A Motion-Tracking DMX512 Controller

Miren Bamforth | 6.111 Project Proposal | Fall 2014

1 Overview

Some modern theatrical lighting instruments are able to rotate in two dimensions; they are referred to as 'moving lights.' Integrating motion tracking capabilities with a moving light would create a real-time spotlight which automatically follows a person or an object onstage. This final project aims to make a device which takes video input from a bird's eye view of a performance space, processes calibration and motion tracking data, and outputs serial data to control a set of moving lights. Currently, following a person or object with a light is done by hand and could be improved with automation.

The motion tracking system is the most important module of the project. It will read the video input and compare multiple frames to find an object which has moved. Initially, motion tracking will be tested with a uniformly colored object on a uniformly colored background; a computer display will act as a debugging tool. As the system's precision increases, the motion tracking will be able to track objects by pattern. Ultimately, the project will be able to track a person and will use a moving light, not a computer screen, for debugging and proof of concept.

Stretch goals of the project include automatic instead of hard-coded calibration, data reading in addition to data writing, and color matching. The ideal outcome is to be able to package the project as a freestanding device which could be used in future performances and research.

2 Design

2.1 Terms

The following are some terms which will be useful to understand the rest of this document. These are industry standard terms used by electrical engineers, theater designers, and technicians.

- **DMX** or **DMX512**: The serial protocol which is used to control theatrical elements such as lighting, fogging machines, and automation
- **Lighting board**: A piece of hardware which outputs DMX512. Typically used to control lighting elements
- **Moving light** or **intelligent lighting**: A theater light which is capable of panning and tilting in place. Usually also able to change color and pattern. Controlled with DMX512
- **Conventional light**: A light which is not capable of changing position, color, pattern, or any aspect other than its brightness. Controlled with DMX512

2.2 Project Overview

The project as a whole can be modeled as four major blocks as shown in Figure 1; these blocks are the motion tracking block, the DMX processing block, the calibration block, and the display and testing block.

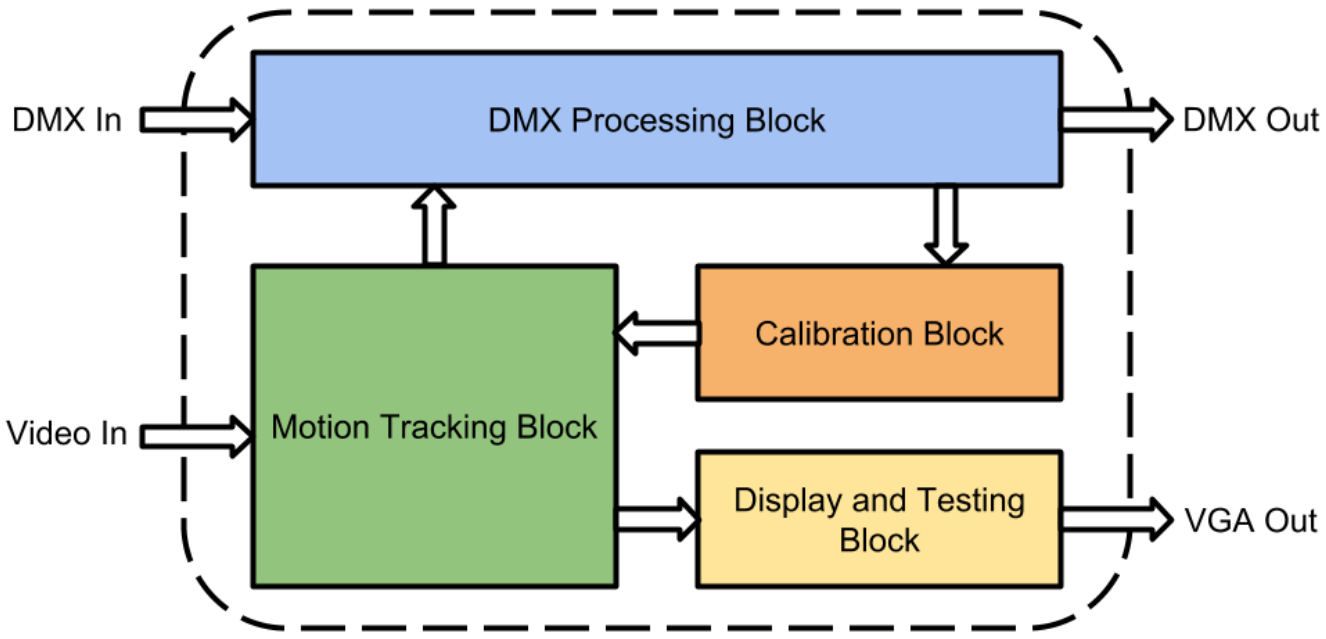

Figure 1: The high level design of the system

2.3 Design Decisions and Motivation

The primary motivation for this project is to make a freestanding device which could be used as part of future projects. Completing this device requires finishing every module, even the optional ones, which is a lofty goal given the time frame and resources of the project. Given this motivation, I made various design choices which lead towards a freestanding device while creating interesting, testable intermediate steps.

At the highest level, the project in its complete form is designed to intercept DMX data ('DMX in' from Figure 1) from a lighting board, process video data ('video in'), combine calibration data with the processed data, and output altered DMX (DMX out) to the lighting instruments for which the original DMX was intended. The specific DMX data bytes which are changed are the bytes that control the pan and the tilt for a given moving light. The system also outputs video (VGA out) for testing purposes and to allow the project to be functional without a DMX-controlled lighting instrument.

The **motion tracking block** operates at 10 frames per second to alleviate the data bandwidth issues that parallel video and DMX processing creates. This block will store video data in memory. It compares two to four different frames each cycle to detect object movement; the final number of compared frames will be chosen based on testing of the precision of the motion detection. Optionally, the motion tracking block can also be used to do color matching between the object and the intelligent light's output color.

The **calibration block** will be hard-coded at first because it is the least important block. If this project becomes a device, an automatic or remote calibration controlled by DMX would be essential, but it is

not necessary for the basic functionality of the project. This block will be simplified first if there is a time crunch at the end of the project.

The **DMX processing block** takes in DMX data and overwrites some of the data based on signals from the motion tracking and calibration blocks. This block is less important than the motion tracking block but more important than the calibration. This block can be split into write-only or read-and-write, so testing with a moving light can occur even if the reading functionality is not yet implemented. The DMX block stores the DMX data in registers for easy access, assuming that there are enough registers for DMX data storage and for the registers required by the rest of the system.

The **display and testing block** outputs motion tracking and calibration data to a computer monitor over VGA for easy debugging and testing without having a moving light in lab.

3 Implementation

This section explains more of the details of the system and verbalizes the block diagram shown in Figure 2.

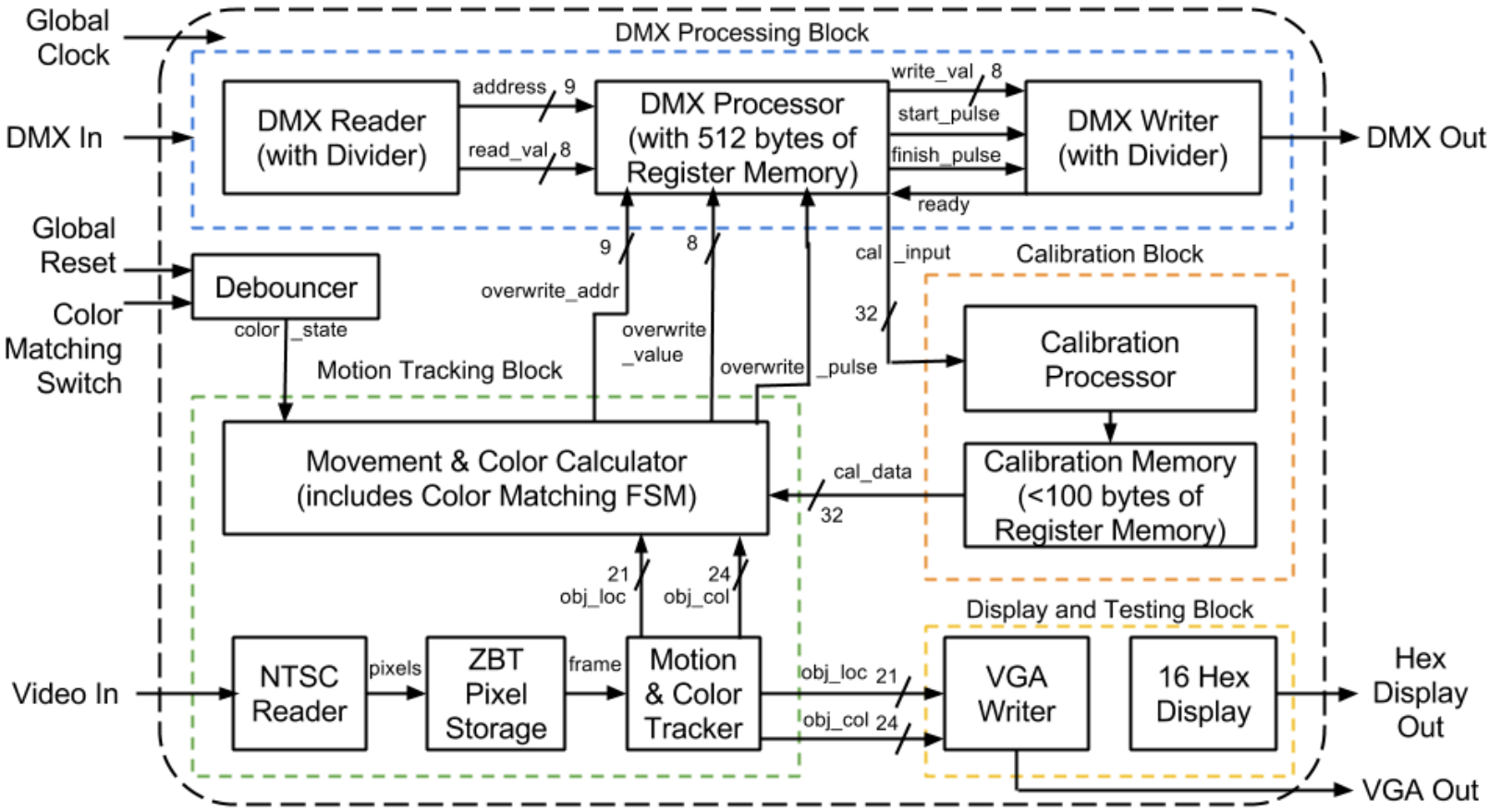

Figure 2: A detailed block diagram showing the individual modules and the links between modules

3.1 Motion Tracking Block

The motion tracking block takes in video data and outputs information for video and DMX output. The first module of this block is the NTSC reader which takes in the raw pixel data from the video stream and stores it as frames in ZBT memory. The motion and color tracker reads frames from ZBT memory and looks for clusters of pixels which either match a certain color or optionally match a certain pattern. At a basic level, the block will average the location of all pixels in a frame within the color range. This algorithm has not yet been fleshed out further, so specification like how many arithmetic operators are necessary is not available. However, it is already clear that this module will be the limiting bandwidth factor for the project. We are operating the motion tracking block at 10 frames per second because of the following approximate bandwidth equation:

$$Bandwidth \geq (10\ frames/second)\,x\,(3\text{x}10^{6}\ pixels/frame)\,x\,(3\ bytes/pixel) = 9\text{x}10^{7}\ bytes/second$$

This bandwidth is high but is within the capabilities of the labkit.

When the motion and color tracker has found the object, it outputs the most recent location of the object's center and the object's most recent color to the movement and color calculator module and the display and testing block. The location data is 21 bits total: 11 bits for horizontal placement and 10 bits for vertical placement. The frames stored in memory are 1024 by 768, hence the choices for 11 bits and 10 bits. The color data is 24 bits; there are 8 bits each for red, green, and blue.

The movement and color calculator module takes in calibration data and location data. Given the distance of the light from the plane of the camera (a value stored in the calibration data for each light), the module calculates the new pan and tilt for that light using basic trigonometry. The pan and tilt bytes are sent to the DMX processor to overwrite the old values. Additionally, if the module is in color mode, the color read from the object is sent to the DMX processor to overwrite the current color button. Color mode is turned on with a debounced button or switch on the labkit. These bytes are sent to the DMX processor with the 9 bit address to overwrite and with an overwrite pulse that prompts the DMX processor to store the new values in memory.

3.2 Calibration Block

The calibration module comes in two forms. At first, the calibration memory will be hard-coded to include information such as the height of the camera, the height of the moving light, and the addresses of the bytes of data which refer to the pan, tilt, and color of the moving light. Without this data, the system could not do the trigonometry to compute the correct changes in the position of the light. If there is enough time at the end of the project, the calibration module will be controlled by reserved bytes of the DMX input data. This DMX data will be processed by the DMX processor and passed to the calibration processor for analysis and storage.

3.3 DMX Processing Block

The DMX processing block consists of three modules. The first module, a DMX reader, is optional and reads the serial stream of data. Whenever it sees the start code as specified by the DMX512 protocol, a counter inside the reader is set to zero. For each packet, the reader outputs the byte of data and 9

bit address of the byte which is equal to the value of the counter when the byte was received.
The DMX processor stores DMX from the DMX input in 512 bytes of registers because 512 bytes is the maximum amount of data transmitted over one DMX universe. The processor overwrites the input data with data received from the motion tracking block. The processor outputs the 512 bytes in the correct order when the DMX writer indicates that it is ready.

The DMX writer indicates that it is ready to the DMX processor whenever it is in the idle state. The DMX processor gives the writer a start pulse to give the writer time to output a start bit before outputting the data bytes in ascending address order. Up to 512 bytes may be sent, but not all 512 bytes must be sent; therefore, the processor sends a finish pulse when it has given the writer all of the non-zero data bytes. The DMX writer outputs the bytes as specified by the DMX512 protocol; the baud rate of DMX512 is 250,000 packets per second. Because there is one start bit, eight data bits, and two stop bits in each packet, this corresponds to a 2.75 MHz clock if one bit is written each cycle. The DMX writer has a divider component which gives it a pulses approximately ever 364 nanoseconds. The DMX reader has a similar divider, but that divider lets the reader over-sample the data to get a more accurate read of the data stream.

There are no arithmetic operations in the DMX block other than Boolean compares. The most challenging part of the block is storing and overwriting values to the 512 bytes worth of memory in an efficient way.

3.4 Display and Testing Block

The display and testing block takes in various data from the system and outputs data which will help with debugging. The VGA module takes in object location and color data from the motion tracking module. It outputs the location of the object in the form of a colored dot matching the color of the object on the screen. This colored dot can be displayed on a black screen or overlaid on top of the corresponding frame for debugging.

Various components will send their state and other data bits to the hex 16 display module which will output values to help with debugging. The input for this block will be changed as different modules enter the testing phase.

3.5 Optional Features

The first optional feature is the extended calibration module as explained in the calibration block section. The second optional feature is the DMX writer module as explained in the DMX block section. The third optional feature is implementing corner detection logic to detect an object instead of using color. Using something other than color will remove issues such as changes in color due to the color of a moving light's beam, but the scope of corner detection is quite large given the amount of time for the project.

4 Testing

The basic testing for each module consists of creating Verilog test benches to be used with ModelSim. As the modules are integrated together, test benches for combinations of modules will be made. DMX

reading and writing can be debugged with ModelSim or with the logic analyzer.

The VGA output will be used for testing by displaying the current location of the object; meanwhile, the hex 16 display will be used to output useful values like states. When this test works with the color reading, the pattern tracking will be implemented and tested in the same way to make sure that the pattern is being processed correctly.

After all of the modules have been integrated, the project will be tested with a DMX-controlled moving light that I will bring to lab. With an actual light, the project's functionality will be easy to troubleshoot. I will need some extra components to do testing with the moving light; the necessary resources for my testing are listed in the resources section.

Finally, if I finish the project early and manage to port some or all of it to a smaller FPGA in time, there is a performance during the week of December 3rd that I am involved in which could be a testing location for my device. The performance is in Kresge Little Theater and uses 13 moving lights. This production could be an opportunity to test my device on a large scale, although it will be a challenge to finish and port the entire project before that date.

5 Schedule

Figure 3 is a Gantt chart showing the proposed schedule for the project.

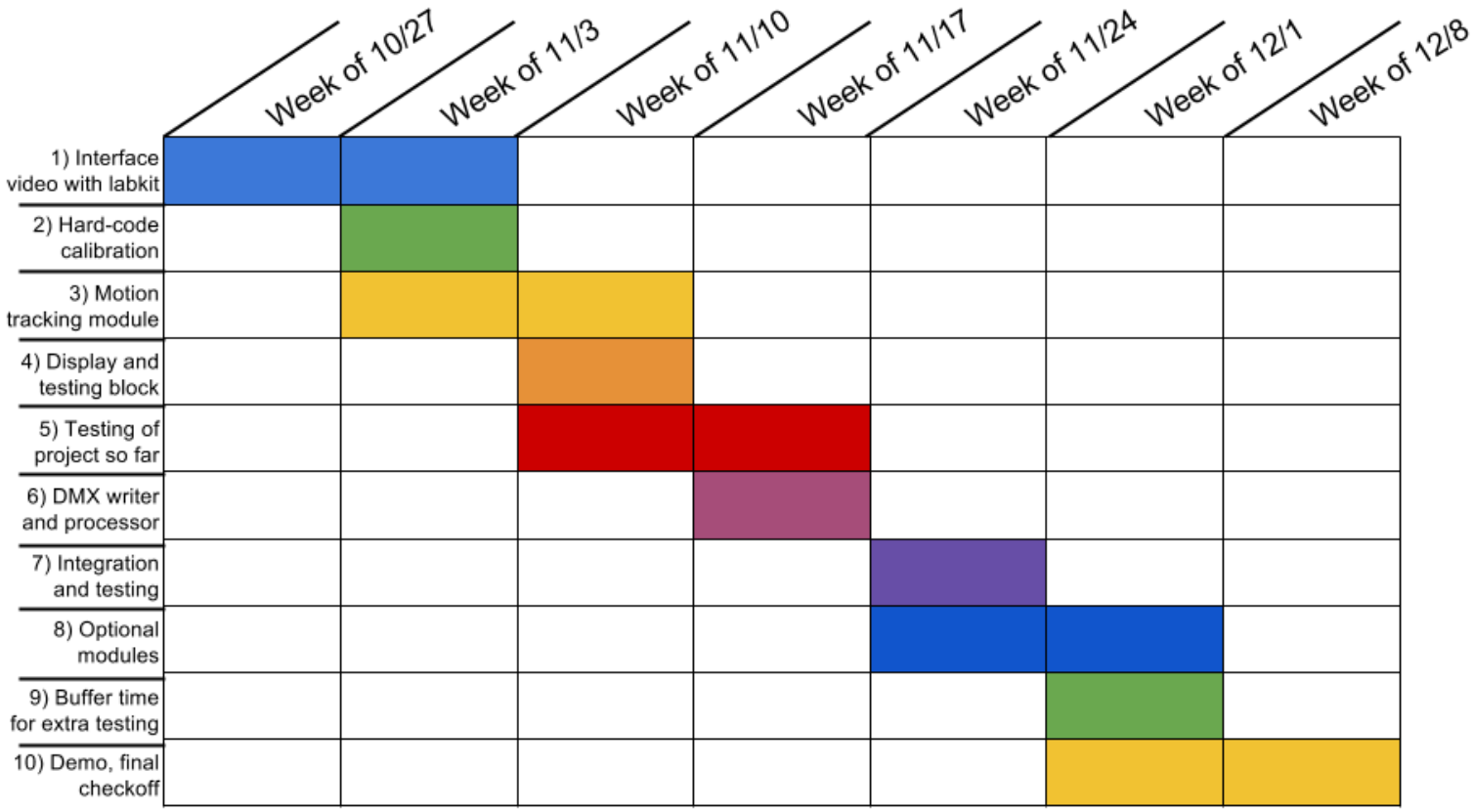

Figure 3: A Gantt chart of the proposed schedule

Here we elaborate on the numbered steps in Figure 3.

1. Interfacing video with the labkit includes reading the hardware documentation for the component video input and writing the NTSC reader and ZBT storage modules
2. Next, the calibration will be hard coded with the values used for testing in lab
3. This step completes the rest of the motion tracking module
4. The display and testing block is created next to allow for extensive testing
5. At this point, the project is tested to see if it meets its baseline requirements
6. The DMX writer and processor are created so that testing with a moving light can occur
7. Integration of all modules and more testing are next to allow for plenty of time in case of errors
8. If there is extra time, optional components will be implemented at the end
9. There is a week of leeway given in case the project is running behind or has unexpected setbacks
10. Finally, the demo and final checkoff occur at the end of the semester. This week is also when the final paper is due

6 Resources

This project needs a few extra parts to build and many extra parts to test. I need one female and one male 5-pin DMX512 connector to wire to the labkit. The connectors are $5 to $10 each. These connectors are soldered to wire leads which are connected to inputs on the labkit. For testing, I need a set up with a moving light, XLR cable which transmits DMX, a stand to which I will attach the moving light and the camera, and possibly a lighting board. I have access to these items through a student group that I work with, but there might be a small rental fee. I can use a USB to DMX converter instead of a lighting board if I can find one. The USB to DMX converter or the lighting board would supply the DMX input by connecting to the male 5-pin connector with XLR cable. The DMX output would connect to the moving light with more XLR cable connected to the female 5-pin connector. I have experience with all of these items, so cost is a more prohibiting factor than complexity.

A description of the DMX512 protocol that will be used for this project can be found here:
http://www.opendmx.net/index.php/DMX512-A

6 Conclusion

The motion-tracking DMX512 controller is a project which I will enjoy building. I have worked extensively with theater and performance technology, but I have never used video integration as part of a project. Since this class is substituting for my thesis, I plan to put many hours into this project to get it as close to a stand-alone device as possible. Testing the system with a real moving light will be very exciting, and I am very motivated to get to that part of the project as soon as possible. Overall, this project will give me new experience which ties into my current research, and I hope that my device will be functional and useful for future projects.