ENSC 305W/440W

Written Progress Report

March 30, 2015

Behnaz Edalat	301127510
Brendan Keane	301167176
Martin Palibroda	301179246
Justin Raine	301041695
Calvin Scott	301169228

Introduction

The vitaMotu Hermes™ is an aftermarket motorized motion controller for existing camera slider systems. The Hermes enables independent videographer and professional filmmakers to capturing smooth time lapse and stop motion recordings at a below-market cost. Controlled via the free Hermes Controller app available on the Apple App Store (pending submission), the vitaMotu Hermes benefits from wireless control, an intuitive user interface allowing customization of all controller parameters. Future enhancements available as free software updates and a durable enclosure (production version) will ensure the vitaMotu Hermes will remain a core tool by vitaMotu customers.

Schedule

In regards to the Gantt chart (Figure 1) originally used to outline the various progress steps in the development of our project, it may be seen that we are on-time or ahead of schedule in most areas, and significantly behind for other areas. Notable sections that have exceeded the timeline in Figure 1 include building the iPhone software application and interfacing the app with the microcontroller. However, this should not be a cause for alarm, as the initial estimation was rough, and developing any mobile application requires a much larger amount of time. Furthermore, while the software development is not yet complete, it is on track for completion approximately one week prior to the deadline. Every other area section is either completed or right on schedule, and currently we are finishing integration and moving forward with continued testing.

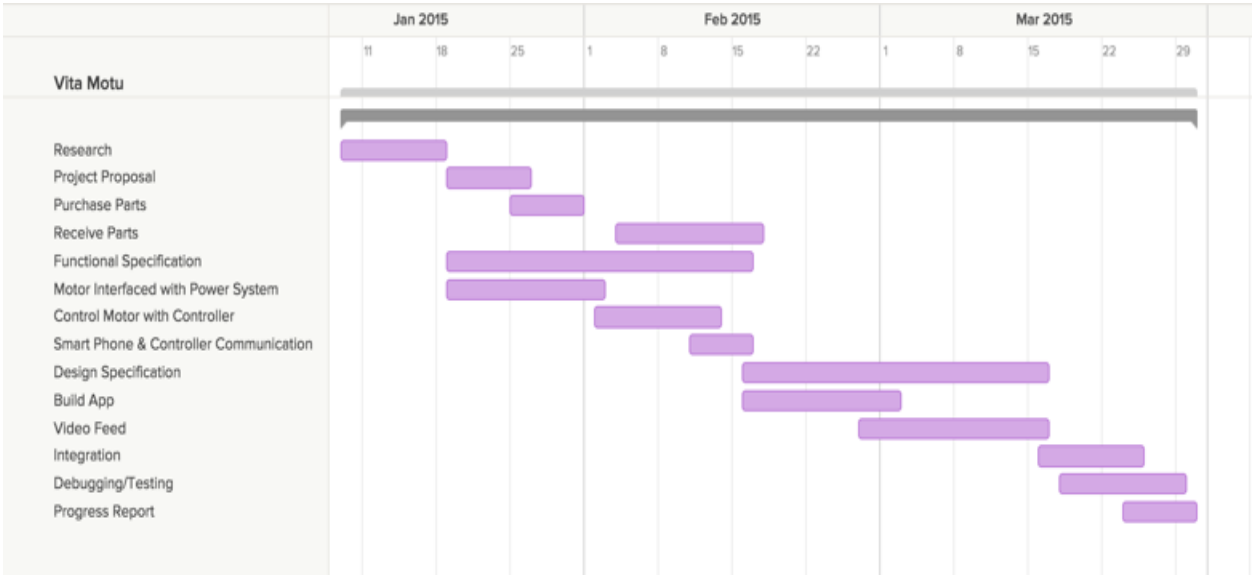

Figure 1: Gantt Chart

Financial

Currently, we have spent 90.8% ($535.97) of our budget and we have $54.43 remaining for last last minute expenses. Estimated and actual costs are compared in details in Table 1 below:

Item	Estimated Cost ($)	Actual Cost ($)	Difference ($)
Raspberry Pi and Power Adaptor	**60.00**	**51.71**	**+8.29**
JR DS8231 Ultra Precision Servo	**90.00**	**27.00**	**+63.00**
Arduino Uno	**40.00**	**39.14**	**+0.59**
Nano Bluetooth Dongle	**5.00**	**0.00**	**+5.00**
Edimax WiFi Adapter	**15.00**	**0.00**	**+15.00**
Cirago Bluetooth and WiFi	**45.00**	**25.82**	**+19.18**
22.2 V 2.6 Ah Battery	**100.00**	**90.51**	**+9.49**
Smart Charger	**37.00**	**0.00**	**+37.00**
Enclosure	**100.00**	**80.00**	**+20.00**
Contingency	**98.40**	**223.53**	**-125.13**
Adafruit Motor Shield for Arduino		50.31	
Breadboarding wire bundle		7.77	
N3 Cable		18.31	
3D Printed Gear		8.00	
Premium Female/Male Jumper Wires		2.52	
5V 2A Switching Power Supply		10.29	
Bluefruit LE - Bluetooth Low Energy		25.82	
Apple Developer Account		119.00	
Total	**590.4**	**535.97**	**54.43**

Table 1: Estimated and Actual cost

Progress and Remediation

Hardware (Completion: ~95%)

All components relating to the physical aspects of the system have been tested and are completely functional. A status summary of the key hardware in our project is detailed below. We are almost on schedule for this section of the project, save for a small problem as mentioned in the following section.

Stepper Motor

- The stepper motor has been tested across varying modes, speeds, and times.
- The achievable motion is mostly within the RPM ranges specified in the Requirements Document.
- The motor provides suitable torque under load.

One potential problem that we are facing is that the motor seems to generate a noticeable amount of heat when operating under 1 RPM, which is a crucial speed range for long length time lapses. Additionally, the maximum speed of the motor is under the estimated maximum speed that we determined in earlier specifications. However it still reaches speeds in an acceptable range for the prototype which currently uses a relatively low cost motor. Installation of a more expensive and powerful motor will resolve this issue. The remaining time spent finishing the hardware for this project will be researching heat sinks.

Motor Mount and Drive Gear

- A wooden motor mount has been created for use in the prototype unit. Production versions will use a CNC'd aluminum equivalent.
- The motor has successfully been mounted to the motor mount.
- The gear for the motor has been designed in Solidworks and printed on a 3D printer.
- Operation has been tested with minimal to no slippage between the drive gear and drive belt.
- The motor assembly has been attached to Cinevate Hedron camera slider's belt system and tested with motor operation.

Arduino Uno Microcontroller

- Completely functional microcontroller.
- Testing has been completed for uploading and running codes for interfacing with the various sub-circuits implemented.
- The Arduino successfully drives the motor via the Adafruit Motor Shield.

The Arduino Uno itself may not have enough pins necessary to complete all of the functions originally outlined. However, core functionality is not inhibited, and the only change required is the lacking of LEDs to display current system information. A production version of the device will use a custom PCB with sufficient I/O to meet the required specifications.

Adafruit Bluetooth LE Wireless Chip

- Seamless circuit configuration when connected to the Arduino microcontroller.
- Serial UART functionality has been successfully tested can successfully send and receive data transmission via the Adafruit sample app.

Firmware (Completion: ~85%)

Programming for the microcontroller has progressed well. Previous problems such as limited memory capacity have been largely resolved as described below. A status summary of the various firmware sections in our project is detailed below.

Drivers

- Required libraries used for interfacing with Bluetooth and motor shield components have been loaded on the Arduino.
- Library functionality has been independently tested and functions correctly.
- All necessary functions have been implemented into the microcontroller firmware.

Earlier in the development of the microcontroller firmware, the available Arudino memory was quickly filled to ~75% by the libraries implemented. As further development was still required, exceeding the available on-board memory was a substantial risk. However, an optional compiling setting was found which only uploads library functions critical to the operation of the program. This proved extremely helpful in fixing the pending memory issue. The completed firmware program has been completed and can successfully be stored in the on-board memory.

Microcontroller Instructions

- The code responsible for Bluetooth packet reading and decoding has been completed.
- Iterative loops for Live, Time Lapse, and Stop Motion modes have been integrated.
- The microcontroller can successfully send START/STOP commands to a connected camera.
- Tracking of slider position has been implemented in order to ensure the motion controller does not move the camera beyond the physical bounds of the slider.

Most of the programming needed for microcontroller processing has been completed. However, control algorithms for conditional Bluetooth disconnection and corresponding error code transfers to the iPhone application requires further development. Additional testing in the next week will be used to confirm that reliable data communication is possible over prolonged periods between the vitaMotu Hermes and the Hermes Controller app.

Software (Completion: ~90%)

The iOS software development of the Hermes Controller app is currently ongoing but on track for completion by April 3, 2015. This allows approximately one week to continue testing for unknown bugs, optimizing code efficiency, and polishing the user interface. A detailed progress summary of the core features of the app is given below.

User Interface

- The UI has been designed and implementation has been completed in Interface Builder (a component of Xcode).
- All responsive user interface elements have been tested and successfully perform live and/or animated updates.

Bluetooth Connection and Data Transfer

- The CoreBluetooth framework has been implemented in the HermesControllerManager class with all necessary methods necessary to initiate a Bluetooth scan, connect to a specified Bluetooth peripheral, and transfer data packets via the Bluetooth radio interface.
- The Bluetooth LE connection process has been tested and successfully scans for nearby devices and is able to connect to the VitaMotu Hermes motion controller with minimal delay.
- While implemented, the data transfer functionality of the HermesControllerManager class has yet to be tested. Testing will commence on Wednesday, April 1.

Data Transfer Packet Specification and Implementation

- The proprietary packet format for communication between the vitaMotu Hermes motion controller and the Hermes Controller app has been completed defined and consists of a 12 byte packet with all necessary data fields. See the vitaMotu Design Document for implantation details.
- Creation of all packet types has been implemented.
- Created packets have successfully been tested to show the packets accurately reflect the user-specified parameters, as input in the UI.

Help View

- The *Help* view (screen) of the mobile app has yet to be completed and represents the only visible indication the app is not yet complete.
- Help documentation – including FAQs, bug reporting information and development credits – have not yet been written.
- Once written, the *Help* contents simply needs to be pasted into the relevant sections of the *Help* view. Copy writing will be completed by Friday, April 3, 2015.

Overall, the Hermes Controller application is on track for completion on April 3, 2015. Any remaining time will be spent performing additional bug testing, improving code efficiency, and improving UI where necessary. Once completed, the app will be submitted to the Apple App Store for review and, ultimately, client availability.

Summary

The vitaMotu team is proud to report that progress in the Hermes motion controller project is largely on track for the upcoming deadline and presentation. Issues that may occur during the final testing phase should be resolved quickly, as our current timeline allows one week for debugging and resolving unexpected issues. While a few changes to the scope of the product were made early in the development process (such as live video streaming to the app) due to technical limitations, our project is largely on track to meet the initial requirements and within budget.