Voice-controlled Video Console

Ben Horkley and Jonathan Surick

Overview

Our project is a voice-controlled camera system, which automatically tracks a speaker standing in front of it, and which responds to specific voice commands by adjusting both the camera and the video being displayed on-screen.

The camera assembly will contain a video camera with a composite video connection to the lab kit, as well as two microphone assemblies, one on either side of the camera. The entire apparatus will be mounted on a servo, which will be plugged into a lab power supply for ± 5V and controlled via digital signals from the lab kit. The signal levels from the side microphones will be used to determine whether the camera is aimed at the speaker, and control logic will adjust the servo position to keep the camera centered on the sound source. A concept diagram of the final design is shown below.

The system will be trainable to recognize several vocally distinct command words, each of which will either control a different video filter or effect (such as black and white, red tint, or sepia), or will control the camera tracking system (overriding the automatic tracking).

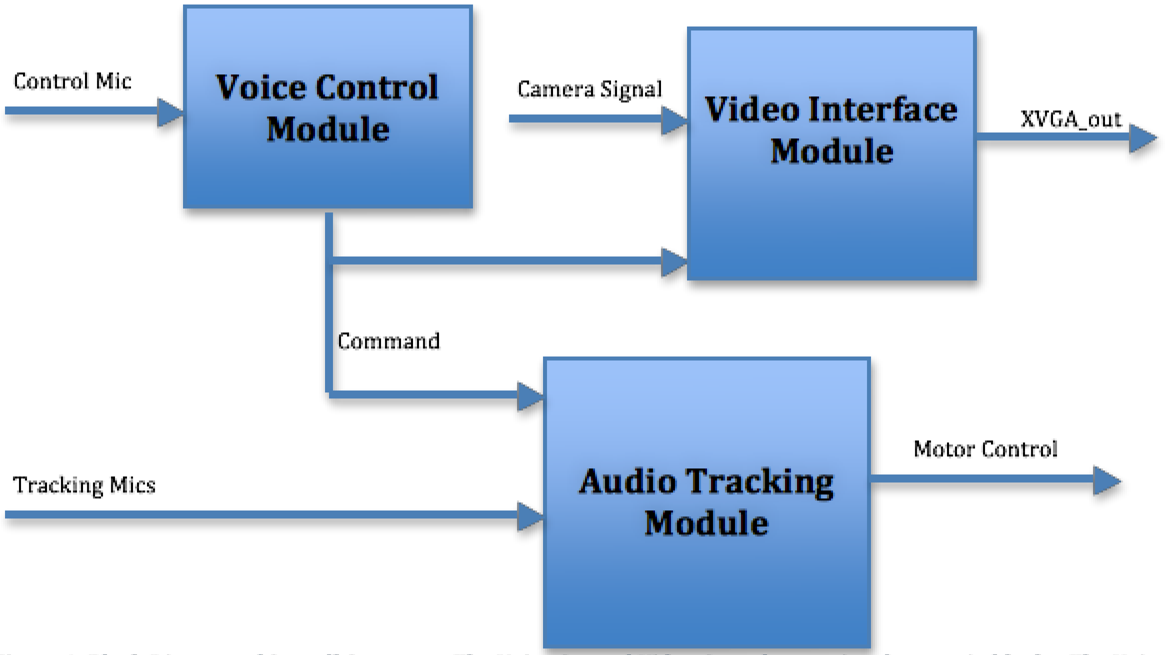

Figure 1: Block Diagram of Overall Structure- The Voice Control Video Console contains three main blocks: The Voice Control Module takes the audio input from the user and determines which of the trained commands has been said. The Audio Tracking Module controls the motion of the camera to either point towards an audio source or to follow voice commands. The Video Interface Module takes the input from the video camera and applies filters before displaying the output on the screen.

The full system is divided into three main distinct modules, as shown in Figure 1: the audio tracking system, the voice control system, and the video filter system. These are described in more detail below, including interfaces for the full modules and the submodules they contain. Each of these main modules will be constructed separately, and is designed to be testable independent of the other two.

Audio Tracking Module

The audio tracking module contains the interfaces for the two microphones mounted on the camera assembly, as well as the servo controlling their movement. It is responsible for sending the control signals necessary to keep the camera centered on the loudest speaking voice in front of the microphone array. The detailed block diagram for this module is seen in Figure 2.

Jonathan will be responsible for this module.

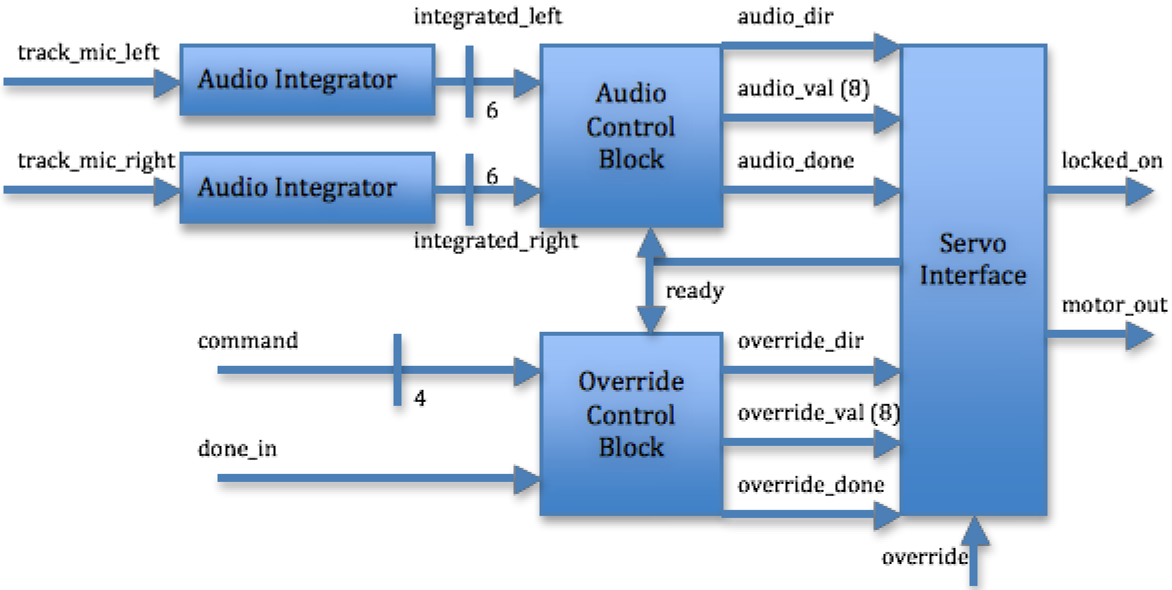

Figure 2:Block Diagram of Audio Tracking Module- This module takes the audio from the two microphones mounted and integrates them to determine the overall volume in from each mic, the audio control block then takes this data and sends signals to the servo to turn the camera towards the louder mic to center the audio source in the camera. The override block takes the voice command generated by the Voice Control Block and passes the instructions on to the Servo Interface. The Interface takes care of turning the signals into PWM so the servo will turn. It also chooses between the override and audio signals based on whether the override signal is on.

Overall I/O

The module will contain two 2-line interfaces with the microphones, controlling the clock signal sent to the microphone chip and reading out serial data from the onboard 12-bit ADC. It also will have a single control line for setting the position of the servo in the camera assembly.

Internal Modules

Each microphone will have a corresponding driver module, which will provide the necessary control signals for reading data out of the built-in ADC, as well as read the incoming audio into a buffer.

The audio integration module will hold an audio data buffer for each microphone. The data in this buffer will be periodically analyzed to determine the loudness levels of the audio data streams which is then passed to the Audio Control Block. The Audio Control Block will take these loudness levels and generate a command word for the servo control module, signifying whether to move the servo more to the left or right or hold its position constant.

Additionally there is an Override Control Block which takes in a command and done signal from the Voice Control Module and also outputs a command word for the servo control module indicating the amount to move the servo and in which direction.

The servo control module will output positive logic pulses to the servo control line, with a rising edge every 20 ms. The width of the pulse will vary between 0.8 and 1.2 ms to control the angular position of the servo. The input to this is chosen based on an override switch, that when shut off takes the output from the audio control block and when active takes the input from the override control block.

Voice Control Module

The voice control module receives incoming audio data from the third microphone, and determines when the speaker is giving one of the trained commands. The voice recognition will operate via push-to-talk; after pressing a button, the speaker will have a one-second period in which their voice is being recorded, after which the audio data will be processed and the command extracted (if one is present).

In order to recognize commands, the model must first be trained with examples of the commands to be used, to be compared while in command mode using dynamic time warping (DTW) on features extracted from the audio data. This training will take place on the FPGA itself. To facilitate this, the module has two major states of operation: training mode and recognition mode. The detailed block diagram of the voice control module can be seen in Figure 3.

The voice control module is the most complex of the three overall sections, and will be worked on by both team members.

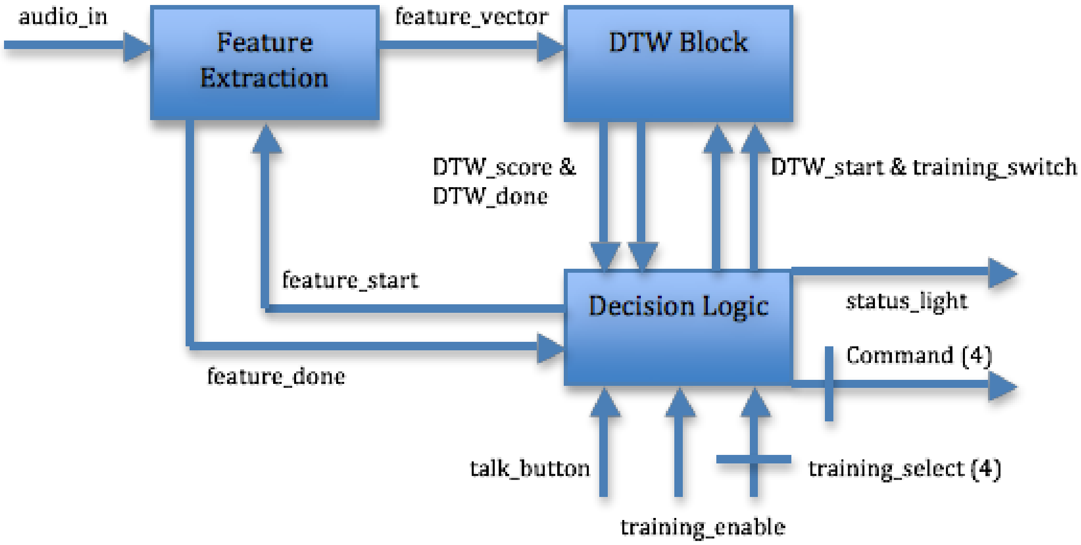

Figure 3: Block diagram of Voice Control Module. Incoming audio is buffered and fed in overlapping 10-ms chunks to the feature extraction module, which feeds data about the signal to the DTW blocks for training or comparison scoring. The decision logic keeps track of when calculations finish, sets the modes on the DTW blocks as needed (from external signals), and processes comparison scores to determine what command if any should be output.

Overall I/O

The voice control module will receive audio data from the third microphone, through the lab kit's microphone input jack and AC97 audio interface. This interface uses the staff-supplied AC97 driver code.

Additionally, there is a single-line switch input to determine whether the module is in training or recognition mode, and a bus input to determine which of the several blocks the training audio data is to be sent to. The width of this bus will be determined by how many distinct commands are needed. Both the control line and bus will most likely be implemented using lab kit switches.

The voice control module has two outputs used to interface with the video filter module. The first is a command bus, encoding the desired filter or combination of filters; the second is an indicator for when the video module should actually read the command bus (to avoid accidental switches of filters, or responses to invalid outputs).

For human interface, this module will also output a status indicator signal to indicate whether the system is currently recording a voice command. This will go to a labeled kit LED.

Internal Modules

The incoming audio data will go to a feature extraction module. This module will process chunks of the incoming data (about 10 ms), and extract a vector of audio features present in that sample. The exact design of this module is still to be finalized, but it will include sufficient memory or registers to hold about 1 second of audio data, a “start” line to trigger buffering of data, a “done” line to indicate when computation on the data block has finished, and an output bus to transmit the processed feature vector.

The feature vectors will be pushed in parallel to several DTW modules, each of which can be set individually to update its training model with that data. Each of these DTW blocks contains an individual control line to select between training mode and recording mode, set by the control submodule in response to outside input. When the DTW block is in training mode, it will add the incoming feature vector to its own internal model for future comparisons. When the DTW block is in recording mode, it will compare the incoming feature vector to its internal model, and output a comparison score to be sent back to the control logic.

In order to properly handle both training and comparing with the same DTW blocks, this module requires some internal state, which is kept by the control logic submodule. This submodule is responsible for taking in the global training enable and training select signals, and setting the mode of each of the DTW blocks appropriately. It also gives “start” signals and receives “done” signals from the feature extraction and DTW blocks to ensure computations are properly ordered and synchronized. Finally, the control logic module is also responsible for processing the comparison scores from the DTW blocks once computation is finished, and determining which command (if any) is matched by an incoming voice command.

Video Interface Module

The video interface module will handle all video interface, both receiving incoming video data from the external camera and pushing outgoing video data to the VGA display. It receives control signals from the voice control module to determine what filtering effects to apply on the outgoing video stream. The detailed block diagram for this module is shown in figure 4.

Ben will be responsible for this module.

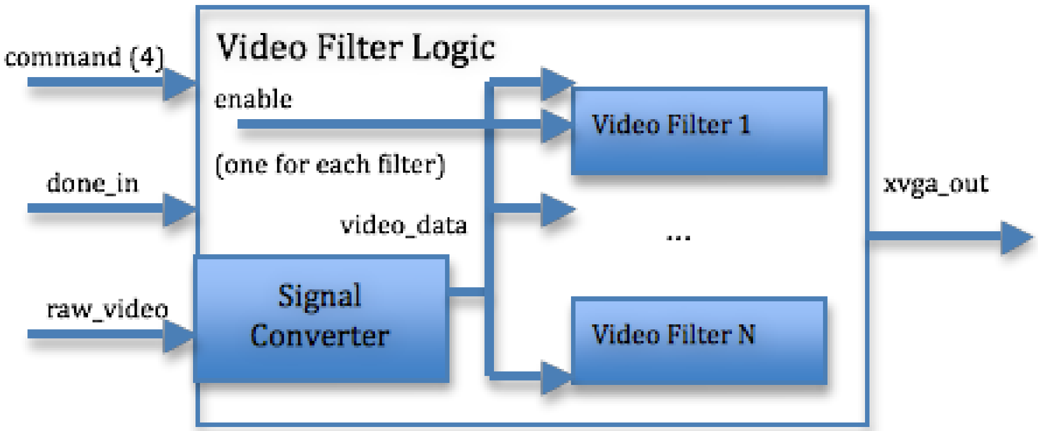

Figure 4: Block diagram of Video Interface Module. This module converts the raw NTSC video from the camera to RGB, and feeds this data to several filter blocks, which are multiplexed to get the final video output. A pulse on done_in locks in a command, which updates internal state and determines which filters are enabled and which is displayed.

Overall I/O

The incoming connections needed by this module as a whole are the connections to the lab kit's composite video port, to interface with the external video camera, and the control signals from the voice control module. The only outgoing connection needed is to the lab kit's VGA port, which will be connected to a monitor at one of the lab workstations.

Internal Modules

The camera interface module will handle decoding incoming NTSC signals from the camera, and converting it to RGB data suitable for VGA output. This code will be modified from the staff-provided video interface modules available on the course website.

The filter module will receive RGB data from the camera interface module, and pass it along to several filter block submodules. Each of these submodules will process the incoming RGB data and produce a filtered output, on a pixel-by-pixel basis. Internal logic will then determine which of these signals to send to the VGA display module.

The VGA display module will receive incoming frames of RGB data and generate properly-formatted XVGA signals, and will connect to the lab kit's VGA port.

External Components

Our project will require a few parts outside the lab kit, although all of them are already available in the lab or do not need to be purchased.

We will be building a frame to hold the camera assembly, which will contain the camera and two microphones mounted on a servo. Both the camera and microphones are available from the lab supplies. To capture speech for voice commands, we will be using the lab kit's headset microphone.

Part numbers
Servo: Hitec HS-311
Mic chip: Digilent Pmod 210-122

Possible Extensions

In addition to displaying the video streaming from the camera, we are also considering displaying a simple graphical equalizer, showing the frequency content of the audio on the voice control microphone in real time. This would require an additional submodule in the voice control module to do Fourier decomposition, an interface between the voice control module and video interface module to carry the vector of Fourier data, and a submodule within the video interface module to process these vectors and display bars of corresponding heights on the display.

At the moment, all video filters being considered would be operating on a per-pixel basis. A possible extension to this is to implement more complex video filters, such as motion blurring. This would require the addition of a frame buffer to the video interface module

Our current design requires the user to press a button before giving voice commands. Ideally, we want commands to be automatically picked out from the incoming data stream without prompting, or for recording to start after an audio trigger (such as clapping). This would definitely be an extension to look into after implementing all other functionality.

Timeline

By November 17, we hope to have proof-of-concept interfaces completed for the tracking and video modules. For tracking, this means being able to manually control the servos which will be moving the assembly; for video, this means having a working RGB display, converted from the signals being sent by the video camera.

By November 24, we aim to have the video and tracking modules essentially completed, and operable using the buttons and switches available on the lab kit. We also will be creating a software prototype of the voice control module, using either Python or Matlab. The major goals of this prototype are to assist in transitioning the system to hardware, as well as to make final determinations of the best feature extraction scheme to use for the final system.

Before leaving for Thanksgiving break, we hope to have mostly finished taking our prototyped system and implementing it on the FPGA. This leaves the 10 days or so after the break for finishing implementation and integration, and doing all other needed debugging and troubleshooting work that may remain.