**线性代数在软件工程学科中的应用方向举例**

摘要：作为一门基础学科，线性代数在工作生活中扮演着无比重要的角色。本文旨在从本专业（软件工程）视角对该学科的大致应用方向做出简单介绍。笔者尽量通过实例化的方式从数字编码，游戏开发，人工智能这3个常见的应用层面来具现对该学科的认识，并不追求全面的概括。

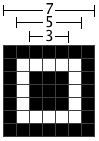
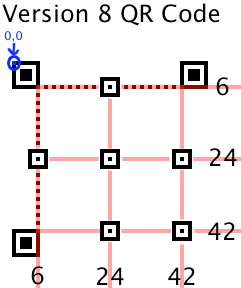
关键词：线性代数，软件工程开发，基础学科应用，数据处理

一．数字编码 —— 基于字符集的信息具现

1.数据编码是计算机处理的关键。如何将现实世界中的信息反映为电路中表示0和1的脉冲信号，是所有本专业问题的关键。该方向也常常涉及一些直接与计算机交互的高级编程，比如数字电路，编译器等。而其中，涉及了大量的相似工作。但问题是，计算机只能处理重复的工作。有人说，线性代数的真正意义在于提供了一种看待世界的视角：万事万物都可以被抽象为某些特征的组合，并在由预制规则定义的框架之下以静态和动态的方式加以观察。而这将大大简化软件开发人员的工作。为了具现这一理论，这里我们简单谈谈二维码。

2.课题简述：基于字符封装的二维码实现原理

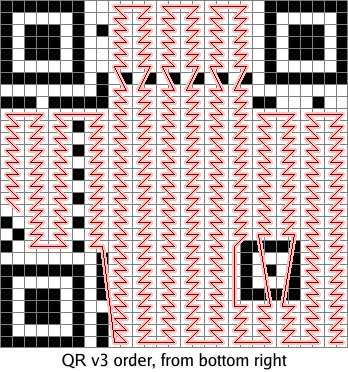
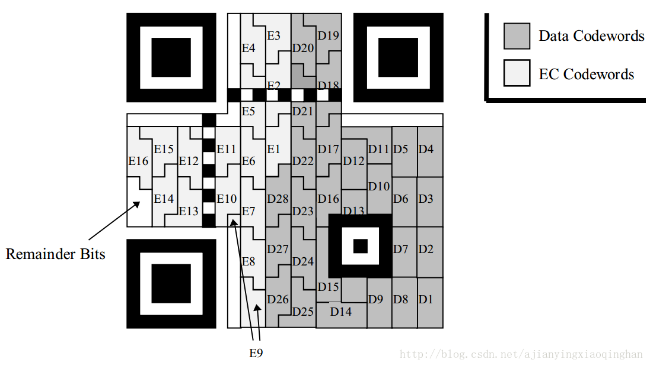
二维码（以下简写作QR）的尺寸总共有40个版本（version），每一个都是一个由0与1构成的矩阵，当然，1代表黑色，0代表白色。计算公式是Size=(Version−1)×4 +21。即最小21\*21，最大为177\*177（BITS/点）向矩阵中填充数据以及进行加密解密的设计，是专业人员所做的主要工作。一个QR的公共组成部分包括三个定位符（三点确定一个矩形）和对齐图案（ver1没有，ver2-6有1个，以上更多），他们的位置有特殊的填充规定。当然，还有时序图案和格式信息，他们一般在定位符的周围分布。

对齐图案 一个5\*5的矩阵 对齐图案安放位置的规定

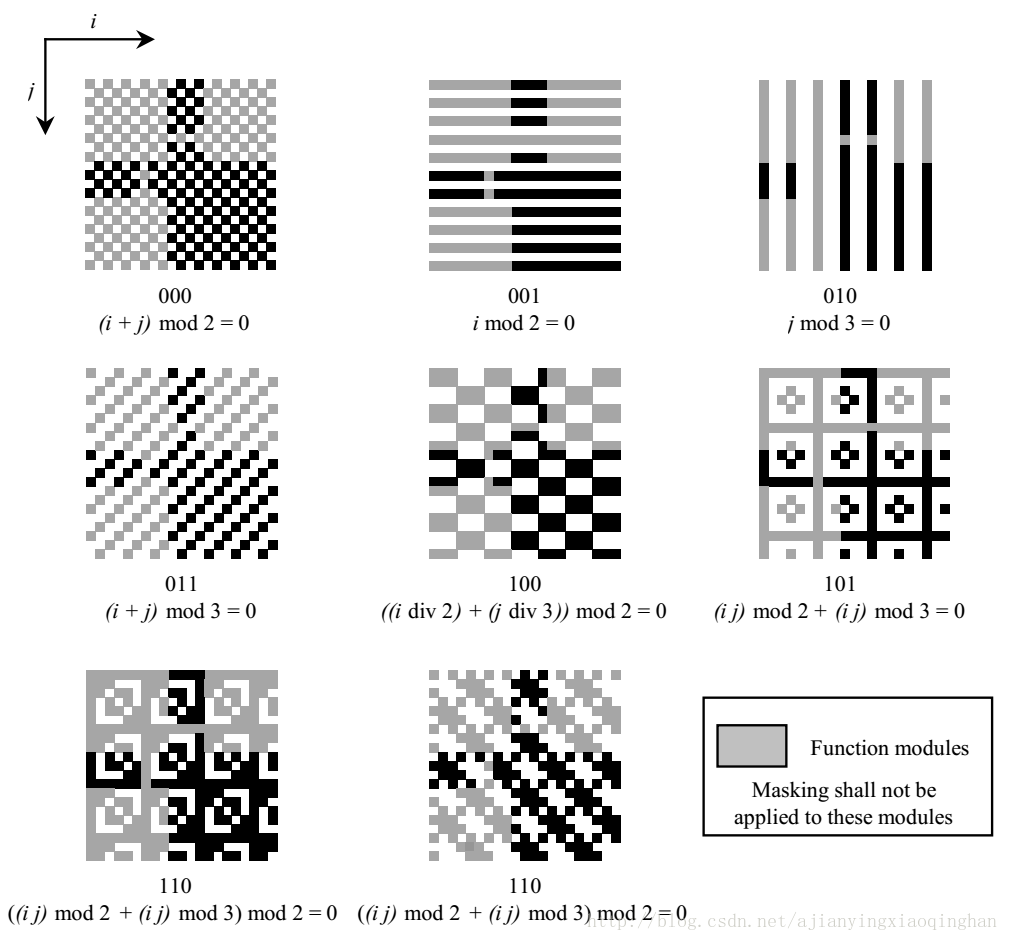
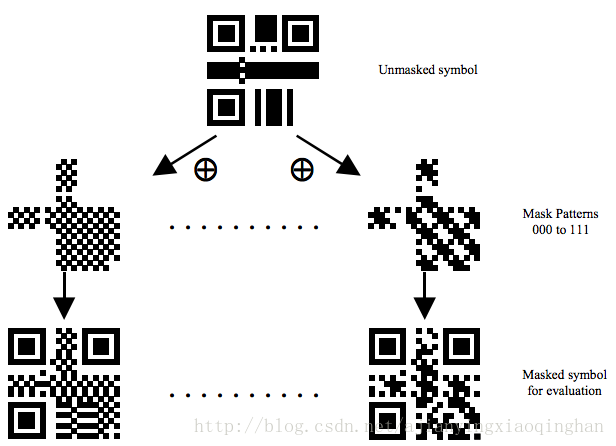
定位符，一个7\*7的矩阵

总而言之，将这些规定的图案填充完毕后，就可以考虑我们要填入的文字信息了（想了解正规的实现规则，请参考https://www.iso.org/standard/62021.html（ISO 18004软件工程标准文件））。对于非数字的文字，我们根据ASCII字符集乃至于Unicode字符集（有中日文）把他们翻译成二进制数。比如在ASCII字符集中，a代表97（转化为二进制为1100001）。

红色位置为数据的填写区域和一个模拟的填写顺序 实际的填数会更加复杂，这也用来进行加密

加密和解密的过程，涉及到基于矩阵的各种运算。我们此处仅以掩码为例做一个演示。实际的二维码在填充完毕后，总会出现大量黑色区域，为此，我们便需要对黑白两色的比例分布做出调整。ISO标准规定了八种不同的蒙版图案。其中，x,y是单个像素的坐标，如果条件表达式（下页左图下方有列出）结果为0，就把当前模块的颜色改为相反颜色，白变黑，黑变白；如果结果不为0，就不改变。我们通过这种简单的条件矩阵加法实现美观。注意到，蒙版图案并不包含定位符和对齐图案等标准区域。

八种常见的蒙版图案和他们的变色条件 施加影响后的变色过程

鉴于这并不是一个学术性的文章，我们不在这里做更多关于实现细节的讨论。

如果您想要参考更多关于这种项目的信息，可以在GITHUB网站上进行检索或者参考以下一个大学生写的不成熟的项目文档：<https://wenku.baidu.com/view/a7135d7db80d6c85ec3a87c24028915f804d841f.html>。

二 . 游戏开发 —— 基于坐标系统的需求实现

1.线性代数贯穿了每一个游戏开发者的一生。大部分坐标系统的代码基础都是矩阵。一些高级游戏开发应用（比如较火的Unity 3D）给程序员们提供了简单可视的游戏开发平台，但其底层依旧是线性映射和向量变换。通过将张量及运算藏在一个特定部分中（封装到类里），不但使玩家，也使得大部分的像我们这样的底层程序员意识不到它的存在（笑）。

1.Using UnityEngine;//使用了Unity3D的一个内置的包（我们称为名称空间）

2.Using System.Collections;//使用“系统”这个对象内的“收藏”空间

3．Public class example2.1.1: {//新建一个类

4. Void gulugulu (){//类里面有一个成员函数（我们称之为“方法”）gulugulu

5. transform.Rotate(0.5 \* Time.deltaTime,0);// 调用使得物体在一个匀速的速度下旋转，不依赖帧的速率

6. }

7.}

请看上述加黑的代码，红字部分的注释表示的旋转工作，就是在规定的增量时间段（Time.deltaTime）内不断将对象的坐标参数乘以子坐标矩阵,根据课本4,8节的内容我们容易知道，这个矩阵表示的就是令物体绕自己的坐标Z轴（我们称为“子坐标”，就是物体以自身重心为原点的坐标，我们会在接下来详细探讨这个问题）旋转a度角。看起来很简单。然而在实现过程中，这并不容易。

这里，我们尝试使用一个简单的二维世界坐标和局部坐标关系的示例来再现开发软件的开发人员（游戏狗的上帝的上帝）的简单思维。

2.项目实践：基于C语言的海龟图的实现

矩阵是二阶的张量，它有两个索引（坐标）。在计算机语言中，它具现为一个二维数组。除了这些数本身携带的信息之外，还可以在每个数的对应位置上携带信息(矩阵亦是如此)。

基于上帝视角的程式是易于编写的。我们假设一个上北下南左西右东的二维世界坐标系。让一个角色向北走再向西走，可以简单地通过两个命令来实现：纵坐标加一个数 横坐标再减一个数。但是如果要让一个角色向左走再向上走，事情就没那么容易。首先要判断这个角色一开始是朝向哪里，如果朝向北方时向左走，那么就是横坐标减。其次要记录现在这个角色朝向哪里，来为下一次的移动做准备。以这个角色为原点的这个二维坐标系（当然现在可以将轴的名字理解为上下左右而不是正负X和正负Y）称为子坐标系。在这个实例中，子坐标系的选取与上一个子坐标系关联，那么上一个坐标系便被称作父坐标系。

坐标系的模型基础是基于拓扑空间，在浅显的应用中，可以理解为向量空间。我们利用数组（矩阵）存储信息的特性，来记录蓝字所表示的以及移动所需的（大部分情况下，父坐标系与子坐标系之间的关系由于是三维的，所以是很抽象的）状态量。在文档所附的源代码中，我们使用了一个二维数组Dstate来表示二维子坐标与世界坐标系之间的关系。Dstate[0][1]存储-1与+1，代表水平量的加减；而Dstate[0][0]存储-1与+1，代表垂直量。他们根据子坐标进行改变。他可以轻松地拓展到三维。关于更具体的介绍，请参考https://blog.csdn.net/biezhihua/article/details/77489399。源代码和一个极其丑陋（因为不是为了玩游戏（认真脸））的可运行的文件已经随文档拷贝。

三．人工智能 —— 基于数学建模的机器学习

1. 先说点题外话。个人对于人工智能一向不是特别感兴趣。究其原因，人工智能的真正的目标是在极其复杂和交互的环境下做出最优的决策。而答案不唯一将成为极其困扰人的事情（尤其对于我来说），这种明确了结果没有最好，只有更好的，又无法脱离暴力实验的长远计划，实在让人提不起兴趣。再回归正题看人工智能应用的本质，就是一堆约束。我们知道，线性规划可以使用矩阵来求解最优解。更加复杂的约束，统称为数学建模。一堆由样本拟合生成的模型所谓“有机”的凑在一起，便形成了模拟人类的假神经网络。在这之中，线性代数提供的“特征性”在“约束”中起了无比重要的作用。这里为了具现一个实现最优解的过程，笔者用Java代码展示了部分最简单的线性回归分析的代码实现。通过不断的输入样本数据(为了大幅简化代码，我们使用了初始化)模拟“训练”的过程，最后得到了最优的结果。其中，线性代数等模型工具负责告诉机器如何将相似化数据转变为重复性工作。实际的问题涉及的数据方面（专业上称为对象及其属性）则比这多得多。可以看到，人工智能的主要成果并不在于你的算法有多么精妙，而在于你的样本数据是否足够多和计算能力是不是够强。这着实让人感到前路多艰。

2.工程分析：基于Java平台的线性回归（训练）

(/\*。。。\*/其中的内容表示注释”)

  1./\*决策公式：最小二乘法 线性回归 y = a\*x + b

  2.b = sum( y ) / n - a \* sum( x ) / n

  3.a = ( n \* sum( xy ) - sum( x ) \* sum( y ) ) / ( n \* sum( x^2 ) - sum(x) ^

4. 2 )  \*/

5. public class LinearRegression {

    6. public static void main(String[] args) {

     7.  int n = 0;

      8.  double[] x = { 1, 2, 3 };

       9.double[] y = { 3, 6, 9 };

        10.double sumx = 0.0, sumy = 0.0, sumx2 = 0.0;

        11.while (n < x.length) {

         12. sumx += x[n];

         13.  sumx2 += x[n] \* x[n];

          14.  sumy += y[n];

          15.  n++;}

         16.double xbar = sumx / n;

         17.double ybar = sumy / n;

         18.double xxbar = 0.0, yybar = 0.0, xybar = 0.0;

         19.for (int i = 0; i < n; i++) {

           20. xxbar += (x[i] - xbar) \* (x[i] - xbar);

           21.yybar += (y[i] - ybar) \* (y[i] - ybar);

           22. xybar += (x[i] - xbar) \* (y[i] - ybar);

        }

         23.double beta1 = xybar / xxbar;

         24.double beta0 = ybar - beta1 \* xbar;

         25.System.out.println("y   = " + beta1 + " \* x + " + beta0);

     26.}}

程序中，8-9行初始化 11-17行计算平均数18-24行根据平均数计算回归直线，第25行打印结果。给定决策的原则，程序便会根据数据生成相应的数学模型。这是人工智能的最优化问题的本质。

总结：线性代数初学总给人一种莫名其妙的感觉。但实际来看用处很多。尽管它是软件工程底层的架构，但其实在前端工作的人员可能真的用不大到多么复杂的知识。但作为一个志存高远的人，既然意识到它在底层设计中的重要性，就更要深入透彻的学习它。这样才能让研究更加深入下去。