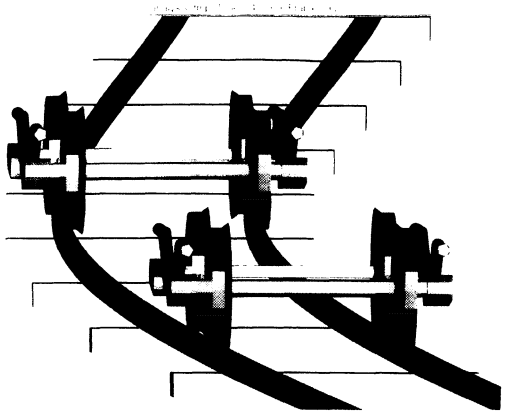
Customizing ADAMS for Rail Applications



A Joint Development of
Dutch Railways
and
TEDAS Mechanical Systems Software GmbH

Customizing ADAMS for Rail Applications

- Background
- ADAMS/Rail 1.0
  - Basic concept
  - Vehicle models
  - Track definition
  - Wheelset models
  - Applications
- Outlook

MSS Needs at Dutch Railway

- Approval of newly designed rail vehicles
- Modifications of existing designs
- Maintenance issues
- Accident reconstruction

Evaluation Process

- Specialized code

- General purpose code

Specialized Programs

- Pros:
  - Taylored to rail applications
  - Analysis based on simplifications
- Cons:
  - Analysis based on simplifications
  - Interfacing to other CAE-tools difficult
  - Adding new vehicle components difficult
  - Few number of installations (<20-30)

General Purpose Code ADAMS

- Pros:
  - General multi-body code, fully nonlinear
  - Industry proven, large number of installations
  - Up-to-date graphical user interface
  - Interfaces to CAD, FEA and controls
- Cons:
  - Not taylored to rail applications

Development Guidelines

- Specification made by railway engineers
- Requirements driven by applications
- Programming done by ADAMS experts

Applications

- Stiffness tests
- Eigenfrequencies
- Derailment safety
- Ride analysis
- Track loads

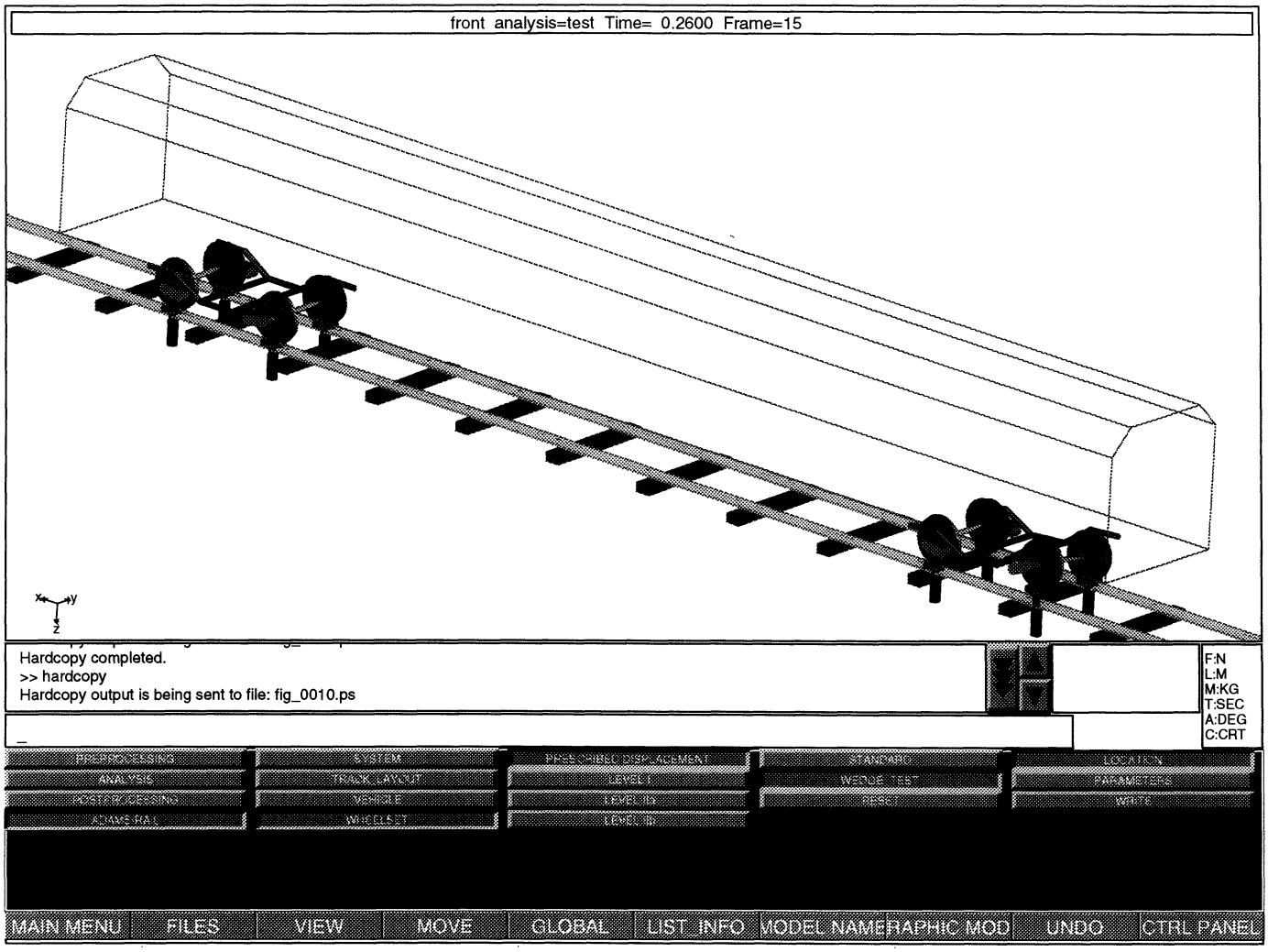
- Different applications require different level of sophistication

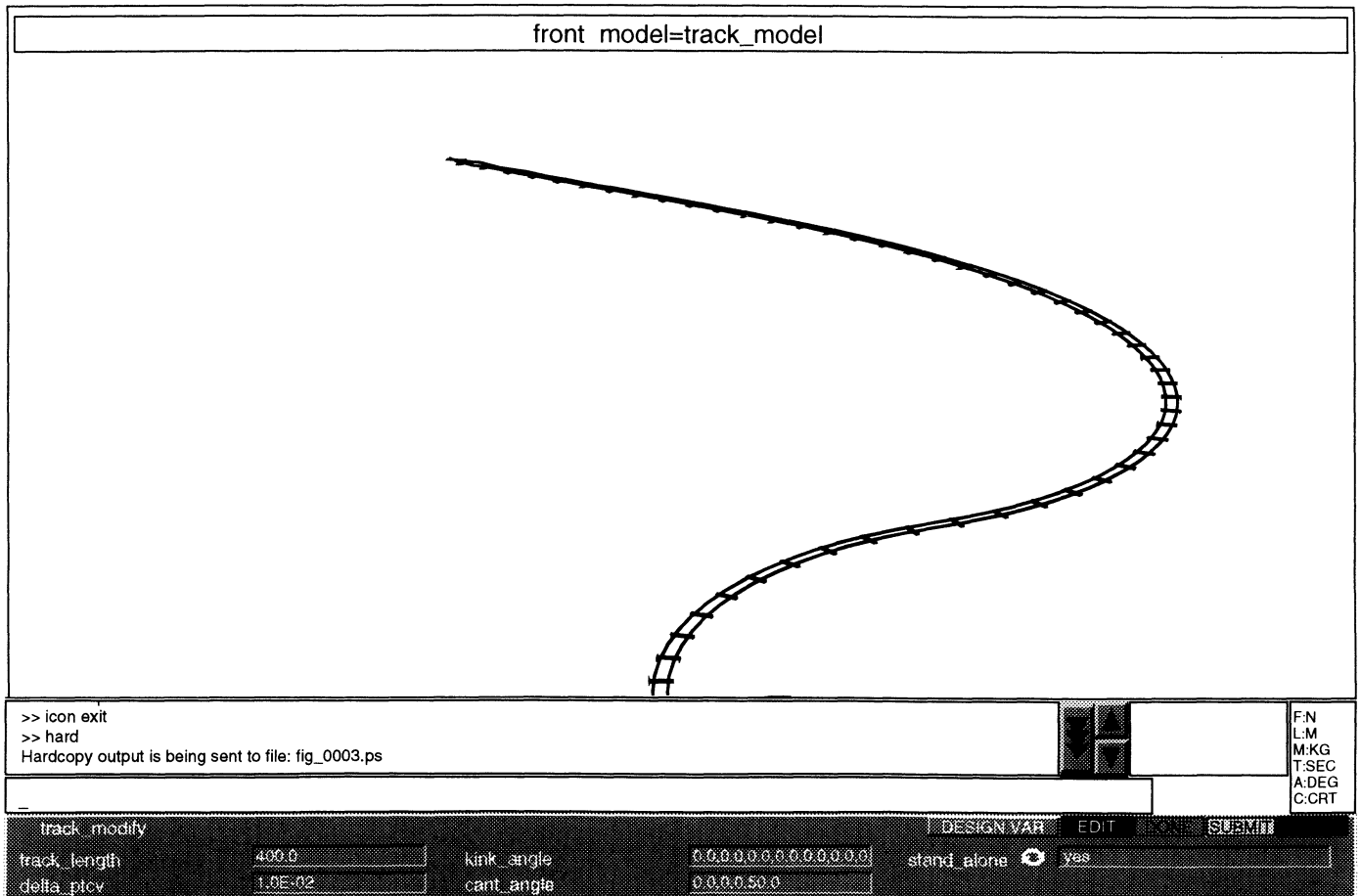
Structure of ADAMS/Rail

- Components
  - Vehicle
  - Track
  - Wheelset, i.e. rail-wheel contact
- Components are treated separately
  - create, modify, read, write
- Automatic system assembly

Track Definition

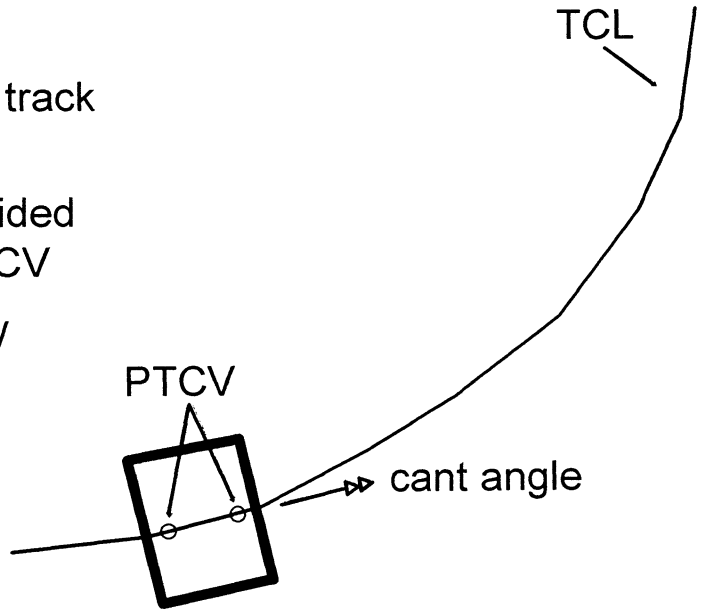
- Analytic description
  - horizontal, vertical curvature
  - cant angle
  - kink angle, inital grade
- Measured data
- Analytic description superimposed with measured data


front analysis=test Time= 0.2600 Frame=15
Hardcopy completed.
>> hardcopy
Hardcopy output is being sent to file: fig_0010.ps
F:N
L:M
M:KG
T:SEC
A:DEG
C:CRT
PREPROCESSING
SYSTEM
PRESCRIBED DISPLACEMENT
STANDARD
LOCATION
ANALYSIS
TRACK_LAYOUT
LEVEL1
WEDGE_TEST
PARAMETERS
POSTPROCESSING
VEHICLE
LEVEL2
PRES
WRITE
RAIL
WHEELSET
LEVEL3
MAIN MENU
FILES
VIEW
MOVE
GLOBAL
LIST INFO
MODEL NAME
RAPHIC MOD
UNDO
CTRL PANEL


front model=track_model
>> icon exit
>> hard
Hardcopy output is being sent to file: fig_0003.ps
F:N
L:M
M:KG
T:SEC
A:DEG
C:CRT
track_modify
DESIGN VAR
EDIT
SUBMIT
track_length
400.0
kink_angle
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
stand_alone
yes
delta_ptcv
1.0E-02
cant_angle
0.0,0.0,0.50.0

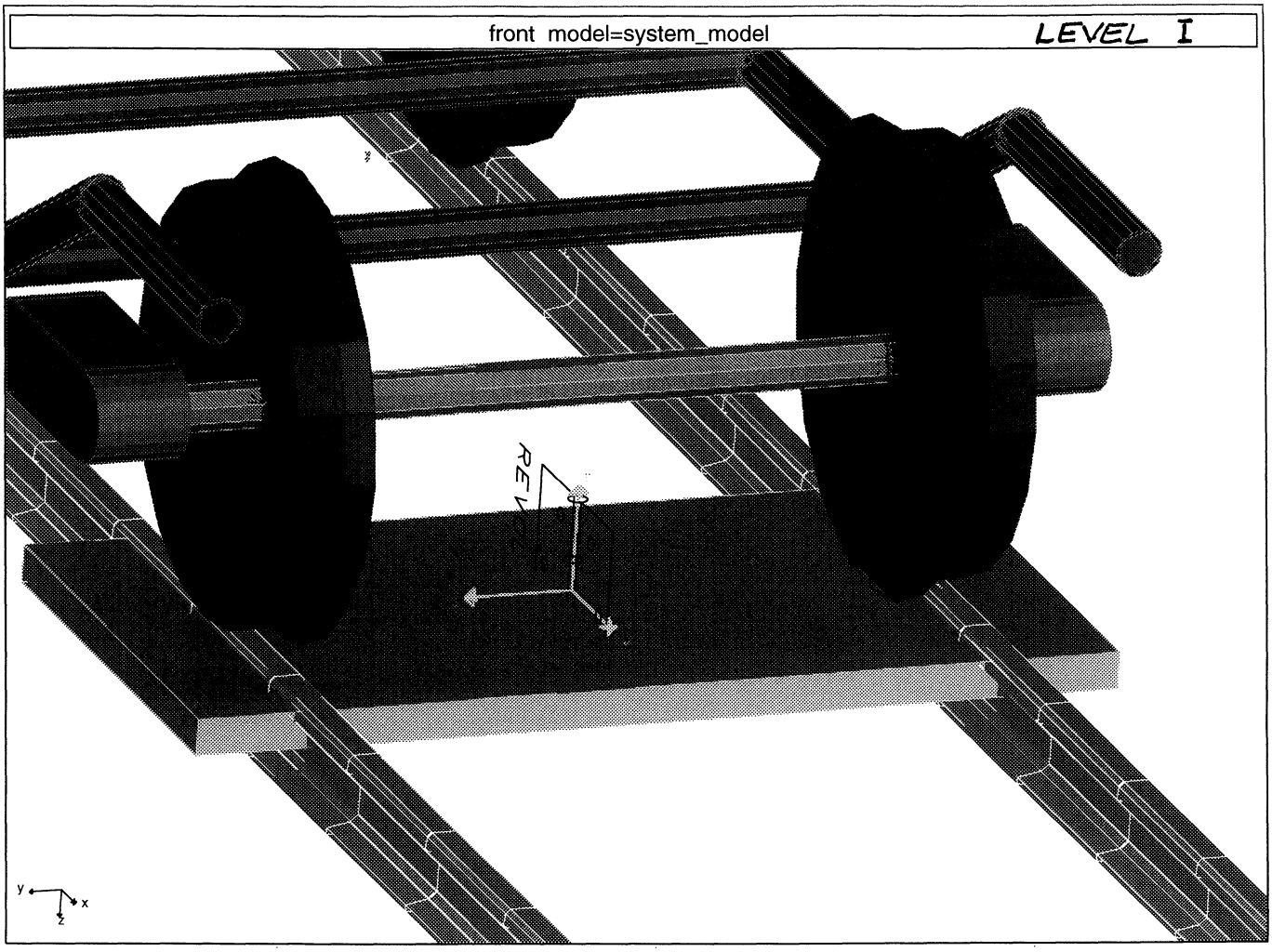
Constraint Realization

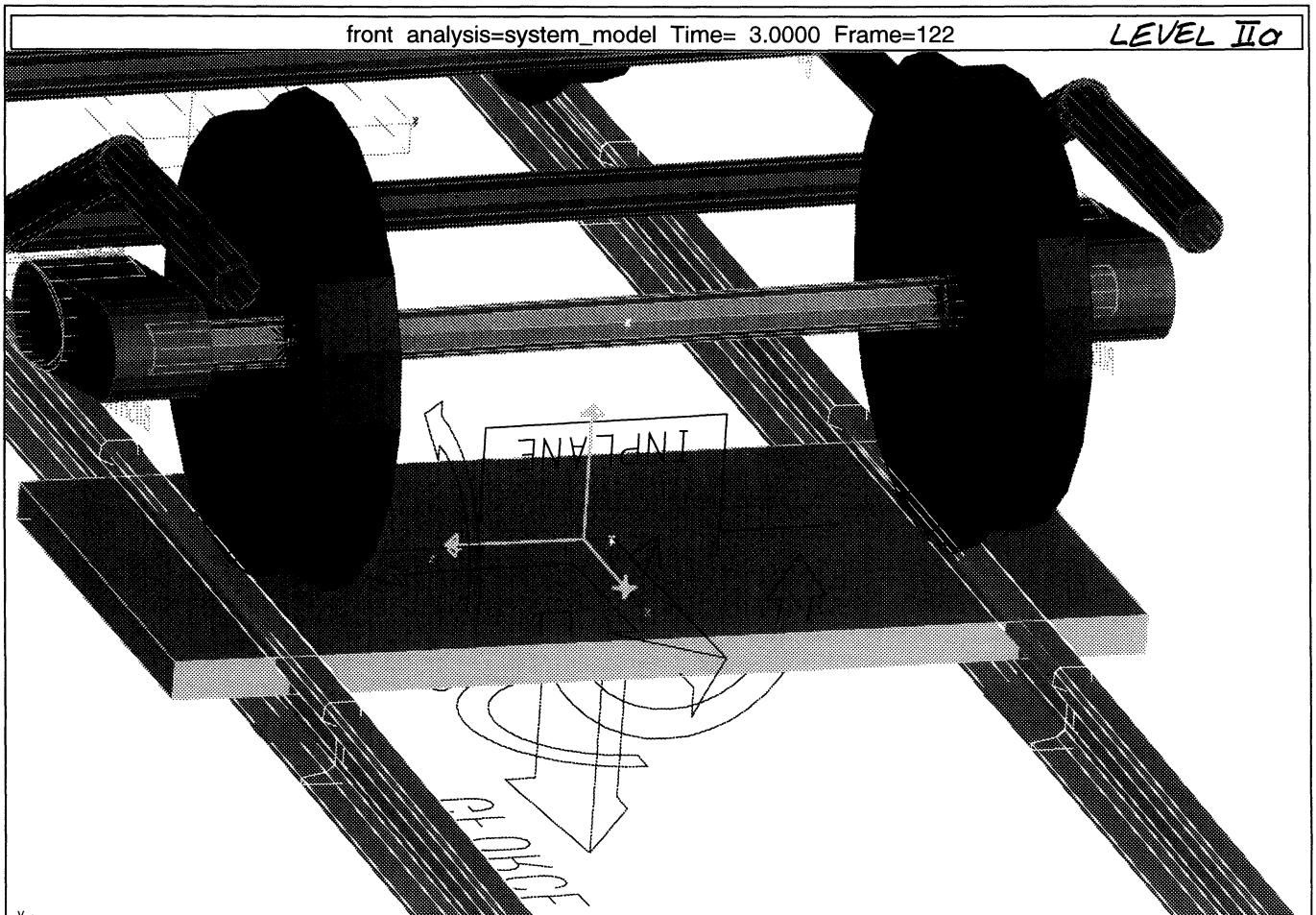
- CURSUB to describe track center line (TCL)
- Track dummy part guided along TCL by two PTCV
- Track cant realized by UCOSUB



Wheelset Levels

- Prescribed displacements
  - Vehicle on hydraulic cylinders
- Level I
  - Longitudinal and yaw degree of freedom
- Level IIa
  - Additional lateral DOF where creep forces are based on a linear model
- Level IIb
  - Additional to level I, axle spin rotation and corresponding non-linear friction forces are included
- Level III: Under development


front model=system_model
LEVEL I


front analysis=system_model Time= 3.0000 Frame=122
LEVEL IIa

Working with ADAMS/Rail

- Create/modify a vehicle, store it on disk
- Create/modify track description, store it on disk
- Create/modify wheelset data, store it on disk

- Assemble the system, specify which vehicle, track and wheelset level
- Perform the analysis wanted

Programming with ADAMS/View

- Use file text write to store information on disk

- How to set initial velocity for all parts

- How to use vectors, indexing

```
!USER_ENTERED_COMMAND ws2a_parameters
!-------------------------------------------------------------------------!
! project ADAMS/RAIL                              tedas/mh 08. feb.94
!
! Purpose: Creating array for gforce
!          Wheelset level IIa
!
!
! Input:
!
! $kalker_f11:t=real:u=1.5
! $kalker_f22:t=real:u=1.5
! $kalker_f23:t=real:u=0.8
! $axle_load:t=real:u=1e3
! $contact_angle:t=real:u=0
! $conicity:t=real:u=0
! $contact_angle_par:t=real:u=0
! $parameter_set_name:t=string:u=ws02a_array
!
! Output:                       Name
!
! gforce_array                  gfo_arr_'parameter_array'
!
! Requirements:
!
! nothing known
!
! Calls:
!
!
!
!-------------------------------------------------------------------------!
!
if condition=(!db_exists("$parameter_set_name"))
  data_element create array general array=$parameter_set_name &
   numbers= 0.0, &
            (half_rolling_line_distance), &
            $kalker_f11, &
            $kalker_f22, &
            $kalker_f23, &
            $axle_load, &
            $contact_angle, &
            $conicity, &
            $contact_angle_par
!
else
  data_element modify array general array=$parameter_set_name &
   numbers= 0.0, &
            (half_rolling_line_distance), &
            $kalker_f11, &
            $kalker_f22, &
            $kalker_f23, &
            $axle_load, &
            $contact_angle, &
            $conicity, &
            $contact_angle_par
end
!
fi ada wr fi=scratch
```

```
!USER_ENTERED_COMMAND wheelset_write
!-------------------------------------------------------------------------!
! project ADAMS/RAIL
!
! Purpose: write out wheelset data file
!
!
! Input:
!
! $ wheelset_file_name:t=string
!
! Output:                    Name
!
! Requirements:
!
! nothing known
!
! Calls:
!
!
!
!-------------------------------------------------------------------------!
file text open file="$wheelset_file_name.wsd" open_mode="overwrite"

file text write form=" "
file text write form="ws2a_parameters parameter_set_name=ws02a_array &"
file text write form="half_rolling_line_distance=%s &" &
                     val=(ws02a_array.numbers[2])
file text write form="kalker_f11=%s &" &
                     val=(ws02a_array.numbers[3])
file text write form="kalker_f22=%s &" &
                     val=(ws02a_array.numbers[4])
file text write form="kalker_f23=%s &" &
                     val=(ws02a_array.numbers[5])
file text write form="axle_load=%s &" &
                     val=(ws02a_array.numbers[6])
file text write form="contact_angle=%s &" &
                     val=(ws02a_array.numbers[7])
file text write form="conicity=%s &" &
                     val=(ws02a_array.numbers[8])
file text write form="contact_angle_par=%s" &
                     val=(ws02a_array.numbers[9])

file text close file="$wheelset_file_name.wsd"


-------------------
file generated by above macro:
-------------------

ws2a_parameters parameter_set_name=ws02a_array &
kalker_f11=1.37E+04 &
kalker_f22=1.11E+04 &
kalker_f23=20.6 &
axle_load=1.0E+04 &
contact_angle=8.0E-02 &
conicity=0.13 &
contact_angle_par=12.5
```

```
!USER_ENTERED_COMMAND vx_part
!HELP_STRING sets init_vx for parts defined in all_parts
!
! $all_parts:t=part:c=0

for var=a_var obj=$all_parts
  if condition=(db_exists(a_var.center_of_mass_marker))
    par cre rigid init part=(a_var) vx=(velocity_x)
  end
end




!USER_ENTERED_COMMAND create_init_velocity
!HELP_STRING set init velocity to all parts
!
! $vx_default:t=real

if condition=(!db_exists("velocity_x"))
  var create var=velocity_x real=$vx_default units=velocity
else
  var modify var=velocity_x real=$vx_default
end
vx_part all_parts=*
```

```
!USER_ENTERED_COMMAND create_axle_list
!
! $axle_number:t=integer:u=1
! $axle_name:t=string:c=0:u=PAR
! $constraint_location:t=location:c=0:u=0,0,0
! $rolling_radius:t=real:c=0:u=0.5

if condition=(!db_exists("axle_number"))
  var create var=axle_number integer=$axle_number
  var cre var=axle_list string=$axle_name
  data cre matrix full mat=axle_loc &
           values=$constraint_location row=3 column=$axle_number units=length
  data cre matrix full mat=roll_rad &
           values=$rolling_radius row=1 column=$axle_number units=length
else
  :
  :
end
```

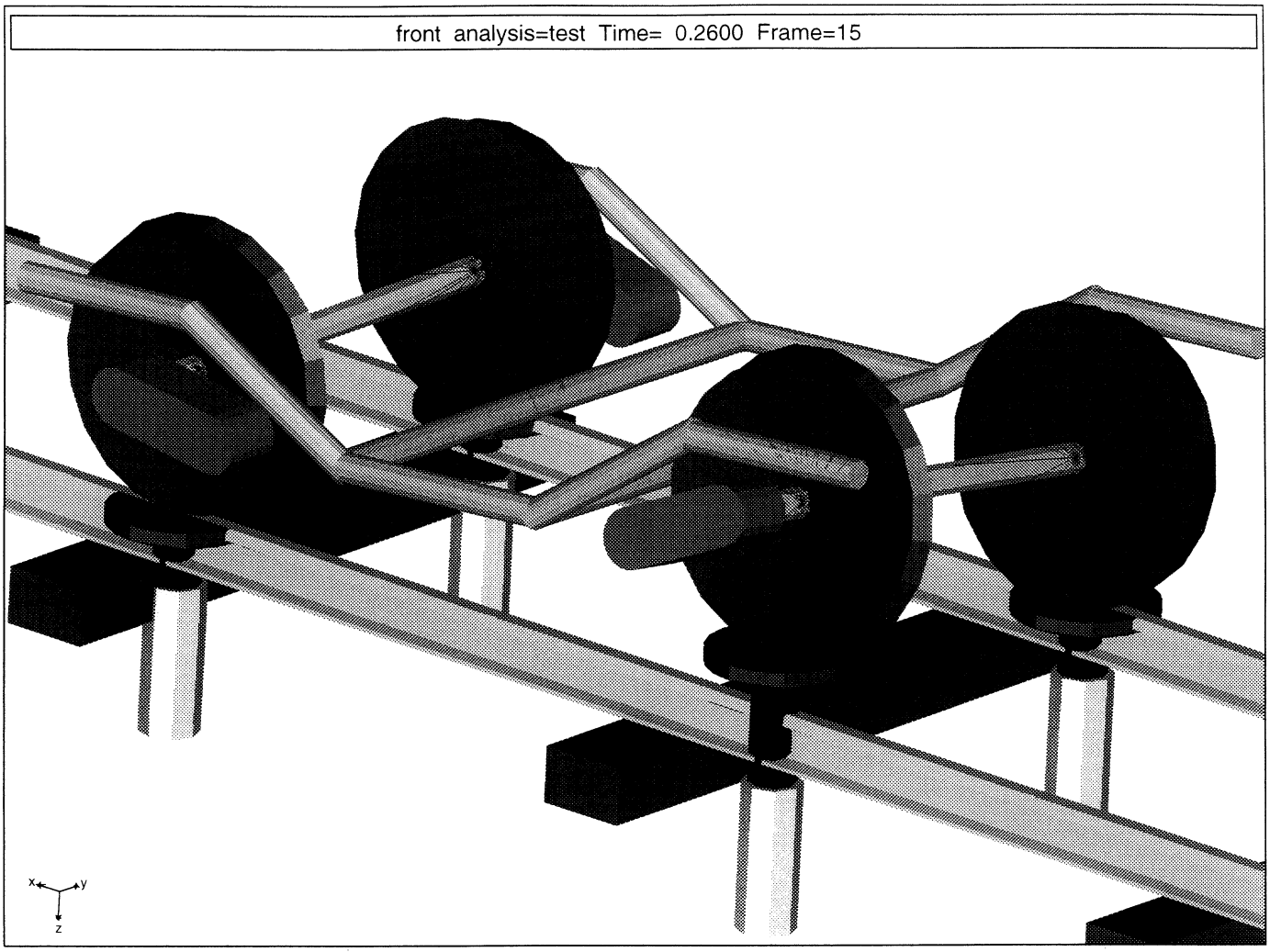
```
!USER_ENTERED_COMMAND add_constraints
!
! $wheel_set_level:t=list(PRESCRIBED_DISPLACEMENT,LEVEL_I,LEVEL_IIa,LEVEL_IIb)
!
!
! add body constraint for prescribed displacement (inplane to prevent x-moveme
!
!
if condition=("$wheel_set_level"=="PRESCRIBED_DISPLACEMENT")
  add_body_constraint body_part=(eval(body_to_be_fixed)) &
       constraint_loc=(eval(body_inp_x_loc [1])), &
                      (eval(body_inp_x_loc [2])), &
                      (eval(body_inp_x_loc [3]))
end

for var=i_var start=1 end=(axle_number)
   if condition=("$wheel_set_level"=="PRESCRIBED_DISPLACEMENT")
      if condition=(!db_exists(eval(axle_list[i_var]//"_trf")))
         ws01 wheel=(eval(axle_list[i_var])) &
                trf_index=(eval(i_var)) &
         con_loc=(eval(axle_loc.values[1,i_var])), &
                 (eval(axle_loc.values[2,i_var])), &
                 (eval(axle_loc.values[3,i_var]))
      end
      cre_dis_con wheel_part=(eval(axle_list[i_var])) &
                  rolling_radius=(eval(roll_rad.values[1,i_var]))
   else
     :
     :
     :
   end
end
```

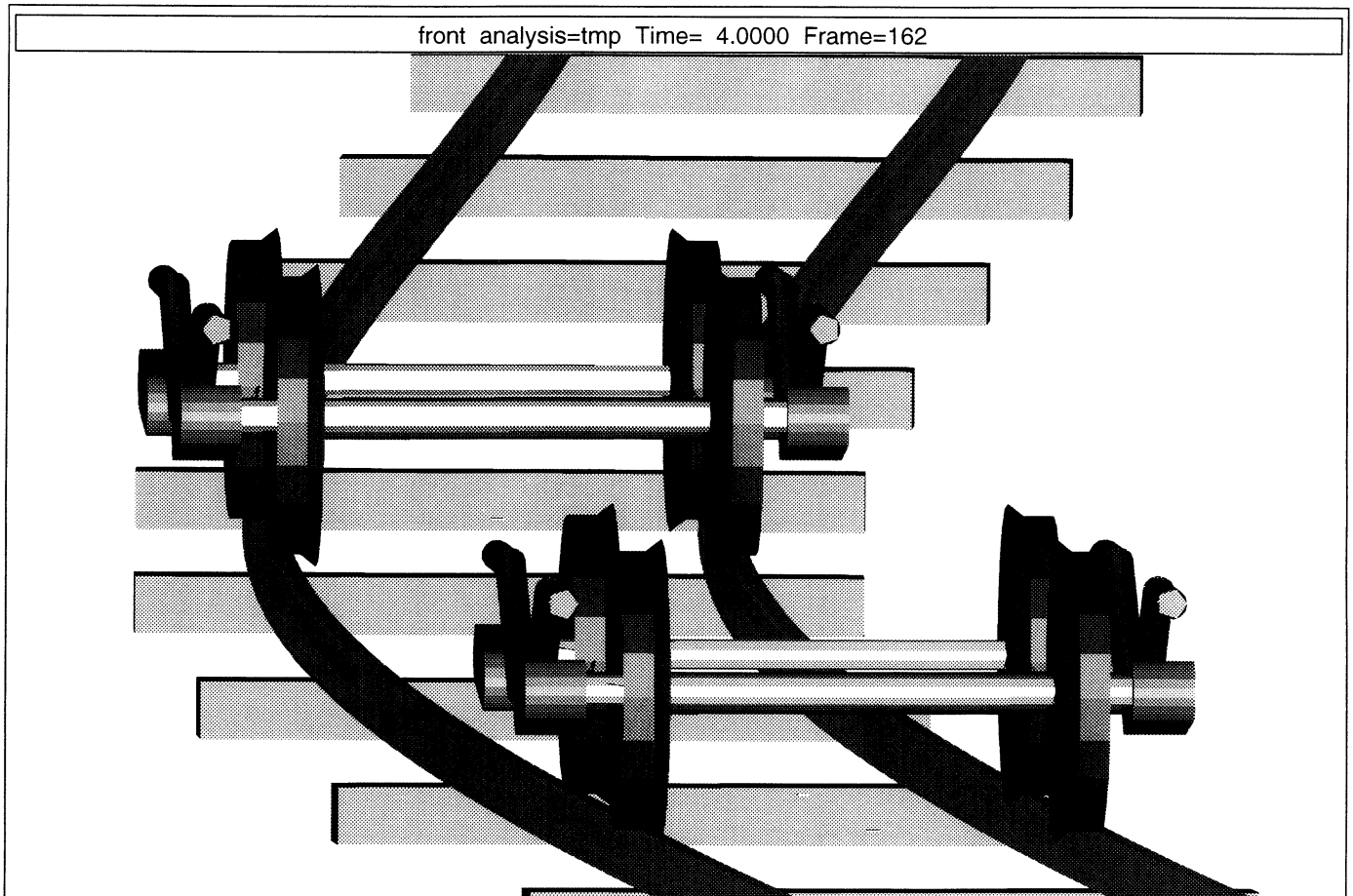
Applications Level I

- Derailment safety
- Vertical comfort
- Eigenmodes/frequencies
- Prescribed displacement
- Roll coefficient
- Static tests

Applications Level II

- Level IIa:
  - Vertical comfort
  - Lateral comfort (for small lateral displacements)
  - Stability (critical speed)
  - ABCD-matrices for spectral analysis
- Level IIb:
  - Longitudianl train dynamics
  - Large traction/braking forces


front analysis=test Time= 0.2600 Frame=15
x
y
z


front analysis=tmp Time= 4.0000 Frame=162

Applications Level III

- Track loads
- Lateral comfort
- Vehicle behaviour in curves
- Stability
- Wear predictions
- Influence profile shape on running quality
- Comparison of suspension design

Outlook

- Productive usage has started
- Non-linear wheelset level III under development
- Feedback (and/or orders) from different countries wanted