sven machon

How can you generate the feather coat of a bird using the weights of the skeleton with python?

Graduation work 2021-2022

Digital Arts and Entertainment

Howest.be

Contents

[ABSTRACT 2](#_Toc84419698)

[INTRODUCTION 3](#_Toc84419699)

[RELATED WORK 3](#_Toc84419700)

[1. Topic 1 3](#_Toc84419701)

[1.1. Subtopic 1 3](#_Toc84419702)

[1.1. Subtopic 2 3](#_Toc84419703)

[2. Topic 2 3](#_Toc84419704)

[CASE STUDY 4](#_Toc84419705)

[1. introduction 4](#_Toc84419706)

[2. Modelling 4](#_Toc84419707)

[2.1. Blockout 4](#_Toc84419708)

[2.2. Zbrush 5](#_Toc84419709)

[3. Texturing 5](#_Toc84419710)

[4. Shading 5](#_Toc84419711)

[5. Lighting 6](#_Toc84419712)

[EXPERIMENTS & RESULTS 7](#_Toc84419713)

[DISCUSSION 7](#_Toc84419714)

[CONCLUSION & FUTURE WORK 7](#_Toc84419715)

[BIBLIOGRAPHY 9](#_Toc84419716)

[APPENDICES 10](#_Toc84419717)

# ABSTRACT

**An abstract explains the aim of the paper in very brief, (the methods, results, etc.). Maximum length 150 words**

# INTRODUCTION

**In the introduction, you write the background of your topic, explain the purpose of the paper more broadly, and explain the hypothesis, and the research question(s).**

# RELATED WORK

Computer generated feather coats are a well known and explored challenge in computer graphics. Several approaches have been explored to achieve a satisfactory result. Such as using field lines [1], [2], Bezier Curves [3], constraint surfaces [4] or using the weights of the skeleton[5], [6]. In this chapter we will discuss some of the techniques that have been used to generate feather coats.

## Animated feather coats using field lines

A picture containing sky, close

Description automatically generated

S. Bangay proposed a technique using a vector field and field lines [2]. By creating a vector field and using the field lines within you can generate a feather coat with minimal key orientations. Using the surface orientation vectors of each vertex and the vertex normal, a vector field is created surrounding the base object. Field lines are then created through following the vector field from each vertex. The vector determined at each point along the field line will then be used to estimate the next point.   
To make up for the two-dimensional nature of a feather, a second field needs to be created perpendicular to the first. Using both fields you can build a coordinate system with the axes corresponding to feather thickness, length and width.  
The use of vectors allows for more control over the feathers. The feathers of a bird are not the same scale everywhere, often the feathers around the head are smaller and more densely packed. By using key vectors that take length into consideration, you can create a feather coat with variation in feather scale.

Figure 1: Surface orientation field and field lines [2]

After creating the vector field, feathers are placed at every vertex of the base object by duplicating a base feather mesh. All feathers will then be transformed from the root up, following the field lines.

## Animated Feather coats using implicit constraint surfaces

Diagram

Description automatically generatedA. Weber and G. Gornowicz [4] presented a way to generate feather coats using implicit constraint surfaces. Before all else, dozens of guide feathers are placed on key locations along the object surface. Next all other feather roots are generated. At these roots the feathers will be instanced by interpolating between the different guides.

The primary goal of this technique is to prevent interpenetration between the feathers. To achieve this an implicit constraint surface is defined for each feather that does not intersect with any other constraint surface.  
This constraint surface must intersect the bird surface at the feather root position, while not intersecting with all other constraint surfaces.

Figure 2: Feather in constraint surface [4]

Every feather has to have root position xr on the surface of the bird S, x is a point on S and nx is the surface normal at point x. Cis a continuous constraint surface. M(∅) is a monotonically increasing scalar mapping function of scalar potential ∅, from hereon referred to as the loft function.x

To ensure that the feathers are places at their root positions a few conditions must be met. The constraint surface must intersect the bird at the root position and the loft function must map zero to zero.

All root positions should have a unique scalar potential value, this can be done by slightly adjusting values of any root positions with similar potential values.  
If two constraint surfaces C1 and C2 with roots xr1 and xr2 intersect, they must do so at the same x

Therefor

Since all root positions have unique scalar values and constraint surfaces with unique scalar potentials will never intersect.

Once the constraint surfaces have been defined you can begin constructing the individual feathers. A feather will be constructed along the associated constraint surface, but there are a few characteristics of the feather that you would want to preserve, to ensure the fidelity of the result. These characteristics are the direction of the shaft, length of the shaft, angle of the barbs off the shaft and the length of the barbs.   
You begin with the shaft direction and then go along the constraint surface for the length of the shaft, before doing the same with the barbs.

## Animated feather coats using the weights of a skeleton

Diagram

Description automatically generatedLiu et al [6] developed a technique to create feather coats by using the weights of the skeleton on a skinned mesh.  
When looking at a real life bird, it can be noticed that for certain species of birds the scattering of the feathers is comparable to that of the bones. Meaning that different types of feather often meet around the joint areas while the size and shape differ less within the region of each bone. By using the skeleton you gain numerous advantages. The usage of weights to interpolate is more intuitive than randomly scattered guides and requires fewer. Having your guides linked to the weights and at the joints also makes it easier to locate any guides that might need changing.

Figure 3: Input model (left) and skeleton (right) [6]

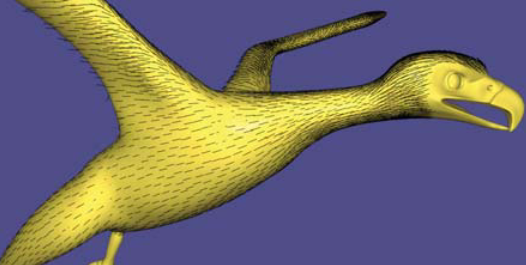
Skeleton-driven character animation makes use of a system of bones and joints bound to the mesh surface. Every bone affects the location of nearby vertices based on weights during animation. In Lui et al ‘s proposed technique the skeleton is expanded upon by associating the bones with a set of feather guides. These guides are user-defined and are then used to generate the body feathers covering the skin surface. The technique presented is, however, not suited for the flight feathers and as such will not be affected by the algorithm but completely controlled by the skeleton. Thanks to the skeleton and weights you are able to generate interpolated feathers on the surface. This has the added benefit of being able to procure the feather density based on the interpolated feather size, seeing as smaller feathers occur in more densely populated areas.  
By interpolating the guides on the sampled roots it is possible to initialize an orientation field which gets smoothed by projecting the orientation of every feather root to its tangent plane.

Figure 4: Orientation field after refinement [6]

A fish swimming in the water

Description automatically generated with low confidenceYou can then generate a feather coat, however this coat is going to be colliding with itself and the surface. To remedy this you adjust every feather individually, in an ordered fashion. To decide which should get adjusted first, you only compare feathers in proximity with each other since it is unlikely that feathers far from each other will collide.   
Once an ordering sequence has been generated each feather is going to get adjusted individually while considering the nearby surface and previously placed feathers. This is done while attempting to stack the feathers as compactly as possible and preserving the orientation field as closely as possible. Because the feather adjustments happen in order, the local surface does not change during adjustments. This means you can consider the prior feathers and nearby skin as one whole, allowing the collisions to be solved using height field projection. This height field is defined as a rectangular grid on the xy-field of the feather with a height value defined in each cell by projecting the local surface onto the grid. To solve the collision, compare the projected feather to the field height and compare the z-values to that of the height field. If no values of the projected feather are lower, no collision occurs.

Figure 5: Growth priority - red (high) to blue (low) [6]

A picture containing text, umbrella, accessory, dome

Description automatically generatedEvery feather individually gets adjusted to lay closer to the surface with interpenetrating with the surface or previous feathers. Since you want to preserve the orientation field as much as possible, every feather is considered rigid and can only be rotated around two axes, y (pitch) and the direction of the orientation field Ô’ (roll).  
The adjusting process then happens in three steps using bisection.  
You start with attempting to solve Fpitch(α)=0 using a big interval, failing this the feather is discarded as the surroundings are too concave or too populated with prior feathers, this will move the feather as close to the local surface as possible. The next step is to try and equalize the distance between the two sides of the feather by solving Froll(β)=0.  
This is achieved with a few rounds, decreasing the intervals each time in order to approximate the closest result.  
Lastly, you repeat the process of step two but with Fpitch(α)=0.   
  
This system can be adapted to make incremental changes in order to make it animation proof.

Figure 6: Individual feather before (left) and after (right) adjustment [6]

# CASE STUDY

## introduction

## Modelling

### Blockout

### Zbrush

## Texturing

## Shading

## Lighting

# EXPERIMENTS & RESULTS

**repeat the main topics, discuss your main findings, discuss the end result.**

# DISCUSSION

**repeat the main topics, discuss your main findings, discuss the end result.**

# CONCLUSION & FUTURE WORK

**repeat the main topics, discuss your main findings, discuss the end result.**

# BIBLIOGRAPHY

# Bibliography

[1] E. Zhang, K. Mischaikow, and G. Turk, “Vector Field Design on Surfaces.”

[2] Hannah. Slay, Shaun. Bangay, S. N. Spencer, SIGGRAPH., and European Association for Computer Graphics., *Animated feather  coats using field lines*. Association for Computing Machinery, 2007.

[3] L. Streit and W. Heidrich, “Generating Feather Coats Using Bezier Curves.”

[4] A. J. Weber and G. Gornowicz, “Collision-free construction of animated feathers using implicit constraint surfaces,” *ACM Transactions on Graphics*, vol. 28, no. 2, Apr. 2009, doi: 10.1145/1516522.1516523.

[5] L. Liu, X. Liu, B. Sheng, Y. Chen, and E. Wu, “Incremental collision-free feathering for animated surfaces,” *Visual Computer*, vol. 33, no. 6–8, pp. 883–890, Jun. 2017, doi: 10.1007/s00371-017-1369-6.

[6] Liu L., Li X., Chen Y., Liu X., Zhang J., and Wu E., “efficient feathering system with collision control,” 2015.

# APPENDICES