Week 4 Lab Prep

1. SVN tutorial

Introduction:

Version control permits you to manage several versions of source code or other files. It is necessary for any software project of useful size. There are many options for Version Control: SVN, CVS, RCS and so on. Today we'll briefly cover some basic stuff about SVN.

How SVN works

SVN works by keeping a "repository" of revision information for a project. The repository is a set of directories and files that documents the entire version information for project. A repository may contain several projects. A programmer "checks out" copies of the files from the repository into a separate working directory. When a programmer modifies these copies and saves a new version, they can then "check in" the modified copies to the repository. The repository then creates new versions of the files and assigns the new version numbers to these copies.

Difference between Repository and Working Directory

The repository is usually saved on the server while we check out the files from the repository into working directory on our local machine. Therefore, we can always have multiple working directories on different machines or even on the same machine of the same or different versions. However, the repository for one project is unique. Moreover, we can delete the working directory without worrying about losing any information since everything is stored in the repository.

Basic commands:

Create a SVN repository

a) ssh to a Linux machine in our department
b) use the following commands to create a SVN repository:
   svnadmin create –fs-type fsfs [repository name]

Simple commands in Linux

a) Import a new module
   svn import [module name] <url> --message "any message"
b) Checkout a module
   svn checkout <url> [directory]
c) Commit changes to the repository
   To commit changes to the repository, just enter your working directory and type:
   svn commit
d) Update the files on your local machine from the database
   svn update
e) Check the status of files in the working directory
   svn status

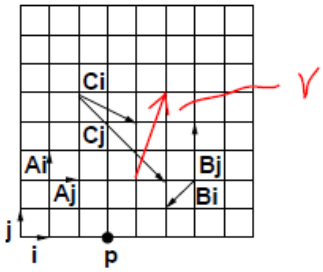
Other Useful SVN Tools:

a) TortoiseSVN: an open source SVN client.
(can be downloaded from http://tortoisesvn.sourceforge.net/)

b) Subversive: SVN plug-in for Eclipse
(installation instructions see:
http://www.eclipse.org/subversive/documentation/gettingStarted/aboutSubversive/install.php)

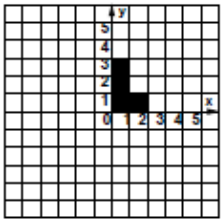
2. Sample Questions for Homework

[q1] (Source: Tamara 314 Homework 1 v2008 modified) The point coordinate P can be expressed as P=3*i+0*j, and the vector *v* can be expressed as *v* = 1*i+3*j, where I and j are basis vectors of unit length along the x and y axes, respectively. Describe the point P in terms of the coordinate system C. Describe the vector *v* in terms of the coordinate system B.



Answer: In coordinate system C: P = (2,-1)
In coordinate system B: *v*=(-1,1)

[q2] (Source: Tamara 314 Homework 1 v2008) For each equation below, sketch the new location L' of the L shape on the grid and provide the OpenGL sequence needed to carry out those operations. Use the function *drawL()*, which draws an L shape with the lower left corner at the current origin as shown below. You may assume the matrix mode is GL_MODELVIERW and that the stack has been initialized with *glLoadIdentity()*. For reference, the OpenGL command syntax is *glRotatef(angle, x, y, z)*, *glTranslatef(x, y, z)*, *glScalef(x, y, z)*.




```
drawL();
```

$$A = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, B = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, C = \begin{bmatrix} 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, D = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

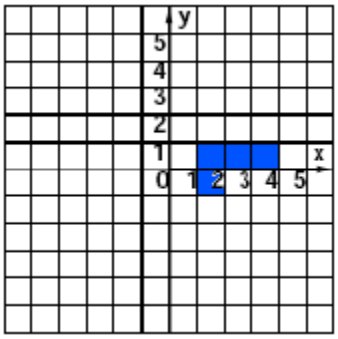
a) L' = ACDL

b) L' = CBADL

Answer:

a)

```
glTranslatef(1.0f, 0.0f, 0.0f);
glRotatef(270.0f, 0.0f, 0.0f, 1.0f);
glTranslatef(-1.0f, 0.0f, 0.0f);
drawL();
```



b)

```
glTranslatef(-1.0f, 0.0f, 0.0f);
glRotatef(270.0f, 0.0f, 0.0f, 1.0f);
glTranslatef(1.0f, 0.0f, 0.0f);
glScalef(2.0f, 1.0f, 1.0f);
drawL();
```