

**Assignment 3, Due Thurs 22nd November 2007, 10.30am. Worth 33% of CA. Demonstrations, see below, must be completed by Tues 20th November 2007, 5.00pm.**

**Answers should be in your own words. For numerical answers, where appropriate, show your working; a numerical answer written down with no explanation of working is likely to get zero marks.**

Late assignments. 10% of marks will be deducted for each day late; assignments seven days late will receive 0%.

**New Computing Department guidelines:** 1. Assignments must be neatly typed and printed; 2. A cover sheet and declaration must be included. A copy of the guidelines and cover sheet are available on my public drive. Please read and consider carefully the section on plagiarism.

The assignment refers to a program in Angel, Interactive Computer Graphics, Addison-Wesley, 2005, Chapter 10, which represents a robot in a *hierarchical* manner using a tree structure. A copy (cleaned up by me) is in `progs\fig2.c`. In particular, note that I have made material (reflectance) arrays global.

It is important that you realise that a game would store moving objects in such a hierarchical structure. The structure in `fig2.c` is a bit crude, but it demonstrates the principle well. If you do a search on `<scene graph>`, you will find out how a game engine represents complex moving objects.

Ideally, I'd make a proper object-oriented *scene graph* based on Angel's code, but I do not have time this year.

Regarding GLSL shaders, I've decided that this is a better learning exercise than an assignment on shaders. I'll set an unassessed practical that will teach you about shaders.

1. (a) Following the model of the existing code for lower and upper arms, and lower and upper legs, add a finger to each of the lower arms).

[20 + 20 = 40 marks]

- (b) Following the model of the existing code, add a menu item and code which will allow the fingers to be rotated. Array `angle` now has 12 slots — add another.

[20 + 20 = 40 marks]

- (c) Write an explanation why *child* nodes move appropriately when their *parent* nodes do.

[20 marks]

Get task 1. done before you attempt task 2.

2. (a) Add fields to the node structure to allow it to contain ambient, diffuse, and specular reflectances (`GLfloat [4]`), and a shininess variable; i.e. potentially, every node can have a separate set of material properties (colour). It may simplify things to use `glColorMaterial` (see notes, chapter 6.7, page 6 -15).

```
typedef struct treenode{
    GLfloat m[16];
    /* add here */
    void (*f)(void);
    struct treenode *sibling;
    struct treenode *child;
}treenode;
```

(b) Then, in `traverse`, activate these just before `root->f()` is invoked.

```
/*
  glMaterialfv(GL_FRONT, GL_SPECULAR, root ??? etc.
*/
```

(c) Then, when you are constructing the tree, you must add code along the following lines, to copy the relevant material properties data to the relevant node fields. I have provided `copyArray`; it may have been possible to get away with assigning arrays via pointers, especially where the `from` arrays are all global variables, and so gave global lifetime; however, in general I consider the latter practice very dangerous, and the likely source of very difficult to locate run-time errors.

```
/*
  copyArray(torso_node.msp, m_sp, 4); copyArray(torso_node.mam, m_am, 4);
  copyArray(torso_node.mdi, m_di, 4); torso_node.msh= m_sh;
*/
```

You should have a different colour for head, torso, upper arms, lower arms, finger, upper legs, lower legs. Keep the light white, and pointing face on. Note that this question should be more tedious than difficult. The code that I've added (or not removed) should give you plenty of hints.

[50 + 50 = 100 marks]

Make sure you have completed task 1. before you attempt task 3.

3. Attempt to make at least *one* set of limbs (e.g. upper arms, lower arms, legs ...) more realistic looking using ellipsoidal shapes. See the code for the head (non-uniformly scaled sphere) to see how to make ellipsoidal shapes. If you have problems here with odd specular or diffuse lighting effects, ignore; scaling may damage normals(?).

[20 + 20 = 40 marks]

4. Add some other body part of your choice or something that the robot is carrying.

[20 + 20 = 40 marks]

5. Draw a *tree* diagram showing the *hierarchy* of robot parts.

[20 marks]

6. Write a brief note on *scene graphs*; 250 words or less; diagram(s) strongly encouraged; examples of systems that use them will be useful.

[40 marks]