

Multiple Ways for Medical Data Visualization Using 3D Slicer

Ismail Mohammed Bahkali

Department of Computer Science
University of Colorado Colorado Springs
Colorado Springs, USA
ibahkali@uccs.edu

Sudhanshu Kumar Semwal

Department of Computer Science
University of Colorado Colorado Springs
Colorado Springs, USA
ssemwal@uccs.edu

Abstract— Computers can process large amounts of data. Medical practitioners can deliver better services and provide more accurate diagnoses and treatment regimens to patients. This document described how 3D Slicer allows Command Line Interface (CLI), Python, Jupyter, and MATLAB in software to process medial data. 3D Slicer has become useful software worldwide since 1997, especially in the medical field for pre-operative visualization and analysis. Today, 3D Slicer is supported by The National Alliance for Medical Imaging Computing (NA-MIC), Neuroimaging Analysis Center (NAC), Biomedical Informatics Research Network (BIRN), The National Center for Image-Guided Therapy (NCIGT), The Harvard Clinical and Translational Science Center (CTSC), and the Slicer Community worldwide as a platform to develop new ideas. In this paper, we demonstrate our knowledge in using the 3D Slicer software.

Keywords-component; 3D Slicer, MATLAB, Jupyter, Python, CLI.

I. INTRODUCTION

3D Slicer is one of the quintessential software commonly used in the medical field [16]. The system is open-source and used to visualize and analyze medical image data sets [2] [16] [20]. It supports a wide range of information formats such as images, surfaces, segmentation, and annotations in either 2D, 3D, and 4D [4] [15]. The platform allows developers to implement and evaluate new features, which can then be distributed to other clinical users. The main programming languages used to code 3D Slicer are C++ and Python. However, it provides a secondary extension module, enabling users to customize the platform's functions and appearance.

According to the article by Fedorov [4], the program is written in Python and C++. It has a full environment and packages for the Python language [5]. 3D Slicer has a built-in Python console functions as a Jupyter notebook kernel for easy prototyping and customization of products for customers, hence analyzing different coding languages that one can utilize to code the program. This paper analyzes and examines how to code 3D Slicer using CLI, Python, Jupyter, and MATLAB. The recommended process of coding 3D Slicer includes GitHub download, CMAKE compilation, generation, and package because the procedure is simple and convenient for the development of a software programming environment.

II. ADVANTAGES OF USING 3D SLICER

3D Slicer has the following advantages:

- It is a free access software compatible with different software such as Linux, Mac OS, and Windows.
- The program is versatile as it uses DICOM files.
- It can be integrated with different image devices, such as MRI, CT scanners, and microscopes.
- 3D Slicer provides real-time images and analysis, which improves surgical operations and efficiency.
- Users are at liberty to customize the platform by adding modules that suit them.
- Visualization allows professionals to share opinions that lead to improved software.
- It can help generate better information, possibly reducing diagnosis time for treatment.
- It reduces the cost of surgery by minimizing surgery planning and process.
- It has broad functionality and scalability.

III. 3D SLICER LIMITATION

Despite the many advantages, 3D Slicer is characterized by the following limitations:

- It requires technical know-how to use the software or to analyze data. Therefore, it limits the use of the software.
- Restarting the program after adding extensions is mandatory; if the right sequences do not follow installing it or using it after installation, there could be errors.
- The software is also not approved for clinical use and is meant for research, conditions that limit the usage in real situations [4].
- It is challenging to construct the environment, meaning that there is no easy way to modify the program.

IV. HOW TO USE A 3D SLICER?

3D Slicer operates in any Windows, Mac, or Linux computer, and the installers are available in 64-bit. It is also recommended that the hardware should have a memory of more than 4GB, a minimum resolution of 1024 by 768, and dedicated graphics [4]. New users get overwhelmed by the number of options available after installing the software. It is crucial to learn how the systems operate for different applications. After launching the software, data should be loaded, as illustrated in Figure 1.

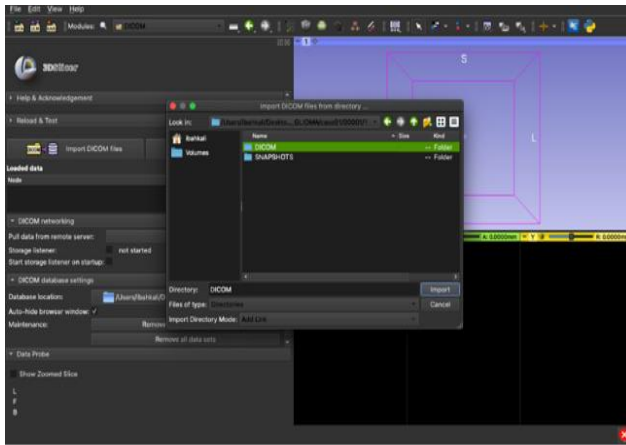


Figure 1. 3D Slicer window for loading data.

After the process above, views can be customized by clicking the controller display push button. The options allow setting viewpoint direction and adding the slice to the 3D viewer. Figure 2 is a screenshot of the 3D Slicer window for managing image views, which we have edited with comments to show different image views.

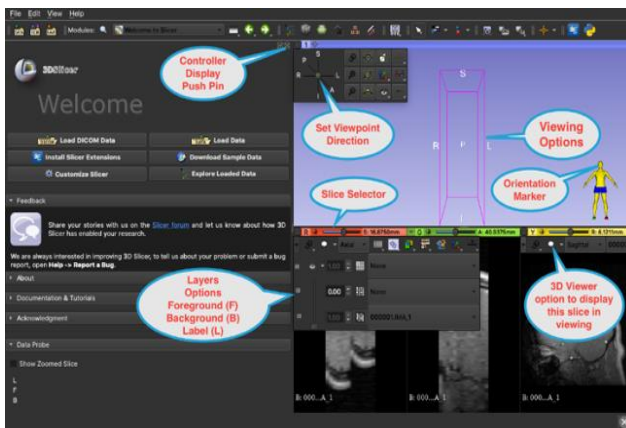


Figure 2. A screenshot of the 3D Slicer window for managing image views.

The third step involves data processing. The 3D Slicer software consists of already included modules (illustrated by Figure 3), which include the following:

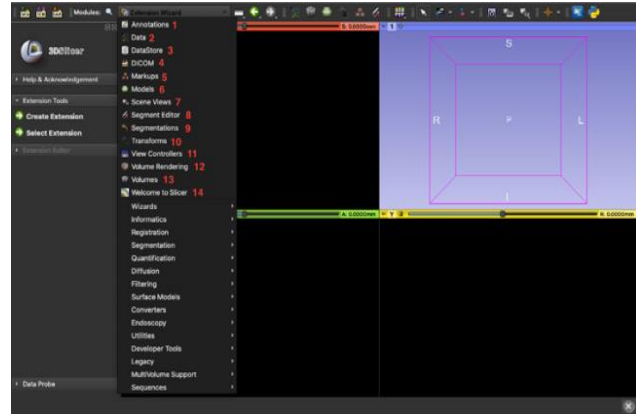


Figure 3. The representation of 3D Slicer built-in modules.

1. **Annotations:** Build and edit additional data linked with a scene. The Markups module will shortly replace the module. Currently supported annotations are fiducial points, rulers, and regions of interest (ROIs).
2. **Data:** It contains all information and allows operation such as search, delete, or rename. The Data module is the fundamental data-organizing point where presenting all loaded data for access and direction is the Data module. It permits regulating the data in folders or patient/study trees (automatically done for DICOM), visualizing any displayable data, transforming whole branches, and many data type-specific features.
3. **DataStore:** Allows users to download and upload datasets.
4. **DICOM** Allows importing, loading, and exporting of DICOM data. Allows sending and receiving data using DICOM networking.
5. **Markups:** Create, edit, and manage markup matrices in two and three dimensions. It can replace fiducial annotations.
6. **Models:** The Models Module loads and balances illustrate parameters of models such as Color, Transparency, and Clipping. Save models via the File menu, Save button. The Add 3D model or a model directory button will permit the user to load any model that Slicer can read and all the VTK models in a directory. Add Scalar Overlay will load a scalar file and associate it with the currently active model. The user can modify the appearance characteristics of the models in the Display pane. Select the model to operate on

from the model selector drop-down menu. Load scalar overlays with a default color lookup table. The user can reassign scalar overlays manually. The user can turn on Clipping for a model in the Display pane and selecting the slice planes that will clip the model are in the Clipping pane. The Model Hierarchy pane enables the user to group models collectively and sets the group's properties.

7. Scene Views: Create, edit, restore, delete scene views. Scene views capture the state of the MRML scene at a given point. The accepted method to utilize them is to load all user data and then adjust the element's visibility and capture impressive scenic views. Unexpected action may happen if the user adds or deletes data from the scene while saving and restoring scene views.
8. Segment Editor: It allows editing segmentation objects by straight illustration and handling segmentation tools on the contained segments in real-time, updating representations are automatically other than the label map one.
9. Segmentation: Any segmentation can hold multiple segments, which match to one structure or ROI. Every segment can include multiple data representations for the same structure, and the module supports automatic transformation between these illustrations and advanced appearance settings, and import/ export features.
10. Transforms: Create and edit transformation matrices.
11. View Controllers: It allows for transforming the views options.
12. Volume Rendering: It implements a 3D visualization of data. It provides advanced tools for the toggling interactive volume rendering of datasets. If supported, hardware-accelerated volume rendering is made available. It permits the selection of preset transfer functions to colorize and set the opacity of data in a task-appropriate approach and tools to customize the transfer functions that specify these parameters.
13. Volumes: It changes the size of different data, adjusting Window, Level, Threshold, Color, and other parameters that regulate the appearance of volume image data in the scene.

14. Welcome to Slicer: A panel features for loading data and customizing the view.

V. PROGRAMMING 3D SLICER

The UI for 3D Slicer consists of three sections: input parameter, results-widgets, and standard 2D and 3D views [14]. The 3D Slicer software architecture is developed in modules, which are coded using different programming languages. The system has three main modules: CLI, Scripted, and Loadable Modules [11]. Accordingly, a developer writes the script modules using Python language that allows easy and extensive access to 3D Slicer's internal systems. The developers further apply C++ language for the loadable modules that control a custom graphical user interface's particular behavior. CLI communicates only through the predefined display, which limits its interaction. Each of the other subsection is developed using specific algorithms to achieve particular functions. Thus, different programming languages are used to create sections of 3D Slicer, which defines their applications. 3D Slicer extension modules can be coded using different languages such as MATLAB, Python, Jupyter, and CLI. MATLAB's functions can be accessed and employed within a 3D Slicer environment using MatlabBridge extension, enabling faster and simple prototyping [9]. As a result of API in Python wrapper, 3D Slicer can be customized using the Python language.

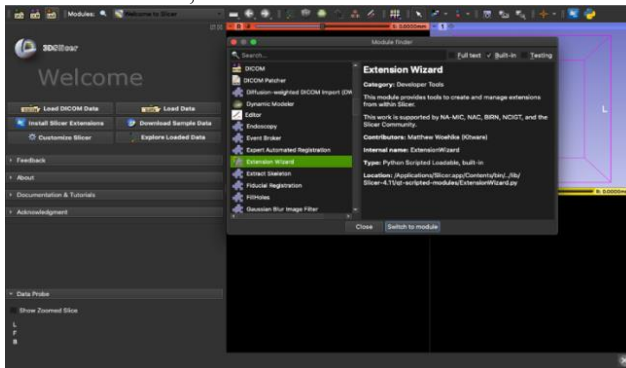
Moreover, the Jupyter language can be implemented in 3D Slicer by following a GitHub centric workflow [17]. The CLI approach can also be used to customize 3D Slicer through input and output strategy, which is more accessible but limited to one batch processing at a time. The recommended steps for customizing 3D Slicer software include the following steps:

- GitHub download.
- CMAKE compilation.
- Generation and package phases.

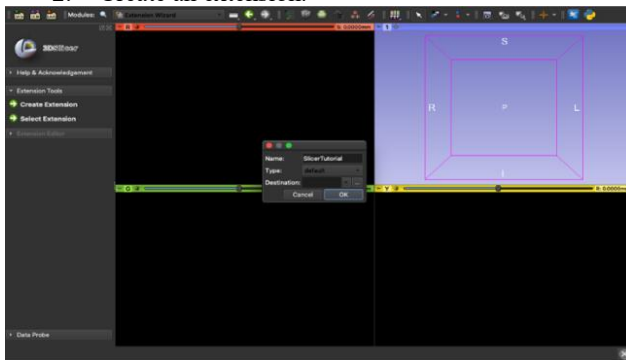
VI. HOW TO CODE 3D SLICER USING PYTHON?

Python is also used in many relevant packages, such as SimpleITK, scikit-learn, and TensorFlow [3]. The 3D Slicer software is written on Visualization Toolkit (VTK), a pipeline based library of graphical data. The program is mainly coded in C++, but the API uses a Python wrapper [1]. The user interface is developed in Qt, and it may be modified using C++ or Python [1]. 3D Slicer has a Python-based platform with an interactor within the graphical user interface capable of managing the software. Thus, developers can use built-in command-line interpreters to modify 3D Slicer software since they have access to the API linked libraries such as VTK, Qt, and simpleITK, which are wrapped with the Python language [13] [14]. Wrapping 3D Slicer software with the Python language makes it possible to develop extension modules for the program, enabling faster and simple prototyping. The steps are:

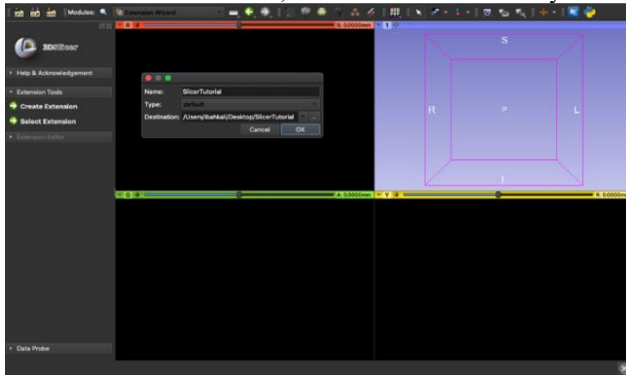
1. Open3DSlicer, press the search button next to Modules, click extension wizard.



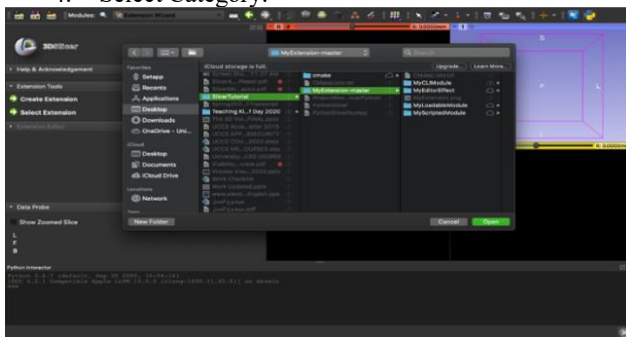
2. Create an extension.



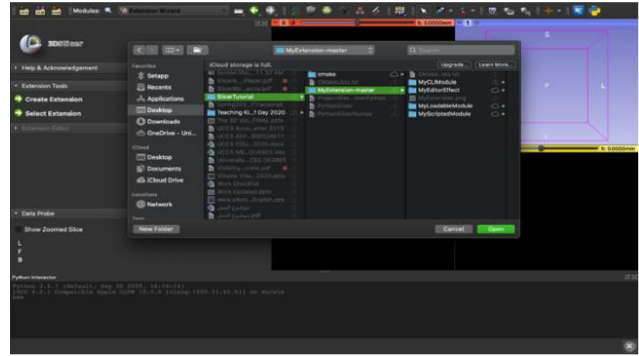
3. Select destination, which is the root directory.



4. Select Category.



5. Add a module to extension.

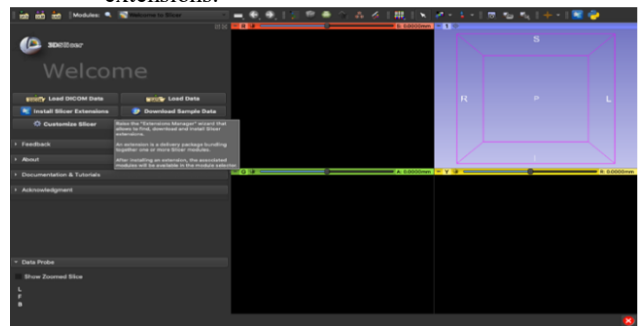


6. Restart 3D Slicer and then open the created module.

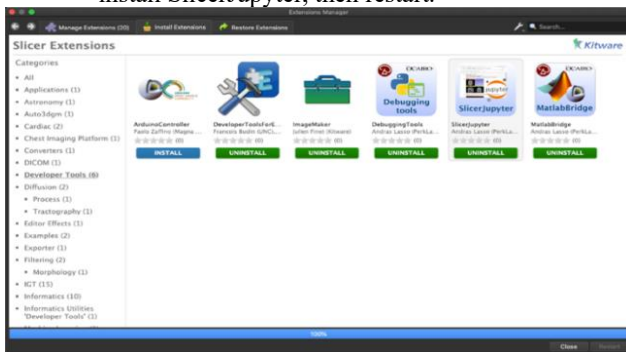
VII. HOW TO CODE 3D SLICER USING JUPYTER?

Jupyter was developed in 2011 as Python notebooks and is an open software web-based application for programming. The software supports numerous languages through the integrated kernels. It can, therefore, be employed to code 3D Slicer for customization purposes [17]. Yaniv examines SimpleITK Jupyter notebooks, which can be programmed using either Python or Language R [17]. The software consists of a single document that describes the image-appraisal workflow using text, equations, tables, and figures. The program is separated into logical parts, known as code chunks. The algorithm provides an opportunity for modifications via the graphical user interface. The SimpleITK Notebook is available using git version control systems and GitHub. Lowekamp [10] described the SimpleITK as an interface that allows algorithms and data structures for the Insight Toolkit (ITK) [7]. The implementation of Jupyter language can follow a GitHub centric workflow, where new codes can be created using the git workflow strategy, which is topic branch-based. As such, a developer uploads code to a forked repository before pulling a request to the primary dataset. The action sets-off testing using CircleCI continuous service [17]. The code is tested and integrated into the system. The Python kernels embedded in the 3D Slicer make it possible to code using Jupyter language, following the procedure outlined by Lowekamp [10]. The steps are:

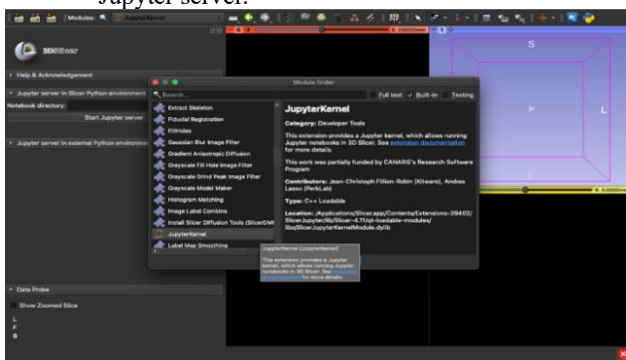
1. Open 3D Slicer software then click install slicer extensions.



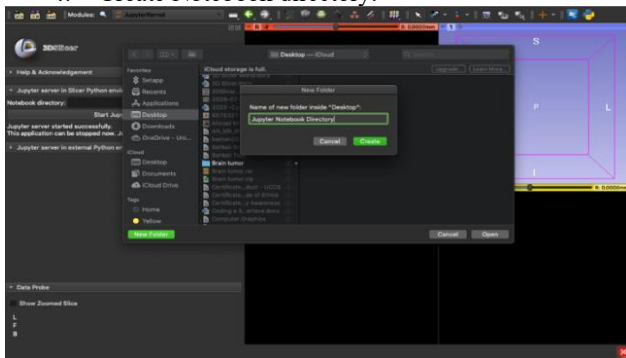
- Click on install extensions, pick developers tools, install SlicerJupyter, then restart.



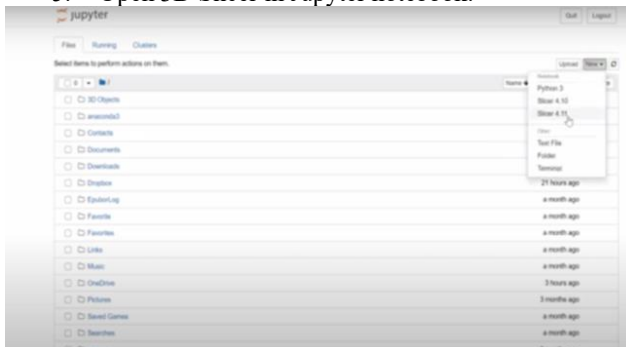
- Launch JupyterKernel from search, press start Jupyter server.



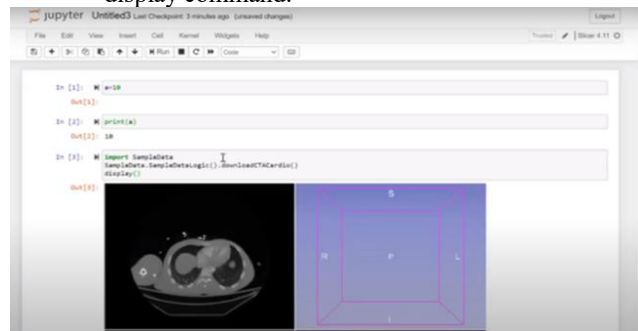
- Create Notebook directory.



- Open 3D Slicer in Jupyter notebook.



- Code using Python, Slicer views can be shown using display command.



VIII. HOW TO CODE 3D SLICER USING MATLAB?

MATLAB is a popular software used for the development of algorithms for computational purposes. It is employed for matrix manipulations, graphing data, creation and implementation of algorithms, and user interface development [1]. However, the program is limited to medical analysis because of the difficulties of importing/exporting, visualization, and processing images. Nevertheless, Lasso [9] proposed a strategy that would enable the running of MATLAB features in 3D Slicer having the architecture of the proposed MatlabBridge mentioned in Figure 1 in [9].

The authors suggested an extension, MatlabBridge, which can start the software, receive input data, run algorithms, and process information using a TCP/IP based OpenIGTLink protocol [9]. The graphical user interfaces are described in the CLI module in XML format. The MatlabBridge should be downloaded and installed using the extension manager feature in 3D Slicer. It has a MatlabModuleGenerator, a helper tool for creating skeleton modules, which can be extended and customized. MATLABCommander allows other 3D Slicer modules to be run by MATLAB functions. Thus, the MATLAB software can be used to program 3D Slicer software via a MatlabBridge [9]. The steps are:

- Open 3D Slicer, click install slicer extensions.
- Click on install extensions, Pick developers tools, Install MatlabBridge, then restart.
- After restarting, Click the welcome to slicer menu, Pick developer tools, Pick MATLAB, Press MATLAB Module Generator.
- Enter Module Name, Click Generate module, File will exist in directory, Press Restart application.
- Click the welcome to slicer menu, Pick MATLAB, Press FirstMatlabModule.
- Download the data of interest (For example, Download MR Head), click clinical sample data, MATLAB, open your module.
- Enter input volume, output volume, and apply.
- Open your Module, click developer tools, MATLAB, MATLAB Module Generator.
- Copy MATLAB Script Directory and Past it on your hard disk.
- Exit 3D Slicer and Discard any modification.

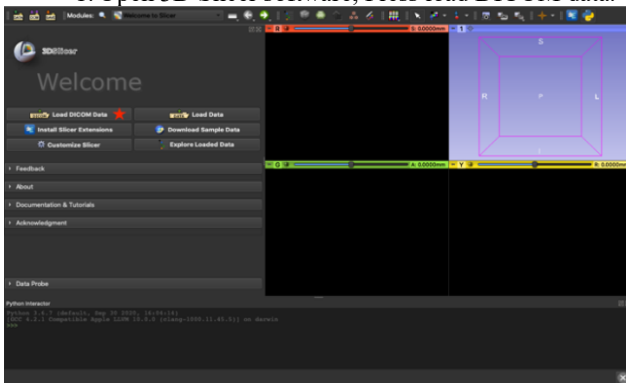
IX. PROGRAMS NEEDED FOR SECONDARY MODIFICATION OF EXTENSION MODULES IN 3D SLICER

3D Slicer provides customization and extension opportunities with an interactive console that gives all the algorithms and data loaded. However, some programs, such as CMake, SVN, Git, Visual Studio, and Qt, are needed to provide a suitable environment to modify the extension modules [11]. The CMake application is used to solve the problem arising from multiple compilers used to program 3D Slicer. It generates a Makefile or Project file, which describes all platforms' construction processes [11]. SVN software stores data and marks changes made to code, allowing the restoration of previous versions [11]. Git is open-source, which is used to manage software versions. Visual Studio is needed for compilation, debugging, and packaging of the developed source code for 3D Slicer [11]. Lastly, the Qt program allows component programming and is mainly used to modify the interface module [11]. CMake, SVN, Git, Visual Studio, and Qt software are needed before changing 3D Slicer extension modules.

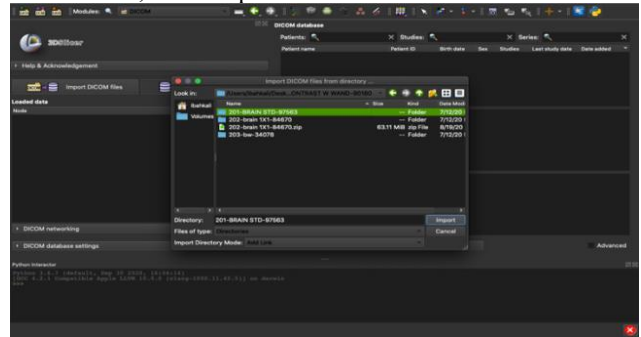
X. HOW TO CODE 3D SLICER USING COMMAND LINE INTERFACE?

One of the 3D Slicer program's main goals is to provide biomedical engineers, software developers, and researchers with an opportunity for quick prototyping and development of images for analysis. Therefore, the tool is open and extensive, with interfaces and design patterns to introduce new functionalities [4]. The key user-level features for modifying the software include DICOM and CLI [4]. The extension code is divided into Python loadable and the C++ CLI module [14]. The former type is detailed later in the study. The C++ CLI module is used for batch processing of input information. Thus, it is limited to the basic input-output setup, where the inputs are the command line parameters [14]. It is also restricted to one-time processing. However, C++ CLI supports easy prototyping by developers. The C++ CLI module allows developers to modify or develop new extension through an input-output phase, which is easier to implement, but it enables only one batch processing at a time. The steps are:

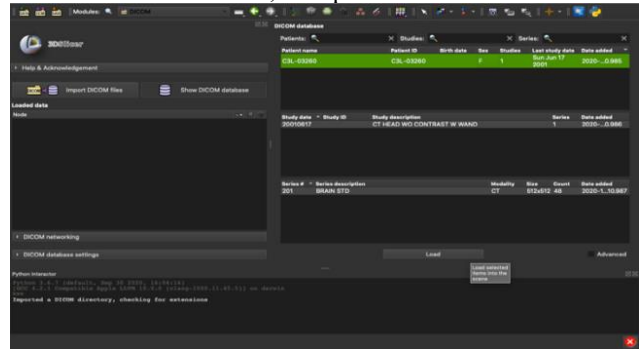
1. Open 3D Slicer software, Press load DICOM data.



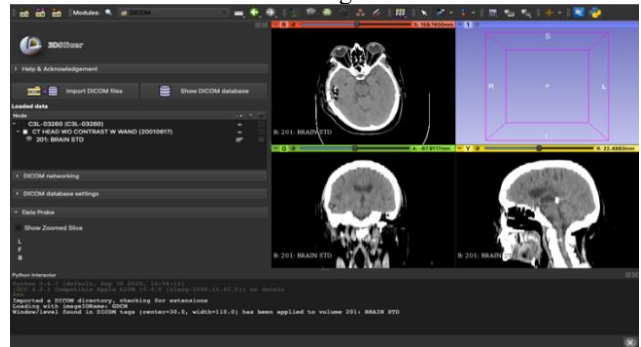
2. Press import DICOM files, Navigate to the dataset file, Press import.



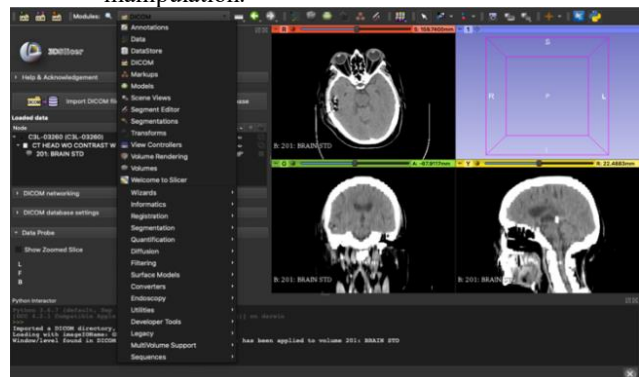
3. Click on the file, then press load.



4. The screen after loading the dataset.



5. Use the Command Phase for different rendering and manipulation.



XI. MODIFYING EXTENSION MODULES IN 3D SLICER

The leading coding languages used for 3D Slicer are C++ and Python. However, it provides a secondary extension module, enabling users to customize the platform's functions and appearance. 3D Slicer allows users to modify the software to suit different applications. Optimizing the software to cater to the medical equipment requirements require developers to be highly accurate at higher speeds and maintain maximum safety during the process. However, the modification process is complicated, and a suitable procedure has been proposed [11]. The recommended steps for customizing 3D Slicer software include GitHub download, CMAKE compilation, and generation and package phases. One acquires source code from GitHub or SVN as the initial step outlined by [11]. However, the developer should configure Git before downloading open software on GitHub [7]. The software should then be installed. The machine information should be set to be utilized by all Git repositories using the following code:

```
$git config--globaluser.name"username"
$git config--globaluser.email@example.com
```

Then, utilize the following code to create a version library:

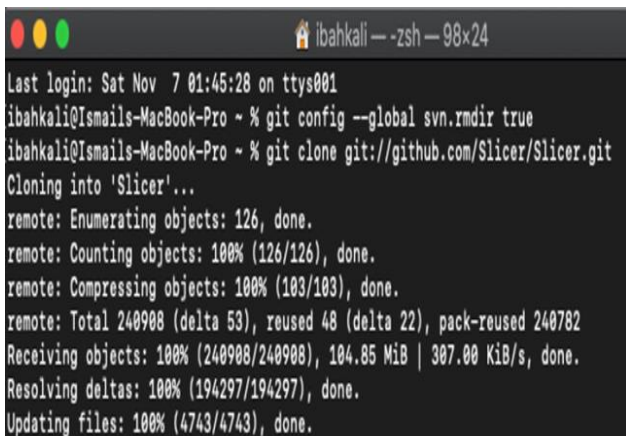
1. Create an empty directory

```
$mkdir mymenu
$cd mymenu
$pwd /Users/ibahkali/mymenu
```

2. Initialize the warehouse git init

```
$git init "Initialized empty Git repository" in:
/Users/ibahkali/mymenu/.git/
```

Lastly, search the source code, select Clone or download, copy the link, open GitBash, type git after \$, put the URL, and press download. The download process is summarized in Figure 4.



```
ibahkali - zsh - 98x24
Last login: Sat Nov 7 01:45:28 on ttys001
ibahkali@Ismails-MacBook-Pro ~ % git config --global svn.rmdir true
ibahkali@Ismails-MacBook-Pro ~ % git clone git://github.com/Slicer/Slicer.git
Cloning into 'Slicer'...
remote: Enumerating objects: 126, done.
remote: Counting objects: 100% (126/126), done.
remote: Compressing objects: 100% (103/103), done.
remote: Total 240908 (delta 53), reused 48 (delta 22), pack-reused 240782
Receiving objects: 100% (240908/240908), 104.85 MiB | 307.00 KiB/s, done.
Resolving deltas: 100% (194297/194297), done.
Updating files: 100% (4743/4743), done.
```

Figure 4. The process of downloading a source code in GitHub

The second step involves compiling and generating the downloaded source code. The procedure can be undertaken using either QT, VS, Git, CMake, SVN, or NSIS [7]. After that, the CMAKE software (Figure 5) should be used to resolve issues arising from multiple compilers [18]. A developer uses the CMake GUI to generate VS project files through the following paths:

- Where the source code is: sets the path to the downloaded source code.
- Where to build the binaries is: sets the space for building and storing a project.

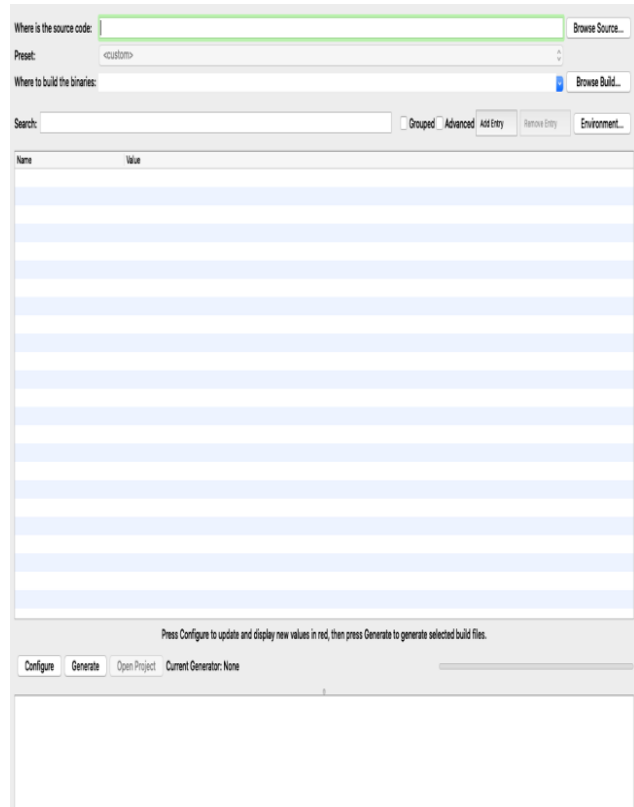


Figure 5. CMake window for path configuration.

The third step is undertaken after the compilation using Visual Studio to generate and debug the code, which might exist or not as a packaged project. The fourth and last phase involves packaging the software. The file might be in packaged or unpackage formats. In the former case, then the file can be produced directly. However, it cannot directly be created if it is not packaged because it is located in the SlicerDMRI.sln file [11]. Clicking the Edit, Application Settings, Modules enables easy importation of the packaged folder [11]. The modules can then be opened and modified according to a user's preference. Figure 6 summarizes the tools and steps needed when undertaking extension module modification in 3D Slicer.

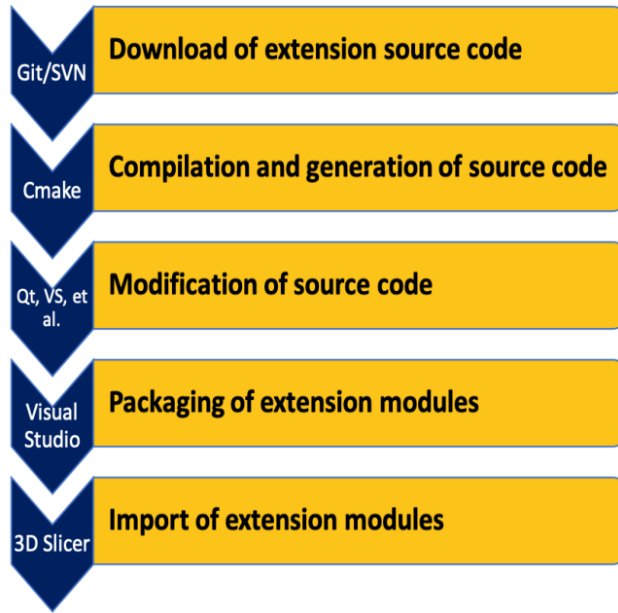


Figure 6. Tools and steps when undertaking extension module modification in 3D Slicer.

XII. CONCLUSION

3D Slicer is an important software, especially in the medical field, because it analyzes sophisticated data through visualization. The software reduces costs and promotes surgical operations safety, which has increased its popularity in medical applications. 3D Slicer is coded mainly using C++ and Python programming languages. However, it provides a secondary extension module, enabling users to customize the platform's functions and appearance. In this regard, the study was undertaken to investigate how a coding environment can be created using different languages to modify the 3D Slicer program and its several paths. It was established that the software could be optimized to meet medical equipment requirements, which need high accuracy, speed, and safety. However, the modification process is complicated, and a suitable procedure was proposed. The recommended steps for customizing 3D Slicer software include GitHub download, CMAKE compilation, and generation and package phases. By using a MatlabBridge extension, MATLAB's functions can be accessed and employed within a 3D Slicer environment, enabling faster and simple prototyping. The Python language can also customize 3D Slicer because its API is done in Python wrapper. Moreover, the Jupyter language can be implemented in 3D Slicer by following a GitHub centric workflow, where new codes are created using the topic branch-based git strategy. The CLI approach can also be used to customize the 3D Slicer through input and output strategy, which is easier but limited to one batch processing at a time. In this paper, we explained these several paths for using 3D Slice so that the usage of 3D Slicer could be more commonplace and grow further.

REFERENCES

- [1] M. Bustamante, "Detection and Quantification of Small Changes in MRI Volumes," 2014.
- [2] D. Chalupa, and J. Mikulka, "A novel tool for supervised segmentation using 3D slicer," *Symmetry*, 2018, pp. 627.
- [3] B. E. Chapman, J. A. Roberts, and A. Sorenson, "Scientific Session Posters and Demonstrations Creating an Open Source Infrastructure for Image Phenotyping in Clinical Research," *SIIM*, 2017.
- [4] A. Fedorov, R. Beichel, J. Kalpathy-Cramer, J. Finet, J.C. Fillion-Robin, S. Pujol, C. Bauer, D. Jennings, F. Fennessy, M. Sonka, and J. Buatti, "3D Slicer as an image computing platform for the Quantitative Imaging Network," *Magnetic resonance imaging*, 2012, pp.1323-1341.
- [5] J. L. Forbes, "Development and Verification of Medical Image Analysis Tools Within the 3D Slicer Environment," 2016.
- [6] D. T. Gering, "A system for surgical planning and guidance using image fusion and interventional MR," (Doctoral dissertation, Massachusetts Institute of Technology), 1999.
- [7] S. Jin, G. Yu, J. Song, J. Chang, J. Cui, "Development environment construction of medical imaging software 3d slicer," *Journal of Complexity in Health Sciences*, 2020, pp. 43-51.
- [8] T. Kapur, S. Pieper, A. Fedorov, J.C. Fillion-Robin, M. Halle, L. O'Donnell, A. Lasso, T. Ungi, C. Pinter, J. Finet, S. Pujol, "Increasing the impact of medical image computing using community-based open-access hackathons," *The NA-MIC and 3D Slicer experience*, 2016, pp. 176-180.
- [9] A. Lasso, K. Alexander, C. Jechel, K. Wang, J. Schreiner, G. Fichtinger, "Running Matlab® functions in 3D Slicer using MatlabBridge," *Imaging Network Ontario*, 2015, pp. 78. www.imno.ca/sites/default/files/2014Proceedings.pdf. Accessed 4 Nov 2020.
- [10] B. C. Lowekamp, D. T. Chen, L. Ibáñez, D. Blezek, "The design of SimpleITK," *Frontiers in neuroinformatics*, 2013, pp. 45.
- [11] L. Ma, C. Zhang, W. Wu, J. Chang, J. Cui, "Secondary development based on 3D Slicer extension modules," *Journal of Complexity in Health Sciences*, 2020, pp. 73-80.
- [12] N. Bruns, "3D Slicer: Universal 3D Visualization Software," *Der Unfallchirurg*, 2019, pp. 662-663.
- [13] F. Pye, N. B. Raja, B. Shirley, Á. T. Kocsis, N. Hohmann, D. JE. Murdock, and E. Jarochowska, "ImageJ and 3D Slicer: open source 2/3D morphometric software," *PeerJ Preprints*, 2019.
- [14] T. Michalík, "Software pro stereotaktickou navigaci v epileptochirurgii," 2019.
- [15] S. Vijayan, S. S. Melo, S. Anamali-Allareddy, F. B. Teixeira, and V. Allareddy, "Segmenting Root Canal Systems Using an Open Source Slicer Software," *Oral Surgery, Oral Medicine, Oral Pathology and Oral Radiology*, 2019, pp. 48-49.
- [16] Y. Muyi, "Medical image segmentation algorithm based on 3D slicer and its application." *Electronic World*, 2016, pp. 14-15.
- [17] Z. Yaniv, B.C. Lowekamp, H.J. Johnson, and R. Beare, "SimpleITK image-analysis notebooks: a collaborative environment for education and reproducible research," *Journal of digital imaging*, 2018, pp.290-303.
- [18] J. Yu, X. Li, Z. Wu, "Design and Implementation of Compiler Theory Demo Module," *Research and Exploration in Laboratory*, 2018, pp. 36.
- [19] X. Zhang, K. Zhang, Q. Pan, J. Chang, "Three-dimensional reconstruction of medical images based on 3D slicer," *Journal of Complexity in Health Sciences*, 2019, pp. 1-2.
- [20] X. Zhou, J. Wang, M. Guo, Z. Gao, "Cross-platform online visualization system for open BIM based on WebGL," *Multimedia Tools and Applications*, 2019, pp. 28575-28590.