Contents

● Introduction
● Background
● Previous Works
● Our Ideas
Introduction
Mirage

inferior mirage

superior mirage
Mirage

inferior mirage

superior mirage

Both by Ludovica Lorenzelli
Our Goal

We want to render realistic, artistic mirages.
Background
Mirage Image: The Physics
Mirage Image: The Physics

High $T \rightarrow$ Low $\rho \rightarrow$ High $\nu \rightarrow$ Low $n$

Low $T \rightarrow$ High $\rho \rightarrow$ Low $\nu \rightarrow$ High $n$

By Ludovica Lorenzelli
Mirage Image: The Physics

High $T \rightarrow$ Low $\rho \rightarrow$ High $\nu \rightarrow$ Low $n$

Low $T \rightarrow$ High $\rho \rightarrow$ Low $\nu \rightarrow$ High $n$

By Ludovica Lorenzelli
Mirage Image: The Physics

High $T$ $\rightarrow$ Low $\rho$ $\rightarrow$ High $\nu$ $\rightarrow$ Low $n$

Low $T$ $\rightarrow$ High $\rho$ $\rightarrow$ Low $\nu$ $\rightarrow$ High $n$

By Ludovica Lorenzelli
Mirage Image: The Physics

superior mirage

inferior mirage
Previous Works
Previous Works

- Modeling refractive index of atmosphere
- Rendering through refractive index model
- Artistic editing by user
Refractive Index Modeling Of Atmosphere

- Simulation of atmospheric phenomena [GSM*06]
- Atmospheric Profile Manager (APM)
- Simplified model of temperature, refractive index similar to real atmosphere
Refractive Index Modeling Of Atmosphere

Background

Inversion Layer

Hot Spot

From [GSM*06]

From [GSM*06]

From slides of [Choi 17]
Rendering Through Refractive Index Model

- Refractive radiative transfer equation [ABW14]
- Applies Hamiltonian optics to the rendering equation

\[ \frac{dv}{ds} = \nabla_x n, \]
\[ \frac{dx}{ds} = \frac{v}{n}. \]
Artistic Editing

- Interactive reflection editing [ROT*09]
- User controllable reflection, defying physics
- Intuitive UI
Artistic Editing

- State of the art in artistic editing of appearance, lighting and material [SPN*14]
- User controlled lighting, BRDF, etc.
- Goal based interaction; user defines goal, system solves.

Images from [ABW14]
Previous Approaches To Mirage Rendering

- Visual simulation of heat shimmering and mirage [ZHF*07]
- Build refractive index from object heat map, renders with ray marching
Artistic Editing
For Mirage Image
Artistic Editing For Mirage Image [Choi 2017]

Supplementary Video: How does the User Interface Works

Artistic Editing for Mirage Image Generation

Submission ID paper1156
Overview

- **Source-to-Destination Point Pair**
  - The user specifies constraints, defined as a set of original position (source) + mirage image position (destination)
- **Perform physically correct** light path calculation
- **Use parametric optimization** to calculate
  - With an appropriate refractive index distribution model for the optimization

This slide is copied from slides of [Choi 17]
Source-to-Destination Point Pair

This slide is copied from slides of [Choi 17]
Source-to-Destination Point Pair

This slide is copied from slides of [Choi 17]
Source-to-Destination Point Pair

This slide is copied from slides of [Choi 17]
Refractive Index Distribution Model

Adapt $\text{APM}[\text{GSM}^\ast 06]$ as a spatial encoding

Images from [GSM*06]
Refractive Index Distribution Model

Plot of **APM** vs. **New Formulation**

This slide is copied from slides of [Choi 17]
Refractive Index Distribution Model

New formulation suggested for the refractive index distribution: **Logistic Approximation**

\[
n(h) = \frac{k_s}{|n_{bg}| + |n_{inv}| + |n_{hotspot}|} \cdot \left( f_{\text{logistic}}(n_{bg}, a_{bg}, 0, h) \\
+ f_{\text{logistic}}(n_{inv}, -a_{inv}, h_{ciso}, h) \\
+ f_{\text{logistic}}(n_{hotspot}, a_{hotspot}, 0, h) \right) + 1,
\]

\[
f_{\text{logistic}}(L, k, x_0, x) := \frac{L}{1 + e^{(x_0 - x)/k}}.
\]

\[
k = \begin{bmatrix} n_{bg} & n_{inv} & n_{hotspot} & h_{ciso} & a_{bg} & a_{inv} & a_{hotspot} & k_s \end{bmatrix}^T.
\]
Spatial Encoding and Optimization

- Let the $m$-dimensional spatial encoding of $\mathbf{n}(\mathbf{x})$ be $\mathbf{L}$

- And let the parameter vector for $\mathbf{L}$ be $\mathbf{k} \in \mathbb{R}^m$
Spatial Encoding and Optimization

- Distance Function
  - Define distance per point pair, as the shortest distance between the Source Point and the light path.
Spatial Encoding and Optimization

- **Distance Function**
  - For i-th pair, spatial encoding $L$, and parameter vector $k$
  
  $$\text{dist}_{L,i}(k) := \inf \{ \| x^*_L(k), i(s) - x_{src_i} \| : s \in [s_0, s_{\text{max}}] \}$$
Spatial Encoding and Optimization

- **Cost Function**
  - For i-th pair, spatial encoding $L$, and parameter vector $k$
    \[
    \text{cost}(k) := \sum_i \text{dist}_{L,i}(k)
    \]
  - solve by **Regression**
    \[
    \arg\min_k \text{cost}(k)
    \]
Rendering method

- Ray-marching algorithm

Image from https://computergraphics.stackexchange.com/questions/161/what-is-ray-marching-is-sphere-tracing-the-same-thing
Rendering method

- Ray-marching algorithm

\[
\frac{dv}{ds} = \nabla_x n, \\
\frac{dx}{ds} = \frac{v}{n}.
\]

for (float s = 0; s < maxStep; s += stepDelta)
{
    dx1 = stepDelta * v / n(x);
    dv1 = stepDelta * gradN(x);
    dx2 = stepDelta * (v + dv1 / 2.0) / n(x + dx1 / 2.0);
    dv2 = stepDelta * gradN(x + dx1 / 2.0);
    dx3 = stepDelta * (v + dv2 / 2.0) / n(x + dx2 / 2.0);
    dv3 = stepDelta * gradN(x + dx2 / 2.0);
    dx4 = stepDelta * (v + dv3) / n(x + dx3);
    dv4 = stepDelta * gradN(x + dx3);
    dx = (dx1 + 2.0 * dx2 + 2.0 * dx3 + dx4) / 6.0;
    dv = (dv1 + 2.0 * dv2 + 2.0 * dv3 + dv4) / 6.0;
    x += dx;
    v += dv;
    /* do something here */
    vec4 projPos = (ProjectionMatrix * ViewMatrix * vec4(x, 1));
    vec3 projPosN = projPos.xyz/projPos.w;
    vec2 depthTexCoordHere = (projPosN.xy + vec2(1.0)) / 2.0;
    if (projPosN.z > texture2D(depthTex, depthTexCoordHere).r)
    {
        color = texture2D(colorTex, depthTexCoordHere).rgb;
        isIntersection = true;
    }
}
Limitations

● This system is focused on creating static scenes
  ○ This formulation does not consider the spatial location of the hot spot and the inversion layer
  ○ When you move the camera, you’ll see a different scene than you intended
● No illumination model
● Poor, unintuitive UI

⇒ Unrealistic mirage scenes
Our Ideas
Areal Hot Spot

- Hot spot is no longer depends on only height
- Hot spot has xy-position and size, defined by user
- Optimize additional parameters using the existing optimizer
- By changing the formula to take into account the area of the hot spot, we can see a more realistic scene even if we move the camera
Areal Hot Spot
Atmospheric Noise

- Temperature variance is not smooth in real
- Air turbulences result in noises in real image
- We can add some noises to the scene for more realistic image
  - Temperature noise to model
  - Geometric noise to ray
Local Illumination Model: Phong Illumination

- Phong Illumination
- Complex global illumination is overkill in large outdoor scene with bright sunlight
- Phong illumination is simple and plausible enough
Texture Filtering

- Mip-mapping
- Anisotropic filtering
- Ray differential
**Better UI**

- Point positioning
- Camera positioning
- More interactive point addition and rendering; user should be able to see result of point addition more frequently
- Point importance
Better UI: Point Importance

- User defines optimization rate (intensity) for each point
- Optimizer has larger margins for less important points
Members & Roles

- Seo Hansol: Setting up the environment, Presentation, Coding

- Lim Mingi: Presentation, Coding
Q & A