Modelling of a dynamical system using hybrid Chi

Citation for published version (APA):

Document status and date:
Published: 01/01/1999

Document Version:
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher’s website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license above, please follow below link for the End User Agreement:
www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:
openaccess@tue.nl
providing details and we will investigate your claim.
Modelling of a dynamical system using hybrid $\chi$.

René van Rooij

January 3, 2000

Contents

1 Introduction 1

2 The model: a single wheelset on a straight railroadtrack 2

3 Implementing the model using hybrid $\chi$ 4

4 Simulation results: hybrid $\chi$ 6

5 Implementing the model using MATLAB 9

6 Simulation results: MATLAB 9

7 Conclusions and recommendations 9

A hybrid $\chi$ code 11

B MATLAB code 13

1 Introduction

Hybrid $\chi$ is a formalism developed by the group Systems Engineering of the Eindhoven University of Technology, Faculty of Mechanical Engineering. This formalism can be used to model and analyze hybrid systems. Characteristic of hybrid systems is that they show discontinuous changes of state, e.g. a mechanical system in which collisions occur. Up till now hybrid $\chi$ has mainly been used to model hybrid industrial systems (e.g. beer breweries). The questions that will be considered in this report are whether or not hybrid $\chi$ can be used for modelling and analysing the dynamical behavior of mechanical systems and (in case the answer to the first question is yes) what advantages hybrid $\chi$ has over more conventional modelling techniques.

The most direct way of answering these questions is to simply try and implement a model of a mechanical system using hybrid $\chi$. In this case a model describing the dynamical behavior of a single wheelset upon a railroad track will be considered. This model which has been developed at the Delft University of Technology by Meijaard and de Pater and shows some interesting characteristics. First of all the model shows non-linear behavior which is caused by the contact between wheel and track. Second of all collisions (discontinuities) between flanges and track play an important part.

In order to answer the second part of the question (what are the advantages of hybrid $\chi$ over other, more conventional, techniques) the model will also be implemented using MATLAB.
In chapter 2 the considered model will be presented. The way in which this model has been implemented using hybrid $x$ is shown in chapter 3. The results obtained using this implementation are presented in chapter 4. The way in which the model has been implemented using MATLAB is shown in chapter 5. And, as to be expected, the results obtained using this implementation are presented in chapter 6. The final conclusions and recommendations are to be found in chapter 7.

2 The model: a single wheelset on a straight railroadtrack

Figure 1 shows a crude sketch of the system that will be considered here.

![Figure 1: Wheelset on railroad track.](image)

In figure 2 the various coordinate frames used to describe the systems kinematics are presented. Only the y-directed displacement ($v$) and the angular displacements around the $z$-axis ($\phi$) will be considered. The displacement of the wheelset along the track ($s$) depends upon the $x$-directed velocity of the system ($V$) which serves as an input parameter to the model.

![Figure 2: The various coordinate frames used to describe the kinematical behavior of the system.](image)

The title of this chapter already includes one assumption being the one regarding the straightness of the railroad track. This assumption implies that there are no forces acting upon the wheelset that are caused by deflections of the track. Forces acting upon the wheelset are caused by the contacts on the one hand and the springs with which the wheelset is fixed to the carriage on the other hand. The springs are fixed to the wheelset as illustrated in figure 3. Clearly, if displacements remain small, the springs influencing $v$ do not influence $\phi$ and vice versa.

Another assumption is that the forces caused by the wheelset-railroadtrack contact can be described using Kalker’s linear contact law. Eventually all this leads to the following set of expressions describing the various forces in terms of the state variables $v$, $\dot{v}$, $\phi$ and $\dot{\phi}$.

$$F_{spring} = c_v \cdot v$$  \hspace{1cm} (1)
Figure 3: Illustration of the way in which the springs are fixed to the wheelset.

\[ F_{\text{contact}} = -K_y \cdot \left( \frac{\dot{v}}{V} - \phi \right) \]  \hspace{1cm} (2)

\[ M_{\text{spring}} = c_\phi \cdot \phi \]  \hspace{1cm} (3)

\[ M_{\text{contact}} = -K_\phi \cdot \left( \frac{\dot{b}^2 \cdot \phi}{V} - \Gamma^2 \cdot v \right) \]  \hspace{1cm} (4)

Herein \( c_x = c_y \) and \( c_\phi = b_1^2 \cdot c_2 \). \( K_y \) and \( K_\phi \) are coefficients with dimension [N] which follow from Kalker's theory.

Using these expression one is now able to express \( \dot{v} \) and \( \dot{\phi} \) in terms of \( v, \dot{v}, \phi \) and \( \phi \).

\[ \ddot{v} = \frac{F_{\text{contact}} - F_{\text{spring}}}{m} \]  \hspace{1cm} (5)

\[ \ddot{\phi} = \frac{M_{\text{contact}} - M_{\text{spring}}}{J} \]  \hspace{1cm} (6)

With this the model is almost completed, the only thing that has to be accounted for yet are the collisions that can occur between wheel-flange and railtrack. Modelling these collisions also calls for yet another assumption, being the one regarding the energy dissipation involved with such a collision. The assumption is that, in case a collision occurs, the \( y \)-directed velocity \( \dot{v} \) changes direction and reduces 5%. Using \( \dot{v}^- \) and \( \dot{v}^+ \) to denote the velocity before and after collision, this can also be formulated in following way:

\[ \dot{v}^+ = -r \cdot \dot{v}^- \]  \hspace{1cm} (7)
The model is now complete and ready for implementation. In the next chapter the implementation using hybrid $\chi$ will be discussed.

3 Implementing the model using hybrid $\chi$

Before starting to describe the actual process of implementing the model using hybrid $\chi$ I'd like to stress that the idea is not to give an exhaustive description of hybrid $\chi$ as it is. I would merely like to show a way in which hybrid $\chi$ can be used for modelling mechanical systems. An extensive description of hybrid $\chi$ itself can be found in [1] and [2].

While implementing the model using hybrid $\chi$ the wheelset was considered as a single object whose state is affected by four other objects containing a mathematical description of the forces that are acting upon the wheelset. From the various possible approaches this one has been chosen because it clearly emphasizes the object oriented approach of hybrid $\chi$.

The objects representing the various forces working upon the wheelset are all more or less alike. The input they need consists of one or two state variables whilst the output is nothing but the value of the force which is acting upon the wheelset. The functional relationship between these input and output variables was already presented in the previous chapter in the form of equations 1 till 4. Below one of these objects namely FSPRING will be discussed in further detail.

The hybrid $\chi$ code describing FSPRING is given below 1.

```
proc FSPRING(v_::displ, fs_::force, cv: real)=

1 |[ v::displ, fs::force
2 |  v--v, fs--fs
3 |  % fs = cv*v ]
```

In the first line of this code the various links through which the 'process' FSPRING can communicate with other processes are defined. There are two different links, one for importing the current value of $v$ ($v_::$ displ) and one for exporting the associated value for $F_{spring}$ ($fs_::$ force). Besides these two links there is another, third input parameter that the process needs namely the stiffness of the spring connecting the wheelset to the carriage $c_v$ which is of type real ($cv$: real).

In the second line of this code the data types of the various variables that will be used within the process are defined. Obviously the displacement $v$ is of type displ and the force $fs$ is of type force.

The third line of the code shows which variable is associated with what link. The value for $v$ is associated to link $v_-$ and the value for $fs$ is associated to the link $fs_-$.

Finally the fourth line contains the actual functional relationship between $v$ and $fs$.

Complete listings of the other objects describing one of the forces acting upon the wheelset ($F_{contact}$, $M_{spring}$ and $M_{contact}$) are given in appendix A.

The object describing the wheelset itself is slightly more complex than the objects describing the functional relationship between state variables and forces, therefore this object will also be discussed in further detail below.

The main problem with the object describing the wheelset is that the collisions of the flanges with the track have to be accounted for. Within hybrid $\chi$ it is possible to model discrete event

---

1 The numbers printed at the start of each line are there purely for ease of use. In the actual hybrid $\chi$ code these numbers are omitted.
processes that run parallel to continuous processes. Both types of processes can influence one another. The way in which this can be done is twofold. First of all it is possible to build guarded command statements in the discrete part that depend upon the value of one of the continuous variables. The tool with which this can be done is the so called nabla operator ($\nabla$). In this case one has to continuously check whether or not $v$ reaches a value at which a flange will make contact with either the left or the right side of the track. If $v$ reaches a level at which contact occurs the state of the system changes instantaneously. As already mentioned in the previous chapter the only change that has to be made is that the velocity $\dot{v}$ changes direction and is reduced by 5%. Within hybrid $\chi$ this state change can be implemented by reassigning a value for $\dot{v}$ the moment one of the guarded command statements becomes true. The code describing the object WHEELSET is given below. The collisions between flanges and track are modelled in lines 12 till 14.

```plaintext
1 proc WHEELSET(fs_, fk_:* force, ms_,mk_:* moment, v1,v2:* displ
2 , v0, dv0, f10, df10, m, J, r, D: real) =
3 1[v:displ, dv:vel, f10, f2: angle, df1: angvel
4 ; v:=v0; dv:=dv0; f1:=f10; df1:=df10
5 | fs_:* fs, f1_:* fk, ms_:* ms, mk_:* mk, v1:* v, v2:* v, dv_:* dv
6 | f1:= f1, f12:* f1, df1:* df1
7 | % v'=dv
8 , dv'=(fk-fs)/m
9 , f1'=df1
10 | df1'=(mk-ms)/J
11 | ![nabla v <-D -> dv:=r*dv; v:=D
12 | ![nabla v > D -> dv:=-r*dv; v:=D
13 ]
14 ]
```

For a more detailed description of the way in which both types of processes (discrete event and continuous) can influence one another see [1] and [2].

All objects of which the complete model should consist have now been defined. The only thing that has to be done now in order to obtain the full model is to connect these objects in the correct way. The latter is done in the small piece of code presented below.

```plaintext
1 syst SYSTEM(v0,dv0,f10,df10,m,J,c1, cv, f2, ms, cb, v1, v2, df1, df2, f3, f4, f5, f6, f7, f8, f9, f10)
2 ![ v1, v2:* displ, dv:* vel, f1, f2:* angle, df1:* angvel, f3, f4:* force,
3 | f5:* moment
4 | FSPRING(f10, f3, cv) || MSPRING(f1, f5, cb, f2)
5 | FCONTACT(f10, f2, v1, v2, cv) || MCONTACT(f1, f5, cv, f2, cb, f3, f4, v1, v2)
6 | WHEELSET(f3, f4, f5, f6, f7, f8, f9, f10, m, J, r, D)
7 ]
```

In this piece of code the system SYSTEM is defined. It consists of the objects FSPRING, MSPRING, FCONTACT, MCONTACT and WHEELSET. These objects are connected together via the channels which are defined in lines 2 and 3. By consequently using the correct channel-names as input parameters for the various objects the connections are established. A graphical representation of the finally obtained system is shown in figure 4.

What remains is the definition of the simulation that one wants to perform. The simulation is defined by assigning the right values for the various system parameters in an experiment. The following small piece of code shows how this is done:

```plaintext
1 xper = ![ SYSTEM( 0.1, 0.1, 0.1, 0.1, 1603, 773.7537938, 563273.943
```

5
Compilation of the finally obtained '.chi' file results in an executable file that can be run and will result in a list of data describing the change of the system state during the course of time. The results of several of such simulations will be discussed in the following chapter which has the promising title:

4 Simulation results: hybrid $\chi$

The simulation results obtained from the model implemented using hybrid $\chi$ will be compared to the results presented in the articles by Meijaard [4] and de Pater [3]. The most important result presented in these articles was a graph showing, in a very schematic way, the dynamical behavior that can be expected as a function of the dimensionless nominal velocity with which the wheelset is dragged along the track. A copy of this figure is shown in figure 5.

The most important conclusions are already incorporated into this picture. For $V$ smaller than 1 the trivial solution $v = 0$ and $\phi = 0$ will be obtained. For values of $V$ slightly larger than 1 the dynamical behavior of the system will become chaotic. For values of $V$ that lie somewhere inbetween 1.165 and 1.318 two different asymmetric periodic solutions exist. 'Asymmetric' in that sense that for these solutions the wheelset will collide with only one side of the track. For values of $V$ larger than 1.318 there's also a symmetric periodic solution (in that case the wheelset collides with both track sides). In case $V$ is larger than 1.67 this symmetric periodic solution is the only one that still exists.

In case the model that has been implemented using hybrid $\chi$ shows the same relation between nominal velocity and dynamical behavior than it is allowed to at least suspect the model to be
Figure 5: Schematic representation of the dynamical behavior of the system as a function of the dimensionless nominal velocity (V) with which the wheelset is dragged along the railroad track.

correct. Whether or not hybrid $\chi$ is accurate enough to analyse systems showing even stronger non-linearities is not the issue here. In order to check this one has to critically look at the solver that is used by hybrid $\chi$. At the moment however, checking whether or not it is possible to model mechanical systems using hybrid $\chi$ is the top priority so accuracy will not be considered here.

The results from the simulations that have been performed are judged from pictures in which $u$ is plot as a function of $\phi$ (or vice versa). For several values of $V$ the results are shown below.
By looking at that part of the curve which describes the last few seconds of the simulation one can judge what kind of dynamical behavior one is dealing with. The pictures above should however not be looked upon as being conclusive evidence that the dynamical behavior of the system is what it is supposed to be, however they do however give an indication. Looking at the results one can therefore conclude that it is very well possible that the implemented model is correct. For example, for $V = 0.99$ the solution $v = 0, \phi = 0$ seems to be stable whereas for $V = 1.01$ this solution seems no longer to be stable. For $V = 1.4$ three different stable solutions were found which corresponds with what was to be expected from the article by Meijaard and de Pater.

All in all it looks like it is very well possible to model and analyse the dynamical behavior of mechanical systems using hybrid $\chi$. The next question was, does this approach have advantages compared to more conservative modelling methods? In order to answer this question the model has also been implemented using MATLAB. The process of implementing this model is described in the next chapter.
5 Implementing the model using MATLAB

When implementing the model using MATLAB an approach was used which is very similar to the approach used by the ODE-solvers currently available under MATLAB. This approach is convenient because the only thing one actually needs in order to solve the system of equations presented in chapter 2 is a somewhat modified version of an ODE-solver. The small modification that is referred to here is the implementation of equation 7. The moment a collision occurs one has to deal with an instantaneous change of state of the system. The standard ODE-solvers available within MATLAB can not handle this type of discontinuities (at least not without any modifications). This problem is avoided by simply using a handcrafted ODE-solver (RUNGE.M) which has equation 7 integrated into it. This already illustrates one of the drawbacks of MATLAB compared to hybrid $\chi$, MATLAB needs very problem specific solvers whereas hybrid $\chi$ doesn’t. Apart from the solver, a function file has been created which contains the differential equations describing the dynamical behaviour of the system inbetween two collisions (WHEELSET.M). A complete listing of both files can be found in appendix B.

6 Simulation results: MATLAB

The results that have been obtained from these MATLAB files are almost similar to the results that have been obtained using hybrid $\chi$. The differences that have been found are of the same order as the difference in accuracy of the solvers that are being used in hybrid $\chi$ ($\pm 10^{-6}$) and MATLAB ($\pm 10^{-16}$) respectively. This observation raises the suspicion that improving the accuracy of the solver used by hybrid $\chi$ will result in results that have an accuracy similar to the results obtained using MATLAB.

7 Conclusions and recommendations

Based upon the final observation in the previous (very small) chapter, it is concluded that the desired accuracy of the final results is a weak argument for preferring MATLAB over hybrid $\chi$.

An advantage of hybrid $\chi$ over MATLAB, in my opinion, is the ease with which discontinuities can be modelled. For a model as small as the one considered here the amount of additional work that has to be done when using MATLAB isn’t such a big deal. However, as more complex models will be considered, this amount of work will most probably increase rapidly. This increase of work will mainly be due to the fact that for each restriction imposed upon the system the solver has to be modified. Therefore, the more restrictions, the more complex the solver will become.

As you can see I'm not yet prepared to make very hard statements regarding the use of hybrid $\chi$ in modelling mechanical systems. However, based upon my experiences in using it I have the strong suspicion there are possibilities in putting hybrid $\chi$ to good use. The most promising possibility I see is the use of hybrid $\chi$ as a sort of modelling tool for multibody systems in which one has to deal with discontinuities such as colliding bodies. I know that some research has been done on this topic but not at this university. Since hybrid $\chi$ is now available, and I have to take a look at multibody systems that do know discontinuities (the CD-drive line system autobalancing unit for example) there is a very good opportunity to increase the knowledge we have in the field of hybrid modelling of mechanical systems. Obviously this will be advantageous to both the Engineering Dynamics and the Systems Engineering department of our faculty. With this in mind I recommend to do some further research in making hybrid $\chi$ more suitable for modeling the above mentioned type of systems. The latter can be done for example by developing some sort of "lego" box which contains some of the most commonly encountered machine parts in mechanical engineering.
References


A hybrid $\chi$ code

type displ = [m],
  vel = [m/s],
  angdispl = [-],
  angvel = [°/-1],
  force = [N],
  moment = [N.m]

proc FSPRING(v_::-* displ, fs_::-* force, cv: real)=
  % [v::displ, fs::force]
  [v-* v, fs-* fs]
  [% fs = cv*v ]

proc FCONTACT(fi2::-* angdispl, dv_::-* vel, fk_::-* force, V,kv: real)=
  % [fi:: angdispl, dv:: vel, fk:: force]
  [fi2-* fi, dv-* dv, fv_* fk]
  [% fk = -ky*(dv/V-fi) ]

proc MSPRING(fi1::-* angdispl, ms_::-* moment, b1,cfi: real)=
  % [fi:: angdispl, ms:: moment]
  [fi1-* fi, ms-* ms]
  [% ms = cfi*b1^2*fi ]

proc MCONTACT(v2::-* displ, dfi_::-* angvel, i&_::-* moment, 
  V,kf, gamma, b, kfi, V: real)=
  % [v:: displ, dfi:: angvel, mk:: moment]
  [v2-* v, dfi-* dfi, mk-* mk]
  [% mk = -kfi*(b^2*dfi/V+v*gamma-2) ]

proc WHEELSET(fs_, fk_::-* force, ms_,mk_::-* moment, v1,v2::-* displ 
  , dv_::-* vel, fi1,fi2::-* angldispl, dfi_::-* angvel 
  , v0, dv0, fi0, dfi0, m, J, r, D: real) =
  % [v:: displ, dv:: vel, fi::angdispl, dfi::angvel, fs,fs::force, ms,ms::moment 
  ; v::=v0; dv::=dv0; fi::=fi0; dfi::=dfi0 
  ; fs_* fs, fk_* fk, ms_* ms, mk_* mk, v1-* v, v2-* v, dv_* dv 
  ; fi1-* fi, fi2-* fi, dfi_* dfi]
  [% v'=dv 
  , dv'=(fk-fs)/m 
  , fi'=dfi 
  , dfi'=(mk-ms)/J 
  *[nabla v < D -> dv::=-r*dv; v::=-D 
  [nabla v > D -> dv::=-r*dv; v::=D 
  ]]

syst SYSTEM(v0,dv0,fi0,dfi0,m,J,cv,cfi,kv,kfi,b1, gamma,V,r,D:real) =
  % [v1,v2::displ, dv::vel, fi1,fi2::angdispl, dfi::angvel, fs,fs::force, 
  ms,ms::moment]
  | FSPRING(v1,fs,cv)    || MSPRING(fi1,ms,b1,cfi) 
  || FCONTACT(fi2,dv,fx,V,kv) || MCONTACT(v2,dfi,ms, gamma,b,kfi,V) 
  || WHEELSET(fs,fs,ms,ms,m,v1,v2,fi,fi2,dfi,fi0,v0,dfi0,m,J,r,D)
xper = [ SYSTEM( 0.1, 0.1, 0.1, 0.1, 1503, 773.7537938, 563273.943
, 289976.9463, 808298.1082, 808298.1082, 0.7175, 0.6
, 0.5, 1.4, 0.95, 0.02 ) ]
## B MATLAB code

### WHEELSET.M

```matlab
% R.A. van Rooij 20 may 1999
% ----------------------------------
% filename: dimensieset.m
% objective: function file to be used in combination with
% rungekutta for analysing the dynamical behaviour
% of a wheelset traveling along a railway track. In
% contrast to the equations used in wheelset.m this
% file uses a dimensioned model
%
% input: x = a column vector containing the state of
% the system
% V = the nominal speed at which the wheelset travels
% along the railroad track
% output: dx = a column vector containing the change of
% state at moment t
% ----------------------------------

function dx=wheelset(x,V)

dx=zeros(4,1);
v=x(1);
phi=x(2);
vpnt=x(3);
phipnt=x(4);

% defining the various necessary system parameters
K=900000;
m=1500;
J=843.75;
cx=937000;
cy=600000;
b=0.75;
b1=0.60;
Gamma=0.5;

F=-K*(vpnt/V-phi);
M=-K*(b^2*phipnt/V+Gamma^2*v);

dx(1)=vpnt;
dx(2)=phipnt;
dx(3)=(F-cy*v)/m;
dx(4)=(M-cx*b1^2*phi)/J;
```

### RUNGE.M

```matlab
% R.A. van Rooij 20 may 1999
% ----------------------------------
% filename: runge.m
% objective: function file to be used in order to solve
```
initial value problems using the "classical"
runge-kutta integration scheme

input: x0 = initial state of the system to be analysed
t = row vector containing the starting time, the
the end time and the time stepsize to be used.
fun = name of the function file which returns the
change of state which is needed in order to
calculate the next system state

output: X = matrix containing the system states at the
various moments in time
T = the vector containing the labels describing
the various moments in time

function [X,T]=runge(x0,t,fun)

r=0.95;  \% dissipative coefficient
t0=t(1); tfinal=t(2); dt=t(3);
t=t0;
T=t;
x=x0;
X=[x];
V=30*0.5;

while t<=tfinal
k1=feval(fun,x,V);
k2=feval(fun,x+dt*k1/2,V);
k3=feval(fun,x+dt*k2/2,V);
k4=feval(fun,x+dt*k3,V);
xt=x+dt*(k1/6+k2/3+k3/3+k4/6);

\% Check whether or not the wheelset collides with one of the rails
if abs(xt(1))>=0.02 \% "if" collision
    dtt=dt;
    while \((abs(xt(1))<=0.02+eps)) \& (abs(xt(1))>=0.02-eps)))
        dt=dt*abs(xt(1)-x(1))/abs(xt(1)-x(1));
        k1=feval(fun,x,V);
        k2=feval(fun,x+dtt*k1/2,V);
        k3=feval(fun,x+dtt*k2/2,V);
        k4=feval(fun,x+dtt*k3,V);
        xt=x+dtt*(k1/6+k2/3+k3/3+k4/6);
    end
    xt(3)=-r*xt(3);
    dtt=dt-dtt;
    k1=feval(fun,xt,V);
    k2=feval(fun,xt+dtt*k1/2,V);
    k3=feval(fun,xt+dtt*k2/2,V);
    k4=feval(fun,xt+dtt*k3,V);
    xt=xt+dtt*(k1/6+k2/3+k3/3+k4/6);
    x=xt;
else
    x=xt;
end

14
end

\% Adding a time step
\[ t = t + dt; \]

\% Storing relevant data for the interval \( 0 \leq t \leq t_{\text{final}} \)
if \( t \geq 0 \)
\[ T = [T \ t]; \]
\[ X = [X \ x]; \]
end
end