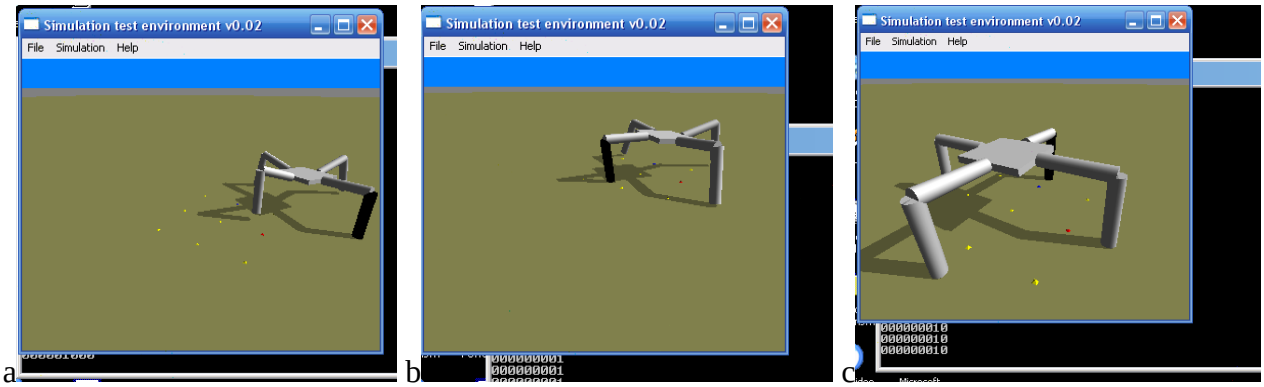

Figure 1: Visual display of the touch sensors firing.

CS295/CS395/CSYS395: Evolutionary Robotics

Programming Assignment 8 of 10

Assigned: Friday, October 21, 2011

Due: Friday, October 28, 2011 by midnight

Description: In this week's assignment you will add sensors to your robot. You will add one binary touch sensor in each of the four lower legs: the sensor will fire when that leg is touching the ground, and will not fire when the leg is off the ground. Adding touch sensors requires four basic steps:

1. Set all touch sensor values to zero at the outset of each time step of the simulation.
2. Add code so that, when nearCallback is called because there is a collision between a leg and the ground plane, ODE knows which leg collided.
3. Set the touch for that leg to 1.
4. For those legs touching the ground, paint them black; those that do not touch the ground, paint white (see Fig. 1).

The following instructions will realize these four steps.

1. Back up Assignment_7 on a flash drive or another computer so that you can always return to your completed seventh assignment.
2. Copy directory Assignment_7, which contains your submitted document and the entire ODE folder. Rename the new directory Assignment_8.
3. ODE allows the user to associate a pointer with each ODE body using `dBodySetData(dBodyID,void *data)`. You will create such a data structure, and store the ID of each ODE body there. That is, the third ODE object will have an ID of 3. At the top of the program, create an array of integers `int IDs[9]`.

4. Just after the array is created, set the ith element of the array to i.

5. In `CreateBox(...)` and `CreateCylinder(...)`, after the ODE body has been created, associate a pointer to the correct element in this array with the object:
   `dBodySetData(body[index],&IDs[index]);`

6. Now, when `nearCallback` is called, you can determine which object is in contact with the ground plane. After these lines in `nearCallback`
   ```
   int g1 = (o1 == ground );
   int g2 = (o2 == ground );
   if (!(g1 ĝ2)) return;,
   ```
   one (but not both) of the objects is an ODE body, and the other is the ground plane. Within the code clause that is executed if a collision has actually occurred—that is, after
   ```
   n = dCollide (o1,o2,N,&contact[0].geom,sizeof(dContact));
   if (n > 0) {
   ```
   Add some code to check which object is the ground plane, and then get the user data from the *other* object. For example if o1 is the ground plane, then o2 is an ODE body. You can get the ID of o2 using
   ```
   int *IDPointer;
   ...
   IDPointer = (int*)dBodyGetData(dGeomGetBody(o2));
   ```
   If o2 is the ground plane, then you need to get the ID of o1.

7. Once you have this ID, print it to the output terminal:
   `printf("%d \n",*IDPointer);`. You should get something like the following when the simulation is running unpaused:
   ```
   5
   7
   ...
   6
   ```
   If the robot is moving randomly (Fig. 1), only the four lower legs (with IDs 5,6,7 and 8) will come into contact with the ground plane.

8. Now, at the top of the program, create a new integer array that will store the values of the four touch sensors:
   `int touches[9];`
   `touches[i]` will be set to 1 when the touch sensor in the ith body part is firing, and zero otherwise. Only `touches[5]` through `touches[8]` will be used, because the first five body parts do not have touch sensors.

9. Calls to `simLoop` and `nearCallback` alternate: After one pass through `simLoop` completes, all of the `nearCallback` calls are made. Once they are complete, a new pass through `simLoop` is made, and so on. So, we need to set all of the touch sensors to zero at the end of `simLoop`, and then set all of the touch sensors that are firing to one during the `nearCallback` calls. Thus, when `simLoop` begins, all elements in `touches` are set correctly. First, back in `nearCallback`, after the print statement you can set the corresponding touch sensor as follows:
```
printf("%d \n",*IDPointer);
touches[*IDPointer] = 1;
```

10. Now, at the very end of `simLoop`, add a `for` loop that sets each element of `touches` to zero.

11. Somewhere within the
`if ( !pause ) {`
clause of `simLoop` create a `for` loop that prints out the values of `touches`. When the simulation is run it should produce something like:
```
00001101
00000101
...
00001000
```
(You can see this in Fig. 1).

12. Now, within `DrawCylinder(i)`, you can include an `if` clause that will draw the ith cylinder black
`dsSetColor (0,0,0);`
if the ith element of `touches` is one, and draw it white
`dsSetColor (1,1,1);`
otherwise. When you compile and run the simulation, you should see that whenever a lower leg comes in contact with the ground it turns black, and turns white when it is lifted above the ground. Capture three images showing different legs doing this as in Fig. 1, copy them into your document and submit to the T.A.