Chapter 2
Position Analysis

2.1 Absolute Cartesian Method

The position analysis of a kinematic chain requires the determination of the joint positions, the position of the centers of gravity, and the angles of the links with the horizontal axis. A planar link with the end nodes A and B is considered in Fig. 2.1. Let (x_A, y_A) be the coordinates of the joint A with respect to the reference frame xOy, and (x_B, y_B) be the coordinates of the joint B with the same reference frame. Using Pythagoras the following relation can be written

$$(x_B - x_A)^2 + (y_B - y_A)^2 = AB^2 = L_{AB}^2, \tag{2.1}$$

where L_{AB} is the length of the link AB. Let ϕ be the angle of the link AB with the horizontal axis Ox. Then, the slope m of the link AB is defined as

$$m = \tan\phi = \frac{y_B - y_A}{x_B - x_A}. \tag{2.2}$$

Let n be the intercept of AB with the vertical axis Oy. Using the slope m and the intercept n, the equation of the straight link, in the plane, is

$$y = mx + n, \tag{2.3}$$

where x and y are the coordinates of any point on this link.

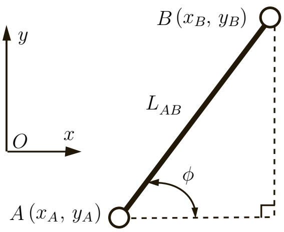

Fig. 2.1 Planar rigid link with two nodes

2.2 Slider-Crank (R-RRT) Mechanism

Exercise
The R-RRT (slider-crank) mechanism shown in Fig. 2.2a has the dimensions: $AB = 0.5$ m and $BC = 1$ m. The driver link 1 makes an angle $\phi = \phi_1 = 45°$ with the horizontal axis. Find the positions of the joints and the angles of the links with the horizontal axis.

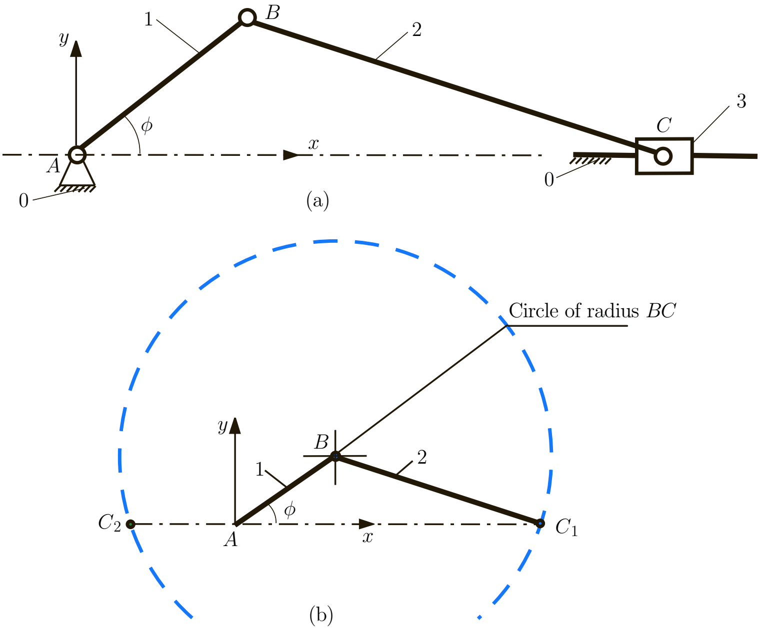

Fig. 2.2 (a) Slider-crank (R-RRT) mechanism and (b) two solutions for joint C: C_1 and C_2

Solution
The MATLAB® program starts with the statements:

```
clear all % clears all variables and functions
clc  % clears the command window and homes the cursor
close all % closes all the open figure windows
```

The MATLAB commands for the input data are:

```
AB=0.5;  BC=1.;
```

The angle of the driver link 1 with the horizontal axis $\phi = 45°$. The MATLAB command for the input angle is:

```
phi=pi/4;
```

where `pi` has a numerical value approximately equal to 3.14159.

Position of Joint A
A Cartesian reference frame xOy is selected. The joint A is in the origin of the reference frame, that is, $A \equiv O$,

$$x_A = 0,\ y_A = 0,$$

or in MATLAB:

```
xA=0; yA=0;
```

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 ϕ is known, the coordinates of the joint B are computed from the following expressions:

$$\begin{aligned} x_B &= AB\cos\phi = (0.5)\cos 45^\circ = 0.353553\,\text{m}, \\ y_B &= AB\sin\phi = (0.5)\sin 45^\circ = 0.353553\,\text{m}. \end{aligned} \tag{2.4}$$

The MATLAB commands for Eq. 2.4 are:

```
xB=AB*cos(phi);
yB=AB*Sin(phi);
```

where `phi` is the angle ϕ in radians.

Position of Joint C
The unknowns are the coordinates of the joint C, x_C and y_C. The joint C is located on the horizontal axis $y_C = 0$ and with MATLAB:

```
yC=0;
```

The length of the segment BC is constant

$$(x_B - x_C)^2 + (y_B - y_C)^2 = BC^2, \tag{2.5}$$

or

$$(0.353553 - x_C)^2 + (0.353553 - 0)^2 = 1^2.$$

Equation 2.5 with MATLAB command is:

```
eqnC='(xB-xCsol)^2+(yB-yC)^2=BC^2';
```

where `xCsol` is the unknown. To solve the equation, a specific MATLAB command will be used. The command:

```
solve('eqn1','eqn2',...,'eqnN','var1','var2',...'varN')
```

attempts to solve an equation or set of equations `'eqn1','eqn2',...,'eqnN'` for the variables `'eqnN','var1','var2',...'varN'`. The set of equations are symbolic expressions or strings specifying equations. The MATLAB command to find the solution `xCsol` of the equation:

```
eqnC='(xB-xCsol)^2+(yB-yC)^2=BC^2'
```

is

```
solC=solve(eqnC,'xCsol');
```

Because it is a quadratic equation two solutions are found for the position of C. The two solutions are given in a vector form: `solC` is a vector with two components `solC(1)` and `solC(2)`. To obtain the numerical solutions the `eval` command has to be used:

```
xC1=eval(solC(1));
xC2=eval(solC(2));
```

The command `eval(s)`, where `s` is a string, executes the string as an expression or statement. The two solutions for x_C, as shown in Fig. 2.2b, are:

$$x_{C_1} = 1.289 \text{ m} \quad \text{and} \quad x_{C_2} = -0.5819 \text{ m}.$$

To determine the correct position of the joint C for the mechanism, an additional condition is needed. For the first quadrant, $0 \leq \phi \leq 90^\circ$, the condition is $x_C > xB$. This MATLAB condition for `xC` located in the first quadrant is:

```
if xC1 > xB xC = xC1; else xC = xC2; end
```

The general form of the `if` statement is:

`if` *expression statements* `else` *statements* `end`

The x-coordinate of the joint C is $x_C = x_{C_1} = 1.2890\,\text{m}$. The angle of the link 2 (link BC) with the horizontal is

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

The MATLAB expression for the angle ϕ_2 is:

```
phi2 = atan((yB-yC)/(xB-xC));
```

The statement `atan(s)` is the arctangent of the elements of `s`. The numerical solutions for B, C, and ϕ_2 are printed using the statements:

```
fprintf('xB = %g (m) \n', xB)
fprintf('yB = %g (m) \n', yB)
fprintf('xC = %g (m) \n', xC)
fprintf('yC = %g (m) \n', yC)
fprintf('phi2 = %g (degrees) \n', phi2*180/pi)
```

The statement `fprintf(f,format,s)` writes data in the real part of array `s` to the file `f`. The data is formated under control of the specified `format` string. The results of the program are displayed as:

```
xB = 0.353553 (m)
yB = 0.353553 (m)
xC = 1.28897 (m)
yC = 0 (m)
phi2 = -20.7048 (degrees)
```

The mechanism is plotted with the help of the command `plot`. The statement `plot(x,y,c)` plots vector `y` versus vector `x`, and `c` is a character string. For the R-RRT mechanism two straight lines *AB* and *BC* are plotted with:

```
plot([xA,xB],[yA,yB],'r-o',[xB,xC],[yB,yC],'b-o')
```

The line *AB* is a red (`r` red), solid line (`-` solid), with a circle (`o` circle) at each data point and the line *BC* is a blue (`b` blue), solid line with a circle at each data point. The graphic of the mechanism obtained with MATLAB is shown in Fig. 2.3. The *x*-axis and *y*-axis are labeled using the commands:

```
xlabel('x (m)')
ylabel('y (m)')
```

and a title is added with:

```
title('positions for \phi = 45 (deg)')
```

On the figure, the joints *A*, *B*, and *C* are identified with the statements:

```
text(xA,yA,' A'),...
text(xB,yB,' B'),...
```

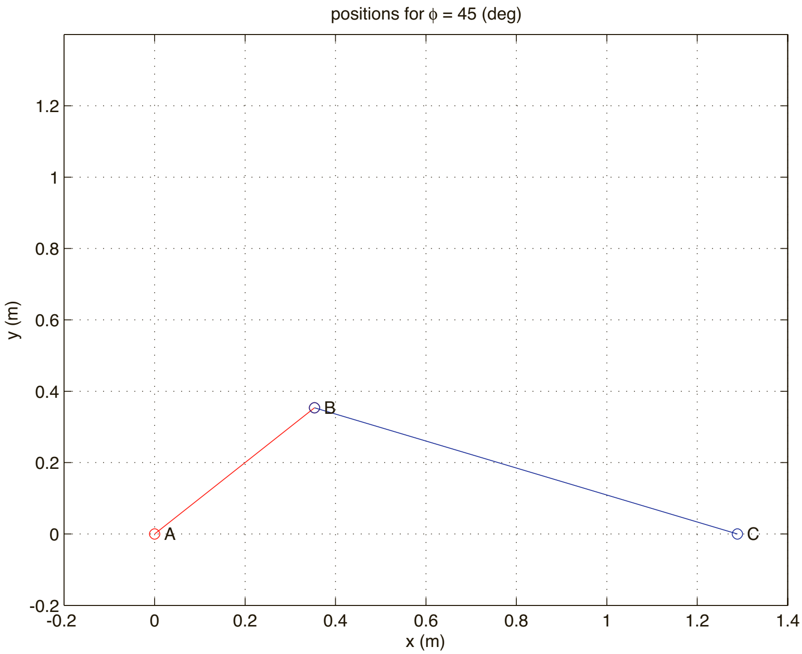

Fig. 2.3 MATLAB graphic of R-RRT mechanism

```
text(xC,yC,' C'),...
axis([-0.2 1.4 -0.2 1.4]),...
grid
```

The commas and ellipses (`...`) after the command are used to execute the commands together. Otherwise, the data will be plotted, then the labels will be added and the data replotted, and so on.

The statement `axis([xMIN xMAX yMIN yMAX])` sets scaling for the x and y axes on the current plot. To improve the graph a background grid was added with the command `grid`.

The MATLAB program for the positions is given in Appendix A.1.

2.3 Four-Bar (R-RRR) Mechanism

Exercise

The considered four-bar (R-RRR) planar mechanism is shown in Fig. 2.4. The driver link is the rigid link 1 (the element AB) and the origin of the reference frame is at A. The following data are given: AB=0.150 m, BC=0.35 m, CD=0.30 m, CE=0.15 m,

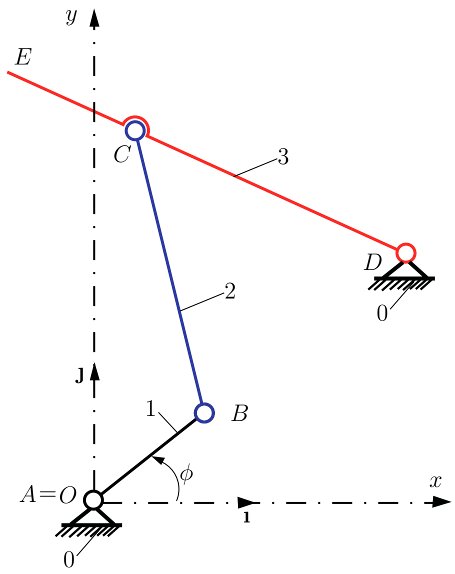

Fig. 2.4 Four-bar (R-RRR) mechanism

x_D=0.30 m, and y_D=0.30 m. The angle of the driver link 1 with the horizontal axis is $\phi = \phi_1 = 45°$. Find the positions of the joints and the angles of the links with the horizontal axis.

Solution

The Cartesian reference frame xyz with the unit vectors $[\mathbf{\imath}, \mathbf{\jmath}, \mathbf{k}]$ is shown Fig. 2.4. Since the joint A is the origin of the reference system $A \equiv O$ the coordinates of A are $x_A = 0$, $y_A = 0$ and the position vector of A is $\mathbf{r}_A = x_A\,\mathbf{\imath} + y_A\,\mathbf{\jmath}$. The position vectors $\mathbf{r}_A$ and $\mathbf{r}_D$ are introduced in MATLAB as:

```
rA = [xA yA 0];
rD = [xD yD 0];
```

In the MATLAB environment, a three-dimensional vector `v` is written as a list of variables `v = [ x y z ]`, where `x`, `y`, and `z` are the spatial coordinates of the vector `v`. The first component of the vector `v` is `x=v(1)`, the second component is `y=v(2)`, and the third component is `z=v(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 ϕ is known, the coordinates of the joint B are computed from the following expressions:

$$x_B = AB\cos\phi = 0.106\,\text{m}, \quad y_B = AB\sin\phi = 0.106\,\text{m}.$$

The position vector of B is $\mathbf{r}_B = x_B\,\mathbf{\imath} + y_B\,\mathbf{\jmath}$. The MATLAB program for this part is:

```
xB = AB*cos(phi); yB = AB*sin(phi); rB = [xB yB 0];
```

Position of Joint C

The unknowns are the coordinates of the joint C, x_C and y_C. Knowing the positions of the joints B and D, the position of the joint C can be computed using the fact that the lengths of the links BC and CD are constants

$$(x_C - x_B)^2 + (y_C - y_B)^2 = BC^2,$$
$$(x_C - x_D)^2 + (y_C - y_D)^2 = CD^2,$$

or

$$\begin{aligned}(x_C - 0.106)^2 + (y_C - 0.106)^2 &= 0.350^2,\\ (x_C - 0.300)^2 + (y_C - 0.300)^2 &= 0.300^2.\end{aligned} \tag{2.6}$$

Equations 2.6 consist of two quadratic equations. Solving this system of equations, two sets of solutions are found for the position of the joint C. These solutions are

$$x_{C_1} = 0.0401\,\text{m},\ \ y_{C_1} = 0.4498\,\text{m} \ \text{ and } \ x_{C_2} = 0.4498\,\text{m},\ \ y_{C_2} = 0.0401\,\text{m}.$$

The MATLAB program for calculating the coordinates of C_1 and C_2 is:

```
eqnC1 = '( xCsol - xB )^2 + ( yCsol - yB )^2 = BC^2';
eqnC2 = '( xCsol - xD )^2 + ( yCsol - yD )^2 = CD^2';
solC = solve(eqnC1, eqnC2, 'xCsol, yCsol');
xCpositions = eval(solC.xCsol);
yCpositions = eval(solC.yCsol);
% first component of the vector xCpositions
xC1 = xCpositions(1);
% second component of the vector xCpositions
xC2 = xCpositions(2);
% first component of the vector yCpositions
yC1 = yCpositions(1);
% second component of the vector yCpositions
yC2 = yCpositions(2);
```

The points C_1 and C_2 are the intersections of the circle of radius BC (with the center at B) with the circle of radius CD (with the center at D), as shown in Fig. 2.5. To determine the correct position of the joint C for this mechanism, a constraint condition is needed: $x_C < x_D$. Because $x_D = 0.300$ m, the coordinates of joint C have the following numerical values:

$$x_C = x_{C_1} = 0.0401\,\text{m} \quad \text{and} \quad y_C = y_{C_1} = 0.4498\,\text{m}.$$

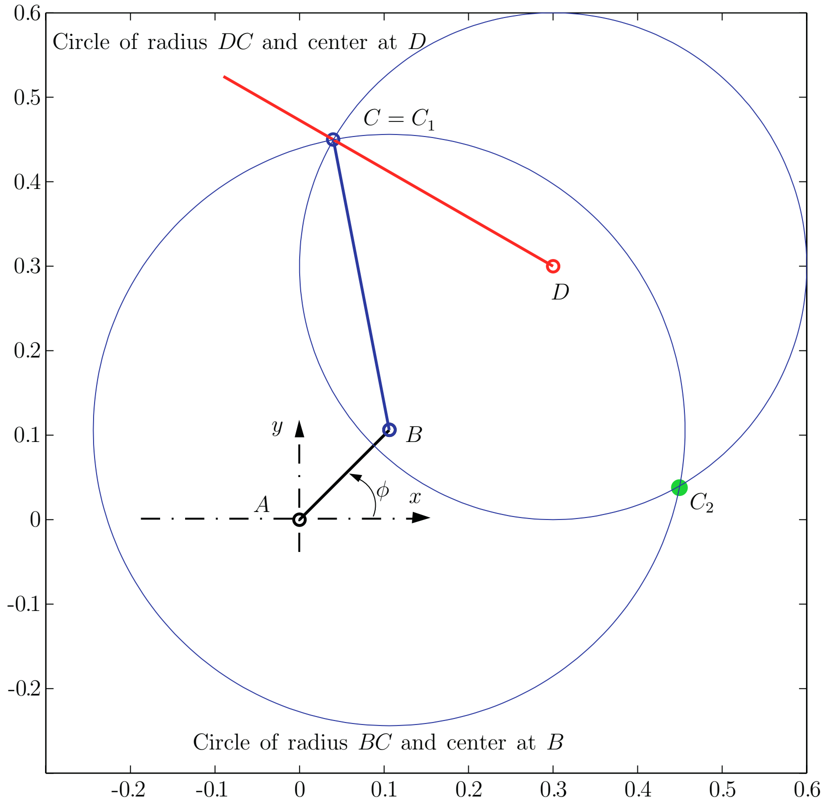

Fig. 2.5 Two solutions for the position of joint C

The MATLAB program for selecting the correct position of C is:

```
if xC1 < xD
    xC = xC1; yC=yC1;
else
    xC = xC2; yC=yC2;
end
rC = [xC yC 0]; % Position vector of C
```

Position of Point E
The unknowns are the coordinates of the point E, x_E and y_E. The position of the point E is determined from the equation

$$(x_E - x_C)^2 + (y_E - y_C)^2 = CE^2, \tag{2.7}$$

or

$$(x_E - 0.0401)^2 + (y_E - 0.4498)^2 = 0.15^2.$$

The joints D, C and E are located on the same straight element DE. For these points, the following equation can be written

$$\frac{y_D - y_C}{x_D - x_C} = \frac{y_E - y_C}{x_E - x_C}, \tag{2.8}$$

or

$$\frac{0.300 - 0.4498}{0.300 - 0.0401} = \frac{y_E - 0.4498}{x_E - 0.0401}.$$

Equations 2.7 and 2.8 form a system from which the coordinates of the point E can be computed. Two solutions are obtained, Fig. 2.6, and the numerical values are

$$x_{E_1} = -0.0899\,\text{m},\ y_{E_1} = 0.5247\,\text{m},$$
$$x_{E_2} = 0.1700\,\text{m},\ \ y_{E_2} = 0.3749\,\text{m}.$$

The MATLAB program for calculating the coordinates of E_1 and E_2 is:

```
eqnE1 = '( xEsol - xC )^2 + ( yEsol - yC )^2 = CE^2 ';
eqnE2 = '(yD-yC)/(xD-xC)=(yEsol-yC )/(xEsol-xC)';
solE = solve(eqnE1, eqnE2, 'xEsol, yEsol');
```

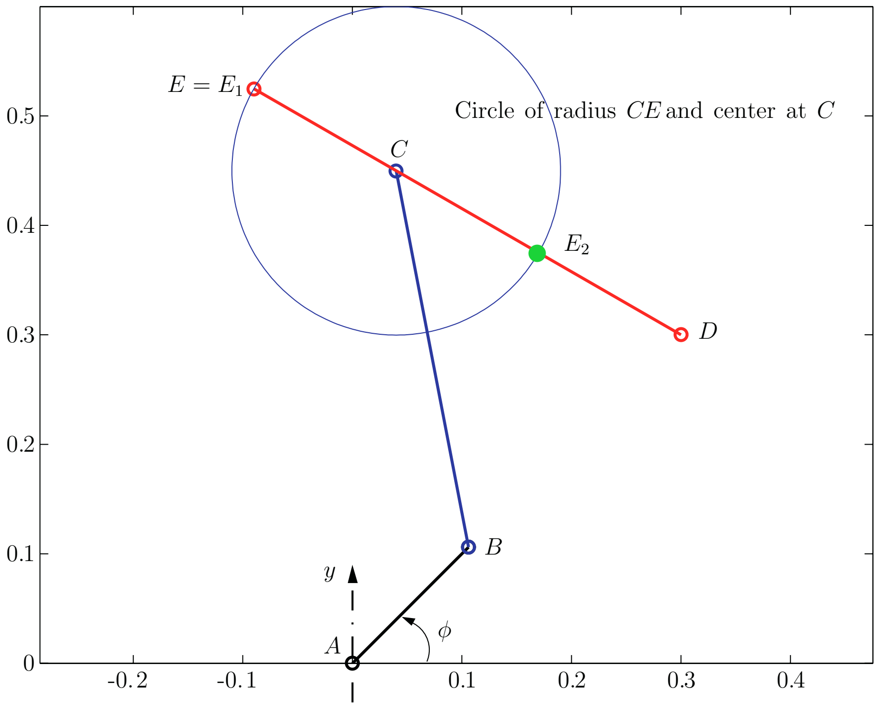

Fig. 2.6 Two solutions for the position of point E

```
xEpositions=eval(solE.xEsol);
yEpositions=eval(solE.yEsol);
xE1 = xEpositions(1); xE2 = xEpositions(2);
yE1 = yEpositions(1); yE2 = yEpositions(2);
```

For continuous motion of the mechanism, a constraint condition is needed, $x_E < x_C$. Using this condition, the coordinates of the point E are

$$x_E = x_{E_1} = -0.0899\,\text{m} \quad \text{and} \quad y_E = y_{E_1} = 0.5247\,\text{m}.$$

The MATLAB program for selecting the correct position of E is

```
if xE1 < xC
    xE = xE1; yE=yE1;
else
    xE = xE2; yE=yE2;
end
rE = [xE yE 0]; % Position vector of E
```

The angles of the links 2, 3, and 4 with the horizontal are

$$\phi_2 = \arctan\frac{y_B - y_C}{x_B - x_C},\ \ \phi_3 = \arctan\frac{y_D - y_C}{x_D - x_C},$$

and in MATLAB

```
phi2 = atan((yB-yC)/(xB-xC));
phi3 = atan((yD-yC)/(xD-xC));
```

The results are printed using the statements:

```
fprintf('rA = [ %g, %g, %g ] (m) \n', rA)
fprintf('rD = [ %g, %g, %g ] (m) \n', rD)
fprintf('rB = [ %g, %g, %g ] (m) \n', rB)
fprintf('rC = [ %g, %g, %g ] (m) \n', rC)
fprintf('rE = [ %g, %g, %g ] (m) \n', rE)
fprintf('phi2 = %g (degrees) \n', phi2*180/pi)
fprintf('phi3 = %g (degrees) \n', phi3*180/pi)
```

The graph of the mechanism using MATLAB for $\phi = \pi/4$ is given by:

```
plot([xA,xB],[yA,yB],'k-o','LineWidth',1.5)
hold on % holds the current plot
plot([xB,xC],[yB,yC],'b-o','LineWidth',1.5)
hold on
plot([xD,xE],[yD,yE],'r-o','LineWidth',1.5)
```

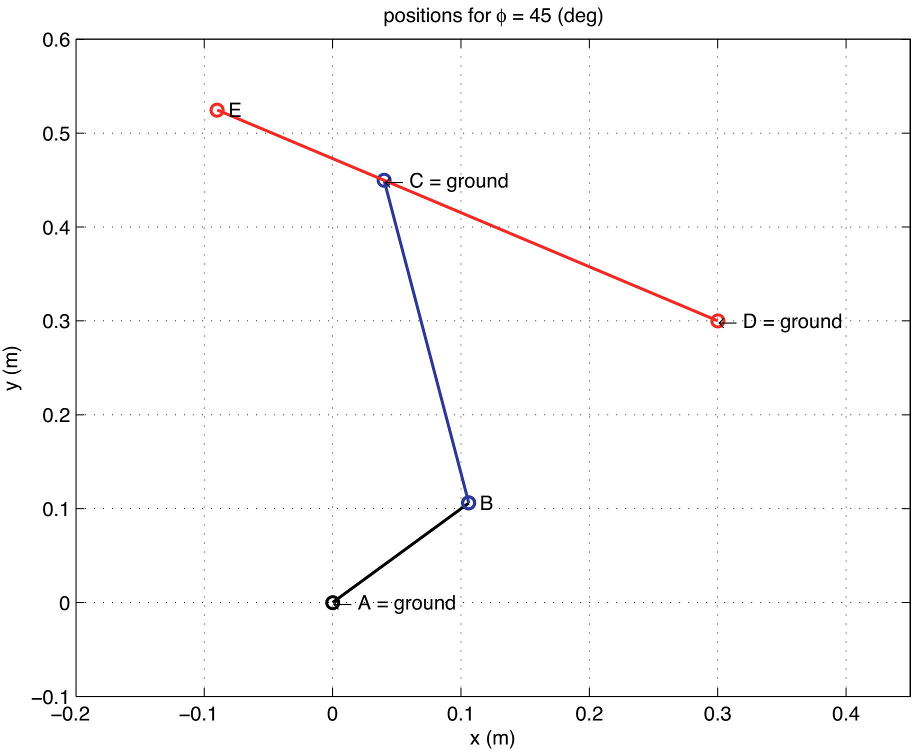

Fig. 2.7 MATLAB graphic of R-RRR mechanism

```
% adds major grid lines to the current axes
grid on,...
xlabel('x (m)'), ylabel('y (m)'),...
title('positions for \phi = 45 (deg)'),...
text(xA,yA,'\leftarrow A = ground',...
'HorizontalAlignment','left'),...
text(xB,yB,' B'),...
text(xC,yC,'\leftarrow C = ground',...
'HorizontalAlignment','left'),...
text(xD,yD,'\leftarrow D = ground',...
'HorizontalAlignment','left'),...
text(xE,yE,' E'), axis([-0.2 0.45 -0.1 0.6])
```

The graph of the R-RRR mechanism using MATLAB is shown in Fig. 2.7. The MATLAB program for the positions and the results is given in Appendix A.2.

2.4 R-RTR-RTR Mechanism

Exercise
The planar R-RTR-RTR mechanism considered is shown in Fig. 2.8. The driver link is the rigid link 1 (the link AB). The following numerical data are given: $AB = 0.15$ m, $AC = 0.10$ m, $CD = 0.15$ m, $DF = 0.40$ m, and $AG = 0.30$ m. The angle of the driver link 1 with the horizontal axis is $\phi = 30°$.

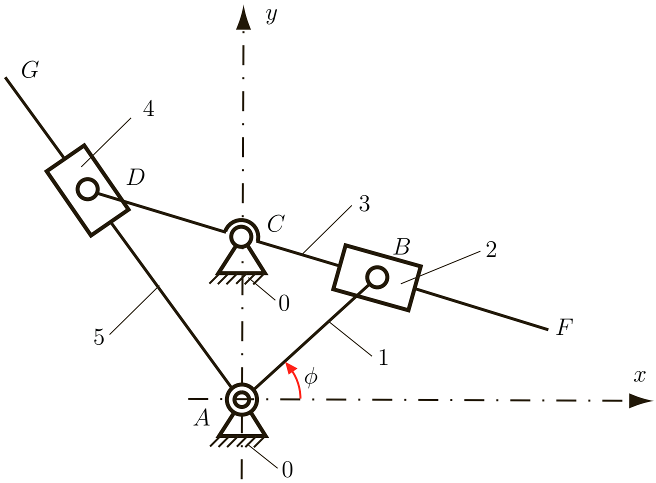

Fig. 2.8 R-RTR-RTR mechanism

Solution
The MATLAB commands for the input data are:

```
AB=0.15; AC=0.10; CD=0.15;   %(m)
phi=pi/6;  %(rad)
DF=0.40; AG=0.30;   % (m)
```

A Cartesian reference frame xOy is selected. The joint A is in the origin of the reference frame, that is, $A \equiv O$, $x_A = 0$, $y_A = 0$.

Position of Joint C
The position vector of C is $\mathbf{r}_C = x_C\,\mathbf{\imath} + y_C\,\mathbf{\jmath} = 0.1\,\mathbf{\jmath}$ m.

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 ϕ is known, the coordinates of the joint B are computed from the following expressions:

$x_B = AB\cos\phi = 0.15\cos 30^\circ = 0.1299\,\text{m},\ y_B = AB\sin\phi = 0.15\sin 30^\circ = 0.075\,\text{m},$

and $\mathbf{r}_B = x_B\mathbf{\imath} + y_B\mathbf{\jmath}$. The MATLAB statements for the positions of the joints A, C, E, and B are:

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

Position of Joint D
The unknowns are the coordinates of the joint D, x_D and y_D. The length of the segment CD is constant:

$$(x_D - x_C)^2 + (y_D - y_C)^2 = CD^2, \tag{2.9}$$

or

$$(x_D - 0)^2 + (y_D - 0.10)^2 = 0.15^2.$$

The points B, C, and D are on the same straight line with the slope

$$m = \frac{(y_B - y_C)}{(x_B - x_C)} = \frac{(y_D - y_C)}{(x_D - x_C)}, \tag{2.10}$$

or

$$\frac{(0.075 - 0.1)}{(0.1299 - 0.0)} = \frac{(y_D - 0.1)}{(x_D - 0.0)}.$$

Equations 2.9 and 2.10 form a system from which the coordinates of the joint D can be computed. To solve the system of equations the MATLAB statement `solve` will be used:

```
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);
% first component of the vector xDpositions
xD1 = xDpositions(1);
% second component of the vector xDpositions
xD2 = xDpositions(2);
% first component of the vector yDpositions
yD1 = yDpositions(1);
% second component of the vector yDpositions
yD2 = yDpositions(2);
```

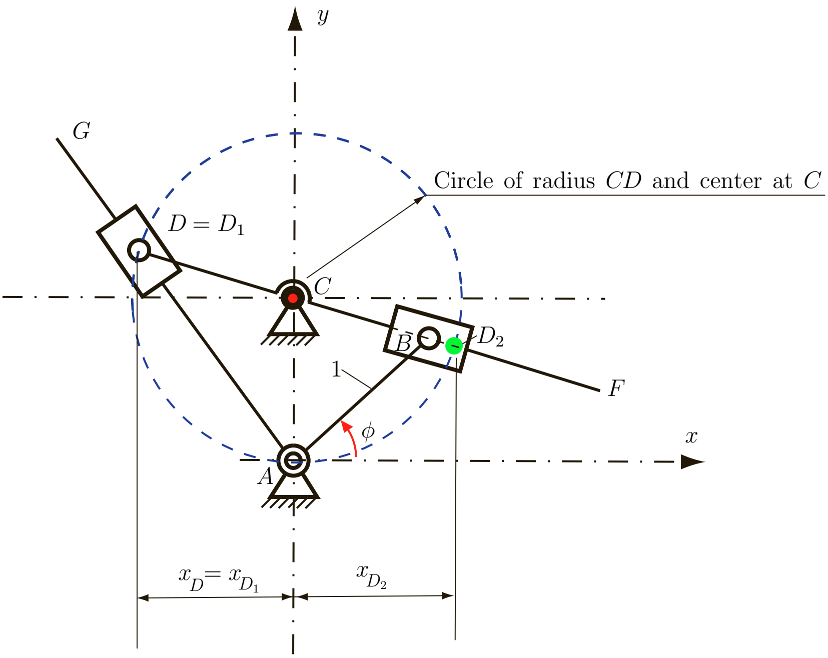

Fig. 2.9 Graphical solutions for joint D

These solutions D_1 and D_2 are located at the intersection of the line BC with the circle centered in C and radius CD (Fig. 2.9), and they have the following numerical values:

$$x_{D1} = -0.1473 \text{ m}, \ y_{D1} = 0.1283 \text{ m},$$
$$x_{D2} = 0.1473 \text{ m}, \ y_{D2} = 0.0717 \text{ m}.$$

To determine the correct position of the joint D for the mechanism, an additional condition is needed. For the first quadrant, $0 \leq \phi \leq 90°$, the condition is $x_D \leq x_C$. This condition with MATLAB is given by:

```
if xD1 <= xC
   xD = xD1; yD=yD1;
else
   xD = xD2; yD=yD2;
end
rD = [xD yD 0]; % Position of D
```

Because $x_C = 0$, the coordinates of the joint D are:

$$x_D = x_{D1} = -0.1473\,\text{m} \quad \text{and} \quad y_D = y_{D1} = 0.1283\,\text{m}.$$

The angles of the links 2, 3, and 4 with the horizontal are

$$\phi_2 = \arctan\frac{y_B - y_C}{x_B - x_C}, \;\; \phi_3 = \phi_2, \;\; \phi_4 = \arctan\frac{y_D}{x_D} + \pi, \;\; \phi_5 = \phi_4,$$

and in MATLAB:

```
phi2 = atan((yB-yC)/(xB-xC));
phi3 = phi2;
phi4 = atan(yD/xD)+pi;
phi5 = phi4;
```

The points F and G are calculated in MATLAB with:

```
xF = xD + DF*cos(phi3) ; yF = yD + DF*sin(phi3) ;
rF = [xF yF 0]; % Position vector of F
xG = AG*cos(phi5) ; yG = AG*sin(phi5) ;
rG = [xG yG 0]; % Position vector of G
```

The results are printed using the statements:

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

The graph of the mechanism in MATLAB for $\phi = \pi/6$ is given by:

```
plot([xA,xB],[yA,yB],'k-o','LineWidth',1.5)
hold on % holds the current plot
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([xA,xD],[yA,yD],'r-o','LineWidth',1.5)
hold on
plot([xD,xG],[yD,yG],'r-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(xF,yF,' F'), text(xG,yG,' G'),...
axis([-0.3 0.3 -0.1 0.3])
```

The MATLAB program for the positions and the results for the R-RTR-RTR mechanism for $\phi = 30^\circ$ is given in Appendix A.3.

2.5 R-RTR-RTR Mechanism: Complete Rotation

For a complete rotation of the driver link AB, $0 \leq \phi \leq 360^\circ$, a step angle of 60° is selected. To calculate the position analysis for a complete cycle the MATLAB statement `for` *var=startval:step:endval, statement* `end` is used. It repeatedly evaluates *statement* in a loop. The counter variable of the loop is *var*. At the start, the variable is initialized to value *startval* and is incremented (or decremented when *step* is negative) by the value *step* for each iteration. The *statement* is repeated until *var* has incremented to the value *endval*. For the considered mechanism the following applies:

```
for phi=0:pi/3:2*pi, Program block, end;
```

2.5.1 Method I: Constraint Conditions

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 ϕ located in the first quadrant $0^\circ \leq \phi \leq 90^\circ$ and the fourth quadrant $270^\circ \leq \phi \leq 360^\circ$ (Fig. 2.10), the following relation exists between x_D and x_C:

$$x_D \leq x_C = 0.$$

For the angle ϕ located in the second quadrant $90^\circ < \phi \leq 180^\circ$ and the third quadrant $180^\circ < \phi < 270^\circ$ (Fig. 2.11), the following relation exists between x_D and x_C:

$$x_D \geq x_C = 0.$$

The following MATLAB commands are used to determine the correct position of the joint D for all four quadrants:

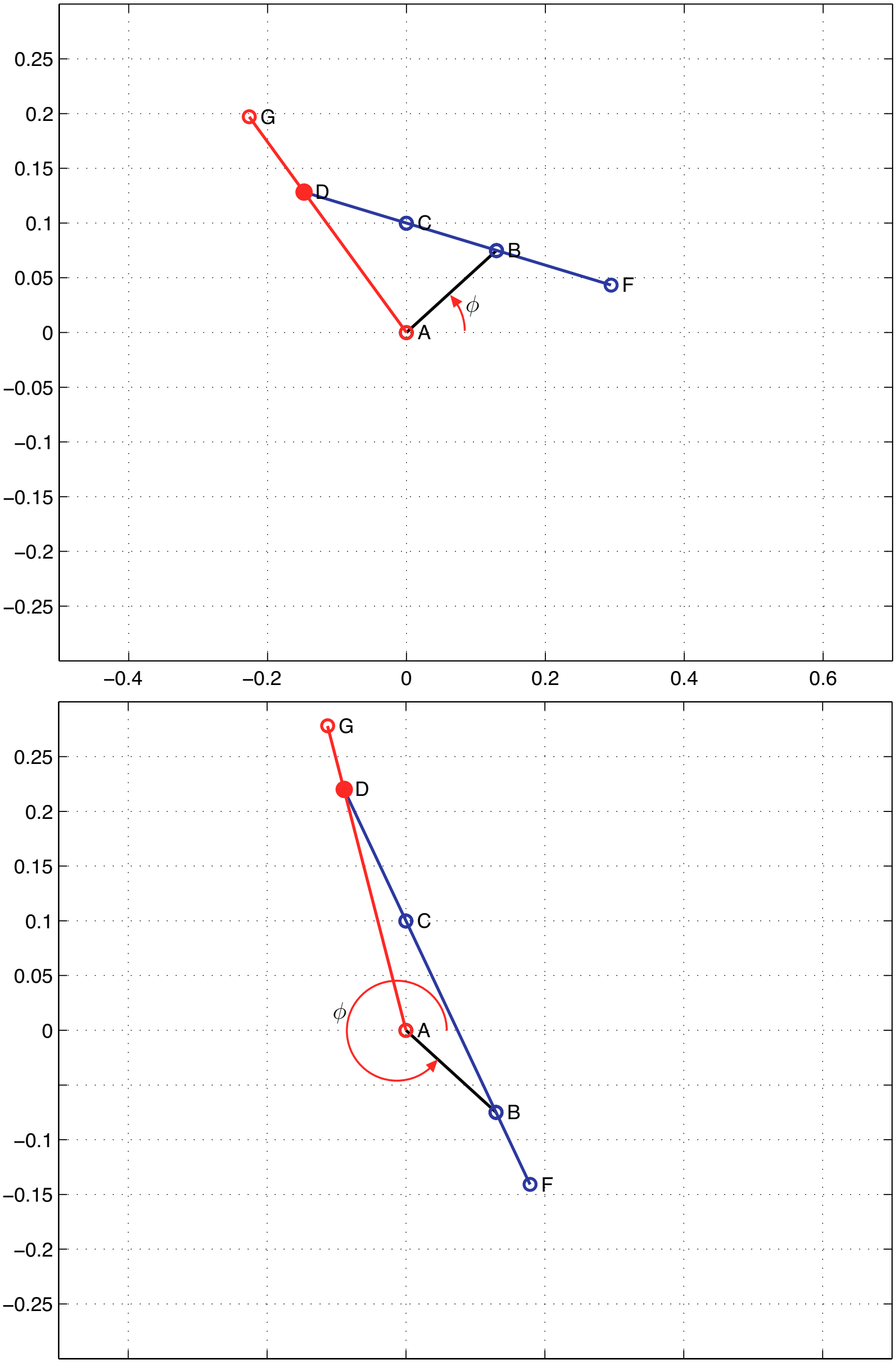

Fig. 2.10 R-RTR-RTR mechanism for $0° < \phi \leq 90°$ and $270° \leq \phi \leq 360°$

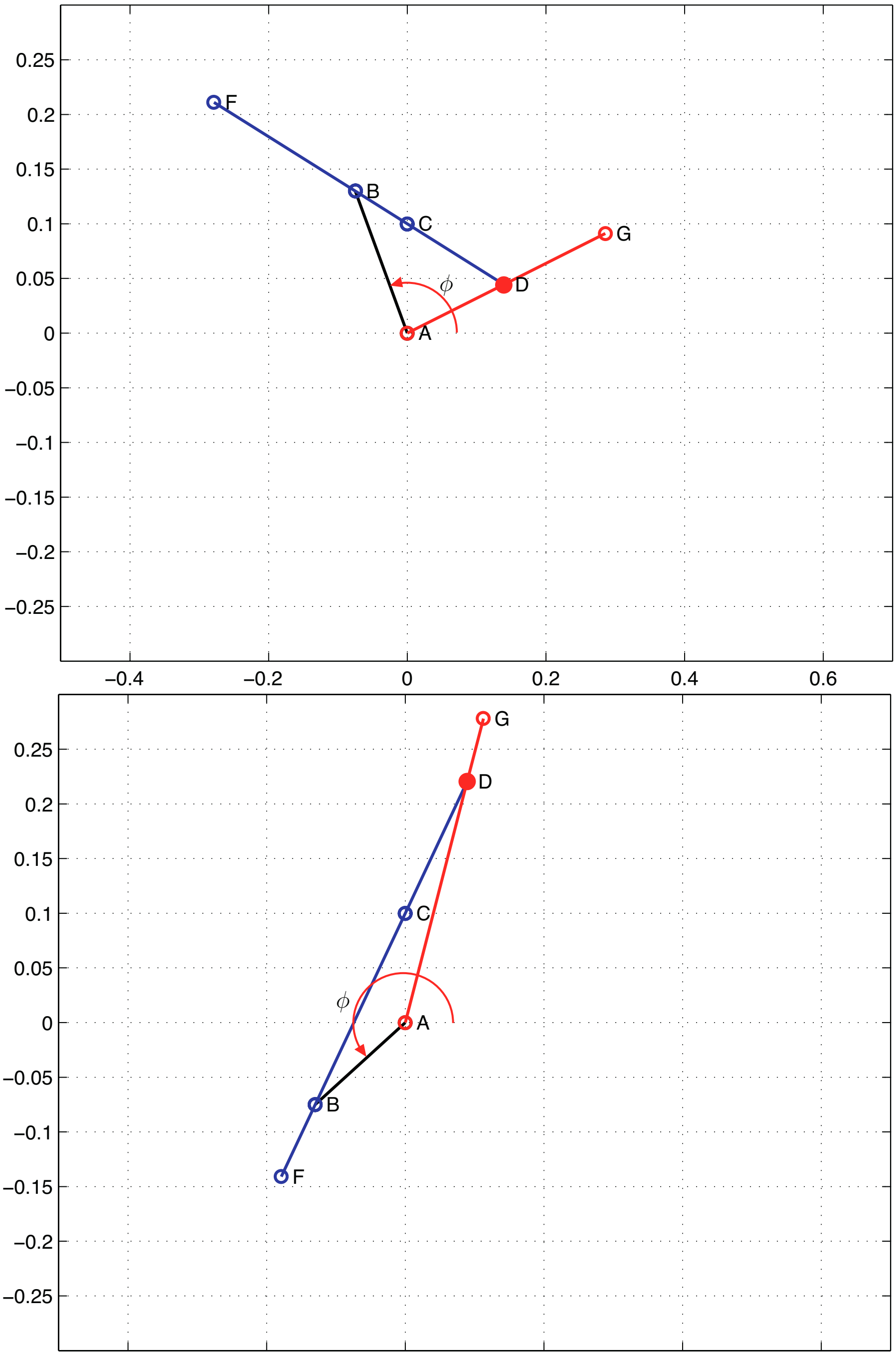

Fig. 2.11 R-RTR-RTR mechanism for $90° < \phi \leq 180°$ and $180° \leq \phi \leq 270°$

```
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
```

where `||` is the logical OR function. The MATLAB program and the results for a complete rotation of the driver link using method I is given in Appendix A.4. The graphic of the mechanism for a complete rotation of the driver link is given in Fig. 2.12. To simplify the graphic the points *E* and *G* are not shown on the figure.

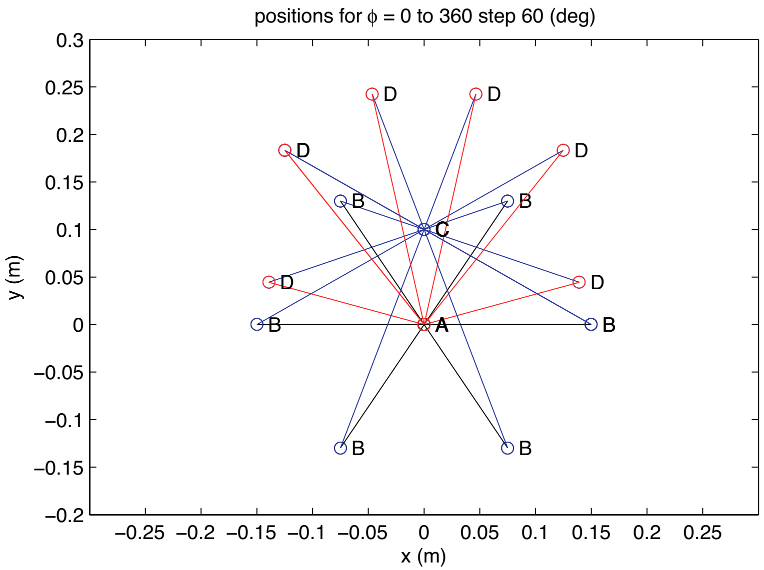

Fig. 2.12 MATLAB graphic of R-RTR-RTR mechanism for a complete rotation of the driver link $0^\circ \leq \phi \leq 360^\circ$

Another way of plotting the simulation of the mechanism for a complete rotation of the driver link is:

```
plot([xA,xB],[yA,yB],'k-o',[xB,xC],[yB,yC],'b-o',...
[xC,xD],[yC,yD],'b-o',[xD,xA],[yD,yA],'r-o'),...
hold off % resets axes properties to their defaults
text(xA,yA,'  A'), text(xB,yB,'  B'),...
text(xC,yC,'  C'), text(xD,yD,'  D'),...
axis([-0.3 0.3 -0.2 0.3]),grid,...
pause(0.8)
```

The MATLAB command `hold off` resets the axes properties to their defaults before drawing new plots and the command `pause(T)` pauses execution for `T` seconds before continuing.

2.5.2 *Method II: Euclidian Distance Function*

Another method for the position analysis for a complete rotation of the driver link uses constraint conditions only for the initial value of the angle ϕ. Next 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:

$$d = \sqrt{(x_P - x_Q)^2 + (y_P - y_Q)^2}. \tag{2.11}$$

In MATLAB, the following function is introduced with a m-file (Dist.m):

```
function d=Dist(xP,yP,xQ,yQ);
d=sqrt((xP-xQ)^2+(yP-yQ)^2);
end
```

For the initial angle $\phi = 0°$, 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 $D = D_0 = D_k$. 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}, 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 are compared. If the distance to the first solution is less than the distance to the second solution, $d^I_k < d^{II}_k$, then the correct answer is $D_{k+1} = D^I_{k+1}$, or else $D_{k+1} = D^{II}_{k+1}$ (Fig. 2.13).

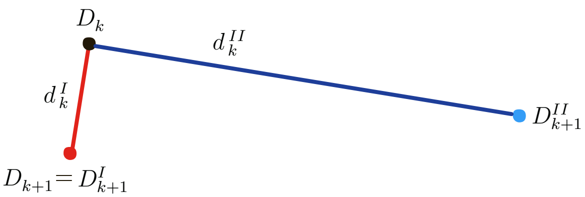

Fig. 2.13 Selection of the correct position: $d^I_k < d^{II}_k \Rightarrow D_{k+1} = D^I_{k+1}$

The following MATLAB statements are used to determine the correct position of the joint D using a single condition for all four quadrants:

```
% at the initial moment phi=0 => increment = 0
increment = 0 ;

% the step has to be small for this method
step=pi/6;
for phi=0:step:2*pi,

xB = AB*cos(phi); yB = AB*sin(phi); rB = [xB yB 0];
fprintf('rB =  [ %g, %g, %g ] (m)\n', rB)
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);
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];
end
```

At the beginning of the rotation the driver link makes an angle `phi=0` with the horizontal and the value of counter `increment` is 0. The MATLAB statement:

```
increment=increment+1;
```

specifies that 1 is to be added to the value in `increment` and the result stored back in `increment`. The value `increment` should be incremented by 1.

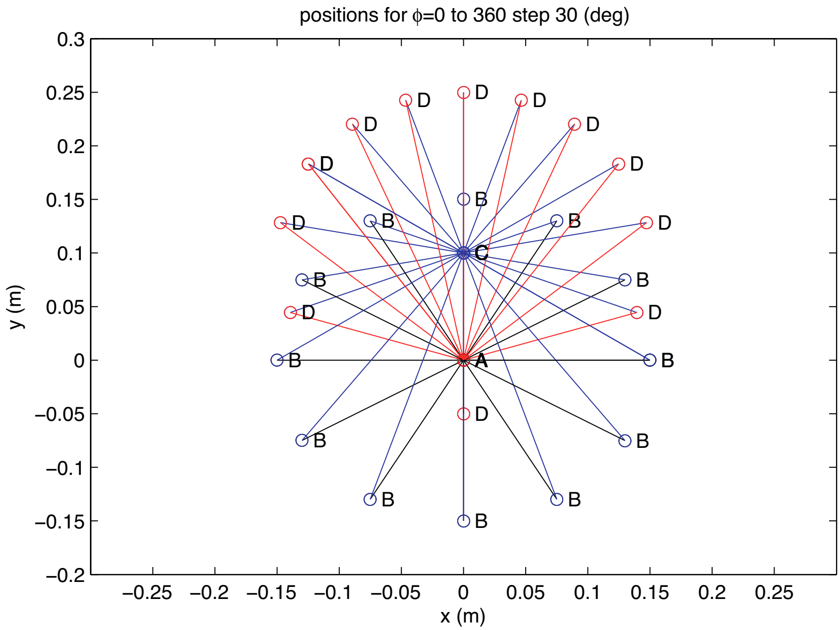

Fig. 2.14 MATLAB graphic of R-RTR-RTR mechanism for a complete rotation of the driver using the Euclidian distance

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.

For the Euclidian distance method the selection of the step of the angle ϕ is very important. If the step of the angle has a large value the method might give wrong answers and that is why it is important to check the graphic of the mechanism.

The MATLAB program for a complete rotation of the driver link using the second method is given in Appendix A.5. The graph of the mechanism for a complete rotation of the driver link (the step of the angle is 30°) is given in Fig. 2.14 (the points E and G are not shown).

2.6 Path of a Point on a Link with General Plane Motion

Exercise: R-RRT Mechanism

The mechanism shown in Fig. 2.2a has $AB = 0.5$ m and $BC = 1$ m. The link 2 (connecting rod BC) has a general plane motion: translation along the x-axis, translation along the y-axis, and rotation about the z-axis. The mass center of link 2 is located at C_2. Determine the path of point C_2 for a complete rotation of the driver link 1.

Solution
The coordinates of the joint B are

$$x_B = AB\cos\phi \quad \text{and} \quad y_B = AB\sin\phi,$$

where $0 \le \phi \le 360°$. The coordinates of the joint C are

$$x_C = x_B + \sqrt{BC^2 - y_B^2} \quad \text{and} \quad y_C = 0.$$

The mass center of the link 2 is the midpoint of the segment BC

$$x_{C_2} = \frac{x_B + x_C}{2} \quad \text{and} \quad y_{C_2} = \frac{y_B + y_C}{2}.$$

The MATLAB statements for the coordinates of C_2 are:

```
AB = .5; BC = 1; xA = 0; yA = 0; yC = 0;
incr = 0;
for phi=0:pi/10:2*pi,
  xB = AB*cos(phi); yB = AB*sin(phi);
  xC = xB + sqrt(BC^2-yB^2);
  incr = incr + 1;
  xC2(incr)=(xB+xC)/2; yC2(incr)=(yB+yC)/2;
end % end for
```

For the complete rotation of the driver link AB, $0 \le \phi \le 360°$, a step angle of $\pi/10$ was selected. For the coordinates of C_2 two vectors:

```
xC2=[xC2(1) xC2(2) ... xC2(incr) ... ]
yC2=[yC2(1) yC2(2) ... yC2(incr) ... ]
```

are obtained. The first components `xC2(1)` and `yC2(1)` are calculated for `phi=0` and `incr=1`. The path of C_2 is obtained by plotting the vector `yC2` in terms of `xC2`:

```
plot(xC2, yC2, '-ko'),...
xlabel('x (m)'), ylabel('y (m)'),...
title('Path described by C2'), grid
```

Figure 2.15 shows two plots: the mechanism for $0 \le \phi \le 360°$ and the closed path described by the point C_2 on the link 2 in general plane motion. The plots are obtained using the program in Appendix A.6.

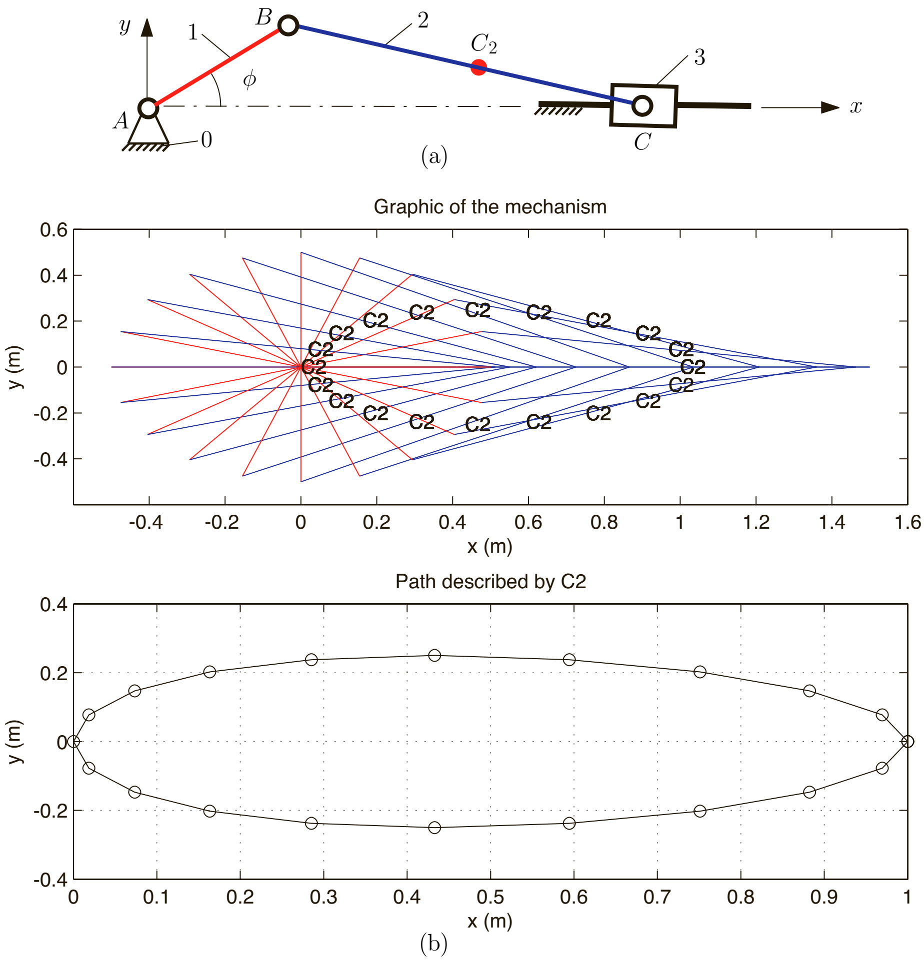

Fig. 2.15 (a) R-RRT mechanism, $AB = 0.5$ m, $BC = 1.0$ m, and $BC_2 = C_2C$; (b) MATLAB plots: mechanism for $0 \leq \phi \leq 360°$ and closed path described by point C_2

R-RRR Mechanism

The mechanism shown in Fig. 2.4 has the dimensions given in Sect. 2.3. The link 2 (link *BC*) has a general plane motion. The positions of the mechanism for $0 \leq \phi \leq 360°$ and the closed path described by the mass center C_2 of the link 2 are shown in Fig. 2.16. The plots are obtained using the program in Appendix A.7.

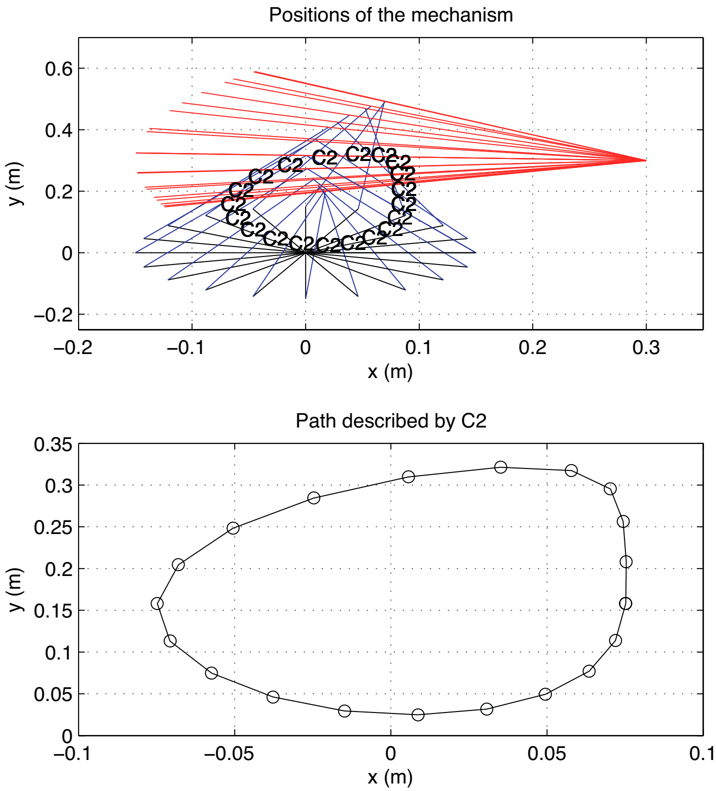

Fig. 2.16 Positions of the R-RRR mechanism for $0 \leq \phi \leq 360°$ and closed path described by the mass center C_2 of link 2.

2.7 Creating a Movie

The R-RTR-RTR mechanism shown in Fig. 2.8 has the dimensions given in Sect. 2.4. This example illustrates the use of movies to visualize the positions of the mechanism for $0 \leq \phi \leq 360°$.

The statement `moviein` is used to create a matrix large enough to hold 12 frames:

```
M = moviein(12);
```

The program has the structure

```
AB=0.15; AC=0.10; CD=0.15; %(m)
xA = 0; yA = 0; xC = 0 ; yC = AC;
% allocate/initialize the matrix to have 12 frames
M = moviein(12);
```

```
incr = 0;
for phi=0:pi/180:2*pi,
xB = AB*cos(phi); yB = AB*sin(phi);
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);
xD1 = xDpositions(1); xD2 = xDpositions(2);
yD1 = yDpositions(1); yD2 = yDpositions(2);
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

plot([xA,xB],[yA,yB],'k-o',...
       [xB,xC],[yB,yC],'b-o',...
       [xC,xD],[yC,yD],'b-o',...
       [xD,xA],[yD,yA],'r-o'),...
text(xA,yA,'  A'), text(xB,yB,'  B'),...
text(xC,yC,'  C'), text(xD,yD,'  D'), grid;

% xlim([Xmin Xmax])
% sets the x limits to the specified values
xlim([-0.3 0.3]);
% ylim([Ymin Ymax])
% sets the x limits to the specified values
ylim([-0.3 0.3]);

incr = incr + 1;

M(:,incr) = getframe; % record the movie

end % end for

movie2avi(M,'RRTRRTR.avi');
```

The statement, `getframe` returns the contents of the current axes, exclusive of the axis labels, title, or tick labels. After generating the movie, the statement, `movie2avi(M,'filename.avi')` creates the AVI movie `filename` from the MATLAB movie `M`. The `filename` input is a string enclosed in single quotes. In this case the name of the movie file is `RRTRRTR.avi`.

http://www.springer.com/978-1-84800-390-3

Mechanisms and Robots Analysis with MATLAB®
Marghitu, D.B.
2009, XII, 480 p., Hardcover
ISBN: 978-1-84800-390-3