

Bone Glow: An Improved Method for the Assignment of Weights for Mesh Deformation

Rich Wareham and Joan Lasenby

Department of Engineering, University of Cambridge,
Trumpington Street, Cambridge, CB2 1PZ, UK
{rjw57,jl221}@cam.ac.uk

Abstract. Many real-time algorithms for mesh deformation driven by animation of an underlying skeleton make use of a set of per-bone weights associated with each vertex. There are few unguided algorithms for the assignment of these weights with a recent proposed solution being *bone heat* [1]. In this paper, we briefly discuss bone heat and provide examples where it performs poorly. We then develop a refinement of bone heat, termed bone glow, which, in our validation, performed as well as bone heat in simple cases while not suffering from bone heat's observed weaknesses.

Keywords: mesh deformation, skeletal animation, bone heat, bone glow, vertex weighting, skinning, realtime, animation.

1 Introduction

The requirement to animate, in real time, on-screen characters for the computer video game industries places a number of restrictions on the mesh deformation scheme used. It must be quick to run, both in its computational requirements and its suitability for implementing on modern graphics accelerators, and quick to use. Most existing methods used in this area rely on associating a meshed surface description of a character with an armatured skeleton. This skeleton is then animated and the mesh is deformed accordingly.

In this paper we concentrate on these *skinning* schemes and, in particular, those which represent the association between skeleton and mesh as a series of weights. Each vertex has a per-bone weight giving a measure, in the range $[0, 1]$, of the influence of the particular bone on that vertex. These weights are normalised so that the sum of all per-bone weights for a particular vertex is 1.

Once assigned, these weights can be used by a number of deformation algorithms. Systems in use in the game industry include Linear Blend or Skeleton Subspace deformation [2], rotor-based deformation [3] and spherical and dual-quaternion interpolation schemes [4,5].

The actual assignment of weights, however, has traditionally been a somewhat manual affair. Existing 3D modelling applications expose the task to artists in the form of 'weight painting' operations whereby the artists 'paint' the weights for one bone onto the mesh with a virtual airbrush.

Automated weight assignment techniques fall into two camps; those that require the artist to specify some parameters, such as bone size or a region of influence for a joint, and those that proceed using only the mesh surface and bone positions. The former methods are best suited to the animation and game industries which have artistic resource to use on a problem. In some cases the latter approach must be taken. A prime example of a use-case in which minimal artist input is required is the emerging field of ‘user generated content’ in which relatively unsophisticated users of an online service author a three-dimensional avatar representation of themselves which they expect to be animated. A method to animate such a character is given only the avatar model and user-authored skeleton and cannot rely on the user to provide sophisticated parameters about the bones.

A recent technique proposed as a suitable algorithm for this unguided skeleton-mesh association is *bone heat* [1]. This method aims to model weight assignment as a heat diffusion system on the surface of the mesh; for each bone, the vertices which have that bone as their nearest visible bone are initialised to have a surface temperature inversely proportional to the square of their closest distance to the bone. A diffusion equation is then solved to ‘smear’ this nearest neighbour weight assignment. This approach has a number of disadvantages, primarily that the technique aims to ‘blur’ an undesirable solution in the hope that it will become more acceptable.

Figure 1 illustrates the effect of incorrect bone heat weight assignment on skinning. Here, the relatively sophisticated real time dual quaternion skinning algorithm built into the Blender 3D animation software [6] was applied to weights calculated from Blender’s internal implementation of bone heat. The bone heat algorithm incorrectly assigned strong weights between the arm bone and chest area leading to unacceptable distortion.

In section 2, we construct a technique which performs at least as well as bone heat in the simple cases and gives a more natural weight assignment where bone heat fails. Our technique uses a different conceptual model to that of bone heat yet generates similar results.

2 Weight Assignment

The method of bone heat solves a diffusion equation over the surface of the mesh. Specifically, for bone i , the diffusion equation solved is

$$-\Delta \mathbf{w}^i + \mathbf{H}(\mathbf{p}^i - \mathbf{w}^i) = 0 \quad (1)$$

where Δ is the discrete surface Laplacian [7], \mathbf{H} is a diagonal matrix with element H_{jj} being the heat contribution of the nearest bone to vertex j , and \mathbf{p}^i is a vector where element $p_j^i = 1$ if bone i is the closest bone to vertex j and $p_j^i = 0$ otherwise. The above equation is solved for \mathbf{w}^i which is a vector in which element j gives the final weight of bone i for vertex j . In [1], \mathbf{H} was constructed in the following manner; if the shortest line segment joining vertex j to the nearest bone was entirely within the mesh volume, the element $H_{jj} = 1/d(j)^2$ with $d(j)$

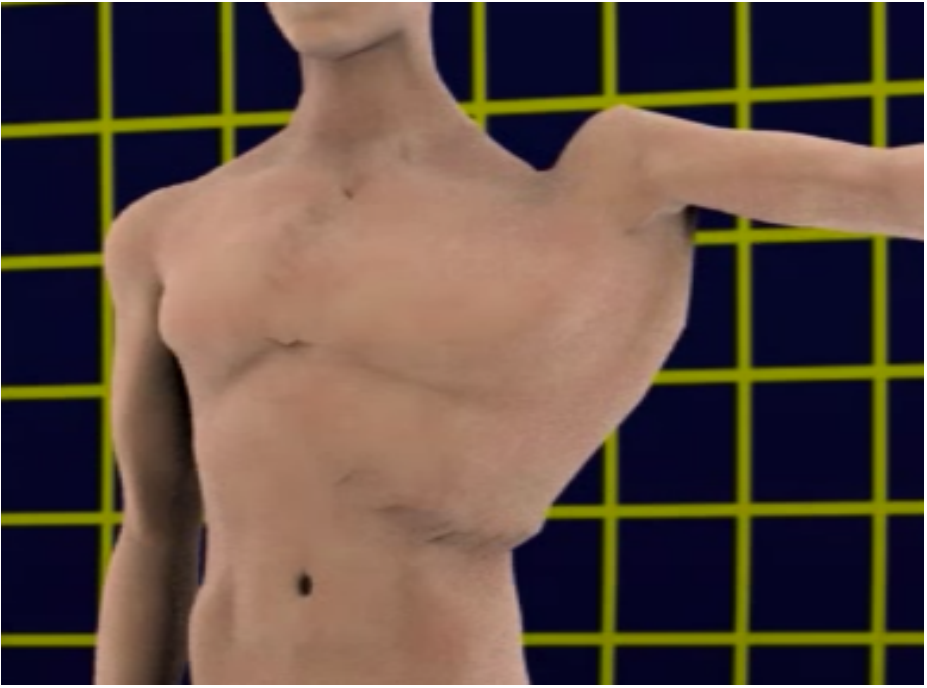


Fig. 1. An example of bone heat failing to correctly assign weights to an arm bone causing obvious distortion in the chest even when a more sophisticated skinning scheme (dual quaternions) is used

being the shortest distance from vertex j to the closest bone. If the shortest line segment was not within the mesh volume, $H_{jj} = 0$.

The bone heat algorithm in effect calculated two sets of weights; an initial set, \mathbf{p}^i , is calculated based upon a nearest visible bone solution. This initial set is then refined, or ‘blurred’ via the diffusion equation above. Such a scheme has merits in terms of the ‘natural’ results the blurring gives but suffers from artifacts arising from this implicit initial solution step. An item of note, shown in fig. 2, is the over-weighting of the arm bone in the chest area. The tip of the arm bone is visible to many of the vertices which make up much of the side of the chest. It is also closer than the upper back bone which, arguably, should be the bone exerting the greatest influence. Using just this nearest-bone initial solution leads to highly displeasing results which the diffusion step must attempt to mitigate.

The nature of bone heat’s initial solution leads to its poor performance in certain cases, especially in the common arm/chest joint case. We aim to modify bone heat so that this initial solution is closer to an acceptable solution thereby relying on the diffusion step only as a post processing operation to increase the smoothness of the final result.

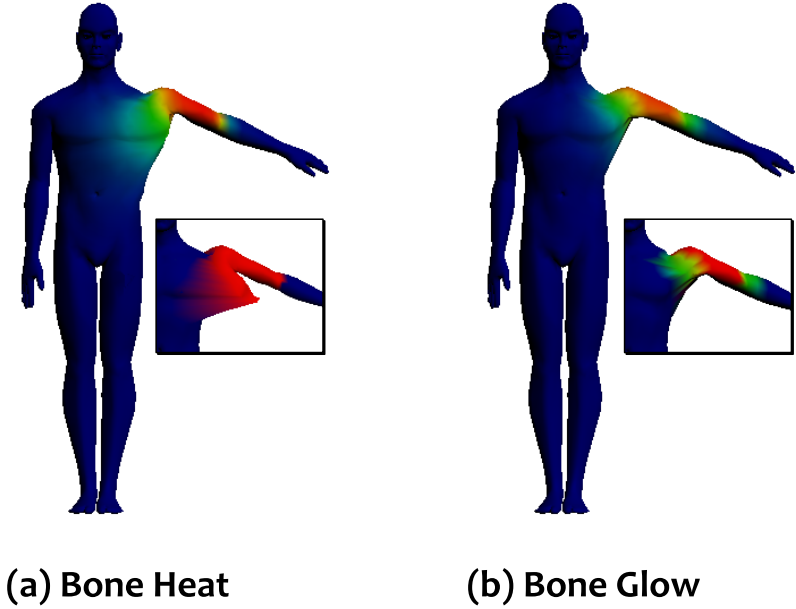


Fig. 2. A comparison of a simple linear blend skinning scheme applied using the weights from a) bone heat and b) bone glow. The weight assigned to each vertex has been indicated using a gradation from blue to red to indicate the range $[0, 1]$. Inset: the solutions without applying a diffusion step.

Our approach will be to define an initial weighting which is not binary for each bone/vertex pair but is instead smooth over the boundaries between the regions of influence of each bone. We then feed this result into the diffusion solver in place of the vector \mathbf{p}^i in eqn. 1.

We calculate a per-vertex influence for each bone by computing the amount of incident light which each vertex receives from the bone under consideration. For this we use an adaptation of the classic lighting integral. To calculate the incident illumination on vertex j from bone i we integrate along the length of the bone calculating the contribution of each point along the bone. Specifically, the illumination, L_j^i , received by vertex j from bone i is given by

$$L_j^i = l^i \int_0^1 l^i V_j^i(\lambda) R_j^i(\lambda) T_j^i(\lambda) d\lambda \quad (2)$$

where l^i is the length of bone i and λ is a normalised co-ordinate along the bone with $\lambda = 0$ being the root of the bone and $\lambda = 1$ being the tip. Each of the terms within the integral represents a different factor in the lighting equation. The $V_j^i(\lambda)$ term is a simple binary function representing the visibility of the bone; if the point at λ on bone i is visible from vertex j then $V_j^i(\lambda) = 1$ and $V_j^i(\lambda) = 0$ otherwise. The $R_j^i(\lambda)$ term gives the proportion of illumination received by vertex

j from the point λ along bone i assuming that the bone is visible. The $T_j^i(\lambda)$ term gives the proportion of illumination transmitted from the point λ along bone i in the direction of vertex j .

There exist well known models for the $R_j^i(\lambda)$ and $T_j^i(\lambda)$ terms which we shall cover below, but first we shall explain the notation used. The location of vertex j , relative to a fixed world frame, is given by the vector \mathbf{v}_j . Similarly the normal to the surface at vertex j is parallel to the unit vector $\hat{\mathbf{n}}_j$. The point on the bone i at normalised co-ordinate λ is represented by the vector \mathbf{b}_λ^i and the axis of bone i is parallel to the unit vector $\hat{\mathbf{a}}_i$. Finally, we define $\mathbf{d}_j^i(\lambda) \equiv \mathbf{v}_j - \mathbf{b}_\lambda^i$. Using the notation $\|\cdot\|$ to mean ‘length of’, we also define the unit vector $\hat{\mathbf{d}}_j^i(\lambda) \equiv \|\mathbf{d}_j^i(\lambda)\|^{-1} \mathbf{d}_j^i(\lambda)$.

To model the received illumination we use a Lambertian falloff; the proportion of incident illumination received is the cosine of the incident angle. We can compute this via the dot product between the incident illumination direction and the vertex normal:

$$R_j^i(\lambda) = \frac{\max(\hat{\mathbf{d}}_\lambda^i \cdot \hat{\mathbf{n}}_j, 0)}{\|\mathbf{d}_j^i(\lambda)\|^2} \quad (3)$$

where we have suppressed, via the $\max()$ function, all light which is received along a path which does not come from within the mesh. In addition we have imposed a quadratic falloff of incident illumination consistent with a local source of light.

The proportion of illumination transmitted from the bone is modelled as falling off as the sine of transmission angle. That is to say that we assume no light is transmitted parallel to the bone and the maximum transmission is perpendicular to the bone axis. We choose the sine function since it may easily be calculated via a cross product:

$$T_j^i(\lambda) = \|\hat{\mathbf{d}}_\lambda^i \times \hat{\mathbf{a}}^i\| \quad (4)$$

Substituting these relations into (2) gives the final illumination integral:

$$L_j^i = l^i \int_0^1 V_j^i(\lambda) \frac{\max(\hat{\mathbf{d}}_\lambda^i \cdot \hat{\mathbf{n}}_j, 0)}{\|\mathbf{d}_\lambda^i\|^2} \|\hat{\mathbf{d}}_\lambda^i \times \hat{\mathbf{a}}^i\| d\lambda. \quad (5)$$

Once the lighting calculation has been performed, an initial set of weights are calculated for each bone. These are used in place of the binary vector \mathbf{p}^i in the diffusion equation above. To ensure a normalised set of initial weights, the j -th element of this initialisation vector is computed as $p_j^i = L_j^i / \sum_i L_j^i$.

Figure 2 shows, as an inset figure, the result of using these initial weights directly. These initial weights, although crude, are already suitable for use in skinning as opposed to the bone heat initial weights which lead to unacceptable levels of mesh distortion.

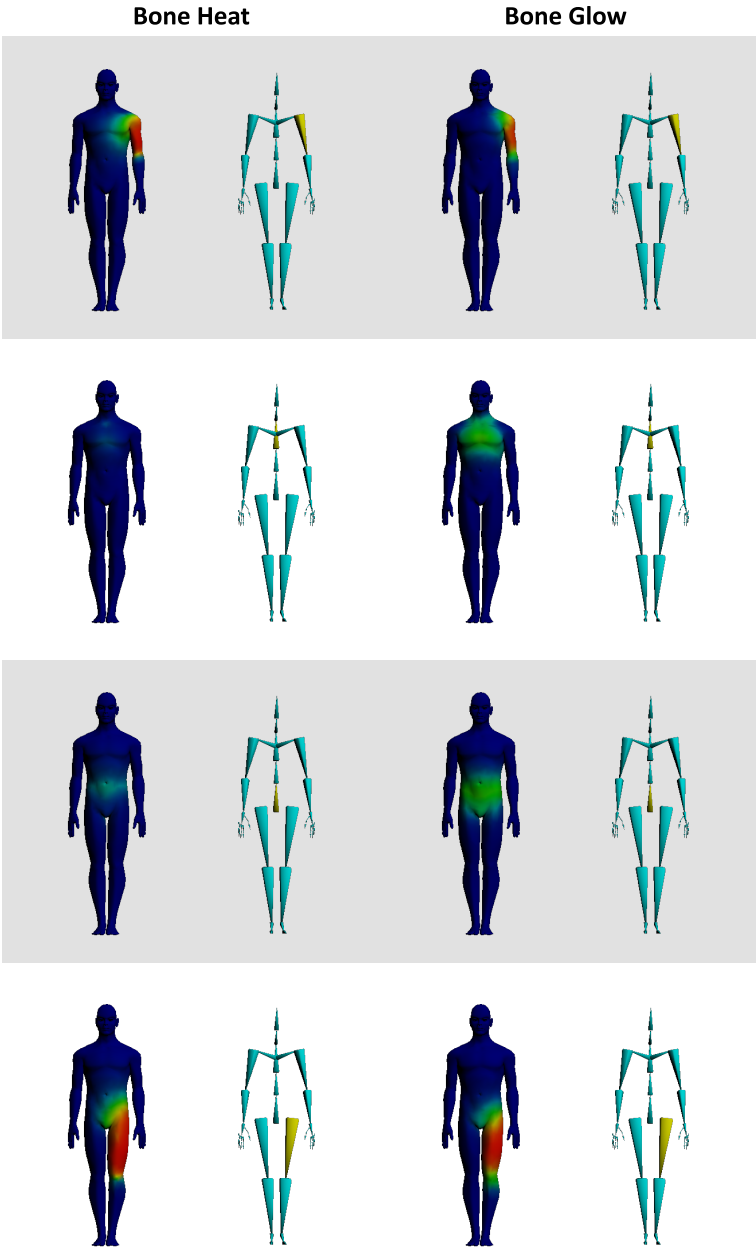


Fig. 3. A side-by-side comparison of bone heat and bone glow applied to a human mesh. In each sub figure, a particular bone has been highlighted and the weight assigned to each vertex has been indicated as in Fig. 2.

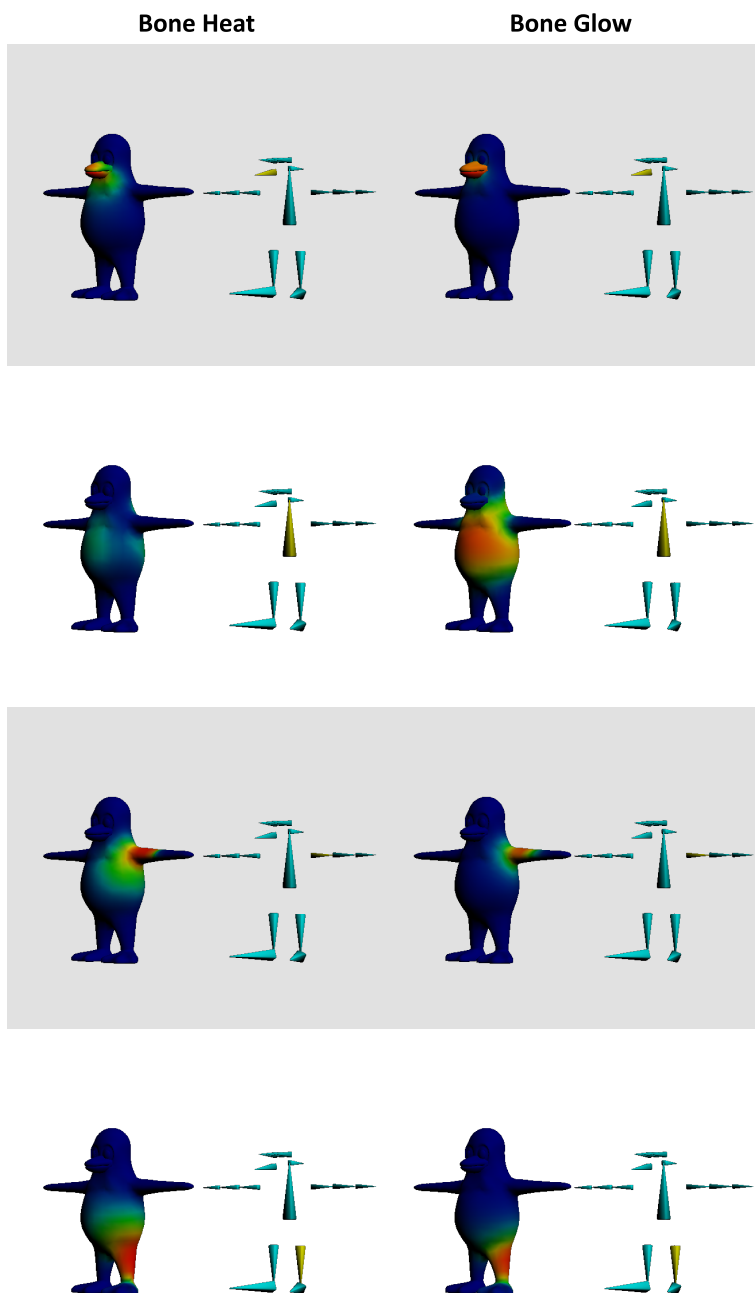


Fig. 4. A side-by-side comparison of bone heat and bone glow applied to a penguin mesh. In each sub figure, a particular bone has been highlighted and the weight assigned to each vertex has been indicated as in Fig. 2.

3 Validation

Although the weighting scheme proposed above is physically motivated in so far as it models a physical process to arrive at an initial weighting, it does not aim to form a set of weights which would lead a particular deformation scheme to closely approximate any physical mesh model. Indeed many usual aims of a mesh deformation scheme, for example volume preservation, are not of primary importance to the games industry. Instead more qualitative metrics of success, such as ‘naturalness’ are important along with a number of non-traditional metrics, such as suitability for implementation on the parallel computing architecture of modern graphics processors.

In this section we attempt to show a degree of validation for bone glow by example; we show examples of weight assignment using both bone heat and bone glow. We then discuss which approach gives rise to solutions with desirable characteristics.

Figure 2 provides a representative illustration of our results; a simple mesh and armature representing a human male were associated automatically using bone heat and bone glow. The underlying armature was posed and the mesh skinned using simple linear blend skinning. Bone heat strongly associates the chest and upper arm bone leading to severe distortion when the arm is posed. Bone glow, on the other hand, correctly assigns most weight to the arm portion of the mesh. The subsequent skinned mesh, although showing some chest deformation, appears far more ‘natural’.

Figures 3 and 4 shows a wider array of examples showing the differences between bone heat and bone glow. The successes of bone heat are mirrored with bone glow. For example, the left leg bone of the male human model has almost identical weights with bone glow and bone heat. Where bone heat and bone glow differ, bone glow appears to produce more ‘natural’ solutions. For example, using bone glow the upper and back bones of the human male figure have strong weighting in the upper and lower torso area, as expected. With bone heat, on the other hand, the weighting is very low.

If we examine the penguin model a similar pattern emerges. The bone glow and bone heat solutions for the limbs are similar—although the bone glow solution is ‘tighter’ around the joint areas. In contrast, the back and beak solutions are clearly superior when using bone glow.

4 Conclusions

We have presented a method for automatically assigning a per-vertex and per-bone weighting to an input mesh given only an underlying skeleton. When combined with an appropriate weight-based deformation scheme it allows an unsophisticated user to rapidly go from a self-authored model and skeleton to animating that model in a natural manner.

In the current implementation, the lighting integral is calculated numerically by stepping along the bone calculating the integrand at each point and summing

the results. Given a suitable approximation for the visibility function this integral could be approximated with an analytic result speeding up the computation considerably.

Acknowledgements. This research was supported by funding from Geomerics Ltd and the UK Department of Trade and Industry. We thank Andrew Kator and Jennifer Legaz for licensing the male human and tux meshes used within this paper under a Creative Commons Attribution 3.0 United States License.

References

1. Baran, I., Popović, J.: Automatic rigging and animation of 3D characters. *ACM Transactions on Graphics* 26(3) (2007)
2. Lewis, J.P., Corder, M., Fong, N.: Pose space deformation: a unified approach to shape interpolation and skeleton-driven deformation. In: *SIGGRAPH 2000: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, New York, NY, USA, pp. 165–172. ACM Press/Addison-Wesley Publishing Co. (2000)
3. Wareham, R.: *Computer Graphics Using Conformal Geometric Algebra*. PhD thesis, University of Cambridge (January 2007)
4. Kavan, L., Collins, S., Žára, J., O’Sullivan, C.: Skinning with dual quaternions. In: *I3D 2007: Proceedings of the 2007 symposium on Interactive 3D graphics and games*, pp. 39–46. ACM, New York (2007)
5. Kavan, L., Žára, J.: Spherical blend skinning: a real-time deformation of articulated models. In: *I3D 2005: Proceedings of the 2005 symposium on Interactive 3D graphics and games*, pp. 9–16. ACM, New York (2005)
6. The Blender project, <http://www.blender.org/>
7. Meyer, M., Desbrun, M., Schröder, P., Baar, A.H.: Discrete differential-geometry operators for triangulated 2-manifolds. In: Hege, H.C., Polthier, K. (eds.) *Visualization and Mathematics III*, pp. 35–57. Springer, Heidelberg (2003)