Platomic User Guide

Aaron Yat Lam

August 4, 2021

Contents

Chapter 1

Quick start guide

1.1 Quick overview of Platomic

Platomic is a graphical application built for Plato capable of quantitatively and qualitatively determining the electrical connectivity between atoms in molecules. It uses Plato's tight binding one script and the Hairy Probe method. This project was submitted in partial fulfilment of the requirements for the MSc degree in Computing Science of Imperial College London (2020-2021).

The source code is available at: https://github.com/aaronlam123/Platomic In addition to the written instructions for the install and getting started process below, you may refer to the video tutorial available here:

1.2 System requirements

System requirements for the Platomic binary:

1. Linux

System requirements for Platomic:

1. Linux
2. OSX

Dependencies (tested on):

1. PyQt5 (5.15.4)
2. pyqtgraph (0.11.1)
3. SciPy (1.5.2)
4. pandas (1.2.1)

5. PyAutoGUI (0.9.52)
6. PyOpenGL (3.1.5)
7. NumPy (1.20.2)

Runs purely on Python 3 (tested on 3.8.5). Requires a copy of Plato for full functionality, which you can ask for permission from @horsfielda. If given permission to access Plato, you may contact me for a Platomic binary + Plato binary distribution. This distribution runs out of the box with no package / dependency installation required. As such it is most suitable for non-expert users who are unfamiliar with Linux and / or would like to skip over the installation of a number of packages.

1.3 Getting started

To run the Platomic binary, navigate to the dist folder, and open a terminal window via right click. Then enter the command "chmod +x setup" to enable execution permissions for the setup bash. Next type "./setup" then "./main" and it should start. Only ./main is needed thereafter to open Platomic.

Alternatively, after downloading the repository and the required dependencies you can start Platomic by running main.py.

In both instances, a copy of Plato, is required to enable full functionality for Platomic. The Plato folder should be located in the same directory as main / main.py.

Chapter 2

Using Platomic

The UI of Platomic consists of three parts, the central window, properties sidebar and console log as seen in Figure 2.1 below. The central window and properties sidebar are of QTabWidget class, allowing for multiple tabs to be displayed depending on which is selected.

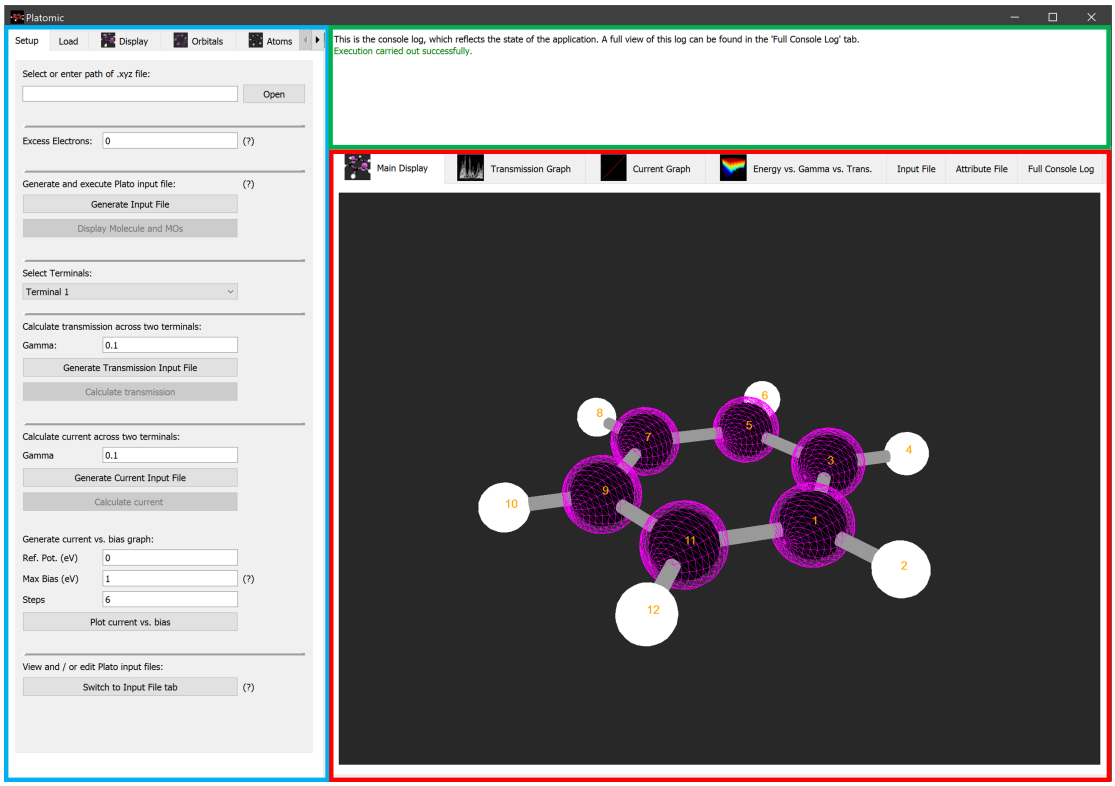

Figure 2.1: UI for Platomic. Red: central window, consisting of the 2D/3D plot and text widgets. Blue: properties sidebar, used to setup the application and change settings. Green: console log, for console outputs and error messages.

2.1 Visualising molecular orbitals

In addition to the walk-through written below, you may refer to the video tutorial available here:

2.2 Transmission graphs

In addition to the walk-through written below, you may refer to the video tutorial available here:

2.3 Current calculations

In addition to the walk-through written below, you may refer to the video tutorial available here:

2.4 Current vs. bias graphs

In addition to the walk-through written below, you may refer to the video tutorial available here:

2.5 Energy vs. gamma vs. transmission plots

In addition to the walk-through written below, you may refer to the video tutorial available here:

2.6 Customisation features

In addition to the walk-through written below, you may refer to the video tutorial available here:

2.7 Load pre-computed Plato output files

In addition to the walk-through written below, you may refer to the video tutorial available here:

Chapter 3

Developing Platomic

3.1 Programs and scripts

3.1.1 atom.py

__init__: initalises Atom object.
decrement_bond_count: decrements bondCount member variable when bond is plot.
get_bondCount: getter for the remaining number of times bonds can be plot.
get_bonding: getter for bonding behaviour as read from attributes.txt.
get_colour: getter for atom colour.
get_eigenenergies: getter for the eigenenergies of the system.
get_eigenenergy: getter for a specific eigenenergy.
get_eigenvector: getter for a specific eigenvector.
get_index: getter for the atom index.
get_isSelectedCurrA: getter for whether the atom is selected for region A.
get_isSelectedCurrB: getter for whether the atom is selected for region B.
get_isSelectedTrans: getter for whether tha atom belongs to a terminal.
get_maxBonds: returns maximum number of bonds permitted for this atom as read from attributes.txt.
get_mi: returns mesh item object associated with the atom.
get_quantum_dict: returns orbital quantum numbers for the atom.
get_radius: returns radius for the atom as read from attributes.txt.
get_skipOrbital: returns flag to skip orbital plotting in Platomic for optimisation reasons.
get_symbol: returns the atomic symbol.
get_total_orbitals: returns total number of orbitals in the system.
get_xyz: returns coordinates of the atom as a list.
reset_bonds: helper function to reset bondCount to maxBonds.
set_bonding: setter for bonding behaviour as read from attributes.txt.
set_colour: setter for atom colour.
set_eigenenergies: setter for the eigenenergies of the system.

set_eigenvector: setter for the 2D array of eigenvectors.
set_isSelectedCurrA: sets atom to be part of region A.
set_isSelectedCurrB: sets atom to be part of region B.
set_isSelectedTrans: sets atom to a specific terminal.
set_maxBonds: sets the maximum number of bonds the atom can form as read from attributes.txt.
set_mi: sets mesh item object to Atom instance.
set_quantum_dict: sets orbital quantum dictionary to Atom instance.
set_radius: sets radius to Atom instance as read from attributes.txt.
set_skipOrbital: sets skip orbital plotting flag for Atom instance as read from attributes.txt.
set_total_orbitals: sets the total number of orbitals in the system.

3.1.2 glaxis.py

Label class

__init__: inherits from gl.GLGraphicsItem.GLGraphicsItem parent class.
setGLViewWidget: sets Label instance to a GLViewWidget.
paint: renders Label onto a GLViewWidget.

GLAxis class

__init__: inherits from GLAxisItem parent class.
add_labels: adds Labels to GLAxis mesh item.
add_tick_values: adds tick values to GLAxis mesh item.
paint: renders GLAxis axes and tick marks.

3.1.3 glview.py

GLView class

__init__: inherits from GLViewWidget parent class.
mousePressEvent: custom mouse click interaction for terminals and regions.
itemsAt: returns list of mesh items in a region.
paintGL: used to render coordinate based labels.
readQImage: captures widget view as an image.

3.1.4 glview3D.py

GLView3D class

__init__: inherits from GLViewWidget parent class.
readQImage: captures widget view as an image.

3.1.5 input.py

correct_quantum: converts Plato magnetic quantum numbers which are whole numbers back to the traditional integer representation. E.g. 0, 1, 2 to -1, 0, 1.
create_all_atoms: creates the Atom array for the molecule.
create_orbital_dict: creates a dictionary with orbital and quantum number information for each element in the molecule.
curr_plato_input: generates Plato current input file.
eig_arr_from_wf: returns 2D array with all eigenvectors split by energy.
find_current_in_file: parses output file for current.
get_last_two_chars: returns last two characters in file which matches a string.
get_line_number: returns line number in file which matches a string.
get_lines_between: returns lines between start and end strings with options.
input_file_setup: generates molecular orbital input file.
isfloat: boolean checks if a string is a float.
isnatnumber: boolean checks if a string is a natural number.
isposfloat: boolean checks if a string is a positive float.
lines_that_contain: returns the line from file which matches a string.
lines_that_start_with: returns the line from file if the line's start matches a string.
process_current_out: processes directory of current output files into arrays.
process_energy_gamma_trans_csv: processes directory of .csv files into arrays.
return_occupied_keys: returns the number of selected terminals.
return_occupied_keys_list: returns the terminals selected.
set_attr_from_file: sets attributes from attributes.txt to Atom array.
set_eig_to_atoms: sets 2D array of eigenvalues, orbital count and quantum dictionary to Atom array.
total_orbitals: returns the total number of orbitals in the system.
trans_plato_input: generates Plato transmission input file.
transmission_headers: parses transmission .csv file for column headers and maps headers to Platomic selections.
xyz_to_plato_input: generates Plato molecular orbital input file.

3.1.6 main.py

__init__: inherits from QtWidgets.QMainWindow class.
draw: updates main display by plotting atoms, bonds and orbitals.
execute: interacts with Plato back-end.

onBiasLineEditChanged: checks bias line edit for invalid inputs for current vs. bias graphs.
onCurrExecuteLoadedButtonClicked: loads current vs. bias graph from pre-computed directory of .out files.
onCurrentSelectionA: called when the user right clicks on an atom, writes to console log user selection and updates member variable for region A.
onCurrentSelectionB: called when the user middle clicks on an atom, writes to console log user selection and updates member variable for region B.
onExcessLineEditChanged: checks excess electrons line edit for invalid inputs.
onExecute3DGraphButtonClicked: executes Plato back-end and plots energy vs. gamma vs. transmission 3D graph.
onExecuteButtonClicked: executes Plato back-end and plots the molecule with molecular orbitals.
onExecuteCurrButtonClicked: executes Plato back-end and prints current to console log.
onExecuteCurrGraphButtonClicked: executes Plato back-end and graphs current vs. bias graph.
onExecuteLoadedButtonClicked: loads molecule and molecular orbitals from pre-computed .out and .wf files.
onGammaEndLineEditChanged: checks gamma end line edit for invalid inputs.
onGammaExecuteLoadedButtonClicked: loads pre-computed directory of .csv files for energy vs. gamma vs. transmission 3D graph.
onGammaLineEditChanged: checks gamma line edit for invalid inputs.
onGammaLineEditChanged2: checks gamma line edit for invalid inputs.
onGammaStartLineEditChanged: checks gamma start line edit for invalid inputs.
onGammaStepsLineEditChanged: checks gamma steps line edit for invalid inputs.
onGenerateCurrInputFileButtonClicked: generates current input file.
onGenerateInputFileButtonClicked: generates molecular orbital input file.
onGenerateTransInputFileButtonClicked: generates transmission input file.
onOpenCsvFileButtonClicked: opens QFileDialog for pre-computed transmission .csv files.
onOpenDirButtonClicked: opens QFileDialog for a pre-computed directory of .out files (for current vs. bias graphs).
onOpenDirGammaButtonClicked: opens QFileDialog for a pre-computed directory of .csv files for energy vs. gamma vs. transmission graphs.
onOpenFileButtonClicked: opens QFileDialog for a .xyz file.
onOpenOutFileButtonClicked: opens QFileDialog for a .out file.
onOpenWfFileButtonClicked: opens QFileDialog for a .wf file.
onReferenceLineEditChanged: checks reference potential line edit for invalid inputs.
onResetViewButtonClicked: resets view of the main display.
onSave3DImageButtonClicked: saves energy vs. gamma vs. transmission plot as a PNG.
onSaveAttributeFileButtonClicked: saves changes to edits made to the attribute file.

onSaveImageButtonClicked: saves main display view as PNG.
onSaveInputFileButtonClicked: saves changes to edits made to the input file.
onStepsLineEditChanged: checks steps (current vs. bias graph) line edit for invalid inputs.
onSwitchToAttrFileTabButtonClicked: switches central window tab to attribute file tab.
onSwitchToInputFileTabButtonClicked: switches central window tab to input file tab.
onToggleAtomsButtonClicked: toggles atoms on / off in the main display.
onTransExecuteButtonClicked: executes Plato back-end and plots transmission graph.
onTransExecuteLoadedButtonClicked: loads transmission graph from pre-computed .csv file.
onTransSelection: called when the user left clicks on an atom, writes to console log user selection and updates member variable for terminal selection.
replaceTextEdit: helper function to update the text edit widget for input files.
setAtomColSliderLabel: called when atom column mesh slider is moved.
setAtomRowSliderLabel: called when atom row mesh slider is moved.
setBondColSliderLabel: called when bond column mesh slider is moved.
setBondRadiusSliderLabel: called when bond radius slider is moved.
setBondRowSliderLabel: called when bond row mesh slider is moved.
setBondThresholdSliderLabel: called when bond threshold slider is moved.
setBrightnessSliderLabel: called when main display brightness slider is moved.
setCheckBoxIndex: called when index coordinate label checkbox is changed.
setCheckBoxPosition: called when position coordinate label checkbox is changed.
setCheckBoxRadius: called when radius coordinate label checkbox is changed.
setCheckBoxSymbol: called when symbol coordinate label checkbox is changed.
setColourASliderLabel: called when orbital alpha colour slider is moved.
setColourBSliderLabel: called when orbital blue colour slider is moved.
setColourComboBox: called when colour for coordinate labels is changed.
setColourGSliderLabel: called when orbital green colour slider is moved.
setColourRSliderLabel: called when orbital red colour slider is moved.
setFontComboBox: called when font for coordinate labels is changed.
setGraphComboBox: called when graph header for transmission graphs is changed, so the transmission plot widget is updated.
setHorizontalSliderLabel: called when the molecular orbital slider is moved.
setOffsetComboBox: called when the offset direction for coordinate labels is changed.
setOrbColSliderLabel: called when the orbital column mesh slider is moved.
setOrbRowSliderLabel: called when the orbital row mesh slider is moved.
setPhiSliderLabel: called when the phi rotation slider is moved.
setScalerSliderLabel: called when the orbital scaler slider is moved.
setSizeComboBox: called when the font size for coordinate labels is changed.
setTerminalComboBox: called when the terminal index is changed.
setThetaSliderLabel: called when the theta rotation slider is moved.

updateAtomColSliderLabel: called when atom column mesh slider is changed.
updateAtomRowSliderLabel: called when atom row mesh slider is changed.
updateBondColSliderLabel: called when bond column mesh slider is changed.
updateBondRadiusSliderLabel: called when bond radius slider is changed.
updateBondRowSliderLabel: called when bond row mesh slider is changed.
updateBondThresholdSliderLabel: called when bond threshold slider is changed.
updateBrightnessSliderLabel: called when main display brightness slider is changed.
updateColourASliderLabel: called when orbital alpha colour slider is changed.
updateColourBSliderLabel: called when orbital blue colour slider is changed.
updateColourGSliderLabel: called when orbital green colour slider is changed.
updateColourRSliderLabel: called when orbital red colour slider is changed.
updateHorizontalSliderLabel: called when the molecular orbital slider is changed.
updateOrbColSliderLabel: called when the orbital column mesh slider is changed.
updateOrbRowSliderLabel: called when the orbital row mesh slider is changed.
updatePhiSliderLabel: called when the phi rotation slider is changed.
updateScalerSliderLabel: called when the orbital scaler slider is changed.
updateThetaSliderLabel: called when the theta rotation slider is changed.
writeErrorToLogs: helper function to write errors to both console logs.
writeToLogs: helper function to write messages to both console logs.

3.1.7 orbital.py

advanced_orbital: generates s, p, d, f orbital mesh data.

3.1.8 plot.py

colours: returns either the colour as a string or RGBA tuple.
current_graph: plots the current vs. bias graph on a plot widget.
draw_advOrbFaces: draws s, p, d, f orbitals with face meshes.
draw_advOrbHorz: draws s, p, d, f vertex orbitals horizontally.
draw_advOrbVert: draws s, p, d, f vertex orbitals vertically.
draw_advOrbWf: draws s, p, d, f orbitals as a wireframe.
draw_atoms: draws atoms using passed in Atom array.
draw_bonds: draws bonds using passed in Atom array.
draw_selection: draws terminal selection highlights.
draw_sphOrbFaces: draws spherical orbitals with face meshes.
draw_sphOrbWf: draws spherical orbitals as a wireframe.
energy_gamma_trans_graph: plots the energy vs. gamma vs. transmission graph.
transmission_graph: plots the transmission graph.

3.2 Directory map

3.2.1 Platomic binary

```
Platomic
├── build (stores metadata, is useful for debugging)
├── dist
│   ├── Plato (Plato repository or binary)
│   ├── benzene.xyz
│   ├── benzene_Au_relaxed.out (example Plato .out file)
│   ├── benzene_Au_relaxed.wf (example Plato .wf file)
│   ├── benzene_Au_relaxed.xyz (example Plato .xyz file)
│   ├── main (main executable)
│   ├── setup (simple bash file to setup Platomic)
│   ├── config
│   │   ├── attributes.txt (configure atom and bond plotting attributes)
│   │   ├── benzene.out (example Plato .out file)
│   │   ├── benzene.wf (example Plato .wf file)
│   │   ├── default.in (template for molecular orbital input file gen)
│   │   ├── default_curr.in (template for current input file gen)
│   │   ├── default_trans.in (template for transmission input file gen)
│   │   ├── icon.png
│   │   ├── icon1.png
│   │   ├── icon2.png
│   │   ├── icon3.png
│   │   ├── icon4.png
│   │   ├── icon5.png
│   │   ├── mainwindow5.ui (PyQt5 UI file for Platomic)
│   │   ├── platomic.png (Platomic application icon)
```

3.2.2 Platomic

```
Platomic
├── atom.py (Atom class)
├── glaxis.py (custom GLAxis class for 3D graphs)
├── glview.py (custom GLViewWidget class for main display)
├── glview3D.py (custom GLViewWidget class for 3D graphs)
├── input.py (processing input and output files)
├── main.py (main program)
├── orbital.py (mesh data generation for s, p, d, f orbitals)
├── plot.py (plotting of atoms, bonds, orbitals and graphs)
├── README.md
├── benzene.xyz
├── benzene_Au_relaxed.out
├── benzene_Au_relaxed.wf
├── benzene_Au_relaxed.xyz
├── Plato
```

```
├── config
│   ├── attributes.txt
│   ├── benzene.out
│   ├── benzene.wf
│   ├── default.in
│   ├── default_curr.in
│   ├── default_trans.in
│   ├── icon.png
│   ├── icon1.png
│   ├── icon2.png
│   ├── icon3.png
│   ├── icon4.png
│   ├── icon5.png
│   ├── mainwindow5.ui
│   └── platomic.png
└── tests (directory used to store unit tests)
    ├── config
    ├── test_files
    ├── coverage.py (used to generate html coverage reports)
    ├── glaxis_test.py
    ├── glview3D_test.py
    ├── glview_test.py
    ├── input_test.py
    ├── main_test.py
    ├── orbital_test.py
    └── plot_test.py
```