Laser Shot: Video-based Alternative to Arcade Light Guns

Tiffany Chen, Spencer Sugimoto, Paul Yang

Introduction

Laser Shot is an alternative to the traditional light gun system used in arcade shooting games. Current light gun systems used in arcade Duck Hunt games are incompatible with LCD screens. To resolve this problem, Laser Shot uses a video camera system to track a laser point on the screen. This shooting system will then be used in an implementation of Duck Hunt, in which ducks will be targets for the laser-based gun.

The implementation of the project is divided into two major components: the dot finder, which tracks the laser, and the Duck Hunt system. The user will be able to interact with the system using labkit buttons and a laser gun. The implementation of Laser Shot and its work distribution are detailed in this proposal.

Dot Finder

The dot finder will use data from the video camera to locate the approximate screen pixel coordinates of the red laser dot. The camera will be positioned to give a clear view of the LCD screen. An overall block diagram of the dot finder is shown in Figure 1. The overall module has a *start* input and *busy*, *x*, and *y* outputs. *start* is used to signal to the dot finder module to begin the process of looking for a dot. *busy* indicates when the module has finished computation and the outputs *x* and *y* are the approximate screen pixel location of the dot.

The process of converting the image from the camera to screen coordinate of the laser dot is performed in three major steps. The first major step is processing the video signal and storing the resulting image into a video input frame buffer. The video signal is decoded by the NTSC decoder module to produce pixel values in the YUV color space. The video signal from the camera is first converted from YUV to RGB values and data from the two fields are merged into a single input frame buffer. Only the red channel (as the laser is red) is stored. Since NTSC resolution is approximately 720 pixels wide by 525 pixels tall, storing an entire frame of data will require 370KB of RAM, assuming 8 bits per pixel. Once the image from the camera is loaded into the input buffer, the next major step will be to process the image to locate the pixel coordinate of the laser dot in the input buffer. The point extractor module runs a series of filters to determine the estimate where the laser dot is. The point extractor module will require internal image buffer to run the processing, which will require another 370KB. Once the coordinates for the laser dot are found in the input buffer, the coordinate transformer module will perform a transformation that converts the coordinates of the dot in the input buffer to coordinates in the LCD screen. Both the point

extractor and coordinate transform module will be supplied coefficient data from a separate module. Overall, the dot finder will require 740KB of memory.

NTSC Decoder / Color Converter / Red Filter / Field Merger

The series of modules takes in the video data from the NTSC decoder and converts the YUV color to RGB, removes green and blue channels, and then generates the appropriate address and data signals to the input frame buffer module to write the red value of the pixel into the appropriate location.

Input Frame Buffer

The input frame buffer is a RAM that holds the full frame image retrieved from the camera. It is 370KB in size and has a single read port, and a single write port. The input frame buffer is read by the point extractor module to get the image for processing.

Point Extractor

The point extractor is by far the most complex sub-module within the dot finder. The point extractor can be decomposed into further sub-modules and a block diagram is shown in Figure 2.

The general algorithm for finding the location of the red laser dot is to first filter the image from the video camera to remove green and blue channels. The color filtering is performed as image data is loaded into the input frame buffer outside of the point extractor module. Once only the red channel data remains, a median filter is used to reduce the noise. Then a threshold filter marks the pixel values that are above a certain level. These pixels are the pixels that correlate to the pixels of the laser dot. Once these pixels are marked, the pixel coordinates of all the marked pixels are averaged to obtain the center pixel location of the laser dot.

Coordinate Transformer

The coordinate transforms the x, y pixel location on the of the image of the LCD screen to the actual x,y coordinates of the LCD screen. The coefficients for the transformation are obtained though a calibration process using the four corners of the LCD screen.

Duck Hunt System

The Duck Hunt system implements the traditional arcade Duck Hunt game. A ROM map contains the background pixel information, and the various objects displayed on the screen are implemented using separate modules. These

modules include the score keeper, bush generator, and duck handler. These four components are fed into a video controller, which controls the layering of these components. The video controller then passes the pixel's RGB display information to the LCD. Audio effects will be stored in a ROM and fed to the audio handler, which sends the appropriate sound signals to the speakers. The specific signals and interactions between these signals are illustrated in Figure 3. A global reset signal resets the game. The screen dimensions will be 1024 x 768.

Audio ROM

This ROM stores the audio information for the game. These sound effects include the shuddering of the bushes as the duck emerges, the "pop" sound of the gun, and the dying sound of the duck. The ROM takes as input the appropriate address from the audio handler and outputs the appropriate sound effect. The ROM should be able to store around 144 kilobytes.

Audio Handler

The audio handler determines which signal should be retrieved from the audio ROM and feeds the sound to the speakers. To determine which audio sound to output, it takes as inputs from the dot finder so that every time a dot's x and y coordinates are asserted, it requests the "pop" sound. The duck's health signal is another signal the handler has as input so that on every positive edge of the signal, the shuddering sound of the bushes will be requested, and on every negative edge, the duck's dying sound will be requested.

Duck Handler

The duck handler handles all signals related to the duck. The duck hander takes in the x and y coordinates of the laser dot from the dot finder and determines whether or not the user has shot the duck and changes its output duck life signal accordingly. The duck handler also receives signals from the duck movement and random number generator to determine the direction and frequency of the duck. The duck handler also implements animation of the duck, such as the flapping of the wings and its death. The corresponding output RGB display signals are sent to the video handler for layering of the various objects on the screen. It also sends enable signals to the random number generator and duck movement modules. The enable signal to the random number generator is asserted when the duck dies or is shot.

Duck Movement

The duck movement generator controls the direction of the duck and outputs the direction when its enable signal is asserted by the duck handler module. The duck can travel at various directions northeast or northwest of the screen.

Random Number Generator

The random number generator how often the duck shows up on the screen so that duck appear at random intervals. It outputs a random number to the duck handler when it receives an enable signal from the duck handler.

Bush Generator

The bush generator creates the bushes and animates the bushes. Its output signal is passed to the video handler so that it is always on the top layer.

Score Keeper

The score keeper keeps track of the number of ducks shot and sends the appropriate information to the video handler to display the scoreboard. Each time the life signal from the duck handler transitions from a high to low, the score keeper will increment the score and update the information on the scoreboard. The scoreboard will also involve character generation to display numbers and letters. The scorekeeper will also keep track of the remaining time left in the game with a timer: if the user does not kill a certain number of ducks within the time frame, the user loses and the game is over. The RGB information associated with the scoreboard will be sent to the video handler.

ROM Map

The ROM map will store the background image for the video output. The RGB information is sent to the video handler.

Video Handler

The video handler determines the layering of the score keeper, bush, duck and background. The handler takes as inputs the RGB information from the score keeper, duck handler, bush generator, and ROM map and outputs the appropriate RGB information after priority encoding. The background is always be at the bottom layer, whereas the duck appears behind the bushes but is on the top layer when it flies out of the bushes.

Testing

The dot finder module will be tested part-by-part as the sub-modules are developed. The NTSC decoder will be first tested by copying the data from the video input to the LCD monitor. If an image is properly displayed, then the NTSC decoder, color converter, field merger, and input frame buffer will be assumed to be working. Then, the various filters in the point extractor module will be tested individually for functionality. Once the point extractor is complete, the coordinate

transformer will be tested to see if the image coordinates can be properly mapped to screen coordinates. The final test setup will incorporate all of these parts in a module that will track the laser dot and display an X on the estimated laser point on the LCD screen.

The duck hunt game system is broken down into very specialized modules, allowing for straight-forward testing. The Video Handler is pivotal to the testing of the other modules, so it will be tested first. Its ability to prioritize images, which both try to occupy the same pixel, will be tested. After the Video Handler works by itself, the ROM Map, which supplies the background; the Bush Generator, which places bushes on top of the background; and the Scoreboard, which displays the score on top of the background will be implemented and tested with the Video Handler. This sets the stage for gameplay.

Next, the modules that concern ducks will be implemented. The Duck Handler will be tested to be sure that it can register a kill, by comparing the input x and y values with the x and y values of the duck it is handling. The Random Number Generator will be tested for ability to produce seemingly random numbers, within a given, preset range. The Duck Movement module will be tested for ability to created straight-line paths for the duck to traverse (by incrementing or decrementing x and y).

The Audio Handler will be tested for its ability to output the correct sounds at the correct time. The Audio ROM needs to store the sounds that Audio Handler needs, so it will be tested for it's storage and recall functionality. After the Audio Handler is shown to work with manual test inputs, it will be tested in concert with the Duck Handler and, after that, the real x and y inputs from the Dot Finder, completing the entire system.

Potential Issues with the Dot Finder:

There are several issues that may arise while detecting the laser dot on the LCD screen. One of the first challenges is making sure that the laser dot is bright enough on the screen. The surface of the LCD screen is not very well suited for reflecting a laser beam and the reflection may be too weak to differentiate from the background. If the laser beam’s reflection is too weak, a pane of lightly frosted glass could be placed over the screen to provide a better reflection surface while still allowing the screen to be seen. Even if the dot were bright enough, another problem would be obtaining a usable image from the video camera. The video camera may over expose the image of the LCD screen, resulting in saturated values. If the pixel values are over saturated, it will be impossible to differentiate the laser dot from the background. The exposure could be reduced though neutral density filters or a reduction in the shutter speed of the video camera. Finally, the last potential issue deals with the accuracy of the dot finder. For the purposes of the Duck Hunt game, the location of the laser

point must be approximated with a reasonable degree of accuracy. If the laser point is moved rapidly, the point will appear as a line in the video frame. Using the point detection algorithm on a line will result in less accurate values for the position of the dot. Furthermore, the transformations that convert the image coordinates to LCD screen coordinates may not be accurate enough. As the modules are developed, tests will be run to determine the severity of the aforementioned issues.

Division of Labor

Paul will be working on most of the dot finder modules, while Tiffany will be implementing the video part of the duck hunt game. Spencer will be in charge of the audio components of the duck hunt game as well as the coordinate transformer and median filter for the dot finder.

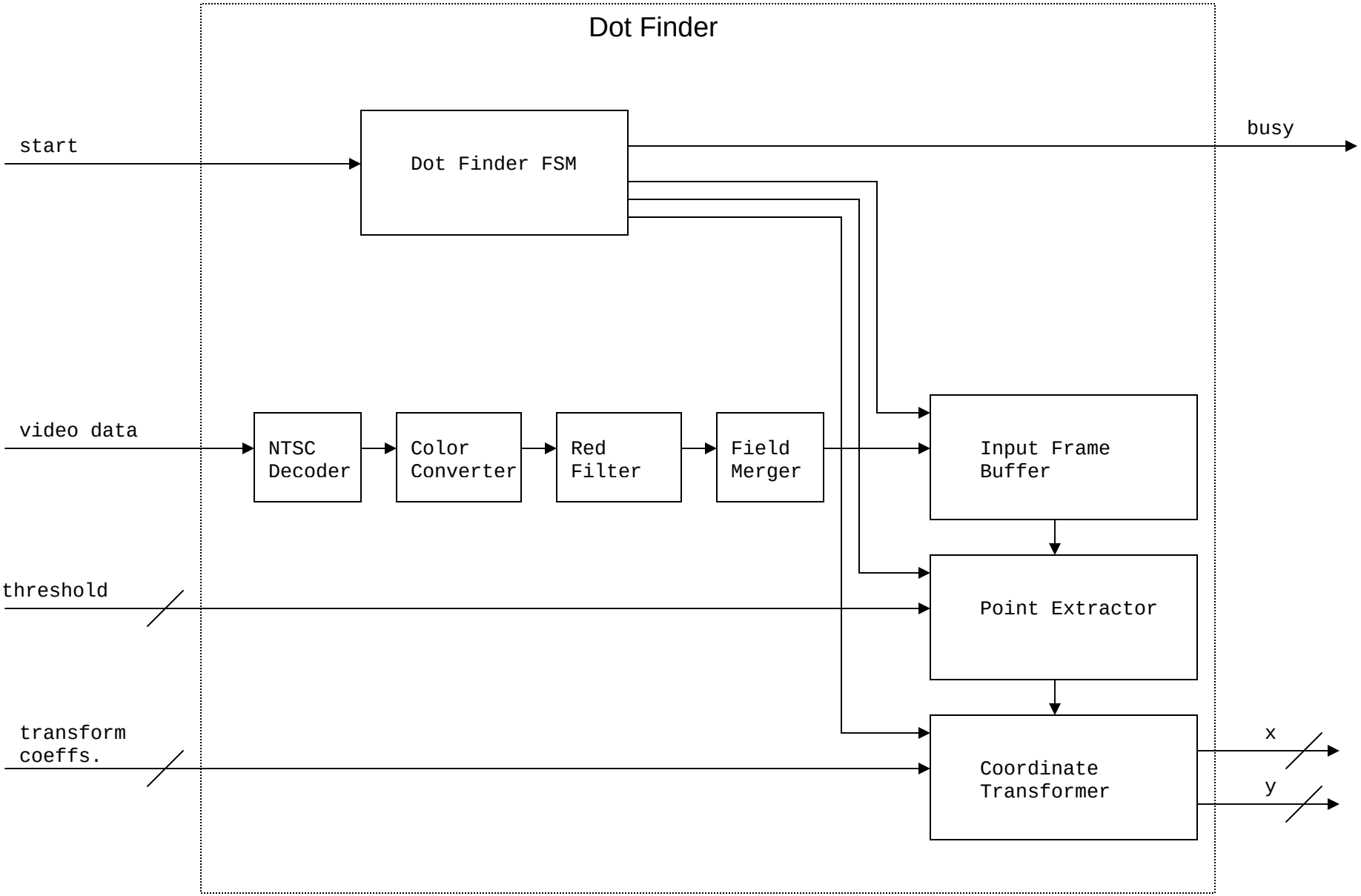

Figure 1: Dot Finder Block Diagram

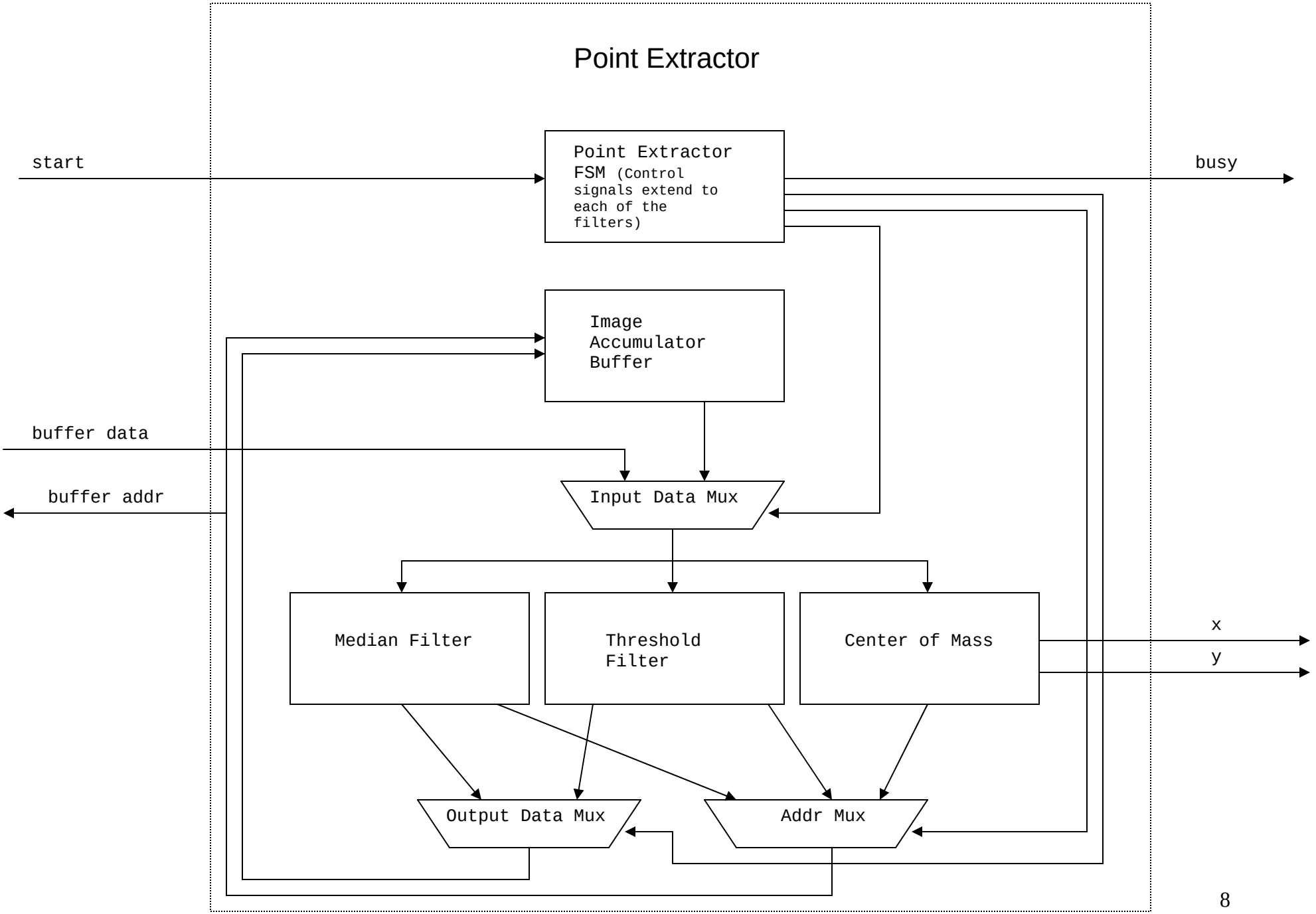

Figure 2: Point Extractor Block Diagram

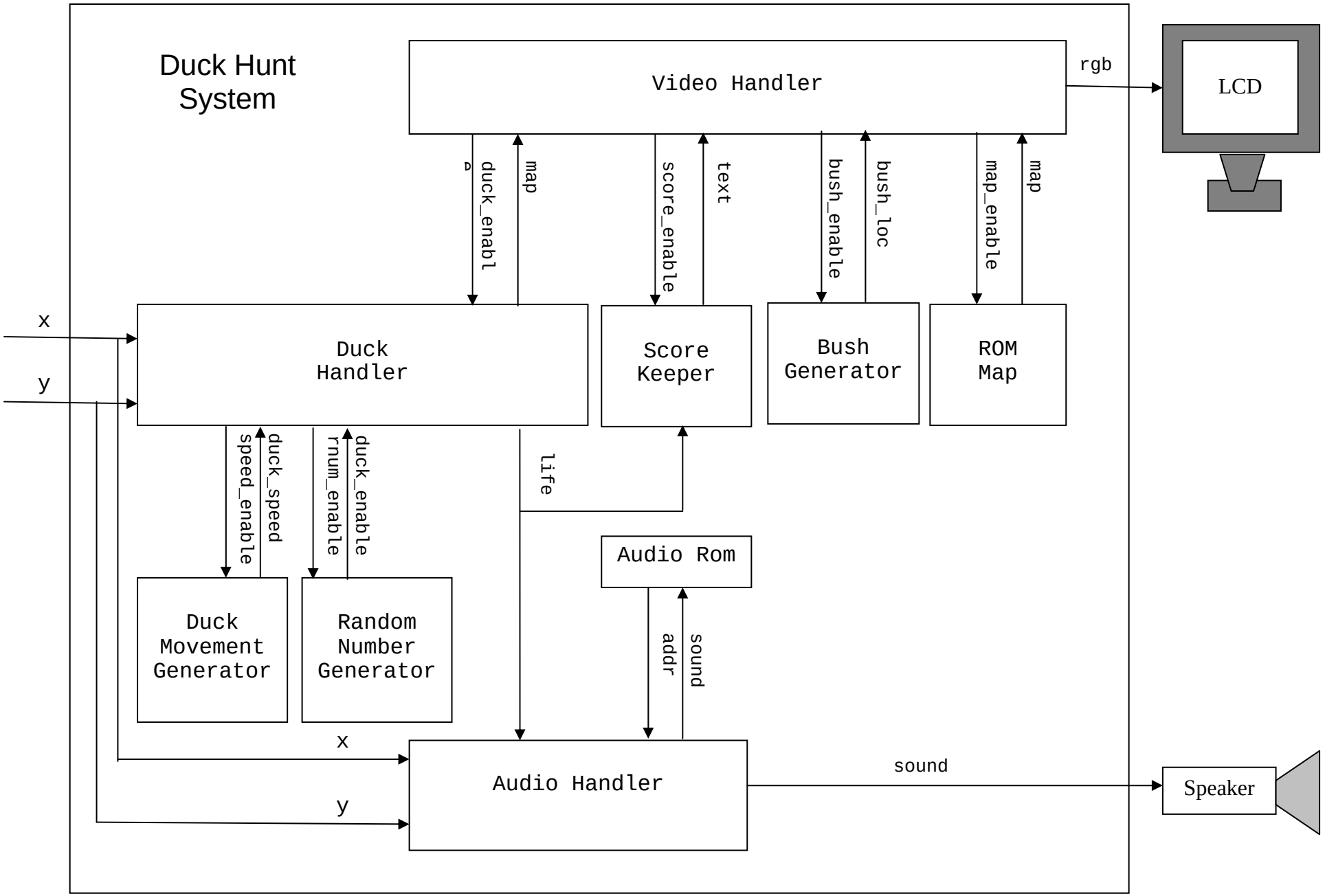

Figure 3: Duck Hunt System