# CS295/CS395/CSYS395: Evolutionary Robotics

## Programming Assignment 6 of 10

Assigned: Friday, October 7, 2011

Due: Friday, October 14, 2011 by midnight

**Description:** In this week's assignment you will be adding eight joints to connect together the nine body parts comprising the robot (Fig. 1). You will accomplish this in the same way as for adding bodies to ODE: add one joint, compile and run the application so that you can see the effect of the addition on the simulation, only then add a second joint, and so on.

1. Back up Assignment_5 on a flash drive or another computer so that you can always return to your completed fifth assignment.

2. Copy directory Assignment_5, which contains your submitted document and the entire ODE folder. Rename the new directory Assignment_6.

3. At the top of the test_buggy code, add a new data structure that will store the eight joints to be created:
   `static dJointID joints[8];`

4. Now create a function `CreateHinge(int index,...)` that will create a joint. You will need to pass this function a number of parameters: you need to tell it which two bodies to attach together; where the joint's fulcrum should be; and the axis of the joint. If you are unfamiliar with these concepts, refer to the Lecture 6 on physical simulation.

5. Inside this function you will use these parameters to:
   join two bodies together (`dJointAttach(...)`),
   set the position of the joint's fulcrum (`dJointSetHingeAnchor(...)`), and
   set the joint's orientation (`dJointSetHingeAxis(...)`).
   For more information about these three functions refer to ODE's documentation.

6. Create another function `DestroyHinge(int index)` that removes the joint from the simulation.

7. Add a line `CreateHinge(0,...);` that creates the first joint, which attaches the right lower leg to the right upper leg (Fig. 1a). This line should be placed just after all of the bodies have been created, but before simLoop is called.

8. Add a second line `DestroyHinge(0);` after simLoop returns.

9. Compile and run the simulation in paused mode. Press CTRL-O repeatedly to step through the simulation. You should see that, as the right upper leg falls, it is stays connected to the right lower leg at the 'knee'. You should see something like that of Fig. 1a. Screen capture this image and store it in your document.

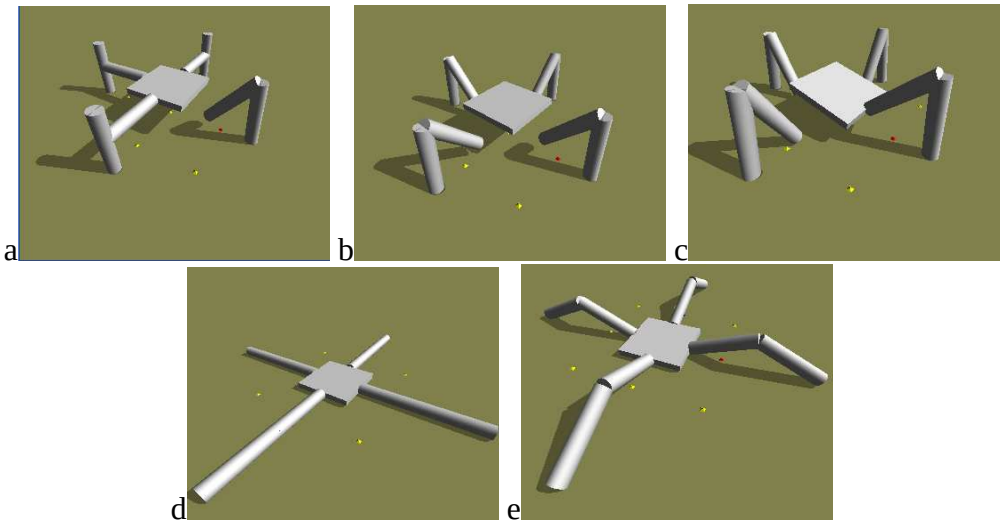


Figure 1: Incremental addition of joints to the quadrupedal robot.

10. Now add the second joint that connects the left upper leg to the left lower leg, recompile and rerun.

11. Add the third joint that connects the front (into the screen) upper leg to the front lower leg, recompile and rerun.

12. Add the fourth joint that connects the back (toward the viewer) upper leg to the back lower leg, recompile and rerun. This should now produces an image like that in Fig. 1b. Screen capture and store in your document.

13. Using the same procedure, add the fifth joint that connects the right upper leg to the main body. Recompile, rerun and step through the simulation to get an image like that in Fig. 1c. Screen capture and copy into your document.

14. Now iteratively add the sixth, seventh and eighth joint connect the remaining three legs to the main body. The robot should now 'sit down', and its legs should flatten out, producing an image as in Fig. 1d.

15. We now need to constrain the range of motion of each joint. For simplicity, we will set each joint to only rotate through $[-45^o, 45^o]$. To do this, you will need to add two more lines to `CreateHinge(...)`:
```
dJointSetHingeParam(joints[index],dParamLoStop,-45*(3.14159/180));
dJointSetHingeParam(joints[index],dParamHiStop,+45*(3.14159/180));
```
What is the `x*(3.14159/180)` for?

16. Now when you recompile and rerun the simulation you should obtain an image as in Fig. 1e. Screen capture and copy into your document, and submit your document.