Handling of plasticity in physics engine

June 19, 2016

1 Introduction

The application of modern computer game techniques enables the description of complex dynamic systems such as military vehicles with a high level of detail while still solving the equations in real-time. Film production and war games, in particular, is a key area that have benefited from simulation technology. In practice, games are often accomplished using an open-source platform such like ODE - Smith (2001-2007), Bullet Physics - Coumans (2003-2016) and Box2D - Catto (2007-2015).

Computational methods used in physics engines are divided to modules that handle collision detection and contact description and modules that handle solution of equations in real-time. Equations need to be solved can further be subdivided to be associated to motion, constraints and collisions. Velocity-based formulation is typically used in constraint based rigid body simulation. Friction is typically taken into account and mechanical joints are handled by constraint equations. Detailed description of various components can be found in e.g. Erleben (2005).

Plasticity is not typically taken into account in gaming solutions. Breaking of various objects typically takes place based on collision or impulse. Nevertheless, breaking of steel or reinforced concrete structures using this approach is not appropriate making a simulation to look unrealistic. Theory for handling of plasticity has been presented already in Terzopoulos and Fleischer (1998). Müller et al. (2004) and Müller et al. (2005) present a method for modeling and animating of elastic and plastic objects in real-time using point based animation. This approach is not been widely used in simulation applications. On major issue is collision handling of deformable objects.

This study will introduce an approach to account plastic deformation in game applications. In the introduced method, the plastic deformation takes place if force or moment exceeds given limit, deformation absorbs energy and joint breaks if plastic capacity is exceeced. The approach is based on using joint motors to model plasticity. Erleben (2005, p. 90) suggests similar method for modelling friction in joints. Adjacent objects are connected by motors. Motor power production limits are estimated based on plastic section modulus. Joint breaking is accounted by summing plastic deformation and comparing it to predefined material based limit. Elastic part of deformation is modelled by employing spring description which is based on modification of existing constraint in Bullet Physics.

Approach presented in this work can be used in gaming industry to provide more realistic simulations without significant extra work. For gaming purposes presented method works best in scenarios where connected parts are relatively heavy. This allows normal integration timestep to be used without stability issues. This kind of metodology also opens large area of combining old structural analysis methods to modern simulation frameworks.

2 Adding plasticity to physics engine

In this section some central terms are explained and main differences between traditional structural analysis and physics engines are reviewed.

Velocity-based formulation is not typically used in structural or mechanical engineering. Erleben (2005, p. 45) provides reasoning and theoretical details why it is so popular in constraint-based rigid body simulation. Main reason is that collision handling can be done without additional procedures. In structural analysis solution method is selected carefully based on needed features. For most complex scenarios finite element method is used. In most cases static small displacement solution using displacement based boundary conditions is used. For large displacement static analysis analysis loading is applied in substeps and displacements are used to update element mesh. Further enhancements are material nonlinearity and dynamic analysis. Physics engine provides dynamic analysis with large displacements.

Material plasticity has typically taken into account in games by using suitable coefficient of restitution. This provides reasonable means to simulate loss of energy in collisions. Simulation of breaking of objects made of ductile material can be made more realistic by splitting rigid objects to multiple parts which are connected by energy absorbing joints. As stress-strain curve may not be familiar to all readers some basics are provided here. Typical stress strain curve of ductile steel is shown in 2.1. Stress-strain curve is not drawn to scale as elastic strain could not be seen as it is typically 0.001 to 0.005. Straing hardening is taken into account in this work mainly by assuming that plasticity in bending expands. Material that starts to yield first is hardened and yielding moves slightly. This can be seen e.g. by bending paperclip. It does not break at low angles but can take few full bends.

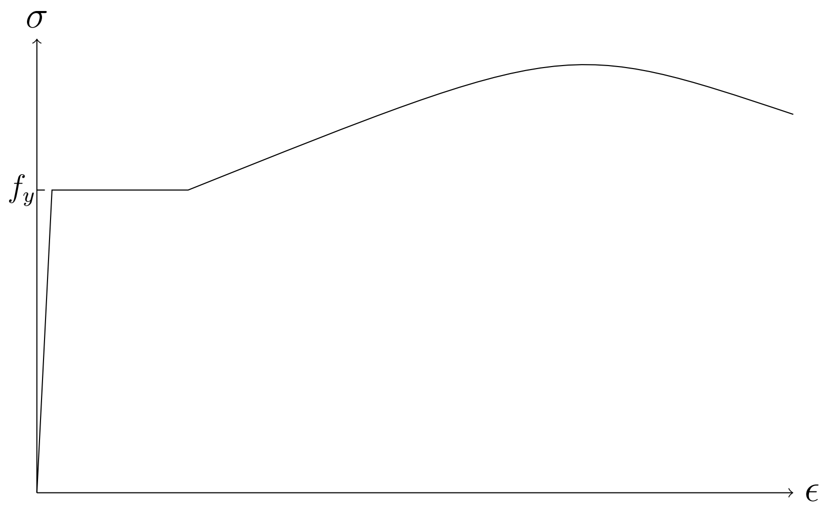

Figure 2.1: Stress-strain curve of ductile steel (not to scale).

Difference between elastic and plastic section modulus is shown in 2.2. If stress is below yield limit, stress and strain are linear within cross section. If cross section is fully plastic, stress is assumed to be at yield level over whole cross section and so plastic section modulus is higher than elastic section modulus. Elastic part is often ignored in this work as displacements due to to elastic deformation are very small.

Basic idea in this work can be tested with any framework having motors and hinge constraints. This can be done by setting target velocity of motor to zero and limiting maximum motor impulse to plastic moment multiplied by timestep.

Further enhancements were created and tested by forking Bullet Physics source code and adding new constraints Nikula (2014-2016). Constraint processing in Bullet Physics is based on ODE, Smith (2001-2007). Mathematical background and detailed examples are available by Smith (2002). Equations 2.1, 2.2 and 2.3 are created for each constraint.

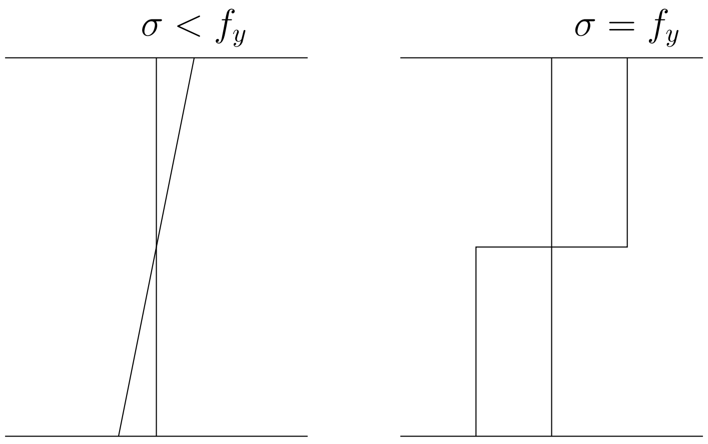

Figure 2.2: Stress distribution under elastic and plastic loads.

$$J_1 v_1 + \Omega_1 \omega_1 + J_2 v_2 + \Omega_2 \omega_2 = c + C\lambda \tag{2.1}$$

$$\lambda \geq l \tag{2.2}$$

$$\lambda \leq h \tag{2.3}$$

Main parameters and corresponding fields in Bullet Physics are described in table 2.1.

Parameter	**Description**	**btConstraintInfo2 pointer**
J_1, Ω_1 J_2, Ω_2	Jacobian	m_J1linearAxis, m_J1angularAxis m_J2linearAxis, m_J2angularAxis
v	linear velocity	
ω	angular velocity	
c	right side vector	m_constraintError
C	constraint force mixing	cfm
λ	constraint force	
l	low limit for constraint force	m_lowerLimit
h	high limit for constraint force	m_upperLimit

Table 2.1: Constraint parameters

References

Catto, E. (2007-2015). *Box2D A 2D Physics Engine for Games*. url: box2d.org.

Coumans, E. (2003-2016). *Bullet Physics Library*. url: bulletphysics.org.

Erleben, K. (2005). *Stable, Robust, and Versatile Multibody Dynamics Animation*. Ph.D. thesis. University of Copenhagen. image.diku.dk/kenny/download/erleben.05.thesis.pdf.

Müller, M., Heidelberger, B., Teschner, M., and Gross, M. (2005). Meshless deformations based on shape matching. In: *ACM Transactions on Graphics (TOG)*, vol. 24, 3, pp. 471–478. url: www.beosil.com/download/MeshlessDeformations_SIG05.pdf.

Müller, M., et al. (2004). Point based animation of elastic, plastic and melting objects. In: *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pp. 141–151. url: lgg.epfl.ch/publications/2004/mueller_2004_PBA.pdf.

Nikula, S. (2014-2016). *Plasticity extension to Bullet Physics Library*. url: https://github.com/simo-11/bullet3.

Smith, R. (2001-2007). *Open Dynamics Engine*. url: ode.org.

Smith, R. (2002). *How to make new joints in ODE*. url: ode.org/joints.pdf.

Terzopoulos, D. and Fleischer, K. (1998). Modeling Inelastic Deformation: Viscoelasticity, Plasticity, Fracture. *Computer Graphics*, pp. 269–278. doi:10.1145/54852.378522, url: web.cs.ucla.edu/~dt/papers/siggraph88/siggraph88.pdf.