REAL TIME CAPTURE OF AUDIO IMAGES AND THEIR USE WITH VIDEO

Adam O'Donovan, Ramani Duraiswami and Nail A.Gumerov

Perceptual Interfaces and Reality Laboratory, Computer Science & UMIACS
University of Maryland, College Park, MD 20742
adamod@gmail.com, {ramani,gumerov}@umiacs.umd.edu

ABSTRACT

Spherical microphone arrays provide an ability to compute the acoustical intensity corresponding to different spatial directions in a given frame of audio-data. These intensities may be exhibited as an image and these images updated at a high frame rate to achieve a video stream if the data capture and intensity computations can be performed sufficiently quickly, there by creating a frame-rate audio camera. We describe how such a camera can be built and the processing done sufficiently quickly using graphics processors. The joint processing of captured frame-rate audio and video images enables applications such as visual identification of noise sources, beamforming and noise-suppression in video conferenceing and others, provided it is possible to account for the spatial differences in the location of the audio and the video cameras. Based on the recognition that the spherical array can be viewed as a central projection camera it is possible to perform such joint analysis. We provide several examples of real-time applications.

1. INTRODUCTION

Over the past few years there have been several publications that deal with the use of spherical microphone arrays (see e.g. [9, 1, 16, 4, 15]). Such arrays are seen by some researchers as a means to capture a representation of the sound field in the vicinity of the array [2], and by others as a means to digitally beamform sound from different directions using the array with a relatively high order beampattern [10, 6], or for nearby sources [7]. Variations to the usual solid spherical arrays have been suggested, including hemispherical arrays [5], open arrays [1], concentric arrays and others. We refer the interested reader to these papers for discussions.

A particularly exciting use of these arrays is to steer it to various directions and create an intensity map of the acoustic power in various frequency bands via beamforming. The resulting image, since it is linked with direction can be used to identify source location (direction), be related with physical objects in the world and identify sources of sound and be used in several applications that we discuss at the end of the paper. This brings up the exciting possibility of creating a "sound camera."

To be useful, two difficulties must be overcome. The first, is that the beamforming requires the weighted sum of the Fourier coefficients of all the microphone signals, and multichannel sound capture, and it has been difficult to achieve frame-rate performance, as would be desirable in applications such as videoconferencing, noise detection, etc. Second, while qualitative identification of sound sources with real-world objects (speaking humans, noisy machines, gunshots) can be done via a human observer who has knowledge of the environment geometry, for precision and automation the sound images must be captured in conjunction with video, and the two must be automatically analyzed to determine correspondence and identification of the sound sources. For this a formulation for the geometrically correct warping of the two images, taken from an array and cameras at different locations is necessary.

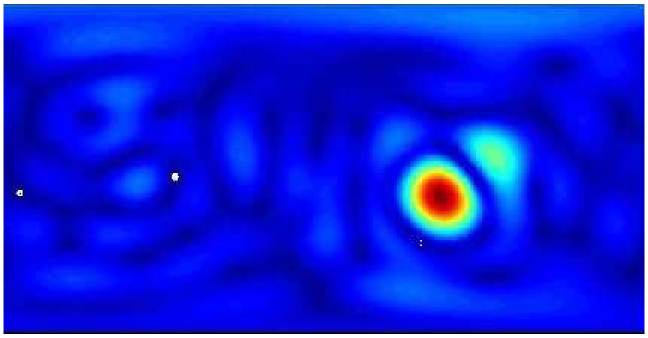

Figure 1: A sound image created by beamforming along a set of 8192 directions (a 128×64grid in azimuth and elevation), and quantizing the steered response power according to a color map.

By recognizing that the spherical array derived sound images satisfy central projection, a property crucial to geometric analysis of multi-camera systems [3], we showed [11] how it was possible to calibrate camera-spherical array systems, and perform vision-guided beamforming. Here, we extend that system to achieve frame-rate sound image creation, beamforming, and the processing of the sound image stream along with a simultaneously acquired video-camera image stream, to achieve "image-transfer," i.e., the ability to warp one image onto the other to determine correspondence. The key innovation that enables speed is to use modern graphics processors (GPUs) to do the processing at frame-rate. In Sec. 2 we provide some background and notation for both spherical arrays and GPUs. In Sec. 3 we briefly describe our experimental setup. In Sec. 4 we provide details that allow us to achieve high frame-rates. Sec. 5 provides experimental results.

2. BACKGROUND

Beamforming with Spherical Microphone Arrays: Let sound be captured at N microphones at locations $\mathbf{\Theta}_s = (\theta_s, \varphi_s)$ on the surface of a solid spherical array. Two approaches to the beamforming weights are possible. The modal approach relies on orthogonality of the spherical harmonics and quadrature on the sphere, and decomposes the frequency dependence. It however requires knowledge of quadrature weights, and theoretically for a quadrature order P (whose square is related to the number of microphones S) can only achieve beampatterns of order $P/2$ [9, 16]. The other requires the solution of interpolation problems of size S (potentially at each frequency), and building of a table of weights [6]. In each case, to beamform the signal in direction $\mathbf{\Theta} = (\theta, \varphi)$ at frequency f (corresponding to wavenumber $k = 2\pi f/c$, where c is the sound speed), we sum up the Fourier transform of the pressure at the different microphones, d_s^k as

$$\psi\left(\Theta; k\right) = \sum_{s=1}^{S} w_N(\mathbf{\Theta}, \mathbf{\Theta}_s, ka) d_s^k\left(\mathbf{\Theta}_s\right). \qquad (1)$$

In the modal case [9] the weights w_N are related to the quadrature weights C_n^m for the locations $\{\mathbf{\Theta}_s\}$, and the b_n coefficients obtained from the scattering solution of a plane wave off a solid sphere

$$w_N(\mathbf{\Theta}, \mathbf{\Theta}_s, ka) = \sum_{n=0}^{N} \frac{1}{2i^n b_n(ka)} \sum_{m=-n}^{n} Y_n^{m*}(\mathbf{\Theta}) Y_n^m(\mathbf{\Theta}_s) C_n^m(\mathbf{\Theta}_s). \qquad (2)$$

For the placement of microphones at special quadrature points, a set of unity quadrature weights C_n^m are achieved. In practice, it was observed [6] that for $\{\mathbf{\Theta}_s\}$ at the the so-called Fliege points, higher order beampatterns were achieved with some noise (approaching that achievable by interpolation $(N+1) = \sqrt{S}$). In our beamformer, we use one order lower than this limit, and the Fliege microphone locations, though we also consider the case where weights are generated separately and stored in a table.

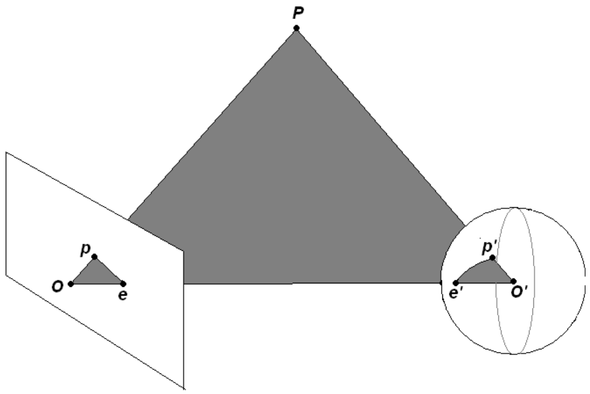

Figure 2: Epipolar geometry between a video camera (left), and a spherical array sound camera. The world point P and its image point p on the left are connected via a line passing through PO.Thus, in the right image the corresponding image point p' lies on a curve which is the image of this line (and vice versa, for image points in the right camera).

Joint Audio-Video processing and Calibration: In [11] we provide a detailed outline of how to use cameras and spherical arrays together and determine the geometric location of a source. The key observation was that the intensity image at different frequencies created via beamforming using a spherical array could be treated as a central projection (CP) camera, since the intensity at each "pixel" is associated with a ray (or its spherical harmonic reconstruction to a certain order). When two CP cameras observe a scene, they share an "epipolar geometry" (Fig. 2, also see [3]). Given two cameras and several correspondences (via a calibration object such as in Fig. 3), a fundamental matrix that encodes the calibration parameters of the camera and the parameters of the relative transformation (rotation and translation) between the two camera frames can be computed. Given a fundamental matrix of a stereo rig it is possible to take points in one cameras coordinate system and relate them to directly to pixels in the second cameras coordinate system. Given more video cameras, a complete solution of the 3D scene structure common to the two cameras can be made, and "image transfer" that allows the transfer of the audio intensity information to actual scene objects made precisely. Given a single camera and a microphone array, the transfer can be accomplished if we assume that the world is planar (or that it is on the surface of a sphere) at a certain range.

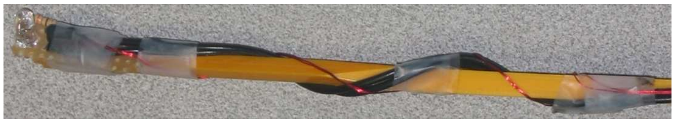

Figure 3: A calibration wand consisting of a Knowles microspeaker and an LED collocated at the end of a pencil was used to obtain the fundamental matrix.

General Purpose GPU Processing: Recently GPUs have become an incredibly powerful computing workhorse for processing computationally intensive highly parallel tasks. Recently NVidia released the Compute Unified Device Architecture (CUDA) along with the G8800 GPU with a theoretical peak speed of 330 Gflops, which is over two orders of magnitude larger than that of a state of the art Intel processor. This release provides a C-like API for coding the individual processors on the GPU that makes general purpose GPU programming much more accessible. CUDA programming, however still requires much trial and error, and understanding of the nonuniform memory architecture to map a problem onto it. In this paper we map the beamforming, image creation, image transfer, and beamformed signal computation problems to the GPU to achieve a frame-rate audio-video camera.

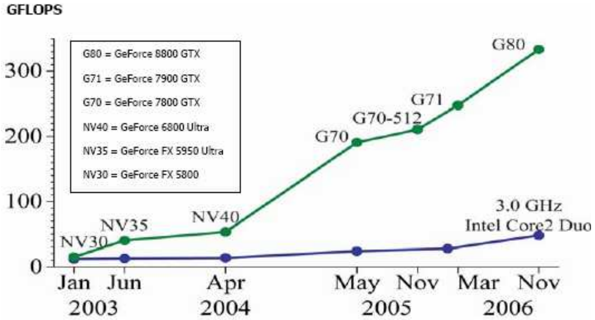

Figure 4: Peak GFlops for NVIDIA GPUs vs Intel CPUs. GPU capabilities have been advancing much faster.

3. EXPERIMENTAL SETUP

Audio information was acquired using a previously developed [6] solid spherical microphone array of radius 10cm whose surface was embedded with 60 microphones. The signals from the microphones are amplified and filtered using two custom 32 channel preamplifiers and fed to two National Instruments PCIe-6259 multi function data acquisition cards. Each audio stream is sampled at a rate of 31250 samples per second. The acquired audio is then shipped to an NVidia G8800 GTX GPU installed in a computer running Windows XP with an Intel Core2 processor and a clock speed of 2.4GHz with 2GB of Ram. The NVidia G8800 GTX GPU utilizes a 16 SIMD multiprocessors with On-Chip Shared memory. Each of these Multiprocessors is composed of 8 separate processors that operate at 1.35GHz for a total of 128 parallel processors. The G8800 is also equipped with 768 MB of onboard memory. In addition to audio acquisition we also acquire video frames from an orange micro IBot USB2.0 web camera at a resolution of 640 x 480 pixels and a frame rate of 10 frames per second. The images are acquired using OpenCV and are also immediately shipped to the onboard memory of the GPU.

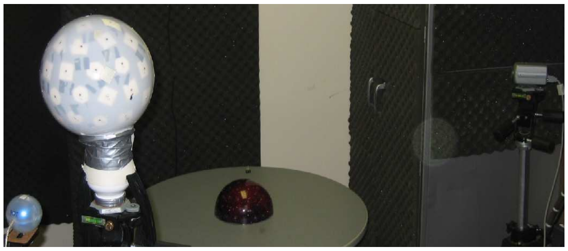

Figure 5: A 2- camera, 2-spherical array system consisting of a webcam and spherical array on the left, a hemispherical array on the centre table, and a video camera on the right. This paper reports results with the single array and camera on the left.

4. REAL-TIME PROCESSING

Since both pre-computed weights [5] and analytically prescribed weights [9] capable of being generated “on-the-fly” are used, we present the generation of images for both cases.

Pre-computed weights: This algorithm proceeds in a two stage fashion: a precomputation phase (run on the CPU) and a run-time GPU component. In stage 1 pixel locations are defined prior to run-time and the weights are computed using any optimization method as described in the literature. These weights are stored on disk and loaded only at Runtime. In general the number of weights that must be computed for a given audio image is equal to PMF where P is the number of audio pixels, M is the number of microphones, and F is the number of frequencies to analyze. Each of these weights is a complex number of size 8 bytes.

After pre-computation and storage of the beamformer weights in the run-time component the weights are read from disk and shipped to the onboard memory of the GPU. A circular buffer of size 2048 x 64 is allocated in the CPU memory to temporarily store the incoming audio in a double buffering configuration. Every time 1024 samples are written to this buffer they are immediately shipped to a pre-allocated buffer on the GPU. While the GPU processes this frame the second half of the buffer is populated. This means that in order to process all of the data in real-time all of the processing must be completed in less then 33ms, to not miss any data.

Once audio data is on the GPU we begin by performing an in place FFT using the cuFFT library in the NVidia CUDA SDK. A matrix vector product is then performed with each frequency’s weight matrix and the corresponding row in the FFT data, using the NVidia CuBlas linear algebra library. The output image is segmented into 16 sub-images for each multi-processor to handle. Each multiprocessor is responsible for compiling the beamformed response power in three frequency bands into the RGB channels of the final pixel buffer object. Once this is completed control is restored to the CPU and the final image is displayed to the screen as a texture mapped quad in OpenGL.

On the fly weight computation: In this implementation there is a much smaller memory footprint. Whereas we needed space to be allocated for weights on the GPU in the previous algorithm, this one only needs to store the location of the microphones. At start up these locations are read from disk and shipped to the GPU memory. Efficient processing is achieved by making use of the addition theorem which states that

$$P_n\left(\cos\gamma\right) = \frac{4\pi}{2n+1}\sum_{m=-n}^{n} Y_n^{-m}\left(\mathbf{\Theta}\right) Y_n^{m}\left(\mathbf{\Theta}_s\right) \qquad (3)$$

where Θ is the spherical coordinate of the audio pixel and Θ_s is the location of the sth microphone, γ is the angle between these two locations and P_n is the Legendre polynomial of order n. This observation reduces the order n^2 sum in Eq. (2) to an order n sum. The P_n are defined by a simple recursive formula that is quickly computed on the GPU for each audio pixel.

The computation of the audio proceeds as follows. First we load the audio signal onto the GPU and perform an inplace FFT. We then segment the audio image into 16 tiles and assign each tile to a multiprocessor of the GPU. Each thread in the execution is responsible for computing the response power of a single pixel in the audio image. The only data that the kernel needs to access is the location of the microphone in order to compute γ and the Fourier coefficients of the 60 microphone signals for all frequencies to be displayed. The weights can then be computed using simple recursive formula for each of the Hankel, Bessel, and Legendre polynomials in Eq. (2).

While performance of the beamformer may be a bit worse, there are several benefits to the on-the-fly approach: 1) frequencies of interest can be changed at runtime with no additional overhead; 2) pixel locations can be changed at runtime with little additional overhead; 3) memory requirements are drastically lower then storing pre-computed weights.

Beamforming: Once a source location of interest is identified, we can isolated audio signal associated with that direction. The intensity of the audio pixel for a given direction is proportional to the Fourier component of the corresponding frequency in the acoustic signal. By computing the intensity of the audio pixels for a given location for all frequencies in the microphone array effective frequency band we can recover and isolate signal. For frequencies outside the effective range of the array we simply append the Fourier coefficients of the raw audio signal from the closest microphone.

5. RESULTS

Vision guided beamforming: Several authors have in the past proposed vision guided beamforming (see e.g., [12]). The idea is that vision based constraints can help us to not steer the beamformer in directions that are not promising. Often these constraints require the source to lie in some constrained region. One crucial difference here is that the quality of the geometric constraints provided by the epipolar geometry is much stronger. We illustrate in

Fig. 6 this example with a case where a speaker's voice is beamformed in the presence of severe noise using location information from vision. Using a calibrated array-camera combination, we applied a standard face detection algorithm to the vision image and then used the epipolar line induced by the mouth region of the vision image to search for the source in the audio image.

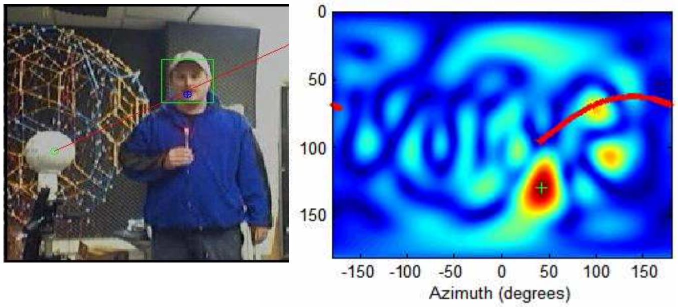

Figure 6: A loudspeaker source was played that overwhelmed the sound of the speaking person, whose face was detected with a face detector [8] and the epipolar line corresponding to the mouth location in the vision image was drawn in the audio image. A search for a local audio intensity peak along this line in the audio image allowed precise steering of the beam, and made the speaker audible. www.umiacs.umd.edu/~ramani/pubs/face_beamform.wmv

Image transfer: Noise source identification via acoustic holography seeks to determine the noise location from remote measurements of the acoustic field. Here we add the capacity to visually identify the source via automatic warping of the sound image. This implementation also has application to areas such as gunshot detection, meeting recording (identifying who's talking), etc. We used the method of precomputed weights. An audio image was generated at a rate of 30 frames/s and video was acquired at a rate of 10 frames/s. In order to reduce the effects of incoherent reverberation and spurious peaks we incorporated a temporal filter of the audio image prior to transfer. Once the audio image is generated a second GPU kernel is assigned to generate the image transfer overlay which is then alpha blended with the video frame. The audio video stereo rig was calibrated according to [11]. The audio image transfer is also performed in parallel on the GPU and the corresponding values are then mapped to a texture and displayed over the video frame. To decrease pixilation artifacts the kernel also performs bilinear interpolation. Though the video frames are only acquired at 10 frames per second the over-laid audio image achieves the same frame rate as the audio camera (30fps). Figure 7 shows an image of the transfer of the sound image onto the video image of a speaking user.

Acknowledgments: We thank Dr. David Luebke of NVIDIA for the donation of the CUDA board and Prof. Amitabh Varshney, UMD for helping arrange the CPU GPU system via NSF Award CISE-RI0403313.

6. REFERENCES

[1] T. Abhayapala & D. Ward, 2002. Theory and Design of High Order Sound Field Microphones Using Spherical Microphone array, IEEE ICASSP 2002, vol. 2, pp. 1949-1952.

[2] R. Duraiswami, D.N. Zotkin, Z. Li, E. Grassi, N.A. Gumerov, L.S. Davis, System for capturing of high-order spatial audio using spherical microphone array and binaural head-tracked playback over headphones with head related transfer function cues, Proc. 119th convention AES, 2005.

[3] R. Hartley and A. Zisserman. Multiple View Geometry in Computer Vision. Cambridge University Press, 2000.

[4] Z. Li, R. Duraiswami, E. Grassi and L.S. Davis, Flexible layout and optimal cancellation of the orthonormality error for spherical microphone arrays, ICASSP2004, IV:41-44, 2004.

[5] Z. Li and R. Duraiswami. "Hemispherical microphone arrays for sound capture and beamforming," Proceedings IEEE WASPAA, 106–109, 2005.

[6] Z. Li and R. Duraiswami. "Flexible and Optimal Design of Spherical Microphone Arrays for Beamforming," IEEE Transactions on Audio, Speech and Language Processing, 15:702-714, 2007

[7] Z. Li, The Capture and Recreation of 3D Auditory Scenes, Ph.D. Thesis, Department of Computer Science, University of Maryland, 2005.

[8] R. Lienhart, L. Liang, and A. Kuranov. "A detector tree of boosted classifiers for real-time object detection and tracking," Proceedings IEEE ICME, 2:277–280, 2003.

[9] J. Meyer & G. Elko, 2002. A Highly Scalable Spherical Microphone Array Based on an Orthonormal Decomposition of the Soundfield, IEEE ICASSP 2002, vol. 2, pp. 1781-1784.

[10] J. Meyer & G. Elko, "Spherical Microphone Arrays for 3D Sound Recording", in Audio Signal Processing, ed. by Y. Huang and J. Benesty, Kluwer Academic Publishers, 2004

[11] A. O'Donovan, R. Duraiswami, and J. Neumann. Microphone Arrays as Generalized Cameras for Integrated Audio Visual Processing. To appear: Proc. IEEE CVPR, 2007.

[12] C. Choi, D. Kong, S. Lee, K. Park, S. Hong, H. Lee, S. Bang; Y. Lee, S. Kim, "Real-time audio-visual localization of user using microphone array and vision camera," Proc. IEEE/RSJ IROS, 1935- 1940, 2005.

[13] S. Whalen, Audio and the Graphics Processing Unit. IEEE tutorial GPGPU, Vis 2004.

[14] E Gallo and N. Tsingos, Efficient 3D Audio Processing with the GPU. http://www-sop.inria.fr/reves/Nicolas.Tsingos/publis/posterfinal.pdf

[15] B. Rafaely, 2005. Analysis and Design of Spherical Microphone Arrays, IEEE Trans. Speech Audio Proc., vol. 13(1), pp. 135-143.

[16] B. Rafaely, 2004. Plane-Wave Decomposition of the Sound Field on a Sphere by Spherical Convolution, J. Acoust. Soc. Am., vol. 116(4), pp. 2149-2157.

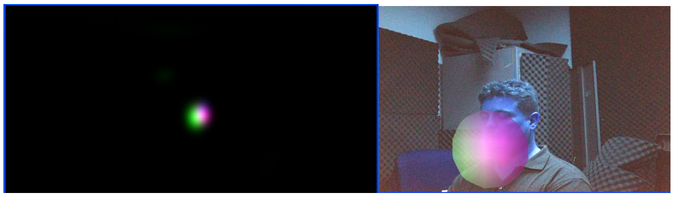

Figure 7: Image transfer example: A person speaks. The spherical array image (left) shows a bright spot at the location corresponding to the mouth. This spot is automatically transferred to the video image (where the spot is much bigger, since the pixel resolution of video is higher), identifying the noise location as the mouth. For a video see www.umiacs.umd.edu/~ramani/audio_imtransf.wmv