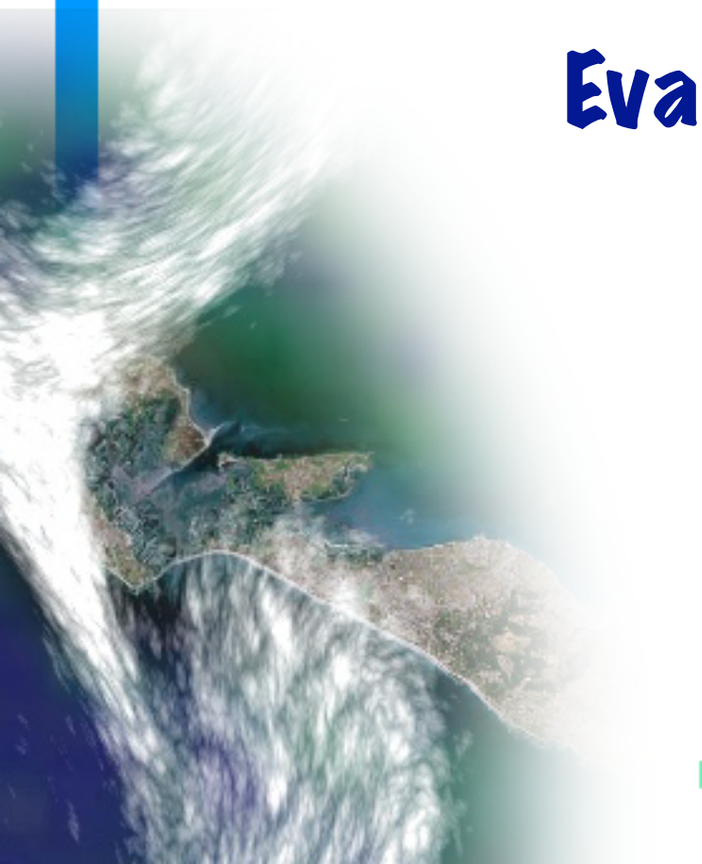
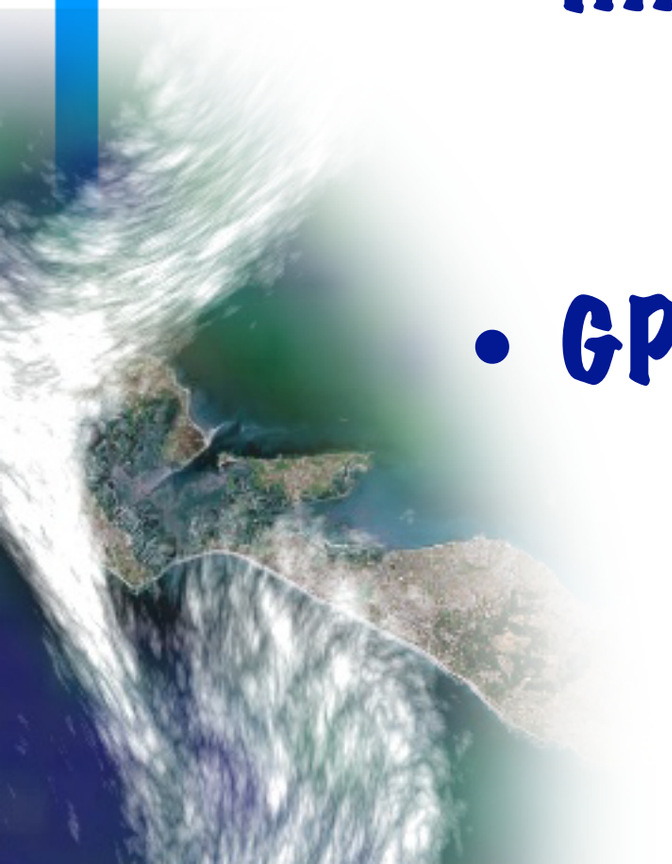
# Real-time flow-noise

Aymeric Augustin
Evasion workgroup - June 8th, 2006



# Real-time flow-noise

- State of the art
- Improved animated Perlin noise
- GPU implementation

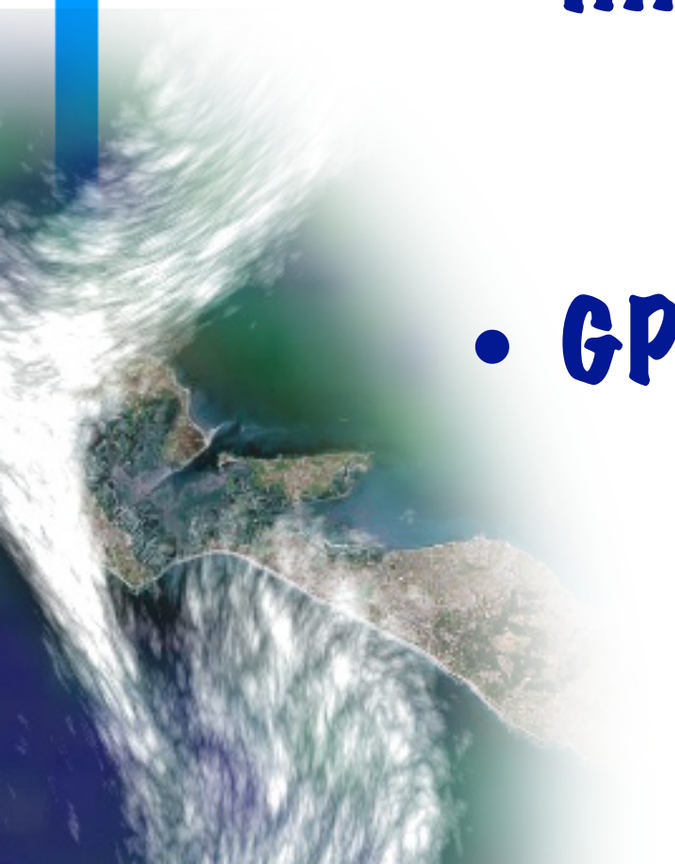


# Real-time flow-noise

- State of the art

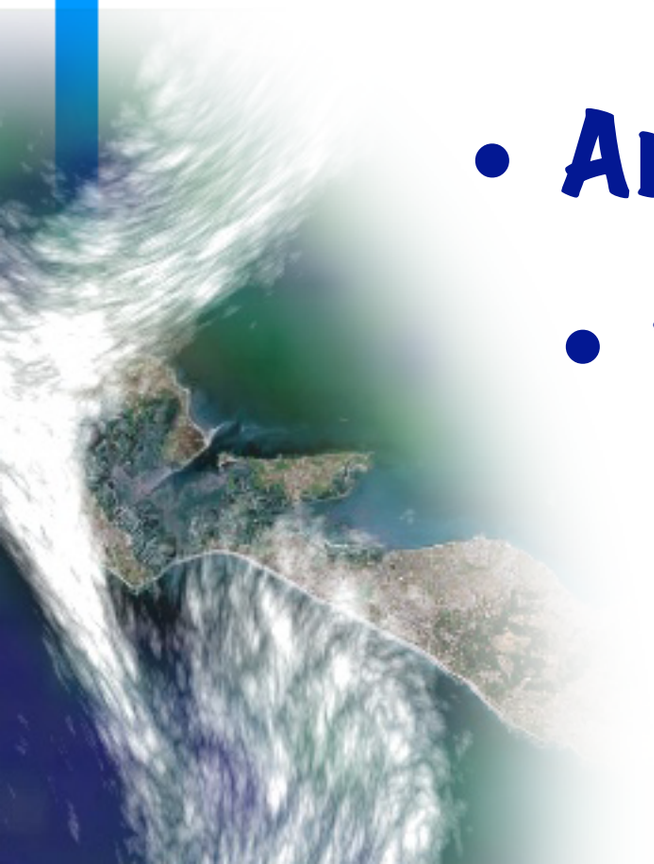
- Improved animated Perlin noise

- GPU implementation



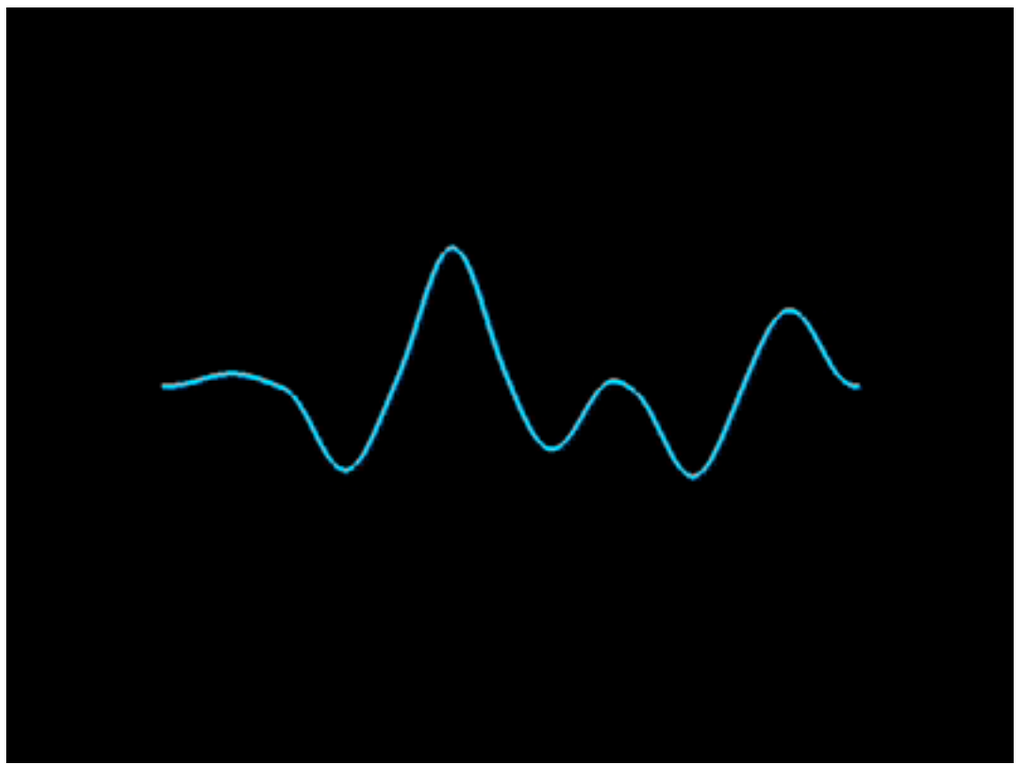
# State of the art

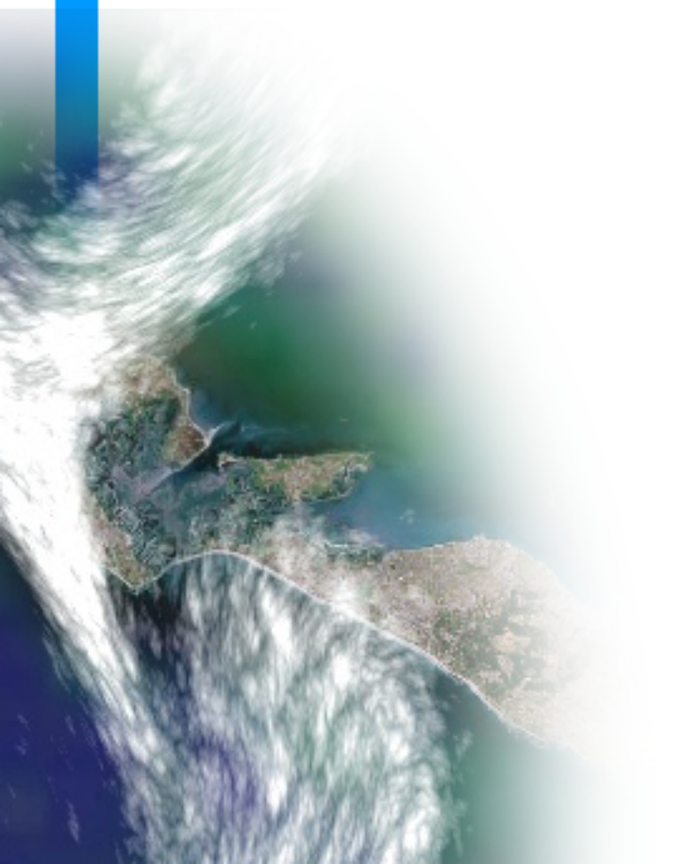
- Procedural textures
  - definition
  - advantages
- Animation issues
  - time and space continuity



# State of the art

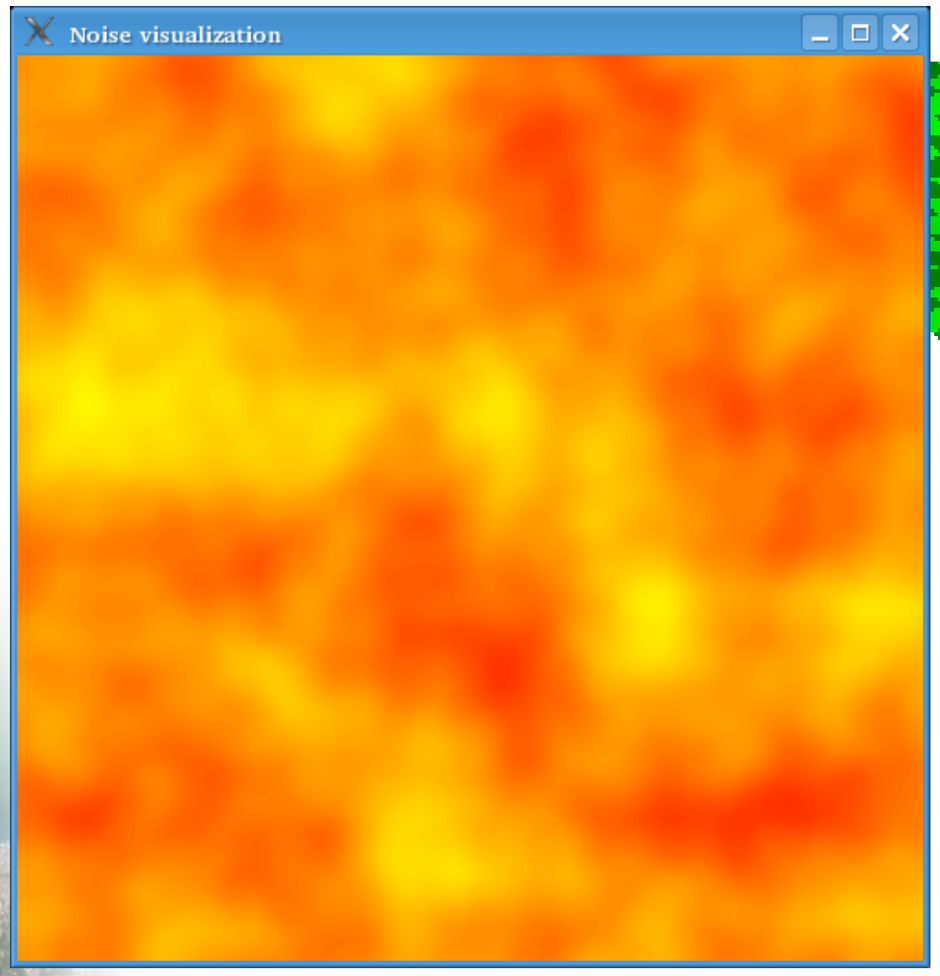
- (more than) An example : Perlin noise
  - 1D :

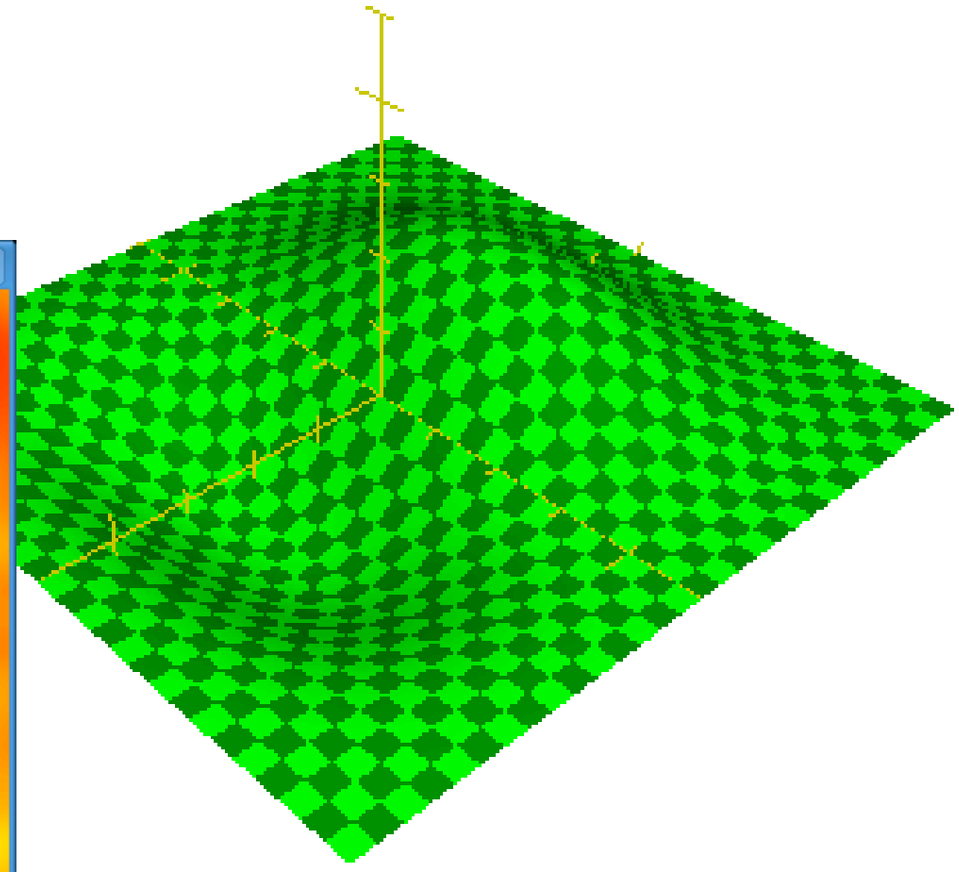


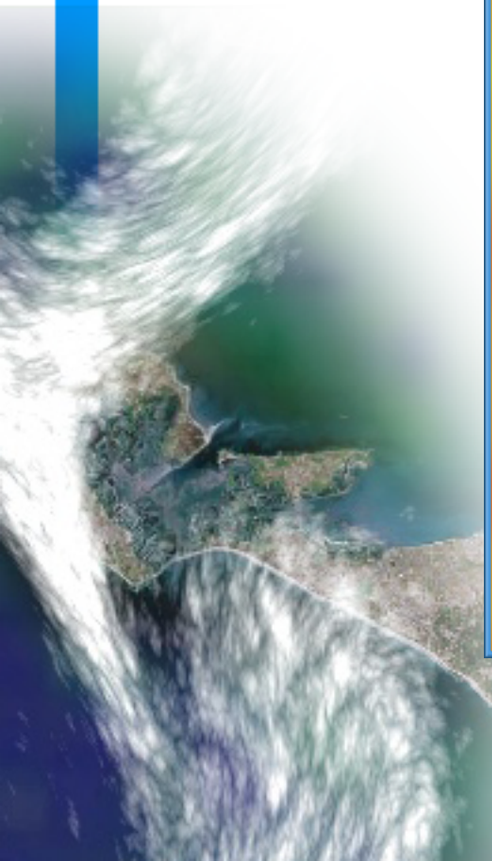


# State of the art

- (more than) An example : Perlin noise
  - 2D :

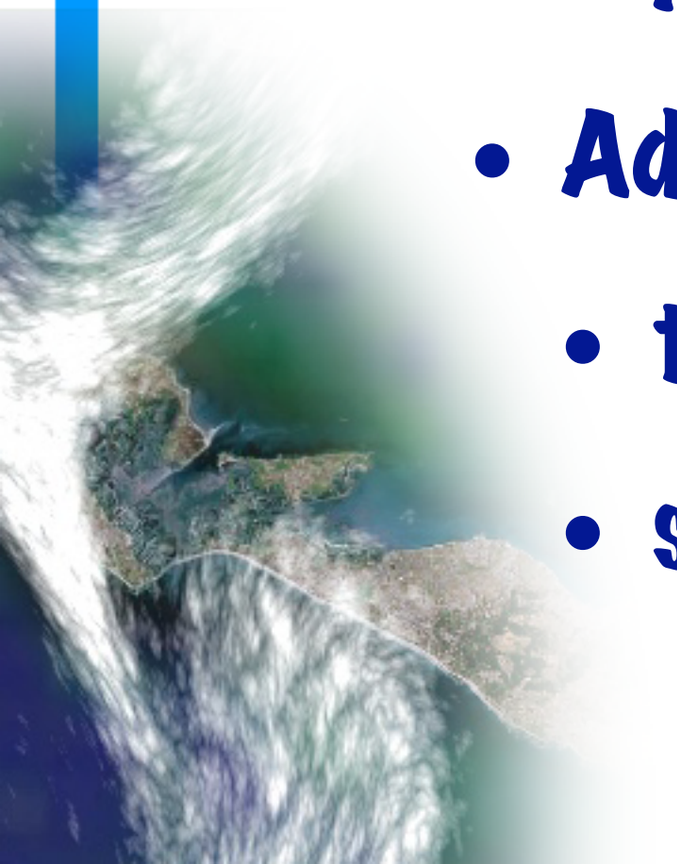




# State of the art

- Texturing fluids
  - flowing and swirling i.e. turbulence
- Flownoise
  - rotate gradients
- Advected textures
  - texture coordinates ‘follow’ the fluid
  - stretching, regeneration, latency

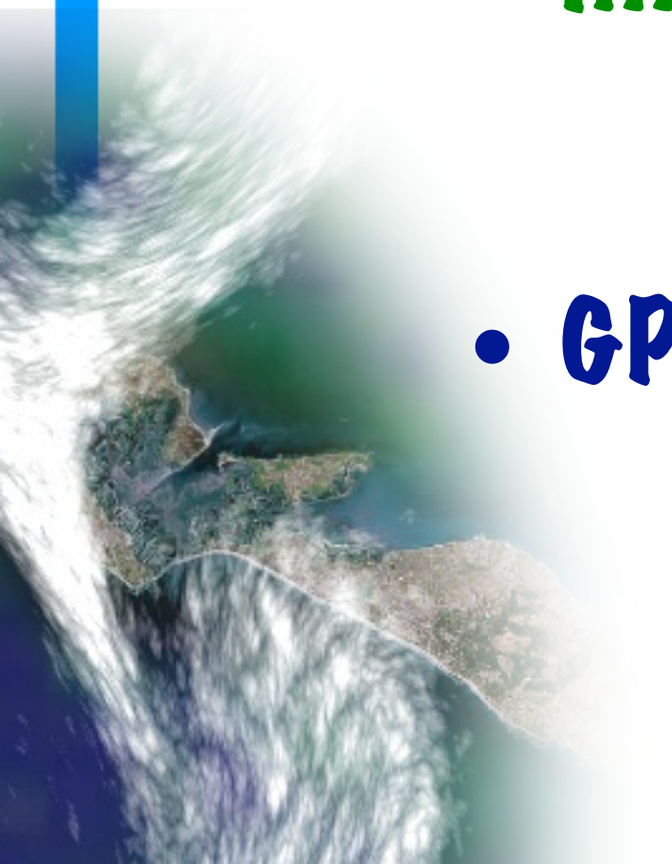


# Real-time flow-noise

- State of the art

- Improved animated Perlin noise

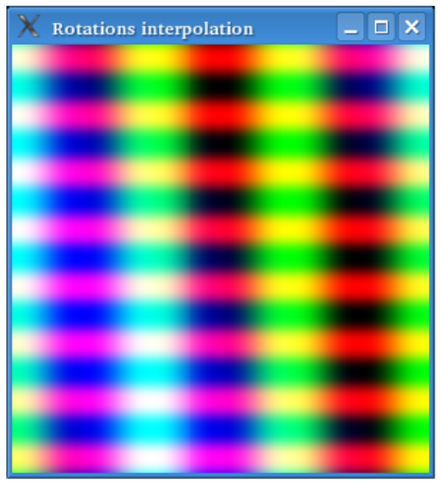
- GPU implementation



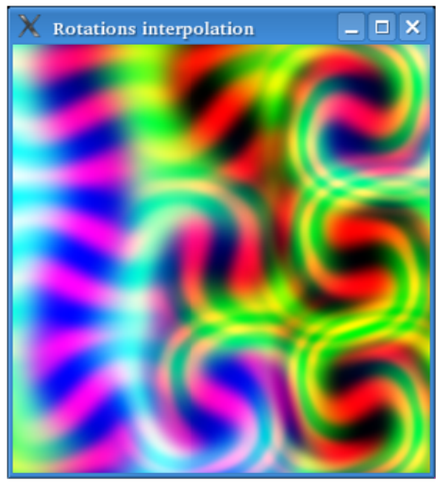
# Improved animated Perlin noise

- **Continuous displacment with respect to a rotation field generates rolling-up**

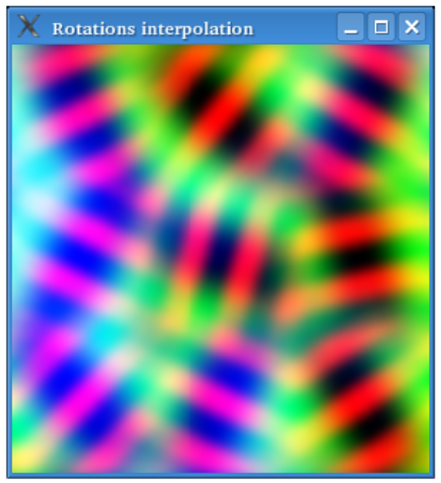
Comparaison des différentes méthodes d'interpolation sur une texture structurée et sur un bruit de Perlin



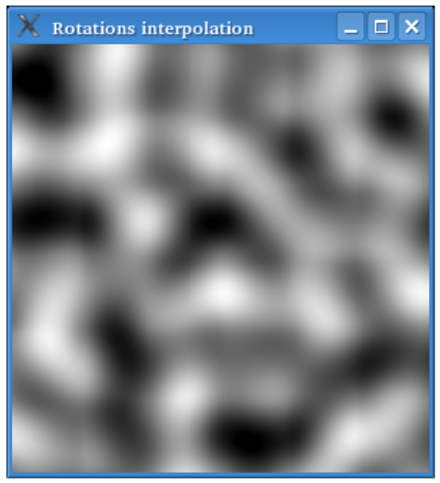

Texture non déformée

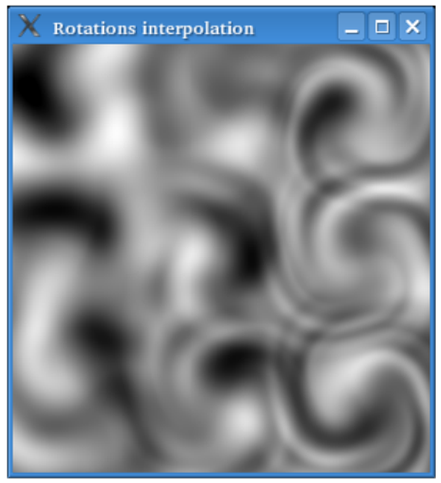


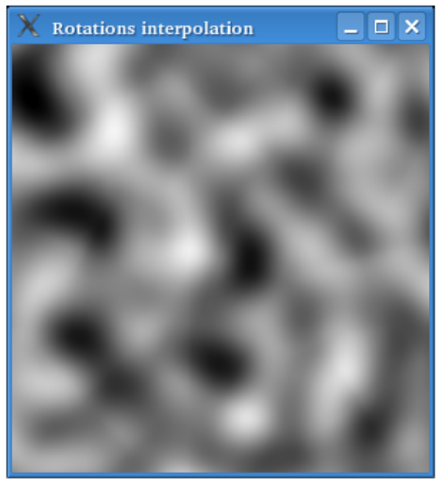

Interpolation du déplacement

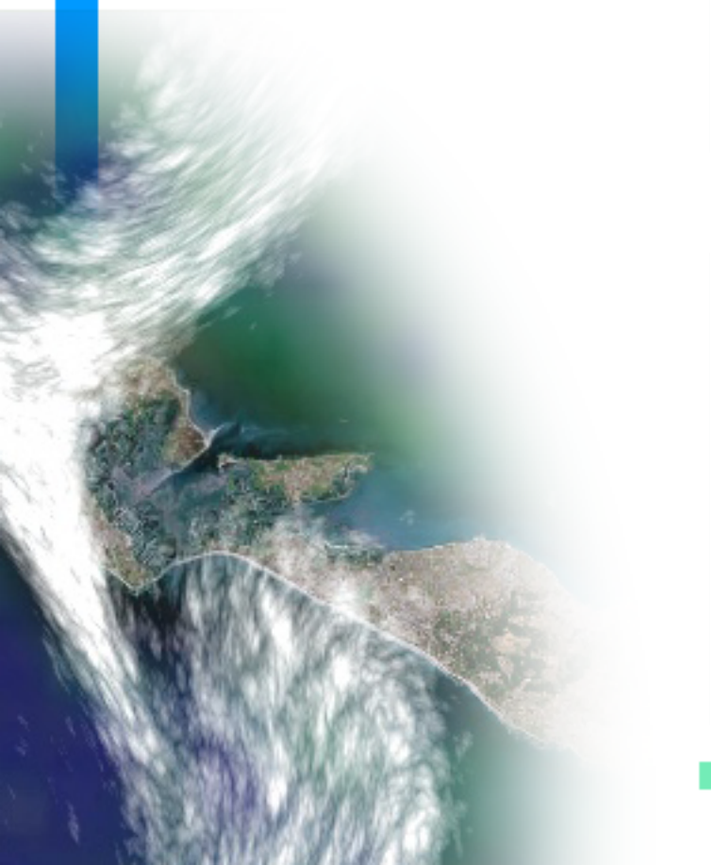



Interpolation du résultat



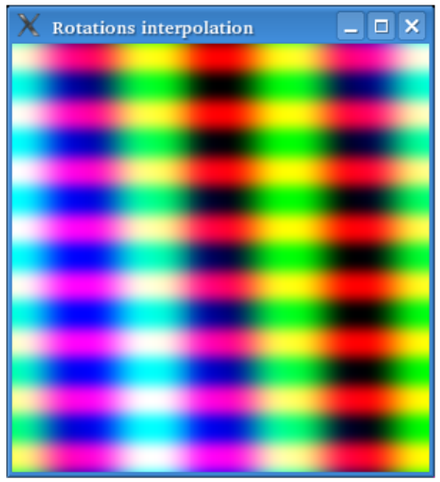



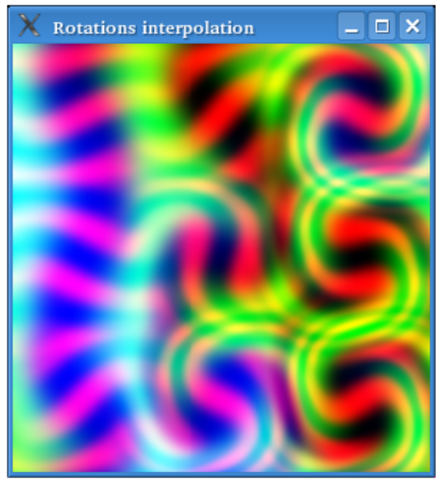



# Improved animated Perlin noise

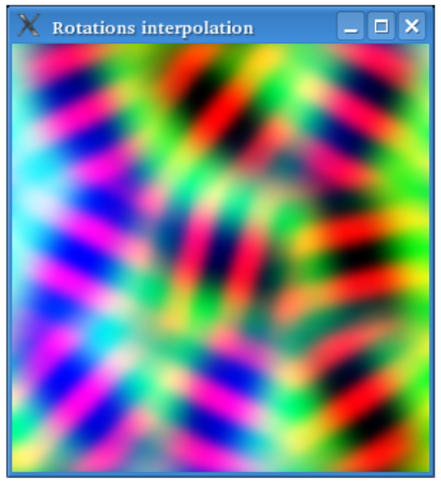
Comparaison des différentes méthodes d'interpolation sur une texture structurée et sur un bruit de Perlin



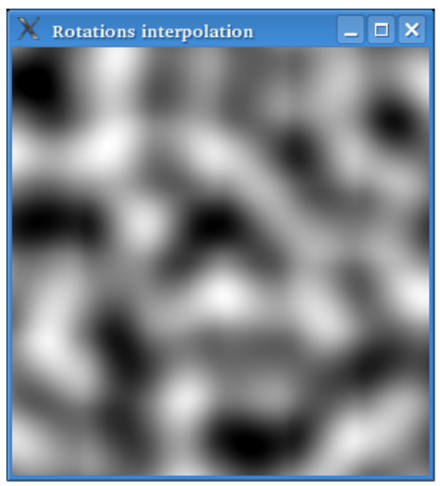

Texture non déformée

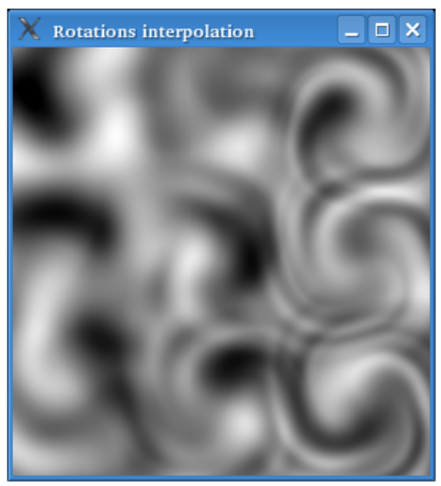


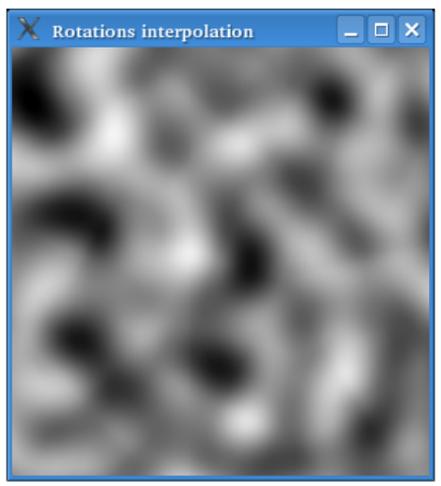

Interpolation du déplacement

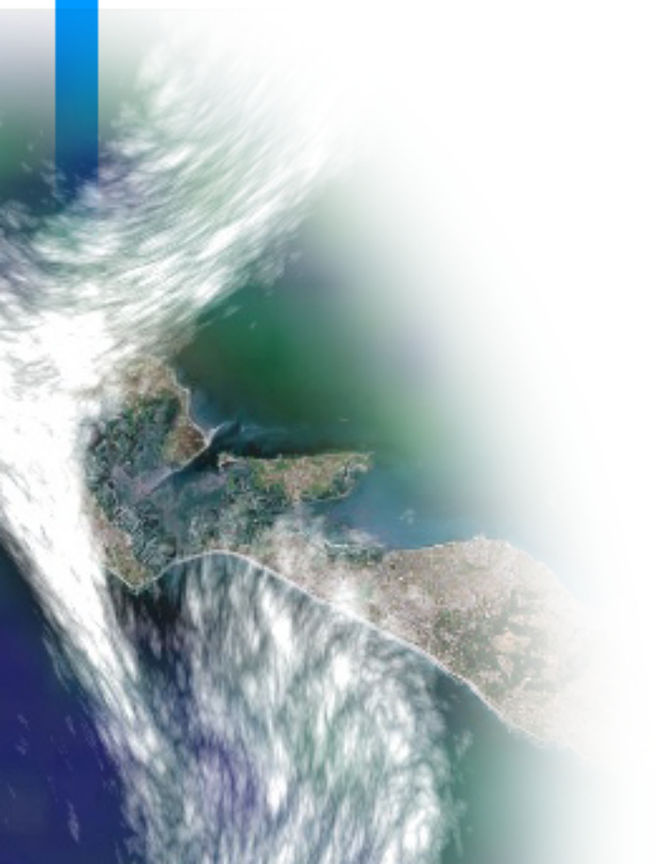



Interpolation du résultat



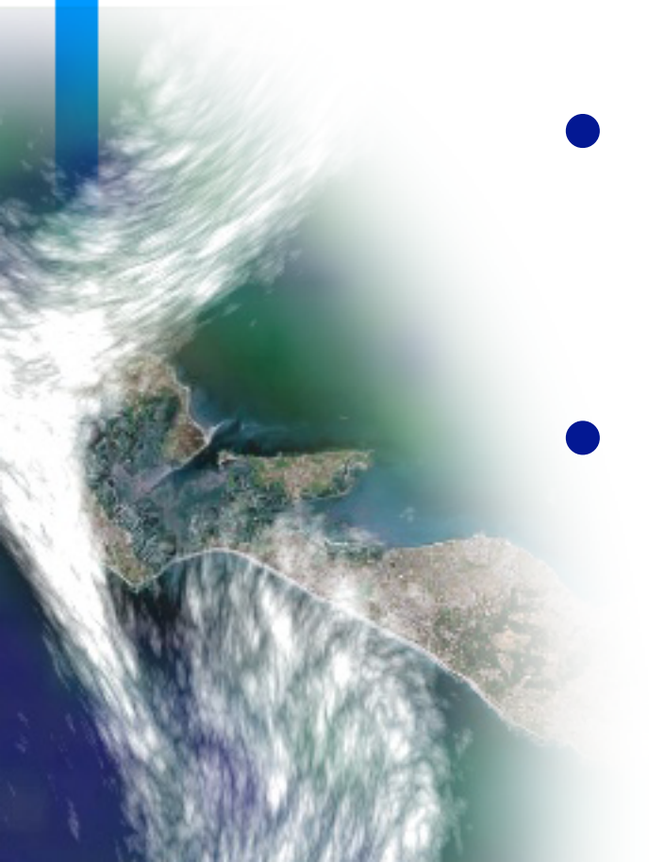



- give up spatial continuity
- noise properties hide it

# Improved animated Perlin noise

- Control rotation spectrum
  - Kolmogorov law : $E(k) = C_K \, e^{-2/3} \, k^{4/3}$
- our discrete version : $W_j \, \Omega_j^2 \sim k_j^{4/3}$
- relation between scale, weight and rotation speed
- still degrees of freedom / control !

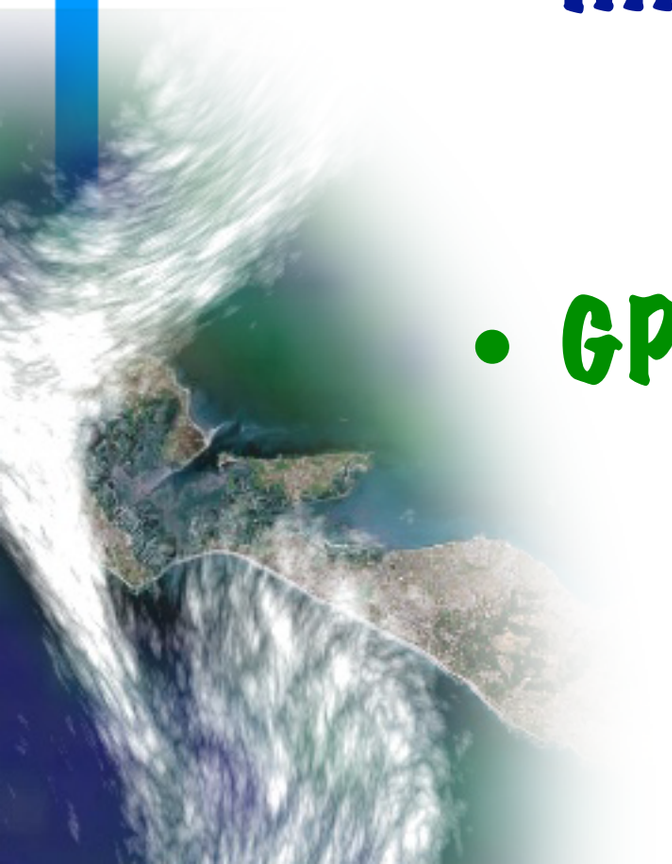


# Real-time flow-noise

- State of the art

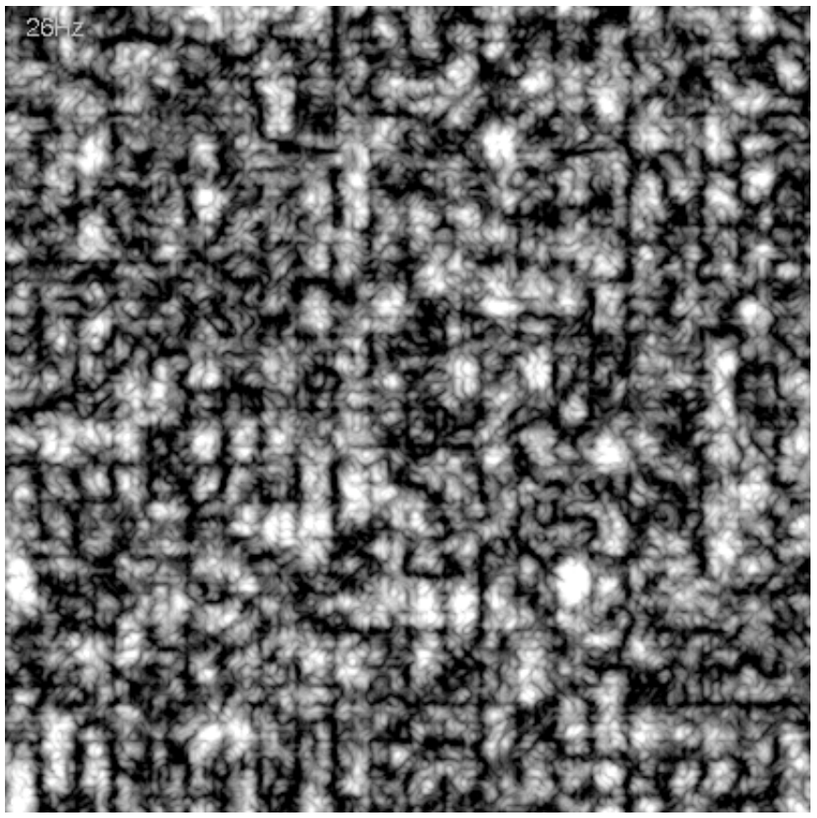
- Improved animated Perlin noise

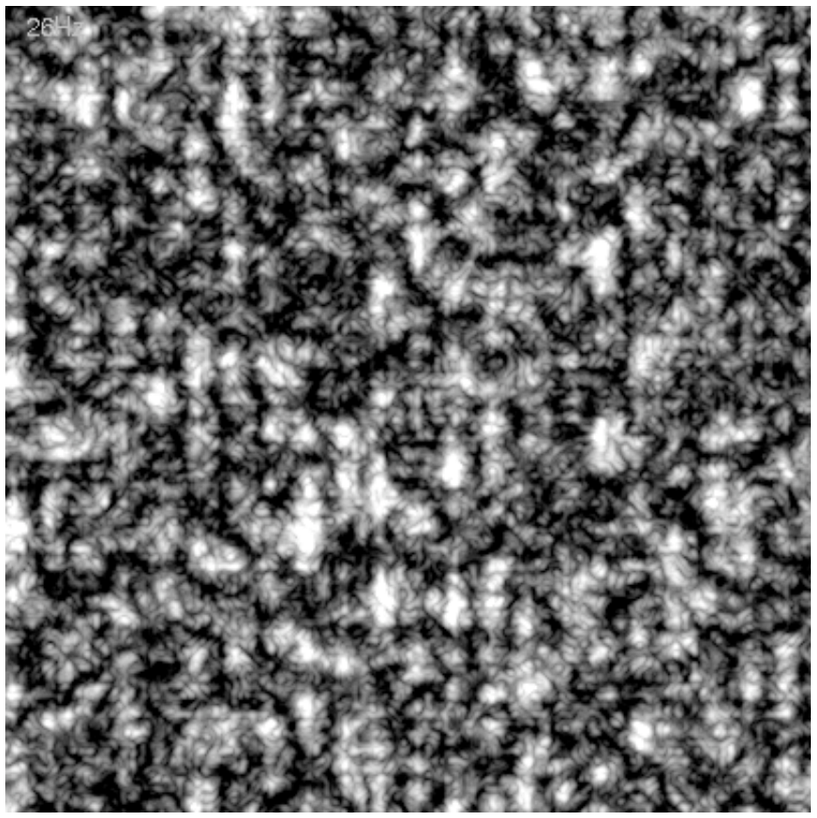
- GPU implementation

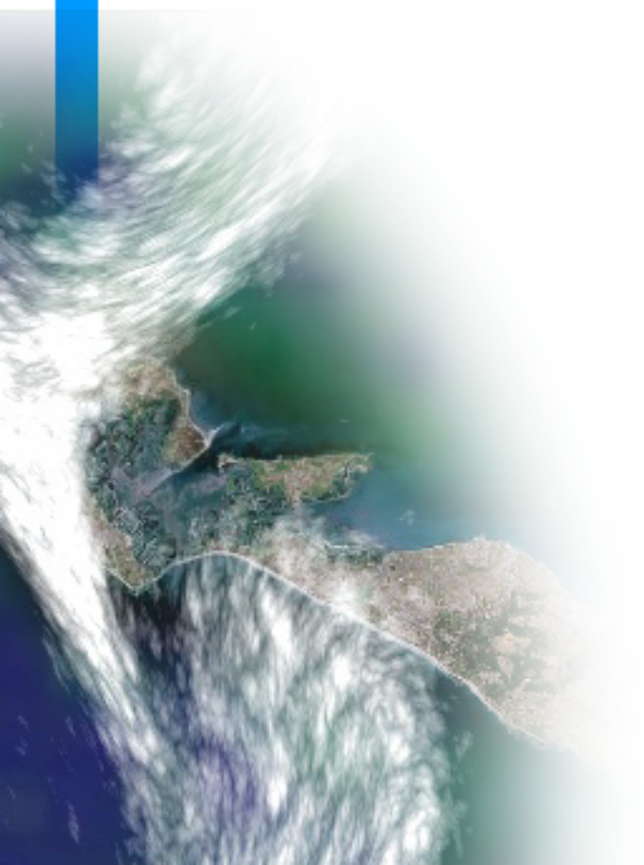


# GPU implementation

- Artifacts due to noise structure
  - add offset between different scales

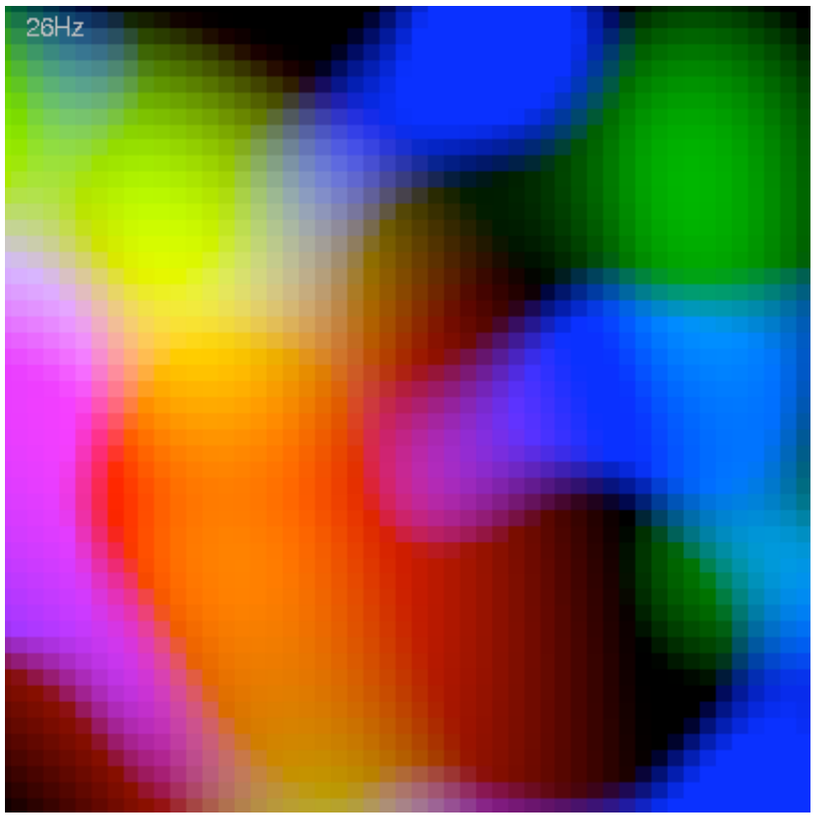


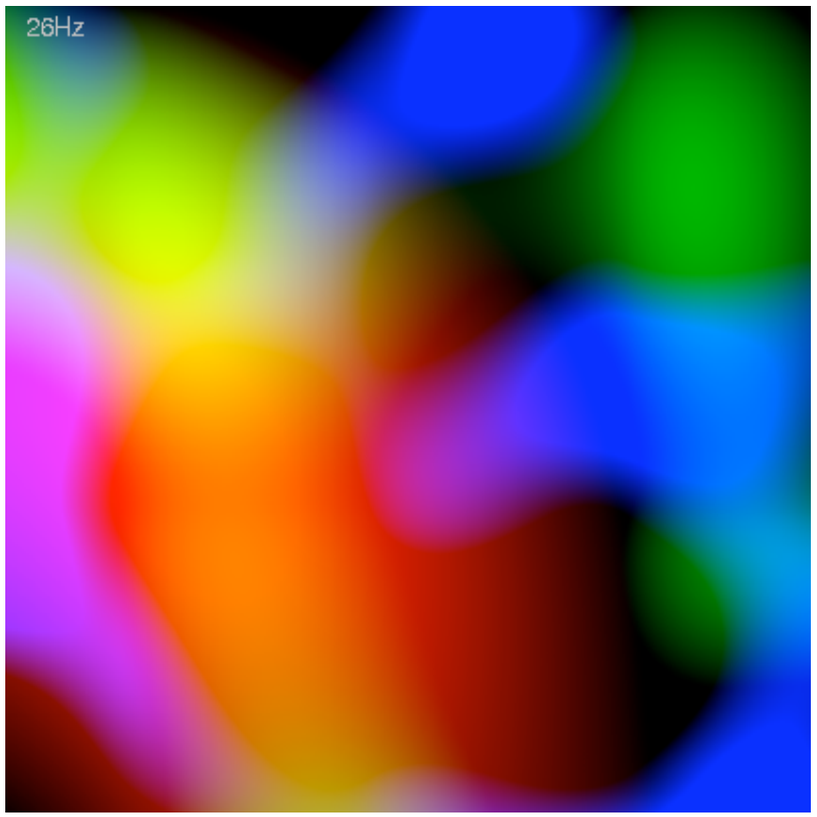



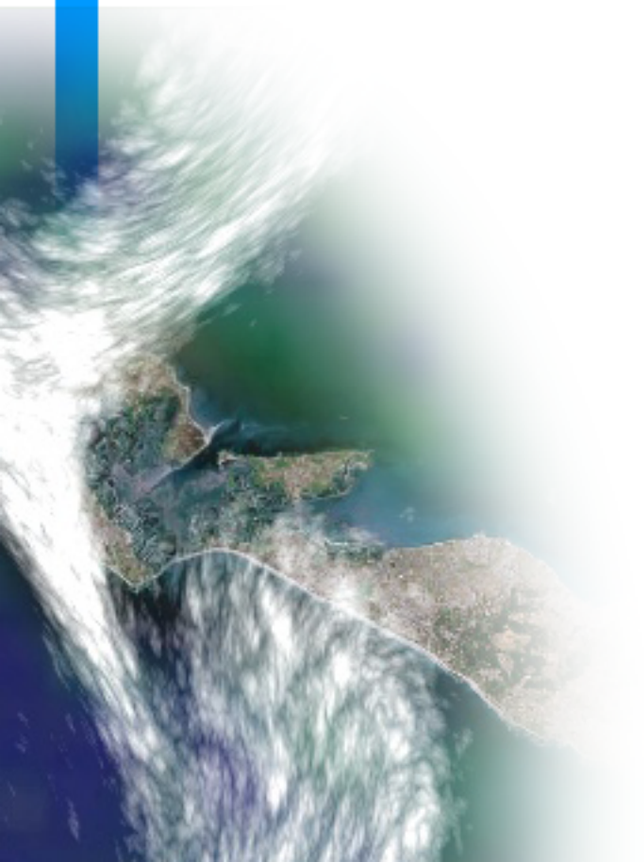



# GPU implementation

- Precision issues for texture coordinates
  - fp16 vs fp32

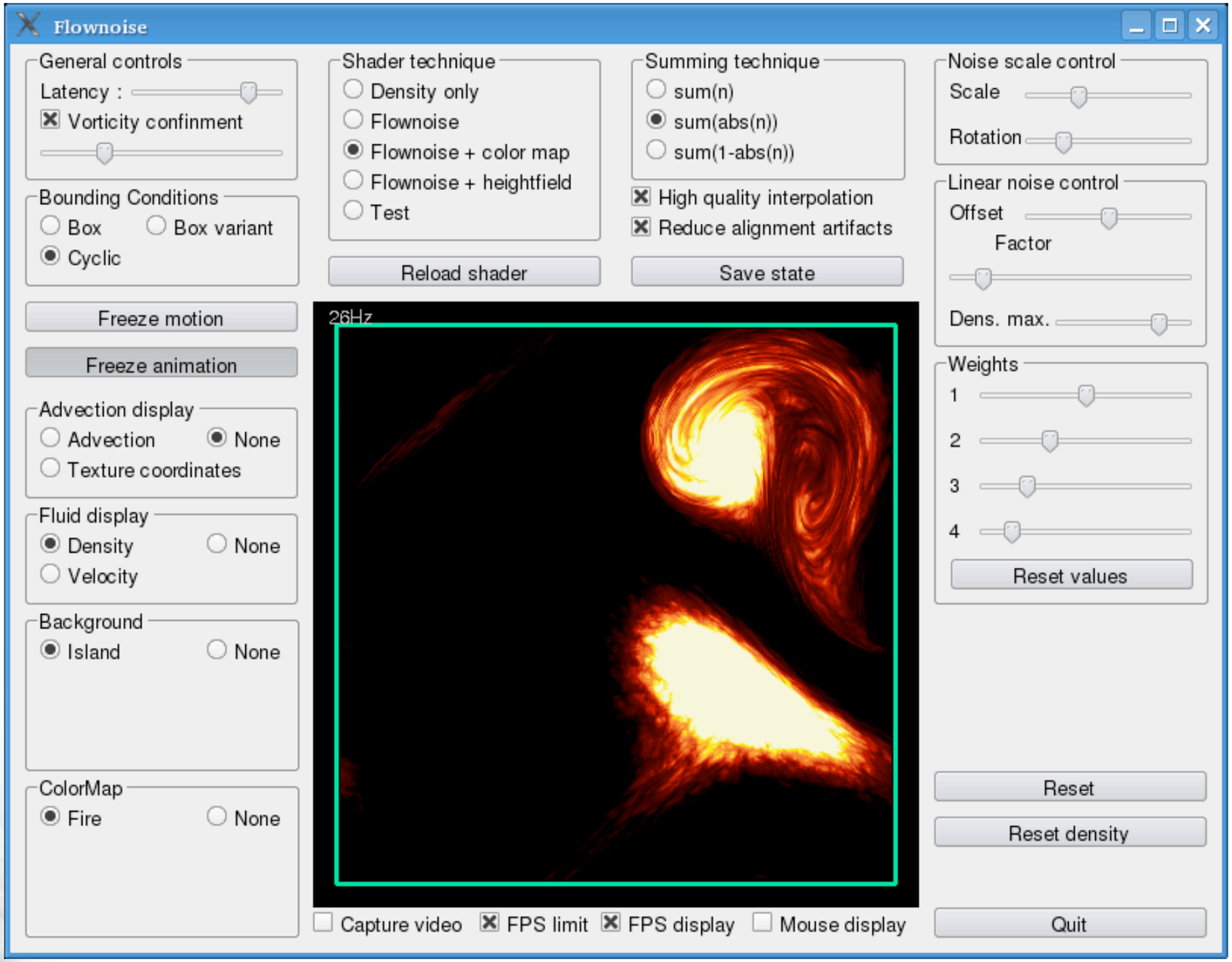


# GPU implementation

- **Overview**

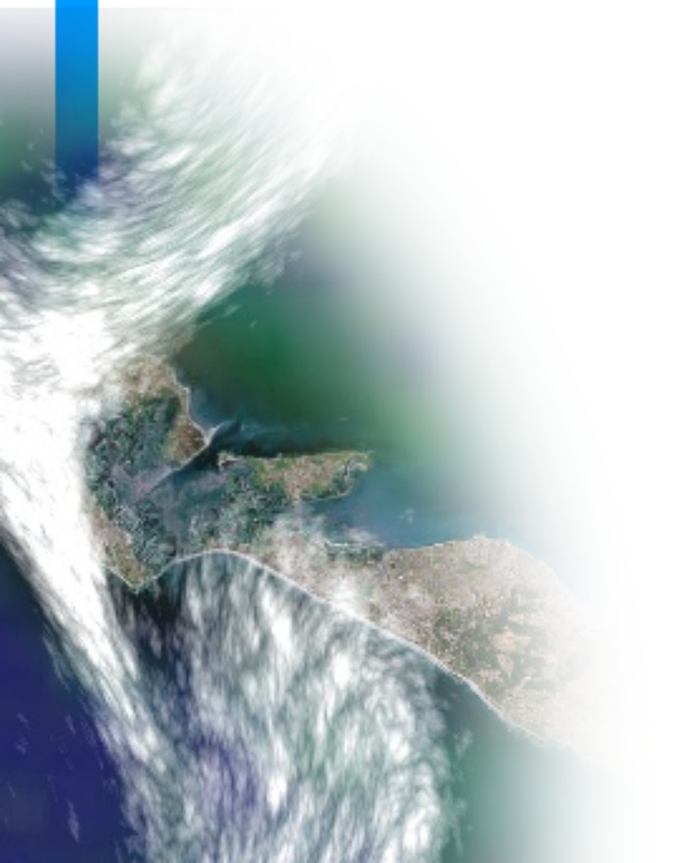


# GPU implementation

- Shader statistics :
  - 20 R-regs, 14 H-regs
  - 771 instructions
  - 48 COSH, 48 SINH, 74 TEX
- Results
  - >25Hz, 400x400 (QuadroFX 1400)

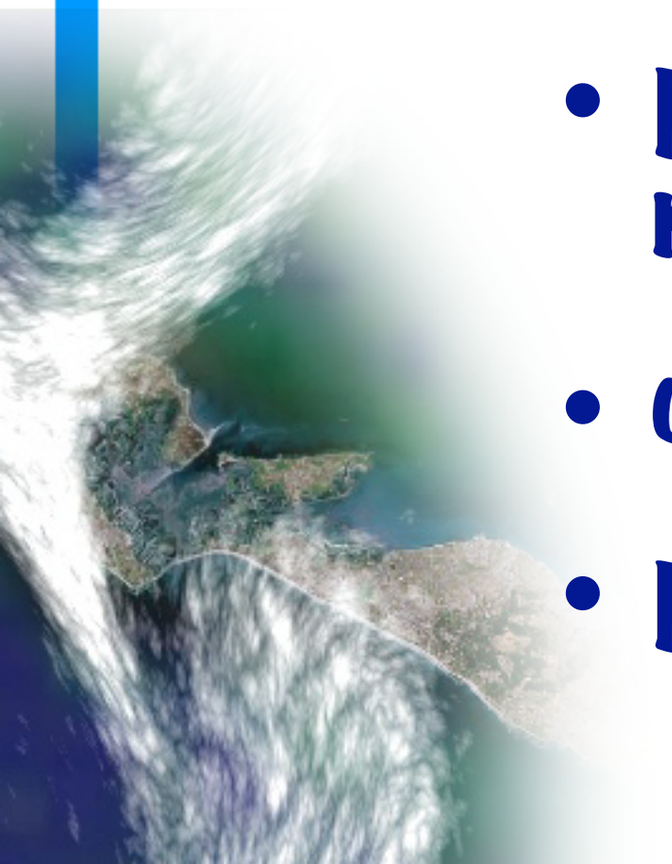
# GPU implementation

- Video



# Real-time flow-noise

- Conclusion
  - simple and artifact-free algorithm
  - real-time on GPU
- Ideas for future research
  - procedural texture coordinates and rotation values
  - create heightfields, possibly with LOD
  - progressive texture update



# Real-time flow-noise

- **Any questions ?**