PLENOPTIC PATH AND ITS APPLICATIONS

Frank Nielsen

Sony Computer Science Laboratories Inc.
Nielsen@csl.sony.co.jp

ABSTRACT

In this paper, we present a method for acquiring, spatially filtering and viewing annotated videos captured with a full field of view multi-head camera moving along a path. We describe our tailored ego-motion recovery algorithm used for calculating the trajectory path of the panoramic head. We then focus on sampling the *plenoptic path* efficiently according to geometric visibility events. Appropriate samplings allow us to filter and compress the panoramic images avoiding some redundancies in the image database. We present several applications and results of plenoptic paths either obtained from indoor shootings or perfectly rendered by computer graphics scripts.

Keywords: Image-based modeling & rendering, Plenoptic sampling, Visibility.

1. INTRODUCTION

The quest for realism in computer graphics is an endless goal. On one hand, advances of object space rendering algorithms provide stunning vivid images that however lack of "real-world" feelings. On the other hand, image-based renderings (IBR) proceed in the image space by modeling and analyzing image sources by means of inverse rendering operations to provide compelling 3d environments but lack of scalability and interactivity. IBRs have become popular with the introduction of photorealistic backdrops in [1][1], light fields [2, 3], compressed surface light fields [4] and photorealistic virtual walkthroughs of static environments [5]. Although the later yet in its infancy, the algorithms often proceed by first georeferencing panoramic frames and then synthesizing novel viewpoints by warping and combining source images based on feature matchings.

2. PLENOPTIC FUNCTION AND PATH

The plenoptic concept [2, 3] is based on the observation that if one knows the 7D function $L(\cdot) = L(X, Y, Z, \theta, \phi, t, \lambda)$ that associates to each cartesian point (X, Y, Z) in 3d space $\mathbb{E}^3$, at any moment t in time, for any orientation (θ, ϕ) and for any wavelength λ the spectrum response, then we can bypass the 3d geometric modeling to provide interactive walkthroughs by "rendering" directly from the plenoptic function $L(\cdot)$. In practice, we can relax the high dimensionality of the function, by freezing time (i.e., considering a static environment), choosing one wavelength (i.e., a color channel – for example Red) and restricting the ray sampling (X, Y, Z, θ, ϕ) to a camera path: the 1d *plenoptic path* $\mathcal{P} = \{P_i = (x, y, z)_i\}$. Brute force sampling of $\mathcal{P}$ can be acquired by moving an omnidirectional camera along a path thus discretizing $\mathcal{P}$ as a set of p points $\mathcal{P} = \{P_i\}_{i=1..p}$. From the 3d discrete sampling of $L_{|\mathcal{P}}(\cdot)$, we can either consider inverse rendering problems (e.g., find the macro/meso geometry and texture attributes, lighting conditions, etc.) or proceed by extrapolating the function for view synthesis. Since massive acquisition is time consuming and meticulous, motorized robot carrying a panoramic head has been used for the task in [5, 6].

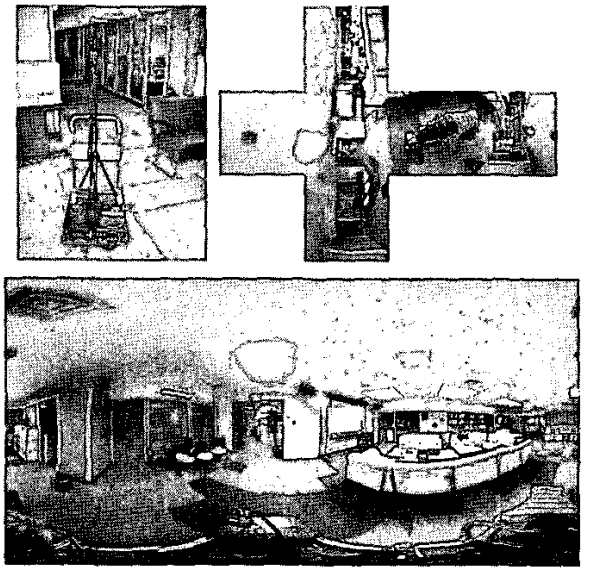

Fig. 1. (Top Left) Multi-camera panoramic head and recording device loaded onto a trolley. (Top Right) Cubic environment map. (Bottom) Equirectangular environment map.

3. ACQUIRING A PLENOPTIC PATH

3.1. Multi-head camera

We use our inhouse multi-head camera to capture full spherical videos. The panoramic head consists of 10 CCD NTSC block cameras sharing roughly a same optical center (i.e., nodal point) whose overlapping field of views capture the full view (4π stera-

Thanks to Sony Corp. FourthVIEW team members for their collaborations. http://www.sony.co.jp/Products/fourthview/

[1]Throughout this paper, we use the most pertinant citation from which other seminal or related papers can be found.

Authorized licensed use limited to: Ecole Polytechnique. Downloaded on October 12, 2009 at 10:15 from IEEE Xplore. Restrictions apply.

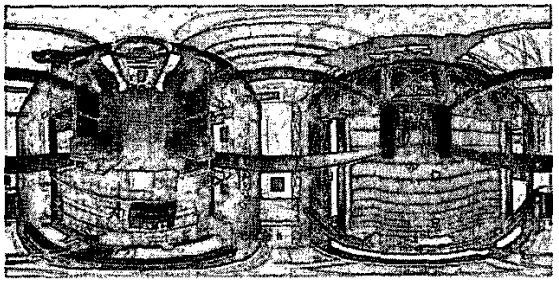

Fig. 2. Surround image computed from a CG script.

dian angle) at 60 interlaced frames per second. We put the camera and recording system onto a trolley and power them with batteries to freely capture surrounding environments. (The trolley is pushed manually by hands as shown in Figure 1.) Algorithms for stitching, viewing and coding the environment maps are explained in details in [7]. We record of the order of $p \propto 10000$ panoramic frames $I_i, i \in \{1, ..., p\}$ with high resolution[2] (i.e., say 2048x1024 image size for equirectangular panoramas).

3.2. Computer graphics scripts

We can also compute full view images and plenoptic paths using ray tracing or radiosity softwares. We chose Povray[3] to compute surround images since its source code is disclosed[4] and some non-trivial CG scripts[5] are available (see Figure 2). Working with CG images is useful for benchmarking since it has the advantage of having a perfect virtual surround camera (ie., no physical obstructions by devices, and no one needed to push the camera on their knees!) and correct imageries (constant lighting, no parallax, no noise, etc.).

4. EGOMOTION RECOVERY

We describe our simple algorithm for recovering the camera extrinsic positions. Since panoramic images have no intrinsic parameters, we end up with a Euclidean path (defined up to a scaling factor) of spatially indexed panoramic images. GPS has been used to roughly annotate the video of outdoor large-scale panoramic paths. Unfortunately, this ends up in too coarse positions that can not be used for view synthesis [8, 9]. Vision algorithms have recently proven successful in the match moving industry, where virtual and real camera paths have to be matched for visual effect productions. Based on feature trackings[6] the position tags $(x, y, \theta)_i, i \in \{1, ..., p\}$ of omnidirectional images are calculated. Below, we sketch yet an easy and fast global egomotion algorithm for our constrained case of a panoramic camera moving on a fixed height plane:

- Compute rough rotation sequence θ_i (pixel-based method).
- Compute rough translation sequence $(x, y)_i$ (feature-based method).
- Refine $(x, y, \theta)_i$ by performing a dense global optimization based on the initial estimations (feature-based method taking properly into account all parameters at once).

Since we do not need to indicate landmarks nor initialize beforehand the path, it makes the acquisition system convenient to use.

4.1. Coarse orientations

Absolute orientation indicates the "North" of the panorama images. Although simple the algorithm performs well without feature matchings. Say our images are in the equirectangular format of size $w \times h$, then for each panoramic frame I_i we average each column pixels to get corresponding 1d ring images R_i. We then register consecutively the ring images R_i, R_{i+1} (eventually with subpixel accuracy [10]) to get the orientation shift. (A pixel unit in the ring image corresponding to a shift of $\frac{2\pi}{w}$ radian.) Averaging the columns can be done within a restricted latitude range by weighting the pixels according to their corresponding vertical subtended angle lengths.

4.2. Coarse translations

Once the images are approximately oriented, we need to find the sequence $(x, y)_i$. As most structure from motion algorithms do, we also reconstruct a set of Euclidean 3d points. We compute 1d translations by chunks as follows: First, partition the path into segments (sequence of consecutive images) where orientations does not change much. This defines a polyline where segment lengths λ_i remained to be calculated (see Figure 5). Given the extremity images I_d and I_{d+k} of a chunk of length k, we compute the translation parameter λ_d from commonly tracked features[7] of $I_d, ..., I_{d+k}$ (see Figure 3) using standard numerical analysis. The sequence $(x, y)_i$ is then obtained from $(\theta, \lambda)_i$ by doing polar to cartesian conversions.

4.3. Parameter refinements

We improve numerically the sequence $(x, y, \theta)_i$ by correlating properly the rotations and translations in a manner similar to methods described in [11, 12]. Although crucial for view synthesis applications, here we roughly analyze combinatorial events happening along the path so that this last step does not improve dramatically the filtering process.

5. FILTERING SPATIALLY THE PLENOPTIC PATH

Once a plenoptic path has been acquired, we would like to sample/annotate it appropriately so that "redundant" images (images bringing not that much new information) are removed. (Sampling is also useful for progressive coding of the plenoptic path, etc.) Chai et al. [13] investigated the sampling of $L(\cdot)$ using spectral analysis. Our study is tailored for plenoptic paths (a subset of plenoptic functions) where geometric decomposition is used.

[2] See [7] for environment representations and equivalent sizes.

[3] See `http://www.povray.org`

[4] Namely, we modify the procedure `create_ray` of the file `render.cpp` in order to output our image format by defining a bijective function that maps to each pixel (x, y) of the CG image its corresponding (θ, ϕ) angular coordinates.

[5] Some scripts can be found at `http://www.irtc.org`

[6] This approach is also mentioned in [6, 12, 11].

[7] We use the Kanade-Lucas-Tomasi library for this purpose. `http://vision.stanford.edu/~birch/klt/`

Authorized licensed use limited to: Ecole Polytechnique. Downloaded on October 12, 2009 at 10:15 from IEEE Xplore. Restrictions apply.

Fig. 3. Features tracked from the plenoptic path sequence.

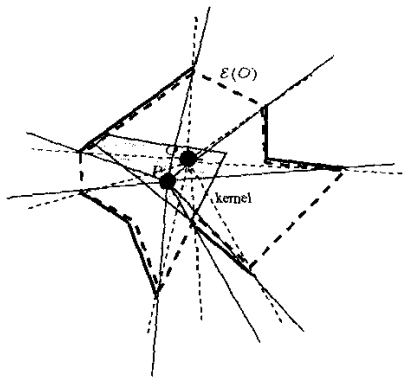

Fig. 4. Envelope $\mathcal{E}(O)$ where O denotes the center. The thick black lines are polylines describing the scene. The dashed star-shaped polygon is the resulting envelope $\mathcal{E}(O)$ from O.

One approach to reduce the number of surround images is to select/distribute viewpoints P_i uniformly along the path $\mathcal{P}$ proportionally to the curve length (path length parameterization). This is efficient if no geometric information is available. Below, we introduce a geometric analysis of combinatorial events arising along $\mathcal{P}$. Let $\mathcal{F}$ denotes the *free* space, that is the space not occluded by any of the n objects of the scene $\mathcal{S} = \{S_1, ..., S_n\}$. $\mathcal{F} = \mathbb{E}^3 \setminus \cup_{i=1}^{n} S_i$. Let $\mathcal{V} \subseteq \mathcal{F}$ be the *viewable* space, that is the portion of free space where the user is allowed to move interactively. Given a position $P \in \mathcal{V}$, let $\mathcal{E}(P)$ denotes the lower envelope surrounding P (see [14] for a glossary of geometric terms). That is $\mathcal{E}(P)$ is defined as the radial function $r(\theta, \phi)$ that for a given (θ, ϕ) angular coordinates, returns the distance to the first hit object of $\mathcal{S}$, if it exists, of a ray emanating from P with direction (θ, ϕ) (see Figure 4). Note that $\mathcal{E}(\cdot)$ is not necessarily continuous. Moving P slightly changes $\mathcal{E}(P)$ smoothly unless it reaches a critical combinatorial event defined by a visibility event (occlusion/disocclusion). Let $\mathcal{A}(\mathcal{S})$ be the partition of $\mathcal{V}$ into elementary visibility cells (see Figure 5). If $\mathcal{S}$ consists of n triangles of $\mathbb{E}^3$ then $\mathcal{A}(\mathcal{S})$ has complexity $O(n^9)$. However, if we restrict to 2d cells cut by a straight line the complexity falls to $O(n^3)$ and becomes tractable (the combinatorial size of all envelopes cut by a straight line being $O(n^3)$, see the Zone theorem [14]). Note that if we constrain the visiblity cells to have a minimum "width" then the complexity becomes linear (i.e., proportional to the path length). For indoor shooting, we consider a floor map (often available in drawing exchange format - DXF format) of the building to get[8] the visibility graph. Note that for n small, we can apply a naive quadratic algorithm to compute its restriction to a polyline. For a given visibility cell, say of volume v and plenoptic path length intersecting it of length l, we choose to sample the path inside this cell according the ratio $\frac{l^3}{v}$. At portions where the visibility information change much (i.e., going from one large visibility cell to another), we increase locally furthermore the sampling rate to provide smoother transitions. The database of images can thus be reduced by 1 or 2 orders of magnitude while keeping its semantics. For complex CG scripts having non-trivial visibility graphs (since n is very large!), we can nevertheless estimate roughly the current viewpoint cell volume v as follows: each pixel $e_{i,j}$ of a RGBZ environment image (color and depth information $d_{i,j}$ stored at each pixel) subtends a corresponding solid angle $a_{i,j}$. Since the solid angles partition the full unit sphere, we simply add their contribution to get $v = \frac{1}{4\pi} \sum_{i,j} a_{i,j} d_{i,j}$. Let us make the assumption that if the volume change *significantly*, this means that we moved from one visibility cell to another. Then we can first probe for combinatorial events along a given plenoptic path and then sample it accordingly (that is render $\mathcal{P}$). The sampling for CG scripts can be done online as we incrementally planify for the next best viewpoint along the path $\mathcal{P}$.

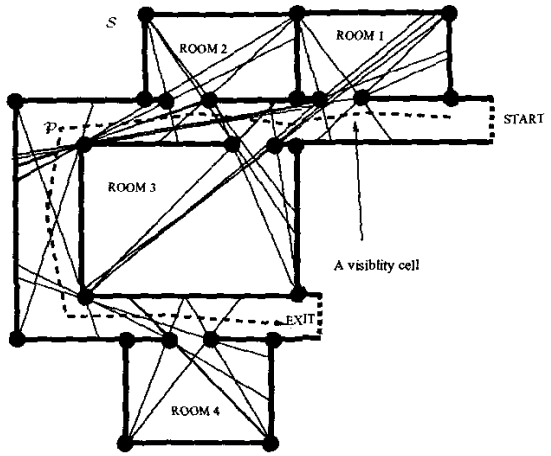

Fig. 5. Schematic illustration of a visibility graph and its cell decomposition.

[8] A 2d visibility graph package is available in CGAL. `http://www.cgal.org`

Authorized licensed use limited to: Ecole Polytechnique. Downloaded on October 12, 2009 at 10:15 from IEEE Xplore. Restrictions apply.

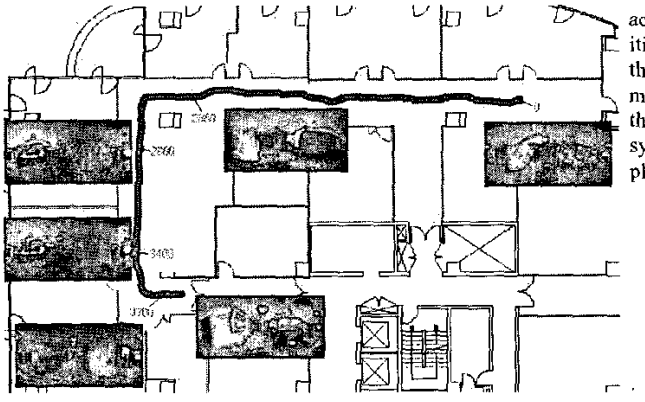

Fig. 6. A plenoptic path and some of its key frames.

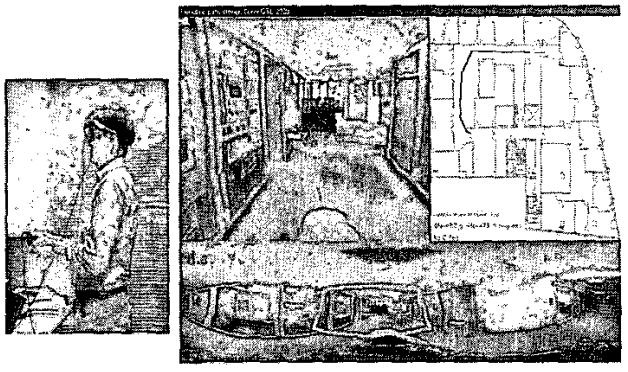

Fig. 7. Interactively walking along the plenoptic path.

6. APPLICATIONS AND RESULTS

We provide experimental results on acquired/synthesized sequences. The implementation has been done in C++ using OpenGL® on a Windows® /Intel® commodity PC. The user can either move interactively, at 60fps refresh rate, on the plenoptic path using the mouse in the viewing/map window (see Figure 7), or by using a head-mounted display (HMD) equipped with a gyroscope. (The tilting of the HMD indicates whether we move forward or backward.) The walkthrough experience can be enriched using multimedia add-ons, such as movie textures (that are calibrated and super-imposed using homographies) or event-triggered[9] image-based operations.

7. CONCLUSION

We presented a framework to acquire or synthesize plenoptic paths whose samplings are computed according to visibility events. We implemented an image-based rendering browser allowing to interactively walk along those plenoptic paths. Ongoing research activities concentrate on further geometric modeling and sampling of the plenoptic path (including non-lambertian reflections, etc.), removing of the moving parts (mainly people moving while sensing the environment) to get clean "static images" using optical flow, system scalability and its automation and, last but not least, its applications in games and telepresence.

[9]For example, we can choose to push the lift button and the lift doors open or close accordingly, etc.

8. REFERENCES

[1] S. E. Chen, "Quicktime VR - An Image-Based Approach to Virtual Environment Navigation," *ACM SIGGRAPH*, pp. 29-38, 1995.

[2] M. Levoy and P. Hanrahan. "Light field rendering," *ACM SIGGRAPH*, pp. 31–42, 1996.

[3] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen, "The lumigraph," *ACM SIGGRAPH*, pp. 43–54, 1996.

[4] W.-C. Chen, J.-Y. Bouguet, M. H. Chu, R. Grzeszczuk, "Light Field Mapping: Efficient Representation and Hardware Rendering of Surface Light Fields," *ACM Transactions on Graphics*. 21 (3), pp. 447-456, 2002.

[5] D. G. Aliaga, T. Funkhouser, D. Yanovsky, I. Carlbom, "Sea Of Images," *IEEE Visualization*, pp. 331-338, 2002.

[6] D. G. Aliaga, D. Yanovsky, T. Funkhouser and I. Carlbom, "Interactive Image-Based rendering Using Feature Globalization", *ACM Symp. on 3D Graphics*, 2003.

[7] F. Nielsen, "High Resolution Full Spherical Videos," *IEEE Intl. Conf. on Information Technology: Coding and Computing*, pp. 260-267, 2002.

[8] M. Hirose, "Space Recording Using Augmented Virtuality Technology," *Intl. Mixed Reality Symp.*, pp. 105-110, 2001.

[9] D. Kimber, J. Foote, and S. Lertsithichai, "FlyAbout: Spatially Indexed Panoramic Video," *ACM Multimedia 2001*, pp. 339-341, 2001.

[10] B. D. Lucas and T. Kanade, "An iterative Image Registration Technique with an Application to Stereo Vision," *7th Intl Joint Conference on Artificial Intelligence (IJCAI)*, pp. 676-679, 1981.

[11] C. J. Taylor, "VideoPlus: A Method for Capturing the Structure and Appearance of Immersive Environments", *IEEE Trans. on Visualization and Computer Graphics*, Vol. 8(2), pp. 171-182, 2002.

[12] M. Antone and S. Teller, "Scalable Extrinsic Calibration of Omni-Directional Image Networks," *Intl. Journal of Computer Vision*, Vol. 49(2/3), pp. 143-174, 2002.

[13] J.-X. Chai, S.-C. Chan, H.-Y. Shum, X. T. "Plenoptic sampling," *ACM SIGGRAPH*, pp. 307-318, 2000.

[14] J.-D. Boissonnat and M. Yvinec, "Algorithmic geometry," Cambridge University Press, 1998.

Authorized licensed use limited to: Ecole Polytechnique. Downloaded on October 12, 2009 at 10:15 from IEEE Xplore. Restrictions apply.