An introduction to image based rendering

Francesco Isgrò
What are we going to see?

- What is image based rendering (IBR)?
- A taxonomy of IBR techniques
- Technical details of few techniques
The concept of view synthesis

- Target: generation of realistic images of a 3D scene
- Applications: computer games, virtual reality, augmented reality
- It has been a Computer Graphics field for years
3D Model → Light Model → Camera Model → Image
- Model: typically a CAD model
  - Artificial model
  - 3D sensors (time of flight, triangulation sensors, .....)
- Colours obtained mapping a texture onto the model
  - Artificial texture
  - Texture obtained from real images
Real images capture objects in real illumination conditions

Starting from real images we can create photo-realistic synthetic images

No need of a complex model: details are in the images

No need of physical simulation: realism is in the images
Two main classes of methods:
- CAD-like approach
- Image-based rendering
CAD-like modeling
A geometric model is recovered from the images using stereopsis

Texture mapped from original images

Stress is on appearance not on geometric accuracy

Needs calibration
Example: the CMU dome

- 51 calibrated cameras
- stereo-matching
- shape reconstruction
- texture mapping from real images
- off-line computation
No 3D model
Scene represented as collection of images
New images created directly from original images
Camera calibration is not required, but it helps
Taxonomy of IBR methods

- Mosaicing
- Interpolation from dense samples
- Non-geometrically valid image mapping
- Geometrically valid image mapping
IBR vs. CAD-like

- IBR methods are very popular
- Both of them need correspondences
- Correspondences = Depth
- IBR is a shortcut
Off-line processing
- CAD-like require less information (only 3D model stored)

On-line processing
- 3D model computation may result too expensive
Mosaicing

- Constructing a panoramic image from a set of images
- Assume the scene projectively planar
  - Not too much depth variation
  - Camera rotating around optical center
- Sparse correspondences needed
Example of mosaicing

- A set of points is tracked across the sequence
- Bad matches removed
- Computing homography
- Each frame warped back to the first frame reference frame
Non geometrically valid mapping

- Image morphing:
  - gradually transform one image into another one
  - needs at least a set of sparse correspondences

- View interpolation[Chen93]:
  - interpolation of two views of the same scene
  - needs dense correspondences
Image morphing: examples
Mainly used to create *in-between* novel views (views on the cameras baseline)

Input is a stereo pair $I$ and $I'$

Dense correspondences are computed

If $p$ and $p'$ are corresponding points then the new point $p_\lambda$ in $I_\lambda$ is

$$p_\lambda = p + \lambda (p' - p)$$

The gray level assigned to $p_\lambda$ is

$$g_\lambda = g + 0.5 (g' - g)$$
Why not geometrically valid?

Reason is that $I_{\lambda}$ is not obtained from a perspective transformation. For instance

\[ x_{\lambda} = \frac{X}{Z} + \lambda \left( \frac{X'}{Z'} - \frac{X}{Z} \right) = \]

\[ = \frac{XZ'}{Z} + \lambda(X'Z - XZ') \]

\[ ZZ' \]
As observed in [Seitz96] $I_\lambda$ is geometrically valid if $I$ and $I'$ are parallel views. In fact being $Z = Z'$

$$x_\lambda = \frac{X + \lambda(X' - X)}{Z}$$
The algorithm works as

1. Rectify the original stereo pair
2. Linearly interpolate obtaining a rectified intermediate image
3. Un-rectify the intermediate image
Image interpolation works well for generating *in-between* novel views.
More general techniques:
- Epipolar transfer
- Trifocal transfer
- Homography plus parallax
Let us assume all cameras have the same intrinsic parameters, given by the identity matrix. Given three views $I$, $I'$ and $I''$ they identify a set of fundamental matrices

- $F$ between $I$ and $I'$
- $F_1$ between $I$ and $I''$
- $F_2$ between $I'$ and $I''$
The image $I''$ can be synthesised from $I$ and $I'$ in the following way:

- $p$ and $p'$ corresponding points
- $F_1p$ is the epipolar line of $p$ in $I''$
- $F_2p'$ is the epipolar line of $p'$ in $I''$
- $p'$ is the intersection of the two epipolar lines $F_2p'$ and $F_1p$
The previous method fails when the novel image intersects the *trifocal plane*

Less problems using the trifocal transfer
The four lines can be back projected into the four planes 1, 2, 3, 4. The four planes intersect in a point O."
The four lines can be back projected into the four planes 1; 2; 3; 4. The four planes intersect in a point.
The four lines can be back projected into the four planes 1, 2, 3, 4. The four planes intersect in a point.
The four lines can be back projected into the four planes 1, 2, 3, and 4. The four planes intersect in a point.
The four lines can be back projected into the four planes 1, 2, 3, 4. The four planes intersect in a point.
The four lines can be back projected into the four planes 1, 2, 3, 4. The four planes intersect in a point.
The four lines can be back projected into the four planes 1; 2; 3; 4. The four planes intersect in a point.
The four lines can be back projected into the four planes 1; 2; 3; 4. The four planes intersect in a point.
The four lines can be back projected into the four planes 1, 2, 3, 4. The four planes intersect in a point.
The four lines can be back projected into the four planes $\tau_1, \tau_2, \tau_3, \tau_4$
The four lines can be back projected into the four planes $\tau_1, \tau_2, \tau_3, \tau_4$.

The four planes intersect in a point.
The four lines can be back projected into the four planes \( \tau_1, \tau_2, \tau_3, \tau_4 \).
The four planes intersect in a point.
The null-space of
\[
\begin{pmatrix}
\tau_1 \\
\tau_2 \\
\tau_3 \\
\tau_4 \\
\end{pmatrix}
\]
is not null.
The four lines can be back projected into the four planes \( \tau_1, \tau_2, \tau_3, \tau_4 \).
The four planes intersect in a point.

\[
\det \begin{pmatrix}
\tau_1 \\
\tau_2 \\
\tau_3 \\
\tau_4
\end{pmatrix} = 0
\]
\[
\sum_{i=1}^{3}(x''\alpha_{i13}p_i - x''x'\alpha_{i33}p_i + x'\alpha_{i31}p_i - \alpha_{i11}p_i) = 0
\]
\[
\sum_{i=1}^{3}(y''\alpha_{i13}p_i - y''x'\alpha_{i33}p_i + x'\alpha_{i32}p_i - \alpha_{i12}p_i) = 0
\]
\[
\sum_{i=1}^{3}(x''\alpha_{i23}p_i - x''y'\alpha_{i33}p_i + y'\alpha_{i31}p_i - \alpha_{i21}p_i) = 0
\]
\[
\sum_{i=1}^{3}(y''\alpha_{i23}p_i - y''y'\alpha_{i33}p_i + y'\alpha_{i32}p_i - \alpha_{i22}p_i) = 0,
\]
\[
\alpha_{ijk} = e'_j M''_{ki} - e''_k M'_{ji}
\]
Given two corresponding points in two original images, the corresponding point in the target image is given by

\[
x'' = \frac{(\alpha_{i11}p_i - x'\alpha_{i31}p_i)}{(\alpha_{i13}p_i - x'\alpha_{i33}p_i)} \\
y'' = \frac{(\alpha_{i22}p_i - y'\alpha_{i32}p_i)}{(\alpha_{i23}p_i - y'\alpha_{i33}p_i)}.
\]
Interpolation from dense samples

- Build a LUT from many image samples
- Explicitly approximate the *plenoptic function*

\[ P = P(\theta, \phi, \lambda, X, Y, Z, t) \]

- It represents the intensity of the light observed from every position and direction in 3-dimensional space
Under some conditions the plenoptic function can be reduced to a 4 parameters function Lumigraph\cite{Gortler96} Light field\cite{Levoy96}:

- dense sample (hundreds of images)
- no correspondences
- effort is in the sampling
Light field example: Lion

Sampling: array $32 \times 16$

Image size $256 \times 256$
Light field example: Buddha

Sampling: array $16 \times 16$

Image size $256 \times 256$
Needs a large number of image samples
Needs to know exact position of cameras
Sampling needs complex engineering
Works well with simple scenes
Not clear their feasibility for real complex scene