Fundamentals of 3D

Lecture 6:
Metric ball trees/Texture synthesis
Advanced coordinate pipelines
Fourier analysis/interpolation

Frank Nielsen
nielsen@lix.polytechnique.fr
19th October 2011
``Texture Synthesis by Non-parametric Sampling''

Alexei A. Efros and Thomas K. Leung

IEEE International Conference on Computer Vision (ICCV'99),

http://graphics.cs.cmu.edu/people/efros/research/EfrosLeung.html
Stochastic texture synthesis

Source Image $I_s$

Target Image $I_t$

Scanline

$\text{L-shape window}$

$2s + 1$

$S$

SSD$(x_s, y_s; x_t, y_t) = \sum_{l=0}^{s} \sum_{c=0}^{s} \text{LShape}(l, c) \left( I_s[x_s + c, y_s + l] - I_t[x_t + c, y_t + l] \right)^2$

$(x_s, y_s) = \text{argmin}_{(x,y) \in I_s} \text{SSD}(x, y; x_t, y_t)$

Fast nearest neighbor queries in high dimensions

2011 Frank Nielsen
Ball tree data structures for nearest neighbor search

Compute a k-means on $S$ with $k=2$
Split $S$ into $S_1$ and $S_2$ according to the two centroids
Perform recursion on $S_1$, and $S_2$ until $|S_1|<n_0$ and $|S_2|<n_0$
Nearest neighbor queries using ball trees

Pruning some of the nodes:
Let $\text{NN}(q)$ denote the current best nearest neighbor of $q$

if $||q-c|| - r_q > r_c$ then PRUNE (do not explore the subtree)

At leaves, perform the naive linear search, and potentially update $\text{NN}(q)$
Careful seeding for k-means: Perform just a careful initialization!!

Interpolate between the two methods:

Let $D(x)$ be the distance between $x$ and the nearest cluster center. Sample proportionally to $(D(x))^\alpha = D^\alpha(x)$

Original Lloyd’s: $\alpha = 0$

Furthest Point: $\alpha = \infty$

k-means++: $\alpha = 2$

1a. Choose an initial center $c_1$ uniformly at random from $\mathcal{X}$.

1b. Choose the next center $c_i$, selecting $c_i = x' \in \mathcal{X}$ with probability $\frac{D(x')^2}{\sum_{x \in \mathcal{X}} D(x)^2}$.

1c. Repeat Step 1b until we have chosen a total of $k$ centers.

Theorem: k-means++ is $\Theta(\log k)$ approximate in expectation
Vp-trees worked best (=fastest) for image patches....

[EECV'08]
Vantage point trees (or vp-trees)

Partition the data according to a vantage point and a distance threshold. Relative distances are thus used.

First split

Split from a vantage point:
- Inner part
- Outer part

do split recursion
Vantage point trees: Pruning condition

If $d(q, p) > r_p + r$ prune the inner branch
If $d(q, p) < r_p - r$ prune the outer branch

For $r_p - r \leq d(q, p) \leq r_p + r$ we have to inspect both branches

Prune outer  Prune inner  Cannot prune
Many ways to partition the point sets

(a) $kd$-Tree
(b) PCA Tree
(c) Ball Tree
(d) $vp$-Tree

Construction Cost

Search Improvement

- $\epsilon$–NN Search
- $k$–NN Search

Comparison of construction cost and search improvement for different tree structures.
GPGPU: General Purpose GPU

<table>
<thead>
<tr>
<th>Descripteur</th>
<th>Dimension</th>
<th>ANN-C++</th>
<th>BF-CUDA</th>
<th>Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>3</td>
<td>1m 33s</td>
<td>53s</td>
<td>1.8</td>
</tr>
<tr>
<td>CP</td>
<td>5</td>
<td>2m 05s</td>
<td>1m 05s</td>
<td>1.9</td>
</tr>
<tr>
<td>CG</td>
<td>5</td>
<td>2m 35s</td>
<td>1m 07s</td>
<td>2.3</td>
</tr>
<tr>
<td>CGP</td>
<td>7</td>
<td>4m 27s</td>
<td>1m 19s</td>
<td>3.3</td>
</tr>
<tr>
<td>C₃</td>
<td>11</td>
<td>6m 40s</td>
<td>1m 17s</td>
<td>5.2</td>
</tr>
<tr>
<td>C₃P</td>
<td>13</td>
<td>5m 43s</td>
<td>1m 12s</td>
<td>4.8</td>
</tr>
</tbody>
</table>

Distance au k=2 plus proche voisin

C: couleur (YUV)
C₃: 3x3 neigh in Y
G: gradient
P: position
k=3
Log-Polar coordinates

$$\rho = \log \sqrt{x^2 + y^2},$$
$$\theta = \arctan \frac{y}{x}.$$
Log-Polar coordinates

Scales and rotations become translations

\[ S_s \mathbf{x} = (sx, sy) \longrightarrow (\log s + \rho(\mathbf{x}), \theta(\mathbf{x})) , \]

\[ R_\phi \mathbf{x} = (x \cos \phi + y \sin \phi, y \cos \phi - x \sin \phi) \longrightarrow (\rho(\mathbf{x}), \phi + \theta(\mathbf{x})) . \]

Data reduction for retinal images...
Spherical coordinates

\[ r = \begin{bmatrix} \cos \phi \sin \theta \\ \sin \phi \\ \cos \phi \cos \theta \end{bmatrix} \]

\[ \theta = \arctan \frac{x}{z} \quad \text{and} \quad \phi = \arctan \frac{y}{\sqrt{x^2 + z^2}}. \]
Spherical coordinates

High Resolution Full Spherical Videos
Frank Nielsen, ITCC'02

(a) (b) (c) (d) (e)

(f) (g) (h) (i) (j)

(k) (l)
Cylindrical coordinates

\[ \theta = \arctan \frac{x}{z}, \quad s = \frac{y}{\sqrt{x^2 + z^2}}. \]

Panoramic image stitching:

Align by a translation into the cylindrical coordinate map...
Environment maps

Equirectangular, mirror ball, cubic, etc.
Environment maps: Mirror ball

Orthographic Projection (Camera)

Unused portion

Chrome Ball

Sphere Map Texture Image

Mapping

\[
\begin{bmatrix}
    u \\
    v
\end{bmatrix} = \begin{bmatrix}
    \frac{r'_x}{2\sqrt{r_x^2 + r_y^2 + (r'_z + 1)^2}} + \frac{1}{2} \\
    \frac{r'_y}{2\sqrt{r_x^2 + r_y^2 + (r'_z + 1)^2}} + \frac{1}{2}
\end{bmatrix}
\]
Environment maps for reflections

Blinn, 1976

1982

Interface, 1985
Lance Williams

http://www.debevec.org/ReflectionMapping/
Environment maps for reflections


Best environment map for real-time graphics?

Dual paraboloid (2 images front/back only)
Transformations and their invariants
Taxonomy of projections:
Multiple centers of projections (MCOP)

Reconstruction from a single MCOP images
Generalizes epipolar geometry
Resolution dependent
Acquisition Example
Multiple centers of projections (MCOP)

Difficult to obtain in practice...
Localization is difficult
Multiple centers of projections (MCOP)

The art of depiction...
Stereo cyclographs...
Image backward vs forward mapping

Image warping

FORWARD_MAPPING($I_s, f$)
1. $\triangleright$ Create a warped image $I_d$ by forward mapping $\triangleright$
2. $\triangleright f$: warping function $\triangleright$
3. Initialize an empty image $I_d$
4. $\triangleright$ for all image lines $\triangleright$
5. for $y \leftarrow 1$ to $h_d$
6. $\triangleright$ for all column pixels $\triangleright$
7. for $x \leftarrow 1$ to $w_d$
8. $\triangleright$ Compute the source-to-destination mapping $\triangleright$
9. $(u, v) \leftarrow f(x, y)$
10. $\triangleright$ Round coordinates to integers $\triangleright$
11. $\triangleright$ (no interpolation required) $\triangleright$
12. $(u_r, v_r) \leftarrow ([u], [v])$
13. $\triangleright$ Should check index bounds $\triangleright$
14. $I_d[u_r, v_r] = I_s[x, y]$

BACKWARD_MAPPING($I_s, f$)
1. $\triangleright$ Destination image $I_d$ of dimension $w_d \times h_d$ $\triangleright$
2. for $v \leftarrow 1$ to $h_d$
3. $\triangleright$ for all column pixels $\triangleright$
4. do for $u \leftarrow 1$ to $w_d$
5. $\triangleright$ Backward mapping requires resampling $\triangleright$
6. $I_d[u, v] = \text{RESAMPLE}(I_s, x, y)$

Resampling
Interpolation
Image Blending: Alpha channel

\[ I[i,j] = \alpha[i,j]F[i,j] + (1 - \alpha[i,j])B[i,j], \]

\[ I = \alpha F + (1 - \alpha)B, \]

Interpretation at the microscopic level...
Image sampling/reconstruction

Discrepancy ? → Reconstruction → Continuous functions
(Analog world) ← ← Discrete images
(Digital devices)

Aliasing

Sampling

Rate ?
Zone plate: Aliasing/ringing effect

1024x1024

256x256
downsampled
using bilinear interpolation
Continuous versus discrete convolutions

\[(f \otimes g)(x) = \int_{t=-\infty}^{\infty} f(t)g(x-t)dt = \int_{t=-\infty}^{\infty} g(t)f(x-t)dt = (g \otimes f)(x).\]

\[C[i, j] = A \otimes B = \sum_{k} \sum_{l} A[k, l]B[i - k, j - l].\]

\[G = \frac{1}{273} \begin{bmatrix}
1 & 4 & 7 & 4 & 1 \\
4 & 16 & 26 & 16 & 4 \\
7 & 26 & 41 & 26 & 7 \\
4 & 16 & 26 & 16 & 4 \\
1 & 4 & 7 & 4 & 1
\end{bmatrix} \]
Fourier analysis

\[ f(x + T) = f(x) \]

Fourier discovered that all periodic signals can be represented as a sum (eventually infinite) of sinusoidal waves: \( \sin(\cdot) \) functions, the basis functions.

Let \( f(\cdot) \) denote the continuous function in the spatial domain and \( F(\cdot) \) denote the dual complex function, also called spectral function.

Euler formula (period 2\( \pi \))

\[
\exp(ix) = \cos x + i \sin x
\]

\[
\exp(ix) = \cos x + i \sin x = \cos(x + 2\pi) + i \sin(x + 2\pi) = \exp(i(x + 2\pi))
\]
Fourier analysis: Duality spatial/spectral domain

**Spatial domain**

\[ f(x, y) = \int_{u=-\infty}^{\infty} \int_{v=-\infty}^{\infty} F(u, v) \exp\left(i2\pi(ux + vy)\right) dudv. \]

**Spectral domain**

\[ F(u, v) = \int_{x=-\infty}^{\infty} \int_{y=-\infty}^{\infty} f(x, y) \exp\left(-i2\pi(ux + vy)\right) dx dy. \]

\[ F(u, v) = A(u, v) + iB(u, v) \]
Fourier analysis: Phase/amplitude

\[ F(u, v) = A(u, v) + iB(u, v) \]

Polar coordinates:

\[ F(u, v) = |F(u, v)| \exp(i\phi(u, v)) \]

\[ |F(u, v)| \]

\[ \phi(u, v) = \arctan \frac{B(u, v)}{A(u, v)} \]

Frequency magnitudes

\[ P(u, v) = A^2(u, v) + B^2(u, v) \]

Power spectrum

Amplitude \hspace{1cm} Phase
Fourier analysis: Convolution theorem

Convolution in spatial domain is a multiplication in Fourier domain

\[ \mathcal{F}(f \otimes g) = \sqrt{2\pi} (\mathcal{F}f) \times (\mathcal{F}g) = \sqrt{2\pi} F \times G \]

Convolution in frequency domain is a multiplication in spatial domain

\[ F \otimes G = \sqrt{2\pi} \mathcal{F}(f \times g) \]
Fourier analysis: Discrete transformations

1D

\[ f_j = \frac{1}{n} \sum_{k=0}^{n-1} x_k \exp(-2\pi i \frac{j k}{n}), \quad \forall \ 0 \leq j \leq n - 1 \]

\[ x_k = \sum_{j=0}^{n-1} f_j \exp(2\pi i \frac{j k}{n}) \]

2D

\[ F(u, v) = \frac{1}{w h} \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} f(x, y) \exp \left(-2\pi i \left( \frac{x u}{w} + \frac{y v}{h} \right) \right) \]

\[ f(x, y) = \sum_{u=0}^{w-1} \sum_{v=0}^{h-1} F(u, v) \exp \left(2\pi i \left( \frac{x u}{w} + \frac{y v}{h} \right) \right) \]
Interpolation/reconstruction filters

Bilinear interpolation

\[ c_{x,y} = (1 - x')y' c_{i,j+1} + x' y' c_{i+1,j+1} + (1 - x')(1 - y') c_{i,j} + x'(1 - y') c_{i+1,j} \]
Interpolation/reconstruction filters

Sinc (Lanczos) is ideal low-pass filter (infinite support)

Fourier transform

The sinc function
\[ \text{sinc}(x) = \frac{\sin \pi x}{\pi x} \]

Spatial domain

Windowed sinc

Fourier domain

Ideal low-pass filter
Mr Angry
Mrs Calm

Low/high frequency perception
Phase correlation

Stitch by 2D translation two images

\[ f_2(x, y) = f_1(x + x_t, y + y_t) \]

\[ F_2(u, v) = F_1(u, v) \exp(-2\pi i(u x_t + v y_t)) \]

\[ \frac{F_1(u, v)F_2^*(u, v)}{|F_1(u, v)F_2^*(u, v)|} = \exp(2\pi i(u x_t + v y_t)) \]

Cross-power spectrum

FFT can be computed in \( O(n \log n) \) time

\( F^* \) is the conjugate function

2011 Frank Nielsen
Phase correlation

\[ \frac{F_1(u, v)F_2^*(u, v)}{|F_1(u, v)F_2^*(u, v)|} = \exp(2\pi i (ux_t + vy_t)) \]

Cross-power spectrum

Algorithme: Calculer le cross-power spectrum des deux images

Calculer la transformation inverse FFT, et chercher le sommet dans l'image spatiale

Extend to rotation and scale using the log-polar transform

(a) input image
(b) scale=4; rotation=45°
(c) log-polar transform of (a)
(d) log-polar transform of (b)
Phase correlation

Extend up to affine transformations


Phase correlation: Detecting the peak

ClairVoyance: A Fast and Robust Precision Mosaicing System for Gigapixel Images

Sub-pixel accuracy if we fit a quadratic function

Clairvoyance system
IEEE IECON 2006
Veuillez commencer vos projets sous Processing et autres Java APIs

Utilisez JMyron sous Processing pour capturer la webcam
http://webcamxtra.sourceforge.net/

Downloads

For updates and support, join the email list.

Processing Library (Popular)
- Download JMyron 0025 for Processing. Includes example projects to help get you started.

MaxMSP External
- Myron for MaxMSP 0021 for Max MSP on OS X - unstable and still in development.

C++
- Do an SVN checkout of the webcamxtra project from Sourceforge. See BUILD.txt for instructions on getting the C++ compiling.

Source
- For revisions 0025 and beyond, we've moved the source to the SourceForge SVN server, so check out the module there. For older versions, the CVS is still open for co's, just closed for ci's. PLEASE submit your patches to this project. Chances are - whatever you added to the code is

Director Xtra
- Download MyronXtra 0025 for Director. Includes example projects to help get you started.

Python
- The first pyMyron alpha support will be available later, a preview to give the developer (Max Oh) some feedback on the email list. Still in early development.

Java
- A usable jar is included in the Processing download. Here is an Eclipse example contributed by Shawn Van Every

Older Versions
- Older releases of this project are released on the SourceForge file releases page.