

文章编号: 1005-8451 (2018) 11-0011-05

# 基于旋转矩阵的ZPW-2000A轨道电路结构图 软件设计

王嗣策

(西南交通大学 信息科学与技术学院, 成都 611756)

**摘要:** 利用旋转矩阵原理对组成三维模型的点阵坐标进行处理, 得到其在不同旋转角度下的以显示屏为图纸的二维坐标。同时结合GDI+图形操作技术, 对列车和钢轨的三维模型及ZPW-2000A型轨道电路结构图进行绘制。采用面向对象的方式将这些图形以及组成它们的点, 线, 面, 封装成类。建立图形与数据的对应关系, 将电路图的内容编写成Excel表格。通过C#编程语言实现表格的读取与写入操作, 并根据表格提供的信息自动绘图。该软件能够显示列车在不同轨道区段上占用和出清组成的三维场景、轨道电路中相关继电器的状态及电流的路径。

**关键词:** 三维模型; GDI+图形操作技术; ZPW-2000A型轨道电路; C#编程语言

**中图分类号:** U284.2 : TP39 **文献标识码:** A

## Software design for structure diagram of ZPW-2000A track circuit based on rotation matrix

WANG Sice

(School of Information Science and Technology, Southwest Jiaotong University, Chengdu 611756, China)

**Abstract:** The principle of rotation matrix was used to process the lattice of the three-dimension model and get the two-dimension coordinates of various rotation angles in the screen. Besides, GDI + graphics operation technology was applied to draw the three-dimension models of the train and the track, the structure diagram of ZPW-2000A track circuit. The object-oriented method was adopted to encapsulate the figures including points, lines and planes into classes. The relationship between graphics and data was established, and the contents of the circuit diagram were compiled into Excel forms. The table reading and writing operation was implemented by C# programming language. Automatic drawing was implemented according to the information provided in the table. The software can show the three-dimension scene of the train occupied and cleared in different track segments, the state of the relays in the track circuit and the path of the current.

**Keywords:** three-dimension model; GDI+ graphic operation technology; ZPW-2000A track circuit; C# programming language

铁路运营过程中, 电路图纸是管理和维护铁路信号设备的重要依据。结合三维场景显示的电路图软件能够提供更多信息, 直观易懂, 成为提高铁路运输效率和职工专业素质的有效工具。常用的图形显示及建模方法有: (1) AutoCAD<sup>[1]</sup>, 具备基本三维设计功能, 并可以通过内嵌语言 Visual Lisp, VBA 和 ARX 等进行二次开发, 但是其开发主要目标是完成自动绘图, 难以实现如电路图的功能仿真等需求<sup>[1]</sup>。(2) LightWave 3D, Unigraphics NX 和 CATIA 等三维建模软件, 三维建模功能强大且各具特色, 但是

难以回避模型导入带来的格式问题。(3) OpenGL, DirectX 等底层图形接口, 具有开发复杂性大, 周期性长和维护困难的缺点<sup>[2]</sup>。

本文以 ZPW-2000A 轨道电路结构图为显示对象进行设计, 采用旋转矩阵原理和作为图形设备接口的 GDI+ 技术实现三维显示的效果。在一定程度上, 降低了开发的难度并增强了设计的灵活性。

## 1 基于旋转矩阵的三维模型设计

### 1.1 旋转矩阵原理简介

旋转矩阵 (Rotation Matrix) 是指与一个向量相乘时, 能够在改变向量方向的同时又不影响其大小

收稿日期: 2018-01-20

作者简介: 王嗣策, 在读硕士研究生。

效果的矩阵。旋转矩阵是一个行列式为1的正交矩阵,它的每个列向量都是单位向量且相互正交。其可被用于三维模型设计,运动学和离散数学的组合优化问题等领域中。

假设最开始空间的笛卡尔坐标系为  $XA, YA, ZA$ , 如此可列出三维空间 A 的矩阵  $VA=\{XA, YA, ZA\}T$ 。通过坐标轴的旋转,空间 A 坐标系的3个坐标轴变换到了图1中红色的3个坐标轴  $XB, YB, ZB$  的位置,得出空间 B 的矩阵  $VB=\{XB, YB, ZB\}T$ 。将两个空间联系起来可以推出  $VB=R \times VA$ , 这里,  $R$  即是旋转矩阵<sup>[3]</sup>。

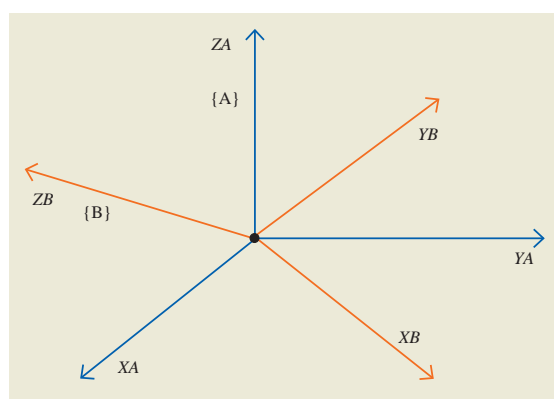


图1 坐标轴的旋转变换

## 1.2 旋转矩阵在图形显示的应用

本文通过点的组合实现三维和二维图形的显示,利用组成图形点阵的坐标在坐标轴的旋转方式,生成旋转矩阵并以此推导出旋转后的相应坐标,实现图形随鼠标拖动进行相应的旋转。旋转矩阵的推导方法如下。

将3个基本旋转的序列复合是求解旋转矩阵的方法之一。在右手笛卡尔坐标系中,三维旋转矩阵都可以用绕  $x$  轴顺时针旋转的角度  $\theta_x$ , 绕  $y$  轴顺时针旋转的角度  $\theta_y$  和绕  $z$  轴顺时针旋转的角度  $\theta_z$  来描述。因此,可以先分别单独计算绕  $x$  轴,  $y$  轴和  $z$  轴三者之一进行相应角度旋转的矩阵  $R_x(\theta_x)$ ,  $R_y(\theta_y)$  和  $R_z(\theta_z)$ 。而最终需要的旋转矩阵刚好可以表示为  $R_x(\theta_x)$ ,  $R_y(\theta_y)$  和  $R_z(\theta_z)$  3 个矩阵的乘积。这种旋转步骤的分解,致使这3个矩阵的生成元很容易理解,分别推导如下。

绕  $x$  轴旋转矩阵的推导:假设在一个三维场景中,点  $P(x, y, z)$  绕  $x$  轴顺时针旋转  $\theta_x$  度得到点  $P'(x', y', z')$ 。其可以被描述成  $x$  坐标保持不变,在  $y$  轴和  $z$

轴组成的  $yoz$  ( $o$  为坐标原点) 平面上进行了一次旋转角为  $\theta_x$  的二维旋转,由此可得:  $x'=x$ ,  $y'=y\cos\theta_x - z\sin\theta_x$ ,  $z'=y\sin\theta_x + z\cos\theta_x$ , 用  $(3 \times 3)$  矩阵表示如下:

$$R_x(\theta_x) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_x & -\sin\theta_x \\ 0 & \sin\theta_x & \cos\theta_x \end{bmatrix} = \exp \left( \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -\theta_x \\ 0 & \theta_x & 0 \end{bmatrix} \right) \quad (1)$$

绕  $y$  轴旋转矩阵的推导:其可以被描述为  $y$  坐标保持不变,在  $z$  轴和  $x$  轴组成的平面  $zox$  上进行了一次旋转角为  $\theta_y$  的二维旋转,  $x'=z\sin\theta_y + x\cos\theta_y$ ,  $y'=y$ ,  $z'=z\cos\theta_y - x\sin\theta_y$ , 用  $(3 \times 3)$  矩阵表示如下:

$$R_y(\theta_y) = \begin{bmatrix} \cos\theta_y & 0 & \sin\theta_y \\ 0 & 1 & 0 \\ -\sin\theta_y & 0 & \cos\theta_y \end{bmatrix} = \exp \left( \begin{bmatrix} 0 & 0 & \theta_y \\ 0 & 0 & 0 \\ -\theta_y & 0 & 0 \end{bmatrix} \right) \quad (2)$$

绕  $z$  轴旋转矩阵的推导:其可以被描述为  $z$  坐标保持不变,在  $x$  轴和  $y$  轴组成的平面  $xoy$  上进行了一次旋转角为  $\theta_z$  的二维旋转,  $x'=x\cos\theta_z - y\sin\theta_z$ ,  $y'=x\sin\theta_z + y\cos\theta_z$ ,  $z'=z$ , 用  $(3 \times 3)$  矩阵表示如下:

$$R_z(\theta_z) = \begin{bmatrix} \cos\theta_z & -\sin\theta_z & 0 \\ \sin\theta_z & \cos\theta_z & 0 \\ 0 & 0 & 1 \end{bmatrix} = \exp \left( \begin{bmatrix} 0 & -\theta_z & 0 \\ \theta_z & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \right) \quad (3)$$

本文使用鼠标水平与竖直拖动窗体焦点的方式移动图纸视角,为方便设计只需要绕  $z$  轴和绕  $y$  轴顺时针方向旋转的矩阵相乘得到旋转矩阵,即  $R=R_y(\theta_y) \times R_z(\theta_z)$ <sup>[4]</sup>。

## 1.3 三维模型的生成与优化

采用旋转矩阵的原理可以得到任意一点经过坐标轴旋转后的坐标,进而可以较为方便地使用二维的绘图 API 绘制三维图形的轮廓。但是从实际的效果上看,经过 GDI+ 绘制输出在屏幕上的仅是能随鼠标拖动而旋转的框架图,无法展现空间上的物体之间的遮挡关系,与实物相差较大且可能对用户造成一定干扰,图2和图3为软件设计过程中,对三维图形遮挡关系的处理效果进行试验的结果对比。

由此可知,需要进行三维图形遮挡关系的处理,方法是采用 C# 的 fill 函数对处于用户视角中的几何体表面的轮廓进行填充。

建立如图4中所绘坐标系,在设定用户视线与屏幕垂直的基础上,以迎着视线的方向作为  $y$  轴的初始方向,与地面垂直的方向为  $z$  轴初始方向,并以此

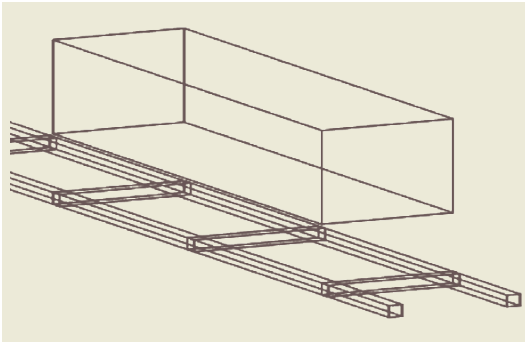


图2 未进行遮挡处理的效果图

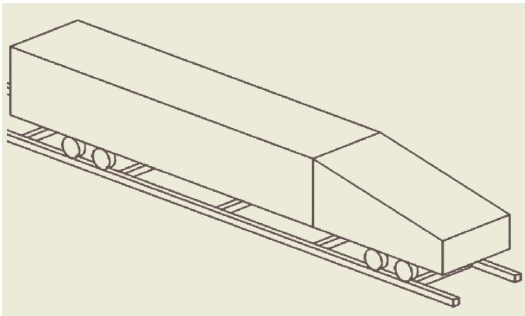


图3 经过遮挡处理的效果图

对几何体的表面进行编号。采用判断语句对绕  $z$  轴逆时针旋转角度  $\beta$  和绕  $y$  轴的逆时针旋转角度  $\alpha$  的多种组合进行判断，结合三维图形确定当下角度最多显示的面（长方体是 3 个面）并对其进行填充，以实现空间遮挡效果，如图 5 所示。

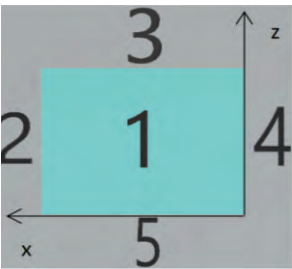


图4 初始位置下的视图显示

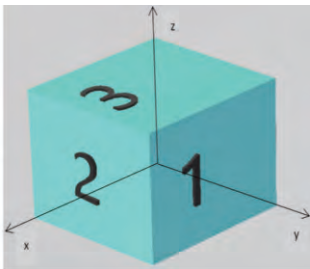


图5 经过旋转后的视图显示

以图 4 中的长方体为例，其初始位置为同一顶点的 3 条棱与初始坐标系重合的情况，按照其分别绕着  $y$  轴和  $z$  轴的从  $0^\circ$  到  $360^\circ$  逐步旋转，其出现在用户视角中的 3 个面的编号如表 1 所示，其中，当  $\alpha$  或  $\beta$  为  $90^\circ$  的倍数时，视角中只能看到长方体的 2 个或 1 个面，可视表 1 中多列出的平面面积为 0，并不影响显示效果。

2 图形信息处理

2.1 图形对象存储与调用

表1 旋转角度与视图中显示的平面关系

$\alpha$ $\beta$	$\beta \in (0, 90^\circ)$	$\beta \in (90^\circ, 180^\circ)$	$\beta \in (180^\circ, 270^\circ)$	$\beta \in (270^\circ, 360^\circ)$
$\alpha \in (0, 90^\circ)$	(1, 2, 3)	(6, 2, 3)	(4, 6, 3)	(1, 4, 3)
$\alpha \in (90^\circ, 180^\circ)$	(1, 3, 4)	(6, 3, 4)	(6, 5, 4)	(1, 5, 4)
$\alpha \in (180^\circ, 270^\circ)$	(1, 4, 5)	(6, 4, 5)	(2, 6, 5)	(1, 2, 5)
$\alpha \in (270^\circ, 360^\circ)$	(1, 5, 2)	(6, 5, 2)	(3, 6, 2)	(1, 3, 2)

采用面向对象的方式将图纸中常用的图形的点，线，面，体的组合封装成类。点和线，线和面，面和体的关系类均是前者组合成后者的关系。最后通过不同几何体的组合形成列车和钢轨的三维模型。

由于软件采用图形类之间的这种组合关系，只需要对处于顶层的图形类的对象进行初始化，即可生成组成该几何体的各部分类的对象，如调用构造函数初始化钢轨的信息，则将依次生成构成这段钢轨的数条横轨和两条竖轨，以及每个长方体对应的 6 个面，12 条棱，8 个顶点。

对上述类的对象分别进行存储是由 C# 语言的 List<T> 类来完成的。List<T> 类表示可通过索引访问的对象的强类型列表。其中，T 为类型参数，代表列表中元素的类型。List<T> 类提供用于对列表进行搜索、排序和操作的方法，其大小可按需动态增加，为使用 C# 语言编写面向对象程序增加了极大的效力和灵活性<sup>[5]</sup>。

2.2 表格编写与导入

为了便于图形的修改以及其它设备图纸的绘制，采用 Excel 表格存储图形数据。采用 C# 窗体应用设计，通过其事件处理的机制，写入浏览文件和表格操作的相关方法，实现在 Windows 窗体中点击按钮导入表格的功能。

选中表格文件后，程序将读取的表格中每一行的类型码，以此确定需要初始化的类的对象，使用表格中相应行的数据为其赋初值，并添加到相应的 List<T> 类列表中。

经过读取后的表格显示如图 6 所示，表格（除表头外）的每一行都与电路图中某一元素的信息一一对应；每行设置 13 列分别记录此元素的不同属性，包括了图形设备的名称，在图纸中的三维坐标，图形的长宽高信息以及绘制导线用到的另一端点的三维坐标，图形的状态信息，还有用于程序进行识别



ZPW2000A型轨道电路结构图软件

导入表格数据	no	名称	x	y	z	length	width	height	xl	yl	zl	state	type
	1	钢轨	-600	0	80	1350	96	8	0	0	0	0	100100000001
	2	列车	-150	0	112	600	96	8	0	0	0	0	101000000001
自动绘图	3	轨道电路	-600	0	0	1350	96	0	0	0	0	0	101100000001
	4	ZPW·PT	-600	230	0	95	80	0	0	0	0	0	110000000001
	5	ZPW·XKD	-500	230	0	95	80	0	0	0	0	0	110000000010
导出表格数据	6	ZPW·PT	-400	230	0	95	80	0	0	0	0	0	110000000011
	7	电缆	-400	330	0	95	80	0	0	0	0	0	110000000100
	8	站防雷	-400	430	0	95	80	0	0	0	0	0	110000000101

图6 导入表格后的界面显示

绘图的类型码信息。

当数据从表格转入程序内存后,将利用程序中的相关函数调用点列表中的信息进行坐标变换的计算,得到新的点的坐标,经过绘图事件的触发即可在屏幕中输出相应图形。

### 3 电路图软件实现

本设计通过编写程序来实现 ZPW-2000A 型轨道电路结构图的平面显示,并可以操作列车和钢轨的三维模型组成不同的列车运行场景,对不同场景下轨道电路的动作进行显示。

#### 3.1 ZPW-2000A型轨道电路结构图组成

软件中视角旋转到俯视角度的 ZPW-2000A 型轨道电路结构如图 7 所示。ZPW-2000A 型轨道电路由室内与室外设施组成。其室内部分包括站防雷及电缆模拟网络,衰减器,发送器及接收器。电室外设施有匹配变压器,补偿电容,传输电缆与调谐区以及调谐区设备引接线<sup>[6]</sup>。

ZPW-2000A 型轨道电路被分成主轨道电路与短小轨道电路。小轨道电路被看作列车行驶前方主轨道电路的延长区段。小轨道的信号经过行驶前方邻近轨道电路接收器处理,再将处理后的信号变成小轨道电路轨道继电器动作条件送入本轨道电路接收器,成为轨道继电器吸起的必要条件中的一部分<sup>[7]</sup>。在图 7 中,发送器产生出高频信号,其中的一些

经由主轨道返回该轨道接收端,而另外的一些通过小轨道电路到达相邻轨道接收端,经过相邻轨道接收器处理变为执行命令送入主轨道接收器<sup>[8]</sup>。当明确主轨道移频信号与小轨道继电器执行条件正确的情况下,使得轨道继电器(GJ)吸起。以此来判断区段的空闲和占用情况。如此可完成轨道电路的全程断轨检测。

#### 3.2 动态显示设计

软件具有电路结构图随鼠标拖动进行变换视图的功能,实现步骤:(1)通过在 C# 窗体程序中设置一系列鼠标操作事件,获取开始点击鼠标与停止拖动鼠标时的横纵坐标。(2)将获取到的鼠标移动数据进行处理,转换成将图纸分别绕 y 轴与 z 轴旋转的弧度值。(3)根据旋转的弧度值生成旋转矩阵,代入各图形的点位信息解出变换后的坐标值。(4)对得到的坐标进行平面覆盖的相关处理,并通过 Refresh() 函数对窗体的图像显示控件 PictureBox 进行刷新变换视图。

软件还具有模拟钢轨的占用与出清状态时电流回路的相应变化,发送报警继电器(FBJ)的吸起与落下的功能,实现步骤:(1)在 Windows 窗体中加入了 ToolStrip 控件建立下拉菜单。(2)将表格中的数据逐行添加到 ToolStrip 控件可供显示的下拉文本项中,为用户提供图纸数据。(3)设置 ToolStrip 控件的索引改变事件,可通过选中某一行的信息更改

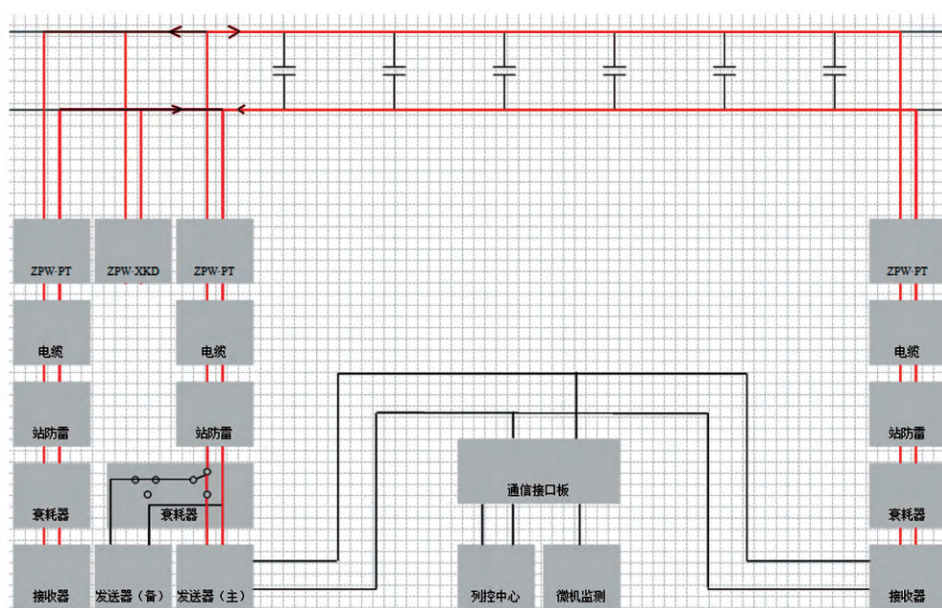


图7 空闲状态的轨道电路结构图

相应图形的状态值。(4)刷新 PictureBox 控件,显示经过状态变换后的图形。

为了便于显示,在编制数据时,将用于模拟列车和钢轨的模型与表示轨道电路及其结构框图部分设置为不同的高度,使其分别处于上下两个平行的平面中,这样在将图纸旋转到图示位置时,可以清晰直观地看到列车在钢轨中占用位置的同时,也可以了解到轨道电路信号的流向。如图8所示在列车第一轮对压入主轨道电路调谐区前方的位置,由发送器发出的信号读取到电流信号一栏的状态为1,显示为红色电流线,经过发送器(主),站防雷及电缆模拟网络,在主轨道电路端经过列车轮形成闭合回路。另一部分则流入调谐区小轨道以实现全程无断轨检查。

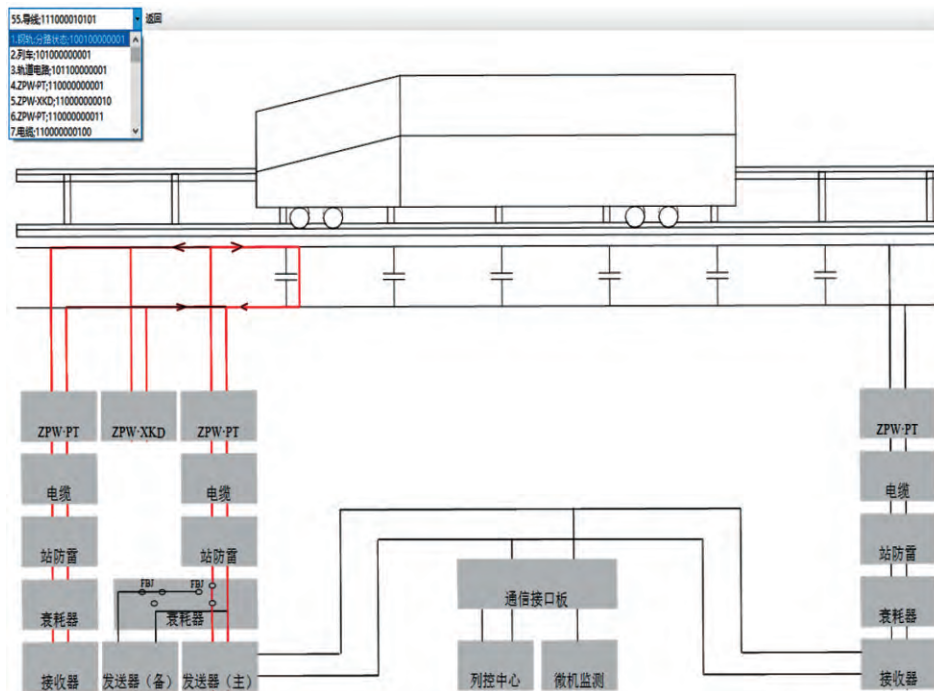


图8 列车占用场景下轨道电路结构图

通过点击选中下拉列表中发送器所在行,实现其状态的改变,并刷新图纸内容,电路走向随之发生相应的变化,如图9所示即为主发送器故障,通过衰耗器内部的发送报警继电器的状态改变,断开与主发送器的连接,接通备用的发送器,达到了故障导向安全的目的。

#### 4 结束语

在 Window 环境下,利用 VS2012 编程开发工具,

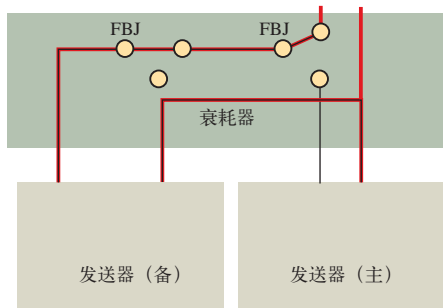


图9 发送器主备系切换

根据图纸内容编制了存储电路图信息的表格,实现了结合三维场景下轨道电路结构图的显示。软件能够显示在轨道占用和出清状态下电流的走向,易于使用者理解和学习。此外,以上建模过程可应用于其它三维场景的设计,并可通过修改表格绘制铁路信号设备的其它图纸,具有良好的实用性。

#### 参考文献:

- [1] 陈小艳,王强,李柏林.利用VTK实现几何模型的三维建模[J].铁路计算机应用,2015,24(7):7-11.
- [2] 刘大园,姚力,巫江.基于AutoCAD.NET的地铁铺轨综合设计图软件开发[J].铁路计算机应用,2013,22(11):36-39.
- [3] 梁艺宝.空间图形旋转变换矩阵推导新导刊[J].中国科教创新导刊,2013(22):34-35.
- [4] 闫志刚.图形学中绕任意轴旋转矩阵的推广[J].计算机时代,2009(12):65-66.
- [5] 孙志辉.C#程序设计[M].北京:人民邮电出版社,2015.
- [6] 魏红星.浅谈ZPW-2000A型无绝缘移频自动闭塞系统[J].科技信息,2012,36(31):195-196.
- [7] 刘利芳.区间信号自动控制[M].北京:科学出版社,2014:89-113.
- [8] 郭进,魏艳,刘利芳.铁路信号基础设备[M].成都:西南交通大学出版社,2008.

责任编辑 陈蓉