第 2 卷 第 2,3 期
1997 年 3 月

中国图象图形学报
China Journal of Image and Graphics

Vol. 2, No. 2,3
Mar. 1997

模板匹配的并行算法*

卢　力　　王能超

（武汉城市建设学院电气与计算机工程系，武汉 430074）　（华中理工大学并行计算研究所，武汉 430074）

摘　要　模板匹配方法是检测图像边缘的重要方法之一。本文对模板匹配方法中的 Prewitt 算子进行了研究，提出了用该算子检测图像边缘的快速算法、向量及分布式算法，并且在并行虚拟环境 PVM 上实现了分布式计算，加速效果明显。

关键词：边缘检测，模板匹配，快速算法，向量算法，分布式算法，PVM

1　引　言

图像的边缘信息无论是对人类或对机器视觉来说都是非常重要的：边缘具有能勾画出区域的形状、能被局部定义以及它能传递大部分图像信息等许多优点，因此边缘检测可以看作是处理许多问题的关键。边缘可以被定义为在局部区域内图像特性的差别，它表现为图像上的不连续性（如表现在图像上灰度级的突变，纹理结构的突变以及彩色的变化等）。一般为了检测灰度的不连续性，可以进行（空间）微分。在数字图像场合，微分常用差分来进行计算。例如，X 方向和 Y 方向的一阶差分可定义为：

$$\Delta_x f(i,j) = f(i,j) - f(i-1,j),$$

$$\Delta_y f(i,j) = f(i,j) - f(i,j-1).$$

因微分值随图像边缘的方向而变化，因此不大有用。经典的边缘提取算法以原始图像为基础，对图像的每一个像素考察其一定邻域内的灰度变化。利用边缘邻近的一阶或二阶方向导数的变化规律，用简单的方法检测出图像边缘。如 Roberts 算子等，其处理的速度快，但存在对噪声的响应敏感的问题，效果不是很理想[1]。方向模板匹配算子（简称模板匹配）处理是边缘提取算法中具有平滑噪声优点的一种常用的方法，它根据各小区域边缘的方向进行连接，能够得到比微分算子较好的结果，其缺点是计算量大、费时。因此，寻求方向模板匹配算子的快速算法和并行算法是解决速度慢这一问题的根本途径[2~4]。本文就 Prewitt 方向模板算子研究其相应的算法，并在并行虚拟环境 PVM 中实现其分布式计算。

2　模板匹配的快速算法

Prewitt 算子的 8 个 3×3 的窗口模板及所代表的边缘方向如图 1 所示[4]，这些算子模板由理想的

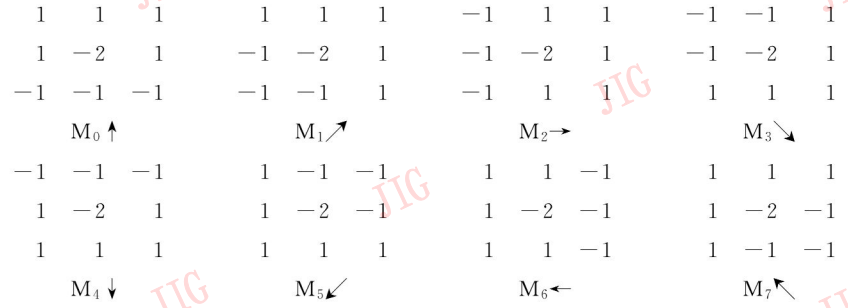

图 1　Prewitt 算子的 8 个模板及所代表的边缘方向

Fig. 1　Eight masks and their represented edge directions of prewitt operator.

北京应用物理与计算数学研究所计算物理实验室基金资助课题。
收稿日期：1996 年 10 月；收到修改稿日期：1997 年 1 月。

边缘子图像构成。把每块模板与图像 3×3 区域匹配，把处在模板内图像的每一点乘以模板内的相应方格中指示的数字，然后把结果相加，其中最大输出值作为图像中心点的值，即是边缘要素的强度，得到最大输出值的方向就是中心点的方向。

设 $f(i,j)$ 为 $M\times N$ 图像中当前点 (i,j) 处的灰度值，$g(i,j)$ 为经 Prewitt 算子模板匹配后点 (i,j) 处的灰度值，记

$$\begin{aligned}
p_0&=f(i-1,j-1),\\
p_1&=f(i-1,j),\\
p_2&=f(i-1,j+1),\\
p_3&=f(i,j+1),\\
p_4&=f(i+1,j+1),\\
p_5&=f(i+1,j),\\
p_6&=f(i+1,j-1),\\
p_7&=f(i,j-1),\\
p_8&=f(i,j),
\end{aligned}$$

则计算

$g(i,j)(i=1,2\cdots M-2;\ j=1,2\cdots N-2)$

的经典算法描述如下：

步 1　计算

$$\begin{aligned}
e_0&=p_0+p_1+p_2+p_3-p_4-p_5\\
&\quad-p_6+p_7-2\times p_8,\\
e_1&=p_0+p_1+p_2+p_3+p_4-p_5\\
&\quad-p_6-p_7-2\times p_8,\\
e_2&=-p_0+p_1+p_2+p_3+p_4+p_5\\
&\quad-p_6-p_7-2\times p_8,\\
e_3&=-p_0-p_1+p_2+p_3+p_4+p_5\\
&\quad+p_6-p_7-2\times p_8,\\
e_4&=-p_0-p_1-p_2+p_3+p_4+p_5\\
&\quad+p_6+p_7-2\times p_8,\\
e_5&=p_0-p_1-p_2-p_3+p_4+p_5\\
&\quad+p_6+p_7-2\times p_8,\\
e_6&=p_0+p_1-p_2-p_3-p_4+p_5\\
&\quad+p_6+p_7-2\times p_8,\\
e_7&=p_0+p_1+p_2-p_3-p_4-p_5\\
&\quad+p_6+p_7-2\times p_8.
\end{aligned}\tag{1}$$

步 2　计算　$g(i,j)=\max\{e_0,e_1\cdots e_7\}$。

将(1)式变形如下：

$$\begin{aligned}
e_0&=(p_0-p_5)+(p_1-p_6)\\
&\quad+(p_2+p_3)-(p_4-p_7)-2\times p_8,\\
e_1&=e_0+2\times(p_4-p_7),\\
e_2&=e_1+2\times(p_5-p_0),\\
e_3&=e_2+2\times(p_6-p_1),\\
e_4&=e_3+2\times(p_7-p_2),\\
e_5&=e_4+2\times(p_0-p_3),\\
e_6&=e_5+2\times(p_1-p_4),\\
e_7&=e_6+2\times(p_2-p_5).
\end{aligned}\tag{2}$$

从而，快速算法描述如下：

步 1　计算

$$\begin{aligned}
x_0&=p_4-p_7,\quad x_1=p_5-p_0,\\
x_2&=p_6-p_1,x_3=p_7-p_2,\\
x_4&=p_0-p_3,\quad x_5=p_1-p_4,\\
x_6&=p_2-p_5,\quad x_7=p_2+p_3.
\end{aligned}$$

步 2　计算

$$\begin{aligned}
e_0&=-x_0-x_1-x_2+x_7-2\times p_8,\\
e_1&=e_0+2\times x_0,\\
e_2&=e_1+2\times x_1,\\
e_3&=e_2+2\times x_2,\\
e_4&=e_3+2\times x_3,\\
e_5&=e_4+2\times x_4,\\
e_6&=e_5+2\times x_5,\\
e_7&=e_6+2\times x_6.
\end{aligned}$$

步 3　计算　$g(i,j)=\max\{e_0,e_1,\cdots,e_7\}$。

对一幅大小为 $M\times N$ 的灰度图像，用经典算法需要 $64(M-2)(N-2)$ 次加减操作和 $8(M-2)(N-2)$ 次移位操作，而用快速算法只需 $19(M-2)(N-2)$ 次加减操作和 $8(M-2)(N-2)$ 次移位操作. 这里结果图像的第 0 和 $M-1$ 行及第 0 和 $N-1$ 列直接赋零值(下同)。如不考虑移位操作所需时间，经典算法与快速算法所用的时间之比在理论上可以达到 3.37。在 PC386 微机上进行实际测试，快速算法比经典算法要快约 3 倍。

3　模板匹配的并行算法

由于图像经模板匹配方法处理后当前中心点的灰度值只与图像处理前的灰度值有关，而与处理后其它点的灰度值无关，故可实施并行处理。下面将讨论用 Prewitt 算子检测图像边缘的向量及分布式并行算法。

3.1　向量算法

设 $M\times N$ 图像第 i 行上的数据依次为 $f(i,0)$，$f(i,1),\cdots,f(i,N-1)$，$i=0,1,\cdots,M-1$，用 $m(k,l)$ 表示第 k 个窗口模板中依行顺序从左到右

的第 l 个加权系数 ($k=0,1,\cdots,7; l=0,1,\cdots,8$),向量算法描述如下:

步 1　建立 $3M$ 个 $N-2$ 维向量:

$$\mathbf{Q}_{0,0}=[f(0,0),f(0,1),\cdots,f(0,N-3)]^T,$$
$$\mathbf{Q}_{0,1}=[f(0,1),f(0,2),\cdots,f(0,N-2)]^T,$$
$$\mathbf{Q}_{0,2}=[f(0,2),f(0,3),\cdots,f(0,N-1)]^T,$$
$$\cdots\cdots$$
$$\mathbf{Q}_{M-1,2}=[f(M-1,2),f(M-1,3),\cdots,f(M-1,N-1)]^T$$

步 2　对 $i=1,1,2,\cdots,M-2$,分别计算

$$\begin{aligned}\mathbf{G}_{k,i}=&m(k,0)\mathbf{Q}_{i-1,0}+m(k,1)\mathbf{Q}_{i-1,1}\\&+m(k,2)\mathbf{Q}_{i-1,2}+m(k,3)\mathbf{Q}_{i,0}\\&+m(k,4)\mathbf{Q}_{i,1}+m(k,5)\mathbf{Q}_{i,2}\\&+m(k,6)\mathbf{Q}_{i+1,0}+m(k,7)\mathbf{Q}_{i+1,1}\\&+m(k,8)\mathbf{Q}_{i+1,2}\\=&[G_{k,i}(j)]^T\end{aligned}$$

其中,$k=0,1,2,\cdots,7$; $j=1,2,\cdots,N-2$.

步 3　计算

$$G(i)=\max_k\{\mathrm{G}_{k,i}\}=\{g(i,j)\},$$
$$i=1,2,\cdots,M-2;$$
$$j=1,2,\cdots,N-2.$$

由于 $m(k,l)$ 与一向量 $\mathbf{Q}$ 相乘,$k=0,1,2,\cdots,7$; $l=0,1,2,\cdots,8$,所以向量算法的加速比在理论上可以达到 $N-2$.

3.2　模板匹配的分布式算法及在并行虚拟环境 PVM 中的实现

由于目前一台大型的并行计算机或并行处理机系统的结点数(处理机台数)要比一幅大规模数字图像的像素点数少得多,因此,Prewitt 算子的分布式算法可对图像数据按分段或分块处理的方法进行。设图像大小为 $M\times N$,并行计算机或并行处理机系统中的结点数为 P,且 M/P 为整数。分布式算法描述如下步骤:

①将图像数据分成 P 段,每段含 $M\times N/P$ 个数据,依次分布到 P 个结点上;

② P 个结点同时进行处理(程序相同);

③依次收集各个结点上的结果数据,得到输出图像数据。

由于对图像数据采用分段的方法并对分段后的每段数据同时进行处理,因而分布式算法的加速比在理论上可以达到 P。

利用 PVM 所提供的消息传递方式[5],我们采用 master-slave 模式编写了用 Prewitt 算子检测图像边缘的分布式并行计算的 C 语言程序。表 1 是在由四台 SGI 工作站联结而成的并行处理机系统上实际测试的结果数据。这里的 512×512、1024×1024、2048×2048 的灰度图像分别由 4 个、16 个、64 个标准的 Lenna 测试图像拼接而成。

表 1　分布式算法的结果数据

Table 1　Result data of the distributed algorithm

图像尺寸	CPU 时间 (s)	墙上时间 (s)	并行加速比	并行效率 (%)
512×512	2.92	0.76	3.84	96.05
1024×1024	11.67	3.02	3.86	96.5
2048×2048	46.61	12.04	3.87	96.75

从上述结果可以看出,并行加速比平均达到 3.857,其平均的并行效率高达96.425%,而且随着图像尺寸的增大,并行加速比增加,并行效率也随之提高,这正是大规模数字图像处理所期望的。实验结果还表明,经分布式并行计算处理后的结果图像其显示效果与串行计算处理后的结果图像的显示效果一样(要对分块或分段后的子图像的边缘进行适当的加工处理)。

参考文献

1　周新伦,柳健,刘华志,等编. 数字图象处理. 北京:国防工业出版社, 1986.

2　吴小培,汪炳权,黄立霞. 模板匹配的快速算法. 信号处理, 1993,9(4): 221～225.

3　郑翔, 黄艺云. 经典边缘检测模板的快速算法. 信号处理,1995, 11(4):317～320.

4　何子凡,蔡长安,罗铁建. 图像边缘检测的并行算法及其在 YH－2 上的实现. 全国第四届并行算法学术会议论文集. 北京: 航空工业出版社,1993:276～280.

5　孙家昶,张林波,迟学斌,等编著. 网络并行计算与分布式编程环境. 北京:科学出版社,1996.

卢力　1986 年毕业于华中师范大学数学系，获理学学士学位。现任武汉城市建设学院电气与计算机工程系讲师，主要研究方向为并行计算和数字图像处理等。

Parallel Algorithm for Template Matching

Lu Li

(*Department of electrical engineering & computer engineering, WUCI, Wuhan* 430074)

Wang Nengchao

(*Parallel Computation Research Institute, HUST, Wuhan* 430074)

Abstract In this paper, a fast algorithm and parallel algorithms for template matching in image edge detection are established. Using PVM, the distributed algorithm is accomplished. This lays a solid foundation for Massively Image Parallel Processing.

Keywords Edge detection, Template matching, Fast algorithm, Verctor algorithm, Distributed algorithm, PVM