Two Bar Linkage Wheel Leg Derivations

IK for 2 Bar

```
syms x theta l1 l2 phi1 phi2
C_mat = simplify([-x*sin(theta), cos(theta); x*cos(theta), sin(theta)]^(-1));
IK_map = [l1*cos(phi1) + l2*cos(phi1-phi2), -l2*cos(phi1 - phi2); -l1*sin(phi1) +
l2*sin(phi1-phi2), l2*sin(phi1 - phi2)];
J_mat = C_mat * IK_map;
J_mat = simplify((J_mat.')^-1);

fprintf("TWO BAR INVERSE KINEMATICS MATRICES: \n")
disp(C_mat);
```

$$\begin{pmatrix} -\frac{\sin(\theta)}{x} & \frac{\cos(\theta)}{x} \\ \cos(\theta) & \sin(\theta) \end{pmatrix}$$

```
disp(IK_map);
```

$$\begin{pmatrix} l_1\cos(\phi_1) + l_2\cos(\phi_1 - \phi_2) & -l_2\cos(\phi_1 - \phi_2) \\ l_2\sin(\phi_1 - \phi_2) - l_1\sin(\phi_1) & l_2\sin(\phi_1 - \phi_2) \end{pmatrix}$$

```
disp(J_mat);
```

$$\begin{pmatrix} \frac{x\cos(\phi_1 - \phi_2 + \theta)}{\sigma_2} & \frac{x\,(l_2\cos(\phi_2 - \phi_1 + \theta) + l_1\cos(\phi_1 + \theta))}{\sigma_1} \\ \frac{\sin(\phi_1 - \phi_2 + \theta)}{\sigma_2} & \frac{l_2\sin(\phi_2 - \phi_1 + \theta) + l_1\sin(\phi_1 + \theta)}{\sigma_1} \end{pmatrix}$$

where

$$\sigma_1 = {l_2}^2\sin(2\,\phi_1 - 2\,\phi_2) - l_1\,l_2\sin(\phi_2)$$

$$\sigma_2 = l_2\sin(2\,\phi_1 - 2\,\phi_2) - l_1\sin(\phi_2)$$

```
fprintf("======================\n");
```

```
======================
```

Consider Full System Position Vector

```
syms x(t) y(t) d1 d2 l1 l2 R gamma(t) phi1(t) phi2(t) theta(t) t

e1 = [1; 0]; % cartesian frame
e2 = [0; 1]; % cartesian frame
d1_hat = e1 * cos(gamma) + e2 * sin(gamma); % robot base frame
```

```
d2_hat = e1 * -sin(gamma) + e2 * cos(gamma); % robot base frame
c1_hat = d1_hat * cos(phi1) + d2_hat * sin(phi1); % link 1 frame
c2_hat = d1_hat * -sin(phi1) + d2_hat * cos(phi1); % link 1 frame
b1_hat = c1_hat * cos(phi2) + c2_hat * sin(phi2); % link 2 frame
b2_hat = c1_hat * -sin(phi2) + c2_hat * cos(phi2); % link 2 frame
a1_hat = b1_hat * cos(theta) + b2_hat * sin(theta); % wheel frame
a2_hat = b1_hat * -sin(theta) + b2_hat * cos(theta); % wheel frame


r_OP = x * e1 + y * e2 - d1 * d1_hat - d2 * d2_hat + l1 * c1_hat + l2 * b1_hat + R
* a1_hat;
r_OP = simplify(r_OP);

fprintf("FULL SYSTEM POSITION VECTOR: \n");
disp(r_OP)
```

$$\begin{pmatrix} x(t) + R\cos(\sigma_1) - d_1\cos(\gamma(t)) + d_2\sin(\gamma(t)) + l_2\cos(\sigma_2) + l_1\cos(\sigma_3) \\ y(t) - d_2\cos(\gamma(t)) + R\sin(\sigma_1) - d_1\sin(\gamma(t)) + l_2\sin(\sigma_2) + l_1\sin(\sigma_3) \end{pmatrix}$$

where

$$\sigma_1 = \gamma(t) + \phi_1(t) + \phi_2(t) + \theta(t)$$

$$\sigma_2 = \gamma(t) + \phi_1(t) + \phi_2(t)$$

$$\sigma_3 = \gamma(t) + \phi_1(t)$$

```
fprintf("=======================\n");
```

```
=======================
```

Full System Velocity Vector

```
v_P = diff(r_OP, t);
v_P = simplify(v_P);
fprintf("FULL SYSTEM VELOCITY VECTOR: \n");
disp_eq(v_P, phi1, phi2, theta, x, y, gamma, t);
```

$$\begin{pmatrix} \dot{x} - R\sin(\sigma_1)\ \left(\dot{\gamma} + \dot{\phi}_1 + \dot{\phi}_2 + \dot{\theta}\right) + d_2\,\dot{\gamma}\cos(\gamma(t)) + d_1\,\dot{\gamma}\sin(\gamma(t)) - l_1\sin(\sigma_3)\ \left(\dot{\gamma} + \dot{\phi}_1\right) - l_2\sin(\sigma_2)\ \left(\dot{\gamma} + \dot{\phi} \right. \\ \dot{y} + R\cos(\sigma_1)\ \left(\dot{\gamma} + \dot{\phi}_1 + \dot{\phi}_2 + \dot{\theta}\right) - d_1\,\dot{\gamma}\cos(\gamma(t)) + d_2\,\dot{\gamma}\sin(\gamma(t)) + l_1\cos(\sigma_3)\ \left(\dot{\gamma} + \dot{\phi}_1\right) + l_2\cos(\sigma_2)\ \left(\dot{\gamma} + \right. \end{pmatrix}$$

where

$$\sigma_1 = \gamma(t) + \phi_1(t) + \phi_2(t) + \theta(t)$$

$$\sigma_2 = \gamma(t) + \phi_1(t) + \phi_2(t)$$

$$\sigma_3 = \gamma(t) + \phi_1(t)$$

```
fprintf("=======================\n");
```

```
=======================
```

```

```

Full System Acceleration Vector

```
a_P = diff(v_P, t);
a_P = simplify(a_P);
fprintf("FULL SYSTEM ACCELERATION VECTOR: \n");
disp_eq(a_P, phi1, phi2, theta, x, y, gamma, t);
```

$$\begin{pmatrix} \ddot{x} - R\sin(\sigma_1)\ \left(\ddot{\gamma} + \ddot{\phi}_1 + \ddot{\phi}_2 + \ddot{\theta}\right) - l_2\cos(\sigma_2)\,\sigma_5 + d_2\,\ddot{\gamma}\cos(\gamma(t)) + d_1\,\ddot{\gamma}\sin(\gamma(t)) - R\cos(\sigma_1)\,\sigma_4 - l_1\sin(\sigma_3) \\ \ddot{y} + R\cos(\sigma_1)\ \left(\ddot{\gamma} + \ddot{\phi}_1 + \ddot{\phi}_2 + \ddot{\theta}\right) - l_2\sin(\sigma_2)\,\sigma_5 - d_1\,\ddot{\gamma}\cos(\gamma(t)) + d_2\,\ddot{\gamma}\sin(\gamma(t)) + l_1\cos(\sigma_3)\ \left(\ddot{\gamma} + \ddot{\phi}_1\right) - R \end{pmatrix}$$

where

$$\sigma_1 = \gamma(t) + \phi_1(t) + \phi_2(t) + \theta(t)$$

$$\sigma_2 = \gamma(t) + \phi_1(t) + \phi_2(t)$$

$$\sigma_3 = \gamma(t) + \phi_1(t)$$

$$\sigma_4 = \left(\dot{\gamma} + \dot{\phi}_1 + \dot{\phi}_2 + \dot{\theta}\right)^2$$

$$\sigma_5 = \left(\dot{\gamma} + \dot{\phi}_1 + \dot{\phi}_2\right)^2$$

```
fprintf("=======================\n");
```

```
=======================
```

Full System Dynamics Equations

```
% define inertia and mass symbols
syms I_A I_B I_C I_D m_A m_B m_C m_D
% define force symbols
syms ax ay bx by cx cy dx dy g
% define torque symbols
syms T_A(t) T_B(t) T_C(t)

% base equations
c_eq1 = m_C * diff(x, t, 2) == cx;
c_eq2 = m_C * diff(y, t, 2) == cy - m_D * g;
c_eq3 = I_D * diff(phi1, t, 2) == m_D * g * cos(gamma) * sqrt(d1^2 + d2^2) - T_C;

% leg 1 equations
```

Helper Functions

```
function equation_s = disp_eq(equation, phi1, phi2, theta, x, y, gamma, t)
    % replace second derivatives
    equation_s = subs(equation, diff(phi1, t, 2), sym('phi_ddot_1'));
    equation_s = subs(equation_s, diff(phi2, t, 2), sym('phi_ddot_2'));
    equation_s = subs(equation_s, diff(theta, t, 2), sym('theta_ddot'));
    equation_s = subs(equation_s, diff(x, t, 2), sym('x_ddot'));
    equation_s = subs(equation_s, diff(y, t, 2), sym('y_ddot'));
    equation_s = subs(equation_s, diff(gamma, t, 2), sym('gamma_ddot'));

    % replace first derivatives
    equation_s = subs(equation_s, diff(phi1, t), sym('phi_dot_1'));
    equation_s = subs(equation_s, diff(phi2, t), sym('phi_dot_2'));
    equation_s = subs(equation_s, diff(theta, t), sym('theta_dot'));
    equation_s = subs(equation_s, diff(x, t), sym('x_dot'));
    equation_s = subs(equation_s, diff(y, t), sym('y_dot'));
    equation_s = subs(equation_s, diff(gamma, t), sym('gamma_dot'));

    % display equation
    disp(equation_s)
end
```