

[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
H04N 7/24 (2006.01)



[12] 发明专利说明书

专利号 ZL 200480004871.5

[45] 授权公告日 2009 年 8 月 19 日

[11] 授权公告号 CN 100531376C

[22] 申请日 2004.2.18

[21] 申请号 200480004871.5

[30] 优先权

[32] 2003.2.21 [33] FR [31] 03/02144

[86] 国际申请 PCT/FR2004/000364 2004.2.18

[87] 国际公布 WO2004/077915 法 2004.9.16

[85] 进入国家阶段日期 2005.8.22

[73] 专利权人 法国电信公司

地址 法国巴黎

[72] 发明人 C·热古

[56] 参考文献

US 2002/163501A1 2002.11.7

US 5896139A 1999.4.20

US 6263496B1 2001.7.17

审查员 张 军

[74] 专利代理机构 北京市中咨律师事务所
代理人 杨晓光 于 静

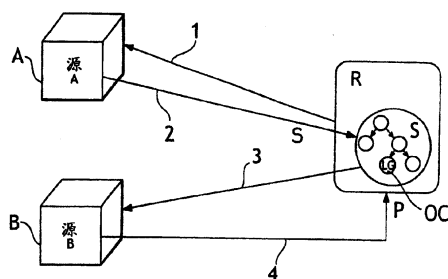
权利要求书 3 页 说明书 12 页 附图 2 页

[54] 发明名称

管理用于显示的图形动画描述的方法及其实施系统和接收机

[57] 摘要

一种管理用于显示的图形动画的描述的方法，该方法特征在于，图形动画由数据定义，所述数据定义将被显示的图形对象的空域-时域排列内容，并且在于，对于所述图形对象的至少一个而言，所述数据包括描述对应于所述图形对象的图元的数据，描述空域-时域排列内容的数据，以及描述独立地存储在适于被询问的存储装置中的图形对象图元的数据，为了得到对应于所述图元的数据，含有利用这样的图元定义的对象空域-时域排列内容包括了用于指定将要被询问的存储装置的数据。



1. 一种管理用于显示的图形动画描述的方法，所述图形动画是由用于描述被存储在第一源中的要显示的图形对象的空域-时域排列内容的一组数据来定义的，其中，所述图形对象是以被独立地存储在第二源中的图元的形式来表示的。

2. 根据权利要求 1 的方法，其中，所述第一和第二源包括服务器装置，所述服务器装置分别适于向远程客户端发送描述空域-时域排列的数据和描述图元的数据。

3. 根据权利要求 1 或 2 的方法，其特征在于，含有由独立地存储的图元所定义的对对象的空域-时域排列内容，包括了用于识别所述数据和/或其所存储的装置的数据。

4. 根据权利要求 1 的方法，其中，所述图元是{动作，多边形，持续时间}型的低级别图元。

5. 根据权利要求 1 的方法，其中，对应于景物图像的所述数据从所述第一源被接收、被解码并且通过预再现处理而被转换成再现图元，所述图元被存储在再现引擎的栈中，并且其中，接收自所述第二源的所述图元被接收、被解码并且被直接发送给所述栈而无需预再现处理。

6. 根据权利要求 1 的方法，其中，描述图形对象的空域-时域排列内容的所述数据组包括标识所述第二源的数据。

7. 一种接收机，包括用于从第一源接收描述要显示的图形对象的空域-时域排列内容的数据的装置，所述图形对象中的至少一个是以图元的形式来表示的，以及处理器装置，用于处理所述数据以显示所述空域-时域排列内容，所述接收机还包括用于从第二源独立地接收所述图元的装置。

8. 根据权利要求 7 的接收机，其特征在于，所述处理器装置直接显

示对应于为所述图形对象而接收的数据的所述图元，并在显示之前对所述空域-时域排列内容的剩余部分进行定位和/或定尺寸预再现处理。

9. 根据权利要求 8 的接收机，其特征在于，所述处理器装置包括图元的栈和再现引擎，所述再现引擎控制和管理在所述栈中存储的图形对象的显示，所述图元对应于为所述图形对象而接收的数据，所述图形对象与在退出所述预再现处理时针对空域-时域排列内容而得到的图元一起被发送到再现图元的栈。

10. 根据权利要求 9 的接收机，其特征在于，所述再现引擎利用显示触发命令在特定时间后动作，以触发所述对应于为所述图形对象而接收的数据的图元的显示。

11. 一种用于实施根据权利要求 1 的管理用于显示的图形动画描述的方法的系统，所述系统包括用于存储描述图形对象的空域-时域排列内容的数据第一源和用于独立地存储表示所述对象中的至少一个的图元的第二源。

12. 一种分解用于显示的图形动画图像的方法，其中，所述图像被分解为描述被存储在第一源中的要显示的图形对象的空域-时域排列内容的数据和指定了其中独立地存储有与所述图形对象中的至少一个相对应的图元的第二源的数据。

13. 根据权利要求 12 的方法，其特征在于，所述图形对象中的至少一个是用多边形形状来表示的，所述对象是以包括共同限定了所述多边形形状的点的坐标的图元的形式来表示的。

14. 根据权利要求 12 的方法，其特征在于，所述图形对象中的至少一个是基于至少一个三角形形状和/或梯形形状来限定的，所述对象是以提供对所述三角形形状和/或梯形形状的限定的图元的形式来表示的。

15. 根据权利要求 13 或 14 的方法，其特征在于，所述图元提供对所

述形状的颜色和/或纹理的限定。

管理用于显示的图形动画描述的方法及其实施系统和接收机

技术领域

本发明涉及图形动画描述技术。

更具体地说，本发明提出一种管理图形景物 (graphics scene) 的方法，以及用于实施该方法的存储系统和接收机。

背景技术

目前有多种图形动画表示格式。

它们采用两种主要途径：

- 一种是基于图形对象的空域 - 时域排列 (spatial-temporal arrangement) 的树状表示，该表示使得在图形对象和子对象之间进行精确的交互，但该表示也需要在如此显示之前进行称作“光栅化”的中间处理。
- 另一种途径是基于多边形架构的再现模式，并且采用简单的图元 (primitive) 来保证快速再现。

第一种途径对应于例如 W3C/SVG 及 MPEG-4/系统/BIFS 所采用的图形描述格式。然而，这第一种途径没有提供最优的图形再现。对于某些动画而言，它也导致计算成本过高，而无必要在其中采用这种技术。

第二种途径提供了有效的图形动画再现；然而，其不能与构成图形动画的图形子对象进行精确的交互，并且再现取决于接收机的显示特性。这第二种途径对应于诸如 Macromedia SWF 的图形格式以及例如常规地用于通过诸如 OpenInventor 的工具进行三维显示的显示表。

发明内容

本发明的一个目的是提出一种减缓现有图形表示技术的缺陷的技术，

现有图形表示技术或者缺乏交互性，或者缺乏图形再现的效率。

为此，本发明提供一种管理用于显示的图形动画描述的方法，该方法的特征在于，图形动画由用于描述将被显示的图形对象的空域-时域排列内容的一组数据定义，并且在于，对于所述图形对象的至少一个而言，所述一组数据包括描述对应于所述图形对象的图元的数据，描述空域-时域排列内容的数据，以及描述被独立地存储的图形对象图元的数据。

应注意到，所提出的技术，尤其在于通过结合多个向量图形表示级别，能够利用最适于给定动画的图形表示，从而节省存储空间。

许多图形表示或动画不需要以简单向量图元合成的形式被描述，但可受益于以较低级图形再现图元的列表的形式的表示。

低级图元为{动作, 多边形, 持续时间}({action, polygon, duration})型, 例如, 在其中所述动作为相加、替换, 或者打破由具有整数非向量坐标的多边形描述的形状。

通过作用于各种表示模式, 这种技术具有这样的进一步优点: 其能够使得图形再现引擎的性能处于完全控制之下, 尤其是通过空域-时域排列的非系统利用。

此外, 所提出的技术可容易地集成到大多数能够绘制向量形状的图形再现设备中。

该方法有利地单独具有或在任何技术上的可行结合中具有如下额外特征:

- 存储装置, 包括适于向远程客户发送数据的服务器装置, 该数据描述了将被显示的图形对象的空域-时域排列内容, 和/或描述图元的数据。

- 空域-时域排列内容, 含有由独立地存储的图元所定义的对象, 该空域-时域排列内容包括了用于识别所述数据和/或其所存储的装置的数据。

- 为显示图形动画, 接收对应于将被显示的图形对象的空域-时域排列内容的数据, 对从所述装置以这种方式接收的数据进行解码, 以及, 如果与此数据相对应的排列包括由独立地存储的图元定义的图形对象, 则接

收和解码与所述图元对应的数据。

- 直接显示对应于为所述图形对象而接收的数据的图元，并且在显示之前，对空域-时域排列内容进行预再现处理；以及

- 将对应于为所述图形对象而接收的数据的图元发送到具有在退出预再现处理时为空域-时域排列内容获得的图元的再现图元栈。

本发明还提供了一种接收机，其包括显示装置和用于接收和解码数据的装置，所述数据描述了将被显示的图形对象的空域-时域排列内容，该接收机的特征在于，其包括用于接收和解码数据的装置，所述数据被独立地存储，并对应于为所述对象在空域-时域排列内容中定义至少一个图形对象的图元，以及处理器装置，用于处理所述数据，以显示空域-时域排列内容和所述图元。

本发明还提供了一种系统，用于实施上面定义的管理将被显示的图形动画的描述的方法，该系统的特征在于，其包括这样的装置，在该装置中描述空域-时域排列内容的数据和描述图形对象图元的数据被独立地存储。

本发明还提供了一种运送一组数据的信号，所述数据定义了用于显示的图形对象和子对象的空域-时域排列，该信号的特征在于，对于至少一个图形对象，所述一组数据包括用于识别被独立地存储的图元的数据和/或用于识别其所存储的装置的数据。

本发明进一步提供了分解用于显示的图形动画图像的方法，该方法的特征在于，所述图像被分解为用于描述将被显示的图形对象的空域-时域排列内容的数据，以及，对于图形对象的至少一个而言，用于定义与其相对应的图元的一组数据，所述空域-时域排列内容对于所述图形对象而言包括了用于指定存储装置的数据，在该存储装置中存储了用于定义所述对象的所述图元的数据。

附图说明

从下面的参考附图给出的说明性的和非限定性的描述中，可显现出本

发明的其它特征和优点，其中：

- 图 1 是表示初始景物接收的示图；
- 图 2 是表示在图 1 中所示的接收机中实现的再现处理的示图；以及
- 图 3 是示图，示出了图像的解码。

具体实施方式

图 1 示出了接收机 R，例如移动电话，其与至少两个外部数据源（服务器 A 和 B）通信，并从该外部数据源接收用于描述将被所述接收机 R 显示的景物和图形对象的二进制数据流。

图形动画以如下方式加载：

接收机 R 请求来自服务器 A 构成的源的图形动画内容。

该服务器 A 向所述接收机 R 发送描述图形对象的空域-时域排列的内容 S（图形景物）。

这由箭头 1 和 2 所示，箭头 1 和 2 表示由接收机 R 向源 A 发送的内容请求，以及由所述源 A 向接收机 R 的对该内容的发送。

当所描述的图形景物包括合成的图形对象 OC 时，接收机 R 询问被指定的服务器 B 关于所述接收机 R 刚从服务器 A 接收到的信息，作为特定的服务器，从服务器 B 将得到与所述合成的对象 OC 相对应的图形图元 P。

那些图形图元有利地是低级的“动作，多边形，持续时间”型的图形图元。

图 1 中的箭头 3 和 4 表示由接收器 R 向服务器 B 发送的传输图形图元的请求，以及由服务器 B 向所述接收机 R 对此图元的传输。

现在参考图 2，其示出了在接收机 R 中实现的再现处理。

由该图可以清楚地看出，接收机 R 包括用于解码初始景物 S 的装置 5，以及用于解码图元 P 的装置 6，该图元 P 是由接收机 R 正在询问的服务器 B 向该接收机 R 发送的。

接收机 R 还包括处理器模块 MT，其包括预再现模块 PR 和再现引擎 MOT。

预再现模块 PR 接收与景物 S 的图像相对应的数据，并对其进行预再现处理，以将其转换为例如 OpenGL 型的再现图元。

预再现模块 PR 的一个功能是使得普通的图形表示适于将对其进行显示的特定设备。

特别地，模块 PR 从该普通的图形表示确定将在屏幕上显示的对象的精确坐标。具体地说，其定义了图像中心的坐标，x 轴和 y 轴的坐标，再现区域的尺寸等。

关于预再现处理的例子，可有利地参考下列文献，例如：

计算机图形学 - 原理与实践 - Foley - Van Dam - Feiner - Hugues - 对象的分层结构与简单 PHIGS - 几何模型，第 286 至 302 页。（*Computer Graphics - Principles and Practice - Foley - Van Dam - Feiner - Hugues - Object Hierarchy and Simple PHIGS - Geometric Modeling*, pp. 286 to 302.）

La realisation de logiciels graphiques interactifs - Collection de la Direction des Etudes et Recherches d'EDF; Travaux dirigés de l'Ecole d'été d'informatique du 7 au 27 juillet 1979; 第 15 至 23 页。 [交互式图形软件产品 - EDF 研究部收藏；1979 年 7 月 7 日至 27 日“数据处理”暑期学校指导材料（directed work）]。

在预再现处理后，得到的图元被存储在图元栈中，该图元栈由图形再现引擎 MOT 进行处理。

再现引擎 MOT 的作用是利用由预再现模块 PR 确定的位置和尺寸元素来控制对象的显示。

例如，由再现引擎 MOT 控制显示的图形对象以类似于如下文献中描述的格式被编码：

“ISO/IEC 14496-1: 2002 - 信息技术 - 音频-视觉对象编码, 第 1 部分: 系统”（*“ISO/IEC 14496-1: 2002 - Information technology - Coding of audio-visual objects, Part 1: Systems”*），其中可特别参考描述二维层和转换节点的段落，当然，同样地可以将本发明用于三维景物。

由再现引擎 MOT 进行的显示控制处理特别地用于管理在不同对象之间的显示冲突，例如，该处理可以是下面文献中描述的类型：

“ISO/IEC 14772-1: 1998 - 信息技术 - 计算机图形学与图像处理 - 虚拟现实模型语言” (**“ISO/IEC 14772-1: 1998 - Information technology - Computer graphics and image processing - The Virtual Reality Modeling Language”**).

当处理器模块 MT 将要准备合成的图形对象 OC 时，与之相对应的图元 P 被直接发送到再现引擎的处理栈，而无需预再现处理。

那些图元可直接在屏幕上显示，而不需要预再现处理，尤其不需要调整尺寸。

因此，通过在屏幕上直接显示由服务器 B 接收的图形图元，**LowGraphics** 对象的再现得以实现。

例如，引擎 MOT 处理图元栈，该图元栈包括由预再现处理以及接收机所接收的合成的图形对象的图元所产生的图元栈，例如，该处理可以是下面出版物中描述的类型：

- 由 J. D. Foley, A. van Dam, S. Feiner 和 J. F. Hughes 编写的 **计算机图形学原理与实践** (Addison-Wesley, 1990) (**Computer Graphics Principles and Practice** by J. D. Foley, A. van Dam, S. Feiner and J. F. Hughes (Addison-Wesley, 1990))
- 由 Mason Woo, Jackie Neider 和 Tom Davis 编写的 **OpenGL 编程指南** (Addison-Wesley, 1997) (**OpenGL Programming Guide** by Mason Woo, Jackie Neider and Tom Davis (Addison-Wesley, 1997))
- 由 Josie Wemecke 编写的 **发明人指导** (**The Inventor Mentor**) (Addison-Wesley, 1994) (**The Inventor Mentor** by Josie Wemecke (Addison-Wesley, 1994))

对相同的图形对象如下给出两种编程的例子：第一个例子对应于对象的标准表示；第二个例子对应于合成的表示，将标准表示和具有低级图元

的表示结合起来。

标准表示

```

Transform { Children [
  Shape {
    geometry IndexedLineSet {
      point [ 0100, 000, 2000, -150.2150]
      colorindex [ 0 1 2 -1 # axes 34 ] # 中心线
      color Color { color [ 0 0 0, .2 .2 .2 ] }
      colorindex [ 0 1 ] # 轴为黑色，中心线为灰色
      colorPerVertex FALSE # 每条折线的颜色
    }
  }
  Shape {
    geometry IndexedLineSet {
      point [ 210, 520, 81.50, 1190, 1470, 17100]
      coordIndex [ 0 1 2 3 4 5 ] # 连接各点
      color Color { color [ .1 .1 .1, .2 .2 .2, .15 .15 .15, .9 .9 .9,
                           .7 .7 .7, 111 ] }
    }
  }
] }

```

合成的表示

```

Transform { Children [
  Shape {
    geometry IndexedLineSet {
      point [ 0100, 000, 2000, -150.2150]
      coordIndex [ 01 2 -1 # axes 34 ] # 中心线
      color Color { color [ 0 0 0, .2 .2 .2 ] }
      colorindex [ 01 ] # 轴为黑色，中心线为灰色
    }
  }
] }

```

```
        colorPerVertex FALSE # 每条折线的颜色
    }
}

LowGraphics {
    startTime      10.8 // 对象 1 将在 10.8 秒内显示
Source "http://www.myserver.com/LowGraphics"
    1}
}
```

很明显，在合成的表示中，程序从在下面地址的服务器中调用名为“*LowGraphics*”的对象的图元：<http://www.myserver.com/LowGraphics>

在合成的表示中，对象“*LowGraphics*”的属性被用来描述所述对象可被处理和组成的方式。

因此，上面给出的例子提出了利用属性“*startTime*”在特定时间后进行作用，以命令触发显示与为对象“*LowGraphics*”接收的数据相对应的图元。

上面例子尤其表明，对象“*LowGraphics*”将在图形景物的持续时间已经过 10.8 秒后被处理。

特别地，此信息使得接收机能够准备下载并且，在可应用处，准备对用于描述所述对象的信号（图 1 中的箭头 3 和 4 分别表示的对发送在 *LowGraphics* 对象中描述的图元的请求和那些图元的发送）进行解码。

可利用其它属性，尤其包括：

- “*endTime*” 属性，用于在给定时间停止对象的显示；
- “*active*” 属性，用于指定是否对象必须被显示或隐藏；
- “*transparency*” 属性，用于指定施加于对象的透明度系数，以使其相对于其它图形对象较透明或较不透明；
- 加载时间（TTL）属性，当“*LowGraphics*”图形对象的信号指定该对象的创建日期 DC，以及接收机在下载日期 DT 下载该对象时，如果该对象的创建和下载之间的已过去时间（DT-DC）大于给定时间 TTL，则指示将不会显示该对象；以及

- “clipping” 属性，用于提供对象将要呈现在其中的区域的尺寸（宽度，高度）。如果对象的尺寸大于所述区域，特别地可以避免显示位于所述区域外的任何部分。

现在参考图 3，其示出了分解图像的处理，其目的在于利用不同的服务器来存储构成图像的元素。

初始图像被分解为图形对象和子对象的空域-时域排列。

这些图形对象中有些可以以低级图元的形式表示，该图元例如{动作，多边形，持续时间}型的图元。

这些合成的对象（OC）被编码（步骤 E_{OC}），以在源 B 中被以再现图元 P 的形式存储。

景物的剩余部分，特别是其它图形对象，以及景物图形对象的一般空域-时域排列被以标准方式编码（步骤 E_S），并被存储在源 A 中。

下面说明图形图元的例子。

图形对象一般用多边形表示。

图形图元可以以点列表（多边形顶点）的形式描述多边形，在其中可联合颜色和纹理。

或者，图元可以仅根据三角形或梯形定义对象。

然后图元仅提供三角形或梯形的定义，并且在其中可联合颜色和纹理。

下面的程序是用于为具有 12 个面、每个面有五个顶点的十二面体进行编码的低级图元的一个例子。

```
#VRML V2.0 utf8
```

```
Viewpoint { description “Initial view” position 0 0 9 }
```

```
# 十二面体：20 个顶点，12 个面。
```

```
#6 种颜色（图元：RGB 和补色：CMY）被绘到面上。
```

```
Transform {
```

```
Translation -1.5 0 0
```

```
Children Shape {
```

```

appearance DEF A Appearance { material Material { } }
geometry DEF IFS
IndexedFaceSet {
coord Coordinate {
point [ # Coords/indices derived from “Jim Blinn’s Corner”
111, 11 -1, 1 -11, 1 -1 -1,
-111, -11 -1, -1 -11, -1 -1 -1,
.6181.6180, -.6181.6180, .618 -1 .618 0, -.618 -1.6180,
1.6180.618, 1.6180 -.618, -1.6180.618, -1.6180 -.618,
0.6181.618, 0 -.6181.618, 0.618 -1.618, 0 -0.618 -1.618

}
coordIndex [
1801213-1, 4951514-1, 21031312-1, 71161415-1, 21201617-1,
11331918-1, 41461716-1, 71551819-1, 416089-1, 21761110-1,
118598-1, 71931011-1,

color Color { # 六种颜色:
color [001, 010, 011, 100, 101, 110]

}
corlorPerVex FALSE #施加到面, 而非顶点 # 此标引给出了很好的对称外观:
colorindex [0, 1, 1, 0, 2, 3, 3, 2, 4, 5, 5, 4 ]

#五种纹理坐标, 用于每个面上的五个顶点。 # 通过适当地标引于其中,
这些纹理可重复利用。纹理坐标

```

```
TextureCoordinate {
```

```
    Point [# 这些是正五边形的坐标:
```

```
    0.6545080.0244717, 0.09549150.206107
```

```
    0.09549150.793893, 0.6545080.975528, 1 0.5,
```

```
# 并且这种特定的标引获得了很好的图像:
```

```
texCoordIndex [
```

```
01234-1, 23401-1, 40123-1, 12340-1,
```

```
23401-1, 01234-1, 12340-1, 40123-1,
```

```
40123-1, 12340-1, 01234-1, 23401-1
```

在 MPEG-4/BIFS(ISO/14496-1)中,这种十二面体的内容大小为 1050 字节。每个面可被分解为三个三角形,且每个三角形包括坐标为 (X, Y) 的三个顶点。

编译后,因此需要发送与三角形的顶点相对应的 $12*3*3*2$ 个整数(三角形的再现是 OpenGL 中的基本图元)。

对移动电话屏幕而言,像素 (X, Y) 可在 2 个字节上编码(最大屏幕尺寸 $255*255$)。

这就构成 $12*3*3*2 = 216$ 字节。

必须加上每个点的颜色分量(3 字节),其形成 $12*3*3*3 = 324$ 字节,也就是说,总共 540 字节。

因此,很显然,所提出的处理得到了对存储容量的显著节省。

可注意到,上述技术可非常普遍地实际应用于所有当前的图形动画描述: MPEG-4/BIFS, SVG, 等等。

应注意到,上面的说明涉及这样的情形,在其中动画数据(空域-时域排列内容,图元)被存储在被远程询问的服务器中。

当然,可利用其它的存储装置(例如 CD-ROM)。

同样地,服务器可利用“推”技术向客户端发送数据,来代替被询问

和采用“拉”技术。

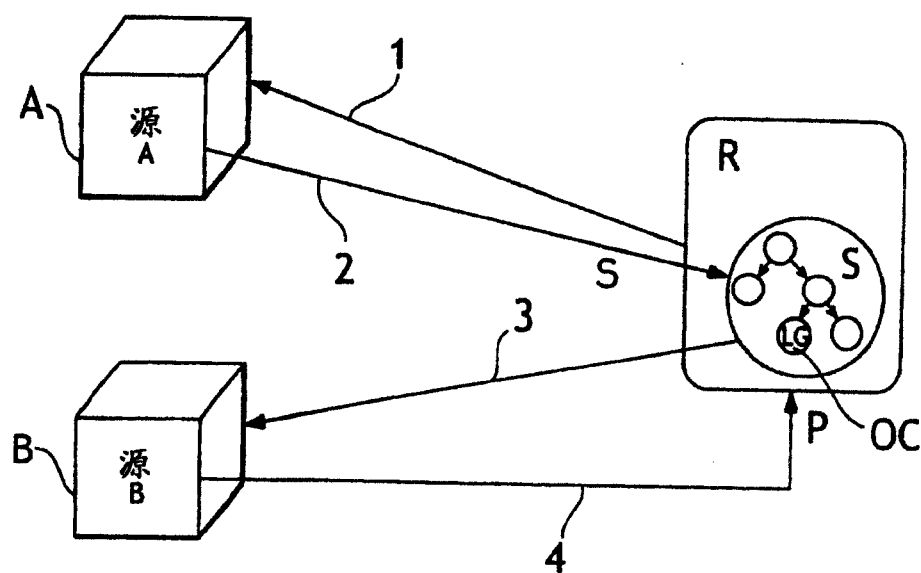


图 1

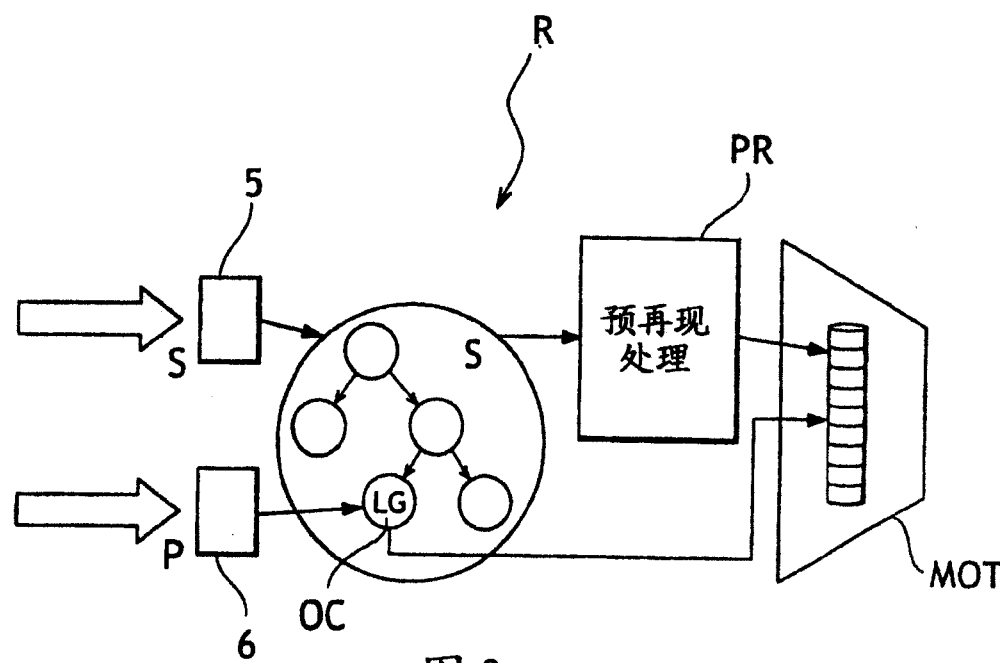


图 2

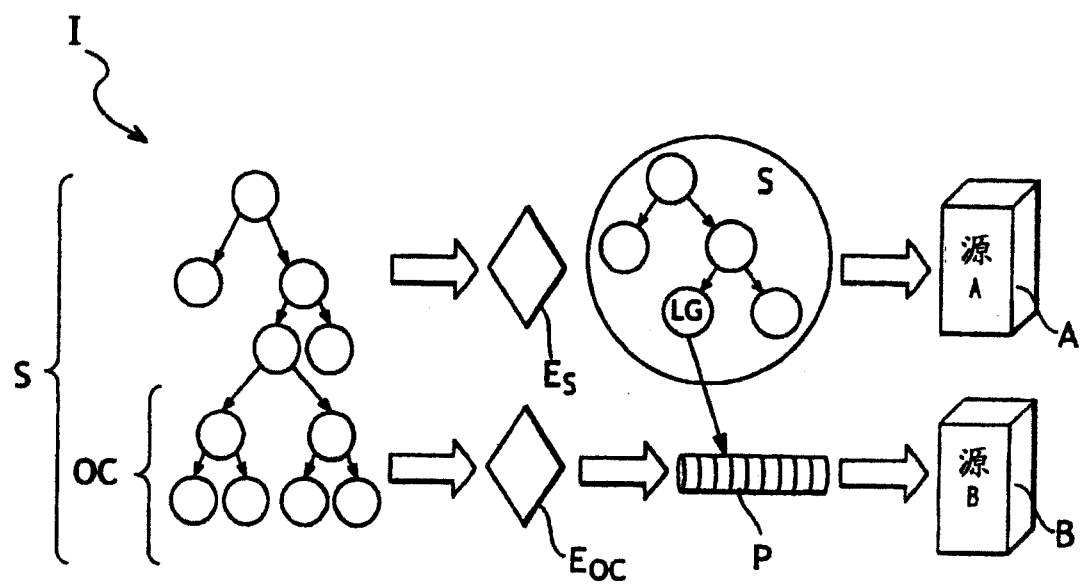


图 3