**Gabriele Minotto**
**Exercise: OPTIMIZATION of double steering-driving wheel in chained form, using polinomial functions**

```
> restart: with(plots): with(LinearAlgebra): with(Threads): with
  (Optimization): with(VectorCalculus):
> # Data Inputs.
  data := [ T = 1, n_OP = 3, K_chosen = 3.4, epsilon = 0.5, n_P =
  100, L = 2]:
  Obs_Pos_Data := [ xp__1 = 2, yp__1 = 2, xp__2 = 11, yp__2 = 4 ,
  xp__3 = 6, yp__3 = 1, xp__4 = 7, yp__4 = 7 ]: <%>:
```

**Model definition: due to the chained form, attitude and steering angles must not be egual to Pi/2, otherwise there are singular configurations and the problem cannot be solved**

```
> eq_tr := []: <%>;
```

$$\left[\ \right] \tag{1}$$

```
> eq_v := [   ]: <%>;
```

$$\left[\ \right] \tag{2}$$

```
>
  chained_form := []: <%>:
  chain_sub := []:
  chained_form_sub := subs(chain_sub,[]):<%>: <chained_form>,
  <chained_form_sub>;
```

$$\left[\ \right], \left[\ \right] \tag{3}$$

**1) Initial transformed conditions (from world to vehicle RF)**

```
> Trasform_V_to_World := [x__w(t) =  x(t)*cos(delta(t)) - y(t)*sin
  (delta(t)), y__w(t) = x(t)*sin(delta(t)) + y(t)*cos(delta(t))]:
  Trasform_World_to_V := simplify(op(solve(Trasform_V_to_World, [x
  (t), y(t)]))): <%%>, <%>;

  in_W_cond := [x__w(t) = 0, y__w(t) = 0, delta(t) = Pi/4 ]:
  #curvatura iniziale parametrica
  fi_W_cond := [x__w(t) = 10, y__w(t) = 10, delta(t) = Pi/4]: <%%>,
  <%>:

  in_cond := [x(t) = subs(in_W_cond, rhs(Trasform_World_to_V[1])),
  y(t) = subs(in_W_cond, rhs(Trasform_World_to_V[2])), alpha__1(t)
  = 0, alpha__2(t) = 0]:
  fi_cond := [x(t) = subs(fi_W_cond, rhs(Trasform_World_to_V[1])),
  y(t) = subs(fi_W_cond, rhs(Trasform_World_to_V[2])), alpha__1(t)
  = 0, alpha__2(t) = 0]: <%%>, <%>;
```

$$\left[\begin{array}{l} x_w(t) = x(t)\cos(\delta(t)) - y(t)\sin(\delta(t)) \\ y_w(t) = x(t)\sin(\delta(t)) + y(t)\cos(\delta(t)) \end{array}\right], \left[\begin{array}{l} x(t) = x_w(t)\cos(\delta(t)) + y_w(t)\sin(\delta(t)) \\ y(t) = -\sin(\delta(t))\,x_w(t) + \cos(\delta(t))\,y_w(t) \end{array}\right] \tag{4}$$

$$\begin{bmatrix} x(t)=0 \\ y(t)=0 \\ \alpha_1(t)=0 \\ \alpha_2(t)=0 \end{bmatrix}, \begin{bmatrix} x(t)=10\sqrt{2} \\ y(t)=0 \\ \alpha_1(t)=0 \\ \alpha_2(t)=0 \end{bmatrix} \quad (4)$$

```
> in_tr_cond := (subs(in_W_cond,in_cond,data,eq_tr)):
  fi_tr_cond := (subs(fi_W_cond,fi_cond,data,eq_tr)): <%%>, <%>;
```

$$[\ ], [\ ] \quad (5)$$

2) Input u__1 e u__2 in symbolic form

```
> u_t := [ ]: <%>;
```

$$[\ ] \quad (6)$$

```
> : <%>:
```

```
> # Generalized velocities and positions.
  sys_vel := []: <%>:
  sys_pos := []: <%>:
```

3) Constraint initial and final conditions - linear system.

```
> Bounds_1 := subs(data,t=subs(data,T),[]): <%>:
```

4) Preparation of Optimization: Target is minimization of trajectory and then the path, bounds as final conditions and minimum distance from obstacles

```
> tr_to_gen := op(solve(eq_tr,[x(t),y(t),delta(t), alpha__1(t),
  alpha__2(t)])): <%>:
  tr_sub := subs(subs(data,sys_pos),tr_to_gen): <%>:
```

Definition of target to minimize

```
> Trajectory := ( );
  Trajectory_sub := subs(tr_sub,Trajectory):
  Target := sum(evalf(subs(t = subs(data,T)/10*k, Trajectory_sub)),
  k = 0..10):
```

$$Trajectory := \quad (7)$$

Definition of bounds made by obstacles

```
> OP[1] := [xp__1, yp__1]:
  OP[2] := [xp__2, yp__2]:
  OP[3] := [xp__3, yp__3]:
  OP[4] := [xp__4, yp__4]:
  OP[5] := [xp__5, yp__5]:
  OP[6] := [xp__6, yp__6]:

  OPV[1] := [subs(x__w(t)=xp__1, y__w(t) = yp__1, rhs
  (Trasform_World_to_V[1])) , subs(x__w(t)=xp__1, y__w(t) = yp__1,
  rhs(Trasform_World_to_V[2]))]:
  OPV[2] := [subs(x__w(t)=xp__2, y__w(t) = yp__2, rhs
  (Trasform_World_to_V[1])) , subs(x__w(t)=xp__2, y__w(t) = yp__2,
  rhs(Trasform_World_to_V[2]))]:
  OPV[3] := [subs(x__w(t)=xp__3, y__w(t) = yp__3, rhs
  (Trasform_World_to_V[1])) , subs(x__w(t)=xp__3, y__w(t) = yp__3,
  rhs(Trasform_World_to_V[2]))]:
```

```
OPV[4] := [subs(x__w(t)=xp__4, y__w(t) = yp__4, rhs
(Trasform_World_to_V[1])) , subs(x__w(t)=xp__4, y__w(t) = yp__4,
rhs(Trasform_World_to_V[2]))]:
OPV[5] := [subs(x__w(t)=xp__5, y__w(t) = yp__5, rhs
(Trasform_World_to_V[1])) , subs(x__w(t)=xp__5, y__w(t) = yp__5,
rhs(Trasform_World_to_V[2]))]:
OPV[6] := [subs(x__w(t)=xp__6, y__w(t) = yp__6, rhs
(Trasform_World_to_V[1])) , subs(x__w(t)=xp__6, y__w(t) = yp__6,
rhs(Trasform_World_to_V[2]))]:

Phi := []:
OPTOT := []:
for i from 1 by 1 to subs(data,n_OP) do
  OPTOT := [op(OPTOT), OPV[i]]:
   #Phi := [ op(Phi), norm(  <(x(t) - OPTOT[i][1]) , (y(t) -
OPTOT[i][2])> ) >= subs(data,epsilon^2)  ]:
   Phi := [ op(Phi),  (  (x(t) - OPTOT[i][1])^2 + (y(t) - OPTOT
[i][2])^2 ) - subs(data,epsilon^2)  >=  0 ];
end do:

Phi;
Phi_sub := subs(tr_sub,Obs_Pos_Data,K = K_chosen,Phi):
Bounds_Phi := []:
for i from 0 by 1 to subs(data,n_P) do
  for j from 1 by 1 to subs(data,n_OP) do
    Bounds_Phi := [ op(Bounds_Phi), subs(t = subs(data,T/n_P*i),
Phi_sub[j])]:
  end do:
end do:
Bounds_Phi_1 := [ sum(Bounds_Phi[k], k = 1..subs(data,n_P)) ]:
op(Bounds_Phi):
```

$$\left[0 \le \left(x(t) - xp_1 \cos(\delta(t)) - yp_1 \sin(\delta(t))\right)^2 + \left(y(t) + \sin(\delta(t))\, xp_1 - \cos(\delta(t))\, yp_1\right)^2 \right. \tag{8}$$
$$- 0.25, 0 \le \left(x(t) - xp_2 \cos(\delta(t)) - yp_2 \sin(\delta(t))\right)^2 + \left(y(t) + \sin(\delta(t))\, xp_2\right.$$
$$\left. - \cos(\delta(t))\, yp_2\right)^2 - 0.25, 0 \le \left(x(t) - xp_3 \cos(\delta(t)) - yp_3 \sin(\delta(t))\right)^2 + \left(y(t)\right.$$
$$\left. + \sin(\delta(t))\, xp_3 - \cos(\delta(t))\, yp_3\right)^2 - 0.25\Big]$$

**Optimization**

```
> Optim := :
  Optim_Free :=  #Optimization WITHOUT obstacles
  Optim_set := Optim[2];
  Optim_set_Free := Optim_Free[2];
Error, `:` unexpected
```

**5) Obtain x__i(t) from optimization**

```
> x__i_final := simplify(evalf(subs(Optim_set,sys_pos))): <%>;
  x__i_final_Free := simplify(evalf(subs(Optim_set_Free,sys_pos))):
  <%>:
Error, invalid input: subs received Optim_set, which is not
valid for its 1st argument
```

$$\begin{bmatrix} 303 \times 1 \text{ Matrix} \\ \text{Data Type: anything} \\ \text{Storage: rectangular} \\ \text{Order: Fortran\_order} \end{bmatrix}$$

Error, invalid input: subs received Optim_set_Free, which is not valid for its 1st argument

**6) Obtain generalized coordinate x, y, delta**

```
> gen_coord := evalf(subs(Optim_set,data,tr_sub)): <%>;

  gen_coord_diff := diff(gen_coord,t): <%>:
  gen_coord_Free := evalf(subs(Optim_set_Free,data,tr_sub)): <%>:
  gen_coord_Free_diff := diff(gen_coord_Free,t): <%>:
```

Error, invalid input: subs received Optim_set, which is not valid for its 1st argument

$$\begin{bmatrix} 303 \times 1 \text{ Matrix} \\ \text{Data Type: anything} \\ \text{Storage: rectangular} \\ \text{Order: Fortran\_order} \end{bmatrix}$$

Error, invalid input: subs received Optim_set_Free, which is not valid for its 1st argument

**7) Find the two inputs**

```
> the_controls := simplify(subs(data,subs(Optim_set,u_t),gen_coord,
  x__i_final,eq_v),size): <%>;
  #subs(data,t=0.6,the_controls[1]);
  the_controls_Free := simplify(subs(data,subs(Optim_set_Free,u_t),
  gen_coord_Free,x__i_final_Free,eq_v),size): <%>:
```

Error, invalid input: subs received Optim_set, which is not valid for its 1st argument

$$\begin{bmatrix} 0 \end{bmatrix}$$

Error, invalid input: subs received Optim_set_Free, which is not valid for its 1st argument

**8) Plot trajectories & controls**

```
> pos_world := subs(subs(data,gen_coord),Trasform_V_to_World): <%>;
  subs(t=subs(data,T),%): <%>;
  pos_world_Free := subs(subs(data,gen_coord_Free),
  Trasform_V_to_World): <%>:
  subs(t = subs(data,T),%):
```

Error, invalid input: subs received gen_coord, which is not valid for its 1st argument

$$\begin{bmatrix} 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 \end{bmatrix}$$

Error, invalid input: subs received gen_coord_Free, which is not valid for its 1st argument

```
> PV := plot([subs(gen_coord_Free,x(t)), subs(gen_coord_Free,y(t)),
  t = 0 .. subs(data,T)], color="DarkOrange",gridlines=true,labels=
```

```
  ["x(t) [m]","y(t) [m]"],title="Trajectory in time, in Car RF",
  axes=boxed):
  C1 := plot([subs(the_controls_Free,v__1(t))], t = 0 .. subs(data,
  T),color=["DarkOrange"],gridlines=true,labels=["t [s]","v(t)
  [m/s]"],title="Traction Velocity in time",axes=boxed,legend=
  ["v__1(t)"]):
  C2 := plot([subs(the_controls_Free,alpha__Dot__1(t)), subs
  (the_controls_Free,alpha__Dot__2(t))], t = 0 .. subs(data,T),
  color=["Red","Green"],gridlines=true,labels=["t [s]", "D(alpha)
  (t) [rad/s]"],title="Controls in time",axes=boxed,legend=["D
  (alpha)__1(t)","D(alpha)__2(t)"]):
  PW := plot([subs(pos_world_Free[1],x__w(t)), subs(pos_world_Free
  [2],y__w(t)), t = 0 .. subs(data,T)], color="DarkOrange",
  gridlines=true,labels=["x(t) [m]","y(t) [m]"],title="Trajectory
  in time, in World RF",axes=boxed):
  AT := plot([subs(subs(data,gen_coord_Free),delta(t))], t = 0 ..
  subs(data,T), color="DarkOrange",gridlines=true,labels=["t [s]",
  "delta(t) [rad]"],title="Attitude in time",axes=boxed):
  display(Array(1..5,[PW,PV,C1,C2,AT]));
Error, invalid input: subs received gen_coord_Free, which is not
valid for its 1st argument
Error, invalid input: subs received the_controls_Free, which is
not valid for its 1st argument
Error, invalid input: subs received the_controls_Free, which is
not valid for its 1st argument
Error, invalid input: subs received pos_world_Free[1], which is
not valid for its 1st argument
Error, invalid input: subs received gen_coord_Free, which is not
valid for its 1st argument
Error, (in plots:-display) element 1 of the rtable is not a
valid plot structure
> PVO := plot([subs(gen_coord,x(t)), subs(gen_coord,y(t)), t = 0 ..
  subs(data,T)], color="DarkOrange",gridlines=true,labels=["x(t)
  [m]","y(t) [m]"],title="Trajectory in time, in Car RF",axes=
  boxed):
  C10 := plot([subs(the_controls,v__1(t))], t = 0 .. subs(data,T),
  color=["DarkOrange"],gridlines=true,labels=["t [s]","v(t) [m/s]
  "],title="Traction Velocity in time",axes=boxed,legend=["v__1(t)
  "]):
  C20 := plot([subs(the_controls,alpha__Dot__1(t)), subs
  (the_controls,alpha__Dot__2(t))], t = 0 .. subs(data,T),color=
  ["Red","Green"],gridlines=true,labels=["t [s]", "D(alpha)(t)
  [rad/s]"],title="Controls in time",axes=boxed,legend=["D(alpha)
  __1(t)","D(alpha)__2(t)"]):
  PWO := plot([subs(pos_world[1],x__w(t)), subs(pos_world[2],y__w
  (t)), t = 0 .. subs(data,T)], color="DarkOrange",gridlines=true,
  labels=["x(t) [m]","y(t) [m]"],title="Trajectory in time, in
  World RF",axes=boxed):
  ATO := plot([subs(subs(data,gen_coord),delta(t))], t = 0 .. subs
  (data,T), color="DarkOrange",gridlines=true,labels=["t [s]",
  "delta(t) [rad]"],title="Attitude in time",axes=boxed):
  POBS := pointplot({seq([op(subs(Obs_Pos_Data,OP[i]))], i = 1 ..
  subs(data,n_OP))}, color = blue, symbolsize = 20):
  POW := display([PWO,POBS]):
  display(Array(1..5,[POW,PVO,C10,C20,ATO]));
  #display(Array(1..2,[PFree,P13]));
```

```
  #display(Array(1..3,[POW, C10,C20]));
Error, invalid input: subs received gen_coord, which is not
valid for its 1st argument
Error, invalid input: subs received the_controls, which is not
valid for its 1st argument
Error, invalid input: subs received the_controls, which is not
valid for its 1st argument
Error, invalid input: subs received pos_world[1], which is not
valid for its 1st argument
Error, invalid input: subs received gen_coord, which is not
valid for its 1st argument
Error, (in plots:-display) expecting plot structures but
received: [PWO]
Error, (in plots:-display) element 1 of the rtable is not a
valid plot structure
> ########################
```