STUDENT SUMMER INTERNSHIP TECHNICAL REPORT

# Mobile Hot Cell for End-of-Life Source Management - Camera Control

DOE-FIU SCIENCE & TECHNOLOGY
WORKFORCE DEVELOPMENT PROGRAM

Date submitted:

December 4, 2020

Principal Investigators:

Christopher Excellent (DOE Fellow Student)
Florida International University

Steven E. Egan (Mentor)
Idaho National Laboratory

Ravi Gudavalli Ph.D. (Program Manager)
Florida International University

Leonel Lagos Ph.D., PMP® (Program Director)
Florida International University

Submitted to:

U.S. Department of Energy
Office of Environmental Management
Under Cooperative Agreement # DE-EM0000598

FIU | Applied Research Center
FLORIDA INTERNATIONAL UNIVERSITY

## ABSTRACT

Mobile Hot Cells (MHC) are very desirable for the removal of radioactive sources that are nearing the end of their life cycles. The processes in place to perform such tasks, however, can be quite expensive and time consuming due to the lack of field-tested MHC's that exist today. This also causes major health and safety risks due to the uncertain designs of those third-party MHCs. To remedy this, Idaho National Laboratory (INL) is developing a Mobile Hot Cell that is safe and easily deployed by applying lessons learned from current designs.

INL's MHC will follow a black cell design, a design that does not utilize the traditional window-master-slave manipulators to function. The black cell design will utilize state-of-the-art robotic manipulators that are programable and teleoperated via a 3D mouse and custom software to perform the disassembly and assembly of the radioactive capsules that are used in the MHC. These manipulators will be coupled with a wide mix of 2D cameras located around the cell. The 2D cameras will work with an LED lighting system that will allow the operator to control the brightness of each light to create shadows or visual cues for the operator when the robot is teleoperated. A stereoscopic camera will also be used for very sensitive meticulous tasks within the MHC. Controlling the stereoscopic camera's function during teleoperation is very important to ensure the successful completion of those tasks. This would easily be done with the vendor provided infrared remote control; however, there is a 15-inch thick steel wall between the operator and the interior of the MHC. This prompts the need for a camera control system that can provide the same level of control that the infrared remote control would have provided. This paper will discuss the two methods used to create that control system for the stereoscopic cameras.

# TABLE OF CONTENTS

ABSTRACT ........ iii
TABLE OF CONTENTS ........ iv
LIST OF FIGURES ........ v
1. INTRODUCTION ........ 1
2. EXECUTIVE SUMMARY ........ 2
3. WORK DESCRIPTION ........ 3
4. ANALYSIS ........ 10
5. CONCLUSION ........ 11
6. REFERENCES ........ 12
APPENDIX A. ........ 13

# LIST OF FIGURES

Figure 1. Comparison between traditional hot cells (left), and black cells (right). ........................ 4
Figure 2. Stereoscopic camera angle diagram. .................................................................................. 5
Figure 3. Camera remote control GUI. ............................................................................................. 7
Figure 4. Button matrix diagram. ...................................................................................................... 8

# 1. INTRODUCTION

This paper focuses on the development of a stereoscopic camera control system for a Mobile Hot Cell that is primarily used for medical devices, but is also used for end-of-life radioactive sources. The paper will first include a brief background on mobile hot cells and stereoscopic cameras to help the reader understand how they work together. Two primary methods were devised for the control system of the stereoscopic cameras. The first method requires additional electronic components to be added to the control board of the camera, while the other is primarily programing based. Both methods will be discussed in further detail in Section 3. The discussion will also include how Virtual Network Computing (VNC) was utilized in this project, how the Graphical User Interface (GUI) was created, as well as the program scripts that were written to create the overall control system. A discussion of the results will then be presented in Section 4.

# 2. EXECUTIVE SUMMARY

This research work has been supported by the DOE-FIU Science & Technology Workforce Development Initiative, an innovative program developed by the U.S. Department of Energy's Office of Environmental Management (DOE-EM) and Florida International University's Applied Research Center (FIU-ARC). During the summer of 2020, a DOE Fellow intern, Christopher Excellent, spent 10 weeks doing a hybrid summer internship, half virtual and half on site at Idaho National Laboratory under the supervision and guidance of Steven E. Egan. The intern's project was initiated on June 15, 2020 and continued through August 22, 2020 with the objective of aiding in the development of a camera control system for Idaho National Laboratory's Mobile Hot Cell.

# 3. WORK DESCRIPTION

## Mobile Hot Cell

INL’s Mobile Hot Cell follows a black cell configuration. This implies that there are no windows for the operators to see directly into the cell, as is the custom in traditional hot cells. Figure 1 shows a comparison between traditional hot cells and black cells. In following a black cell set up, the MHC will utilize robotic manipulators in place of the traditional master slave manipulators. The robotic manipulators will create a more precise and versatile operation within the cell. This is because the robotic manipulators can be programed to perform tasks with pinpoint accuracy. They can also be automated to perform repetitive and tedious tasks, therefore reducing operator fatigue. Another advantage of the robotic manipulators is the safety areas that can be programmed to prevent the operator from moving the arms into dangerous areas and damaging them.

The window, which is the vision system in the traditional cells, will be replaced by a multitude of 2D cameras located strategically in the cell to provide views of the cell at different angles. The combination of these camera angles will allow the operator to have a full view of the work area and the areas around the cell; this is an attractive feature of the black cell design. The cameras used in the MHC are off-the-shelf cameras and can withstand radiation long enough for the tasks to be completed without them burning out. This was proven after several tests were performed to verify the survivability of the cameras and the robots. The vision system will also include a stereoscopic camera that will be used to provide the operator with a 3D image of the components with which the manipulators will be interacting. A brief description of how these stereoscopic cameras work will be discussed in the next subsection followed by how it will be used in the MHC.

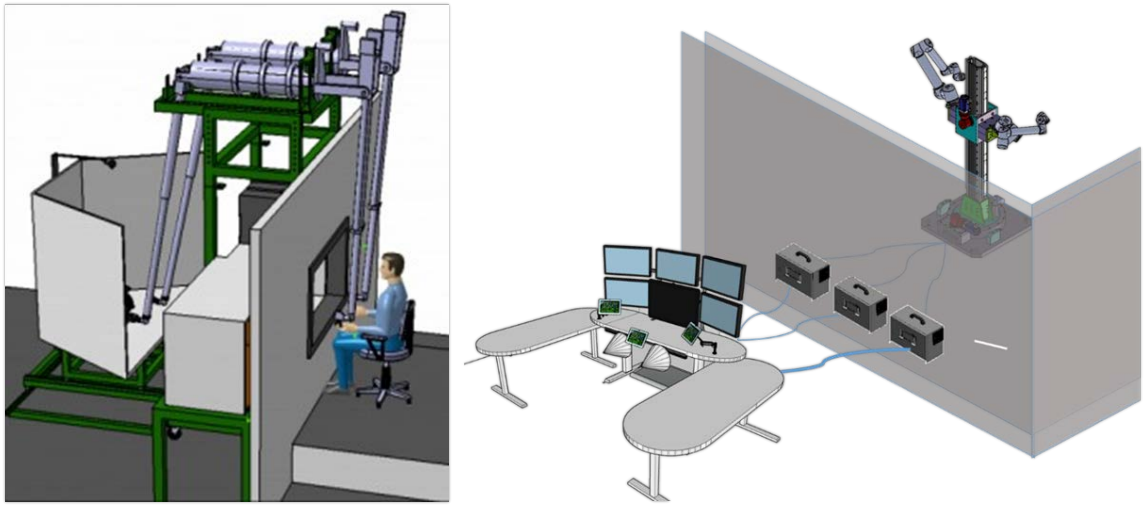
**Figure 1. Comparison between traditional hot cells (left), and black cells (right).**

## Stereoscopic Cameras

Stereoscopic cameras are cameras with two or more lenses with separate image sensors that are used to simulate human binocular vision. Each lens is located at approximately 2.5-inches apart, which is equivalent to the intra-ocular distance, i.e., the distance between one’s eyes. This allows 3D images to be captured or recorded by having each camera capture an image of the same item of interest at a few inches to the right or left, as seen in Figure 2. Those images can then be processed through computer programing to create 3D images. In the MHC, the 3D images captured by the stereoscopic cameras will be projected on an auto-stereoscopic display. The auto-stereoscopic display allows 3D images to be seen without the need for 3D glasses. This is important because it eliminates the need to remove and wear 3D glasses by the operator when he or she needs to switch from looking at a 2D display to a 3D one or vice versa. Utilizing this camera in the MHC facilitates working with small components during the assembly and disassembly of the radioactive capsules, as well as utilizing the manipulators to move objects around in the MHC. The key to utilizing this camera is being able to control its various functions through the 15-inch thick wall; however, this cannot be accomplished with the camera’s infrared remote control because of the wall thickness. Therefore, the need for a reliable and user-friendly method of controlling the cameras is prompted. The way the team accomplished this will be discussed in the next subsection.

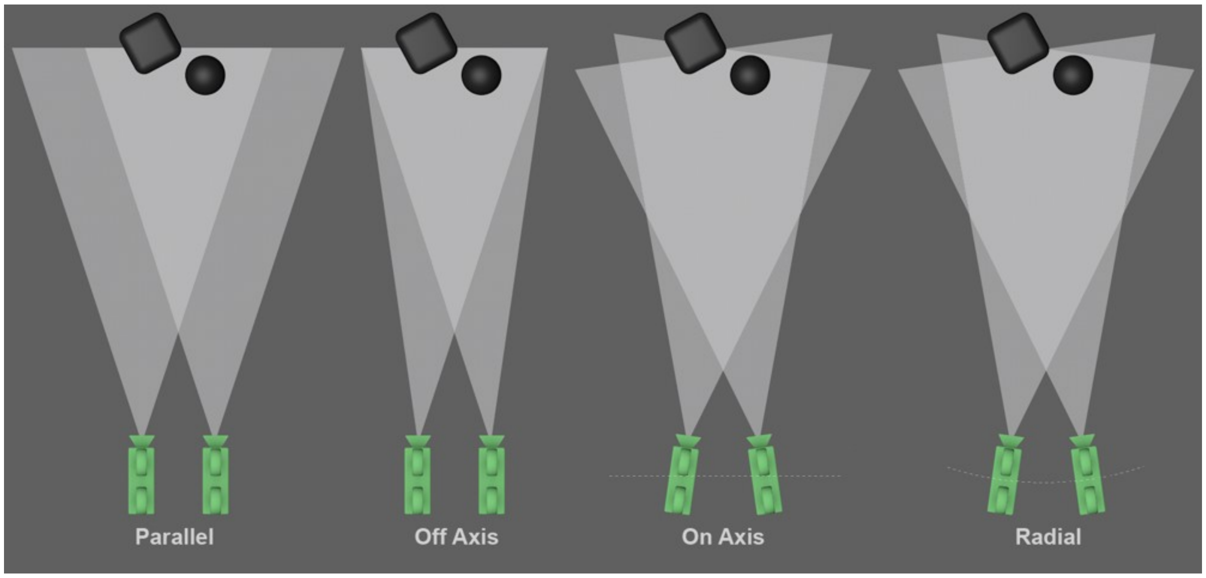

**Figure 2. Stereoscopic camera angle diagram.**

## Camera Control Methodology

As mentioned in the previous section, a reliable and user-friendly remote control is necessary for the stereoscopic cameras, and since there is the issue of a 15-inch thick wall between the MHC and the outside controls, the remote control will need to be wired. With that in mind, the team decided to create a control system that works in the following manner; a camera will be connected to a Raspberry Pi and both will be placed inside of the mobile hot cell. That Raspberry Pi will contain a program script that will be used to control the various functions of the cameras. To run the aforementioned camera control program, a Virtual Network Computing (VNC) connection will be established between the Raspberry Pi and a computer (located outside of the cell) via an Ethernet connection. This method works because VNC allows a user to fully control another computer through a Wi-Fi or Ethernet connection. Since the Raspberry Pi is miniature computer, another computer should be able to control it without any issues as long as the user has the proper IP address and has the VNC software installed on both computers respectively.

The VNC software that we are utilizing to create this control system is Real VNC. The team decided to utilize this VNC software because it permits a quick, simple, and safe way to establish a communication between the two computers. While it is true that there are other methods of controlling computers in the same way that the VNC does, those methods tend to be more complex to use and are time consuming to set up. The complexity of those other methods primarily reside in the fact that they require the user to be quite proficient in programming. Since we want the camera controlling scheme to be user friendly, the Real VNC software allows us to make it just that. Furthermore, once the software is installed on both the Raspberry Pi and the external computer, we can then begin the next step for the control system for the camera.

To add to the user-friendly aspect of the camera control system, we wanted to create a Graphic User Interface (GUI) that would allow the user to control the camera by simply clicking on buttons that would appear on the computer screen. The GUI was designed to resemble the remote control that the camera came with to allow for intuitive use. We utilized Tkinter, a programming library that allows users to create GUIs, to build the GUI on Python (see Figure 3**)**. This library allowed us to construct a window that will contain the GUI and the buttons, as well as define the function of each button. In APPENDIX A, snippets of how Tkinter was used to create the GUI can be found, as well as a brief explanation. The team originally planned to use the GUI by pairing it up with the keypad board that the vendor provided with the camera. This method would require us to build a separate circuit and connect it to the Raspberry Pi, which would enable us to trigger the buttons on the keypad board from the computer program.

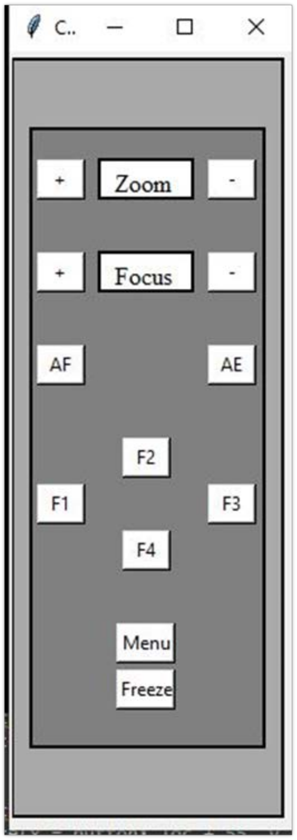

**Figure 3. Camera remote control GUI.**

To make this happen, we had to gain a basic understanding of button matrices and how they work. Figure 4 shows a circuit for a button matrix as well as how the current flows. As can be seen in the image, a button matrix contains rows and columns, and where these lines intersect is where a button would be located. The figure also shows how currents can flow in one specific direction by using diodes. Understanding this concept, we built an extension to this circuit and connected each row and column to an input and output pin on the Raspberry Pi. This allowed us to replicate the keypad board, and for the diodes, we utilized transistors to help us control the current flow. Our final circuit schematic follows a similar patter as the circuit shown in Figure 4 with the difference being that our outputs lead to a General Pin Input/Output (GPIO) on the Raspberry Pi. These pins would be used to send the electric signal from the Raspberry Pi to the keypad therefore triggering the individual buttons on the keypad.

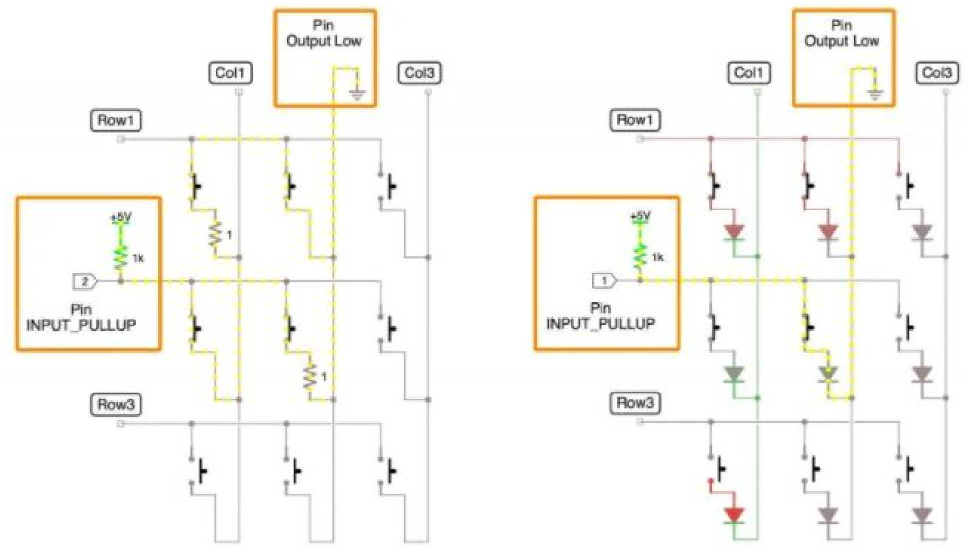

**Figure 4. Button matrix diagram.**

While this plan seemed promising, the team was unable to proceed with it because the manufacturers did not program the keypad board to work with the camera. To remedy this, the team decided to take a slightly different approach. The team decided to utilize the same GUI, but instead control the buttons through serial communication. This required us to initiate the serial communication in our python programming as seen in the snippet below.

```
import serial
ser = serial.Serial(
        port="/dev/serial0",\
        baudrate= 115200,\
        parity=serial.PARITY_NONE,\
        stopbits=serial.STOPBITS_ONE,\
        bytesize=serial.EIGHTBITS,\
        timeout=1)
```

Once the serial communication was established, we were then able to send commands to the camera by following the serial protocol that the manufacturers provided to us. The following lines of code show the portion of the program that reads the serial commands. Once the user runs this code, they would input commands such as “zt”, “zw” and “fn”. These commands would make the camera zoom in, zoom out, or focus-in respectively.

Upon the successful completion of this program, a separate program was written to have these serial commands preplaced in the GUI, so that once the buttons were pressed, they would execute the serial commands. The programing lines below depict how this was done.

```
1. def zoom_plus():
2.     ser.writelines("#z100 \r")
3.     line = ser.readline()
4.     print(line)
5.     sleep(0.1)
6.     print("Zoom was increased")
7.     sleep(0.1)
```

In this program script, line 2 contains the command “`#z100 \r`”, which tells the camera to zoom in for a value of 100. Line 3 then passes that command to the serial port for the camera to interpret. That entire program function, named `zoom_plus()` as seen in line 1, is then passed into the button functions so that when the buttons are clicked on in the GUI, the camera can zoom in; the lines of code that build the buttons can be seen below. A similar pattern was repeated for each individual button and each had their specific commands.

```
8. #+ button
9. buttonPlus= Button(win, text = "+", command = zoom_plus,
   height = 1, width = 1, bg="white",compound = LEFT)
10.     buttonPlus.place(x = buttonX_loc, y = buttonY_loc )
```

Lines 8 through 10 are how the buttons are created and localized in the GUI. In line 9, where it scripts “`command = zoom_plus`”, we can see how the zoom function was called into the zoom button therefore allowing our buttons on the GUI to do the specific task of zooming in. Once we set up all of the button functions for the GUI in the same manner with their respective commands, we were able to successfully create a user-friendly control system for the cameras. In the next section all the methods are analyzed with a discussion of their pros and cons.

# 4. ANALYSIS

As stated in the previous section, we began by scripting a program that would allow the user to directly send commands to the camera from the serial prompt. This program script works great because it allows the user greater flexibility with the commands they can send to the camera. That is because the user can have the list of commands next to them and script the specific commands they want to send to the camera at will. The disadvantage of this method is that the user would need to learn the protocol that needs to be followed in order to send those commands to the camera. In this case, the user would need to learn the basics of ASCII serial protocol format in order to send these commands. An example of this is as follows: If the user wanted the camera to zoom in for a value of 10, they would need to type in "`#z10 \r`" into the serial window. Breaking this command down, the "#" in that line of code is a header and must always be typed in before any command is inputted. The "`z10`" is the command to zoom in for an increment of 10 while the "`[SPACE]\r`" is the checksum and carriage return (the closing arguments) that must be typed in for the computer chip on the camera to be able to interpret the command that has been sent. Controlling the camera in this manner, although it has greater benefits, can be tedious and may require some practice for the reader to be able to accurately send commands 100% percent of the time. This can be remedied by pre-scripting the header format into the program so that the user only must type in the command (`z10`) and the closing argument (`[SPACE]\r`) in the serial window.

Since this method may not work for everyone, we decided to incorporate the serial commands into the GUI so that the user can simply click on buttons to have the camera zoom in, change focus, or do anything else. As we demonstrated in the previous section, we were able to set up the zoom button in the GUI to work with the zoom command of the serial protocol. The same thing can be done for all the buttons in the GUI and should additional commands be required, additional buttons can be created to undertake those specific commands in the GUI.

# 5. CONCLUSION

To conclude, we were able to successfully develop a user-friendly control system for the stereoscopic camera that is meant to be used in the Mobile Hot Cell. The team was able to develop two ways of controlling the cameras through python programing; one requires the user to directly write in the serial window and the other is done through a Graphical User Interface that only requires the user to click on the buttons. The first method that the team initially explored would have been a great way to control the camera; however, since the manufacturers did not set up their keypad to be used in that manner, the team decided to go through the two ways previously described. It is projected that this camera control system will also be used in the Sample Prep Lab, another project at INL. Through the completion of this project, the Mobile Hot Cell can work more efficiently with greater success.

## 6. REFERENCES

1. "Stereo Camera." *Wikipedia, the Free Encyclopedia*, Wikimedia Foundation, Inc, 15 Feb. 2005, en.wikipedia.org/wiki/Stereo_camera. Accessed 20 Nov. 2020.
2. "3D Stereo Rendering." *C4D Cafe*, www.c4dcafe.com/reviews/r13/3dstereo.html

# APPENDIX A

**TKinter Utilization**

Initialization of the Tkinter library:

```
1. from Tkinter import *
2. import Tkinter as tk
3. …
4. …
5. …
6. win.pack()
7. gui.mainloop()
```

Lines 1 and 2 of the codes are utilized to call the Tkinter library and assign a reference name to it in the python program respectively. Line 3 through 4 is where the user would write their lines of code to create their Graphical User Interface (GUI) window and the list of commands and functions they wish to place in the program. Finally, line 6 makes it so that every component of the GUI remains together when the used moves the GUI window around in the monitor through a mouse click. Line 7 is used to keep the GUI window open until the user decides to close it. This format can be used to create various GUIs and multiple examples and instructions can be found online on how to use this library.