VECTOR FLUID: A VECTOR GRAPHICS DEPICTION OF SURFACE FLOW

(PUBLISHED 2010)

向量流體：表面流的向量影像表述

PRESENTER: XIN-WEI, LIN

DATE: 2024/09/18

OUTLINE

- Introduction
- Previous Work
- Method
- Results
- Applications
- Discussion & Conclusion

INTRODUCTION(1/3)

- Background
  - Less studies on NPR.
  - No research about Vector Graphics.
- Goal
  - 2D surface flow similar to marbling or suminagashi, using vector graphics.
- Limitations of Traditional Methods
  - too dissipative, blobby or complex.

INTRODUCTION(2/3)

Shortcomings	Solutions
More complicated shapes, more computation causes.	• Non-topological-changing consideration • Adaptive Refinement • GPU Enhancement

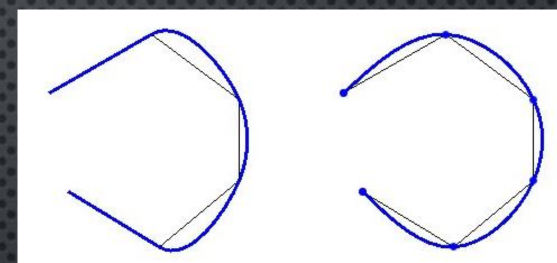
INTRODUCTION(3/3)

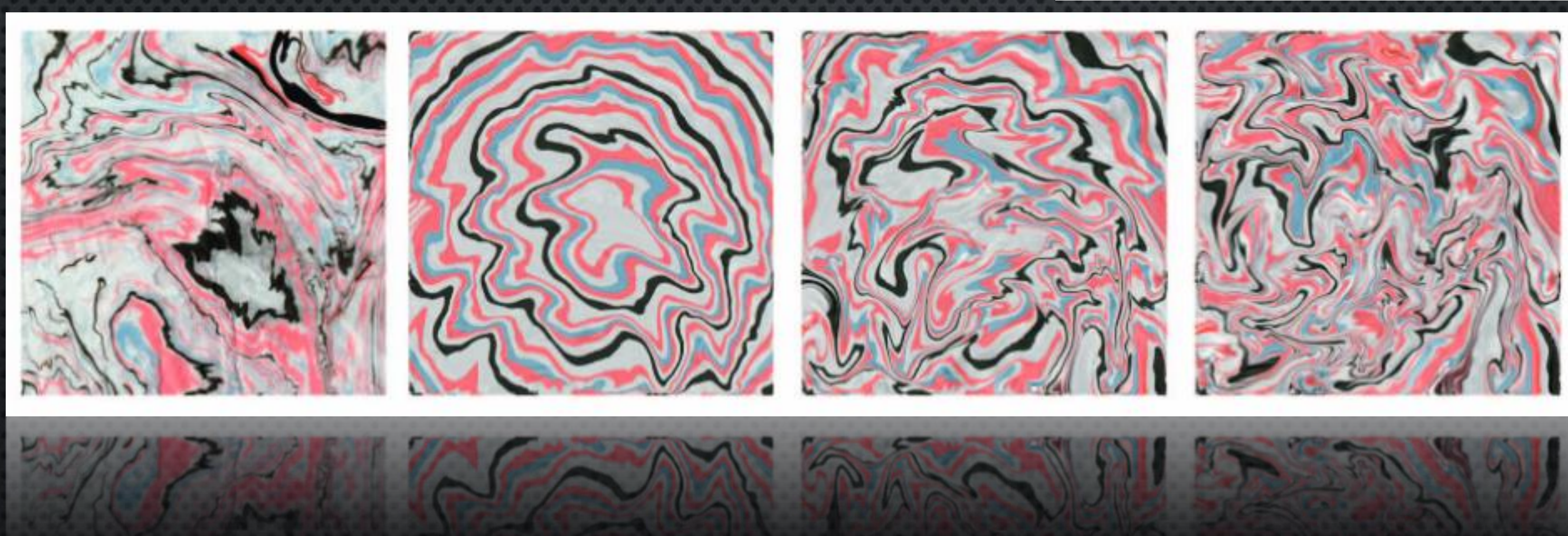
- Key components of algorithm
  - Ignorance of topological changes;
  - Adaptive Refinement;
  - Shapes are depicted with vector graphics

PREVIOUS WORK (1/2) - ARTISTIC EXPRESSION

- DIGITAL MARBLING: A MULTISCALE FLUID MODEL

  ACAR AND BOULANGER (2006)





PREVIOUS WORK (2/2) - SURFACE TRACKING

- FAST AND ROBUST TRACKING OF FLUID SURFACES

  MÜLLER ET AL. (2003)

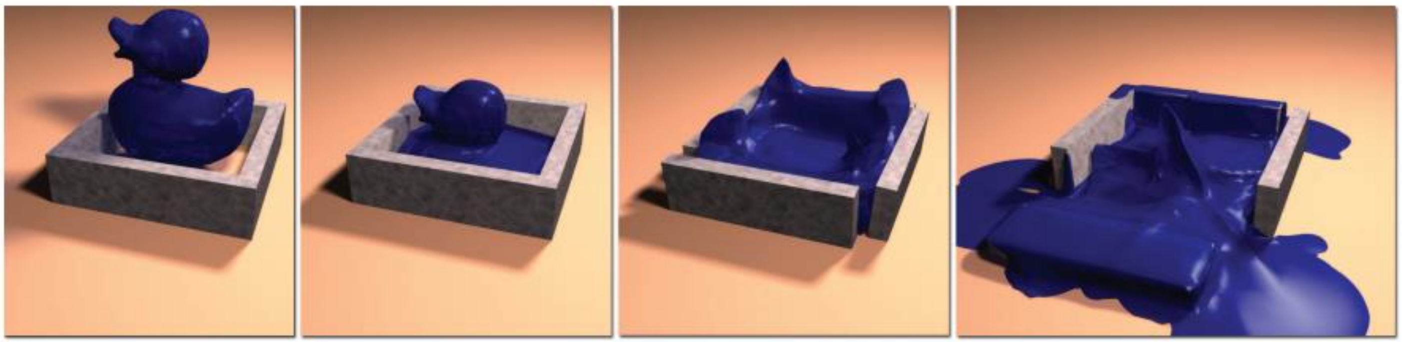


Figure 12: *Unbounded Eulerian fluid simulation showing thin sheets, sharp features and shallow puddles rendered off-line.*

Figure 13: *Two way interaction with rigid bodies.*

METHOD (1/7) - GENERALLY SPEAKING

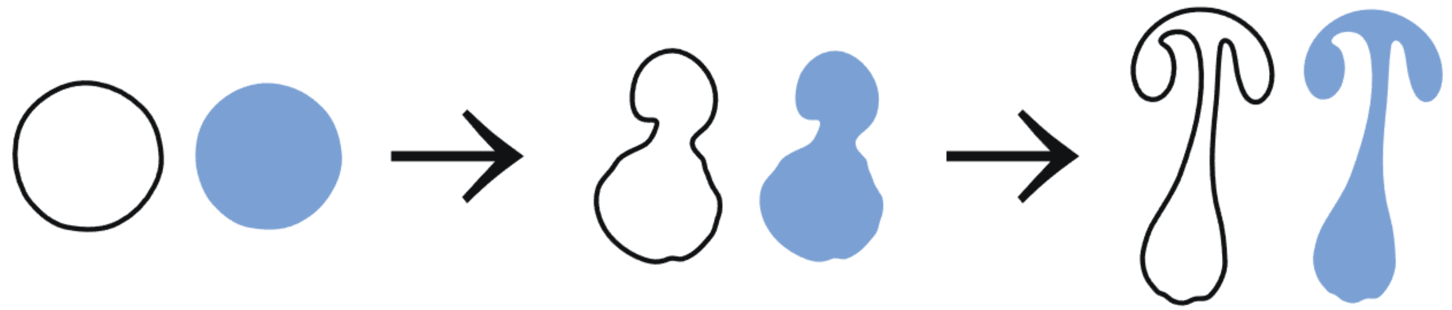


Figure 2: *Workflow of our method: We first place closed contours of a painted region in the fluid field, then advect or stretch them along the fluid flow. The rendering of the region is done just as a concave polygon is rendered.*

METHOD (2/7) - CONTOUR ADVECTION(1/2)

- Simplicity
  - finite differential grid
  - semi Lagrangian advection method
- Contour
  - A sequence of discrete points
  - Advect them in the Lagrangian manner
- prevent collisions
  - fourth order accuracy Runge-Kutta scheme

METHOD (3/7) - CONTOUR ADVECTION(2/2)

- SCHEME FUNCTION:

$$v^{t+\Delta t} = \phi(\Delta t, v^t, u^t)$$

(IF DISTANCE > d):

$$v_{new} = \phi(\Delta t, \frac{1}{2}\left(v_0^{t-\Delta t} + v_1^{t-\Delta t}\right), u^{t-\Delta t})$$

METHOD (4/7) - RENDERING AND EXPORT

Vector Graphics	regular concave polygon
Rendering	OpenGL
For Contour (p, q)	draw triangle(0,p,q) (inverting the existing values between 0 and 1)
Solid Region	filled with value 1

METHOD (5/7) - **ADAPTIVE REFINEMENT**(1/2)

- WHY ADAPTATION NEEDED?
  - EXCESSIVE COMPUTATION
- INSERT MORE POINTS ON **TIGHTLY CURVE**

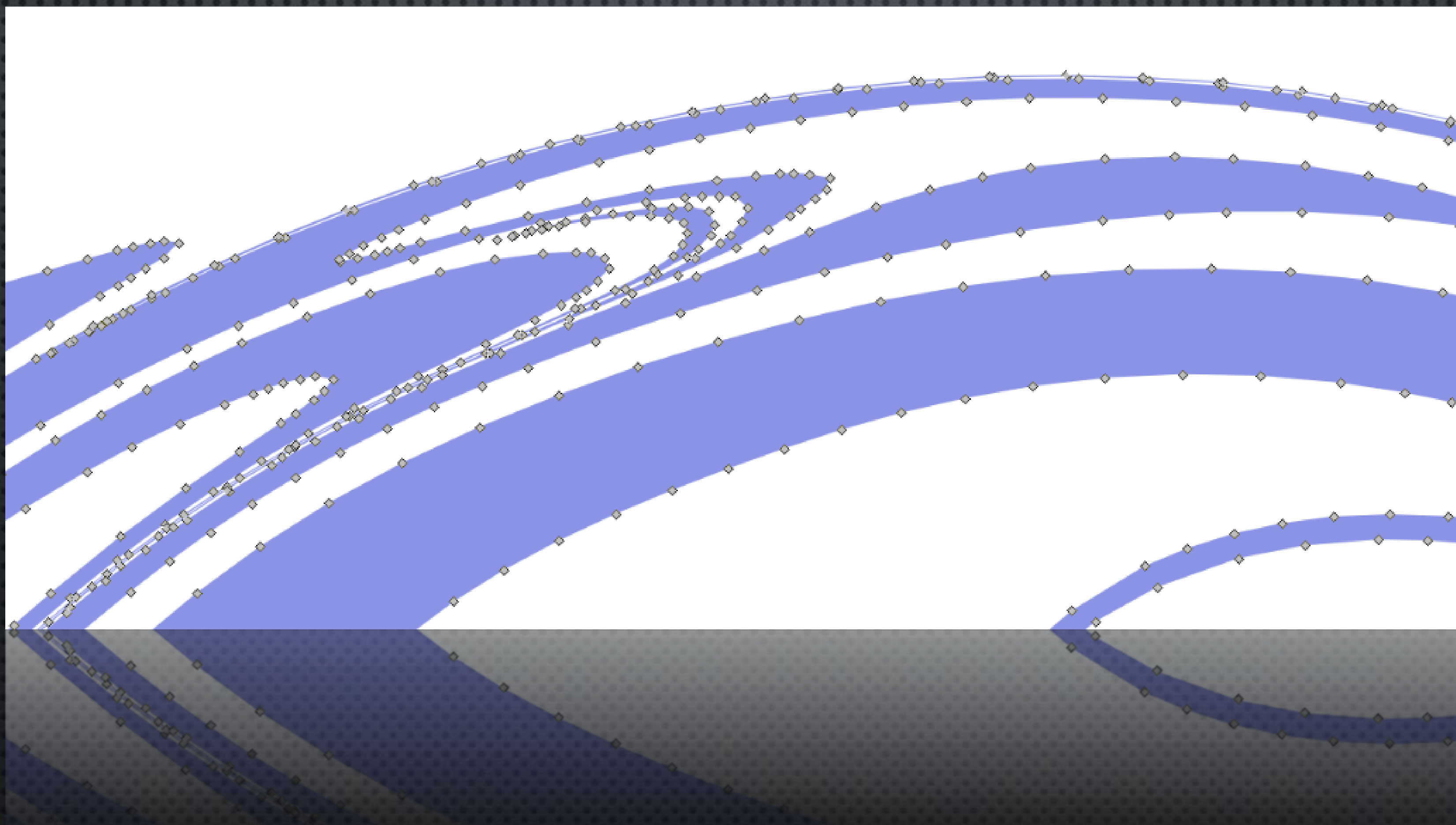
$$d_i = d_{max} c_i^{curvature} c_i^{turbulence} + \varepsilon \quad (1)$$

$$c_i^{curvature} = exp(-|\kappa|) \quad (2)$$

$$c_i^{turbulence} = exp(-|\nabla \times u(v_i)|) \quad (3)$$

$$d_i \leftarrow \sum_{n=-\omega}^{\omega} G(\alpha, n) d_{i+n} \text{ , } \omega = k d_{max}^{-1} \quad (4)$$

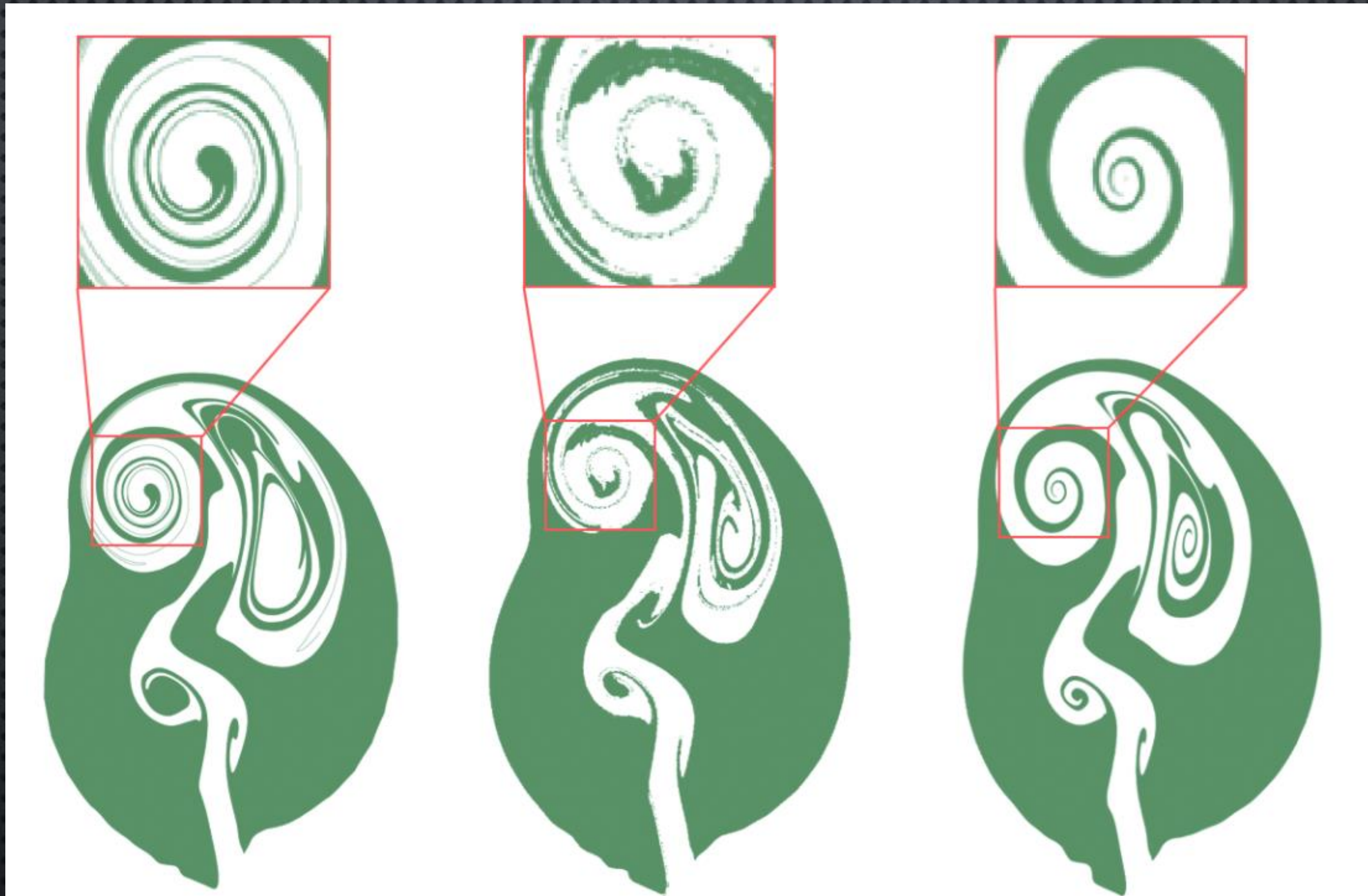
METHOD (6/7) - **ADAPTIVE REFINEMENT**(2/2)



METHOD (7/7) – GPU ACCELERATION

- Contour Advection
  - Task positions within each kernel are advanced in parallel.
- Contour Subdivision
  - The distances between tasks are checked, and subdivision or collapsing is performed if necessary.
- Task Diffusion
  - Pairs of kernels are randomly selected, and the number of tasks is evenly distributed between them.

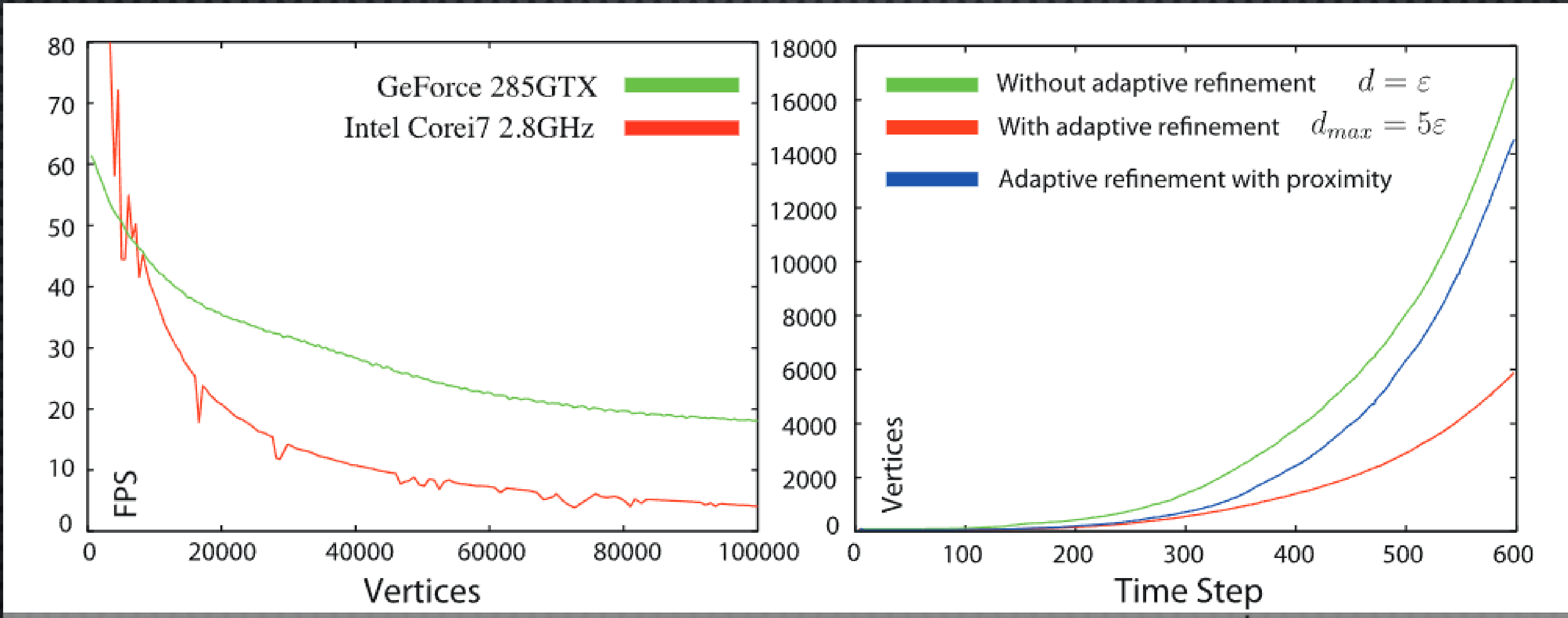
RESULTS(1/2) – METHOD COMPARISON



Our method with 11126 Vertices | Particle level-set method | Eulerian Advection with Acar's Edge Accentuation

RESULTS(2/2) - PERFORMANCE

- LIMITATIONS:
  - A TRACTABLE NUMBER OF TIME STEPS IS LIMITED TO AROUND A FEW THOUSAND.
  - CONTOURS ARE INTERACTIVELY TRACKED UP TO AROUND 100,000 VERTICES.



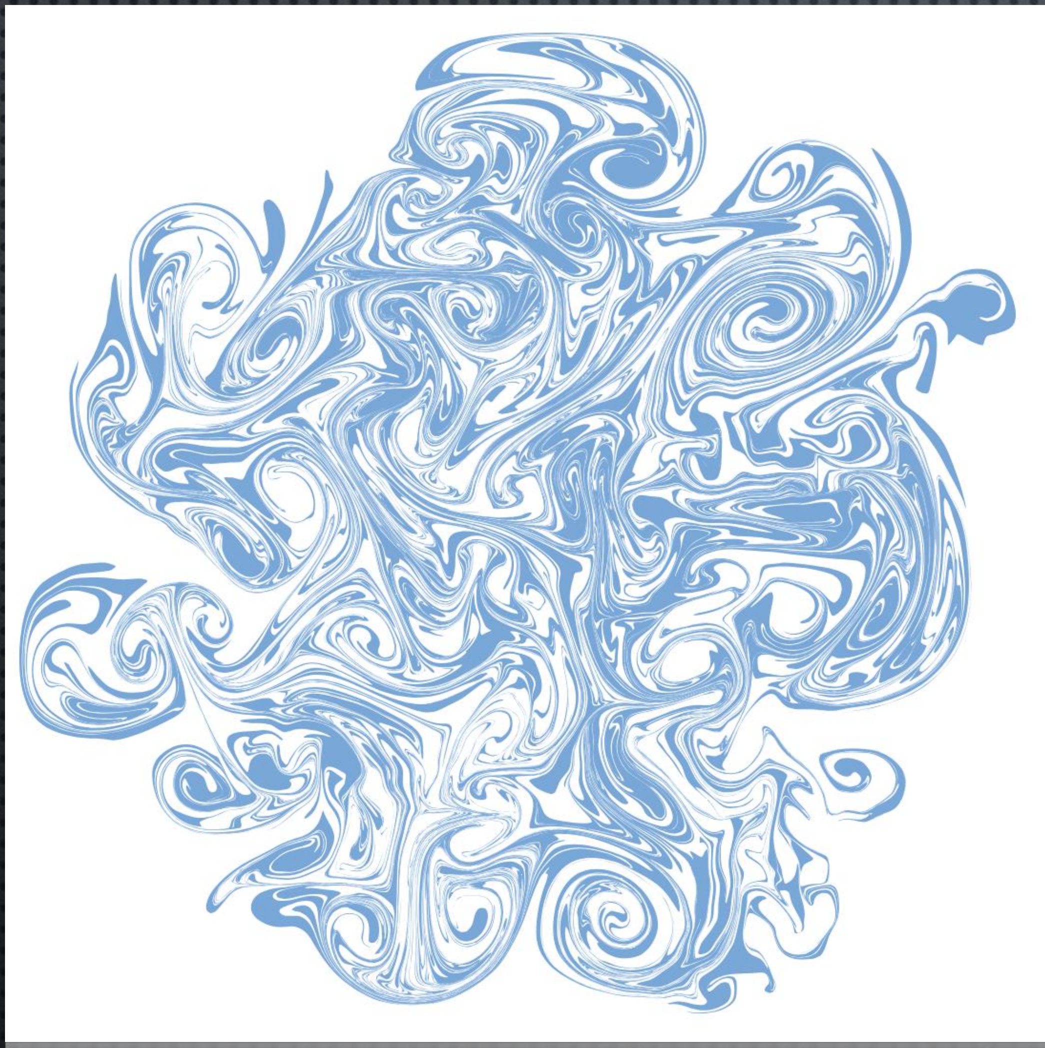
APPLICATIONS(1/5) - PROPERTIES

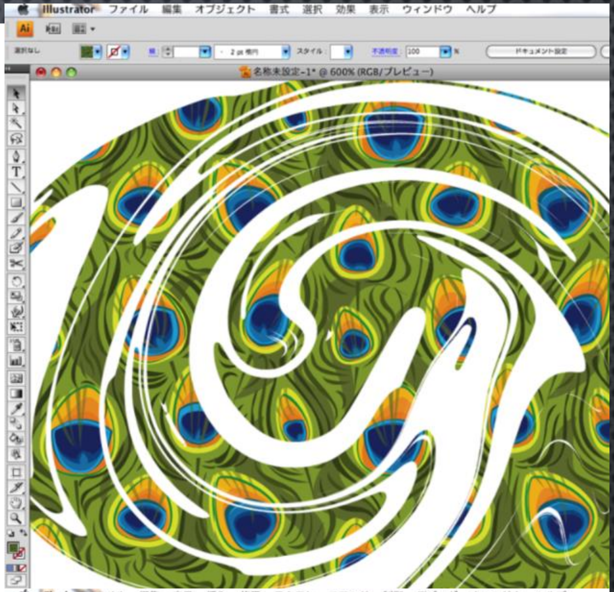
- HARDWARE & SOFTWARE

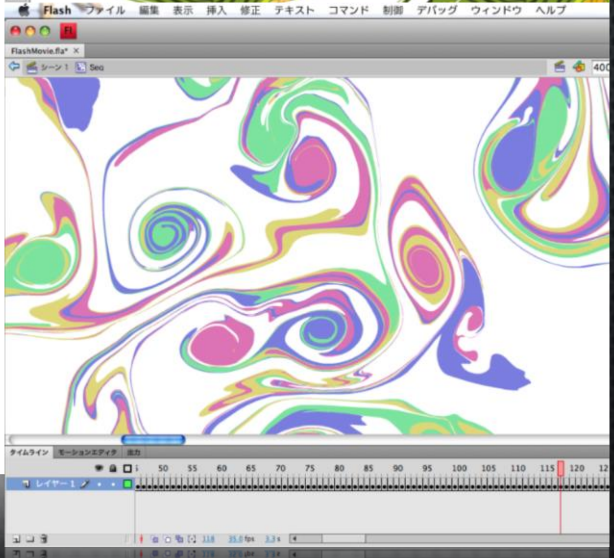
CPU	Intel Core i7 2.8GHz
GPU	Nvidia GTX 285(1024MB gRAM)
Operating System	Linux

- USERS ARE ALLOWED TO DROP PIGMENTS ON THE SURFACE OF WATER AND **DISTURB IT SIMPLY BY MOUSE DRAGGING.**

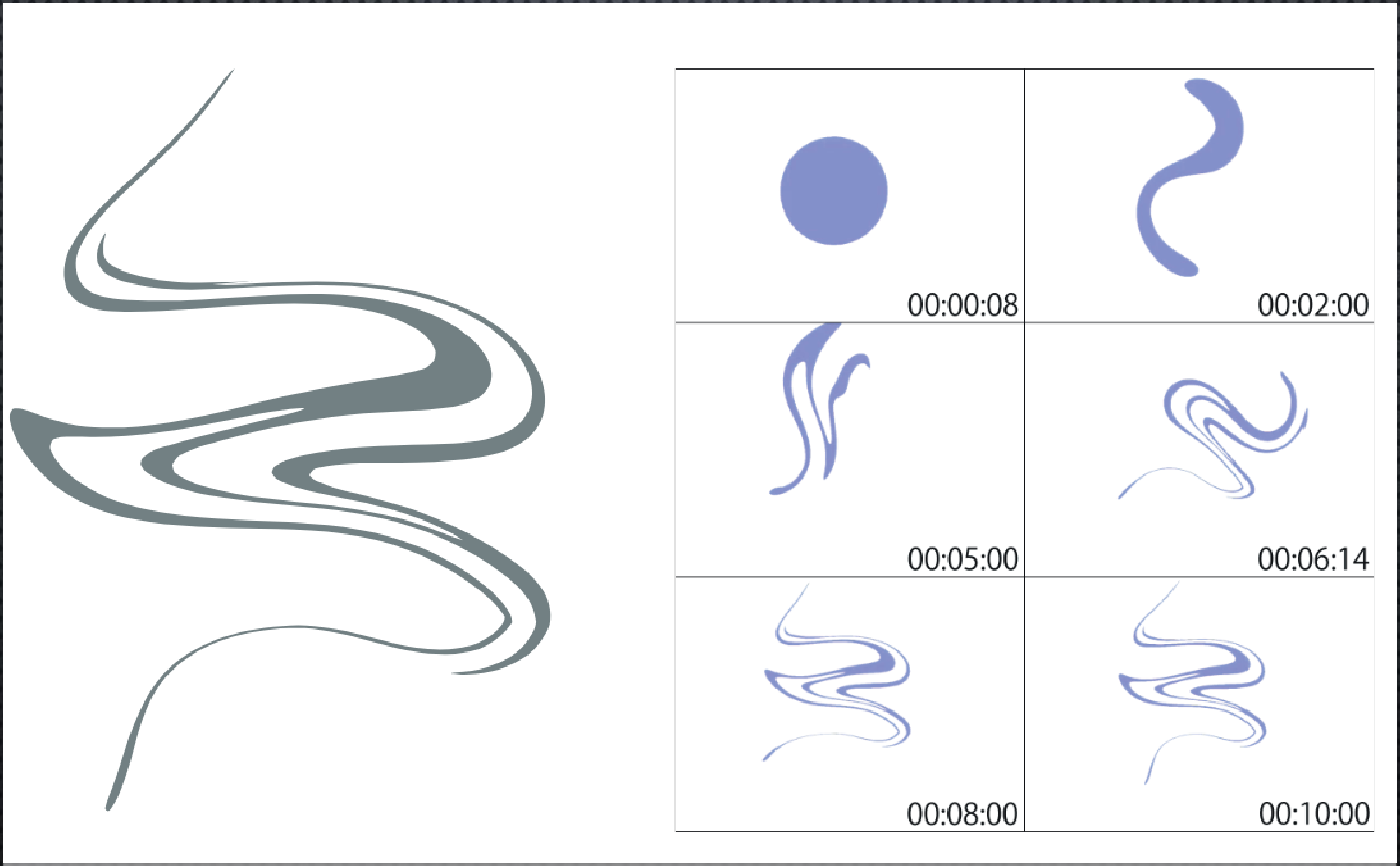
APPLICATIONS(2/5) - MARBLING AND SUMI-NAGASHI





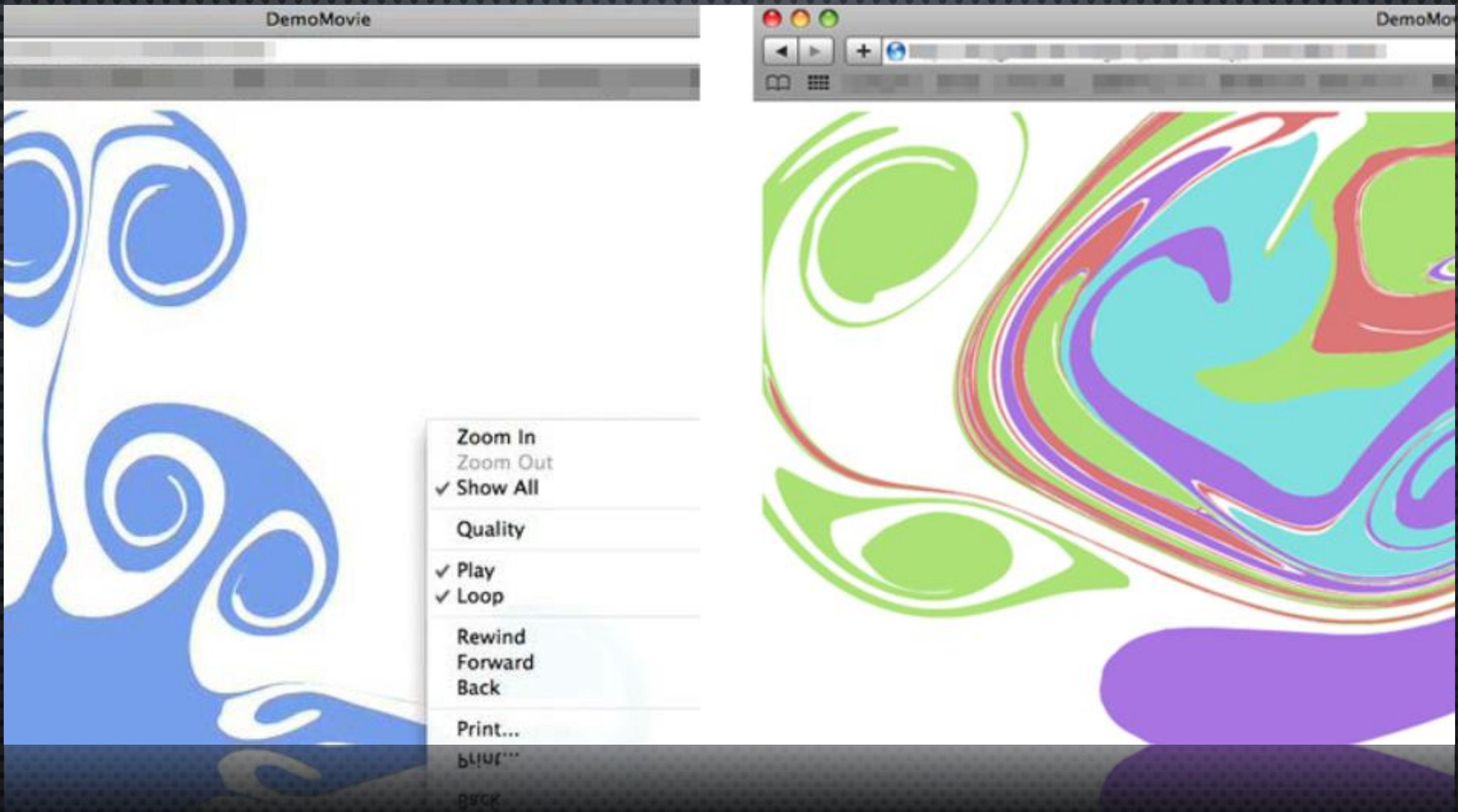


APPLICATIONS(3/5) – SHAPE DESIGNING TOOL

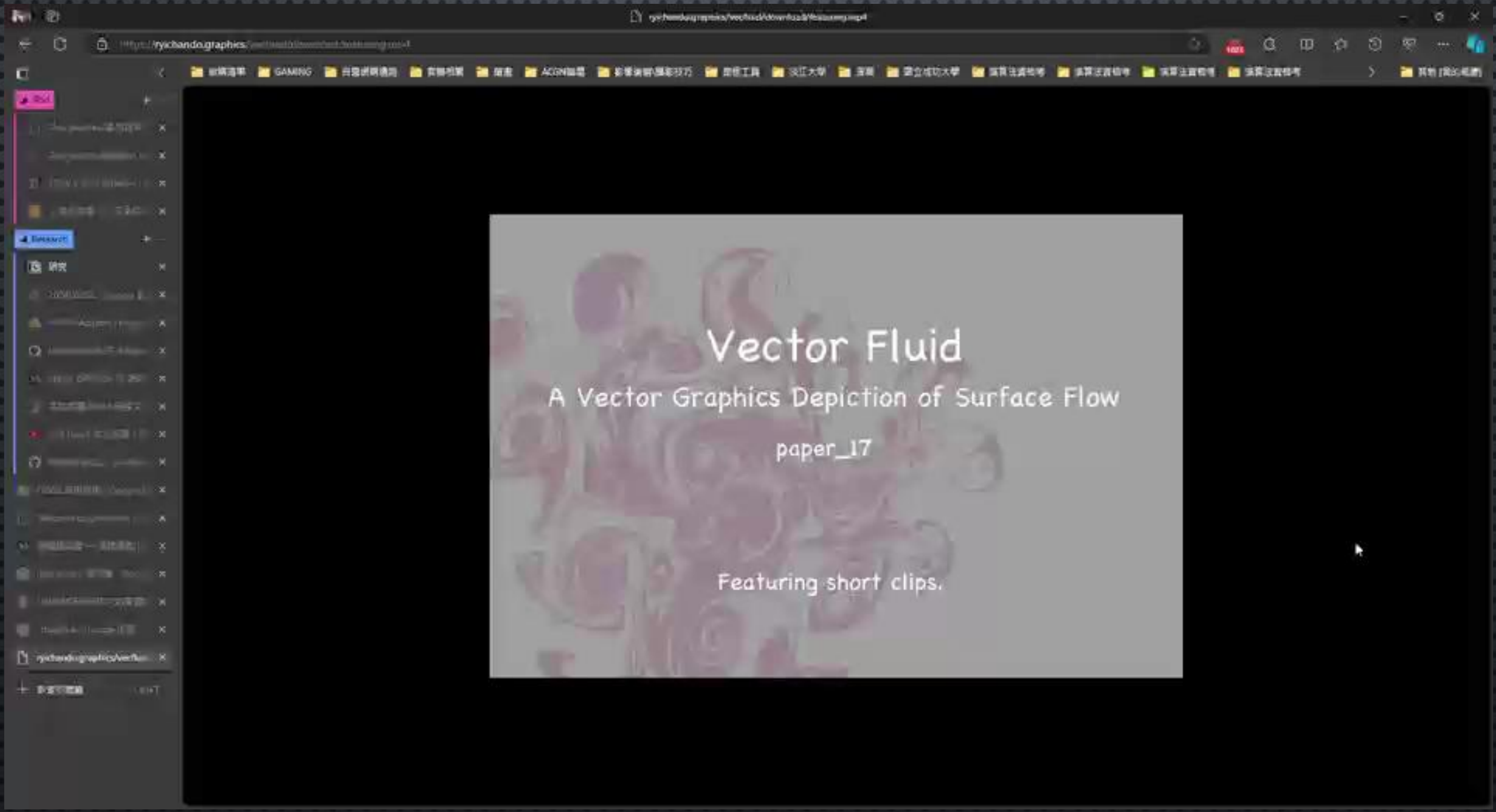


APPLICATIONS (4/5) – TARGET DRIVEN DESIGN

APPLICATIONS (5/5) - FLASH ANIMATION



DEMO


Vector Fluid
A Vector Graphics Depiction of Surface Flow
paper_17
Featuring short clips.

Discussion & Conclusion

- Topology of concave polygons remains unchanged during advection.
- Front-tracking algorithm with adaptive refinement.
- Ported the algorithm to GPU.
- Applicable to marbling, shape design, animation, and target-driven design.
- Future work: enhance method for more complex scenes.

THAT’S ALL
THANK YOU!