第 17 卷 第 3 期　　电 子 科 学 学 刊　　Vol. 17 No. 3
1995 年 5 月　　JOURNAL OF ELECTRONICS　　May 1995

平行多视点算法实时求取深度信息

王 新 成　　朱维乐　顾德仁
（清华大学电子工程系　北京　100084）　（电子科技大学电子工程系　成都　610054）

摘要　引入了平行多视点的概念，从平行多视点获取的多幅图象中可以求得深度信息. 通过利用图象的 FFT 变换，将时域中很难解决的问题变换到频域中来解决，避免了复杂的匹配运算，极大地简化了求取图象点、面深度信息的问题，并且提高了深度求取的实时性、全面性和准确性. 介绍了算法原理及特点，并给出了实验结果及分析.

关键词　平行多视点，深度信息，算法，实时

1　平行多视点图象的获取

在本文的实验中，暂用单摄象机平移得到多视点. 如图 1 所示配置. 摄像机作平移，分别在 $V_1, V_2, \cdots, V_N$ 点处摄得 N 幅图象，其中 $|V_1V_2| = |V_2V_3| = \ldots = |V_{N-1}V_N|$. 平移间距的大小应根据对物点实际深度的估计及摄像机的焦距进行选取. 这样摄得的多幅图象，我们就将其作为本文实验所用的平行多视点图象.

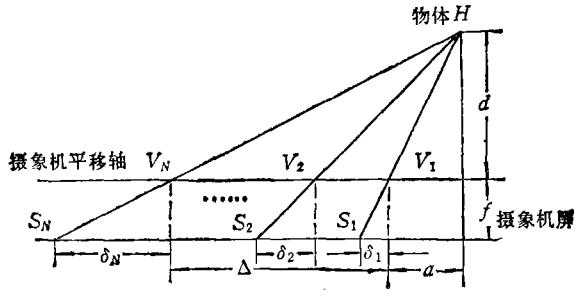


图 1　获取平行多视点图象及深度恢复算法原理说明

2　深度获取的算法原理

如图 1 所示，$S_1, S_2, \cdots, S_N$ 点为物点 H 分别在 N 幅图象上的 N 个象点. 在图中，f 为摄象机焦距；d 为物点 H 的深度；$\delta_1, \delta_2, \cdots, \delta_N$ 分别为象点 $S_1, S_2, \cdots, S_N$ 点相对于光轴的偏移量.

由图 1，可设 $|V_1V_2| = |V_2V_3| = \ldots = |V_{N-1}V_N| = \Delta/(N-1)$，根据相似关系：

1993-05-12 收到，1994-10-25 定稿
王新成　1965 年生，博士后，研究领域有：体成像、动画、图象压缩编码、三维计算机视觉等.
朱维乐　1942 年生，教授，现从事图文电视及 HDTV 研究.
顾德仁　1924 年生，教授，博士生导师，主要从事信号与信息处理研究工作.

$$\begin{cases}\delta_N/f = (\Delta + a)/d, \\ \delta_1/f = a/d \ (a \text{ 的定义见图 1}),\end{cases}$$

则

$$(\delta_N - \delta_1)/f = \Delta/d.$$

令 $\delta = \delta_N - \delta_1$，

则

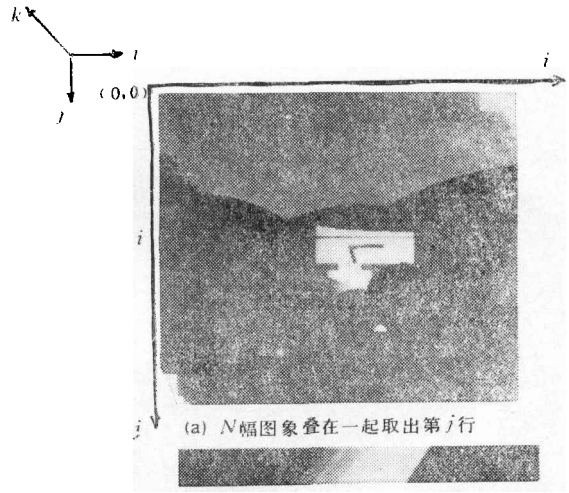
$$d = (\Delta \cdot f)/\delta, \tag{1}$$

δ 表示物点H的象点的总位移量. 注意: 由(1)可知，δ 与 $\delta_2,\delta_3,\cdots,\delta_{N-1}$ 无关，只与首末两幅图象中物点的象点的偏移量有关. 这只是从 δ 的表达式看出的，但在实际计算中正是利用了各偏移点(即 $S_1,S_2,\cdots,S_N$ 点)在同一直线上这一事实(这点是显然的，证略)，才简化了深度的计算；否则就又变为立体视觉，陷入复杂的匹配运算求对应点了. 这也是本方法显著特点之一. 下面将用图片具体说明.

现在假定图象尺寸为 256×256，将摄取的N幅图象叠在一起，如图 2(a)所示. j 表示行数，i 表示一行中的第 i 个象素，k 表示第 k 幅画面. 取出任意的第 j 行，取出的 N行就形成了 $N \times 256$ 的块图象，如图 2(b) 所示. 在块图象中，相同灰度的点形成了很多斜线. 每条斜线上的点是同一物点的象点. 其斜率即为

$$K = \Delta/\delta = \Delta/(\delta_N - \delta_1) \tag{2}$$

由(1),(2)两式得: $d = K \cdot f$，这样就把深度 d （绝对深度）的求取问题转化为直线斜率 K 的计算问题了. 但直接在空域中计算斜率 K不容易. 为便于计算直线斜率 K，我们将图象变换到频域中来考虑，利用了傅里叶变换的旋转性. 这也是该方法为什么能简化深度计算的关键所在. 为此，我们特将傅里叶变换的旋转性作一讨论.



(b) $N \times 256$ 块图象

图 2　获得 $N \times 256$ 块图象说明

设有二维傅里叶变换对: $f(x,y) \longleftrightarrow F(u,v)$，这里，我们引入极坐标: $x = r \cdot \cos\theta, y = r \cdot \sin\theta, u = \omega \cdot \cos\varphi, v = \omega \cdot \sin\varphi$，则 $f(x,y)$ 和 $F(u,v)$ 分别地变为 $f(r,\theta)$ 和 $F(\omega,\varphi)$. 无论在连续的或离散的傅里叶变换对中用直接代入方法都可以证明: $f(r,\theta+\theta_0) \longleftrightarrow F(\omega,\varphi+\theta_0)$. 换言之，如果 $f(x,y)$ 被旋转 θ_0，则 $F(u,v)$ 亦被旋转同一角度.

利用这一性质，在空域中对图象中一条直线斜率不好求，可将其变换到频域中来求. 这时需要注意的是，时频域中的两条直线是相互垂直的. 这同时也利用了傅里叶变换的周期性和对称性. 当然，对于本文算法，采用其它具有周期性的正交变换，如 DCT 等也是可行的. 但这里，我们采用傅里叶变换，因为傅里叶变换是目前我们用得最多的，并有各种快速算法. 这为实时处理和简化获取图象深度信息的计算提供了有力的支持. 下面

讨论该算法的计算原理.

由图 2(b), 现在在空域中形成了近似的平行线段,则在傅里叶变换的幅度谱中,能量主要集中在与空域中平行线相垂直的方向上, 即灰度值较大的点都集中在这一方向上. 这样可先用一门限检测出这些点, 这可采用图象二值化、直线拟合及直线方向定位算法, 求出直线斜率. 这样求得的直线斜率的精度肯定会提高, 因为这基本上是单线拟合. 如设原直线斜率为K,频域中拟合直线斜率为 K', 则 $K \cdot K' = -1$. 求出了 K', 就可计算出 K. 这样,求深度 d 的问题又转化为求频域中拟合直线斜率 K' 的问题了.

对一行来说, $N \times 256$ 的块图象,需要求出 256 点的深度信息. 采用滑动窗,从左到右,依次取出$N \times N$小块进行计算. 因为我们要求块内的近似平行线的斜率,为消除傅里叶变换的周期性的影响,先对$N \times N$块图象进行梯度运算,接着进行 FFT 变换. 这样,在幅度谱中就可消除横向及纵向两条明显的亮线, 而只剩下一条有用的明显的斜线了. 采用上面所述图象二值化及直线拟合技术, 即可得到该斜线的拟合直线斜率, 进而求出该$N \times N$块图象中心点的绝对深度,如图 3 所示.

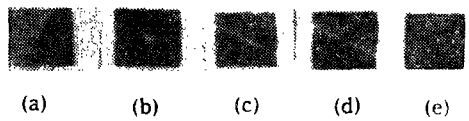


图 3 $N \times N$ 块图象中心点深度的求取

(a) 原图($N \times N$) (b) FFT 变换幅度谱 (c) 原图梯度运算后的 FFT 变换幅度谱

(d) 无用区域置为零后的结果图象 (e) 阈值处理后的结果图象

当$N \times N$滑动窗在 $N \times 256$ 块图象中滑动完后, 可得到 $256-N$ 点的深度值. 对于开始及最后 $N/2$ 点的深度值, 可用曲线拟合或插值法得到. 这样, 即可得到整行 256 点的深度值(其中有些点无法计算出斜率 K, 故无法计算出深度值 d). 依此一行一行地计算下去,可求出整幅图象的深度图.

经过上面对整幅图象的处理后,我们只能得到稀疏的、非均匀分布的深度图, 因为所用滑动窗很小,而且在一幅图象中难免有多块无灰度变化的区域,这时就无法直接计算出该点深度. 为此必须为获取均匀网格点的深度值进行内插或补插. 内插方法, 例如, 线性、B 样条、松弛算法等,在图形学中已有很多描述,这里就不再赘述. 由均匀网格点数据可以方便地绘制出深度图.

在本文中将只获得稀疏深度图,后期内插或补插结果不讨论,以集中说明本文算法的优越性.

3 实验结果

3.1 *仿真实验* 首先, 通过我们自行配置的以 IBM386 微机、VG-32 图象处理板、VEGA 高分显示卡及彩显为主的 RISG 原理系统,生成平移间距为 1 的多幅仿真图象,采用本算法计算出深度图. 图 4 示出了原图象及其深度图象的灰度图形式.

从图 4(b) 中仍可看出图 4(a) 中各个物体的基本轮廓,并且距离近的物体亮,距离远的物体暗. 这正显示了图象的深度层次. 表 1 示出了滑动窗宽度$N = 8$ 和 $N = 16$ 时仿

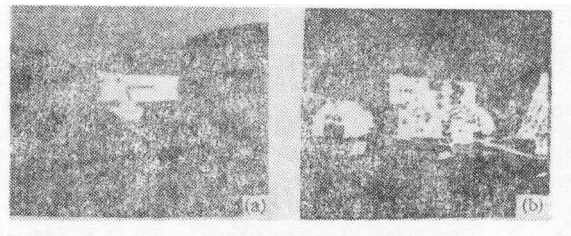


图 4　仿真图象求取深度实验(图象尺寸为 256×256)
(a) 原图象　(b) 深度图象的灰度图形式

表 1　$N=8$ 及 $N=16$ 时仿真图象求取深度比较(深度单位：象素)

点的编号	实际深度	$N=16$			$N=8$		
		求出深度	绝对误差	相对误差	求出深度	绝对误差	相对误差
1	518.208	523.847	5.639	1.088%	507.729	10.479	2.022%
2	641.277	641.713	0.436	0.068%	611.155	30.122	4.697%
3	600.210	611.155	10.945	1.824%	611.155	10.945	1.824%
4	506.132	496.775	9.357	1.849%	507.729	1.597	0.315%
5	572.260	582.006	9.746	1.703%	595.705	23.445	4.097%
6	587.072	528.510	58.472	9.960%	733.386	146.314	24.923%
7	359.752	391.381	31.629	8.792%	471.462	111.710	31.052%
8	266.991	291.931	24.940	9.341%	327.491	60.500	22.660%
9	2218.985	2424.396	205.411	9.257%	1462.444	756.541	34.094%
10	2149.082	2353.657	204.575	9.519%	1273.425	875.657	40.746%

真图象中 10 点(图 4(a) 中带“+”的点，前 5 点为边缘点)的绝对误差及相对误差.

由表 1 可知：$N=8$ 时，边缘点求出深度值的相对误差<5%，非边缘点求出深度值的相对误差<41%；$N=16$ 时，边缘点求出深度值的相对误差<2%，非边缘点求出深度值的相对误差<10%. 根据我们对多幅不同的仿真图象求取深度图，都可得到上述结果. 故我们可得，N越大，精度越高.

下面我们对图 4 所示图象 (图象尺寸：256×256. 总象素数=65536) 给出一统计结果，如表 2 所示.

表 2　滑动窗宽度 N 与求出深度点数的关系

滑动窗宽度 (N)	$N=8$	$N=16$	$N=32$
原图象中能求出深度值点数 (S)	10835	17881	23373
在原图象中所占百分比 $\left(\frac{100S}{65536}\right)$	16.53%	27.28%	35.66%

由表 2 可知，随着N值的增大：$N=8,16,32$，能求出深度值的点数也逐渐增多，在原图象中所占百分比从 16.53% 增加到 35.66%. 根据对多幅不同的仿真图象求出深度图的统计结果表明，都可得出这一结果，即随着N值增大，能求出深度值点数在原图象中所占百分比从百分之十几增大到百分之几十. 由此可得，N越大，能求出深度值点数越多.

从计算机模拟的结果可见，该算法的精确度是很高的. 当输入是边缘点(即定位精度很高)时，所恢复的绝对深度与理论值几乎一致(误差在 3‰～5% 范围内)；当是非边缘点(即本身定位误差较大)时，所恢复的绝对深度与理论值相比，能计算出深度值的点，只要滑动窗宽度选择得当，其误差可控制在容许的范围内.

3.2 *实摄图象实验* 这时增加了 CCD SHARP 型摄象机、PC VISION85 图象采集卡、滑动平台以及必要的光照条件及景物(垂直线条较明显). 在本实验中，暂用单摄像机平移得到多视点. 通过实验证明了这一原理算法在实用中的可行性及优越性.

图 5 示出了采用本算法计算出的深度图. 表 3 示出了任选10点(前 5 点为边缘点)求出深度值误差比较及与滑动窗宽度的关系. 由表 3 可知，滑动窗宽度 $N=8$ 时，边缘点求出深度值相对误差<6%，非边缘点求出深度值相对误差<43%；$N=16$ 时，边缘点求出深度值相对误差<3%，非边缘点求出深度值相对误差<11%. 根据我们对多幅不同的实摄图象求取深度图，都可得到上述结果，即 N 越大，求出深度值误差越小. 下面给出多幅图象(其中 Img1 表示图 5(a) 所示图象. 其它图象及其深度图的图片未给出)的统计结果，如表 4 所示(图象尺寸都为 256×256).

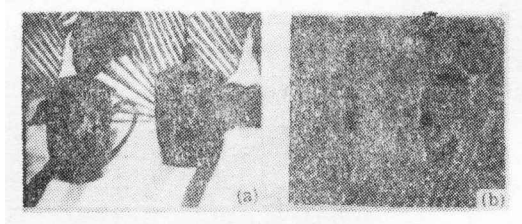


图 5 实摄图象求取深度实验(图象尺寸 256×256)
a) 原图象 (b) 深度图象的灰度图形式

表 3 滑动窗为 8×8，16×16 时原图象中任选 10 点求出深度值误差比较

	滑动窗为 8×8	滑动窗为 16×16
点的编号	求出点深度相对误差(%)	求出点深度相对误差(%)
1	2.309	0.115
2	0.714	2.346
3	5.196	1.715
4	4.823	2.022
5	3.271	1.988
6	17.221	5.699
7	23.667	4.320
8	38.069	9.214
9	24.337	8.723
10	42.127	10.112

由表 4 可知，$N=8$ 时，原图象中有 50% 以上的点可求出深度值，最高可达 70% 以上；$N=16$ 时，原图象中有 80% 以上的点可求出深度值，最高可达 90% 以上. 在同一 N 值下，原图象中景物越多，明显的黑斑或亮斑区域越少，能求出深度值的点数越多；反之，能求出深度值的点越少. 随着 N 值的增大，能求出深度值点数在原图象中所占百分比

表 4　滑动窗宽度 N 与图象中能求出深度值点数的关系

滑动窗宽度（N）		$N=8$	$N=16$
原图象中能求出深度值点数（S）	Img1	38378	54438
	Img2	38549	54846
	Img3	45541	56420
	Img4	48973	60275
	Img5	37811	57356
在原图象中所占百分比 $\left(\frac{100S}{65536}\%\right)$	Img1	58.56%	83.07%
	Img2	58.82%	83.69%
	Img3	69.49%	86.09%
	Img4	74.73%	91.97%
	Img5	57.70%	87.52%

从 60% 左右增加到 80% 以上，即 N 越大，能求出深度值点数越多。

从实摄图象实验结果可知，本算法的精确度是很高的，所恢复的绝对深度，对于边缘点，误差可在百分之几左右；对于非边缘点，误差在百分之几十以内。

4　讨论

根据实验结果，给出运行时间比较。需要说明的是，我们的计算是在带有 80387 协处理器的 PC386 微机（主频 33MHz）上进行的。表 5 中仿真图象指的是图 4 所示图象，实摄图象指的是图 5 所示图象。

表 5　运行时间比较

滑动窗宽度（N）		$N=8$	$N=16$
运行时间（s）	仿真图象	33	54
	实摄图象	116	165

由表 5 可知，采用本算法时，只需几十秒到两分左右。用其它具有不同景物的图象进行计算，可得相同结果，其运行时间总可控制在 1min 到 2min 之内。这就可以达到准实时实现。这与目前见于文献的其它算法比较都要快得多。要进一步提高平行多视点算法的速度，可从进一步减少以下三种算法的运算次数着手：傅里叶变换算法、图象二值化算法、直线拟合算法。

5　总结

本文具体给出了平行多视点求取图象点、行及整幅图象深度值的实用算法，并给出了实验结果：包括仿真实验及实摄图象实验结果。将它与其它算法从计算的复杂程度、求出深度的全面性、精度及计算时间等四个方面进行比较，进一步证明了平行多视点算法各方面的优越性，突出说明了它在实时性方面的特点及贡献。实验结果证明了该算法的可行性。平行多视点算法不失为一种优越的新方法。

在本文实验中，用单摄象机平移得到多视点，离实用阶段还有一段距离。在实用中，应是特制的多镜头摄象（镜头间距可作一定程度的调整），一次得到多幅画面。这样，计算效果可进一步提高。这方面工作我们还在作进一步研究。

参 考 文 献

[1] Zhu X K, *et al*. AMSE Review, 1991, 19(3): 25—32.
[2] Nasrabadi M. IEEE Trans. on PAMI, 1992, PAMI-14(5): 566—572.
[3] Dhar M, Rangarajan K, Mujumdar J. Asia-Pacific Engineering Journal, Part A [Electrical Engineering], 1992. 2(2): 217—231.
[4] Sparr G. Image and Vision Computing, 1992, 10(10): 683—688.

REAL-TIME IMPLEMENTATION OF CALCULATING DEPTH INFORMATION BY IMAGE SEQUENCES ACQUIRED FROM THE PARALLEL MULTI-VIEWPOINTS

Wang Xincheng
(*Department of Electronic Engineering, Tsinghua University, Beijing* 100084)

Zhu Weile Gu Deren
(*Department of Electronic Engineering, University of Electronic Science and Technology of China, Chengdu* 610054)

Abstract A new algorithm of calculating depth information by image sequences acquired from the parallel multi-viewpoints is discussed. By means of orthogonal transformation of image (in this paper, the authors use fast Fourier transformation), the matching problem, which is difficult to solve in time domain, is put into transformation field in frequency domain.

Key words Parallel multi-viewpoint, Depth information, Algorithm, Real-time