Motion Field Texture Synthesis

Chongyang Ma[1,3] Li-Yi Wei[2]
Baining Guo[3,1] Kun Zhou[4]

[1]Tsinghua University [2]Microsoft Research
[3]Microsoft Research Asia [4]Zhejiang University

Fluid animation

Physics-based simulation

- ☺ Physics and visual reality
- ☹ Computationally expensive

[Fedkiw et al. 2001]

Procedural turbulence

- ☺ Computational efficiency
- ☹ Limited generality

[Kim et al. 2008]

Our goal

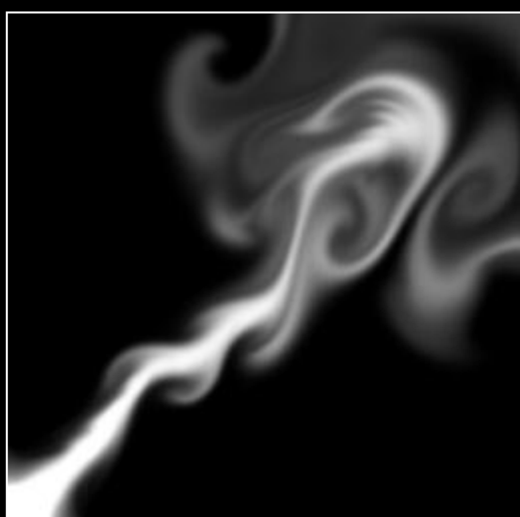
- General & flexible effects
  - may or may not be based on physics reality
- User friendly & controllability
  - only needs to supply exemplar
- Easy computation
  - fast
  - stable

Our approach

Example-based motion field texture synthesis

small-scale detail exemplar

large-scale coarse motion



input

synthesis

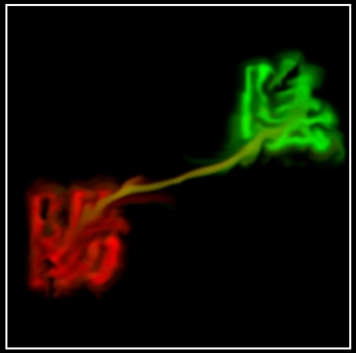
output

Why?

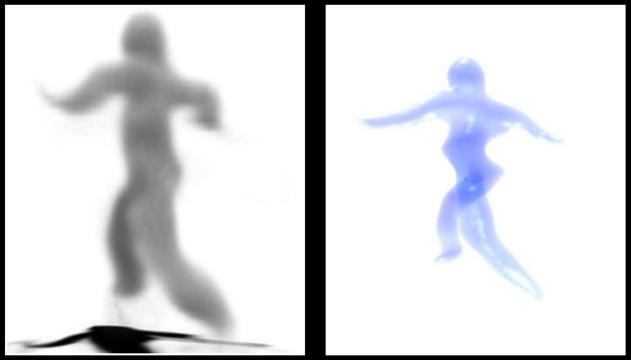
- Characteristics of many animation effects
  - A large-scale motion + repetitive small-scale details
- Users’ preference
  - Direct only the large-scale motion
  - Avoid tedious manual work for motion details
- Generality of data-driven methods
  - Example-based vs. procedural

Related work: animation

Control for large scale fluid motion



[Fattal and Lischinski 2004]

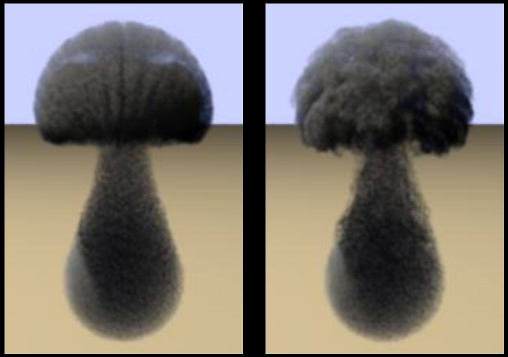


[McNamara et al. 2004]

Related work: animation

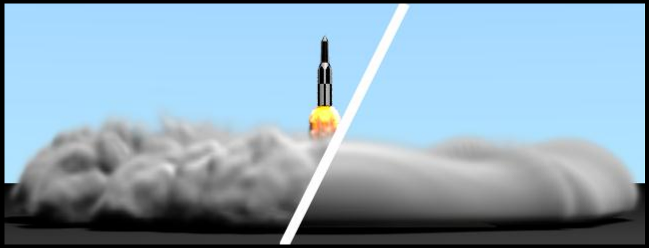
Synthesize detailed fluid motion

- Kolmogorov’s 5/3 power law
- Procedural noise



[Schechter and Bridson 2008]

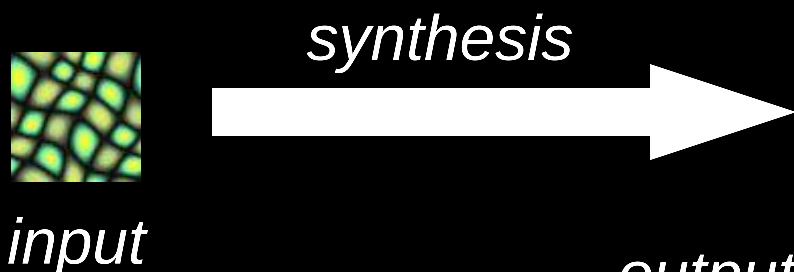
[Kim et al. 2008]

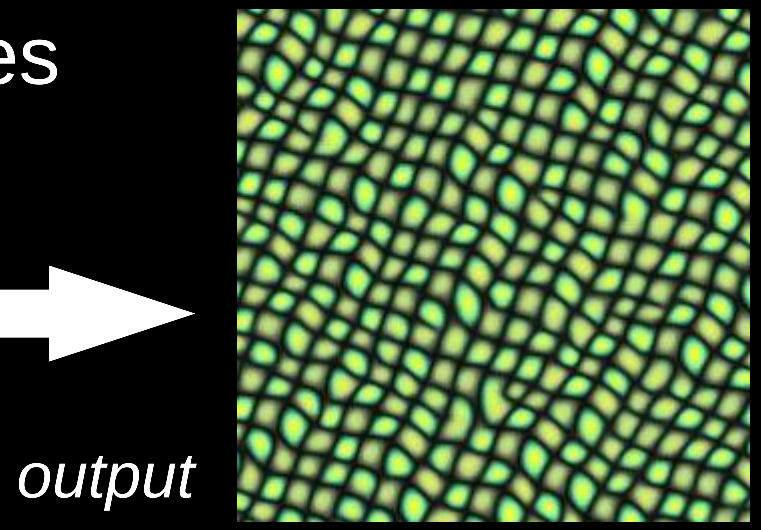


[Narain et al. 2008]

Related work: texture synthesis

Original for color images





Later extended for other data categories

- Video, geometry, character motion, etc
- See [Wei et al. 2009] for survey

Algorithm

Motion field

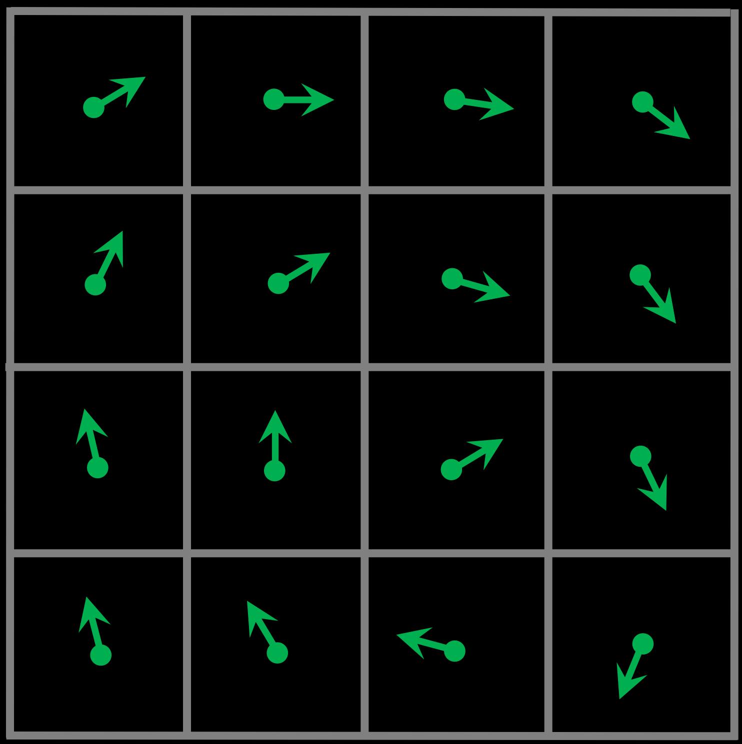
Velocity/displacement vectors defined over regular grids

2D motion field

$$\left(u\left(x,y\right),v\left(x,y\right)\right)$$

3D motion field

$$\left(u\left(x,y,z\right),v\left(x,y,z\right),w\left(x,y,z\right)\right)$$

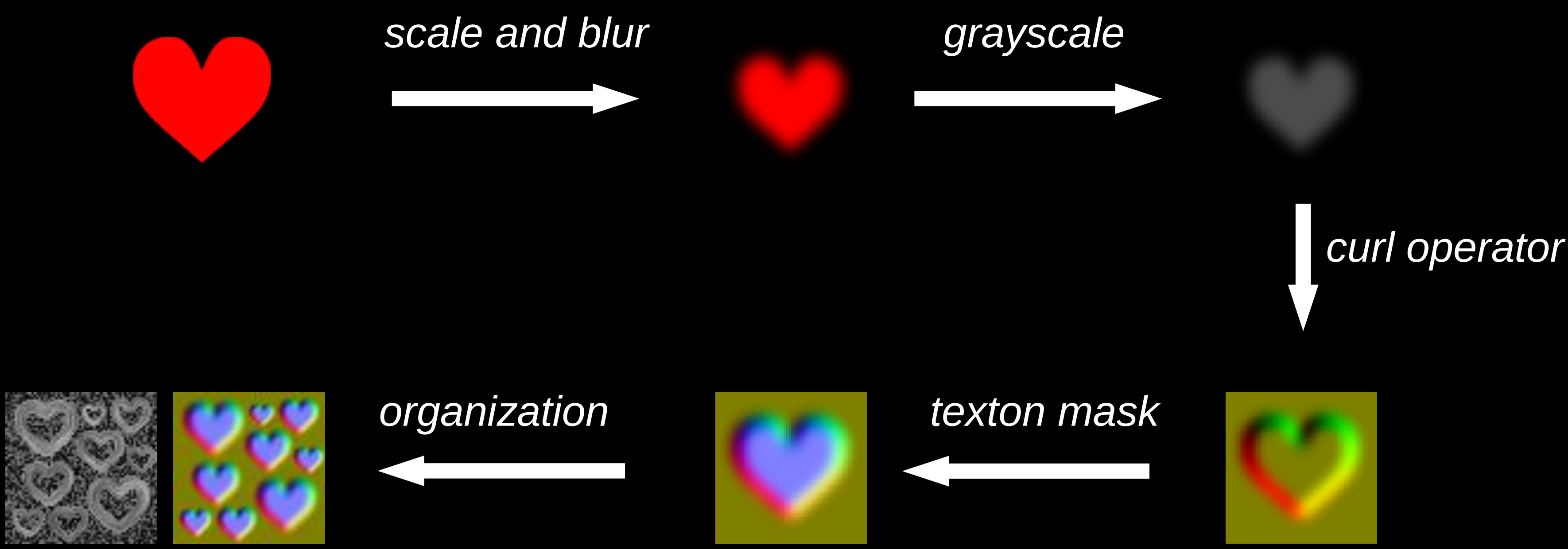


Data acquisition

- For both exemplar and large-scale motion
  - Physics-based simulation
  - Procedural flow
  - Captured motion data
  - Manual doodling
- To obtain interesting exemplars
  - Derive from images by curl operator $\nabla \times \Psi = \left(\frac{\partial \Psi}{\partial y}, -\frac{\partial \Psi}{\partial x} \right)$

Exemplar derivation

From color image to motion exemplar



Algorithm pipeline

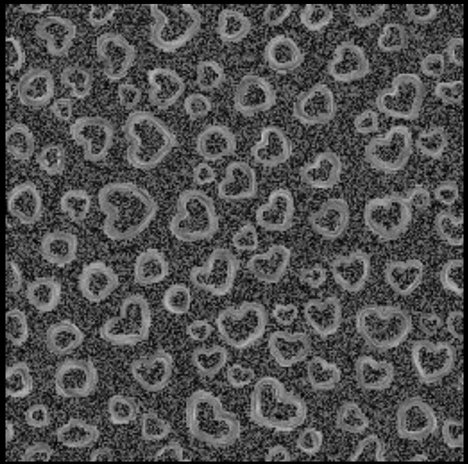
- Initialization
  - Random Initialization for the first frame
  - Advection from last frame
- Optimization-based synthesis of detail motion
- Combination with original motion
  - User-specified constant weight
- Final rendering

Algorithm pipeline

- Initialization
  - Random Initialization for the first frame
  - Advection from last frame
- Optimization-based synthesis of detail motion
- Combination with original motion
  - User-specified constant weight
- Final rendering

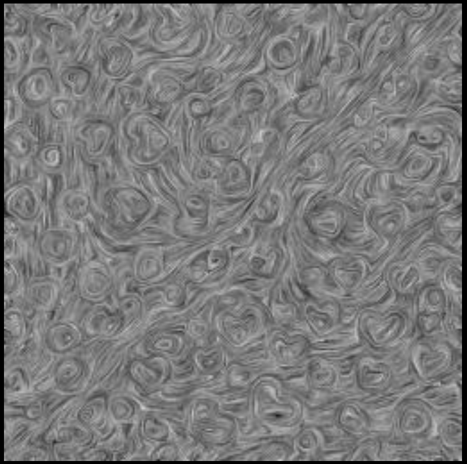
Algorithm pipeline

- Initialization
  - Random Initialization for the first frame
  - Advection from last frame
- Optimization-based synthesis of detail motion
- Combination with original motion
  - User-specified constant weight
- Final rendering



Algorithm pipeline

- Initialization
  - Random Initialization for the first frame
  - Advection from last frame
- Optimization-based synthesis of detail motion
- Combination with original motion
  - User-specified constant weight
- Final rendering



Algorithm pipeline

- Initialization
  - Random Initialization for the first frame
  - Advection from last frame
- Optimization-based synthesis of detail motion
- Combination with original motion
  - User-specified constant weight
- Final rendering

Algorithm pipeline

- Initialization
  - Random Initialization for the first frame
  - Advection from last frame
- Optimization-based synthesis of detail motion
- Combination with original motion
  - User-specified constant weight
- Final rendering

Synthesize motion details

Adapt texture optimization [Kwatra et al. 2005]

$$E_t\left(\mathbf{x};\left\{\mathbf{z}_p\right\}\right)=\sum_{p\in X^{\dagger}}\left|\mathbf{x}_p-\mathbf{z}_p\right|^2+O\left(\mathbf{x}\right)$$

neighborhood similarity additional constraints

Solver

Least squares [Kwatra et al. 2005]

Discrete solver [Han et al. 2006]

Motion vs. color

- Coordinate transformation
  - For natural appearance of motion details
- Vector projection
  - For 2D-to-3D synthesis

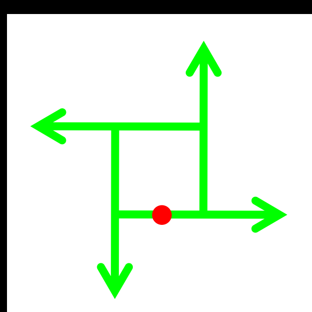
Coordinate transformation

diagonal large-scale motion

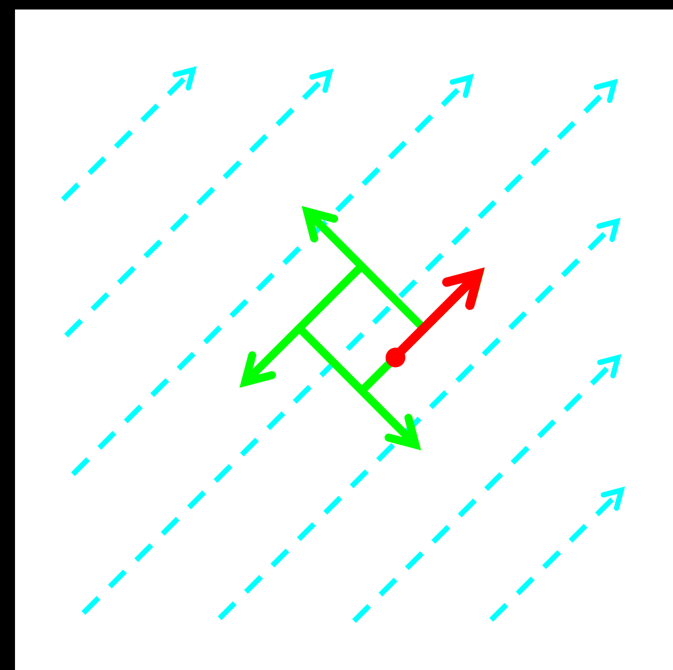
matched neighborhood

wrong value at the red point

correct value at the red point



exemplar (swirl pattern)



output

Vector projection

Solid color textures

Remain invariant with respect to different views

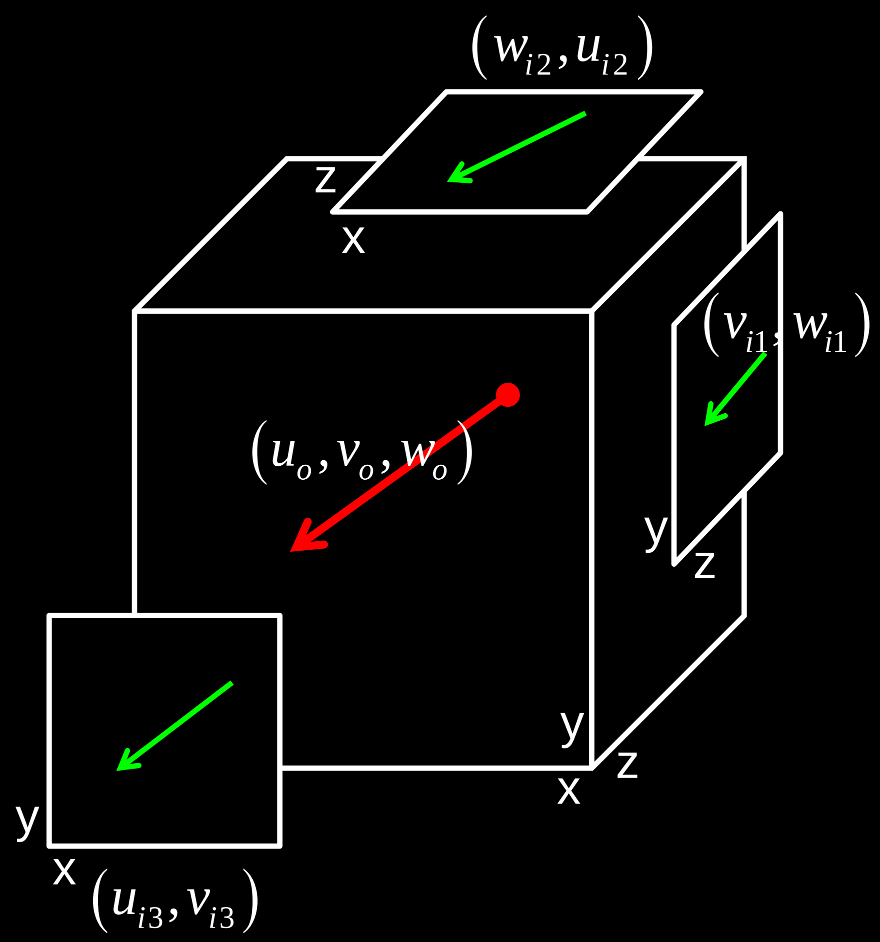
$$\left(r_i\left(x,y\right),g_i\left(x,y\right),b_i\left(x,y\right)\right)_{i=1,2,3}\rightarrow\left(r_o\left(x,y,z\right),g_o\left(x,y,z\right),b_o\left(x,y,z\right)\right)$$

3D motion vectors

Subject to vector projection

$$\left.\begin{matrix}\left(u_{i3}\left(x,y\right),v_{i3}\left(x,y\right)\right)\\ \left(v_{i1}\left(y,z\right),w_{i1}\left(y,z\right)\right)\\ \left(w_{i2}\left(z,x\right),u_{i2}\left(z,x\right)\right)\end{matrix}\right\}\rightarrow\left(u_o\left(x,y,z\right),v_o\left(x,y,z\right),w_o\left(x,y,z\right)\right)$$

Vector projection



A 2D input specifies only the corresponding projected components of a 3D output.

$$u_o = \frac{u_{i2} + u_{i3}}{2}$$

$$v_o = \frac{v_{i3} + v_{i1}}{2}$$

$$w_o = \frac{w_{i1} + w_{i2}}{2}$$

Boundary conditions

- Constrained texture synthesis
  - Normal component
  - Vector magnitude
- Additional energy term:

sub-vector component

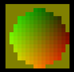
$$E_n(\mathbf{x}) = \sum_{p \in X} \lambda_p \left| \mathbf{x}_p^n - \mathbf{b}_p \right|^2$$

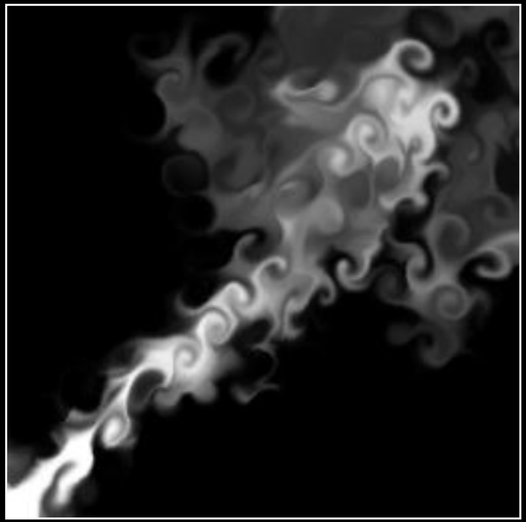
specified boundary condition

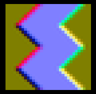
a Gaussian weighting function

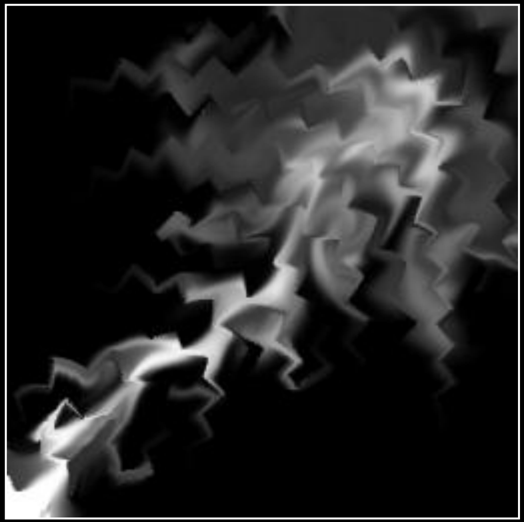
Results

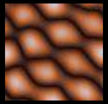
2D results

 circle

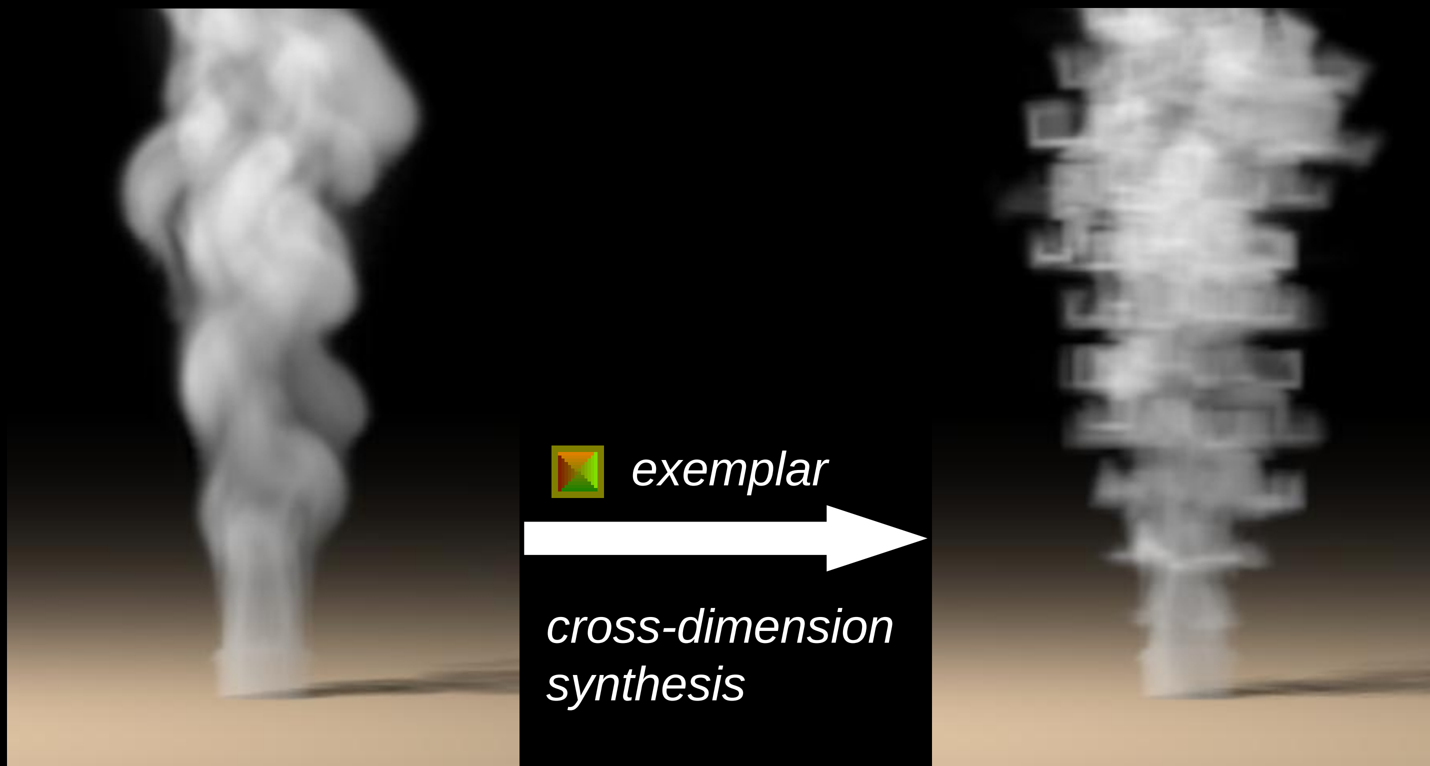


 lightning



 295

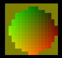
3D result



original *our result*

3D result

original



exemplar

cross-dimension synthesis

our result

Boundary conditions

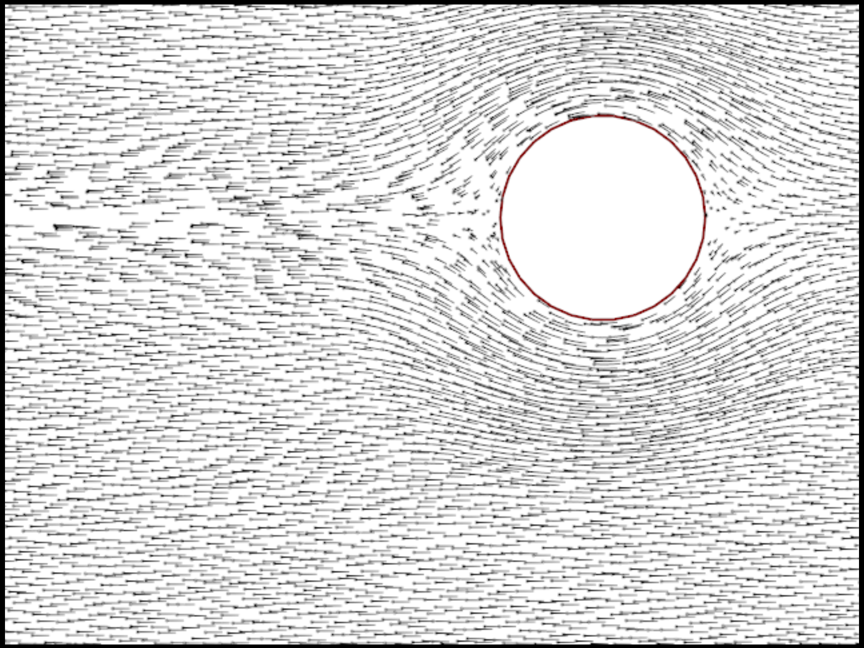
Constrain normal components with respect to obstacles

original

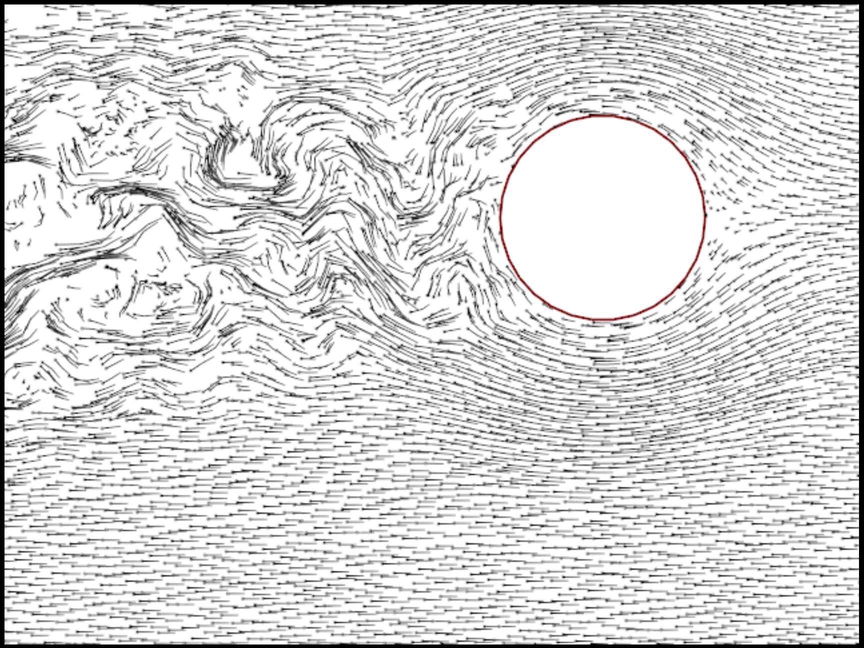
our result

Boundary conditions

Constrain magnitude within a triangular region



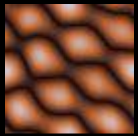
original



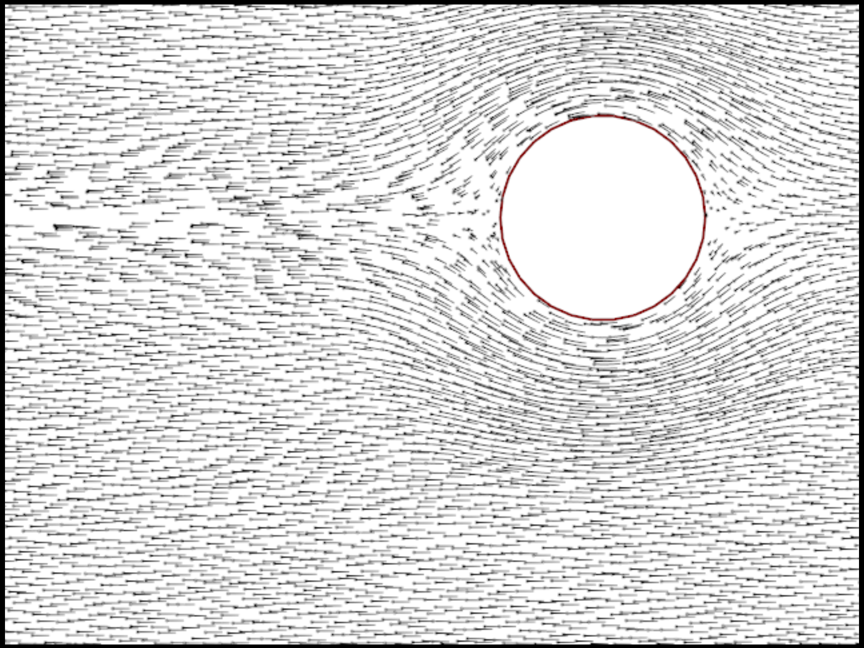
curl noise [Bridson et al.2007]

Boundary conditions

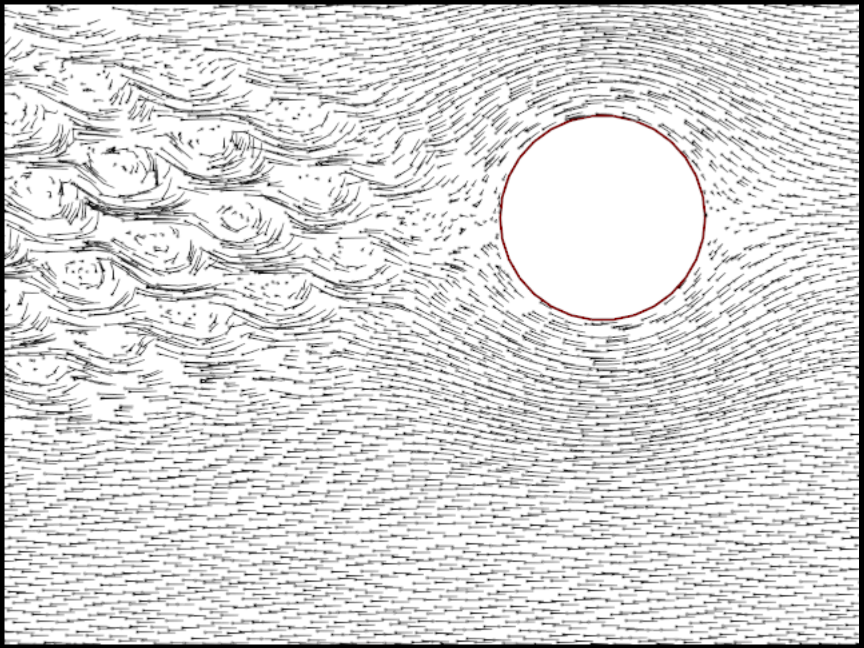
Constrain magnitude within a triangular region



exemplar



original

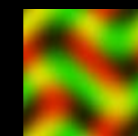


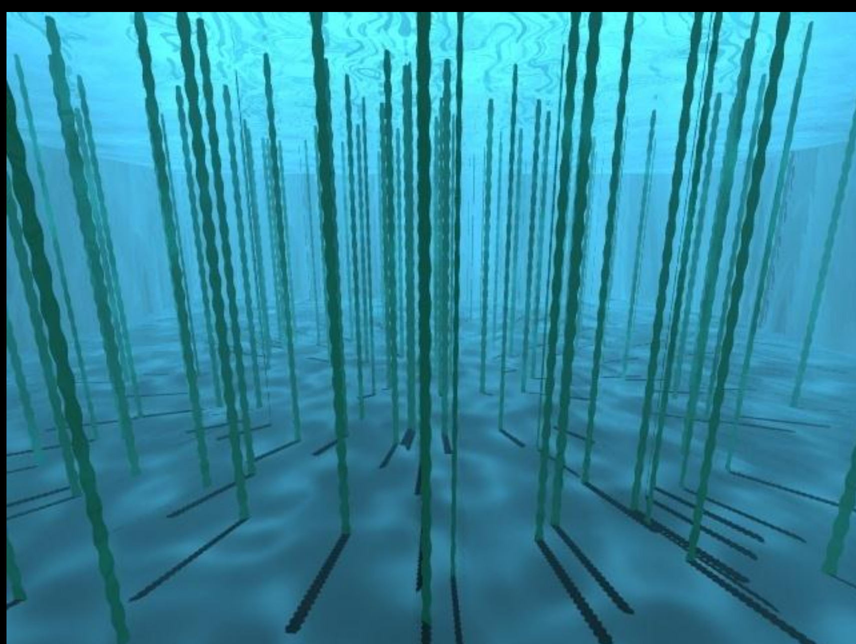
our result

Group motion

Vertex displacement for triangle meshes

Sinusoidal exemplar for all the three views





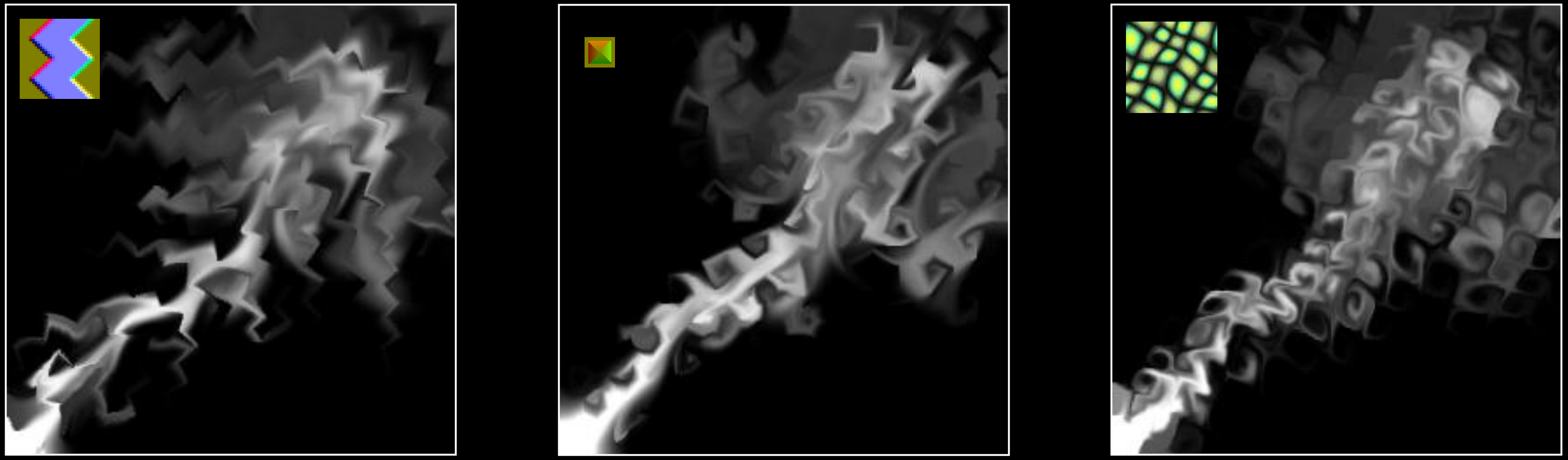
original

our result

Conclusion

Automatic synthesis of motion details from texture exemplars

Enables non-physics-based artistic effects



Future work

- Real-time application
  - Lazy evaluation [Dong et al. 2008]
- Lagrangian particle system
  - Regular pixels/voxels vs. irregular mesh vertices
- Other quantities
  - Agent positions [Kyriakou and Chrysanthou 2008]

Acknowledgements

Xin Tong

Xin Sun

Zhong Ren

Kangkang Yin

Qiming Hou

Minmin Gong

Su Wang

Kai Xu

Matt Scott

Bridson et al. 2007

Kim et al. 2008

NVIDIA

Thank you!