宣言

```
% 定数パラメータを定義
syms m_1 I_1 l_1 L_1 real
syms m_2 I_2 l_2 L_2 real

% 時間に依存する文字を定義
syms theta_1_t(t)
syms theta_2_t(t)

% 時間に依存しない文字としての定義
syms theta_1 omega_1 alpha_1 real
syms theta_2 omega_2 alpha_2 real

% 端点を動くものとして億
syms x_1_top(t) y_1_top(t)
syms x_2_top(t) y_2_top(t)

% 各端点にかかる力を定義
syms Fx_1_top Fy_1_top real
syms Fx_2_top Fy_2_top real
```

時間依存の置き換えの準備

```
theta_1_replaced = [theta_1_t, diff(theta_1_t, t), diff(theta_1_t, t, t)];
theta_1_replacing = [theta_1, omega_1, alpha_1];

theta_2_replaced = [theta_2_t, diff(theta_2_t, t), diff(theta_2_t, t, t)];
theta_2_replacing = [theta_2, omega_2, alpha_2];

subs_t = @(input) subs( input, ...
    [ ...
        theta_1_replaced, theta_2_replaced ...
    ], ...
    [ ...
        theta_1_replacing, theta_2_replacing ...
    ] ...
    );
```

拘束条件の準備

```
subs_restriction_1_top = @(input) subs( input, [x_1_top, y_1_top], sym([0,0]) );
subs_restriction_2_top = @(input) subs( input, [x_2_top, y_2_top], [x_1_top, y_1_top] + l_1 * |
```

重心の加速度を定義する

```
pG_1 = [x_1_top, y_1_top] + L_1 * [cos(theta_1_t), sin(theta_1_t)];
pG_2 = [x_2_top, y_2_top] + L_2 * [cos(theta_2_t), sin(theta_2_t)];

aG_1 = formula( diff( pG_1, t, t ) );
aG_2 = formula( diff( pG_2, t, t ) );
```

トルクを計算する

```
torque_1_F_1_top = cross( [x_1_top, y_1_top,0] - [pG_1, 0], [Fx_1_top, Fy_1_top, 0] );
torque_1_F_2_top = cross( [x_2_top, y_2_top,0] - [pG_1, 0], [-Fx_2_top, -Fy_2_top, 0] );
torque_2_F_2_top = cross( [x_2_top, y_2_top,0] - [pG_2, 0], [Fx_2_top, Fy_2_top, 0] );

torque_1_F_1_top = formula( torque_1_F_1_top );
torque_1_F_2_top = formula( torque_1_F_2_top );
torque_2_F_2_top = formula( torque_2_F_2_top );

torque_1_F_1_top = torque_1_F_1_top(3);
torque_1_F_2_top = torque_1_F_2_top(3);
torque_2_F_2_top = torque_2_F_2_top(3);
```

方程式を立てる

```
Netwon_eq = [
    m_1 * aG_1(1) == Fx_1_top + (-Fx_2_top);
    m_1 * aG_1(2) == Fy_1_top + (-Fy_2_top);
    m_2 * aG_2(1) == Fx_2_top;
    m_2 * aG_2(2) == Fy_2_top;
    I_1 * diff( theta_1_t, t, t ) == torque_1_F_1_top + torque_1_F_2_top;
    I_2 * diff( theta_2_t, t, t ) == torque_2_F_2_top;
    ];

% 拘束を代入する
% restriction_2 が restriction_1 に依存するから、先に restriction_2 から代入する
Netwon_eq = subs_restriction_2_top( Netwon_eq );
Netwon_eq = subs_restriction_1_top( Netwon_eq );

% diff( theta_1, t ) などを omega_1 などに置き換え
% 置き換えないと equationsToMatrix が使えない
Netwon_eq = subs_t(Netwon_eq);

variables = [alpha_1, alpha_2, Fx_1_top, Fy_1_top, Fx_2_top, Fy_2_top];
[A,B] = equationsToMatrix(Netwon_eq, variables);
if det(A) == 0
    error('det(A) == 0')
end
X = inv(A) * B;

subs_X = @(input) subs(input, variables, X');
```

運動量が外力によってのみ変化しているか確認する

```
% 各セグメントの重心速度の定義
vG_1 = formula( diff( pG_1, t ) );
vG_2 = formula( diff( pG_2, t ) );

% 運動量とその変化速度を定義
momentum = m_1 * vG_1 + m_2 * vG_2;
dmomentum = diff( momentum, t );
```

```
% 拘束を代入する
% restriction_2 が restriction_1 に依存するから、先に restriction_2 から代入する
dmomentum = subs_restriction_2_top( dmomentum );
dmomentum = subs_restriction_1_top( dmomentum );

% diff( theta_1, t ) などを omega_1 などに置き換え
dmomentum = subs_t( dmomentum );

% alpha_1 などを置き換え
dmomentum = subs_X( dmomentum );

% 外力を置き換えたものを引いて、0 になるか確認
simplify( dmomentum - subs_X([Fx_1_top, Fy_1_top]) )
```

ans = $\begin{pmatrix} 0 & 0 \end{pmatrix}$