Astronomy and Computing 19 (2017) 45–59

Contents lists available at ScienceDirect

Astronomy and Computing

journal homepage: www.elsevier.com/locate/ascom

Full length article

# SlicerAstro: A 3-D interactive visual analytics tool for HI data☆

D. Punzo [a,*], J.M. van der Hulst [a], J.B.T.M. Roerdink [b], J.C. Fillion-Robin [c], L. Yu [d,e]

[a] Kapteyn Astronomical Institute, University of Groningen, Landleven 12, 9747 AD Groningen, The Netherlands
[b] Johann Bernoulli Institute for Mathematics and Computer Science, University of Groningen, Nijenborgh 9, 9747 AG Groningen, The Netherlands
[c] Kitware Inc, Clifton Park, NY, United States
[d] University Medical Center Groningen, Center for Medical Imaging North East Netherlands, University of Groningen, Hanzeplein 1, 9713 GZ Groningen, The Netherlands
[e] Hangzhou Dianzi University, Zhejiang, China

ARTICLE INFO

*Article history:*
Received 27 January 2017
Accepted 19 March 2017
Available online 28 March 2017

*Keywords:*
Radio lines: galaxies
Scientific visualization
Visual analytics
Agile software development
Object oriented development
Empirical software validation

ABSTRACT

SKA precursors are capable of detecting hundreds of galaxies in HI in a single 12 h pointing. In deeper surveys one will probe more easily faint HI structures, typically located in the vicinity of galaxies, such as tails, filaments, and extraplanar gas. The importance of interactive visualization in data exploration has been demonstrated by the wide use of tools (e.g. `Karma`, `Casaviewer`, `VISIONS`) that help users to receive immediate feedback when manipulating the data. We have developed `SlicerAstro`, a 3-D interactive viewer with new analysis capabilities, based on traditional 2-D input/output hardware. These capabilities enhance the data inspection, allowing faster analysis of complex sources than with traditional tools. `SlicerAstro` is an open-source extension of `3DSlicer`, a multi-platform open source software package for visualization and medical image processing.

We demonstrate the capabilities of the current stable binary release of `SlicerAstro`, which offers the following features: (i) handling of FITS files and astronomical coordinate systems; (ii) coupled 2-D/3-D visualization; (iii) interactive filtering; (iv) interactive 3-D masking; (v) and interactive 3-D modeling. In addition, `SlicerAstro` has been designed with a strong, stable and modular C++ core, and its classes are also accessible via `Python` scripting, allowing great flexibility for user-customized visualization and analysis tasks.



## 1. Introduction

Upcoming neutral hydrogen (HI) surveys (e.g., Verheijen et al., 2009; Duffy et al., 2012) will deliver large datasets. The daily dataflow will be of the order of TBytes and several hundreds of galaxies will be detected. To find and characterize HI objects, automated processing methods must use all of the three-dimensional (3-D) information (two positional dimensions and one spectral dimension) that the surveys make available.

In this context, 3-D visualization techniques provide a powerful tool to inspect the sources under study. In fact, the 3-D view of a galaxy simultaneously presents both its HI distribution and its kinematics providing an immediate overview of the structures and coherence in the data (Oosterloo, 1995; Goodman, 2012; Punzo et al., 2015). In addition, user interaction in the 3-D environment provides capabilities which astronomers can use to quickly analyze complex sources found by automated pipelines (e.g., `Duchamp` and `SoFiA`; Whiting, 2012; Serra et al., 2015). These sources include interacting galaxies, tidal tails, filaments, and stripped galaxies, and the majority will not exceed dimensions greater than $10^8$ voxels.[1]

Performing interactive 3-D rendering (and analysis) of HI sources is computationally affordable using a modern desktop (Punzo et al., 2015). This has stimulated further development of 3-D visualization tools for astronomical purposes. For example, different package developments have recently been undertaken, exploiting: the rendering engine of `Blender`,[2] an open source software for 3-D animations (Taylor, 2015; Kent, 2015; Naiman, 2016); indirect volume rendering[3] available in the Visualization ToolKit, `VTK`,[4] and `Mayavi2`[5] (Vogt et al., 2016); stereoscopic visualization

☆ This code is registered at the ASCL with the code entry ascl:2016ascl.soft11021P.

* Corresponding author.
*E-mail address:* D.Punzo@astro.rug.nl (D. Punzo).

http://dx.doi.org/10.1016/j.ascom.2017.03.004
2213-1337/

[1] Voxels are 3-D pixels.
[2] https://www.blender.org/.
[3] In scientific visualization and computer graphics, volume rendering is a set of techniques used to display a 2-D projection of a 3-D discretely sampled dataset.
[4] http://www.vtk.org/.
[5] http://code.enthought.com/projects/mayavi/.

and 3-D interaction hardware using the gaming engine Unity[6] (Ferrand et al., 2016); and a large-scale, hybrid visualization and supercomputing environment (Vohl et al., 2016). Although the previous packages have introduced 3-D rendering solutions to visualize 3-D astronomical datasets, they do not fully satisfy our visualization requirements (see Section 2.1).

In this paper, we present SlicerAstro[7] (Punzo et al., 2016b), an extension of 3DSlicer[8] (a multi-platform open source software package for visualization and medical image processing; Fedorov et al., 2012), that aims to provide an interactive 3-D visual analytics tool based on traditional 2-D input/output hardware.

In Section 2.1 we describe the design of SlicerAstro. In Section 3 we show how interactive filtering and 3-D visualization can boost the inspection of faint complex sources. In Section 4 we describe the interactive 3-D masking capabilities available in SlicerAstro. In Section 5 we show how 3-D visualization, coupled with interactive modeling, provides additional capabilities helping the discovery and analysis of subtle structures in the 3-D domain. In Section 6 we discuss the efficiency of such visual analytics techniques for helping astronomers in the analysis of complex sources.

## 2. The SlicerAstro environment

An exhaustive review of open-source 3-D visualization packages in Punzo et al. (2015) led to the choice of 3DSlicer as the preferred platform for the development of SlicerAstro. The most important deciding factors included the following:

(I) 3DSlicer is an open-source platform with a Berkeley Software Distribution (BSD) license, which allows for free utilization of the software;
(II) the software has a flexible environment for code development and collaboration;
(III) 3DSlicer has adequate documentation for both developers and users;
(IV) the 3DSlicer software has a large number of active developers;
(V) the 3DSlicer interface already has numerous quantitative features (e.g., data probing, setting fiducial markups[9] and listing their position, 2-D/3-D rulers and calculating statistics in a selected volume).

Several of the medical visualization tools present in 3DSlicer suit the needs of astronomical applications. For example, 3DSlicer optimizes the display layout and the process of navigating through data for parallel two-dimensional visualizations (e.g., movies of channel maps).

In addition, 3DSlicer has been adopted by Kitware[10] as key open-source platform similarly to VTK, ITK[11] and Paraview[12] which Kitware has been supporting for more than 15 years. This guarantees long-term support and future updates of 3DSlicer.

### 2.1. Design

Punzo et al. (2015) analyzed and reviewed the requirements for the visualization of HI in and around galaxies. These include handling the loading and writing of Flexible Image Transport System (FITS) files (Pence et al., 2010), the ability to display astronomical World Coordinates System (WCS; Calabretta and Greisen, 2002; Greisen et al., 2006), interactive 3-D high-quality rendering capabilities (i.e., *graphics processing unit* (GPU)-accelerated ray casting rendering Roth, 1982; Schroeder et al., 2006) and interactive linking between 1-D/2-D/3-D views. Interactive visualization which allows the user to extract quantitative information directly from the visual presentation is also of primary importance: probing the data with a cursor; displaying coordinate axes in the 2-D views; performing 3-D segmentation[13] techniques; linked 1-D/2-D/3-D region of interest (ROI) selection and the ability to calculate statistics (e.g., mean, *rms*, maximum, minimum, etc.) in a specific area or volume. Another requirement is to couple analysis techniques such as interactive smoothing and tilted-ring model fitting to visualization. Therefore, comparative visualization (multiple views, overlaid visualizations, etc.) is fundamental for comparing the raw data with the smoothed version and/or the models. The last requirement is interoperability[14] with virtual observatory (VO) tools (Taylor et al., 2011). Moreover, in order to facilitate collaborative work, the source code must be open, modular, well documented, and well maintained.

The current version of the 3DSlicer software provides several of these capabilities: CPU and GPU rendering based on the VTK, interface optimized for 2-D visualization with a high-level of linking between the 2-D and 3-D views, 2-D and 3-D segmentations techniques, high-level of modularity in the source code, embedded python console in the user interface for fast interaction with the 3DSlicer application programming interface (API),[15] presence of detailed documentation for both users and developers. In addition, we made a number of contributions to the 3DSlicer source: we added more types of units in the 3DSlicer standards and factorized the DataProbe module and widgets that control the 2-D views to allow their customization by 3DSlicer extensions.

In addition, to fulfill the requirements, the following capabilities have to be added:

(I) proper visualization of astronomical data-cubes using the FITS data format;
(II) enabling interactive smoothing in all three dimensions;
(III) interactive 3-D selection of HI sources;
(IV) interactive HI data modeling coupled to visualization;
(V) generation of flux density profiles and histograms of the voxel intensities;
(VI) introduction of the SAMP protocol to enable interoperability with Topcat (Taylor, 2005), and other VO tools and catalogs.

These software capabilities are particular to astronomical applications and, therefore, it is optimal to implement them in an extension of 3DSlicer, i.e. SlicerAstro, rather than in its core.

In the next sections we will discuss the implementation and deployment of such capabilities and use the HI emission in and around WEIN069 (Ramatsoku et al., 2016), a galaxy in a region in the sky where a filament of the Perseus-Pisces Supercluster (PPScl) crosses the plane of the Milky Way, as an example.

6 https://unity3d.com/.
7 https://github.com/Punzo/SlicerAstro.
8 https://www.slicer.org/.
9 A fiducial markup or fiducial is an object placed in the field of view of an imaging system which appears in the image produced, for use as a point of reference or a measure.
10 https://www.kitware.com/.
11 https://itk.org/.
12 http://www.paraview.org/.
13 Image segmentation is the process of partitioning an image into disjoint regions that are uniform with respect to some property.
14 Interoperability is the ability of different information technology systems and software applications to communicate, exchange data, and use the information that has been exchanged.
15 The API is a set of subroutine definitions, protocols, and tools for building application software.

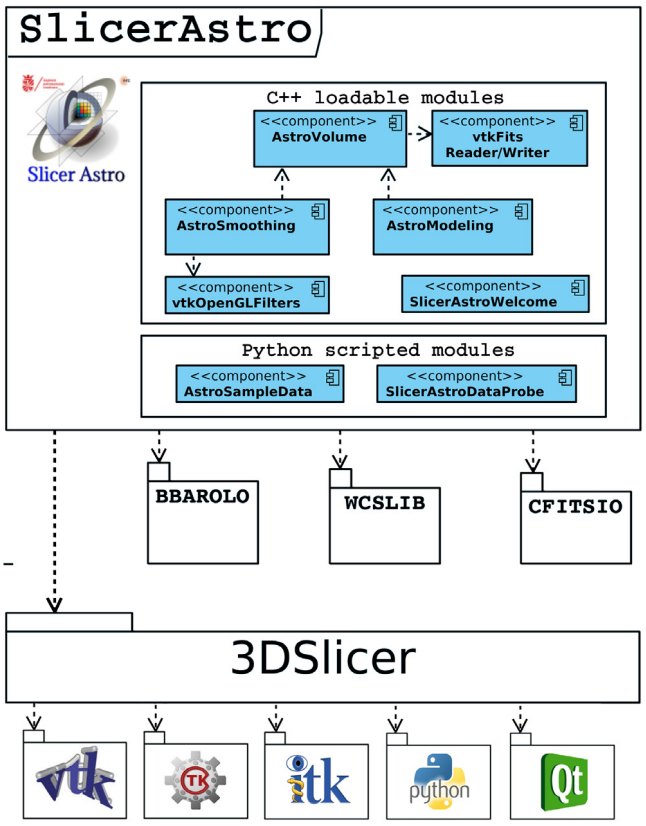


**Fig. 1.** The architecture of `SlicerAstro` is shown in the diagram. The dashed arrows indicate the dependency of a component on another one. The loadable modules are the main components of `SlicerAstro`. The `AstroVolume` module is the core module of `SlicerAstro` and it provides an interface for handling the loading and writing of FITS files, the control of the 2-D and 3-D color transfer functions, and the display of the astronomical world coordinates system (WCS; Calabretta and Greisen, 2002; Greisen et al., 2006). The `AstroSmoothing` and `AstroModeling` modules take care of specific operations (smoothing and modeling respectively), with their own interface widgets for user interaction. The scripted modules have the role of utilities such as downloading sample datasets. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

### 2.2. Implementation

The `3DSlicer` plug-in mechanism enables the rapid development of custom modules in different programming languages and for different levels of integration:

(1) The *command-line interface modules* are standalone executables with a limited input/output argument complexity (simple argument types and no user interaction).
(2) The *loadable modules* are plugins implemented in the `C++` language that are integrated tightly in the `3DSlicer` core software. These modules have access to all other `3DSlicer` core modules and the internals of the application and they can define custom, interactive graphical user interfaces.
(3) The *scripted modules* are written in the `Python` language. These modules can be developed and modified without rebuilding or restarting `3DSlicer` and they have similar access to the application internals as loadable modules.

All objects (volumetric images, surface models, transforms, etc.) in `3DSlicer` are stored in a hierarchical structure of nodes encoded in the Medical Reality Modeling Language (MRML). Each MRML node has its own list of custom attributes that can be used to specify additional characteristics for the data object. This method of storage enables the modules to have access to the MRML tree, allowing new extensions to leverage existing processing and visualization functions without directly interfering with other modules.

In addition, `3DSlicer` and its extensions are developed using a `CMake`-based[16] build system which greatly helps the development, packaging and testing of multi-platform software.

The `SlicerAstro` functionality is implemented as multiple plug-in modules, bundled as one downloadable extension. This modularization makes development and maintenance faster and affordable. Moreover, extensions are built everyday for the nightly build of `3DSlicer` to identify breakage with the core. The architecture of `SlicerAstro` is shown in Fig. 1. `SlicerAstro` uses the `CTK`[17] and `Qt`[18] packages for user interface widgets, and the `VTK` library for the visualization (i.e. 2-D and 3-D rendering). `SlicerAstro` depends also on: `CFITSIO` (Pence, 2010), `WCSLIB` (Calabretta, 2011) and $^{3D}$`Barolo` (Di Teodoro and Fraternali, 2015a). The loadable modules are the main components of `SlicerAstro`, while the scripted modules have the role of utilities such as presenting a welcoming interface and capabilities to download sample datasets. The `AstroVolume` component is the core module (see Section 2.3); `AstroSmoothing` and `AstroModeling` modules take care of specific operations (smoothing and modeling respectively), with their own interface widgets for user interaction (see Sections 3 and 5).

`SlicerAstro` development focuses on HI datasets. Therefore, we currently provide modules which are mainly aimed for the analysis of HI data-cubes. However, `SlicerAstro` can potentially enhance also the inspection of other datasets such as mm/submm molecular line data and optical integral field spectroscopic data. We will elaborate more in the potential of `SlicerAstro` for such datasets in Section 6.

### 2.3. Interface framework

The `AstroVolume` module provides an interface for handling the loading and writing of FITS files, MRML nodes that store the data in the `3DSlicer` object-tree, the display of the WCS and the control of the 2-D and 3-D color transfer functions.

In Fig. 3, we show the implementation of the `3DSlicer` and `SlicerAstro` interface. On the top, the main menu shows several options for loading and writing files (including FITS files) and for editing the `3DSlicer` settings. The data loaded from a FITS file are stored in a `vtkMRMLAstroVolumeNode` object. The instantiated MRML nodes and their properties can be inspected in the `SubjectHierarchy` module (see Fig. 2). The output of source finder pipelines, that is, *object masks*, are loaded as `vtkMRMLAstroLabelMapVolumeNode` objects. These masks are delivered as a data-cube where non-detected voxels in the original data-cube have a value of 0 and detected voxels have an integer value corresponding to the ID of the object they belong to. Due to the complex 3-D nature of the sources (Sancisi et al., 2008) and the noisy character of the data, constructing a fully automated and reliable pipeline is not trivial (Popping et al., 2012) and visualization can help in identifying or rejecting very faint signals (Punzo et al., 2016a). For example, in Fig. 3, `SlicerAstro` shows the visualization of the HI emission in and around WEIN069 and its mask, generated with `SoFiA` (Serra et al., 2014). The data-cube contains three sources, WEIN069 and two companions, each identified as a separate source with its own mask. In addition there is a tidal tail and a very faint filament that is connecting two of the galaxies.

16 https://cmake.org/.
17 http://www.commontk.org.
18 https://www.qt.io/.

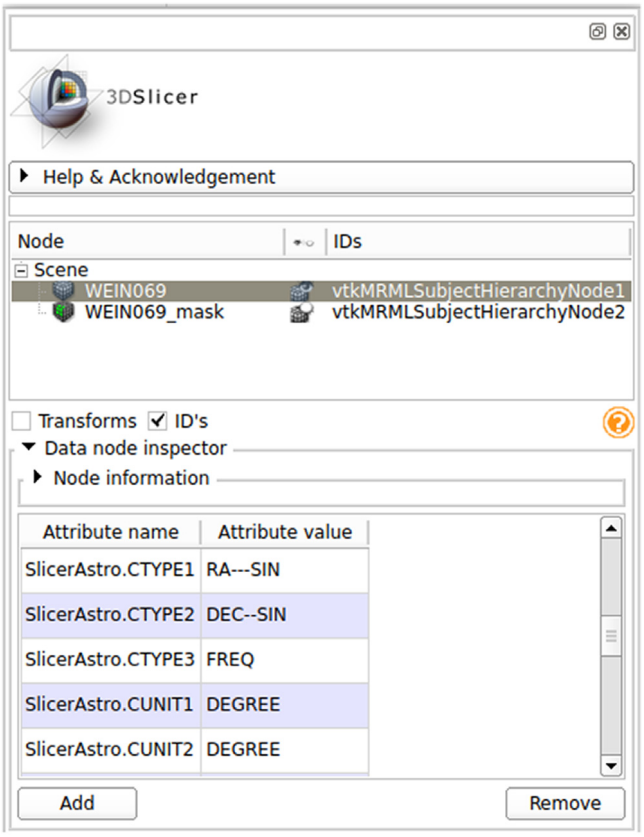


**Fig. 2.** The interface widgets of the `SubjectHierarchy` module. In the top panel, the interface includes the widgets for selecting MRML nodes representing the data-cubes. In the bottom panel, the interface includes a tool to inspect and modify the FITS keywords.

The left panel in Fig. 3, includes the widgets for changing the 2-D and 3-D color transfer functions for `vtkMRMLAstroVolumeNode` objects. In the case of `vtkMRMLAstroLabelMapVolumeNode` objects, volume rendering is not available, but it is possible to use the `MaskVisualization` widget to convert the `vtkMRMLAstroLabelMapVolumeNode` object to a `vtkMRMLSegmentationNode` object. The `vtkMRMLSegmentationNode` class is a core class of `3DSlicer` that handles the display of data segmentation both in the 2-D and 3-D views, as shown in Fig. 3, and they can be overlaid on the data of a `vtkMRMLAstroVolumeNode` object. The segmentation objects can also be interactively modified (see Section 4 for more information) and can be exported for 3-D printing (or imported in `Blender`) by saving them in the STL file format.

Moreover, the layout includes interface widgets to control the display properties (e.g., user interaction to rotate the 3-D view), a window displaying the 3-D World Coordinate and data values of the position of a data probe in the linked 2-D views. The 2-D views also have quantitative World Coordinate axes.

Finally, the MRML infrastructure allows the user to save the session as a *scene*. Reloading such a *scene* restores the session. One can also share interesting visualizations with colleagues using the `Datastore` module. This module saves a bundle with all the necessary information (the data, the visualization views, screenshots and text comments) on the Kitware servers. Other users can download these bundles.

### 2.4. Rendering and user interactions

In `3DSlicer` the visualization representations are rendered with the Visualization Toolkit, `VTK` (Schroeder et al., 2006). In `SlicerAstro` the data are rendered in 3-D with the `VTK` implementation of the ray casting algorithm, a direct volume rendering method (Roth, 1982). Ray casting offers very high-quality results (i.e., free of artifacts), but it is computationally expensive. On the other hand, ray casting is a massively parallel algorithm. On modern desktops the `VTK` GPU implementation offers interactive rendering with a high ($>5$) frame rate (FPS) for data-cubes not exceeding $10^9$ voxels. The use of such high quality rendering is mandatory in our case. In fact, other methods can produce many rendering artifacts in the noisy regions of an HI data-cube. In particular, indirect volume rendering techniques are very ineffective at signal-to-noise-ratio $\lesssim$2, because they have to fit geometries to very noisy data that do not have well-defined closed borders (Punzo et al., 2015).

In `SlicerAstro`, the masks are visualized as segmentations (i.e., `3DSlicer` renders them with indirect volume rendering), because they are supposed to be noise-free by definition.

`3DSlicer` offers several 2-D/3-D linked navigation and interaction tools such as *crosshair*, *fiducials*, *region of interest* (ROI), *ruler* and slice views linked with 3-D views (for more information, we refer to Fedorov et al., 2012 and the `3DSlicer` online documentation[19]). All these features are extremely useful for navigating and probing the data. However, the 3-D visualization paradigm used in `3DSlicer` and `SlicerAstro` is limited by the use of 2-D input and output hardware such as a standard monitor and mouse. An obvious limitation in 3-D is that it is not straightforward to select features or pick positions (i.e., voxels) in the 3-D space in an intuitive manner. Complementary visualization in 2-D (linked to the 3-D one) can partially address these deficiencies.

In `3DSlicer` all the modules are accessible at run-time from the `Python` console (`Python` version 2.7.11 is bundled and delivered together with the `3DSlicer` binaries). Note, however, that of the packages often used in astronomy only `numpy` is part of this bundle. This allows additional flexibility for user-customized visualization and analysis tasks using all `3DSlicer` and `SlicerAstro` capabilities. The `Python` console and automated `Python` scripts are a very powerful tool for interacting with the data itself. Some examples are: accessing the array containing the data, modifying the data and calculating statistics in a region of interest. Moreover, the MRML objects store everything that the user visualizes and changes in the interface. This allows the user to perform the same actions by using `Python` scripts. An example for applying smoothing to a data-cube, performing the rendering, and saving the result as a video is shown in the appendix, Section 7. For example, this framework is extremely useful for creating screenshots and videos for a large number of sources. For more information, we also refer to the online documentation.[20]

## 3. Interactive filtering

Future blind HI surveys will detect a large variety of galaxies with additional complex features such as tails, extra-planar gas, and filaments. These faint structures can be found in nearby, well resolved galaxies and groups of marginally resolved galaxies. They have a very low signal-to-noise ratio ($\sim$1), but are extended over many pixels. Efficiently separating such signals from the noise is not straightforward (Punzo et al., 2016a). Moreover, in the case of Apertif (Verheijen et al., 2009) and ASKAP (Johnston et al., 2008), it is estimated that tens of such sub-cubes will be collected weekly (Punzo et al., 2015). This is a large volume of data, and a coupling between the filtering algorithms and 3-D visualization can enhance the inspection process of large numbers of galaxies and masks provided by source finder algorithms.

[19] https://www.slicer.org/wiki/Documentation/Nightly.
[20] https://github.com/Punzo/SlicerAstro/wiki.

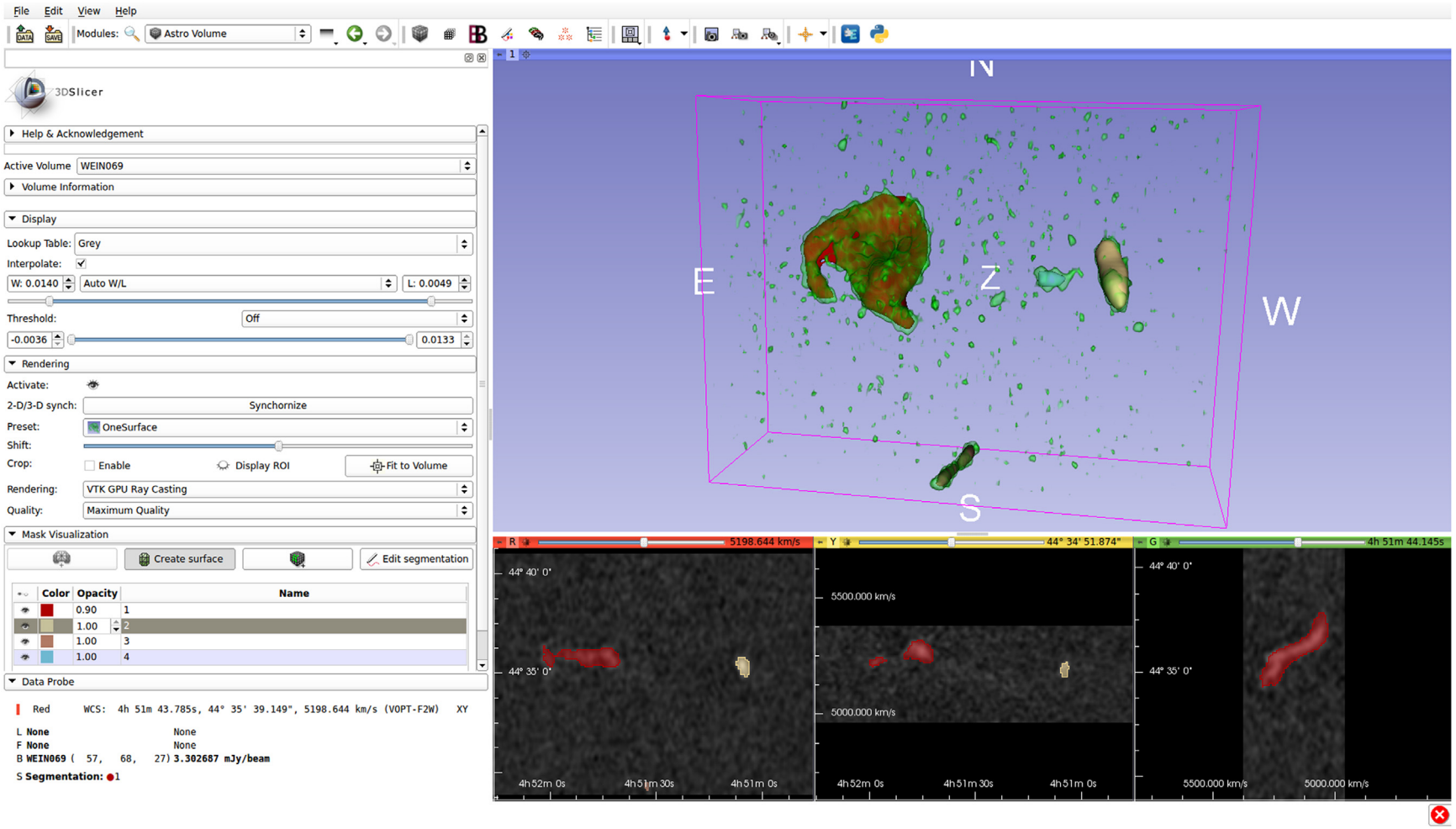


**Fig. 3.** Visualization of the H I emission in and around WEIN069 (Ramatsoku et al., 2016) and its mask, generated with `SoFiA`, in `SlicerAstro`. The data-cube contains three sources, i.e., WEIN069 and two companion galaxies, a tidal tail and a very faint filament that connects two galaxies. In the 3-D view the data are rendered in green and highlighted at an intensity level equal to 3 times the root mean square (*rms*) noise. The colored segmentations represent the mask and each color refers to a specific source ID as shown in the table widget in the left panel. The left panel includes also interface widgets to control the 2-D and 3-D color transfer functions and a data probe window. Quantitative information such as WCS coordinates are shown both in the data probe window and along the axes in the 2-D views. The white labels in the 3-D view represent the four cardinal directions (*N*, *S*, *E*, *W*) and the line-of-sight direction (representing frequency/wavelength or velocity/redshift, z, hence the symbol *Z*). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

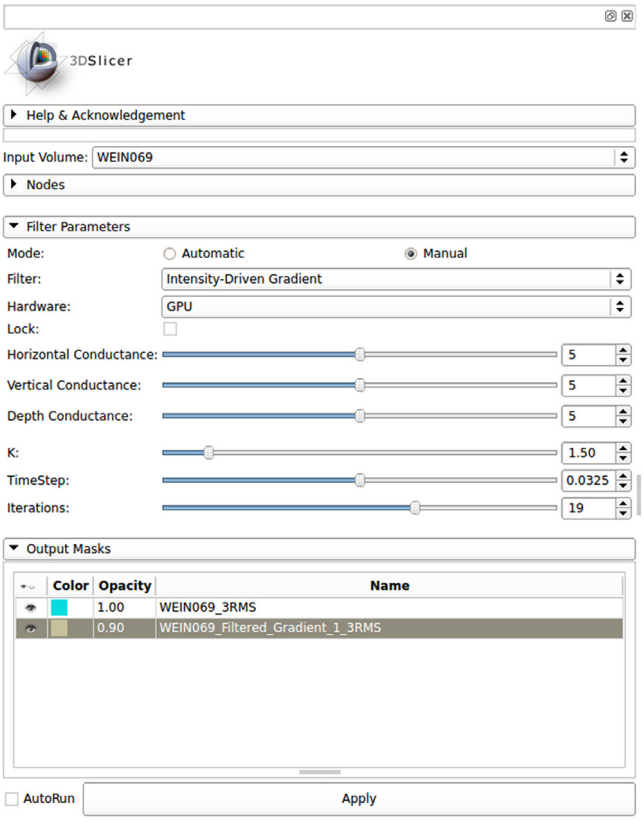


**Fig. 4.** The interface widgets of the `AstroSmoothing` module. The interface includes a widget for changing the input parameters for the smoothing and a table showing the output segmentations generated after the smoothing process.

In Fig. 4, we show the interface of the `AstroSmoothing` module. This includes the widgets for changing the input (such as the filter choice, the computational hardware and the smoothing parameters) and visualizing the output segmentation objects generated by the smoothing process.

Three filters are currently available in the `AstroSmoothing` module:

(1) *Box* (or *mean*) filter: it replaces each pixel value in the volume with the mean value of its neighbors including the value of the pixel itself. Both isotropic (where the 3-D kernel has the same dimensions along all the three axes) and anisotropic implementations are available.
(2) *Gaussian*: it applies a 3-D convolution operator to the volume. It preserves the shape of the objects better than the box filter, but is computationally more expensive. Both isotropic and anisotropic (the kernel can have different dimensions along the 3 axes and it can be rotated) implementations are available.
(3) *Intensity-Driven Gradient*: it uses an adaptive diffusion process (i.e., operates on the differences between neighboring pixels, rather than on the pixel values directly). High signal-to-noise regions are unaffected, but low signal-to-noise, extended, regions are enhanced.

These algorithms are available in `SlicerAstro` as parallelized implementations on both CPU and GPU hardware, offering interactive performance when processing data-cubes of dimensions up to $10^7$ voxels and very fast performance ($< 3.5$ sec) for larger ones (up to $10^8$ voxels). The intensity-driven gradient filter, due to its adaptive characteristics, is the optimal choice for HI data (Punzo et al., 2016a). Therefore, it is the default method when the automatic mode has been chosen. This algorithm preserves the detailed structure of the signal with high signal-to-noise ratio ($> 3$) at the highest resolution, while smoothing only the faint part of the signal (signal-to-noise ratio $< 3$). For more information regarding the filters and their performance, default parameters, advantages and disadvantages, we refer to Punzo et al. (2016).

After running the smoothing process, `SlicerAstro` displays automatically a comparative layout composed of two 3-D views, one of the original data (top left panel) and one of the filtered data (top right panel), and three 2-D views of the original data (lower three panels) for the inspection of the data, as shown in Fig. 5. In this particular case, the 3-D visualization of the filtered data highlights immediately the presence of the faint filament between two galaxies that was hardly visible in the original version of the data. Moreover, the coupling between 3-D visualization and interactive filtering enables a user to manually and iteratively search the best smoothing parameters for maximally enhancing the local signal-to-noise ratio of the very faint component.

We will show in the next section how any segmentations generated by the smoothing module (or converted from loaded masks as shown in Section 2.3) can be interactively modified in the `SegmentationEditor` module of `3DSlicer`.

## 4. Interactive 3-D masking

Twenty years ago, Norris (1994) pointed out that the main challenge for visualizing astronomical data in 3-D was to develop a 3-D visualization tool with interactive capabilities for data inspection and with interactive and quantitative analysis capabilities. Nowadays, 3-D interactive visualization is achievable thanks to the use of massively parallel hardware such as GPUs (see Section 2.4). On the other hand, volumetric data interaction tools (e.g., picking a voxel or selecting a region of interest in 3-D) are necessary for performing data analysis in a 3-D environment.

An optimized 3-D selection technique, based on 2-D input/output hardware, is still a partially open-problem, not only in astronomy, but also in medical visualization and computer science. Moreover, the optimal selection technique highly depends on the specifications of the use case. Our requirements for a 3-D selection tool are interactivity and a minimal number of user-operations for achieving the selection (i.e., user-friendliness). For a review of the state-of-the-art 3-D selection algorithms we refer to Yu et al. (2012) and Yu et al. (2016).

For our application, we opt for the `CloudLasso` technique (Yu et al., 2012). The `CloudLasso`, operated on grid data, is based on the application of the Marching Cubes (MC) algorithm (Wyvill et al., 1986; Lorensen and Cline, 1987) for the identification of regions of voxels with signal inside a user-drawn lasso; i.e., `CloudLasso` is a lasso-constrained Marching Cubes method. The `CloudLasso` method allows us to spatially select structures with high signal-to-noise ratio ($> 3$) within a lasso region. Even if disjoint structures lie visually behind one another, they can be all selected without including the noisy regions in between. For operating the intended selection, a threshold has to be chosen. The `CloudLasso` algorithm, therefore, comprises the following two steps:

(A) Volume selection: the subset of the volume where the intensities of the voxels exceed a threshold is computed using Marching Cubes.
(B) Threshold tuning: interactive adjustment of the intensity threshold.

The selection operated by the `CloudLasso` algorithm is highly dependent on the user interactions. In fact, the user has to choose the orientation of the camera which gives the best view of the data, perform the 2-D selection on the screen and select the optimal

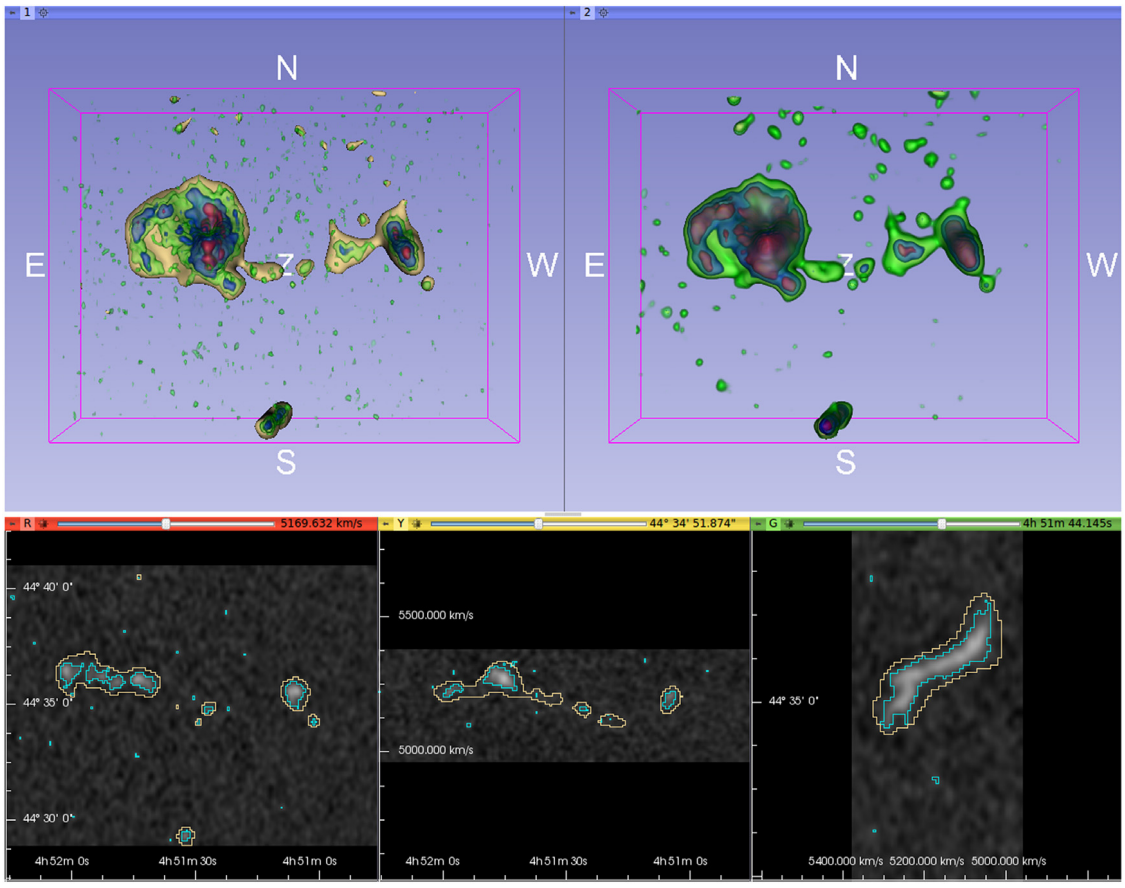


**Fig. 5.** A comparative layout of the output generated by the `AstroSmoothing` module is shown. The layout is composed of two 3-D views and three 2-D views. In the left 3-D view and the 2-D views the data are shown. In the right 3-D view the filtered version of the data is shown. The data are rendered in different colors that highlight the data at different intensity levels: green, blue and red correspond to 3, 7 and 15 times the *rms* noise respectively. The colored segmentations represent masks automatically calculated by the filtering algorithm. The light blue and yellow segmentations (visualized as contour plots in the 2-D views) are a 3 *rms* thresholding of the input data and the filtered data, respectively. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

threshold. Therefore, the user experience and knowledge of the data are of primary importance in the `CloudLasso` selection. The technique allows the user to perform a refinement of the selection interactively by tuning the threshold or through Boolean Operations.

Although the threshold tuning step can be improved or replaced by more complex techniques to identify and classify the signal in the selection, the `CloudLasso` technique is the most reliable choice in our case, because it leaves any classification to the user (leveraging his/her knowledge about the data). For example, connectivity operators (Heijmans, 1999) can be applied after the thresholding to distinguish the various islands of signal and to label them with IDs. Moreover, `MAX-TREE` algorithms (Moschini et al., 2014) can automatically provide a tree classification of the data. Finally, more advanced selection techniques can be employed (e.g., `Cast` selections Yu et al., 2016). The common element in these techniques is the idea to classify (in different ways) the information in the data. However, due to the very noisy nature of HI data, separating the HI signal from the noise is not trivial (Punzo et al., 2016a) and, therefore, it is quite challenging to build an automated algorithm to classify the data (see also Giese et al., 2016).

In the `SegmentationEditor` module of `3DSlicer` we implemented an `AstroCloudLasso` segmentation editing capability, optimized and specialized for the selection of HI data. A segmentation editor is a `3DSlicer` tool that enables user interaction with the data and creation/modification of segmentations both in the 2-D and 3-D views. In Fig. 6 we show the interface widgets of the `AstroCloudLasso` segmentation editor. The default value of the threshold is set to 3 times the *rms* value of the data-cube under study. In Fig. 7, we show how the selection procedure is performed in a 3-D view of `3DSlicer` and the results for each segment are shown (i.e., we repeated the selection procedure four times). The tool can also perform 2-D selections (on the 2-D views), it can erase the segment under the selection (both in the 2-D and 3-D views) if the erase mode has been enabled, and it can interactively adjust the selection of the intensity threshold if the automatic updating mode has been enabled.

The `AstroCloudLasso` segmentation editor effect can be used for two applications:

(A) interactively modify a mask as shown in Fig. 7 (note that `SlicerAstro` can save the new mask as a FITS file). This framework can be used as a modification tool of the masks generated from source finder pipelines.
(B) selecting regions of interest for further analysis.

In the next section, we will apply the segmentation as a selection for operating tilted-ring modeling in the region of interest.

## 5. Interactive modeling

In the case of HI in galaxies one can extract additional information from fitting the observations with a so called *tilted-ring* model (Warner et al., 1973). Such a model describes the observed HI distribution of the galaxy as a set of concentric, inclined, and rotating rings. Each ring is characterized by the following parameters: the center of the spatial coordinates, and the systemic

velocity, rotation velocity, velocity dispersion, inclination, position angle as a function of the galactocentric radius. A model is specified by a set of ring (radially varying) parameters plus a set of global parameters (e.g., ring width).

To compute a model the rings are populated with an ensemble of HI clouds using a Monte Carlo method. The cloud ensembles are integrated along each line-of-sight in the data-cube and convolved with a 3D-Gaussian representing the properties of the observing beam and the resolution in the frequency domain.

A tilted-ring model is necessarily an oversimplification of the HI distribution inside galaxies. When the orbits are significantly non-circular, for example in the presence of a bar (Bosma, 1978), the tilted-ring model will not be able to represent the data accurately. Furthermore, there is a degree of degeneracy between some of the ring parameters (e.g., inclination, position angle and rotational velocity). In many cases, however, the tilted-ring model serves as a good approximation and can provide a deeper understanding of the kinematics and morphology of a galaxy, including asymmetries in surface density and velocity, the presence of gas at anomalous velocities, of extra-planar gas, of inflows and outflows, etc. It is for example rather easy to locate the presence of extra-planar gas once the symmetric and regularly rotating disk is modeled (see Section 5.3).

It is, therefore, very useful to add model fitting capabilities to a visual analytics tool for HI data. Such a capability enables an interactive comparison between the data and models so that the quality of the model can be assessed interactively. This is possible by embedding the model routine in the visualization interface. This will also enable interactive tuning of the model parameters using the visualization interface.

Modern 3-D tilted-ring modeling software can generate symmetric models that reproduce the data with a minimal user input and interaction (Di Teodoro and Fraternali, 2015b; Kamphuis et al., 2015a). In the next section we will briefly review such software libraries. We will also describe the integration of one of them in `SlicerAstro` and show how it provides additional capabilities for the detection and analysis of subtle structures in the 3-D domain. Two use cases will be investigated in Sections 5.2 and 5.3: using the 3-D selection tool (shown in Section 4) to perform the tilted-ring model fitting only in a region of interest (i.e., excluding non-symmetric, non-regular, HI structures such as tidal tails) and using the symmetrical properties of automated tilted-ring model fitting to locate extra-planar gas.

### 5.1. Requirements

Tilted-ring model fitting is rather complex. Therefore we chose to rely on an external state-of-the-art package rather than designing a new one. In order to be able to wrap an external model fitting package into `SlicerAstro` the following requirements can be formulated:

(I) 3-D model fitting capabilities;
(II) automatic estimation of the initial parameters for the fitting;
(III) parallelization on CPU and/or GPU for fast execution;
(IV) developed as a modern, modular source code, preferably C++.

Currently, two software packages are available: $^{3D}$`Barolo` (Di Teodoro and Fraternali, 2015a), an automated procedure which fits tilted-ring models to HI data-cubes; and `Fat` (Kamphuis et al., 2015b), a similar package built on top of `TiRiFic` (Józsa et al., 2012, 2007).

As shown in Table 1, $^{3D}$`Barolo` is currently our optimal choice because, being developed in C++, it satisfies the fourth requirement. The fourth requirement ensures high performance, a rather simple and smooth integration process, and long-term maintainability. On the contrary, `Fat` has been developed in `IDL` which introduce with it several compiling and linking issues (also, the license of `IDL` is not compatible the open-source BSD license of `SlicerAstro`).

**Table 1**
Requirements for the model fitting external library, described in Section 5. We compare two software packages: `TiRiFic/Fat` (Józsa et al., 2007; Kamphuis et al., 2015a) and $^{3D}$`Barolo` (Di Teodoro and Fraternali, 2015b).

| Requirements | `TiRiFic/Fat` | $^{3D}$`Barolo` |
|---|---|---|
| I: fitting capabilities | ✓ | ✓ |
| II: parameter estimation | ✓ | ✓ |
| III: CPU/GPU parallelization | ✗ | ✗ |
| IV: C++ Development | ✗ | ✓ |

Although both state-of-the-art packages lack of a concrete parallelization strategy, they still have sufficiently fast performance for fitting the data for sources in data volumes up to $10^6$ voxels and for generating a single model for sources representing a data volume of up to $10^8$ voxels (see Section 5.2). However, user interaction with the modeling routine in the `AstroModeling` module will greatly benefit from a parallel 3-D tilted-ring model fitting source code.

In the following use cases[21] we will show how the `AstroModeling` module, exploiting 3-D interactive visualization and $^{3D}$`Barolo`, helps in the modeling and analysis of complex sources.

### 5.2. Use case A: analysis of sources with tidal tails

Although $^{3D}$`Barolo` is a powerful fitting routine, it is designed to fit models of galaxies with a thin regularly rotating disk. Therefore, $^{3D}$`Barolo` (or other current tilted-ring modeling algorithms) cannot recognize, for example, tidal tail structures and separate them from the central regularly rotating body of the galaxy.

In this section, we show how to use the `AstroModeling` module for the manual quality control of the models. This framework enhances the analysis of gravitational perturbed galaxies such as WEIN069. In fact, the 3-D selection tool described in Section 4 can be used to select a region of interest for which $^{3D}$`Barolo` provides the best results. For example, in the case of WEIN069 the user can separate the two kinematic components, i.e., the regularly rotating disk and the tidal tail, and perform the calculations only on the central disk.

Figs. 8 and 9 show in blue a selection of the central body of WEIN069 and the parameters chosen for running the fitting routine in $^{3D}$`Barolo`. The fitting results are shown in Figs. 9 and 10: the yellow segmentation, in the 2-D and 3-D views, represents the model, while the green rendering, in the 3-D view, represents the data. The visualization highlights the model and the data at the intensity level chosen in the *contour level* interface widget (in this case three times the value of the *rms* noise in the input data-cube).

The overlay of the segmentation of the model on the 3-D rendering of the data facilitates the inspection of the model. In the case of Fig. 10, the horizontal and vertical axes of the third 3-D view (middle left panel) are the velocity and the declination dimensions, respectively. It is immediately clear that the rotation curve of the

[21] In software and systems engineering, a use case is a list of actions or event steps, typically defining the interactions between a role (known in the Unified Modeling Language as an actor) and a system, to achieve a goal. The actor can be a human or other external system.

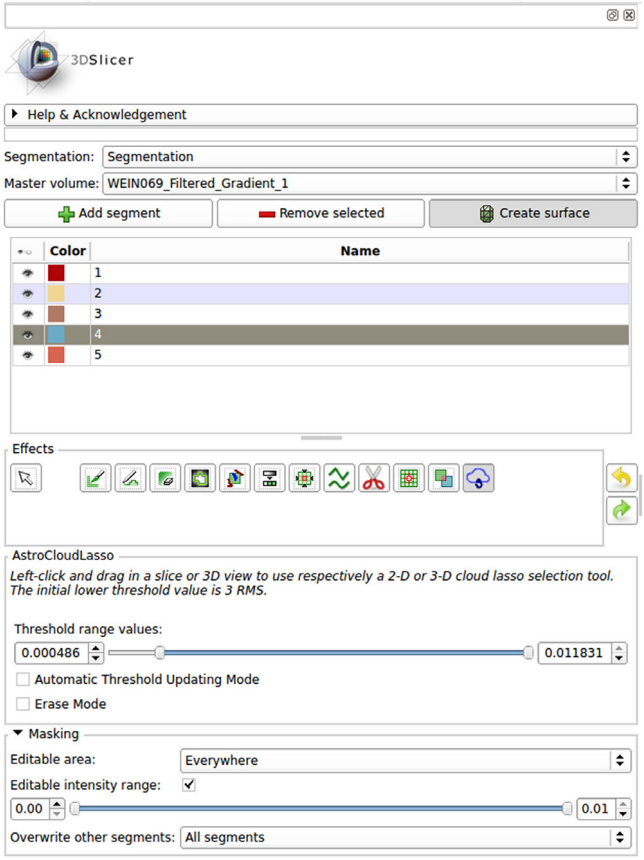


**Fig. 6.** The interface widgets of the `SegmentationEditor` module and `AstroCloudLasso` segmentation effects. The interface includes widgets for selecting the segment to modify, the segmentation editor effect and the parameters relative to the chosen effect.

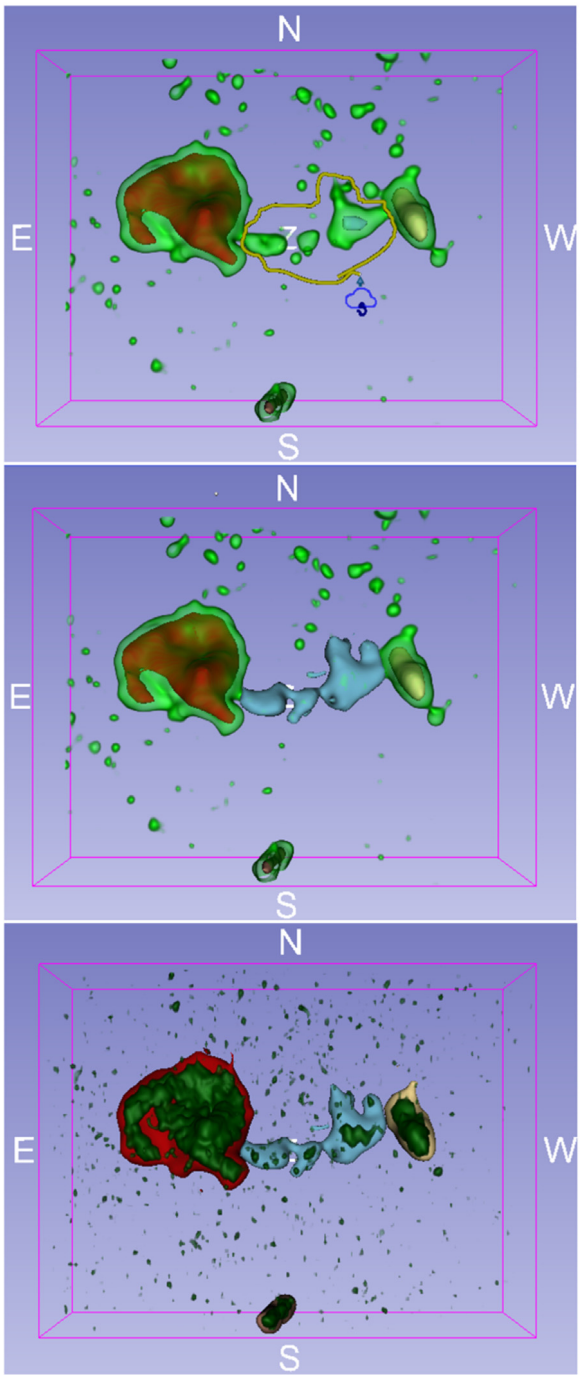


**Fig. 7.** Usage of the `AstroCloudLasso` segmentation editor effect in 3-D. A smoothed version of the WEIN069 data is rendered in green in the top and middle panels. In the bottom panel, the original version of WEIN069 data is rendered. The three renderings highlight the data at the intensity level equal 3 times the *rms*. In the top panel, the colored segmentations represent the mask shown in Fig. 3. In order to visualize clearly both the data and the mask, the data are rendered with a higher opacity in the bottom panel compared to the upper panels. Similarly, the opacity of the segmentations is decreased. The `AstroCloudLasso` selection tool is visualized as a yellow tube drawn by the user with the 2-D cursor indicated by the blue cloud. This tool computes a selection in 3-D space from the 2-D user-selection. It builds a closed surface at the value of the intensity level specified in the settings widget (Fig. 6) and visualizes the modified segment as shown in the middle panel. In the bottom panel, we show all the modified segments.

model in the inner rings does not rise fast enough. User interactions with the 3-D view such as camera zooming and rotation enhance the 3-D perspective giving an even better overview of the differences. On the other hand, for checking the data pixel by pixel (e.g., for data probing) it is better to use a two-dimensional representation.

Finally, the `AstroModeling` module provides a table widget in the interface (see Fig. 9) that can be used to refine the model and update the visualization. All the ring parameters of the model are available in the table. The refining process of the output model is crucial. In fact, the tilted-ring model fitting is a process with a high degree of degeneracy between the parameters, and the fitting results strongly depend on the value of the initial parameters, especially for the inclination. Therefore, the models must be carefully checked, compared with the data, and refined.

The computational time needed by $^{3D}$`Barolo` to fit the data depends on several factors: the number of voxels, the number of rings of the model and the *goodness* (i.e., whether the error in the estimate is $<10\%$) of the initial parameters. However, it is not possible to provide unbiased benchmarks for a fitting routine (i.e., the performance highly depends on the input parameters). To give an example, the time for fitting a source extended up to $10^6$ voxels, using 20 rings and having a reliable estimation of the input parameters is $\sim$ 2.5 min (exploiting 1 CPU core at 2.60 GHz). On the other hand, once the fitting has been performed, recalculating a single new model with the same size in voxels and number of rings takes less than 2 s (the process includes getting the model from $^{3D}$`Barolo` and creating the 3-D segmentation in `SlicerAstro` as well). The computational complexity of this second step is $O(Nr)$, where $N$ is the number of voxels of the source and $r$ the number of rings. Despite the fact that the framework is not interactive, it

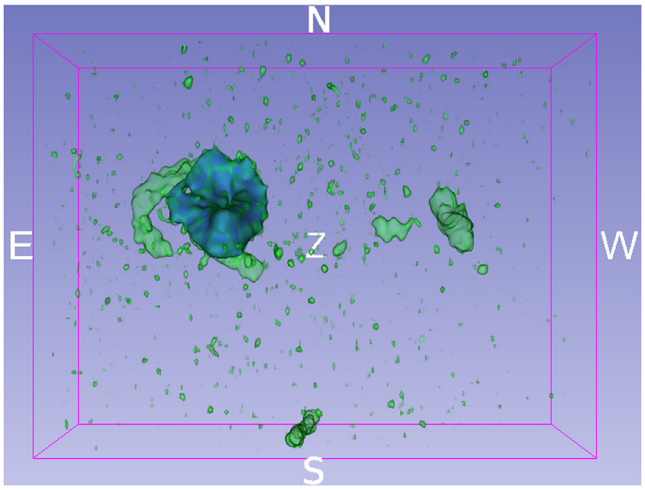


**Fig. 8.** 3-D view of WEIN069 rendered in green. It highlights the data at the intensity level equal 3 times the *rms*. The blue segmentation represents a 3-D selection (see Section 4). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

is still fast enough to provide a powerful tool to refine models and compare them with the data.

### 5.3. Use case B: finding anomalous velocity gas

It has been demonstrated that the gas distribution of some spiral galaxies (e.g., NGC2403; Fraternali et al., 2002) is not composed of just a cold *regular* thin disk. Stellar winds and supernovae can produce extra-planar gas (e.g., a galactic fountain; Bregman, 1980). In this case, modeling is used to constrain the 3-D structure and kinematics of the extra-planar gas which is visible in the data as a faint kinematic component in addition to the disk.

The `AstroModeling` module uses the output model of $^{3D}$`Barolo` for visually highlighting the different components in the data-cube. After visualizing the model of the symmetric cold thin disk as a segmentation, it is immediately possible to locate any unusual features in the data-cube of interest and already get an idea of their properties, thus directing further modeling. For example, a model of the extra-planar gas above or below the disk with a slower rotation and a vertical motion provides quantitative information about the rotation and the infall velocity of such gas.

In Fig. 11 we show as an example the analysis that we performed on NGC2403. The input parameters for the fitting have not been edited, therefore $^{3D}$`Barolo` performed an automatic estimation of the initial parameters. In the 3-D view `SlicerAstro` illustrates the data of the NGC2403 observations rendered in green and the tilted-ring model generated with $^{3D}$`Barolo` as a white segmentation. The white segmentation is rendered with the maximum opacity in order to obscure all the data that have been fitted by $^{3D}$`Barolo`. This combination gives an immediate overview of the extra-planar gas present in the NGC2403 observations (Fraternali et al., 2002). Since $^{3D}$`Barolo` mostly fits the symmetric regularly rotating part of the galaxy, it therefore is a powerful tool for locating anomalous features in the data, such as the extra-planar gas in NGC2403.

## 6. Summary

`SlicerAstro` is an open-source project and its binaries are currently available in the extensions manager of `3DSlicer`.[22] The

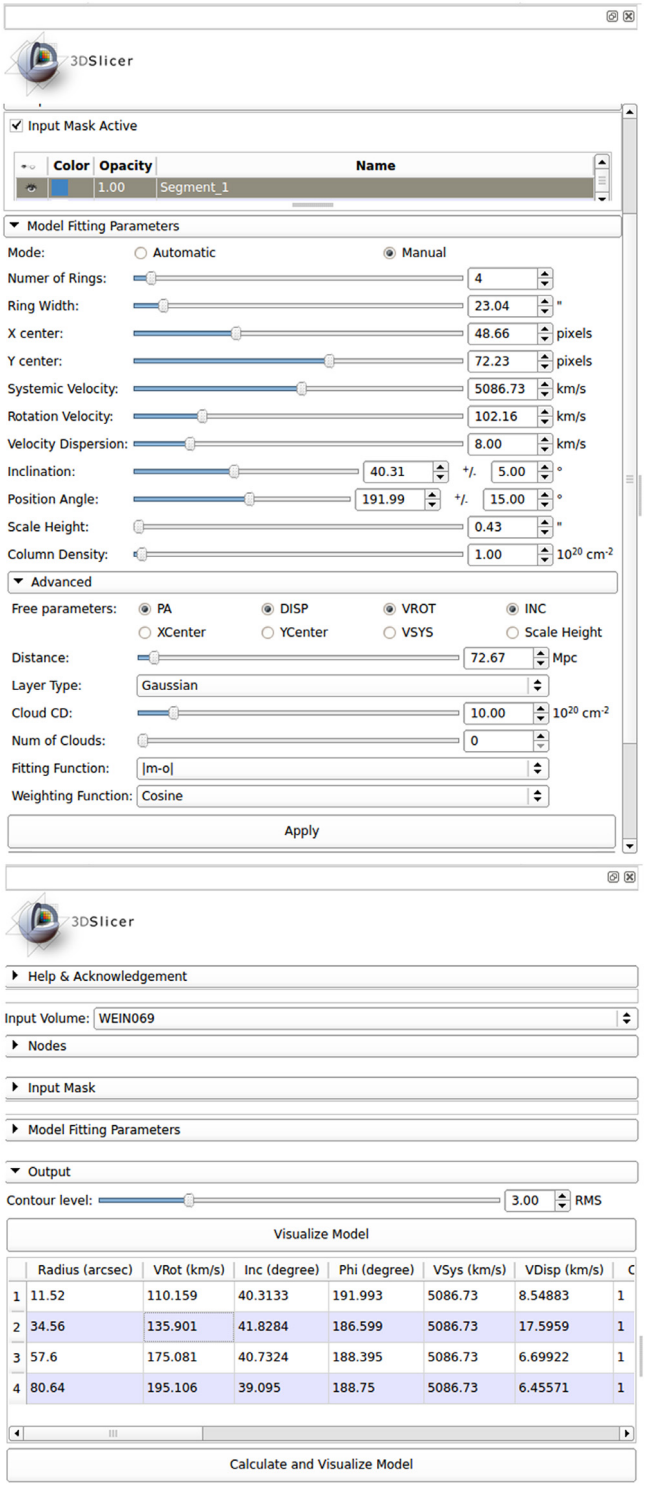


**Fig. 9.** The interface widgets of the `AstroModeling` module. In the top panel, the interface includes the widgets for selecting a segment that will be used as mask for the modeling and the input parameters for the model fitting. In manual mode one can specify the fitting method (i.e. the kind of residuals between the model and the data to be minimized) and the weighting function, i.e. the weights (as a function of angle from the minor axis) given to the residuals before fitting. These weights correct for the effect that the line of sight component of the circular velocity in a rotating disk approaches zero when approaching the minor axis. (for more information see Di Teodoro and Fraternali, 2015a). In the bottom panel, the interface includes a *Contour level* widget to choose the threshold value for the segmentation of the model, an editable table with the parameters of the rings of the output model (see Fig. 10), and push buttons for updating the model.

[22] The user guide is available at the following link: https://github.com/Punzo/SlicerAstro/wiki#get-slicerastro.

novelty of SlicerAstro over traditional astronomical viewers is the use of 3-D interactive tools for the visualization and analysis of HI in and around galaxies. SlicerAstro has been designed with a strong, stable and modular C++ core, but it can be used also via Python scripting, allowing great flexibility for user-customized visualization and analysis tasks (see Section 2.1).

Although SlicerAstro is still under development, it already offers several new qualitative and quantitative visualization and analysis features for the inspection of HI and other spectral line data. The overall advantage of SlicerAstro compared to traditional viewers (e.g., KARMA, Casaviewer and VISIONS; Gooch, 1996; McMullin et al., 2007; van der Hulst et al., 1992) is that it bundles analytical operations such as smoothing and modeling with the visualization. These visual analytics techniques enhance the visualization itself. More important is in our view the interactivity offered by SlicerAstro. Interactivity is key to enhancing the inspection and analysis of complex datasets. In fact, precisely the interactive and coupled 3-D/2-D visualization aspects (e.g., volume rendering, navigation, changing color/opacity function, selecting regions of interest in the 3-D space) which are (partially) missing in the traditional tools and which disclose powerful visual analytics capabilities.

In Section 2.3, we presented the main module, AstroVolume. This module provides: a user interface for loading and writing FITS files; the display of astronomical World Coordinates; control of 2-D and 3-D color transfer functions; MRML nodes for storing the data; and data conversion tools for masks and 3DSlicer segmentation objects. Fig. 3 showed how 3-D visualization gives an immediate overview of the HI emission in and around WEIN069 (i.e. three interacting galaxies and a tidal tail Weinberger et al., 1995; Ramatsoku et al., 2016) and the mask generated by automated source finder pipelines such as SoFiA or Duchamp. 3-D visualization highly enhances and accelerates the inspection of the data and of the masks, allowing efficient manual quality control of part (i.e., complex galaxies or groups of galaxies) of the large datasets that will be provided by the SKA precursors.

In addition, we presented the AstroSmoothing module in Section 3. Fig. 5 showed the filtered version of WEIN069, obtained with a newly implemented intensity-driven gradient filter (Punzo et al., 2015). The 3-D visualization highlights immediately the presence of a faint filament between two galaxies that was hardly visible in the original data-cube. The coupling between the interactive smoothing algorithms (available in the parallelized version both on CPUs and GPUs) and the 3-D visualization allows for a detailed inspection of the result and a manual, iterative, search for the best smoothing parameters for maximally enhancing the local signal-to-noise ratio of the very faint signal.

Finally, we introduced the AstroCloudLasso selection tool in Section 4. This is a 3-D interactive selection tool (Yu et al., 2012), optimized for HI data, added by Slicer-Astro in the SegmentationEditor of 3DSlicer. We showed how to use this tool to create and modify segmentation objects in the 3-D views (Fig. 7). The tool can be also used in the 2-D views for a 2-D selection. CloudLasso is an intuitive and efficient 3-D selection method, which is crucial for allowing manual modification of masks generated automatically by source finder pipelines (e.g., adding very faint signal missed by automated pipelines). A second application of the tool is to select a region of interest (ROI). The ROI can be successively used to perform calculations, such as tilted-ring model fitting, in the selection.

In Section 5, we demonstrated that 3-D visualization, coupled to modeling, provides additional capabilities helping the discovery and analysis of subtle structures in the 3-D domain. We integrated $^{3D}$ Barolo, a tilted-ring model fitting package, in SlicerAstro, providing an interface to set the input parameters for the fitting (Fig. 8). Moreover, the interface includes a widget for editing the ring parameters of the output model, for recalculating and visualizing the model on top of the data. We also showed that 3-D is a powerful tool not only to provide a region of interest for the calculations, but also for the inspection of the model (Fig. 10) and the data not fitted by the model (e.g. extra-planar gas in NGC2403, Fig. 11).

The efficiency and the effectiveness of the visual analytics techniques implemented in SlicerAstro have been tested. Quantifying the results for the efficiency of the modeling capabilities in SlicerAstro is not straightforward as the speed depends on the size of the data-cube and on the input parameters. However, even in the case of a moderately large and well resolved object such as NGC2403 (dimension $\sim 1.4 \times 10^6$ voxels), the model fitting is performed in less than 2 min. In addition, modifying manually the parameters of the output model is interactive. The 3-D smoothing algorithms in SlicerAstro have interactive performance. An adaptive smoothing operation on NGC2403 data-cube is performed in less than 0.1 s exploiting the computing power of a GPU (i.e., GeForce GTX860M). The main advantage of SlicerAstro is that the combination of these smoothing and modeling capabilities with *interactive* 3-D visualization provides an immediate overview of all the coherent 3-D structures of the data, masks and models. This is very powerful and it definitely does increase the effectiveness of the visualization and, thus, the efficiency of the astronomical users in the manual exploration of many datasets.

We conclude that interactive 3-D quantitative and comparative visualization, 3-D user interaction and analysis capabilities available in SlicerAstro form an effective new tool that will boost, in terms both of efficiency and quality, the analysis of complex sources in the context of large data-flows that will be provided by the SKA precursors. However, in order to fulfill all the visualization requirements defined in the Section 2.1 (and extensively discussed in Punzo et al., 2015), some quantitative features still have to be incorporated in SlicerAstro. For example, a tool displaying the histogram of the flux intensities of the data-cube will greatly help the user in setting the 2-D color function. The capability to display flux density profiles (i.e. linked 1-D visualization) is also necessary, especially when dealing with unresolved sources. In addition, capabilities for overlaying (in an automated way) other datasets (including datasets with different grids and projection systems) will enhance the inspection of multi-wave bands datasets and are under development. Furthermore, a dedicated tool in SlicerAstro for easily displaying position–velocity (P–V) diagrams will improve the inspection and comparison of models. Specialized analysis tasks on 3-D selections (e.g., calculating statistics, moment maps, etc., in regions of interest) can be performed by running scripts in the 3DSlicer Python console. On the other hand, customized quantitative tasks can also be added, as core modules, in SlicerAstro, similar to the implementation of the AstroModeling module. These capabilities will be integrated in future updates of SlicerAstro.

The implementation of VO interoperability and the advantages of such connectivity will be considered and analyzed further in the case of SlicerAstro. In fact, the SAMP protocol and the FITS format are no longer globally accepted standards (Mink, 2015; Mink et al., 2015; Thomas et al., 2015). Other scientific fields such as medical imaging can provide insights on how to improve the astronomical standards. For example, the Digital Imaging and Communications in Medicine (DICOM) (Mildenberger et al., 2002) protocol is a remarkable example of a standard for handling, storing, printing, and transmitting information in medical imaging. DICOM includes a file format definition and a network communications protocol universally accepted and used by the medical scientific community.

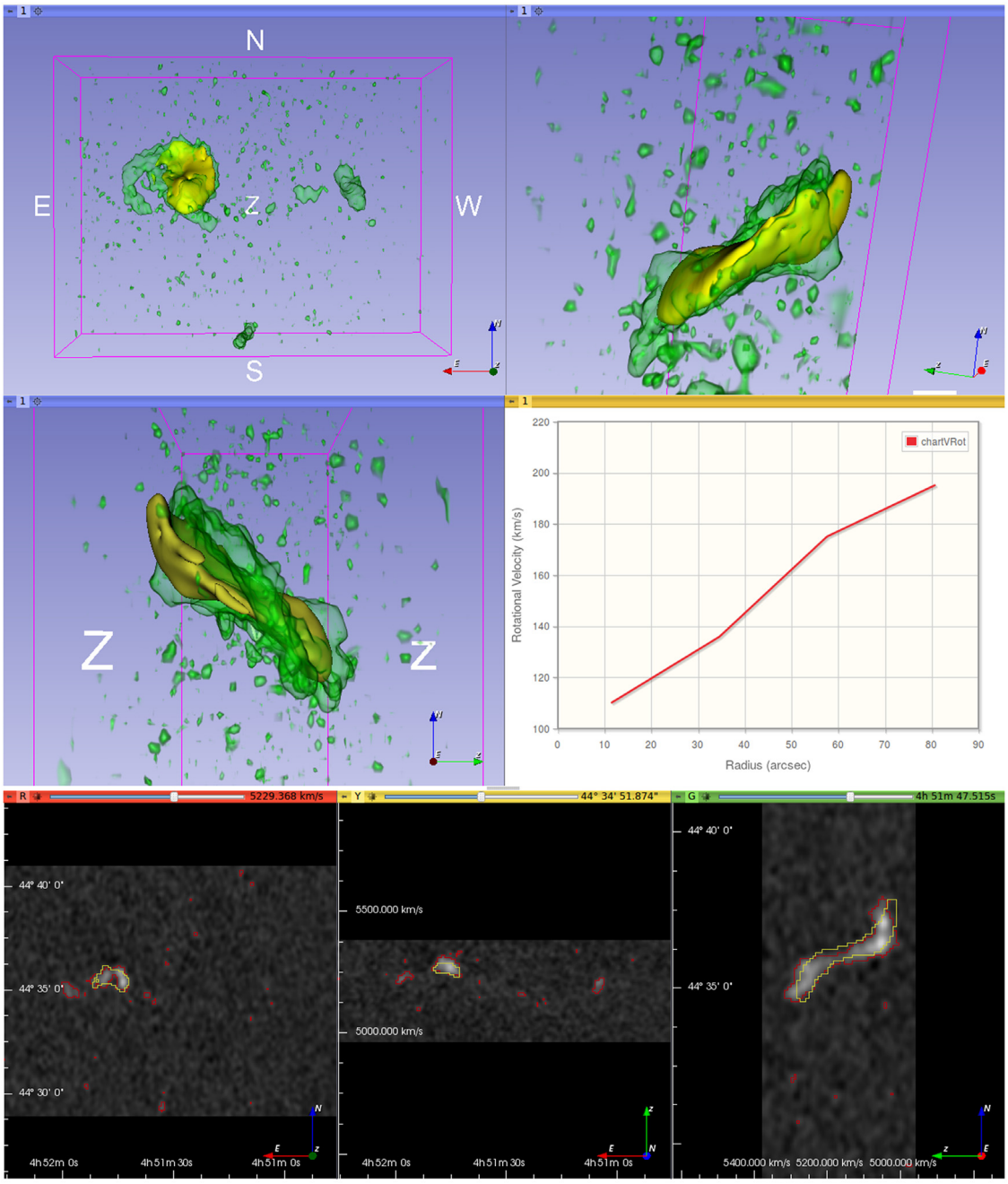


**Fig. 10.** Comparative layout of the output generated by the `AstroModeling` module. The layout is composed of three 3-D views, three 2-D views and a chart view. The WEIN069 data are shown in the 3-D view and the 2-D views. In the 3-D views the data are rendered in green. Each 3-D view has a different camera position: top-left, viewing direction along the velocity axis; top-right, the origin of the camera is in the center of the data-cube and the view is parallel to the geometrical major axis of the galaxy; middle-left, the viewing direction is along the RA axis. The white labels *z* and *Z* indicate the line of sight velocity (or redshift z) direction (i.e., increasing from *z* to *Z*). The green arrow in the lower right corner of the top right panel points to the plane indicated by the symbol *z*. The middle-right view has plotting capabilities and the different parameters of the rings of the output model can be shown. The bottom views are slices of the data-cube: bottom-left, XY; bottom-middle, XZ; bottom-right, ZY. In the 2-D views the data are displayed with a grayscale color function. The yellow segmentation (in the 3-D and 2-D views) represents the fitted model. The red segmentation (in the 2-D views) is a contour plot of the data. The rendering and the segmentations highlight the data and model at the *rms* value chosen in the *Contour level* widget (Fig. 9). In the chart view, the values of the ring parameters of the fitted model are plotted (it is possible to switch the plots in the chart view menu under the awl widget). The values of the parameters are also reported in the table widget on the `AstroModeling` module window interface (see Fig. 9). The values in the table are editable and can be used to refine the model. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

`SlicerAstro` is a project under continuous development and we have adopted an agile development approach (i.e. development cycles are driven by user-feedback). In addition, the software is open-source and third parties are encouraged to contribute. More important, any idea, feedback, criticism or bug can be reported at the following link in the tracker issue.[23] Finally, although the development of `SlicerAstro` thus far mainly focused on 3-D HI

23 https://github.com/Punzo/SlicerAstro/issues.

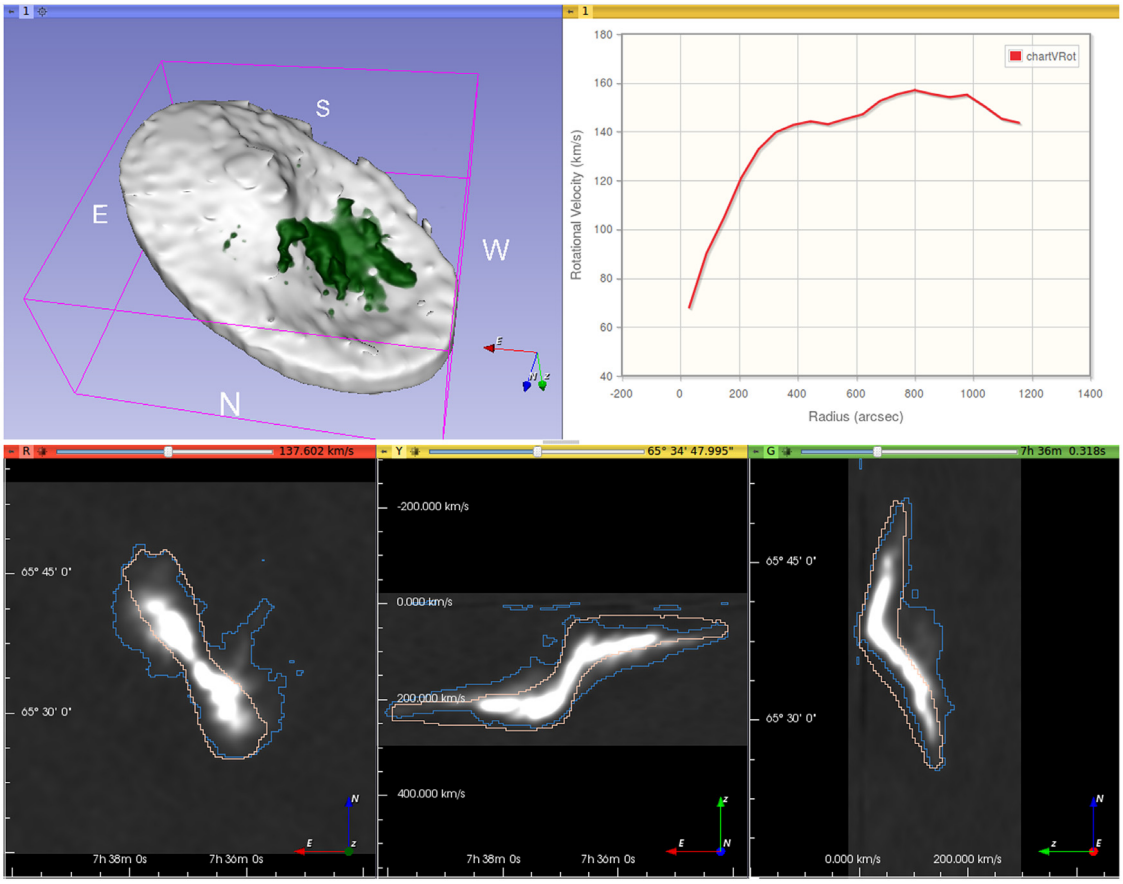


**Fig. 11.** Illustration of the H I data and model, fitted by [3D] `Barolo`, of NGC2403 from the THINGS survey (Walter et al., 2008). The galaxy is very well resolved. The comparative layout is the same as used in Fig. 10. The top-left view is a 3-D view of NGC2403. The white labels represent the four cardinal directions (*N*, *S*, *E*, *W*). The green arrow points along the line of sight. The white segmentation (in the 3-D view) represents the model that fits the regular disk. The model has been fitted in *automatic* mode (i.e., no mask and no input parameters have been provided to [3D] `Barolo` from the graphical user interface). The dark green rendering of the data, from an intensity level of 3 times the *rms*, clearly shows unsettled gas in the inner region. The top-right view has plotting capabilities and the different parameters of the rings of the output model can be shown. The bottom views are slices of the data-cube: bottom-left, XY; bottom-middle, XZ; bottom-right, ZY. The blue and the pink segmentations (in the 2-D views) are contours of the data and model, respectively, at the *rms* value chosen in the *Contour level* widget (see Fig. 9). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

data, it will also be a useful tool for any other type of 3-D astronomical data such as mm/submm molecular line data and optical integral field spectroscopic data. Molecular line data and optical/NIR spectroscopic data have the additional complication that often more than one spectral line is present in a single spectral window. This makes the visualization more complex, though clever stacking of the known spectral lines can e.g. enhance the signal to noise and stacking tools can help to bring out the kinematic behavior of gas which emits multiple spectral lines. One can also think of additional tools to visualize line ratios by e.g. superposing spectral different lines in different colors interactively. In conclusion, though SlicerAstro is useful for other types of astronomical 3-D data, additional tools will be required tailored to the kind of data and the scientific questions to be addressed.

## Appendix

Below, the `Python` code provides an example for applying smoothing to a data-cube, performing the rendering and saving the last as a video. The script can be copy and pasted in the `3DSlicer Python` console, or it can be launched from the command line with the following command:

```
./Slicer --python-script script.py
```

More information is provided at the following link:
https://github.com/Punzo/SlicerAstro/wiki.

```
# Load a data cube in SlicerAstro
slicer.util.loadVolume("/full_path
    /WEIN069.fits",{"center":True})
mw = slicer.util.mainWindow()
ms = mw.moduleSelector()

# Smooth the data cube in
    automatic mode (CPU)
ms.selectModule('AstroSmoothing')
smowidget = slicer.modules.
    astrosmoothing.
    widgetRepresentation()
smowidget.onApply()

# Setup the Render for the data
    cube and its filtered version
ms.selectModule('AstroVolume')
```

```
astrovolumewidget = slicer.modules
   .astrovolume.
   widgetRepresentation()
astrovolumewidget.
   onCurrentQualityControlChanged
   (1)
volumes = slicer.mrmlScene.
   GetNodesByClass("
   vtkMRMLAstroVolumeNode")
volumefiltered = volumes.
   GetItemAsObject(1)
smomrmlpara.SetInputVolumeNodeID(
   volumefiltered.GetID())
astrovolumewidget.
   onCurrentQualityControlChanged
   (1)

# Create videos
ms.selectModule('ScreenCapture')
screencapturewidget = slicer.
   modules.screencapture.
   widgetRepresentation()
instance = screencapturewidget.
   self()
# For the data cube
viewNode = slicer.util.getNode('
   vtkMRMLViewNode1')
instance.viewNodeSelector.
   setCurrentNode(viewNode)
instance.numberOfStepsSliderWidget
   .setValue(360)
instance.videoExportCheckBox.
   setChecked(1)
instance.videoFormatWidget.
   setCurrentIndex(1)
instance.videoFileNameWidget.
   setText("WEIN069.mp4")
instance.videoLengthSliderWidget.
   setValue(6)
instance.onCaptureButton()
# For the filtered version
viewNode = slicer.util.getNode('
   vtkMRMLViewNode2')
instance.viewNodeSelector.
   setCurrentNode(viewNode)
instance.numberOfStepsSliderWidget
   .setValue(360)
instance.videoExportCheckBox.
   setChecked(1)
instance.videoFormatWidget.
   setCurrentIndex(1)
instance.videoFileNameWidget.
   setText("WEIN069_smoothed.mp4")
instance.videoLengthSliderWidget.
   setValue(6)
instance.onCaptureButton()
```

## Acknowledgments

We thank M.A. Ramatsoku and M.A.W. Verheijen for proving us with the HI data of WEIN069. Support also came from S. Pieper (Isomics, Inc.), A. Lasso (Laboratory of Percutaneous Surgery at Queen's University) and E. di Teodoro (Australian National University) in the form of feedback and assistance.

Finally, we thank the reviewers for their constructive comments, which helped us to improve the paper.

D. Punzo and J.M van der Hulst acknowledge the support from the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2013)/ERC Grant Agreement nr. 291-531.

L. Yu was partially supported from National Natural Science Foundation of China (Grant No. 61502132).

We are grateful to the various agencies and programs that funded support and development of 3DSlicer over the years.

## References

Bosma, A., 1978. The distribution and kinematics of neutral hydrogen in spiral galaxies of various morphological types (Ph.D. thesis), Groningen Univ., (1978).

Bregman, J.N., 1980. The galactic fountain of high-velocity clouds. Astrophys. J. 236, 577–591. doi:10.1086/157776.

Calabretta, M.R., (2011).

Calabretta, M.R., Greisen, E.W., 2002. Representations of celestial coordinates in FITS. Astron. Astrophys. 395, 1077–1122. doi:10.1051/0004-6361:20021327. arXiv:astro-ph/0207413.

Di Teodoro, E.M., Fraternali, F., (2015a).

Di Teodoro, E.M., Fraternali, F., 2015b. 3D BAROLO: a new 3D algorithm to derive rotation curves of galaxies. Mon. Not. R. Astron. Soc. 451, 3021–3033. doi:10.1093/mnras/stv1213. arXiv:1505.07834.

Duffy, A.R., Meyer, M.J., Staveley-Smith, L., Bernyk, M., Croton, D.J., Koribalski, B.S., Gerstmann, D., Westerlund, S., 2012. Predictions for ASKAP neutral hydrogen surveys. Mon. Not. R. Astron. Soc. 426, 3385–3402. doi:10.1111/j.1365-2966.2012.21987.x. arXiv:1208.5592.

Fedorov, A., Beichel, R., Kalpathy-Cramer, J., Finet, J., Fillion-Robin, J.-C., Pujol, S., Bauer, C., Jennings, D., Fennessy, F., Sonka, M., Buatti, J., Aylward, S., Miller, J., Pieper, S., Kikinis, R., 2012. 3D slicer as an image computing platform for the quantitative imaging network. 30 (9), 1323–1341. doi:10.1016/j.mri.2012.05.001.

Ferrand, G., English, J., Irani, P., 2016. 3D visualization of astronomy data cubes using immersive displays, ArXiv ePrints. arXiv:1607.08874.

Fraternali, F., van Moorsel, G., Sancisi, R., Oosterloo, T., 2002. Deep HI Survey of the Spiral Galaxy NGC 2403. Astronom. J. 123, 3124–3140. doi:10.1086/340358. arXiv:astro-ph/0203405.

Giese, N., van der Hulst, T., Serra, P., Oosterloo, T., 2016. Non-parametric estimation of morphological lopsidedness. Mon. Not. R. Astron. Soc. 461, 1656–1673. doi:10.1093/mnras/stw1426. arXiv:1606.07387.

Gooch, R., 1996. Karma: a Visualization Test-Bed. In: Jacoby, G.H., Barnes, J. (Eds.), Astronomical Data Analysis Software and Systems V. In: Astronomical Society of the Pacific Conference Series, 101, p. 80. http://www.atnf.csiro.au/computing/software/karma/.

Goodman, A.A., 2012. Principles of high-dimensional data visualization in astronomy. Astron. Nachr. 333, 505. doi:10.1002/asna.201211705. arXiv:1205.4747.

Greisen, E.W., Calabretta, M.R., Valdes, F.G., Allen, S.L., 2006. Representations of spectral coordinates in FITS. Astron. Astrophys. 446, 747–771. doi:10.1051/0004-6361:20053818. arXiv:astro-ph/0507293.

Heijmans, H.J.A.M., 1999. Connected morphological operators for binary images. Comput. Vis. Image Underst. 73, 99–120.

Johnston, S., Taylor, R., Bailes, M., Bartel, N., Baugh, C., Bietenholz, M., Blake, C., Braun, R., Brown, J., Chatterjee, S., Darling, J., Deller, A., Dodson, R., Edwards, P., Ekers, R., Ellingsen, S., Feain, I., Gaensler, B., Haverkorn, M., Hobbs, G., Hopkins, A., Jackson, C., James, C., Joncas, G., Kaspi, V., Kilborn, V., Koribalski, B., Kothes, R., Landecker, T., Lenc, A., Lovell, J., Macquart, J.-P., Manchester, R., Matthews, D., McClure-Griffiths, N., Norris, R., Pen, U.-L., Phillips, C., Power, C., Protheroe, R., Sadler, E., Schmidt, B., Stairs, I., Staveley-Smith, L., Stil, J., Tingay, S., Tzioumis, A., Walker, M., Wall, J., Wolleben, M., 2008. Science with ASKAP. The Australian Square Kilometre Array pathfinder. Exp. Astron. 22, 151–273. doi:10.1007/s10686-008-9124-7. arXiv:0810.5187.

Józsa, G.I.G., Kenn, F., Klein, U., Oosterloo, T.A., 2007. Kinematic modeling of disk galaxies. I. A new method to fit tilted rings to data cubes. Astron. Astrophys. 468, 731–774. doi:10.1051/0004-6361:20066164. arXiv:astro-ph/0703207.

Józsa, G.I.G., Kenn, F., Oosterloo, T.A., Klein, U., (2012).

Kamphuis, P., Józsa, G.I.G., Oh, S.-. H., Spekkens, K., Urbancic, N., Serra, P., Koribalski, B.S., Dettmar, R.-J., 2015. Automated kinematic modeling of warped galaxy discs in large H I surveys: 3D tilted-ring fitting of HI emission cubes. Mon. Not. R. Astron. Soc. 452, 3139–3158. doi:10.1093/mnras/stv1480. arXiv:1507.00413.

Kamphuis, P., Józsa, G.I.G., Oh, S.-. H., Spekkens, K., Urbancic, N., Serra, P., Koribalski, B.S., Dettmar, R.-J., (2015b).

Kent, B.R., 2015. 3D Scientific visualization with blender, Print ISBN: 978-1-6270-5611-3, Online ISBN: 978-1-6270-5612-0.

Lorensen, W.E., Cline, H.E., 1987. Marching cubes: A high resolution 3D surface construction algorithm. SIGGRAPH Comput. Graph. (ISSN: 0097-8930) 21 (4), 163–169. doi:10.1145/37402.37422. http://doi.acm.org/10.1145/37402.37422.

McMullin, J.P., Waters, B., Schiebel, D., Young, W., Golap, K., 2007. CASA architecture and applications. In: Shaw, R.A., Hill, F., Bell, D.J. (Eds.), Astronomical Data Analysis Software and Systems XVI. In: Astronomical Society of the Pacific Conference Series, 376, p. 127.

Mildenberger, P., Eichelberg, M., Martin, E., 2002. Introduction to the DICOM standard. Eur. J. Radiol. 12 (4), 920–927.

Mink, J.D., 2015. Astronomical data formats: What we have and how we got here. Astron. Comput. 12, 128–132. doi:10.1016/j.ascom.2015.07.001. arXiv:1507.03646.

Mink, J., Mann, R.G., Hanisch, R., Rots, A., Seaman, R., Jenness, T., Thomas, B., O'Mullane, W., 2015. The past, present, and future of astronomical data formats. In: Taylor, A.R., Rosolowsky, E. (Eds.), Astronomical Data Analysis Software an Systems XXIV (ADASS XXIV). In: Astronomical Society of the Pacific Conference Series, 495, p. 11 arXiv:1411.0996.

Moschini, U., Teeninga, P., Wilkinson, M.H.F., Giese, N., Punzo, D., van der Hulst, J.M., Trager, S.C., 2014. Towards better segmentation of large floating point 3D astronomical datasets: first results. In: Proceedings of the 2014 Conference on Big Data from Space (BiDS 2014). pp. 232–235.

Naiman, J.P., 2016. AstroBlend: An astrophysical visualization package for blender. Astron. Comput. 15, 50–60. doi:10.1016/j.ascom.2016.02.002. arXiv:1602.03178.

Norris, R., 1994. The challenge of astronomical visualization. Astronomical Data Analysis Software and Systems III 61, 51.

Oosterloo, T., 1995. Visualization of radio data. Proceedings of the Astronomical Society of Australia 12, 215.

Pence, W.D., (2010).

Pence, W.D., Chiappetti, L., Page, C.G., Shaw, R.A., Stobie, E., 2010. Definition of the Flexible Image Transport System (FITS), version 3.0. Astron. Astrophys. 524, A42. doi:10.1051/0004-6361/201015362.

Popping, A., Jurek, R., Westmeier, T., P., , Flöer, L., Meyer, M., Koribalski, B., 2012. Comparison of potential ASKAP HI survey source finders. Publ. Astron. Soc. Aust. 29, 318–339. doi:10.1071/AS11067. arXiv:1201.3994.

Punzo, D., van der Hulst, J.M., Roerdink, J.B.T.M., Oosterloo, T.A., Ramatsoku, M., Verheijen, M.A.W., 2015. The role of 3-D interactive visualization in blind surveys of HI in galaxies. Astron. Comput. 12, 86–99. doi:10.1016/j.ascom.2015.05.004. arXiv:1505.06976.

Punzo, D., van der Hulst, J.M., Roerdink, J.B.T.M., 2016. Finding faint HI structure in and around galaxies: Scraping the barrel. Astron. Comput. 17, 163–176. doi:10.1016/j.ascom.2016.09.002. arXiv:1609.03782.

Punzo, D., van der Hulst, T., Roerdink, J., Fillion-Robin, J.-C., (2016b).

Ramatsoku, M., Verheijen, M.A.W., Kraan-Korteweg, R.C., Józsa, G.I.G., Schröder, A.C., Jarrett, T.H., Elson, E.C., van Driel, W., de Blok, W.J.G., Henning, P.A., 2016. The WSRT ZoA Perseus-Pisces filament wide-field HI imaging survey - I. HI catalog and atlas. Mon. Not. R. Astron. Soc. 460, 923–941. doi:10.1093/mnras/stw968. arXiv:1605.02603.

Roth, S.D., 1982. Ray casting for modeling solids. Elsevier 18, 109–144. doi:10.1016/0146-664X(82)90169-1.

Sancisi, R., Fraternali, F., Oosterloo, T., van der Hulst, T., 2008. Cold gas accretion in galaxies. Astron. Astrophys. Rev. 15, 189–223. doi:10.1007/s00159-008-0010-0. arXiv:0803.0109.

Schroeder, W., Martin, K., Lorensen, B., 2006. The visualization toolkit (4th ed.). Kitware.

Serra, P., Westmeier, T., Giese, N., Jurek, R., Flöer, L., Popping, A., Winkel, B., van der Hulst, T., Meyer, M., Koribalski, B., Staveley-Smith, L., Courtois, H., (2014).

Serra, P., Westmeier, T., Giese, N., Jurek, R., Flöer, L., Popping, A., Winkel, B., van der Hulst, T., Meyer, M., Koribalski, B.S., Staveley-Smith, L., Courtois, H., 2015. SOFIA: a flexible source finder for 3D spectral line data. Mon. Not. R. Astron. Soc. 448, 1922–1929. doi:10.1093/mnras/stv079. arXiv:1501.03906.

Taylor, M., Boch, T., Fitzpatrick, M., Allan, A., Paioro, L., Taylor, J., Tody, D., 2011. IVOA Recommendation: SAMP - Simple Application Messaging Protocol Version 1.3, ArXiv ePrints. arXiv:1110.0528.

Taylor, M.B., 2005. TOPCAT STIL: Starlink Table/VOTable processing software. In: Shopbell, P., Britton, M., Ebert, R. (Eds.), Astronomical Data Analysis Software and Systems XIV. In: Astronomical Society of the Pacific Conference Series, 347, p. 29.

Taylor, R., 2015. FRELLED: A realtime volumetric data viewer for astronomers. Astron. Comput. 13, 67–79. doi:10.1016/j.ascom.2015.10.002. arXiv:1510.03589.

Thomas, B., Jenness, T., Economou, F., Greenfield, P., Hirst, P., Berry, D.S., Bray, E., Gray, N., Muna, D., Turner, J., de Val-Borro, M., Santander-Vela, J., Shupe, D., Good, J., Berriman, G.B., Kitaeff, S., Fay, J., Laurino, O., Alexov, A., Landry, W., Masters, J., Brazier, A., Schaaf, R., Edwards, K., Redman, R.O., Marsh, T.R., Streicher, O., Norris, P., Pascual, S., Davie, M., Droettboom, M., Robitaille, T., Campana, R., Hagen, A., Hartogh, P., Klaes, D., Craig, M.W., Homeier, D., 2015. Learning from FITS: Limitations in use in modern astronomical research. Astron. Comput. 12, 133–145. doi:10.1016/j.ascom.2015.01.009. arXiv:1502.00996.

van der Hulst, J.M., Terlouw, J.P., Begeman, K.G., Zwitser, W., Roelfsema, P.R., 1992. The Groningen image processing system, GIPSY. In: Worrall, D.M., Biemesderfer, C., Barnes, J. (Eds.), Astronomical Data Analysis Software and Systems I. In: Astronomical Society of the Pacific Conference Series, 25, p. 131. https://www.astro.rug.nl/~gipsy/articles/gipsypaper.html.

Verheijen, M., Oosterloo, T., Heald, G., van Cappellen, W., 2009. HI surveys with APERTIF. In: Panoramic Radio Astronomy: Wide-Field 1–2 GHz Research on Galaxy Evolution. p. 10.

Vogt, F.P.A., Owen, C.I., Verdes-Montenegro, L., Borthakur, S., 2016. Advanced data visualization in astrophysics: The X3D pathway. Astrophys. J. 818, 115. doi:10.3847/0004-637X/818/2/115. arXiv:1510.02796.

Vohl, D., Barnes, D.G., Fluke, C.J., Poudel, G., Georgiou-Karistianis, N., Hassan, A.H., Benovitski, Y., Wong, T. H., Kaluza, O., Nguyen, T.D., Bonnington, C.P., (2016).

Walter, F., Brinks, E., de Blok, W.J.G., Bigiel, F., Kennicutt, Jr., R.C., Thornley, M.D., Leroy, A., 2008. THINGS: The HI nearby galaxy survey. Astron. J. 136, 2563–2647. doi:10.1088/0004-6256/136/6/2563. arXiv:0810.2125.

Warner, P.J., Wright, M.C.H., Baldwin, J.E., 1973. High resolution observations of neutral hydrogen in M33 - II. The velocity field. Mon. Not. R. Astron. Soc. 163.

Weinberger, R., Saurer, W., Seeberger, R., 1995. Penetrating the "zone of avoidance". I. A compilation of optically identified extragalactic objects within $|b| <= 5deg$. Astron. Astrophys. Suppl. 110, 269.

Whiting, M.T., 2012. DUCHAMP: a 3D source finder for spectral-line data. Mon. Not. R. Astron. Soc. 421, 3242–3256. doi:10.1111/j.1365-2966.2012.20548.x. arXiv:1201.2710.

Wyvill, G., McPheeters, , Wyvill, B., 1986. Data structure forsoft objects. Vis. Comput. (ISSN: 1432-2315) 2 (4), 227–234. doi:10.1007/BF01900346.

Yu, L., Efstathiou, K., Isenberg, P., Isenberg, T., 2012. Efficient structure-aware selection techniques for 3D point cloud visualizations with 2DOF input. IEEE Trans. Vis. Comput. Graphics (ISSN: 1077-2626) 18 (12), 2245–2254. doi:10.1109/TVCG.2012.217.

Yu, L., Efstathiou, K., Isenberg, P., Isenberg, T., 2016. CAST: Effective and efficient user interaction for context-aware selection in 3D particle clouds. IEEE Trans. Vis. Comput. Graphics (ISSN: 1077-2626) 22 (1), 886–895 doi.ieeecomputersociety.org/10.1109/TVCG.2015.2467202.