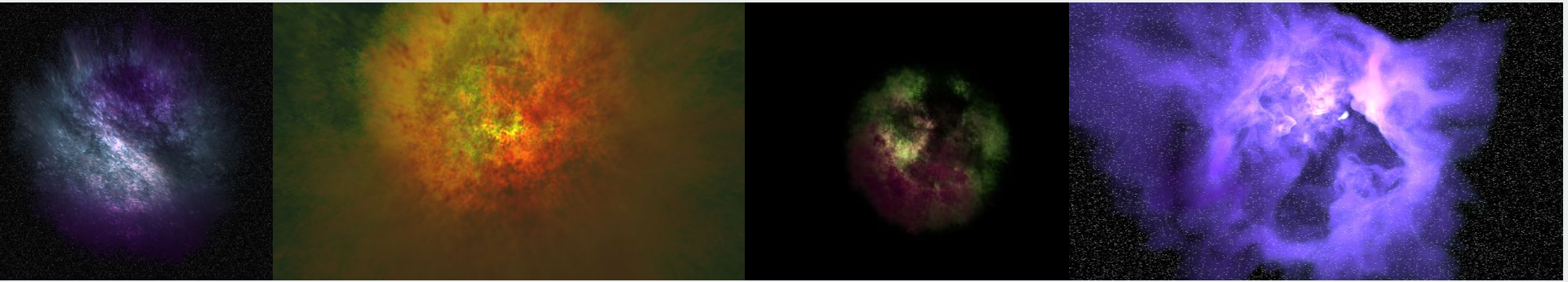
Aix*Marseille université initiative d'excellence
Master Géométrie et Informatique Graphique

MAVERICK

Grenoble INP ENSIMAG | UGA

# Procedural Generation of galactic dust and nebulas



Author : Erwan LERIA
Supervisor : Fabrice NEYRET

18/09/2020

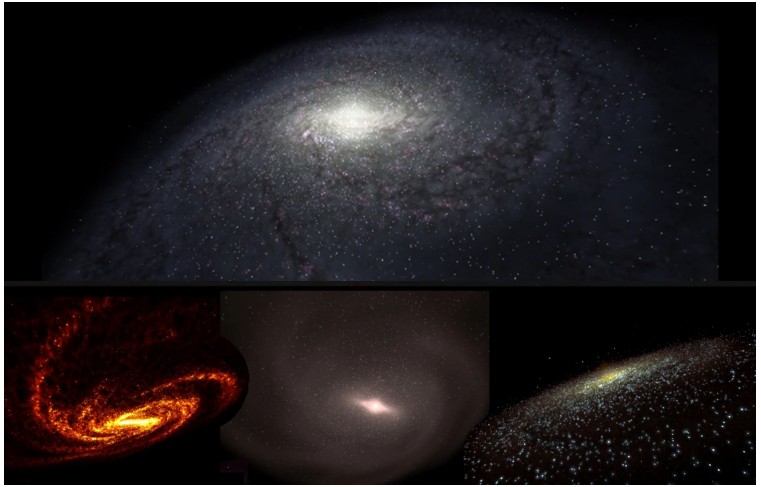
# Summary

- Context
- Previous works
- Our nebulas model
- Evaluations and future works

# Context

# Context

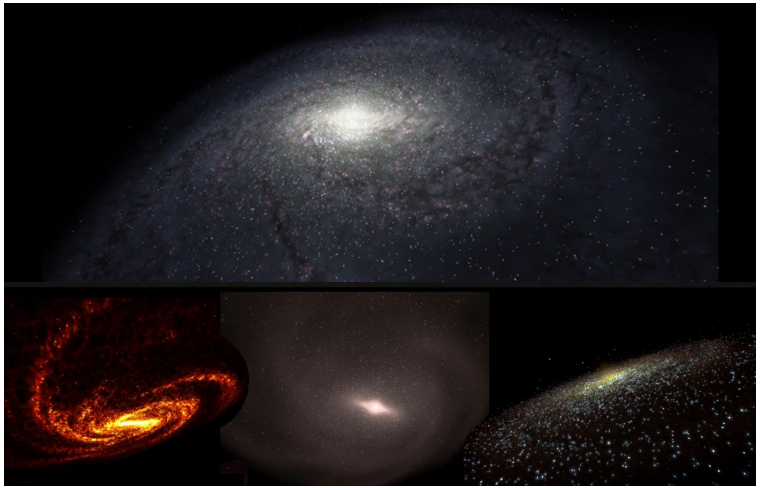
- ANR veRTIGE / Galaxy
  - INRIA, RSA Cosmos & Observatoire de Paris-Meudon



Picture from SkyExplorer, a RSA Cosmos Software

# Context

- Produire des scènes
  - vastes et détaillées
  - haute qualité graphique ~ Hubble
  - temps réel
  - sans données explicites (à la volée)



Images issues de SkyExplorer, un logiciel de RSA Cosmos

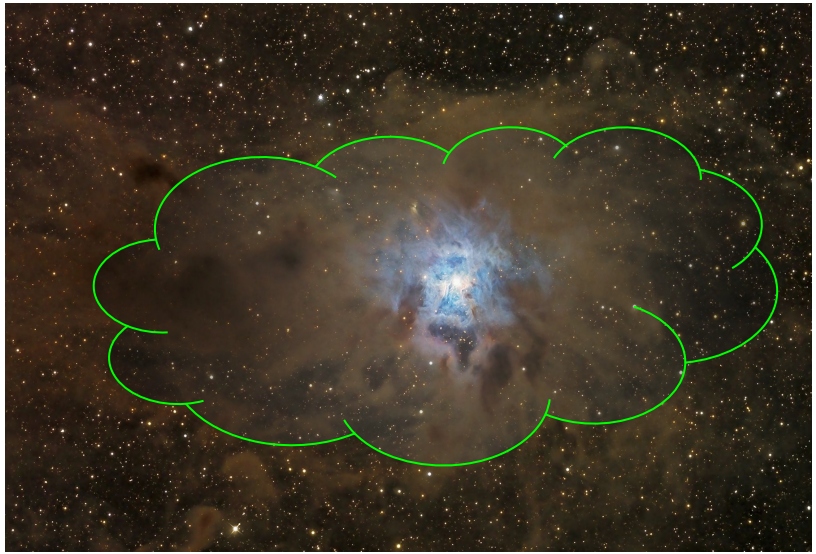
# Context

- Caractérisation d’une nébuleuse

“Iris Nebula” - NGC 7023, image sauvegardée par la NASA

# Context

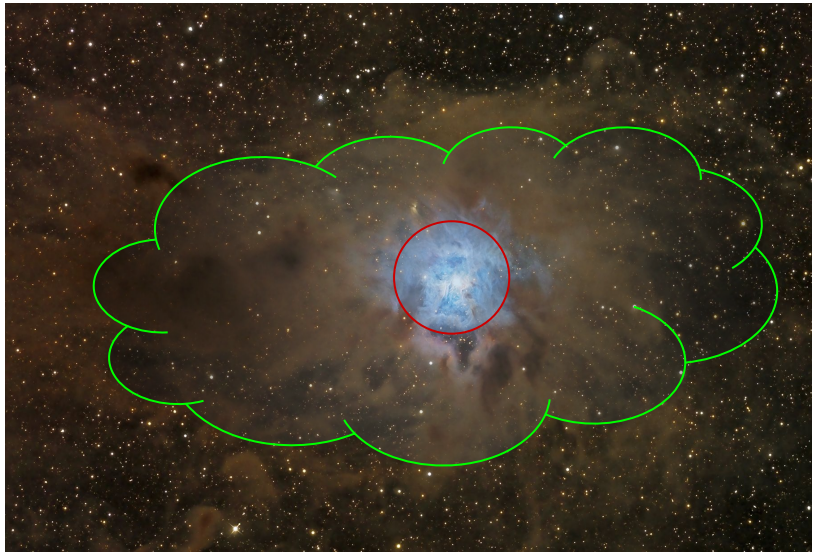
- Caractérisation d'une nébuleuse
  - nuage de poussière galactique



"Iris Nebula" - NGC 7023, image sauvegardée par la NASA

# Context

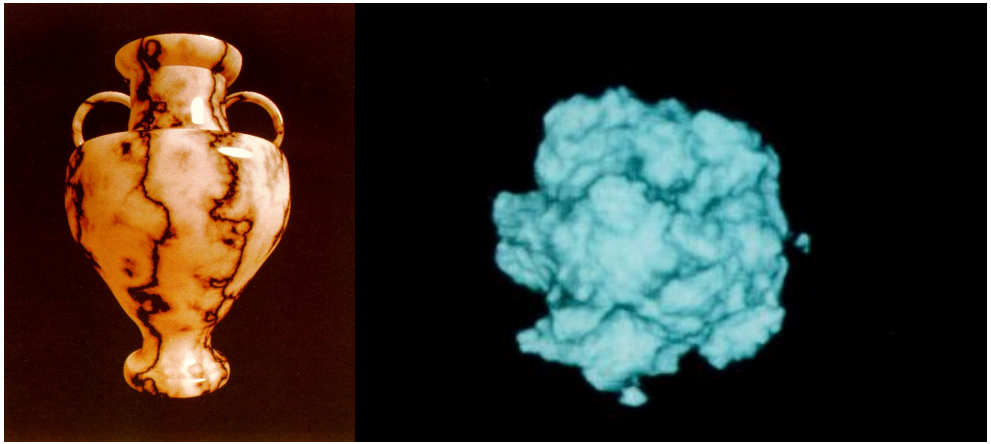
- Caractérisation d'une nébuleuse
  - nuage de poussière galactique
  - étoile perturbant le nuage
  - apparition de filaments et d'une zone de vide autour de cette étoile



"Iris Nebula" - NGC 7023, image sauvegardée par la NASA

# Previous Works

# Procedural Noise



Gauche : vase, *An image synthesizer* [Sig85]

Droite : fractal sphere, *Hypertexture* [Sig89]

# Procedural Noise

- Générer des textures 2D ou 3D
  - aléatoires
  - continues
  - contrôlables
  - à la volée

# Procedural Noise

- Bruit : $b(x)$
- Bruit fractal additif : $N_+(x) = \sum_i^{\eta} \frac{1}{2^i} \cdot b(2^i x)$
- Bruit fractal multiplicatif : $N_*(x) = \prod_i^{\eta} b(2^i x)$

# Volume Rendering

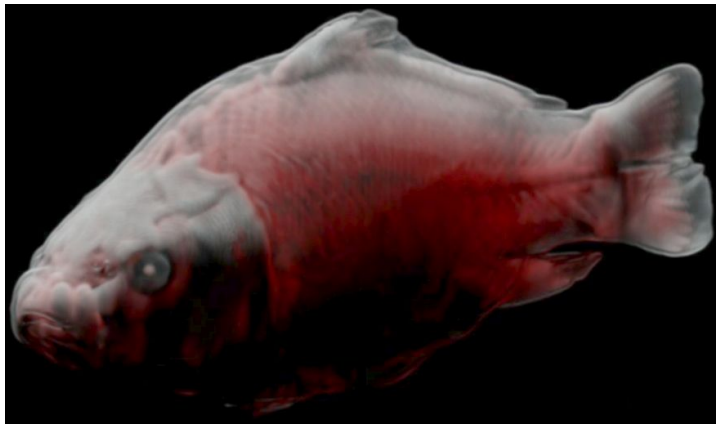


Image : transluscent volume shading, *Interactive Translucent Volume Rendering and Procedural Modeling* [IEEE02]

# Volume Rendering

- Intensité d'un pixel → équation Transport de la lumière

$$I_\lambda(x) = \int_0^\infty e^{-\int_0^l \sigma_t dl} \cdot (\sigma_s \quad) \cdot \psi \cdot \sum_0^{si} L_{s_i} \cdot e^{-\int_0^{s_i} \sigma_t dl} dl$$

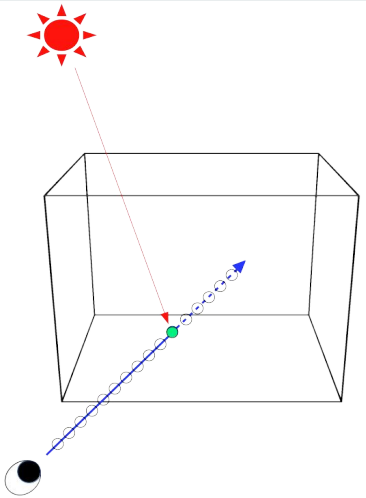
# Volume Rendering

- Intensité d'un pixel → discrétisation de l'équation

$$I_{rgb}(x) = \sum_{0}^{\infty} \prod_{0}^{l} e^{-\sigma_t \Delta_l} \cdot \frac{\sigma_s}{\sigma_t} \cdot Illumination$$

transparence accumulée couleur locale éclairage incident

# Volume Rendering



- Intensité d'un pixel → algorithme
  voxel courant de rayon(pixel):

$$C_{acc} = C_{acc} + T_{acc} \cdot (1 - T_{loc}) \cdot C_{loc} \cdot Illumination$$

  - *Cacc* : couleur accumulée
  - *Cloc* : couleur locale
  - *Illumination* : éclairage parvenu des sources de lumière
  - *Tloc :* transparence locale
  - *Tacc* : transparence accumulée

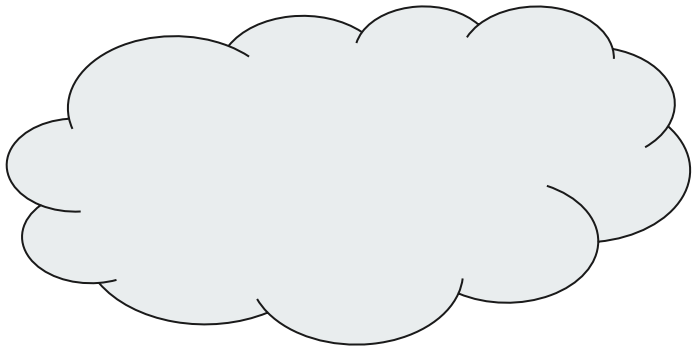
# Notre modèle de nébuleuse

# Notre modèle de nébuleuses

- Cahier des charges:
  - générer à la volée une forme plausible
  - détails compris
  - en faire le rendu temps réel
  - mettre en oeuvre et compléter les outils existant

# Notre modèle de nébuleuses

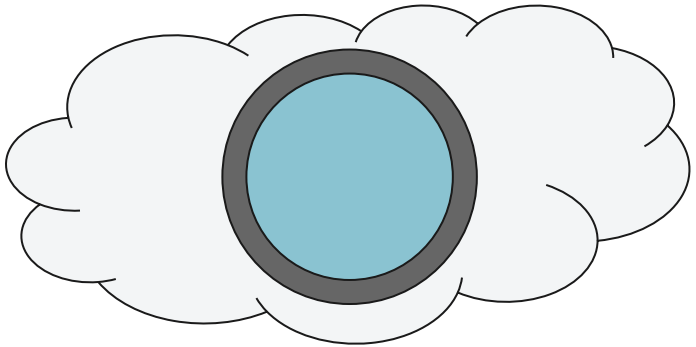
- Nuage de poussière
    - grain: densité = f( noise(P) )
    - forme:
        - mask → sigma=f(noise(P)) m(P)



# Notre modèle de nébuleuses

- Nuage de poussière
- Bulle
  - coque sphérique floue: compress(P)=$Gauss_{r,ep}(|P-S|)$
  - déformation :r+noise, ep+noise
  - Transparence is :

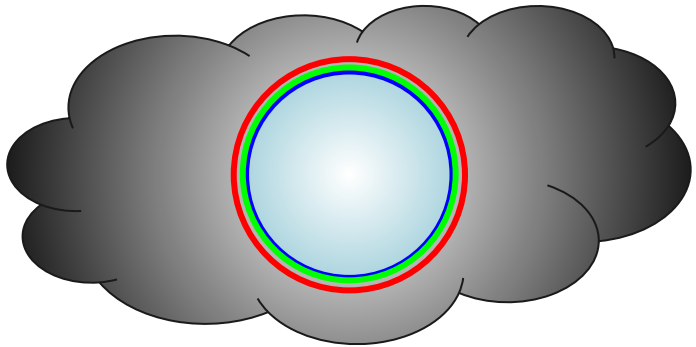
$$e^{-\alpha_{rgb}\cdot\tau(n(stretch(v)))\cdot compress(m(v))\Delta l}$$



# Notre modèle de nébuleuses

- Nuage de poussière
- Bulle
- Lumière  Cacc += Tacc* Cloc * Illu
  - Illu(P) =  estimation analytique

    = L * exp(-σs )
  - Cloc = σs/σt *Illu
  - σs = colormap(|P-S|-I0 )



# Notre modèle de nébuleuses

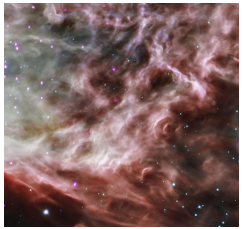
- Paramètres graphiques:
  - nuage:
    - forme: masque
    - densité
    - noise
  - bulle
    - radius, épaisseur
    - noise
- Paramètres physiques:
  - atténuation
  - couleurs physiques
    - $\sigma_t$(rgb)
    - $\sigma_s$(rgb) = colormap
  - fond (sky)

https://www.shadertoy.com/view/tsdyDr

https://www.shadertoy.com/view/Ws3yzH

# Démo

# Évaluations et travaux futurs

- Evaluation:
  - la vraie:  la fausse: 
  - perfs

- Travaux futurs
  - perf : passer les vides,  adapter precision à densité
  - améliorer filament: anisotropie, …, (bruit multiplicatif )
  - améliorer couleurs: couleurs physiques (ionization du gaz)
  - animer
  - intégrer dans le système

# Questions ?

# Annexe

# Évaluations et travaux futurs

- Fragment Shader webGL/GLSL
  - https://www.shadertoy.com/user/Leria
- Performances temps-réel, résolution ***800x450*** sur *Nebula 16* :
  - 60 fps → Nvidia GeForce GTX 1080 Ti
  - 27 fps → Nvidia GeForce GTX 770
  - 27 fps → Nvidia GeForce GTX 1050
  - 06 fps → Nvidia GeForce 920M

# Bruit procédural

- Comment générer un signal aléatoire et continu ?
  - Définition d'une grille régulière de dimension *n*

# Bruit procédural

- Comment générer un signal aléatoire et continu ?
  - Définition d'une grille régulière à *n* dimension
  - Définition de gradients aléatoires à chaque point de la grille

# Bruit procédural

- Comment générer un signal aléatoire et continu ?
  - Définition d'une grille régulière à *n* dimension
  - Définition de gradients aléatoires à chaque point de la grille
  - Génération d'une courbe à partir de l'interpolation des gradients

# Aliasing

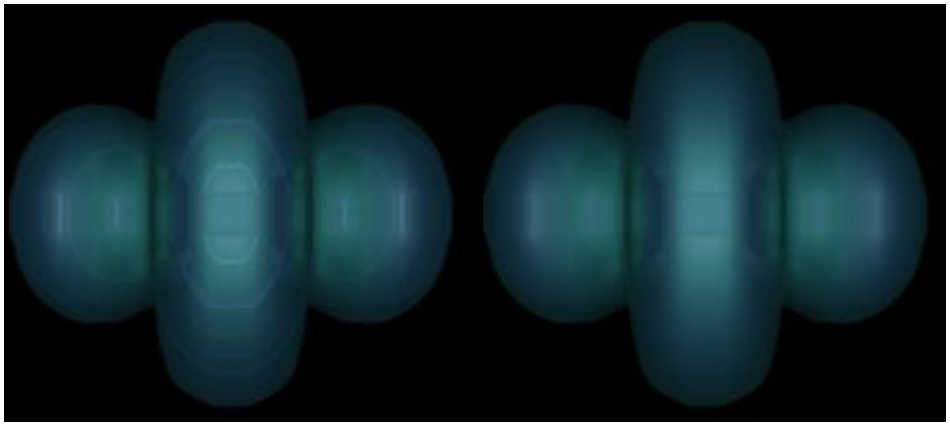


Image : Semi-transparent isosurface rendering of the spherical harmonic function without (left) and with correction (right)

# Aliasing

- Échantillonnage de la fonction de densité supérieur aux pas de rayon traversant le volume :
  - Apparition de tranches sur le volume
  - Utilisation de la pré-intégration pour “lisser” la surface