Apparatus for Wavefront Error Sensor Measurement: Manufacturing Status Review

February 7, 2018

Authors: Robert Belter, Ali Colic, Jake Crouse, Lucas Droste, Diego Gomes, Ankit Hriday, Owen Lyke, Brandon Noirot, Owen Shepherd, and Brandon Stetler

Customer: Eliot Young
Southwest Research Institute &
NASA Glenn Research Center
1.0 – OVERVIEW

1.1 – Motivation
1.2 – Objectives
1.3 – CONOPS
1.4 – Functional Block Diagram
1.5 – Baseline Design
1.6 – Critical Manufacturing Elements
Project AWESoMe will characterize the performance of two wavefront detectors in order to determine which to use on an upcoming balloon optical payload.

Potential benefits of RCWS:
• Simplicity in design - One CMOS vs. many microlenslets.
• Optics systems generally have a system for changing the focal length, can therefore use the main image detector.
• The RCWS method has the potential to perform equally or even better than the currently used methods on aerial platforms.
• Future missions could then choose SHA or RCWS systems based on performance data
1.2 - Objectives

- Quantitatively compare the SHA and RCWS wavefront sensors as a function of source intensity
- Develop a prototype Roddier sensor to be used in the comparison
- Design and build a test platform that facilitates data collection with required precision and accuracy
- Develop forward-predictive models to drive the design and validate results
- Present preliminary results
1.3.1 – CONOPS Part 1
1.3.2 – CONOPS Part 2

**Method 1 – Error**
Measurement with SHA*

**Diverging**
**Converging**

**Focus**

*Note: No collimating lens is needed here, correction applied in software.*

**Method 2 – Error**
Measurement with RCWS*

**Image 1**

**Focus**

**Image Sensor**

*~1mm* aft of focus

**Image 2**

**Image Sensor**

*~1mm* fore of focus

**Focus**
1.4 – Functional Block Diagram (FBD)
1.4.1 – FBD Part 1
1.4.2 – FBD Part 2

Computing Unit

Result Calculation
- Amplitude Calculation
- Rate Calculation
- Data Formatting

Raw Data Analysis
- SHA* Interpreting Algorithm
- RCWS* Interpreting Algorithm
- Optical Simulation

Image Data
Optical Computational Model

User
Amplitude
Results
Measured Error
Rate

1.5 – Baseline Design

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image Source</td>
<td>Provide known conditioned state at the input to the system</td>
</tr>
<tr>
<td>Optical System</td>
<td>Introduce wavefront error and focus image to sensors</td>
</tr>
<tr>
<td>Shack-Hartmann Array</td>
<td>Test Article #1</td>
</tr>
<tr>
<td>Roddier Curvature Wavefront Sensor</td>
<td>Test Article #2</td>
</tr>
<tr>
<td>Testbed</td>
<td>Align, isolate, and protect optical components</td>
</tr>
<tr>
<td>Environmental Sensor System</td>
<td>Track environmental changes</td>
</tr>
<tr>
<td>RCWS Algorithm</td>
<td>Compute RCWS Zernike amplitudes from RCWS data</td>
</tr>
<tr>
<td>Test Control Software</td>
<td>Automate test procedure and perform data handling</td>
</tr>
</tbody>
</table>

Changes since CDR:

- Alignment stages are purchased rather than manufactured due to required precision
- Range of defocus distances for RCWS significantly reduced because of results from predicted images
- Shroud designed to reduce stray light and air movement in the optical path
1.5.1 – Baseline Design
1.6 – Critical Manufacturing Elements

<table>
<thead>
<tr>
<th>Region</th>
<th>CPE</th>
<th>Description of CPE Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>Documentation</td>
<td>Test Procedure</td>
<td>Ensures testing is complete and proceeds smoothly</td>
</tr>
<tr>
<td></td>
<td>Equipment Handling Procedure</td>
<td>Avoid costly mistakes and damage to equipment</td>
</tr>
<tr>
<td></td>
<td>Data Format Specification</td>
<td>Drives software development and integration</td>
</tr>
<tr>
<td>Software</td>
<td>Wavefront Reconstruction</td>
<td>Meets requirement to develop RCWS system</td>
</tr>
<tr>
<td></td>
<td>Manual Control</td>
<td>Minimum success of project</td>
</tr>
<tr>
<td></td>
<td>Test Control</td>
<td>Allows for much larger data set</td>
</tr>
<tr>
<td></td>
<td>Teensy Program</td>
<td>Meets environmental sensing requirements</td>
</tr>
<tr>
<td>Hardware</td>
<td>Light Enclosure</td>
<td>Reduces light and air movement effects on results</td>
</tr>
<tr>
<td></td>
<td>Image Source</td>
<td>Provides the test image</td>
</tr>
<tr>
<td></td>
<td>Mirror Mounts</td>
<td>Complicated part to manufacture, must hold mirrors</td>
</tr>
<tr>
<td></td>
<td>Environmental HW</td>
<td>Meets environmental sensing requirements</td>
</tr>
</tbody>
</table>
2.0 – SCHEDULE

2.1 – Schedule Overview
2.2 – Manufacturing Plan
2.3 – Testing Plan
Integration and testing as well as the final experiment cannot proceed without manufacturing and documentation complete.
2.2 – Manufacturing Plan

March 2\textsuperscript{nd} : Manufacturing complete.

Hardware and software development occurs independently.

Feb 19\textsuperscript{th} : Independent software aspects combined into main driver
2.3 – Testing Plan

March 12th: Integration and characterization on optical table.

March 26th: Experiment begins.

April 2nd: Need for re-test evaluated from data.

April 9th: Margin for re-test and data analysis.

April 20th: Additional margin for course requirements.
3.1 – Documentation
3.2 – Software
3.3 – Hardware

3.0 – MANUFACTURING
3.1.1 – Equipment Handling Procedures
3.1.2 – Testing Procedures
3.1.3 – Data Format Standardization

3.1 – DOCUMENTATION
3.1.1 – Equipment Handling Procedures

Component Handling Guidelines:
• Details procedures to be used when transporting/installing each component
• General guidelines for handling, storing, and cleaning optical components
• Specific details for working in the SwRI lab

Component Location Schedule and Checklist Document:
• Tracks when and where components are moved between CU and SwRI
• Eliminates resource conflicts
• Team member sign off sheet for each critical step of transport, installation, and testing

Drafts of both items complete, to be finalized by Feb. 16th for integration
3.1.2 – Testing Procedures

A preliminary Experimental Procedure has been created. Describes the generalized flow of the experimental process to help direct the development of the interfaces and automation.

- Detailed experimental procedure stepping any user through the experimental process
  - Dependent on user-computer interface, which is still under development.

- Verification and Validation testing procedures
  - Detailing step-by-step processes for each of the verification and validation tests (six tests).
  - Not necessarily automated.

Test procedures shall be completed by March 5th.
A specification of data formats is necessary to unify individual SW development and manage resulting data for post-processing.

The specification is finalized, allowing software development to continue and integrate smoothly.
3.2 – SOFTWARE

3.2.1 – Wavefront Reconstruction
3.2.2 – Manual Control
3.2.3 – Test Control
3.2.4 – Teensy Program
3.2.1 – Wavefront Reconstruction

\[
\frac{I_1(\vec{r}) - I_2(-\vec{r})}{I_1(\vec{r}) + I_2(\vec{r})} = \frac{f(f-l)}{2l} \left[ \nabla^2 z(\vec{u}) - \frac{\partial}{\partial u} z(\vec{u}) \delta_c \right] u = f\vec{r}/l
\]

Wavefront reconstruction is a critical portion of the RCWS package. The mathematical solution and software implementation may be the limiting factor in RCWS performance.

**Current Issues:**

- Few solutions are published for reference
- Some examples such as Large Synoptic Survey Telescope code appear to be specialized, and with limited documentation
- The team’s lack of optics experience is limiting the efficacy in evaluating the problem, specifically as to whether or not Airy disk patterns will mask the required structures
3.2.1 – Wavefront Reconstruction

Next Steps:

• Blob detection software can be used to determine first 6 Zernike modes, already implemented for modes 1-4.
• A sub-team is dedicated to studying the mathematical concepts in order to understand and solve the Poisson equation.

Expect to spend February and 2 weeks of March to solve this.

Fortunately, this problem is isolated, thus, does not hinder progress in the rest of the project. However, the wavefront reconstruction is critical for a fair evaluation of the performance of the RCWS.
3.2.2 – Manual Control

The experiment can be completed without automated test control as long as the provided software for various hardware items works. This has been verified for:

- ThorLabs WFS150-7AR with the ThorLabs-provided software
- QHY174-M and the ASI120MM detectors using COTS software
- ThorLabs motorized optical stages utilizing the ThorLabs-provided software

Proves that all required data can be collected manually. Test control software will allow for faster execution and increased reliability of results.
3.2.3 – Test Control

The test control software is designed to automate the collection of 500+ data points. Information determined at each system state includes:

- SHA wavefront measurement
- RCWS fore and aft-focus images
- Environmental sensor data
- Predicted wavefront and images
3.2.3.1 – Automation APIs

Outlined boxes are where programs with APIs will be utilized for automation.
3.2.3.1 – Automation API Status

**Purpose:** The automatic image capture software is to simplify the collection of many iterations of data through computer control of both the image sensors and motorized stages because we have limited time in the optics lab to test.

**Current State:** Currently becoming familiar with first- and third-party drivers for the image sensors and motorized stages, as these will be the primary method of interfacing with the hardware and will allow for automation of testing.

**Next Step:** Experimentation with driver functions and creation of rudimentary control programs to validate the functionality on the test hardware.

**Biggest Challenge:** Debugging the initial problems with utilizing the OS-specific drivers, as there seems to be a very specific set of steps to connect to the hardware.

### Automatic Image Capture Progress

- **CMOS Image Capture Automation:** The manufacturer of the CMOS image sensors don’t directly provide drivers, so we are reliant on 3rd party drivers to automate the image capture and exposure time. 15%
- **SHA Image Sensor Automation:** The SHA is from ThorLabs, so they provide drivers to support user-developed applications, along with explanations of the provided functions, which will aid development. 20%
- **Motorized Stage Automation:** The motorized stages are also from ThorLabs which come with their own 1st party drivers. This aids with end-user application development. 20%

**Estimated Time Remaining:** 50 hours, based on the amount of hardware which needs to be automated.
3.2.3.2 – Environmental Sensor Data

Outlined boxes display where the environmental sensor data requires programming.
3.2.3.2 – Environmental Sensor Data Status

**Purpose:** This software needs to be written in order to transmit the environmental sensor data from the Teensy to the test computer over serial.

**Current State:** Currently specified data format in which the data will be transmitted for storage (proof of concept was completed last semester).

**Next Step:** Create an emulator which generates dummy data in the correct format in order to validate data storage so we can move on to writing the sensor-reading code.

**Biggest Challenge:** Implementing data streaming at the same time as other control functions.

---

**Teensy Communication Software**

**Teensy Communication:** We proved last semester that we could stream data to a computer from the Teensy over serial, as well as sample the environmental sensors at the acceptable rates.

**Estimated Time Remaining:** 10 hours, since the group members already have experience with Teensy serial communication and reading sensors.
Purpose: The optical model in Zemax will be polled for each system state to determine expected wavefront shape. This will validate the results of the wavefront sensors and further help determine accuracy of the RCWS method.

Current State: Python API into Zemax has verified key access and the access the optical path specification file.

Next Step: Programmatically change values in lens file, then save expected Zernike amplitudes to text file.

Biggest Challenge: Changing the desired data point in the lens file programmatically.
3.2.4 – Environmental Sensor Drivers

Programming Scope:

- Samples from 12 sensors at upwards of 1 kHz
- Data from 12 sensors is time-stamped and written to serial
- Central program handles PC-side serial monitoring / data collection and data storage.

Status:

- Serial monitoring has been tested on the Teensy, PC
- SPI transfers have been tested, need to be scaled for 6 sensors
- Code mostly requires re-tooling, possible GUI addition at this point

Estimated Time Remaining: 8 hours for scaling existing code to 12 sensors and incorporating communication standards.
3.3 – HARDWARE

3.3.1 – Light Enclosure
3.3.2 – Image Source
3.3.3 – Mirror Mounts
3.3.4 – Environmental Sensors
3.3.1 – Light Enclosure

**Purpose:** Reduces air movement and incident light
- Inside painted black to absorb reflected light
- Interlocking design blocks direct light entry
- Black fabric will cover silver optical table surface
- Minimal volume reduces air movement

**Status:**
- Construction complete
- Still needs to have sealing ring installed on optical table
**Purpose:** Source directs light into optical fiber to be diffused before pinhole

**Scope:**
- Mechanical and electronic components to be purchased
- Components require minimal modification

Manufactured by team
In Progress Components:
- Emitter electrical connections and mounts
- Soldering and simple mounting with PCB standoffs
- Testing individual components for out-of-box functionality
  - AC-DC converter works as expected (nominal 12V output)
  - DC-DC booster to be tested (variable voltage and current output)

Next Steps:
- Mechanical rail mounting system, ~3 hrs machining
- Emitter Mount, ~4 hours machining
- Alignment, ~3 hrs
- Secondary shroud, ~2 hrs
Unique notched design of mirror mounts allows for close spacing of beams per optical path specs.

- Stock has been ordered from online metals
- Expected machine time: ~16 hrs for 2 mounts

Other mounts to manufacture:
- Pellicle Mount: ~6 hrs
- CMOS Camera Mount: ~4 hrs
3.3.4 – Environmental Sensors

**Parts Not Yet Acquired:**
- Sensor Circuit Boards + Teensy Connector Board
- Temperature caps (manufactured)
- Adhesive Pads

**Parts Acquired:**
- ICs (OR gates, ADXL345, ADT7320)
- Connectors
- Capacitors
- Teensy

- The ICs, connectors, capacitors, and the Teensy will all be soldered to the designed PCBs
- Temperature caps are non-crucial, and will be 3D-printed for the temperature sensors
- Adhesive pads will be used for mounting the accelerometers
3.3.4 – Environmental Sensors

Circuit Board Status:

• Design completed / reviewed
• PCB order placed January 25th, expected February 5th
• Tasks remaining: populating PCBs, make cables for sensors
  – Estimate: 10-12 hours for soldering, error-checking
4.0 – BUDGET

4.1 – Budget Overview
4.2 – Ordering Status
4.1 – Budget Overview

Class + EEF Funding:
- 54.4% spent, 63.6% expected use

Equipment on Loan:
- ThorLabs Shack-Hartmann Array
- QHY174M detector
- ThorLabs motorized stages
- Lab space
4.2 – Ordering Status

Planned:
• All planned items have been ordered

Pending:
• ThorLabs KDC101 Motor Controller (1 of 3)
• Gloves
• Masks
• LED heatsink
• OR gates
• Aluminum rod
• Fasteners
• Powdercoat

All other components have been received. Lead times not expected for any of above items.
QUESTIONS?
Test Control FBD

1. Wavefront Reconstruction
2. Motor Control
3. Central Program:
   - Automation of connected peripherals
   - Parsing input data (images, zernikes, env.)
   - Data storage and saving / file creation
4. CMOS Data Capture
5. SHA Data Capture
6. Test Info
7. Desired Positioning
   - Actual Positioning
   - Motors / Stages
1. User specifies a “schedule” of states to test
2. Test control program reads the test specification and creates timestamped sub-folders for each state of the system.
3. Each sub-folder can stand alone with required state information, raw data from sensors, predicted results, and reconstructed wavefronts.
Environmental Sensors – Timing Diagram

1 Cycle, $\Delta t = 1\text{ms}$

Actual System Cycle:

- Read 6 Bytes of Accelerometer Data (x6)
- Read 2 Bytes of Temperature Data (x6)
- Computer reads approximately 48 bytes from Serial connection with Teensy
- Margin

Previously Tested / Simulated Cycle:

1 Cycle, $\Delta t = 1\text{ms}$

- Read 6 Bytes of Accelerometer Data (x6)
- Read 6 Bytes of Accelerometer Data (x6)
- Computer reads approximately 48 bytes from Serial connection with Teensy
- Sensor Switching
- Margin

- 0.078ms
- 0.078ms
- 0.044ms
- 0.0061ms
- 0.794ms
Environmental Sensors – Data Rate Testing

- Serial Read Data Rates were constructed using Python with pyserial, Windows 10, and a Teensy 3.6.
- For Data checking, the Teensy had been set to count from 0 to 99 endlessly while streaming the data over serial. The data was detected to have zero errors.
- Tests were run with varying time intervals of 1, 5, 8, 10, 12, 20, 30, 40, 50, and 60s

<table>
<thead>
<tr>
<th>Serial Testing Results</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Average Data Rate</strong>: 1086.041 kbps</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Generic SPI Device Testing Results</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Average Switching Time</strong>: 0.5104 μs</td>
</tr>
</tbody>
</table>

- SPI testing was completed with two generic accelerometers, testing read time and the time required to switch between sensors
## Component Choices:
- Microcontroller: Teensy 3.6
- PC Interface: Serial USB connection
- Accelerometer: ADXL 344
- Temperature Sensor: ADT7320
- [Overall Schematic](#)

### ADXL344 Requirements and Performance

| Requirement            | Sampling Rate | SPI Data Rate | Resolution | Filtering?
|-------------------------|---------------|---------------|------------|-------------
| ADXL-344 Performance    | 3.2 kHz (Max) | 5 MHz (≈ 5 Mbps) | ±3.9 mg    | Yes         |

### ADT7320 Requirements and Performance

| Requirement | Sampling Rate | SPI Data Rate | Resolution (16 bit/13 bit) [°C] | Accuracy [°C] | Filtering?
|-------------|---------------|---------------|----------------------------------|---------------|-------------
| Requirement:| 1 Hz          | 0.55 MHz      | ±0.15                            | ±0.5          | No          |
| ADT7320     | 4 Hz (Max)    | 5 MHz (≈ 5 Mbps) | ±0.0078 (±0.0625) | ±0.31         | No          |
Wavefront Reconstruction

\[
\frac{I_1(\mathbf{r}) - I_2(-\mathbf{r})}{I_1(\mathbf{r}) + I_2(\mathbf{r})} = \frac{f(f-l)}{2l} \left[ \nabla^2 z(\mathbf{u}) - \frac{\partial}{\partial u} z(\mathbf{u}) \delta_c \right]_u = f \tau/l
\]

This is the data collected by the test setup.

The wavefront \(z\) is the structure of interest that should be solved for.

- This expression leaves out terms due to diffraction because of assumptions made in the original paper.
- To obtain a general solution this relationship would need to be re-derived including the diffraction terms.
- At that point a forward-model would exist, but a simple solution for \(z\) is not guaranteed.
- A numerical solution could likely be found, however the work required exceeds the capabilities of the team.