```
//-------------------------------------------------------------------------------------------
/*! ¥file
    ¥brief  create acrobot for ODE
    ¥author Akihiko Yamaguchi
    ¥date   Dec.26 2007 */
//-------------------------------------------------------------------------------------------
// dynamics and collision objects
static dWorld world;
static dSimpleSpace space (0);
static dPlane plane;
static dBody body[3];
static const int JOINT_NUM(2);
/*mod*/static const int JOINT_STATE_DIM(JOINT_NUM*2);  // 関節状態の次元
/*mod*/static const int BASE_STATE_DIM(13);  // ベース状態の次元
//*mod*/static const int STATE_DIM(4);
static dHingeJoint joint[JOINT_NUM];
static dFixedJoint base_joint; // 支柱を地面(z=0)に固定する
static dJointGroup contactgroup;
static dBox  LinkBase;
/*mod*/static dCapsule Link1, Link2;

const int  MAX_CONTACTS (10); // maximum number of contact points per body
//-------------------------------------------------------------------------------------------
/*mod*/static double joint_state[JOINT_STATE_DIM];
/*mod*/static double base_state[BASE_STATE_DIM];
//-------------------------------------------------------------------------------------------


//===========================================================================================
//! ¥brief シミュレーションオブジェクトを作成
void create_world( void )
//===========================================================================================
{
  // acrobot の回転軸（以下，支柱）は ここでは 直方体(Box)にしています.
  const dReal param_h0     = 0.05; // 支柱（直方体）の高さ[m]
  const dReal param_wx0    = 0.05; // 同幅(x)
  const dReal param_wy0    = 0.80; // 同幅(y)
  const dReal param_z0     = 1.20; // 支柱の垂直位置[m]

  const dReal param_l1     = 0.50; // 第1リンク（支柱に近いリンク）の長さ[m]
  const dReal param_d1     = 0.15; // 同直径[m]
  const dReal param_l2     = 0.50; // 第2リンク（支柱に近いリンク）の長さ[m]
  const dReal param_d2     = 0.15; // 同直径[m]

  const dReal density      = 1000.0;  // 各リンクの密度[kg/m^3]. 参考(?)`人体の密度' は 900~1100 kg/m^3 (wikipedia)

  int i;
  contactgroup.create (0);
  world.setGravity (0,0,-9.8);  // 重力 [m/s^2]
  dWorldSetCFM (world.id(),1e-5);
  plane.create (space,0,0,1,0); // 地面（平面）.

  i=0; {
    body[i].create (world);
    body[i].setPosition (0.0, 0.0, param_z0); // 支柱の中心座標
    dReal xx=param_wx0, yy=param_wy0, zz=param_h0;
    dMass m;
    m.setBox (density,xx,yy,zz);
    body[i].setMass (&m);
    LinkBase.create (space,xx,yy,zz);
    LinkBase.setBody (body[i]);
  }
  i=1; {
    body[i].create (world);
    body[i].setPosition (0.0, 0.0, param_z0-0.5*param_l1); // リンク1の中心座標
    dReal rad=0.5*param_d1, len=param_l1-2.0*rad;
    dMass m;
    m.setCappedCylinder (density,3,rad,len);  // direction(3): z-axis
    body[i].setMass (&m);
    Link1.create (space,rad,len);
    Link1.setBody (body[i]);
  }
  i=2; {
    body[i].create (world);
    body[i].setPosition (0.0, 0.0, param_z0-param_l1-0.5*param_l2); // リンク2の中心座標
    dReal rad=0.5*param_d2, len=param_l2-2.0*rad;
    dMass m;
    m.setCappedCylinder (density,3,rad,len);  // direction(3): z-axis
    body[i].setMass (&m);
    Link2.create (space,rad,len);
    Link2.setBody (body[i]);
  }

  i=0; {
    const dReal *pos = body[0].getPosition();
    joint[i].create (world);
    joint[i].attach (body[0],body[1]);
    joint[i].setAnchor (pos[0],pos[1],pos[2]); // 回転中心=支柱の中心(=原点)
    joint[i].setAxis (0.0,1.0,0.0); // 回転軸=y軸
    // joint[i].setParam (dParamHiStop, +0.5*M_PI); // 関節の可動範囲を制約するときに使う
    // joint[i].setParam (dParamLoStop, -0.5*M_PI); // acrobot の場合は省略
  }
  i=1; {
    const dReal *pos = body[1].getPosition();
    joint[i].create (world);
    joint[i].attach (body[1],body[2]);
    joint[i].setAnchor (pos[0],pos[1],pos[2]-0.5*param_l1); // 回転中心=リンク1とリンク2の間
    joint[i].setAxis (0.0,1.0,0.0); // 回転軸=y軸
  }
  base_joint.create(world);
  base_joint.attach(body[0].id(),0); // 支柱(body[0]) と 平面(0)の間の固定リンク. 支柱が固定される.
  base_joint.set();
}
//-------------------------------------------------------------------------------------------

```

```
106
107 //=============================================================================================
108 //! ¥brief 描画関数
109 void draw_world( void )
110 //=============================================================================================
111 {
112   int i;
113   dsSetColor (0,0.5,1);
114   dsSetTexture (DS_WOOD);
115   dReal rad, len;
116   dReal sides[4];
117   dBox *blink;
118   dCapsule *clink;
119   dsSetColorAlpha (1.0, 0.0, 0.0, 0.6);
120   i=0; blink=&LinkBase; blink->getLengths(sides); dsDrawBox (body[i].getPosition(),body[i].getRotation(),sides);
121   dsSetColorAlpha (0.0, 0.5, 1.0, 0.6);
122   i=1; clink=&Link1; clink->getParams(&rad, &len); dsDrawCappedCylinder (body[i].getPosition(),body[i].getRotation(),len,rad);
123   i=2; clink=&Link2; clink->getParams(&rad, &len); dsDrawCappedCylinder (body[i].getPosition(),body[i].getRotation(),len,rad);
124 }
125 //---------------------------------------------------------------------------------------------
126
127
128
129
```