Melting Objects

M. W. Jones
Department of Computer Science, University of Wales Swansea
Singleton Park, Swansea SA2 8PP, UK
m.w.jones@swan.ac.uk

ABSTRACT

This paper describes a technique for producing realistic animations of melting objects. The work presented here introduces a method that accurately models both thermal flow and the latent heat during the phase change. The mechanism for energy transfer to the model is via both boundary conditions and radiation. Emphasis is made on accurately modelling the solid object and the method is particularly suited to rigid solids with complex surface geometry. The underlying objects are constructed using *Volume Graphics* modelling techniques (specifically voxelization), and a numerical simulation computes the time-dependent heat flow throughout the object. A technique for computing the phase transition is given, and details for rendering the melting objects are provided. The melting is controlled by material parameters, such as specific heat capacity, thermal conductivity, latent heat and temperature. Examples are given of melting ice and a plastic man.

1 INTRODUCTION

One of the goals of computer graphics is to provide methods for creating visually realistic imagery of natural phenomenon. A great deal of research has been carried out in order to produce realistic animations of water [KWF+01, Sta99], fracturing [OH99], explosions [YOH00] and gaseous phenomena (e.g. smoke [Sta01]), but relatively little has been produced in the area of accurate animations of melting solids. Their aim has been to produce techniques that are simple to use, not too costly to compute, and yet provide suitable animation of the phenomenon being simulated for rendering. Numerical simulation techniques have been proposed for many of the above areas as computing hardware is now at the stage where the computation of the simulation is practical (for situations where a full solution / simulation is not necessary to obtain the desired effect).

The overall aim of this paper is to demonstrate methods for depicting the realistic animation of the melting of solids. In order to model the melting of a solid, the energy distribution throughout the object has to be simulated accurately. Heat energy must be provided to the object which raises its temperature until the melting temperature is reached. At this stage any additional energy is required to break chemical bonds in order to change the object's state from solid to liquid. This additional energy is known as the latent heat of melting, and during this transition phase, heat energy is absorbed without raising the temperature. The object will enter the liquid state once enough energy has been provided.

Journal of WSCG, Vol.11, No.1., ISSN 1213-6972
WSCG'2003, February 3-7, 2003, Plzen, Czech Republic.

2 PREVIOUS WORK

Blobby objects [WMW86] provided a means for rendering flexible looking objects, or objects with a liquid like behaviour. Previous research on the melting of objects for computer graphics used blobby modelling techniques to model the objects, and particle system techniques to move the primitives according to external and internal dynamic forces. Miller and Pearce [MP89] used such a technique for their globular dynamics system. An object (liquid or solid) is made of individual primitives which are governed by their *globular dynamics*. The system of particles are rendered by centring spherical fields (blobs) at each primitive and using a method similar to rendering the isosurface of the resulting potential field. Their method can produce liquid like behaviour and through careful management, turn solids into liquids, but there is no concept of heating involved. Internal forces are entirely due to attraction as a function of distance and external forces are due to Newtonian mechanics.

Desbrun and Gascuel [DG96] expanded this method by introducing smoothed particle hydrodynamics. Particles within the system are modelled using pressure and viscosity to govern their motion. This results in a physically based system of equations rather than ad-hoc systems. The surface of the object is rendered by isosurfacing the density function of the particles. This gives realistic liquid motion (and would be useful for transporting liquid away from the melting solid).

The above methods use particle motion rather than energy distribution, and therefore have not taken account of heat transfer, temperature changes and latent heat, all of which will be necessary to correctly model melting objects.

Terzopoulos et al [TPF91] introduce the use of the heat equation for deformable models. The object is discretized using a set of hexahedral cells with each vertex representing a particle within the object. A temperature is stored at each particle, and springs connecting the particles along the material coordinate axes conduct thermal energy. The spring-mass model lends itself particularly well to thermoelasticity which they model by allowing the spring stiffness to vary within the thermoelastic temperature range of the object, before setting it to zero at the melting point. Tonnesen [Ton91] also uses the heat equation with a system of particles.

All of the above particle based methods suffer from three drawbacks. Firstly particle to particle interaction has to be calculated for all particles (within the sphere of influence). This can be costly to compute as many particles will be required to model and render objects with high detail. Secondly, there is some difficulty in ensuring that the surface rendered from the potential fields is volume preserving (and that the system of implicit primitives models the surface accurately). For example, Desbrun and Gascuel [DG96] indicate that a variation of up to 10% of the volume can occur (although it is far lower for most of the time). Finally only Terzopoulos et al [TPF91] and Tonnesen [Ton91] accurately model thermal energy in their models, but they omit the problem of modelling latent heat and energy absorbtion through radiation. They also create models that combine a macroscopic view of thermal energy with a microscopic view of particle interaction throughout the object.

More recently Fujishiro and Aoki [FA01] approximated thawing ice by using erosion and dilation operations on binary voxelised data (each of their voxels is two-state – inside or outside). Their method provides a certain amount of realism (irregular melting) by using different sized structuring elements (in their case different radii spheres) in areas where the object surface can see more of the heat source. Their model is not physically based as there is no calculation of heat energy. The visual appearance of their method could be improved by using a grey-level voxelization as is used here. For later comparison with the proposed method, their method took over 2 hours on a 195Mhz SGI O2 for a 128^3 data set.

This paper introduces a model of computing for animations of melting objects. Like recent approaches for realistic animation of phenomena [KWF$^+$01, Sta99, OH99, YOH00, Sta01] this method involves numerical simulation (in this case the *Heat Equation*) for the calculation of the time-dependent energy distribution throughout the object. The method presented here introduces a model that accurately models both thermal flow and latent heat which is an important part of the physical process of melting. Emphasis here has been on accurately modelling the underlying solid object, and the method is particularly useful for rigid solids with complex surfaces. By using the phase mixture formulation and voxelization, the work here produces volume preserving models and rendering methods. The result is a fast (volume preserving) physically based animation that allows effective control of the melting process by using well defined parameters derived from the physical properties of the material undergoing the phase transition. In addition to utilising the heat equation, a practical method for dealing with latent heat during the phase transition is presented, and a means by which objects can be converted to easily simulated models via voxelization is given. This also leads to a natural rendering technique for the results of the simulation using volume rendering techniques. In particular, for the example of ice melting, employing volume rendering techniques has allowed an accurate refraction model to be used.

An additional advantage of the voxel approach to melting objects is that it provides a structure that allows the fast computation of heating due to radiative transfer which is also demonstrated in this paper.

3 HEAT TRANSFER

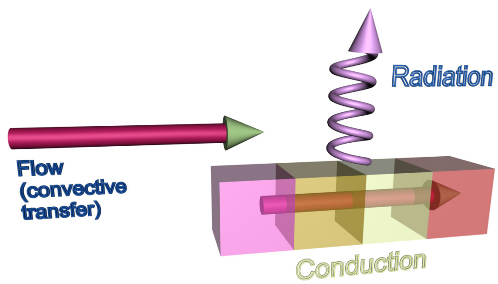


Figure 1: Heat transfer.

To establish the domain of the problem solved, we need to understand the various ways in which heat transfer can take place in an object. Heat transfer occurs [Hol96] whenever there is a temperature gradient present. Energy is transfered from the high temperature region to the lower temperature region, and can take place using one or more of *conduction*, *convection* and *radiation* (see Figure 1). If the (hot) box of Figure 1 is fully insulated, only conduction takes place. If we remove that insulation, the box now conducts heat to the surrounding medium (for example (cool) air). The temperature gradient in the surrounding air will govern the heat transfer between media. Where an air flow occurs across the object (as in Figure 1), the air will carry the heat away at a faster rate, and therefore cool the object faster. Although conduction takes place at the surface interface, the temperature gradient in the flowing air must be calculated by relating temperature to the flow field which gives a numerical model for convection. Thermal radiation is that electromagnetic radiation which is due to temperature difference between two surfaces. Energy is radiated from blackbodies at a rate proportional to its surface area and the forth power of the absolute temperature of the body. As object surfaces consist of a multitude of substances an emissivity factor appropriate for that surface attenuates the amount. The net exchange between bodies also depends upon the area that can be viewed from each surface of the other. This problem and its solution is well known in computer graphics as radiosity (and its various forms of solution) is based upon radiation heat transfer.

In order to model a melting object accurately, heat conduction throughout the object must be modelled as the temperature raises. Where the temperature rises to the melting point, additional energy will be absorbed without raising the temperature as latent heat. Where the object has changed to a liquid, convection will now take place and the temperature gradient across the liquid must be calculated. Heat will also be conducted at the liquid to air interface, and convection will also take place in the air. The liquid will also run across the melting surface and collect at the lowest point. Radiation will also need to be calculated.

From this description it can be seen that the problem of accurately modelling melting objects will result in an extremely complex numerical simulation. For the puposes of this work a few simplifications have been made. Firstly some objects do not radiate – for example snow and ice have very low emissivity, and so object cooling via thermal radiation has been ignored, although object heating via radiation has been modelled (Section 9). The model removes the requirement to model convection by assuming any energy transferred to (or from) the air is replaced by energy from the surrounding air to give the air at the surface a constant temperature. Assumptions about the melt are given in Section 6. The model for computing the conductance within the object is developed in the next section.

4 THE HEAT EQUATION

The numerical model for producing an animation of a melting object is developed in Sections 4 to 7. An introduction to the general area of numerical methods is given by Gerald and Wheatley [GW99]. Holman [Hol96] provides a text on the area of heat transfer and provides solutions for modelling the various methods of heat transfer as mentioned above.

Considering Figure 2, the rate at which energy flows into a three-dimensional cell and the rate at which energy is internally generated is equal to the rate at which energy flows out of the cell and the rate at which the internal temperature increases. The solution requires to find the time dependent temperature, u, of the cell, and the energy flow through the body.

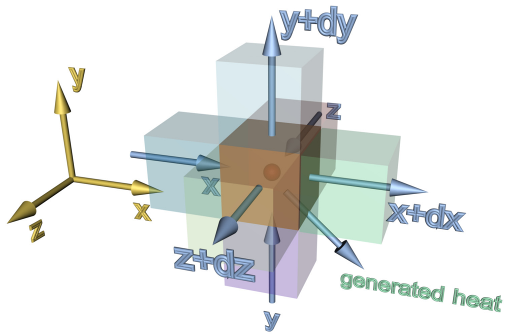


Figure 2: Modelling Temperature.

If we assume for the moment that the object consists of one row of cells parallel to the x axis, which are fully insulated so only conduction takes place (e.g. Figure 1), then the rate at which energy flows into an element at point x is $-kdydz\left(\frac{du}{dx}\right)$, where k is the constant of thermal conductivity for the material, $dydz$ gives the cross sectional area and the minus sign is inserted so that the second law of thermodynamics is satisfied (($\frac{du}{dx}$) is the temperature profile along the object and will be negative if a higher temperature is at the left, but in that case the flow is in the positive x direction, hence the minus sign).

The rate at which energy flows from the element is similar, but for the point $x + dx$, giving:
$-kdydz\left[\frac{du}{dx} + \frac{d}{dx}\left(\frac{du}{dx}\right)dx\right]$
i.e. the gradient at x plus the change in the gradient between x and $x + dx$.

The rate of change in internal temperature is dependent on the volume, V, of the cell ($dx \times dy \times dz$, units m^3), the density of the material in the cell (ρ, units kg/m^3) and the amount of energy required to heat 1kg of the material by 1 degree (specific heat capacity, c, units J/(kg$^{\text{O}}$C)), giving $c\rho V\frac{du}{dt}$.

By assuming no heat is generated by the cell we obtain:

$$-kdydz\left(\frac{\partial u}{\partial x}\right)=$$
$$-kdydz\left[\frac{\partial u}{\partial x}+\frac{\partial}{\partial x}\left(\frac{\partial u}{\partial x}\right)dx\right]+c\rho V\frac{\partial u}{\partial t} \quad (1)$$

which reduces to:

$$k\left(\frac{\partial^2 u}{\partial x^2}\right)=c\rho\frac{\partial u}{\partial t} \quad (2)$$

and leads to the standard diffusion equation for three-dimensions (the *Heat Equation*):

$$k\nabla^2 u=c\rho\frac{\partial u}{\partial t} \quad (3)$$

where k is the thermal conductivity of the material J/sec/($^{\mathrm{O}}$Cm), c is the specific heat capacity J/(kg$^{\mathrm{O}}$C), ρ is the density kg/m^3 and $\nabla^2 u$ is the Laplacian (which when extended into three-dimensions becomes):

$$\nabla^2 u=\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}+\frac{\partial^2 u}{\partial z^2}\right) \quad (4)$$

The above *Heat Equation* describes analytically how heat flows in an object with the assumption that the object is both homogenous (with constant thermal properties) and does not generate heat from within (although it would be simple to introduce both into the model).

5 NUMERICAL SOLUTION

Using the explicit method in order to solve the above equation for a specific situation requires that the spatial domain be discretised into a regular lattice. The partial derivatives are replaced by finite differences, and an explicit equation can be derived which can be used to compute the solution based upon the initial conditions.

The time (first) derivative of Equation 3 is replaced with the forward difference approximation:

$$\frac{\partial u}{\partial t}=\frac{u_o^{j+1}-u_o^j}{\Delta t} \quad (5)$$

where o is the (three-dimensional) position and j is the time step. Each time step is Δt after the previous.

The spatial (second) derivatives (Equation 4) are replaced with central difference approximations, giving:

$$\nabla^2 u=\frac{\left(u_{x-1}^j-2u_o^j+u_{x+1}^j\right)}{(\Delta x^2)}+\frac{\left(u_{y-1}^j-2u_o^j+u_{y+1}^j\right)}{(\Delta y^2)}+\frac{\left(u_{z-1}^j-2u_o^j+u_{z+1}^j\right)}{(\Delta z^2)} \quad (6)$$

where the subscript indicates the specific 6-neighbour of the current voxel, u_o.

By working with cubical cells ($h=\Delta x=\Delta y=\Delta z$), Equation 6 reduces to:

$$\nabla^2 u=\frac{u_{x-1}^j+u_{x+1}^j+u_{y-1}^j+u_{y+1}^j+u_{z-1}^j+u_{z+1}^j-6u_0^j}{(h^2)} \quad (7)$$

Substituting Equations 5 and 7 into Equation 3 and rearranging gives:

$$u_0^{j+1}= r\times(u_{x-1}^j+u_{x+1}^j+u_{y-1}^j+u_{y+1}^j+u_{z-1}^j+u_{z+1}^j)+(1-6r)*u_o^j \quad (8)$$

where:

$$r=\frac{k\Delta t}{c\rho h^2} \quad (9)$$

Equation 8 gives the explicit method for computing the simulation in three-dimensions. The temperature of the cell u_o at time $j+1$ can be computed from u_o and its neighbouring cells at time j. Given boundary conditions Equation 8 will calculate the time varying solution of temperature throughout the solid. For **stability** a value of $r\leq\frac{1}{6}$ is required in Equation 8. Since values for specific heat (c), density (ρ), grid spacing (h) and thermal conductivity (k) are known, the time step (Δt) can be chosen to enable $r\leq\frac{1}{6}$.

An implementation using the above equation will model the heat flow throughout an object, but will not take into account the effect of the phase transition the object will undergo at its melting point or the addition of energy through radiation, and therefore additional measures must be taken to ensure the accurate modelling of melting.

6 PHASE TRANSITION

The melting of ice into water is an example of a phase transition. As energy is supplied to a kilogram of ice, the temperature will raise by approximately 1 degree for every 2kJ supplied up to the phase transition temperature. At 0$^{\mathrm{O}}$C energy is required to change state from ice to water. At this transitional stage, the energy supplied to the ice will be used to break the chemical bonds, and is known as the latent heat of melting (L).

Latent heat is the energy that changes the phase of a substance without changing its temperature. To transform 1kg of ice into water at 0$^{\mathrm{O}}$C requires 333kJ of heat.

In general, a simulation involving a phase transition, must accurately simulate the absorbtion or release of latent energy on the moving boundary that separates the solid and liquid phases of the object. This involves the accurate tracking of the boundary, and the modelling of the energy at the boundary.

The finite difference solution derived in Section 4 for a fixed grid can be adapted to model the energy where a phase transition is involved by using the *enthalpy* formulation [LMTS96]. Equation 3 is reformulated as:

$$k\nabla^2 u = c\rho \frac{\partial H}{\partial t} \tag{10}$$

where H is the enthalpy function, otherwise known as the total heat content, and is defined in the situation where there is an isothermal phase change, for a temperature u, as:

$$\begin{aligned} &H(u) = \int_{u_r}^{u} \rho c_s(u) du \\ &(u < u_f) \\ &H(u) = \int_{u_r}^{u_f} \rho c_s(u) du + \rho L + \int_{u_f}^{u} \rho c_l(u) du \\ &(u \geq u_l) \end{aligned} \tag{11}$$

where L is the latent heat, c_s and c_l are the specific heat capacities in the solid and liquid phases respectively, and u_f and u_r are the freezing point and a reference temperature below u_f respectively.
The heat content, H, at a temperature, u, of an object of density ρ, takes into account the total energy required to heat the object including the latent energy, L, where the object has undergone a phase change.
This can be implemented using a primary temperature data structure and a secondary enthalpy corrective (latent heat) data structure. The numerical simulation works on the primary data structure, but at the end of each time step, any increase in temperature over the melting point of the solid is converted into energy and added to the secondary latent heat data structure (or subtracted during refreezing). Only when the energy of that cell has reached the required latent heat value, is the cell considered to be fully melted. This has the result that whilst the latent heat of the cell is 0 the cell is completely solid and once above L (or $L\rho h^3$, depending upon the energy conversion) it is liquid. In between a phase mixture exists which needs to be taken into account.

7 MODELLING AND PHASE MIXTURE

During the phase transition the solid and liquid parts may coexist at equal temperature. Ice and water coexisting at a temperature of 0^{o}C and thermally insulated will result in a situation where there is no heat flow between the constituent parts. If more energy is provided directly to the ice, further melting will take place until the ice is fully melted. In reality a volume cell containing both ice and water which is heated will result in the raising of the temperature of the water, aswell as providing energy to the ice for further melting. Heat flow will also take place between the ice and water. A fully accurate simulation will have to track the boundary that exists between the two phases of material (which also have different thermo-physical properties) and determine the heat flow between them (even including convection in the liquid part) and the transfer of heat out of the cell. A simplifying assumption taken here is that any energy flowing into a cell is directed at the solid mass only and does not heat the liquid. This results in the correct amount of energy being stored as latent heat for melting and still results in a model that converges. Once fully melted the liquid cell is removed and replaced by air. This part of the numerical simulation could be improved as mentioned in Section 10. The phase mixture in each cell is tracked during the simulation using the solid mass fraction, ϕ which is expressed as:

$$\phi = \begin{cases} 1 & \text{solid region} \\ 0 < \phi < 1 & \text{mix} \\ 0 & \text{liquid region} \end{cases} \tag{12}$$

This leads to a natural way of modelling objects by expressing their volume occupancy of each volume element. Complex surfaces can be evaluated to produce a volume which corresponds to the solid mass fraction formulation. Each cubic voxel is tested against the volume object, and ϕ is set to zero if it is outside (also the temperature is initialised to the initial air temperature to be used). Each interior voxel is set to one and its temperature is set to the object temperature. Unlike Fujishiro and Aoki [FA01] the method here calculates a grey-level function rather than binary voxel data by using the volume occupancy of transverse voxels (voxels through which the surface passes). There are a number of ways for which the volume occupancy function can be calculated:

- Already **voxelised data**:
  - **Either**: Rescale the exisiting grey-level values between zero and one. Set all interior voxels to one (8 vertices all have values greater than threshold), and all exterior voxels to zero. The remaining grey-level values can be used to approximate volume occupancy.
  - **Or**: Use the tiling tetrahedra [PT90] algorithm (similar to marching cubes but does not have ambiguity problems) on the data using the surface threshold. For each voxel geometrically calculate the volume inside the surface by examining each tetrahedron in turn.
- **Triangular mesh** data:
  - **Either**: Voxelise the mesh using, for example, the distance field method [Jon96].

Proceed using one of the two above methods. Although the perception may be that this is a lengthy and complicated process, by using an octree, careful ordering of distance calculations, and only calculating the voxel values in the immediate vicinity of the surface, it is possible to voxelise large meshes on high resolution grids in seconds – e.g. 2000 triangles on a 60^3 grid takes about 1 second on an Athlon 1.4GHz.

  – **Or**: Approximate via a sampling method (can be slow for many samples) or calculate geometrically.

- **CSG**: The occupancy is known for each voxel, so computation is straightforward from the object definition.

- **Implicit Functions**: Voxelize or sample the implicit function, or calculate volume directly from the function if available.

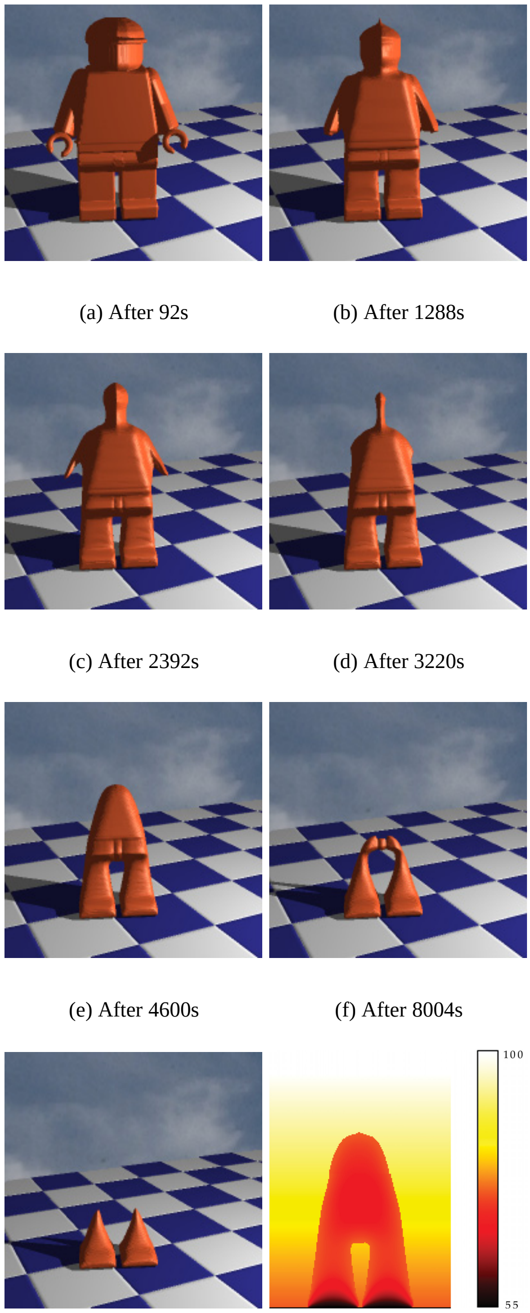
These methods lead to the solid mass fraction function which is used in the numerical simulation. To render the melting surface the solid mass fraction field ϕ can be used with volume rendering or direct surface rendering. Images here are direct surface rendered from the field ϕ with a threshold of 0.5.

8 RENDERING

Table 1 shows thermal data for a number of objects collected from numerous sources. Thermal properties vary according to temperature and pressure of the object being measured. For example the thermal conductivity of ice has been measured as 2.4W/m$^{\text{o}}$C at -20$^{\text{o}}$C and 2.2W/m$^{\text{o}}$C at 0$^{\text{o}}$C. The figures should be treated as approximate as they have been reported for different temperatures and pressures.

An animation of a melting block of ice was computed using the solid mass fraction to represent the phase mixture, and using the model developed for thermal flow and latent heat storage. The rendering was been carried out by vlib [WC01] using the solid mass fraction as the volume function to be rendered. The use of volume rendering has allowed an accurate refraction model based upon the refractive index and size of the solid object. The animation accompanies this submission.

Figure 3 shows an animation of a more complex surface melting. In this case a phase mixture volumetric dataset was created from Sramek's voxelised legoman [SK98], and used in the simulation. Figure 3(h) shows a slice through the volume at 4324 seconds, and demonstrates the heat distribution at that point in time. Note that Figures 3 and 4 both have the boundary condition that the air temperature is related to the distance



(a) After 92s (b) After 1288s

(c) After 2392s (d) After 3220s

(e) After 4600s (f) After 8004s

(g) After 9200s (h) Temperature of middle slice at 4324s

Figure 3: 24cm high wax legoman. $k = 0.18$ W/(m$^{\text{o}}$C), $\rho = 1000$kgm^{-3}, $c = 2000$ J/kg$^{\text{o}}$C, $h = 0.001$m, $L = 105$kJ, boundary conditions – see Section 8.

Substance	Melting Point (^{O}C)	Density (kg/m^3)	Specific Heat Capacity (kJ/kg)	Thermal Conductivity W/mK	Latent Heat of Fusion kJ/kg
Ice	0	917	2.06	2.2	333
Water	N/A	1000	4.186	0.57	N/A
Wax	60	1000	2	0.18	105
Glass	550	2500	0.84	0.8	N/A
Silver	962	10500	0.235	406	111
Aluminium	659	2700	0.9	210	399
Gold	1063	19300	0.128	293	64

Table 1: Approximate data for various substances. (Values vary according to temperature)

from the floor, starting at 65^{O}C at the floor and 100^{O}C at the top. This was to reflect the fact that the air would be cooler in the vicinity of the floor which is at 55^{O}C. It is apparent from Figure 3 that the model using just heat transfer through conduction produces animations that are too regular and unnatural looking. One method for creating a more realistic ”irregular” melting is by introducing hot radiating objects into the scene.

9 RADIATIVE TRANSFER

As already stated, one of the simplifying assumptions of the model is that the object does not cool by radiating heat. The model presented here is extended to include object heating by radiative transfer to the object.

Heating due to radiation is governed by the visibility of the point being considered from the heat source. Where the heat source is visible from the object it will provide heat that can be absorbed, reflected and transmitted. In this model we regard all incident heat as being absorbed by the object. Reflected heat could be modelled simply like light as it is also electromagnetic radiation and follows the same principal of specular and diffuse reflection. To further simplify calculation the heat source is assigned an energy value E_h representing the energy that is provided to one voxel at 1 metre. For each visible voxel v, the energy due to radiation, E_r, from the heat source located at h is:

$$E_r = \frac{E_h cos\theta}{|\vec{hv}|^2} \tag{13}$$

where

$$cos\theta = \frac{\vec{hv}.\vec{n}}{|\vec{hv}|} \tag{14}$$

and $\vec{n}$ is the normalised gradient calculated using central differences from ϕ.

The voxel visibility is calculated by considering all voxels along the path from the voxel to the radiant using the fast voxel traversal algortihm of Amanatides and Woo [AW87]. The additional calculation at most doubles the amount of computation required.

Figure 4 shows the same dataset but this time thermal radiation has been included. The legoman data set is $171 \times 98 \times 242 = 4,055,436$ elements. Each time-step is computed in around 0.8 seconds on an Athlon 1.4GHz. A 400 frame animation (8000 time steps) took just over 1hr30m to compute. This compares very favourably with Fujishiro and Aoki's [FA01] method which took 2 hours on a 195MHz SGI O2 to compute 30 iterations of a $128 \times 128 \times 128 = 2,097,152$ voxel dataset. Without radiative transfer, each time-step takes around 0.4 seconds to compute.

10 CONCLUSION

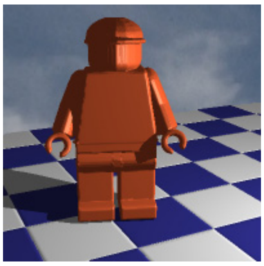
This paper has presented a method for animating melting solids. The numerical simulation has modelled heat flow, latent heat and radiative transfer accurately with the melting process governed by well defined material properties. Using the solid mass fraction formulation for the melt has provided a useful tool for both rendering and modelling complex surfaces accurately. Suggestions for converting (voxelizing) various graphical models have been made. Future work could increase the realism by allowing the pooling of the liquid on the object and ground. This could be achieved by using smoothed particle hydrodynamics such as the method by Desbrun and Gascuel [DG96]. The method as presented has been just concerned with introducing melting, for increased realism Newtonian mechanics should be incorporated so that parts that break off could be tracked correctly.

References

[AW87] J. Amanatides and A. Woo. A fast voxel traversal algorithm for ray tracing. *Proc. of Eurographics '87, Amsterdam, The Netherlands*, pages 3–9, August 1987.

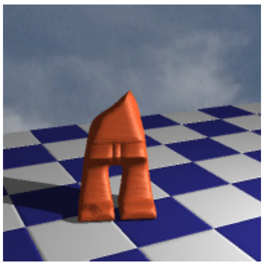
[DG96] M. Desbrun and M-P. Gascuel. Smothed particles: A new paradigm for animating highly deformable bodies. In *Computer Animation and Simulation*, pages 61–76, August 1996.

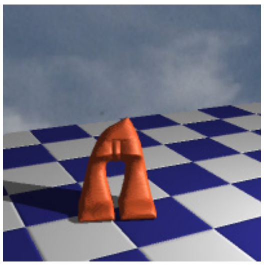
[FA01] I. Fujishiro and E. Aoki. *Volume Graphics 2001*, pages 69–80. Springer-Verlag, Wien, 2001.



(a) After 92s (b) After 1012s

(c) After 1932s (d) After 2576s





(e) After 3680s (f) After 6164s

(g) After 7636s (h) After 17020s

Figure 4: Same paremeters as Figure 3, but with radiation from a heat source at the top left.

[GW99] C. Gerald and P. Wheatley. *Applied Numerical Analysis*. Addison-Wesley, 6th edition, 1999.

[Hol96] J. P. Holman. *Heat Transfer*. McGraw-Hill, 8th edition, 1996.

[Jon96] M. W. Jones. The production of volume data from triangular meshes using voxelisation. *Computer Graphics Forum*, 15(5):311–318, December 1996.

[KWF+01] A. Kunimatsu, Y. Watanabe, H. Fujii, T. Saito, K. Hiwada, T. Takahashi, and H. Ueki. Fast simulation and rendering techniques for fluid objects. *Computer Graphics Forum*, 20(3):57–66, 2001.

[LMTS96] R. W. Lewis, K. Morgan, H. R. Thomas, and K. N. Seetharamu. *The Finite Element Method in Heat Transfer Analysis*. Wiley, 1996.

[MP89] Gavin Miller and Andrew Pearce. Globular dynamics: A connected particle system for animating viscous fluids. *Computers and Graphics*, 13(3):305–309, 1989.

[OH99] J. O'Brien and J. Hodgins. Graphical modeling and animation of brittle fracture. In *Proceedings of SIGGRAPH 1999*, pages 137–146. ACM SIGGRAPH, August 1999.

[PT90] B. A. Payne and A. W. Toga. Surface mapping brain function on 3D models. *IEEE Computer Graphics and Applications*, 10(5):33–41, September 1990.

[SK98] M. Sramek and A. E. Kaufman. Object voxelization by filtering. In *Volume Visualization Symposium*, pages 111–118. IEEE Computer Society, 1998.

[Sta99] Jos Stam. Stable fluids. In *Proceedings of SIGGRAPH 1999*, pages 121–128, August 1999.

[Sta01] Jos Stam. Interacting with smoke and fire in real time. *Communications of the ACM*, 43(7):76–83, 2001.

[Ton91] D. Tonnesen. Modeling liquids and solids using thermal particles. In *Graphics Interface'91*, pages 255–262. Morgan Kaufmann, June 1991.

[TPF91] D. Terzopoulos, J. Platt, and K. Fleischer. Heating and melting deformable models. *Visualization and Computer Animation*, 2(2):68–73, 1991.

[WC01] A. S. Winter and M. Chen. vlib: A volume graphics API. In *Volume Graphics 2001*, pages 133–148. Springer, 2001.

[WMW86] G. Wyvill, C. McPheeters, and B. Wyvill. Data structures for soft objects. *The Visual Computer*, 2:227–234, 1986.

[YOH00] G. Yngve, J. O'Brien, and J. Hodgins. Animating explosions. In *Proceedings of SIGGRAPH 2000*, pages 29–36. ACM SIGGRAPH, July 2000.