**基于SIMD与多核编程的电子相册的设计与实现**

作者：崔悦 李亮

学号：3118311046 3118311015

指导教师：朱利教授

2018年11月

摘 要

传统的电子相册淡入淡出的实现，对图片按像素进行循环串行处理，当图片分辨率较高时，像素数量可达到百万级别，传统的方法效率比较低，融合速度较慢，甚至会出现视觉感受不流畅的问题。

本文的主要工作是设计和实现一个电子相册应用，主要功能是读取对应路径下图片进行融合处理，以淡入淡出的效果进行显示。文中分析了传统图像融合方法中存在的问题，用不同方法进行加速融合并进行了评测和对比。

本文采用opencv开源库图像处理技术和SSE和多核编程技术作为基本技术，使用了vs2017集成开发环境搭建了系统编码环境，完成了基于SSE和多核编程的电子相册的编码，并分别对此应用进行了功能测试和性能测试。测试结果说明，应用满足所有业务需求，实现了所有功能，达到了性能要求。

**关 键 词**：SEE；多线程；电子相册；淡入淡出

**论文类型**：应用研究

目录

[摘 要 2](#_Toc530687377)

[1 背景和意义 4](#_Toc530687378)

[2系统主要相关技术和基础理论 4](#_Toc530687379)

[2.1 OpenCV介绍 4](#_Toc530687380)

[2.2 SSE介绍 5](#_Toc530687381)

[2.3多核编程介绍 5](#_Toc530687382)

[3基于SSE和多核编程应用的分析与实现 6](#_Toc530687383)

[3.1应用需求描述 6](#_Toc530687384)

[3.1.1功能需求描述 6](#_Toc530687385)

[3.1.2性能需求描述 6](#_Toc530687386)

[3.2原理介绍与具体实现 6](#_Toc530687387)

[3.2.1 本节内容介绍 6](#_Toc530687388)

[3.2.2图像过渡的基本原理 6](#_Toc530687389)

[3.2.2图片过渡效果的四种实现 7](#_Toc530687390)

[3.2.3以上四种方法的性能评测和比较 9](#_Toc530687391)

[4结论与展望 11](#_Toc530687392)

# 1 背景和意义

开发一个能够高效进行图像融合的应用是非常有意义的，尤其是对于当期图像和视频分辨率日趋增大的情况。并且Intel SIMD指令集直到目前已经发展多代，从最初MMX寄存器只有64位到现在AVX达到512位，硬件的加速发展也推动着软件开发的发展，很多编程者对这些指令集在软件中的应用并不多，这也是一种资源的浪费。

# 2系统主要相关技术和基础理论

为了保证基于SSE和多核编程应用的顺利实现，并在实现过程中减少开发风险，提升代码的可移植行，增加应用的可靠性。本系统将采用opencv开源库作为图像处理基本技术，将opencv和SIMD与多线程编程结合起来，提高效率，加速图像的融合。

## 2.1 OpenCV介绍

OpenCV是一个基于BSD许可（开源）发行的跨平台计算机视觉库，可以运行在Linux、Windows、Android和Mac OS操作系统上。它轻量级而且高效——由一系列 C 函数和少量 C++ 类构成，同时提供了Python、Ruby、MATLAB等语言的接口，实现了图像处理和计算机视觉方面的很多通用算法。

OpenCV的优点

计算机视觉市场巨大而且持续增长，且这方面没有标准API，如今的计算机视觉软件大概有以下三种：

1、研究代码（慢，不稳定，独立并与其他库不兼容）

2、耗费很高的商业化工具（比如Halcon, MATLAB+Simulink）

3、依赖硬件的一些特别的解决方案（比如视频监控，制造控制系统，医疗设备）这是如今的现状，而标准的API将简化计算机视觉程序和解决方案的开发，OpenCV致力于成为这样的标准API。

OpenCV致力于真实世界的实时应用，通过优化的C代码的编写对其执行速度带来了可观的提升，并且可以通过购买Intel的IPP高性能多媒体函数库（Integrated Performance Primitives）得到更快的处理速度。右图为OpenCV与当前其他主流视觉函数库的性能比较。

## 2.2 SSE介绍

SIMD全称Single Instruction Multiple Data，单指令多数据流，能够复制多个操作数，并把它们打包在大型寄存器的一组指令集。

支持SIMD的处理器在音频解码、视频回放、3D游戏等应用中显示出优异的性能。

SIMD技术的大致发展脉络是

1996 MMX 64bit Registers

1999-2008 SSE-SSE4 128bit Register

2010 AVX 256bit Register

SSE(SSE(Streaming SIMD Extensions)是SIMD技术的一种，它包括70条指令，其中包含单指令多数据浮点计算、以及额外的SIMD整数和高速缓存控制指令，可同时对4个32位单精度浮点数进行运算处理。其优势在于更高分辨率的图像浏览和处理、更高精度和更快响应速度。

## 2.3多核编程介绍

通过对多核与单核多线程的比较可更清楚的解释多核编程。

首先介绍并行与并发：并行指两件（多件）事情在同一时刻一起发生；并发：两件（多件）事情在同一时刻只能有一个发生，由CPU快速切换，从而给人的感觉是同时进行

使用多线程来实现并行计算来缩短计算时间时，只有在多核CPU下才行，单核CPU下启用多线程最终总的计算计算一样，因为CPU在同一时间，只能服务于一个线程，

在单核CPU下运用多线程仅仅能实现快速响应用户的请求，避免因io或网络阻塞而导致界面停留卡顿。

# 3基于SSE和多核编程应用的分析与实现

## 3.1应用需求描述

### 3.1.1功能需求描述

1）相册获取

程序对相对路径下的若干相册进行读取(每个相册是一个文件夹，里面有若干相片)，相册数量理论上没有限制，每个相册相片数量理论上没有限制。

2) 相片融合

就单独一个相册来说，对其中相片按序进行两两融合并实现淡入淡出显示，要求视觉上没有突兀感。

3) 相册展示

每个相册有单独的窗口进行显示，可拖动大小和位置，方便进行布局。

### 3.1.2性能需求描述

预期融合速度接近传统方法的8倍：理论依据是由于采用128位寄存器进行多数据处理，每次可同时处理8个字节数据，相比较传统每次处理一个字节，。

## 3.2原理介绍与具体实现

### 3.2.1 本节内容介绍

电子相册的基本功能是将多幅图片轮流显示，并增加前后两张图片的过渡效果。其中，过渡效果的实现是本实验的重点，所以，本节的主要内容将分为以下几点进行展示：

1图像过渡的基本原理

2图片过渡效果的普通实现

3使用SSE4加速图片过渡效果的处理速度

4使用SSE4和多线程加速图片过渡效果的处理速度

5对图片过渡效果普通实现（即1）加速的另外一种方法

6以上四种方法的比较和总结

### 3.2.2图像过渡的基本原理

图像过渡效果的基本原理是将两幅图像相同位置上的RGB值进行加权求和，总权值为1，计算公式如下：

Result = (A-B)\*fade+B

其中，A代表A图片的对应像素RGB值，B代表B图片对应像素的RGB值，fade代表权重，Result代表两幅图像对应像素的融合值。本实验中为提高计算速度，将权重扩大128倍再除以128，其中除以128可用移位操作代替。公式变为：

Result = ((A-B)\*fade)>>7+B

>>7代表右移7位。

### 3.2.2图片过渡效果的四种实现

**1普通实现**

实现代码如下：

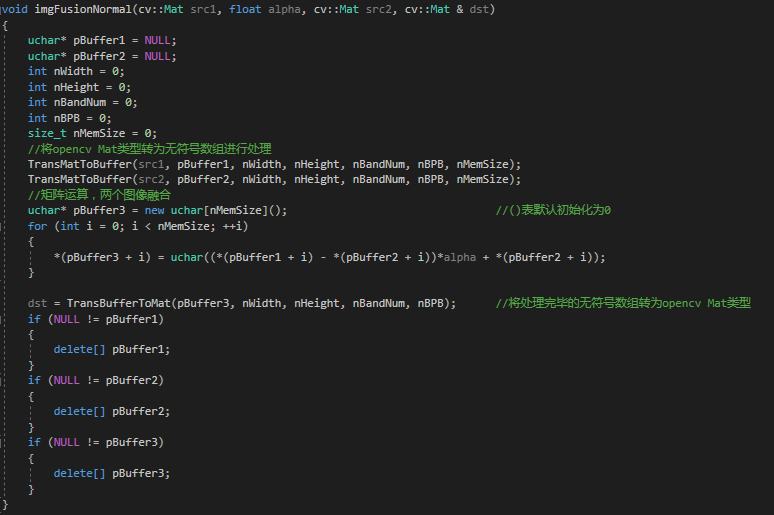


图1

函数使用两个opencv Mat类型的图像数据作为输入，对每一个像素值进行加权求和。融合效果的一个展示如下（使用的图片是两张高清照片）：



图2

**2对图片过渡效果普通实现（循环体改造）**

在进行SSE4实验中，发现循环体是同时对8个像素值进行加权求和，于是想到可以对普通实现进行修改，将循环体进行优化，每次循环同时对8个像素值进行加权求和。实现代码如下：

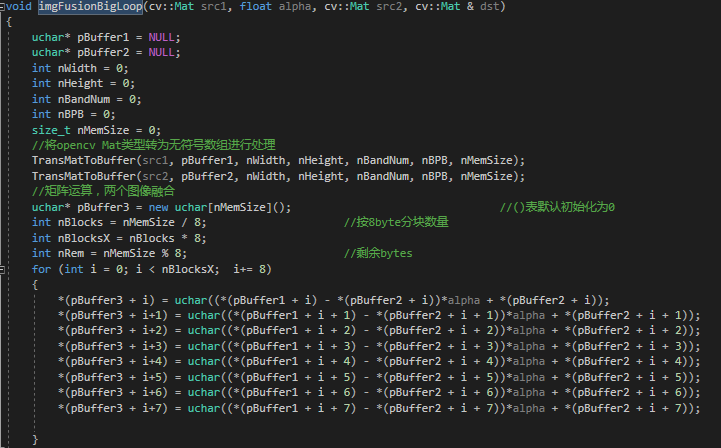


图3

可以看见，循环体结构的改变。

**3使用SSE4加速**

实现代码如下：

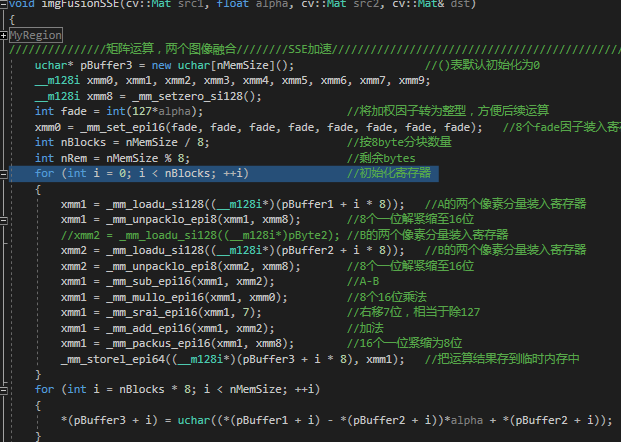


图4

在计算两幅图像的加权平均时，将两幅图像的八个RGB值一次装入128位寄存器，然后相减，在乘以权重，在做一次移位和加法操作，得到16位的RGB值，对16位的RGB值进行一次紧缩，得到8位的RGB值，最后将寄存器中的数据返回到相应的内存地址中进行展示。

**4使用SSE4和多线程加速图片过渡效果的处理速度**

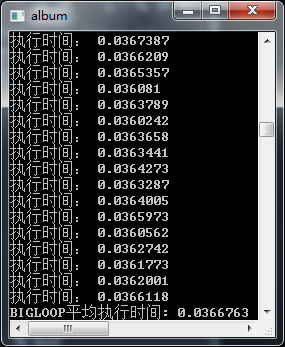
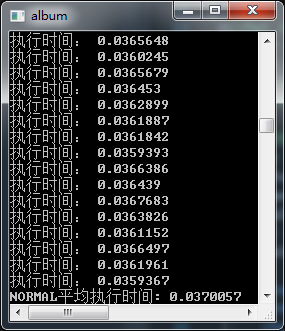
使用openMP进行多线程程序设计，在图4高亮行上面加上一句

#pragma omp parallel for

对for循环使用自动进行优化为多谢线程并行进行执行，可实现简单的多线程并行。另外，此种方法必须消除循环间的数据依赖，否则虽然能够加速运行，但是却得到错误的结果。

### 3.2.3以上四种方法的性能评测和比较

分别调用以上四种方法对两幅图像（1920x1080）的融合进行测试。在调用函数前后分别计时，以得到各个方法的实际运行时间（单位为秒）。为了消除由计算机状态影响导致的某一次执行时间的不稳定，所以对以上融合效果分别调用100次，求其平均作为性能度量的最终依据。实验结果如下：



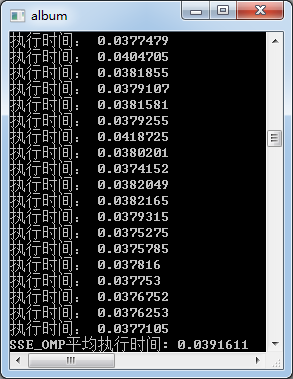
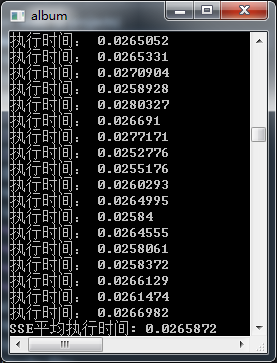


图5

根据实验结果可以看到，使用SSE4对图像过渡效果进行加速相较于普通方法能明显提高运行速度，使用多线程又能够提高性能，但是提高幅度较小。将循环体结构进行改造出乎意外地取得了最好的性能，分析其原因应该是循环体每次循环的执行都需要浪费较大的时间用于变量的重新赋值。

表1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 使用技术 | 普通实现(NORMAL) | 循环体改造(BIGLOOP) | SSE | SSE+OPENMP |
| 性能提升（以NORMAL为基准） | 1 | 0.90% | 28% | -5.80% |

# 4结论与展望

从以上的评测和比价可见，SSE和多线程都起到了加速的效果，其中SSE加速效果最明显，而使用openmp技术和循环体改造加速效果甚微，甚至起到了负面效果；这启发我们在编程过程中要有意识的编写可以并行处理的代码块，编写出局部性好的代码块，了解机器支持的并行处理指令集，发挥硬件加速的优势，从而开发出高效的软件产品。