就是这张图片的指纹。组合的次序并不重要，只要保证所有图片都采用同样次序就行了。

如果图片放大或缩小，或改变纵横比，结果值也不会改变。增加或减少亮度或对比度，或改变颜色，对哈希值都不会太大的影响。均值哈希算法最大的优点是计算速度快。

5.3.3 关键帧与噪声视频的匹配算法

关键帧与噪声视频的匹配数学模型如下：

$$\eta = \sum \Phi_{(th)}\left[\mathrm{H}(\mathrm{f_i}), H(y_k)\right]$$

其中，$\mathrm{f_i}$ 、$\mathrm{y_k}$ 分别为源视频的关键帧第 i 帧和带噪声视频的第 k 帧，我们以 3 帧的步长遍历带噪声的视频并与源视频关键帧进行匹配（实验证明，以 3 帧为步长进行遍历能使我们的模型的运行的效率最大化），函数 H 为本问题的核心——均值哈希算法，它返回输入图片的哈希指纹。函数 Φ 结合阈值 th 计算两个哈希指纹的汉明权重与汉明距离，并输出带噪声视频中与关键帧 $\mathrm{f_i}$ 相似的帧数 η。通过判断 η 的值，我们可以判定带噪声视频是否是源视频的复制品。通常阈值 th 的选取遵循“如果哈希指纹不相同的数据位不超过 7，就说明两张图片很相似；如果大于 15，就说明这是两张不同的图片”原则。

5.3.4 实验结果与分析

我们通过均值哈希算法来实现关键帧与噪声帧的匹配，实验结果如下：

图 16 视频一关键帧与噪声帧匹配

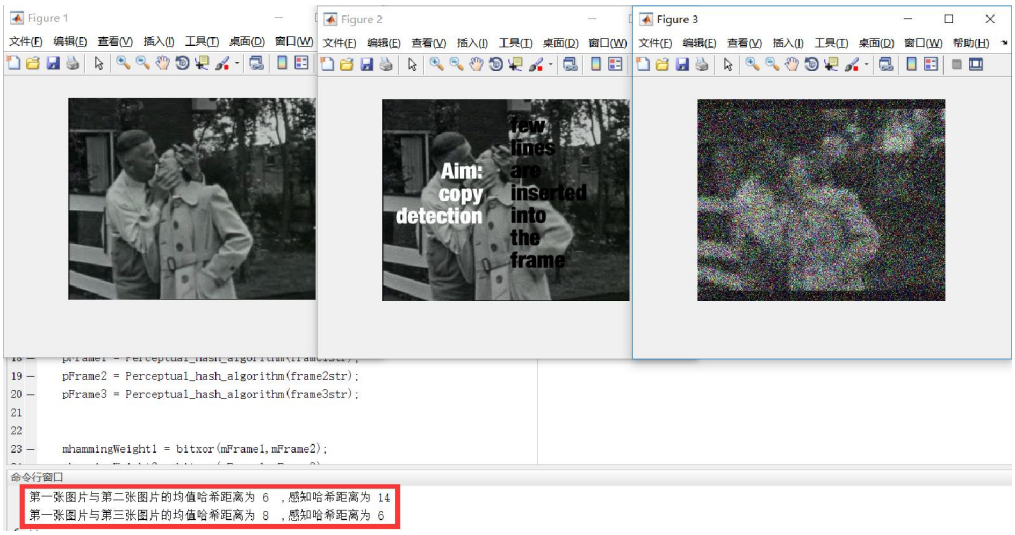


图 17 视频二关键帧与噪声帧匹配

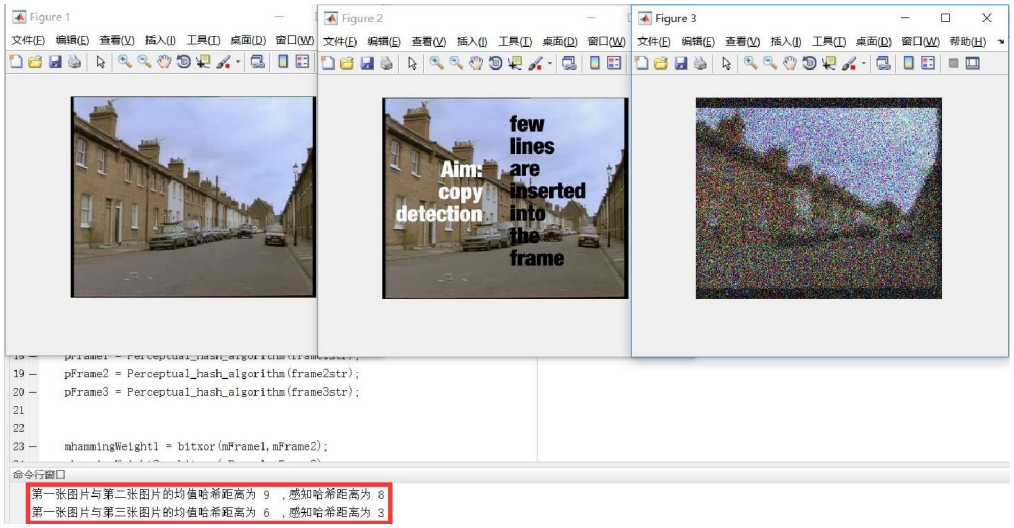


图 18 视频三关键帧与噪声帧匹配

如图 16 中左下角所示，源视频关键帧（第一张图）与相对应的噪声帧（第三张图）的均值哈希的汉明距离为 13，远小于 64。

如图 17 中左下角所示，源视频关键帧（第一张图）与相对应的噪声帧（第三张图）的均值哈希的汉明距离为 8，远小于 64。

如图 18 中左下角所示，源视频关键帧（第一张图）与相对应的噪声帧（第三张图）的均值哈希的汉明距离为 6，远小于 64。

综上所述，只要设定合适的阈值，就可以实现检测出增加噪声后的视频中是否有相似的关键帧，从而推断源视频是否被盗用。

5.4 问题四:关键帧检测的改进

5.4.1 感知哈希模型简述

感知哈希（Perceptual hash algorithm）是基于离散余弦变换（Discrete Cosine Transform，DCT）的图像哈希算法。

DCT 是可分离的变换，其变换核为余弦函数。DCT 除了具有一般的正交变换性质外， 它的变换阵的基向量能很好地描述人类语音信号和图像信号的相关特征。因此，在对语音信号、图像信号的变换中，DCT 变换被认为是一种准最佳变换。

在关键帧检测的改进问题中，我们使用二维离散余弦变换。该算法对图像块进行处理，取图像块的低频系数构造特征矩阵。

下面是感知哈希算法简单的步骤：

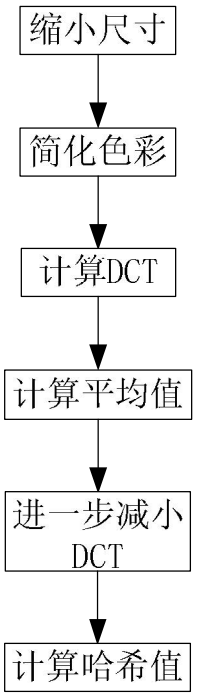


图 19 感知哈希算法步骤

第一步，缩小尺寸。最快速的去除高频和细节，只保留结构明暗的方法就是缩小尺寸。将图片缩小到 8x8 的尺寸，总共 64 个像素。摒弃不同尺寸、比例带来的图片差异。

第二步，简化色彩。将缩小后的图片，转为 64 级灰度。也就是说，所有像素点总共只有 64 种颜色。

第三步，计算 DCT（离散余弦变换）。DCT 是把图片分解频率聚集和梯状形，虽然 JPEG 使用 8*8 的 DCT 变换，在这里使用 32*32 的 DCT 变换。

第四步，缩小 DCT。虽然 DCT 的结果是 32*32 大小的矩阵，但我们只要保留左上角的 8*8 的矩阵，这部分呈现了图片中的最低频率。

第五步，计算平均值。计算所有 64 个值的平均值。

第六步，进一步减小 DCT。这是最主要的一步，根据 8*8 的 DCT 矩阵，设置 0 或 1 的 64 位的 hash 值，大于等于 DCT 均值的设为”1”，小于 DCT 均值的设为“0”。结果并不能告诉我们真实性的低频率，只能粗略地告诉我们相对于平均值频率的相对比例。只要图片的整体结构保持不变，哈希值就不变。能够避免伽马校正或颜色直方图被调整带来的影响。

第七步，计算哈希值。将 64bit 设置成 64 位的长整型，组合的次序并不重要，只要保证所有图片都采用同样次序就行了。将 32*32 的 DCT 转换成 32*32 的图像。将上一步的比较结果，组合在一起，就构成了一个 64 位的整数，这就是这张图片的指纹。组合的次序并不重要，只要保证所有图片都采用同样次序就行了（例如，自左到右、自顶向下、big-endian）。

得到指纹以后，就可以对比不同的图片，看看 64 位中有多少位是不一样的。在理论上，这等同于计算“汉明距离”（Hammingdistance）。如果不相同的数据位不超过 7，就说明两张图片很相似；如果大于 15，就说明这是两张不同的图片。

5.4.2 关键帧检测的改进算法

我们的关键帧检测的改进算法是 5.1 中颜色直方图差异法与感知哈希法的融合。它在计算第 i 帧、第 i + ξ 帧的 RGB 特征直方图的基础上，同时计算第 i 帧、第 i + ξ 帧的哈希指纹。我们的关键帧检测的改进方法是添加了两帧图片哈希指纹差异的约束。只有当第 i 帧、第 i + ξ 帧的 RGB 差异大于平均值，且第 i 帧、第 i + ξ 帧的哈希指纹大于某个 th 阈值的时候，才判定为关键帧。

5.4.3 模型性能评估

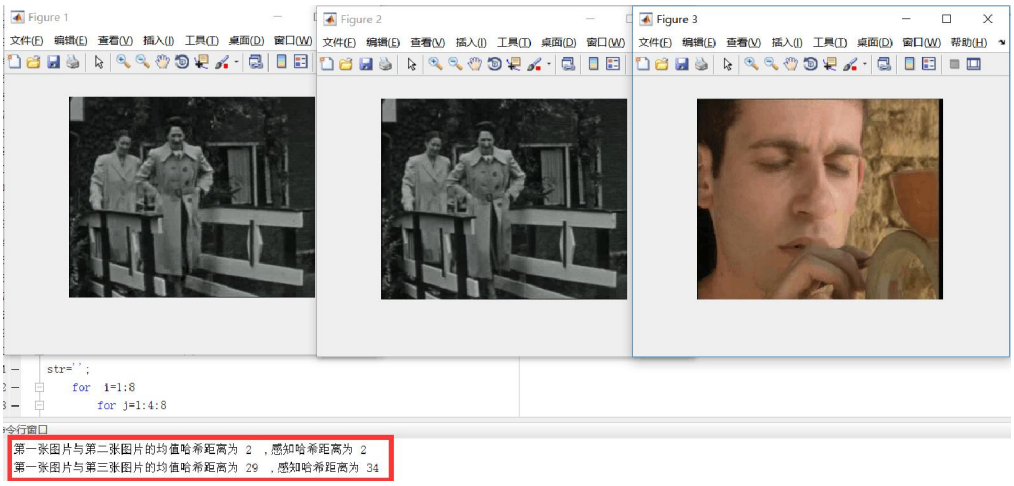


图 20 视频三关键帧与噪声帧匹配

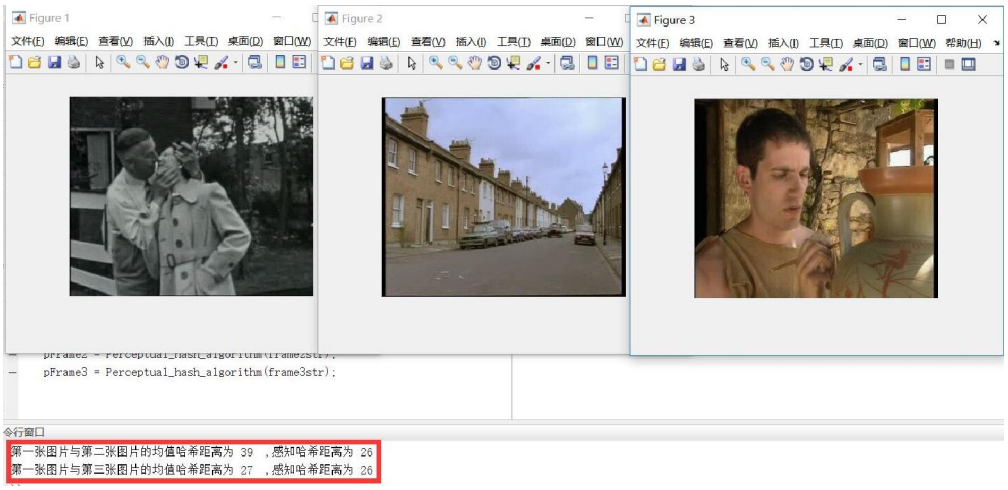


图 21 视频三关键帧与噪声帧匹配

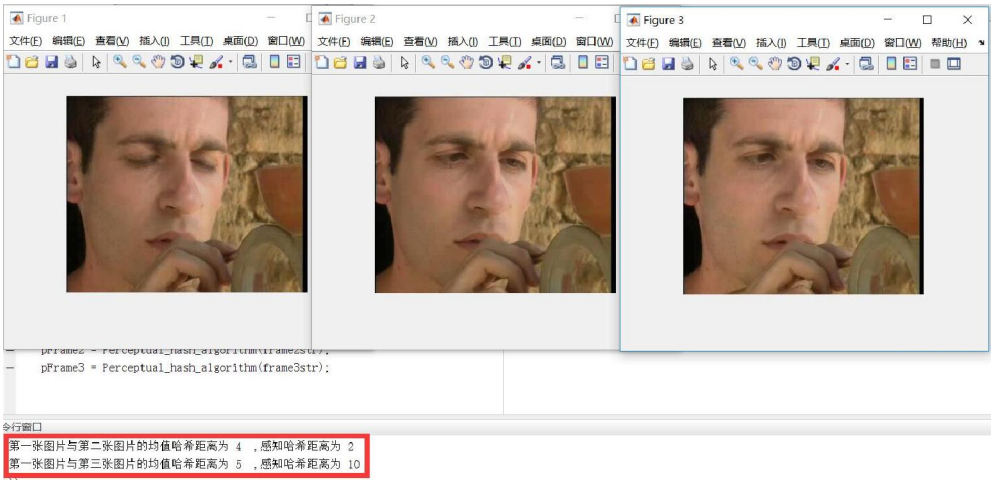


图 22 视频三关键帧与噪声帧匹配

在本小节中，我们实现了融合了感知哈希算法改进了关键帧的检测。图 20、图 21、图 22 对感知哈希算法进行了不同组别的评估。

由图 20 可以看出，第一张图片与第二张图片极为相似，因此第一张图片与第

二张图片的感知哈希汉明距离为 2。而第一张图片与第三张图片分属于完全不同的视频，因此它们的感知哈希汉明距离为 34，符合我们的预期。

由图 21 可以看出，三张图片完全不同，因此它们的感知哈希汉明距离都为 26，偏大，符合我们的预期。

由图 22 可以看出，三张图片极为相似，可能来自相邻近的帧，它们的两两间的感知哈希汉明距离分别为 2，10。符合我们的预期。

6 模型的评价与改进

6.1 模型的创新点与优势：

①首先对视频数据及元素间的相关性进行分析，处理，简化了计算。

②在 SURF 算法基础上，添加预仿射 Max 模型，实现预测仿射变化，去除不满足变化的野值，从而达到高精度图像匹配。实验数据得知该模型可行。

③在噪点帧与关键帧匹配中，使用均值哈希算法，成功基于汉明距离条件进行匹配，实验得知汉明距离偏大，代表两帧越不相关，汉明距越小，代表两帧越相似。其中的阈值筛选，可通过实验数据划分得到。这与哈希算法理论相一致。

④将哈希算法用于噪点视频，并结合关键帧提取改良升级为感知哈希法。

⑤问题二、问题三、问题四中所提及的模型，我们为了验证模型的可信度，使用了大量的数据纵横交错检验，采用正确-错误的数据对模型进行验证。实验数据分析得模型可信度较高。

⑥所建立的模型与实际紧密联系，由一些利用简单的模型就能达到很好的效果，有很好的通用性和推广性。

⑦对视频帧进行了逐帧处理，避免了数据偶然缺失的麻烦。

⑧运用 Matlab 软件进行计算，可信度高。

⑨论文中图形与数据相结合更具有说服力。

6.2 模型的不足和改进

①在关键帧提取时，我们的方法对于某些视频来说，关键帧偶尔会聚集在间隙不大的某几帧，从而使得误差增大。

改进：针对关键帧提取的不足，在问题四的模型中，我们将均值哈希法融合进关键帧检测之中，升级为感知哈希法，使得获取的关键帧能更具有代表性。

②在处理数据和求解过程中不可避免的出现各种误差，在一定也影响到模型求解的精确度。

参考文献

[1]唐亮，林智慧，梁涛.基于 Matlab GUI 的数字图像处理仿真平台的设计阴.数字技术与应用，2015(07):179-180.

[2]李慧，蔺启忠，刘庆杰，等.基于 FAST 和 SURF 的遥感图像自动配准方法[J].国土资源遥感，2012,24(2):28-33.

[3]葛永新，杨丹，张小洪.基于特征点对齐度的图像配准方法[J].电子与信息学报，2007, 2(29),425 一 428.

[4] Swaminathan A, Mao Y N, Wu M. Robust and secure imageHashing[J]. IEEE Transactions on Information Forensics &Security, 2006, 1(2): 215-230.

[5] Huang Z Q, Liu S G. Robustness and discrimination orientedHashing combining texture and invariant vector distance[C]//Proceedings of the 26th ACM International Conference
on Multimedia, Seoul, Oct 22- 26, 2018. New York: ACM,2018: 1389-1397.

[6] Monga V, Evans B L. Perceptual image Hashing via featurepoints: performance evaluation and tradeoffs[J]. IEEE Tran-sactions on Image Processing, 2006, 15(11): 3453-3466.

[7] Lv X D, Wang Z J. Perceptual image Hashing based onshape contexts and local feature points[J]. IEEE Transactionson Information Forensics & Security, 2012, 7(3): 1081-1093.

[8] Kozat S S, Venkatesan R, Mihcak M K. Robust perceptual
image Hashing via matrix invariants[C]//Proceedings of the
2004 IEEE International Conference on Image Processing,
Singapore, Oct 24-27, 2004. Piscataway: IEEE, 2004: 3443-
3446.

[9] Monga V, Mihcak M K. Robust and secure image Hashingvia non-negative matrix factorizations[J]. IEEE Transactionson Information Forensics & Security, 2007, 2(3): 376-390.

[10] Lv X D, Wang Z J. An extended image Hashing concept:content-based fingerprinting using FJLT[J]. EURASIP Journalon Information Security, 2009(1): 1-16.

附 录

(有证明的加证明)1、证明

1、代码

问题一代码

```
img_file = dir('images\3-1\');
img_dir  = 'images\3-1\' ;
NOF = max(size(img_file) - 2);
imglist=img_file(3 : NOF + 2);
video = VideoReader('3-1.mpg');

step = 3;
frame_len = NOF + 2;
img_diff(round(frame_len / step)) = 0;
```

%遍历视频，计算两图片的RGB的特征值，分别作差取绝对值进行求和，得到RGB特征点的差异值

```
for i = 1 : step : frame_len
    frame = imread(strcat(img_dir,imglist(i).name));
    rHist = double(imhist(frame(:,:,1), 256));
    gHist = double(imhist(frame(:,:,2), 256));
    bHist = double(imhist(frame(:,:,3), 256));


    if i < (frame_len - step)
        frame1 = imread(strcat(img_dir,imglist(i +
step).name));
        r1Hist = double(imhist(frame1(:,:,1), 256));
        g1Hist = double(imhist(frame1(:,:,2), 256));
        b1Hist = double(imhist(frame1(:,:,3), 256));
    end

    img_diff(i)= sum(abs(rHist-r1Hist) +
abs(gHist-g1Hist) + abs(bHist-b1Hist))/3;
```

%显示运行进度

```
   clc;
   X = ['Process: ', num2str(int64((i/frame_len)*100)),
'%'];
```

```
    disp(X)
end

%剔除离群值求平均，算出阈值
Threshold=trimmean(img_diff,0.3) * 11;

for i = 1 : step : frame_len

   tmp = 0;
    if (i  >  frame_len - step)
        if( 0 == tmp ) %只进入一次
           imwrite(read(video, i), ['.\3-keyframe\'
num2str(i) '.jpg']);
           tmp = tmp + 1;
        end
    end

    if(img_diff(i) > Threshold)
       imwrite(read(video, i), ['.\3-keyframe\'
num2str(i) '.jpg']);
    end;
end
```

问题二代码。

```
clc;
%读取问题一模型提取的原视频关键帧
boxImage =
imread('D:/matlab/Video-Keyframe-Extraction-Using-RGB-
Features-in-Matlab-master/keyframe/three/2283.jpg');
%将RGB图像转化为灰度图
boxImage = rgb2gray(boxImage);
%读取待测视频
obj =
VideoReader('D:/matlab/Video-Keyframe-Extraction-Using
-RGB-Features-in-Matlab-master/3-3.mpg');
%遍历待测视频每帧，记录总帧数
numFrames = obj.NumberOfFrames;
```

```
numzeros= 4;
nz = strcat('%0',num2str(numzeros),'d');

%待测视频的逐帧预处理

for k = 1:numFrames
    sceneImage = read(obj,k);
    sceneImage = rgb2gray(sceneImage);

%提取SURF特征点

    boxPoints = detectSURFFeatures(boxImage);
    scenePoints = detectSURFFeatures(sceneImage);

 %根据特征点生成图像的特征向量

[boxFeatures, boxPoints] = extractFeatures(boxImage,
boxPoints);
[sceneFeatures, scenePoints] =
extractFeatures(sceneImage, scenePoints);

%初步建立一个匹配对，即粗匹配（含野值）

boxPairs = matchFeatures(boxFeatures, sceneFeatures);
matchedBoxPoints = boxPoints(boxPairs(:, 1), :);
matchedScenePoints = scenePoints(boxPairs(:, 2), :);
num1=length(matchedScenePoints);

%预测仿射变化，去除不满足变化的野值

if num1>3
[tform, inlierBoxPoints, inlierScenePoints] =
estimateGeometricTransform(matchedBoxPoints,
matchedScenePoints, 'affine');
num=length(inlierScenePoints);

fprintf('第 %d 帧\n',k);


%汉明距

if num>50
figure(2);
showMatchedFeatures(boxImage, sceneImage,
inlierBoxPoints, inlierScenePoints, 'montage');

title('筛选-细匹配');

figure(1);
showMatchedFeatures(boxImage, sceneImage,
matchedBoxPoints, matchedScenePoints, 'montage');

title('粗匹配');
```

```
fprintf('当前帧匹配点数：%d\n',num);
end
end
end
```

问题三代码：

```
d = dir('images\1-4\');
NOF = max(size(d) - 2);
imglist = d(3 : NOF + 2);
hammingDistance = zeros(NOF-1, 1);

Threshold = 8;
similarImgNum = 0;

% read a keyframe for test
keyFrame = Mean_hash_algorithm('651.jpg');

%逐帧遍历
for i=1:NOF-1

    noiseFrame =
Mean_hash_algorithm(strcat('images\1-4\',
imglist(i).name));
    hammingWeight = bitxor(keyFrame, noiseFrame); %计算汉
明权重
    hammingDistance(i) = sum(hammingWeight(:) == 1); %计
算汉明距离

    if(hammingDistance(i) < 7)
        similarImgNum = similarImgNum + 1;
    end
end;

fprintf ('number of Similar Image is %d', similarImgNum);
if (similarImgNum > 3)
    fprintf ('one fragment was stolen');
else
```

```
    fprintf ('No fragments were stolen');
end


% function :binary to hex
function str=BinToHex(A)
str='';
    for  i=1:8
        for j=1:4:8
            temp = dec2hex(bin2dec(num2str(A(i,j:j+3))));
            str=strcat(str,num2str(temp));
        end
    end
end


% function: Mean_hash_algorithm
% img_dir : directory of input  image
function A= Mean_hash_algorithm(img_dir)

    img=imread(img_dir);
    temp=rgb2gray(img);

    img2=imresize(temp,[8 8]);
    img2=uint8(double(img2) / 4);

    average=mean(mean(img2));

    img2(img2 < average) = 0;
    img2(img2 >= average) = 1;
    A = img2;

    Str = BinToHex(A); %compute fingerprint of image
end
```

问题四代码：

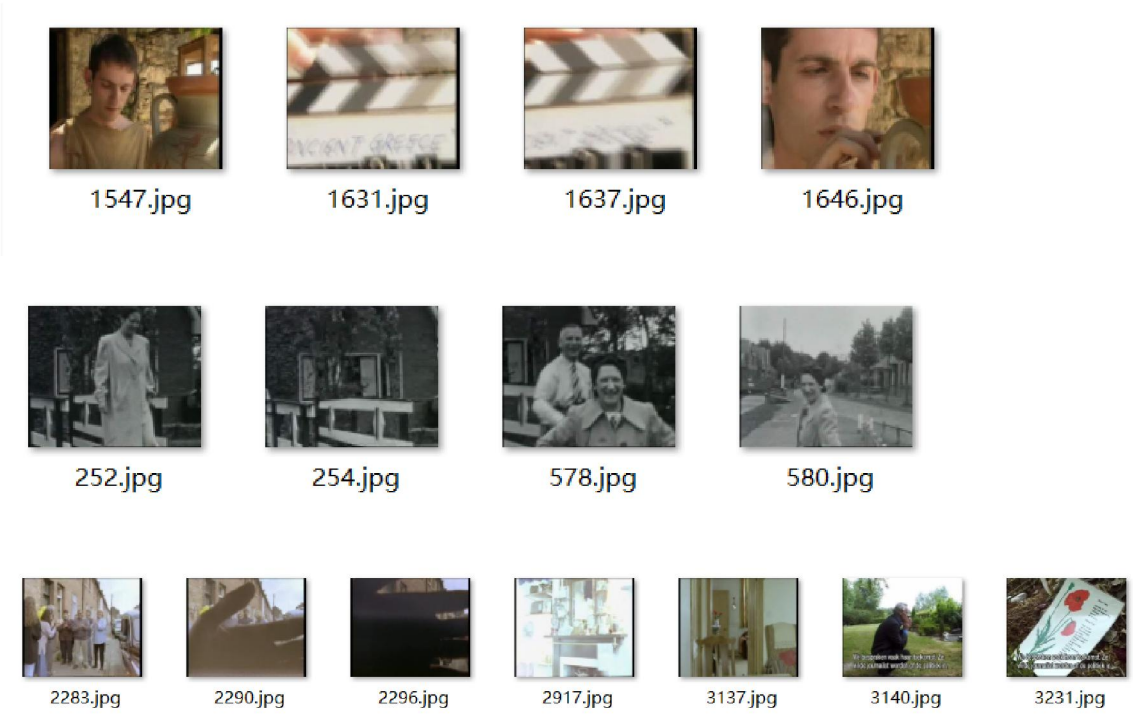
```
img_file = dir('images\1-1\');
img_dir  = 'images\1-1\' ;
NOF = max(size(img_file) - 2);
imglist = img_file(1:NOF + 2);
video = VideoReader('1-1.mpg');

step = 3;
```

```
frame_len = NOF + 2;
img_diff(round(frame_len / step)) = 0;
hammingDistance(round(frame_len / step)) = 0;
hammingDistanceThreshold = 20;

%遍历视频，计算RGB特征点的差异值，融合感知哈希算法
for i = 1 : step : frame_len
    frame1 = imread(strcat(img_dir,imglist(i).name));
    rHist = double(imhist(frame1(:,:,1), 256));
    gHist = double(imhist(frame1(:,:,2), 256));
    bHist = double(imhist(frame1(:,:,3), 256));
    pHash_of_frame1 =
Perceptual_hash_algorithm(strcat(img_dir,imglist(i).na
me));

    if i < (frame_len - step)
        frame2 = imread(strcat(img_dir,imglist(i +
step).name));
        r1Hist = double(imhist(frame2(:,:,1), 256));
        g1Hist = double(imhist(frame2(:,:,2), 256));
        b1Hist = double(imhist(frame2(:,:,3), 256));
        pHash_of_frame2 =
Perceptual_hash_algorithm(strcat(img_dir,imglist(i +
step).name));
    end

    img_diff(i)= sum(abs(rHist-r1Hist) +
abs(gHist-g1Hist) + abs(bHist-b1Hist))/3;

    hammingWeight = bitxor(pHash_of_frame1,
pHash_of_frame2);%哈希权重

    hammingDistance(i)=sum(hammingWeight(:)==1);%哈希距
离


    %显示运行进度
    clc;
    X = ['Process: ', num2str(int64((i/frame_len)*100)),
'%'];
    disp(X)
```

```
end

%剔除离群值求平均，算出阈值
Threshold = trimmean(img_diff,0.1)* 11 ;

for i = 1 : step : frame_len
    tmp = 0;
    if (i >  frame_len - step)
        if( 0 == tmp)
           imwrite(read(video, i), ['.\1-keyframe\'
num2str(i) '.jpg']);
           tmp = tmp + 1;
        end
    end


    if( img_diff(i)>Threshold && hammingDistance(i) >
hammingDistanceThreshold )
      imwrite(read(video, i), ['.\1-keyframe\'
num2str(i) '.jpg']);
    end;

end


% function: Perceptual_hash_algorithm
% img_dir : directory of input  image
function pHash= Perceptual_hash_algorithm( img_dir )
    dIdx = zeros(1, 64);
    mean = 0.0;
    k = 1;
    pHash = zeros(1,64);

    img=imresize(rgb2gray(imread(img_dir)),[8 8]);
    dctCoeff = dct2(img);

    for ii = 1 : 8
        for jj =1 : 8
           dIdx(k) = dctCoeff(ii,jj);
           mean = mean + dctCoeff(ii,jj)/64;
           k = k + 1;
        end
    end
```

```
    for ii = 1 : 64
        if (dIdx(ii) >= mean)
            pHash(ii) = 1;
        else
            pHash(ii) = 0;
        end
    end
end
```

2、图（以下为从源视频筛选出来的部分关键帧）



3、表