

高分辨率灰度图像的快速多分辨率着色

胡 伟 秦开怀

(清华大学计算机科学与技术系 北京 100084)

摘 要 灰度图像的计算机快速着色有着广泛的应用前景,尤其是对老照片或者旧影片的彩色化处理.基于最优化的着色方法利用少量的人工标色,就能够得到好的着色效果,但是该方法在处理较高分辨率图像时需要消耗大量的存储和计算资源,有时甚至不能得到最后的结果.在最优着色方法的基础上,文中利用图像着色的特征,提出了使用最优着色的金字塔模型进行多分辨率着色处理的新方法,始终控制矩阵计算的规模,能够很快地得到视觉效果一致甚至更好的着色效果.实验结果也证明新方法无论在计算时间、存储需求还是最后着色的质量上,尤其是针对高分辨率图像,比原方法有了很大的改进.

关键词 计算机着色;灰度图像;多分辨率;边缘检测;优化

中图法分类号 TP391

DOI号: 10.3724/SP.J.1016.2009.01062

Fast Multi-Resolution Colorization of High-Resolution Gray Images

HU Wei QIN Kai-Huai

(Department of Computer Science & Technology, Tsinghua University, Beijing 100084)

Abstract Colorization could be used to increase the visual appeal of old black and white photos and movies. By means of colorization using optimization, one can add colors automatically to a monochrome image or movie after annotating the image with a few color scribbles, based on a simple premise—neighboring pixels in space that have similar intensities should have similar colors. However, the previous method may fail to deal with high-resolution gray images well because of its quite large size of computation. In this paper, the authors utilize features of the image colorization as an optimization problem to overcome this limitation. An image pyramid model with the edge detection is presented to reduce the computation size for the colorization. As a result, the new method can colorize high-resolution gray images to advantage in computing speed and visual effects.

Keywords colorization; gray images; multi-resolution; edge detection; optimization

1 引 言

计算机着色(colorization)首先是 Wilson Markle 于 1970 年提出的概念,即通过计算机处理能够在

原来的黑白照片、电影或者电视中加上彩色效果.最常用的计算机着色方法是利用图像分割技术(segmentation)将一幅图像分割成许多区域,然后每个区域赋上不同的颜色.但是现有图像分割技术很难准确分割图像的不同区域,使得该方法在处理

一般的图像时都会出现问题,尤其是存在复杂边缘和颜色平缓过渡的情况时^[1].

2002 年, Welsh 等人提出了一种半自动的着色方法^[2]. 该方法根据一幅彩色参考图像的信息给灰度图像进行着色,主要的依据是着色像素以及周围像素灰度值与参考图像的匹配. 在此基础上,采用分块多分辨率着色的改进算法被应用到人体脸部灰度图像着色中^[3]. 因为必须寻找到非常合适的参考图像,该方法的应用受到很大的局限. 2004 年, Levin 等人提出了使用优化进行计算机着色的方法^[1]. 优化着色方法基于一个简单的假定:相邻的像素之间如果有相似的灰度值,那么就会有相似的颜色值. 利用这一假定,计算机着色的过程转化为一个优化求解的过程,而人工进行少量的标色工作则成为优化求解的约束条件. 实验证明该方法能够在少量人工标色的情况下得到非常好的计算机着色效果. 后续针对着色的研究更多着眼于特殊图像的处理,例如卡通、手绘漫画^[4]等.

虽然优化着色方法取得了很大的进步^[5],但是在实际使用过程中仍然会遇到很多问题,尤其是图像分辨率比较高的情况下,计算时间和着色质量都不能得到很好的保证. 在此工作基础上,我们提出了多分辨率优化着色方法,很好地解决了该方法所遇到的问题,使得采用优化技术进行计算机着色的方法能够真正得到广泛应用.

本文将在第 2 节中介绍我们实现的多分辨率优化着色方法的理论基础、基本思想、具体算法和性能分析等;第 3 节将给出实验结果和数据比较;第 4 节进行总结和展望.

2 多分辨率优化着色方法

本节将首先对优化着色方法进行简要介绍,在此基础上进一步描述多分辨率优化着色方法的基本思想和具体算法,最后对新算法的性能做一个简单的分析.

2.1 优化着色方法(Colorization Using Optimization)

优化着色方法在视频处理中常用的 YUV 颜色空间进行图形着色^[1],其中 Y 表示单一的灰度(亮度)通道,U 和 V 表示色度通道^[6]. YUV 颜色空间与常用的 RGB 颜色空间可以通过以下公式进行换算:

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.147 & -0.289 & 0.436 \\ 0.615 & -0.515 & -0.100 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix},$$

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1.00 & 0.00 & 1.14 \\ 1.00 & -0.39 & -0.58 \\ 1.00 & 2.03 & 0.00 \end{bmatrix} \begin{bmatrix} Y \\ U \\ V \end{bmatrix} \quad (1)$$

灰度图像实际上就是当像素颜色用 YUV 颜色空间表示时,所有像素 U、V 值都为零的图像. 计算机着色过程可以表示为:通过对一组灰度的输入值 $Y(x, y, t)$ 进行处理后,输出对应的色度值 $U(x, y, t)$ 和 $V(x, y, t)$. 我们用 $Y(r)$ 代替 $Y(x, y, t)$ 来表示一个特定像素 r 的灰度值,用 $U(r)$ 、 $V(r)$ 表示像素 r 的色度值. 优化着色方法假定相邻的像素之间如果有相似的 Y 值,那么就会有相似的 U 和 V 值,那么相邻像素之间颜色值差距最小的情况就是最佳的着色情况. 因为 U、V 相对独立,求解最佳的 U 值分布可以表示为最小化

$$J(U) = \sum_r (U(r) - \sum_{s \in N(r)} w_{rs} U(s))^2 \quad (2)$$

的过程. 式(2)中 $N(r)$ 表示与像素 r 位置相邻的像素集合, s 为该集合中的一个像素. 对于一幅图像,两个像素相邻只是在位置上的连接,即像素 r 周围一圈 8 个像素(r 在图像边缘时相邻像素会减少),而对于视频序列则需要考虑相邻帧之间像素关系. 式(2)中 w_{rs} 是一个和为 1 的权值函数,表示像素 r 与 s 的亮度值 $Y(r)$ 和 $Y(s)$ 的相似程度,可以使用的权重函数包括图像分割中常用的

$$w_{rs} \propto e^{-(Y(r)-Y(s))^2/2\sigma_r^2} \text{ 或者}$$

$$w_{rs} \propto 1 + (Y(r) - \mu_r)(Y(s) - \mu_r)/\sigma_r^2,$$

其中 σ_r 和 μ_r 分别表示像素 r 以及周围相邻像素的 Y 值的方差和平均值. 式(2)的最小化求解过程实际上是一个稀疏线性系统的最小平方求解过程,人工标色就是增加求解的约束条件. Levin 方法使用 Matlab 的稀疏线性系统最小平方问题求解方法进行静态图像着色,而图像序列着色则使用多重网格方法求解^[7]. 当计算完 U 值后再使用相同方法计算 V 值,就得到了最后的彩色图像.

根据以上介绍可以看出,最小化 $J(U)$ 的过程导致求解一个大规模稀疏线性系统. 所有像素将组成一个大规模的稀疏矩阵,当矩阵达到一定规模时,求解将非常耗时. 800×600 或者 1024×768 的图像如今非常常见,这就意味着矩阵规模将达到几十万甚至上百万的量级. 对于普通 PC 而言,这种规模的矩阵求解经常是难以承受的. 而这只是考虑静态图像的情况,一旦需要进行视频着色,计算规模更会急剧增加. 虽然可以使用多重网格方法来缓解计算能力的问题,但多重网格法同样需要更多的时间计算

超大规模矩阵,并且计算过程中可能会出现计算精度降低的问题,影响最终的着色质量.

2.2 基本思想

对于一般的优化着色方法(后文简称为 Levin 方法),人工标色是一个增加约束条件的过程,但人工标色实际上是一个像素 U 、 V 值的赋值过程. 假设需要对一幅 100×100 的灰度图像着色,按照式(2)可以建立一个 10000×10000 规模的稀疏矩阵进行优化求解. 如果标色像素达到 9999 个,而未标色像素只有一个,那么按照 Levin 方法就会在以上大规模矩阵的基础上添加 9999 个约束. 如果将标色看作是赋值,即减少未知量的过程,那么 10000 个像素中就存在 9999 个已知的,代入式(2)中,最后得到一个 1×1 规模的矩阵进行优化求解. 前后两种方式虽然相似,但是求解的计算量却相差非常大.

Levin 方法之所以使用添加约束是因为一幅图像中人工标色的像素只占非常小的比例,这样两种方法的计算量差距很小,而后一种方式构造求解过程会相对复杂些. 但是我们是否有可能将标色的像素转化成绝大多数呢? 答案是可行的. 下面首先介绍新方法的基本根据和原则:

- (1) 彩色图像中 U 、 V 值显著改变的临界线,也是 Y 值显著改变的临界线;
- (2) 同类色中相近颜色的区别在视觉上主要是 Y 值作用的结果;

- (3) 灰度图像的尺寸越小,求解速度越快;
- (4) 相同尺寸的灰度图像,经过标色的像素越多,求解速度越快.

第 1 个原则是 Levin 方法假定的自然推广,既然相邻像素相似的灰度意味着相似的 U 和 V 值,那么不相似的 U 和 V 值也就意味着不相似的 Y 值. 因为视觉上对 U 、 V 变化的不敏感性,我们可以假定同类色中颜色的区别主要是 Y 值作用的结果,这就是第 2 个原则. 图 1 很好地说明了原则 2 的正确性. 图 1(a)为一片草地的照片,每个像素都有着不同的 Y 、 U 和 V 值. 我们在照片中随机抽取一点,然后将图 1(a)中所有像素保留 Y 值,而 U 和 V 值统一赋值为抽取点的 U 和 V 值. 进行两次检测后得到图 1(b)和图 1(c)的效果. 虽然所有像素的 U 和 V 值都完全相同,但因为保留了 Y 值,在视觉效果上后面两图与原始图像的差别却并不很明显,这说明眼睛对 U 和 V 的敏感程度远低于 Y .

第 3 和第 4 个原则源于矩阵规模对求解速度的影响. 当图像尺寸越小的时候,矩阵规模越小;根据上面分析,标色的像素越多,未知的像素越少,矩阵规模也越小.

基于原则 3,低分辨率图像的计算机着色速度快. 很自然地,我们可以将高分辨率的灰度图像和标色图像转变到低分辨率下进行快速着色. 如图 2 所示,(a)为灰度图像和标注图像的合成,图像分辨率

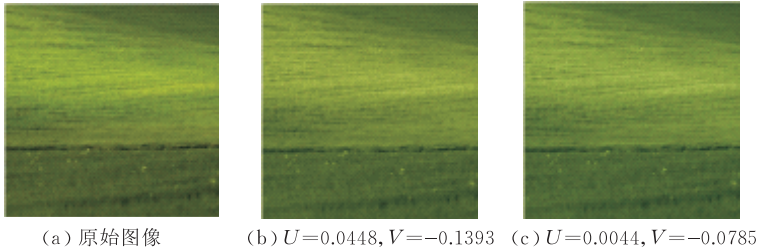


图 1 原则 2 的示意图

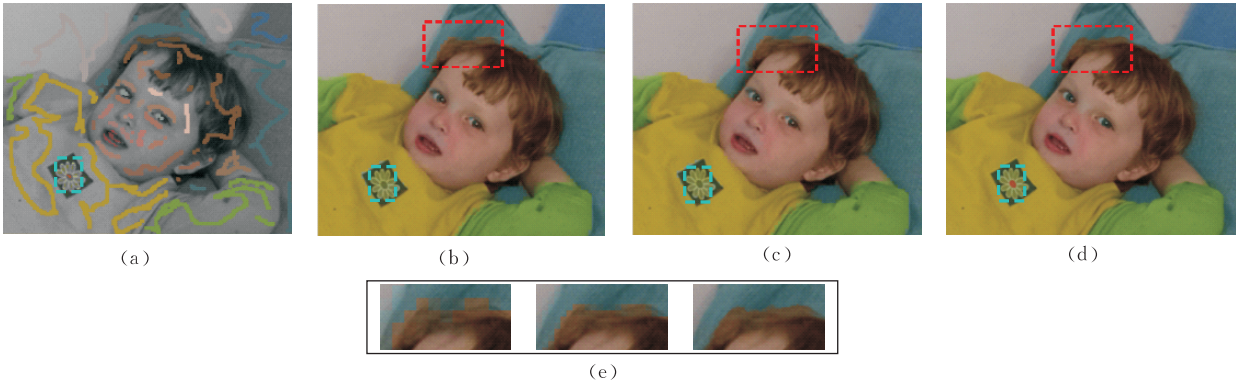


图 2 直接使用低分辨率着色效果比较

为 320×256 ; (b) 是在原图 1/8 分辨率下使用 Levin 方法进行着色后, 按照缩放比例将计算得到的 U 、 V 值直接赋给初始灰度图像得到的结果, 整个优化求解过程耗时约 0.101s; (c) 是在原图 1/4 分辨率下着色后直接赋给初始灰度图像得到的结果, 耗时约 0.423s; (d) 是在原图 1/2 分辨率下着色后直接赋给初始灰度图像得到的结果, 耗时约 1.716s; (e) 中从左至右为 (b)、(c) 和 (d) 中用红色虚框标注的区域. 以上计算结果在 Windows XP 上使用 Matlab 6.5 得到, 计算机 CPU 为酷睿 2 双核 2.8GHz, 内存 2GB RAM.

图 2(e) 中可以看到, 在不同颜色的交界区域出现了错误着色, 而且不同颜色的交界区域正好属于灰度图像边缘, 这与原则 1 相符. 同时注意图 2 中蓝色虚框标注的区域: (a) 中标的红色在很低分辨率情况下被剔除, 以至 (b) 和 (c) 中不存在所标红色的效果. 总之, 并不能简单地使用低分辨率着色来取代高分辨率着色的效果. 直接使用低分辨率着色结果的问题主要在两方面: (1) 不同颜色交界的边缘区域的像素会出现错误着色; (2) 低分辨率着色可能会使某些人工标色失效.

虽然直接使用低分辨率着色结果出现很多问题, 但对比 (b)、(c) 和 (d), 可知大部分非颜色交界区域的像素在使用低分辨率着色结果时, 可以达到比较满意的着色效果, 这正好说明了原则 2 的正确性. 同时根据原则 4, 标色的像素越多, 则计算过程也会越快. 那么能否利用低分辨率下比较满意的着色效果, 使之作为高分辨率下的标色参与 Levin 方法的

计算? 如果可行的话, 虽然高分辨率下图像尺寸增大, 但是利用低分辨率的着色结果使得标色的像素更多, 计算速度也会得到提高. 为此, 我们提出解决此问题的新方法的基本思想: 构造灰度图像和标色图像的金字塔模型, 使用 Levin 方法对低分辨率进行着色, 然后将低分辨率着色结果放大到高分辨率, 去除颜色交界边缘的有害信息之后与高分辨率的标色图像进行合并, 形成新的标色图像参与着色计算; 同时始终控制参与 Levin 计算的矩阵规模, 提高整个计算的速度. 下面对该方法进行详细介绍.

2.3 算 法

我们的新方法使用了如下算法:

1. 假设图像 G 和 I 表示高分辨率灰度图像和人工输入的标色图像, T 表示需要输出的彩色图像, 根据边缘提取方法得到 G 的边缘图像 E . Levin 方法改进后的优化计算过程用 $\text{newLevin}(G, I)$ 表示, 即 $T = \text{newLevin}(G, I)$, newLevin 中标色将是一个减小未知量的过程, 而不是添加约束的过程;
2. 令 $i = L, L-1, \dots, 1$ 表示 G 和 I 对应的多分辨率金字塔模型^[8]的层次编号 (i 的值越大表示分辨率越低);
3. G_i 、 E_i 和 I_i 表示第 i 层的灰度图像、边缘图像和人工标色图像;
4. 从分辨率最低的 $i = L$ 开始, 计算 $T_L = \text{newLevin}(G_L, I_L)$;
5. 根据 $i+1$ 层得到的结果 T_{i+1} 构建第 i 层的着色图像 T_i . 计算公式如下:
$$T_i = \text{newLevin}(G_i, T_{i+1} - E_i + I_i) \quad (3)$$
6. 当 $i=1$ 时, $T=T_1$, 计算过程结束.

从上述计算过程可以看出新方法使用低分辨率的着色结果构建高分辨率的输入. 为解决上面提到

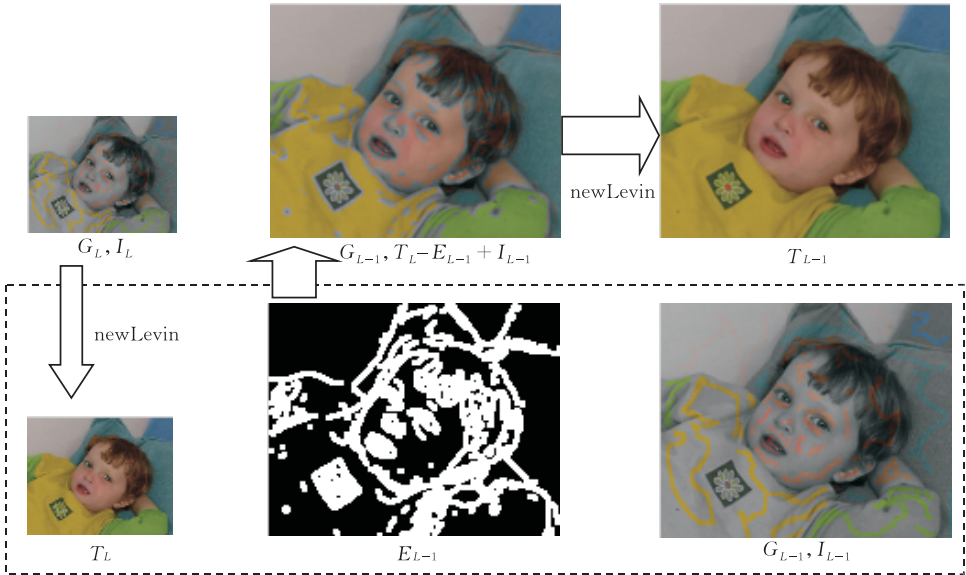


图 3 新算法实现过程示意图

的问题,关键的步骤体现在式(3)中. 针对第一个问题,我们在低分辨率着色结果上剔除了边缘部分的着色,即 $T_{i+1} - E_i$;而针对第二个问题,我们在高分辨率下继续添加人工标色,就得到了高分辨率下最终的标色图像 $T_{i+1} - E_i + I_i$;在该图像中,仅有少部分像素未被标色,根据原则 4,将极大地提高计算速度. 分析以上过程可知,整个过程中始终不需要解大规模的矩阵运算. 图 3 清楚地说明了算法的实现过程. 图 3 中,观察在最低分辨率的 L 层的输入 (G_L, I_L) 可知未标色像素占据了图像的绝大部分,但是此时图像分辨率很低,Levin 方法求解非常快;而 $L-1$ 层的输入 $(G_{L-1}, T_L - E_{L-1} + I_{L-1})$ 中,虽然图像分辨率提高 4 倍,但是标色的像素已经占据图像的绝大部分(对于普通图像一般在 90% 以上),求解的速度甚至比 L 层更快. 注意在 L 层,图 2 中蓝框标注的红点因为分辨率过低而消失,但是在 $L-1$ 层增加 I_{L-1} ,使得该标注重新参加运算. 依此类推下去,最后能以很快的运算速度得到非常高分辨率图像的准确着色结果.

2.4 性能分析

下面对新算法的性能进行粗略的分析. 假设图像 A 的初始像素数为 M . 同时假设求解稀疏线性最小平方问题的计算速度与稀疏矩阵规模成线性关系,即求解 $(2N \times 2N)$ 矩阵需要时间为 $N \times N$ 矩阵的 4 倍. 如果 A 中存在比例为 ξ 的像素被人工标色,求解完整的 M^2 矩阵所需时间为 T ,那么使用 newLevin 方法,对 A 直接进行着色的计算时间为 $(1-\xi)T$.

使用新方法后,对该图像作 L 层的金字塔模型分解,层次编号为 $L, \dots, 1$,那么第 i 层的像素数为 $M/4^{i-1}$. 假设边缘提取检测到的像素占整个图像像素的比例为 ω (实际分辨率增大时 ω 值会减小). 在理想情况下,第 L 层的人工标色像素比例等于 ξ ,而从第 $L-1$ 到第 1 层中,标色的像素比例为 $1-\omega$. 所以计算 newLevin(G_L, I_L) 的时间为 $(1-\xi)T/4^{L-1}$,而计算 $T_i (i=L-1, \dots, 1)$ 的时间为 $\omega T/4^{i-1}$. 那么整个计算的时间总和为

$$T_{\text{total}} = \frac{(1-\xi)T}{4^{L-1}} + \sum_{i=L-1}^1 \frac{\omega \cdot T}{4^{i-1}} \\ = \left(\frac{(1-\xi)}{4^{L-1}} + \frac{4}{3} \left(1 - \frac{1}{4^L} \right) \omega \right) \cdot T \quad (4)$$

从式(4)可知,当 L 趋于无穷时,

$$T_{\text{total}} = \lim_{L \rightarrow \infty} \left(\frac{(1-\xi)}{4^{L-1}} + \frac{4}{3} \left(1 - \frac{1}{4^L} \right) \omega \right) \cdot T = \frac{4\omega}{3} T \quad (5)$$

即在理想情况下,只要边缘检测提取出来的图像边缘像素不超过整个图像像素的 75%,新方法将比直接使用 newLevin 方法的速度更快. 而基于统计规律,一幅普通图像的边缘像素比例一般不超过 20%,对于分辨率很高的图像,则这一比例往往低于 10%,那么使用新方法将会比直接使用 Levin 方法提高 10 倍以上的速度. 在实际计算中,当分辨率增大时,构建求解稀疏矩阵的时间代价会逐渐增大,但因为同时 ω 值会减小,性能分析在总体上符合实际计算结果,后面的实验结果证明了这点.

上述算法中,生成 E_i 必须使用边缘检测算法. 如今的边缘检测算法很多,如梯度算子、方向算子、拉普拉斯算子和 Canny 算子等. 虽然这些方法在高精度图像边缘检测方面尚不能做到很完美,但即使是最基本的算法也能比较完整地描绘出图像的边缘轮廓,这对于新方法要求生成的 E_i 已经足够了;而且因为 T_{i+1} 中一个像素将影响到第 i 层的多个像素,如果使用非常精确的边缘检测算法,在 $T_{i+1} - E_i$ 时可能保留错误的着色信息. 既然新方法本来就要求粗糙的边缘检测方法,所以新方法中只简单地使用了局部灰度值比较来提取边缘,这样处理的另一个好处是计算非常迅速,一幅 1024×768 的图像做一次边缘检测的时间也在 0.5s 左右,在整个着色过程中所占时间比例非常小. 对于某些特殊的图像,使用普通的边缘检测方法很难得到令人满意的结果. 例如当图像有很多噪音时,式(3)中的 ω 值增大很多,会影响新方法的计算性能. 为了消除噪音影响,可以使用基于小波的边缘检测等许多改进算法.

3 实验结果和数据比较

我们在 Matlab 上实现了本文介绍的新方法,以下所有测试数据均在操作系统为 Windows XP Professional、CPU 为酷睿 2 双核 2.8GHz 处理器、系统内存 2GB 的普通 PC 上获得,使用的运算平台为 Matlab 6.5 Release 13.

图 4 是使用新方法和 Levin 方法根据完全相同的输入进行着色的两对结果图像. 从左至右分别为初始人工标色图像、新方法着色图像和 Levin 方法着色图像. (a) 的图像分辨率为 320×256 , (b) 的图像分辨率为 768×576 . 使用新方法时,图 4(a) 金字塔模型为 3 层, (b) 中金字塔模型为 4 层,即最低分辨率分别为初始图像的 1/8 和 1/16. 因为图 4(b) 的分辨率较高,Levin 方法使用 Matlab 已经无法计算

出结果,所以使用了 Levin 提供的多重网格求解库 (C++实现) 进行计算. 可以看出改进后的新方法在视觉效果上已经接近甚至超过 Levin 方法,如(b)中左侧红框标注蓝色天空的着色效果,Levin 方法

存在不自然的色彩过渡,还有左侧紫色边框标注的白色云彩,Levin 方法比新方法丧失了更多的细节. 最主要原因在于直接使用 Levin 方法求解,矩阵规模过大,求解精度上会存在问题.

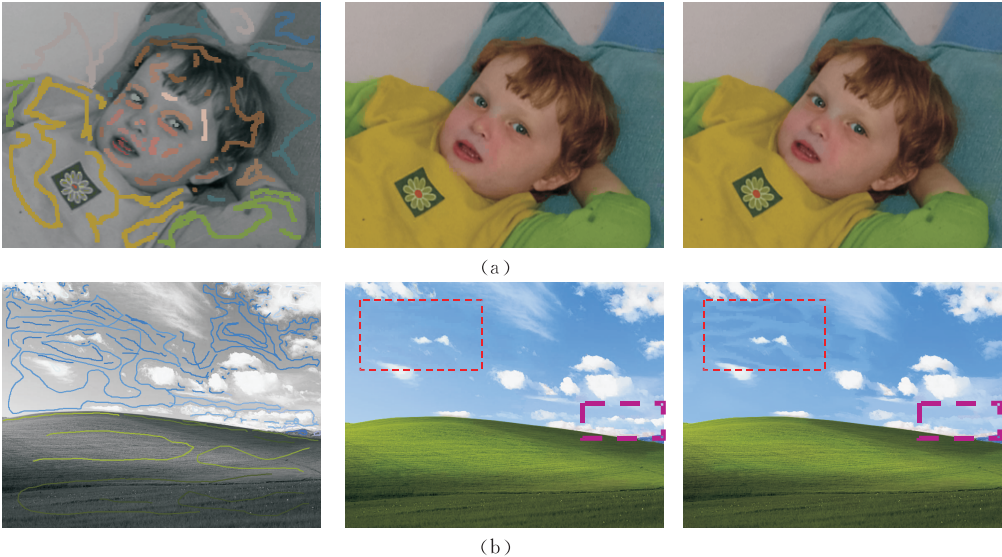


图 4 新方法与 Levin 方法的着色效果比较

表 1 中列举了图 4 中两种着色方法的测试数据. 表 1 中第 1 和第 2 部分是 Matlab 实现的新方法计算图 4(a)和图 4(b)的测试数据,可以看到未知像素的比例随着分辨率的提高逐渐减少;表 1 中第 3 部分是 Levin 方法分别使用 Matlab 和多重网格连接库计算图 4(a)和图 4(b)的测试数据,可以看到使用 Matlab 已经无法计算 768×576 分辨率的图像. 对比这些数据可以看出,新方法在计算速度上比 Levin 方法有非常明显的优势. 当图像分辨率达到 768×576 时(图 4(b)),Matlab 实现的 Levin 方法

已经无法计算出最后结果,而使用 Matlab 实现的新算法却能在 5s 以内完成计算. Levin 方法使用多重网格方法计算时,虽然能够得到最后结果,但是可能会出现计算精度上的不足,造成最后的结果出现误差,图 4(b)中 Levin 方法出现的着色错误就说明了这一问题.

下面我们以附图 1 为例进一步说明新方法在计算速度上的优势. 图中原灰度图像分辨率为 1024×768 . 图 5 给出了该原始图像在不同分辨率下使用不同方法的计算速度曲线. 计算中新方法使用的金字塔模型的层数为 4. 图 5 的横坐标为输入的灰度图

表 1 新方法和 Levin 方法的运算数据比较

(1) 图 4(a),新方法计算数据		
	未知像素比例/%	Matlab 计算时间/s
第 3 层	78.48	0.235
第 2 层	23.64	0.360
第 1 层	12.12	0.782
总和		1.377
(2) 图 4(b),新方法计算数据		
	未知像素比例/%	Matlab 计算时间/s
第 4 层	88.01	0.031
第 3 层	8.20	2.031
第 2 层	5.87	0.516
第 1 层	3.42	1.453
总和		4.031
(3) Levin 方法计算数据		
	Matlab 计算时间/s	MultiGrid 计算时间/s
图 4(a)	5.656	2.140
图 4(b)	无法计算	11.750

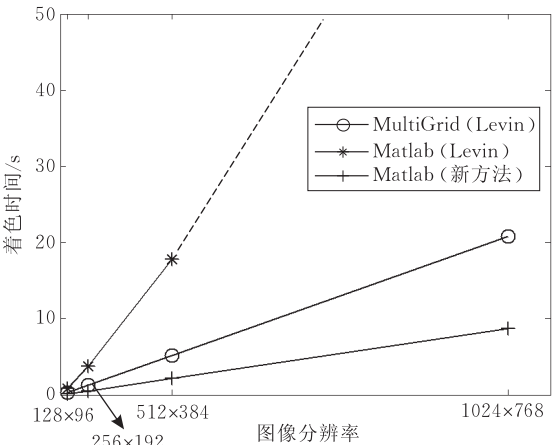


图 5 计算速度曲线示意图

像分辨率,纵坐标表示着色的时间(s).图中*线表示使用 Matlab 实现 Levin 方法求解的计算时间曲线,其中虚线部分表明已经无法计算该分辨率下的图像;o线表示 Levin 方法使用多重网格求解的计算时间曲线;+线表示使用 Matlab 实现新方法求解的计算时间曲线.从图可知新方法在 Matlab 下计算速度达到 Levin 方法 Matlab 计算速度的 10 倍以上.如果使用 C++实现新算法的多重网格方法,那么对于高分辨率灰度图像着色有望达到实时.

4 总结和展望

本文在 Levin 方法的基础上,对灰度图像着色提出一系列改进方法. Levin 方法第一次使用优化方法解决着色问题,但是它并不能得到广泛的应用.新的着色方法结合多分辨率以及边缘检测等图像处理的常用方法,极大提高了灰度图像的着色速度,减少了算法占用的计算和存储资源,突破了原有方法处理速度慢和难以处理高分辨率图像的限制,使得基于优化的灰度图像着色成为真正实用化的技术方法,可以在黑白照片和黑白电视电影的彩色化工作中发挥巨大的作用.附图 2 是使用本方法对毛主席经典黑白照片进行着色的结果^①.

本方法虽然在 YUV 颜色空间进行运算,但是同样可以使用 NTSC 或者是 Lab 颜色空间.因为这三种颜色空间均为一个灰度(亮度)通道和两个颜色通道的组合.同时 2.2 节中介绍的新方法基本根据和原则同样适用于 NTSC 和 Lab 颜色空间.如果在后两种颜色空间中进行运算,只需要将公式(1)中相应颜色空间与 RGB 空间的转换关系进行调整即可.因为 YUV 等颜色空间在本方法中只是运算的中间平台,故而使用不同的颜色空间对着色结果不会产生视觉上的明显差异.因为 YUV 或者 NTSC 颜色

空间与 RGB 颜色空间的转换相比 Lab 颜色空间更加简单,所以我们在计算时使用 YUV 颜色空间.

图像可以作为纹理在 GPU(Graphics Process Units)中进行处理,而整个方法实现中的金字塔模型构建、边缘检测、多重网格以及最小平方问题求解都已经能够在 GPU 中完成.同时,新方法始终控制了矩阵规模,使得整个新方法的处理过程完全可以在 GPU 中实现,最终使得高分辨率图像或者视频着色达到实时.这将是我們下一步的工作.

参 考 文 献

- [1] Levin A, Lischinski D, Weiss Y. Colorization using optimization//Proceedings of the ACM SIGGRAPH 2004. Los Angeles, California, USA, 2004: 689-694
- [2] Welsh T, Ashikhmin M, Mueller K. Transferring color to greyscale images. ACM Transactions on Graphics, 2002, 21(3): 277-280
- [3] Zhao Guo-Ying, Li Hua. Journal of Computer-Aided Design & Computer Graphics, 2004, 16(8): 1051-1056(in Chinese) (赵国英, 李华. 人体脸部灰度图像上色的改进算法. 计算机辅助设计与图形学学报, 2004, 16(8): 1051-1056)
- [4] Qu Y, Wong T T, Heng P A. Manga colorization. ACM Transactions on Graphics, 2006, 25(3): 1214-1220
- [5] Liron Y, Guillermo S. Fast image and video colorization using chrominance blending. IEEE Transactions on Image Processing, 2006, 15(5): 1120-1129
- [6] Jack K. Video Demystified. 3rd Edition. Amsterdam, The Netherlands: Elsevier Science & Technology, 2001
- [7] Press W, Teukolsky S, Vetterling W, Flannery B. Numerical Recipes in C: The Art of Scientific Computing. Cambridge, UK: Cambridge University Press, 1992
- [8] Hertzmann A, Jacobs C, Oliver N et al. Image analogies//Proceedings of the ACM SIGGRAPH 2001. Los Angeles, California, 2001: 327-340



HU Wei, born in 1979, Ph. D., lecturer of Department of Computer Science and Technology, Beijing University of Chemical Technology. His research interests focus on interactive global illumination, and real-time rendering.

QIN Kai-Huai, born in 1958, Ph. D., professor of Department of Computer Science and Technology, Tsinghua University. His research interests include computer graphics, computer aided geometric design, physics-based geometric modeling, wavelets, image processing and visualization, surgical planning and simulation, virtual reality and CAD/CAM.

① 黑白照片来源于新华网《纪念毛泽东诞辰 110 周年》专题
http://www.xinhuanet.com/newscenter/mzd/index.htm

Background

Computer-assisted colorization is very attractive for automatically colorizing gray photos and movies. It is still difficult to colorize targets quickly and accurately without users' intervention, although many researchers have achieved great progress in colorization. Methods that utilize image segmentation or region tracking cannot be performed reliably when there exist fuzzy or complex region boundaries in the images to be colorized. Transferring colors from a reference color image to the target gray image has also been used for colorization, however, it is hard to find such a satisfying reference image for any gray image. Colorization using optimization can

obtain high-quality colorizations by solving an optimization problem, with a few color scribbles to annotate the image, but it also has the limitation on its considerable computation cost.

Based on the optimization colorization technique, a multi-resolution colorization approach making use of a pyramid model with edge detection is presented to deal with large-size images and movies quickly. The experiments and comparisons in this paper show that the new multi-resolution approach achieves much better results than previous techniques in both quality and efficiency.



附图 1 新方法着色结果,分辨率为 1024×768



附图 2 毛主席经典黑白照片着色