```
1
2  //-------------------------------------------------------------------------------------------
3  /*! ¥file
4      ¥brief  4-legged robot simulator - server
5      ¥author Akihiko Yamaguchi
6      ¥date   Mar.13 2007  */
7  //-------------------------------------------------------------------------------------------
8  #ifndef ODE_MINOR_VERSION
9    #error ODE_MINOR_VERSION should be set in compile
10   #error   ex. -DODE_MINOR_VERSION=10
11 #endif
12 //-------------------------------------------------------------------------------------------
13 #include <ode/ode.h>
14 #include <drawstuff/drawstuff.h>
15 #include <iostream>
16 #undef PACKAGE_BUGREPORT
17 #undef PACKAGE_NAME
18 #undef PACKAGE_STRING
19 #undef PACKAGE_TARNAME
20 #undef PACKAGE_VERSION
21 #include <octave/config.h>
22 #include <octave/Matrix.h>
23 //-------------------------------------------------------------------------------------------
24 #include <cstdlib>
25 #include <cstdio>
26 #include <cstring>
27 #include <unistd.h>
28 #include <sys/types.h>
29 #include <sys/socket.h>
30 #include <sys/un.h>
31 //-------------------------------------------------------------------------------------------
32 #include "protocol.h"
33 //-------------------------------------------------------------------------------------------
34 #ifdef _MSC_VER
35 #pragma warning(disable:4244 4305)  // for VC++, no precision loss complaints
36 #endif
37 // select correct drawing functions
38 #ifdef dDOUBLE
39 #define dsDrawBox dsDrawBoxD
40 #define dsDrawSphere dsDrawSphereD
41 #define dsDrawCylinder dsDrawCylinderD
42 #define dsDrawCapsule dsDrawCapsuleD
43 #define dsDrawConvex dsDrawConvexD
44 #endif
45 //-------------------------------------------------------------------------------------------
46 using namespace std;
47 //-------------------------------------------------------------------------------------------
48 #include "robot.cpp"
49 //-------------------------------------------------------------------------------------------
50
51
52 //===========================================================================================
53 //! ¥brief ふたつのオブジェクト o1, o2 が衝突しそうならこのコールバック関数が呼ばれる
54 //! ¥note 衝突しているかいないかはこの関数で（ユーザが）判定し，衝突していれば接触点にリンクを追加する.
55 static void nearCallback (void *data, dGeomID o1, dGeomID o2)
56 //===========================================================================================
57 {
58   // exit without doing anything if the two bodies are connected by a joint
59   dBodyID b1 = dGeomGetBody(o1);
60   dBodyID b2 = dGeomGetBody(o2);
61   if (b1 && b2 && dAreConnectedExcluding(b1,b2,dJointTypeContact)) return;
62
63   dContact contact[MAX_CONTACTS];   // up to MAX_CONTACTS contacts per box-box
64   for (int i=0; i<MAX_CONTACTS; i++)
65   {
66     contact[i].surface.mode = dContactBounce | dContactSoftCFM;
67     contact[i].surface.mu = dInfinity;
68     contact[i].surface.mu2 = 0;
69     contact[i].surface.bounce = 0.1;
70     contact[i].surface.bounce_vel = 0.1;
71     contact[i].surface.soft_cfm = 0.01;
72   }
73   if (int numc = dCollide (o1,o2,MAX_CONTACTS,&contact[0].geom,sizeof(dContact)))
74   {
75     for (int i=0; i<numc; i++)
76     {
77       dJointID c = dJointCreateContact (world.id(),contactgroup.id(),contact+i);
78       dJointAttach (c,b1,b2);
79     }
80   }
81 }
82 //-------------------------------------------------------------------------------------------
83
84
85 //===========================================================================================
86 //! ¥brief start simulation - set viewpoint
87 static void start()
88 //===========================================================================================
89 {
90   #if ODE_MINOR_VERSION>=10
91     dAllocateODEDataForThread(dAllocateMaskAll);
92   #endif
93
94   static float xyz[3] = {0.75,1.3,1.0};
95   static float hpr[3] = {-120.0,-16.0,0.0};
96
97   dsSetViewpoint (xyz,hpr);
98   // cerr << "Press 'R' to reset simulation¥n" << endl;
99 }
100 //-------------------------------------------------------------------------------------------
101
102 //===========================================================================================
103 //! ¥brief キーイベントのコールバック関数
104 //! ¥param[in] cmd  入力キー
105 static void keyEvent (int cmd)
```

```
106 //==========================================================================================
107 {
108   // if (cmd=='r'||cmd=='R')
109   // {
110   //   create_world();
111   // }
112 }
113 //------------------------------------------------------------------------------------------
114 
115 static void getJointState (double state[JOINT_STATE_DIM])
116 {
117   for (int j(0);j<JOINT_NUM;++j)
118   {
119     state[j] = joint[j].getAngle();
120     state[JOINT_NUM+j] = joint[j].getAngleRate();
121   }
122   // cerr<<"joint1= ";for(int j(0);j<JOINT_STATE_DIM;++j)cerr<<" "<<state[j];cerr<<endl;
123 }
124 //------------------------------------------------------------------------------------------
125 
126 static void getBaseState (double state[BASE_STATE_DIM])
127 {
128   state[0]  = body[0].getPosition()[0];  // x
129   state[1]  = body[0].getPosition()[1];  // y
130   state[2]  = body[0].getPosition()[2];  // z
131   state[3]  = body[0].getQuaternion()[0]; // quaternion (w)
132   state[4]  = body[0].getQuaternion()[1]; // quaternion (x)
133   state[5]  = body[0].getQuaternion()[2]; // quaternion (y)
134   state[6]  = body[0].getQuaternion()[3]; // quaternion (z)
135 
136   state[7]  = body[0].getLinearVel()[0];  // vx
137   state[8]  = body[0].getLinearVel()[1];  // vy
138   state[9]  = body[0].getLinearVel()[2];  // vz
139   state[10] = body[0].getAngularVel()[0] ; // wx
140   state[11] = body[0].getAngularVel()[1] ; // wx
141   state[12] = body[0].getAngularVel()[2] ; // wx
142 }
143 //------------------------------------------------------------------------------------------
144 
145 
146 
147 //------------------------------------------------------------------------------------------
148 static int    global_file_descriptor(-1);
149 static int    window_x(400), window_y(400);
150 // static const  dReal time_step (0.0005); // シミュレーションきざみ幅(0.5[ms])
151 static ColumnVector  input_torque (JOINT_NUM, 0.0);
152 //------------------------------------------------------------------------------------------
153 
154 
155 
156 void stepSimulation (const dReal &time_step)
157 {
158   for (int j(0); j<JOINT_NUM; ++j)
159     joint[j].addTorque(input_torque(j));
160   // cerr<<"torque= "<<input_torque.transpose()<<endl;
161 
162   // シミュレーション
163   space.collide (0,&nearCallback);
164   world.step (time_step);
165   // time += time_step;
166   // remove all contact joints
167   contactgroup.empty();
168 }
169 //------------------------------------------------------------------------------------------
170 
171 
172 bool oct_robot_server (void)
173 {
174   TXData data;
175   while (1)
176   {
177     if (global_file_descriptor<0)
178     {
179       cerr<<"connection terminated (unexpected error)."<<data.command<<endl;
180       exit(1);
181     }
182     read (global_file_descriptor, (char*)&data, sizeof(data));
183     switch (data.command)
184     {
185       case  ORS_START_SIM        :
186         return true;
187       case  ORS_STOP_SIM         :
188         return false;
189       case  ORS_STEP_SIM         :
190         stepSimulation (data.dvalue);
191         break;
192       case  ORS_RESET_SIM        :
193         create_world();
194         break;
195       case  ORS_DRAW_WORLD       :
196         return true;
197       case  ORS_SET_TORQUE       :
198         input_torque(data.step) = data.dvalue;
199         break;
200       case  ORS_SET_WINDOWSIZE   :
201         if (data.step==0)  window_x=data.ivalue;
202         else if (data.step==1)  window_y=data.ivalue;
203         break;
204       case  ORS_GET_JOINT_NUM    :
205         write (global_file_descriptor, (char*)&JOINT_NUM, sizeof(JOINT_NUM));
206         break;
207       case  ORS_GET_JSTATE_DIM   :
208         write (global_file_descriptor, (char*)&JOINT_STATE_DIM, sizeof(JOINT_STATE_DIM));
209         break;
210       case  ORS_GET_BSTATE_DIM   :
```

```
211         write (global_file_descriptor, (char*)&BASE_STATE_DIM, sizeof(BASE_STATE_DIM));
212         break;
213       case  ORS_GET_JOINT_STATE  :
214         getJointState (joint_state);
215         // cerr<<"joint2= ";for(int j(0);j<JOINT_STATE_DIM;++j)cerr<<" "<<joint_state[j];cerr<<endl;
216         write (global_file_descriptor, (char*)joint_state, sizeof(double)*JOINT_STATE_DIM);
217         break;
218       case  ORS_GET_BASE_STATE   :
219         getBaseState (base_state);
220         write (global_file_descriptor, (char*)base_state, sizeof(double)*BASE_STATE_DIM);
221         break;
222       default :
223         cerr<<"in oct_robot_server(): invalid command "<<data.command<<endl;
224         return false;
225     }
226   }
227 }
228 //-------------------------------------------------------------------------------------------
229
230 void setup_server (void)
231   // ref. http://www.ueda.info.waseda.ac.jp/~toyama/network/example1.html
232 {
233   int    fd1;
234   struct sockaddr_un    saddr;
235   struct sockaddr_un    caddr;
236   int    len;
237
238   if ((fd1 = socket (PF_UNIX, SOCK_STREAM, 0)) < 0)
239   {
240     perror("socket");
241     exit(1);
242   }
243
244   bzero ((char *)&saddr, sizeof(saddr));
245   // ソケットの名前を代入
246   saddr.sun_family = AF_UNIX;
247   strcpy (saddr.sun_path, SOCK_NAME);
248   // ソケットにアドレスをバインド
249   unlink(SOCK_NAME);
250   if (bind(fd1, (struct sockaddr *)&saddr,
251            sizeof(saddr.sun_family) + strlen(SOCK_NAME)) < 0){
252     perror("bind");
253     exit(1);
254   }
255   // listen をソケットに対して発行
256   if (listen(fd1, 1) < 0)
257   {
258     perror("listen");
259     exit(1);
260   }
261
262   len = sizeof(caddr);
263   /*
264     * accept()により、クライアントからの接続要求を受け付ける。
265     * 成功すると、クライアントと接続されたソケットのディスクリプタが
266     * fd2に返される。このfd2を通して通信が可能となる。
267     * fd1は必要なくなるので、close()で閉じる。
268     */
269   if ((global_file_descriptor = accept(fd1, (struct sockaddr *)&caddr, (socklen_t*)&len)) < 0)
270   {
271       perror("accept");
272       exit(1);
273   }
274   close(fd1);
275 }
276 //-------------------------------------------------------------------------------------------
277
278
279 //===========================================================================================
280 /*! ¥brief 描画(OpenGL)のコールバック関数.
281     ¥param[in] pause 停止モードなら true (0以外)
282
283     シミュレーションのきざみ time_step=0.0005[s] に対して描画は 50 fps 程度で十分なので,
284     1 frame ごとに simStepsPerFrame=1.0/time_step/FPS=40 回ダイナミクスのシミュレーションを回す.  */
285 static void simLoop (int pause)
286 //===========================================================================================
287 {
288 //   static dReal time(0.0); // シミュレーション時間
289   if (!pause)
290   {
291     if (!oct_robot_server())  dsStop();
292   }
293
294   draw_world();
295 }
296 //-------------------------------------------------------------------------------------------
297
298 static void stopSimulation (void)
299 {
300   close (global_file_descriptor);
301   global_file_descriptor = -1;
302 }
303 //-------------------------------------------------------------------------------------------
304
305
306 int main (int argc, char **argv)
307 {
308   dsFunctions fn; // OpenGL 出力用オブジェクト
309   fn.version = DS_VERSION;
310   fn.start = &start;
311   fn.step = &simLoop;
312   fn.command = &keyEvent;
313   fn.stop = &stopSimulation;
314   char path_to_textures[]="textures";
315   fn.path_to_textures = path_to_textures;  //! ¥note カレントディレクトリに textures へのリンクが必要
```

```
316
317   #if ODE_MINOR_VERSION>=9
318   # if ODE_MINOR_VERSION==9
319     dInitODE();
320   # elif ODE_MINOR_VERSION>=10
321     dInitODE2(0);
322   # endif
323   #endif
324
325   setup_server();
326   if (oct_robot_server())
327   {
328     create_world();
329     // run simulation
330     if(window_x>0 && window_y>0)
331       dsSimulationLoop (argc,argv,window_x,window_y,&fn);
332     else
333       while(oct_robot_server());
334   }
335   if(global_file_descriptor!=-1)
336   {
337     close (global_file_descriptor);
338     global_file_descriptor = -1;
339   }
340
341   #if ODE_MINOR_VERSION>=9
342     dCloseODE();
343   #endif
344   return 0;
345 }
346 //-----------------------------------------------------------------------------
347
```