Definitions

- What is FPP?
- What is PTC?

Cornell-KEK support: Profs. Dave Rubin and Makoto Tobiyama

History

- Start FPP around 1998/Start PTC around 2000
- Put into MAD-X with help of Frank Schmidt and Eric Macintosh completed by 2002
- Spin added around 2007 in FPP and PTC (Barber pressure)
- Magnet modulation added in FPP and PTC (Schmidt pressure)
- Continuous collaboration with David Sagan since the beginning: FPP/PTC fully added in the early 2010 in BMAD
- PTC painfully added in ORBIT under the guidance Alexander Molodozhentsev (It is unfortunately a “hack” but it worked)
- Made SAD compatible under the guidance of David Sagan
- Analysis tools fully “complexified”, old Fortran 77 routines are obsolescent (Deniau pressure)

FPP : taylor series + analysis

- FPP overloads the “DA” or “TPSA” package of Berz
- It also overloads my Fortran 77 analysis tools (Normal Form etc...)
- It has a polymorphic type (REAL_8): real and Taylor
- Since 2011, it has a new analysis package based on a complex TPSA
- Normal form with magnet modulation, radiative maps, coasting beam, spin with SO(3) and quaternions
- Maps of all sorts and kinds: Lie transforms, generating functions, partially inverted maps

PTC

- It is (generally symplectic) integrator: a tracking code
- Novel structures: physical beam lines are not identified to tracking structures: figure 8, colliders, recirculators
- Has the concept of a fibre to deal with the novel structures
- It uses a polymorphic type so anything can become a Taylor series: phase space, spin, a_n, b_n, etc..
- Radiation, beam envelopes, magnet modulation, spin, etc...
- It is inside MAD-X (Schmidt/Macintosh)
- It is inside BMAD (Sagan)
- Hacked job in ORBIT (Modolzhentsev)
- Pretty much supports SAD magnets (as does BMAD, Sagan pressure)

FPP and PTC: New?

- FPP is new technology but nothing scientifically new

Explained in a Fermilab Lecture in the late 1980s

1) Berz Taylor TPSA overloaded
2) Polymorph type created: can be real or Taylor at execution time (Bengtsson)
3) A complex TPSA with normal form for transverse, spin (quaternion), magnet modulation.

- PTC has a different beam structure: New idea

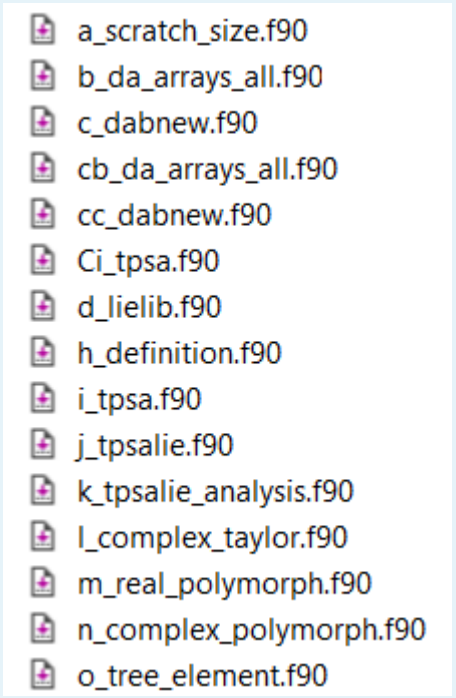
Realization of a fibre bundle structure (idea of Hirata and Forest 1992)

1) Trackable beam line is succession of containers called “fibres”
2) The fibre is discrete equivalent of the “s” variable
3) Each fibre points to a physical magnet
4) Magnets are not cloned as in the real world

Production of Taylor Maps (Sagan)

- It is possible to replace a magnet by a Taylor map from PTC or elsewhere
- It is possible to produce Taylor maps for tracking and incorporate it elsewhere
- Symplectic/Unitary algorithm are possible on the Taylor maps (Orbital-Spin)

FPP alone



Mini tracking code → y_multipole.f90

Main program → Z_FPP_normal_form.f90

One-turn Taylor map computation

```
program FPP
 use my_multipole
implicit none
integer order,nd
real(dp) closed_orbit(6)
! polymorph
type(real_8) x(6)
longprint = .false.
call set_multipole
call find_closed_orbit(closed_orbit)
write(6,*)  " closed orbit ",   closed_orbit

order=8

nd=1
call c_init_all(order,nd,0,0)
call alloc(x)

x=closed_orbit

call print(x)

call multipole(x)

call print(x)

x=closed_orbit


x(1)=closed_orbit(1)+dz_8(1)
x(2)=closed_orbit(2)+dz_8(2)
call print(x)
call multipole(x)


call print(x)

end program FPP
```

```
program FPP_id
 use my_multipole
implicit none
integer order,nd
real(dp) closed_orbit(6)
! polymorph
type(real_8) x(6)
! TPSA Maps
type(c_damap) id,one_turn
longprint = .false.
call set_multipole
call find_closed_orbit(closed_orbit)
write(6,*)  " closed orbit ",   closed_orbit

order=8

nd=1
call c_init_all(order,nd,0,0)
call alloc(x);call alloc(id,one_turn)

x=closed_orbit

call print(x)

call multipole(x)

call print(x)

id=1
x= closed_orbit + id

call print(x)

call multipole(x)

one_turn=x
call print(one_turn)

end program FPP_id
```

```
subroutine multipolep(x)
implicit none
type(real_8) x(2)
 x(1)=x(1)+Lpol*x(2)
 x(2)=x(2)-bnpol(1)-bnpol(2)*x(1)   &
   -bnpol(3)*x(1)**2-bnpol(4)*x(1)**3
end subroutine multipolep
```

program FPP

```
 closed orbit  -1.127016653792583E-002
-7.531047606624180E-018
           0
 -1.127016653792583E-002
 -7.531047606624180E-018
 -1.127016653792583E-002
 -7.535205098774256E-018

 Properties, NO =    8, NV =    2, INA =   26
 ********************************************

   0 -0.1127016653792583E-01   0  0
   1   1.000000000000000       1  0


 Properties, NO =    8, NV =    2, INA =   27
 ********************************************

   0 -0.7531047606624180E-17   0  0
   1   1.000000000000000       0  1


 Properties, NO =    8, NV =    2, INA =   26
 ********************************************

   0 -0.1127016653792583E-01   0  0
   1   1.000000000000000       1  0
   1  0.1000000000000000       0  1


 Properties, NO =    8, NV =    2, INA =   27
 ********************************************

   0 -0.7535205098774256E-17   0  0
   1 -0.7745966692414834E-01   1  0
   1  0.9922540333075851       0  1
   2  -1.000000000000000       2  0
   2 -0.2000000000000000       1  1
   2 -0.1000000000000000E-01   0  2
```

program FPP_id

```
closed orbit  -1.127016653792583E-002 -7.531047606624180E-018
           0
-1.127016653792583E-002
-7.531047606624180E-018
-1.127016653792583E-002
-7.535205098774256E-018

Properties, NO =    8, NV =    2, INA =   25
********************************************

  0 -0.1127016653792583E-01   0  0
  1   1.000000000000000       1  0


Properties, NO =    8, NV =    2, INA =   31
********************************************

  0 -0.7531047606624180E-17   0  0
  1   1.000000000000000       0  1


          2  Dimensional DA map (around chosen orbit in map%x0)

Properties, NO =    8, NV =    2, INA =  161
********************************************

  0 -0.1127016653792583E-01   0.000000000000000       0  0
  1   1.000000000000000       0.000000000000000       1  0
  1  0.1000000000000000       0.000000000000000       0  1


Properties, NO =    8, NV =    2, INA =  160
********************************************

  0 -0.7535205098774256E-17   0.000000000000000       0  0
  1 -0.7745966692414834E-01   0.000000000000000       1  0
  1  0.9922540333075851       0.000000000000000       0  1
  2  -1.000000000000000       0.000000000000000       2  0
  2 -0.2000000000000000       0.000000000000000       1  1
  2 -0.1000000000000000E-01   0.000000000000000       0  2

 No Spin Matrix
 c_quaternion is identity
No Stochastic Radiation
```

Normal form

```
program Z_FPP_normal_form

    use my_multipole
    implicit none
    integer order,nd
    real(dp) closed_orbit(6)
    ! polymorph
    type(real_8) x(6)
    ! TPSA Maps
    type(c_damap) id,one_turn,diagonal,rotation
    type(c_normal_form) normal_form
    longprint = .false.
    call set_multipole
    call find_closed_orbit(closed_orbit)
    write(6,*)  " closed orbit ",   closed_orbit

    order=3

    nd=1
    call c_init_all(order,nd,0,0)
    call alloc(x);call alloc(id,one_turn);
    call alloc(normal_form);
    call alloc(diagonal,rotation)

    id=1
    x= closed_orbit + id

    call multipole(x)

    one_turn=x

    call c_normal(one_turn,normal_form)

    rotation=normal_form%atot**(-1)*one_turn*normal_form%atot
    call clean(rotation,rotation,1.d-6)
    call print(rotation)
    diagonal=c_phasor()**(-1)*rotation*c_phasor()
    call clean(diagonal,diagonal,1.d-6)
    call print(diagonal)

end program Z_FPP_normal_form
```

```
closed orbit  -1.127016653792583E-002 -7.531047606624180E-018
          0
1  0.99612701665379      0.87925915931623E-01
2  0.99612701665379     -0.87925915931623E-01
 The order of the planes has been guessed using the algorithm in c_locate_planes
 Hopefully it is correct! Please check!
Order guessed ->  1

          2  Dimensional DA map (around chosen orbit in map%x0)

Properties, NO =    3, NV =    2, INA =  258
*********************************************

  1  0.9961270166537924       0.000000000000000       1  0
  1  0.8792591593162297E-01   0.000000000000000       0  1
  3  0.6114646075503689       0.000000000000000       3  0
  3  -6.927382090420237       0.000000000000000       2  1
  3  0.6114646075504240       0.000000000000000       1  2
  3  -6.927382090420256       0.000000000000000       0  3


Properties, NO =    3, NV =    2, INA =  259
*********************************************

  1 -0.8792591593162302E-01   0.000000000000000       1  0
  1  0.9961270166537927       0.000000000000000       0  1
  3   6.927382090420240       0.000000000000000       3  0
  3  0.611464607zz5503596       0.000000000000000       2  1
  3   6.927382090420270       0.000000000000000       1  2
  3  0.6114646075503209       0.000000000000000       0  3

 No Spin Matrix
 c_quaternion is identity
No Stochastic Radiation

          2  Dimensional DA map (around chosen orbit in map%x0)

Properties, NO =    3, NV =    2, INA =  243
*********************************************

  1  0.9961270166537926     -0.8792591593162299E-01   1  0
  3  0.6114646075503567       6.927382090420251       2  1


Properties, NO =    3, NV =    2, INA =  244
*********************************************

  1  0.9961270166537926      0.8792591593162299E-01   0  1
  3  0.6114646075503567      -6.927382090420251       1  2

 No Spin Matrix
 c_quaternion is identity
No Stochastic Radiation
```

FPP + PTC

- a_scratch_size.f90
- b_da_arrays_all.f90
- c_dabnew.f90
- cb_da_arrays_all.f90
- cc_dabnew.f90
- Ci_tpsa.f90
- d_lielib.f90
- h_definition.f90
- i_tpsa.f90
- j_tpsalie.f90
- k_tpsalie_analysis.f90
- l_complex_taylor.f90
- m_real_polymorph.f90
- n_complex_polymorph.f90
- o_tree_element.f90

FPP ←

PTC →

- Sa_extend_poly.f90
- Sb_sagan_pol_arbitrary.f90
- Sc_euclidean.f90
- Sd_frame.f90
- Se_status.f90
- Sf_def_all_kinds.f90
- Sg_sagan_wiggler.f90
- Sh_def_kind.f90
- Si_def_element.f90
- Sk_link_list.f90
- Sl_family.f90
- Sm_tracking.f90
- Sma0_beam_beam_ptc.f90
- Sma_multiparticle.f90
- Sn_mad_like.f90
- So_fitting.f90
- Sp_keywords.f90
- Spb_fake_gino_sub.f90
- Sq_orbit_ptc.f90
- Sr_spin.f90
- Sra_fitting.f90
- Ss_fake_mad.f90
- St_pointers.f90

User defined module →

Main program →

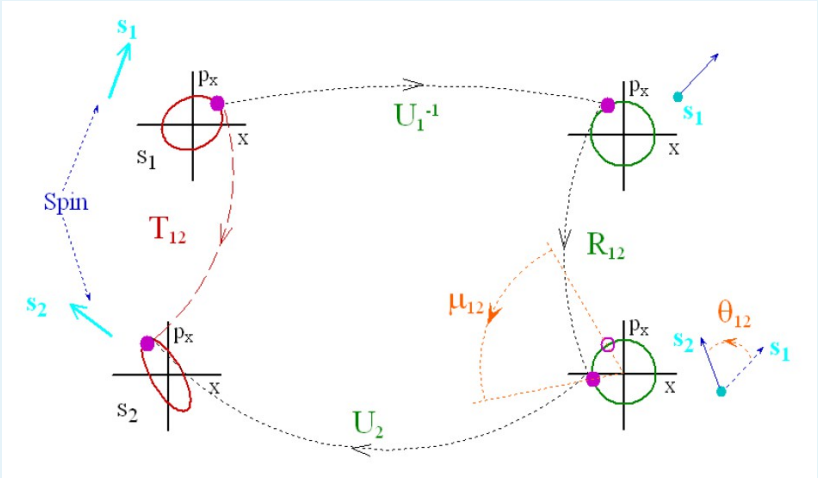
- y_als_lattice.f90
- z_spin_phase_advance_isf_for_sad.f90

Courant-Snyder Loop

```
state=nocavity0+SPIN0
CALL FIND_ORBIT(ALS,CLOSED,1,STATE,c_1d_5)
state=nocavity0
no=2
call init_all(STATE,no,1)


!!!! Polymorphic probe is created in the usual manner
   XS0=CLOSED
   ID_S=1
   XS=XS0+ID_S

!!!! get spin polymorphic probe after one turn
CALL propagate(ALS,XS,+STATE,FIBRE1=1)


! Copy probe_8 into a complex damap
c_map=XS

! Normalize it c_map = C_n%atot^(-1) o R o C_n%atot
call c_normal(c_map,c_n,dospin=my_true,phase=phase,nu_spin=nu_spin)

U=c_n%Atot  ! Non-descript U exiting normal form
call c_full_canonise(U,U_c,D,F,A,b,R,phase,nu_spin)

 p => als%start

do i=1,als%n

 CALL propagate(ALS,XS,+STATE,FIBRE1=i,fibre2=i+1)

  xs0=xs ! Saving orbit  ! (9b)
  U=XS ! copying in map  ! (9c)
  ! U = U_c o  R = D o f o A o b o R
  call c_full_canonise(U,U_c,D,F,A,b,R,phase,nu_spin)
```

COMPUTE SOMETHING HERE

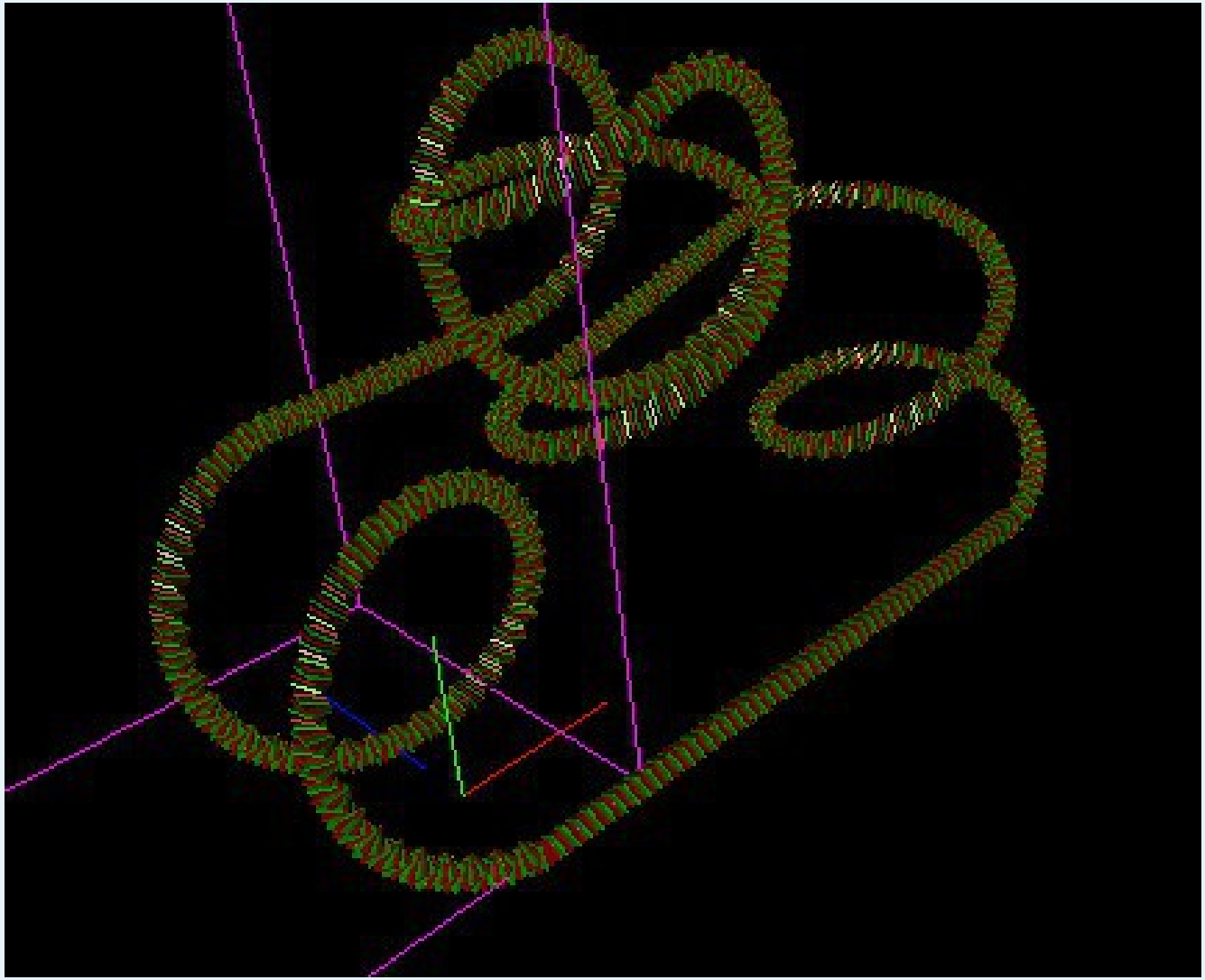
```
 XS=XS0+U_c
 p=>p%next
enddo
```



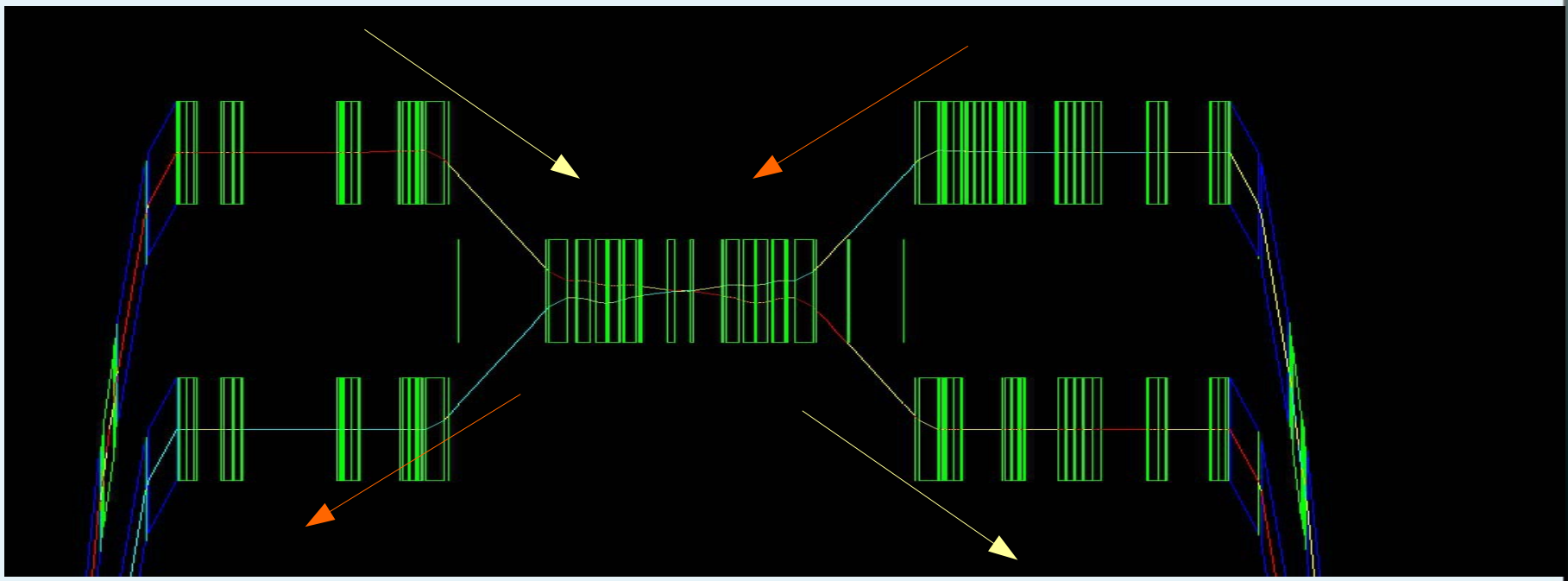
Compute something?

```
 write(mft,*) "position, Element ", i, p%mag%name

  betax_1=(U_c%v(1).sub.'1000')**2+(U_c%v(1).sub.'0100')**2
  betax_2=(U_c%v(1).sub.'0010')**2+(U_c%v(1).sub.'0001')**2

if(use_quaternion) call makeso3(D)
 ISF=2
 ISF=D%s*ISF

 write(mft,*) " 2< x^2 > "
   call print(fonction_FLOQUET,mft)

 write(mft,*) " Ripken Beta_x_1 Beta_x_2 ",betax_1,betax_2

Write(mft,*) " ISF vector n "
call print(ISF,mft,prec)

write(mft,*) " phase x"
call print(phase(1),mft,prec)

write(mft,*) " phase y"
call print(phase(2),mft,prec)

if(state%nocavity) then
 write(mft,*) "  Time "
 call print(phase(3),mft,prec)
endif

write(mft,*) " phase spin"
call print(nu_spin,mft,prec)
```

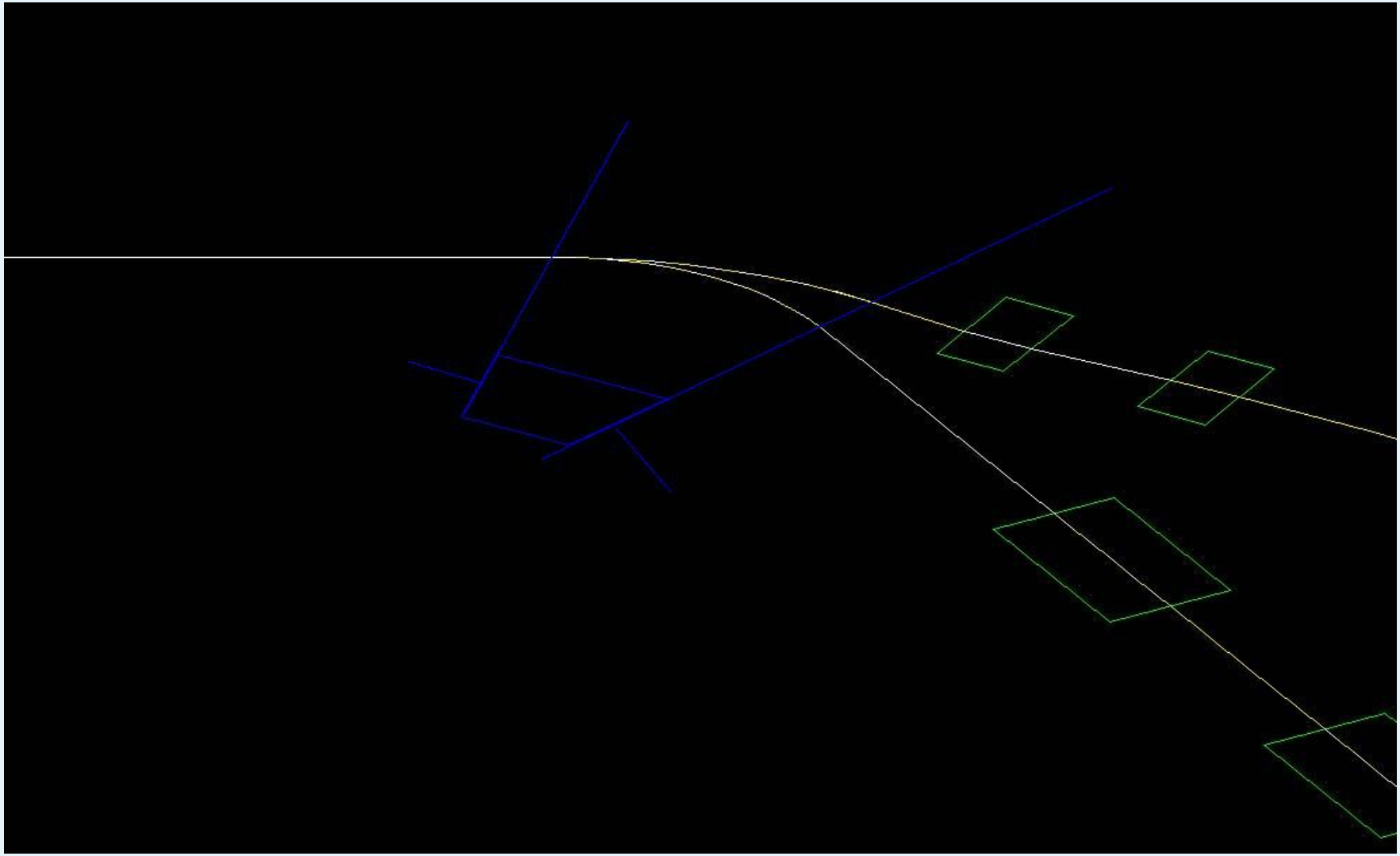
Some Strange Machida/Mohri Object



LHC



Some Recirculor



Future

1) Putting Laurent Deniau new TPSA in FPP
2) Of course anyone can incorporate FPP or PTC in their code
3) Perhaps one day, the full geometrical part of PTC will be integrated in a CAD environment or in a full code (BMAD is a prototype)
4) Always opened to perturbation theory applications: Qiang (LBL), Zhou (KEK), etc...