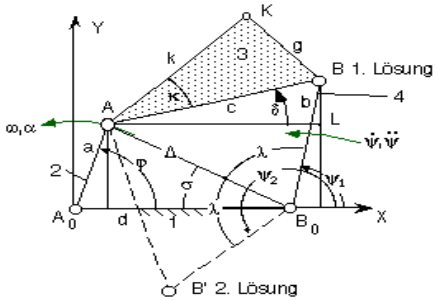
MDA-Chapter 3: Design of Mechanisms - Project WS2009-Problem2 - Solution

Munich Univ. of Applied Sciences - Prof. Dr. Oskar Wallrapp, FK06

created by Prof. O.Wallrapp - 12.12.2009
updated:

Crank-rocker mechanism: Basic functions




```
> restart;
  with(plots): with(plottools): with(StringTools):    # for plots
  and animation

> alias(Degree = Pi/180);
  1.0*Degree;
```

$$Degree$$

$$0.005555555556\,\pi$$

```
> X := d - a*cos(phi):
  Y := a * sin(phi):
  sigma := arctan(Y/X):
  Delta := sqrt(X^2 + Y^2):
  lambda := simplify(arccos((Delta^2 + b^2 - c^2) / (2 * b *
  Delta)) ):
> fpsi := Pi - (sigma + lambda);
```

$$fpsi := -\arctan\left(\frac{a\sin(\phi)}{d-a\cos(\phi)}\right)+\arccos\left(\frac{1}{2}\,\frac{-d^2+2\,d\,a\cos(\phi)-b^2+c^2-a^2}{b\sqrt{d^2-2\,d\,a\cos(\phi)+a^2}}\right)$$

```
> fpsi2 := Pi - (sigma - lambda);
```

$$fpsi2 := 2\,\pi-\arctan\left(\frac{a\sin(\phi)}{d-a\cos(\phi)}\right)-\arccos\left(\frac{1}{2}\,\frac{-d^2+2\,d\,a\cos(\phi)-b^2+c^2-a^2}{b\sqrt{d^2-2\,d\,a\cos(\phi)+a^2}}\right)$$

```
> fphii := Pi + arccos((d^2 + (c - a)^2 - b^2) / (2 * (c - a) *
  d)):
  fphia  := arccos((d^2 + (c + a)^2 - b^2) / (2 * (c + a) * d)):
  fphi0 := fphii - fphia;
```

$$fphi0 := \pi+\arccos\left(\frac{1}{2}\,\frac{d^2+(c-a)^2-b^2}{(c-a)\,d}\right)-\arccos\left(\frac{1}{2}\,\frac{d^2+(c+a)^2-b^2}{(c+a)\,d}\right)$$

```
> fpsii := Pi - arccos((d^2 + b^2 - (c - a)^2) / (2 * b * d)):
  fpsia := Pi - arccos((d^2 + b^2 - (c + a)^2) / (2 * b * d)):
  fpsi0 := fpsii - fpsia;
```

$$fpsi0 := -\arccos\left(\frac{1}{2}\,\frac{d^2+b^2-(c-a)^2}{b\,d}\right)+\arccos\left(\frac{1}{2}\,\frac{d^2+b^2-(c+a)^2}{b\,d}\right)$$

```
> mumin1:=arccos((c^2 + b^2 - (d - a)^2)/(2 * b * c));
```

$$mumin1 := \arccos\left(\frac{1}{2}\,\frac{c^2+b^2-(d-a)^2}{b\,c}\right)$$

```
> mumin2:= Pi-arccos((c^2 + b^2 - (d + a)^2)/(2 * b * c));
```

$$mumin2 := \pi-\arccos\left(\frac{1}{2}\,\frac{c^2+b^2-(d+a)^2}{b\,c}\right)$$

```
> mu := arccos((c^2+b^2-Delta_f^2)/(2*b*c)); fmu := subs(Delta_f
  = Delta, mu);
```

$$\mu := \arccos\left(\frac{1}{2}\,\frac{c^2+b^2-Delta\_f^2}{b\,c}\right)$$

$$fmu := \arccos\left(\frac{1}{2}\,\frac{c^2+b^2-(d-a\cos(\phi))^2-a^2\sin(\phi)^2}{b\,c}\right)$$

```
> Grashof:=min(a, b, c, d) + max(a, b, c, d);
```

$$Grashof := \min(a,b,c,d)+\max(a,b,c,d)$$

```
> fdelta := arctan((b * sin(psi) - a * sin(phi)), (d + b * cos
  (psi) - a * cos(phi)));
```

$$fdelta := \arctan(b\sin(\psi)-a\sin(\phi),\,d+b\cos(\psi)-a\cos(\phi))$$

```
> xK := A0x + a * cos(phi + gamma0) + k * cos(kappa + delta +
  gamma0);
  yK := A0y + a * sin(phi + gamma0) + k * sin(kappa + delta +
  gamma0);
```

$$xK := A0x+a\cos(\phi+\gamma0)+k\cos(\kappa+\delta+\gamma0)$$

$$yK := A0y+a\sin(\phi+\gamma0)+k\sin(\kappa+\delta+\gamma0)$$

2) Calculations based on the graphical results

From the graphic solution we have

```
> par_erg1 := [A0x=13.3, A0y=-159.3, gamma0=0, a = 40, b=100, c=
  100, d=80, k=200, kappa=0];
```

$$par\_erg1 := [A0x=13.3,\ A0y=-159.3,\ \gamma0=0,\ a=40,\ b=100,\ c=100,\ d=80,\ k=200,\ \kappa=0]$$

```
> evalf(subs(par_erg1, fpsi0/Degree));   # check of toggle angles
  psi0 and phi0
```

$$64.66706140$$

```
> evalf(subs(par_erg1, fphi0/Degree));
```

$$225.5846915$$

```
> evalf(subs(par_erg1, mumin1/Degree));
```

$$23.07391807$$

```
> mumin_erg1 := evalf(subs(par_erg1,mumin2/Degree));    # minimal
  transmission angle
```

$$mumin\_erg1 := 106.2602047$$

```
> phia_erg1 := evalf(subs(par_erg1, fphia));          # angles of
  toggle point phi a and phi i
  evalf(%/Degree);
```

$$phia\_erg1 := 0.7751933733$$

$$44.41530859$$

```
> phii_erg1 := evalf(subs(par_erg1, fphii));
  evalf(%/Degree);
```

$$phii\_erg1 := 4.712388981$$

$$270.0000000$$

```
> psia_erg1 := evalf(subs(par_erg1,phi=phia_erg1, fpsi)); #
  angles of toggle point psi a and psi i:
  evalf(%/Degree);
```

$$psia\_erg1 := 1.369438406$$

$$78.46304095$$

```
> psii_erg1 := evalf(subs(par_erg1,phi=phii_erg1, fpsii)); #
  angle of toggle point a:
  evalf(%/Degree);
```

$$psii\_erg1 := 2.498091545$$

$$143.1301023$$

```
> xKpar := evalf(subs(par_erg1, delta=fdelta, psi=fpsi, par_erg1,
  xK));
  yKpar := evalf(subs(par_erg1, delta=fdelta, psi=fpsi, par_erg1,
  yK));
```

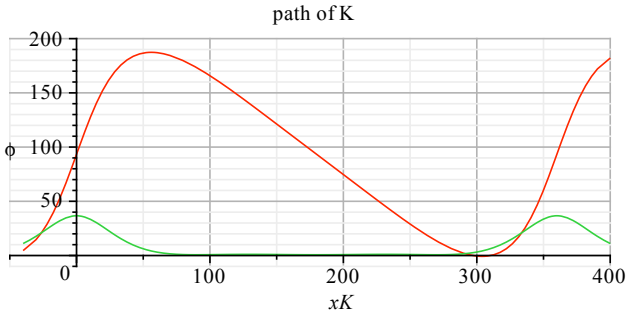
$$xKpar := 13.3 + 40.\cos(\phi) + 200.\cos\left(\arctan\left(100.\sin\left(-1.\arctan\left(\frac{40.\sin(\phi)}{80. - 40.\cos(\phi)}\right) + \arccos\left(\frac{0.005000000000\,(-8000. + 6400.\cos(\phi))}{\sqrt{8000. - 6400.\cos(\phi)}}\right)\right) - 40.\sin(\phi), 80. + 100.\cos\left(-1.\arctan\left(\frac{40.\sin(\phi)}{80. - 40.\cos(\phi)}\right) + \arccos\left(\frac{0.005000000000\,(-8000. + 6400.\cos(\phi))}{\sqrt{8000. - 6400.\cos(\phi)}}\right)\right) - 40.\cos(\phi)\right)\right)$$

$$yKpar := -159.3 + 40.\sin(\phi) + 200.\sin\left(\arctan\left(100.\sin\left(-1.\arctan\left(\frac{40.\sin(\phi)}{80. - 40.\cos(\phi)}\right) + \arccos\left(\frac{0.005000000000\,(-8000. + 6400.\cos(\phi))}{\sqrt{8000. - 6400.\cos(\phi)}}\right)\right) - 40.\sin(\phi), 80. + 100.\cos\left(-1.\arctan\left(\frac{40.\sin(\phi)}{80. - 40.\cos(\phi)}\right) + \arccos\left(\frac{0.005000000000\,(-8000. + 6400.\cos(\phi))}{\sqrt{8000. - 6400.\cos(\phi)}}\right)\right) - 40.\cos(\phi)\right)\right)$$

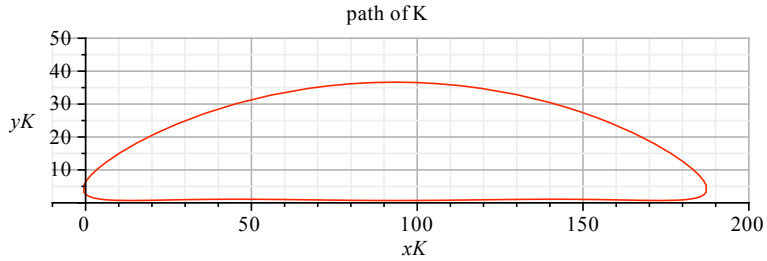
```
> evalf(subs(phi=3,xKpar));
```

$$100.8526246$$

```
> phi:= phiG*Degree:
  plot([xKpar,yKpar], phiG=-40..400, gridlines=true,labels=['xK',
  'phi'],title="path of K", view=[-50..400, -10..200]);
  phi := 'phi':
```




```
> plo_K := plot([xKpar,yKpar, phi=0..2*Pi],gridlines=true,labels=
  ['xK','yK'],title="path of K", view=[-10..200, 0..50],scaling =
  constrained):
  display(plo_K);
```




```
> Grashof:= min(subs(par_erg1, a), subs(par_erg1, b), subs
  (par_erg1, c), subs(par_erg1,d)) + max(subs(par_erg1, a), subs
  (par_erg1, b), subs(par_erg1, c), subs(par_erg1, d));
```

$$Grashof := 140$$

```
> evalf(subs(par_erg1,c + d));
```

We have a solution: full revolution of the crank
Transmission angle mu_min = 23 degrees - it is very small

Animation of crank-rocker-mechanism with par1(psi=fpsi)

```
> pa := par_erg1;
```

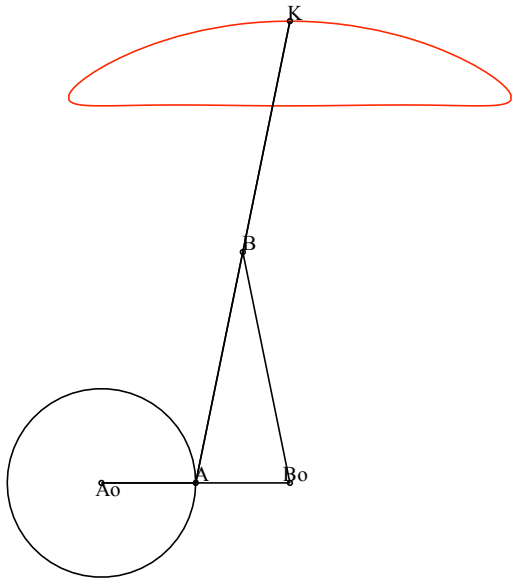
$pa := [A0x = 13.3, A0y = -159.3, \gamma 0 = 0, a = 40, b = 100, c = 100, d = 80, k = 200, \kappa = 0]$

```
> fpsii := fpsi:
  rA0x := subs(pa, A0x):
  rA0y := subs(pa, A0y):
  rB0x := subs(pa, A0x + d * cos(gamma0)):
  rB0y := subs(pa, A0y + d * sin(gamma0)):
  rax := subs(pa, A0x + a * cos(phi + gamma0)):
  ray := subs(pa, A0y + a * sin(phi + gamma0)):
  rbx := subs(psi=fpsii, pa, A0x + b * cos(psi + gamma0) + d *
  cos(gamma0)):
  rby := subs(psi=fpsii, pa, A0y + b * sin(psi + gamma0) + d *
  sin(gamma0)):
  rkx := subs(delta = fdelta, psi = fpsii, pa, xK):
  rky := subs(delta = fdelta, psi = fpsii, pa, yK):
  plo_K1 := plot([xKpar,yKpar, phi=0..2*Pi], view=[-10..200,
  -200..50],scaling = constrained,gridlines=true):
  lmax :=  max(subs(pa,a), subs(pa,b), subs(pa,c), subs(pa,d)):

> data:='data':
  data:=[]:
  framecount:=30:
  for i from 0 to framecount do
        phi:=evalf(2*Pi*i/framecount,2):
        textk := Join( ["phi    = ",convert(evalf(phi      *
  180/Pi,4), string)," "]);
        data:=[op(data),
               display([
                     plo_K1,
                textplot([0, lmax/2, textk]),
                       circle([rA0x, rA0y],subs(pa,a)),
                       circle([rA0x, rA0y],lmax/100),
                       circle([rax, ray],lmax/100),
                       circle([rbx, rby],lmax/100),
                       circle([rkx, rky],lmax/100),
                       circle([rB0x, rB0y],lmax/100),
                line([rA0x, rA0y], [rB0x,rB0y]),
                line([rA0x, rA0y], [rax,ray]),
                line([rB0x, rB0y], [rbx,rby]),
                line([rax, ray], [rbx, rby]),
                line([rbx, rby], [rkx, rky]),
                line([rkx, rky], [rax, ray]),
                       textplot([rA0x+lmax/30,rA0y-
  lmax/30, "Ao"]),
                       textplot([rax+lmax/30,ray+lmax/30,
  "A"]),
                       textplot([rbx+lmax/30,rby+lmax/30,
  "B"]),
                       textplot([rkx+lmax/30,rky+lmax/30,
  "K"]),
                       textplot([rB0x+lmax/30,rB0y+
  lmax/30, "Bo"])
                ])]:
  od :
  display(data, insequence=true,scaling=constrained,axes=none,
  title="Motion of c.r.mecha par set 1");
  phi:='phi': i:='i':
```

Motion of c.r.mecha par set 1

phi = 0.



Check of the point of K

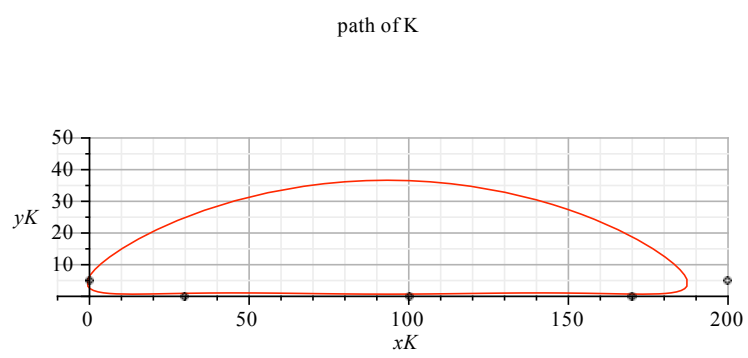
```
> val_xK := [0, 30, 100, 170, 200 ];      # xK mm
```

$val\_xK := [0, 30, 100, 170, 200]$

```
> val_yK := [5, 0, 0, 0, 5];     # yK mm
```

$val\_yK := [5, 0, 0, 0, 5]$

```
> display(plo_K,
    point([val_xK[1], val_yK[1]],lmax/50),
    point([val_xK[2], val_yK[2]],lmax/100),
    point([val_xK[3], val_yK[3]],lmax/100),
    point([val_xK[4], val_yK[4]],lmax/100),
    point([val_xK[5], val_yK[5]],lmax/100),
  gridlines=true);
```



There are some errors at point 5.

2 Find a mechanism for a 5 given points of K

Given is
function fpsi(phi , par) , where par = {a=S, b=c=2.5*S, k=5*S, d=2*S, A0x, A0x, gamma0, kappa=0}
parReq := {a=S, b=c=2.5*S, k=5*S, d=2*S, kappa=0};
Values of xK and yK

```
> par_erg1;
```

$$[A0x = 13.3,\ A0y = -159.3,\ \gamma0 = 0,\ a = 40,\ b = 100,\ c = 100,\ d = 80,\ k = 200,\ \kappa = 0]$$

```
> parReq := {a=S, b=2.5*S, c=2.5*S, k=5*S, d=2*S,  kappa=0,
  gamma0=0};
```

$$parReq := \{a = S,\ b = 2.5\,S,\ c = 2.5\,S,\ d = 2\,S,\ \gamma0 = 0,\ k = 5\,S,\ \kappa = 0\}$$

```
> evalm(val_xK);
```

$$[\,0\ \ 30\ \ 100\ \ 170\ \ 200\,]$$

```
> evalm(val_yK);
```

$$[\,5\ \ 0\ \ 0\ \ 0\ \ 5\,]$$

```
> val_phi := [phi1, phi2, phi3, phi4, phi5];
```

$$val\_phi := [\phi1, \phi2, \phi3, \phi4, \phi5]$$

We setup j = 1..nfct requirements to satisfy psi(phi(j), a, b, c) = val_psi(j)(eps) , where nfct = 5

```
> xKpar := evalf(subs(delta=fdelta, psi=fpsi, parReq, xK));
```

$$xKpar := A0x + S\cos(\phi) + 5.\,S\cos\left(\arctan\left(2.5\,S\sin\left(-1.\arctan\left(\frac{S\sin(\phi)}{2.\,S - 1.\,S\cos(\phi)}\right)\right.\right.\right.$$
$$\left.\left.\left. + \arccos\left(\frac{0.2000000000\,\left(-5.00\,S^2 + 4.\,S^2\cos(\phi)\right)}{S\sqrt{5.\,S^2 - 4.\,S^2\cos(\phi)}}\right)\right)\right) - 1.\,S\sin(\phi),\ 2.\,S + 2.5\,S\cos\left(\right.$$
$$-1.\arctan\left(\frac{S\sin(\phi)}{2.\,S - 1.\,S\cos(\phi)}\right) + \arccos\left(\frac{0.2000000000\,\left(-5.00\,S^2 + 4.\,S^2\cos(\phi)\right)}{S\sqrt{5.\,S^2 - 4.\,S^2\cos(\phi)}}\right)\Bigg)$$
$$- 1.\,S\cos(\phi)\Bigg)\Bigg)$$

```
> yKpar := evalf(subs(delta=fdelta, psi=fpsi, parReq, yK));
```

$$yKpar := A0y + S\sin(\phi) + 5.\,S\sin\left(\arctan\left(2.5\,S\sin\left(-1.\arctan\left(\frac{S\sin(\phi)}{2.\,S - 1.\,S\cos(\phi)}\right)\right.\right.\right.$$
$$\left.\left.\left. + \arccos\left(\frac{0.2000000000\,\left(-5.00\,S^2 + 4.\,S^2\cos(\phi)\right)}{S\sqrt{5.\,S^2 - 4.\,S^2\cos(\phi)}}\right)\right)\right) - 1.\,S\sin(\phi),\ 2.\,S + 2.5\,S\cos\left(\right.$$
$$-1.\arctan\left(\frac{S\sin(\phi)}{2.\,S - 1.\,S\cos(\phi)}\right) + \arccos\left(\frac{0.2000000000\,\left(-5.00\,S^2 + 4.\,S^2\cos(\phi)\right)}{S\sqrt{5.\,S^2 - 4.\,S^2\cos(\phi)}}\right)\Bigg)$$
$$- 1.\,S\cos(\phi)\Bigg)\Bigg)$$

```
> nfct := nops(val_yK);
```

$$nfct := 5$$

```
> Rq := [
    subs(phi=(val_phi[1]), xKpar),
    subs(phi=(val_phi[2]), xKpar),
    subs(phi=(val_phi[3]), xKpar),
    subs(phi=(val_phi[4]), xKpar),
    subs(phi=(val_phi[5]), xKpar),
    subs(phi=(val_phi[1]), yKpar),
    subs(phi=(val_phi[2]), yKpar),
    subs(phi=(val_phi[3]), yKpar),
    subs(phi=(val_phi[4]), yKpar),
    subs(phi=(val_phi[5]), yKpar)
  ]:
```

```
> evalf(subs(par_erg1[1], S=40, phi1=300*Degree,  Rq[1]=val_xK[1]
  ));  # test
  evalf(subs(par_erg1[2], S=40, phi1=300*Degree,  Rq[6]=val_yK[1]
  ));  # test
```

$$-0.50831523 = 0.$$
$$3.1807681 = 5.$$

```
> evalf(subs(par_erg1[1], S=40, phi2=250*Degree,  Rq[2]=val_xK[2]
  ));  # test
  evalf(subs(par_erg1[2], S=40, phi2=250*Degree,  Rq[7]=val_yK[2]
  ));  # test
```

$$29.00601682 = 30.$$
$$0.9415515 = 0.$$

```
> evalf(subs(par_erg1[1], S=40, phi3=170*Degree,  Rq[3]=val_xK[3]
```

```
));  # test
evalf(subs(par_erg1[2], S=40, phi3=170*Degree,  Rq[8]=val_yK[3]
));  # test
```

$$102.6102921 = 100.$$

$$0.7329618 = 0.$$

```
> evalf(subs(par_erg1[1], S=40, phi4=95*Degree,  Rq[4]=val_xK[4])
);  # test
evalf(subs(par_erg1[2], S=40, phi4=95*Degree,  Rq[9]=val_yK[4])
);  # test
```

$$169.6800114 = 170.$$

$$0.7259263 = 0.$$

Let's try an optimization!

```
> ferror := 0:
  for i from 1 to 5 do
  ferror := ferror + (Rq[i]   - evalm(val_xK[i]))^2:
  ferror := ferror + (Rq[i+5] - evalm(val_yK[i]))^2:
  od:
  ferror:
```

```
> pinit := {par_erg1[1], par_erg1[2], S=40, phi1=300*Degree,
  phi2=250*Degree, phi3=170*Degree, phi4=95*Degree, phi5=57*
  Degree };
> evalf(subs(pinit, ferror));
```

$$pinit := \left\{A0x = 13.3, A0y = -159.3, S = 40, \phi1 = \frac{5}{3}\pi, \phi2 = \frac{25}{18}\pi, \phi3 = \frac{17}{18}\pi, \phi4 = \frac{19}{36}\pi, \phi5 = \frac{19}{60}\pi\right\}$$

$$174.4923212$$

```
> with(Optimization):
  parOpt := Minimize(ferror,initialpoint = pinit, assume =
  nonnegative );
Warning, undefined value encountered
Error, (in Optimization:-NLPSolve) number expected for float[8]
parameter, got Float(undefined)+Float(undefined)*I
```

Solution by Mathematica

```
> parOpt := [S=42.665209406524674, phi1=5.37937, phi2=4.39077,
  phi3=3.14159, phi4=1.89242, phi5=0.90382,
   A0x = 14.6696, A0y = -170.816, gamma0=0, a = 42.6652, b =
  106.663, c = 106.663, d = 85.3304, k = 213.326, kappa=0];
```

$$parOpt := [S = 42.665209406524674, \phi1 = 5.37937, \phi2 = 4.39077, \phi3 = 3.14159, \phi4 = 1.89242, \phi5 = 0.90382, A0x = 14.6696, A0y = -170.816, \gamma0 = 0, a = 42.6652, b = 106.663, c = 106.663, d = 85.3304, k = 213.326, \kappa = 0]$$

```
> par_erg1;
```

$$[A0x = 13.3, A0y = -159.3, \gamma0 = 0, a = 40, b = 100, c = 100, d = 80, k = 200, \kappa = 0]$$

This is a better solution.

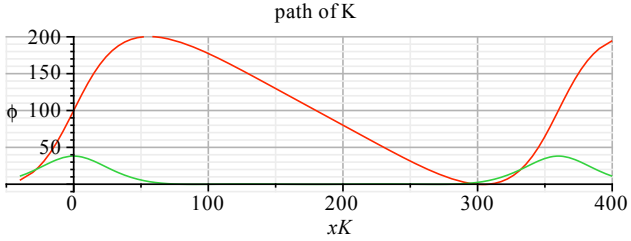
```
> xK;
```

$$A0x + a\cos(\phi + \gamma0) + k\cos(\kappa + \delta + \gamma0)$$

```
> xKpar := evalf(subs(delta=fdelta, psi=fpsi, parOpt, xK)):
  yKpar := evalf(subs(delta=fdelta, psi=fpsi, parOpt, yK)):
> evalf(subs(phi=1,yKpar));
```

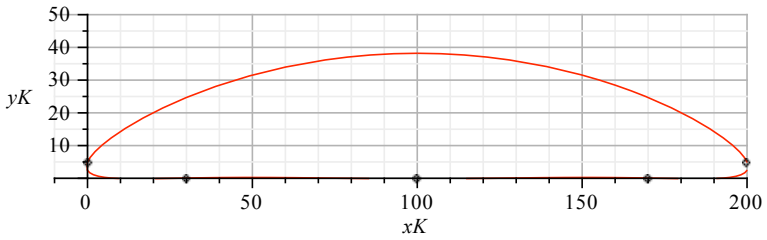
$$3.1910089$$

```
> phi:= phiG*Degree:
  plot([xKpar,yKpar], phiG=-40..400, gridlines=true,labels=['xK',
  'phi'],title="path of K", view=[-50..400, -10..200]);
  phi := 'phi':
```




```
> plo_K := plot([xKpar,yKpar, phi=0..2*Pi],gridlines=true,labels=
  ['xK','yK'],title="path of K after Optim", view=[-10..200, 0.
  .50],scaling = constrained):
  display(plo_K):
> display(plo_K,
    point([val_xK[1], val_yK[1]],lmax/50),
    point([val_xK[2], val_yK[2]],lmax/100),
    point([val_xK[3], val_yK[3]],lmax/100),
    point([val_xK[4], val_yK[4]],lmax/100),
    point([val_xK[5], val_yK[5]],lmax/100),
  gridlines=true);
```

path of K after Optim




```
> Grashof:= min(subs(parOpt, a), subs(parOpt, b), subs(parOpt,
  c), subs(parOpt,d)) + max(subs(parOpt, a), subs(parOpt, b),
  subs(parOpt, c), subs(parOpt, d));
```

Grashof := 149.3282

```
> evalf(subs(parOpt,c + d));
```

191.9934

```
> evalf(subs(parOpt, fpsi0/Degree));   # check of toggle angles
  psi0 and phi0
```

64.66706132

```
> evalf(subs(parOpt, fphi0/Degree));
```

225.5846913

```
> evalf(subs(parOpt, mumin1/Degree));
```

23.07391807

```
> mumin_erg1 := evalf(subs(parOpt,mumin2/Degree));   # minimal
  transmission angle
```

mumin_erg1 := 106.2602047

```
> phia_erg1 := evalf(subs(parOpt, fphia));          # angles of
  toggle point phi a and phi i
  evalf(%/Degree);
```

phia_erg1 := 0.7751933737

44.41530862

```
> phii_erg1 := evalf(subs(parOpt, fphii));
  evalf(%/Degree);
```

phii_erg1 := 4.712388981

270.0000000

```
> psia_erg1 := evalf(subs(parOpt,phi=phia_erg1, fpsi)); # angles
  of toggle point psi a and psi i:
  evalf(%/Degree);
```

psia_erg1 := 1.369438406

78.46304095

```
> psii_erg1 := evalf(subs(parOpt,phi=phii_erg1, fpsii)); # angle
  of toggle point a:
  evalf(%/Degree);
```

psii_erg1 := 2.498091544

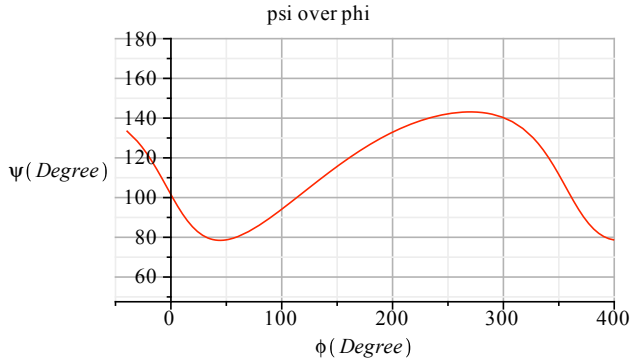
143.1301023

3 Kinematic Analysis

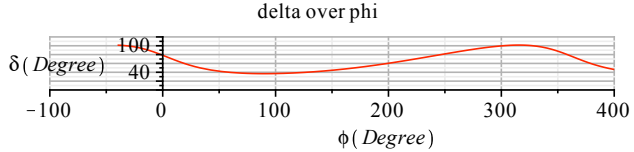
```
> par := parOpt;
```

par := [*S* = 42.665209406524674, *ϕ1* = 5.37937, *ϕ2* = 4.39077, *ϕ3* = 3.14159, *ϕ4* = 1.89242, *ϕ5* = 0.90382, *A0x* = 14.6696, *A0y* = −170.816, *γ0* = 0, *a* = 42.6652, *b* = 106.663, *c* = 106.663, *d* = 85.3304, *k* = 213.326, *κ* = 0]

```
> phi:= phiG*Degree:
  plo_psi1 := plot(subs(par, fpsi)/Degree, phiG=-40..400,
  gridlines=true,labels=['phi(Degree)','psi(Degree)'],title="psi
  over phi", view=[-50..400, 50..180]):
  phi := 'phi':
  display(plo_psi1);
```



```
> phi:= phiG*Degree:
  plot(subs(psi=fpsi, par, fdelta)/Degree,phiG=-40..400,
  gridlines=true,labels=['phi(Degree)','delta(Degree)'],title=
  "delta over phi",view=[-100..400, 0..120]);
  phi := 'phi':
```




```
> mumin_erg := evalf(subs(par,mumin2/Degree))* degrees;
```

$$mumin\_erg := 106.2602047\ degrees$$

```
> phi0_erg := evalf(subs(par,fphi0/Degree))* degrees;
```

$$phi0\_erg := 225.5846913\ degrees$$

```
> psi0_erg := evalf(subs(par,fpsi0/Degree))* degrees;
```

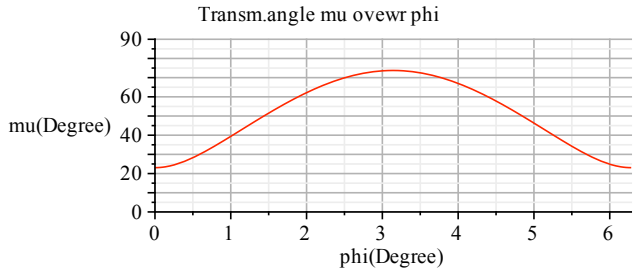
$$psi0\_erg := 64.66706132\ degrees$$

```
> mu_f2 := proc(phi)
          global par;
          if (evalf(subs(par,arccos((1/2)*(c^2+b^2-d^2+2*d*a*cos
  (phi)-a^2)/(b*c)))) > evalf(Pi/2)) then subs(par, Pi - arccos(
  (1/2)*(c^2+b^2-d^2+2*d*a*cos(phi)-a^2)/(b*c))) else subs(par,
  arccos((1/2)*(c^2+b^2-d^2+2*d*a*cos(phi)-a^2)/(b*c))) end if;
  end proc:

> evalf(mu_f2(0));      # Test for phi = 0
```

$$0.4027158416$$

```
> # Note: ' proc '  allows to evaluate the proc containing if
  statements in the plot !!
>
  plot('mu_f2(phi)'/Degree, phi =0..2*Pi, labels=["phi(Degree)",
  "mu(Degree)"],title="Transm.angle mu ovewr phi",gridlines=true,
  view=[0..2*Pi,0..90]);
  phi := 'phi':
```



4 Linear velocity vK (phi, omega=phip=const) of points K / H

we want to have a average velocity at point K3 of 200 mm/s

```
> omega := 'omega';
```

$$\omega := \omega$$

```
> vKx := diff(xKpar, phi)*omega:
  vKy := diff(yKpar, phi)*omega:
  evalf(subs(phi=phi3, parOpt, vKx));
> omega_par := solve(%=200, omega);
```

$$-56.88693334\ \omega$$

$$omega\_par := -3.515745853$$

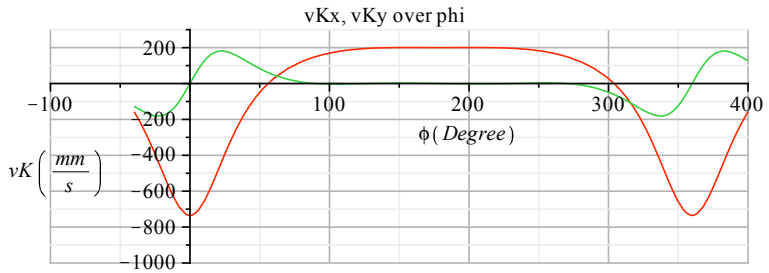
```
> evalf(subs(phi=phi3, parOpt, omega=omega_par, vKx));
```

$$199.9999999$$

```
> evalf(subs(phi=phi3, parOpt, omega=omega_par, vKy));
```

$$0.00002214726203$$

```
> phi:= phiG*Degree:
  plot([subs(omega=omega_par, vKx), subs(omega=omega_par, vKy)],
  phiG=-40..400,gridlines=true,labels=['phi(Degree)','vK (mm/s)
  '],title="vKx, vKy over phi",view=[-100..400, -1000..300]);
  phi := 'phi':
```

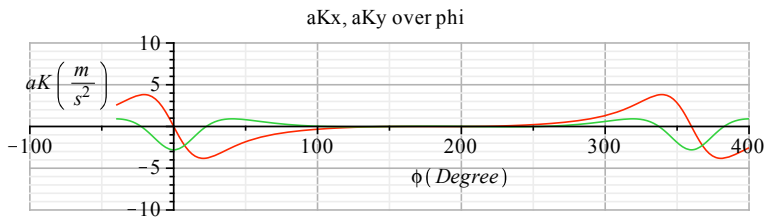



```
> evalf(subs(phi=0, parOpt, omega=omega_par, vKx));
```

$$-734.8469221$$

The acceleration

```
> aKx := diff(vKx, phi) * omega/1000:
  aKy := diff(vKy, phi) * omega/1000:
> phi:= phiG*Degree:
  plot([subs(omega=omega_par, aKx), subs(omega=omega_par, aKy)],
  phiG=-40..400,gridlines=true,labels=['phi(Degree)','aK (m/s^2)
  '],title="aKx, aKy over phi",view=[-100..400, -10..10]);
  phi := 'phi':
```



5 Input torque Min to drive the machine

sum of power = 0 = + Min omega - (Tx*vCMx + Ty*vCMy + Fg*vCMy)
herein, vCM = vK, aCM = aK
for vK in mm/s we get Nmm

```
> par_dyn := {mcutter = 1.5, g=9.81 };
  Tx := mcutter * aCMx;
  Ty := mcutter * aCMx;
  Fg := mcutter * g;
```

$$par\_dyn := \{g = 9.81, mcutter = 1.5\}$$

$$Tx := mcutter\ aCMx$$

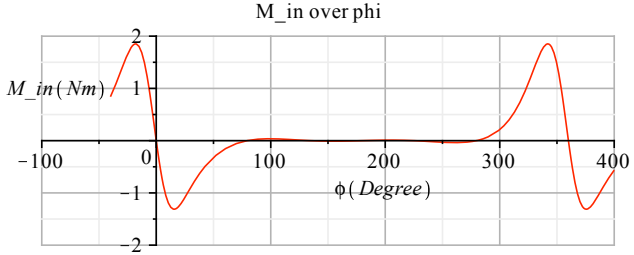
$$Ty := mcutter\ aCMx$$

$$Fg := mcutter\ g$$

```
> M_in := (Tx*vCMx + Ty*vCMy + Fg*vCMy) / omega ;
```

$$M\_in := \frac{mcutter\ aCMx\ vCMx + mcutter\ aCMx\ vCMy + mcutter\ g\ vCMy}{\omega}$$

```
> M_inpar := subs(par_dyn, vCMx=vKx, vCMy=vKy, aCMx=aKx, aCMy=
  aKy, omega=omega_par,  M_in/1000):   # to get torque in Nm
> phi:= phiG*Degree:
  plot(M_inpar, phiG=-40..400,gridlines=true,labels=['phi(Degree)
  ','M_in(Nm)'],title="M_in over phi",view=[-100..400, -2..2]);
  phi := 'phi':
```




```
> evalf(subs(phi=0, M_inpar));
```

$$-2.2\ 10^{-9}$$

```
> evalf(subs(phi=phi3, parOpt, M_inpar));
```

$$-9.907926756\ 10^{-8}$$

```
> diff(M_inpar, phi):
  erg := fsolve( %=0, phi=20*Pi/180.);
  Min_max := evalf(subs(phi=erg,  M_inpar));
```

$$erg := 0.2724680528$$

$$Min\_max := -1.312333077$$

```
> diff(M_inpar, phi):
  erg := fsolve( %=0, phi=340*Pi/180.);
  Min_max := evalf(subs(phi=erg,  M_inpar));
```

$$erg := 5.968816757$$

$$Min\_max := 1.852661506$$

End of Problem 1