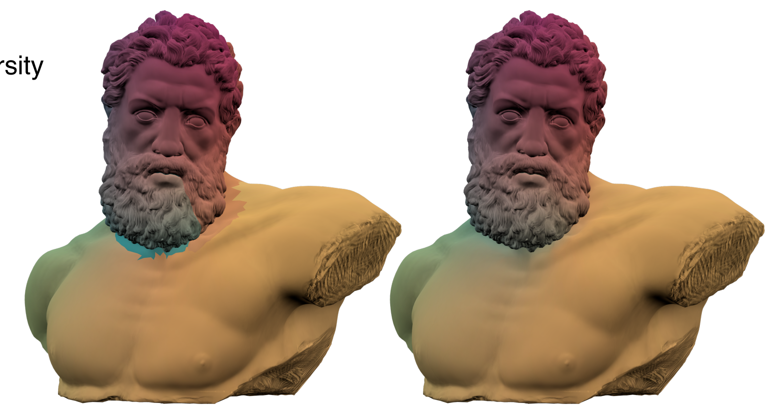
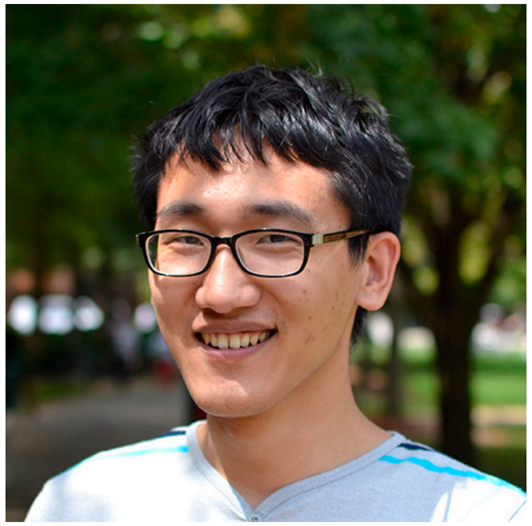
Seamless: Seam erasure and seam-aware decoupling of shape from mesh resolution

Songrun Liu*, George Mason University
Zachary Ferguson*, George Mason University
Alec Jacobson, University of Toronto
Yotam Gingold, George Mason University
(*Joint first authors)

CraGL Creativity and Graphics Lab
dgp dynamic graphics project
George Mason University
Computer Science University of Toronto



Thank you for the introduction.

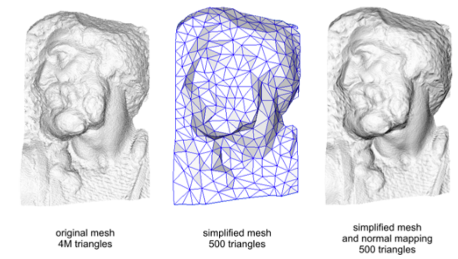


Songrun Liu

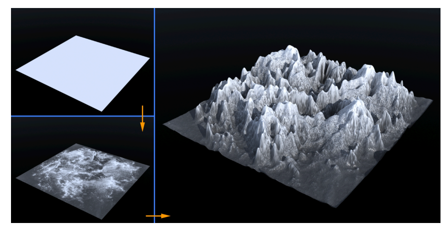
I will be presenting the entire paper as my co-author, Songrun Liu, could not make it today.

TEXTURES

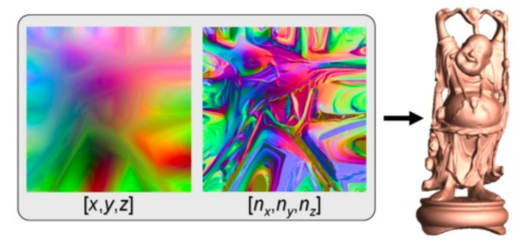
Color Map



Normal Map



Displacement Map



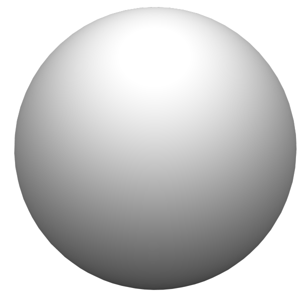
Geometry Images [Gu et al. 2002]

Textures are ubiquitous in computer graphics. They can be used to apply color <click> to a surface, <click> normals of a surface, <click> displacement of the surface, <click> and even the 3D positions of the entire surface, as Gu et al. showed with geometry images in 2002.

TEXTURE MAPPING

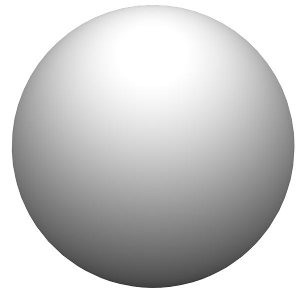
Given a 3D model <click> and a 2D texture image <click> we want to apply the 2D signal onto our 3D surface <click>.

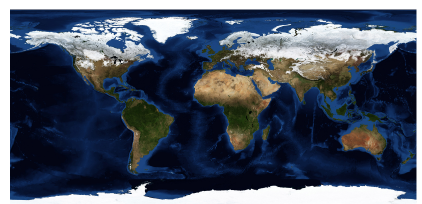
TEXTURE MAPPING



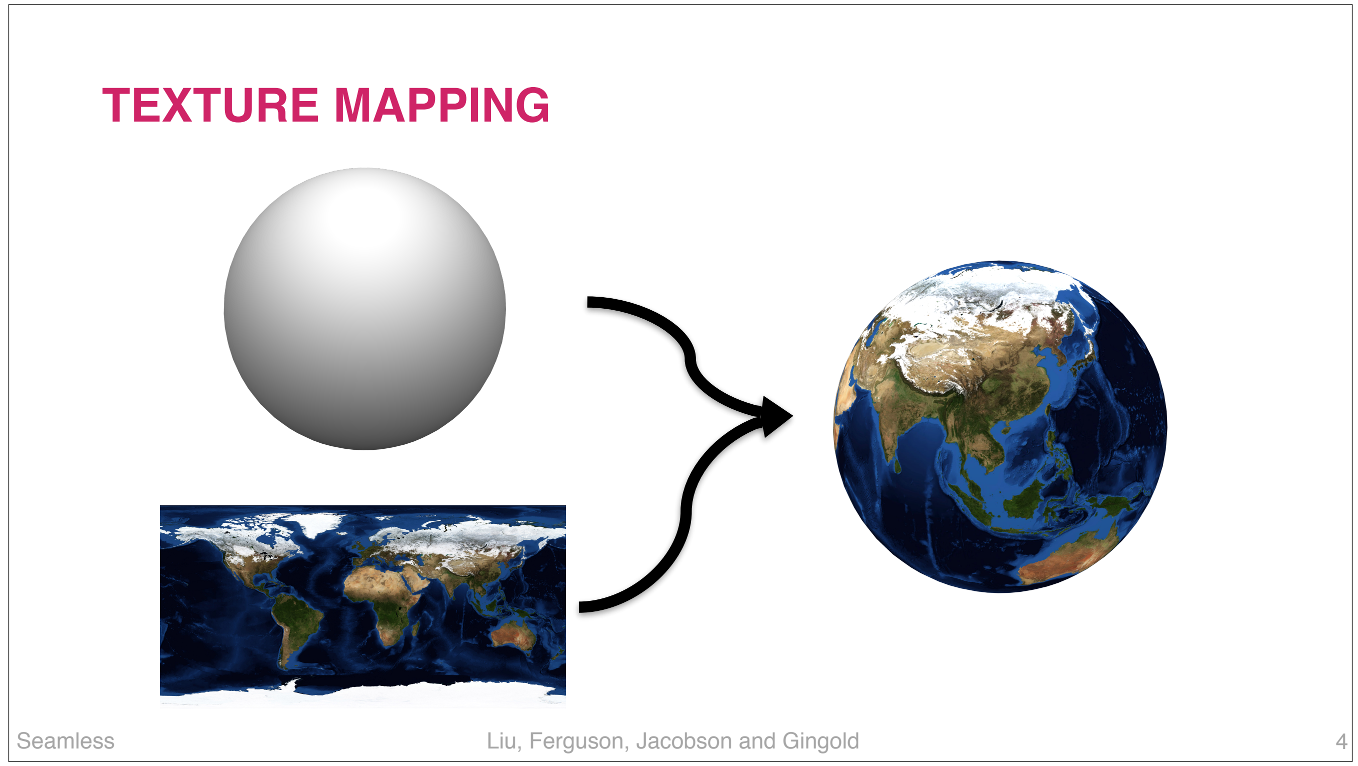
Given a 3D model <click> and a 2D texture image <click> we want to apply the 2D signal onto our 3D surface <click>.

TEXTURE MAPPING



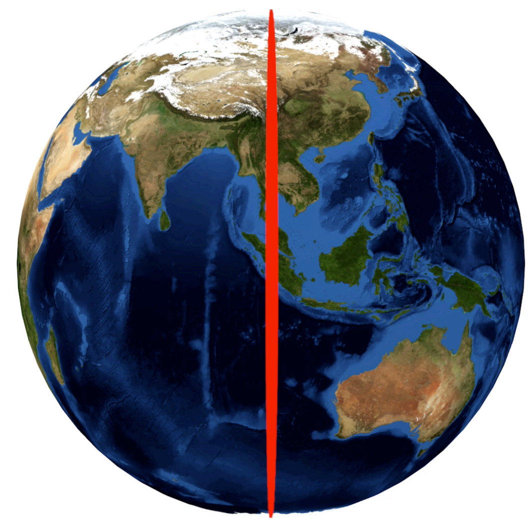


Given a 3D model <click> and a 2D texture image <click> we want to apply the 2D signal onto our 3D surface <click>.



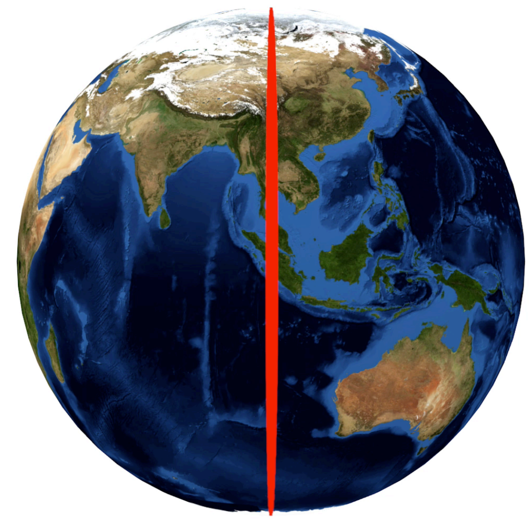
Given a 3D model <click> and a 2D texture image <click> we want to apply the 2D signal onto our 3D surface <click>.

2D PARAMETERIZATION



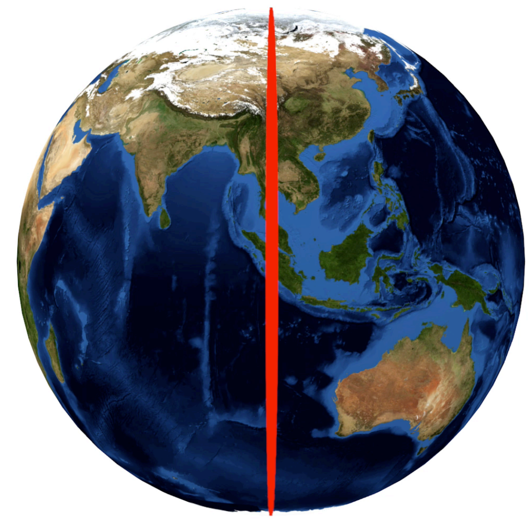
In order to apply the texture we need to create a mapping from our 3D model to a 2D parameterization. Because our model is in general complex, and we want to avoid overlaps, we need to introduce seams <click>. Using these seams as cuts in the mesh, we map to the plane <play>.

2D PARAMETERIZATION



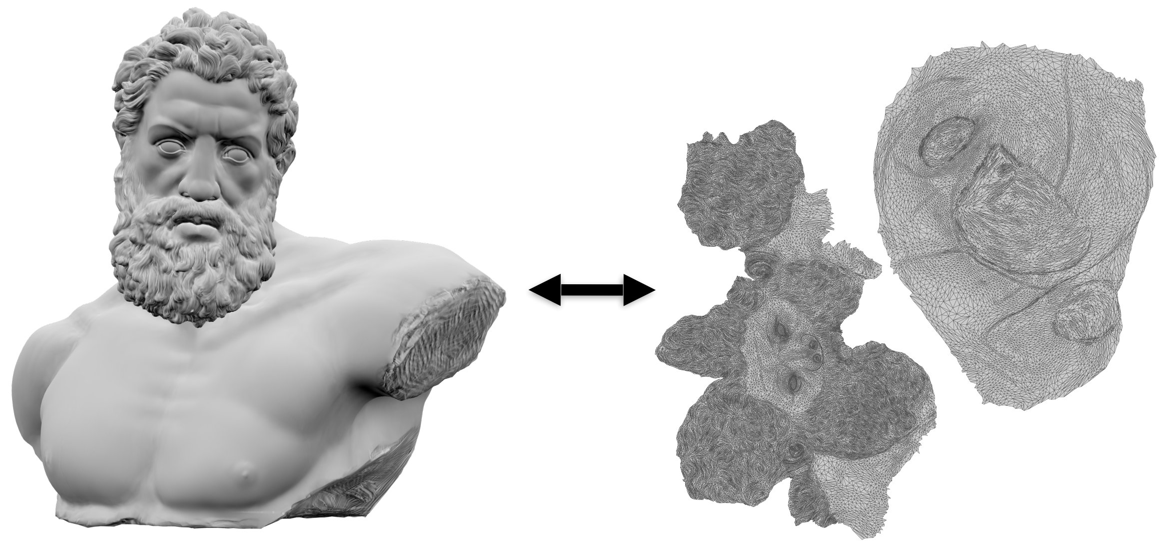
In order to apply the texture we need to create a mapping from our 3D model to a 2D parameterization. Because our model is in general complex, and we want to avoid overlaps, we need to introduce seams <click>. Using these seams as cuts in the mesh, we map to the plane <play>.

2D PARAMETERIZATION



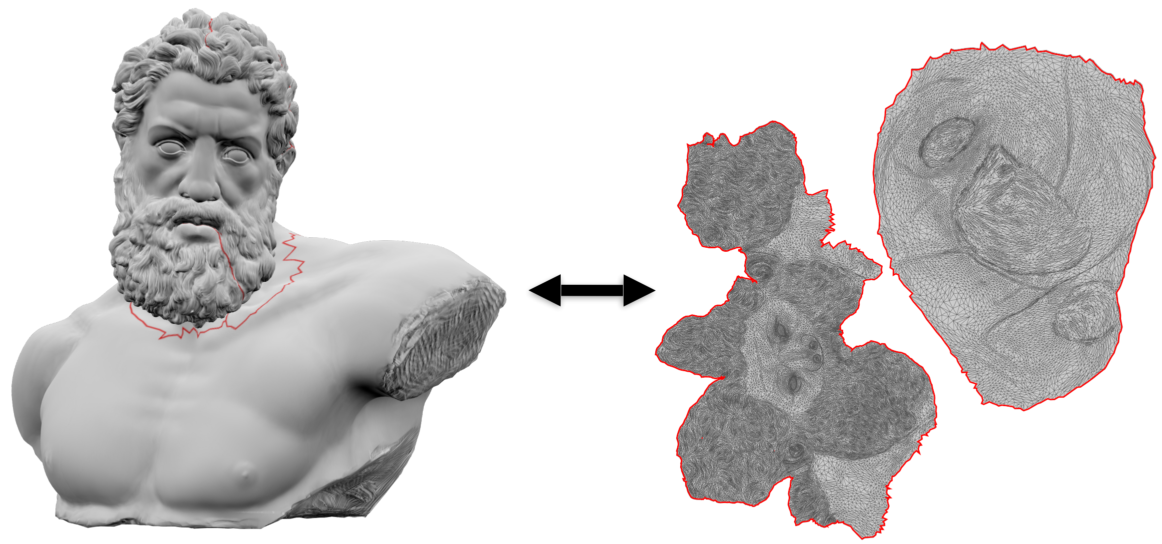
In order to apply the texture we need to create a mapping from our 3D model to a 2D parameterization. Because our model is in general complex, and we want to avoid overlaps, we need to introduce seams <click>. Using these seams as cuts in the mesh, we map to the plane <play>.

SEAMS



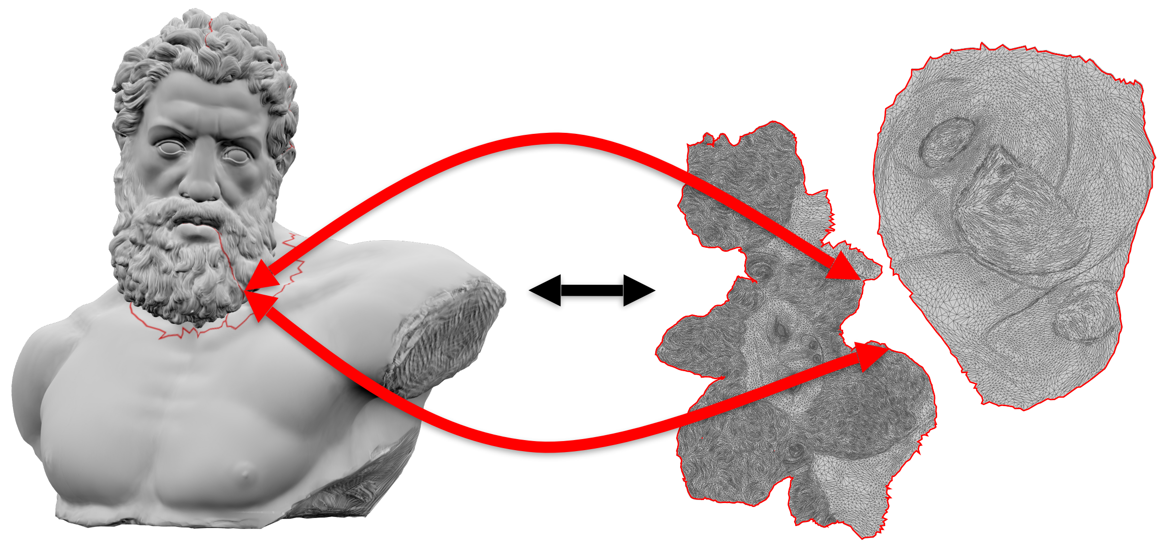
Here is a more realistic example. Given a parameterization, seams <click> are edges in 3D that map <click> to two different locations in our parameterization.

SEAMS



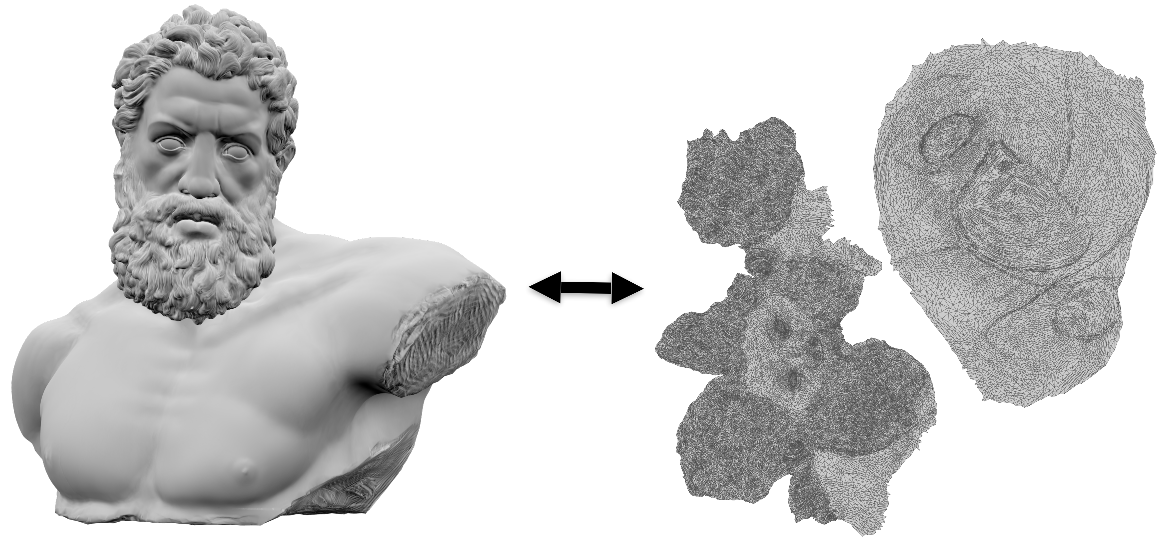
Here is a more realistic example. Given a parameterization, seams <click> are edges in 3D that map <click> to two different locations in our parameterization.

SEAMS



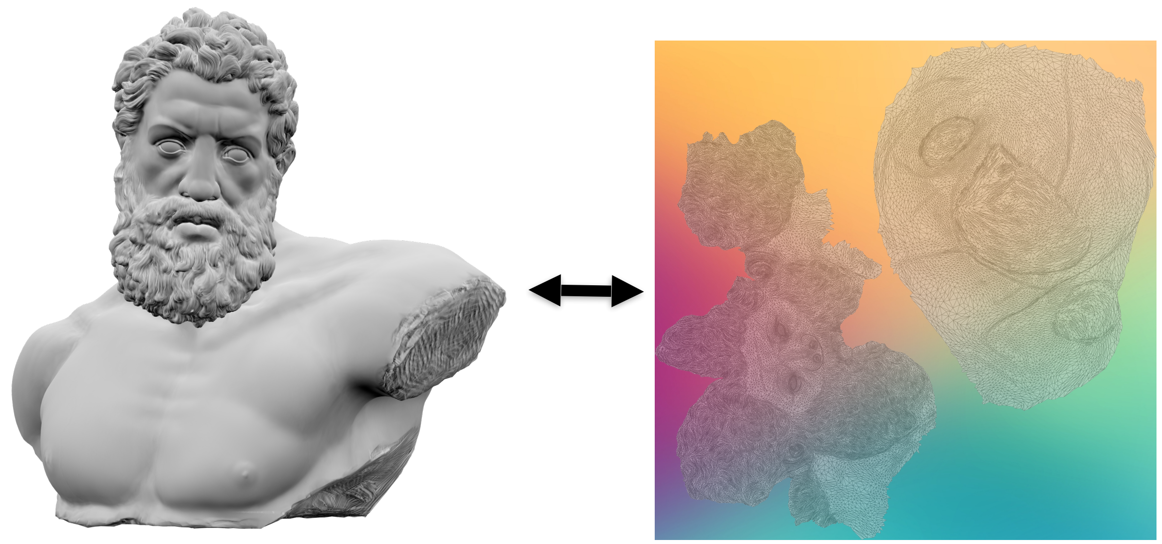
Here is a more realistic example. Given a parameterization, seams <click> are edges in 3D that map <click> to two different locations in our parameterization.

SEAM DISCONTINUITIES



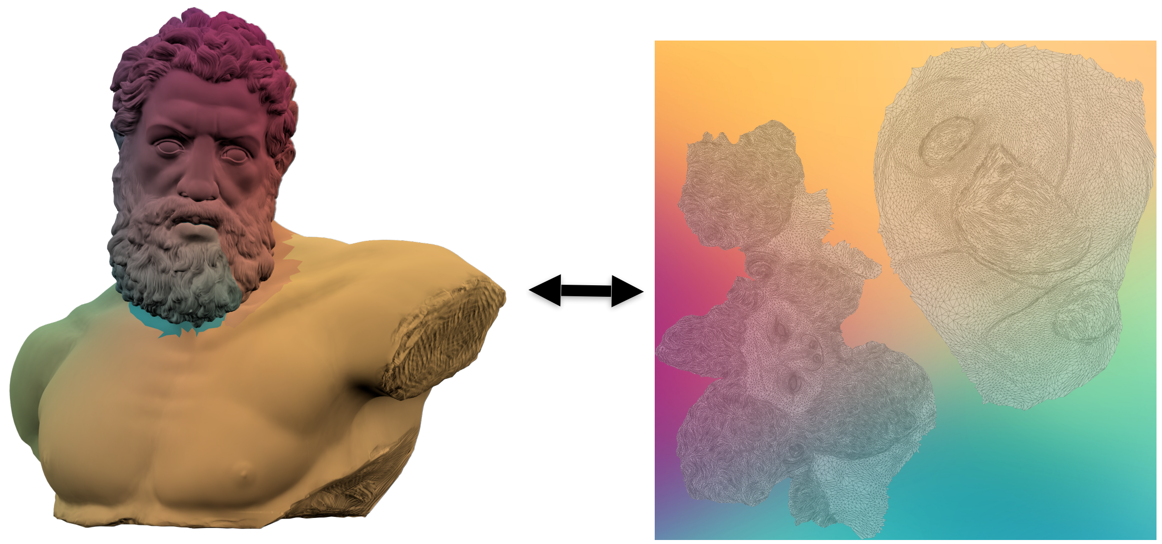
These seams, however, can cause problems. Say we want to map this colorful image <click> onto our model <click>. Visual artifacts <click> or seam discontinuities occur along the seam edges. This is a result of evaluating two different values for a single seam edge.

SEAM DISCONTINUITIES



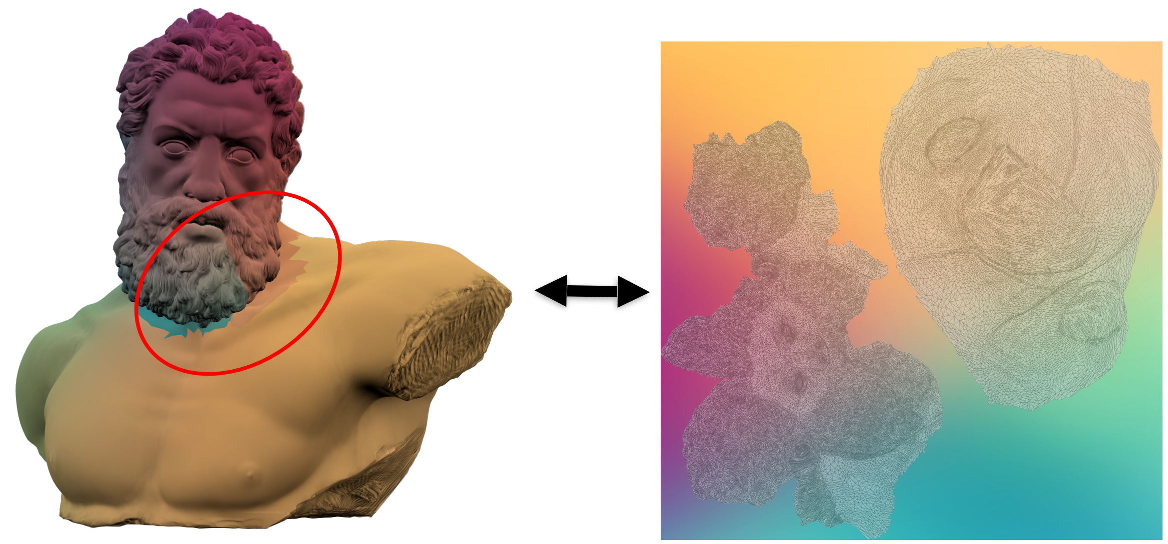
These seams, however, can cause problems. Say we want to map this colorful image <click> onto our model <click>. Visual artifacts <click> or seam discontinuities occur along the seam edges. This is a result of evaluating two different values for a single seam edge.

SEAM DISCONTINUITIES



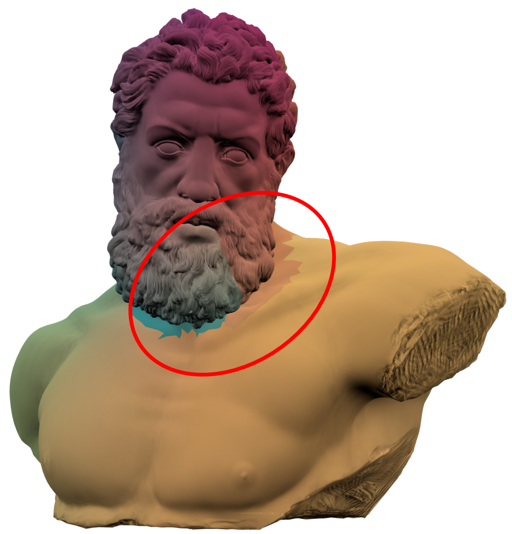
These seams, however, can cause problems. Say we want to map this colorful image <click> onto our model <click>. Visual artifacts <click> or seam discontinuities occur along the seam edges. This is a result of evaluating two different values for a single seam edge.

SEAM DISCONTINUITIES



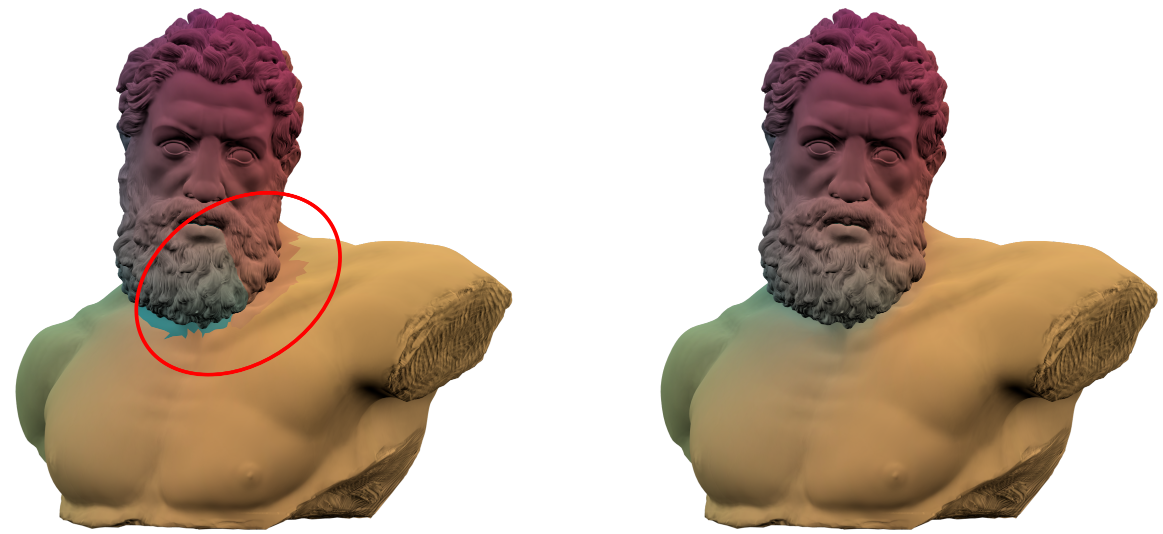
These seams, however, can cause problems. Say we want to map this colorful image <click> onto our model <click>. Visual artifacts <click> or seam discontinuities occur along the seam edges. This is a result of evaluating two different values for a single seam edge.

SEAM DISCONTINUITIES



Our goal is therefore to remove these seam discontinuities <click>.

SEAM DISCONTINUITIES

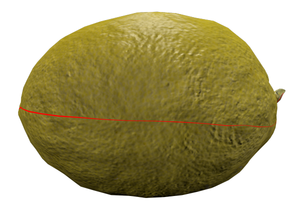


Our goal is therefore to remove these seam discontinuities <click>.

DISCONTINUITIES IN GEOMETRY IMAGES

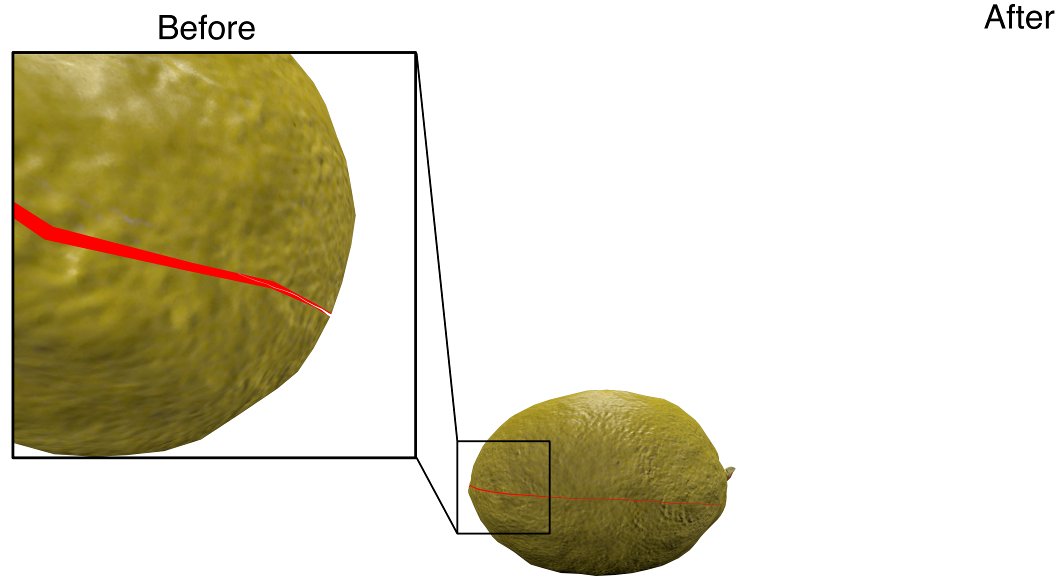
Before

After



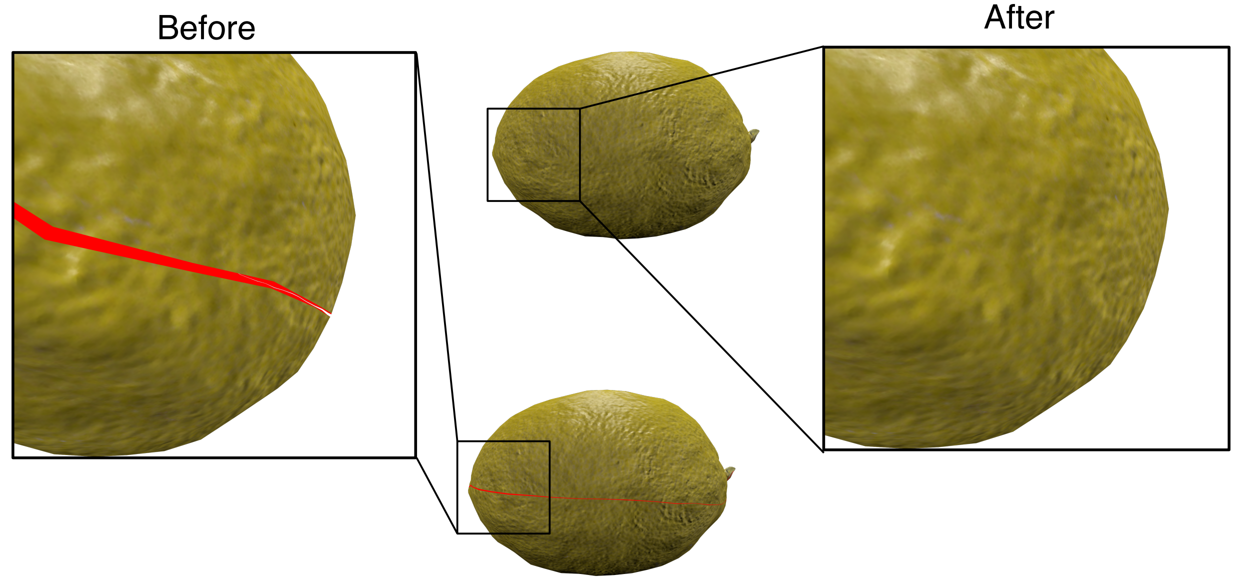
Seam discontinuities in geometry images result in cracks <click> as each seam half edge lands in a different 3D position. Here back faces are drawn in red. You can clearly see before that the inside of the lemon is visible. <click>The seam edges agree on a 3D location after our algorithm.

DISCONTINUITIES IN GEOMETRY IMAGES



Seam discontinuities in geometry images result in cracks <click> as each seam half edge lands in a different 3D position. Here back faces are drawn in red. You can clearly see before that the inside of the lemon is visible. <click>The seam edges agree on a 3D location after our algorithm.

DISCONTINUITIES IN GEOMETRY IMAGES



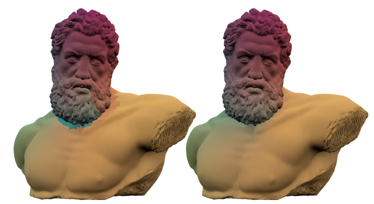
Seam discontinuities in geometry images result in cracks <click> as each seam half edge lands in a different 3D position. Here back faces are drawn in red. You can clearly see before that the inside of the lemon is visible. <click>The seam edges agree on a 3D location after our algorithm.

CONTRIBUTIONS

We present four novel techniques to solve the problem of seams. First, <click> we present our seam erasure which erases seam artifacts from texture images. Second, <click> we introduce a seam aware decimation in which we can decimate our model, collapsing surface elements, while reusing the same texture. Third, <click> we introduce a seam straightening algorithm to better assist our decimations. Lastly,<click> we show how skinning weights can be stored in textures to adaptively tessellate and deform a model. <click> Let us start with the seam erasure.

CONTRIBUTIONS

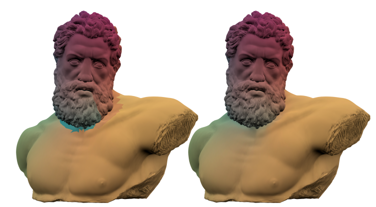
Seam Erasure



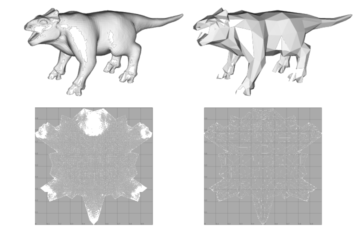
We present four novel techniques to solve the problem of seams. First, <click> we present our seam erasure which erases seam artifacts from texture images. Second, <click> we introduce a seam aware decimation in which we can decimate our model, collapsing surface elements, while reusing the same texture. Third, <click> we introduce a seam straightening algorithm to better assist our decimations. Lastly,<click> we show how skinning weights can be stored in textures to adaptively tessellate and deform a model. <click> Let us start with the seam erasure.

CONTRIBUTIONS

Seam Erasure



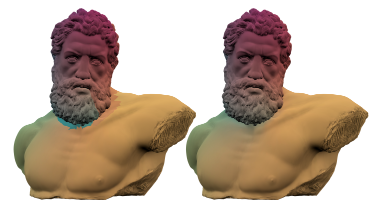
Seam Aware Decimation



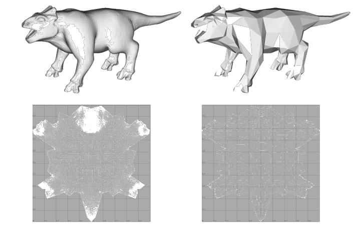
We present four novel techniques to solve the problem of seams. First, <click> we present our seam erasure which erases seam artifacts from texture images. Second, <click> we introduce a seam aware decimation in which we can decimate our model, collapsing surface elements, while reusing the same texture. Third, <click> we introduce a seam straightening algorithm to better assist our decimations. Lastly,<click> we show how skinning weights can be stored in textures to adaptively tessellate and deform a model. <click> Let us start with the seam erasure.

CONTRIBUTIONS

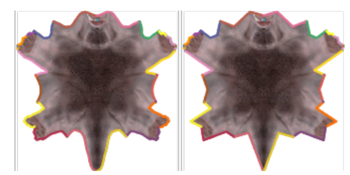
Seam Erasure



Seam Aware Decimation



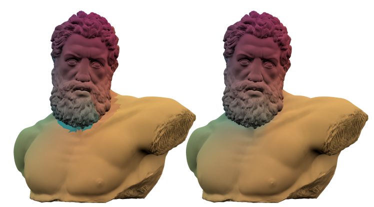
Seam Straightener



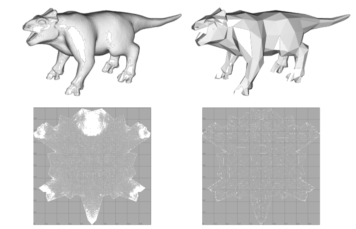
We present four novel techniques to solve the problem of seams. First, <click> we present our seam erasure which erases seam artifacts from texture images. Second, <click> we introduce a seam aware decimation in which we can decimate our model, collapsing surface elements, while reusing the same texture. Third, <click> we introduce a seam straightening algorithm to better assist our decimations. Lastly,<click> we show how skinning weights can be stored in textures to adaptively tessellate and deform a model. <click> Let us start with the seam erasure.

CONTRIBUTIONS

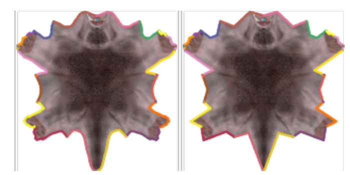
Seam Erasure



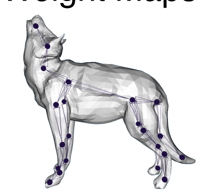
Seam Aware Decimation



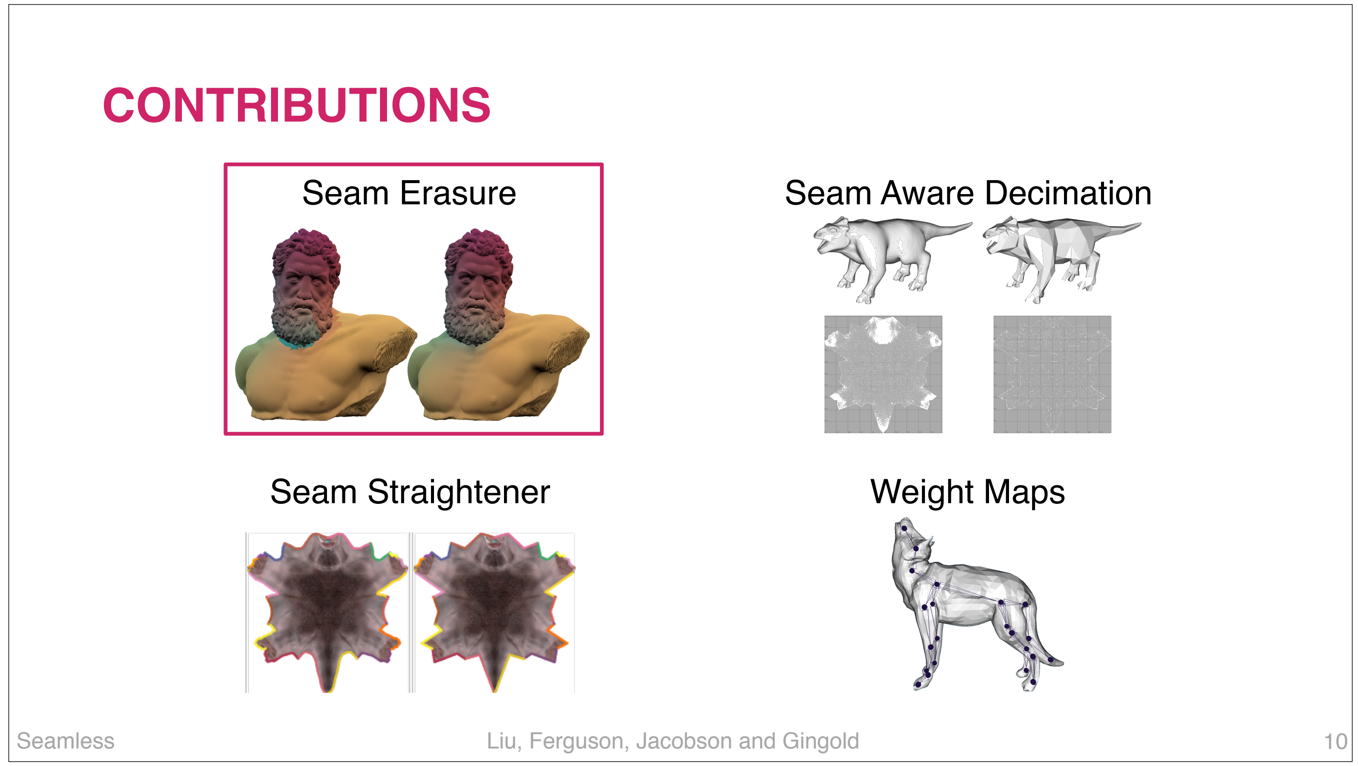
Seam Straightener



Weight Maps



We present four novel techniques to solve the problem of seams. First, <click> we present our seam erasure which erases seam artifacts from texture images. Second, <click> we introduce a seam aware decimation in which we can decimate our model, collapsing surface elements, while reusing the same texture. Third, <click> we introduce a seam straightening algorithm to better assist our decimations. Lastly,<click> we show how skinning weights can be stored in textures to adaptively tessellate and deform a model. <click> Let us start with the seam erasure.



We present four novel techniques to solve the problem of seams. First, <click> we present our seam erasure which erases seam artifacts from texture images. Second, <click> we introduce a seam aware decimation in which we can decimate our model, collapsing surface elements, while reusing the same texture. Third, <click> we introduce a seam straightening algorithm to better assist our decimations. Lastly,<click> we show how skinning weights can be stored in textures to adaptively tessellate and deform a model. <click> Let us start with the seam erasure.

OVERVIEW

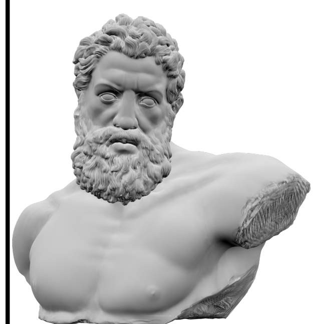
INPUT:

OUTPUT:

Our algorithm takes as input <click> a 3D model with a 2D parameterization <click> and <click> a texture image. We then output a texture with the seam erased <click>. This is a one time preprocess and requires no additional runtime cost.

OVERVIEW

INPUT:

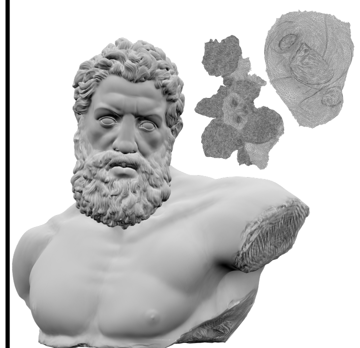


OUTPUT:

Our algorithm takes as input <click> a 3D model with a 2D parameterization <click> and <click> a texture image. We then output a texture with the seam erased <click>. This is a one time preprocess and requires no additional runtime cost.

OVERVIEW

INPUT:

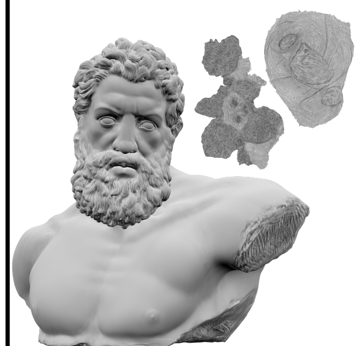


OUTPUT:

Our algorithm takes as input <click> a 3D model with a 2D parameterization <click> and <click> a texture image. We then output a texture with the seam erased <click>. This is a one time preprocess and requires no additional runtime cost.

OVERVIEW

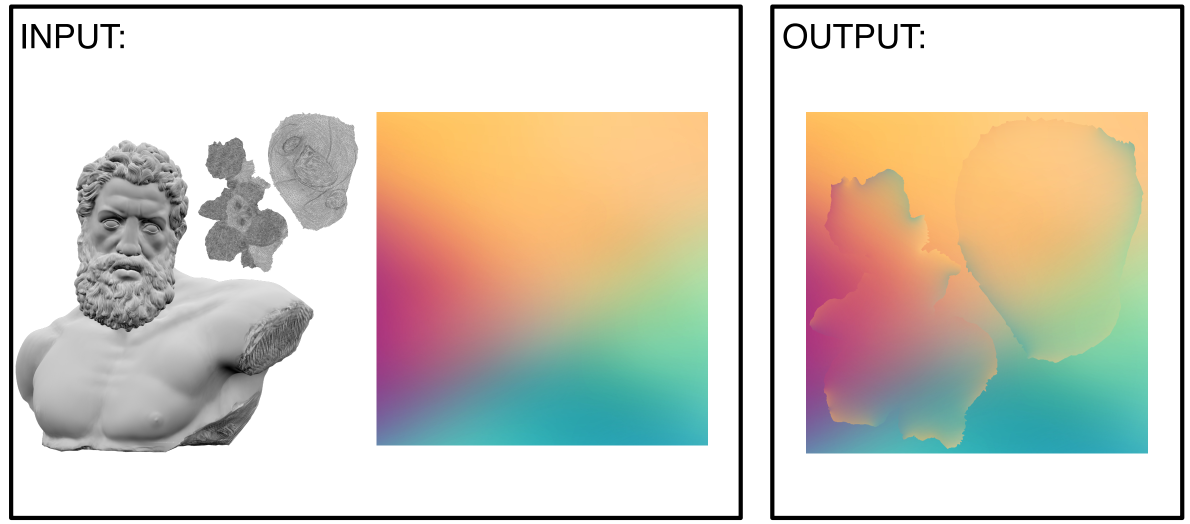
INPUT:



OUTPUT:

Our algorithm takes as input <click> a 3D model with a 2D parameterization <click> and <click> a texture image. We then output a texture with the seam erased <click>. This is a one time preprocess and requires no additional runtime cost.

OVERVIEW



Our algorithm takes as input <click> a 3D model with a 2D parameterization <click> and <click> a texture image. We then output a texture with the seam erased <click>. This is a one time preprocess and requires no additional runtime cost.

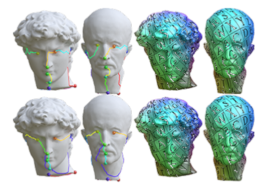
RELATED WORKS

Previous attempts have been made to fix the problem of seams. <click> Some control the parameterization. <click> Others modify the rendering pipeline, and <click> some avoid parameterization all together. Our work is orthogonal to the choice of the original parameterization and stands to make this family of work more useful. <click> The closest approach to ours was developed in industry and briefly described by Iwanicki [2013], who also optimize texture values.

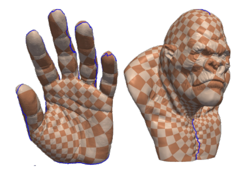
We share the motivation to operate on existing parameterizations and leave the rendering pipeline unchanged.

RELATED WORKS

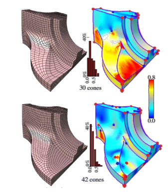
Control the Parameterization



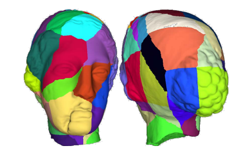
[Aigerman et al. 2015]



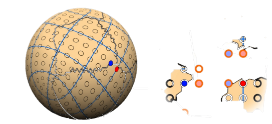
[Kovalsky et al. 2016]



[Myles and Zorin 2013]



[Purnomo et al. 2004]



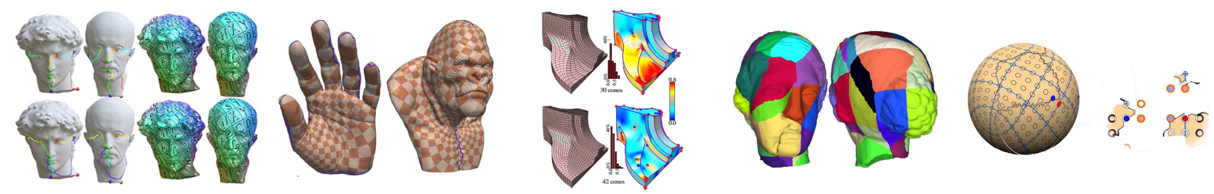
[Ray et al. 2010]

Previous attempts have been made to fix the problem of seams. <click> Some control the parameterization. <click> Others modify the rendering pipeline, and <click> some avoid parameterization all together. Our work is orthogonal to the choice of the original parameterization and stands to make this family of work more useful. <click> The closest approach to ours was developed in industry and briefly described by Iwanicki [2013], who also optimize texture values.

We share the motivation to operate on existing parameterizations and leave the rendering pipeline unchanged.

RELATED WORKS

Control the Parameterization



[Aigerman et al. 2015] [Kovalsky et al. 2016] [Myles and Zorin 2013] [Purnomo et al. 2004] [Ray et al. 2010]

Modify Rendering Pipeline

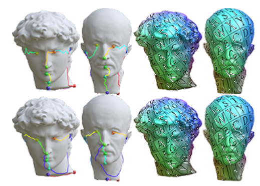
[Toth 2013] [Piponi and Borshukov 2000]

Previous attempts have been made to fix the problem of seams. <click> Some control the parameterization. <click> Others modify the rendering pipeline, and <click> some avoid parameterization all together. Our work is orthogonal to the choice of the original parameterization and stands to make this family of work more useful. <click> The closest approach to ours was developed in industry and briefly described by Iwanicki [2013], who also optimize texture values.

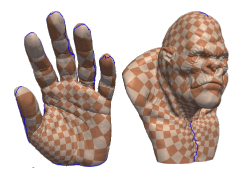
We share the motivation to operate on existing parameterizations and leave the rendering pipeline unchanged.

RELATED WORKS

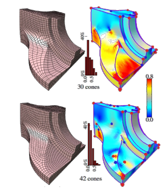
Control the Parameterization



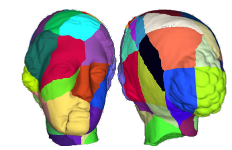
[Aigerman et al. 2015]



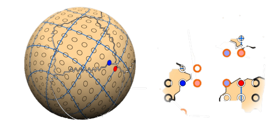
[Kovalsky et al. 2016]



[Myles and Zorin 2013]

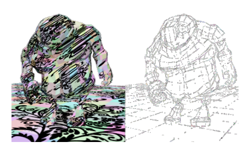


[Purnomo et al. 2004]



[Ray et al. 2010]

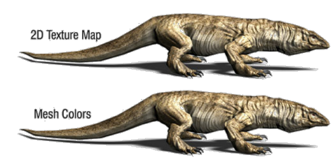
Modify Rendering Pipeline



[Toth 2013]

[Piponi and Borshukov 2000]

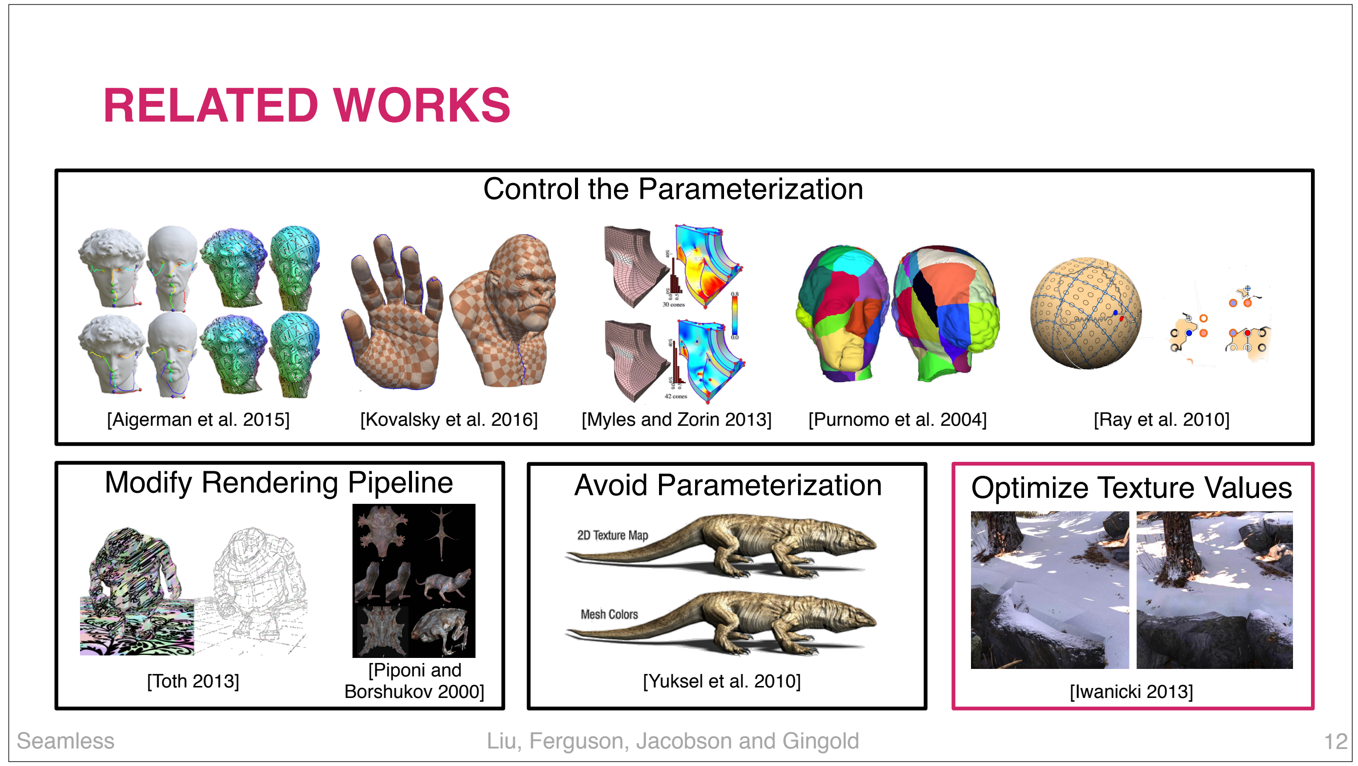
Avoid Parameterization



[Yuksel et al. 2010]

Previous attempts have been made to fix the problem of seams. <click> Some control the parameterization. <click> Others modify the rendering pipeline, and <click> some avoid parameterization all together. Our work is orthogonal to the choice of the original parameterization and stands to make this family of work more useful. <click> The closest approach to ours was developed in industry and briefly described by Iwanicki [2013], who also optimize texture values.

We share the motivation to operate on existing parameterizations and leave the rendering pipeline unchanged.



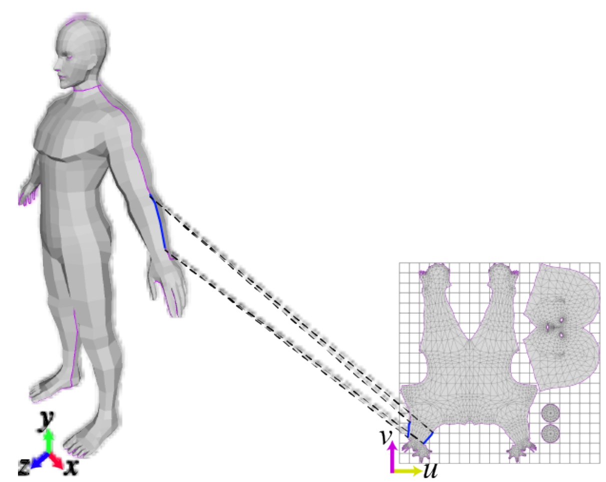
Previous attempts have been made to fix the problem of seams. <click> Some control the parameterization. <click> Others modify the rendering pipeline, and <click> some avoid parameterization all together. Our work is orthogonal to the choice of the original parameterization and stands to make this family of work more useful. <click> The closest approach to ours was developed in industry and briefly described by Iwanicki [2013], who also optimize texture values.

We share the motivation to operate on existing parameterizations and leave the rendering pipeline unchanged.

VALUE ALONG AN EDGE

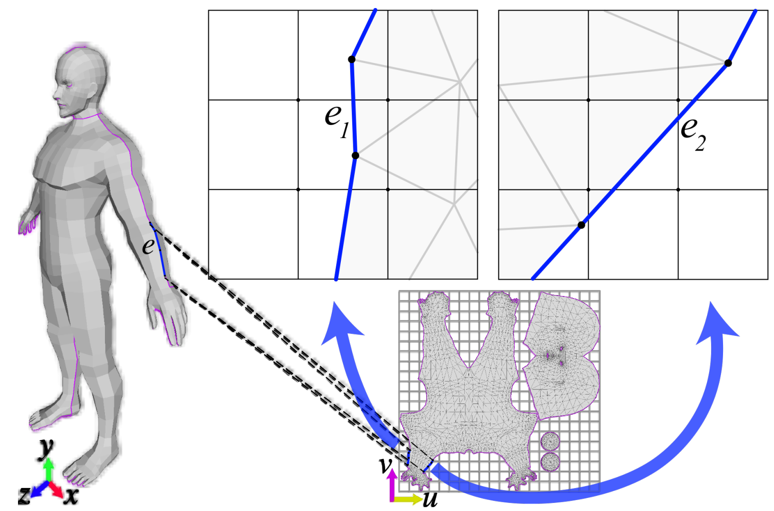
<click> For each edge along the seam there are two copies in the parameterization. <click> The value along the edge is a interpolation of surrounding texel values. <click>

VALUE ALONG AN EDGE



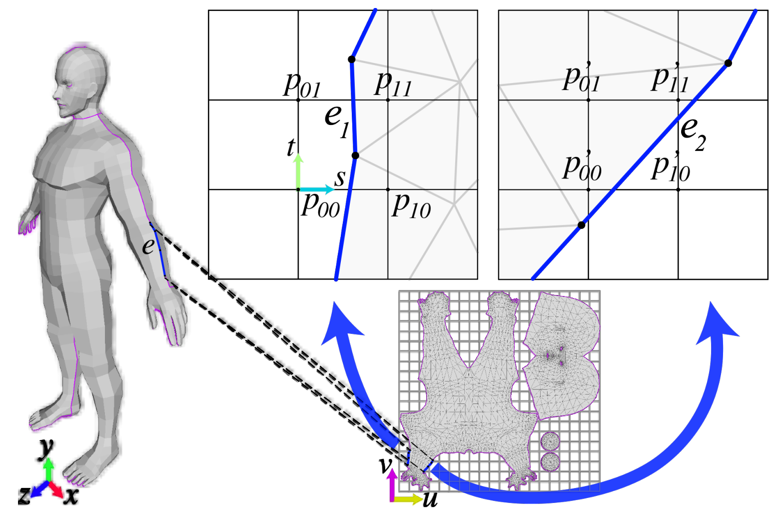
<click> For each edge along the seam there are two copies in the parameterization. <click> The value along the edge is a interpolation of surrounding texel values. <click>

VALUE ALONG AN EDGE



<click> For each edge along the seam there are two copies in the parameterization. <click> The value along the edge is a interpolation of surrounding texel values. <click>

VALUE ALONG AN EDGE

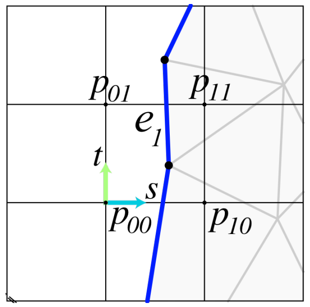


<click> For each edge along the seam there are two copies in the parameterization. <click> The value along the edge is a interpolation of surrounding texel values. <click>

BILINEAR INTERPOLATION

Bilinear interpolation:

$$\mathrm{Bilerp}(s,t) = (1-t)((p_{10}-p_{00})s + p_{00}) + t((p_{11}-p_{01})s + p_{01})$$

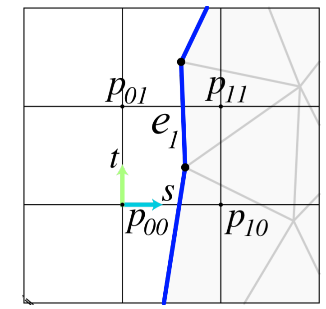


Bi-linear interpolation is used to determine values along the seam. Bilinear interpolation interpolates along the rows then between the interpolated values. <click> <read slide>

BILINEAR INTERPOLATION

Bilinear interpolation:

$$\text{Bilerp}(s,t) = (1-t)((p_{10}-p_{00})s + p_{00}) + t((p_{11}-p_{01})s + p_{01})$$



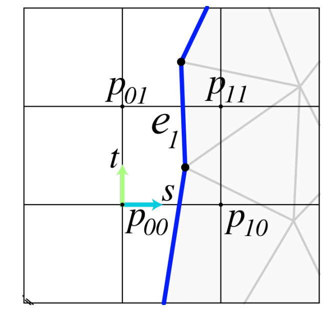
This is linear in $\mathbf{p}$ and quadratic in edge parameter γ:

$$\text{Bilerp}(e) = m(e(\gamma))\mathbf{p}$$

Bi-linear interpolation is used to determine values along the seam. Bilinear interpolation interpolates along the rows then between the interpolated values. <click> <read slide>

BILINEAR INTERPOLATION

Bilinear interpolation:



$$\text{Bilerp}(s,t) = (1-t)((p_{10}-p_{00})s + p_{00}) + t((p_{11}-p_{01})s + p_{01})$$

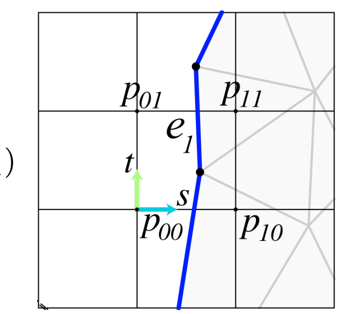
This is linear in $\mathbf{p}$ and quadratic in edge parameter γ:

$$\text{Bilerp}(e) = m(e(\gamma))\mathbf{p} = \gamma^2 \cdot a(\gamma)^{\mathsf{T}}\mathbf{p} + \gamma \cdot b(\gamma)^{\mathsf{T}}\mathbf{p} + c(\gamma)^{\mathsf{T}}\mathbf{p}$$

Bi-linear interpolation is used to determine values along the seam. Bilinear interpolation interpolates along the rows then between the interpolated values. <click> <read slide>

BILINEAR INTERPOLATION

Bilinear interpolation:



$$\mathrm{Bilerp}(s,t) = (1-t)((p_{10}-p_{00})s + p_{00}) + t((p_{11}-p_{01})s + p_{01})$$

This is linear in $\mathbf{p}$ and quadratic in edge parameter γ:

$$\mathrm{Bilerp}(e) = m(e(\gamma))\mathbf{p} = \gamma^2 \cdot a(\gamma)^\mathsf{T}\mathbf{p} + \gamma \cdot b(\gamma)^\mathsf{T}\mathbf{p} + c(\gamma)^\mathsf{T}\mathbf{p}$$

$\gamma \in [0,1], a, b, c$ are sparse column vectors of coefficients for e, and $\mathbf{p}$ is the column vector of all samples in the texture.

Bi-linear interpolation is used to determine values along the seam. Bilinear interpolation interpolates along the rows then between the interpolated values. <click> <read slide>

SEAM ENERGY

$$\int_0^1 \|m(e_1)\mathbf{p} - m(e_2)\mathbf{p}\|^2 d\gamma$$

We integrate the squared difference of edge values. The edge value is linear in p. P is independent of gamma, so <click> we can pull it out. The integral can be analytically integrated <click> to be a square matrix. <click> We then sum over all edge pairs to produce our final seam energy. This energy is quadratic in p.

SEAM ENERGY

$$\mathbf{p}^\mathsf{T}\left(\int_0^1 \|m(e_1) \quad - m(e_2) \quad \|^2 d\gamma\right)\mathbf{p}$$

We integrate the squared difference of edge values. The edge value is linear in p. P is independent of gamma, so <click> we can pull it out. The integral can be analytically integrated <click> to be a square matrix. <click> We then sum over all edge pairs to produce our final seam energy. This energy is quadratic in p.

SEAM ENERGY

$$\mathbf{p}^{\mathsf{T}} \left(\qquad M_{e_1,e_2} \qquad \right) \mathbf{p}$$

We integrate the squared difference of edge values. The edge value is linear in p. P is independent of gamma, so <click> we can pull it out. The integral can be analytically integrated <click> to be a square matrix. <click> We then sum over all edge pairs to produce our final seam energy. This energy is quadratic in p.

SEAM ENERGY

$$\mathbf{p}^{\mathsf{T}} M \mathbf{p} = \sum_{e_1, e_2 \in \text{seams}} \mathbf{p}^{\mathsf{T}} \left(\qquad M_{e_1,e_2} \qquad \right) \mathbf{p}$$

We integrate the squared difference of edge values. The edge value is linear in p. P is independent of gamma, so <click> we can pull it out. The integral can be analytically integrated <click> to be a square matrix. <click> We then sum over all edge pairs to produce our final seam energy. This energy is quadratic in p.

POSSIBLE SOLUTIONS

$$\mathbf{p}^{\mathsf{T}} M \mathbf{p} = 0$$

It is important to note this energy is under constrained. The trivial solution as well as all constant textures <click> are seamless <click> <click> <click>.

POSSIBLE SOLUTIONS

It is important to note this energy is under constrained. The trivial solution as well as all constant textures <click> are seamless <click> <click> <click>.

OPTIMIZATION

Our total energy is:

$E(\mathbf{p}) =$

In order to choose a solution we introduce a set of regulatory energies. <click> We use a least squares approach to express the reliability of interior texels. <click> This however preserves the outside values too exactly, resulting in poor interpolation. Therefore, <click> we add a gradient domain term to allow for global effects resulting from enforcing seam continuity. As this animation plays the gradient weight increases. <play> <click> For smoothness, we introduce an energy term that measures the integrated C^1 discontinuity across seams. This is implemented similarly to the seam energy.

<click> We optimize this energy subject to being in the null space of the seam energy. <click> <read slide> a weighted term for the seam. <click> Where w_seam much larger then all other weights.

OPTIMIZATION

Our total energy is:

$$E(\mathbf{p}) = w_{\text{change}} \|\mathbf{p} - \mathbf{p_0}\|^2$$

In order to choose a solution we introduce a set of regulatory energies. <click> We use a least squares approach to express the reliability of interior texels. <click> This however preserves the outside values too exactly, resulting in poor interpolation. Therefore, <click> we add a gradient domain term to allow for global effects resulting from enforcing seam continuity. As this animation plays the gradient weight increases. <play> <click> For smoothness, we introduce an energy term that measures the integrated C^1 discontinuity across seams. This is implemented similarly to the seam energy.

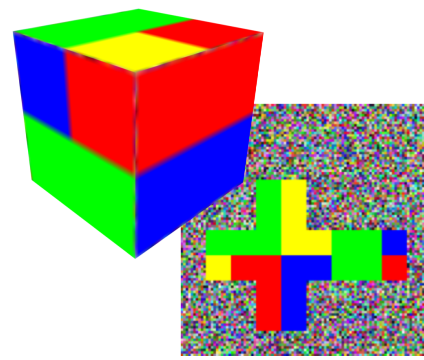
<click> We optimize this energy subject to being in the null space of the seam energy. <click> <read slide> a weighted term for the seam. <click> Where w_seam much larger then all other weights.

OPTIMIZATION

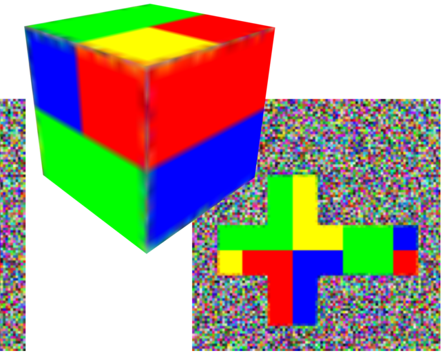
Our total energy is:

$$E(\mathbf{p}) = w_{\text{change}}\|\mathbf{p} - \mathbf{p_0}\|^2$$

Original



With Only E_{Change}



In order to choose a solution we introduce a set of regulatory energies. <click> We use a least squares approach to express the reliability of interior texels. <click> This however preserves the outside values too exactly, resulting in poor interpolation. Therefore, <click> we add a gradient domain term to allow for global effects resulting from enforcing seam continuity. As this animation plays the gradient weight increases. <play> <click> For smoothness, we introduce an energy term that measures the integrated C^1 discontinuity across seams. This is implemented similarly to the seam energy.

<click> We optimize this energy subject to being in the null space of the seam energy. <click> <read slide> a weighted term for the seam. <click> Where w_seam much larger then all other weights.

OPTIMIZATION

Our total energy is:

$$E(\mathbf{p}) = w_{\text{change}}\|\mathbf{p} - \mathbf{p_0}\|^2 \quad + \quad w_{\nabla}\|\nabla\mathbf{p} - \nabla\mathbf{p_0}\|^2$$

In order to choose a solution we introduce a set of regulatory energies. <click> We use a least squares approach to express the reliability of interior texels. <click> This however preserves the outside values too exactly, resulting in poor interpolation. Therefore, <click> we add a gradient domain term to allow for global effects resulting from enforcing seam continuity. As this animation plays the gradient weight increases. <play> <click> For smoothness, we introduce an energy term that measures the integrated C^1 discontinuity across seams. This is implemented similarly to the seam energy.

<click> We optimize this energy subject to being in the null space of the seam energy. <click> <read slide> a weighted term for the seam. <click> Where w_seam much larger then all other weights.

OPTIMIZATION

Our total energy is:

$$E(\mathbf{p}) = w_{\text{change}}\|\mathbf{p} - \mathbf{p_0}\|^2 \quad + \quad w_{\nabla}\|\nabla\mathbf{p} - \nabla\mathbf{p_0}\|^2$$

In order to choose a solution we introduce a set of regulatory energies. <click> We use a least squares approach to express the reliability of interior texels. <click> This however preserves the outside values too exactly, resulting in poor interpolation. Therefore, <click> we add a gradient domain term to allow for global effects resulting from enforcing seam continuity. As this animation plays the gradient weight increases. <play> <click> For smoothness, we introduce an energy term that measures the integrated C^1 discontinuity across seams. This is implemented similarly to the seam energy.

<click> We optimize this energy subject to being in the null space of the seam energy. <click> <read slide> a weighted term for the seam. <click> Where w_seam much larger then all other weights.

OPTIMIZATION

Our total energy is:

$$E(\mathbf{p}) = w_{\text{change}}\|\mathbf{p} - \mathbf{p_0}\|^2 \qquad + \qquad w_{\nabla}\|\nabla\mathbf{p} - \nabla\mathbf{p_0}\|^2$$

In order to choose a solution we introduce a set of regulatory energies. <click> We use a least squares approach to express the reliability of interior texels. <click> This however preserves the outside values too exactly, resulting in poor interpolation. Therefore, <click> we add a gradient domain term to allow for global effects resulting from enforcing seam continuity. As this animation plays the gradient weight increases. <play> <click> For smoothness, we introduce an energy term that measures the integrated C^1 discontinuity across seams. This is implemented similarly to the seam energy.

<click> We optimize this energy subject to being in the null space of the seam energy. <click> <read slide> a weighted term for the seam. <click> Where w_seam much larger then all other weights.

OPTIMIZATION

Our total energy is:

$$E(\mathbf{p}) = w_{\text{change}}\|\mathbf{p} - \mathbf{p_0}\|^2 \quad + \quad w_{\nabla}\|\nabla\mathbf{p} - \nabla\mathbf{p_0}\|^2$$

In order to choose a solution we introduce a set of regulatory energies. <click> We use a least squares approach to express the reliability of interior texels. <click> This however preserves the outside values too exactly, resulting in poor interpolation. Therefore, <click> we add a gradient domain term to allow for global effects resulting from enforcing seam continuity. As this animation plays the gradient weight increases. <play> <click> For smoothness, we introduce an energy term that measures the integrated C^1 discontinuity across seams. This is implemented similarly to the seam energy.

<click> We optimize this energy subject to being in the null space of the seam energy. <click> <read slide> a weighted term for the seam. <click> Where w_seam much larger then all other weights.

OPTIMIZATION

Our total energy is:

$$E(\mathbf{p}) = w_{\text{change}}\|\mathbf{p} - \mathbf{p_0}\|^2 \quad + \quad w_{\nabla}\|\nabla\mathbf{p} - \nabla\mathbf{p_0}\|^2 \quad + \quad w_{C^1}E_{C^1}(\mathbf{p})$$

In order to choose a solution we introduce a set of regulatory energies. <click> We use a least squares approach to express the reliability of interior texels. <click> This however preserves the outside values too exactly, resulting in poor interpolation. Therefore, <click> we add a gradient domain term to allow for global effects resulting from enforcing seam continuity. As this animation plays the gradient weight increases. <play> <click> For smoothness, we introduce an energy term that measures the integrated C^1 discontinuity across seams. This is implemented similarly to the seam energy.

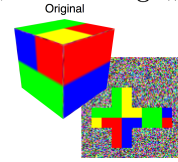
<click> We optimize this energy subject to being in the null space of the seam energy. <click> <read slide> a weighted term for the seam. <click> Where w_seam much larger then all other weights.

OPTIMIZATION

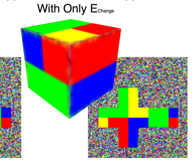
Our total energy is:

$$E(\mathbf{p}) = w_{\text{change}}\|\mathbf{p} - \mathbf{p_0}\|^2 \quad + \quad w_{\nabla}\|\nabla\mathbf{p} - \nabla\mathbf{p_0}\|^2 \quad + \quad w_{C^1}E_{C^1}(\mathbf{p})$$

Original



With Only E_{change}



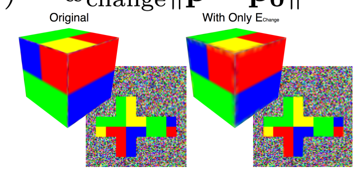
In order to choose a solution we introduce a set of regulatory energies. <click> We use a least squares approach to express the reliability of interior texels. <click> This however preserves the outside values too exactly, resulting in poor interpolation. Therefore, <click> we add a gradient domain term to allow for global effects resulting from enforcing seam continuity. As this animation plays the gradient weight increases. <play> <click> For smoothness, we introduce an energy term that measures the integrated C^1 discontinuity across seams. This is implemented similarly to the seam energy.

<click> We optimize this energy subject to being in the null space of the seam energy. <click> <read slide> a weighted term for the seam. <click> Where w_seam much larger then all other weights.

OPTIMIZATION

Our total energy is:

$$E(\mathbf{p}) = w_{\text{change}}\|\mathbf{p}-\mathbf{p_0}\|^2 \quad + \quad w_{\nabla}\|\nabla\mathbf{p}-\nabla\mathbf{p_0}\|^2 \quad + \quad w_{C^1}E_{C^1}(\mathbf{p})$$



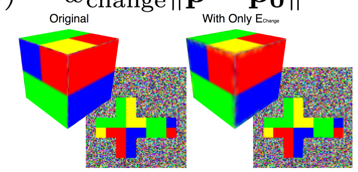
In order to choose a solution we introduce a set of regulatory energies. <click> We use a least squares approach to express the reliability of interior texels. <click> This however preserves the outside values too exactly, resulting in poor interpolation. Therefore, <click> we add a gradient domain term to allow for global effects resulting from enforcing seam continuity. As this animation plays the gradient weight increases. <play> <click> For smoothness, we introduce an energy term that measures the integrated C^1 discontinuity across seams. This is implemented similarly to the seam energy.

<click> We optimize this energy subject to being in the null space of the seam energy. <click> <read slide> a weighted term for the seam. <click> Where w_seam much larger then all other weights.

OPTIMIZATION

Our total energy is:

$$E(\mathbf{p}) = w_{\text{change}}\|\mathbf{p} - \mathbf{p_0}\|^2 \quad + \quad w_{\nabla}\|\nabla\mathbf{p} - \nabla\mathbf{p_0}\|^2 \quad + \quad w_{C^1}E_{C^1}(\mathbf{p})$$



Subject to $E_{\text{seam}}(\mathbf{p}) = \mathbf{p}^{\mathsf{T}}M\mathbf{p} = 0$

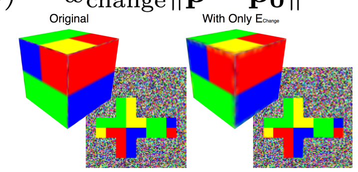
In order to choose a solution we introduce a set of regulatory energies. <click> We use a least squares approach to express the reliability of interior texels. <click> This however preserves the outside values too exactly, resulting in poor interpolation. Therefore, <click> we add a gradient domain term to allow for global effects resulting from enforcing seam continuity. As this animation plays the gradient weight increases. <play> <click> For smoothness, we introduce an energy term that measures the integrated C^1 discontinuity across seams. This is implemented similarly to the seam energy.

<click> We optimize this energy subject to being in the null space of the seam energy. <click> <read slide> a weighted term for the seam. <click> Where w_seam much larger then all other weights.

OPTIMIZATION

Our total energy is:

$$E(\mathbf{p}) = w_{\text{change}}\|\mathbf{p} - \mathbf{p_0}\|^2 \quad + \quad w_{\nabla}\|\nabla\mathbf{p} - \nabla\mathbf{p_0}\|^2 \quad + \quad w_{C^1}E_{C^1}(\mathbf{p})$$



Subject to $E_{\text{seam}}(\mathbf{p}) = \mathbf{p}^{\mathsf{T}}M\mathbf{p} = 0$

We impose the null space constraint via the penalty method by adding:

$$w_{\text{seam}}E_{\text{seam}}(\mathbf{p})$$

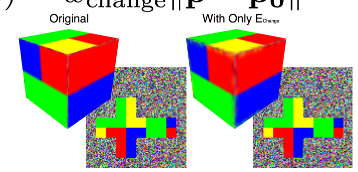
In order to choose a solution we introduce a set of regulatory energies. <click> We use a least squares approach to express the reliability of interior texels. <click> This however preserves the outside values too exactly, resulting in poor interpolation. Therefore, <click> we add a gradient domain term to allow for global effects resulting from enforcing seam continuity. As this animation plays the gradient weight increases. <play> <click> For smoothness, we introduce an energy term that measures the integrated C^1 discontinuity across seams. This is implemented similarly to the seam energy.

<click> We optimize this energy subject to being in the null space of the seam energy. <click> <read slide> a weighted term for the seam. <click> Where w_seam much larger then all other weights.

OPTIMIZATION

Our total energy is:

$$E(\mathbf{p}) = w_{\text{change}}\|\mathbf{p} - \mathbf{p_0}\|^2 \quad + \quad w_{\nabla}\|\nabla\mathbf{p} - \nabla\mathbf{p_0}\|^2 \quad + \quad w_{C^1}E_{C^1}(\mathbf{p})$$



Subject to $E_{\text{seam}}(\mathbf{p}) = \mathbf{p}^{\mathsf{T}}M\mathbf{p} = 0$

We impose the null space constraint via the penalty method by adding:

$$w_{\text{seam}}E_{\text{seam}}(\mathbf{p})$$

with weights

$$w_{\text{seam}} \gg w_{\text{change}}, w_{\nabla}, w_{C^1}$$

In order to choose a solution we introduce a set of regulatory energies. <click> We use a least squares approach to express the reliability of interior texels. <click> This however preserves the outside values too exactly, resulting in poor interpolation. Therefore, <click> we add a gradient domain term to allow for global effects resulting from enforcing seam continuity. As this animation plays the gradient weight increases. <play> <click> For smoothness, we introduce an energy term that measures the integrated C^1 discontinuity across seams. This is implemented similarly to the seam energy.

<click> We optimize this energy subject to being in the null space of the seam energy. <click> <read slide> a weighted term for the seam. <click> Where w_seam much larger then all other weights.

Seam Erasure: Results

Let us see some results of the seam erasure.

COLOR MAP

Before

After

Here the texture values define the surface color. <click> See how the color changes sharply across the seam. <click> After the seam erasure, the edges agree on a smoothed value.

COLOR MAP

Before

After

Here the texture values define the surface color. <click> See how the color changes sharply across the seam. <click> After the seam erasure, the edges agree on a smoothed value.

COLOR MAP

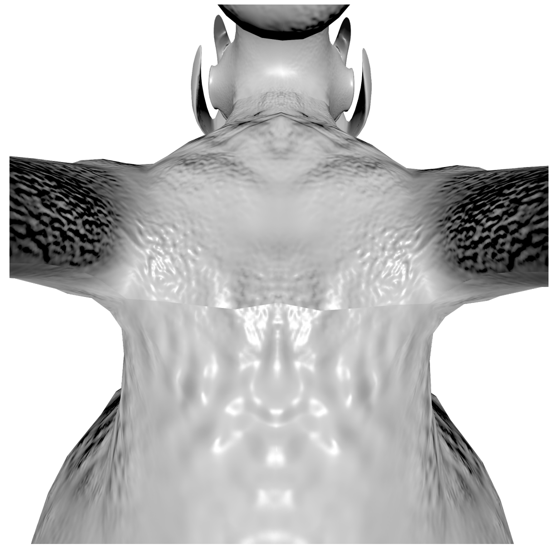
Before After

Here the texture values define the surface color. <click> See how the color changes sharply across the seam. <click> After the seam erasure, the edges agree on a smoothed value.

NORMAL MAP

Before

After

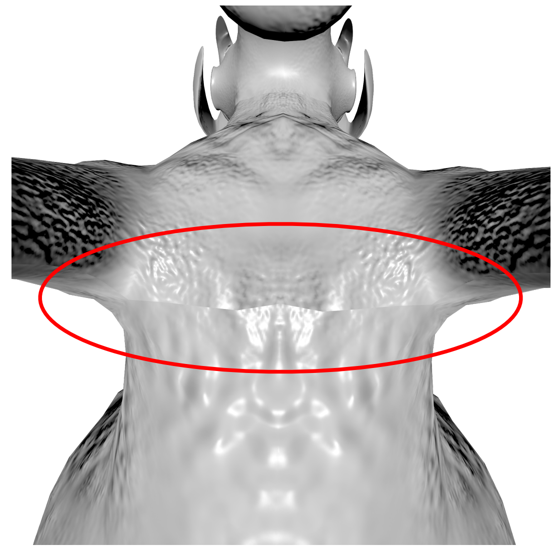


Here the seam is visible when defining surface normals in the texture. <click> Notice how artists place seams in hard to see areas. For example on the belly of this cow model. Hiding seams takes a great deal of effort. <click> Our research stands to fix this process by erasing the seams independent of where it is placed.

NORMAL MAP

Before

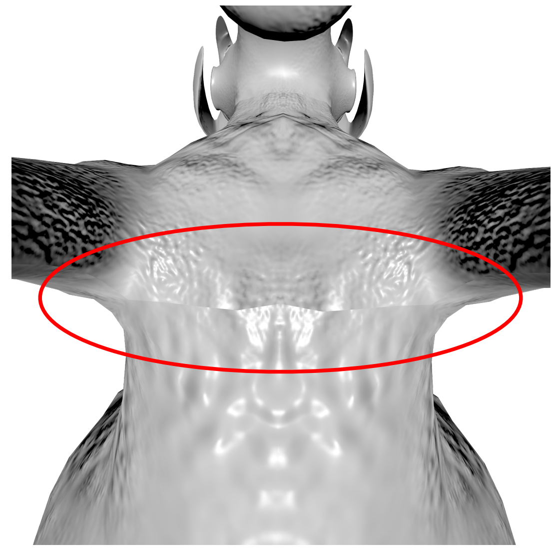
After



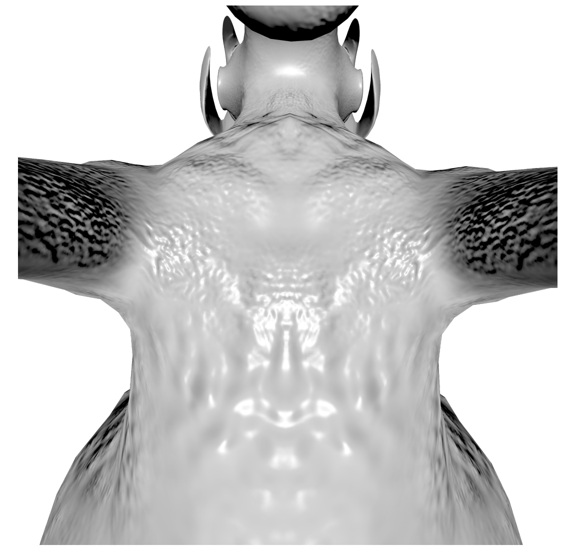
Here the seam is visible when defining surface normals in the texture. <click> Notice how artists place seams in hard to see areas. For example on the belly of this cow model. Hiding seams takes a great deal of effort. <click> Our research stands to fix this process by erasing the seams independent of where it is placed.

NORMAL MAP

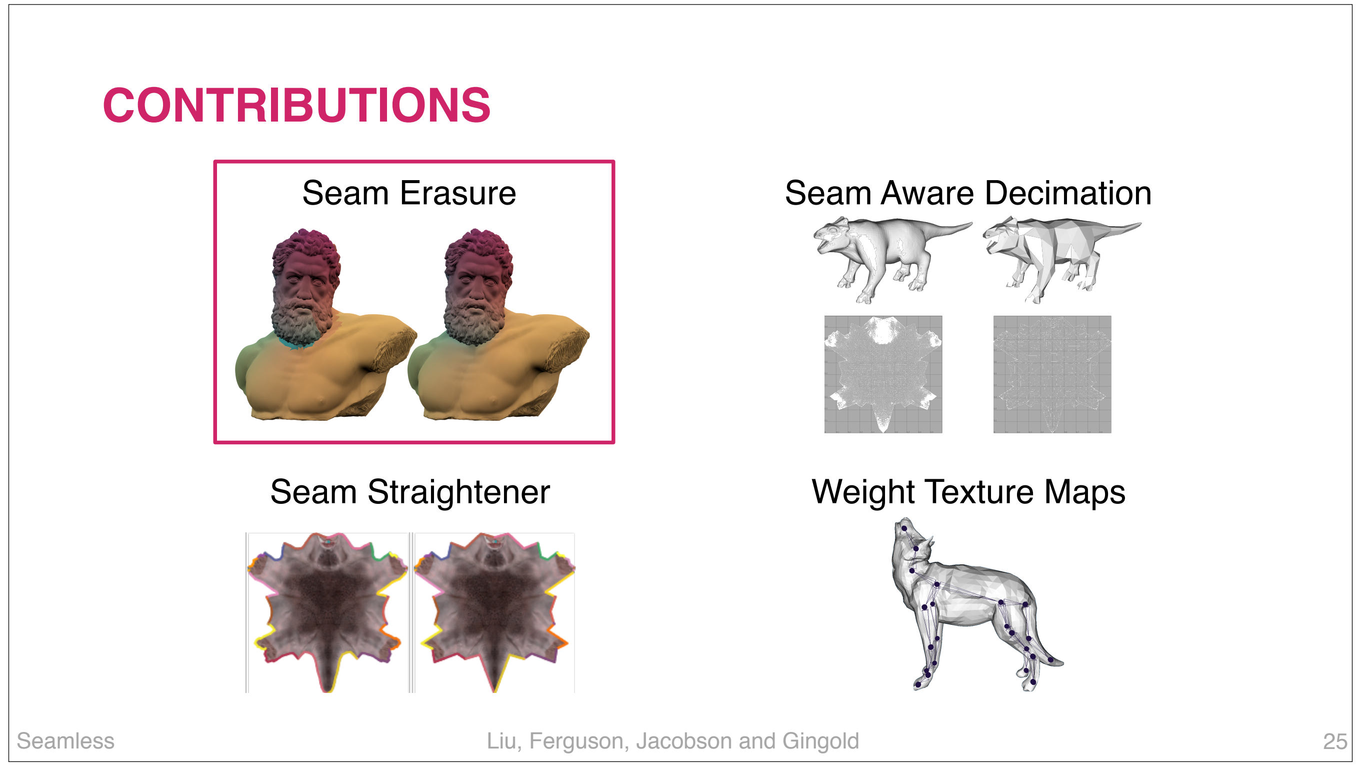
Before



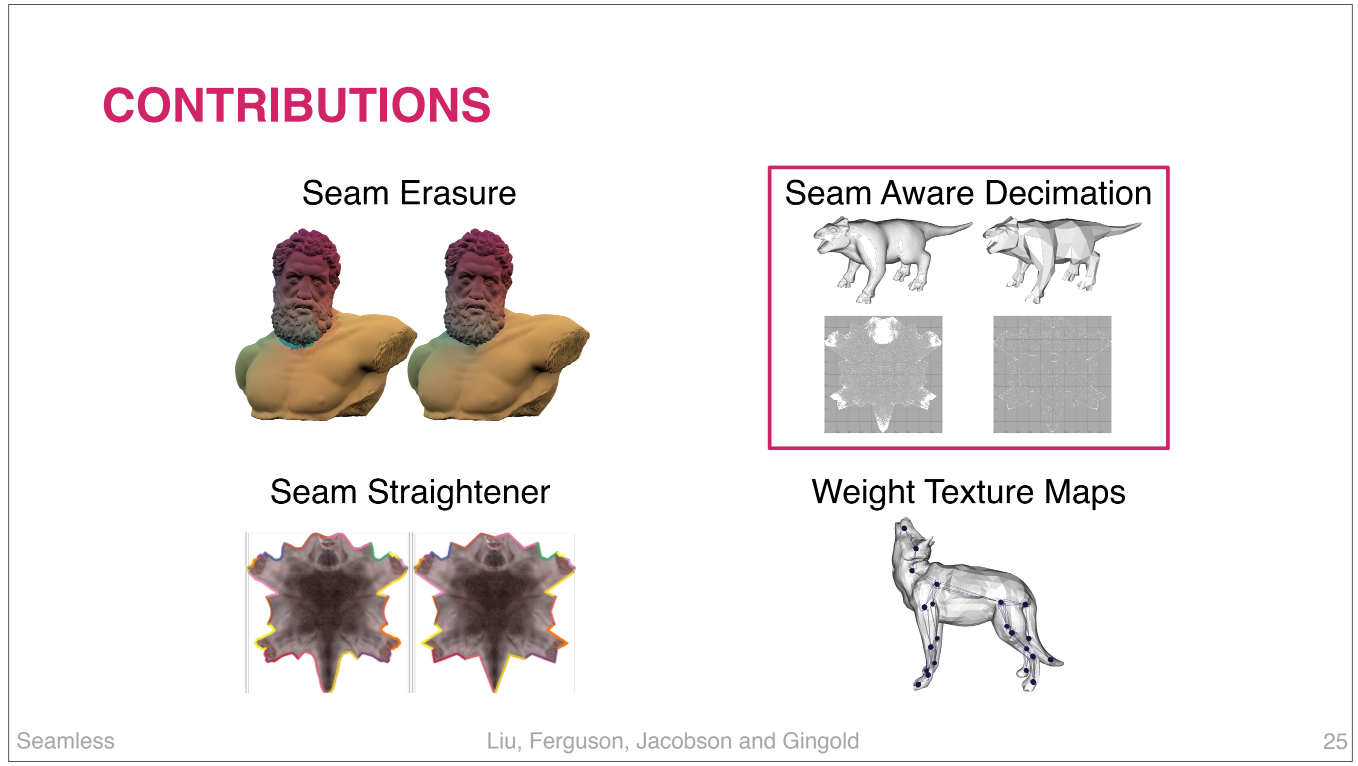
After



Here the seam is visible when defining surface normals in the texture. <click> Notice how artists place seams in hard to see areas. For example on the belly of this cow model. Hiding seams takes a great deal of effort. <click> Our research stands to fix this process by erasing the seams independent of where it is placed.

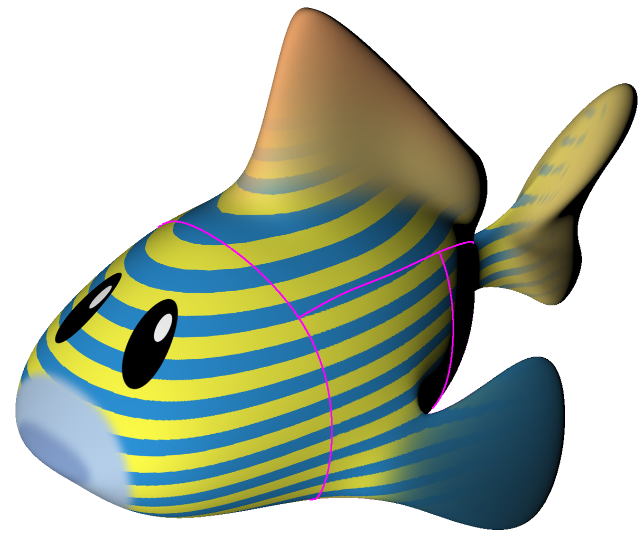


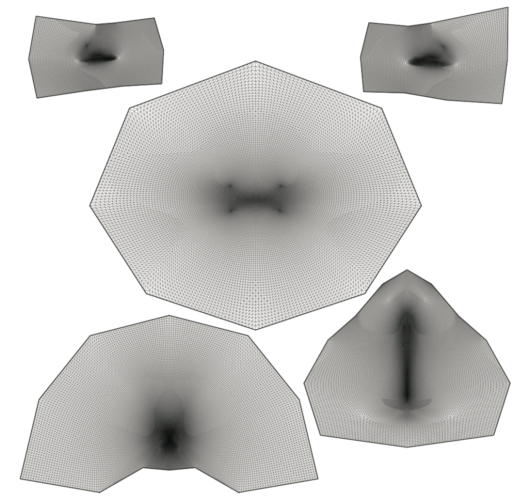
We have now fixed the problem of seams in textures, but if we want to reuse these textures on varying level of detail models, we may introduce seams. All of our work erasing the seam would have to be done over again. <click> To avoid this we implement a seam aware decimation that will prevent seam artifacts arising from decimation.



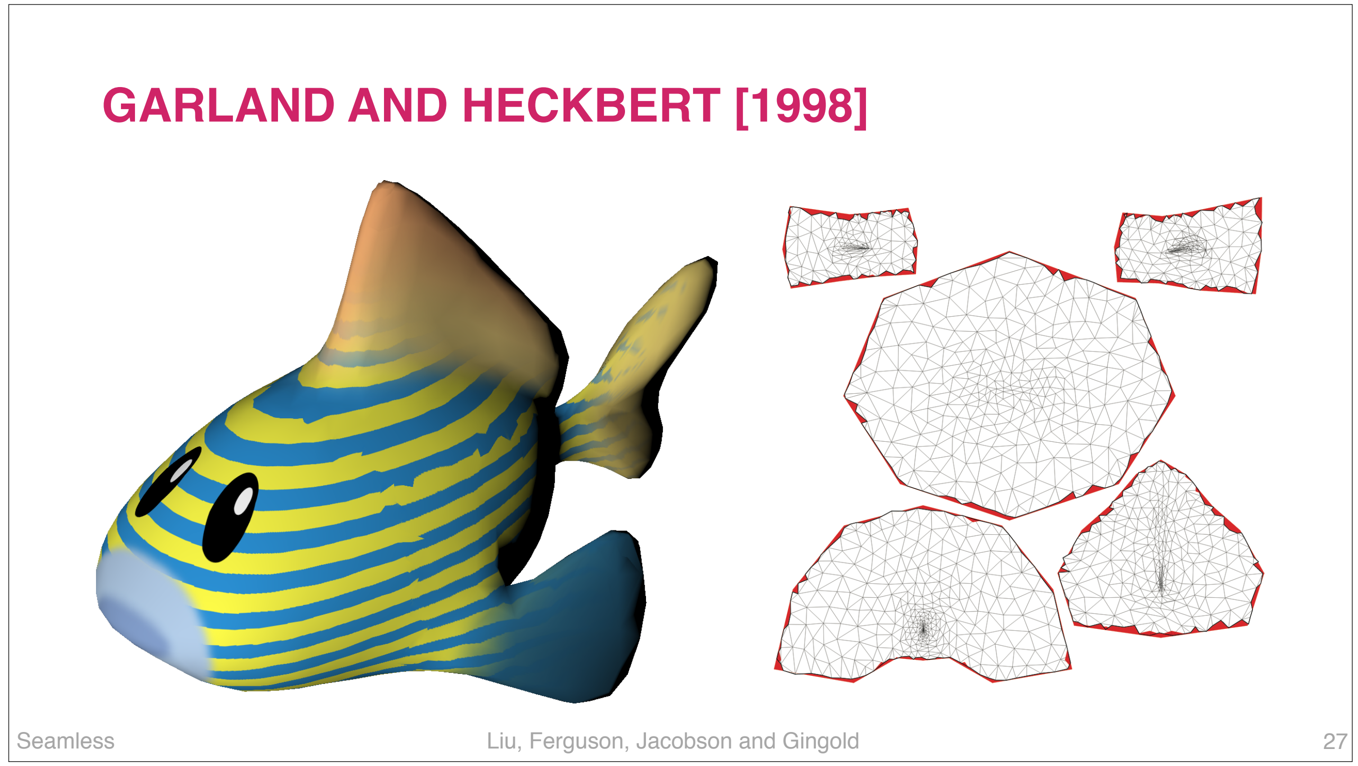
We have now fixed the problem of seams in textures, but if we want to reuse these textures on varying level of detail models, we may introduce seams. All of our work erasing the seam would have to be done over again. <click> To avoid this we implement a seam aware decimation that will prevent seam artifacts arising from decimation.

ORIGINAL MESH



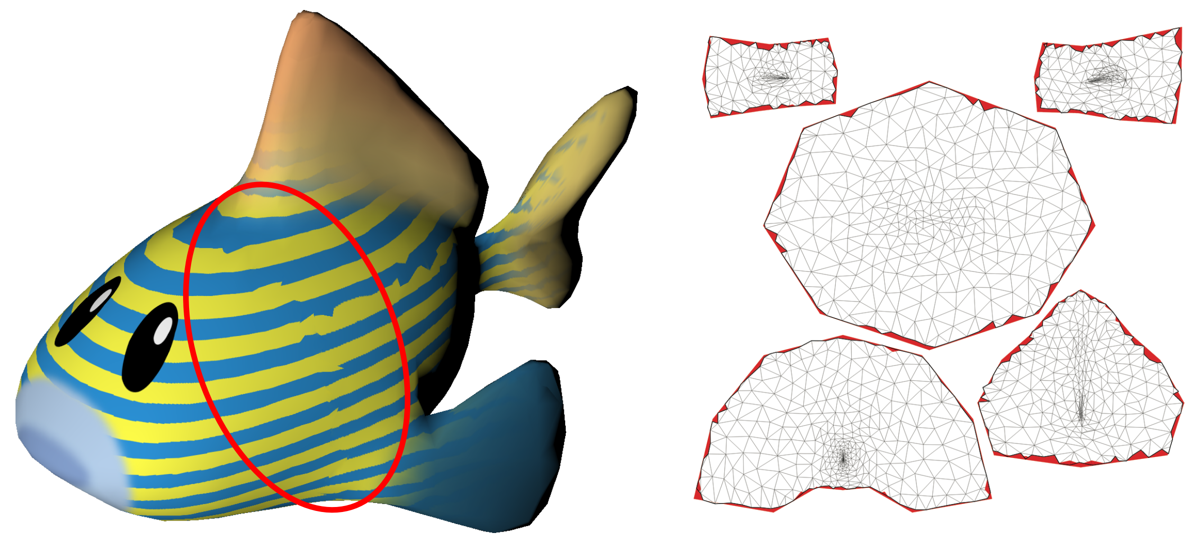


To best understand our approach, let us compare the decimation results of others.
We want to decimate this dense fish model. Seams of the model are drawn in magenta.

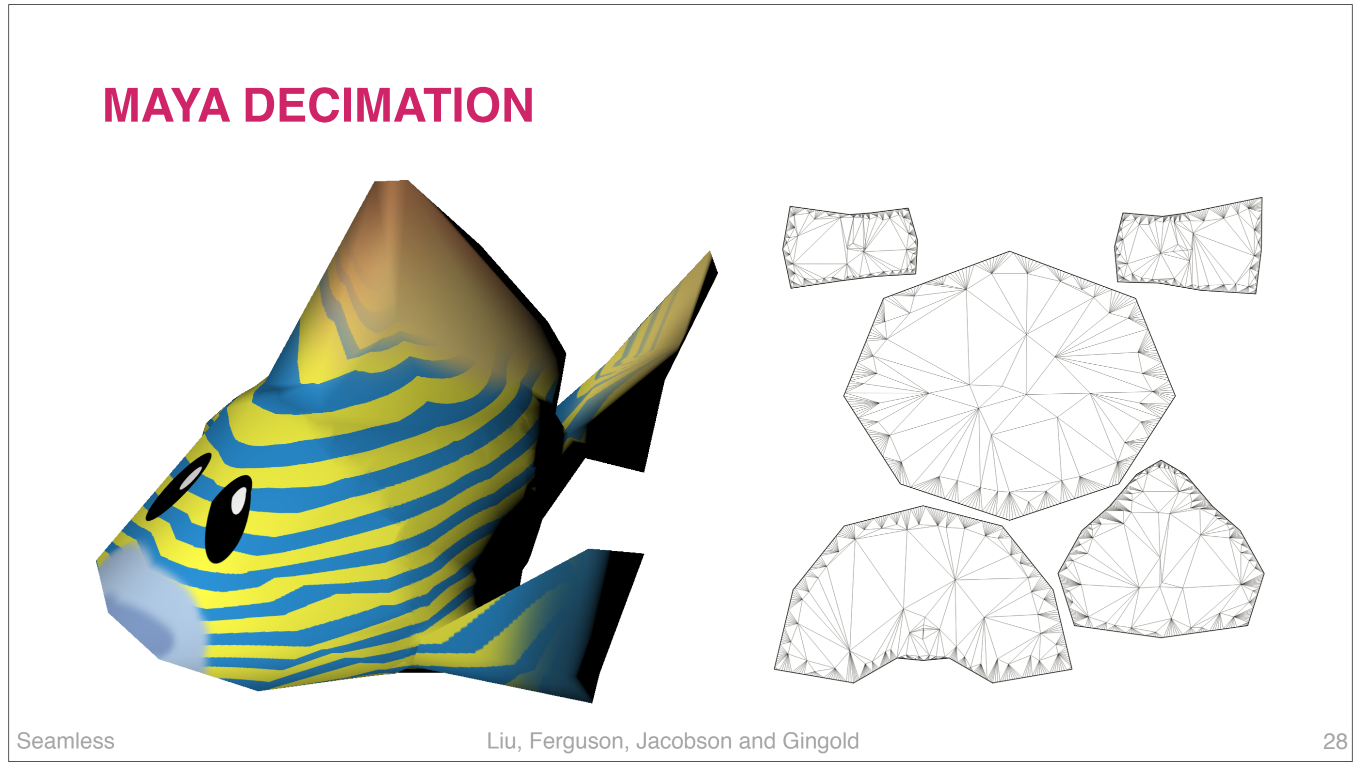


Garland and Heckbert [1998] (implemented by MeshLab [Cignoni et al. 2008]) do not preserve seams precisely, leading to artifacts in the texture. Red areas near seams in the inset parameterization indicate this deviation in the parametric domain.

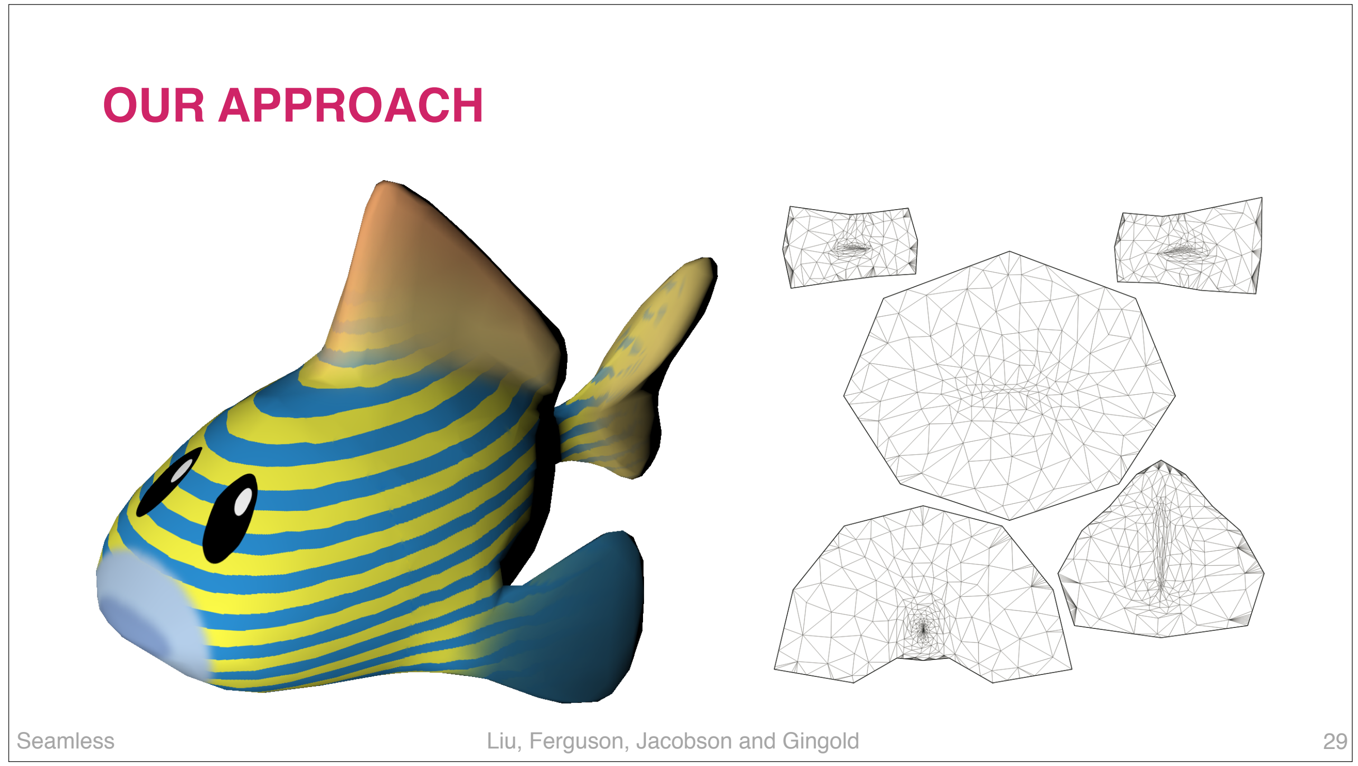
GARLAND AND HECKBERT [1998]



Garland and Heckbert [1998] (implemented by MeshLab [Cignoni et al. 2008]) do not preserve seams precisely, leading to artifacts in the texture. Red areas near seams in the inset parameterization indicate this deviation in the parametric domain.



Maya [2017] prevents decimation of seams entirely, leading to suboptimal allocation of mesh vertices.



Our seamless decimation allows the same texture to be used across all decimation levels—notably along seams. Given a seam-free textures, we describe criteria that must be satisfied to be able to collapse an edge without introducing a discontinuity, and conditions that the new vertex's uv parametric coordinates must satisfy.

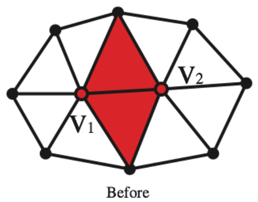
GREEDY EDGE COLLAPSE

Based on Garland and Heckbert [1998]'s n-D Quadric Error Metric

We base our seam aware decimation on Garland and Heckbert's nD Quadric Error Metric. <click> Edges are collapsed <click> greedily according to the quadric error metric. <Read slide> This is algorithm may cause seams discontinuities, so we introduce additional criteria to prevent them.

GREEDY EDGE COLLAPSE

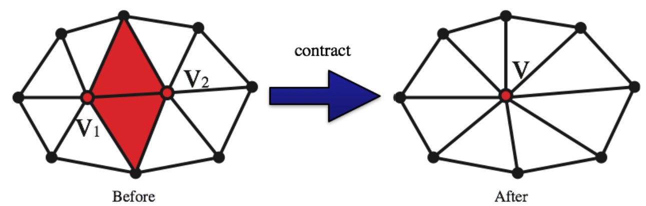
Based on Garland and Heckbert [1998]'s n-D Quadric Error Metric



We base our seam aware decimation on Garland and Heckbert's nD Quadric Error Metric. <click> Edges are collapsed <click> greedily according to the quadric error metric. <Read slide> This is algorithm may cause seams discontinuities, so we introduce additional criteria to prevent them.

GREEDY EDGE COLLAPSE

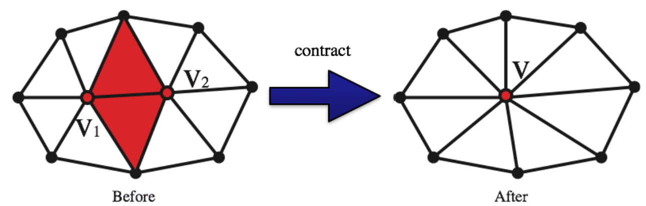
Based on Garland and Heckbert [1998]'s n-D Quadric Error Metric



We base our seam aware decimation on Garland and Heckbert's nD Quadric Error Metric. <click> Edges are collapsed <click> greedily according to the quadric error metric. <Read slide> This is algorithm may cause seams discontinuities, so we introduce additional criteria to prevent them.

GREEDY EDGE COLLAPSE

Based on Garland and Heckbert [1998]'s n-D Quadric Error Metric

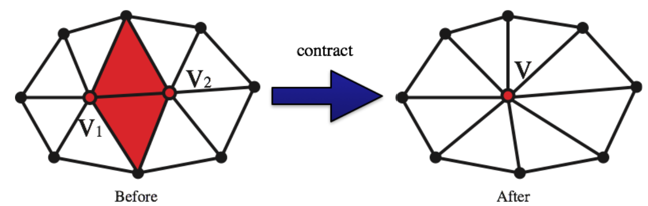


- Each face defines a plane (e.g. 5-D for [x, y, z, u, v])

We base our seam aware decimation on Garland and Heckbert's nD Quadric Error Metric. <click> Edges are collapsed <click> greedily according to the quadric error metric. <Read slide> This is algorithm may cause seams discontinuities, so we introduce additional criteria to prevent them.

GREEDY EDGE COLLAPSE

Based on Garland and Heckbert [1998]'s n-D Quadric Error Metric

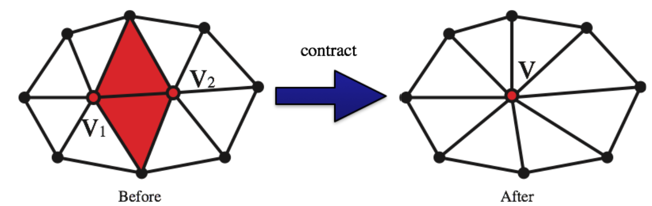


- Each face defines a plane (e.g. 5-D for [x, y, z, u, v])
- Edge error metric = sum of squared distances to face's planes

We base our seam aware decimation on Garland and Heckbert's nD Quadric Error Metric. <click> Edges are collapsed <click> greedily according to the quadric error metric. <Read slide> This is algorithm may cause seams discontinuities, so we introduce additional criteria to prevent them.

GREEDY EDGE COLLAPSE

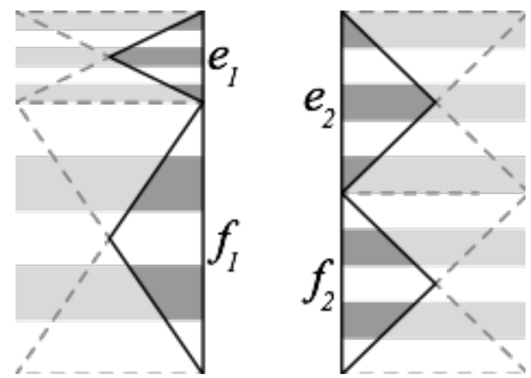
Based on Garland and Heckbert [1998]'s n-D Quadric Error Metric



- Each face defines a plane (e.g. 5-D for [x, y, z, u, v])
- Edge error metric = sum of squared distances to face's planes
- New vertex position minimizes the edge error metric and keeps the edge error metric.

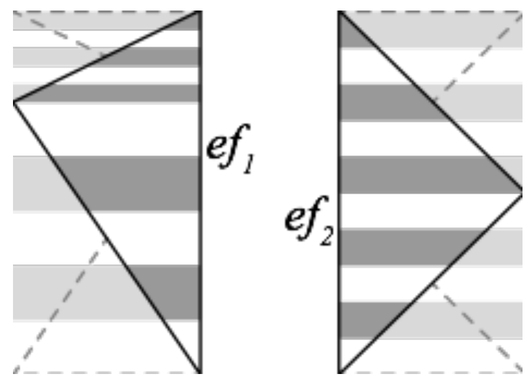
We base our seam aware decimation on Garland and Heckbert's nD Quadric Error Metric. <click> Edges are collapsed <click> greedily according to the quadric error metric. <Read slide> This is algorithm may cause seams discontinuities, so we introduce additional criteria to prevent them.

LENGTH RATIO CRITERIA



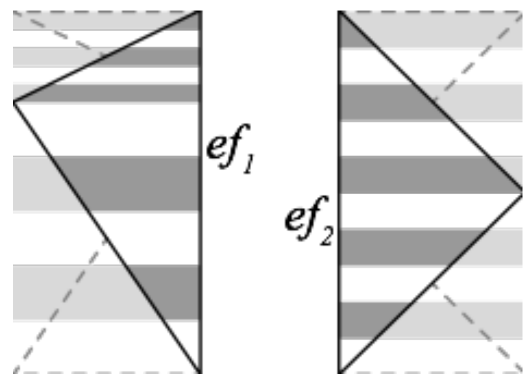
Preserving only the uv shape of seams can still introduce discontinuities due to mismatched sampling. In this example, <read point 1>. <click> To avoid this scenario we implement a length ratio criteria. <click>

LENGTH RATIO CRITERIA



Preserving only the uv shape of seams can still introduce discontinuities due to mismatched sampling. In this example, <read point 1>. <click> To avoid this scenario we implement a length ratio criteria. <click>

LENGTH RATIO CRITERIA

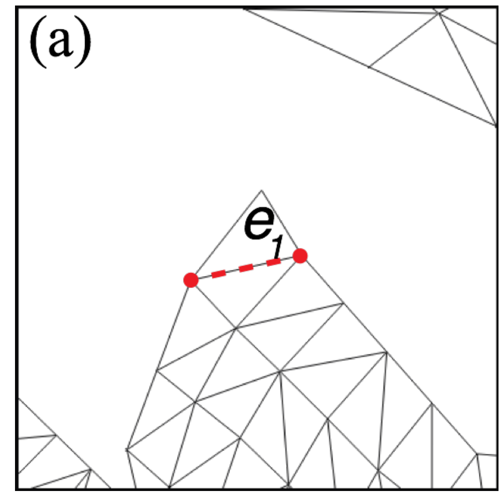


- Merging e_1 f_1 and e_2 f_2 will cause the stripe texture to be misaligned across the seam.
- Length Ratio Criteria:

$$\frac{\|e_1\|}{\|f_1\|} = \frac{\|e_2\|}{\|f_2\|}$$

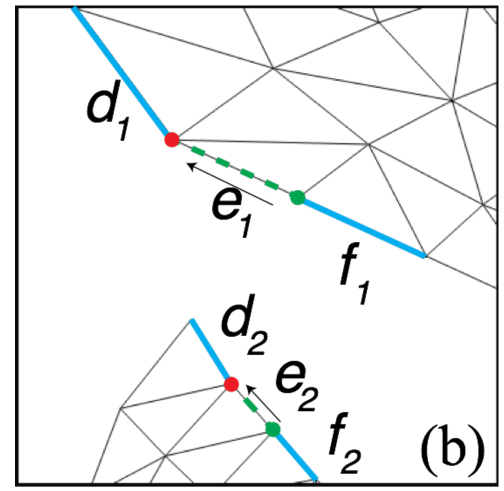
Preserving only the uv shape of seams can still introduce discontinuities due to mismatched sampling. In this example, <read point 1>. <click> To avoid this scenario we implement a length ratio criteria. <click>

LINK CONDITION



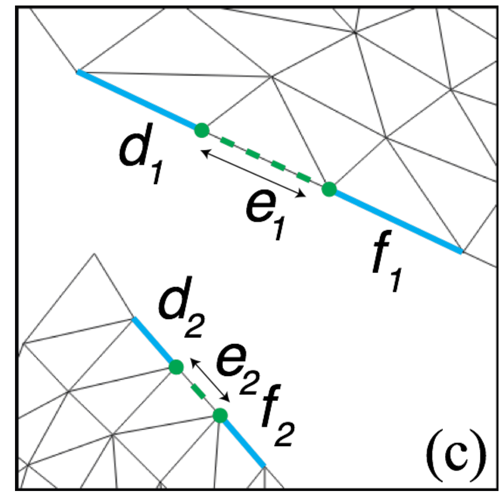
To prevent changes in the topology of the mesh we, implement a link condition. That is, we prevent collapsing a non-seam edges whose endpoints are both on the seam.

TWO UNIFIABLE EDGES



In this example e and f are unifiable because they are collinear and satisfy the length ratio condition. However, e and d cannot be unified because they are not collinear. Therefore, when collapsing e, the only satisfying new vertex placement in uv is for endpoints to move to the location of the red vertices.

THREE UNIFIABLE EDGES

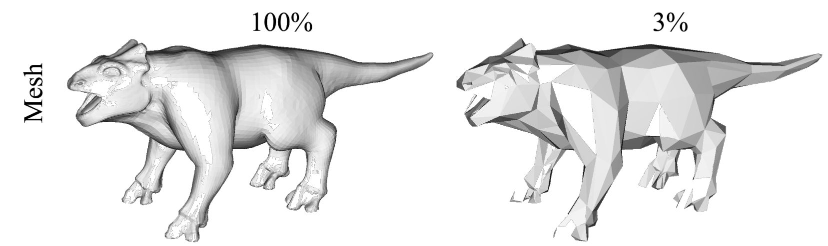


In this example, e, f and e, d are unifiable, so both endpoints of e1 and e2 are free to move. The new vertex placement in uv and xyz will be determined by minimizing the quadric metric subject to collinearity constraints.

Seam Aware Decimation: Results

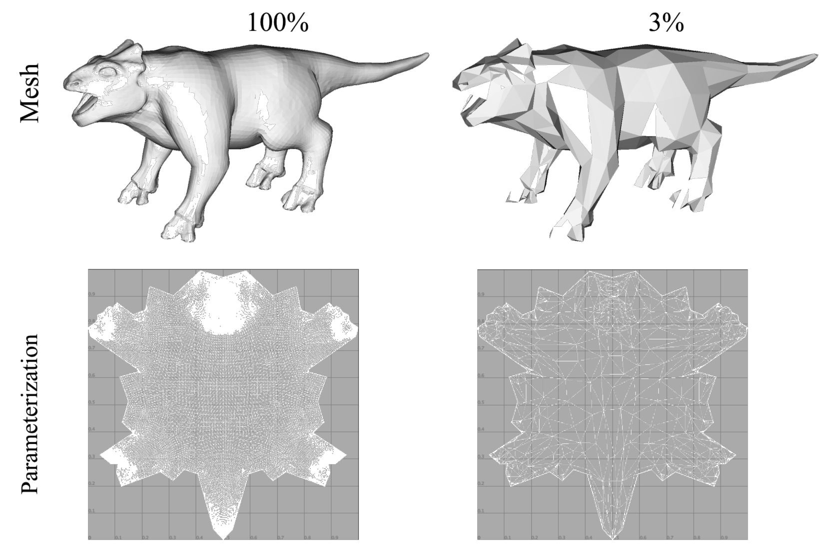
Let us see some results of our decimation algorithm.

DECIMATION RESULT

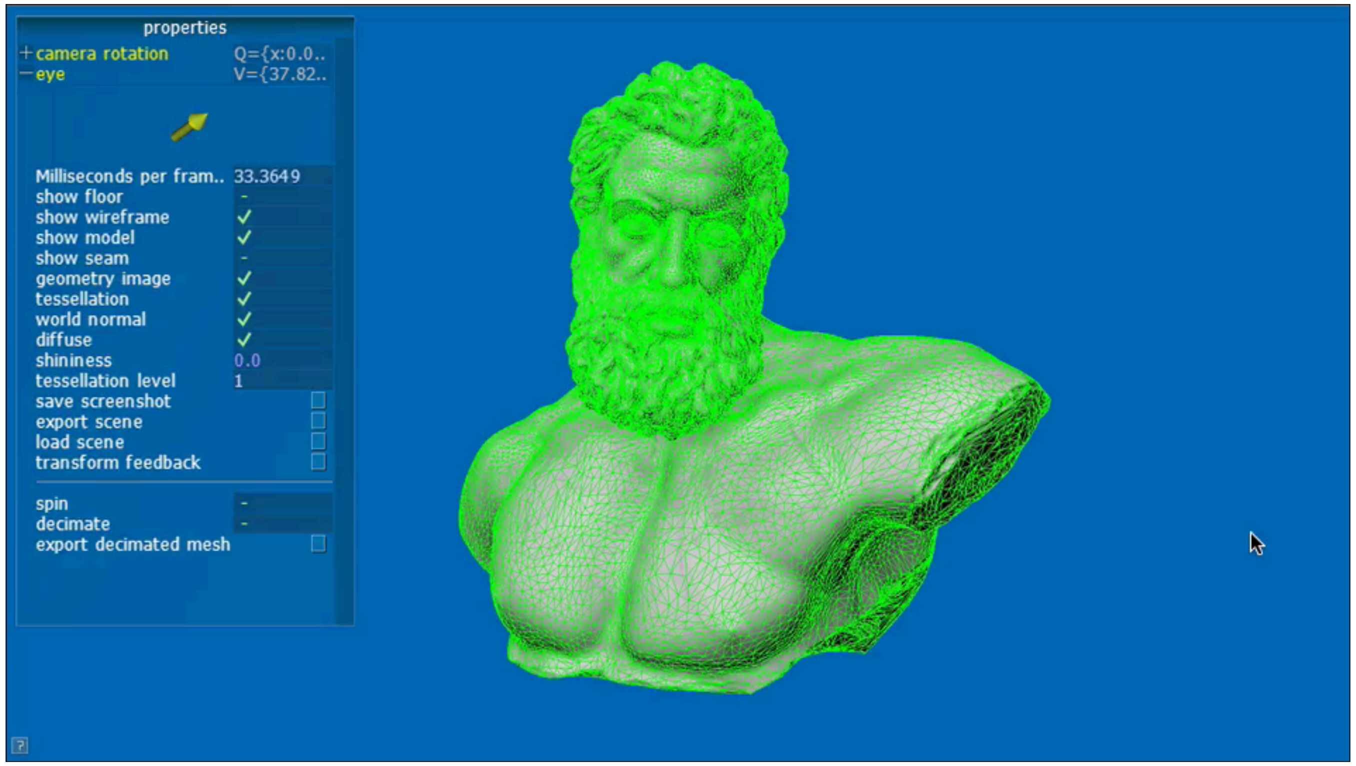


Here, this animal model is heavily decimated to 3% of its original edges.
<click> In the parameterization, the seam boundary shape is preserved.

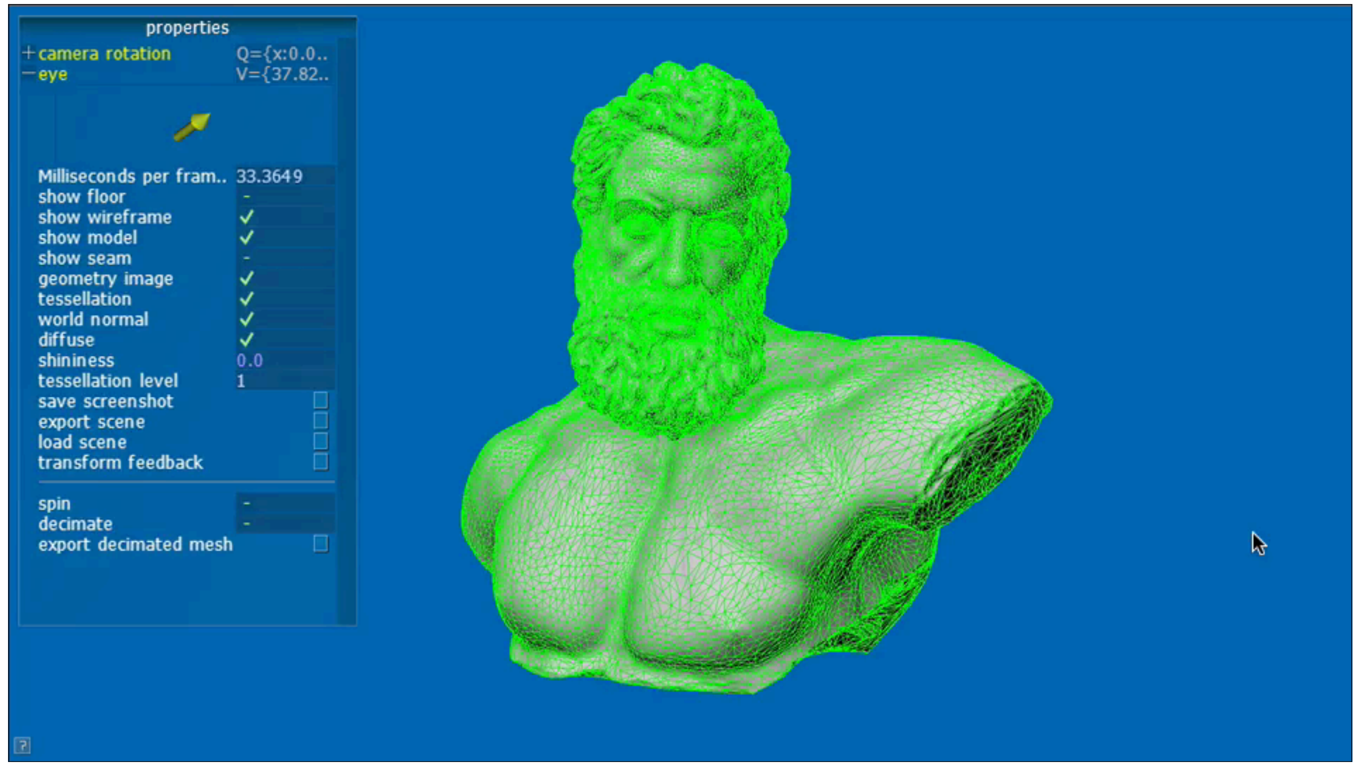
DECIMATION RESULT



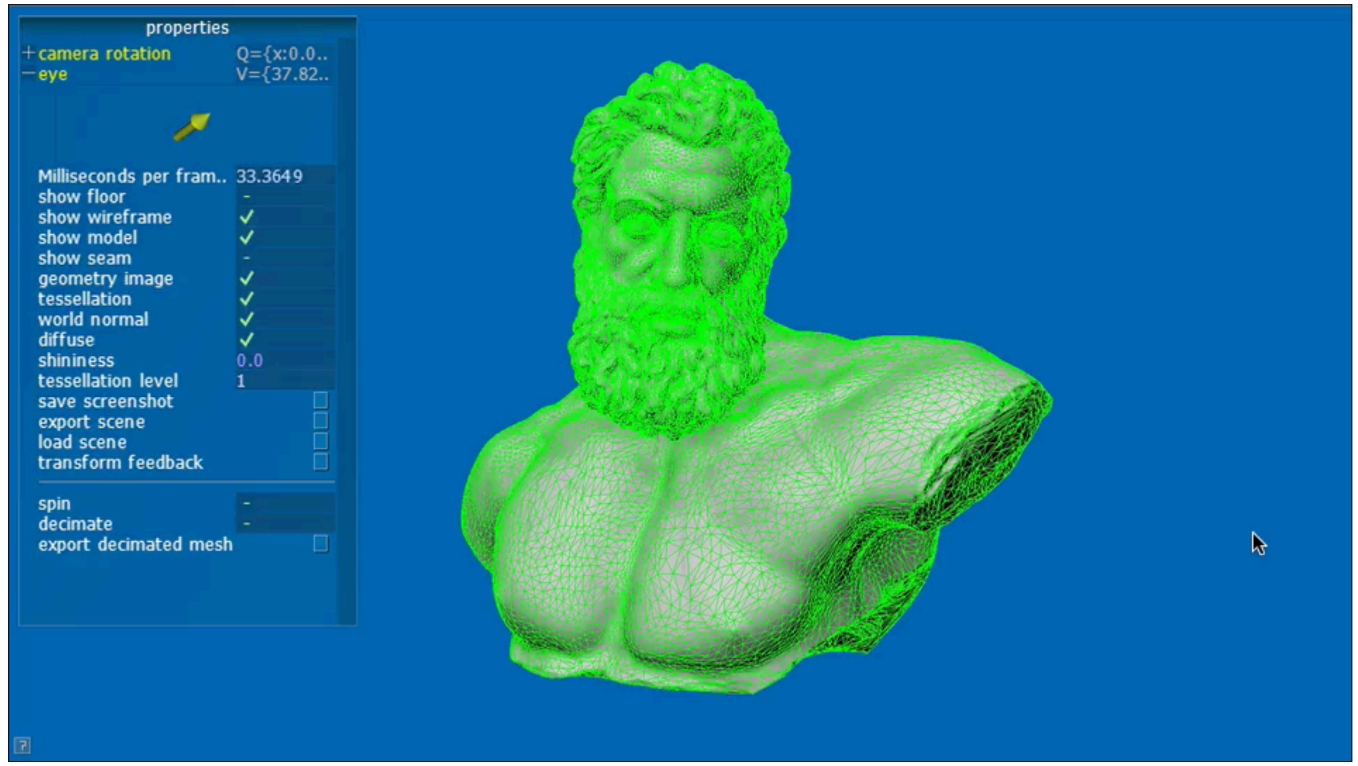
Here, this animal model is heavily decimated to 3% of its original edges.
<click> In the parameterization, the seam boundary shape is preserved.



This Hercules model is decimated using our algorithm. Watch as edges are collapsed <play until tessellation> . We can now adaptively tessellate the surface reusing the same texture. <play>



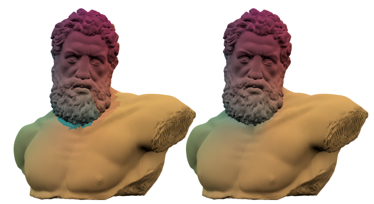
This Hercules model is decimated using our algorithm. Watch as edges are collapsed <play until tessellation> . We can now adaptively tessellate the surface reusing the same texture. <play>



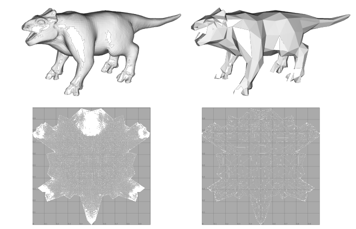
This Hercules model is decimated using our algorithm. Watch as edges are collapsed <play until tessellation> . We can now adaptively tessellate the surface reusing the same texture. <play>

CONTRIBUTIONS

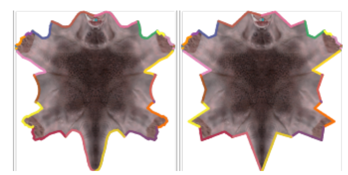
Seam Erasure



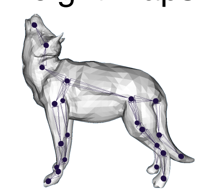
Seam Aware Decimation



Seam Straightener



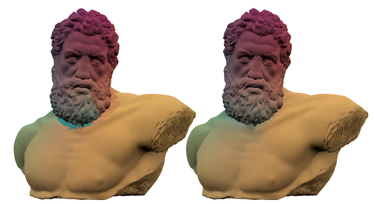
Weight Maps



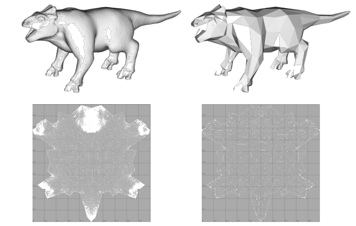
To improve the quality of our decimations we <click> introduce a seam straightening algorithm to increase the number of collinear seam edges.

CONTRIBUTIONS

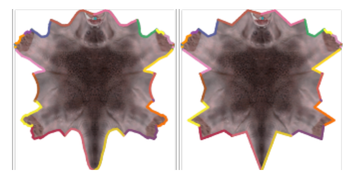
Seam Erasure



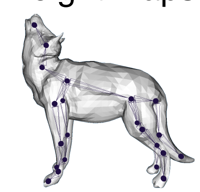
Seam Aware Decimation



Seam Straightener

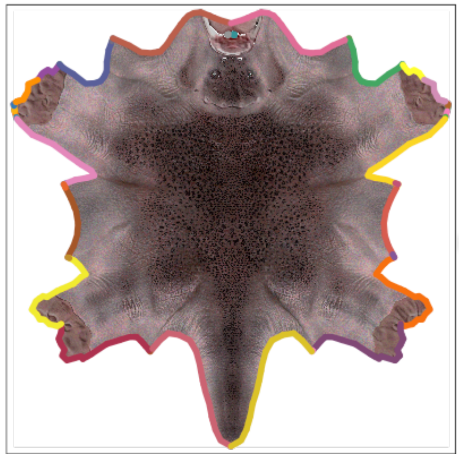


Weight Maps



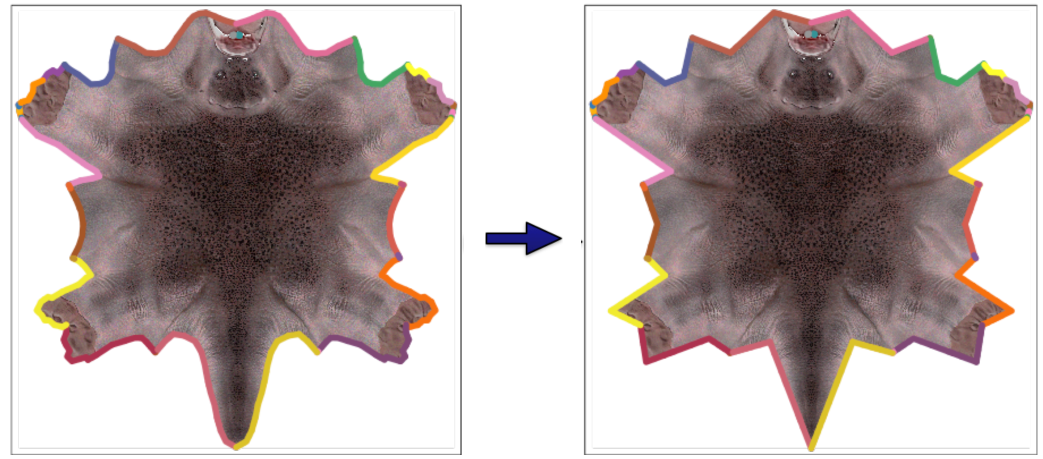
To improve the quality of our decimations we <click> introduce a seam straightening algorithm to increase the number of collinear seam edges.

SEAM STRAIGHTENER



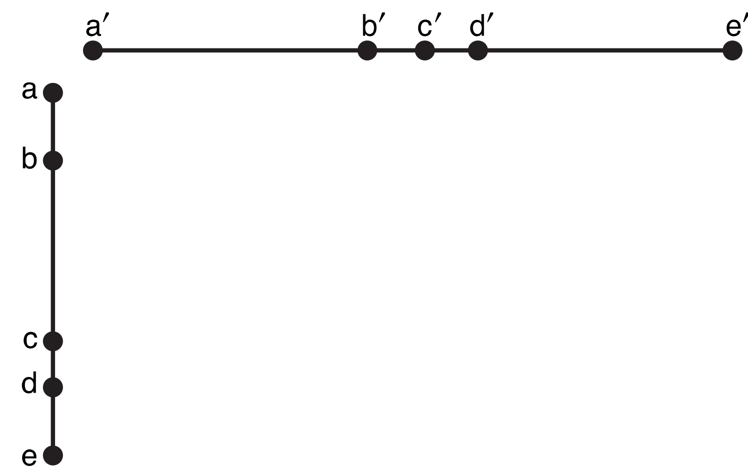
Seam parameterizations of most models do not often meet the collinear edge criteria, and consequently decimation is hindered. We propose preprocessing a given model's uv parameterization to straighten seam edges <click> and therefore increase the effectiveness of our seam-aware mesh decimation.

SEAM STRAIGHTENER



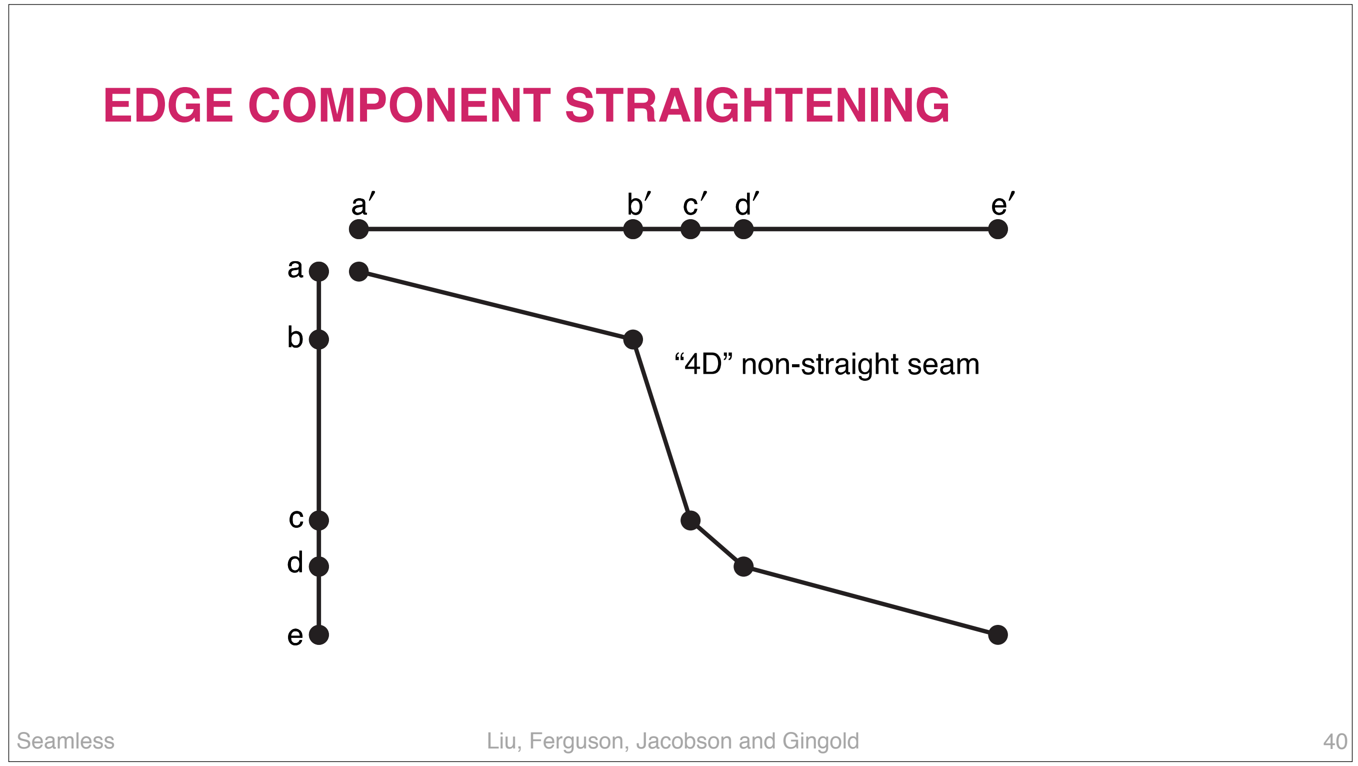
Seam parameterizations of most models do not often meet the collinear edge criteria, and consequently decimation is hindered. We propose preprocessing a given model’s uv parameterization to straighten seam edges <click> and therefore increase the effectiveness of our seam-aware mesh decimation.

EDGE COMPONENT STRAIGHTENING



Our original vertices are illustrated here with mismatched edge length ratios. <click> We treat this as a 4D curve, show here in the 2D, the dimensions are [u, v, u', v']. <click> We straighten the curve using iterative Ramer-Douglas-Peucker method. <click> Because we straighten according to arc-length, the edge-length ratios match.

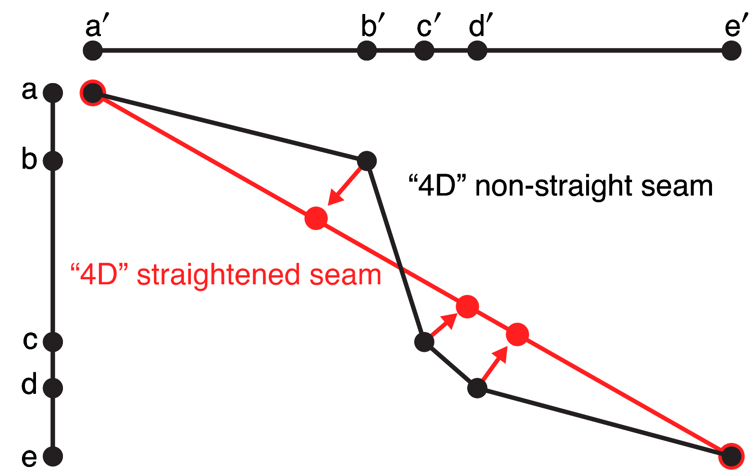
Two halves of a seam with original vertices {a, b, c, d, e } and {a′, b′, c′, d′, e′ } are illustrated as black chains with mismatching edge length ratios (left and top).We straighten the seam, treating it as a 4D curve (illustrated here as a black 2D curve reduced to a red curve). The vertices are repositioned along the curve in 4D, and these define the parametric vertex positions along the original seams (right and bottom). Because their parametrizations agree, edge-length ratios now also agree.



Our original vertices are illustrated here with mismatched edge length ratios. <click> We treat this as a 4D curve, show here in the 2D, the dimensions are [u, v, u', v']. <click> We straighten the curve using iterative Ramer-Douglas-Peucker method. <click> Because we straighten according to arc-length, the edge-length ratios match.

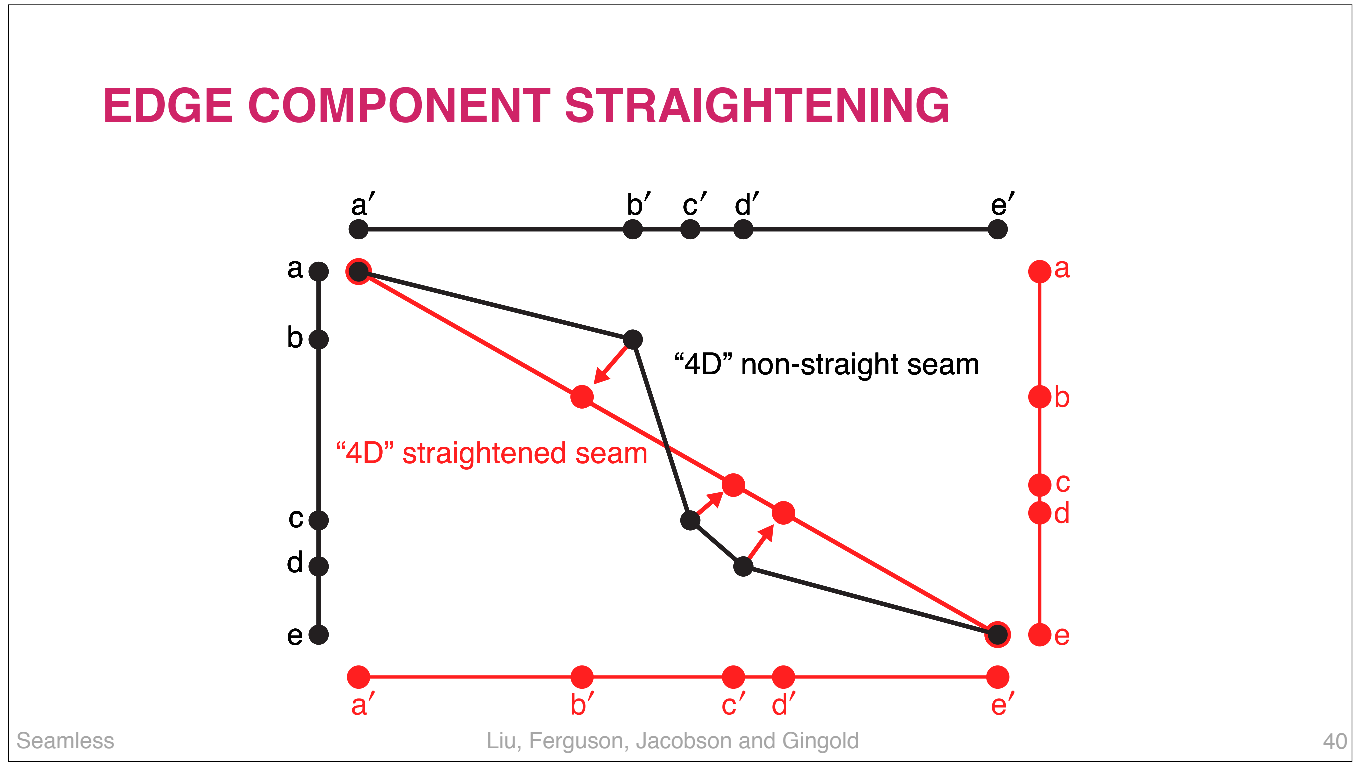
Two halves of a seam with original vertices {a, b, c, d, e } and {a', b', c', d', e' } are illustrated as black chains with mismatching edge length ratios (left and top).We straighten the seam, treating it as a 4D curve (illustrated here as a black 2D curve reduced to a red curve). The vertices are repositioned along the curve in 4D, and these define the parametric vertex positions along the original seams (right and bottom). Because their parametrizations agree, edge-length ratios now also agree.

EDGE COMPONENT STRAIGHTENING



Our original vertices are illustrated here with mismatched edge length ratios. <click> We treat this as a 4D curve, show here in the 2D, the dimensions are [u, v, u', v']. <click> We straighten the curve using iterative Ramer-Douglas-Peucker method. <click> Because we straighten according to arc-length, the edge-length ratios match.

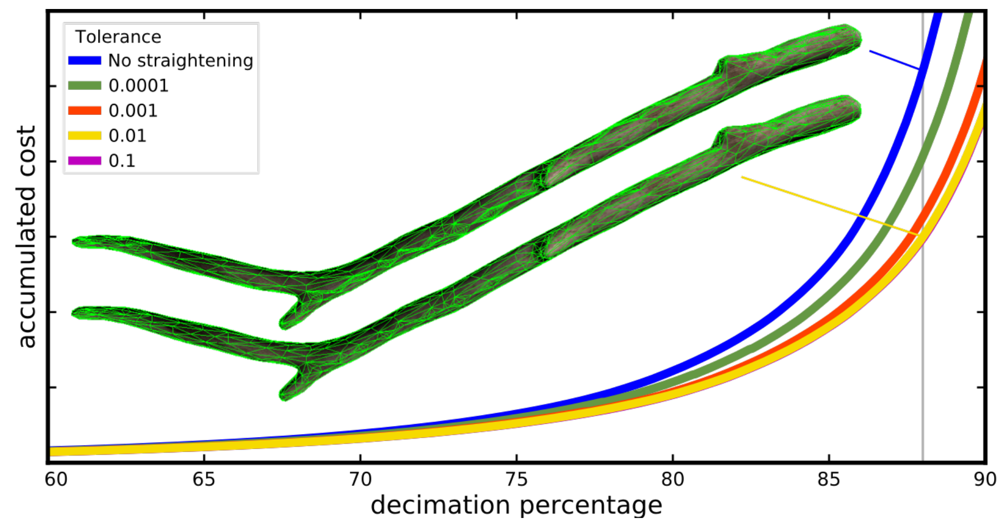
Two halves of a seam with original vertices {a, b, c, d, e } and {a′, b′, c′, d′, e′ } are illustrated as black chains with mismatching edge length ratios (left and top).We straighten the seam, treating it as a 4D curve (illustrated here as a black 2D curve reduced to a red curve). The vertices are repositioned along the curve in 4D, and these define the parametric vertex positions along the original seams (right and bottom). Because their parametrizations agree, edge-length ratios now also agree.



Our original vertices are illustrated here with mismatched edge length ratios. <click> We treat this as a 4D curve, show here in the 2D, the dimensions are [u, v, u', v']. <click> We straighten the curve using iterative Ramer-Douglas-Peucker method. <click> Because we straighten according to arc-length, the edge-length ratios match.

Two halves of a seam with original vertices {a, b, c, d, e } and {a', b', c', d', e' } are illustrated as black chains with mismatching edge length ratios (left and top).We straighten the seam, treating it as a 4D curve (illustrated here as a black 2D curve reduced to a red curve). The vertices are repositioned along the curve in 4D, and these define the parametric vertex positions along the original seams (right and bottom). Because their parametrizations agree, edge-length ratios now also agree.

SEAM STRAIGHTENING RESULTS



The accumulated metric cost of edge collapses to decimate a model (stick) decreases as more straightening is performed. Each curve represents a different straightening tolerance. Straightening increases the number of seam edges that can be collapsed, allowing for a more effective use of the mesh resolution and therefore a lower total error.

UN-COLLAPSIBLE EDGES

Example	# Un-Collapsible Edges Before	# Un-Collapsible Edges After
Chimp	805	171
Hercules	626	290
Animal	369	17
Wolf	374	173

Here we can see the number of un-collapsible edges before straightening. <click> After straightening this number decreases.

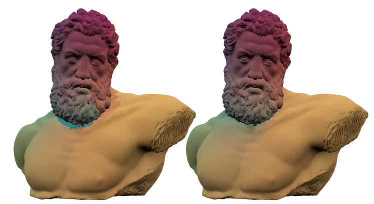
UN-COLLAPSIBLE EDGES

Example	# Un-Collapsible Edges Before	# Un-Collapsible Edges After
Chimp	805	171
Hercules	626	290
Animal	369	17
Wolf	374	173

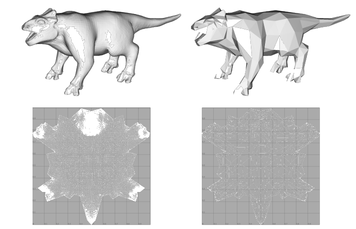
Here we can see the number of un-collapsible edges before straightening. <click> After straightening this number decreases.

CONTRIBUTIONS

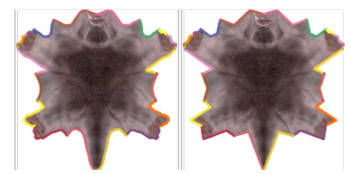
Seam Erasure



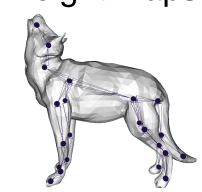
Seam Aware Decimation



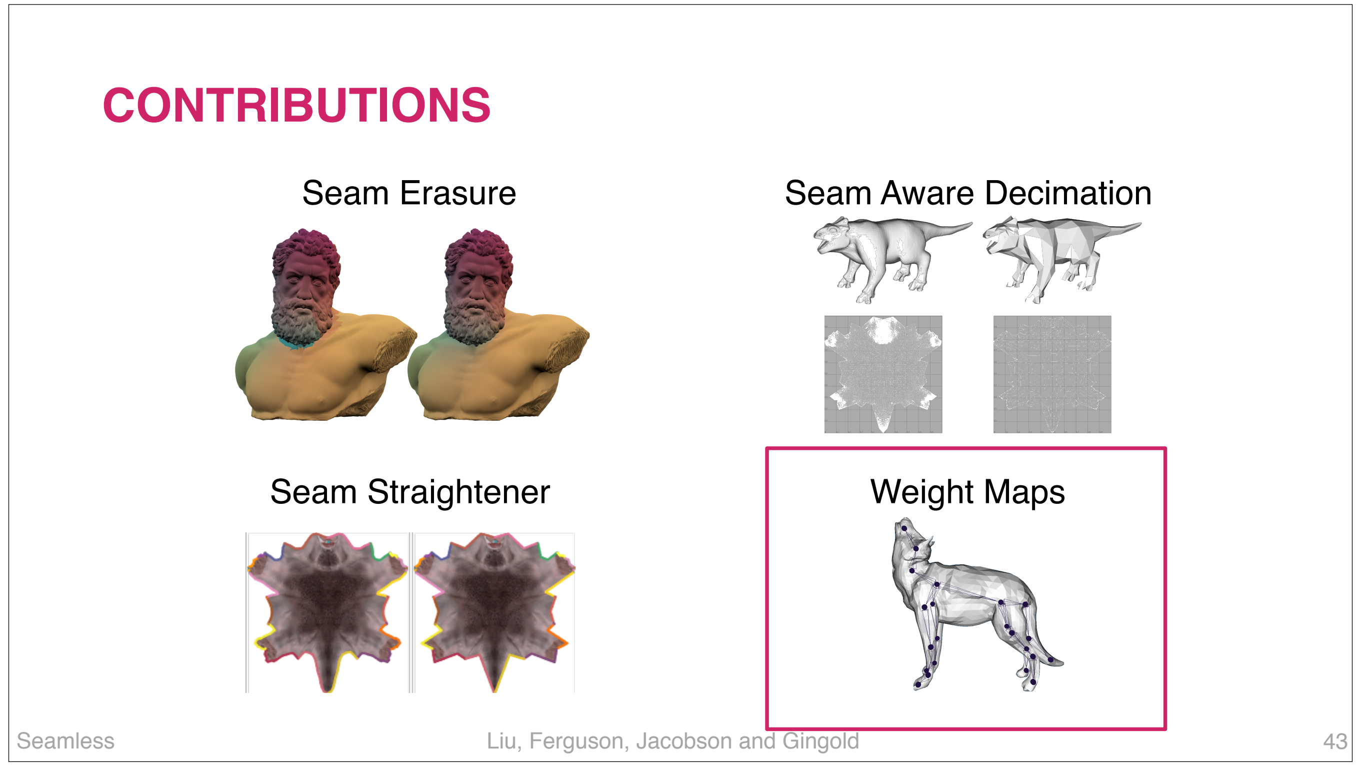
Seam Straightener



Weight Maps

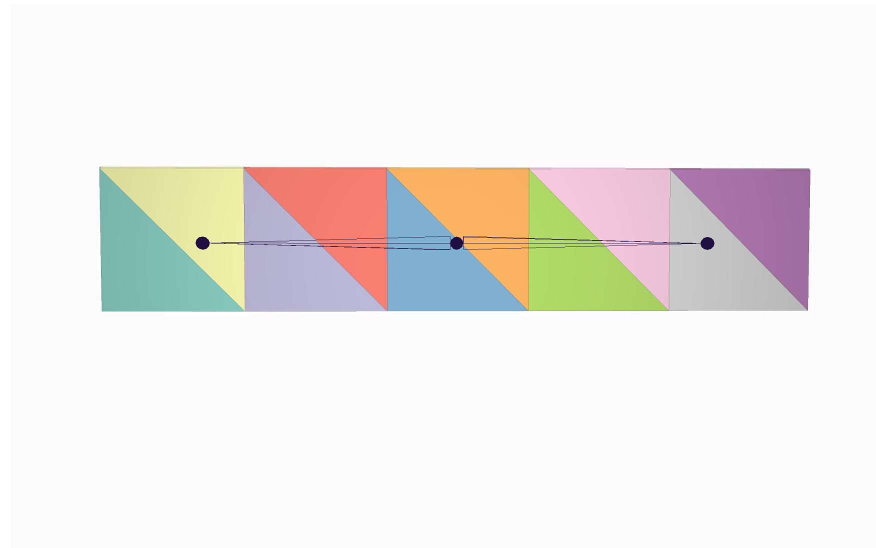


In total, we have erased the seam from texture and designed a decimation algorithm that allows texture reuse. <click> We can combine both of these techniques to aid deformation of 3D models by storing skinning weights in a texture image and sending a minimal number of triangles to the GPU.



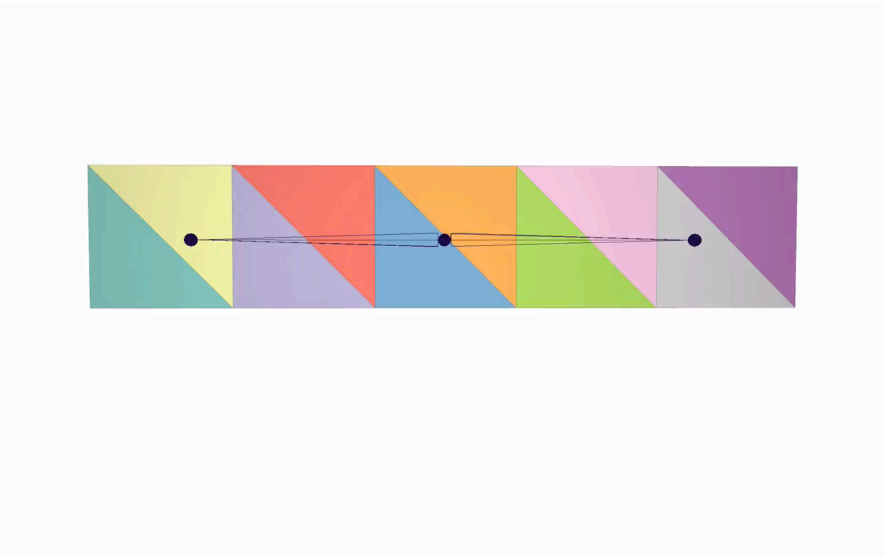
In total, we have erased the seam from texture and designed a decimation algorithm that allows texture reuse. <click> We can combine both of these techniques to aid deformation of 3D models by storing skinning weights in a texture image and sending a minimal number of triangles to the GPU.

PER-VERTEX SKINNING



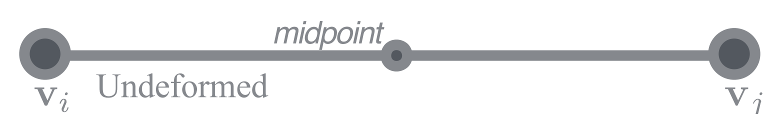
Linear blend skinning weights are stored per-vertex; a mesh is deformed by deforming its vertices. As a result, triangles stay triangles and edges between vertices stay straight.

PER-VERTEX SKINNING



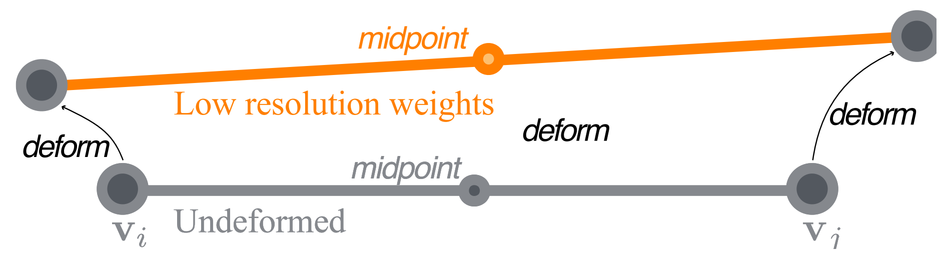
Linear blend skinning weights are stored per-vertex; a mesh is deformed by deforming its vertices. As a result, triangles stay triangles and edges between vertices stay straight.

PER-VERTEX SKINNING



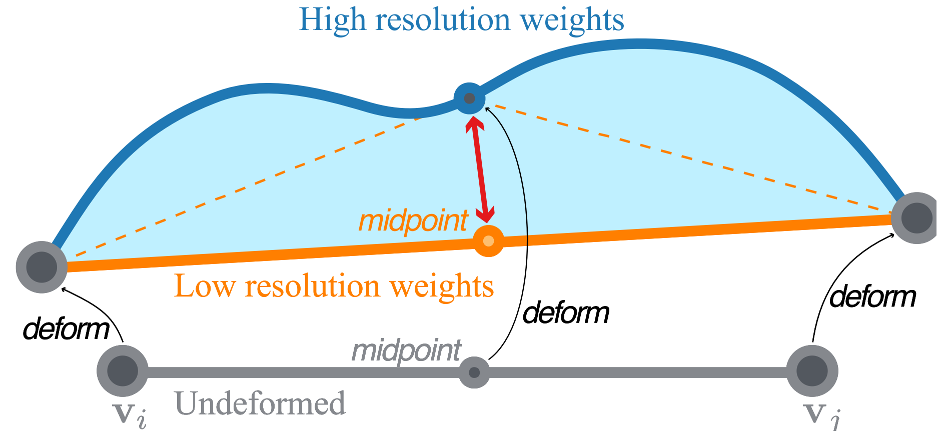
However, the underlying deformation function has a natural definition throughout the mesh with arbitrary weights everywhere <click> (we store these high resolution weights as weight map textures).

PER-VERTEX SKINNING



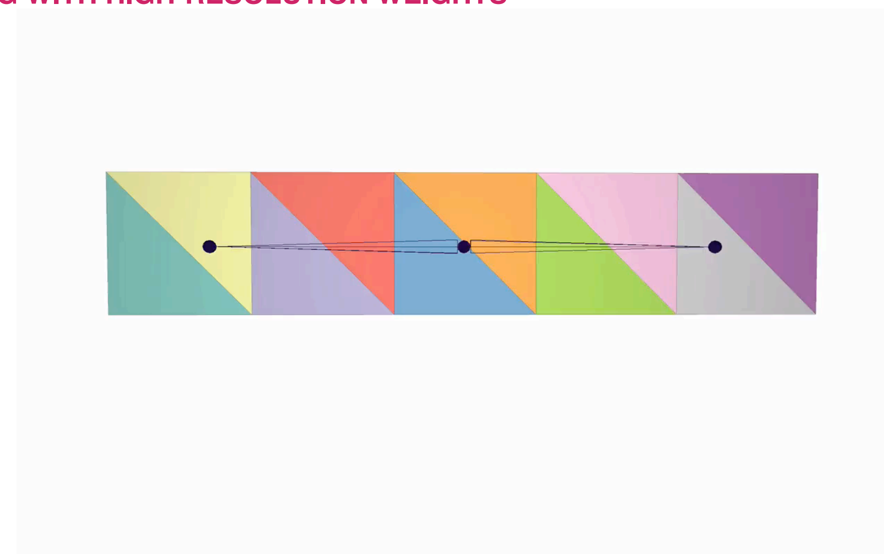
However, the underlying deformation function has a natural definition throughout the mesh with arbitrary weights everywhere <click> (we store these high resolution weights as weight map textures).

PER-VERTEX SKINNING



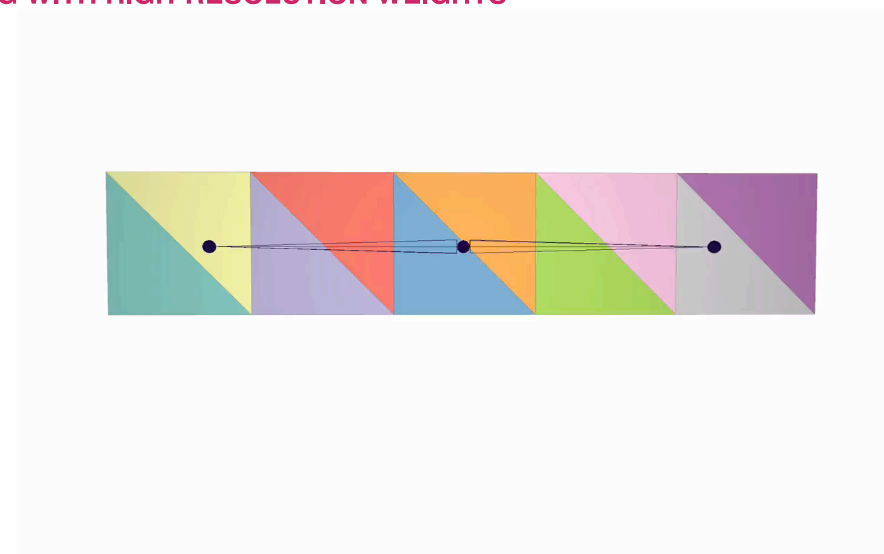
However, the underlying deformation function has a natural definition throughout the mesh with arbitrary weights everywhere <click> (we store these high resolution weights as weight map textures).

SKINNING WITH HIGH-RESOLUTION WEIGHTS



With high resolution weights <click> we get more visually pleasing deformation by rendering with adaptive re-meshing.

SKINNING WITH HIGH-RESOLUTION WEIGHTS

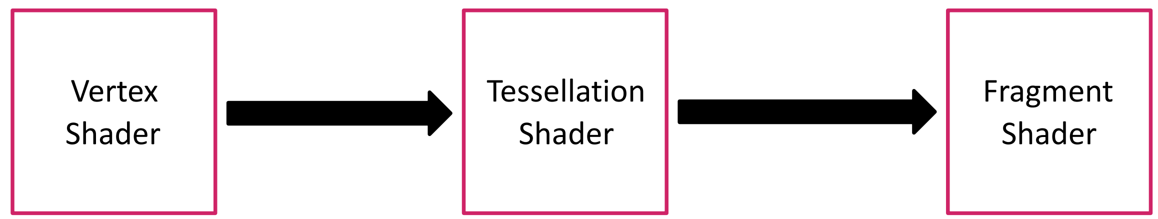


With high resolution weights <click> we get more visually pleasing deformation by rendering with adaptive re-meshing.

MODERN GPU PIPELINE

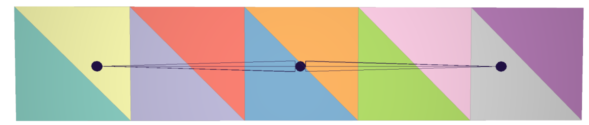
Adaptive re-meshing on the CPU is computationally intensive and impractical for complicated models. Luckily, modern GPU's tessellation shader <click> allows us to implement real-time adaptive subdivision via existing graphics hardware.

MODERN GPU PIPELINE



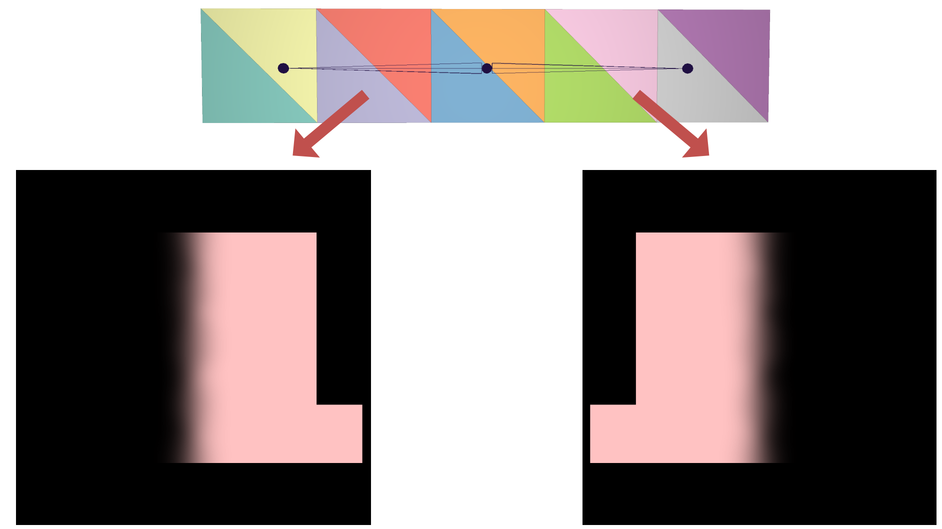
Adaptive re-meshing on the CPU is computationally intensive and impractical for complicated models. Luckily, modern GPU's tessellation shader <click> allows us to implement real-time adaptive subdivision via existing graphics hardware.

WEIGHTS MAP AS TEXTURES



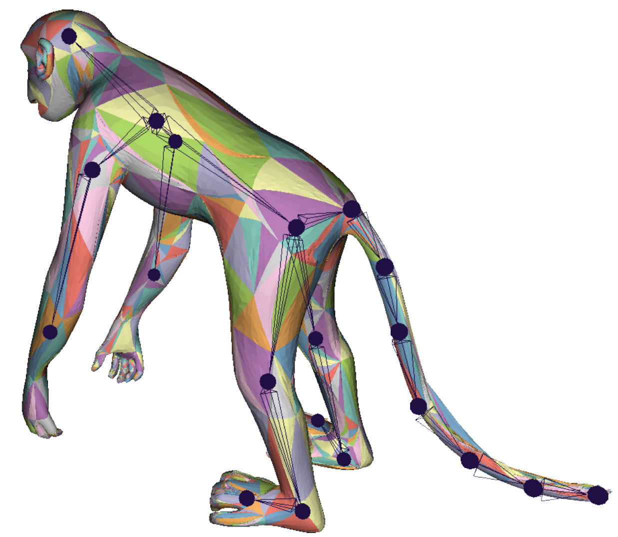
High-resolution skinning weights can be store as textures for access from the GPU. <click> In our experiments, we save each bone’s weights map as a texture image.

WEIGHTS MAP AS TEXTURES



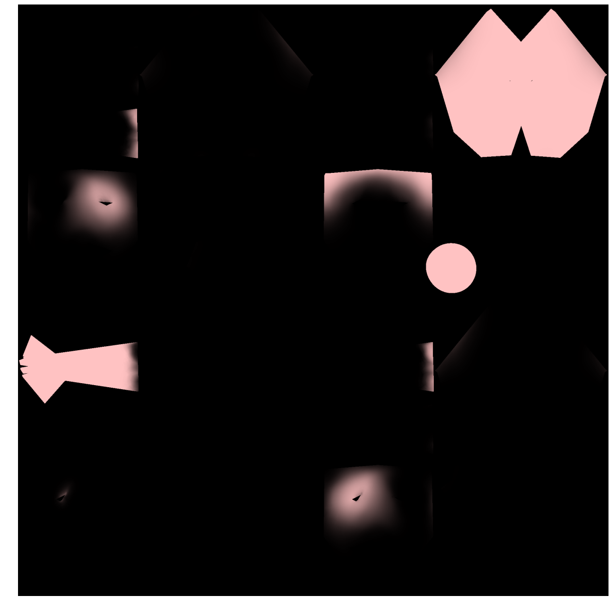
High-resolution skinning weights can be store as textures for access from the GPU. <click> In our experiments, we save each bone’s weights map as a texture image.

SKIN COMPLICATED MODEL WITH WEIGHT TEXTURES



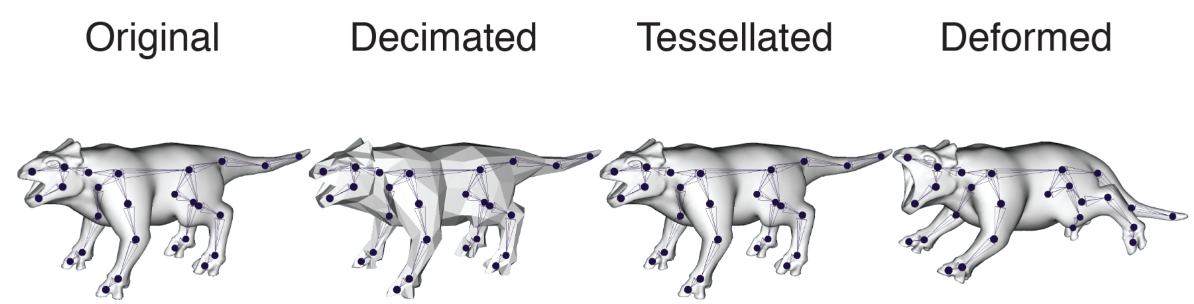
For models with a lot of bones, <click>we can easily pack multiple textures into one to save the resource of texture units, which is often limited by the hardware.

SKIN COMPLICATED MODEL WITH WEIGHT TEXTURES



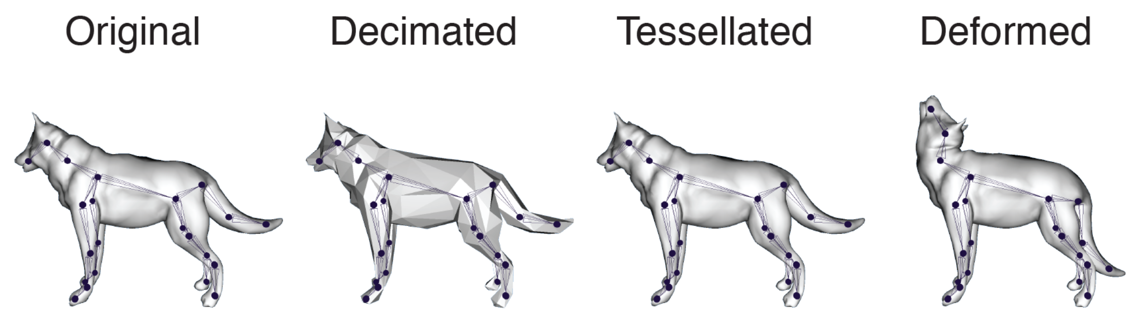
For models with a lot of bones, <click>we can easily pack multiple textures into one to save the resource of texture units, which is often limited by the hardware.

DEFORMATION WITH WEIGHT MAPS



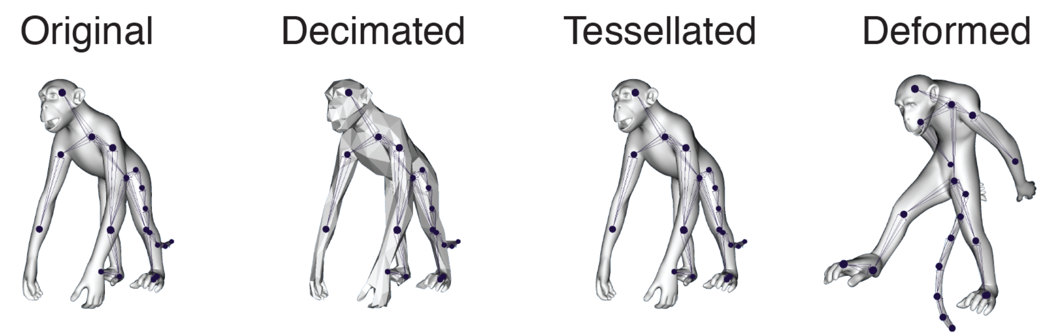
Decimating a detailed input mesh into a coarse mesh and accompanying vector-valued displacement map, normal map, and skin weight map. These allow our coarse model to reproduce the shape and deformability of the detailed mesh.

DEFORMATION WITH WEIGHT MAPS



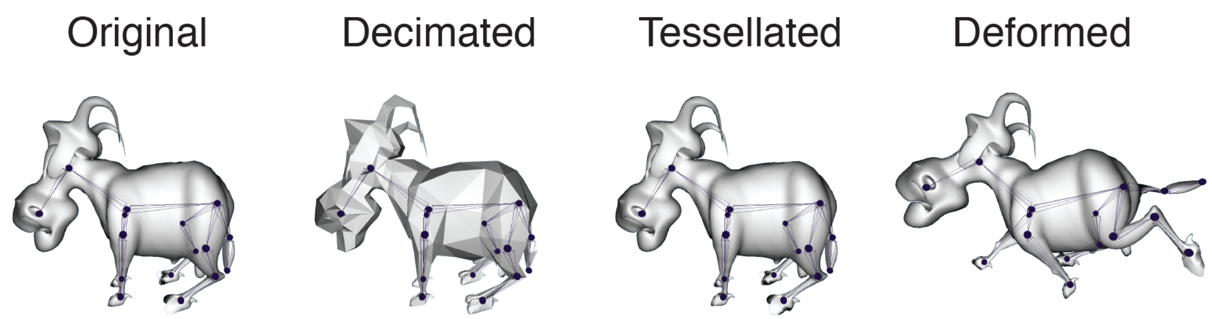
Decimating a detailed input mesh into a coarse mesh and accompanying vector-valued displacement map, normal map, and skin weight map. These allow our coarse model to reproduce the shape and deformability of the detailed mesh.

DEFORMATION WITH WEIGHT MAPS



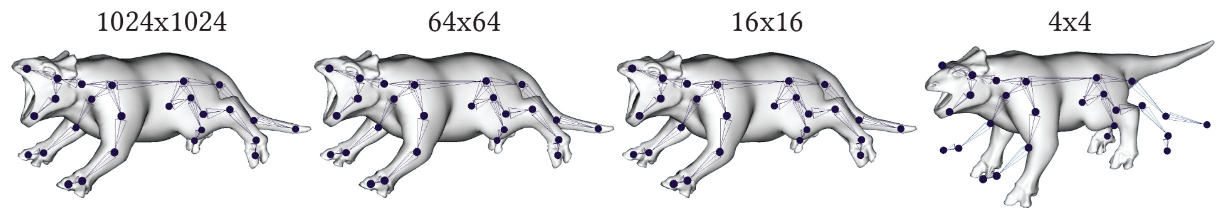
Decimating a detailed input mesh into a coarse mesh and accompanying vector-valued displacement map, normal map, and skin weight map. These allow our coarse model to reproduce the shape and deformability of the detailed mesh.

DEFORMATION WITH WEIGHT MAPS



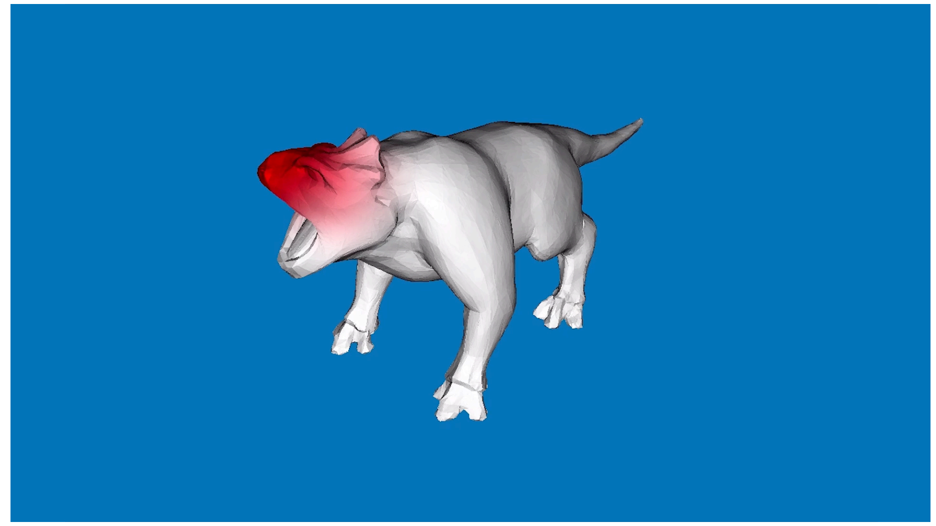
Decimating a detailed input mesh into a coarse mesh and accompanying vector-valued displacement map, normal map, and skin weight map. These allow our coarse model to reproduce the shape and deformability of the detailed mesh.

RESOLUTION OF WEIGHT MAPS

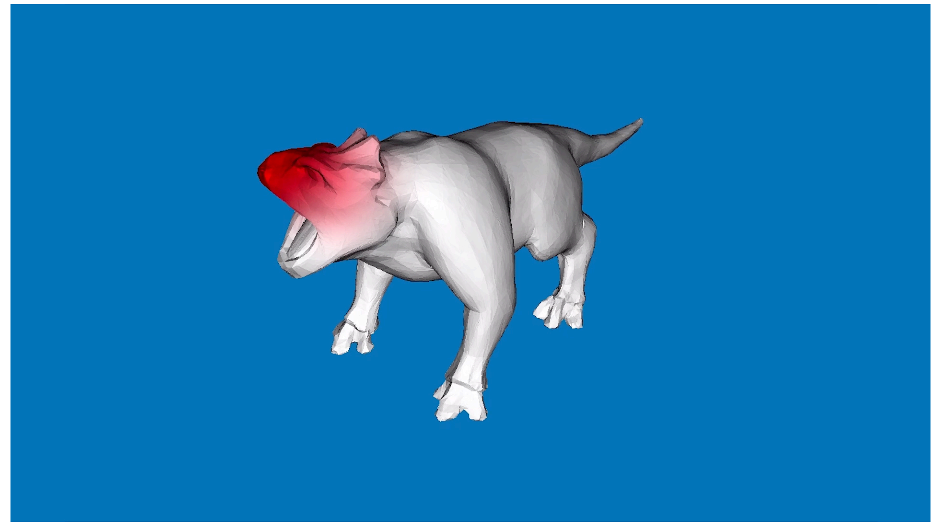


We are able to get complex poses without seams for as small as 16 x 16 textures. Smaller textures result in a constant function in order to preserve the seam continuity.

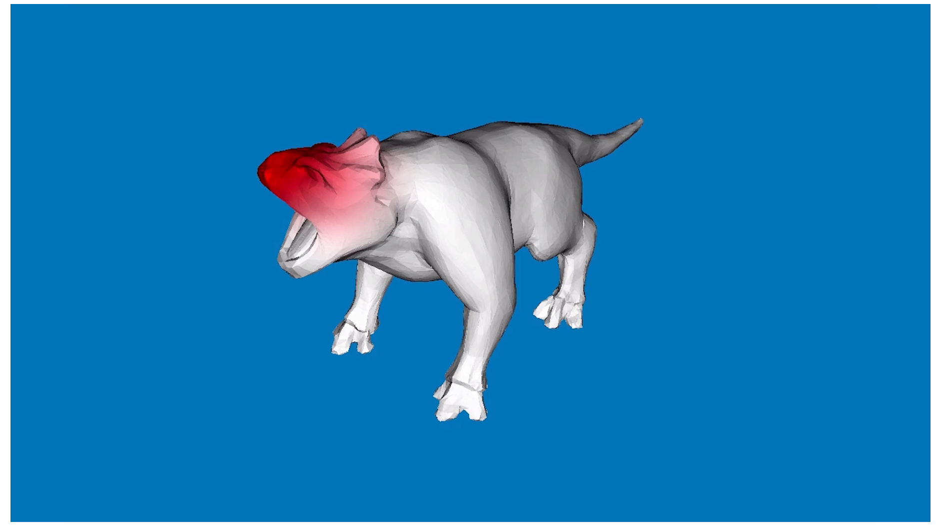
WEIGHT PAINTING



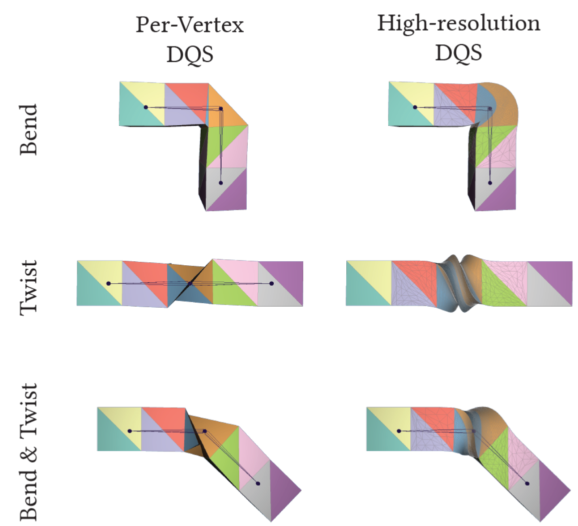
WEIGHT PAINTING



WEIGHT PAINTING



DUAL QUATERNION SKINNING WITH WEIGHT MAPS



Our approach can be extend to other skinning approaches such as Dual quaternion skinning and <click>

FREE-FORM DEFORMATION WITH WEIGHT MAPS

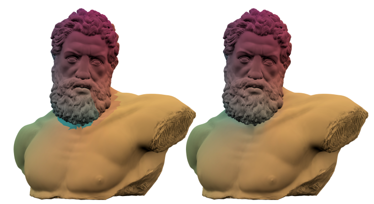
free-form deformation.

CONCLUSION

In total we have <click> erased the seam from our textures in a one time preprocess. This does not change the rendering pipeline and therefore can be easily integrated in current workflows. We also showed <click> how we can decimate a model without introducing seam artifacts. This allows models with the same parametric domain to share texture images. To aid in our decimation <click> we introduced a seam straightening algorithm. Increasing the number of collinear seam edges and therefore the quality of decimation. Lastly, we combined our seamless texture and decimated models <click> to show how skinning weights for deformations can be stored in textures and adaptively used to create weight maps.

CONCLUSION

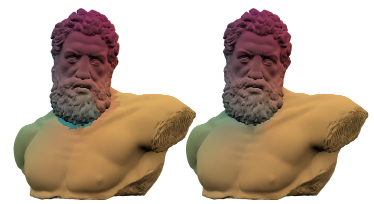
Seam Erasure



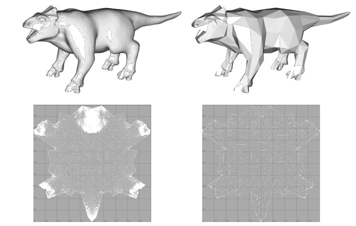
In total we have <click> erased the seam from our textures in a one time preprocess. This does not change the rendering pipeline and therefore can be easily integrated in current workflows. We also showed <click> how we can decimate a model without introducing seam artifacts. This allows models with the same parametric domain to share texture images. To aid in our decimation <click> we introduced a seam straightening algorithm. Increasing the number of collinear seam edges and therefore the quality of decimation. Lastly, we combined our seamless texture and decimated models <click> to show how skinning weights for deformations can be stored in textures and adaptively used to create weight maps.

CONCLUSION

Seam Erasure



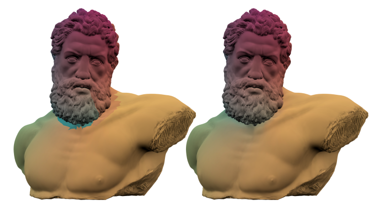
Seam Aware Decimation



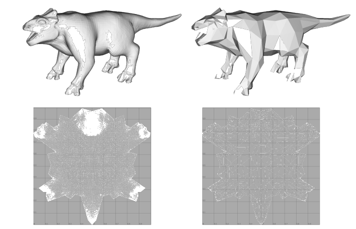
In total we have <click> erased the seam from our textures in a one time preprocess. This does not change the rendering pipeline and therefore can be easily integrated in current workflows. We also showed <click> how we can decimate a model without introducing seam artifacts. This allows models with the same parametric domain to share texture images. To aid in our decimation <click> we introduced a seam straightening algorithm. Increasing the number of collinear seam edges and therefore the quality of decimation. Lastly, we combined our seamless texture and decimated models <click> to show how skinning weights for deformations can be stored in textures and adaptively used to create weight maps.

CONCLUSION

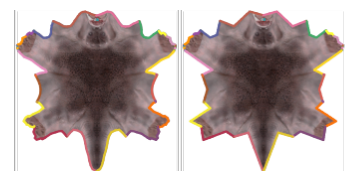
Seam Erasure



Seam Aware Decimation



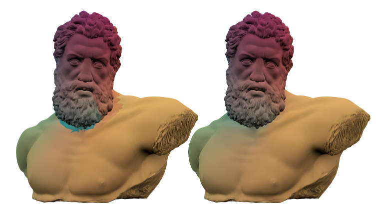
Seam Straightener



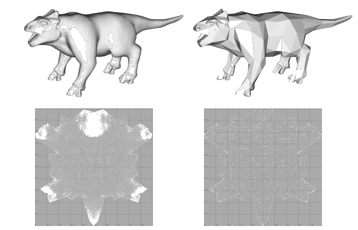
In total we have <click> erased the seam from our textures in a one time preprocess. This does not change the rendering pipeline and therefore can be easily integrated in current workflows. We also showed <click> how we can decimate a model without introducing seam artifacts. This allows models with the same parametric domain to share texture images. To aid in our decimation <click> we introduced a seam straightening algorithm. Increasing the number of collinear seam edges and therefore the quality of decimation. Lastly, we combined our seamless texture and decimated models <click> to show how skinning weights for deformations can be stored in textures and adaptively used to create weight maps.

CONCLUSION

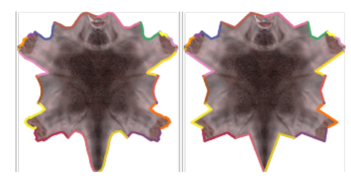
Seam Erasure



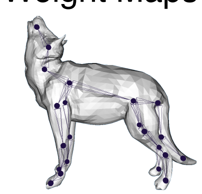
Seam Aware Decimation



Seam Straightener



Weight Maps



In total we have <click> erased the seam from our textures in a one time preprocess. This does not change the rendering pipeline and therefore can be easily integrated in current workflows. We also showed <click> how we can decimate a model without introducing seam artifacts. This allows models with the same parametric domain to share texture images. To aid in our decimation <click> we introduced a seam straightening algorithm. Increasing the number of collinear seam edges and therefore the quality of decimation. Lastly, we combined our seamless texture and decimated models <click> to show how skinning weights for deformations can be stored in textures and adaptively used to create weight maps.

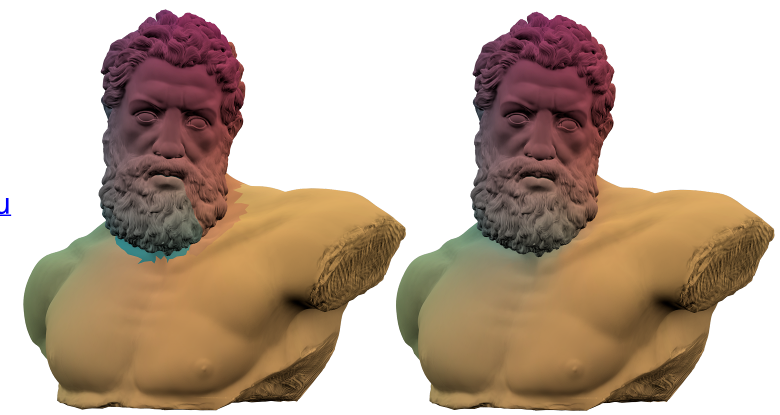
LIMITATIONS AND FUTURE WORK

- Limitations:
  - Low resolution result is constant
  - Non-overlapping parametrization
  - Tangent space normal maps

- Future Work:
  - Minimize the bilinear reconstruction error of the displacement and geometry images
  - Volumetric textures (trilinear interpolation)

SEAMLESS: SEAM ERASURE AND SEAM-AWARE DECOUPLING OF SHAPE FROM MESH RESOLUTION

Project page and Source code:
https://cragl.cs.gmu.edu/seamless/



Contact:
Zachary Ferguson, zfergus@nyu.edu
Songrun Liu, sliu11@gmu.edu

We are grateful to Guilin Liu and Jyh-Ming Lien, Keenan Crane, and Turbosquid users Sumatra3d, Deniz Ozemre, SpinQuad1976, Pabong, Tornado Studio, and mnphmnmn.

This work was supported by the US NSF, Google, NSERC, and Adobe Systems Inc.

The source code is currently accessible online, and if you have any questions in the future feel free to contact either Songrun or myself.