ARL-TR-7506 ● Oct 2015

US Army Research Laboratory

The Automation of the Transonic Experimental Facility (TEF) and the Aerodynamic Experimental Facility (AEF)

by Charith R Ranawake

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

ARL-TR-7506 ● OCT 2015

US Army Research Laboratory

The Automation of the Transonic Experimental Facility (TEF) and the Aerodynamic Experimental Facility (AEF)

by Charith R Ranawake
Weapons and Materials Research Directorate, ARL

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED (From - To)
October 2015	Final	05/2015–08/2015

4. TITLE AND SUBTITLE
The Automation of the Transonic Experimental Facility (TEF) and the Aerodynamic Experimental Facility (AEF)

5a. CONTRACT NUMBER
W911NF-10-2-0076

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Charith R Ranawake

5d. PROJECT NUMBER
AH80

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
US Army Research Laboratory
ATTN: RDRL-WML-E
Aberdeen Proving Ground, MD 21005

8. PERFORMING ORGANIZATION REPORT NUMBER
ARL-TR-7506

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT
The Aerodynamic and Transonic Experimental Facilities at the US Army Research Laboratory (ARL) maintain free-flight spark ranges to evaluate the aerodynamic and flight dynamic performances of gun-launched projectiles. Currently, firing a projectile and capturing its shadowgraph image on film is a manual process—which is tedious and limiting when conducting multiple tests. This project addressed the automation of the free-flight spark ranges by enhancing ARL's software for remotely controlling the operation of the ranges and integrating digital cameras. Initially, modifications were made to the existing FilmScanner program to make it compatible with Vidar scanners that do not utilize the drivers developed using the TWAIN communications protocol. Then, a prototype system consisting of a range interface box, Weather Station, and digital camera connected via Ethernet to a computer was built to serve as a development platform. The existing RangeController program was enhanced to control the prototype system. The enhancements include adding functionality to remotely take weather measurements from the Weather Station, control the camera settings, operate the camera using an external trigger, and record the camera images after the shot. These enhancements will help significantly reduce human intervention, which will decrease the required range time for testing.

15. SUBJECT TERMS
TEF, AEF, range digitization

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Sidra Silton
Unclassified	Unclassified	Unclassified	UU	30	19b. TELEPHONE NUMBER (Include area code) 410-306-0792

Contents

List of Figures

Acknowledgments

The author would like to thank the following individuals for their contributions and support during this project:

- Dr Sidra I Silton for her guidance, support, and vast knowledge about this project.
- Mr Ilmars Celmins for his contributions, attentiveness to details, and assistance with this project.
- Mr Kenneth Paxton for laboriously wiring up the devices in the range so the RangeController Program could be tested.

INTENTIONALLY LEFT BLANK.

Student Biography

Charith Ranawake is a sophomore at the University of Maryland, College Park, in the Business, Society, and Economy Scholars Program. He is a computer science major but plans to double major in business management. His past research experience includes working as a volunteer programmer for a professor at George Mason University, where he helped develop a computer-security software packaged; also, completing a Science and Engineering Apprentice Program internship in which he extensively tested BRL–CAD's raytracer, a solid-geometry modeling system, on the primitive Bag of Triangles. After college, Charith plans on working with both cybersecurity and operating systems.

INTENTIONALLY LEFT BLANK.

1. Background

The US Army Research Laboratory (ARL) at Aberdeen Proving Ground, Maryland, currently operates 2 free-flight spark ranges—one each at the Transonic Experimental Facility (TEF) and Aerodynamic Experimental Facility (AEF)—that are used to obtain aerodynamic and flight-dynamic performance information about projectiles to benefit research and development for the Army. These ranges produce dual, orthogonal shadowgraph images of a projectile at multiple positions as it travels downrange. From the shadowgraph images, measurements of the actual flight motion can be obtained and analyzed to determine the aerodynamic and flight-dynamic behavior of the projectile.[1] Figures 1 and 2 display how the TEF and AEF spark ranges, respectively, are currently set up—each station set a certain distance away from another, prepared to capture the shadowgraph image. Despite the fact the 2 ranges look dissimilar due to their varying sizes, they use the same basic devices to acquire the shadowgraph images of the projectile. These devices include an infrared (IR) screen, a spark box, and 2 films at each station. The IR screen is a sensor that sends a signal to the range interface box, which controls the spark box. The spark box will then release a short-duration, high-intensity spark after a predefined delay. From this spark, a pair of orthogonal shadowgraph images are captured by the films. In AEF, a single spark is split by a mirror and the projectile's shadow is captured directly on the film in each plane. In TEF, due to the size of the range, 2 cameras and 2 spark boxes (one on the wall and one in the floor) are used to take a picture of screens mounted on the opposite wall and ceiling that show the projectile shadow when the spark flashes. Because a camera is used to take a picture of the screen, each film contains 2 images of the projectile, the actual projectile (fuzzy) and its shadow (clear), as shown in Fig. 3.

Fig. 1 Transonic Experimental Facility's spark range

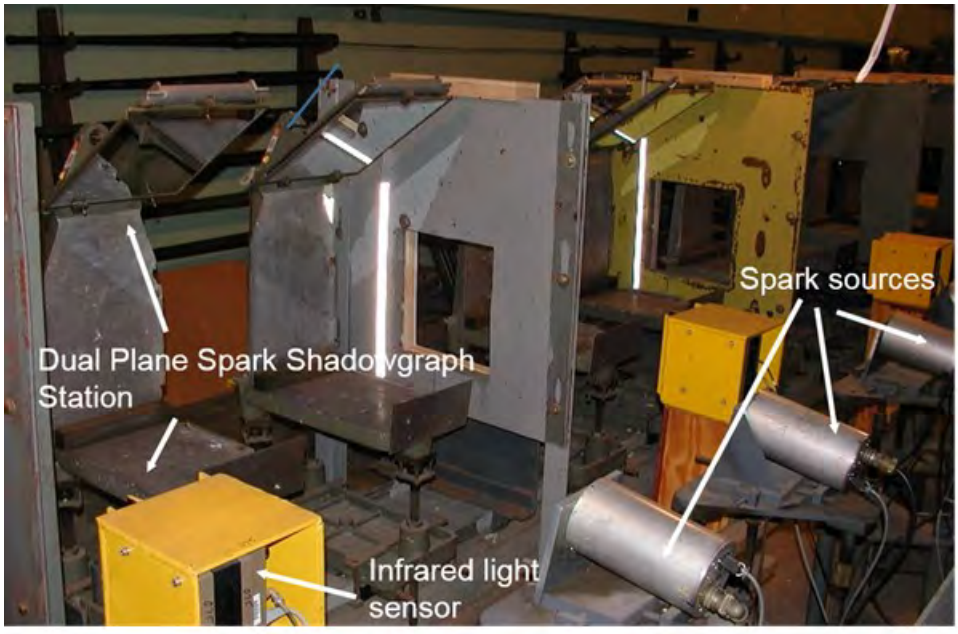

Fig. 2 Aerodynamic Experimental Facility's spark range

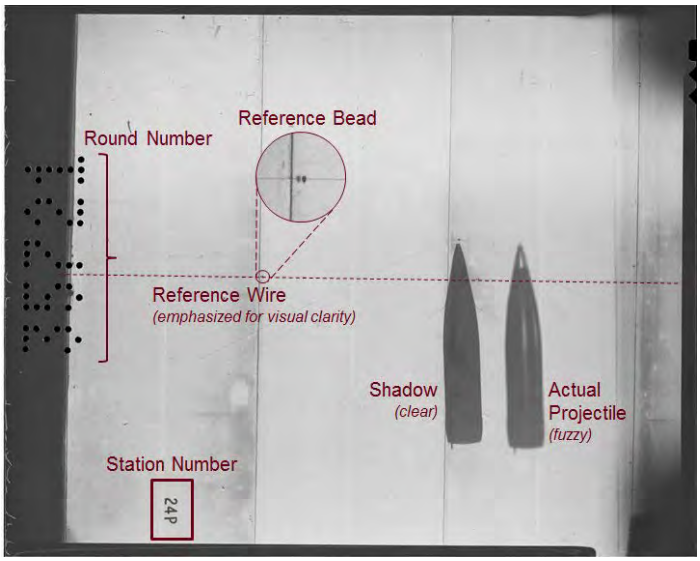

Fig. 3 Shadowgraph of gun-launched projectile in TEF

The current procedure for conducting multiple shots for a single test is extremely inefficient and burdensome due to its manual and labor-intensive process. The inefficiency can be observed by reviewing the basic procedures in the AEF where, once a test is in full swing, it takes 30–45 minutes—at best—to conduct a single shot. The basic procedure for each shot is

1) Turn lights off in range
2) Load film at each station
3) Fire shot
4) Sparks triggered as shot passes IR Screen
5) Shadowgraph image captured on film
6) Retrieve films from each station
7) Develop films

After completing these steps for each shot, another day is required to run the 3 Spark Range Film Package programs: FilmScanner, FilmSorter, and FilmReader.

2. Spark-Range Film Package

2.1 Overview

The Spark-Range Film Package,[2] comprising 3 programs, allows the user to obtain usable data for analysis of the projectile of interest from the films. The first step is to digitize the films obtained from the free-flight spark range for each shot using the FilmScanner program. After all of the films have been digitized, the FilmSorter program sorts and organizes the scanned images based on the station number. This program also allows the user to perform basic image enhancements in case the shot did not come out clearly on the film. The last program in the Spark-Range Film Package is the FilmReader program. After running the first 2 programs and receiving the respective outputs, the FilmReader program takes measurements of the projectile shadow and reference locations from each scanned image. The output of the FilmReader program is used as input for the aerodynamic analysis.[3]

To continue the process of completely automating the ranges inside the TEF and AEF, modifications and improvements were made to the FilmScanner program.

2.2 FilmScanner Program

Prior to modifying the program, the FilmScanner program was only compatible with Vidar scanners that utilize the drivers developed by Vidar using the standardized TWAIN applications programming interface and communications protocol. The main modification to the FilmScanner Program was to make the existing user interface compatible with the Vidar scanners that do not use the TWAIN drivers.

This modification was implemented by using a C++ Dynamic Linked Library (DLL) that the Vidar Company supplied with its scanner. The Vidar DLL is composed of several functions that help perform the basic scanning operations, such as accepting or ejecting the film. This DLL needed to be called from the FilmScanner's graphic user interface (GUI) in order to interface with the scanner when the TWAIN drivers were not used. However, it was not possible to call the Vidar DLL directly from the existing GUI due to the memory issues that occurred. These memory issues occurred with several of the parameters in the Vidar DLL functions containing data structures because the 2 programs were written in different programming languages: Visual Basic (GUI) and C++ (DLL). The elements encapsulated in the data structures in the Vidar DLL contain data types that are unsupported in Visual Basic, so it was impossible to populate the data structures in the Visual Basic GUI. The solution to this problem was to create a

"wrapper" DLL in C++ that would be called from the Visual Basic GUI. Only the crucial parameters from the user input, which would not conflict with any data types between the 2 languages, would be passed into the "wrapper" DLL. The rest of the parameters, which are immutable, were hard-coded in the "wrapper" DLL in order to fully populate each data structure. The Vidar DLL functions were then called from the "wrapper" DLL using the data structures just created. Most of the calls for each function of the "wrapper" DLL are event driven, meaning each function is called by entering a value in the textbox, selecting an option from the combo box, or pressing a button on the form. (This solution is outlined in Fig. 4.) By taking all of these steps, the memory issues associated with passing data structures with unsupported data types became nonexistent.

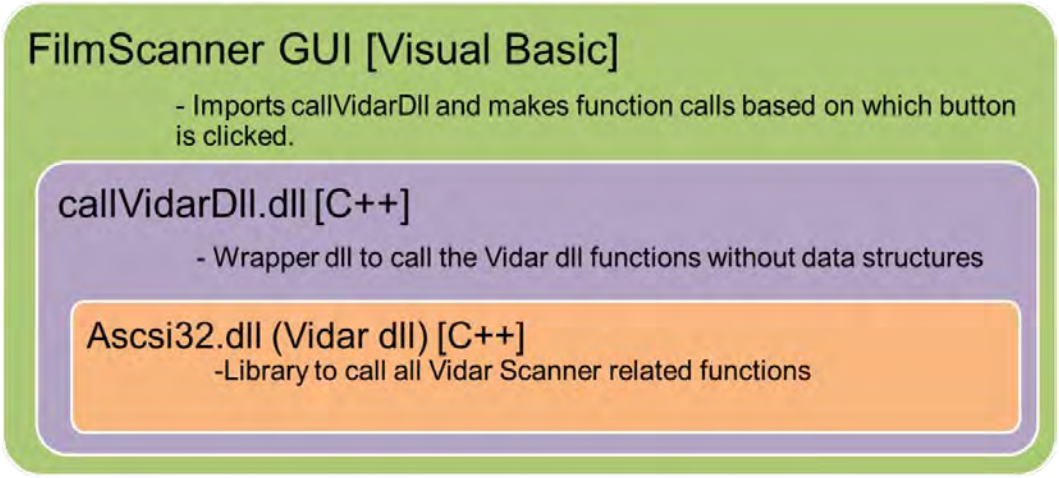

Fig. 4 FilmScanner program's hierarchy

Figure 5 displays the original FilmScanner GUI, while Fig. 6 shows the updated version. Basic design modifications were made to properly incorporate the Vidar Scanners that do not utilize the TWAIN drivers as well as to make the user interface more efficient.

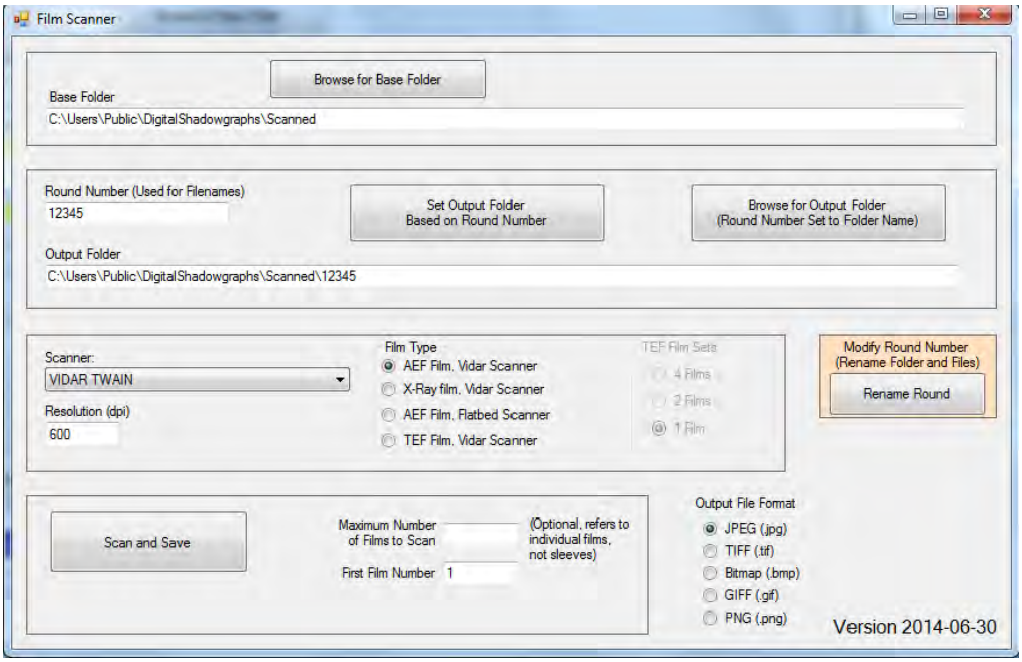

Fig. 5 FilmScanner GUI before modifications

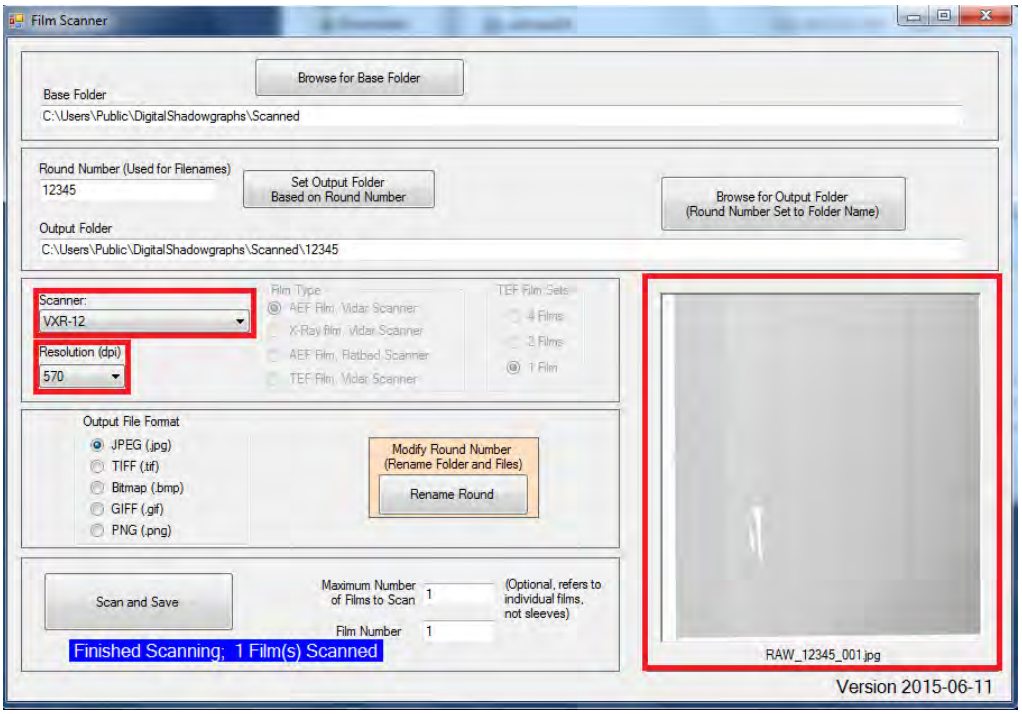

Fig. 6 FilmScanner GUI after modifications

The first change made to the FilmScanner GUI was the addition of the “VXR-12” scanner to the program, which can be seen in the top–left red box in Fig. 6. This Vidar scanner, which does not use the TWAIN drivers, is actively being used by

ARL to digitize the films. (The functionality of the scanner was added to the FilmScanner program because the old scanner that used the TWAIN drivers broke.)

The next change to this program was the replacement of the resolution textbox with a combo box, which is in the red box underneath the Scanner combo box in Fig. 6. User input from a textbox has a higher rate for producing several errors in this program. The most common issue that can occur is entering a value for the resolution that is not compatible with the scanner. To fix this problem, whenever the user selects a scanner, the program communicates with the scanner to determine which resolutions it supports. The program stores these resolutions in a combo box that the user can easily select. This functionality of receiving supported resolutions currently works with either scanner, regardless of whether or not it uses the TWAIN drivers.

The last addition to the FilmScanner GUI was a picture box (bottom–right corner of Fig. 6). After a film has been scanned, the image is directly uploaded to the picture box, allowing the user to view it. The picture box continually updates itself after each film has been scanned, if there are multiple films in the tray. This feature permits the user to quickly check whether the scanned image came out properly instead of having to open each image after the program finished scanning the films.

3. RangeController Program

Although modifications and improvements were made to the FilmScanner Program, the ARL would like to increase the automation and efficiency of the free-flight spark ranges. This involves incorporating digital cameras in the ranges and into the Range Controller program to take digital images of the projectile shadowgraphs. The addition of the digital cameras would make the FilmScanner and FilmSorter program obsolete. The digital camera the ARL's Flight Sciences Branch is currently testing is an 8MP Gigabit Ethernet Camera made by Imperx. The criteria for testing the feasibility of using the cameras on the ranges include the following: ability to adequately acquire the image of the projectile's shadowgraph, ease by which the cameras can be controlled programmatically, and price.

The RangeController program is a Visual Basic.NET program that was written to control all the devices on the range.[4] When received from ArrowTech Associates, the RangeController program only had functionality to control the range interface box, a device that is used to control the IR screens and spark boxes at each station. The functionality added to the RangeController program as part of this project was the ability to control the digital cameras and a Weather Station. The cameras will replace the film on the ranges to capture the projectile's shadowgraph image, while the Weather Station will record the most current temperature, pressure, and

humidity and automatically store the information. With the addition of this functionality, the goal of fully automating the ranges to minimize human involvement and improve efficiency will soon become a reality.

Figure 7 is an image of all of the range devices controlled directly by the Range Controller program connected to an Ethernet switch; the Ethernet switch is also connected to a computer. Connecting all of the devices to the same computer via the Ethernet port posed a problem that required an unconventional solution. Since all of the devices were on different subnets, none of them would be in the same range as the computer. Thus, the computer would not be able to detect any of the devices. The solution to this problem was to assign multiple static internet protocol (IP) addresses to the computer to match each respective device. Adding multiple static IP addresses to the computer allowed all of the devices to be in the same range, eliminating most of the connectivity issues. Although most of the connectivity issues were fixed, a Socket Exception persisted: "No connection could be made because the target machine actively refused it." This connectivity issue caused the devices to either lose their connection to the computer after a certain period of time or to not even allow the computer to connect to the device. The Socket Exception issue came from an incorrect setting for the firewall on the ARL disk image of the stand-alone computer. The firewall setting was set to refuse some inbound connections through the Ethernet port, which explains why most of the attempts to connect to the devices were unsuccessful. Once the firewall setting was adjusted to allow inbound connections through the Ethernet port, the issue was resolved.

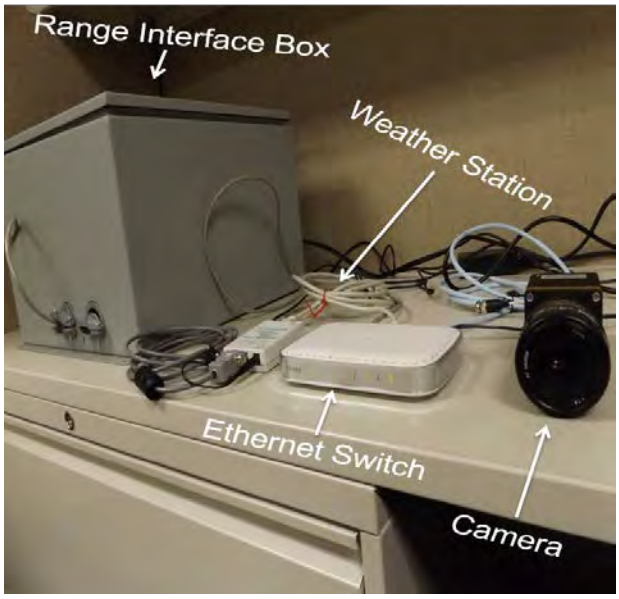

Fig. 7 Setup of devices in cubicle

Figures 8–15 are screenshots of the modified RangeController GUI containing the functionality to control the range interface box, the Weather Station, and the digital cameras.

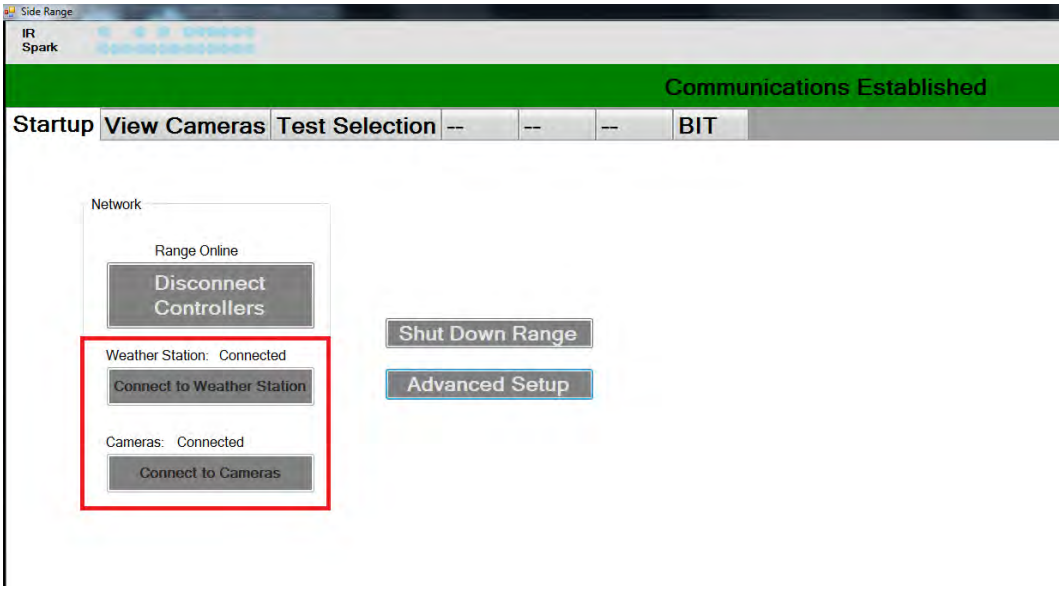

Fig. 8 RangeController program's startup tab with Weather Station connected

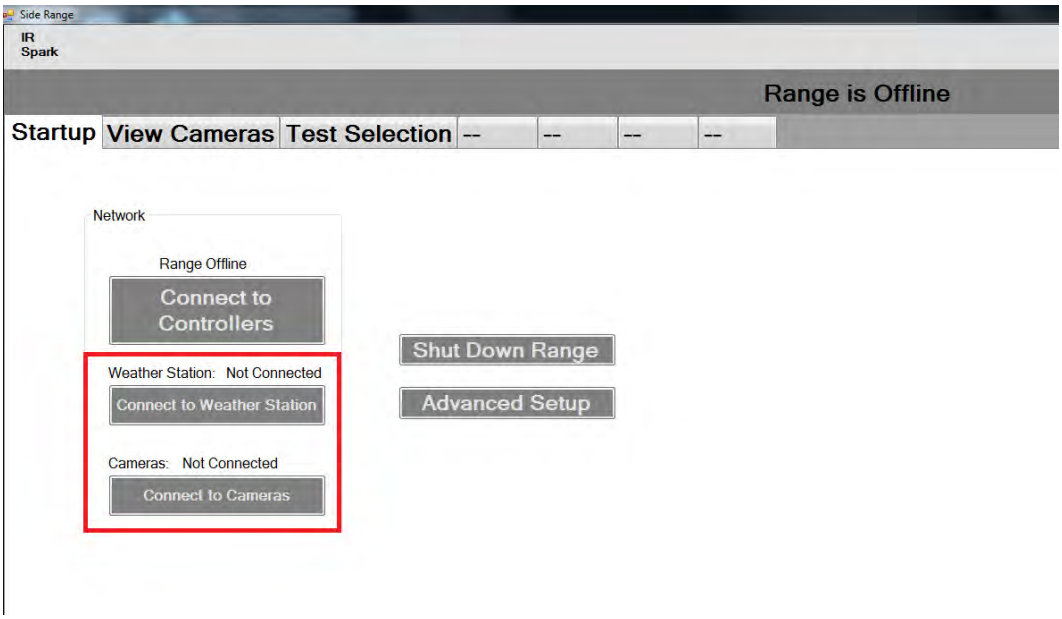

Fig. 9 RangeController program's startup tab with Weather Station not connected

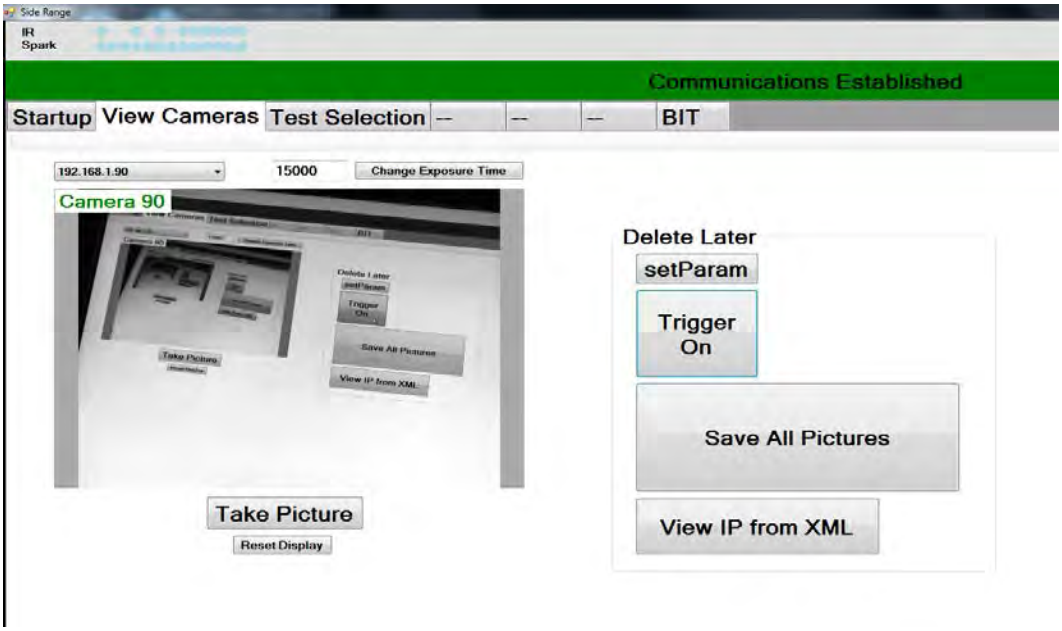

Fig. 10 RangeController program's view-cameras tab

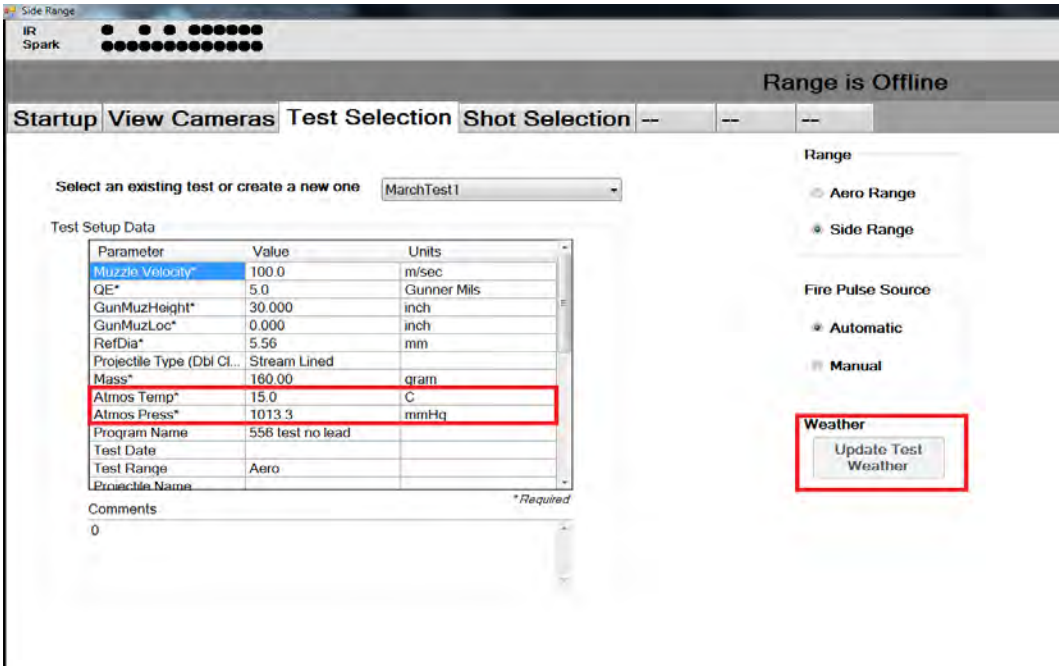

Fig. 11 RangeController program's test-selection tab with Weather Station not connected

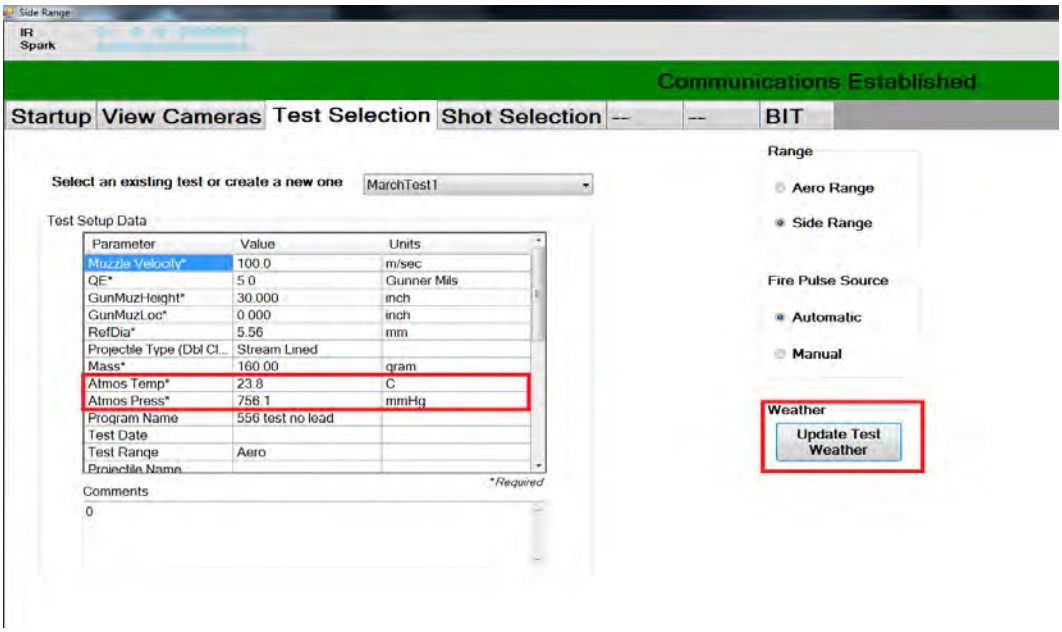

Fig. 12 RangeController program's test-selection tab with Weather Station connected

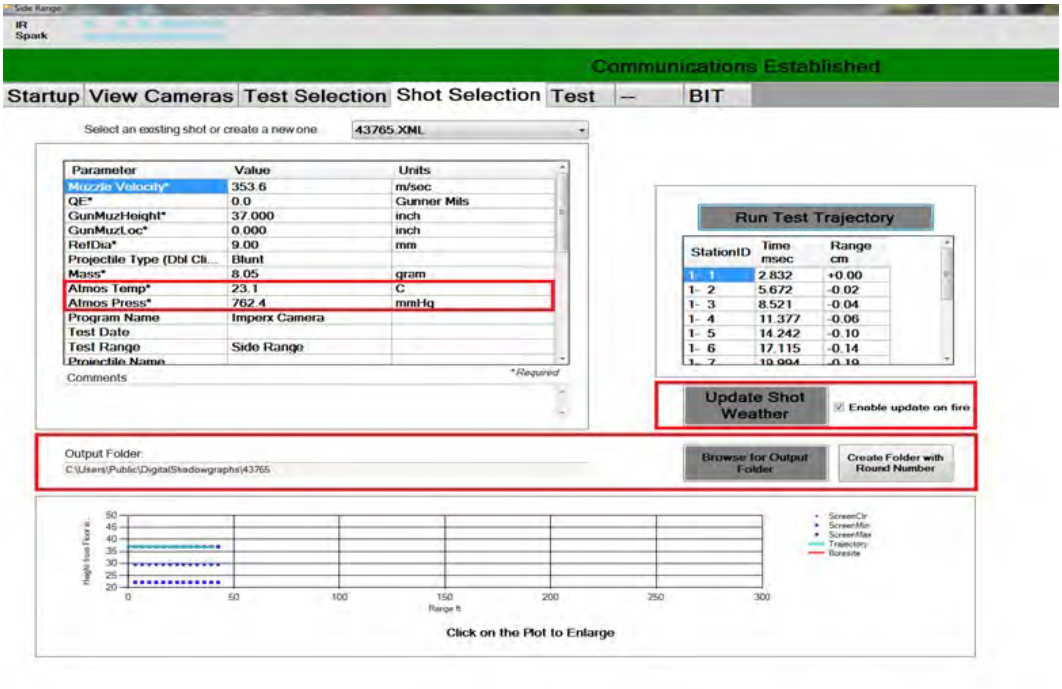

Fig. 13 RangeController program's shot-selection tab with Weather Station connected

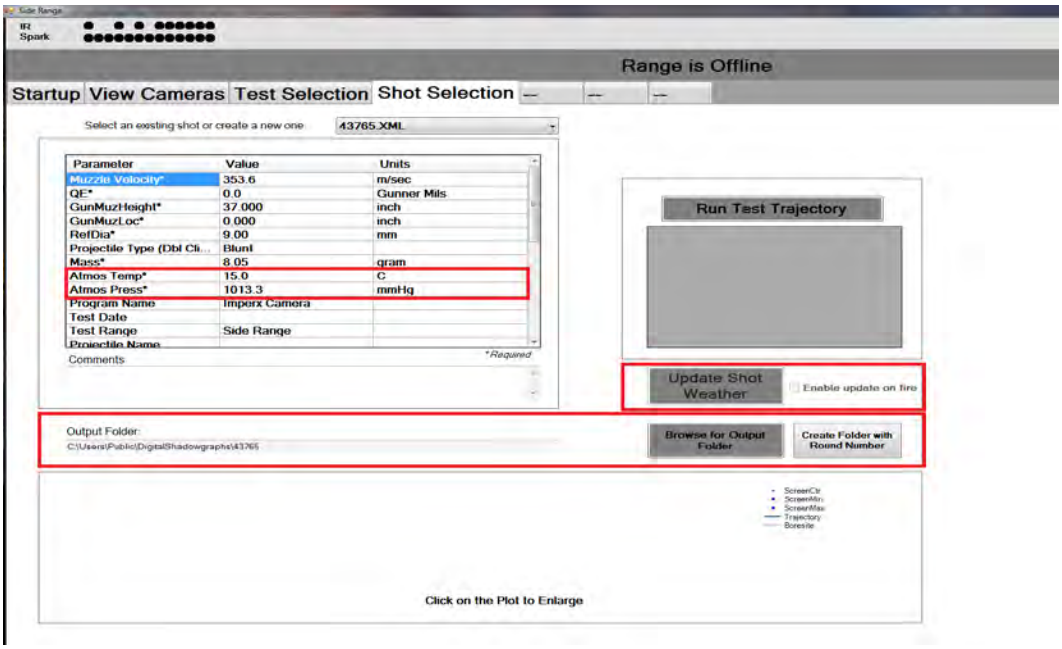

Fig. 14 RangeController program's shot-selection tab with Weather Station not connected

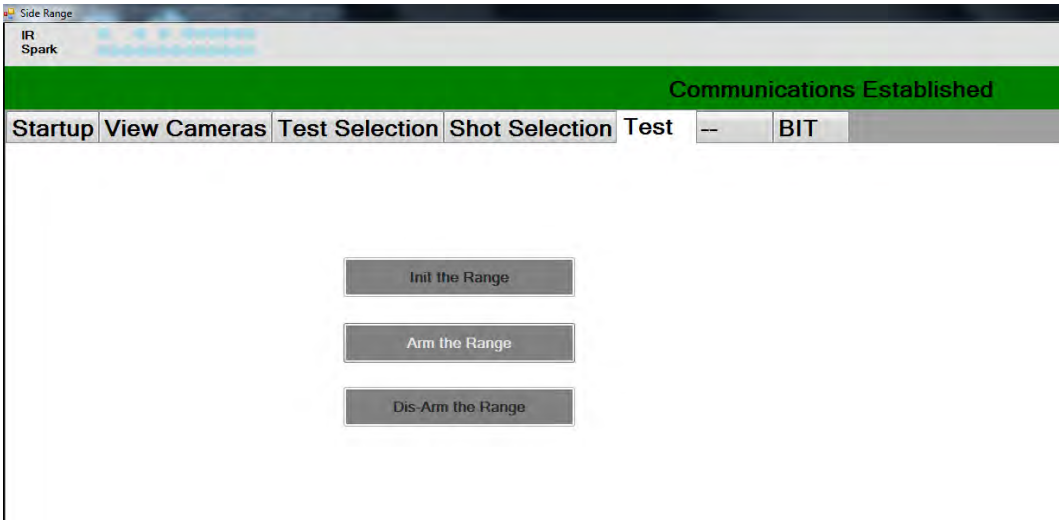

Fig. 15 RangeController program's test tab

The RangeController program, when it is first run, will look like either Fig. 8 or Fig. 9, depending on whether a device is connected to the computer or not. The Startup tab serves the purpose of connecting to the devices, disconnecting the devices, and setting up information about the range. The first change made to this tab of the GUI was to automatically connect to the Weather Station and digital cameras on the Form Load. If the status of connectivity was completely successful, the GUI would look like Fig. 8. The labels above the respective button for the devices will be changed to "Connected", while blackening out the "Connect to…"

buttons. The purpose of blackening out these buttons is to disable the ability to "reconnect" to the devices if the devices were already connected. If the status of connectivity was unsuccessful, the GUI would look like Fig. 9. The labels above the respective button would display "Not connected", and the program will allow the user to hit the buttons to reconnect to the devices.

Figure 10 is a screenshot of the second tab of the RangeController program, labeled "View Cameras". This tab allows the user to view any of the streams of any camera on the range by its IP address. Eventually, each camera will have its own unique IP address that will correlate with the number of the station it is placed on. Other functionalities of this tab include adjusting the exposure time of the camera and taking a picture from the stream. The main reason the capability to view the streams of the cameras was added to the RangeController program was to test if each camera was outputting the correct information. This functionality is very applicable: If a shot image did not come out as it was supposed to, the user would be able to view the camera stream from behind the computer as opposed to physically entering the range and locating the camera to determine whether or not it is still usable.

The only addition to the Test Selection tab was the "Update Test Weather" button (see Figs. 11 and 12). This button retrieves the most current temperature, pressure, and humidity data and stores these into the "Test Setup Data" Excel spreadsheet. Although humidity is not currently displayed in the screenshot of the Test Selection tab, it is the third item from the bottom in the Excel spreadsheet.

The difference between Figs. 11 and 12 is that the first shows the RangeController GUI when the Weather Station is not connected, while the second shows the Weather Station connected. To avoid confusion in the program when the Weather Station is not connected, the "Update Test Weather" button would not be able to be selected.

The changes to the Test Selection tab were also made to the Shot Selection tab, in addition to some other minor changes. Similar to the screenshots shown in Figs. 11 and 12, Figs. 13 and 14 display the RangeController program when the Weather Station is and is not connected, respectively. The first difference between the Shot Selection tab and Test Selection tab is that there is an area to display the output folder for the camera images. This was implemented by adding a button to browse for a folder in which to store all the images for each shot. When a shot file is created, a folder with the number of the shot is created inside the folder that was selected from the browser dialog. If the folder had already been selected, only that path will be adjusted to store the file images in that folder. In addition to selecting a folder to output the film images, an "Enable update on fire" checkbox was added to the

weather section. Like the update weather buttons, the "Enable update on fire" checkbox will be disabled if the Weather Station is disconnected. The underlying functionality of the checkbox will occur in the Test tab.

The last tab, as displayed in Fig. 15, is the Test tab, which has added capabilities if the "Enable update on fire" checkbox is selected. If this checkbox is selected, which it is by default, the program automatically retrieves the most current weather information and places it into the respective Excel spreadsheet when the "Arm the Range" button is hit. If the Weather Station has not been connected at this point in the program, a message box will appear to notify the operator that the weather information cannot be updated.

The purpose for incorporating the Weather Station in the RangeController program is to automatically have the weather information stored in the respective locations. This is important: Before this added functionality, technicians at the range recorded the information separately using the Weather Station's software and manually input it into the Excel spreadsheets. This manual input of data into the Excel spreadsheets can be a major source for error in the analyses of the projectiles—the technicians might input a false value or forget to input it all together. By having an automatic input of information from the Weather Station, the errors associated with each shot significantly decrease.

4. Algorithms

4.1 Weather Station

The first step taken to incorporate the Weather Station into the RangeController program was to connect to the device by creating a transmission control protocol (TCP) Client with its own IP address. In order to receive information from the Weather Station, commands must be issued to the network stream of the TCP Client. Thus, it was necessary to open the network stream of the TCP Client and pass the stream into a StreamReader and a StreamWriter variable. These 2 variables facilitate communication with the Weather Station. Some of the pre-defined commands that the Weather Station is programmed with are "*SRTC", "*SRHi", "*SRH2". These commands request the temperature in degrees Celsius, the pressure in millimeters of mercury (mmHg), and the relative humidity (by percent), respectively, as shown in Fig. 16. The StreamWriter variable uses these commands to request information from the Weather Station. After writing the command into the StreamWriter, the Weather Station acknowledges the command and responds by writing the requested information into the StreamReader variable. In order to store the information into a different variable, the information must be read

character by character instead of all at once since there is no carriage return. In the RangeController program, global variables were used to store the weather information so that they could be accessed at any point in the program.

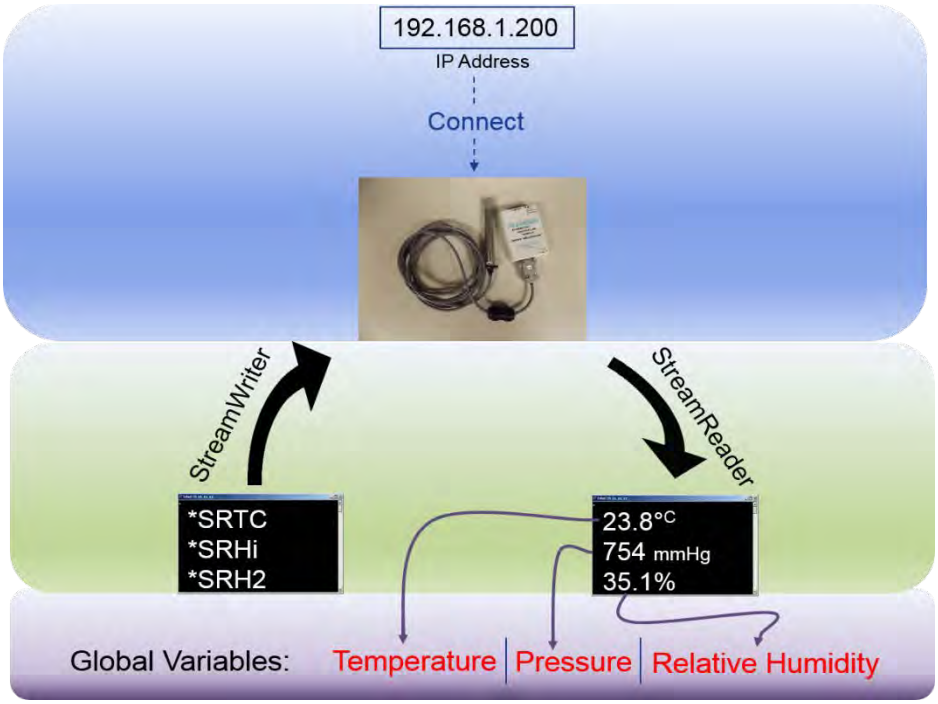

Fig. 16 Program flow for the Weather Station

4.2 Cameras

The first step in incorporating the Gigabit Ethernet Cameras into the RangeController program was to write a class that would control the individual parameters and functions of each camera. In the RangeController program, this class was named “Source”. This Source class was instantiated every time a camera devices was discovered. This means there was one Source object for each camera and an array of Source objects in the RangeController program. Eventually, the range may require different resolution Gigabit Ethernet Cameras; having the functionality to modify individual parameters will simplify the process of controlling the cameras.

After writing the Source class, the next step was to programmatically search for all of the Gigabit Ethernet Camera devices connected to the Ethernet port of the computer. Then, after all of the devices had been found, the IP address of each of these devices was stored in a devices array.

Figure 17 depicts the set of processes that would be completed for each element in the device array. The first step was to connect to the camera device using its IP

address. Instead of using a TCP Client to connect to the cameras, functions from the eBUS software development kit (SDK) made by Pleora Technologies were used. This SDK was encapsulated with several classes that made the process of incorporating the cameras into the RangeController program more convenient. The second step was to create an object from the Source class that was written. When the class is instantiated, the connected device variable should be passed into the constructor of that class. Then, using the functions from the eBUS SDK, the device should be configured for streaming. The fourth step was to set the necessary parameters of each camera. Three of the parameters that must be set for each camera are the TriggerMode, TriggerSource, and ExposureTimeRaw. Just before the projectile is fired, these parameters must be set to "On", "External", and "40", respectively. Setting the trigger mode to "On" and the trigger source to External lets the camera know that when the external trigger is signaled it should capture a signal frame after a certain delay—which in this instance is the exposure time in milliseconds. The external trigger that will be used on the range occurs when the projectile passes the IR screen, and the IR screen sends a 5-volt charge to the back of the camera. After these settings are set for each camera, the last step for properly incorporating the cameras into the RangeController program is to start the stream. Starting the stream for each camera does not perform any image acquisition; it just tells the camera to start waiting for the external trigger so that it can capture the frame.

Fig. 17 Steps conducted for each ethernet camera

After the shot is complete, all of the frames that were captured on each camera are downloaded to the computer, making sure to only download a picture if the camera was able to capture a frame. (Sometimes a projectile might not make it to the very end of the range—thus, not triggering some of the cameras.) The program also waits until the shot has been completed to download all of the images because downloading several images at once will exhaust the Central Processing Unit, which can lead to problems in executing other parts of the code.

5. Conclusions

The Spark-Range Film Package has been a major milestone on the way to completing the end goal of this project: to fully automate the 2 ranges. Despite

improvements to the FilmScanner program—making it compatible with the Vidar Scanners that do not utilize the TWAIN drivers, adding a picture box, and changing the resolution textbox to a combo box—there is still a large amount of human interaction to acquire the measurement data about projectiles fired at the TEF and AEF ranges. By incorporating a Weather Station and digital cameras in AEF and/or TEF, the time taken to conduct a single shot as part of a larger test will be reduced to 1–2 minutes from 30–45 minutes. The RangeController program now has the functionality to control these devices. This program is able to automatically record the weather information as well as take and save frames from an external trigger for a Gigabit Ethernet Camera. The obstacle the ARL has encountered is finding a low-priced yet effective camera that is able to capture the image of the projectile. The current camera that was tested with this software was an 8MP Gigabit Ethernet Camera made by Imperx. With the current lens, the camera is not able to pick up any image (black and noisy image after adjusting parameters) because it is not sensitive enough in low-light situations. In the future, a better lens will be tested with this camera; also, a less-expensive Gigabit Ethernet camera that uses Pleora's eBUS SDK will be found.

6. References and Notes

1. Braun WF. The free flight aerodynamics range. Aberdeen Proving Ground (MD): Army Ballistic Research Laboratories (US); 1958 July. Report No.: 1048. Also available at http://www.dtic.mil/dtic/tr/fulltext/u2/202249.pdf.

2. Haire SM. Transonic Experimental Facility (TEF) free-flight spark range: automation of film reading and processing. Aberdeen Proving Ground (MD): Army Research Laboratory (US); 2015 Aug. Report No.: ARL-TR-7387. Also available at http://www.arl.army.mil/www/default.cfm?technical_report=7475.

3. Arrow Tech Associates, Inc. ARFDAS: ballistic range data analysis system: user and technical manual. South Burlington (VT); 1997 May.

4. Private communication, Charith Ranawake with Mark Steinhoff, Arrow Tech Associates; 2015 July.

List of Symbols, Abbreviations, and Acronyms

AEF	Aerodynamic Experimental Facility
ARL	US Army Research Laboratory
DLL	Dynamic Linked Library
GUI	graphic user interface
IP	internet protocol
IR	infrared
mmHg	millimeter of mercury
SDK	software development kit
TCP	transmission control protocol
TEF	Transonic Experimental Facility

1 (PDF) DEFENSE TECHNICAL INFORMATION CTR DTIC OCA

2 (PDF) DIRECTOR US ARMY RESEARCH LAB RDRL CIO LL IMAL HRA MAIL & RECORDS MGMT

1 (PDF) GOVT PRINTG OFC A MALHOTRA

34 (PDF) DIR USARL
RDRL WM
M CHEN
RDRL WML
P PEREGINO
M ZOLTOSKI
RDRL WML A
W OBERLE
RDRL WML B
N TRIVEDI
RDRL WML C
S AUBERT
RDRL WML D
R BEYER
RDRL WML E
R ANDERSON
W AUBRY
D CASHEN
I CELMINS
F FRESCONI
J GARNER
B GUIDOS
P KEPPINGER
J KRANZ
G OBERLIN
K PAXTON
T PUCKET
J SAHU
S SILTON
J SMITH
P WEINACHT
K WILLAN
RDRL WML F
T BROWN
M ILG
B KLINE
J MALEY
B NELSON
RDRL WML G
J SOUTH
RDRL WML H
J NEWILL
RDRL WMM
J ZABINSKI
H E MAUPIN
RDRL WMP
D LYON

4 (PDF) ARROWTECH ASSOCIATES
W HATHAWAY
C HILLMAN
M STEINHOFF
C RANAWAKE

1 (PDF) DRDC
D CORRIVEAU