
Introduzione ai Particle Systems nella Computer Graphics

Docente:

Diego Romano

Studente:

Fabiola Salomone

Università degli Studi di Napoli "Federico II"



Introduzione

1

Stato dell'arte

2

L'algoritmo - Particelle Transitorie

3

L'algoritmo - Particelle Filamentose

4

Implementazione in OpenGL - Particelle Transitorie

5

Implementazione in OpenGL - Particelle Filamentose

6

Conclusioni

7

Introduzione

Oggetti Solidi vs Oggetti Fuzzy

Oggetti Solidi



Superficie liscia, ben definita e lucida

Geometria netta, modellabile, con mesh e poligoni

Esempi : sfera, cubo, automobile

Oggetti Fuzzy



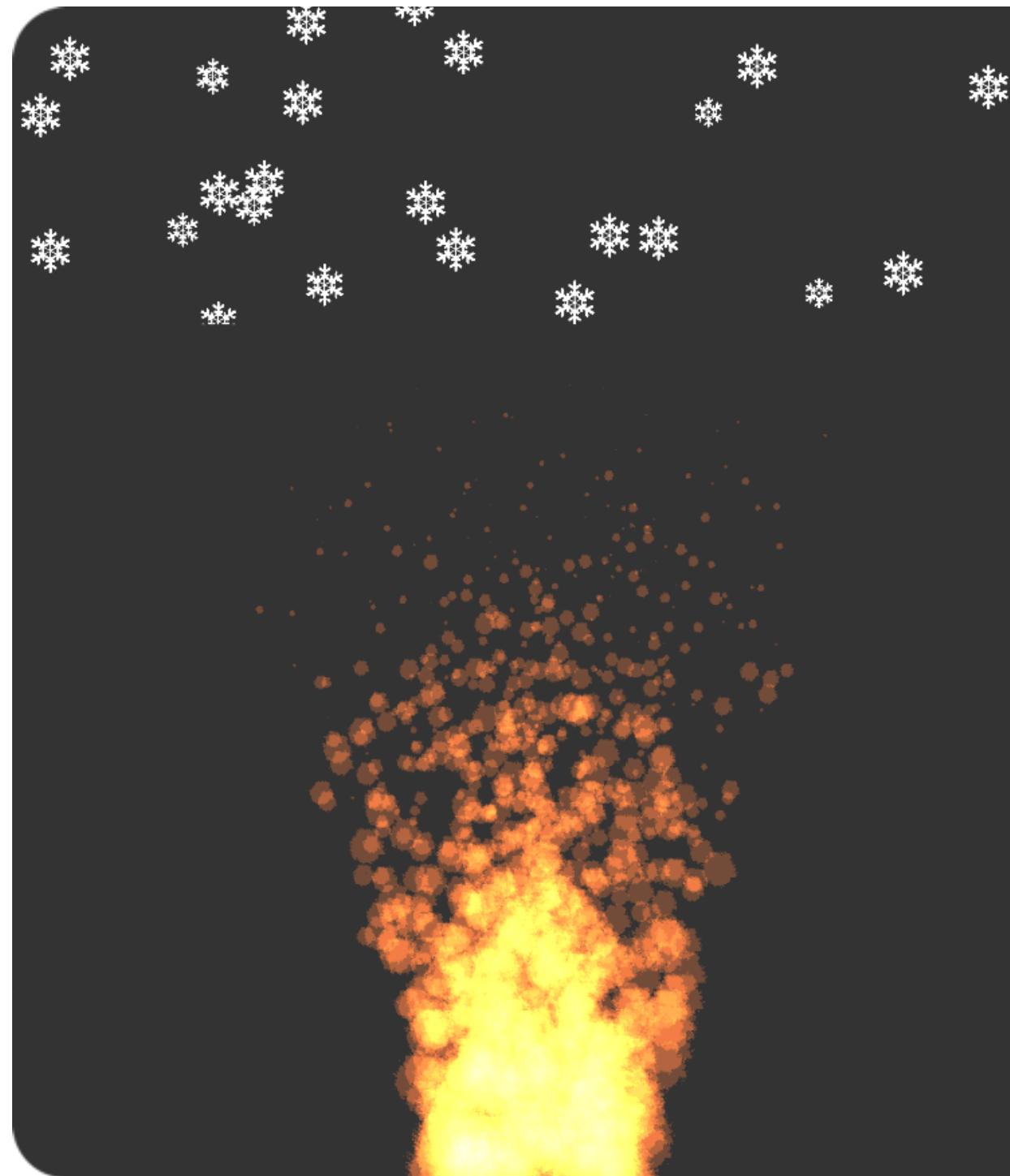
Superficie irregolare, indefinita, stonata

Geometria complessa, volumetrica o dinamica

Esempi : fuoco, fumo, nebbia

Introduzione

Oggetti Fuzzy nella Computer Graphics



Oggetti Fuzzy con particelle
transitorie



Oggetti Fuzzy con particelle
filamentose

Stato dell'arte

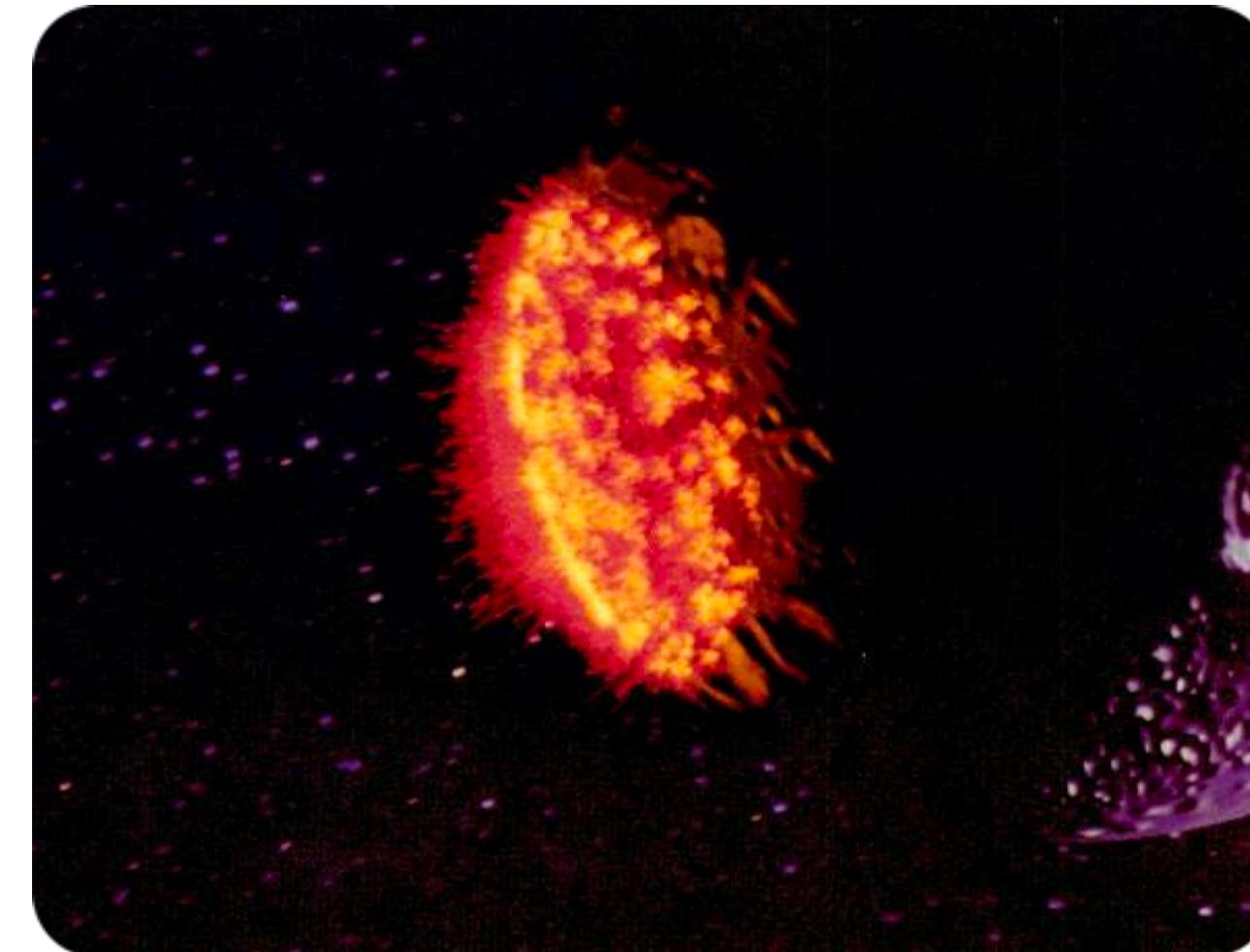
Rendering di oggetti



Volumetric Simulation

Jim Blinn (1982)

- Light Reflection Functions
- Volume based
- Physically-based scattering
- Voyager/Anelli di Saturno (NASA)



Particle Systems

William T.Reeves (1983)

- “Fuzzy” Objects
- Stochastic dynamics
- Birth/death di particelle
- Effetti per Star Trek II

Stato dell'arte

Rendering di oggetti



Volumetric Simulation

Jim Blinn (1982)

- Light Reflection Functions
- Volume based
- Physically-based scattering
- Voyager/Anelli di Saturno (NASA)



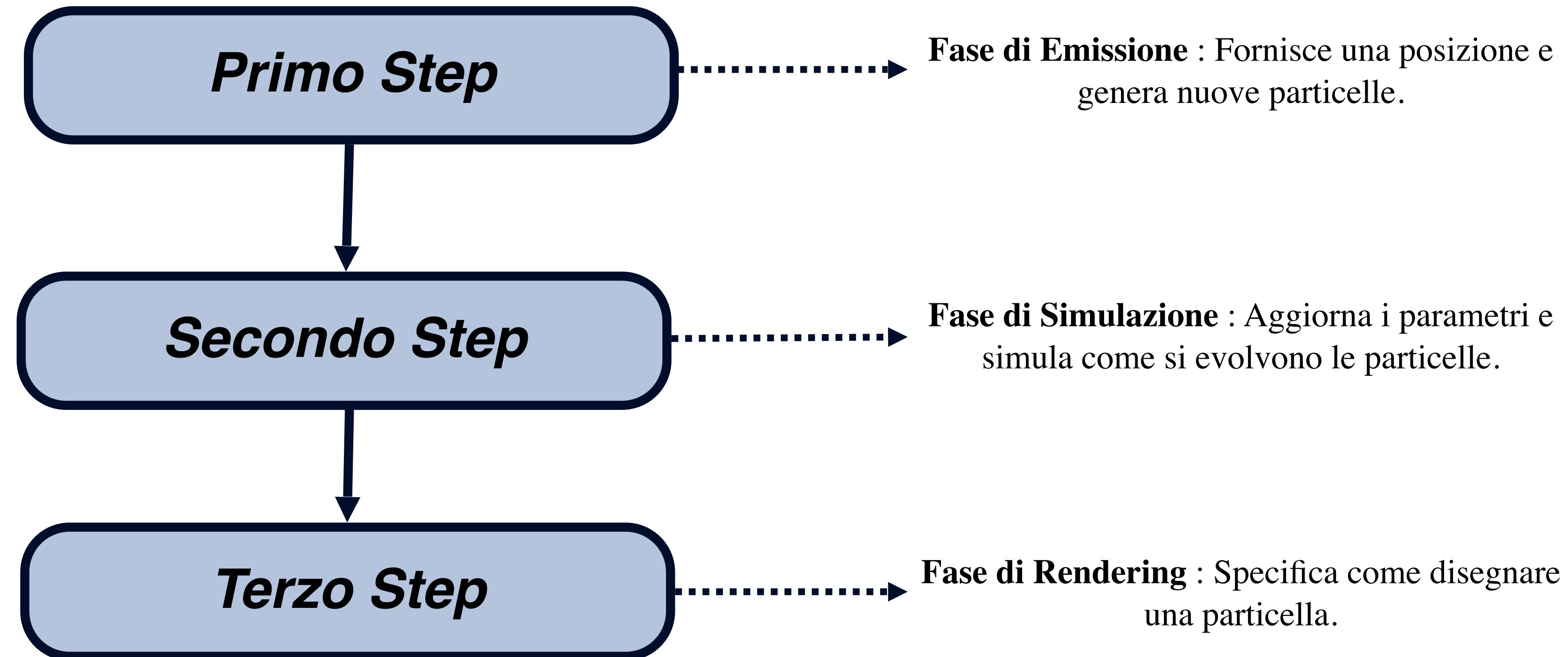
Particle Systems

William T. Reeves (1985)

- Structured Particle Rendering
- Shading probabilistico su insiemi strutturali
- Stocastico, approssimato
- The Adventures of André and Wally B

L'algoritmo - Particelle Transitorie

Schema esecutivo



L'algoritmo - Particelle Transitorie

Schema esecutivo

Primo Step

- Si modula l'intensità delle particelle nel tempo per fotogramma con funzioni lineari.
- Si calcola il numero di particelle per fotogramma: calcolato con media e varianza o proporzionale all'area dello schermo.
- Si generano le particelle tramite processi stocastici controllati.
- Si assegna ad ogni particella parametri d'inizializzazione come : posizione iniziale, velocità iniziale, colore iniziale, trasparenza iniziale, dimensione iniziale, forma e durata di vita.

L'algoritmo - Particelle Transitorie

Schema esecutivo

Secondo Step

- Si struttura il sistema a gerarchia ad albero, con sistemi figli influenzati dal sistema padre, utile per effetti complessi.
- Si muovono le particelle e si e aggiornano continuamente i loro attributi visivi (colore, dimensione, trasparenza) che variano in base alla velocità, all'accelerazione e si definisce quanto velocemente cambiano gli attributi visivi nel tempo.
- Si presta attenzione alla scomparsa della particella che avviene quando: la sua durata scade; la sua intensità visiva è troppo bassa; è troppo distante dal sistema padre.

L'algoritmo - Particelle Transitorie

Schema esecutivo

Terzo Step

- Si trattano le particelle come sorgenti luminose puntiformi, non si disegnano direttamente ma si calcola la luce che aggiungono ai pixel
- Si determina il Colore e intensità della luce in base al colore emesso e alla trasparenza della particella.
- La quantità di luce aggiunta a un pixel non dipende dalla distanza tra la particella e il punto di visualizzazione.
- Si disegnano le forme delle particelle con antialiasing al fine di prevenire aliasing temporale e sfarfallio (strobing)
- Si fa in modo che se molte particelle colpiscono lo stesso pixel, l'algoritmo limita l'intensità dei colori (RGB) al massimo del frame buffer per evitare sovraccarico visivo.

L'algoritmo - Particelle filamentose

Schema esecutivo

Primo Step

- Si posizionano le particelle nello spazio 3D secondo una griglia, regole procedurali e distribuzioni casuali controllate da parametri
- Si assegna ad ogni particella i parametri iniziali (es. forma, colore, orientamento) tramite distribuzioni probabilistiche.
- Si fa in modo che le particelle formano una struttura gerarchica ricorsiva e interdipendente, con relazioni definite da variazioni stocastiche.
- Si applicano vincoli spaziali e ambientali per determinare dove e quante particelle emettere.

L'algoritmo - Particelle filamentose

Schema esecutivo

Secondo Step

- Si aggiornano nel tempo i parametri delle particelle: posizione, velocità, colore, forma, orientamento, dimensione e deformazioni.
- Si applicano forze esterne: gravità, vento, luce, collisioni o vincoli.
- Si introducono variazioni casuali per simulare l'irregolarità naturale e l'imprevedibilità.
- Si gestisce lo stato della particella: attiva/inattiva, durata, transizioni di stato.
- Si calcolano le interazioni tra particelle e ambiente, come terreno, vento e ostacoli.

L'algoritmo - Particelle filamentose

Schema esecutivo

Terzo Step

- Si proiettano le particelle da 3D a 2D con prospettiva per definirne la posizione visiva.
- Si calcola la luce (ambientale, diffusa, speculare), anche in modo probabilistico.
- Si simulano ombre interne ed esterne con tecniche approssimative (es. shadow mask).
- Si ordinano le particelle in base alla profondità per garantire una corretta sovraposizione visiva (painter's algorithm o bucket sort).
- Si rappresentano le particelle graficamente con primitive semplici e antialiasing per migliorarne la qualità.
- Si combinano gli elementi nell'immagine finale rispettando trasparenze e colori.

Implementazione in OpenGL - Particelle Transitorie

Inizializzazione scena grafica - Codice

L' impostazioni della scena in OpenGL viene fatta in tal modo :

```
glEnable(GL_BLEND);
glBlendFunc(GL_SRC_ALPHA, GL_ONE);
glEnable(GL_LINE_SMOOTH);
glEnable(GL_POINT_SMOOTH);
```

Setup della camera :

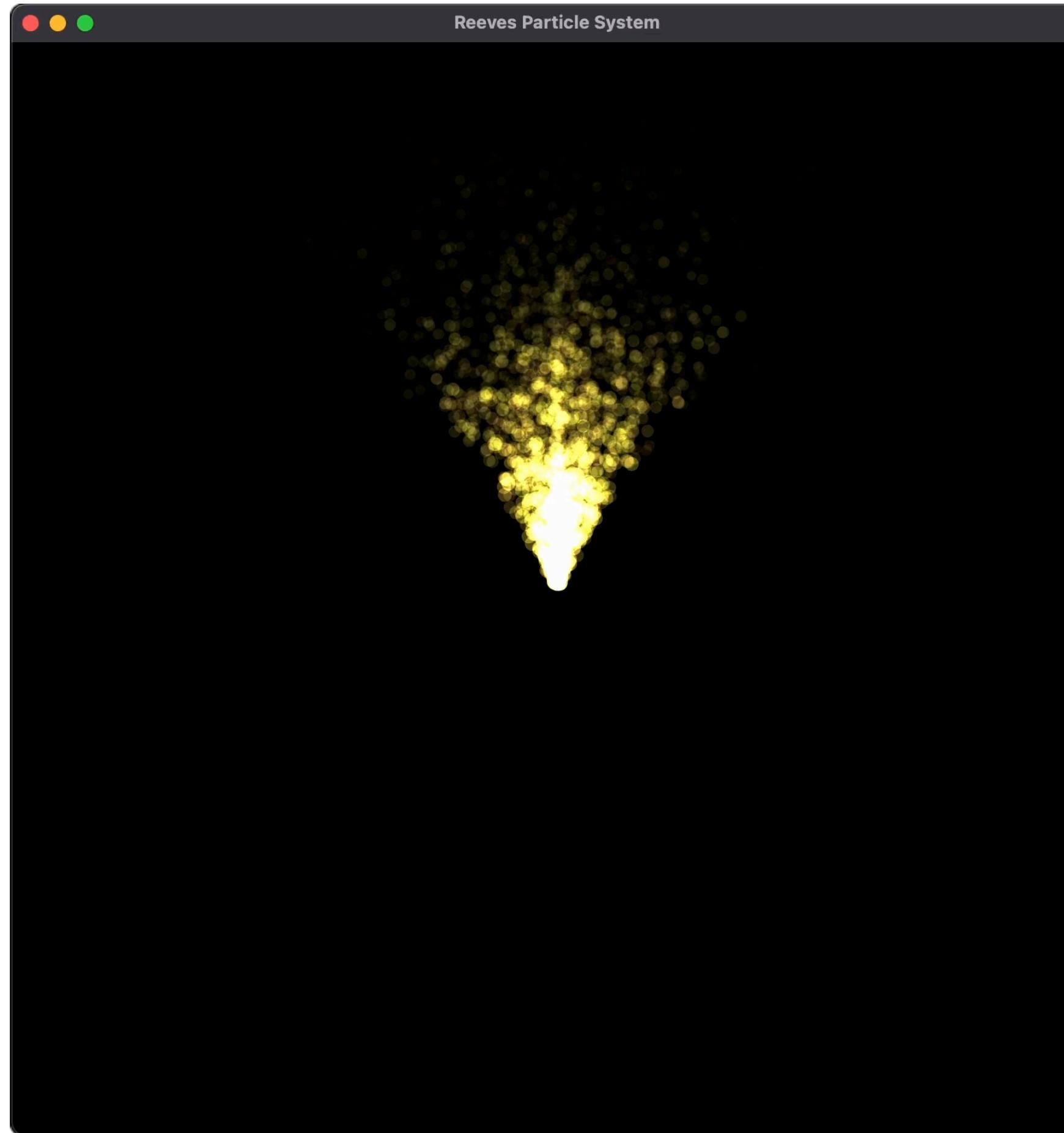
```
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluPerspective(45.0f, aspect, 0.1f, 1000.0f);
```

Setup della vista :

```
gluLookAt(camX + panX, camY + panY, camZ,
          panX, panY, 0.0f,
          0.0f, 1.0f, 0.0f);
glScalef(scene_scale_x, scene_scale_y, 1.0f);
```

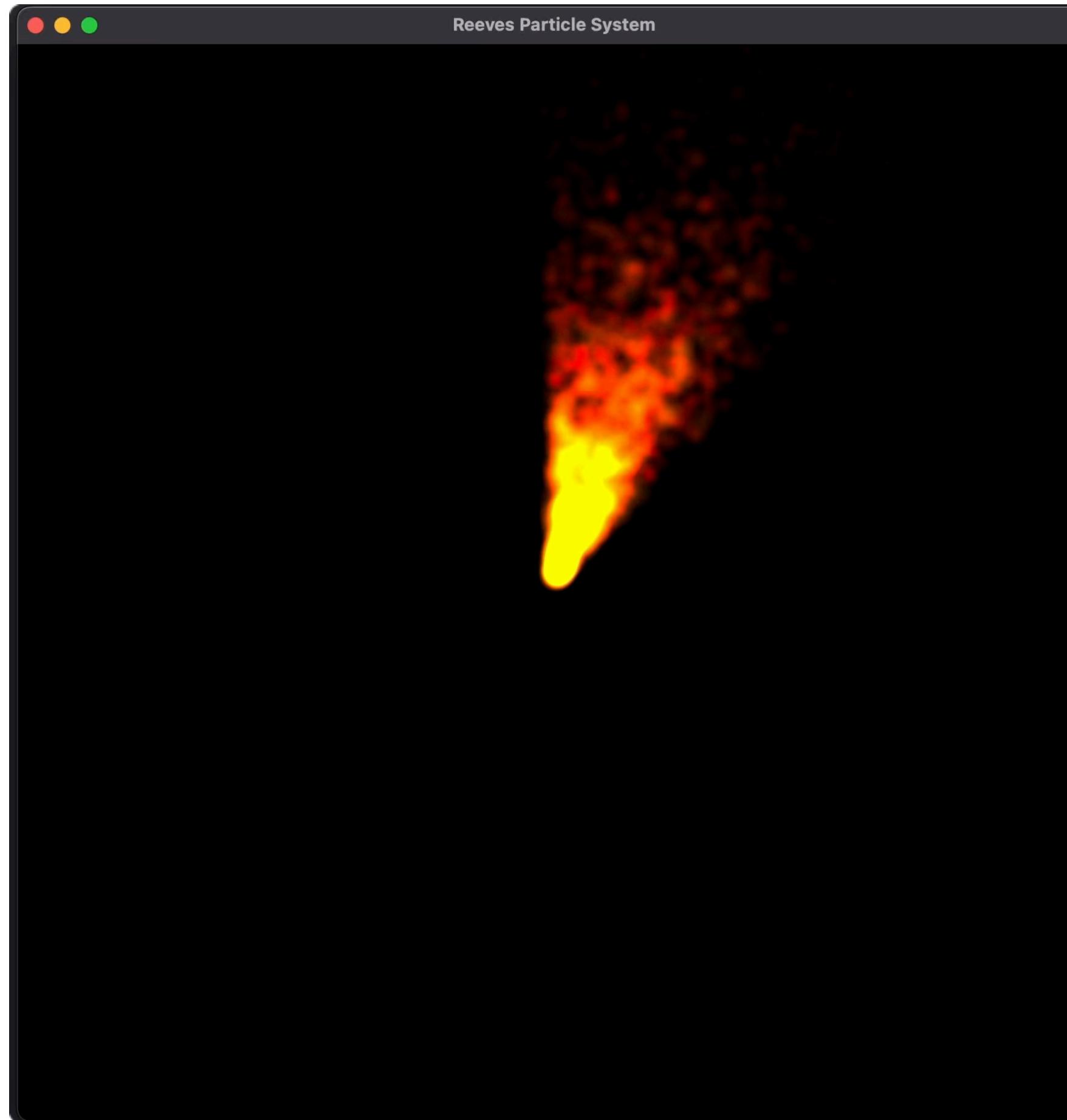
Implementazione in OpenGL - Particelle Transitorie

Codice della Scena 1



Implementazione in OpenGL - Particelle Transitorie

Codice della Scena 2



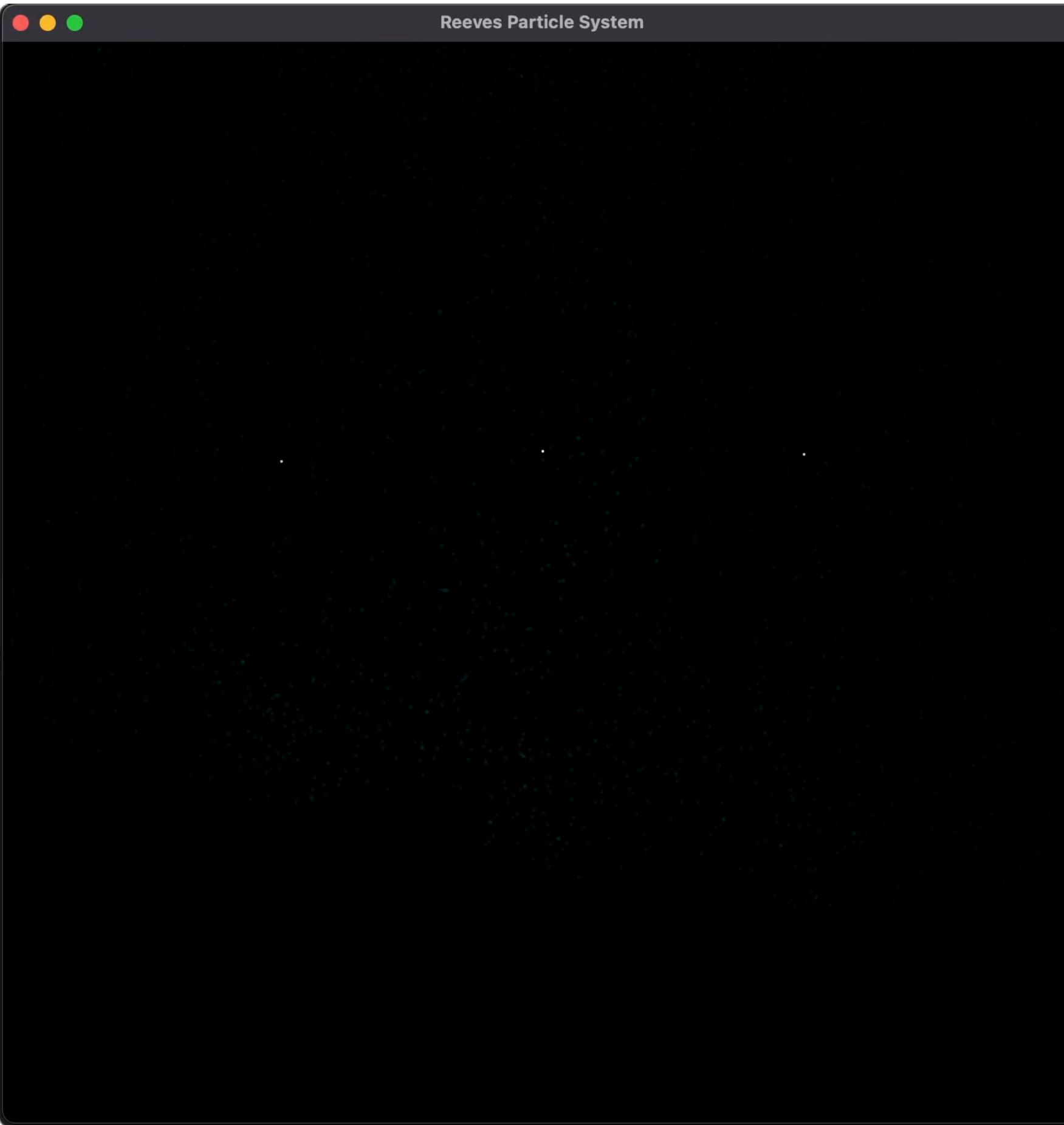
Implementazione in OpenGL - Particelle Transitorie

Codice della Scena 3



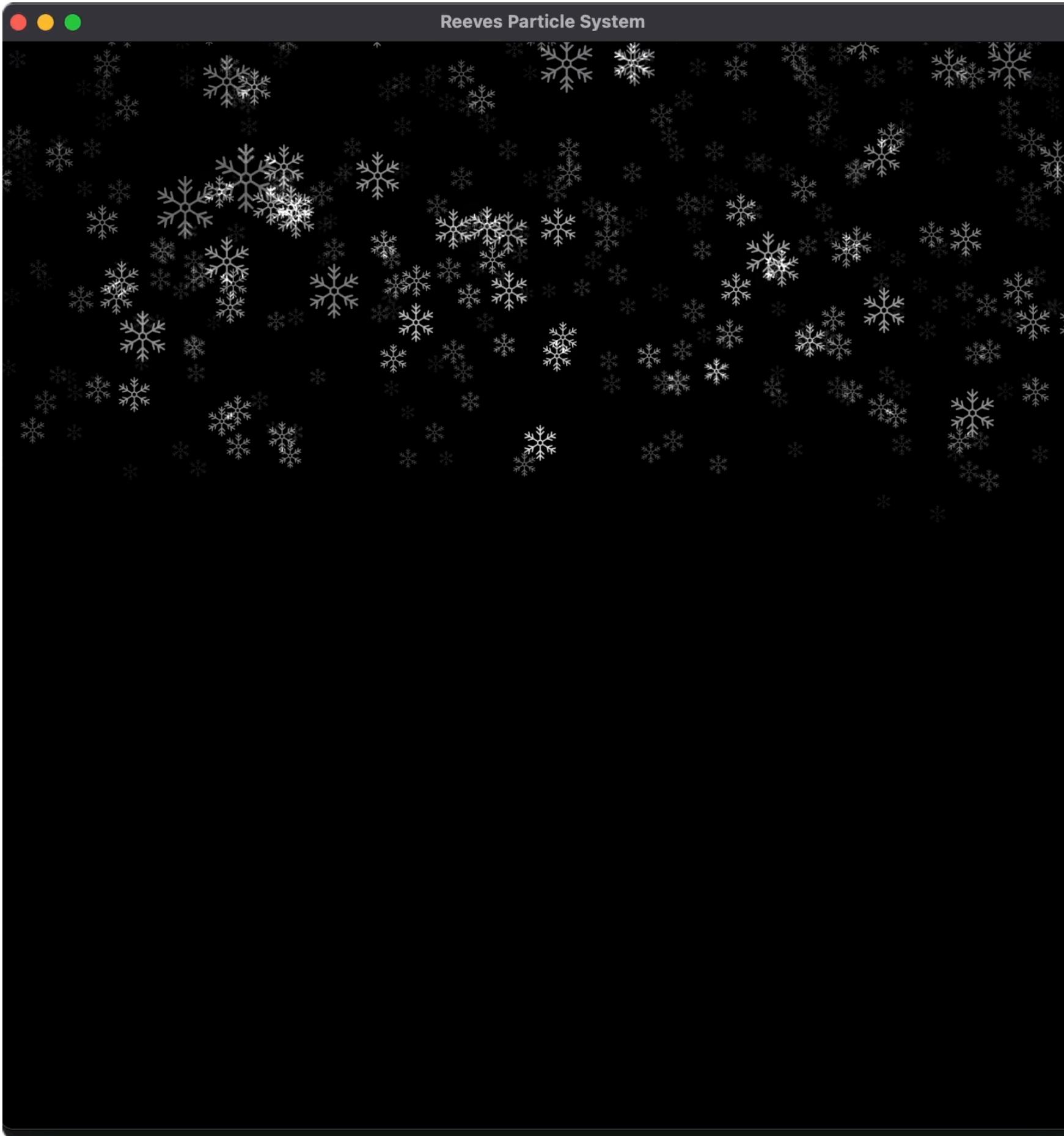
Implementazione in OpenGL - Particelle Transitorie

Codice della Scena 4



Implementazione in OpenGL - Particelle Transitorie

Codice della Scena 5



Implementazione in OpenGL - Particelle Transitorie

Codice della Scena 6



Implementazione in OpenGL - Particelle Filamentose

Inizializzazione scena grafica - Codice

L' impostazioni della scena in OpenGL viene fatta in tal modo :

```
glEnable(GL_LINE_SMOOTH);  
glEnable(GL_DEPTH_TEST);
```

Setup della camera:

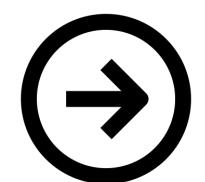
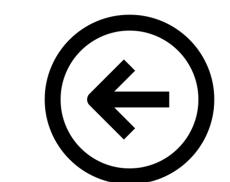
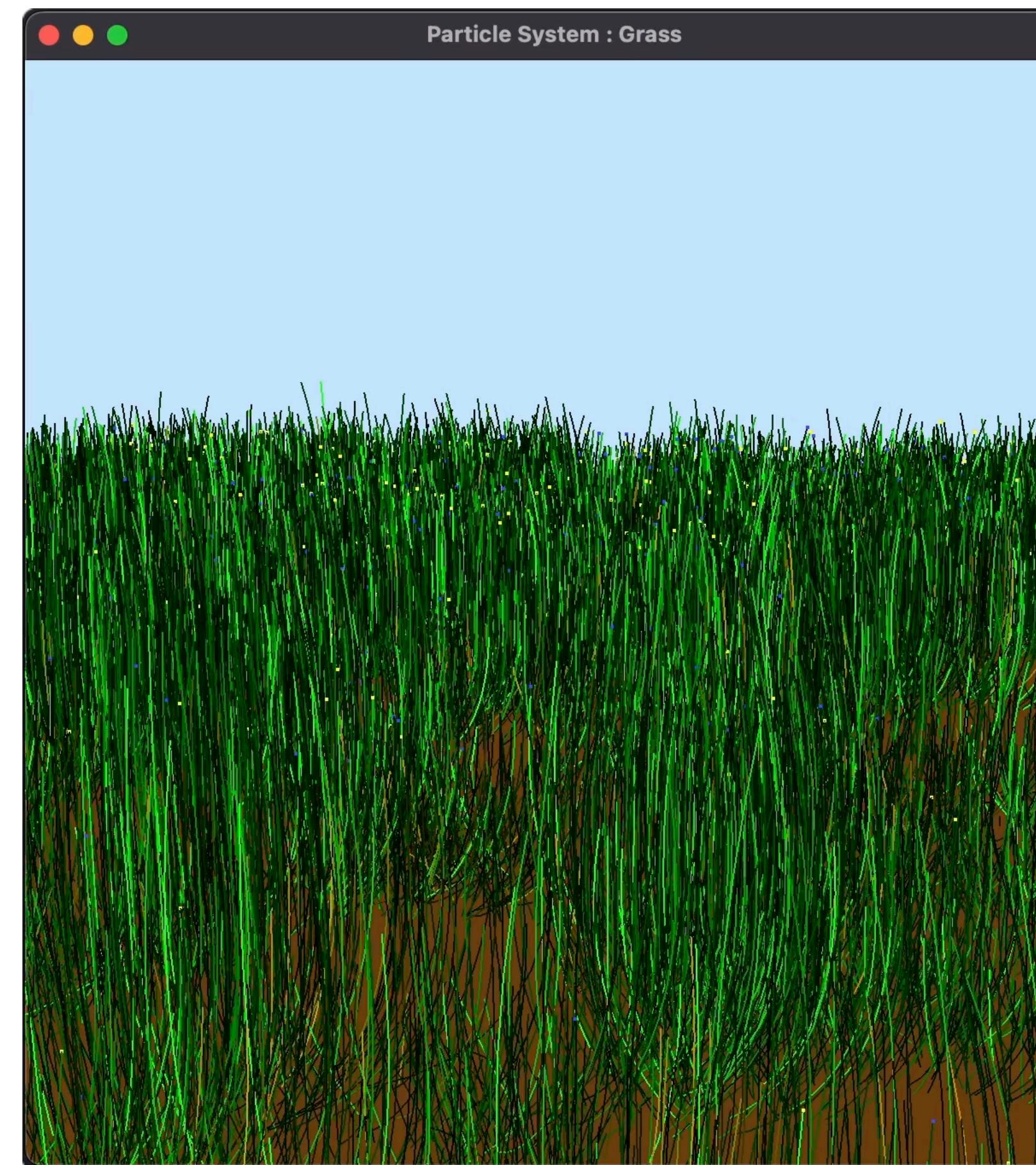
```
glMatrixMode(GL_PROJECTION);  
glLoadIdentity();  
gluPerspective(45.0, ratio, 0.1f, 100.0f);
```

Setup della vista :

```
gluLookAt(camX, 2.0, camZ,  
          camX, 0.5, 0.0,  
          0.0, 1.0, 0.0);
```

Implementazione in OpenGL - Particelle Filamentose

Codice della Scena 1



Conclusioni

Vantaggi

- **Modellazione efficiente di fenomeni naturali non solidi** (fuoco, fumo, erba, acqua) difficilmente gestibili tramite mesh deterministic.
- **Supporto nativo per effetti visivi avanzati:** motion blur, fading, trasparenze, easing non-lineari, emissione condizionale.
- **Algoritmi approssimativi e stocastici**(sampling, randomizzazione controllata) : riducono drasticamente il costo computazionale rispetto a soluzioni deterministiche.
- **Proceduralità e controllo gerarchico (padre-figlio)** : facilitano l'animazione di sistemi complessi e organici.
- **Elevata scalabilità:** rappresentazione compatta (punti, sprite, linee) con ottimo rapporto qualità/prestazioni anche su GPU.

Svantaggi

- **Assenza di interazione tra particelle:** nessuna collisioni, fusioni o turbolenze realistiche.
- **Imprevedibilità stocastica:** piccole variazioni nei parametri iniziali possono portare a grandi divergenze nei risultati.
- **Limitazioni di rendering fisico:** mancano scattering volumetrico, rifrazione e riflessione accurata, rispetto a metodi basati su path-tracing.
- **Ridotta versatilità geometrica:** il sistema assume geometrie semplici, non adatto a modelli ramificati o scene forestali complesse.

Futuri sviluppi

- **Combinare** : particelle con mesh per oggetti solidi per sfruttare i punti di forza di entrambi.
- **Shadow volumes/3D texture** : usare per gestire scene con oggetti sovrapposti.
- **Estensione del sistema** : a modelli ramificati o scene forestali complesse.
- **Shader personalizzati e simulazioni ambientali** : aggiungere simulazioni ambientali dinamiche (ostacoli, stagionalità).
- **Ottimizzazione** : tramite LOD (Level of Detail) e pipeline parallele su GPU.

Grazie per l'attenzione !