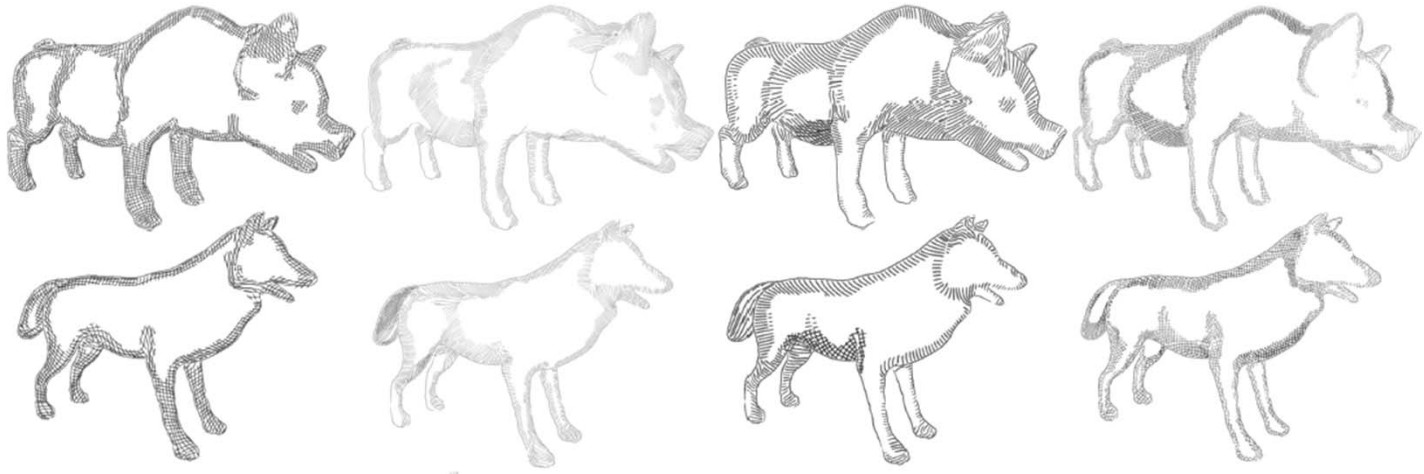
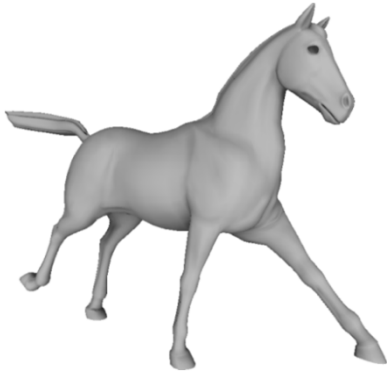
Learning hatching for pen-and-ink illustrations of surfaces



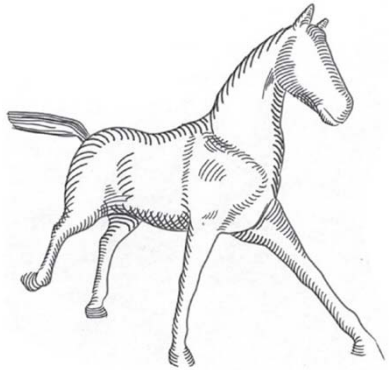
Evangelos Kalogerakis[1,2], Derek Nowrouzehahrai[1,3,4], Simon Breslav[1,5], Aaron Hertzmann[1]

[1]University of Toronto, [2]Stanford University, [3]Disney Research Zurich, [4]University of Montreal, [5]Autodesk Research

Goal: Synthesis of hatching illustrations

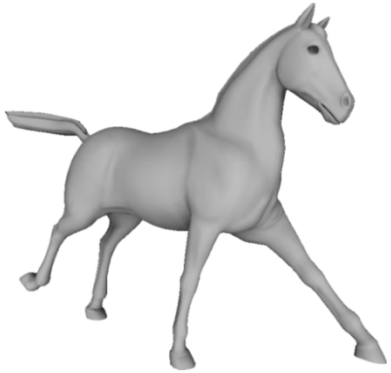


Exemplar shape

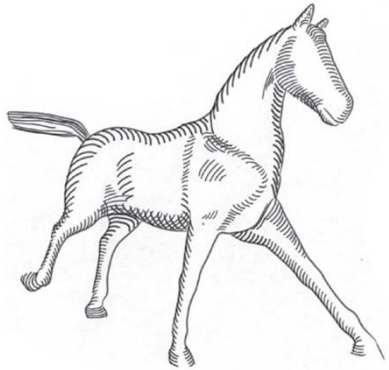


Artist's illustration

Goal: Synthesis of hatching illustrations



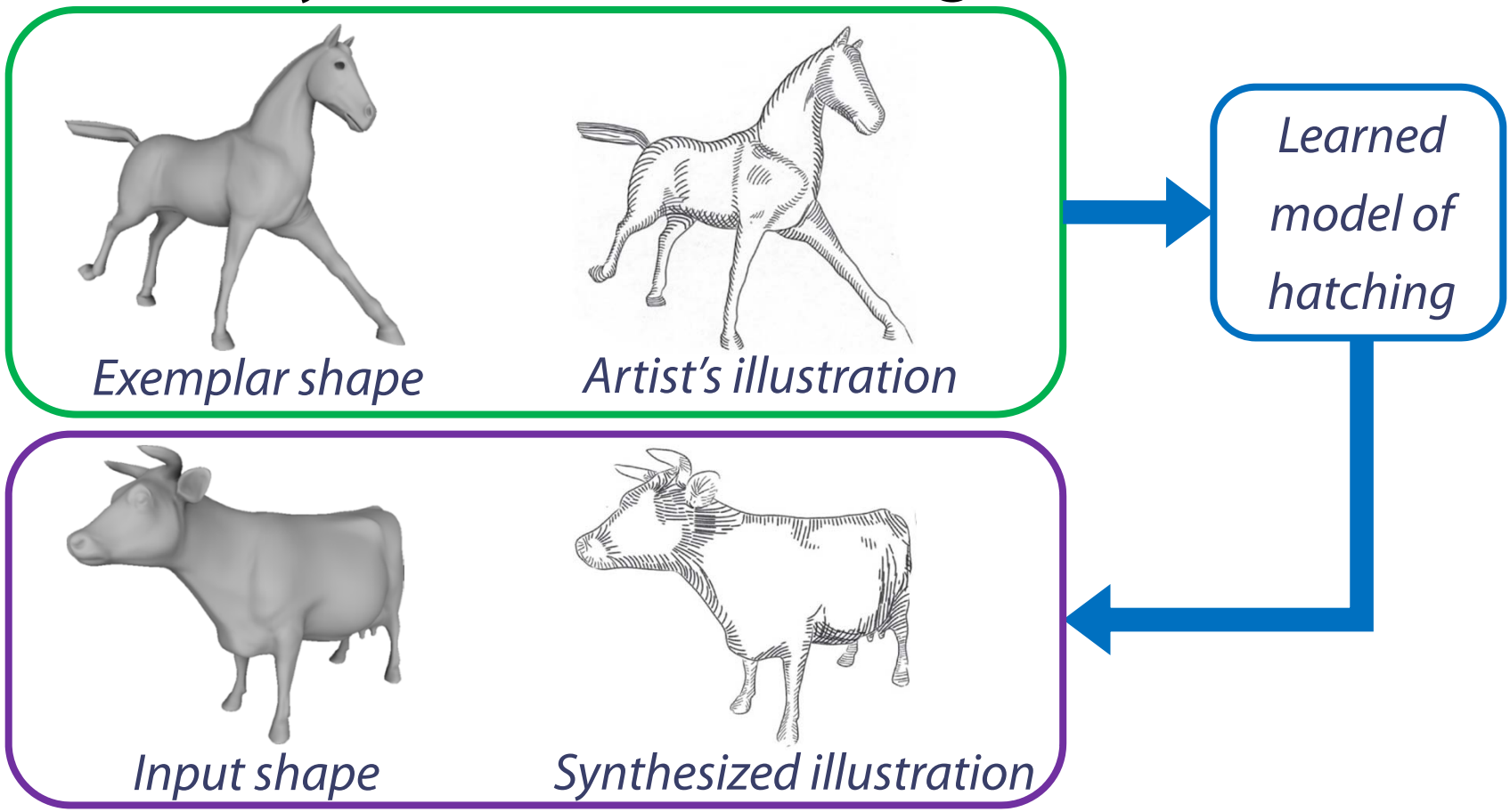
Exemplar shape



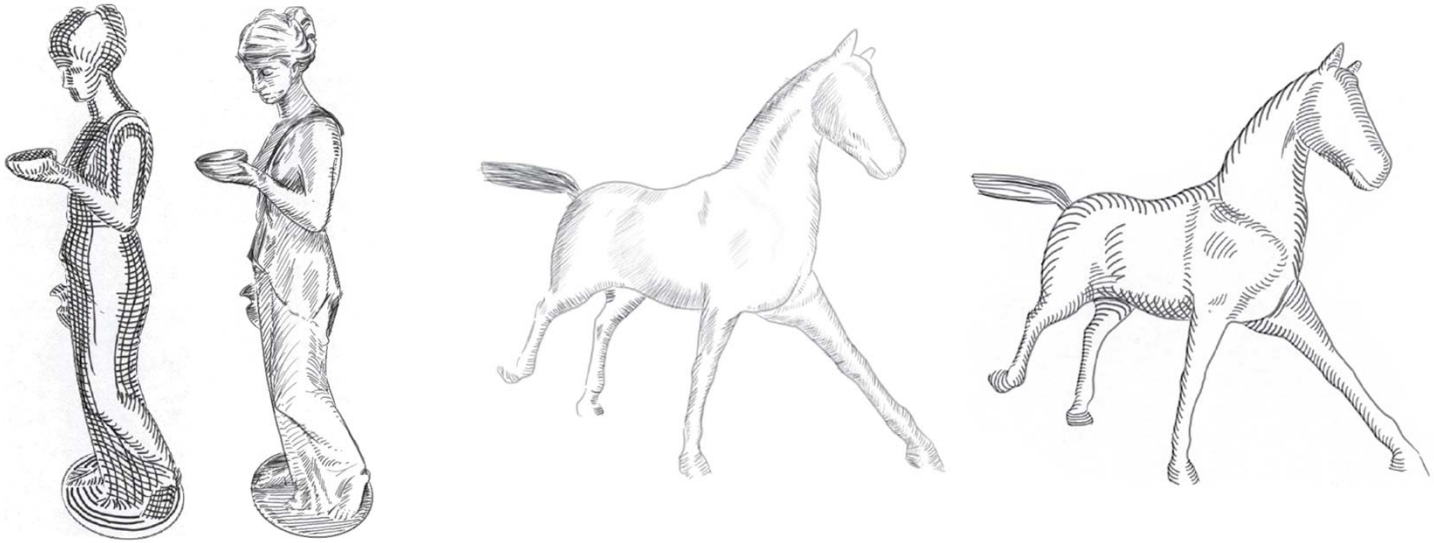
Artist's illustration

Learned model of hatching

Goal: Synthesis of hatching illustrations



Challenge: understanding hatching styles



Related work: hatching smooth surfaces

Iso-parametric curves

[Saito and Takahashi 1990, Winkenbach and Salesin 1996]

Related work: hatching smooth surfaces

Iso-parametric curves

[Saito and Takahashi 1990, Winkenbach and Salesin 1996]

Smooth curvature directions and shading-based tone

[Elber 1998, Hertzmann and Zorin 2000]

Related work: hatching smooth surfaces

Iso-parametric curves

[Saito and Takahashi 1990, Winkenbach and Salesin 1996]

Smooth curvature directions and shading-based tone

[Elber 1998, Hertzmann and Zorin 2000]

Shading gradients

[Singh and Schaefer 2010]

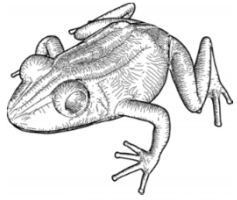
Related work: hatching smooth surfaces

Iso-parametric curves

[Saito and Takahashi 1990, Winkenbach and Salesin 1996]

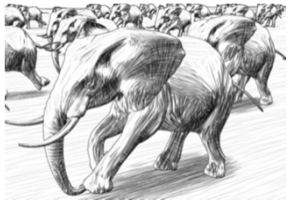
Smooth curvature directions and shading-based tone

[Elber 1998, Hertzmann and Zorin 2000]



Shading gradients

[Singh and Schaefer 2010]

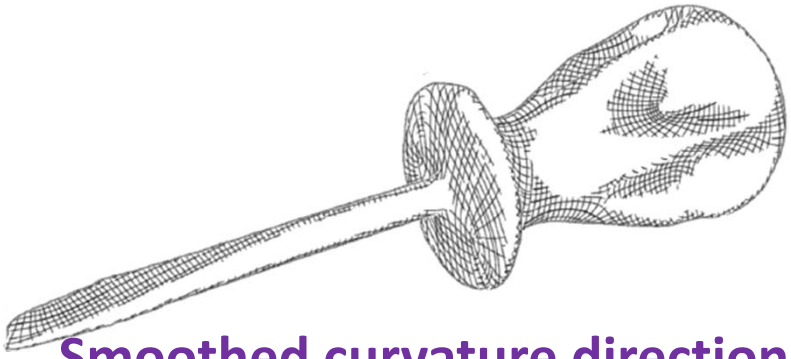


Real-time hatching

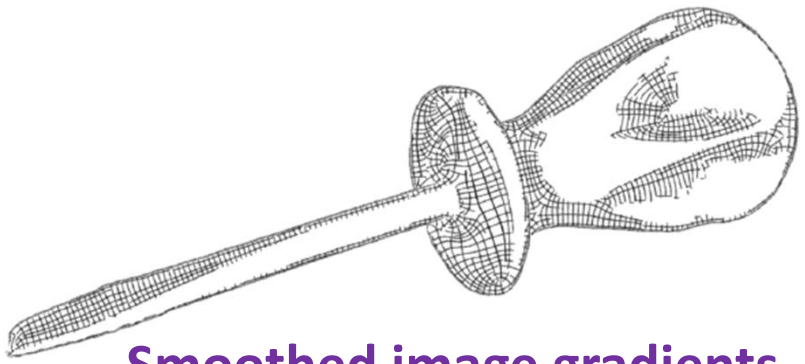
[Praun et al. 2001, Kim et al. 2008]

Related work: hatching smooth surfaces

Artist's illustration



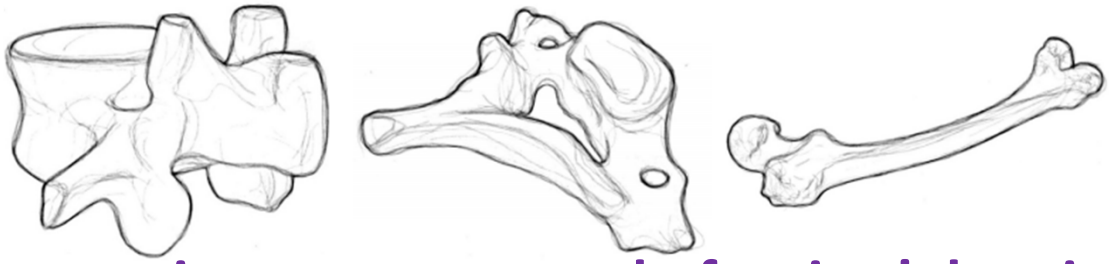
Smoothed curvature directions
[Hertzmann and Zorin 2000]



Smoothed image gradients
[Singh and Schaefer 2010]

Related work: where do people draw lines?

[Cole et al. 2008]



Average images composed of artists' drawings

Predicted line drawing

Our approach

Learns a model of hatching style from a single artist's drawing of an input shape

Our approach

Learns a model of hatching style from a single artist's drawing of an input shape

Can transfer the hatching style to different views of the exemplar shape as well as different shapes

Our approach

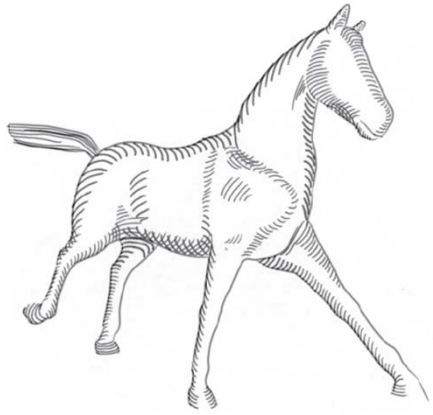
Learns a model of hatching style from a single artist's drawing of an input shape

Can transfer the hatching style to different views of the exemplar shape as well as different shapes

The hatching style is determined by hatching properties related to hatching tone and orientations

Hatching properties

Hatching level

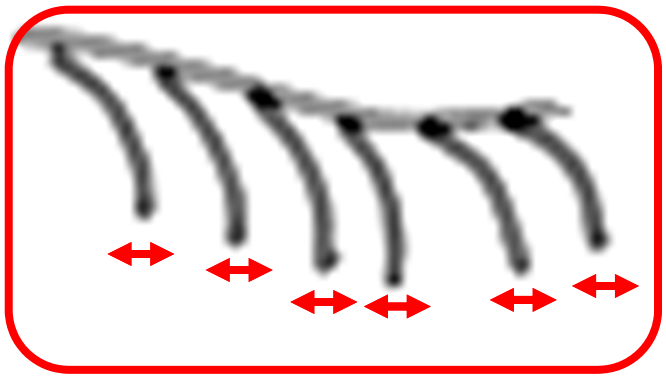


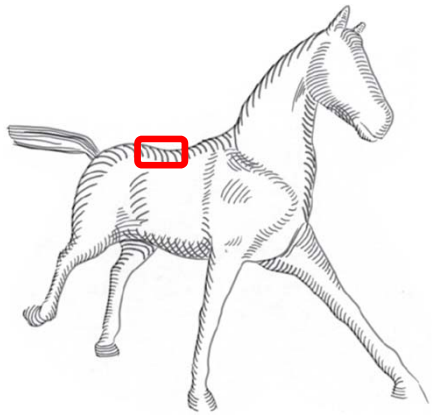
- No hatching
- Hatching
- Cross-hatching

Hatching properties

Hatching level

Stroke thickness



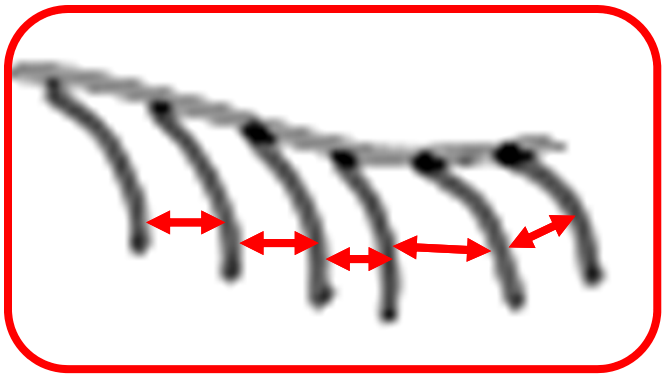


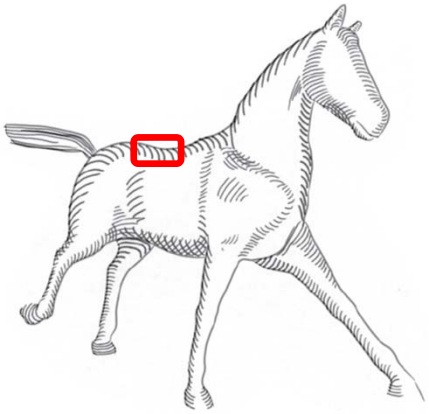
Hatching properties

Hatching level

Stroke thickness

Stroke spacing





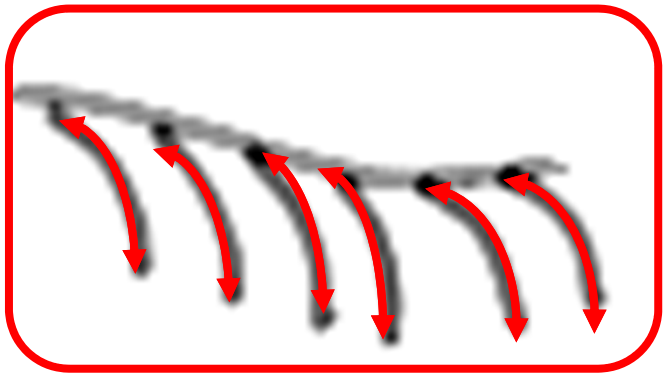
Hatching properties

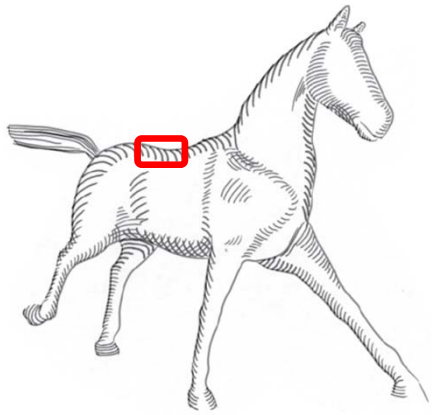
Hatching level

Stroke thickness

Stroke spacing

Stroke length





Hatching properties

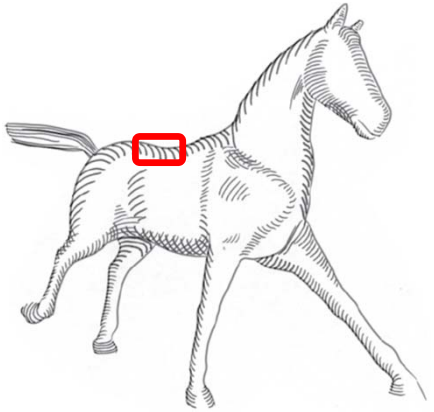
Hatching level

Stroke thickness

Stroke spacing

Stroke length

Stroke intensity



Hatching properties

Hatching level

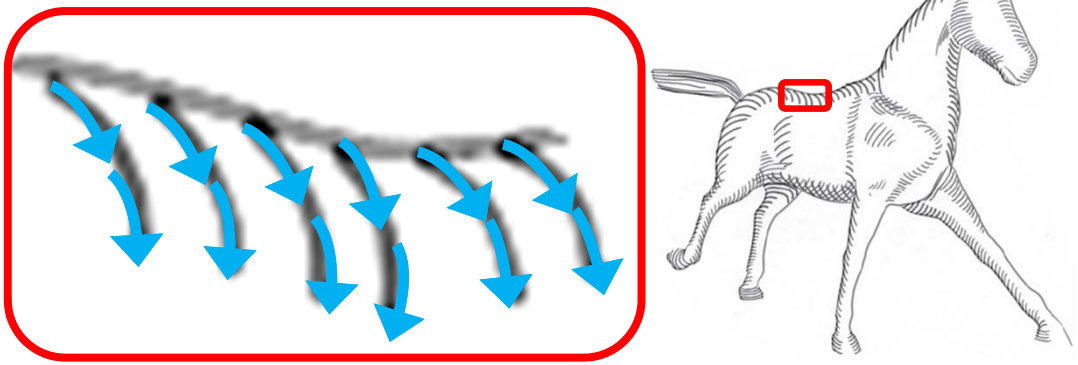
Stroke thickness

Stroke spacing

Stroke length

Stroke intensity

Hatching orientations



Hatching properties

Hatching level

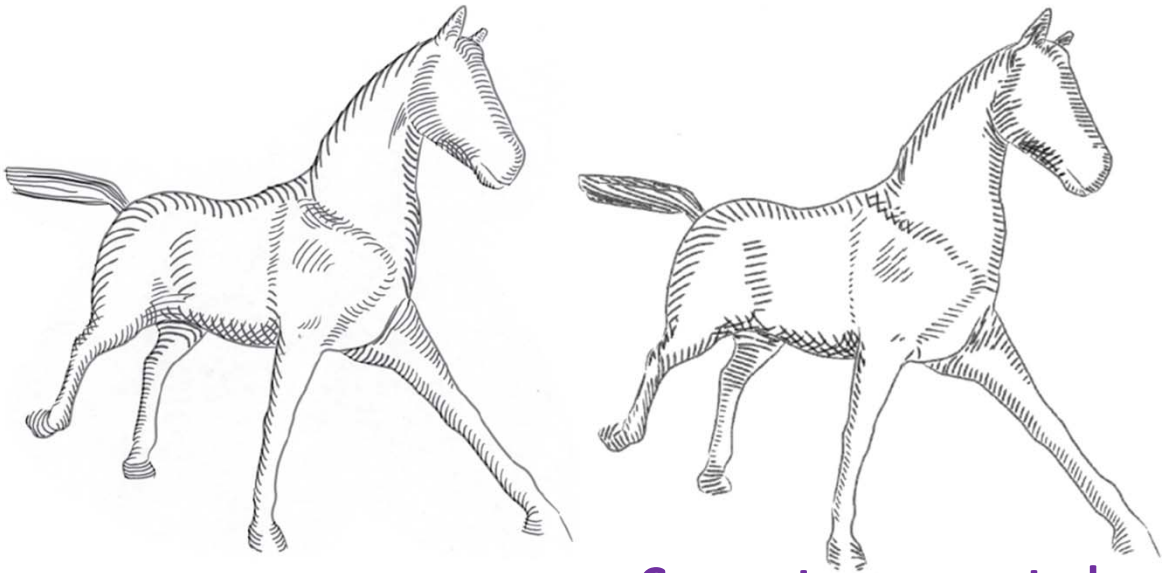
Stroke thickness

Stroke spacing

Stroke length

Stroke intensity

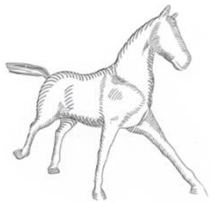
Hatching orientations



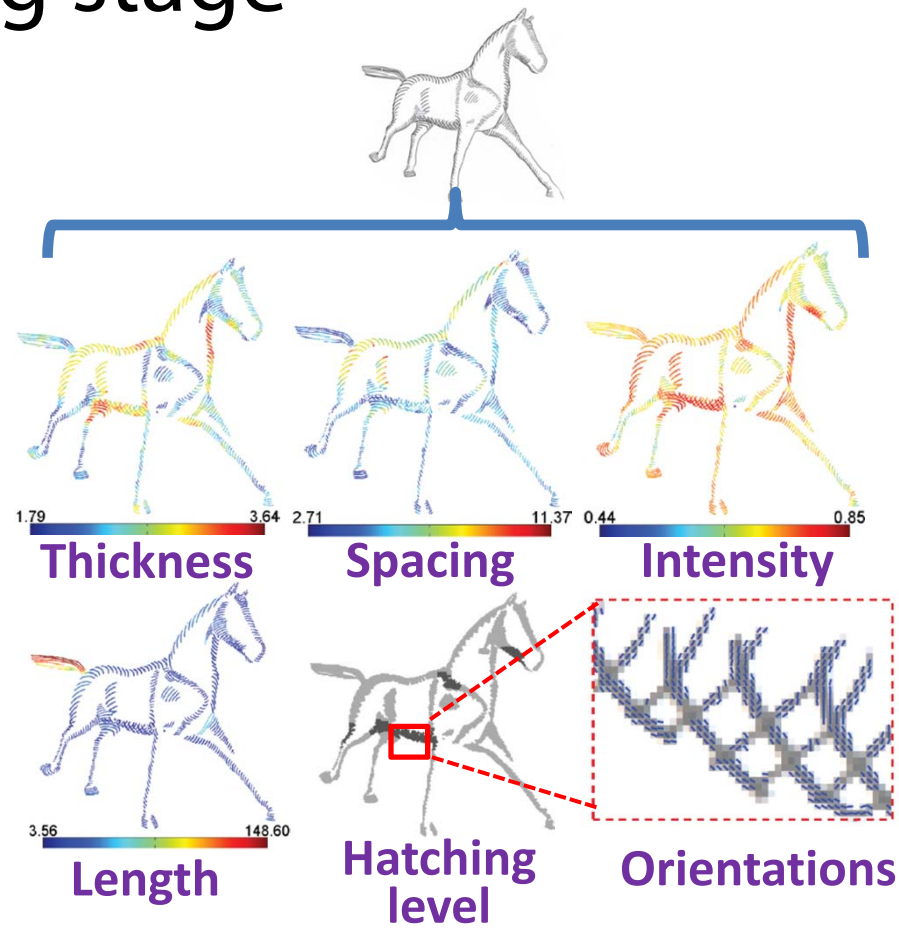
Artist's illustration

Computer-generated illustration

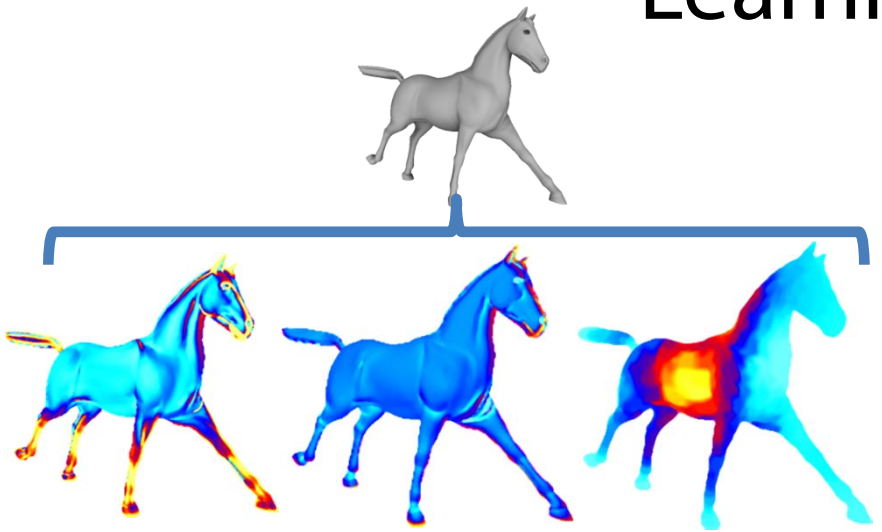
Learning stage



Learning stage

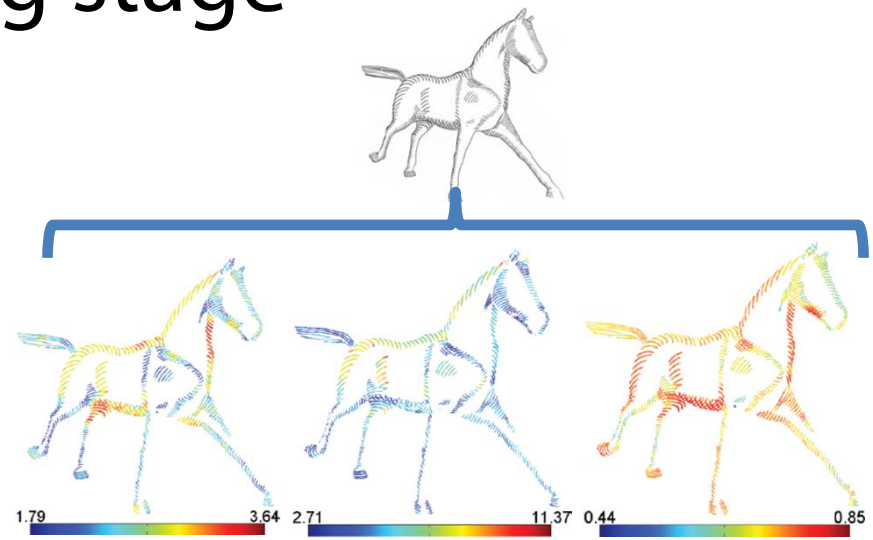


Learning stage

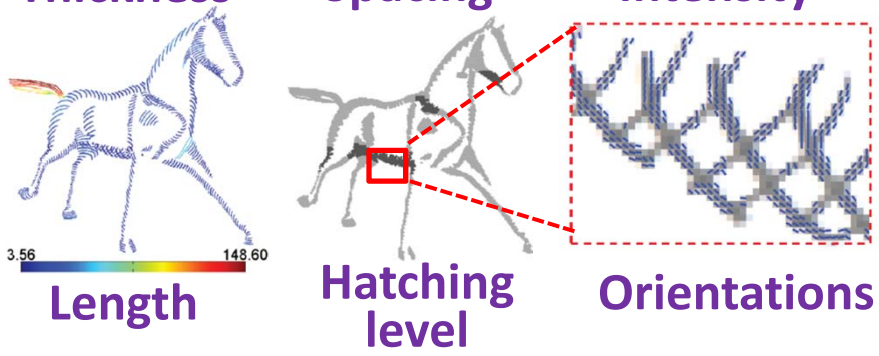


Shape features

Image-space features

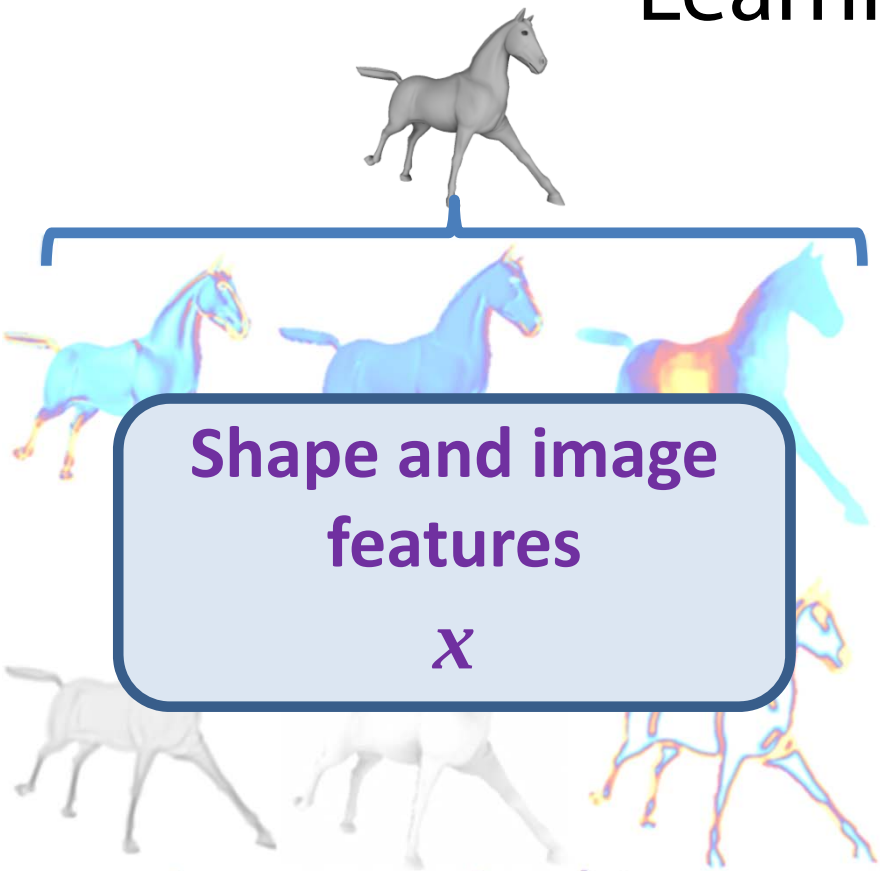


Thickness Spacing Intensity



Length Hatching level Orientations

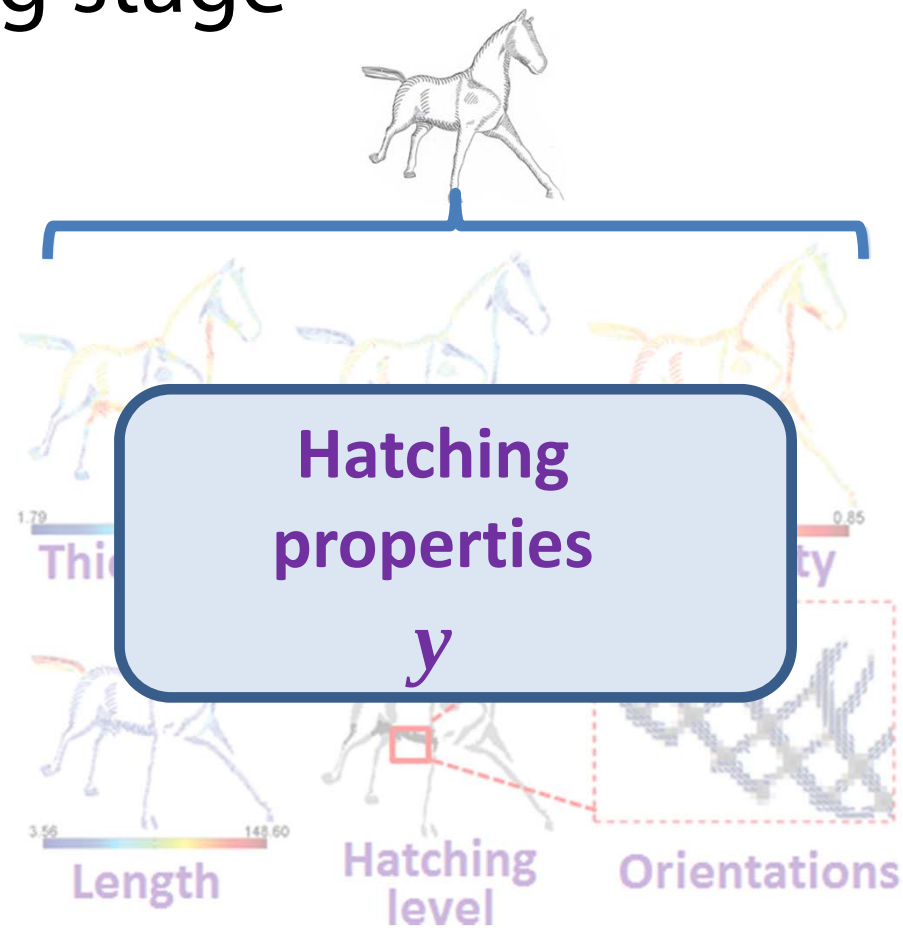
Learning stage



Shape and image features

x

Image-space descriptors



Hatching properties

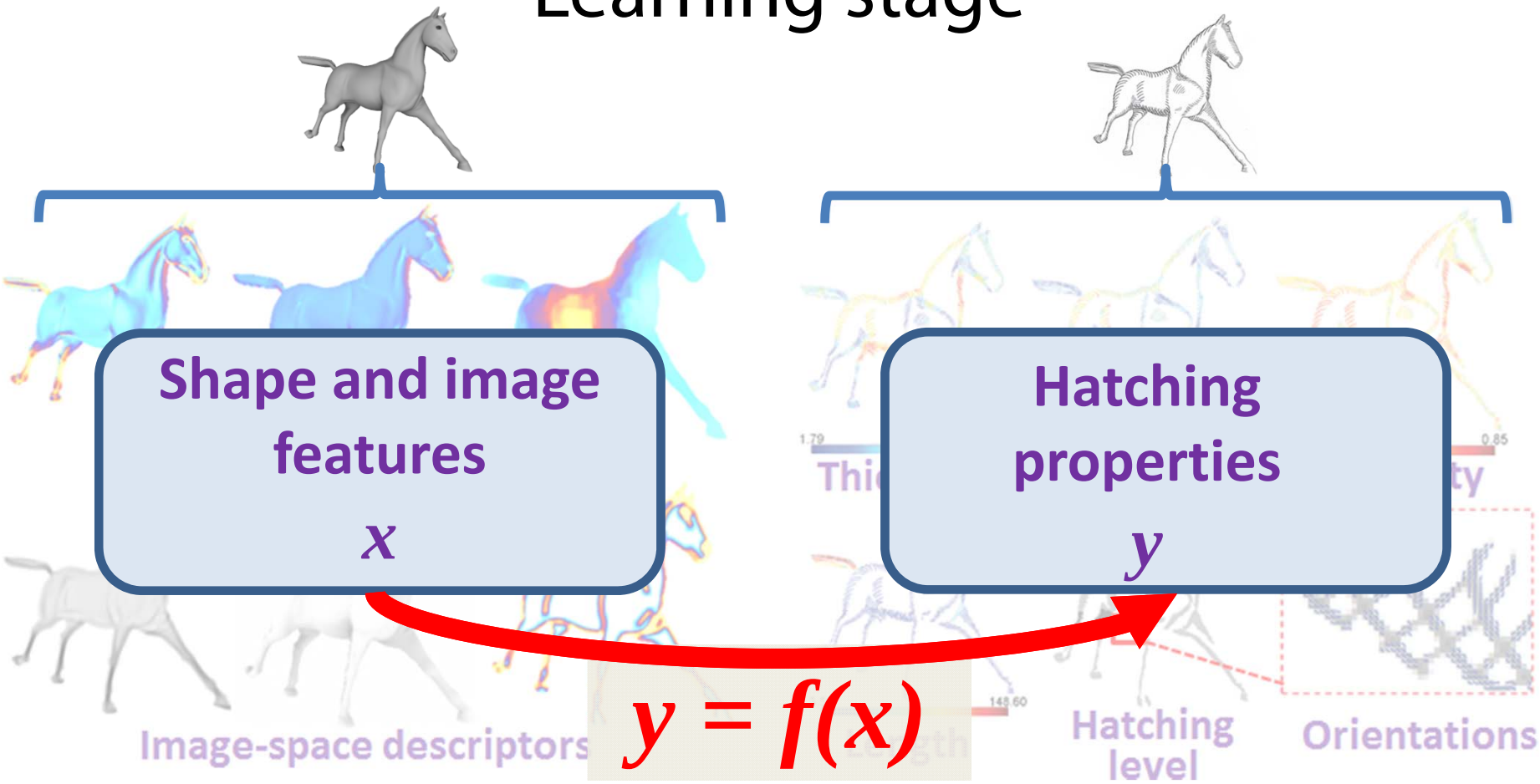
y

Length

Hatching level

Orientations

Learning stage



Learning hatching orientations

Linear model expressing hatching orientations as a weighted sum of selected orientation features.

$$f(\boldsymbol{\theta}; \mathbf{w}) = \sum_k w_k \, \mathbf{v}_k$$

$$\mathbf{v} = [\cos(2\theta), \sin(2\theta)]^T$$

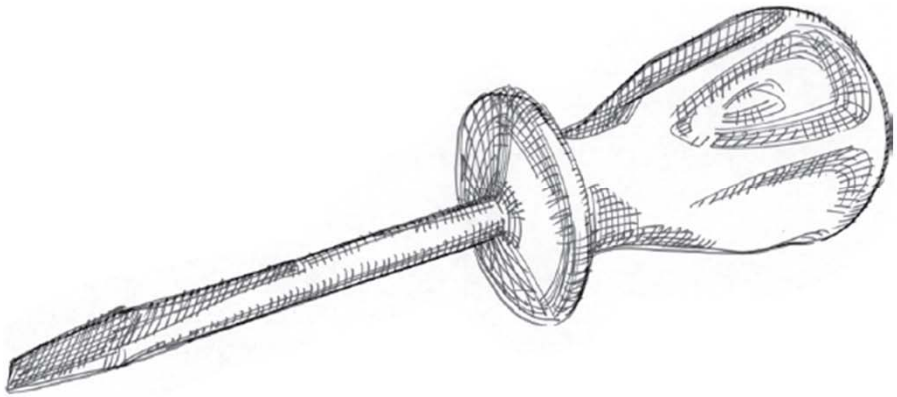
Learning hatching orientations

Linear model expressing hatching orientations as a weighted sum of selected orientation features.

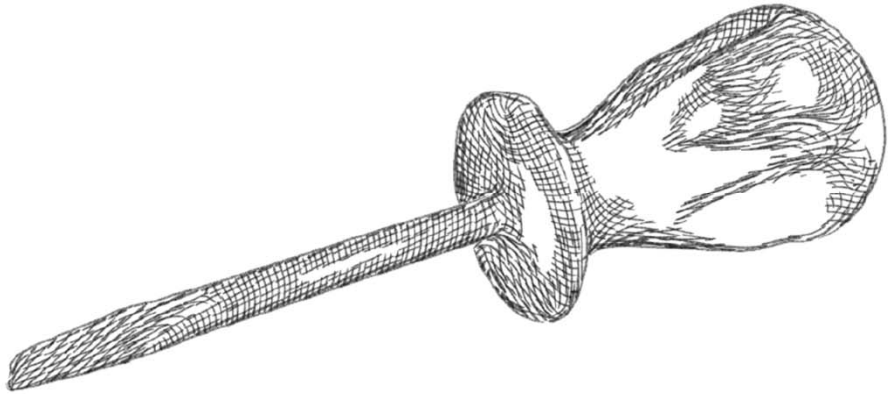
$$f(\boldsymbol{\theta}; \mathbf{w}) = \sum_k w_k \mathbf{v}_k$$

$$\mathbf{v} = [\cos(2\theta), \sin(2\theta)]^T$$

Learning hatching orientations

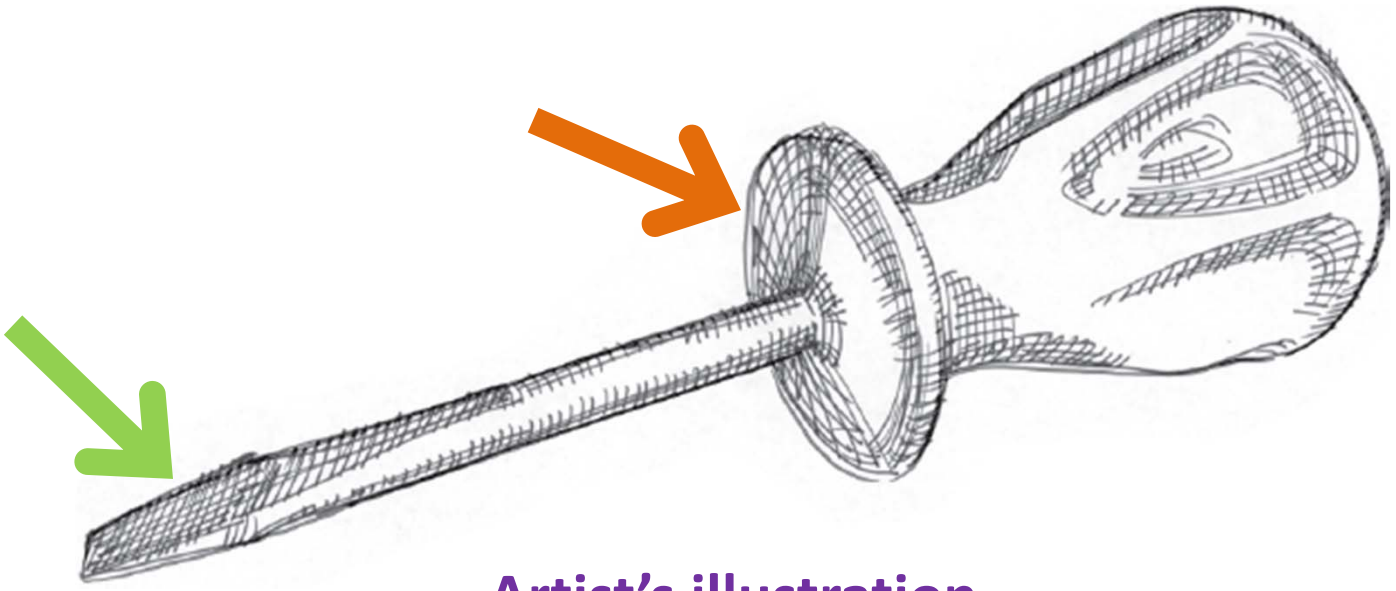


Artist's illustration



Fitting a single model across the illustration

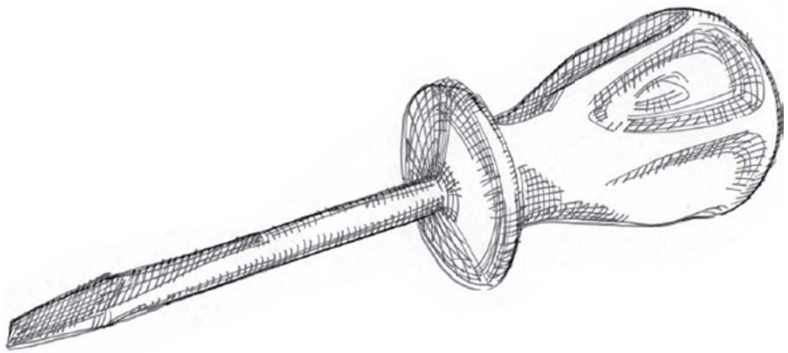
Learning orientation fields

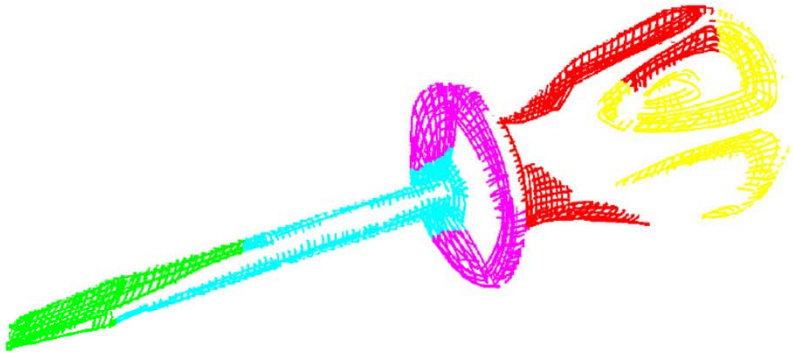


Artist's illustration

Mixture of experts model

Simultaneous segmentation & model fitting for each segment





$\vec{f}_1 = \nabla a_2$
$\vec{f}_2 = .54(\vec{k}_{max,1}) + .46(\vec{r}_\perp)$

$\vec{f}_1 = .73(\nabla I_3) + .27(\vec{r})$
$\vec{f}_2 = .69(\vec{k}_{max,2}) + .31(\nabla I_{\perp,3})$

$\vec{f}_1 = .77(\vec{e}_{b,3}) + .23(\nabla I_3)$
$\vec{f}_2 = \vec{v}$

$\vec{f}_1 = .59(\vec{e}_{b,3}) + .41(\nabla(\vec{L}\cdot\vec{N})_3)$
$\vec{f}_2 = .63(\vec{e}_{a,3}) + .37(\nabla(\vec{L}\cdot\vec{N})_{\perp,3})$

$\vec{f}_1 = .88(\nabla a_3) + .12(\nabla(\vec{L}\cdot\vec{N})_3)$
$\vec{f}_2 = .45(\vec{k}_{max,2}) + .31(\nabla a_{\perp,3}) + .24(\vec{e}_{a,3})$

Learning stroke properties

Map features to thickness, intensity, spacing, length

$$y = \prod_k (a_k x_k \ + b_k)^{\alpha_k}$$

Learning stroke properties

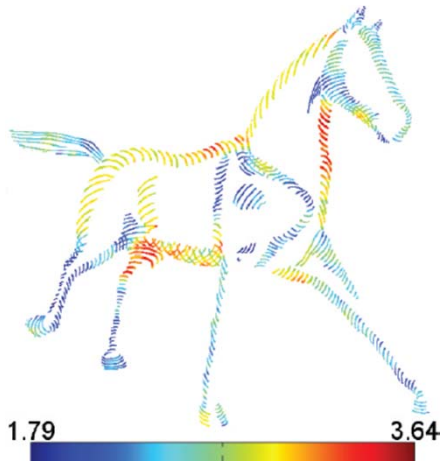
Map features to thickness, intensity, spacing, length

$$y = \prod_k (a_k x_k + b_k)^{\alpha_k}$$

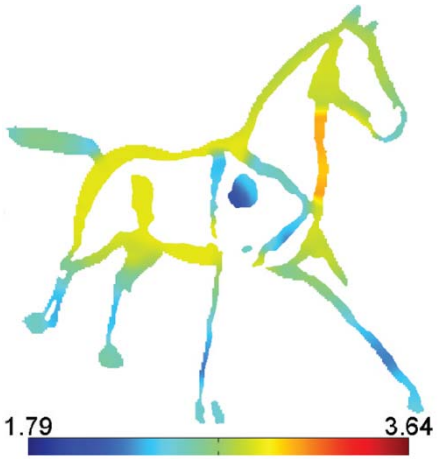
Learning stroke properties

Map features to **thickness**

$$y = \prod_k (a_k x_k + b_k)^{\alpha_k}$$



Extracted thickness

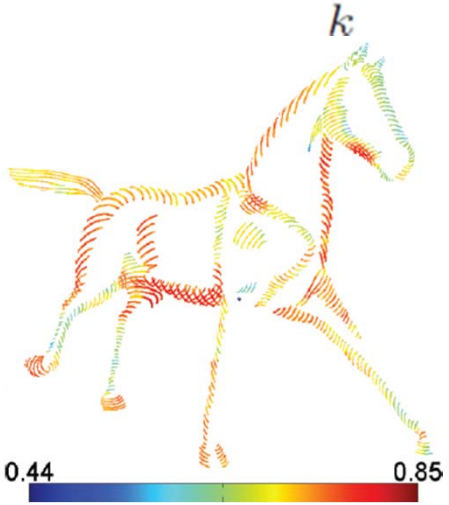


Learned thickness

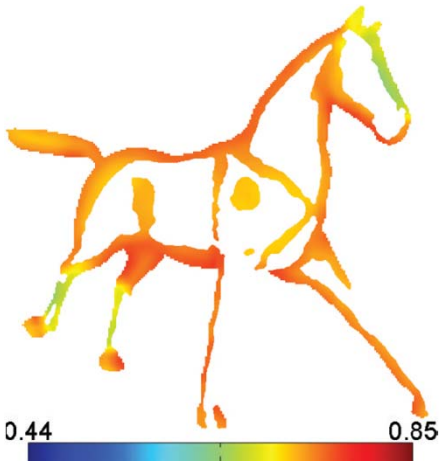
Learning stroke properties

Map features to **intensity**

$$y = \prod_k (a_k x_k + b_k)^{\alpha_k}$$



Extracted intensity

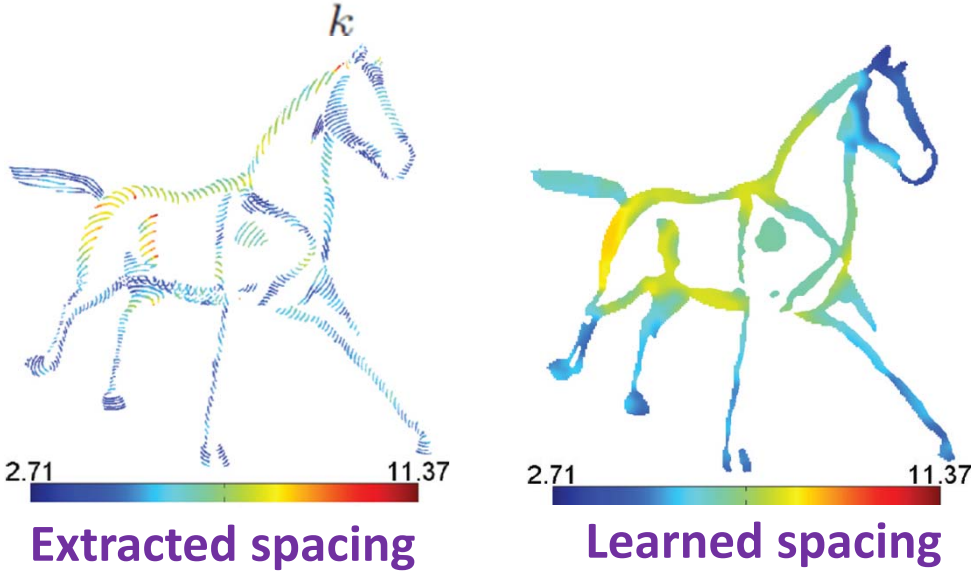


Learned intensity

Learning stroke properties

Map features to **spacing**

$$y = \prod_k (a_k x_k + b_k)^{\alpha_k}$$

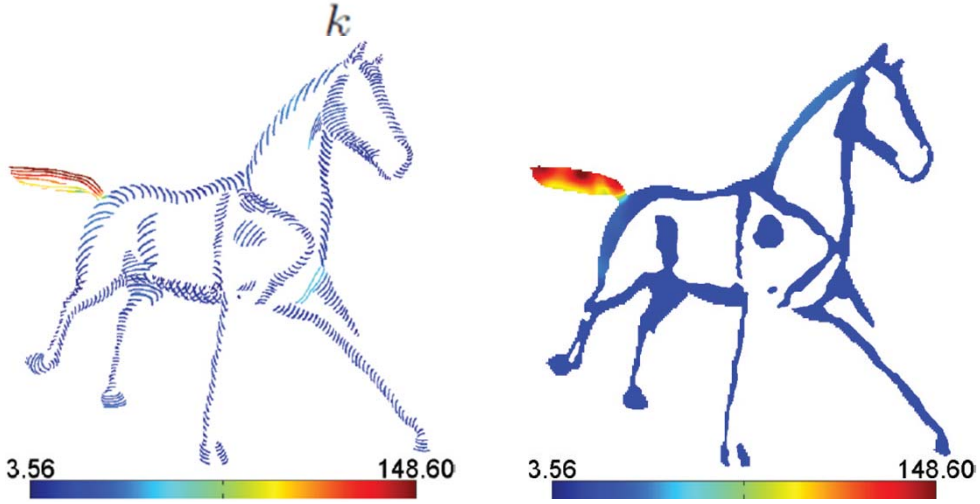


Extracted spacing **Learned spacing**

Learning stroke properties

Map features to **length**

$$y = \prod_k (a_k x_k + b_k)^{\alpha_k}$$

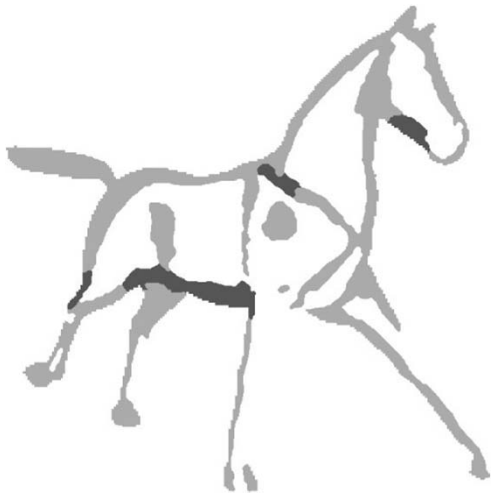


Extracted length **Learned length**

Learning hatching level and segment labels

Map features to discrete values with Joint Boosting + CRF

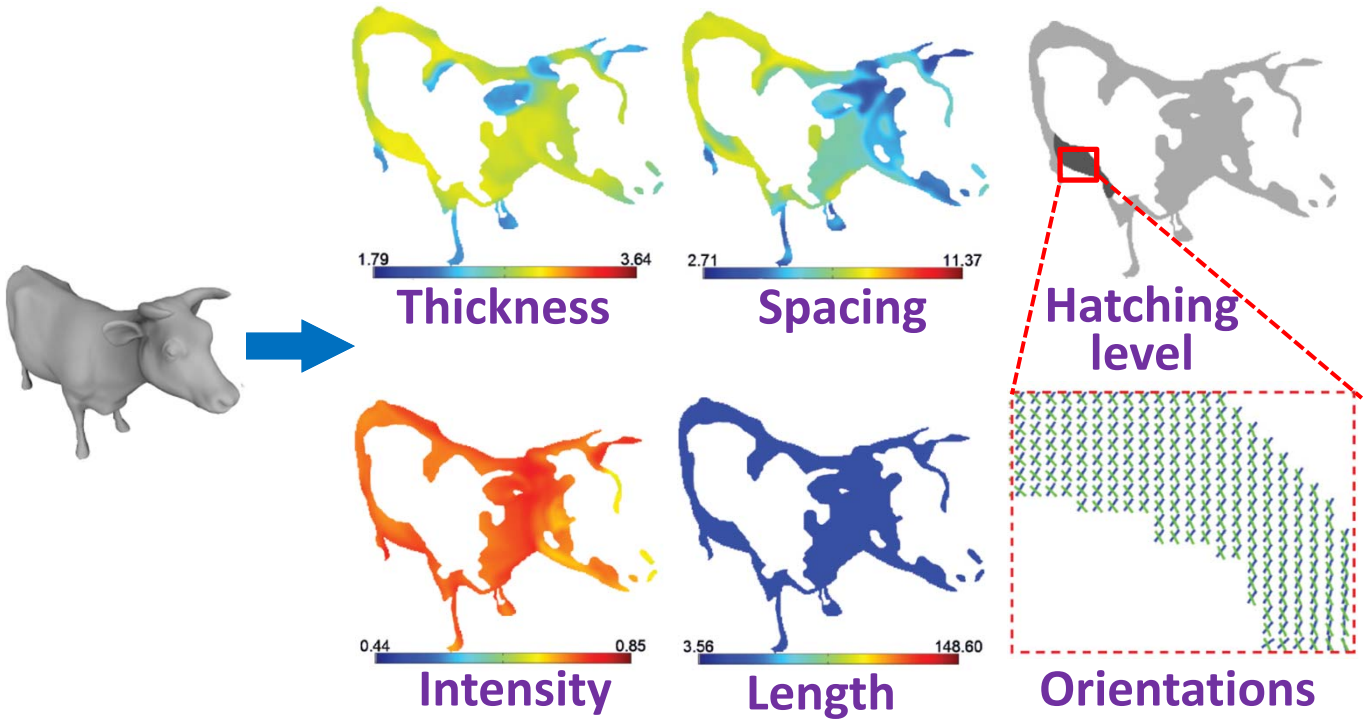
Extracted hatching level



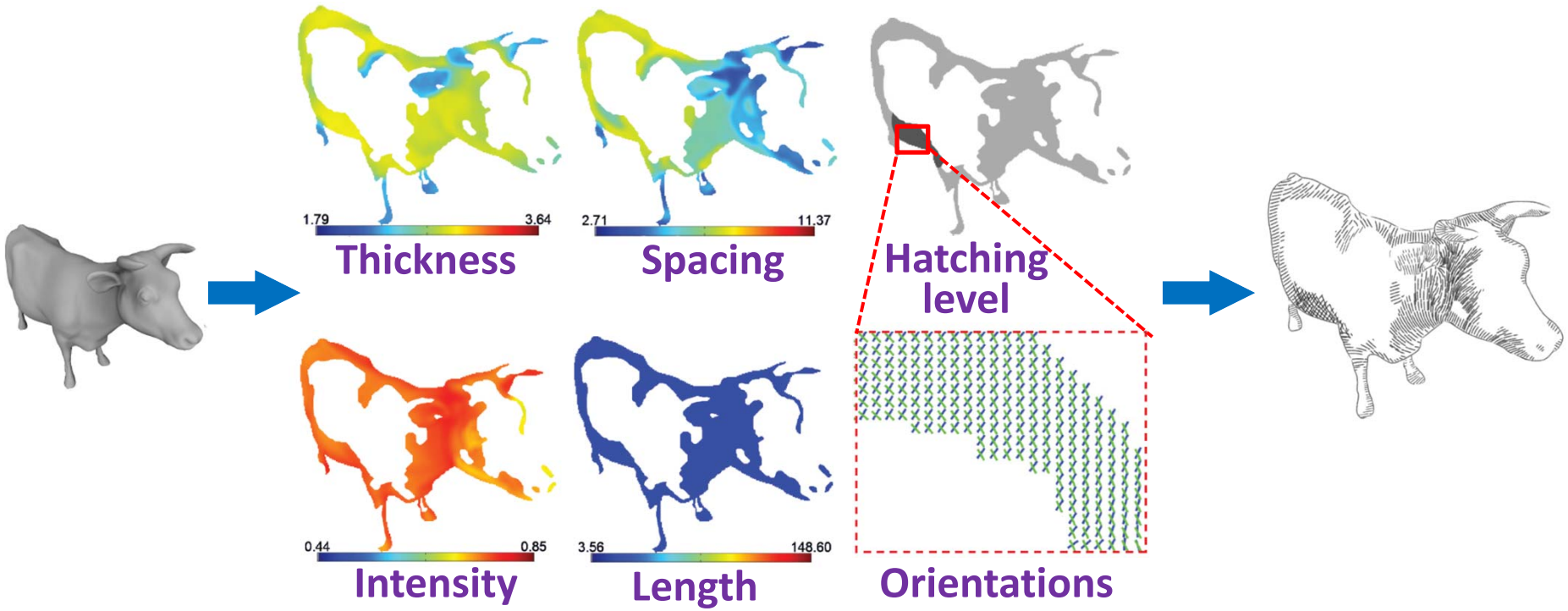
Learned hatching level

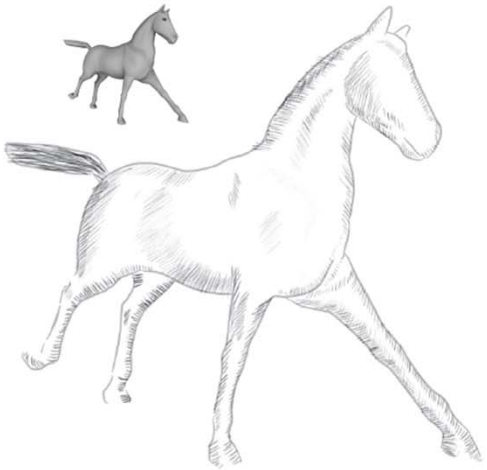
Synthesis stage

Synthesis stage

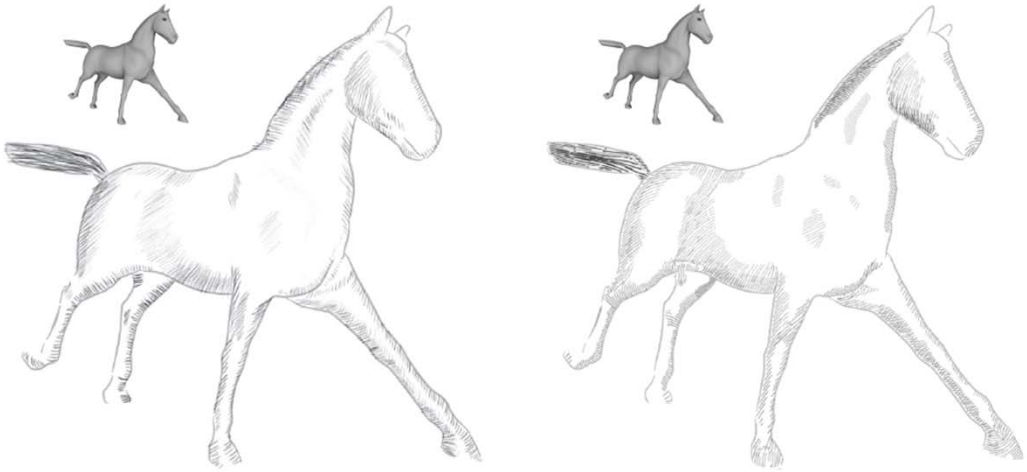


Synthesis stage

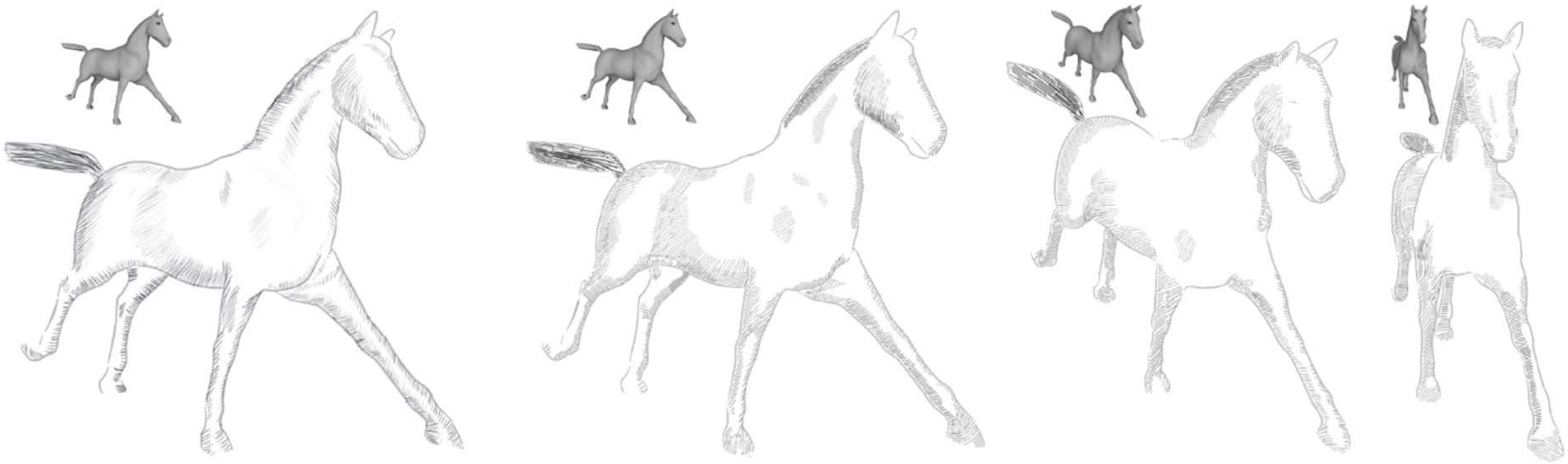




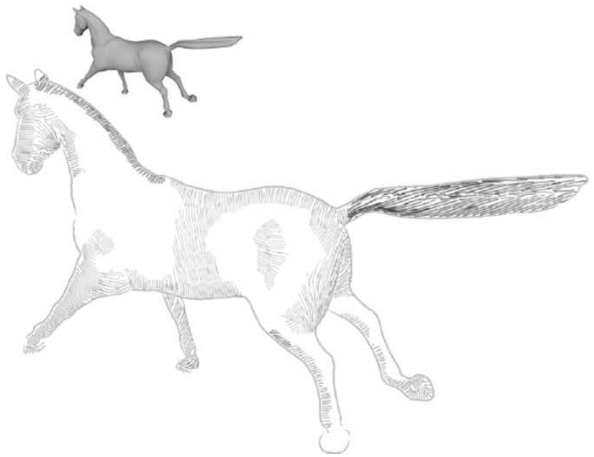
Artist’s illustration

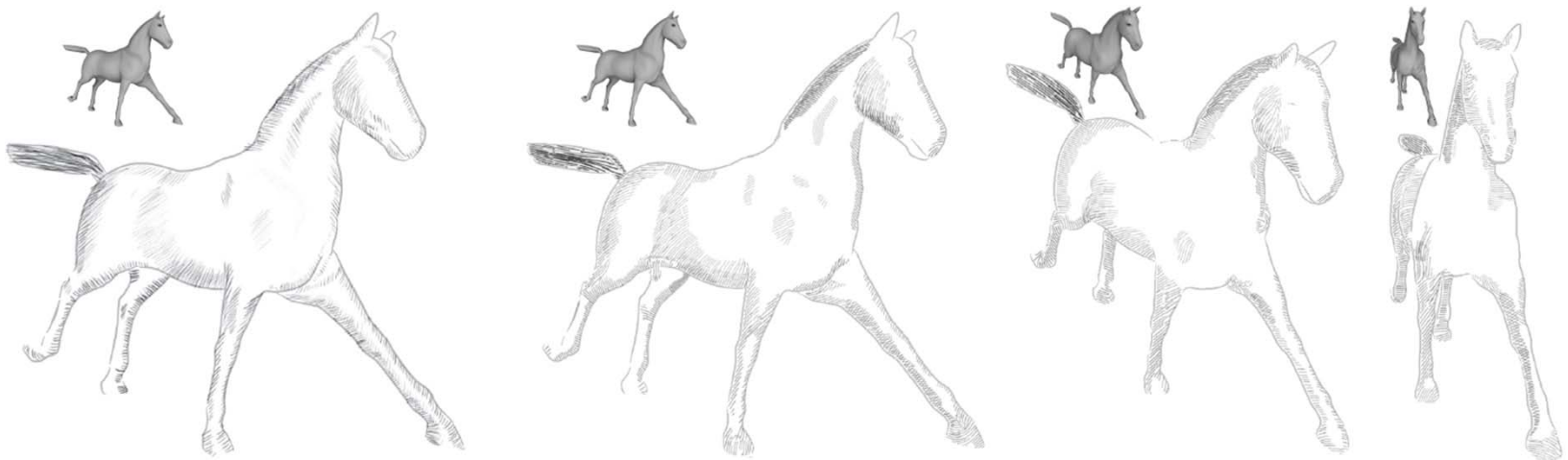


Artist’s illustration

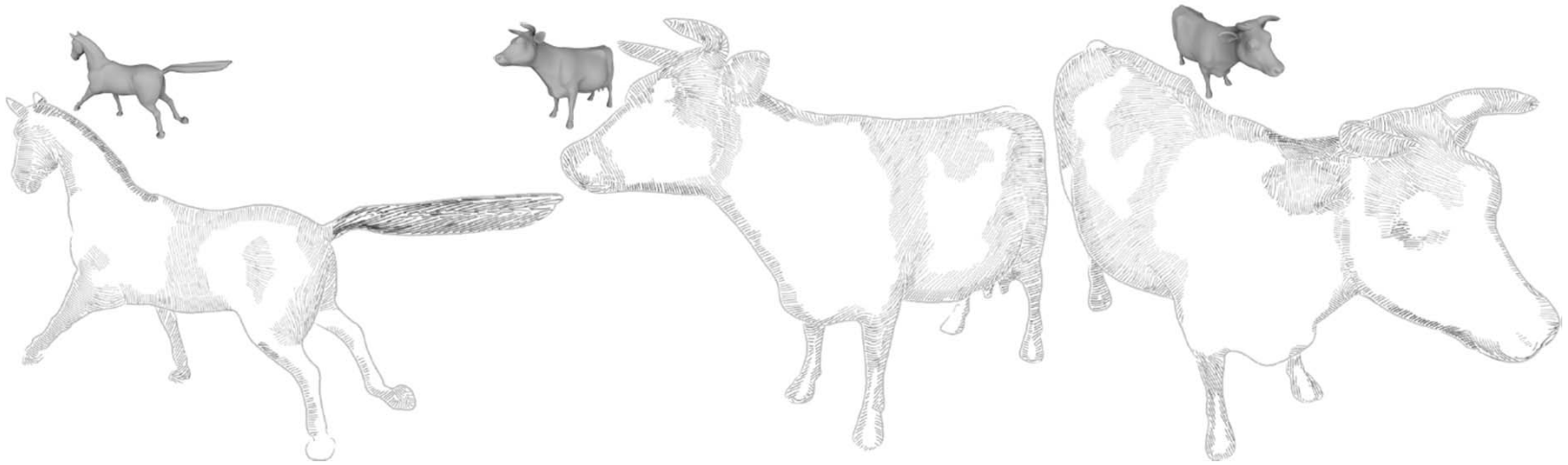


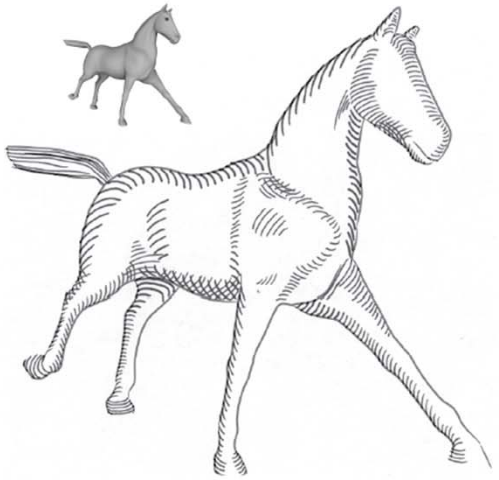
Artist's illustration



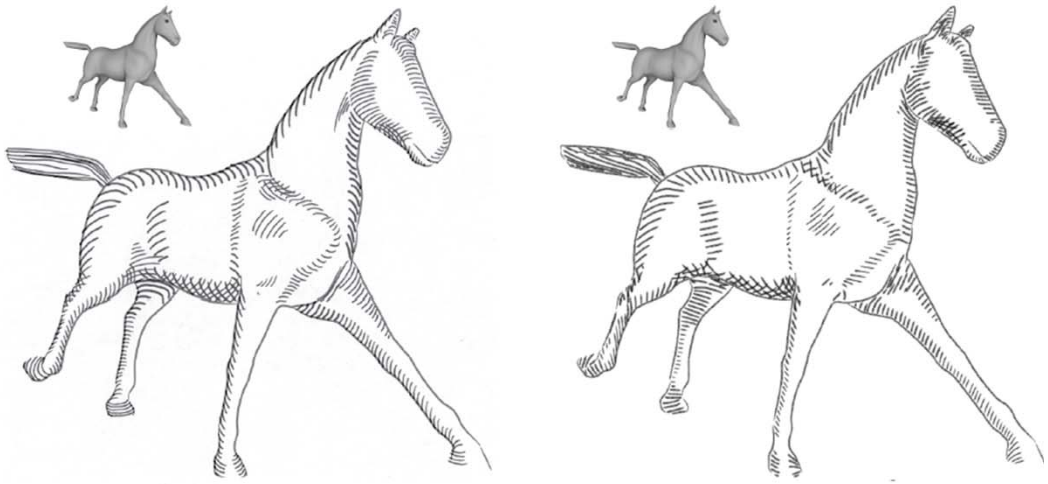


Artist's illustration

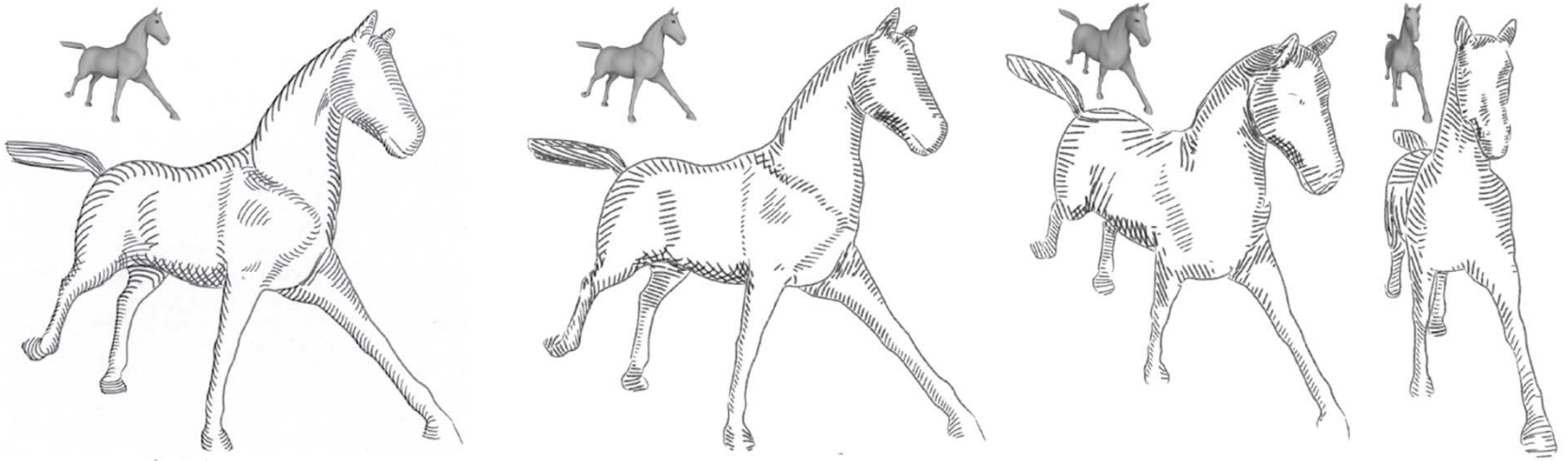




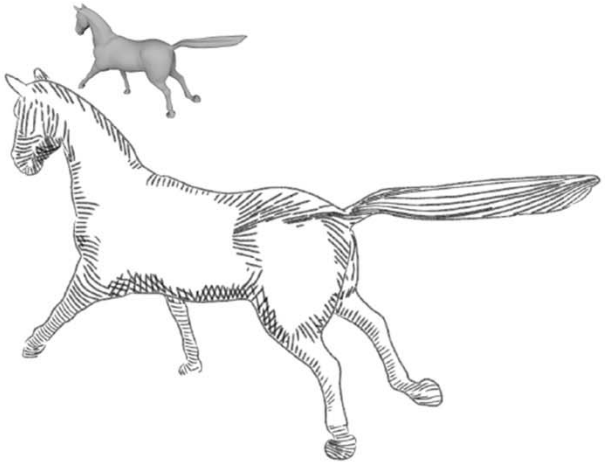
Artist’s illustration

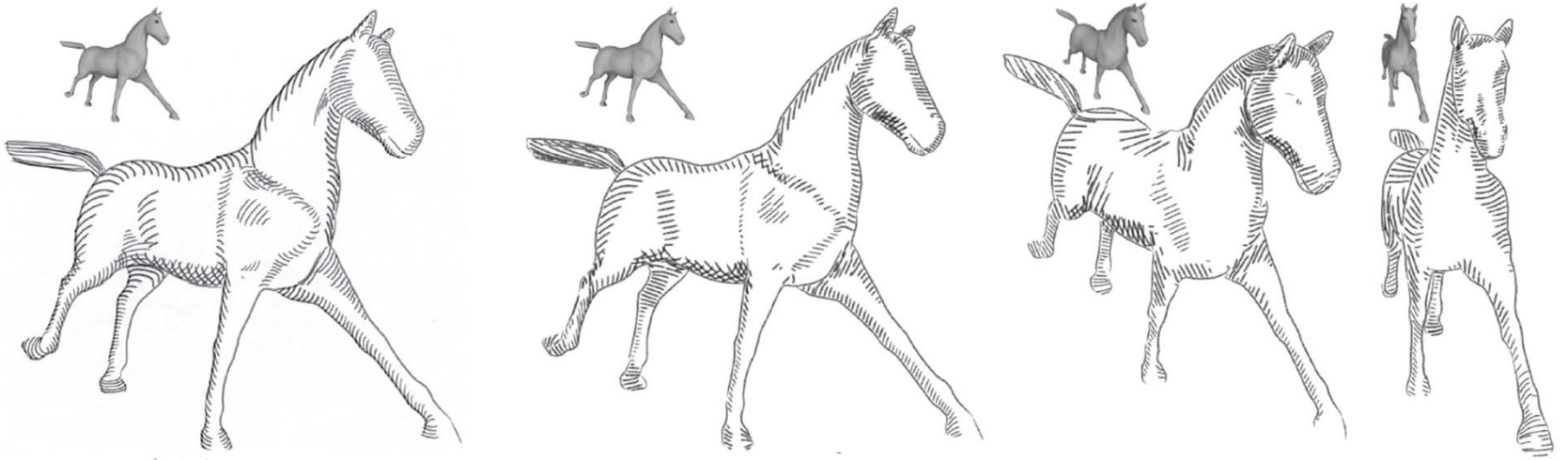


Artist's illustration

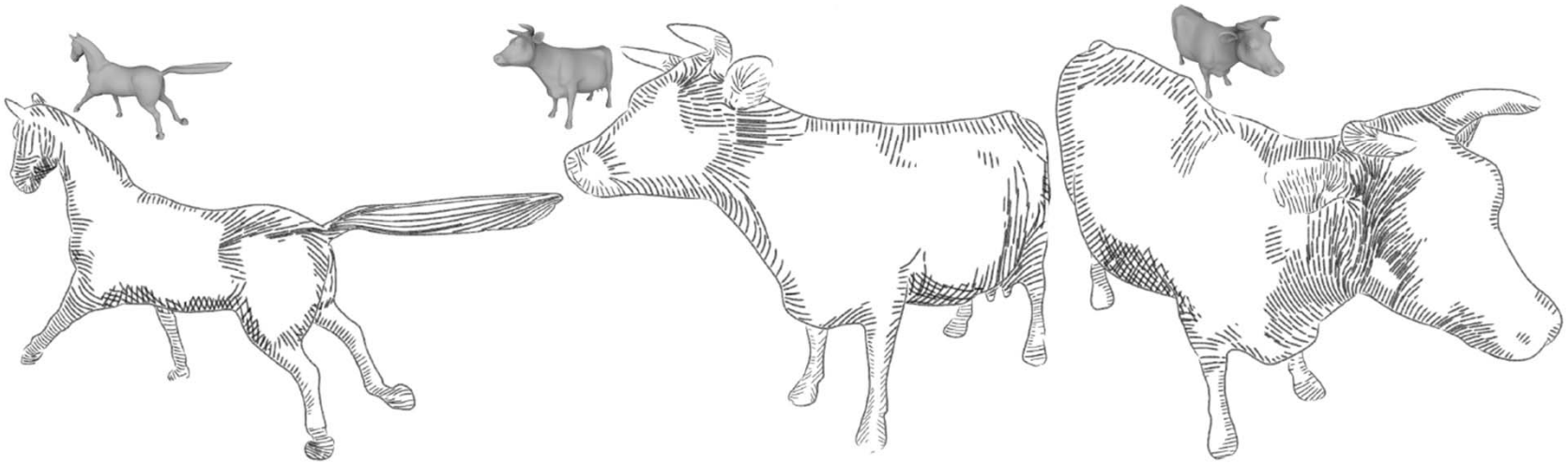


Artist's illustration

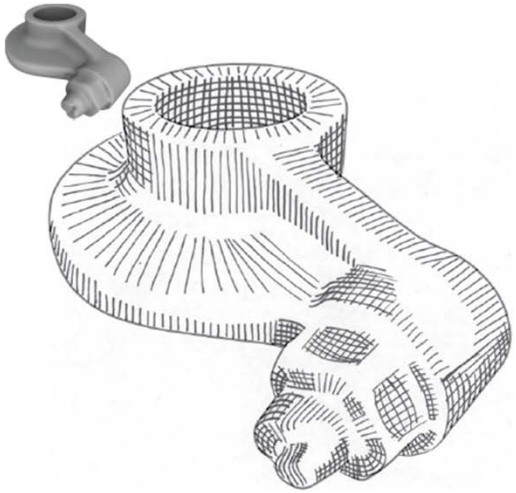


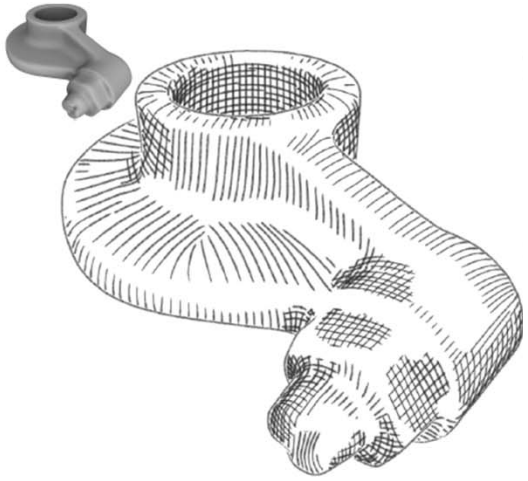


Artist's illustration

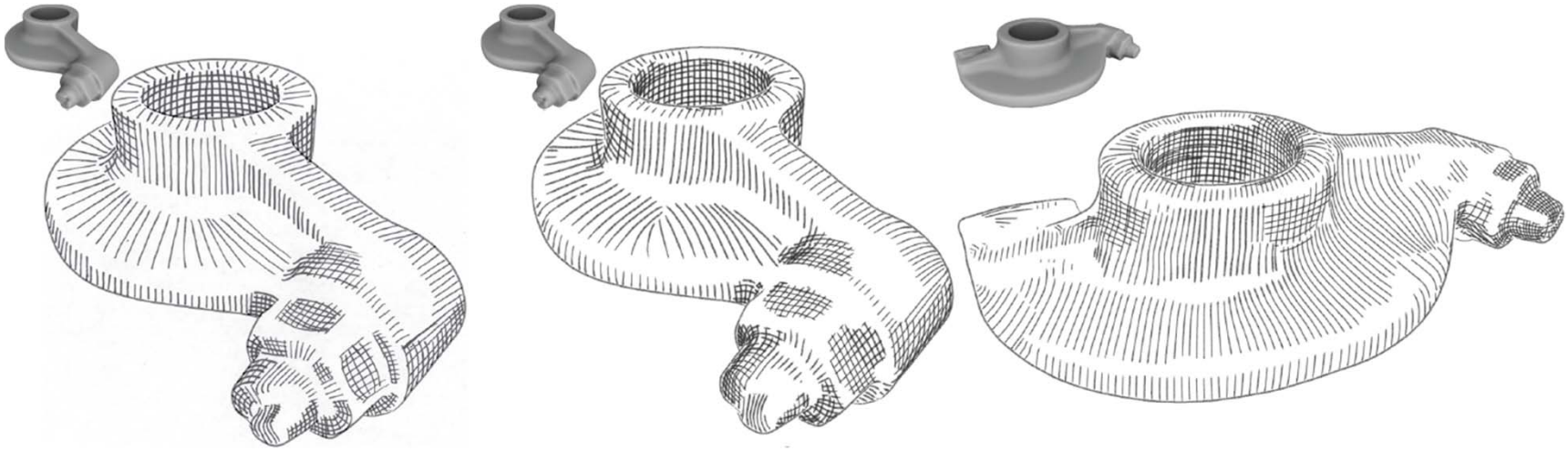


Artist’s illustration

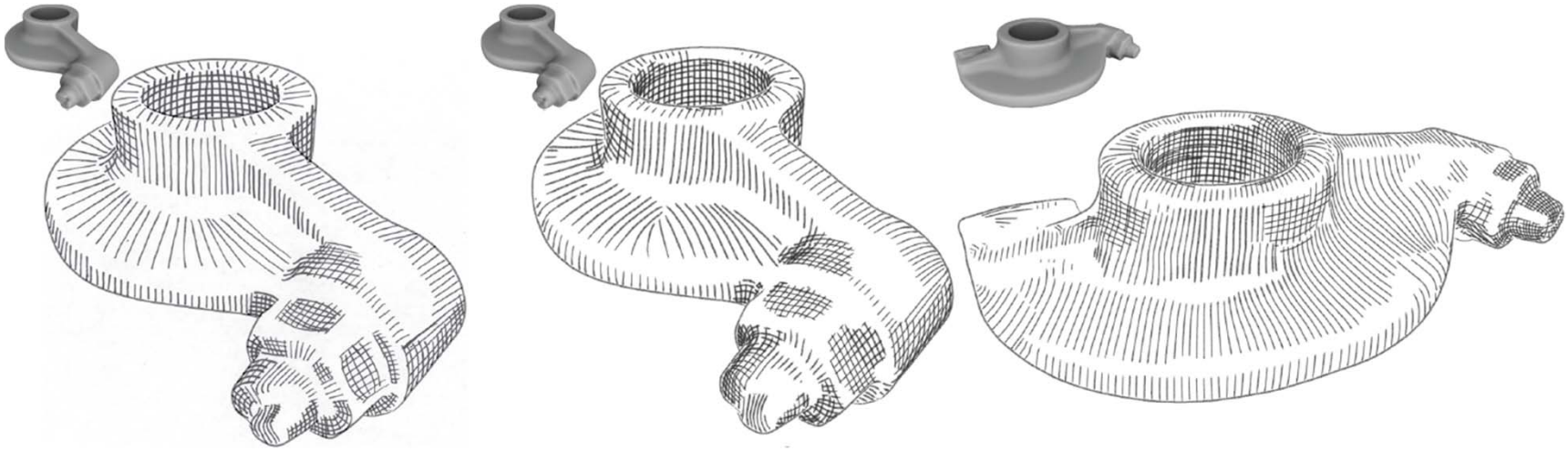




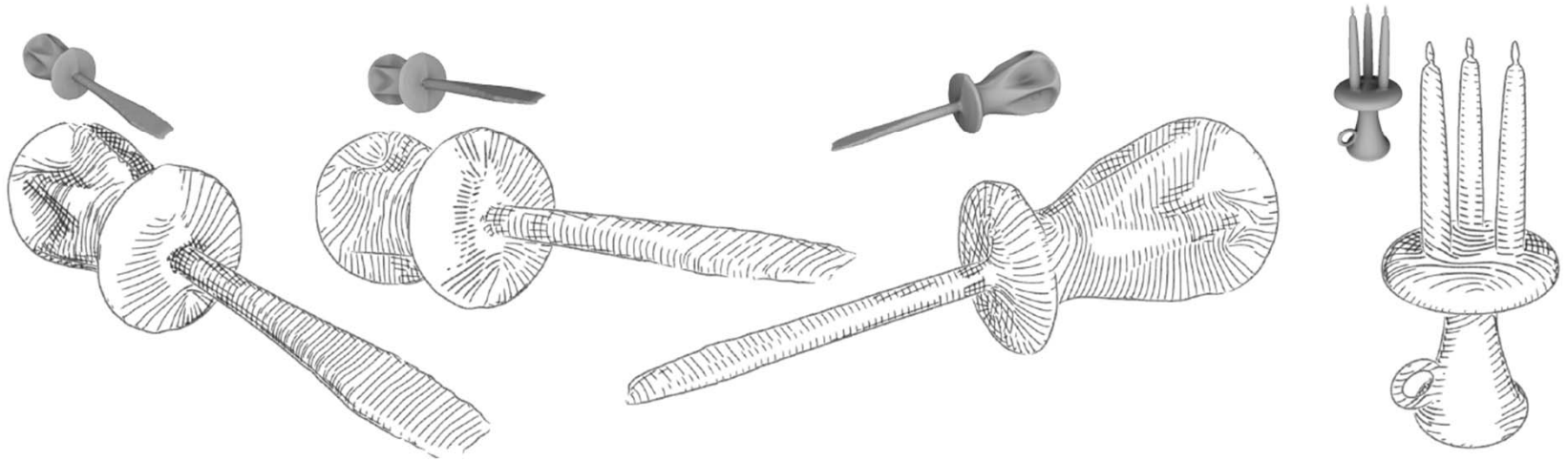
Artist’s illustration

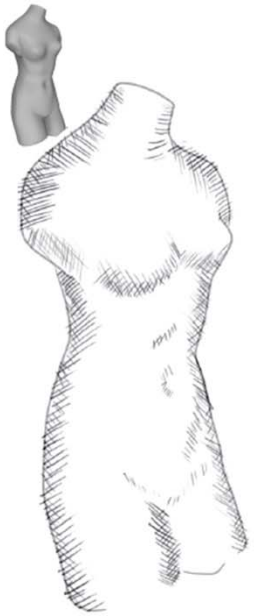


Artist's illustration

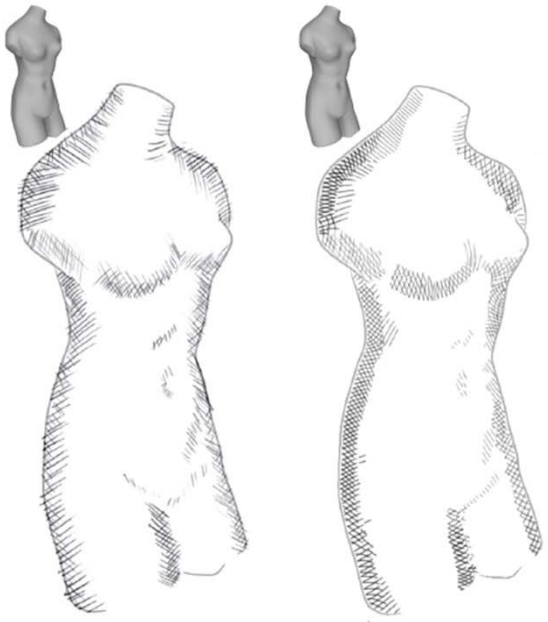


Artist's illustration

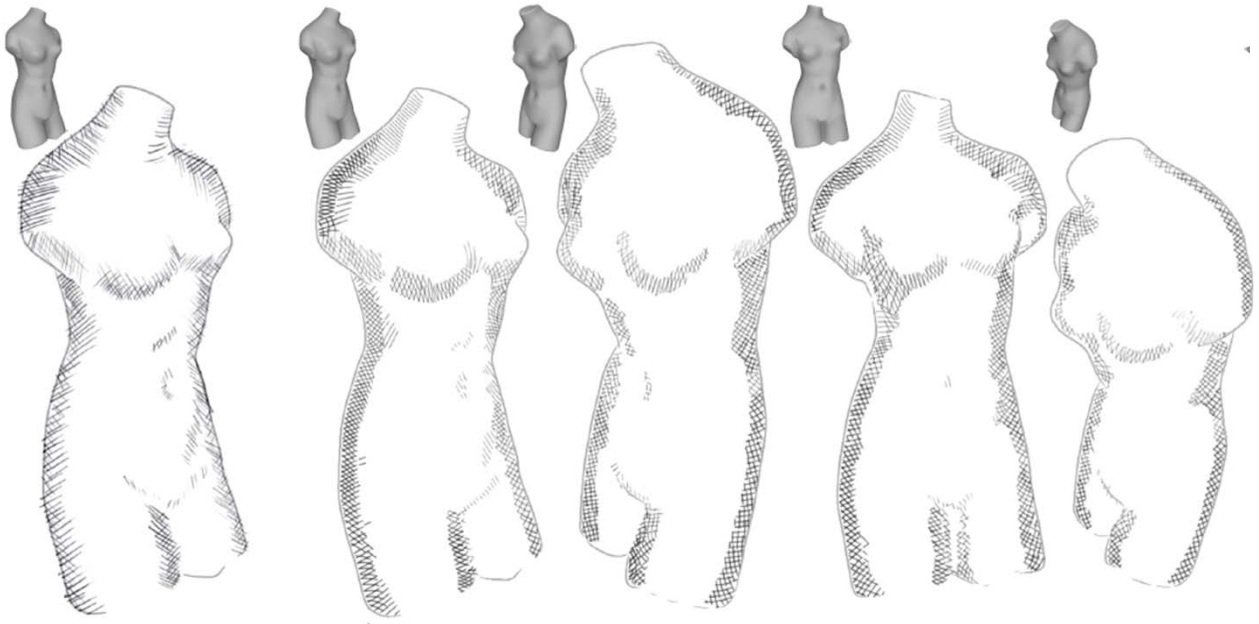




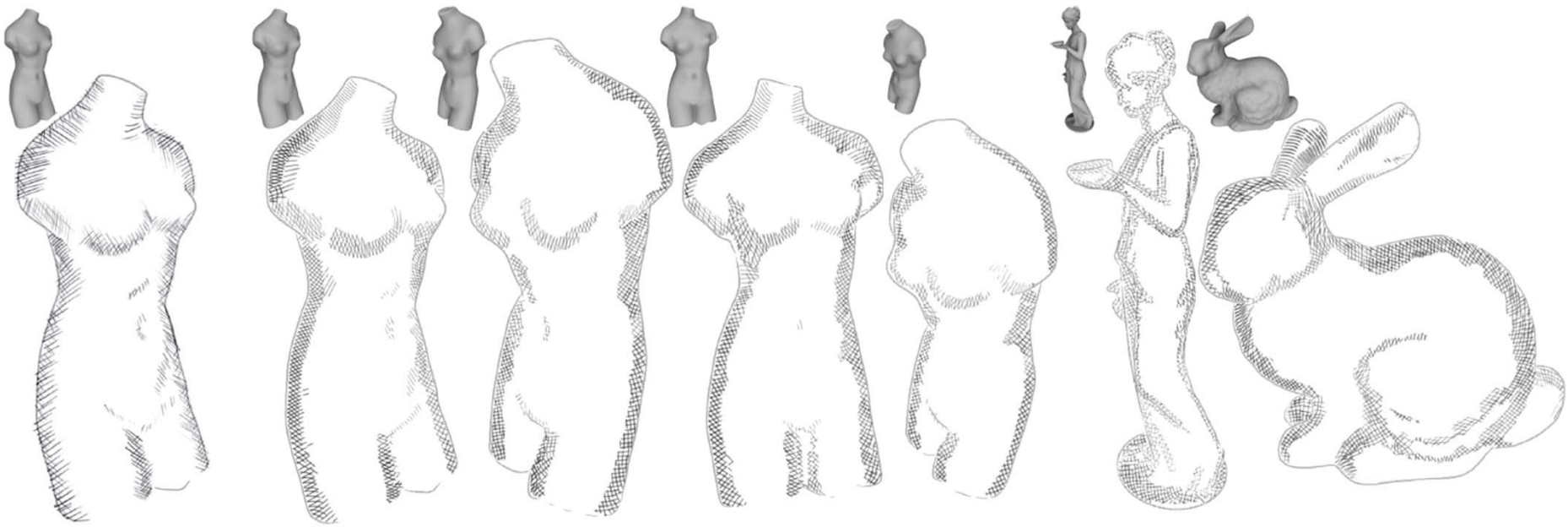
Artist's illustration



Artist's illustration

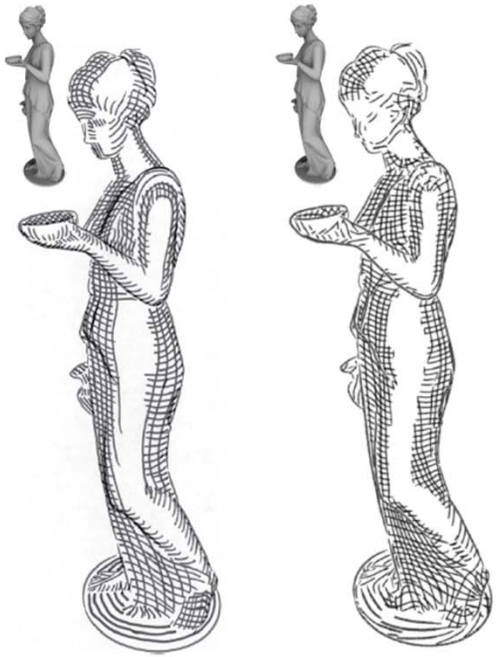


Artist's illustration

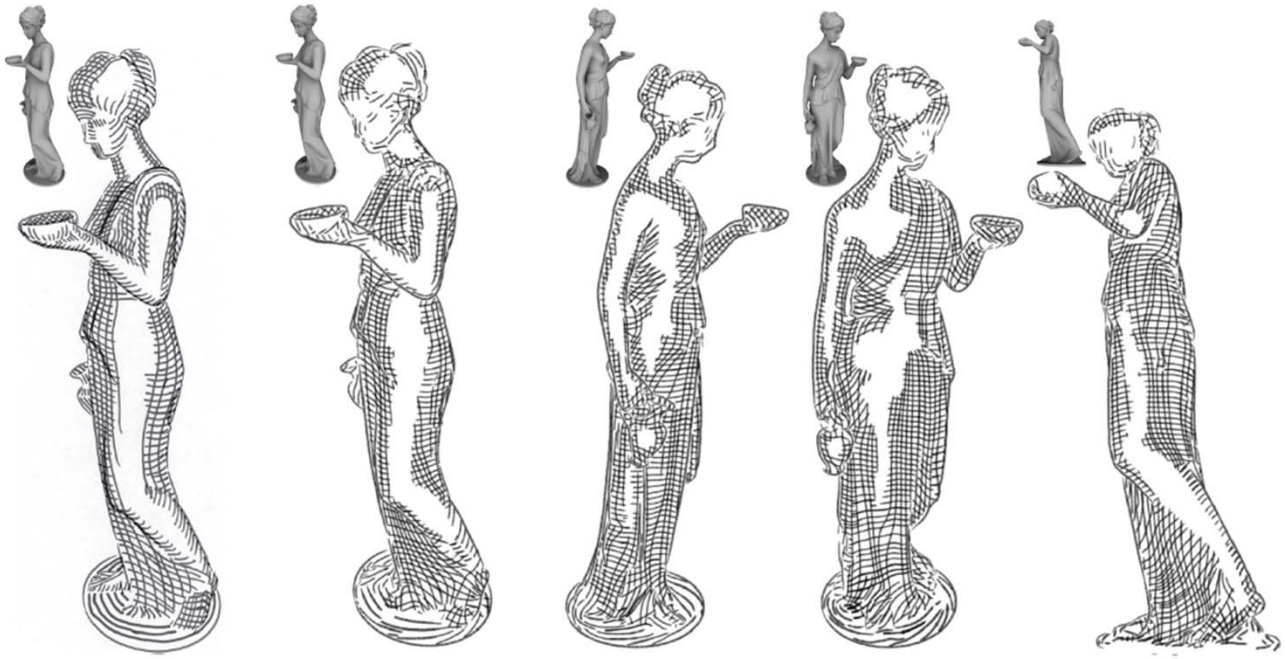


Artist's illustration

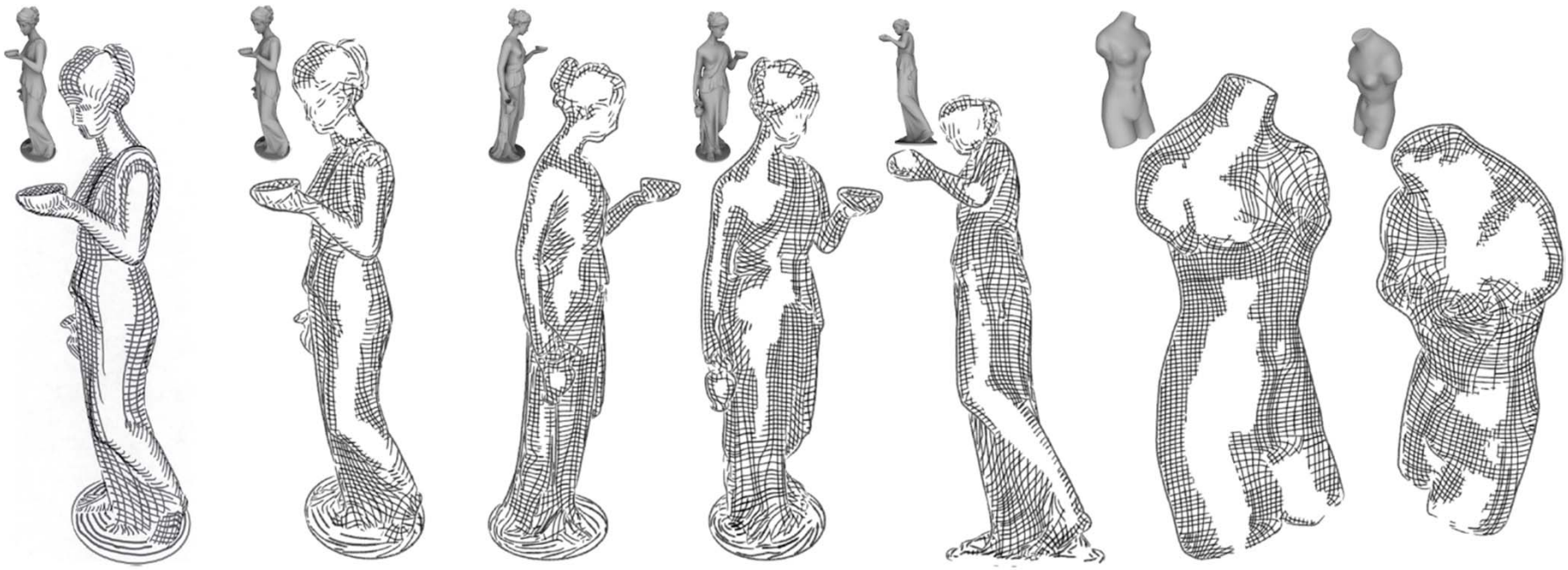
Artist's illustration



Artist's illustration



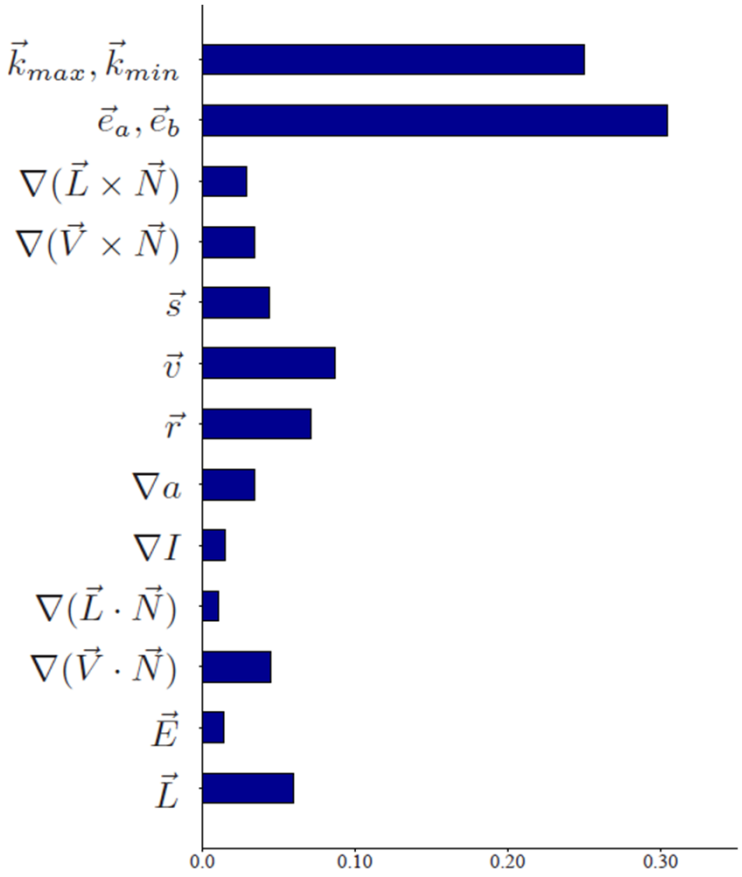
Artist's illustration



Artist's illustration

Analysis of features used

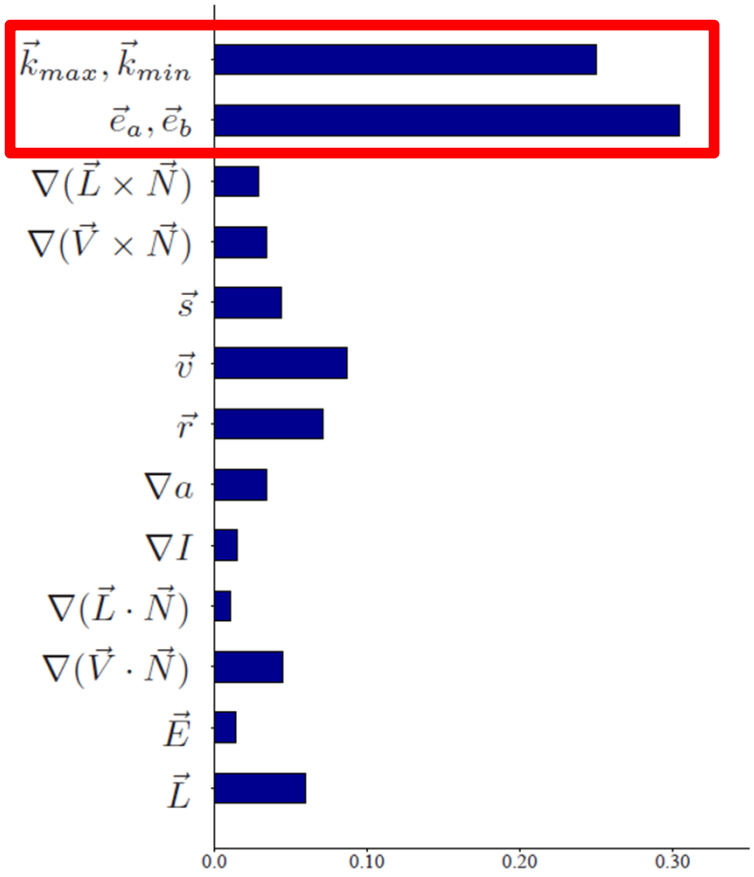
Orientation features:



Analysis of features used

Orientation features:

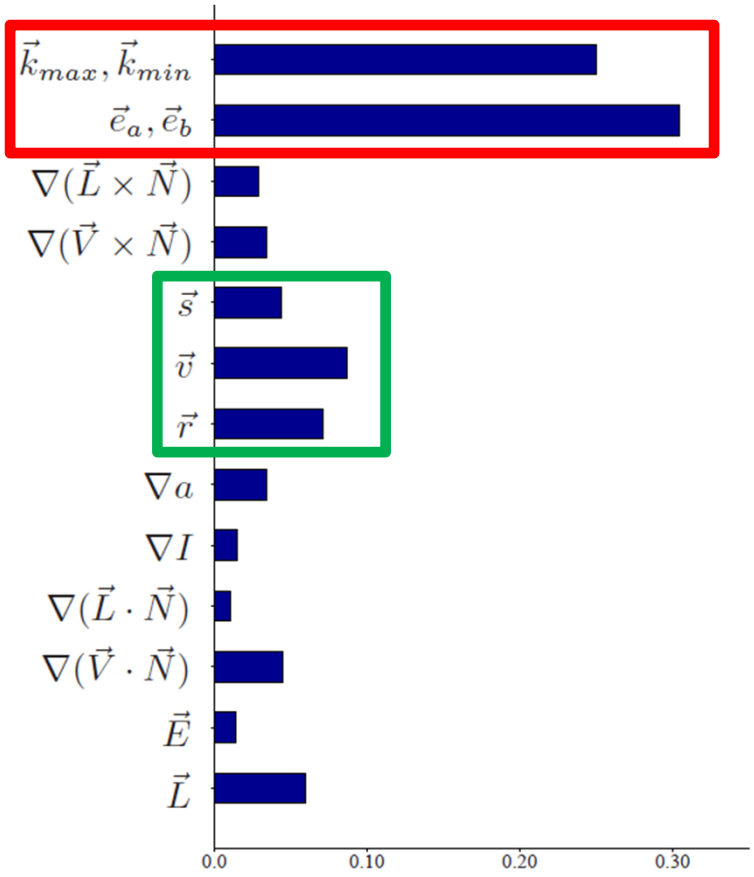
- Principal curvatures and local symmetry axes dominate



Analysis of features used

Orientation features:

- Principal curvatures and local symmetry axes dominate
- Also orientations aligned with feature lines are also important



Analysis of features used

Hatching level: *image intensity, shading features*

Stroke thickness: *shape descriptors, curvature, shading features, image gradients, location of feature lines, depth*

Spacing: *shape descriptors, curvature, derivatives of curvature, shading features*

Intensity: *shape descriptors, image intensity, shading features, depth, location of feature lines*

Length: *shape descriptors, curvature, radial curvature, shading feature, image intensity, image gradient*

Segment label: *shape descriptors*

Summary

- An algorithm that **learns hatching styles**

Summary

- An algorithm that **learns hatching styles**
- Learns from a **single drawing**

Summary

- An algorithm that **learns hatching styles**
- Learns from a **single drawing**
- Synthesizes hatching illustrations in the input artist's style **for novel views and shapes**

Limitations

- We do not always exactly match the artist's illustration - aspects of hatching style **are lost**

Limitations

- We do not always exactly match the artist's illustration - aspects of hatching style **are lost**
- Pre-processing stage relies on **thresholds** to robustly extract hatching properties.

Limitations

- We do not always exactly match the artist's illustration - aspects of hatching style **are lost**
- Pre-processing stage relies on **thresholds** to robustly extract hatching properties.
- Computation time **is large**
  (5h-10h for training, 0.5-1h for synthesis)

Future Work

- Analyze **larger set of drawings**

Future Work

- Analyze **larger set of drawings**
- Extend our framework to analyze other **forms of art**

Future Work

- Analyze **larger set of drawings**
- Extend our framework to analyze other **forms of art**
- Applications to **field design on surfaces**

Thank you!

Acknowledgements:

Seok-Hyung Bae, Patrick Coleman, Vikramaditya Dasgupta, Mark Hazen, Thomas Hendry, Olga Vesselova, Olga Veksler, Robert Kalnins, Philip Davidson, David Bourguignon, Xiaobai Chen, Aleksey Golovinskiy, Thomas Funkhouser, Andrea Tagliasacchi, Richard Zhang, Aim@Shape, VAKHUN, Cyberware repositories

NSERC CRSNG

Canada Foundation for Innovation

Fondation canadienne pour l'innovation

Ontario MINISTRY OF RESEARCH AND INNOVATION

KAUST