

---

# **scikit-surgerydavinci Documentation**

**Thomas Dowrick**

**Oct 08, 2020**



---

## Contents

---

<b>1</b>	<b>Installing</b>	<b>3</b>
<b>2</b>	<b>Example Usage</b>	<b>5</b>
2.1	Contributing . . . . .	5
2.2	Useful links . . . . .	5
<b>3</b>	<b>Licensing and copyright</b>	<b>7</b>
<b>4</b>	<b>Acknowledgements</b>	<b>9</b>
4.1	latest . . . . .	9
	<b>Python Module Index</b>	<b>15</b>
	<b>Index</b>	<b>17</b>





Augmented reality for image-guided laproscopic surgery.

scikit-surgerydavinci is part of the [SNAPPY](#) software project, developed at the [Wellcome EPSRC Centre for Interventional and Surgical Sciences](#), part of [University College London \(UCL\)](#).

Author: Thomas Dowrick

scikit-surgerydavinci supports Python 3.6.



# CHAPTER 1

---

## Installing

---

You can install using pip:

```
pip install scikit-surgerydavinci
```





## CHAPTER 2

---

### Example Usage

---

Single video feed:

```
python sksurgerydavinci -s 0 -m MODEL_DIR
```

-m specifies the location of a directory containing vtl/stl/vtp/ply models.

Stereo video feed:

```
python sksurgerydavinci -s 0 1 -m MODEL_DIR
```

Mock stereo feed using only one input:

```
python sksurgerydavinci -s 0 -1 -m MODEL_DIR
```

If 3 or more screens are available, each video feed will run full screen on a separate display. Video feeds can be assigned to specific displays using the -o argument:

```
python sksurgerydavinci -s 0 1 -o 2 3 1
```

More details on command line arguments can be viewed using:

```
python sksurgerydavinci -h
```

## 2.1 Contributing

Please see the [contributing guidelines](#).

## 2.2 Useful links

- [Source code repository](#)

- [Documentation](#)

## CHAPTER 3

---

### Licensing and copyright

---

Copyright 2019 University College London. scikit-surgerydavinci is released under the BSD-3 license. Please see the [license file](#) for details.



Supported by Wellcome and EPSRC.

## 4.1 latest

### 4.1.1 skysurgerydavinci package

#### Subpackages

#### skurgerydavinci.ui package

#### Submodules

#### skurgerydavinci.ui.gui module

GUI for AR Davinci

**class** skysurgerydavinci.ui.gui.**UI** (*overlay\_window*)

Bases: PySide2.QtWidgets.QWidget

User interface class, to combine vtk render view with buttons/sliders etc

**add\_crop\_buttons** ()

Add buttons to crop the video stream.

**add\_exit\_button** ()

Add a button to exit the program

**add\_opacity\_slider** ()

Create a QSlider that controls VTK model opacity

### **add\_record\_buttons ()**

Add buttons to take screenshot and record video. Button callbacks should be implemented in the Viewer class.

### **add\_toggle\_stereo\_button ()**

Add button to switch between L and R views

### **add\_vtk\_models (vtk\_models)**

Add VTK models to the UI and vtk\_overlay\_window

**Parameters** **vtk\_models** (*List of VTKSurfaceModels*) – List of vtk models to add

### **add\_vtk\_overlay\_window (overlay\_window)**

Add the vtk\_overlay\_window to the UI layout.

**Parameters** **overlay\_window** – vtk\_overlay\_window

### **create\_toggle\_checkbox\_for\_model (model)**

Create a checkbox that will toggle on/off model visibility.

**Parameters** **model** – VTKSurfaceModel

**Returns** QCheckBox

### **exit\_signal = <PySide2.QtCore.Signal object>**

### **on\_exit\_clicked ()**

Close the application when the exit button has been clicked

### **set\_all\_opacities (opacity\_percent)**

Set the opacity for all VTK models

**Parameters** **opacity\_percent** (*int*) – Target opacity value, in %

### **set\_screen (screen)**

Link the widget to a particular screen

**Parameters** **screen** – QtGuiApplication.screen() object

### **staticMetaObject = <PySide2.QtCore.QMetaObject object>**

## **sksurgerydavinci.ui.sksurgerydavinci\_command\_line module**

Command line processing

sksurgerydavinci.ui.sksurgerydavinci\_command\_line.**main** (*args=None*)

Entry point for ardaVIN application

## **sksurgerydavinci.ui.sksurgerydavinci\_demo module**

Hello world demo module

sksurgerydavinci.ui.sksurgerydavinci\_demo.**create\_sample\_model** ()

Create an empty VTKSurfaceModel object, and set the actor to a vtkConeSource

sksurgerydavinci.ui.sksurgerydavinci\_demo.**run\_demo** (*args*)

Show message

## Module contents

scikit-surgerydavinci

## sksurgerydavinci.widgets package

### Submodules

#### sksurgerydavinci.widgets.Viewers module

Mono/Stereo viewing windows for Ardavinci

**class** sksurgerydavinci.widgets.Viewers.**FakeOverlayOnVideoFeed**

Bases: object

Implement empty methods to replicate OverlayOnVideoFeed.

**on\_record\_start** ()  
Intentionally Blank.

**on\_record\_stop** ()  
Intentionally Blank.

**set\_roi** ()  
Intentionally Blank.

**start** ()  
Intentionally Blank.

**stop** ()  
Intentionally Blank.

**class** sksurgerydavinci.widgets.Viewers.**FakeVTKOverlayWindow**

Bases: object

Implement empty methods to replicate VTKOverlayWindow.

**add\_vtk\_models** (*models*)  
Intentionally Blank.

**get\_foreground\_camera** ()  
Intentionally Blank.

**save\_scene\_to\_file** (*fname*)  
Intentionally Blank.

**set\_foreground\_camera** (*camera*)  
Intentionally Blank.

**set\_geometry** ()  
Intentionally Blank.

**set\_screen** ()  
Intentionally Blank.

**class** sksurgerydavinci.widgets.Viewers.**MockStereoViewer** (*video\_source*)

Bases: *sksurgerydavinci.widgets.Viewers.StereoViewerBase*

Mock stereo viewer, duplicating a single camera input to multiple screens.

**Parameters** *video\_source* – OpenCV compatible video source (int or filename)

**staticMetaObject = <PySide2.QtCore.QMetaObject object>**

**class** `sksurgerydavinci.widgets.Viewers.MonoViewer` (*video\_source*)

Bases: `sksurgerydavinci.widgets.Viewers.StereoViewerBase`

Generates a VTK interactor UI with a single video stream as background. :param video\_source: OpenCV compatible video source (int or filename)

Only use the left\_view of StereoViewerBase. Set the other views to non-existent views.

**staticMetaObject = <PySide2.QtCore.QMetaObject object>**

**class** `sksurgerydavinci.widgets.Viewers.StereoViewer` (*left\_source, right\_source*)

Bases: `sksurgerydavinci.widgets.Viewers.StereoViewerBase`

Stereo viewer, creates an overlay window for each video input.

#### Parameters

- **left\_source** – OpenCV compatible video source (int or filename)
- **right\_source** – OpenCV compatible video source (int or filename)

**staticMetaObject = <PySide2.QtCore.QMetaObject object>**

**class** `sksurgerydavinci.widgets.Viewers.StereoViewerBase`

Bases: `PySide2.QtWidgets.QWidget`

Base class for StereoViewers.

Child classes implment the left/right/ui video feeds as appropriate.

**add\_vtk\_models** (*models*)

Add VTK models to all widgets.

**Parameters models** (`VTKSurfaceModel`) – List of models to add.

**are\_widgets\_on\_different\_screens** (*screens*)

Check if each widget has been placed on it's own screen. :param screens: List of screen numbers corresponding to the screen

each widget is displayed on.

**maximize\_left\_and\_right\_widgets** ()

Fullscreen view for each widget. `QWidget.setFullScreen()` isn't working properly in all cases, so resizing manually instead.

**on\_autocrop\_started** ()

Start auto cropping.

**on\_autocrop\_stopped** ()

Stop auto cropping.

**on\_crop\_clicked** ()

Set the ROI on the left view, and copy it to the right.

**on\_record\_start\_clicked** ()

Start recording a video. In the `MockStereoViewer` we only need to record the 'left' input, as the right view is a duplicate of the left. The proper `StereoViewer` class extends this method to also record the right view.

**on\_record\_stop\_clicked** ()

Stop recording data to file and restore button settings.



**on\_screenshot\_clicked()**  
Save a screenshot to disk, using date and time as filename

**run\_before\_quit()**  
Clean up the VTK interactor instances before quitting

**set\_all\_roi(roi)**  
Set the roi for left/right views.

**set\_external\_screens(output\_screens)**  
Set the display screens for each widget, if external monitors are available.  
**Parameters output\_screens** (*List of ints*) – List of screens on which to put widgets

**set\_widget\_screens(screens\_to\_use)**  
Move each of the widgets to a particular screen. If each widget is on it's own screen, run fullscreen.  
**Parameters screens\_to\_use** – List of QScreen objects.

**start()**  
Start all widgets.

**staticMetaObject = <PySide2.QtCore.QMetaObject object>**

**sufficient\_displays\_for\_stereo\_view()**  
Check if there are enough additional displays to give each window it's own screen.

**sync\_camera\_view\_between\_windows()**  
Set all the foreground cameras to the same vtkCamera, so that camera changes are synchronised.

**update\_autocrop()**  
Automatically crop the incoming video stream using AutoCropper.

## Module contents

### Submodules

#### sksurgerydavinci.davinci\_xi\_auto\_cropping module

AutoCropper Module.

**class** sksurgerydavinci.davinci\_xi\_auto\_cropping.**AutoCropBlackBorder** (*threshold=1, min\_size=0*)

Bases: object

Class to generate a ROI for an image based on a threshold value. Originally developed for auto cropping video feed from the DaVinci Xi and Tile Pro, where there is a central image (which we want to keep), surrounded by a black region, and then other stuff that we're not interested in.

By searching for the start/end of the black region, we can auto crop the bit we do want.

For speed, only the red channel of the image is scanned when detecting black areas (See comment in set\_roi for more details).

**Parameters threshold** – Threshold value

**: param min\_size: Minimum size (applies to both width and length)** of the expected output.

### **get\_bounds** (*vector*)

Given a vector, skip any leading values that are > threshold, then find the first contiguous range of values that are below the threshold value. If the whole vector is above the threshold, return the start/end indexes. e.g. [0 0 0 50 50 50 0 0 0 ] return 3, 5 - [50 50 50]

[25 25 0 0 25 25 25 0 0] returns 4, 7 [25 25 25] as the leading 25s are skipped.

[100 100 100 100 100 100] returns 0, 5.

**Parameters** **vector** – Input vector, in which to find the start/end bounds

**Return start** First element that is above the threshold.

**Return end** First element following start, that is below threshold.

### **get\_roi** (*img*)

Calculate the ROI. Find the x/y extent of the ROI by averaging row and column values, and then finding the area of interest.

## Module contents

scikit-surgerydavinci

- modindex
- genindex
- search

**d**

skurgerydavinci.davinci\_xi\_auto\_cropping,  
13

**s**

skurgerydavinci, 14

**u**

skurgerydavinci.ui, 11  
skurgerydavinci.ui.gui, 9  
skurgerydavinci.ui.skurgerydavinci\_command\_line,  
10  
skurgerydavinci.ui.skurgerydavinci\_demo,  
10

**w**

skurgerydavinci.widgets, 13  
skurgerydavinci.widgets.Viewers, 11



## A

add\_crop\_buttons() (*sksurgerydavinci.ui.gui.UI method*), 9  
 add\_exit\_button() (*sksurgerydavinci.ui.gui.UI method*), 9  
 add\_opacity\_slider() (*sksurgerydavinci.ui.gui.UI method*), 9  
 add\_record\_buttons() (*sksurgerydavinci.ui.gui.UI method*), 9  
 add\_toggle\_stereo\_button() (*sksurgerydavinci.ui.gui.UI method*), 10  
 add\_vtk\_models() (*sksurgerydavinci.ui.gui.UI method*), 10  
 add\_vtk\_models() (*sksurgerydavinci.widgets.Viewers.FakeVTKOverlayWindow method*), 11  
 add\_vtk\_models() (*sksurgerydavinci.widgets.Viewers.StereoViewerBase method*), 12  
 add\_vtk\_overlay\_window() (*sksurgerydavinci.ui.gui.UI method*), 10  
 are\_widgets\_on\_different\_screens() (*sksurgerydavinci.widgets.Viewers.StereoViewerBase method*), 12  
 AutoCropBlackBorder (*class in sksurgerydavinci.davinci\_xi\_auto\_cropping*), 13

## C

create\_sample\_model() (*in module sksurgerydavinci.ui.sksurgerydavinci\_demo*), 10  
 create\_toggle\_checkbox\_for\_model() (*sksurgerydavinci.ui.gui.UI method*), 10

## E

exit\_signal (*sksurgerydavinci.ui.gui.UI attribute*), 10

## F

FakeOverlayOnVideoFeed (*class in sksurgerydavinci.widgets.Viewers*), 11

FakeVTKOverlayWindow (*class in sksurgerydavinci.widgets.Viewers*), 11

## G

get\_bounds() (*sksurgerydavinci.davinci\_xi\_auto\_cropping.AutoCropBlackBorder method*), 13  
 get\_foreground\_camera() (*sksurgerydavinci.widgets.Viewers.FakeVTKOverlayWindow method*), 11  
 get\_roi() (*sksurgerydavinci.davinci\_xi\_auto\_cropping.AutoCropBlackBorder method*), 14

## M

main() (*in module sksurgerydavinci.ui.sksurgerydavinci\_command\_line*), 10  
 maximize\_left\_and\_right\_widgets() (*sksurgerydavinci.widgets.Viewers.StereoViewerBase method*), 12  
 MockStereoViewer (*class in sksurgerydavinci.widgets.Viewers*), 11  
 MonoViewer (*class in sksurgerydavinci.widgets.Viewers*), 12

## O

on\_autocrop\_started() (*sksurgerydavinci.widgets.Viewers.StereoViewerBase method*), 12  
 on\_autocrop\_stopped() (*sksurgerydavinci.widgets.Viewers.StereoViewerBase method*), 12  
 on\_crop\_clicked() (*sksurgerydavinci.widgets.Viewers.StereoViewerBase method*), 12  
 on\_exit\_clicked() (*sksurgerydavinci.ui.gui.UI method*), 10

`on_record_start()` (*sksurgery-davinci.widgets.Viewers.FakeOverlayOnVideoFeed method*), 11  
`on_record_start_clicked()` (*sksurgery-davinci.widgets.Viewers.StereoViewerBase method*), 12  
`on_record_stop()` (*sksurgery-davinci.widgets.Viewers.FakeOverlayOnVideoFeed method*), 11  
`on_record_stop_clicked()` (*sksurgery-davinci.widgets.Viewers.StereoViewerBase method*), 12  
`on_screenshot_clicked()` (*sksurgery-davinci.widgets.Viewers.StereoViewerBase method*), 12

**R**

`run_before_quit()` (*sksurgery-davinci.widgets.Viewers.StereoViewerBase method*), 13  
`run_demo()` (*in module sksurgery-davinci.ui.sksurgerydavinci\_demo*), 10

**S**

`save_scene_to_file()` (*sksurgery-davinci.widgets.Viewers.FakeVTKOverlayWindow method*), 11  
`set_all_opacities()` (*sksurgerydavinci.ui.gui.UI method*), 10  
`set_all_roi()` (*sksurgery-davinci.widgets.Viewers.StereoViewerBase method*), 13  
`set_external_screens()` (*sksurgery-davinci.widgets.Viewers.StereoViewerBase method*), 13  
`set_foreground_camera()` (*sksurgery-davinci.widgets.Viewers.FakeVTKOverlayWindow method*), 11  
`set_geometry()` (*sksurgery-davinci.widgets.Viewers.FakeVTKOverlayWindow method*), 11  
`set_roi()` (*sksurgery-davinci.widgets.Viewers.FakeOverlayOnVideoFeed method*), 11  
`set_screen()` (*sksurgerydavinci.ui.gui.UI method*), 10  
`set_screen()` (*sksurgery-davinci.widgets.Viewers.FakeVTKOverlayWindow method*), 11  
`set_widget_screens()` (*sksurgery-davinci.widgets.Viewers.StereoViewerBase method*), 13  
*sksurgerydavinci (module)*, 14  
*sksurgerydavinci.davinci\_xi\_auto\_cropping (module)*, 13  
*sksurgerydavinci.ui (module)*, 11  
*sksurgerydavinci.ui.gui (module)*, 9  
*sksurgerydavinci.ui.sksurgerydavinci\_command\_line (module)*, 10  
*sksurgerydavinci.ui.sksurgerydavinci\_demo (module)*, 10  
*sksurgerydavinci.widgets (module)*, 13  
*sksurgerydavinci.widgets.Viewers (module)*, 11  
`start()` (*sksurgerydavinci.widgets.Viewers.FakeOverlayOnVideoFeed method*), 11  
`start()` (*sksurgerydavinci.widgets.Viewers.StereoViewerBase method*), 13  
*staticMetaObject (sksurgerydavinci.ui.gui.UI attribute)*, 10  
*staticMetaObject (sksurgery-davinci.widgets.Viewers.MockStereoViewer attribute)*, 11  
*staticMetaObject (sksurgery-davinci.widgets.Viewers.MonoViewer attribute)*, 12  
*staticMetaObject (sksurgery-davinci.widgets.Viewers.StereoViewer attribute)*, 12  
*staticMetaObject (sksurgery-davinci.widgets.Viewers.StereoViewerBase attribute)*, 13  
*StereoViewer (class in sksurgery-davinci.widgets.Viewers)*, 12  
*StereoViewerBase (class in sksurgery-davinci.widgets.Viewers)*, 12  
`stop()` (*sksurgerydavinci.widgets.Viewers.FakeOverlayOnVideoFeed method*), 11  
`sufficient_displays_for_stereo_view()` (*sksurgerydavinci.widgets.Viewers.StereoViewerBase method*), 13  
`sync_camera_view_between_windows()` (*sksurgerydavinci.widgets.Viewers.StereoViewerBase method*), 13

**U**

*UI (class in sksurgerydavinci.ui.gui)*, 9  
`update_autocrop()` (*sksurgery-davinci.widgets.Viewers.StereoViewerBase method*), 13