

(12) 发明专利

(10) 授权公告号 CN 1971591 B

(45) 授权公告日 2010.05.19

(21) 申请号 200610162844.1

US 5862255 A, 1999.01.19, 全文.

(22) 申请日 2006.11.24

US 4874936 A, 1989.10.17, 全文.

(30) 优先权数据

审查员 王咪娜

2005-339079 2005.11.24 JP

2005-339080 2005.11.24 JP

(73) 专利权人 佳能株式会社

地址 日本东京

(72) 发明人 内山晋二 尼莎·萨布拉玛尼阿玛

佐藤清秀

(74) 专利代理机构 中国国际贸易促进委员会专

利商标事务所 11038

代理人 王萍

(51) Int. Cl.

G06K 19/06 (2006.01)

G06K 9/18 (2006.01)

(56) 对比文件

CN 1438603 A, 2003.08.27, 全文.

CN 1077299 A, 1993.10.13, 全文.

CN 1334544 A, 2002.02.06, 全文.

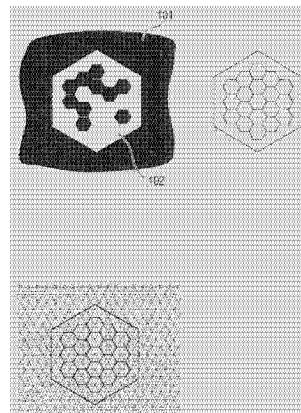
权利要求书 2 页 说明书 13 页 附图 11 页

(54) 发明名称

用于检测二维代码的方法和二维代码检测器

(57) 摘要

一种二维代码，包括可以根据亮度或颜色与背景区别开来的多边形外部区域，以及位于与外部区域的周界具有预定距离处的内部区域。内部区域被分成预定大小的单元，以便通过每一个单元的亮度或颜色来代表信息。这些单元被布置得彼此毗邻，每一个单元都具有六边形的形状。



1. 一种用于检测二维代码的方法,包括下列步骤:

从背景中检测在图像上投影的二维代码的多边形外部区域;

根据图像中的多边形外部区域的形状识别内部比特单元的位置;以及

根据图像中的比特单元的亮度或颜色获取比特值,

其中,二维代码包括能够根据亮度或颜色与背景区别开来的多边形外部区域,以及位于与多边形外部区域的周界具有预定距离处的内部区域,内部区域具有多个六边形比特单元,这些比特单元被布置得彼此毗邻。

2. 根据权利要求 1 所述的方法,

其中在所述检测步骤中,检测到四边形外部区域作为所述多边形外部区域,并且所述识别步骤包括下列步骤:

根据在检测步骤中检测到的所述四边形外部区域的形状和对应的物理外部区域的实际形状之间的关系,计算代表二维投影变换的信息;以及

使用代表二维投影变换的信息计算图像中的每一个比特单元的位置。

3. 根据权利要求 1 所述的方法,

其中,多边形外部区域具有六边形的形状。

4. 根据权利要求 3 所述的方法,

其中,每一个比特单元都具有正六边形的形状。

5. 根据权利要求 4 所述的方法,

其中,比特单元是这样排列的,以便连接六边形外部区域的相对顶点的线和与其交叉的正六边形比特单元的任何相对的边成直角地交叉。

6. 根据权利要求 1 所述的方法,

其中,比特单元具有其高度和宽度相等的六边形的形状。

7. 根据权利要求 6 所述的方法,

其中,外部区域具有四边形的形状。

8. 根据权利要求 7 所述的方法,

其中,使用四个六边形单元来确定二维代码的取向,所述四个六边形单元中的每一个都位于四边形外部区域的四个顶点的其中一个的附近。

9. 一种二维代码检测器,包括:

多边形外部区域检测设备,适用于从背景中检测在图像上投影的二维代码的多边形外部区域;

比特单元识别设备,适用于根据图像中的多边形外部区域的形状识别内部比特单元的位置;以及

比特值获取设备,适用于根据图像中的比特单元的亮度或颜色获取比特值,

其中,二维代码包括能够根据亮度或颜色与背景区别开来的多边形外部区域,以及位于与多边形外部区域的周界具有预定距离处的内部区域,内部区域具有多个六边形比特单元,这些比特单元被布置得彼此毗邻。

10. 根据权利要求 9 所述的二维代码检测器,

其中所述多边形外部区域检测设备检测到四边形外部区域作为所述多边形外部区域,并且所述比特单元识别设备包括:

根据由多边形外部区域检测设备检测到的所述四边形外部区域的形状和对应的物理外部区域的实际形状之间的关系，计算代表二维投影变换的信息的装置；以及使用代表二维投影变换的信息计算图像中的每一个比特单元的位置的装置。

11. 根据权利要求 9 所述的二维代码检测器，

其中，多边形外部区域具有六边形的形状。

12. 根据权利要求 11 所述的二维代码检测器，

其中，每一个比特单元都具有正六边形的形状。

13. 根据权利要求 12 所述的二维代码检测器，

其中，比特单元是这样排列的，以便连接六边形外部区域的相对顶点的线和与其交叉的正六边形比特单元的任何相对的边成直角地交叉。

14. 根据权利要求 9 所述的二维代码检测器，

其中，比特单元具有其高度和宽度相等的六边形的形状。

15. 根据权利要求 14 所述的二维代码检测器，

其中，外部区域具有四边形的形状。

16. 根据权利要求 15 所述的二维代码检测器，

其中，使用四个六边形单元来确定二维代码的取向，所述四个六边形单元中的每一个都位于四边形外部区域的四个顶点的其中一个的附近。

用于检测二维代码的方法和二维代码检测器

技术领域

[0001] 本发明涉及二维代码的配置，并涉及用于从图像检测二维代码的方法和设备。

背景技术

[0002] [相关技术 1]

[0003] 图 18 示出了叫做 MaxiCode 的已知的二维代码。在此二维代码中，在位于二维代码的中心部分的具有不同半径的同心环状的黑白区域的周围有规律地排列着微小的正六边形单元。每一个正六边形单元都被涂成黑色或白色，从而代表了二进制信息。通过使用照相机等等捕获代码的图像，读出代码的内容。在下文中将把这种类型的二维代码称为“二维代码 1”。

[0004] 图 19 和 20 所示的二维代码也是已知的。这些二维代码中的每一个都具有可以根据亮度从背景区别开来的正方形外部区域，以及通过将代码的内部区域划分为大小相等的区域而形成的小的正方形单元。信息通过小的正方形单元的亮度 / 暗度来表达。通常，这样的二维代码的图像是通过照相机来捕获的，并且从捕获到的图像识别和检测代码的周界（轮廓）或外部形状。根据检测到的周界（轮廓）内分配的每一个小的正方形单元的亮度，读出由代码代表的信息。在下文中，将把这种类型的二维代码称为“二维代码 2”。

[0005] 一般而言，“二维代码 1”和“二维代码 2”要被附加到产品等中，并且用于管理诸如制造信息和运输信息之类的信息，其中制造信息包括产品号码、产品 ID 以及制造日期，运输信息包括产地、目的地以及运输日期 / 时间。读取这样的二维代码是根据这样的前提执行的：代码的图像是由照相机大致从代码的正面的位置（即，照相机的光轴大致正好垂直于代码的位置）捕获的，并且捕获的代码图像充分大。

[0006] [相关技术 2]

[0007] 在混合现实 (MR) 的领域中（其中虚拟对象被重叠在真实空间上以便显示在真实空间中），执行下列处理以呈现虚拟图像。估计真实的照相机相对于位于真实空间中的正方形标记的位置和取向。根据真实的照相机的估计位置和取向，设置虚拟照相机的参数。然后，根据设置的参数，呈现虚拟图像。

[0008] 这样的 MR 应用中使用的技术的一个示例是 ARToolkit，这在 X. Zhang, S. Fronz, N. Navab：“Visualmarker detection and decoding in AR systems : A comparative study,” Proc. of International Symposium on Mixed and Augmented Reality {ISMAR' 02}，2002（以下简称为“Zhang 等人的著作”）中进行了描述。在 ARToolkit 中，使用了具有正方形周界（轮廓）或外部形状的标记。从所捕获的图像检测标记周界（轮廓）的四边形形状。然后，根据检测到的四边形形状与原始正方形形状相比如何变形（即，检测到的四边形由于透视投影的效果而变形），估计照相机的位置和取向。此外，为了将一个标记与多个标记区别开来，用不同的图案施加于各个标记的内部。通过对从所捕获的图像获得的四边形的内部的图案执行称为“图案匹配”的处理，来区别每一个标记。在下文中，将把这种类型的二维代码称为“二维代码 3”。

[0009] 图 22 示出了 MR 应用中使用的标记的另一个示例, 这是“Zhang 等人的著作”中所描述的。此标记与“二维代码 3”的不同之处在于, 标记的内部被分成 4×4 网格图案。此外, 根据其中心位于一个划分的网格单元的中心的每一个圆形的区域是否为黑色, 实现区别每一个标记的功能。然而, 此标记与“二维代码 3”的相同之处在于, 根据原始正方形轮廓, 估计照相机的位置和取向。在下文中, 将把这种类型的二维代码称为“二维代码 4”。

[0010] 在 MR 领域提出了具有正方形轮廓的其他类型的标记。例如, 如图 23 所示的标记具有多个小的正方形单元, 位于其正方形轮廓的外面。这些小的正方形单元被用来标识正方形轮廓在所捕获的图像内的旋转方向。标记的内部被分成 4×4 个正方形单元。用于区别标记的信息由每一个小的正方形单元的亮度来代表。在下文中, 将把这种类型的二维代码称为“二维代码 5”。

[0011] 上文所描述的 MR 应用的“二维代码 3”、“二维代码 4”和“二维代码 5”所共有的一个特征是, 它们都具有正方形轮廓。

[0012] “相关技术 1”中所描述的“二维代码 1”具有位于代码的中心的同心圆, 这些同心圆被小的正六边形单元围绕起来。这些正六边形单元彼此联锁 (interlock) 起来, 每一个单元都被涂成黑色或者白色。这些黑色和白色单元代表了二进制数据。使用从最里边的同心圆以 60 度的间隔辐射出的线作为参考, 标识六边形单元的位置。换句话说, 只有在从代码 (索引) 的大致正面的位置捕获“二维代码 1”的图像的情况下 (即, 使用其光轴沿大致着索引 (index) 所附着的平面的法线方向的照相机捕获图像时), 才可以检测到索引。这对“二维代码 2”也成立。执行上文所描述的相关技术 1 的二维代码的图像的捕获的前提是, 索引图像从照相机的光轴大致正好垂直于索引平面的位置捕获, 而不是从照相机的光轴不正好垂直于索引平面的任意位置捕获图像。

[0013] 上文所描述的“二维代码 3”、“二维代码 4”和“二维代码 5”中的每一个都具有正方形轮廓的外部区域。此区域的亮度低于背景的亮度。在相关技术 2 的技术中, 执行估算, 以便根据二维代码的轮廓的变形来确定照相机相对于二维代码的位置和取向。变形是由于三维特征投影到二维平面中而发生的。即, 执行相关技术 2 的索引的图像捕获的前提不是它们的图像从正好垂直于索引平面的方向捕获。相反地, 执行这些索引的图像捕获的前提是, 图像从可能偏离索引平面的法线方向的方向捕获。

[0014] 对于“二维代码 3”, 为了区别各个索引, 通过图案匹配技术, 识别正方形轮廓内部的任意图案。预先独立地存储了索引内部的图案, 供进行图案匹配时使用。然后, 在通过从所捕获的图像中剪辑索引的内部区域所获得的图像被规范化之后, 将规范化的图像和存储的图案进行比较, 以便根据规范化的图像是否相似于存储的图案来识别索引。然而, 此技术有一个缺点: 随着要使用的图案的数量或要被同时捕获图像的索引的数量增大, 要进行图案匹配的对的数量也增大。这可能会导致进行计算所需要的时间显著地增加。此外, 图案的数量的增加产生相似的图案, 会导致可能发生的对图案的错误识别。如此, 实际可以同时使用不超过大约几十个图案。

[0015] 另一方面, 在“二维代码 4”和“二维代码 5”中, 内部区域被相等地分成小的正方形单元, 由每一个单元的亮度 (黑色或白色) 来代表比特值。如此, “二维代码 4”和“二维代码 5”没有上文所描述的与“二维代码 3”关联的缺点。然而, 如下面所描述的, “二维代码 4”和“二维代码 5”仍有改进的余地。

[0016] 为了使二维代码包含尽可能多的信息,必须增大比特单元的数量。然而,由于必须从所捕获的图像中识别各个比特单元,因此,在缩小每一个比特单元的大小方面存在限制。换句话说,二维代码的可识别性取决于投射到图像上 / 或图像上捕获的比特单元的大小。一般而言,对于 MR 应用,较小的二维代码是合乎需要的。然而,当要记录在二维代码中的信息量增大时,二维代码的大小必须在物理上增大,以便它在所捕获的图像中看起来充分大。即,二维代码的大小和其中包含的信息量处于折衷的关系。如此,为了缩小二维代码的大小并且还要尽可能多地记录信息,需要在代码的内部区域高密度地排列比特单元。

[0017] “二维代码 4”具有排列在正方形网格中的圆形的内部比特单元。利用这种类型的二维代码,当缩小每一个圆形比特单元的大小时,圆形单元之间的间隔的大小变得重要。例如,当被涂成黑色的圆形比特单元的大小被缩小到所捕获的图像中的像素的宽度的大小时,单元看起来是与单元周围的白色空间混合,从而产生了中性亮度的区域。这会导致在读出比特时错误的数量增加。即,希望内部单元的排列彼此靠近,它们之间没有间隔。

[0018] 在“二维代码 5”中,内部区域被相等地分成彼此相邻的正方形比特单元。如此,这种类型的二维代码没有与“二维代码 4”关联的上述问题。此外,即使在标记所在的图像从不正好垂直于标记平面的方向被捕获的情况下,也可以检测到该标记。如此,将从哪个方向捕获小的内部正方形单元的图像是不明确的。即,这些单元的何种类型的四边形形状将出现在投影的图像中是不明确的。在“二维代码 5”中,根据所捕获的图像中的各个正方形单元的亮度,执行将代码转换为基于比特的表示法的处理。如此,当读取亮度时,将读出每一个正方形单元的中心点(亮度参考点)的亮度。为了准确地读取亮度,必须捕获代码的图像,以便在所捕获的图像中的正方形单元的亮度参考点之间有足够的恒定的距离。换句话说,当原始索引中的单元的图像被投影到图像平面中时,越是接近于圆形形状(其中心位于亮度参考点中),投影图像中的每一个单元都出现在图像平面中,那么,原始索引中的单元的大小的减小程度就越大。通过减小单元的大小,会增大为整个索引区分配的比特的数量,每一个单元都代表了一个比特值。相比之下,当单元的形状是正方形时(表明每一个单元看上去是四边形,且各角位于图像平面中),这些单元可能不会被有效地高密度地排列起来。

[0019] 另一方面,在“二维代码 4”中,单个比特单元是圆形的。然而,在单元之间有不需要的间隔,如此,这些单元可能不会被高密度地排列起来。

发明内容

[0020] 本发明是在考虑到上述情况的背景下作出的。从上文可以看出,在“二维代码 4”和“二维代码 3”在比特单元的排列的密度方面有改进的余地。

[0021] 根据本发明的一个方面的二维代码,其包括可以根据亮度或颜色从背景区别开来的多边形外部区域,以及位于与外部区域的周界具有每一个单元的亮度或颜色来代表信息。单元被放置得彼此毗邻,每一个单元都具有六边形的形状。

[0022] 根据本发明的一个方面的检测二维代码的方法包括下列步骤:从背景中检测在图像上投影的二维代码的多边形外部区域,根据图像中的多边形外部区域的形状判断内部比特单元的位置,以及根据比特单元的亮度或颜色获取比特值。二维代码包括可以根据亮度或颜色从背景区别开来的多边形外部区域,以及位于与六边形外部区域的周界具有预定距

离处的内部区域，内部区域具有多个六边形比特单元，比特单元被放置得彼此毗邻。

[0023] 根据本发明的一个方面的二维代码检测器包括：多边形外部区域检测设备，适用于从背景中检测在图像上投影的二维代码的多边形外部区域；比特单元识别设备，适用于根据图像中的多边形外部区域的形状识别内部比特单元的位置；以及比特值获取设备，适用于根据图像中的比特单元的亮度或颜色获取比特值。其中，二维代码包括能够根据亮度或颜色与背景区别开来的多边形外部区域，以及位于与多边形外部区域的周界具有预定距离处的内部区域，内部区域具有多个六边形比特单元，这些比特单元被布置得彼此毗邻。

[0024] 通过下面的参考附图进行的详细描述，本发明的其他特征和优点将变得显而易见，其中，相同的附图标记在整个图中表示相同或类似的部分。

附图说明

[0025] 本说明书收入的并构成本说明书的一部分的附图说明了本发明的实施例，并且与说明书一起用于说明本发明的原理。

[0026] 图 1A 到 1C 说明了根据本发明的示范性实施例的二维代码的配置。

[0027] 图 2A 到 2C 说明了根据本发明的示范性实施例的另一个二维代码的配置。

[0028] 图 3 是说明了根据本发明的示范性实施例的二维代码检测器的方框图。

[0029] 图 4 是说明了由根据本发明的示范性实施例的二维代码检测器所执行的处理过程的流程图。

[0030] 图 5A 到 5E 是说明了用于检测二维代码的正方形轮廓的方法的示意图。

[0031] 图 6 是示出了二维代码的原始六边形轮廓和在投影的图像上获得的二维代码的轮廓，并说明了原始轮廓和投影轮廓之间的投影变换的示意图。

[0032] 图 7 是说明了根据检测到的六边形轮廓计算内部比特单元的中心位置的方案的示意图。

[0033] 图 8 是说明了二维代码的投影图像中的特定比特单元的中心点和邻近像素之间的关系的示意图。

[0034] 图 9A 到 9D 是说明了根据本发明的示范性实施例的二维代码的比特单元和根据已知技术的二维代码的比特单元之间的比较的示意图。

[0035] 图 10A 到 10D 说明了根据本发明的示范性实施例的二维代码的配置。

[0036] 图 11 是说明了由根据本发明的示范性实施例的二维代码检测器所执行的处理过程的流程图。

[0037] 图 12A 到 12E 是说明了用于检测二维代码的正方形轮廓的方法的示意图。

[0038] 图 13 是示出了二维代码的原始正方形轮廓和在投影图像上获得的二维代码的轮廓，并说明了原始轮廓和投影轮廓之间的投影变换的示意图。

[0039] 图 14A 到 14D 是说明了根据本发明的示范性实施例的二维代码的比特单元和根据已知技术的二维代码的比特单元之间的比较的示意图。

[0040] 图 15 说明了两种情况下的二维代码，其中一种情况在白色外部区域上提供了比特单元，另一种情况在黑色外部区域上提供了比特单元。

[0041] 图 16 说明了基于六边形轮廓的索引的示例，这是一种根据已知技术的二维代码类型。

- [0042] 图 17 说明了基于六边形轮廓的索引的示例,这是一种根据已知技术的二维代码类型。
- [0043] 图 18 说明了基于六边形轮廓的索引的示例,这是一种根据已知技术的二维代码类型。
- [0044] 图 19 说明了基于六边形轮廓的索引的示例,这是一种根据已知技术的二维代码类型。
- [0045] 图 20 说明了基于六边形轮廓的索引的示例,这是一种根据已知技术的二维代码类型。
- [0046] 图 21 说明了基于六边形轮廓的索引的示例,这是一种根据已知技术的二维代码类型。
- [0047] 图 22 说明了基于六边形轮廓的索引的示例,这是一种根据已知技术的二维代码类型。
- [0048] 图 23 说明了基于六边形轮廓的索引的示例,这是一种根据已知技术的二维代码类型。

具体实施方式

- [0049] 现在将根据附图详细描述本发明的优选实施例。
- [0050] [第一示范性实施例]
- [0051] 请参看图 1A 到 1C,它们示出了根据本发明的示范性实施例的二维代码。图 1A 示出了二维代码 102,它具有白色的正六边形外部区域,并在黑色背景 101 上形成。在二维代码 102 的内部,在与二维代码 102 的正六边形周界(轮廓)具有预定距离处,提供了一个区域。此内部区域被分成正六边形单元,每一个单元都被涂成黑色或者白色。图 1B 中示出了每一个单元和外周界(轮廓)之间的关系。在此示例中,六边形单元被排列成相对于六边形周界(轮廓)的对角线倾斜 30 度。每一个六边形单元和外周界(轮廓)之间的间隔的长度被设置为在对角线方向等分六边形单元的线的长度的一半。图 1C 示出了此排列。如该图所示,由虚线表示的三角网作为额外的线覆盖在图 1B 中的二维代码上。此三角网由平行线的组每隔一定间隔构成,每 120 度角布置一组平行线。
- [0052] 如此,在图 1A 到 1C 中所示出的二维代码 102 中,六边形单元是这样排列的,以便在对角线方向连接六边形的外周界(轮廓)的顶点的线和正六边形单元的任何对边成直角地交叉。
- [0053] 下面,将描述用于读取二维代码的二维代码检测器的优选配置的示例。图 3 是示出了用于检测图 1 中所示的这样的二维代码的二维代码检测器的配置的方框图。
- [0054] 二维代码检测器包括用于捕获图 1 中的二维代码的图像的照相机 301。所捕获的图像被根据总线控制器 304 的仲裁通过图像加载单元 302 写入到存储器 305 中。存储器 305 中记录了用于执行根据本发明的示范性实施例的处理过程的程序。CPU(中央处理单元)303 根据该过程执行处理。总线控制器 304 对于 CPU 303、图像加载单元 302 以及存储器 305 之间的数据的输入 / 输出执行仲裁。
- [0055] 请参看图 4,流程图示出了图 3 中所示的二维代码检测器的处理过程。
- [0056] 在步骤 S401 中,由照相机 301 捕获二维代码的图像,并且通过图像加载单元 302

获取所捕获的图像。在步骤 S402 中, 检测作为在图像上投影的二维代码(索引)的外周界(轮廓)的六边形形状。由照相机 301 执行此图像捕获过程时, 不一定要让照相机的取向使得照相机的光轴正好垂直于索引平面。相反地, 当在 MR 应用中使用时, 可以从各种方向捕获索引图像。在此情况下, 图像平面的垂直和水平轴以及索引的六边形周界(轮廓)的各边不一定排成一线。

[0057] 下面将参考图 5A 到 5E 描述用于检测外周界(轮廓)的步骤 S402 的处理。图 5A 示出了从正好垂直于索引的平面的方向查看的索引。在此图中没有示出内部单元, 在对步骤 S402 的处理的此处描述中忽略这些单元。例如, 可以由照相机捕获图 5B 所示出的图像。甚至在索引具有正六边形形状的情况下, 所捕获的图像中的索引也可以看起来像凸起的六边形, 除非索引是从照相机的光轴垂直于索引平面的位置捕获的。此外, 图像中的索引也可以取向为任何方向。此外, 所捕获的图像中的黑色区域和白色区域用于代表亮度的级别, 黑色区域和白色区域之间的边界区域代表中间亮度级别。图 5C 示出了通过二进制化图 5B 所示的图像而获得的二进制图像。此图像二进制化可以通过简单的阈值处理来实现。然而, 可以应用任何二进制化方案来获得这样的图像。从此二进制图像, 通过像素跟踪技术, 检测形成白色区域的周界(轮廓)的像素, 以便获得形成了具有图 5D 所示出的一个像素的宽度的闭环的像素排列。当对此像素排列(alignment)执行虚线近似法时, 获得了图 5E 所示的六边形。此时, 在索引的原来的形状不是六边形的情况下, 可以通过虚线近似法发现, 断点的数量不是六个。如此, 可以判断原来的形状是否是六边形。

[0058] 在步骤 S402 的处理的上述描述中, 忽略了索引的内部区域中的比特单元。然而, 由于在内部比特单元和六边形周界(轮廓)之间提供了空间, 因此, 内部单元不影响步骤 S402 的处理, 只要索引在图像平面中占据了比预定面积更大的面积(即, 如此捕获图像, 以便空间的宽度大于所捕获的图像中的一个像素的宽度)。

[0059] 然后, 在步骤 S403 中, 根据检测到的六边形轮廓计算各个内部比特单元的中心位置。下面将参考图 6 描述此计算。如该图所示, 作为索引的原始轮廓的正六边形的顶点被排序为 (X_n, Y_n) ($n = 0, 1, \dots, 5$), 而在图像中检测到的六边形轮廓的顶点被排序为 (x_n, y_n) 。这些六边形通过二维投影变换相关联, 该二维投影变换通过下面的公式来表达。

$$[0060] \quad x_n = \frac{H_{11}X_n + H_{12}Y_n + H_{13}}{H_{31}X_n + H_{32}Y_n + 1}, y_n = \frac{H_{21}X_n + H_{22}Y_n + H_{23}}{H_{31}X_n + H_{32}Y_n + 1} \quad (1)$$

[0061] 此关系表达式可以通过如下所示的矩阵运算而写成齐次坐标变换公式。

$$[0062] \quad \begin{bmatrix} wx_n \\ wy_n \\ w \end{bmatrix} = \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & 1 \end{bmatrix} \begin{bmatrix} X_n \\ Y_n \\ 1 \end{bmatrix} \quad (2)$$

[0063] 在此情况下, 矩阵

$$[0064] \quad \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & 1 \end{bmatrix} \quad (3)$$

[0065] 称为“单应性矩阵”。顶点 (X_n, Y_n) 和 (x_n, y_n) 具有六个对应的点。可以导出公式(1) 和关联的公式(2) 的六个关系表达式。换句话说, 通过对联立方程进行求解, 可以从六个对应的点之间的关系导出上述单应性矩阵(3) 或公式(1) 中的八个参数 $H_{11}, H_{12}, H_{13}, H_{21}$ 、

H_{22} 、 H_{23} 、 H_{31} 和 H_{32} 。由于未知数的数量是八个,因此,为了唯一地确定八个未知数,需要四个对应的点。如此,从六个对应的点计算八个参数导致联立方程中存在冗余,希望使用最小二乘方法等等来计算参数。

[0066] 一旦获得了公式(1)或(2)中所示的关系表达式,空间(X, Y)中的任何点都可以映射到空间(x, y)。即,该公式允许计算索引中的任意比特单元的中心点出现在投影图像上的位置。例如,参看图7,可以获得中心点701在投影图像中的坐标。利用上文所描述的配置,可以在步骤S403的处理中计算内部比特单元的中心点出现在投影图像平面中的位置。

[0067] 随后,在步骤S404中,获取所获得的各个比特单元在图像平面中的中心位置的亮度值。由于在步骤S403的处理中计算出的中心位置是基于与六边形轮廓的图像捕获关联的投影变换的,因此,亮度值将不一定是整数值。即,要获取的亮度值不直接代表对应的像素。如此,亮度值可以被设置为最近的相邻像素的值或基于最近的四个像素的值通过诸如线性内插之类的内插方案来进行计算。使用图8对此进行了描述。假设在步骤S403中获取的比特单元在图像平面中的中心位置位于该图所示的点801处。点801的坐标值不是整数值,表明点801不对应于图像中的像素的中心。在步骤S404的处理中,获取此点801的亮度。在此示例中,使用上文所提及的最近的相邻像素的值进行的亮度计算意味着,距离点801最近的像素802的值被用作点801的亮度值。使用上文所提及的线性内插方案进行的亮度计算意味着,从最近的四个像素,即像素802、803、804和805的值内插亮度值。

[0068] 然后,在步骤S405中,所获得的亮度值被转换为比特值。最简单的比特转换方案是阈值处理,其中,通过亮度值和预定阈值之间的比较,亮度值被确定为0或1。可以使用步骤S402的二进制化处理中使用的值作为预定的阈值。然而,也可以使用任何其他方案来确定阈值。此外,步骤S405的处理可以并入到在步骤S406中对所有比特单元执行了比特转换过程之后执行的处理中,而不是在步骤S404和步骤S406的处理之间执行。在此情况下,也可能在步骤S403和步骤S404中的每个处理中获取每一个比特单元的亮度值,然后,根据亮度值的分布,将所获得的亮度值转换为比特值。

[0069] 在上文中,描述了根据此示范性实施例的从二维代码中读取比特值的处理,该二维代码呈现正六边形的形状,其具有内部被分成联锁的正六边形单元的区域。由于正六边形具有旋转对称性,因此,前面的处理过程没有提供例如关于投影图像中的哪个方向对应于索引的顶部的信息。鉴于此,可以使用这样的方案:事先向多个预定单元中的每一个分配一种颜色(黑色或白色),以便可以确定索引的旋转方向。作为替换,可以将另一个单元置于六边形轮廓的外部,以便可以确定旋转方向,与“二维代码5”的情况相同。

[0070] 在上文中,描述了从二维代码中读取比特值的方法,该二维代码具有正六边形轮廓,并在六边形轮廓的内部具有联锁的正六边形单元。这样的二维代码是有利的,原因有下面几个。

[0071] 图9A示出了根据此示范性实施例的与图1B所示的二维代码相同的二维代码,在该二维代码的内部区域中,排列有联锁的正六边形比特单元。图9B示出了根据与相关技术2的“二维代码5”中使用的方案相似的方案排列比特单元的情况。如该图所示,在内部区域排列着联锁的正方形单元。图9C示出了图9A所示的其中一个单元被覆盖到图9B所示的其中一个单元上的情况。从该图可以看出,六边形单元的相对的边之间的距离与正方形单元的相对的边之间的距离相同。如上文所描述的,从任何角度捕获这些二维代码的图像。

如此,当由照相机捕获图 9C 所示的单个单元的图像然后覆盖在一起时,可以获得图 9D 所示的图像。通常,照相机将像素排列在网格中,以便获取外部对象等等的亮度和颜色,作为像素的值。如此,如上文使用图 8 并结合步骤 S403 的处理所描述的,将从相对于投影图像中的比特单元的中心点最近的相邻像素的像素值或四个最近的像素的像素值获得图 9C 所示的每一个比特单元的亮度或颜色。从图 9D 可以看出,当比特单元是正方形时,在图像中投射的四边形的顶点附近的那些部分的大小增大,因为与图像中的六边形的顶点相比,这些顶点与中心点的距离更远。这表明,位于四边形的顶点附近的这些部分很少影响中心点的亮度或颜色。由于此二维代码的图像的捕获是在代码的图像可以从各个方向捕获的前提下执行的,因此,通过与中心点具有相同距离的点来定义比特单元,可以缩小比特单元的大小。即,希望比特单元的形状尽可能地接近于圆形,以降低比特单元的大小。虽然图 9A 所示的比特单元的大小是图 9B 所示的比特单元的 0.866 倍,但是,当从各个方向捕获它们的图像时,这些比特单元具有大致相同的最小可读的比特大小。尽管图 9A 的二维代码包含 19 个比特单元,而图 9B 的二维代码只包含 16 个单元。然而,图 9A 所示的总共 19 个比特单元的面积与图 9B 所示的总共 16 个比特单元的面积大致相同。如此,根据本发明的本示范性实施例,在二维代码中可以记录更多的比特数,而不会增大二维代码的大小。

[0072] 第一个示范性实施例的二维代码包括具有正六边形轮廓的外部区域,以及位于外部区域中的内部区域,在该内部区域中,如图 1 所示的那样布置了正六边形的比特单元。然而,轮廓和每一个比特单元的形状不需要是正六边形,可以是任何六边形形状。例如,二维代码可以具有如图 2 所示的那样的矩形轮廓。只要联锁的六边形比特单元被布置在与二维代码的轮廓有预定距离的位置处,并且可以根据投影图像平面中的(变形的)轮廓的形状,确定投影图像平面中的每一个比特单元的位置,就可以实现本发明的本质。

[0073] 此外,作为本发明的实施例的示例,描述了这样的情况:每一个比特单元都被分配了一种颜色(黑色或白色),每一个单元的亮度都通过阈值处理被转换为二进制值(1 或 0)。然而,作为上述实施例的修改方案,也可能使用多个亮度值,以便将它们转换为 n 个值。还可能向各个单元分配多个不同的颜色。也是在此情况下,如上文所描述的,可以使用相邻像素通过例如最近邻技术或诸如双线性内插法之类的内插技术获得各个单元的颜色。然后,选择与所获得的颜色最接近的多个注册的候选颜色(m 个颜色),以便分配给各个单元。如此,比特单元可以被转换为最多 m 个值。

[0074] [第二示范性实施例]

[0075] 图 10A 到 10D 说明了根据本发明的第二示范性实施例的二维代码。参看图 10A,在黑色背景 1001 上形成二维代码 1002。二维代码 1002 具有被涂成白色的并具有正方形轮廓的外部区域,以及在与轮廓有预定距离的位置处提供的内部区域。内部区域被分成被联锁在一起的六边形单元。每一个六边形单元都具有相等的高度和宽度,并被涂成黑色或白色。

[0076] 图 10B 示出了六边形单元和正方形轮廓之间的关系。图 10C 示出了一个六边形单元,以及一个四边形,作为具有相等的高度和宽度的额外的线。该图中的值“s”是六边形单元和四边形的高度和宽度。六边形单元的两个顶点位于四边形的右边和左边的中点,其余的四个顶点位于四边形的上边和下边中,其中每一个都与右边或左边有 s/4 的距离。图 10D 示出了四个六边形单元 1003、1004、1005 和 1006,其中每一个六边形单元都位于正方形轮廓的各个顶点的最邻近的附近处。这些六边形单元将在步骤 S1103 到 S1106 的处理中使

用。每一个六边形单元 1003、1004、1005 和 1006 都分别用预定的颜色填充：在此示例中，分别为白色、黑色、黑色、黑色。

[0077] 图 11 是示出了根据此示范性实施例的处理过程的流程图，该处理过程是使用上文所描述的二维代码检测器执行的。

[0078] 在步骤 S1101 中，由照相机 201 捕获如图 10 所示的二维代码的图像，通过图像加载单元 202 获取所捕获的图像。在步骤 S1102 中，从所捕获的图像检测作为二维代码的轮廓的四边形。

[0079] 参看图 12A 到 12E，将描述用于检测轮廓的步骤 S1102 的此处理。图 12A 示出了当从视轴垂直于索引平面的位置查看时该索引的形状。在此描述中，忽略了内部比特单元，并且在图中没有示出。图 12B 示出了由照相机捕获的索引的图像的示例。即使在索引的原来的形状是正方形的情况下，它在所捕获的图像中看起来也不是正方形，而是凸的四边形，除非它是从照相机的光轴垂直于索引的平面的位置捕获的。此外，在图像平面中，索引也可以朝任何方向。此外，每一个白色和黑色单元都代表图像中的亮度的级别，黑色和白色区域之间的边界区域被观察为具有中等级别的亮度。图 12C 示出了通过二进制化图 12B 所示的图像而获得的二进制图像。此图像二进制化可以通过简单的阈值处理来实现。然而，可以应用任何二进制化方案来获得这样的图像。从此二进制图像，通过像素跟踪技术，检测形成被涂成白色的区域的轮廓的像素，以便获得形成了具有图 12D 所示的一个像素的宽度的闭环的像素排列。当对此像素排列执行虚线近似法时，获得了图 12E 所示的四边形。此时，在原始索引的形状不是正方形的情况下，通过虚线近似法可以发现，断点的数量不是四个。如此，可以判断原来的形状是否是正方形。在步骤 S1102 的处理的上述描述中，没有提及索引的内部区域中的比特单元。由于在内部比特单元和正方形轮廓之间提供了空间，因此，内部单元不影响步骤 S1102 的处理，只要索引在整个图像中占据了比预定面积更大的面积（即，空间的宽度大于所捕获的图像中的一个像素的宽度）。

[0080] 然后，在步骤 S1103 中，根据图像平面中的检测到的四边形周界（轮廓），计算在二维代码的四个顶点的最近的附近提供的如图 10D 所示的各个内部比特单元 1003、1004、1005，以及 1006 的中心位置。下面将参考图 13 描述此计算。如该图所示，作为索引的原始轮廓的正方形的顶点被排序为 (X_n, Y_n) ($n = 0, 1, 2, 3$)，而在图像中检测到的四边形轮廓的顶点被排序为 (x_n, y_n) 。这些轮廓可以通过以下公式表达的二维投影变换相关联。

$$[0081] \quad x_n = \frac{H_{11}X_n + H_{12}Y_n + H_{13}}{H_{31}X_n + H_{32}Y_n + 1}, y_n = \frac{H_{21}X_n + H_{22}Y_n + H_{23}}{H_{31}X_n + H_{32}Y_n + 1} \quad (4)$$

[0082] 此关系表达式可以通过如下所示的矩阵运算而写成齐次坐标变换公式。

$$[0083] \quad \begin{bmatrix} wX_n \\ wY_n \\ w \end{bmatrix} = \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & 1 \end{bmatrix} \begin{bmatrix} X_n \\ Y_n \\ 1 \end{bmatrix} \quad (5)$$

[0084] 在此情况下，矩阵

$$[0085] \quad \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & 1 \end{bmatrix} \quad (6)$$

[0086] 叫做“单应性矩阵”。顶点 (X_n, Y_n) 和 (x_n, y_n) 具有四个对应的点。可以导出公式

(4) 和关联的公式 (5) 的四个关系表达式。换句话说,通过对联立方程进行求解,可以从四个对应的点之间的关系导出上述单应性矩阵 (6) 或公式 (4) 中的八个参数 H11、H12、H13、H21、H22、H23、H31 和 H32。

[0087] 一旦获得了公式 (4) 或关联的公式 (5) 中所示的关系表达式,空间 (X, Y) 中的任何点都可以映射到空间 (x, y)。即,该公式允许计算索引中的任意比特单元的中心点出现在投影图像上的位置。请注意,此示范性实施例的重要的特征是,单元 1003、1004、1005 以及 1006 的中心点位于正方形周界 (轮廓) 的对角线上。在正方形中,上边和下边之间的距离等于右边和左边之间的距离。如此,可以根据检测到的四边形周界 (轮廓) 获得图像平面上的四个六边形单元的中心位置,不管二维代码的正方形轮廓的图像所取向的方向。然而,不可能只根据图像中的四边形轮廓将各个单元 1003、1004、1005 和 1006 彼此区别开来。换句话说,在步骤 S1103 的处理中,获得四个六边形单元的中心点,而不将各个单元彼此区别开来。

[0088] 随后,在步骤 S1104 中,使用与参考图 4 描述的步骤 S403 的处理中使用的方案相似的方案,获取四个六边形单元的所获得的中心位置的亮度值。

[0089] 然后,在步骤 S1105 中,将每一个所获得的亮度值转换为二进制值 (黑色或白色)。最简单的比特转换方案是阈值处理,其中,通过亮度值和预定阈值之间的比较,亮度值被确定为 0 或 1 (白色或黑色)。可以使用步骤 S1102 的二进制化处理中使用的值作为预定的阈值。然而,也可以使用任何其他方案来确定阈值。通过步骤 S1103 到步骤 S1105 的处理,已经确定了位于各个顶点的最近的附近处的每一个六边形单元的颜色 (黑色或白色)。根据此结果,在步骤 S1106 中,可以确定图像中的二维代码的旋转方向 (取向)。在此确定过程中,如上文所描述的,由于单元 1003、1004、1005 和 1006 分别被涂成白色、黑色、黑色和黑色,因此可以通过比较在步骤 S1104 的处理中所获得的单元的那些亮度值来识别每一个单元的颜色。

[0090] 通过上文所描述的处理,判断投影到图像平面上的二维代码与原来的形状相比如何变形,以及二维代码在图像平面中取向为哪个方向。现在,根据上述结果,将执行读出内部比特单元的处理。在步骤 S1107 中,使用公式 (4) 或 (5) 计算图像平面上的任何一个比特单元 (除了图 10D 中的比特单元 1003、1004、1005 和 1006 之外) 的中心位置。然后,在步骤 S1108 中,获取图像平面中的单元的中心点的亮度值。此用于获取亮度值的处理类似于步骤 S1104 的处理。在步骤 S1109 中,所获得的亮度值被转换为比特值。此转换处理类似于步骤 S1105 的处理,其中,四个六边形单元的亮度值被转换为二进制值,并通过阈值处理等等来实现。在步骤 S1110 中,判断是否对所有比特单元执行了该处理。如果结果是否定的,则重复步骤 S1107 到步骤 S1109 的处理,直到对所有比特单元执行了该处理。此外,步骤 S1109 的处理可以并入到在步骤 S1110 中对所有比特单元执行了比特转换处理之后执行的处理中,而不是在步骤 S1108 和步骤 S1110 的处理之间执行。在此情况下,当执行步骤 S1107、S1108 和 S1110 的处理时,已经获得了所有比特单元的亮度值。因此,也可能根据步骤 S1107、S1108 和 S1110 的每个处理中所获得的亮度值的分布,将每一个亮度值转换为比特值。

[0091] 在上文中,描述了从二维代码中读取比特值的处理,该二维代码具有正方形轮廓,并且在正方形轮廓的内部具有联锁的单元,其中每一个单元都呈现六边形形状,具有相等

的高度和宽度。上文还描述了用于根据正方形轮廓检测六边形的检测器和方法。这样的二维代码是有利的，原因有下面几个。

[0092] 请参看图 14A, 其示出了二维代码, 该二维代码基本上与图 10B 中所示的二维代码相同。在此示范性实施例中实现的二维代码的正六边形轮廓的内部, 布置着相邻的比特单元, 每一个比特单元都具有六边形的形状, 具有相等的高度和宽度。图 14B 示出了这样的情况: 使用与用于上文所描述的相关技术 2 的“二维代码 5”的方案相似的方案提供相邻的正方形比特单元。如图 14A 和 14B 所示, 其中一个六边形单元和其中一个正方形单元带有阴影。在这些图中, 图 14A 的单元看起来小于图 14B 的单元。然而, 当它们被覆盖在一起时, 如图 14C 所示, 图 14B 的单元只比图 14A 的单元大较小的空间。如上文所描述的, 这些二维代码将由照相机从任何方向进行图像捕获。如此, 当图 14A 和 14B 的单元的所捕获的图像被覆盖在一起时, 可以获得如图 14D 所示的图像。通常, 照相机将像素排列在网格中, 以便获得外部对象等等的亮度和颜色等作为像素的值。如此, 如上文使用图 6 并结合步骤 S1103 和步骤 S1107 的处理所描述的, 将从基于投影图像中的比特单元的中心点的最邻近的相邻像素的像素值或四个最邻近的像素的像素值获得这些比特单元中的每一个比特单元的亮度或颜色。从图 14D 可以看出, 当比特单元是正方形时, 图像平面中的四边形的顶点的周围的那些部分的大小增大, 这是因为与六边形的顶点相比, 这些顶点与中心点的距离更远。这表明, 这些部分很少影响中心点的亮度或颜色。由于将从各个方向捕获这些二维代码的图像, 因此, 通过与中心点具有相同距离的点来定义比特单元, 可以缩小比特单元的大小。即, 希望比特单元的轮廓的形状尽可能地接近于圆形, 以降低比特单元的大小。换句话说, 图 14A 和 14B 的二维代码在根据该前提被捕获图像时, 这些比特单元具有大致相同的最小可读比特大小。然而, 图 14B 中的二维代码只有 16 个比特单元, 而图 14A 中的二维代码具有 18 个比特单元。如此, 根据本发明的示范性实施例, 在二维代码中可以记录更多的比特数, 而不会对代码的最小可识别大小产生负面影响。

[0093] 此外, 将每一个六边形比特单元的高度和宽度设定得相等会产生下列优点。当代码用于 MR 应用时, 二维代码的正方形轮廓是非常重要的特征。当估计照相机的位置和取向时, 在观察二维代码时, 正方形轮廓将不会产生特定方向性, 允许从各个视点进行观察。然而, 如果每一个比特单元的形状被设置为正六边形, 而正方形轮廓的形状仍保持原状, 则难以在正方形轮廓的内部布置联锁的比特单元, 同时在轮廓和单元之间提供恒定的距离。例如, 如图 16 所示, 在轮廓的顶边或底边和比特单元之间提供的距离比在轮廓的右边或左边和比特单元之间提供的距离长。换句话说, 与如图 10D 所示的二维代码不同, 位于图 16 的正方形轮廓的四个顶点附近的正六边形单元的四个中心点的位置相对于对角线不是对称的(即, 中心点不在正方形轮廓的对角线上)。如此, 在此情况下, 不可能根据通过步骤 S1103 到步骤 S1106 的处理检测到的四边形轮廓, 确定图像平面中的四边形轮廓的旋转方向或取向。由于各个比特单元的位置相对于对角线不是对称的, 因此, 不可能从在确定图像平面中的四边形轮廓的取向之前执行的轮廓检测结果中获得比特单元的中心点。另一方面, 当使用如图 10D 所示的具有六边形的形状的高度和宽度相等的比特单元时, 可能在正方形轮廓的顶点的最近的附近布置四个比特单元的中心位置, 以便相对于顶点是对称的(中心点位于对角线上)。在使用这些四个比特单元检测图像平面中的四边形轮廓的取向之后, 可以计算除了这四个比特单元之外的比特单元的中心点。此外, 也可能通过在正方形轮廓的外部

提供索引，确定图像平面中的四边形的取向，类似于相关技术 2 的“二维代码 5”。然而，在这样的情况下，增大了二维代码的总的大小。如果这样的大的空间可以用于索引，可以增大正方形轮廓的大小，如此，根据本发明的示范性实施例，可以在索引中分配更多的比特数。

[0094] 当希望在二维代码中表达更多数量的比特值时，可以布置更多数量的比特单元，如图 17 所示。如此，具有相等高度和宽度的比特单元有助于将比特单元布置在正方形中。

[0095] 在上述示范性实施例中，每一个比特单元都被涂成黑色或者白色，比特单元的亮度值通过阈值处理被转换为二进制比特值（1 或 0）。然而，也可能使用多个亮度值，它们将被转换为 n 个灰度值。还可能向各个单元分配多个不同的颜色。也是在此情况下，如上文所描述的，可以通过最近邻技术或诸如使用相邻像素的双线性内插法之类的内插法获得各个单元的颜色。然后，选择与所获取的颜色最接近的多个注册的候选颜色（m 个颜色），以便分配给各个单元。如此，比特单元可以被转换最多为 m 个色值。

[0096] 在上述示范性实施例中，正方形轮廓内的外部区域用白色填充，各个比特单元通过黑色和白色来表达。然而，外部区域也可以用黑色或任何其他半色调或颜色填充。此外，如图 15 所示，具有黑色边界区域的二维代码和具有白色边界区域的二维代码可以一起使用，以便可以通过外部区域的颜色表达额外的一个比特。

[0097] 此外，在上述示范性实施例中，描述了这样的情况：比特单元 1003、1004、1005 以及 1006 分别用白色、黑色、黑色和黑色填充，用于确定二维代码的旋转方向或取向。然而，也可以使用诸如黑色、白色、白色、白色之类的其他颜色布置。此外，具有白色、黑色、黑色以及白色的布置的二维代码和具有黑色、白色、白色以及白色的二维代码可以一起使用，以便可以通过这两种布局之间的差异表达额外的一个比特。

[0098] [其他示范性实施例]

[0099] 也可以使用由多个设备构成的系统实现与上文所描述的二维代码检测器的功能相似的功能。

[0100] 本发明也包含这样的布置：其中，用于实现上文所描述的实施例的功能的软件程序被提供到具有能够执行直接来自记录介质或通过有线或者无线通信读取的程序的计算机的系统或设备，以便通过执行提供到系统或设备的计算机的程序来实现等效的功能。

[0101] 如此，要被提供到计算机并安装在计算机中的用于实现上文所描述的实施例的功能的程序代码也是实现本发明的特征。即，用于实现上文所描述的实施例的功能的计算机程序可以包含在本发明中。

[0102] 在此情况下，程序不局限于任何形式，如目标代码、由解释器执行的程序、提供到操作系统的脚本数据等等。

[0103] 用于提供程序的记录介质的示例包括磁记录介质，如软盘、硬盘、磁带等等；光学 / 磁光记录介质，如磁光（MO）盘、光盘只读存储器（CD-ROM）、可记录 CD（CD-R）、可重写 CD（CD-RW）、数字多用途盘 ROM（DVD-ROM）、DVD-R、DVD-RW、等等；非易失性半导体存储器等等。

[0104] 用于通过有线 / 无线通信提供程序的方法的示例包括，在网络上的服务器上存储数据文件（程序数据文件），它们可以充当在客户端计算机上实现本发明的计算机程序，如计算机程序本身，具有自动安装功能的压缩的和自解压文件等等；并且将程序数据文件下载到对服务器进行访问的客户端计算机。在此情况下，程序数据文件可以被分段为多个分

段文件，并将分段文件存储在不同的服务器上。也就是说，服务器设备可以将用于实现本发明的功能处理的程序数据文件下载到多个用户。

[0105] 即，用于将实现上述示范性实施例的功能的程序数据文件下载到多个用户的服务器设备也包含在本发明中。

[0106] 此外，也可以作出这种布置，其中，根据本发明的程序被加密并存储在诸如 CD-ROM 之类的记录介质中，并在此状态下分发给用户，同时用于对加密信息进行解密的密钥信息从因特网主页下载给例如满足某些条件的用户，以便可以使用密钥信息执行加密的程序并安装到计算机中。

[0107] 此外，除了上文所描述的实施例的功能通过读出的程序在计算机上执行来实现之外，上文所描述的实施例的功能可以由在计算机上运行的操作系统来实现，该操作系统基于程序的指令执行部分或所有实际处理。

[0108] 此外，上文所描述的功能可以通过从记录介质读出的程序被写入到存储器中来实现，其中该存储器被提供到插入到计算机的功能扩展板或连接到计算机的功能扩展单元，之后，在计算机上运行的 OS 基于程序的指令执行部分或所有实际处理。

[0109] 由于在不偏离本发明的范围的情况下，可以作出本发明的许多显然广泛不同的实施例，应该理解，本发明不限于其特定实施例，除非如所附的权利要求所限定。

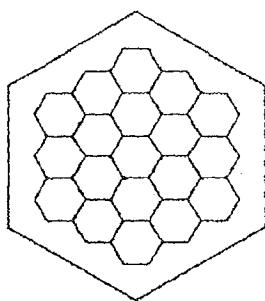
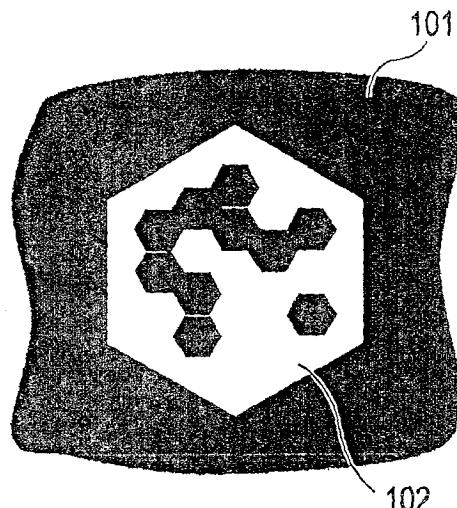


图 1B

图 1A

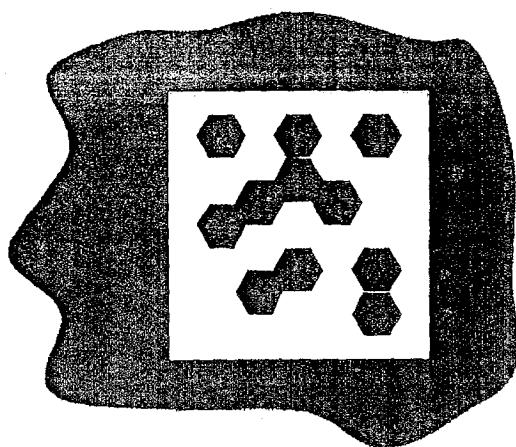
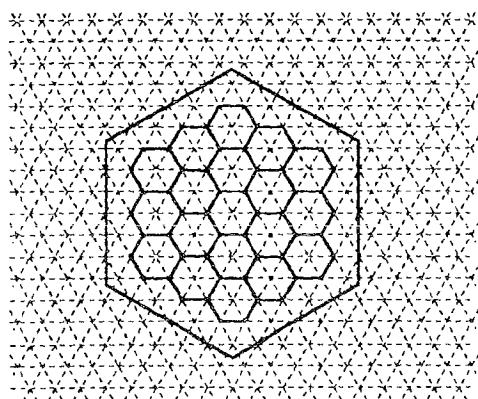


图 2A

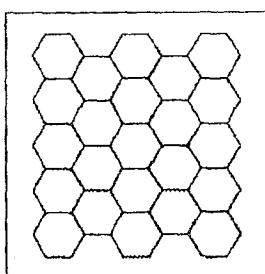


图 2B

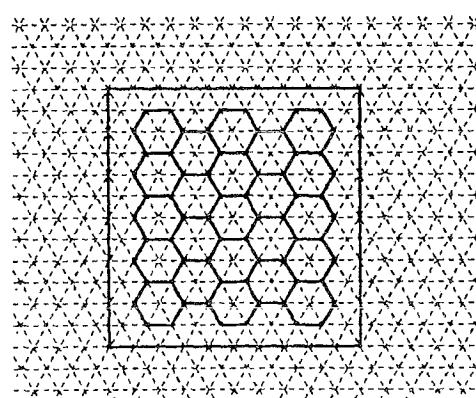


图 2C

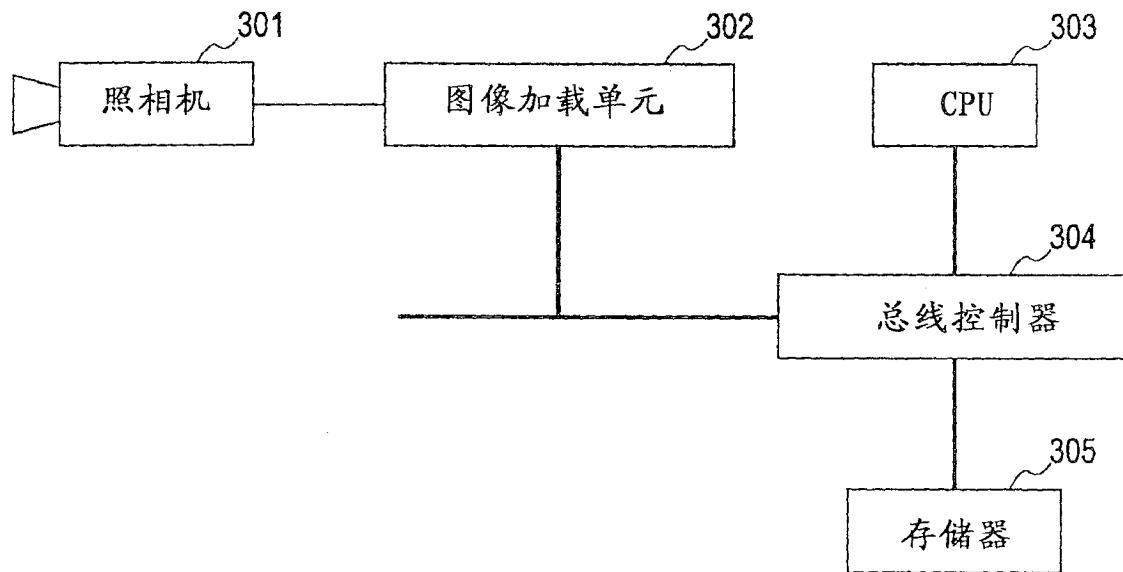


图 3

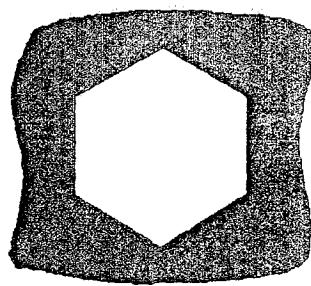
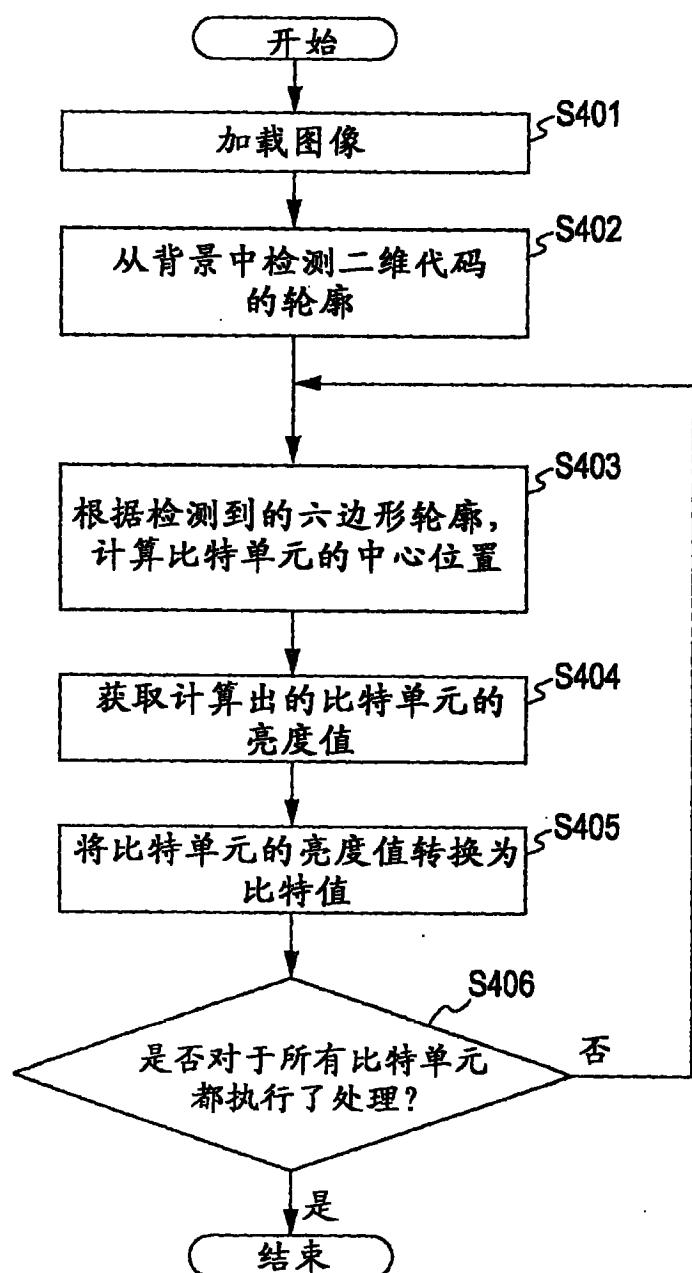


图 5A

图 4

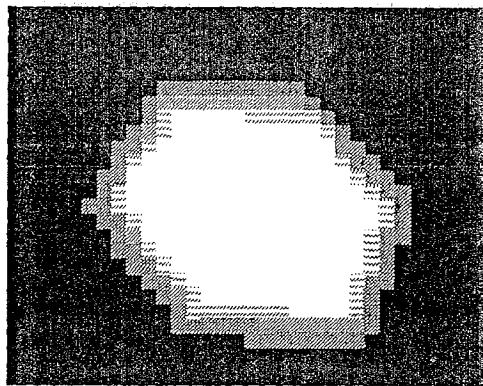


图 5B

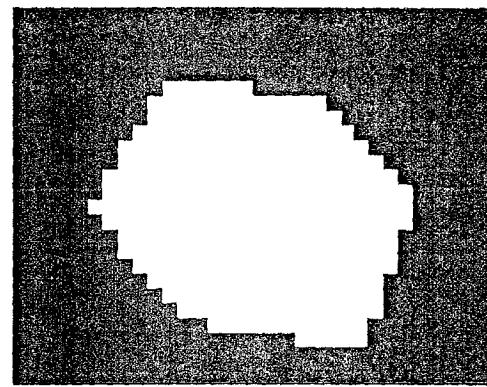


图 5C

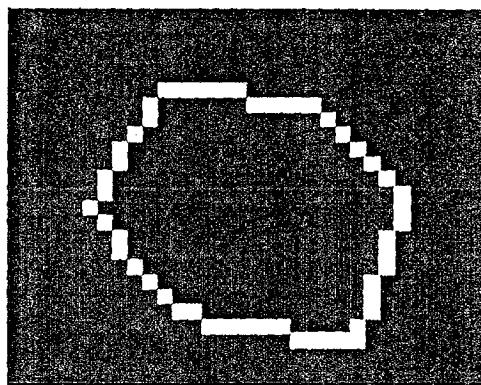


图 5D

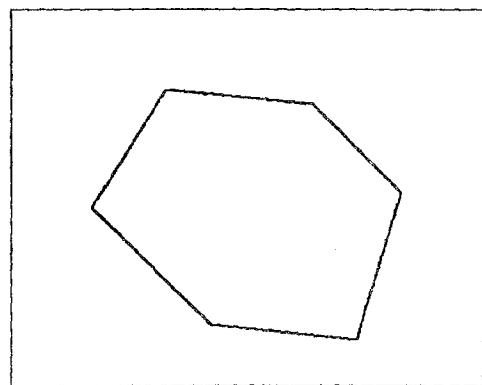


图 5E

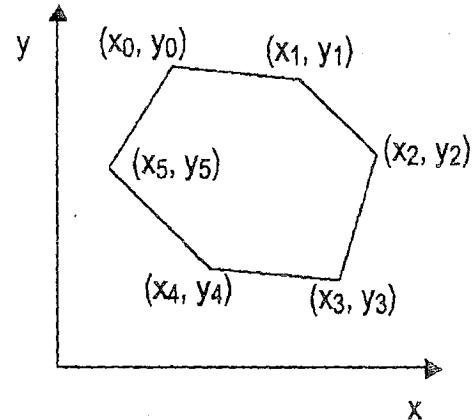
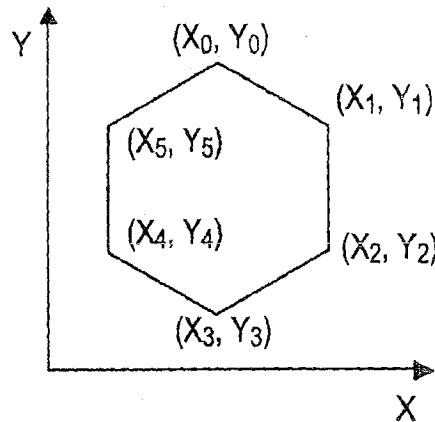


图 6

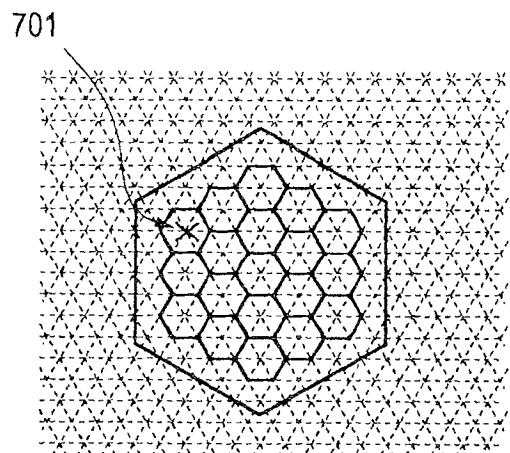


图 7

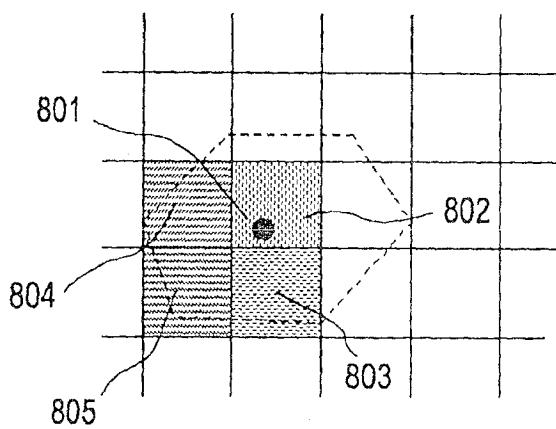


图 8

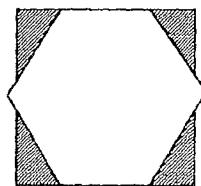
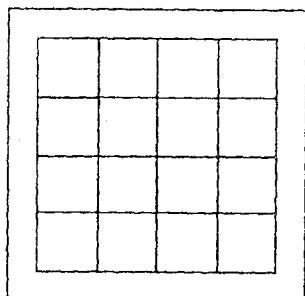
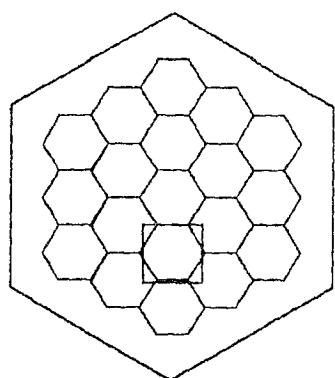


图 9A

图 9B

图 9C

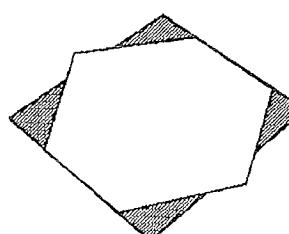


图 9D

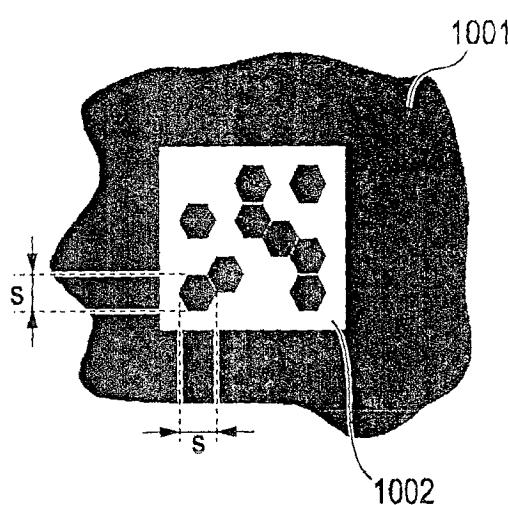


图 10A

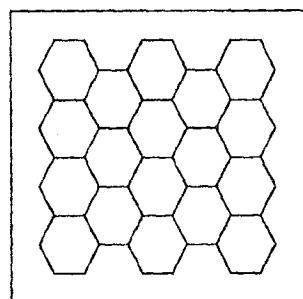


图 10B

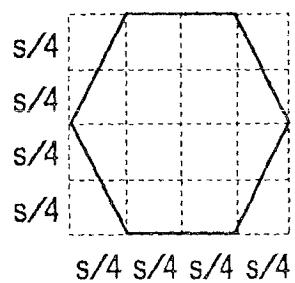


图 10C

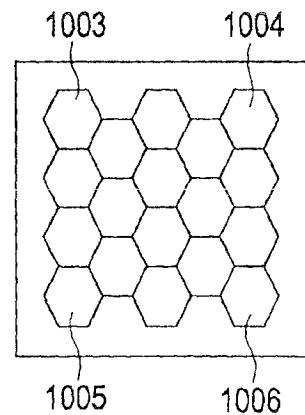


图 10D

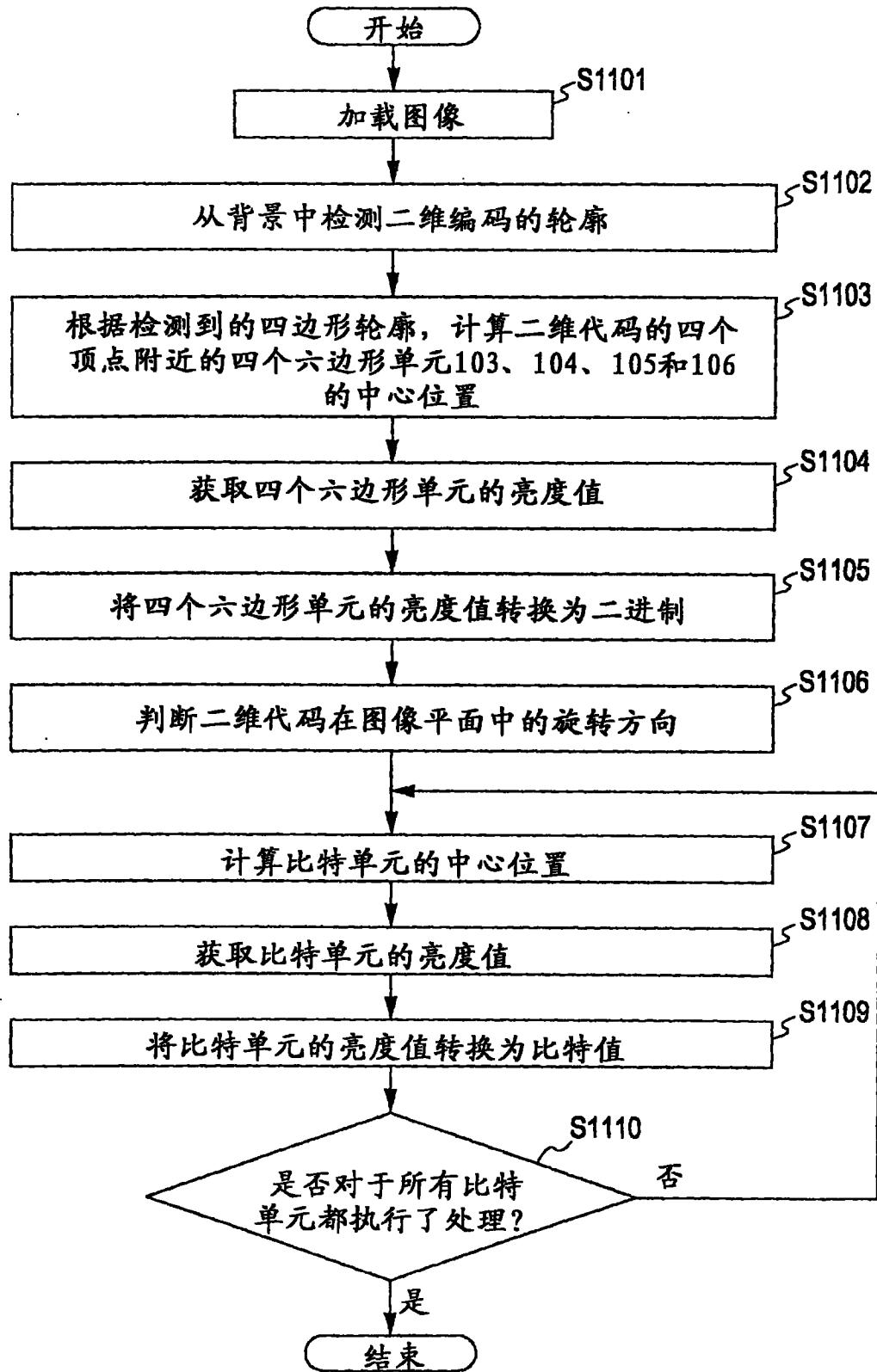


图 11

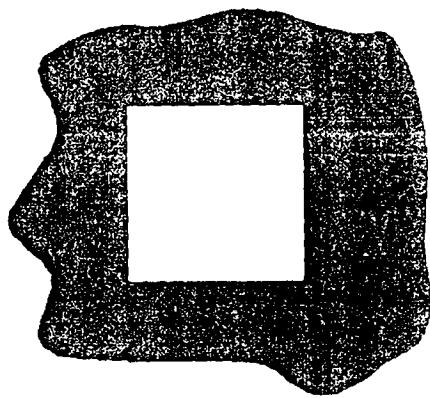


图 12A

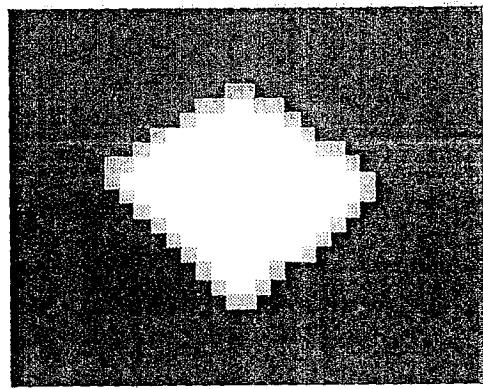


图 12B

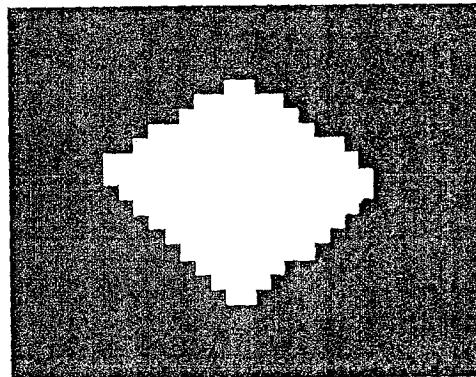


图 12C

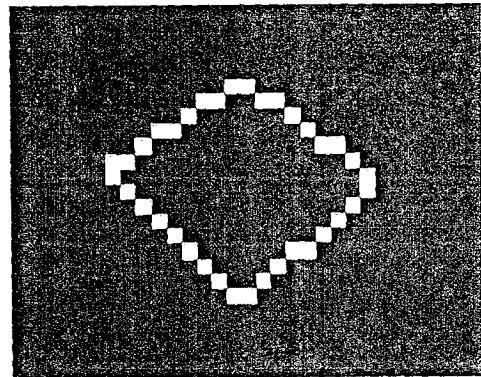


图 12D

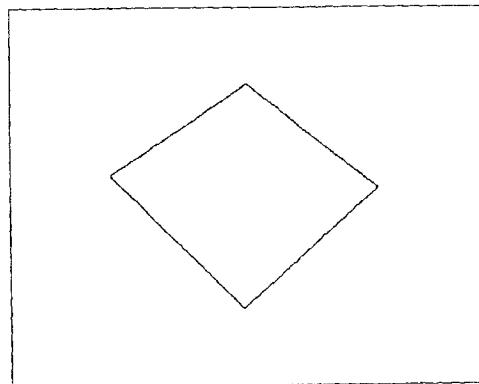


图 12E

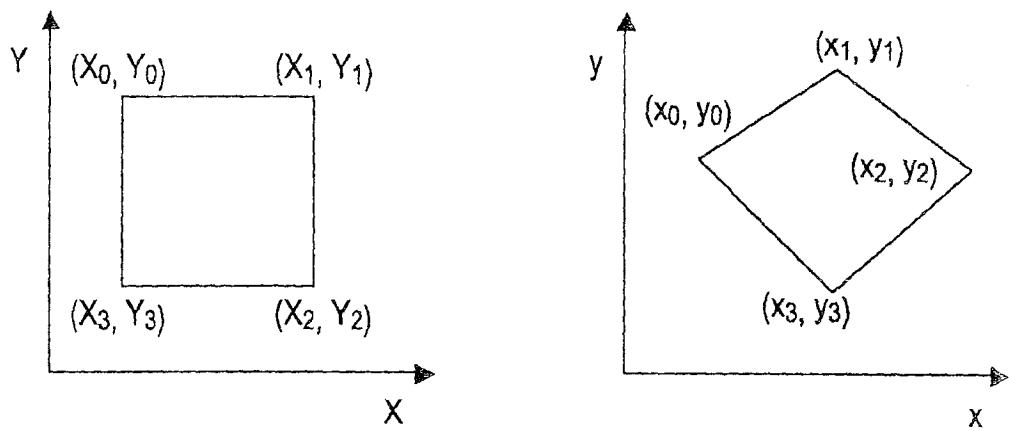


图 13

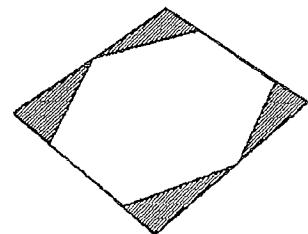
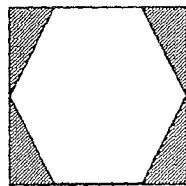
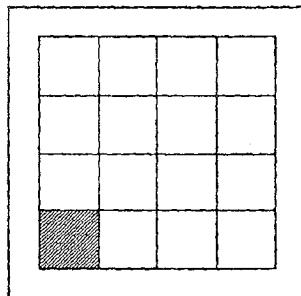
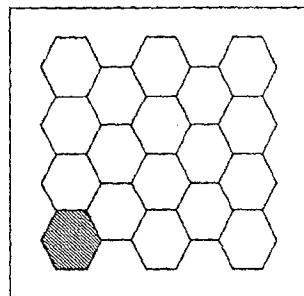


图 14C

图 14D

图 14A

图 14B

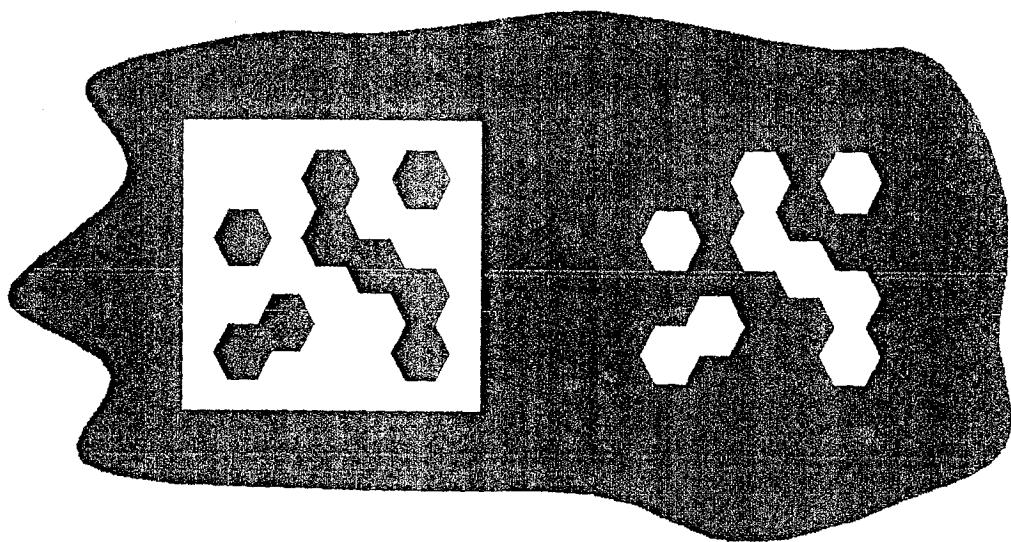


图 15

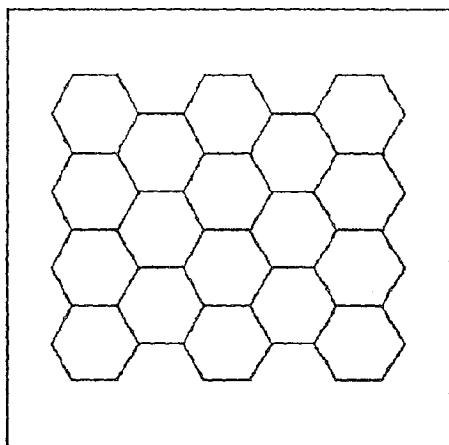


图 16

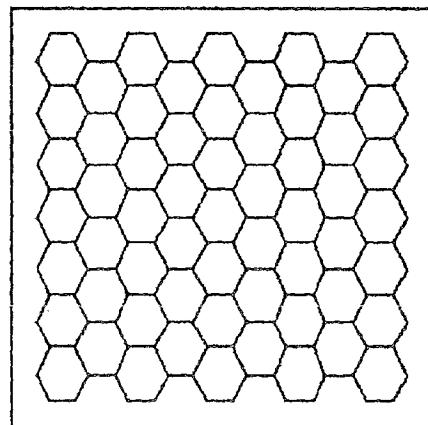


图 17

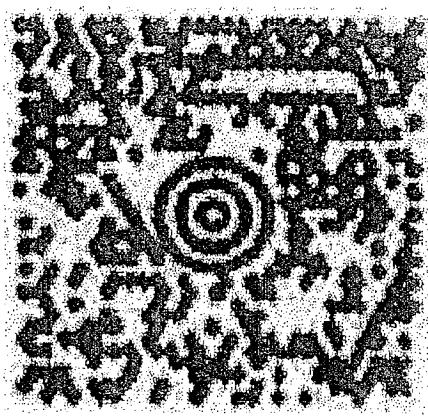


图 18

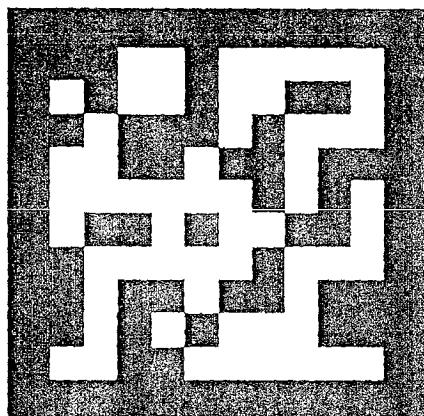


图 19

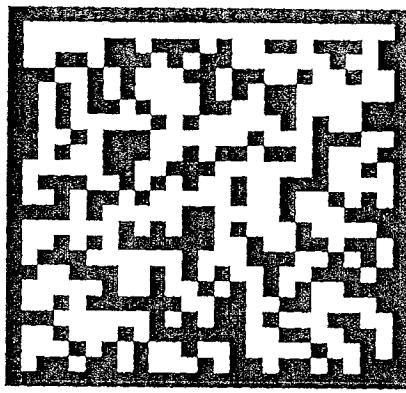


图 20



图 21

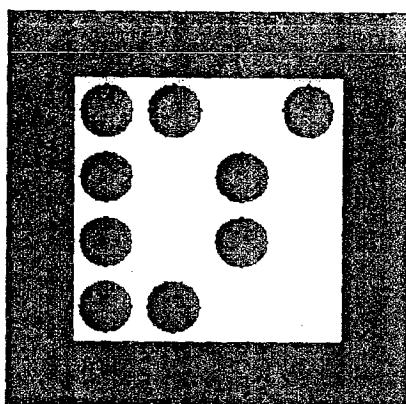


图 22

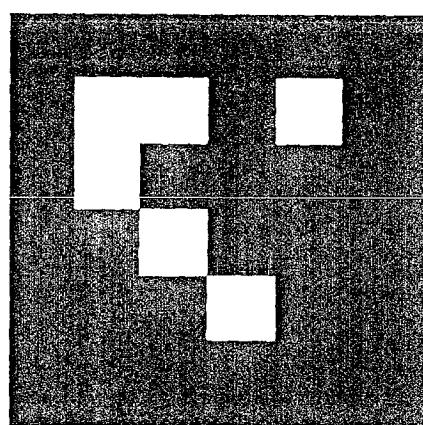


图 23