CS 213,2001 年秋季

实验室作业 L4:代码优化

分配时间:10 月 11 日

截止日期:10 月 25 日晚上 11:59

Sanjit Seshia (sanjit+213@cs.cmu.edu) 是这项任务的负责人。

1 简介

本作业涉及优化内存密集型代码。图像处理提供了许多可以从优化中受益的函数示例。在本实验中,我们将考虑两种图像处理操作: 旋转,将图像逆时针旋转 90°图像。

, 和平滑,它“平滑”或“模糊”

在本实验中,我们将考虑将图像表示为二维矩阵M,其中Mi,j表示 M 的第(i, j) 个像素的值。像素值是红、绿和蓝的三元组 (RGB) 值。我们只会考虑方形图像。令N表示图像的行数（或列数）。行和列按照 C 风格从 0 到N − 1 进行编号。

给定这种表示形式,旋转运算可以非常简单地实现为以下两个矩阵运算的组合:

- 转置:对于每个(i, j) 对， Mi,j和Mj,i互换。
- 交换行:行i与行N − 1 − i交换。

这种组合如图 1 所示。

平滑操作是通过将每个像素值替换为其周围所有像素的平均值（在以该像素为中心的最大 3 × 3 窗口中)来实现的。考虑图 2。像素 M2[1][1] 和 M2[N-1][N-1] 的值如下所示:

$$M2[1][1] = \frac{\sum_{i=0}^{2}\sum_{j=0}^{2} M1[i][j]}{9}$$

$$M2[N-1][N-1] = \frac{\sum_{i=N-2}^{N-1}\sum_{j=N-2}^{N-1} M1[i][j]}{4}$$

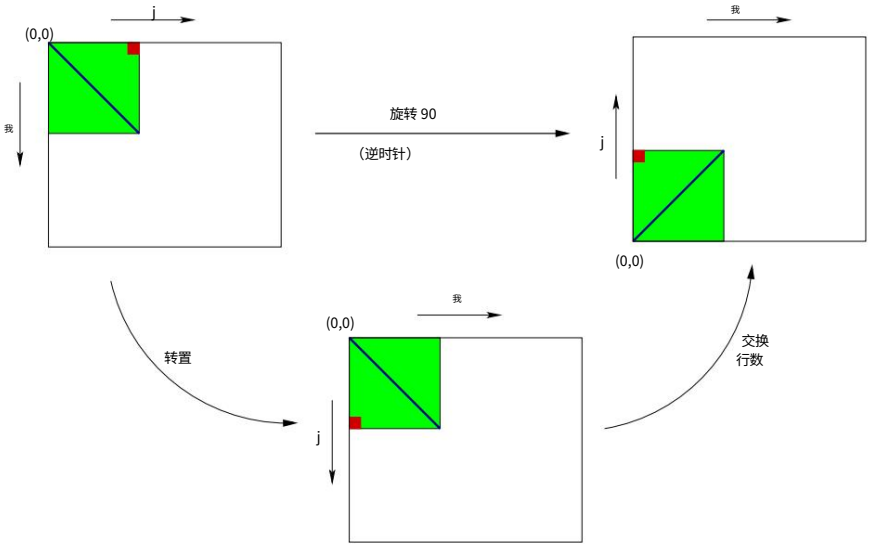

图 1:将图像逆时针旋转 90°

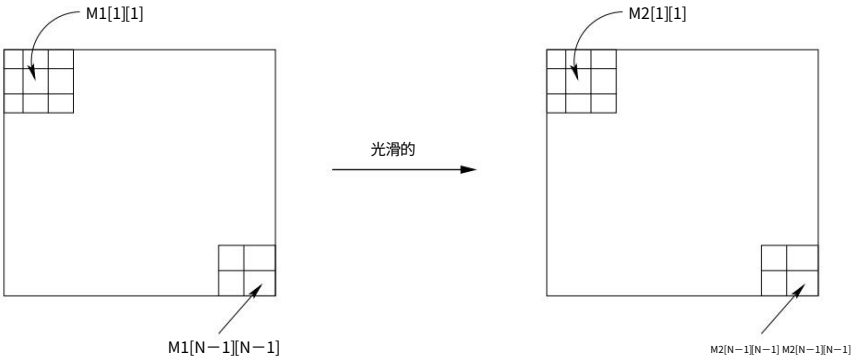

图 2:平滑图像

2 物流

您可以在最多两人的小组中解决此作业中的问题。唯一的“上交”将是电子的。对作业的任何澄清和修改都将发布在课程网站上

页。

3 分发说明

特定于站点：在此处插入一段内容，解释讲师如何将perflab-handout.tar文件分发给学生。

首先将 perflab-handout.tar 复制到您计划在其中完成工作的受保护目录。
然后输入命令：tar xvf perflab-handout.tar。这将导致许多文件被解压到该目录中。您将修改和提交的唯一文件是 kernels.c。
driver.c 程序是一个驱动程序，可让您评估解决方案的性能。使用命令 make driver 生成驱动程序代码并使用命令 ./driver 运行它。

查看文件 kernels.c，您会注意到一个 C 结构团队，您应该在其中插入有关组成您的编程团队的一两个人的请求的识别信息。立即执行此操作，以免忘记。

4 实施概述

数据结构

核心数据结构处理图像表示。像素是一个结构体，如下所示：

```
typedef struct { 无符号短
    红； /* R值*/ unsigned Short green; /* G值 */ unsigned
    Short blue; /* B 值 */ } 像素;
```

可以看出，RGB 值具有 16 位表示形式（“16 位颜色”）。图像 I 表示为一维像素数组，其中第(i, j) 个像素为 I[RIDX(i,j,n)]。这里n是图像矩阵的维数，RIDX是一个宏，定义如下：

```
#定义 RIDX(i,j,n) ((i)*(n)+(j))
```

请参阅文件 defs.h 以获取此代码。

旋转

以下 C 函数计算将源图像 src 旋转 90° 的结果，并将结果存储在目标图像 dst 中。暗淡是图像的尺寸。

```
无效naive_rotate（int暗淡,像素* src,像素* dst){
    整数 i,j;

    for(i=0; i < 暗淡; i++)
        for(j=0; j < 暗淡; j++)
            dst[RIDX(dim-1-j,i,dim)] = src[RIDX(i,j,dim)];

    返回;
}
```

上面的代码扫描源图像矩阵的行,复制到目标图像矩阵的列。
您的任务是使用代码移动、循环展开和阻塞等技术重写此代码,使其尽可能快地运行。

请参阅文件 kernels.c 以获取此代码。

光滑的

平滑函数将源图像 src 作为输入,并在目标图像 dst 中返回平滑结果。这是实现的一部分：

```
无效naive_smooth（int暗淡,像素* src,像素* dst){
    整数 i,j;

    for(i=0; i < 暗淡; i++)
        for(j=0; j < 暗淡; j++)
            dst[RIDX(i,j,dim)] = avg(dim, i, j, src); /* 平滑第 (i,j) 个像素 */

    返回;
}
```

函数 avg 返回第 (i,j) 个像素周围所有像素的平均值。您的任务是优化平滑（和平均)以尽可能快地运行。 （注意:函数 avg 是一个本地函数,您可以完全摆脱它,以其他方式实现平滑。）

该代码（以及 avg 的实现)位于文件 kernels.c 中。

绩效衡量标准

我们的主要性能指标是CPE或每个元素的周期数。如果函数需要C 个周期来运行大小为N × N的图像,则 CPE 值为C/N2 。表 1 总结了上面所示的简单实现的性能,并将其与优化的实现进行了比较。显示了 5 个不同N 值的性能。所有测量均在 Pentium III Xeon Fish 机器上进行。

优化实现与原始实现的比率（加速)将构成您的实现得分。为了总结不同 N 值的总体影响,我们将计算这 5 个值的结果的几何平均值。也就是说,如果N = {32, 64, 128, 256, 512} 的测量加速比为R32、 R64、 R128、 R256和R512 ,那么我们将整体性能计算为

$$R = \sqrt[5]{R32 \times R64 \times R128 \times R256 \times R512}$$

测试用例	1	2	3	4	5	
方法	N 64 128 256 512 1024 几何。意思是					
朴素旋转 (CPE)	14.7 40.1 46.4 65.9 94.5					
优化轮换 (CPE)	8.0	8.6 14.8 22.1 25.3				
加速（天真/选择）	1.8	4.7	3.1	3.0	3.7	3.1
方法	氮32	64 128 256 512 几何。意思是				
朴素光滑（CPE）	695 698 702 717 722					
优化平滑（CPE）	41.5 41.6 41.2 53.5 56.4					
加速（天真/选择）	16.8 16.8 17.0 13.4 12.8					15.2

表 1:优化实施与初始实施的 CPE 和比率

假设

为了让生活更轻松,您可以假设N是 32 的倍数。您的代码必须针对所有此类N 值正确运行，
但我们将仅针对表 1 中所示的 5 个值来测量其性能。

5 基础设施

我们提供了支持代码来帮助您测试实现的正确性并衡量其性能。本节介绍如何使用此基础设施。作业每个部分的具体细节是

在下一节中描述。

注意 :您要修改的唯一源文件是 kernels.c。

版本控制

您将编写旋转和平滑例程的许多版本。帮助您比较性能
对于您编写的所有不同版本,我们提供了一种“注册”功能的方法。

例如,我们为您提供的文件 kernels.c 包含以下函数：

```
无效register_rotate_functions（）{
    add_rotate_function(&旋转,rotate_descr);
}
```

该函数包含一次或多次调用添加旋转函数。在上面的例子中，
添加旋转函数将函数旋转与字符串旋转描述符（ASCII)一起注册
函数功能的描述。请参阅文件 kernels.c 以了解如何创建字符串描述。这
字符串的长度最多为 256 个字符。

文件 kernels.c 中提供了用于平滑内核的类似函数。

司机

您将编写的源代码将与我们提供到驱动程序二进制文件中的目标代码链接。要创建此二进制文件,您需要执行命令

```
unix> 制作驱动程序
```

每次更改 kernels.c 中的代码时,都需要重新制作驱动程序。要测试您的实现,您可以运行以下命令:

```
unix> ./驱动程序
```

该驱动程序可以在四种不同的模式下运行:

- ·默认模式,在该模式下运行所有版本的实现。
- ·自动评分器模式,其中仅运行rotate() 和smooth() 函数。这是我们使用驱动程序对您的交卷进行评分时将运行的模式。
- ·文件模式,其中仅运行输入文件中提到的版本。
- ·转储模式,其中每个版本的一行描述转储到文本文件中。然后,您可以编辑此文本文件以仅保留您想要使用文件模式测试的版本。您可以指定转储文件后是否退出或者是否要运行您的实现。

如果不带任何参数运行,驱动程序将运行您的所有版本(默认模式)。其他模式和选项可以通过驱动程序的命令行参数指定,如下所示:

-g :仅运行rotate()和smooth()函数(自动评分器模式)。

-f <funcfile> :仅执行 <funcfile> 中指定的版本(文件模式)。

-d <dumpfile> :将所有版本的名称转储到名为 <dumpfile> 的转储文件中,一行对应一个版本(转储模式)。

-q :将版本名称转储到转储文件后退出。与-d 一起使用。例如,要退出打印转储文件后,立即键入 ./driver -qd dumpfile。

-h :打印命令行用法。

团队信息

重要提示:开始之前,您应该在 kernels.c 中的结构中填写有关您团队的信息(组名称、团队成员姓名和电子邮件地址)。此信息与数据实验室的信息一样。

6 作业细节

优化旋转(50 分)

在这一部分中,您将优化轮换以实现尽可能低的 CPE。您应该编译驱动程序,然后使用适当的参数运行它来测试您的实现。

例如,使用提供的原始版本(用于旋转)运行驱动程序会生成如下所示的输出:

unix> ./驱动程序
队名:博维克
成员 1:Harry Q. Bovik
电子邮件 1:bovik@nowhere.edu

旋转:版本 = naive_rotate:朴素基线实现:

暗淡	64	128	256	第512章	1024	意思是
您的 CPE	14.6	40.9	46.8	63.5	90.9	
基线 CPE 14.7		40.1	46.4	65.9	94.5	
加速	1.0	1.0	1.0	1.0	1.0	1.0

优化平滑（50分）

在这一部分中,您将优化平滑以获得尽可能低的 CPE。

例如,使用提供的简单版本运行驱动程序（为了平滑)会生成如下所示的输出:

unix> ./驱动程序

平滑:版本 = naive_smooth:朴素基线实现:

暗淡	32	64	128	256	第512章	意思是
您的 CPE	695.8 698.5 703.8 720.3 722.7					
基线 CPE 695.0 698.0 702.0 717.0 722.0						
加速	1.0	1.0	1.0	1.0	1.0	1.0

一些忠告。查看为旋转和平滑生成的汇编代码。注重内在优化
使用类中介绍的优化技巧循环（在循环中重复执行的代码）。光滑的是
与旋转函数相比,计算密集度更高,内存敏感度更低,因此优化在某种程度上
不同的口味。

编码规则

你可以编写任何你想要的代码,只要它满足以下条件:

- 必须采用ANSI C 语言。不得使用任何嵌入式汇编语言语句。
- 它不得干扰时间测量机制。如果您的代码打印任何内容,您也会受到处罚
  无关的信息。

您只能修改kernels.c中的代码。您可以定义宏、附加全局变量和其他
这些文件中的过程。

评估

您的旋转和平滑解决方案各占您成绩的 50%。每个分数将基于
下列:

- 正确性:如果有错误的代码导致司机抱怨,您将不会获得任何奖励!这包括代码
  对测试尺寸正确操作,但对其他尺寸的图像矩阵操作错误。如前面提到的,
  您可以假设图像尺寸是 32 的倍数。

- CPE:如果您的旋转和平滑实现正确并且平均CPE 分别高于阈值Sr和Ss ,您将获得满分。如果您的正确实现比所提供的简单实现更好,您将获得部分荣誉。

  特定地点:作为讲师,您需要决定您的全额学分阈值Sr和Ss以及部分学分的规则。我们通常使用线性标度,如果学生确实尝试解决实验室问题,则最低限度约为 40%。

7 上交指示

特定于站点:在此处插入一段内容,告诉每个团队如何提交其kernels.c文件。例如,以下是我们在 CMU 使用的交接指令。

完成实验后,您将提交一个文件 kernels.c,其中包含您的解决方案。以下是提交解决方案的方法:

- 确保您已将识别信息包含在kernels.c 的团队结构中。
- 确保rotate() 和smooth() 函数与您最快的实现相对应,因为这些函数
  当我们使用驱动程序对您的作业进行评分时,这是唯一要测试的功能。
- 删除任何无关的打印语句。
- 创建以下形式的团队名称:
  - “ID”,其中 ID 是您的 Andrew ID(如果您独自工作),或者
  - “ID1+ID2” ,其中 ID1是第一个团队成员的 Andrew ID,ID2是第一个团队成员的 Andrew ID
    第二个团队成员。

  这应该与您在 kernels.c 结构中输入的团队名称相同。
- 要提交 kernels.c 文件,请输入:

  make handin TEAM=团队名称

  其中 teamname 是上述团队名称。
- 提交后,如果您发现错误并希望提交修改后的副本,请键入

  make handin TEAM=团队名称 VERSION=2

  每次提交时不断增加版本号。
- 您可以通过查看来验证您的提交

  /afs/cs.cmu.edu/academic/class/15213-f01/L1/handin

  您在此目录中具有列出和插入权限,但没有读取或写入权限。

祝你好运!