

```
BeginPackage["Driver`"]

Driver::usage =
  "Driver[xA,yA,AB,phi,omega,alpha] computes the driver link position, velocity and :
Begin["`Private`"]

Driver[xA_,yA_,AB_,phi_,omega_,alpha_] :=
Block[ { xB, yB ,vBx, vBy, aBx, aBy },

xB = xA + AB Cos[phi] ;
yB = yA + AB Sin[phi] ;

vBx = - AB omega Sin[phi] ;
vBy =  AB omega Cos[phi] ;

aBx = - AB omega^2 Cos[phi] - AB alpha Sin[phi] ;
aBy = - AB omega^2 Sin[phi] + AB alpha Cos[phi] ;

Return[ { xB, yB, vBx, vBy, aBx, aBy } ] ;
]

End[ ]

EndPackage[ ]

Driver`

Driver[xA,yA,AB,phi,omega,alpha] computes
  the driver link position, velocity and acceleration vectors.

Driver`Private`

Driver`Private`
```

```
(* Driver mechanism *)

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

<<Driver.m ;

(* Input data *)
AB = 0.20 ;      (* m *)
phi = Pi/6 ;    (* rad *)
omega = 5. ;    (* rad/s *)
alpha = 0. ;    (* rad/s^2 *)

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

(* Position of joint B *)
xB = Driver[xA,yA,AB,phi,omega,alpha][[1]] ;
yB = Driver[xA,yA,AB,phi,omega,alpha][[2]] ;

(* Velocity of joint B *)
vBx = Driver[xA,yA,AB,phi,omega,alpha][[3]] ;
vBy = Driver[xA,yA,AB,phi,omega,alpha][[4]] ;

(* Acceleration of joint B *)
aBx = Driver[xA,yA,AB,phi,omega,alpha][[5]] ;
aBy = Driver[xA,yA,AB,phi,omega,alpha][[6]] ;

Print["rB = ",{xB,yB,0}," [m]"];
Print["vB = ",{vBx,vBy,0}," [m/s]"];
Print["aB = ",{aBx,aBy,0}," [m/s^2]"];

rB = {0.173205, 0.1, 0} [m]

vB = {-0.5, 0.866025, 0} [m/s]

aB = {-4.33013, -2.5, 0} [m/s^2]
```

```

BeginPackage["Position`"]

PosRRR::usage =
  "PosRRR[xM,yM,xN,yN,MP,NP] computes the position vectors for RRR dyad"

Begin["`Private`"]

PosRRR[xM_,yM_,xN_,yN_,MP_,NP_] :=
Block[

{ xPSol, yPSol, xP1, yP1, xP2, yP2, eqRRR1, eqRRR2, solRRR },

eqRRR1 = (xM-xPSol)^2 + (yM-yPSol)^2 == MP^2 ;
eqRRR2 = (xN-xPSol)^2 + (yN-yPSol)^2 == NP^2 ;

solRRR = Solve[ { eqRRR1 , eqRRR2 }, { xPSol, yPSol } ] ;

xP1 = xPSol/.solRRR[[1]] ;
yP1 = yPSol/.solRRR[[1]] ;
xP2 = xPSol/.solRRR[[2]] ;
yP2 = yPSol/.solRRR[[2]] ;

Return[ { xP1, yP1, xP2, yP2 } ] ;
]

End[ ] (* PosRRR *)

PosRRT::usage =
  "PosRRT[xM,yM,xN,yN,MP,theta] computes the position vectors for RRT \
dyad"

Begin["`Private`"]

PosRRT[xM_,yM_,xN_,yN_,MP_,theta_] :=
Block[

{ xPSol, yPSol, xP1, yP1, xP2, yP2, eqRRT, solRRT, eqRRT1, eqRRT2 },

If[ (theta==Pi/2) || (theta==3*Pi/2),

  xP1 = xP2 = xN ;
  eqRRT = (xM-xN)^2 + (yM-yPSol)^2 == MP^2 ;
  solRRT = Solve[ eqRRT, yPSol ] ;
  yP1 = yPSol/.solRRT[[1]] ;
  yP2 = yPSol/.solRRT[[2]] ,

  eqRRT1 = (xM-xPSol)^2 + (yM-yPSol)^2 == MP^2 ;
  eqRRT2 = Tan[theta] == (yPSol-yN)/(xPSol-xN) ;
  solRRT = Solve[ { eqRRT1 , eqRRT2 }, { xPSol , yPSol } ] ;
  xP1 = xPSol/.solRRT[[1]] ;
  yP1 = yPSol/.solRRT[[1]] ;
  xP2 = xPSol/.solRRT[[2]] ;
  yP2 = yPSol/.solRRT[[2]] ;

] ;

Return[ { xP1, yP1, xP2, yP2 } ] ;
]

End[ ] (* PosRRT *)

```

```
EndPackage[ ]
```

```
Position`
```

```
PosRRR[xM,yM,xN,yN,MP,NP] computes the position vectors for RRR dyad
```

```
Position`Private`
```

```
Position`Private`
```

```
PosRRT[xM,yM,xN,yN,MP,theta] computes the position vectors for RRT dyad
```

```
Position`Private`
```

```
Position`Private`
```

```
(* R-RTR-RRT mechanism *)

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

<<Driver.m ;
<<Position.m ;

(* Input data *)

AB = 0.20 ;      (* m *)
AD = 0.40 ;      (* m *)
CD = 0.70 ;      (* m *)
CE = 0.30 ;      (* m *)
yE = 0.35 ;      (* m *)
phi = Pi/4 ;     (* rad *)
omega = 5. ;     (* rad/s *)
alpha = 0. ;     (* rad/s^2 *)

(* Position Vectors *)

xA = yA = 0 ;

xB = Driver[xA,yA,AB,phi,omega,alpha][[1]] ;
yB = Driver[xA,yA,AB,phi,omega,alpha][[2]] ;

xD = 0 ;
yD = -AD ;

phi3 = ArcTan[(yB-yD)/(xB-xD)] ;

xC = xD + CD Cos[phi3] ;
yC = yD + CD Sin[phi3] ;

phi5 = Pi ;

(* xP=0; yP=yE; *)

xE1 = PosRRT[xC,yC,0,yE,CE,phi5][[1]] ;
xE2 = PosRRT[xC,yC,0,yE,CE,phi5][[3]] ;

(* Choose the correct solution *)
If[ (xE1<xC), xE=xE1, xE=xE2 ] ;

Print["rB = ",{xB,yB,0}," [m] "];
Print["phi3 = ",phi3*180/N[Pi]," [deg]"];
Print["rC = ",{xC,yC,0}," [m] "];
Print["rE = ",{xE,yE,0}," [m] "];

rB = {0.141421, 0.141421, 0} [m]

phi3 = 75.3612 [deg]

rC = {0.176907, 0.277277, 0} [m]

rE = {-0.114145, 0.35, 0} [m]
```

In[220]:=

```
(* R-RRR-RRT mechanism *)

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

<<Driver.m ;
<<Position.m ;

(* Input data *)

AB = 0.15 ;      (* m *)
BC = 0.40 ;      (* m *)
CD = 0.37 ;      (* m *)
CE = 0.23 ;      (* m *)
EF = CE ;        (* m *)
La = 0.30 ;      (* m *)
Lb = 0.45 ;      (* m *)
Lc = CD ;        (* m *)
phi = Pi/4 ;     (* rad *)
omega = N[Pi] ;  (* rad/s *)
alpha = 0. ;     (* rad/s^2 *)

(* Position Vectors *)

xA = yA = 0 ;

xB = Driver[xA,yA,AB,phi,omega,alpha][[1]] ;
yB = Driver[xA,yA,AB,phi,omega,alpha][[2]] ;

xD = La ;
yD = Lb ;

xC1 = PosRRR[xB,yB,xD,yD,BC,CD][[1]] ;
yC1 = PosRRR[xB,yB,xD,yD,BC,CD][[2]] ;
xC2 = PosRRR[xB,yB,xD,yD,BC,CD][[3]] ;
yC2 = PosRRR[xB,yB,xD,yD,BC,CD][[4]] ;

(* Choose the correct solution *)
If[ (yC1>yB), xC=xC1;yC=yC1, xC=xC2;yC=yC2 ] ;

phi3 = ArcTan[(yC-yD)/(xC-xD)] + Pi ;

xE = xC + CE Cos[phi3] ;
yE = yC + CE Sin[phi3] ;

xF = - Lc ;

phi5 = Pi/2 ;

yF1 = PosRRT[xE,yE,-Lc,0,EF,phi5][[2]] ;
yF2 = PosRRT[xE,yE,-Lc,0,EF,phi5][[4]] ;

(* Choose the correct solution *)
If[ (yF1<yE), yF=yF1, yF=yF2 ] ;

Print["Positions"];
Print["rB = ",{xB,yB,0}," [m] "];
Print["rC = ",{xC,yC,0}," [m] "];
Print["rE = ",{xE,yE,0}," [m] "];
Print["rF = ",{xF,yF,0}," [m] "];
```

```
phi2 = ArcTan[(yC-yB)/(xC-xB)] + Pi ;
```

```
phi4 = ArcTan[(yE-yF)/(xE-xF)] ;
```

```
Print["Angles"];  
Print["phi = ",phi," [rad]"];  
Print["phi2 = ",phi2," [rad]"];  
Print["phi3 = ",phi3," [rad]"];  
Print["phi4 = ",phi4," [rad]"];
```

Positions

```
rB = {0.106066, 0.106066, 0} [m]
```

```
rC = {-0.0696798, 0.46539, 0} [m]
```

```
rE = {-0.299481, 0.474956, 0} [m]
```

```
rF = {-0.37, 0.256034, 0} [m]
```

Angles

```
phi =  $\frac{\pi}{4}$  [rad]
```

```
phi2 = 2.02569 [rad]
```

```
phi3 = 3.09999 [rad]
```

```
phi4 = 1.25917 [rad]
```

```
(* R-RRT mechanism *)

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

<<Driver.m ;
<<Position.m ;

(* Input data *)

AC = 0.10 ;          (* m *)
BC = 0.30 ;          (* m *)
AP = 0.50 ;          (* m *)
phi = Pi/4 ;        (* rad *)
n = 30 ;             (* rpm *)
omega = N[Pi]*n/30 ; (* rad/s *)
alpha = 0. ;         (* rad/s^2 *)

(* Position Vectors *)

xA = yA = 0 ;

xC = AC ;
yC = 0 ;

xP = Driver[xA,yA,AP,phi,omega,alpha][[1]] ;
yP = Driver[xA,yA,AP,phi,omega,alpha][[2]] ;

xB1 = PosRRT[xC,yC,xP,yP,BC,phi][[1]] ;
yB1 = PosRRT[xC,yC,xP,yP,BC,phi][[2]] ;
xB2 = PosRRT[xC,yC,xP,yP,BC,phi][[3]] ;
yB2 = PosRRT[xC,yC,xP,yP,BC,phi][[4]] ;

(* Choose the correct solution *)
If[ (yB1>yC), xB=xB1;yB=yB1, xB=xB2;yB=yB2 ] ;

Print["rB = ",{xB,yB,0}," [m]"];

rB = {0.256155, 0.256155, 0} [m]
```

```

BeginPackage["VelAcc`"]

VelAccRRR::usage =
  "VelAccRRR[xM,yM,xN,yN,xP,yP,vMx,vMy,vNx,vNy,aMx,aMy,aNx,aNy] computes the velocity;

Begin["`Private`"]

VelAccRRR[xM_,yM_,xN_,yN_,xP_,yP_,vMx_,vMy_,vNx_,vNy_,aMx_,aMy_,aNx_,aNy_] :=
Block[

{ vPxSol, vPySol, aPxSol, aPySol, vPx, vPy, aPx, aPy,
eqRRR1v, eqRRR2v, solRRRv, eqRRR1a, eqRRR2a, solRRRa },

(* Velocity *)

eqRRR1v = (xM-xP) (vMx-vPxSol) + (yM-yP) (vMy-vPySol) == 0 ;
eqRRR2v = (xN-xP) (vNx-vPxSol) + (yN-yP) (vNy-vPySol) == 0 ;

solRRRv = Solve[ { eqRRR1v , eqRRR2v }, { vPxSol, vPySol } ] ;

vPx = vPxSol/.solRRRv[[1]] ;
vPy = vPySol/.solRRRv[[1]] ;

(* Acceleration *)

eqRRR1a = (xM-xP) (aMx-aPxSol) + (vMx-vPx)^2 + (yM-yP) (aMy-aPySol) + (vMy-vPy)^2 == 0 ;
eqRRR2a = (xN-xP) (aNx-aPxSol) + (vNx-vPx)^2 + (yN-yP) (aNy-aPySol) + (vNy-vPy)^2 == 0 ;

solRRRa = Solve[ { eqRRR1a , eqRRR2a }, { aPxSol, aPySol } ] ;

aPx = aPxSol/.solRRRa[[1]] ;
aPy = aPySol/.solRRRa[[1]] ;

Return[ { vPx, vPy, aPx, aPy } ] ;
]

End[ ] (* VelAccRRR *)

VelAccRRT::usage =
  "VelAccRRT[xM,yM,xN,yN,xP,yP,vMx,vMy,vNx,vNy,aMx,aMy,aNx,aNy,theta,dtheta,ddtheta]

Begin["`Private`"]

VelAccRRT[xM_,yM_,xN_,yN_,xP_,yP_,vMx_,vMy_,vNx_,vNy_,aMx_,aMy_,aNx_,aNy_,theta_,dtheta_,ddtheta_]
Block[

{ vPxSol, vPySol, aPxSol, aPySol, vPx, vPy, aPx, aPy,
eqRRTv, eqRRTa, eqRRT1v, eqRRT2v, solRRTv , eqRRR1a, eqRRR2a, solRRRa },

(* Velocity *)

eqRRT1v = (xM-xP) (vMx-vPxSol) + (yM-yP) (vMy-vPySol) == 0 ;
eqRRT2v = Sin[theta] (vPxSol-vNx) + Cos[theta] dtheta (xP-xN)
- Cos[theta] (vPySol-vNy) + Sin[theta] dtheta (yP-yN) == 0 ;

solRRTv = Solve[ { eqRRT1v , eqRRT2v }, { vPxSol , vPySol } ] ;

vPx = vPxSol/.solRRTv[[1]] ;
vPy = vPySol/.solRRTv[[1]] ;

```

```

(* Acceleration *)

eqRRT1a = (xM-xP) (aMx-aPxSol) + (vMx-vPx)^2 + (yM-yP) (aMy-aPySol) + (vMy-vPy)^2 == 0 ;
eqRRT2a =
+ Sin[theta] (aPxSol-aNx) - Cos[theta] (aPySol-aNy)
+ ( 2 Cos[theta] (vPx-vNx) - Sin[theta] dtheta (xP-xN)
+ 2 Sin[theta] (vPy-vNy) + Cos[theta] dtheta (yP-yN) ) dtheta
+ ( Cos[theta] (xP-xN) + Sin[theta] (yP-yN) ) ddtheta == 0 ;

solRRTa = Solve[ { eqRRT1a , eqRRT2a }, { aPxSol , aPySol } ] ;

aPx = aPxSol/.solRRTa[[1]] ;
aPy = aPySol/.solRRTa[[1]] ;

Return[ { vPx, vPy, aPx, aPy } ] ;
]

End[ ] (* VelAccRRT *)

AngVelAcc::usage =
  "AngVelAcc[xM,yM,xN,yN,vMx,vMy,vNx,vNy,aMx,aMy,aNx,aNy,theta] computes the angular

Begin["`Private`"]

AngVelAcc[xM_,yM_,xN_,yN_,vMx_,vMy_,vNx_,vNy_,aMx_,aMy_,aNx_,aNy_,theta_]:=
Block[

{ dtheta, ddtheta },

dtheta =
( Cos[theta] (vMy-vNy) - Sin[theta] (vMx-vNx) )
/ ( Sin[theta] (yM-yN) + Cos[theta] (xM-xN) ) ;

ddtheta =
( Cos[theta] (aMy-aNy) - Sin[theta] (aMx-aNx) -
( Cos[theta] dtheta (yM-yN) + 2 Sin[theta] (vMy-vNy)
- Sin[theta] dtheta (xM-xN) + 2 Cos[theta] (vMx-vNx) ) dtheta )
/ ( Cos[theta] (xM-xN) + Sin[theta] (yM-yN) ) ;

Return[ { dtheta, ddtheta } ] ;
]

End[ ] (* AngVelAcc *)

AbsVelAcc::usage =
  "AbsVelAcc[xM,yM,xN,yN,vMx,vMy,aMx,aMy,dtheta,ddtheta] computes the absolute veloc:

Begin["`Private`"]

AbsVelAcc[xM_,yM_,xN_,yN_,vMx_,vMy_,aMx_,aMy_,dtheta_,ddtheta_]:=
Block[

{ vNx, vNy, aNx, aNy },

vNx = vMx - dtheta (yN-yM) ;
vNy = vMy + dtheta (xN-xM) ;

aNx = aMx - ddtheta (yN-yM) - dtheta^2 (xN-xM) ;
aNy = aMy + ddtheta (xN-xM) - dtheta^2 (yN-yM) ;

Return[ { vNx, vNy, aNx, aNy } ] ;
]

```

```
End[ ] (* AbsVelAcc *)
```

```
EndPackage[ ]
```

```
VelAcc`
```

```
VelAccRRR[xM,yM,xN,yN,xP,yP,vMx,vMy,vNx,vNy,aMx,aMy,aNx,  
  aNy] computes the velocity and acceleration vectors for RRR dyad
```

```
VelAcc`Private`
```

```
VelAcc`Private`
```

```
VelAccRRT[xM,yM,xN,yN,xP,yP,vMx,vMy,vNx,vNy,aMx,aMy,aNx,aNy,theta,dtheta,  
  ddtheta] computes the velocity and acceleration vectors for RRT dyad
```

```
VelAcc`Private`
```

```
VelAcc`Private`
```

```
AngVelAcc[xM,yM,xN,yN,vMx,vMy,vNx,vNy,aMx,aMy,aNx,aNy,  
  theta] computes the angular velocity and acceleration of a link.
```

```
VelAcc`Private`
```

```
VelAcc`Private`
```

```
AbsVelAcc[xM,yM,xN,yN,vMx,vMy,aMx,aMy,dtheta,  
  ddtheta] computes the absolute velocity and acceleration vectors.
```

```
VelAcc`Private`
```

```
VelAcc`Private`
```

```

(* R-RRR-RRT mechanism *)

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

<<Driver.m ;
<<Position.m ;
<<VelAcc.m ;

(* Input data *)

AB = 0.15 ;          (* m *)
BC = 0.40 ;          (* m *)
CD = 0.37 ;          (* m *)
CE = 0.23 ;          (* m *)
EF = CE ;           (* m *)
La = 0.30 ;          (* m *)
Lb = 0.45 ;          (* m *)
Lc = CD ;            (* m *)
phi = Pi/4 ;         (* rad *)
omega = N[Pi]*100/30 ; (* rad/s *)
alpha = 0. ;          (* rad/s^2 *)

(* Position Vectors *)

xA = yA = 0 ;

xB = Driver[xA,yA,AB,phi,omega,alpha][[1]] ;
yB = Driver[xA,yA,AB,phi,omega,alpha][[2]] ;

xD = La ;
yD = Lb ;

xC1 = PosRRR[xB,yB,xD,yD,BC,CD][[1]] ;
yC1 = PosRRR[xB,yB,xD,yD,BC,CD][[2]] ;
xC2 = PosRRR[xB,yB,xD,yD,BC,CD][[3]] ;
yC2 = PosRRR[xB,yB,xD,yD,BC,CD][[4]] ;

(* Choose the correct solution *)
If[ (yC1>yB), xC=xC1;yC=yC1, xC=xC2;yC=yC2 ] ;

phi3 = ArcTan[(yC-yD)/(xC-xD)] + Pi ;

xE = xC + CE Cos[phi3] ;
yE = yC + CE Sin[phi3] ;

xF = - Lc ;

phi5 = Pi/2 ;

xP = - Lc ;
yP = 0 ;

yF1 = PosRRT[xE,yE,xP,yP,EF,phi5][[2]] ;
yF2 = PosRRT[xE,yE,xP,yP,EF,phi5][[4]] ;

(* Choose the correct solution *)
If[ (yF1<yE), yF=yF1, yF=yF2 ] ;

phi2 = ArcTan[(yC-yB)/(xC-xB)] + Pi ;

phi4 = ArcTan[(yE-yF)/(xE-xF)] ;

```

```

Print["Positions"];
Print["rB = ", {xB, yB, 0}, " [m]"];
Print["rC = ", {xC, yC, 0}, " [m]"];
Print["rE = ", {xE, yE, 0}, " [m]"];
Print["rF = ", {xF, yF, 0}, " [m]"];

Print["Angles"];
Print["phi = ", phi, " [rad]"];
Print["phi2 = ", phi2, " [rad]"];
Print["phi3 = ", phi3, " [rad]"];
Print["phi4 = ", phi4, " [rad]"];

(* Velocity and acceleration vectors *)

vBx = Driver[xA, yA, AB, phi, omega, alpha][[3]] ;
vBy = Driver[xA, yA, AB, phi, omega, alpha][[4]] ;
aBx = Driver[xA, yA, AB, phi, omega, alpha][[5]] ;
aBy = Driver[xA, yA, AB, phi, omega, alpha][[6]] ;

vDx = 0 ; vDy = 0 ;
aDx = 0 ; aDy = 0 ;

vCx = VelAccRRR[xB, yB, xD, yD, xC, yC, vBx, vBy, vDx, vDy, aBx, aBy, aDx, aDy][[1]] ;
vCy = VelAccRRR[xB, yB, xD, yD, xC, yC, vBx, vBy, vDx, vDy, aBx, aBy, aDx, aDy][[2]] ;
aCx = VelAccRRR[xB, yB, xD, yD, xC, yC, vBx, vBy, vDx, vDy, aBx, aBy, aDx, aDy][[3]] ;
aCy = VelAccRRR[xB, yB, xD, yD, xC, yC, vBx, vBy, vDx, vDy, aBx, aBy, aDx, aDy][[4]] ;

omega3 = AngVelAcc[xC, yC, xD, yD, vCx, vCy, vDx, vDy, aCx, aCy, aDx, aDy, phi3][[1]] ;

alpha3 = AngVelAcc[xC, yC, xD, yD, vCx, vCy, vDx, vDy, aCx, aCy, aDx, aDy, phi3][[2]] ;

vEx = AbsVelAcc[xD, yD, xE, yE, vDx, vDy, aDx, aDy, omega3, alpha3][[1]] ;
vEy = AbsVelAcc[xD, yD, xE, yE, vDx, vDy, aDx, aDy, omega3, alpha3][[2]] ;

aEx = AbsVelAcc[xD, yD, xE, yE, vDx, vDy, aDx, aDy, omega3, alpha3][[3]] ;
aEy = AbsVelAcc[xD, yD, xE, yE, vDx, vDy, aDx, aDy, omega3, alpha3][[4]] ;

vPx = vPy = 0 ;
aPx = aPy = 0 ;

omega5 = alpha5 = 0 ;

vFx =
VelAccRRR[xE, yE, xP, yP, xF, yF, vEx, vEy, vPx, vPy, aEx, aEy, aPx, aPy, phi5, omega5, alpha5][[1]] ;
vFy =
VelAccRRR[xE, yE, xP, yP, xF, yF, vEx, vEy, vPx, vPy, aEx, aEy, aPx, aPy, phi5, omega5, alpha5][[2]] ;
aFx =
VelAccRRR[xE, yE, xP, yP, xF, yF, vEx, vEy, vPx, vPy, aEx, aEy, aPx, aPy, phi5, omega5, alpha5][[3]] ;
aFy =
VelAccRRR[xE, yE, xP, yP, xF, yF, vEx, vEy, vPx, vPy, aEx, aEy, aPx, aPy, phi5, omega5, alpha5][[4]] ;

Print["Velocities"];
Print["vB = ", {vBx, vBy, 0}, " [m/s]"];
Print["vC = ", {vCx, vCy, 0}, " [m/s]"];
Print["vE = ", {vEx, vEy, 0}, " [m/s]"];
Print["vF = ", {vFx, vFy, 0}, " [m/s]"];

Print["Accelerations"];
Print["aB = ", {aBx, aBy, 0}, " [m/s^2]"];
Print["aC = ", {aCx, aCy, 0}, " [m/s^2]"];
Print["aE = ", {aEx, aEy, 0}, " [m/s^2]"];
Print["aF = ", {aFx, aFy, 0}, " [m/s^2]"];

omega2 = AngVelAcc[xB, yB, xC, yC, vBx, vBy, vCx, vCy, aBx, aBy, aCx, aCy, phi2][[1]] ;

```

```

alpha2 = AngVelAcc[xB,yB,xC,yC,vBx,vBy,vCx,vCy,aBx,aBy,aCx,aCy,phi2][[2]] ;
omega4 = AngVelAcc[xE,yE,xF,yF,vEx,vEy,vFx,vFy,aEx,aEy,aFx,aFy,phi4][[1]] ;
alpha4 = AngVelAcc[xE,yE,xF,yF,vEx,vEy,vFx,vFy,aEx,aEy,aFx,aFy,phi4][[2]] ;

Print["Angular velocities"];
Print["omega = ",{0,0,omega}," [rad/s]"];
Print["omega2 = ",{0,0,omega2}," [rad/s]"];
Print["omega3 = ",{0,0,omega3}," [rad/s]"];
Print["omega4 = ",{0,0,omega4}," [rad/s]"];

Print["Angular accelerations"];
Print["alpha = ",{0,0,alpha}," [rad/s^2]"];
Print["alpha2 = ",{0,0,alpha2}," [rad/s^2]"];
Print["alpha3 = ",{0,0,alpha3}," [rad/s^2]"];
Print["alpha4 = ",{0,0,alpha4}," [rad/s^2]"];

Positions

rB = {0.106066, 0.106066, 0} [m]
rC = {-0.0696798, 0.46539, 0} [m]
rE = {-0.299481, 0.474956, 0} [m]
rF = {-0.37, 0.256034, 0} [m]

Angles

phi =  $\frac{\pi}{4}$  [rad]
phi2 = 2.02569 [rad]
phi3 = 3.09999 [rad]
phi4 = 1.25917 [rad]

Velocities

vB = {-1.11072, 1.11072, 0} [m/s]
vC = {0.0702851, 1.68835, 0} [m/s]
vE = {0.113976, 2.73787, 0} [m/s]
vF = {0., 2.77458, 0} [m/s]

Accelerations

aB = {-11.6314, -11.6314, 0} [m/s^2]
aC = {7.42779, -7.11978, 0} [m/s^2]
aE = {12.0451, -11.5456, 0} [m/s^2]
aF = {0., -7.60013, 0} [m/s^2]

Angular velocities

omega = {0, 0, 10.472} [rad/s]

```

```
omega2 = {0, 0, -3.28675} [rad/s]
```

```
omega3 = {0, 0, -4.56707} [rad/s]
```

```
omega4 = {0, 0, -0.520622} [rad/s]
```

Angular accelerations

```
alpha = {0, 0, 0.} [rad/s^2]
```

```
alpha2 = {0, 0, -47.7584} [rad/s^2]
```

```
alpha3 = {0, 0, 18.391} [rad/s^2]
```

```
alpha4 = {0, 0, -55.1071} [rad/s^2]
```

```

(* R-RRT mechanism *)

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

<<Driver.m ;
<<Position.m ;
<<VelAcc.m ;

(* Input data *)

AC = 0.10 ;          (* m *)
BC = 0.30 ;          (* m *)
AP = 0.50 ;          (* m *)
phi = Pi/4 ;         (* rad *)
n = 30 ;             (* rpm *)
omega = N[Pi]*n/30 ; (* rad/s *)
alpha = 0. ;         (* rad/s^2 *)

(* Position Vectors *)

xA = yA = 0 ;

xC = AC ;
yC = 0 ;

xP = Driver[xA,yA,AP,phi,omega,alpha][[1]] ;
yP = Driver[xA,yA,AP,phi,omega,alpha][[2]] ;

xB1 = PosRRT[xC,yC,xP,yP,BC,phi][[1]] ;
yB1 = PosRRT[xC,yC,xP,yP,BC,phi][[2]] ;
xB2 = PosRRT[xC,yC,xP,yP,BC,phi][[3]] ;
yB2 = PosRRT[xC,yC,xP,yP,BC,phi][[4]] ;

(* Choose the correct solution *)
If[ (yB1>yC), xB=xB1;yB=yB1, xB=xB2;yB=yB2 ] ;

Print["Positions"];
Print["rB = ",{xB,yB,0}, " [m]"];

(* Velocity and acceleration vectors *)

vPx = Driver[xA,yA,AP,phi,omega,alpha][[3]] ;
vPy = Driver[xA,yA,AP,phi,omega,alpha][[4]] ;
aPx = Driver[xA,yA,AP,phi,omega,alpha][[5]] ;
aPy = Driver[xA,yA,AP,phi,omega,alpha][[6]] ;

vCx = 0 ; vCy = 0 ;
aCx = 0 ; aCy = 0 ;

vBx =
VelAccRRT[xC,yC,xP,yP,xB,yB,vCx,vCy,vPx,vPy,aCx,aCy,aPx,aPy,phi,omega,alpha][[1]] ;
vBy =
VelAccRRT[xC,yC,xP,yP,xB,yB,vCx,vCy,vPx,vPy,aCx,aCy,aPx,aPy,phi,omega,alpha][[2]] ;
aBx =
VelAccRRT[xC,yC,xP,yP,xB,yB,vCx,vCy,vPx,vPy,aCx,aCy,aPx,aPy,phi,omega,alpha][[3]] ;
aBy =
VelAccRRT[xC,yC,xP,yP,xB,yB,vCx,vCy,vPx,vPy,aCx,aCy,aPx,aPy,phi,omega,alpha][[4]] ;

Print["Velocities"];
Print["vB = ",{vBx,vBy,0}, " [m/s]"];

```

```
Print["Accelerations"];
Print["aB = ",{aBx,aBy,0}, " [m/s^2]"];

phi3 = ArcTan[(yB-yC)/(xB-xC)] ;

omega3 = AngVelAcc[xB,yB,xC,yC, vBx,vBy,vCx,vCy,aBx,aBy,aCx,aCy][[1]] ;

alpha3 = AngVelAcc[xB,yB,xC,yC, vBx,vBy,vCx,vCy,aBx,aBy,aCx,aCy][[2]] ;

Print["Angular velocities"];
Print["omega3 = ",{0,0,omega3}, " [rad/s]"];

Print["Angular accelerations"];
Print["alpha3 = ",{0,0,alpha3}, " [rad/s^2]"];

Positions

rB = {0.256155, 0.256155, 0} [m]

Velocities

vB = {-0.999913, 0.609559, 0} [m/s]

Accelerations

aB = {-1.80234, -4.25501, 0} [m/s^2]

Angular velocities

omega3 = {0, 0, 3.90354} [rad/s]

Angular accelerations

alpha3 = {0, 0, -2.25292} [rad/s^2]
```

```

(* R-RTR-RTR mechanism *)

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

<<Driver.m ;
<<VelAcc.m ;

(* Input data *)

AB = 0.14 ;          (* m *)
AC = 0.06 ;          (* m *)
AE = 0.25 ;          (* m *)
CD = 0.15 ;          (* m *)
phi = Pi/6 ;         (* rad *)
n = 50 ;             (* rpm *)
omega = n*N[Pi]/30 ; (* rad/s *)
alpha = 0. ;         (* rad/s^2 *)

(* Position Vectors *)

xA = yA = 0 ;

xB = Driver[xA,yA,AB,phi,omega,alpha][[1]] ;
yB = Driver[xA,yA,AB,phi,omega,alpha][[2]] ;

xC = 0 ;
yC = AC ;

phi3 = ArcTan[(yB-yC)/(xB-xC)] ;

xD = xC - CD Cos[phi3] ;
yD = yC - CD Sin[phi3] ;

xE = 0 ;
yE = -AE ;

phi5 = Pi + ArcTan[(yD-yE)/(xD-xE)] ;

(* Velocity and acceleration vectors *)

vBx = Driver[xA,yA,AB,phi,omega,alpha][[3]] ;
vBy = Driver[xA,yA,AB,phi,omega,alpha][[4]] ;
aBx = Driver[xA,yA,AB,phi,omega,alpha][[5]] ;
aBy = Driver[xA,yA,AB,phi,omega,alpha][[6]] ;

vCx = vCy = 0 ;
aCx = aCy = 0 ;

omega3 = AngVelAcc[xB,yB,xC,yC, vBx,vBy,vCx,vCy,aBx,aBy,aCx,aCy,phi3][[1]] ;
alpha3 = AngVelAcc[xB,yB,xC,yC, vBx,vBy,vCx,vCy,aBx,aBy,aCx,aCy,phi3][[2]] ;

vDx = AbsVelAcc[xC,yC,xD,yD,vCx,vCy,aCx,aCy,omega3,alpha3][[1]] ;
vDy = AbsVelAcc[xC,yC,xD,yD,vCx,vCy,aCx,aCy,omega3,alpha3][[2]] ;
aDx = AbsVelAcc[xC,yC,xD,yD,vCx,vCy,aCx,aCy,omega3,alpha3][[3]] ;
aDy = AbsVelAcc[xC,yC,xD,yD,vCx,vCy,aCx,aCy,omega3,alpha3][[4]] ;

vEx = vEy = 0 ;
aEx = aEy = 0 ;

omega5 = AngVelAcc[xD,yD,xE,yE, vDx,vDy,vEx,vEy,aDx,aDy,aEx,aEy,phi5][[1]] ;

```

```

alpha5 = AngVelAcc[xD,yD,xE,yE, vDx,vDy,vEx,vEy,aDx,aDy,aEx,aEy,phi5][[2]] ;

Print["Positions"];
Print["rB = ",{xB,yB,0}," [m]"];
Print["rD = ",{xD,yD,0}," [m]"];

Print["Velocities"];
Print["vB = ",{vBx,vBy,0}," [m/s]"];
Print["vD = ",{vDx,vDy,0}," [m/s]"];

Print["Accelerations"];
Print["aB = ",{aBx,aBy,0}," [m/s^2]"];
Print["aD = ",{aDx,aDy,0}," [m/s^2]"];

Print["Angles"];
Print["phi = ",phi," [rad]"];
Print["phi3 = ",phi3," [rad]"];
Print["phi5 = ",phi5," [rad]"];

Print["Angular velocities"];
Print["omega = ",{0,0,omega}," [rad/s]"];
Print["omega3 = ",{0,0,omega3}," [rad/s]"];
Print["omega5 = ",{0,0,omega5}," [rad/s]"];

Print["Angular accelerations"];
Print["alpha = ",{0,0,alpha}," [rad/s^2]"];
Print["alpha3 = ",{0,0,alpha3}," [rad/s^2]"];
Print["alpha5 = ",{0,0,alpha5}," [rad/s^2]"];

Positions

rB = {0.121244, 0.07, 0} [m]

rD = {-0.149492, 0.0476701, 0} [m]

Velocities

vB = {-0.366519, 0.63483, 0} [m/s]

vD = {0.0671766, -0.814473, 0} [m/s]

Accelerations

aB = {-3.32396, -1.91909, 0} [m/s^2]

aD = {4.61708, -1.81183, 0} [m/s^2]

Angles

phi =  $\frac{\pi}{6}$  [rad]

phi3 = 0.0822923 [rad]

phi5 = 2.03621 [rad]

Angular velocities

omega = {0, 0, 5.23599} [rad/s]

omega3 = {0, 0, 5.44826} [rad/s]

omega5 = {0, 0, 0.917134} [rad/s]

```

Angular accelerations

`alpha = {0, 0, 0.} [rad/s^2]`

`alpha3 = {0, 0, 14.5681} [rad/s^2]`

`alpha5 = {0, 0, -5.77155} [rad/s^2]`

```

BeginPackage["Force`"]

ForceMomentum::usage =
  "ForceMomentum[m,aCM,ICM,ddtheta] computes the total force and moment of a rigid l:

Begin["`Private`"]

ForceMomentum[m_,aCM_,ICM_,ddtheta_]:=
Block[

{ g, Fin, G, F, M },

g = 9.807 ;
Fin = - m aCM ;
G = { 0, -m g, 0 } ;
F = Fin + G ;
M = - ICM { 0, 0, ddtheta } ;

Return[ { F, M } ] ;
]

End[ ] (* ForceMomentum *)

ForceRRR::usage =
  "ForceRRR[F2,M2,F3,M3,rM,rN,rP,rC2,rC3] computes the joint reactions for the RRR d:

Begin["`Private`"]

ForceRRR[F2_,M2_,F3_,M3_,rM_,rN_,rP_,rC2_,rC3_]:=
Block[

{ F12, F12Sol, F12xSol, F12ySol, F43, F43Sol, F43xSol, F43ySol,
rPC2, rPC3, rPM, rPN, F32, eqRRR1, eqRRR2, eqRRR3, eqRRR4, solRRR },

F12Sol = { F12xSol, F12ySol, 0 } ;
F43Sol = { F43xSol, F43ySol, 0 } ;

rPC2 = rC2 - rP ;
rPC3 = rC3 - rP ;
rPM = rM - rP ;
rPN = rN - rP ;

eqRRR1 = (F12Sol+F43Sol+F2+F3)[[1]] == 0 ;
eqRRR2 = (F12Sol+F43Sol+F2+F3)[[2]] == 0 ;
eqRRR3 = (Cross[rPC2,F2]+Cross[rPM,F12Sol]+M2)[[3]] == 0 ;
eqRRR4 = (Cross[rPC3,F3]+Cross[rPN,F43Sol]+M3)[[3]] == 0 ;

solRRR = Solve[ {eqRRR1, eqRRR2, eqRRR3, eqRRR4},
{F12xSol,F12ySol,F43xSol,F43ySol} ] ;

F12 = F12Sol/.solRRR[[1]] ;
F43 = F43Sol/.solRRR[[1]] ;

F32 = - F2 - F12 ;

Return[ { F12, F43, F23 } ] ;
]

End[ ] (* ForceRRR *)

```

```

ForceRRT::usage =
    "ForceRRT[F2,M2,F3,M3,rM,rN,rP,rC2] computes the joint reactions for the RRR dyad."

Begin["`Private`"]

ForceRRT[F2_,M2_,F3_,M3_,rM_,rN_,rP_,rC2_] :=
Block[

{ F12, F12Sol, F12xSol, F12ySol, F43, F43Sol, F43xSol,
F43ySol, F32, eqRRT1, eqRRT2, eqRRT3, eqRRT4, solRRT, rNC2,
rNM, rNP, rNQ, rQP, rQ, rQSol, xQSol, yQSol, eqRRTQ1, eqRRTQ2, solRRTQ },

F12Sol = { F12xSol, F12ySol, 0 } ;
F43Sol = { F43xSol, F43ySol, 0 } ;

rPC2 = rC2 - rP ;
rPM = rM - rP ;
rPN = rN - rP ;

eqRRT1 = (F12Sol+F43Sol+F2+F3)[[1]] == 0 ;
eqRRT2 = (F12Sol+F43Sol+F2+F3)[[2]] == 0 ;
eqRRT3 = (F43Sol.rPN) == 0 ;
eqRRT4 = (Cross[rPC2,F2]+Cross[rPM,F12Sol]+M2)[[3]] == 0 ;

solRRT = Solve[ {eqRRT1, eqRRT2, eqRRT3, eqRRT4},
{F12xSol,F12ySol,F43xSol,F43ySol} ] ;

F12 = F12Sol/.solRRT[[1]] ;
F43 = F43Sol/.solRRT[[1]] ;

F32 = - F2 - F12 ;
F23 = - F32 ;

If[ M3[[3]]==0 , rQ = rP ,

rQSol = { xQSol, yQSol, 0 } ;
rPQ = rQSol - rP ;

eqRRTQ1 = rPQ[[2]]/rPQ[[1]] - rPN[[2]]/rPN[[1]] == 0 ;
eqRRTQ2 = (Cross[rPQ,F43]+M3)[[3]] == 0 ;
solRRTQ = Solve[ {eqRRTQ1, eqRRTQ2} , {xQSol,yQSol} ] ;
rQ = rQSol/.solRRTQ[[1]] ;
] ;

Return[ { F12, F43, F23, rQ } ] ;
]

End[ ] (* ForceRRT *)

ForceRTR::usage =
    "ForceRTR[F2,M2,F3,M3,rM,rP,rC2] computes the joint reactions for the RTR dyad."

Begin["`Private`"]

ForceRTR[F2_,M2_,F3_,M3_,rM_,rP_,rC2_] :=
Block[

{ rC3, F12, F12Sol, F12xSol, F12ySol, F43, F43Sol, F43xSol, F43ySol,
F32, eqRTR1, eqRTR2, eqRTR3, eqRTR4, solRTR, rMC2, rMP, rPQ, rQ,
rQSol, xQSol, yQSol, eqRTRQ1, eqRTRQ2, solRTRQ },

F12Sol = { F12xSol, F12ySol, 0 } ;
F43Sol = { F43xSol, F43ySol, 0 } ;

```

```

rMC2 = rC2 - rM ;
rMP = rP - rM ;

eqRTR1 = (F12Sol+F43Sol+F2+F3)[[1]] == 0 ;
eqRTR2 = (F12Sol+F43Sol+F2+F3)[[2]] == 0 ;
eqRTR3 = (F12Sol+F2).rMP == 0 ;
eqRTR4 = (Cross[rMC2,F2]+Cross[rMP,(F3+F43Sol)]+M2+M3)[[3]] == 0 ;

solRTR = Solve[ {eqRTR1, eqRTR2, eqRTR3, eqRTR4},
{F12xSol,F12ySol,F43xSol,F43ySol} ] ;

F12 = F12Sol/.solRTR[[1]] ;
F43 = F43Sol/.solRTR[[1]] ;

F32 = - F2 - F12 ;
F23 = - F32 ;

If[ M3[[3]]==0 , rQ = rP ,

rQSol = { xQSol, yQSol, 0 } ;
rMQ = rQSol - rM ;
rPQ = rQSol - rP ;

eqRTRQ1 = rMQ[[2]]/rMQ[[1]] - rMP[[2]]/rMP[[1]] == 0 ;
eqRTRQ2 = (Cross[rPQ,F23]+M3)[[3]] == 0 ;
solRTRQ = Solve[ {eqRTRQ1, eqRTRQ2} , {xQSol,yQSol} ] ;
rQ = rQSol/.solRTRQ[[1]] ;
] ;

Return[ { F12, F43, F23, rQ } ] ;
]

End[ ] (* ForceRTR *)

FMDriver::usage =
  "FMDriver[F1,M1,F21,rA,rB,rC1] computes the joint reaction and torque of the motor.

Begin["`Private`"]

FMDriver[F1_,M1_,F21_,rA_,rB_,rC1_] :=
Block[

{ F01, rAC1, rAB, Mm },

F01 = - F1 - F21 ;

rAC1 = rC1 - rA ;
rAB = rB - rA ;
Mm = - Cross[rAC1,F1] - Cross[rAB,F21] - M1 ;

Return[ { F01, Mm } ] ;
]

End[ ] (* FMDriver *)

EndPackage[ ]

Force`

ForceMomentum[m,aCM,ICM,ddtheta] computes the total force and moment of a rigid link.

```

Force`Private`

Force`Private`

ForceRRR[F2,M2,F3,M3,rM,rN,rP,rC2,rC3] computes the joint reactions for the RRR dyad.

Force`Private`

Force`Private`

ForceRRT[F2,M2,F3,M3,rM,rN,rP,rC2] computes the joint reactions for the RRR dyad.

Force`Private`

Force`Private`

ForceRTR[F2,M2,F3,M3,rM,rP,rC2] computes the joint reactions for the RTR dyad.

Force`Private`

Force`Private`

FMDriver[F1,M1,F21,rA,rB,rC1] computes the joint reaction and torque of the motor.

Force`Private`

Force`Private`

```

(* R-RRT mechanism *)

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

<<Driver.m ;
<<Position.m ;
<<VelAcc.m ;
<<Force.m ;

(* Input data *)

AC = 0.10 ;          (* m *)
BC = 0.30 ;          (* m *)
AP = 0.50 ;          (* m *)
wS = 0.05 ;          (* m *)
hS = 0.02 ;          (* m *)
phi = Pi/4 ;         (* rad *)
n = 30 ;             (* rpm *)
omega = N[Pi]*n/30 ; (* rad/s *)
alpha = 0. ;          (* rad/s^2 *)
g = 9.807 ;          (* m/s^2 *)
rho = 8000 ;          (* Kg/m^3 *)
d = 0.001 ;          (* m *)
h = 0.01 ;           (* m *)
Mext = {0, 0, -100} ; (* Nm *)

(* Position Vectors *)

xA = yA = 0 ;

xC = AC ;
yC = 0 ;

xP = Driver[xA,yA,AP,phi,omega,alpha][[1]] ;
yP = Driver[xA,yA,AP,phi,omega,alpha][[2]] ;

xB1 = PosRRT[xC,yC,xP,yP,BC,phi][[1]] ;
yB1 = PosRRT[xC,yC,xP,yP,BC,phi][[2]] ;
xB2 = PosRRT[xC,yC,xP,yP,BC,phi][[3]] ;
yB2 = PosRRT[xC,yC,xP,yP,BC,phi][[4]] ;

(* Choose the correct solution *)
If[ (yB1>yC), xB=xB1;yB=yB1, xB=xB2;yB=yB2 ] ;

Print["Positions"];
Print["rB = ",{xB,yB,0}," [m]"];

(* Velocity and acceleration vectors *)

vPx = Driver[xA,yA,AP,phi,omega,alpha][[3]] ;
vPy = Driver[xA,yA,AP,phi,omega,alpha][[4]] ;
aPx = Driver[xA,yA,AP,phi,omega,alpha][[5]] ;
aPy = Driver[xA,yA,AP,phi,omega,alpha][[6]] ;

vCx = 0 ; vCy = 0 ;
aCx = 0 ; aCy = 0 ;

vBx =
VelAccRRT[xC,yC,xP,yP,xB,yB,vCx,vCy,vPx,vPy,aCx,aCy,aPx,aPy,phi,omega,alpha][[1]] ;
vBy =
VelAccRRT[xC,yC,xP,yP,xB,yB,vCx,vCy,vPx,vPy,aCx,aCy,aPx,aPy,phi,omega,alpha][[2]] ;

```

```

aBx =
VelAccRRT[xC,yC,xP,yP,xB,yB,vCx,vCy,vPx,vPy,aCx,aCy,aPx,aPy,phi,omega,alpha][[3]] ;
aBy =
VelAccRRT[xC,yC,xP,yP,xB,yB,vCx,vCy,vPx,vPy,aCx,aCy,aPx,aPy,phi,omega,alpha][[4]] ;

Print["Velocities"];
Print["vB =",{vBx,vBy,0}," [m/s]"];

Print["Accelerations"];
Print["aB =",{aBx,aBy,0}," [m/s^2]"];

phi3 = ArcTan[(yB-yC)/(xB-xC)] ;

omega3 = AngVelAcc[xB,yB,xC,yC,vBx,vBy,vCx,vCy,aBx,aBy,aCx,aCy][[1]] ;

alpha3 = AngVelAcc[xB,yB,xC,yC,vBx,vBy,vCx,vCy,aBx,aBy,aCx,aCy][[2]] ;

Print["Angular velocities"];
Print["omega3 =",{0,0,omega3}," [rad/s]"];

Print["Angular accelerations"];
Print["alpha3 =",{0,0,alpha3}," [rad/s^2]"];

aA = { 0, 0, 0 } ;
aB = {aBx, aBy, 0 } ;
aC = {0, 0, 0 } ;
aP = {aPx, aPy, 0 } ;

(* Forces and moments *)

m1 = rho AP h d ;
aC1 = (aA+aP)/2 ;
IC1 = m1 (AP^2+h^2)/12 ;
F1 = ForceMomentum[m1,aC1,IC1,alpha][[1]] ;
M1 = ForceMomentum[m1,aC1,IC1,alpha][[2]] ;

m2 = rho wS hS d ;
aC2 = aB ;
IC2 = m2 (wS^2+hS^2)/12 ;
F2 = ForceMomentum[m2,aC2,IC2,alpha][[1]] ;
M2 = ForceMomentum[m2,aC2,IC2,alpha][[2]] ;

m3 = rho BC h d ;
aC3 = (aB+aC)/2 ;
IC3 = m1 (BC^2+h^2)/12 ;
F3 = ForceMomentum[m1,aC1,IC1,alpha3][[1]] ;
M3 = ForceMomentum[m1,aC1,IC1,alpha3][[2]] ;

Print["Forces and moments of links"];
Print["F1 = ",F1," [N]"];
Print["M1 = ",M1," [Nm]"];
Print["F2 = ",F2," [N]"];
Print["M2 = ",M2," [Nm]"];
Print["F3 = ",F3," [N]"];
Print["M3 = ",M3," [Nm]"];

rA = {xA, yA, 0} ;
rB = {xB, yB, 0} ;
rC = {xC, yC, 0} ;
rP = {xP, yP, 0} ;

rC3 = (rB+rC)/2 ;
F03 = ForceRRT[F3,M3+Mext,F2,M2,rC,rA,rB,rC3][[1]] ;
F12 = ForceRRT[F3,M3+Mext,F2,M2,rC,rA,rB,rC3][[2]] ;
F23 = ForceRRT[F3,M3+Mext,F2,M2,rC,rA,rB,rC3][[3]] ;

```

```
rQ = ForceRRT[F3,M3+Mext,F2,M2,rC,rA,rB,rC3][[4]] ;
```

```
F21 = - F12 ;
```

```
rC1 = (rA+rP)/2 ;
```

```
F01 = FMDriver[F1,M1,F21,rA,rB,rC1][[1]] ;
```

```
Mm = FMDriver[F1,M1,F21,rA,rB,rC1][[2]] ;
```

```
Print["Joint reactions"];
```

```
Print["F03 = ",F03," [N]"];
```

```
Print["F12 = ",F12," [N]"];
```

```
Print["F23 = ",F23," [N]"];
```

```
Print["rQ = ",rQ," [m]"];
```

```
Print["F01 = ",F01," [N]"];
```

```
Print["Mm = ",Mm," [N]"];
```

Positions

```
rB = {0.256155, 0.256155, 0} [m]
```

Velocities

```
vB = {-0.999913, 0.609559, 0} [m/s]
```

Accelerations

```
aB = {-1.80234, -4.25501, 0} [m/s^2]
```

Angular velocities

```
omega3 = {0, 0, 0.256155} [rad/s]
```

Angular accelerations

```
alpha3 = {0, 0, 0.256155} [rad/s^2]
```

Forces and moments of links

```
F1 = {0.0697886, -0.322491, 0} [N]
```

```
M1 = {0, 0, 0.} [Nm]
```

```
F2 = {0.0144187, -0.044416, 0} [N]
```

```
M2 = {0, 0, 0.} [Nm]
```

```
F3 = {0.0697886, -0.322491, 0} [N]
```

```
M3 = {0, 0, -0.000213548} [Nm]
```

Joint reactions

```
F03 = {242.56, -242.278, 0} [N]
```

```
F12 = {-242.645, 242.645, 0} [N]
```

```
F23 = {242.63, -242.6, 0} [N]
```

```
rQ = {0.256155, 0.256155, 0} [m]
```

```
F01 = {-242.714, 242.967, 0} [N]
```

```
Mm = {0., 0., 124.379} [N]
```

```

(* R-RTR-RTR mechanism *)

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

<<Driver.m ;
<<VelAcc.m ;
<<Force.m ;

(* Input data *)

AB = 0.14 ;          (* m *)
DF = 0.40 ;          (* m *)
EG = 0.50 ;          (* m *)
CD = 0.15 ;          (* m *)
AC = 0.06 ;          (* m *)
AE = 0.25 ;          (* m *)
wS = 0.05 ;          (* m *)
hS = 0.02 ;          (* m *)
phi = Pi/6 ;         (* rad *)
n = 50 ;             (* rpm *)
omega = n*Pi/30 ;    (* rad/s *)
alpha = 0. ;         (* rad/s^2 *)
g = 9.807 ;          (* m/s^2 *)
rho = 8000 ;         (* Kg/m^3 *)
d = 0.001 ;          (* m *)
h = 0.01 ;           (* m *)
Mext = {0, 0, -100} ; (* Nm *)

(* Position Vectors *)

xA = yA = 0 ;

xB = Driver[xA,yA,AB,phi,omega,alpha][[1]] ;
yB = Driver[xA,yA,AB,phi,omega,alpha][[2]] ;

xC = 0 ;
yC = AC ;

phi2 = phi3 = ArcTan[(yB-yC)/(xB-xC)] ;

xD = xC - CD Cos[phi3] ;
yD = yC - CD Sin[phi3] ;

xE = 0 ;
yE = -AE ;

xF = xC+(DF-CD)Cos[phi3] ;
yF = yC+(DF-CD)Sin[phi3] ;

xG = xE + EG Cos[phi5] ;
yG = yE + EG Sin[phi5] ;

(* Velocity and acceleration vectors *)

vBx = Driver[xA,yA,AB,phi,omega,alpha][[3]] ;
vBy = Driver[xA,yA,AB,phi,omega,alpha][[4]] ;
aBx = Driver[xA,yA,AB,phi,omega,alpha][[5]] ;
aBy = Driver[xA,yA,AB,phi,omega,alpha][[6]] ;

vCx = vCy = 0 ;
aCx = aCy = 0 ;

```

```

omega2 = omega3 = AngVelAcc[xB,yB,xC,yC, vBx,vBy,vCx,vCy,aBx,aBy,aCx,aCy,phi3][[1]] ;
alpha2 = alpha3 = AngVelAcc[xB,yB,xC,yC, vBx,vBy,vCx,vCy,aBx,aBy,aCx,aCy,phi3][[2]] ;

vDx = AbsVelAcc[xC,yC,xD,yD,vCx,vCy,aCx,aCy,omega3,alpha3][[1]] ;
vDy = AbsVelAcc[xC,yC,xD,yD,vCx,vCy,aCx,aCy,omega3,alpha3][[2]] ;
aDx = AbsVelAcc[xC,yC,xD,yD,vCx,vCy,aCx,aCy,omega3,alpha3][[3]] ;
aDy = AbsVelAcc[xC,yC,xD,yD,vCx,vCy,aCx,aCy,omega3,alpha3][[4]] ;

vEx = vEy = 0 ;
aEx = aEy = 0 ;

phi4 = phi5 = Pi + ArcTan[(yD-yE)/(xD-xE)] ;

omega4 = omega5 = AngVelAcc[xD,yD,xE,yE, vDx,vDy,vEx,vEy,aDx,aDy,aEx,aEy,phi5][[1]] ;
alpha4 = alpha5 = AngVelAcc[xD,yD,xE,yE, vDx,vDy,vEx,vEy,aDx,aDy,aEx,aEy,phi5][[2]] ;

aA = { 0, 0, 0 } ;
aB = { aBx, aBy, 0 } ;
aC = { aCx, aCy, 0 } ;
aD = { aDx, aDy, 0 } ;
aE = { aEx, aEy, 0 } ;

(* Forces and moments *)

m1 = rho AB h d ;
aC1 = (aA+aB)/2 ;
IC1 = m1 (AB^2+h^2)/12 ;
F1 = ForceMomentum[m1,aC1,IC1,alpha][[1]] ;
M1 = ForceMomentum[m1,aC1,IC1,alpha][[2]] ;

m2 = rho wS hS d ;
aC2 = aB ;
IC2 = m2 (wS^2+hS^2)/12 ;
F2 = ForceMomentum[m2,aC2,IC2,alpha2][[1]] ;
M2 = ForceMomentum[m2,aC2,IC2,alpha2][[2]] ;

m3 = rho DF h d ;
xC3 = xC+(DF/2-CD)Cos[phi3] ;
yC3 = yC+(DF/2-CD)Sin[phi3] ;
aC3x = AbsVelAcc[xC,yC,xC3,yC3,vCx,vCy,aCx,aCy,omega3,alpha3][[3]] ;
aC3y = AbsVelAcc[xC,yC,xC3,yC3,vCx,vCy,aCx,aCy,omega3,alpha3][[4]] ;
aC3 = {aC3x, aC3y, 0} ;
IC3 = m3 (DF^2+h^2)/12 ;
F3 = ForceMomentum[m3,aC3,IC3,alpha3][[1]] ;
M3 = ForceMomentum[m3,aC3,IC3,alpha3][[2]] ;

m4 = rho wS hS d ;
aC4 = aD ;
IC4 = m4 (wS^2+hS^2)/12 ;
F4 = ForceMomentum[m4,aC4,IC4,alpha4][[1]] ;
M4 = ForceMomentum[m4,aC4,IC4,alpha4][[2]] ;

m5 = rho EG h d ;
xC5 = xE + EG/2 Cos[phi5] ;
yC5 = yE + EG/2 Sin[phi5] ;
aC5x = AbsVelAcc[xE,yE,xC5,yC5,vEx,vEy,aEx,aEy,omega5,alpha5][[3]] ;
aC5y = AbsVelAcc[xE,yE,xC5,yC5,vEx,vEy,aEx,aEy,omega5,alpha5][[4]] ;
aC5 = {aC5x, aC5y, 0} ;
IC5 = m5 (EG^2+h^2)/12 ;
F5 = ForceMomentum[m5,aC5,IC5,alpha5][[1]] ;
M5 = ForceMomentum[m5,aC5,IC5,alpha5][[2]] ;

```

```

Print["Forces and moments of links"];
Print["F1 = ",F1," [N]"];
Print["M1 = ",M1," [Nm]"];
Print["F2 = ",F2," [N]"];
Print["M2 = ",M2," [Nm]"];
Print["F3 = ",F3," [N]"];
Print["M3 = ",M3," [Nm]"];
Print["F4 = ",F4," [N]"];
Print["M4 = ",M4," [Nm]"];
Print["F5 = ",F5," [N]"];
Print["M5 = ",M5," [Nm]"];

(* Joint reactions *)

rA = {xA, yA, 0} ;
rB = {xB, yB, 0} ;
rC = {xC, yC, 0} ;
rD = {xD, yD, 0} ;
rE = {xE, yE, 0} ;
rF = {xF, yF, 0} ;
rG = {xG, yG, 0} ;

rC5 = {xC5,yC5,0} ;
F05 = ForceRTR[F5,M5+Mext,F4,M4,rE,rD,rC5][[1]] ;
F34 = ForceRTR[F5,M5+Mext,F4,M4,rE,rD,rC5][[2]] ;
F45 = ForceRTR[F5,M5+Mext,F4,M4,rE,rD,rC5][[3]] ;
rP = ForceRTR[F5,M5+Mext,F4,M4,rE,rD,rC5][[4]] ;

F43 = - F34 ;
rC3 = {xC3,yC3,0} ;
rC3D = rD-rC3 ;
M43 = Cross[rC3D,F43] ;
F03 = ForceRTR[F3+F43,M3+M43,F2,M2,rC,rB,rC3][[1]] ;
F12 = ForceRTR[F3+F43,M3+M43,F2,M2,rC,rB,rC3][[2]] ;
F23 = ForceRTR[F3+F43,M3+M43,F2,M2,rC,rB,rC3][[3]] ;
rQ = ForceRTR[F3+F43,M3+M43,F2,M2,rC,rB,rC3][[4]] ;

F21 = - F12 ;
rC1 = (rA+rB)/2 ;
F01 = FMDriver[F1,M1,F21,rA,rB,rC1][[1]] ;
Mm = FMDriver[F1,M1,F21,rA,rB,rC1][[2]] ;

Print["Joint reactions"];
Print["F05 = ",F05," [N]"];
Print["F34 = ",F34," [N]"];
Print["F45 = ",F45," [N]"];
Print["rP = ",rP," [m]"];
Print["F03 = ",F03," [N]"];
Print["F12 = ",F12," [N]"];
Print["F23 = ",F23," [N]"];
Print["rQ = ",rQ," [m]"];
Print["F01 = ",F01," [N]"];
Print["Mm = ",Mm," [Nm]"];

Forces and moments of links

F1 = {0.0186142, -0.0990915, 0} [N]

M1 = {0, 0, 0.} [Nm]

F2 = {0.0265917, -0.0631033, 0} [N]

M2 = {0, 0, -0.000028165} [Nm]

```

```
F3 = {0.0492489, -0.33315, 0} [N]
M3 = {0, 0, -0.00621962} [Nm]
F4 = {-0.0369367, -0.0639614, 0} [N]
M4 = {0, 0, 0.0000111583} [Nm]
F5 = {-0.0553516, -0.410666, 0} [N]
M5 = {0, 0, 0.00481155} [Nm]
Joint reactions
F05 = {268.165, 135.057, 0} [N]
F34 = {-268.072, -134.583, 0} [N]
F45 = {268.109, 134.647, 0} [N]
rP = {-0.149492, 0.0476701, 0} [m]
F03 = {-256.745, -272.179, 0} [N]
F12 = {-11.4028, 137.993, 0} [N]
F23 = {11.3762, -137.93, 0} [N]
rQ = {0.121243, 0.07, 0} [m]
F01 = {-11.4214, 138.092, 0} [N]
Mm = {0., 0., 17.5356} [Nm]
```