Automatic Cameraman

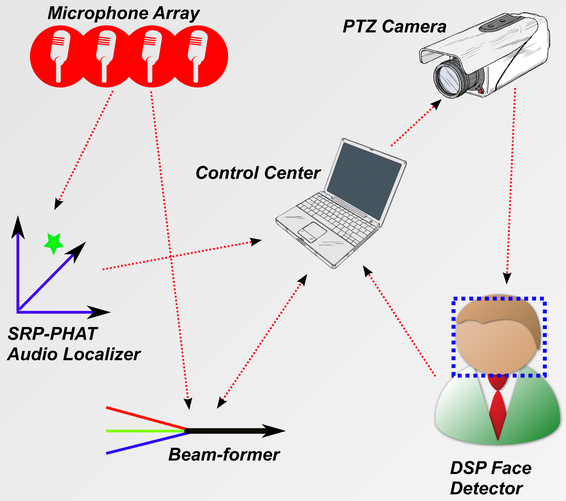
Yoav Freund*, Evan Ettinger*, Brian McFee*, Shankar Shivappa+
*Computer Science and Engineering, +Electrical and Computer Engineering
University of California at San Diego

Abstract

An automatic camera pointing system is presented. Using seven microphones and a single PTZ-camera placed in an ad-hoc fashion, we are able to locate and track a human speaker. We utilize an audio localization technique to first direct where the camera should point, and then make fine adjustments to this pointing with a face detector running on a DSP-board. We also apply beam-forming to the audio and can record video using the system. Applications include video conferencing, automatic lecture recording, and is applicable in many security domains.

The audio localization is based upon a generalized correlation technique (Steered Response Power PHAse Transform, SRP-PHAT). Here we utilize knowledge of the microphone and camera positions to explore the target space for potential localized audio sources. This technique relies on estimating the delays between microphones, which occurs when a localized sound source is emitted. Each microphone records a delayed version of the audio, and using a correlation technique we can leverage these delays into a localization algorithm. The face detector, is a standard Viola and Jones detector implemented on a TI DaVinci DSP board. We are able to in real-time locate a person and record their audio and video, and moreover switch to new speakers as they appear. Demo of the system is available on request.

Cameraman Architecture



Functional overview of the Automatic Cameraman

The automatic cameraman is composed of four major hardware components.

The first component is a Sony SNC-RZ30N Pan/Tilt/Zoom (PTZ) camera. The PTZ camera can be controlled by HTTP requests over an ethernet interface, and outputs NTSC video over a composite video connector.

The second component is the microphone array, which consists of seven microphones and a MOTU 896HD audio digitizer. One of the microphones is placed on the PTZ camera, and the remaining six are placed ad-hoc throughout the environment. The seven audio signals are sent over a firewire connection to the control system.

The third component is a Texas Instruments TMS320DM6446 DaVinci Digital Media System. This board contains a 600MHz C64X+ digital signal processor (DSP) and 300MHz ARM central processor, and runs the MontaVista embedded Linux distribution. All of the video processing (face detection and light tracking) is performed by custom codecs on the DSP, and results are communicated back to the control system through XML-RPC requests over ethernet.

The final component is an Apple Mac-Book Pro, which operates the audio processing and camera control software. The audio signals from the MOTU are processed by the audio localizer, and results are then sent to the control program. The signals are also merged into a single signal by a beam-former, and the resulting output is made available in the form of an emulated audio device. The control program handles requests from the audio and video systems, makes policy decisions, and issues requests to the PTZ camera.

Calibration

Because we do not assume knowledge of the microphone arrangement, the system must be calibrated.

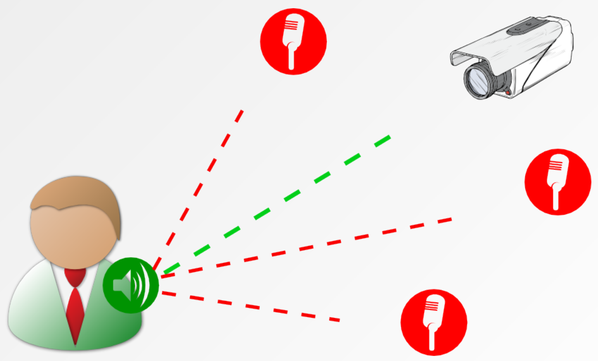
The goal of the calibration procedure is to learn a mapping from the space of microphone time delays to the physical dimensions of the room, and the camera's pan-tilt coordinate system.

In order to learn this mapping, we constructed a hand-held device that emits white noise and a green light which can be tracked by the PTZ camera. The noise emitted by the calibration device is recorded by a wireless microphone, and this signal is used as the ground truth to estimate the distance from the device to each microphone in the array. Meanwhile, the camera is directed to keep the calibration device at the center of the frame, and records the current pan and tilt angles (φ,θ). When the user is fixed at a specific location (x) in the room, this process yields the measurement:

$$m(x)=(d_1, d_2, \ldots, d_7, \varphi, \theta),$$

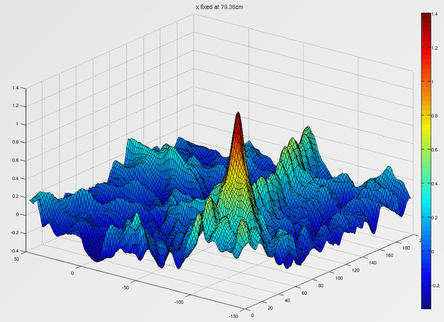
where d_i is the estimated distance from the calibration device to the i^{th} microphone. We record this set of measurements for several points throughout the room, and formulate an optimization problem to solve for the positions of the microphones. The pan/tilt angles are used to orient the learned coordinate system to the camera's natural coordinates. Note that because one of the microphones is fixed to the camera, this also provides an estimate of the distance to the calibration device from the origin of the camera's coordinate system.

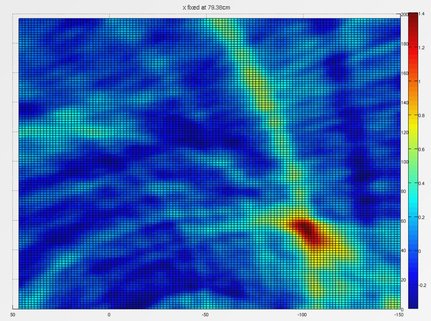
Once we have learned the positions of the microphones, we can then estimate the location of any noise source in the room.



The system is calibrated by taking measurements of audio delays and camera angles throughout the environment.

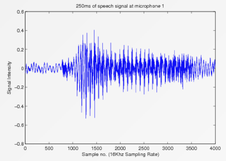
Audio Localization

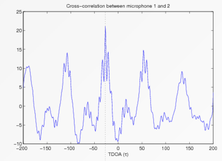


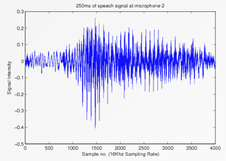


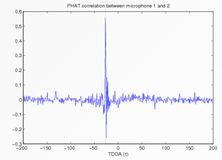
The audio localization works by leveraging both the known microphone positions found in the calibration process along with a method of time delay estimation between pairs of microphones. For each pair of microphones we can calculate the PHAT correlation series (see lower-right figure), which is a generalized cross correlation between the two audio signals. The PHAT correlation has the property that it spikes at the delay at which the two signals match correctly and is near zero at other delays, whereas the regular cross correlation is much noisier. We can efficiently calculate for all pairs of microphones the PHAT correlation series using an FFT.

We then make a 3cm spaced uniform grid over the entire region of interest in front of the camera, and for each grid point x we calculate the expected delays between pairs of microphones for a sound source emitted at x. We create a score for position x by summing the PHAT series value at these delays (see heat map figures to the left). The position with the greatest score is where we believe the sound source to be emitted from, and we alert the controller of this localization event. In practice, the score function is very steeply spiked at the true sound source location. The actual implementation uses a coarse-to-fine griding along with grid contraction steps to speed up the evaluation process. We can localize a 100ms audio frame in under 25ms of CPU time.









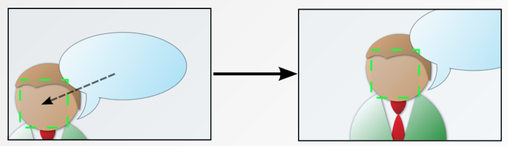
Face Detection

If the source of an audio localization event is a person speaking, we want the cameraman to keep the frame centered on the speaker. To accomplish this, a real-time face detector operates on the video stream from the PTZ camera, and reports back to the controller any time a face is detected in the stream. The controller can then integrate the face detection coordinates with the incoming audio localization events in order to determine the correct pan and tilt settings for the camera. If the detected face does not occupy a large enough portion of the image, the controller gradually zooms in until the detection window is sufficiently large.

The face detector is based on the cascaded detector of Viola & Jones. We use a pre-trained cascade provided by Intel's OpenCV library, but the evaluation engine has been reimplemented as a codec to run on the DaVinci's DSP.

Once a face is detected, the detector enters a tracking mode for the next several frames. When tracking a face, we use a simple template-matching algorithm, rather than the full detection cascade. This allows the cameraman to follow faces, even during brief periods of occlusion or out-of-plane rotation, which may not be successfully detected by the cascade.

Patches of images which are considered faces (either by the detection cascade or the tracker) are saved to disk. Because these saved images include tracking results which are highly likely to be faces, but would be incorrectly classified by the detection cascade, we can use them to train more robust classifiers.

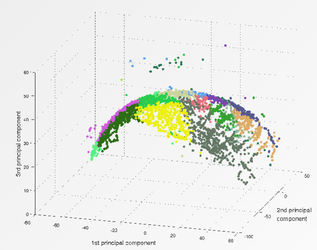


The camera centers the user's face.

Low-Dimensional Manifolds

For an incoming audio signal, measuring the time delay of arrival between each pair of microphones results in a vector in C(7,2) = 21-dimensional space. However, due to physical constraints, there are really only 3 major degrees of freedom, corresponding to the speaker's position within the environment. Therefore, we can expect that the time delay measurements should lie on or near a low-dimensional manifold in the 21-dimensional ambient space. Furthermore, the dimensionality of the manifold should remain low, even if new features are appended to the measurement vector, such as microphone gain levels, echo structure, and filter response power.

Compactly modeling this manifold would allow for several interesting applications, including learning optimal filters for speech reconstruction dependent on the location of the speaker. Random Projection Trees (RPTrees) provide a novel data structure for representing low-dimensional manifolds, which may be incorporated to make accurate location predictions when more features than simple time delays are available.



A 2-dimensional manifold in R^{21}, visualized in R^3.