Position Analysis

The planar R-RTR-RTR mechanism considered is shown in Fig. 1. The driver link is the rigid link 1 (the link \(AB\)). The following numerical data are given: \(AB = 0.140\) m, \(AC = 0.060\) m, \(AE = 0.250\) m, \(CD = 0.150\) m. The angle of the driver link 1 with the horizontal axis is \(\phi = 30^\circ\).

**Position analysis for an input angle**

*Position of joint A*

A Cartesian reference frame \(xOy\) is selected. The joint \(A\) is in the origin of the reference frame, that is, \(A \equiv O\),

\[
x_A = 0, \quad y_A = 0.
\] (1)

*Position of joint C*

The coordinates of the joint \(C\) are

\[
x_C = 0, \quad y_C = AC = 0.060\) m.
\] (2)

*Position of joint E*

The coordinates of the joint \(E\) are

\[
x_E = 0, \quad y_E = -AE = -0.250\) m.
\] (3)

*Position of joint B*

The unknowns are the coordinates of the joint \(B\), \(x_B\) and \(y_B\). Because the joint \(A\) is fixed and the angle \(\phi\) is known, the coordinates of the joint \(B\) are computed from the following expressions

\[
x_B = AB \cos \phi = 0.140 \cos 30^\circ = 0.121\) m,
\]
\[
y_B = AB \sin \phi = 0.140 \sin 30^\circ = 0.070\) m.
\] (4)

*Position of joint D*

The unknowns are the coordinates of the joint \(D\), \(x_D\) and \(y_D\). The joint \(D\) is located on the line \(BC\):

\[
\frac{y_D - y_C}{x_D - x_C} = \frac{y_B - y_C}{x_B - x_C} \quad \text{or} \quad (x_B - x_C)(y_D - y_C) = (x_D - x_C)(y_B - y_C)
\] (5)
Furthermore, the length of the segment $CD$ is constant:

$$(x_C - x_D)^2 + (y_C - y_D)^2 = CD^2. \quad (6)$$

The Eqs. (5) and (6) form a system from which the coordinates of the joint $D$ can be computed. To solve the system of equations, a specific MATLAB/Mathematica™ command will be used. Two sets of solutions are found for the position of the joint $D$. These solutions are located at the intersection of the line $BC$ with the circle centered in $C$ and radius $CD$ (Fig. 2), and they have the following numerical values:

$$x_{D1} = -0.149 \text{ m}, \quad y_{D1} = 0.047 \text{ m},$$
$$x_{D2} = 0.149 \text{ m}, \quad y_{D2} = 0.072 \text{ m}.$$

To determine the correct position of the joint $D$ for the mechanism, an additional condition is needed.

For the first quadrant, $0 \leq \phi \leq 90^\circ$, the condition is $x_D \leq x_C$.

Because $x_C = 0 \text{ m}$, the coordinates of the joint $D$ are

$$x_D = x_{D1} = -0.149 \text{ m},$$
$$y_D = y_{D1} = 0.047 \text{ m}.$$

**Angle $\phi_2$**

The angle of link 2 (or link 3) with the horizontal axis is calculated from the slope of the straight line $BC$:

$$\phi_2 = \phi_3 = \arctan \frac{y_B - y_C}{x_B - x_C}.$$

**Angle $\phi_4$**

The angle of link 5 (or link 4) with the horizontal axis is obtained from the slope of the straight line $ED$:

$$\phi_4 = \phi_5 = \arctan \frac{y_E - y_D}{x_E - x_D}.$$

The MATLAB/Mathematica™ program for the input angle $\phi = 30^\circ$ is given in Program 1.
Position analysis for a complete rotation

For a complete rotation of the driver link $AB$, $0 \leq \phi \leq 360^\circ$, a step angle of $\phi = 60^\circ$ is selected.

**Method I**

Method I uses constraint conditions for the mechanism for each quadrant. For the mechanism, there are several conditions for the position of the joint $D$.

For the angle $\phi$ located in the first quadrant $0^\circ \leq \phi \leq 90^\circ$ (Fig. 2), and the fourth quadrant $270^\circ \leq \phi \leq 360^\circ$ (Fig. 5), the following relation exists between $x_D$ and $x_C$:

$$x_D \leq x_C.$$

For the angle $\phi$ located in the second quadrant $90^\circ < \phi \leq 180^\circ$ (Fig. 3), and the third quadrant $180^\circ < \phi < 270^\circ$ (Fig. 4), the following relation exists between $x_D$ and $x_C$:

$$x_D \geq x_C.$$

The MATLAB/Mathematica™ program for a complete rotation of the driver link using method I is given in Program 2. The graph of the mechanism for a complete rotation of the driver link is given in Fig. 6.

**Method II**

Another position analysis method for a complete rotation of the driver link uses constraint conditions for the initial value of the angle $\phi$. For the mechanism, the correct position of the joint $D$ is calculated using a simple function, the Euclidian distance between two points $P$ and $Q$:

$$d = \sqrt{(x_P - x_Q)^2 + (y_P - y_Q)^2}. \quad (7)$$

For the initial angle $\phi = 0^\circ$, the constraint is $x_D \leq x_C$, so the first position of the joint $D$, that is, $D_0$, is calculated for the first step $D = D_0 = D_k$, $k = 0$. For the next position of the joint, $D_{k+1}$, there are two solutions $D^I_{k+1}$ and $D^II_{k+1}$, $k = 0, 1, 2, \ldots$. In order to choose the correct solution of the joint, $D_{k+1}$, it is compared the distances between the old position, $D_k$, and each new calculated positions $D^I_{k+1}$ and $D^II_{k+1}$. The distances between the known solution $D_k$ and the new solutions $D^I_{k+1}$ and $D^II_{k+1}$ are $d^I_k$ and $d^II_k$. If the distance to the first solution is less than the distance to the second solution,
$d_k^I < d_k^II$, then the correct answer is $D_{k+1} = D_{k+1}^I$, or else $D_{k+1} = D_{k+1}^II$ (Fig. 7). With this algorithm the correct solution is selected using just one constraint relation for the initial step and then, automatically, the problem is solved. In this way it is not necessary to have different constraints for different quadrants.

The MATLAB/Mathematica$^{TM}$ program for a complete rotation of the driver link using the second method is given in Program 3.
Circle of radius $CD$

$D = D_1$

$A(0,0)$

$x_D = x_{D1}$

$x_D = x_{D2}$

Fig. 2
Fig. 6
Circle of radius $CD$

$D_k$

$D_{k+1} = D_{k+1}^l$

$C$

$D_{k+1}^H$

$A(0,0)$

$d_k$

$d_k^H$

Fig. 7
clear; clc; close all

% Input data
AB = 0.14; AC = 0.06; AE = 0.25; CD = 0.15; % (m)
phi = pi/6; % (rad)

% Select the dimensions
DF = 0.4; EG = 0.5; % (m)

xA = 0; yA = 0; rA = [xA yA 0]; % Position of A
xC = 0; yC = AC; rC = [xC yC 0]; % Position of C
xE = 0; yE = -AE; rE = [xE yE 0]; % Position of E
xB = AB*cos(phi); yB = AB*sin(phi); rB = [xB yB 0]; % Position of B

% Position of joint D
% Distance formula: CD=constant
eqnD1 = '(xDsol - xC)^2 + (yDsol - yC)^2 = CD^2';
% Slope formula: B, C, and D are on the same straight line
eqnD2 = '(yB - yC) / (xB - xC) = (yDsol - yC) / (xDsol - xC)';
% Simultaneously solve above equations
solD = solve(eqnD1, eqnD2, 'xDsol, yDsol');
% solve symbolic solution of algebraic equations
% Two solutions for xD - vector form
xDpositions = eval(solD.xDsol);
% eval execute string as an expression or statement
% Two solutions for yD - vector form
yDpositions = eval(solD.yDsol);
% Separate the solutions in scalar form
xD1 = xDpositions(1); % first component of the vector xDpositions
xD2 = xDpositions(2); % second component of the vector xDpositions
yD1 = yDpositions(1); % first component of the vector yDpositions
yD2 = yDpositions(2); % second component of the vector yDpositions

% Select the correct position for D for the given input angle
if xD1 <= xC
    xD = xD1; yD = yD1;
else
    xD = xD2; yD = yD2;
end
rD = [xD yD 0]; % Position of D

% Angles of the links with the horizontal
phi2 = atan((yB-yC)/(xB-xC));
phi3 = phi2;
phi4 = atan((yD-yE)/(xD-xE))+pi;
phi5 = phi4;

% Positions of the points F and G
xF = xD + DF*cos(phi3);
yF = yD + DF*sin(phi3);
rF = [xF yF 0]; % Position vector of F

xG = xE + EG*cos(phi5);
yG = yE + EG*sin(phi5);
rG = [xG yG 0]; % Position vector of G
fprintf('Results
'); fprintf('
');

fprintf('rA = [ %g, %g, %g] (m)
', rA);
fprintf('rC = [ %g, %g, %g] (m)
', rC);
fprintf('rE = [ %g, %g, %g] (m)
', rE);
fprintf('rB = [ %g, %g, %g] (m)
', rB);
fprintf('rD = [ %g, %g, %g] (m)
', rD);
fprintf('phi2 = phi3 = %g (degrees)
', phi2*180/pi);
fprintf('phi4 = phi5 = %g (degrees)
', phi4*180/pi);
fprintf('rF = [ %g, %g, %g] (m)
', rF);
fprintf('rG = [ %g, %g, %g] (m)
', rG);

% Graphic of the mechanism
plot([xA,xB],[yA,yB],’r-o’,’LineWidth’,1.5);
hold on
plot([xD,xC],[yD,yC],’b-o’,’LineWidth’,1.5);
hold on
plot([xC,xB],[yC,yB],’b-o’,’LineWidth’,1.5);
hold on
plot([xB,xF],[yB,yF],’b-o’,’LineWidth’,1.5);
hold on
plot([xE,xD],[yE,yD],’g-o’,’LineWidth’,1.5);
hold on
plot([xD,xG],[yD,yG],’g-o’,’LineWidth’,1.5);
grid on,

xlabel('x (m)'), ylabel('y (m)'),...
title('positions for \phi = 30 (deg)'),...
text(xA,yA,’\leftarrow A = ground’,’HorizontalAlignment’,’left’),...
text(xB,yB,’B’),...
text(xC,yC,’\leftarrow C = ground’,’HorizontalAlignment’,’left’),...
text(xD,yD,’D’),...
text(xE,yE,’\leftarrow E = ground’,’HorizontalAlignment’,’left’),...
text(xF,yF,’F’), text(xG,yG,’G’)

Results

\[ r_A = [0, 0, 0] \text{ (m)} \]
\[ r_C = [0, 0.06, 0] \text{ (m)} \]
\[ r_E = [0, -0.25, 0] \text{ (m)} \]
\[ r_B = [0.121244, 0.07, 0] \text{ (m)} \]
\[ r_D = [-0.149492, 0.0476701, 0] \text{ (m)} \]
\[ \phi_2 = \phi_3 = 4.715 \text{ (degrees)} \]
\[ \phi_4 = \phi_5 = 116.666 \text{ (degrees)} \]
\[ r_F = [0.249154, 0.0805999, 0] \text{ (m)} \]
\[ r_G = [-0.224396, 0.196818, 0] \text{ (m)} \]
Positions for φ = 30 (deg)

A = ground
B
C = ground
D
E = ground
F

\(x (\text{m})\)

\(y (\text{m})\)
% Program 2
% R–RTR–RTR
% Position analysis – complete rotation
clear; clc; close all

% Input data
AB=0.14; AC=0.06; AE=0.25; CD=0.15; (%m)
xA = 0; yA = 0; rA = [xA yA 0]; % Position vector of A
xC = 0 ; yC = AC ; rC = [xC yC 0]; % Position vector of C
xE = 0 ; yE = −AE ; rE = [xE yE 0]; % Position vector of E

fprintf('Results 
'); fprintf('
');
fprintf('rA =  [ %g, %g, %g] (m)
', rA);
fprintf('rC =  [ %g, %g, %g] (m)
', rC);
fprintf('rE =  [ %g, %g, %g] (m)
', rE);

% complete rotation phi=0 to 2*pi step pi/3
for phi=0:pi/3:2*pi,

fprintf('phi =  %g deegres 
', phi*180/pi);

% Position of joint B – position of the driver link
xB = AB*cos(phi); yB = AB*sin(phi); rB = [xB yB 0];
fprintf('rB =  [ %g, %g, %g] (m)
', rB);

% Position of joint D
eqnD1 = ‘’( xDsol − xC )^2 + ( yDsol − yC )^2 = CD^2 ‘’;
% Slope formula: B, C, and D are on the same straight line
eqnD2 = ‘’( yB − yC ) / ( xB − xC ) = ( yDsol − yC ) / ( xDsol − xC )’’;
% Simultaneously solve above equations
solD = solve(eqnD1, eqnD2, ’xDsol, yDsol’);
xDpositions = eval(solD.xDsol);
yDpositions = eval(solD.yDsol);
% Separate the solutions in scalar form
xD1 = xDpositions(1); xD2 = xDpositions(2);
yD1 = yDpositions(1); yD2 = yDpositions(2);

% Select the correct position for D for the angle phi
% see the drawings for each quadrant
if (phi>=0 && phi<=pi/2)||(phi >= 3*pi/2 && phi<=2*pi)
    if xD1 <= xC xD = xD1; yD=yD1; else xD = xD2; yD=yD2; end
else
    if xD1 >= xC xD = xD1; yD=yD1; else xD = xD2; yD=yD2; end
end
% &&  short-circuit logical AND
% ||  short-circuit logical OR
rD = [xD yD 0];
fprintf('rD =  [ %g, %g, %g] (m)
', rD);

% Angles of the links with the horizontal
phi2 = atan((yB−yC)/(xB−xC));
phi3 = phi2;
fprintf('phi2 = phi3 = %g (degrees) 
', phi2*180/pi);
phi4 = atan((yD−yE)/(xD−xE));
phi5 = phi4;
fprintf('phi4 = phi5 = %g (degrees) 
', phi4*180/pi);
fprintf('
');
% Graphic of the mechanism

plot([xA,xB],[yA,yB],’r’-o’, [xB,xC],[yB,yC],’b’-o’, [xC,xD],[yC,yD],’b’-o’)
hold on
plot([xD,xE],[yD,yE],’g’-o’)
xlabel(‘x (m)’),...
ylabel(‘y (m)’),...
title(‘positions for \phi = 0 to 360 step 60 (deg)’),...
text(xA,yA,’ A’),...
text(xB,yB,’ B’),....
text(xC,yC,’ C’),....
text(xD,yD,’ D’),....
text(xE,yE,’ E’)

end % end for
Results

\[ \begin{align*}
\mathbf{r_A} & = [0, 0, 0] \text{ (m)} \\
\mathbf{r_C} & = [0, 0.06, 0] \text{ (m)} \\
\mathbf{r_E} & = [0, -0.25, 0] \text{ (m)} \\
\phi & = 0 \text{ degrees} \\
\mathbf{r_B} & = [0.14, 0, 0] \text{ (m)} \\
\mathbf{r_D} & = [-0.137872, 0.119088, 0] \text{ (m)} \\
\phi_2 & = \phi_3 = -23.1986 \text{ (degrees)} \\
\phi_4 & = \phi_5 = -69.517 \text{ (degrees)} \\
\phi & = 60 \text{ degrees} \\
\mathbf{r_B} & = [0.07, 0.121244, 0] \text{ (m)} \\
\mathbf{r_D} & = [-0.112892, -0.0387698, 0] \text{ (m)} \\
\phi_2 & = \phi_3 = 41.1829 \text{ (degrees)} \\
\phi_4 & = \phi_5 = 61.8778 \text{ (degrees)} \\
\phi & = 120 \text{ degrees} \\
\mathbf{r_B} & = [-0.07, 0.121244, 0] \text{ (m)} \\
\mathbf{r_D} & = [0.112892, -0.0387698, 0] \text{ (m)} \\
\phi_2 & = \phi_3 = -41.1829 \text{ (degrees)} \\
\phi_4 & = \phi_5 = 61.8778 \text{ (degrees)} \\
\phi & = 180 \text{ degrees} \\
\mathbf{r_B} & = [-0.14, 1.71451e-17, 0] \text{ (m)} \\
\mathbf{r_D} & = [0.137872, 0.119088, 0] \text{ (m)} \\
\phi_2 & = \phi_3 = 23.1986 \text{ (degrees)} \\
\phi_4 & = \phi_5 = 69.517 \text{ (degrees)} \\
\phi & = 240 \text{ degrees} \\
\mathbf{r_B} & = [-0.07, -0.121244, 0] \text{ (m)} \\
\mathbf{r_D} & = [0.0540425, 0.199926, 0] \text{ (m)} \\
\phi_2 & = \phi_3 = 68.8824 \text{ (degrees)} \\
\phi_4 & = \phi_5 = 83.1508 \text{ (degrees)} \\
\phi & = 300 \text{ degrees} \\
\mathbf{r_B} & = [0.07, -0.121244, 0] \text{ (m)} \\
\mathbf{r_D} & = [-0.0540425, 0.199926, 0] \text{ (m)} \\
\phi_2 & = \phi_3 = -68.8824 \text{ (degrees)} \\
\phi_4 & = \phi_5 = -83.1508 \text{ (degrees)} \\
\phi & = 360 \text{ degrees} \\
\mathbf{r_B} & = [0.14, -3.42901e-17, 0] \text{ (m)} \\
\mathbf{r_D} & = [-0.137872, 0.119088, 0] \text{ (m)} \\
\phi_2 & = \phi_3 = -23.1986 \text{ (degrees)} \\
\phi_4 & = \phi_5 = -69.517 \text{ (degrees)} \\
>>
positions for $\phi = 0$ to 360 step 60 (deg)
% Program 3
% R−RTR−RTR
% Position analysis − complete rotation
% Euclidian distance function
% the program use the function: Dist(x1,y1,x2,y2)
% the function is defined in the program Dist.m

clear; clc; close all

% Input data
AB=0.14; AC=0.06; AE=0.25; CD=0.15; %(m)
xA = 0; yA = 0; rA = [xA yA 0]; % Position vector of A
xC = 0 ; yC = AC ; rC = [xC yC 0]; % Position vector of C
xE = 0 ; yE = −AE ; rE = [xE yE 0]; % Position vector of E

fprintf('Results \n'); fprintf('\n');
fprintf('rA =  [ %g, %g, %g] (m)\n', rA);
fprintf('rC =  [ %g, %g, %g] (m)\n', rC);
fprintf('rE =  [ %g, %g, %g] (m)\n', rE);

increment = 0 ; % at the initial moment phi=0 => increment = 0
step=pi/6; % the step has to be small for this method
for phi=0:step:2*pi,
    fprintf('phi =  %g degrees \n', phi*180/pi);
        % Position of joint B
    xB = AB*cos(phi); yB = AB*sin(phi); rB = [xB yB 0];
    fprintf('rB =  [ %g, %g, %g] (m)\n', rB);
        % Position of joint D
eqnD1 = ‘( xDsol − xC )^2 + ( yDsol − yC )^2 = CD^2 ‘;
eqnD2 = ‘( yB − yC ) / ( xB − xC ) = ( yDsol − yC ) / ( xDsol − xC )’;
solD = solve(eqnD1, eqnD2, ‘xDsol, yDsol’);
xDpositions = eval(solD.xDsol);
yDpositions = eval(solD.yDsol);
        % Separate the solutions in scalar form
    xD1 = xDpositions(1); xD2 = xDpositions(2);
yD1 = yDpositions(1); yD2 = yDpositions(2);

    % select the correct position for D only for increment == 0
    % the selection process is automatic for all the other steps
    if increment == 0
        if xD1 <= xC xD = xD1; yD=yD1; else xD = xD2; yD=yD2; end
    else
        dist1 = Dist(xD1,yD1,xDold,yDold);
        dist2 = Dist(xD2,yD2,xDold,yDold);
        if dist1 < dist2 xD = xD1; yD=yD1; else xD = xD2; yD=yD2; end
    end
    xDold=xD;
yDold=yD;

    increment=increment+1;
    rD = [xD yD 0];
    fprintf('rD =  [ %g, %g, %g] (m)\n', rD);

    phi2 = atan((yB−yC)/(xB−xC)); phi3 = phi2;
    fprintf('phi2 = phi3 = %g (degrees) \n’, phi2*180/pi);
    phi4 = atan((yD−yE)/(xD−xE)); phi5 = phi4;
    fprintf('phi4 = phi5 = %g (degrees) \n’, phi4*180/pi);
end
% Graphic of the mechanism

plot([xA,xB],[yA,yB],’r-o’,[xB,xC],[yB,yC],’b-o’,[xC,xD],[yC,yD],’b-o’) hold on plot([xD,xE],[yD,yE],’g-o’) xlabel(’x (m)’),...
ylabel(’y (m)’),....
title(’positions for \phi = 0 to 360 step 30 (deg)’),...
text(xA,yA,’ A’),....
text(xB,yB,’ B’),....
text(xC,yC,’ C’),....
text(xD,yD,’ D’),....
text(xE,yE,’ E’) end
function d=Dist(xP,yP,xQ,yQ);
d=sqrt((xP−xQ)^2+(yP−yQ)^2);
end
Results

\( \mathbf{r}_A = [0, 0, 0] \) (m)
\( \mathbf{r}_C = [0, 0.06, 0] \) (m)
\( \mathbf{r}_E = [0, -0.25, 0] \) (m)

\( \phi = 0 \) degrees
\( \mathbf{r}_B = [0.14, 0, 0] \) (m)
\( \mathbf{r}_D = [-0.137872, 0.119088, 0] \) (m)
\( \phi_2 = \phi_3 = -23.1986 \) (degrees)
\( \phi_4 = \phi_5 = -69.517 \) (degrees)

\( \phi = 30 \) degrees
\( \mathbf{r}_B = [0.121244, 0.07, 0] \) (m)
\( \mathbf{r}_D = [-0.149492, 0.0476701, 0] \) (m)
\( \phi_2 = \phi_3 = 4.715 \) (degrees)
\( \phi_4 = \phi_5 = -63.3338 \) (degrees)

\( \phi = 60 \) degrees
\( \mathbf{r}_B = [0.07, 0.121244, 0] \) (m)
\( \mathbf{r}_D = [-0.112892, -0.0387698, 0] \) (m)
\( \phi_2 = \phi_3 = 41.1829 \) (degrees)
\( \phi_4 = \phi_5 = -61.8778 \) (degrees)

\( \phi = 90 \) degrees
\( \mathbf{r}_B = [8.57253e-18, 0.14, 0] \) (m)
\( \mathbf{r}_D = [-1.60735e-17, -0.09, 0] \) (m)
\( \phi_2 = \phi_3 = 90 \) (degrees)
\( \phi_4 = \phi_5 = -90 \) (degrees)

\( \phi = 120 \) degrees
\( \mathbf{r}_B = [-0.07, 0.121244, 0] \) (m)
\( \mathbf{r}_D = [0.112892, -0.0387698, 0] \) (m)
\( \phi_2 = \phi_3 = -41.1829 \) (degrees)
\( \phi_4 = \phi_5 = 61.8778 \) (degrees)

\( \phi = 150 \) degrees
\( \mathbf{r}_B = [-0.121244, 0.07, 0] \) (m)
\( \mathbf{r}_D = [0.149492, 0.0476701, 0] \) (m)
\( \phi_2 = \phi_3 = -4.715 \) (degrees)
\( \phi_4 = \phi_5 = 63.3338 \) (degrees)

\( \phi = 180 \) degrees
\( \mathbf{r}_B = [-0.14, 1.71451e-17, 0] \) (m)
\( \mathbf{r}_D = [0.137872, 0.119088, 0] \) (m)
\( \phi_2 = \phi_3 = 23.1986 \) (degrees)
\( \phi_4 = \phi_5 = 69.517 \) (degrees)

\( \phi = 210 \) degrees
\( \mathbf{r}_B = [-0.121244, -0.07, 0] \) (m)
\( \mathbf{r}_D = [0.102307, 0.169696, 0] \) (m)
\( \phi_2 = \phi_3 = 46.9961 \) (degrees)
\( \phi_4 = \phi_5 = 76.3005 \) (degrees)

\( \phi = 240 \) degrees
\( \mathbf{r}_B = [-0.07, -0.121244, 0] \) (m)
\( \mathbf{r}_D = [0.0540425, 0.199926, 0] \) (m)
\( \phi_2 = \phi_3 = 68.8824 \) (degrees)
\( \phi_4 = \phi_5 = 83.1508 \) (degrees)
\text{phi} = 270 \text{ degrees}
\text{rB} = [ -2.57176e^{-17}, -0.14, 0] \text{ (m)}
\text{rD} = [ 1.92882e^{-17}, 0.21, 0] \text{ (m)}
\phi_2 = \phi_3 = 90 \text{ (degrees)}
\phi_4 = \phi_5 = 90 \text{ (degrees)}

\text{phi} = 300 \text{ degrees}
\text{rB} = [ 0.07, -0.121244, 0] \text{ (m)}
\text{rD} = [ -0.0540425, 0.199926, 0] \text{ (m)}
\phi_2 = \phi_3 = -68.8824 \text{ (degrees)}
\phi_4 = \phi_5 = -83.1508 \text{ (degrees)}

\text{phi} = 330 \text{ degrees}
\text{rB} = [ 0.121244, -0.07, 0] \text{ (m)}
\text{rD} = [ -0.102307, 0.169696, 0] \text{ (m)}
\phi_2 = \phi_3 = -46.9961 \text{ (degrees)}
\phi_4 = \phi_5 = -76.3005 \text{ (degrees)}

\text{phi} = 360 \text{ degrees}
\text{rB} = [ 0.14, -3.42901e^{-17}, 0] \text{ (m)}
\text{rD} = [ -0.137872, 0.119088, 0] \text{ (m)}
\phi_2 = \phi_3 = -23.1986 \text{ (degrees)}
\phi_4 = \phi_5 = -69.517 \text{ (degrees)}

>>
positions for $\phi = 0$ to 360 step 30 (deg)
(* POSITION ANALYSIS - input angle phi *)

Apply[Clear,Names["Global`*"]];
Off[General::spell];
Off[General::spell1];

(* Input data *)
AB = 0.14 ;
AC = 0.06 ;
AE = 0.25 ;
CD = 0.15 ;

(* Input angle *)
phi = N[Pi]/6 ;

(* Position of joint A *)
xA = yA = 0;

(* Position of joint C *)
xC = 0 ;
yC = AC ;

(* Position of joint E *)
xE = 0 ;
yE = -AE ;

(* Position of joint B *)
xB = AB Cos[phi] ;
yB = AB Sin[phi] ;

(* Position of joint D *)
eqnD1 = ( xDsol - xC )^2 + ( yDsol - yC )^2 - CD^2 == 0 ;
eqnD2 = ( yB - yC ) / ( xB - xC ) == ( yDsol - yC ) / ( xDsol - xC ) ;
solutionD = Solve [ { eqnD1 , eqnD2 } , { xDsol , yDsol } ] ;

(* Two solutions for D *)
xD1 = xDsol /. solutionD[[1]] ;
yD1 = yDsol /. solutionD[[1]] ;
xD2 = xDsol /. solutionD[[2]] ;
yD2 = yDsol /. solutionD[[2]] ;

(* Select the correct position for D *)
If [ xD1 <= xC , xD = xD1 ; yD = yD1 , xD = xD2 ; yD=yD2 ] ;

(* Print the solutions for B and D *)
Print["xB = ",xB," m"];
Print["yB = ",yB," m"];
Print["xD = ",xD," m"];
Print["yD = ",yD," m"];

(*********)
(*Link 2*)
(*********)
phi2=ArcTan[(yB-yC)/(xB-xC)];
"\phi2 = \phi3 = \text{ArcTan}[(yB-yC)/(xB-xC)]"
Print["\phi2 = ",\phi2," rad = ",\phi2+180/N[Pi]," deg "];

(*********)
(*Link 4*)
(*********)
phi4=ArcTan[(yD-yE)/(xD-xE)];
"\phi4 = \phi5 = \text{ArcTan}[(yD-yE)/(xD-xE)]"
Print["\phi4 = ",\phi4," rad = ",\phi4+180/N[Pi]," deg "];
(* Graph of the mechanism *)
markers = Table [
  Point [ { xA , yA } ] ,
  Point [ { xB , yB } ] ,
  Point [ { xC , yC } ] ,
  Point [ { xD , yD } ] ,
  Point [ { xE , yE } ]
] ;
name = Table [
  Text [ "A" ,{0  , 0},{-1 , 1} ] ,
  Text [ "B" ,{xB , yB} ,{ 0  , -1} ] ,
  Text [ "C" ,{xC , yC} ,{ -1,-1} ] ,
  Text [ "D" ,{xD , yD} ,{ 0  , -1} ] ,
  Text [ "E" ,{xE , yE} ,{ -1, 1} ]
] ;
graph = Graphics [
  { { RGBColor [ 1 , 0 , 0 ] ,
    Line [ { {xA,yA},{xB,yB} } ] } ,
  { RGBColor [ 0 , 1 , 0 ] ,
    Line [ { {xB,yB},{xD,yD} } ] } ,
  { RGBColor [ 0 , 0 , 1 ] ,
    Line [ { {xD,yD},{xE,yE} } ] } ,
  { RGBColor [ 1 , 1 , 1 ] ,
    PointSize [ 0.01 ] , markers } ,
  { name } ] ] ;
Show [ Graphics [ graph ] ,
  PlotRange -> { { -.25 , .25 } ,
  { -.3 , .25 } } ,
  Frame -> True,
  AxesOrigin -> {xA,yA} ,
  FrameLabel -> {"x","y"} ,
  Axes -> {True,True} ,
  AspectRatio -> Automatic ] ;
xB = 0.121244 m
yB = 0.07 m
xD = -0.149492 m
yD = 0.0476701 m
φ2 = φ3 = ArcTan[(yB-yC)/(xB-xC)]
φ2 = 0.0822923 rad = 4.715 deg
φ4 = φ5 = ArcTan[(yD-yE)/(xD-xE)]
φ4 = -1.10538 rad = -63.3338 deg
(POSITION ANALYSIS - Complete rotation (Method I))

Apply[Clear,Names["Global"],]
Off[General::spell];
Off[General::spell1];

(* Input data *)
AB = 0.14 ;
AC = 0.06 ;
AE = 0.25 ;
CD = 0.15 ;

(* Position of joint A *)
xA = yA = 0;

(* Position of joint C *)
xC = 0 ;
yC = AC ;

(* Position of joint E *)
xE = 0 ;
yE = -AE ;
increment = 0 ;

For [ phi = 0, phi <= 2*N[Pi] , phi += N[Pi]/3 ,

(* Position of joint B *)
xB = AB Cos[phi] ;
yB = AB Sin[phi] ;

(* Position of joint D *)
eqnD1 = ( xDsol - xC )^2 + ( yDsol - yC )^2 - CD^2 == 0 ;
eqnD2 = ( yB - yC ) / ( xB - xC ) == ( yDsol - yC ) / ( xDsol - xC ) ;
solutionD = Solve [ { eqnD1 , eqnD2 } , { xDsol , yDsol } ] ;

(* Two solutions for D *)
xD1 = xDsol /. solutionD[[1]] ;
yD1 = yDsol /. solutionD[[1]] ;
xD2 = xDsol /. solutionD[[2]] ;
yD2 = yDsol /. solutionD[[2]] ;

(* Select the correct position for D *)
If [ 0 <= phi <= N[Pi]/2 || 3 N[Pi]/2 <= phi <= 2 N[Pi] ,
   If [ xD1 <= xC , xD = xD1 ; yD = yD1 , xD = xD2 ; yD = yD2] ,
   If [ xD1 >= xC , xD = xD1 ; yD = yD1 , xD = xD2 ; yD = yD2] ] ;

(* Print phi and the solutions for B and D *)
Print["phi = ",phi," rad = ",phi 180/N[Pi]," deg"] ;
Print["xB = ",xB," m"] ;
Print["yB = ",yB," m"] ;
Print["xD = ",xD," m"] ;
Print["yD = ",yD," m"] ;

(* Graph of the mecanism *)
markers = Table [ { Point [ { xA , yA } ] ,
   Point [ { xB , yB } ] ,
   Point [ { xC , yC } ] ,
   Point [ { xD , yD } ] ,
   Point [ { xE , yE } ] } ] ;
name = Table [ {
   Text [ "A",{0,0},{-1,1} ],
   Text [ "B",{xB,yB},{0,-1} ],
   Text [ "C",{xC,yC},{-1,-1} ],
   Text [ "D",{xD,yD},{0,-1} ],
   Text [ "E",{xE,yE},{-1,1} ]
} ];

graph [ increment ] = Graphics [ 
   { RGBColor [ 1,0,0 ],
     Line [ {xA,yA},{xB,yB} ] },
   { RGBColor [ 0,1,0 ],
     Line [ {xB,yB},{xD,yD} ] },
   { RGBColor [ 0,0,1 ],
     Line [ {xD,yD},{xE,yE} ] },
   { RGBColor [ 1,1,1 ],
     PointSize [ 0.01 ], markers },
   { name } ] ;

Show [ Graphics [ graph [ increment ] ] ,
   PlotRange -> { {-0.25,.25} ,
                  { -0.3,.25} } ,
   Frame -> True,
   AxesOrigin -> {xA,yA},
   FrameLabel -> {"x","y"},
   Axes -> {True,True},
   AspectRatio -> Automatic ] ;

increment++;
Program 2

Diagram 1

Diagram 2
(* All the positions on the same graphic *)

Show [Table[graph [i] , { i , increment-1 } ] ,
      PlotRange -> { { -.25 , .25 } ,
                     { -.3 , .25 } } ,
      Frame -> True,
      AxesOrigin -> {xA,yA},
      FrameLabel -> {"x","y"},
      Axes -> {True,True},
      AspectRatio -> Automatic ];
(* POSITION ANALYSIS - Complete rotation (Method II) *)

Apply[Clear, Names["Global"]];
Off[General::spell1];
Off[General::spell];

(* Euclidian distance function *)
Dist[xP_, yP_, xQ_, yQ_] := Sqrt[(xP - xQ)^2 + (yP - yQ)^2];

(* Input data *)
AB = 0.14;
AC = 0.06;
AE = 0.25;
CD = 0.15;

(* Position of joint A *)
xA = yA = 0;

(* Position of joint C *)
xC = 0;
yC = AC;

(* Position of joint E *)
xE = 0;
yE = -AE;
increment = 0;

For [phi = 0, phi <= 2*N[Pi], phi += N[Pi]/6, 

(* Position of joint B *)
xB = AB Cos[phi];
yB = AB Sin[phi];

(* Position of joint D *)
eqnD1 = (xDsol - xC)^2 + (yDsol - yC)^2 - CD^2 == 0;
eqnD2 = (yB - yC)/(xB - xC) == (yDsol - yC)/(xDsol - xC);
solutionD = Solve [{eqnD1, eqnD2}, {xDsol, yDsol}];

(* Two solutions for D *)
xD1 = xDsol /. solutionD[[1]]; yD1 = yDsol /. solutionD[[1]]; xD2 = xDsol /. solutionD[[2]]; yD2 = yDsol /. solutionD[[2]]; 

(* Select the correct position for D *)
If[increment == 0, If[xD1 < xC, xD = xD1; yD = yD1, xD = xD2; yD = yD2], dist1 = Dist[xD1, yD1, xDold, yDold]; dist2 = Dist[xD2, yD2, xDold, yDold];
If[dist1 < dist2, xD = xD1; yD = yD1, xD = xD2; yD = yD2];
xDold = xD;
yDold = yD;
increment++;

(*
Print["phi = ", phi, " rad = ", phi 180/N[Pi], " deg"];
Print["xB = ", xB, " m"]; Print["yB = ", yB, " m"];
Print["xD = ", xD, " m"]; Print["yD = ", yD, " m"];
*)
apply[lear, names[
General::spell],
General::spell1];

(* Euclidian distance function *)
Dist[xP_, yP_, xQ_, yQ_] := Sqrt[(xP - xQ)^2 + (yP - yQ)^2];

(* Input data *)
AB = 0.14;
AC = 0.06;
AE = 0.25;
CD = 0.15;

(* Position of joint A *)
xA = yA = 0;

(* Position of joint C *)
xC = 0;
yC = AC;

(* Position of joint E *)
xE = 0;
yE = -AE;

increment = 0;
For[phi = 0, phi <= 2*Pi, phi += Pi/6, 

(* Position of joint B *)
xB = AB*Cos[phi];
yB = AB*Sin[phi];

(* Position of joint D *)
eqnD1 = (xDsol - xC)^2 + (yDsol - yC)^2 - CD^2 == 0;
eqnD2 = (yB - yC)/(xB - xC) == (yDsol - yC)/(xDsol - xC);
solutionD = Solve[{eqnD1, eqnD2}, {xDsol, yDsol}];

(* Two solutions for D *)
xD1 = xDsol/.solutionD[[1]];
yD1 = yDsol/.solutionD[[1]];
xD2 = xDsol/.solutionD[[2]];
yD2 = yDsol/.solutionD[[2]];

(* Select the correct position for D *)
If[increment == 0, 
If[xD1 < xC, xD = xD1; yD = yD1, xD = xD2; yD = yD2],
dist1 = Dist[xD1, yD1, xDold, yDold];
dist2 = Dist[xD2, yD2, xDold, yDold];
If[dist1 < dist2, xD = xD1; yD = yD1, xD = xD2; yD = yD2];
]
xDold = xD;
yDold = yD;
increment++;

(* Print *)
Print["phi = ", phi, " rad = ", phi*180/Pi, " deg"];
Print["xB = ", xB, " m"];
Print["yB = ", yB, " m"];
Print["xD = ", xD, " m"];
Print["yD = ", yD, " m"];

markers = Table[{Point[{xA, yA}], 
                 Point[{xB, yB}], 
                 Point[{xC, yC}], 
                 Point[{xD, yD}], 
                 Point[{xE, yE}]},
               {i, 1, 5}];

name = Table[{Text["A", {0, 0}, {-1, 1}], 
               Text["B", {xB, yB}, {0, -1}], 
               Text["C", {xC, yC}, {-1, -1}],
               Text["D", {xD, yD}, {0, -1}], 
               Text["E", {xE, yE}, {-1, 1}]}];

graph[increment] = Graphics[
{RGBColor[1, 0, 0], 
  Line[{{xA, yA}, {xB, yB}}], 
  RGBColor[0, 1, 0], 
  Line[{{xB, yB}, {xD, yD}}], 
  RGBColor[0, 0, 1], 
  Line[{{xD, yD}, {xE, yE}}], 
  RGBColor[1, 1, 1], 
  PointSize[0.01], markers},
name];

Show[Graphics[graph[increment]],
  PlotRange -> {{-.25, .25}, {-0.3, .25}},
  Frame -> True,
  AxesOrigin -> {xA, yA},
  FrameLabel -> {"x", "y"},
  Axes -> {True, True},
  AspectRatio -> Automatic];

] (* End of FOR loop *)
(* All positions on the same graphic *)

Show [Table[graph [i] , { i , increment } ] ,
   PlotRange -> { { -.25 , .25 } ,
                  { -3 , .25 } } ,
   Frame -> True,
   AxesOrigin -> {xA,yA},
   FrameLabel -> {"x","y"},
   Axes -> {True,True},
   AspectRatio -> Automatic ] ;