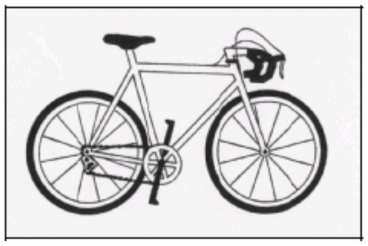
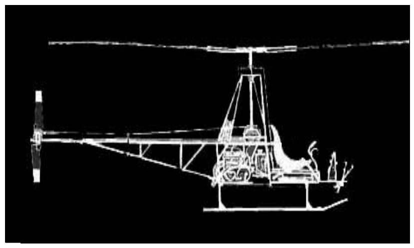
CSE 170 Computer Graphics

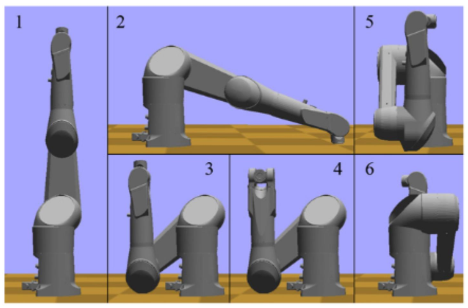
Project 1 – Animating a Hierarchical Object

Deadline:
Wed/Thu Labs (Mar 21/22) - presentation and submission until end of lab
(The deadline is your last lab before Spring Break)

Late Projects:
We will accept presentations and submissions until April 4/5, but under a 20% penalty







Description

You will model a hierarchical object of your choice and animate it using transformations. Use of any functionality in SIG or from external tools is acceptable but the requirements listed in this document have to be implemented with your own code.

You can choose which object to build and animate. The pictures above give some examples but you can certainly find many other interesting options. It can be any object as long as you follow the requirements described below.

Requirements

1. (25%) Parts: you will have to model a hierarchical object of at least 5 parts interconnected by joints which will be animated. The joints have to create at least 5 "degrees of freedom" in total, and produce movements in 3 dimensions. You also have to place your object in a scene with a floor and a few other simple objects around in order to give some context to the scene. The objects may be all based on primitives (tubes, capsules, spheres, etc.), or you may as well use GsModel objects loaded from .obj models retrieved from the web.

Below are some examples for different objects taking into account their typical parts:

Bicycle	Crane	Helicopter	Character
- Wheels	- Wheels	- Body	- Body
- Frame	- Cabin	- Main rotor axis	- Right Arm
- Pedals	- Main arm	- Main rotor blades	- Left Arm
- Fork	- Secondary arm	- Tail rotor axis	- Right Leg
- Handlebar	- Hook	- Tail rotor blades	- Left Leg
- Floor with objects	- Floor with objects	- Floor with objects	- Floor with objects

2. (30%) Animation: Your object will have two types of animation: global motion (to move it around your scene), and joint motion (to move the joints of the object). If you are implementing a vehicle what you want to achieve is a controllable vehicle where all of the parts correctly move in a synchronized way.

Up/Down and Left/Right keys: these keys are reserved for animating the global motion of your object. You may use additional keys if you need (for ex. like PageUp/PageDown for elevation), but at least the arrow keys have to meaningfully control the global motion of your object. For a bicycle or a vehicle what makes sense is to change the linear velocity of your vehicle with the Up/Down keys and change the direction of the wheels with the Left/Right keys. For other objects you just have to find a meaningful way to control the global motion of your object with the arrow keys. If for your object there is no meaningful way to control its global motion you may just use the arrow keys to move it around the scene.

Keys Q/A, W/S, E/D, etc.: These keys are reserved to control the joints of your object. Use as many keys as you need (at least 5 pairs) to achieve a meaningful control of the individual parts of your object. Each pair of keys (Q/A, W/S, E/D, etc.) will control one degree of freedom of your object along its normal and opposite directions. For example you can use Q/A to test a steering wheel rotation towards the left and right directions, by increments for each key press. The keys may be used only to test the joint movements of your object, while other keys and controls (like the arrow keys) are reserved for proper global object movement.

→ *Note that the animations must be implemented by using transformation matrices, either using the scene graph functionality or by multiplying matrices directly to your objects.*

3. (30%) Visualization. In order to test and improve your project you also have to develop the following visualization tools/effects:

a) Include at least two different camera modes to be switched with the space bar: the first mode can be a fixed one, but the second mode is from a moving camera, like from inside the vehicle or for example from an animated bird's eye view camera moving around a circle on the top of your scene. You may experiment with additional modes if you wish but at least one moving camera should be there. To manipulate the camera just access the GsCamera object of the viewer with method camera() and change its parameters directly. As an example, the following simple code controls a 3 seconds camera movement:

```
double lt, t0=gs_time();
do
{       lt = gs_time()-t0;
        camera().eye.x += 0.001f;
        camera().center.x += 0.001f;
        camera().up.x += 0.001f;
        render();
        ws_check();
        message().setf("local time=%f",lt);
} while (lt<3.0f);
```

b) Include at least one textured object. You may just load a .obj file of your choice which already comes with a texture or you may texture a GsModel yourself. The make_primitive() method of GsModel does not create texture coordinates, but you can study how the vertices of the primitives are created and then add the texture coordinates directly to a GsModel. You may also just reuse your textured torus in your project or just texture a very simple object for example to have a grass texture on your floor plane.

c) You also have to display on the floor a correct shadow of your hierarchical object. You may just use your geometric projection solution for PA3 and apply it to your hierarchical object.

4. (10%) Overall Project Quality. As usual, it is also a requirement that your project looks good. Note that you do not need to make a complex project or go beyond the minimum requirements to get all points; it is perfectly fine to stay with simple primitives as long as everything works and looks carefully prepared.

This project will certainly require several days of implementation, so come to all the labs to have continuous support during your development!!

5. (5%) Presentation and Project Submission
As usual, you will present your project to the TA by the deadline and then submit your code. You are encouraged to show your results to your colleagues in the lab, and everyone is welcome to come to other labs to see the projects. Unfortunately it is not feasible to have everyone show their projects to everyone else, but we are going to post your results in a web page.

⇨ As part of your submission, in addition to uploading your code as usual, you will also be required to supply one .png or .jpg snapshot of your project. This image should be the best representative image to highlight your project. We will later organize a web page with all pictures. Please name your picture in the following format:

```
proj1_[your last name]_[your first name].jpg/png
```

(Note: if for any reason you do not want your image posted in a web page, just let me know.)

Summary of Requirements

1. (25%) Parts: at least 5 connected parts with 5 degrees of freedom (moving in 3D), and a scene with extra objects.

2. (30%) Animation: global motion controlled with arrow keys and individual animation of main degrees of freedom (at least 5) with Q/A, W/S, E/D, etc, keys.

3. (30%) Visualization: include a) static and moving camera (10%), b) textured objects (10%), and c) a shadow of the hierarchical object (10%).

4. (10%) Overall Quality: your project has to look good! (You may get full points here even if you do not complete all items in requirement 3.)

5. (5%) Presentation and Submission: Do not forget to submit your representative image!

⇨ Heads-up about Project 2 (the final project of the class): For Project 2 you will be able to choose your topic and to work on it in groups. It is common to re-use several pieces of Project 1 in Project 2. We will have class presentations of all final projects on May 9.