第4部　ロボット制御演習 (Robot Control)

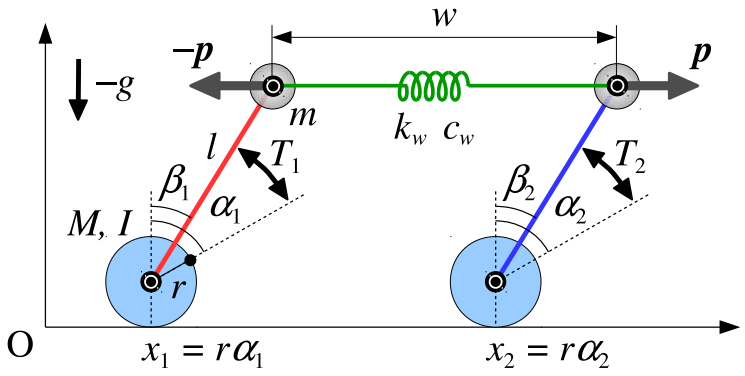
宇都宮大学工学研究科 吉田勝俊

2018.1.22 版

目 次

12　対戦型ロボット・シミュレータの作成

第 3 部 **11.4 節** p30 では，倒れすぎると転倒する車輪型倒立ロボットを作った．これを 2 台用意して，先端をバネとダンパーの直動リンクで結合した次のマルチロボットシステムを考える．



w はリンクの長さ，$\boldsymbol{p}$ は右側先端にリンクが及ぼす力である．作用・反作用により左側先端には $-\boldsymbol{p}$ がかかる．振り子棒とリンクは回転対偶をなし摩擦は無いものとする．また，リンクの質量は無視する．

ここで，左側のロボットのトルク T_1 に新たな制御入力 $u(t)$ を

$$T_1 = \overbrace{\underbrace{\mathrm{trap}(\beta, 0, \beta_0)}_{\text{制御の打切}} \cdot \overbrace{(K_1\alpha + K_2\dot{\alpha} + K_3\beta + K_4\dot{\beta})}^{\text{PD 制御による倒立安定化}} + u(t)}^{\text{第 3 部 11.4 節 p30}} \tag{12.1}$$

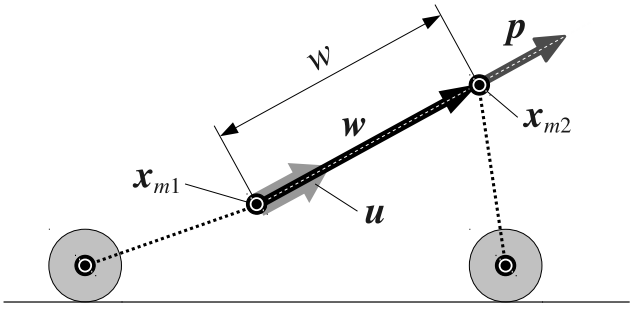
のように追加して，右側のロボットを転倒させる問題を考える．もし具体的な $u(t)$ を適切に構成できれば，手押し相撲を模したような対戦的挙動が実現するに違いない．

12.1　リンク結合の数理モデル

まず，リンクの自然長を w_0 とおき，$\boldsymbol{p}$ の大きさ (負値も認める) を

$$p \equiv -k_w(w - w_0) - c_w\dot{w} \tag{12.2}$$

で定めることにする．次に，ロボットの姿勢 $(\alpha_1, \beta_1, \alpha_2, \beta_2)$ から，$w, \dot{w}$ および $\boldsymbol{p}$ の方向を割り出すため，次のような幾何学を考える．



各先端の位置ベクトル $\boldsymbol{x}_{mi}$ は，第 3 部の式 (8.4) p20 にあるように，

$$\boldsymbol{x}_{mi} = \begin{bmatrix} r\alpha_i + l\sin\beta_i \\ r + l\cos\beta_i \end{bmatrix} \quad (i = 1, 2) \tag{12.3}$$

であった．これより左先端から右先端に向う変位ベクトル

$$\boldsymbol{w} \equiv \boldsymbol{x}_{m2} - \boldsymbol{x}_{m1} = \begin{bmatrix} r(\alpha_2 - \alpha_1) + l(\sin\beta_2 - \sin\beta_1) \\ l(\cos\beta_2 - \cos\beta_1) \end{bmatrix} = \begin{bmatrix} w_x \\ w_y \end{bmatrix} \tag{12.4}$$

が得られる．ゆえに，姿勢 $(\alpha_1, \beta_1, \alpha_2, \beta_2)$ におけるリンク長は

$$\begin{aligned} w = |\boldsymbol{w}| &= \sqrt{w_x^2 + w_y^2} \\ &= \sqrt{\Big(r(\alpha_2 - \alpha_1) + l(\sin\beta_2 - \sin\beta_1)\Big)^2 + \Big(l(\cos\beta_2 - \cos\beta_1)\Big)^2} \end{aligned} \tag{12.5}$$

と書ける．また $\dot{w}$ については，$\boldsymbol{w}$ の時間微分

$$\dot{\boldsymbol{w}} = \begin{bmatrix} r(\dot{\alpha}_2 - \dot{\alpha}_1) + l(\dot{\beta}_2\cos\beta_2 - \dot{\beta}_1\cos\beta_1) \\ l(-\dot{\beta}_2\sin\beta_2 + \dot{\beta}_1\sin\beta_1) \end{bmatrix} = \begin{bmatrix} \dot{w}_x \\ \dot{w}_y \end{bmatrix} \tag{12.6}$$

と，微分公式 $(\sqrt{x^2+y^2})^{\cdot} = (\dot{x}x + \dot{y}y)/\sqrt{x^2+y^2}$ より，

$$\dot{w} = (\dot{w}_x w_x + \dot{w}_y w_y)/w \tag{12.7}$$

となる．次に $\boldsymbol{p}$ を求める．振り子棒とリンクの間に摩擦はないので，$\boldsymbol{p}$ の方向は $\boldsymbol{w}$ の方向に一致する．ゆえに振り子の先端が受ける力 $\boldsymbol{p}$ は，$\boldsymbol{w}$ 方向の単位ベクトル $\boldsymbol{u}$ に，$\boldsymbol{p}$ の大きさ p を乗じたものとして

$$\boldsymbol{p} = p\boldsymbol{u}, \quad \boldsymbol{u} \equiv \boldsymbol{w}/w, \quad p = \text{式 (12.2) p37} \tag{12.8}$$

のように表せる．

振り子の先端には，以上の $\boldsymbol{p}$ のほか，第 3 部 (11.13) p31 の床の反力 $\boldsymbol{f}$ が加わる．具体的には，左の先端には合力 $\boldsymbol{f}_1 - \boldsymbol{p}$ が作用し，右の先端には合力 $\boldsymbol{f}_2 + \boldsymbol{p}$ が作用する．これらの作用を，姿勢角 α, β に関する一般化力 $\boldsymbol{\mathcal{F}}_{\boldsymbol{f}} = (f_\alpha, f_\beta)^T$ で表すと，第 3 部の変換公式 (11.12) p31 より，

$$\boldsymbol{\mathcal{F}}^1_{\boldsymbol{f}} = \begin{bmatrix} f^1_\alpha \\ f^1_\beta \end{bmatrix} = \begin{bmatrix} r & 0 \\ l\cos\beta_1 & -l\sin\beta_1 \end{bmatrix} (\boldsymbol{f}_1 - \boldsymbol{p}) \qquad \text{※左} \tag{12.9}$$

$$\boldsymbol{\mathcal{F}}^2_{\boldsymbol{f}} = \begin{bmatrix} f^2_\alpha \\ f^2_\beta \end{bmatrix} = \begin{bmatrix} r & 0 \\ l\cos\beta_2 & -l\sin\beta_2 \end{bmatrix} (\boldsymbol{f}_2 + \boldsymbol{p}) \qquad \text{※右} \tag{12.10}$$

となる (左右で $\boldsymbol{p}$ の符号に注意せよ).

実習 4.1　Code 8 を実行せよ．結合ロボットの運動が確認できる．そのまま実行すると，共倒れとなる．プログラムの 90 行付近

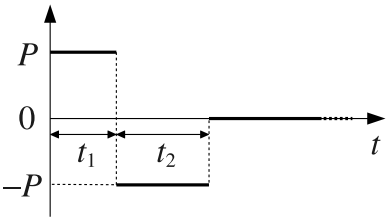
```
x01=[0; 0; 0; 2];  //左側の初期値
```

の 2 の部分 (β の初期角速度) だけを変更して，左の勝ちを作れるか？ 試みよ．

12.2 バンバン制御

車の位置や速度を，フル加速とフルブレーキングだけでコントロールするような制御方式を**バンバン制御** (bang-bang control) という．すなわち，制御入力が最大値と最小値しかとらないような制御方式をバンバン制御という．このようなバンバン制御の入力を，(12.1) p37 の制御入力 $u(t)$ に与えて，左のロボットを勝利に導こう．

例えば，段数が 2 のバンバン制御入力の例を以下に示す．

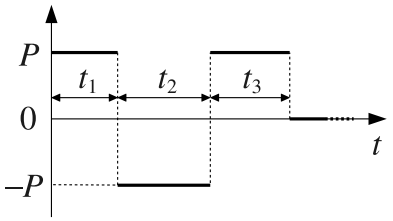


$$u(t) = \begin{cases} P & (0 \le t < t_1) \\ -P & (t_1 \le t < t_1 + t_2) \\ 0 & (\text{それ以外}) \end{cases} \tag{12.11}$$

この $u(t)$ は，$t = 0$ から t_1 秒間は $u(t) = P$ を出力し，続く t_2 秒間は逆向きの $u(t) = -P$ を出力する．それ以外の時刻では $u(t) = 0$，すなわち出力を停止する．Scilab では `if` 文を用いて，例えば，次のように表せる．

```
t1=0.36; t2=0.78; #切替時間
if (t<t1)
  u = P;
elseif (t<t1+t2)
  u = -P;
else
  u = 0;
end
```

同様にして，3 段の場合は，



$$u(t) = \begin{cases} P & (0 \le t < t_1) \\ -P & (t_1 \le t < t_1 + t_2) \\ P & (t_1 + t_2 \le t < t_1 + t_2 + t_3) \\ 0 & (\text{それ以外}) \end{cases} \tag{12.12}$$

```
t1=0.36; t2=0.78; t3=0.38; #切替時間
if (t<t1)
  u = P;
elseif (t<t1+t2)
  u = -P;
elseif (t<t1+t2+t3)
  u = P;
else
  u = 0;
end
```

となる．同様にして，切替時間 $t_4, t_5, \cdots$ を追加し，場合分けを追加すれば，任意の段数のバンバン制御入力が実現できる．

実習 4.2　**Code 9** を実行せよ．左のロボットが行使する 3 段のバンバン制御により，左のロボットが勝利する．なお，今回のアニメーションでは，水平方向の視点をこの系の重心に置いた．(重心運動に追従するカメラからの映像になっている)

13　研究課題

グループで協力して，以下の問題を解決せよ．分らないことがあれば，まずグループ内で解決を試みよ．それでも分らなければ，他のグループにも相談せよ．

問題 4.1　左 (赤) のロボットに与えるバンバン入力を調整して (切替時刻および段数),

1. 左 (赤) の勝ち
2. 右 (青) の勝ち
3. 引き分け
4. 共倒れ

を全て再現せよ．対応するバンバン入力を，数式と数値で記録せよ．

問題 4.2　左 (赤) のロボットが勝利し，なおかつ勝利 (右 (青) のロボットの先端が床に触れる) までの所要時間が最短となるような，バンバン入力を構成せよ．

問題 4.3　左 (赤) のロボットが勝利し，なおかつ右 (青) のロボットが左側に倒れるような，バンバン入力を構成せよ．

問題 4.4　支給プログラム **Code 9** では，右 (青) 用のバンバン入力 `bang2(t)` は空であり，左 (赤) が勝利した．この `bang2(t)` に何らかの処理を追加して，左 (赤) に勝て．ただし，次のルールに従うこととする．

- `bang2(t)` の中身以外の部分は，支給プログラム **Code 9** p42 のままとする．
- `bang2(t)` の中身の変更方法は問わない．

参考文献

[1] 吉田勝俊著 :「機械力学」（宇都宮大学大学生協）

[2] 吉田ほか 2 名 : “Artificial wrestling: A dynamical formulation of autonomous agents fighting in a coupled inverted pendula framework”, *Mechanical Engineering Journal*, Vol.2, No.5 (2015), `http://dx.doi.org/10.1299/mej.14-00518`

A　プログラム例

Code 8: “wip2.sce” (Scilab)

```
clear;clf();
//////////////// 諸元 ///////////////
M=1, J=0.1, m=5, r=0.2, l=1, g=9.8;
bmax=0.29;  //転倒角
mu=0.3;     //摩擦係数
w0=3;       //リンクの自然長
////////////////////////////////////
///// ステップ関数ほか /////
function y = step( x )
    s = 100;
    y = 1.0./(1.0+exp(-s*x));                  //ステップ関数
endfunction
function y = trap( x, x0, w )
    y = step((x-x0)+w).*step(-(x-x0)+w);     //台形関数
endfunction
function y = sgn( x )
    s = 1e4;
    y = (2.0/(1+exp(-s*x))-1.0);               //符号関数
endfunction
///// 床からの反力（ペナルティー法） /////
function ff = yuka(x)
    a=x(1); da=x(2); b=x(3); db=x(4);
    xm = [r*a+l*sin(b); r+l*cos(b)];
    dxm = [r*da+l*db*cos(b); -l*db*sin(b)];
    Kp=1e4; Cp=90; //これらの数値を変えると床の性質が変わる
    R = step(-xm(2))*(-Kp*xm(2)-Cp*dxm(2)); //垂直抗力
    F = -mu*R*sgn(dxm(1));                     //摩擦力
    ff = [F; R];                               //床からの反力
endfunction
///// リンクからの反力 /////
function pp = rinku(x)
    a1=x(1); da1=x(2); b1=x(3); db1=x(4);
    a2=x(5); da2=x(6); b2=x(7); db2=x(8);
    //リンクからの力
    ww = [r*(a2-a1)+l*(sin(b2)-sin(b1));...
                    l*(cos(b2)-cos(b1))];
    dww = [r*(da2-da1)+l*( db2*cos(b2)-db1*cos(b1));...
                        l*(-db2*sin(b2)+db1*sin(b1))];
    w = norm(ww); //wwの長さ
    dw = ( dww(1)*ww(1)+dww(2)*ww(2) )/w; //wwの長さの時間微分
    uu = ww/w; //単位ベクトル化
    ck = 5000; cw = 100; p = -ck*(w-w0)-cw*dw; //反力の大きさ
    pp = p*uu;
endfunction
///// 振り子先端に作用する力→一般化力 /////
function Ff = sentan(x,F)
    global r l;
    b=x(3);
    Ff = [r, 0; l*cos(b), -l*sin(b)]*F; //一般化力への変換公式
endfunction
///// 倒立制御入力 /////
function u = toritu(x,a0)
    a=x(1); da=x(2); b=x(3); db=x(4);
    /// 制御の打ち切り×(位置制御 + 倒立制御) ///
    u = trap(b,0,bmax)*(1*(a-a0) + 0.5*da + 40*b + 4*db); //a0は目標位置
endfunction
///// 運動方程式の右辺 ////
function h = uhen(x,F)
  a=x(1); da=x(2); b=x(3); db=x(4);
  A = [(M+m)*r^2+J, m*l*r*cos(b); ...
       m*l*r*cos(b),    m*l^2];
  bb = [m*l*r*db^2 *sin(b); m*g*l*sin(b)] + F; //Fは一般化力
  h = A\bb;
endfunction
///// 運動方程式の定義 /////
function dx = eom(t,x)
  pp = rinku(x);    //リンクからの反力
  ///// 左 /////
  a=x(1); da=x(2); b=x(3); db=x(4);
  ff=yuka([a; da; b; db]);                 //床反力
  Ff=sentan([a; da; b; db], ff - pp);      //振り子先端への力→一般化力
  T =toritu([a; da; b; db],0);             //倒立制御
  FT=[T; -T];                              //トルク→一般化力
  h=uhen([a; da; b; db], FT+Ff);           //加速度で整理した右辺
  dx(1) = x(2); dx(2) = h(1);              //運動方程式
  dx(3) = x(4); dx(4) = h(2);
  ///// 右 /////
```

```
  a=x(5); da=x(6); b=x(7); db=x(8);
  ff=yuka([a; da; b; db]);                    //床反力
  Ff=sentan([a; da; b; db], ff + pp);         //振り子先端への力→一般化力
  T =toritu([a; da; b; db],w0/r);             //倒立制御
  FT=[T; -T];                                 //トルク→一般化力
  h=uhen([a; da; b; db], FT+Ff);              //加速度の右辺
  dx(5) = x(6); dx(6) = h(1);                 //運動方程式
  dx(7) = x(8); dx(8) = h(2);
endfunction
///// 運動方程式を解く /////
tn=501; tt=linspace(0,0.01*tn,tn);
x01=[0; 0; 0; 2];        //左側の初期値
x02=[w0/r; 0; 0; 0];     //右側の初期値
xx=ode( [x01;x02], 0, tt, eom );
// ロボット描画用の関数
function R=Rot(a)              //回転
 R=[cos(a),-sin(a);...
    sin(a), cos(a)];
endfunction
function yy=Trans(xx,rr)     //平行移動
 yy(1,:)=xx(1,:) + rr(1)*ones(xx(1,:));
 yy(2,:)=xx(2,:) + rr(2)*ones(xx(2,:));
endfunction
function endp = draw_robot(a,b,sp) //ロボット描画
    x=r*a; //車輪中心の水平変位
    sn=5; sq=linspace(0,2*%pi*(sn-1)/sn,sn);
    spoke=r*[cos(sq);sin(sq)];          //原点にあるスポークの外周点
    spoke=Rot(-a) * spoke;              //角度a回転したスポークの外周点
    spoke=Trans(spoke,[x;r]);           //位置xのスポークの外周点
    xv=[x*ones(1:sn);spoke(1,:)]; yv=[r*ones(1:sn);spoke(2,:)];
    xsegs(xv,yv,2); xarc(x-r, 2*r, 2*r, 2*r, 0, 360*64); //スポーク; 車輪
    rod=[0,0;0,l];                      //原点にある棒の両端点
    rod=Rot(-b) * rod;                  //角度βだけ回転した棒の両端点
    rod=Trans(rod,[x;r]);               //位置xの棒の両端点
    plot( rod(1,:), rod(2,:),sp);       //棒の描画
    gc=gce(); gc.children.thickness=2;  //線の太さ
    endp=rod(:,2);                      //棒の先端
endfunction
function draw_pair(x)
    a1=x(1); b1=x(3); ep1=draw_robot(a1,b1,"r-");   //r 赤
    a2=x(5); b2=x(7); ep2=draw_robot(a2,b2,"b-");   //b 青
    plot([ep1(1),ep2(1)],[ep1(2),ep2(2)],"g-");     //リンクの描画
    gc=gca(); gc.isoview="on"; xgrid(4);            //縦横比1; グリッド;
    gc.data_bounds=[-2.5,-1;5.5,2];                 //座標軸の範囲
endfunction
drawlater; draw_pair( [x01;x02] ); drawnow;
sleep(2000); //2s待ち
realtimeinit(0.01); //アニメーションの時間刻み
for i=1:10:tn
    realtime(i);
    drawlater; clf();           //描画延期; 描画消去;
    draw_pair( xx(:,i) );       //描画消去;ロボット描画;
    xlabel(sprintf("%d / %d", i, tn));
    drawnow;                    //画面更新;
end
```

Code 9: “wip2-bb.sce” (Scilab)

```
clear;clf();
///// 左（赤）のバンバン入力 /////
function u=bang1(t)
    P = 6; //バンバン入力の強度
    /// 以下は3段の場合 ///
    t1=0.36; t2=0.78; t3=0.39; //切替時間の変更は0.01単位
    if (t<t1)
        u = P;
    elseif (t<t1+t2)
        u = -P;
    elseif (t<t1+t2+t3)
        u = P;
    else
        u = 0;
    end
endfunction
///// 右（青）のバンバン入力 /////
function u=bang2(t)
    u = 0; //0を返す = 何もしない
endfunction
/////////////// 諸元 ///////////////
M=2, J=0.1, m=5, r=0.2, l=1, g=9.8;
bmax=0.9;   //転倒角
```

```
mu=0.3;      //摩擦係数
w0=3;        //リンクの自然長
////////////////////////////////////
///// ステップ関数ほか /////
function y = step( x )
    s = 100;
    y = 1.0./(1.0+exp(-s*x));                  //ステップ関数
endfunction
function y = trap( x, x0, w )
    y = step((x-x0)+w).*step(-(x-x0)+w);       //台形関数
endfunction
function y = sgn( x )
    s = 1e4; //#1×10の4乗のこと = 10000
    y = (2.0/(1+exp(-s*x))-1.0);               //符号関数
endfunction
///// 床からの反力（ペナルティー法） /////
function ff = yuka(x)
    a=x(1); da=x(2); b=x(3); db=x(4);
    xm = [r*a+l*sin(b); r+l*cos(b)];
    dxm = [r*da+l*db*cos(b); -l*db*sin(b)];
    Kp=1e4; Cp=90; //これらの数値を変えると床の性質が変わる
    R = step(-xm(2))*(-Kp*xm(2)-Cp*dxm(2)); //垂直抗力
    F = -mu*R*sgn(dxm(1));                     //摩擦力
    ff = [F; R];                               //床からの反力
endfunction
///// リンクからの反力 /////
function pp = rinku(x)
    a1=x(1); da1=x(2); b1=x(3); db1=x(4);
    a2=x(5); da2=x(6); b2=x(7); db2=x(8);
    //リンクからの力
    ww = [r*(a2-a1)+l*(sin(b2)-sin(b1));...
                    l*(cos(b2)-cos(b1))];
    dww = [r*(da2-da1)+l*( db2*cos(b2)-db1*cos(b1));...
                        l*(-db2*sin(b2)+db1*sin(b1))];
    w = norm(ww); //wwの長さ
    dw = ( dww(1)*ww(1)+dww(2)*ww(2) )/w; //wwの長さの時間微分
    uu = ww/w; //単位ベクトル化
    ck = 5000; cw = 100; p = -ck*(w-w0)-cw*dw; //反力の大きさ
    pp = p*uu;
endfunction
///// 振り子先端に作用する力→一般化力 /////
function Ff = sentan(x,F)
    global r l;
    b=x(3);
    Ff = [r, 0; l*cos(b), -l*sin(b)]*F; //一般化力への変換公式
endfunction
///// 倒立制御入力 /////
function u = toritu(x,a0)
    a=x(1); da=x(2); b=x(3); db=x(4);
    /// 制御の打ち切り×(位置制御 + 倒立制御) ///
    u = trap(b,0,bmax)*(15*b + 1*db); //対戦用に「倒立制御のみ」に変更
endfunction
///// 運動方程式の右辺 ////
function h = uhen(x,F)
    a=x(1); da=x(2); b=x(3); db=x(4);
    A = [ (M+m)*r^2+J, m*l*r*cos(b); ...
          m*l*r*cos(b),    m*l^2];
    bb = [m*l*r*db^2 *sin(b); m*g*l*sin(b)] + F; //Fは一般化力
    h = A\bb;
endfunction
///// 運動方程式の定義 /////
function dx = eom(t,x)
    pp = rinku(x);    //リンクからの反力
    ///// 左 /////
    a=x(1); da=x(2); b=x(3); db=x(4);
    ff=yuka([a; da; b; db]);              //床反力
    Ff=sentan([a; da; b; db], ff - pp); //振り子先端への力→一般化力
    T =toritu([a; da; b; db],0);          //倒立制御
    T = T + bang1(t);                     //バンバン入力をトルクに加算
    FT=[T; -T];                           //トルク→一般化力
    h=uhen([a; da; b; db], FT+Ff);        //加速度で整理した右辺
    dx(1) = x(2); dx(2) = h(1);           //運動方程式
    dx(3) = x(4); dx(4) = h(2);
    ///// 右 /////
    a=x(5); da=x(6); b=x(7); db=x(8);
    ff=yuka([a; da; b; db]);              //床反力
    Ff=sentan([a; da; b; db], ff + pp); //振り子先端への力→一般化力
    T =toritu([a; da; b; db],w0/r);       //倒立制御
    T = T + bang2(t);                     //バンバン入力をトルクに加算
    FT=[T; -T];                           //トルク→一般化力
    h=uhen([a; da; b; db], FT+Ff);        //加速度の右辺
    dx(5) = x(6); dx(6) = h(1);           //運動方程式
```

```
    dx(7) = x(8); dx(8) = h(2);
endfunction
///// 運動方程式を解く /////
tn=801; tt=linspace(0,0.01*tn,tn);
x01=[0; 0; 0; 0];        //左側の初期値
x02=[w0/r; 0; 0; 0];     //右側の初期値
xx=ode( [x01;x02], 0, tt, eom );
///// ロボット描画用の関数 /////
function R=Rot(a)            //回転
    R=[cos(a),-sin(a);...
       sin(a), cos(a)];
endfunction
function yy=Trans(xx,rr)     //平行移動
    yy(1,:)=xx(1,:) + rr(1)*ones(xx(1,:));
    yy(2,:)=xx(2,:) + rr(2)*ones(xx(2,:));
endfunction
function endp = draw_robot(a,b,x,sp) //ロボット描画
//    x=r*a; //車輪中心の水平変位
    sn=5; sq=linspace(0,2*%pi*(sn-1)/sn,sn);
    spoke=r*[cos(sq);sin(sq)];       //原点にあるスポークの外周点
    spoke=Rot(-a) * spoke;           //角度a回転したスポークの外周点
    spoke=Trans(spoke,[x;r]);        //位置xのスポークの外周点
    xv=[x*ones(1:sn);spoke(1,:)]; yv=[r*ones(1:sn);spoke(2,:)];
    xsegs(xv,yv,2); xarc(x-r, 2*r, 2*r, 2*r, 0, 360*64); //スポーク; 車輪
    rod=[0,0;0,l];                   //原点にある棒の両端点
    rod=Rot(-b) * rod;               //角度βだけ回転した棒の両端点
    rod=Trans(rod,[x;r]);            //位置xの棒の両端点
    plot( rod(1,:), rod(2,:),sp);       //棒の描画
    gc=gce(); gc.children.thickness=2;  //線の太さ
    endp=rod(:,2);                      //棒の先端
endfunction
function draw_pair(x)
    a1=x(1); b1=x(3); a2=x(5); b2=x(7);
    /// 重心まわりの描画 ///
    xM1=r*a1; xM2=r*a2; xm1=xM1+l*sin(b1); xm2=xM2+l*sin(b2);
    xG=(M*(xM1+xM2)+m*(xm1+xm2))/(2*M+2*m);      //重心の座標
    ep1=draw_robot(a1,b1,r*a1-xG,"r-");          //r 赤
    ep2=draw_robot(a2,b2,r*a2-xG,"b-");          //b 青
    plot([ep1(1),ep2(1)],[ep1(2),ep2(2)],"g-"); //リンクの描画
    gc=gca(); gc.isoview="on";                   //縦横比1
    gc.data_bounds=[-3,0;3,2];                   //座標軸の範囲
endfunction
drawlater; draw_pair( [x01;x02] ); drawnow;
sleep(2000); //2s待ち
realtimeinit(0.01); //アニメーションの時間刻み
for i=1:10:tn
    realtime(i);
    drawlater; clf();           //描画延期; 描画消去;
    draw_pair( xx(:,i) );       //ロボット描画
    xlabel(sprintf("%d / %d", i, tn));
    drawnow;                    //画面更新
end
///// 転倒時刻の測定 /////
i_1win=0; i_2win=0;
for i=1:tn
    b=xx(3,i); ep1=r+l*cos(b);
    b=xx(7,i); ep2=r+l*cos(b);
    if ( ep1<0 & i_2win==0 )
        i_2win=i;
    end
    if ( ep2<0 & i_1win==0 )
        i_1win=i;
    end
end
///// 転倒時刻の表示 /////
sy=1.5;
if ( i_1win==0 & i_2win==0 )             //引き分け
    xstring(-2,sy,"DRAW !");
elseif ( i_1win>0 & i_2win==0 ) //1の勝ち
    t_win=tt(i_1win);                    //勝った時刻
    xstring(-2,sy,sprintf("Red wins at  t = %.2f (i=%d)",t_win,i_1win));
elseif ( i_1win==0 & i_2win>0 ) //2の勝ち
    t_win=tt(i_2win);                    //勝った時刻
    xstring(-2,sy,sprintf("Blue wins at  t = %.2f (i=%d)",t_win,i_2win));
else
    xstring(0,sy,"DRAW (both lose)");    //共倒れ
end
gc=gce(); gc.font_size=5; drawnow;
```