Track Constraints For Rigid Body Dynamics

(Document Version: 1)

„Pure Vernunft darf niemals siegen“

Tocotronic

Moving along a track in a dynamic simulator can be done by defining special constraints for something we want to call a trackjoint. The trackjoint connects an anchor of a moving body with a track system that is fixed on another body.

We consider the relevant entities;

s : Arc length along the curve

$\boldsymbol{P}(s)$: Position on a curve, as a function of the arc length

$\mathbf{T}(s)$: Tangent vector of the curve at $\boldsymbol{P}(s)$.

$\mathbf{N}(s)$: Normal vector of the curve at $\boldsymbol{P}(s)$.

$\mathbf{B}(s)$: Binormal vector of the curve at $\boldsymbol{P}(s)$.

T,N,B are unit vectors perpendicular on each other, according to [Frenet/Serret] they obey the following equations along the curve;

$$\begin{aligned} d\mathbf{T}(s)/ds &= c(s)*\mathbf{N}(s) \\ d\mathbf{N}(s)/ds &= -c(s)*\mathbf{T}(s) + t(s)*\mathbf{B}(s) \\ d\mathbf{B}(s)/ds &= - t(s)*\mathbf{N}(s) \end{aligned} \tag{1}$$

where;

c(s) : Curvature at arc length s.

t(s) : Torsion at arc length s.

ds : The length along the curve at which the moving body is moved in one simulation step along the track curve.

On most curves the above equations have no idea about what we would call an up-direction. We want to identify the B - vector with that direction, but we will not always be happy with that and might want to tweak the up direction for different reasons. To do this, we introduce the twisted vectors **wT,wN,wB**, which will be the vectors yielded by applying a rotation function wt(s) to the frame and rotating it around **T** with the angle wt.

wt(s): Twist at s; angle to rotate the TNB-Frame around **T**.
wT(s): Twisted **T** vector at s.
wN(s): Twisted **N** vector at s.
wB(s): Twisted **B** vector at s.

It happens to be **wT**(s) = **T**(s).

Since we always have to have these frames, consisting of a position ***P***, and some three axes **T,N,B**, defining an orientation, we want to introduce a notation ***F***, with ***F.P***, ***F*.T**, ***F*.N**, ***F*.B**, being the position and the respective vectors of what [PhysX] calls a pose, meaning being at some position and being orientated. In this terminology we have a comoving frame ***F***(s) and a twisted frame ***wF***(s) with ***F*.T**(s) = ***wF*.T**(s) and ***F.P***(s) = **wF.P**(s) for every parameter s along the curve.

In the most general case we see our track system as being attached to one rigid body and another rigid body moving along those (maybe moving) tracks. So there is a notion of some:

Fm(t) : The pose of the moving body Bm at simulation step t.
Ft(t) : The pose of the body Bt at simulation step t, to which the tracks are attached.
F(s,t) : The point and orientations on the track at parameter s and at time step t.

A(t) : Anchor of Bm; the point and the directions fixed on Bm that are supposed to align with the track. Since it is fixed with the body it will only be a function of time if formulated in the global coordinate system.

t : Simulation step. Imagine it as a point in time with dt being infinitesimal small. In a simulation world it will become the simulation step number with dt = 1.

For a rigid body simulator we have to formulate our constraints in terms of the following equation;

<u>**Jm**</u>***Vm** + <u>**Rm**</u>***Wm** + <u>**Jt**</u>***Vt** + <u>**Rt**</u>***Wt** = 0 (2)

see [Smith], where;

Vm : Linear velocity of the body Bm's COM (Center Of Mass) moving along the track.

Wm : Rotation velocity of the body Bm's COM moving along the track.

Vt : Linear velocity of the body Bt's COM the track is attached to.

Wt : Rotation velocity of the body Bt's COM the track is attached to.

<u>**Jm**</u>, <u>**Rm**</u>, <u>**Jt**</u>, <u>**Rt**</u> : 3xn Matrices, specifying the contribution of the respective velocities to the contraints. Here n is the number of constraints that make up a joint. For our trackjoint we will need six contraints. The total of those four matrices can get combined to one big matrix called 'The Jacobian' [Smith]. In that case the above equation will look like this:

<u>**J**</u>***V** = 0

But this is for art, let us go on with (2): The physics engines (e.g. [ODE] or [PhysX]) take those J and R from our trackjoint and will try to apply constraint forces in order to change the velocities, so that they fulfill the above equation. Note that it is an equation about velocities only.

Each simulation step t, the values will get calculated by our trackjoint for the actual situation, then the engine will fix the velocities and move the two bodies accordingly to their new positions. Since the relative movement of Bm's anchor and Bt's track position will be along $\boldsymbol{F}.\mathbf{T}$(s) and not along the curve $\boldsymbol{F}.\boldsymbol{P}$(s), we take the relative movement along $\boldsymbol{F}.\mathbf{T}$ as our;

$$ds = (\boldsymbol{A}.\boldsymbol{P}(t+dt)-\boldsymbol{A}.\boldsymbol{P}(t))*\boldsymbol{F}.\mathbf{T}(s,t) - (\boldsymbol{F}.\boldsymbol{P}(s,t+dt)-\boldsymbol{F}.\boldsymbol{P}(s,t))*\boldsymbol{F}.\mathbf{T}(s,t)$$

we demand as a precondition that $\boldsymbol{A}.\boldsymbol{P}(t) = \boldsymbol{F}.\boldsymbol{P}(s,t)$, as it would be in a perfect simulation, so we get;

$$ds = (\boldsymbol{A}.\boldsymbol{P}(t+dt) - \boldsymbol{F}.\boldsymbol{P}(s,t+dt)) * \boldsymbol{F}.\mathbf{T}(s,t) \qquad (3)$$

There are some concerns about the fact that the simulation would not be perfect and how this would influence the validity of (3). But first we will have some kind of error correction (see below) and above all secondly, this formular for ds has the advantage to immediately correct any aberrations along F.T: if A.P would for some reason get advanced along the curve, a greater ds would follow with the official track position; if F.P would be advanced, ds would move back. Any other choice would lead the F.P to advance or fall behind our A.P during the simulation and we would have to correct it.

With this ds, we find the new supposed pose of Bm's anchor on the track, $\boldsymbol{wF}$(s+ds), but with a small margin of error, since our anchor was actually moved (and rotated) along the tangent (and according to the values at the start of ds). Also the computation will yield to numerical inaccuracies and the starting values might not be totally exact. But make no mistake: if we could compute the equations with infinite accuracy and the timestep would be infinitesimal small and the starting data exact, equation (2) would guarantee that all future poses would be correct, even if it is only calculating the velocities (see [Smith]).

To correct these errors, an error reduction parameter is used, by extending (2) to;

$$\underline{\mathbf{Jm}}*\mathbf{Vm} + \underline{\mathbf{Rm}}*\mathbf{Wm} + \underline{\mathbf{Jt}}*\mathbf{Vt} + \underline{\mathbf{Rt}}*\mathbf{Wt} = erp/dt * \mathbf{E} \qquad (4)$$

erp : Error reduction parameter, runs from 0 (no reduction) to 1 (full reduction inside one simulation step).

E : An n - dimensional vector, providing one number per constraint that describes the amount of error as a distance or angle.

The above V and W refer to the movement of the centers of masses of the bodies, from those we get the respective velocities of the anchor and the twisted frame as:

$$\begin{aligned}
\mathbf{Va} &= \mathbf{Vm} + \mathbf{Wm} \times (\boldsymbol{A.P} - \boldsymbol{Fm.P}) \\
\mathbf{Wa} &= \mathbf{Wm} \\
\mathbf{Vp} &= \mathbf{Vt} + \mathbf{Wt} \times (\boldsymbol{wF.P} - \boldsymbol{Ft.P}) \qquad (5) \\
\mathbf{Wp} &= \mathbf{Wt}
\end{aligned}$$

(with 'x' being the vector cross product). Note again that these hold only for the centers of masses (COM) of the bodies. The Fm and Ft will be the poses of the COMs of the bodies and have to get calculated as such.

Our obligation now is to calculate the J,R and E for our constraints;

Constraint 1, no relative movement along *wF*.N; on a track a linear movement sideways the track is prevented by the rails, so the relative movement in that direction would be zero. If there would be an aberration, a small correcting velocity will get introduced in the opposite direction;

$$(\mathbf{Va} - \mathbf{Vp}) * \boldsymbol{wF}.\mathbf{N} = erp/dt * -(\boldsymbol{A.P} - \boldsymbol{wF.P}) * \boldsymbol{wF}.\mathbf{N}$$

The relative velocity on the left side is the velocity of the anchor as seen from the point on the track, an aberration of the anchor in positive wF.N direction has to lead to a compensating velocity in the opposite direction, hence the '-'.

With (5) and the rule for the vector spat product, $(\mathbf{A}\times\mathbf{B})*\mathbf{C} = (\mathbf{B}\times\mathbf{C})*\mathbf{A} = (\mathbf{C}\times\mathbf{A})*\mathbf{B}$ we get;

$$((\mathbf{Vm} + \mathbf{Wm} \times (\boldsymbol{A.P} - \boldsymbol{Fm.P})) - (\mathbf{Vt} + \mathbf{Wt} \times (\boldsymbol{wF.P} - \boldsymbol{Ft.P})) * \boldsymbol{wF}.\mathbf{N}$$
$$= \boldsymbol{wF}.\mathbf{N}*\mathbf{Vm} + ((\boldsymbol{A.P}-\boldsymbol{Fm.P})\times\boldsymbol{wF}.\mathbf{N})*\mathbf{Wm} - \boldsymbol{wF}.\mathbf{N}*\mathbf{Vt} - ((\boldsymbol{wF.P}-\boldsymbol{Ft.P})\times\boldsymbol{wF}.\mathbf{N})*\mathbf{Wt}$$

from which we can take the elements of (4) as;

$$\begin{aligned}
\mathbf{Jm1} &= \boldsymbol{wF}.\mathbf{N}\\
\mathbf{Rm1} &= (\boldsymbol{A.P} - \boldsymbol{Fm.P}) \times \boldsymbol{wF}.\mathbf{N}\\
\mathbf{Jt1} &= -\boldsymbol{wF}.\mathbf{N}\\
\mathbf{Rt1} &= \boldsymbol{wF}.\mathbf{N} \times (\boldsymbol{wF.P} - \boldsymbol{Ft.P})\\
E1 &= (\boldsymbol{wF.P} - \boldsymbol{A.P}) * \boldsymbol{wF}.\mathbf{N}\\
f1min &= infinite\\
f1max &= infinite
\end{aligned} \qquad (6)$$

Here fmin will limit the maximum force applied in $-\boldsymbol{wF}.\mathbf{N}$ direction and fmax be the maximum force applied in $\boldsymbol{wF}.\mathbf{N}$ direction to guarantee (4). We might want to limit these forces, in which case (4) would be violated if the necessary forces become greater.

Constrain 2, no relative movement in -*wF*.B direction; the track supports the anchor by applying a force in the ***wF*.B** direction, but if anything would lift it, the constraint will offer no resistance. This plays quite similar to constraint 1;

$$(\mathbf{Va} - \mathbf{Vp}) * \boldsymbol{wF}.\mathbf{B} = erp/dt * -(\boldsymbol{A.P} - \boldsymbol{wF.P}) * \boldsymbol{wF}.\mathbf{B}$$

yielding;

Jm2 = ***wF*.B**

Rm2 = (***A.P* - *Fm.P***) x ***wF*.B**

Jt2 = -***wF*.B**

Rt2 = ***wF*.B** x (***wF.P*** - ***Ft.P***)

E2 = (***wF.P* - *A.P)*** * ***wF*.B**

f2min= 0

f2max= infinite (7)

Constraint 3, a screwdriver rotation around *wF*.T according to torsion and twist; while the anchor is moving along the tangent direction it is supposed to follow the torsion of the curve t(s) from (1) plus our twist wt(s).

***wF*.T** * (**Wa**-**Wp**) - (t(s) + dwt(s)/ds) * ***wF*.T** * (**Va**-**Vp**)
= erp/dt * (***A*.T**x***wF*.T** + **A.N**x***wF*.N** + **A.B**x***wF*.B**) * ***wF*.T**

If the curve's torsion would be zero and the twist would not change and we would have a perfect aligment of the N and B axes of the anchor and the twisted frame, the above equation would say that there'll be no relative rotation around wF.T. But since we have to rotate with the torsion and the changing twist as we move along the curve in wF.T direction we have to subtract the second term to demand some specific amount of rotation. That the torsion computes into the equation like this can be seen from (1): t(s) specifies an amount of rotation around T for the vectors N and B. Additionally we demand a little rotation to correct the aberrations of the vectors N and B as far as the correcting rotation is along wF.T, of course. From (5) we get for the left side;

wF*.T** * (**Wm**-**Wt**) - (t(s) + dwt(s)/ds) * ***wF*.T** * ((**Vm** + **Wm** x (A.P* - *Fm.P***)) - (**Vt** + **Wt** x (***wF.P*** - ***Ft.P***))

wich yields us;

Jm3 = -(t + dwt(s)/ds) * ***wF*.T**

Rm3 = ***wF*.T** - (t + dwt(s)/ds) * (***A.P* - *Fm.P***) x ***wF*.T**

Jt3 = (t + dwt(s)/ds) * ***wF*.T**

Rt3 = -***wF* .T** + (t + dwt(s)/ds) * (***wF.P* - *Ft.P***) x ***wF*.T**

E3 = (***A*.N** x ***wF*.N** + ***A*.B** x ***wF*.B**) * ***wF*.T**

t3min = infinite

t3max = infinite (8)

The forces in this case actually become torques.

Constraint 4, no rotation around the normal direction; from (1) you can see that there is no such rotation by the missing terms for dT/ds and dB/ds which would need some component in B or T direction respectively to rotate around N. Note that this relates to F.N and not to the twisted wF.N.

***F*.N** * (**Wa-Wp**) = ***F*.N** * (**Wm-Wt**)

= erp/dt * (***A*.T**x***wF*.T** + ***A*.N**x***wF*.N** + ***A*.B**x***wF*.B**)*** *F*.N**

Still we would allow such a rotation along F.N, if it brings our anchor and the twisted frame closer together.

Jm4 = 0

Rm4 = ***F*.N**

Jt4 = 0

Rt4 = -***F*.N**

E4 = (***A*.T** x ***wF*.T** + ***A*.N** x ***wF*.N** + ***A*.B** x ***wF*.B**) * ***F*.N**

t4min = infinite

t4max = infinite (9)

Constraint 5, proper rotation around F.B; from (1) you see that the curvature c(s) is the rotational change of F around F.B as you go along the curve, since the rotation of N and T around B would be;

$$\mathbf{RN} = d\mathbf{N}/ds - (d\mathbf{N}/ds*\mathbf{B})*\mathbf{B} = -c*\mathbf{T}$$
$$\mathbf{RT} = d\mathbf{T}/ds - (d\mathbf{T}/ds*\mathbf{B})*\mathbf{B} = c*\mathbf{N}$$

so;

$$\boldsymbol{F}.\mathbf{B} *(\mathbf{Wa}-\mathbf{Wp}) - c(s) * \boldsymbol{wF}.\mathbf{T} * (\mathbf{Va}-\mathbf{Vp}) = erp/dt * (\boldsymbol{A}.\mathbf{T}\times\boldsymbol{wF}.\mathbf{T} + \boldsymbol{A}.\mathbf{N}\times\boldsymbol{wF}.\mathbf{N} + \boldsymbol{A}.\mathbf{B}\times\boldsymbol{wF}.\mathbf{B})*\boldsymbol{F}.\mathbf{B}$$

The left side is;

$$\boldsymbol{F}.\mathbf{B} * (\mathbf{Wm}-\mathbf{Wt}) - c(s) * \boldsymbol{wF}.\mathbf{T} * ((\mathbf{Vm} + \mathbf{Wm} \times (\boldsymbol{A.P} - \boldsymbol{Fm.P})) - (\mathbf{Vt} + \mathbf{Wt} \times (\boldsymbol{wF.P} - \boldsymbol{Ft.P}))$$

So we get;

$$\mathbf{Jm5} = - c(s) * \boldsymbol{wF}.\mathbf{T}$$
$$\mathbf{Rm5} = \boldsymbol{F}.\mathbf{B} - c(s) * (\boldsymbol{A.P} - \boldsymbol{Fm.P}) \times \boldsymbol{wF}.\mathbf{T}$$
$$\mathbf{Jt5} = c(s) * \boldsymbol{wF}.\mathbf{T}$$
$$\mathbf{Rt5} = - \boldsymbol{F}.\mathbf{B} + c(s)* (\boldsymbol{wF.P} - \boldsymbol{Ft.P}) \times \boldsymbol{wF}.\mathbf{T}$$
$$E5 = (\boldsymbol{A}.\mathbf{T} \times \boldsymbol{wF}.\mathbf{T} + \boldsymbol{A}.\mathbf{N} \times \boldsymbol{wF}.\mathbf{N} + \boldsymbol{A}.\mathbf{B} \times \boldsymbol{wF}.\mathbf{B})*\boldsymbol{F}.\mathbf{B}$$
$$t5min = infinite$$
$$t5max = infinite \qquad (10)$$

Constraint 6, a motor for accelerating and braking; instead of applying forces to reach this goal, it is much more natural for a physics engine, to take a constraint. Since we deal with velocities, we have to specify a target velocity vTarget, that the engine would try to achieve by limited forces.

$$\boldsymbol{wF}.\mathbf{T} * (\mathbf{Va}-\mathbf{Vp}) = vTarget$$

So the left side becomes;

wF*.T((**Vm** + **Wm** x (***A.P* - *Fm.P***)) - (**Vt** + **Wt** x (***wF.P*** - ***Ft.P***))

this yields;

Jm6 = ***wF*.T**
Rm6 = (***A.P* - *Fm.P***) x ***wF*.T**
Jt6 = -***wF*.T**
Rt6 = ***wF*.T** x (***wF.P*** - ***Ft.P***)
E6 = dt/erp * vTarget
f6min = motor_force_min
f6max = motor_force_max (11)

Simple as it seems and true as it is (because it works) - this is the result of a very hard fight.

Hinterzarten den 30.8.2017 Marc-Michael Horstmann

References;

[Frenet/Serret] https://en.wikipedia.org/wiki/Frenet%E2%80%93Serret_formulas

[PhysX] https://developer.nvidia.com/gameworks-physx-overview

[Smith] Russ Smith, Constraints in Rigid Body Dynamics, Game Programming Gems, Volume 4. Charles River Media, 2004.

[ODE] http://ode.org/