
AUDRI Documentation

Author

May 14, 2018

Contents:

1	audri module	1
2	config module	3
3	data module	7
4	gui package	9
4.1	Submodules	9
4.2	gui.conf module	9
4.3	gui.controls module	10
4.4	gui.main module	11
4.5	gui.panel module	11
4.6	gui.sim module	12
5	visualiser package	15
5.1	Submodules	17
5.2	visualiser.util module	17
5.3	visualiser.vehicles module	18
5.4	visualiser.visualiser module	19
6	Indices and tables	23
	Python Module Index	25

Provides functionality for automated driving. AUDRI is able to receive feature vectors, learn from them, and then decide upon an action to perform when presented with another feature vector

class `audri.Agent`

Bases: `object`

A learning agent who will learn how to drive from data it is given, and is able to use its training to command a car

action (*stateVector*)

Return the predicted label for the given state vector

Parameters `stateVector` – (`dict`) The unlabeled state vector

load (*name*)

save (*name*)

train (*data*)

Train the agent using a set of data

Parameters `data` – (`list`) A list of `dict` state vectors

CHAPTER 2

config module

Configuration classes for all of the components

```
class config.GUIConfig
```

```
    Bases: object
```

```
    Configuration used by the gui module
```

```
    tuple attributes store main text at index 0, and tooltip text at index 1
```

```
    CompareText = 'Compete with AUDRI'
```

```
        (str) Text for the button in the MainFrame, linking to the Simulator in compare mode
```

```
    ConfAppearance = 'Appearance'
```

```
        (str) Header text for the appearance parameters section in the ConfigFrame
```

```
    ConfBackground = ('Scroll background', 'Whether the background should scroll down the v
```

```
    ConfCarScale = ('Car scale', 'Multiplied against the normal size of the car sprite')
```

```
    ConfCarSpeed = ('Car speed', '(metres per second)\nThe forward speed of the main car')
```

```
    ConfExperiment = 'Experiment parameters'
```

```
        (str) Header text for the experimental parameters section in the ConfigFrame
```

```
    ConfFPS = ('Frames per second', 'Frame rate - how many times per second the screen is r
```

```
    ConfInnerPadding = 15
```

```
        (int) Padding between labels and their connected control in the ConfigFrame
```

```
    ConfObstFreq = ('Obstacle frequency', '(seconds)\nTime between obstacles spawning')
```

```
    ConfObstScale = ('Obstacle scale', 'Multiplied against the normal size of the obstacle
```

```
    ConfObstSpeed = ('Obstacle speed', '(metres per second)\nThe forward speed of the obst
```

```
    ConfOffroad = ('Allow the car to drive offroad', 'If checked, the main car will be abl
```

```
    ConfRandomSeed = ('Random seed', 'The value used to control pseudo-random number gener
```

```
    ConfRecordFreq = ('Snapshot interval', 'Gap in seconds between recordings of the state
```

ConfRowPadding = 10

ConfSave = ('Save', 'Save configuration and return to the main menu')

ConfText = 'Configuration'
(*str*) Text for the button in the *MainFrame*, linking to the *ConfigFrame*

ConfTickrate = ('Tickrate', 'Tick rate - how many times per second the game logic should run')

ConfTitle = 'Configuration'
(*str*) Header text in the *ConfigFrame*

DataText = 'Gather training data'
(*str*) Text for the button in the *MainFrame*, linking to the *Simulator* in training mode

Font = {'family': 'sans-serif', 'size': 10}
(*dict*) Data used to create a normal font

FontBold = {'family': 'sans-serif', 'size': 10, 'weight': 'bold'}
(*dict*) Data used to create a bold font

FontHeading = {'family': 'sans-serif', 'size': 15, 'weight': 'bold'}
(*dict*) Data used to create a header font

FontMonospace = {'family': 'monospace', 'size': 10}
(*dict*) Data used to create a monospaced font

Height = 600
(*int*) Height of the main window

MainButtonXPad = 15

MainButtonYPad = 15

ModeText = ['manual', 'AUDRI', 'compare']
list of *str* mapping simulator modes to a description of that mode, used by *SimulatorPanel*

PanelWidth = 220
(*int*) Width of the Panel

SimPopupDupWarn = 'This name is already in use, overwrite?'
Duplicate warning message in (*NameDatasetPopup*)

SimPopupPad = 10
Padding in (*NameDatasetPopup*)

SimPopupText = 'Choose a name for this training set'
Message in (*NameDatasetPopup*)

SimPopupTitle = 'Name the training set'
Window title of the popup for naming a training set (*NameDatasetPopup*)

Subtitle = 'Teaching a computer to drive through example'
(*str*) Subtitle shown in the *MainFrame*

TestText = 'Test AUDRI'

Theme = 'clam'
(*str*) Tkinter theme to use

Title = 'AUDRI Simulator'
(*str*) Main title shown in the titlebar and the *MainFrame*

TooltipWidth = 250

VisualiserWidth = 500
 (int) Width of the *visualiser.visualiser.SimulatorVisualiser*

class config.SimulatorConfig

Bases: *object*

Configuration used by the *visualiser* class

CarScale = 0.35
 (float) Scale of the car's sprite

CarSpeed = 11
 (float) Speed of the main car in metres per second

FPS = 70
 (int) Targeted frames per second to be drawn

LaneWidth = 86
 (int) Width of a lane

ObstacleInterval = 2
 (float) Seconds between spawning obstacles

ObstacleSpeed = 8
 (float) Speed of obstacle vehicles in metres per second
 Should be less than *CarSpeed*

OffroadAllowed = False
 (bool) Whether the car can go offroad

OffroadWidth = 120
 (int) Width of a an offroad lane

PixelMetreRatio = 50
 (float) Ratio of pixels to metres, affects the visualisation of speeds

RandomSeed = 'geqJQD6MfJ'
 Random seed used to influence when vehicles spawn Can be *int*, *str*, and others. See *random.seed()* for details

RecordInterval = 0.2
 (float) Seconds between each snapshot

ScrollBackground = True
 (bool) Whether the background should move

TickRate = 5
 (int) Targeted milliseconds between each simulation tick

class config.Singleton

Bases: *type*

Metaclass that provides singleton behaviour

Author Adam Forsyth <adam@adamforsyth.net>

Source <https://stackoverflow.com/a/6798042>

_instances = {}
 (dict) stores the instances of each class

Provides functionality for manipulating training data sets and models

`data.DATA_DIR = '/home/docs/checkouts/readthedocs.org/user_builds/audri/checkouts/latest/'`
(*str*) Path to store data in

`data.DATA_ORDER = ['action', 'aheadDistance', 'currentLane']`
list specifying the order of data

`data.MODEL_DIR = '/home/docs/checkouts/readthedocs.org/user_builds/audri/checkouts/latest/'`
(*str*) Path to store models in

`data.dataExists` (*name*)

Parameters *name* – (*str*) The name of the data set to search for

Returns *bool* indicating if a data set named *name* exists

`data.loadData` (*name*)

Parameters *name* – (*str*) The name of the data set to load

Returns *dict* containing the training data

Returns *dict* containing the experimental settings used

`data.modelExists` (*name*)

Parameters *name* – (*str*) The name of the model to search for

Returns *bool* indicating if a model named *name* exists

`data.saveData` (*name*, *data*)

Save a data set to a file with a given name.

Additionally saves a metadata file containing the experimental settings used during the training

Parameters

- **name** – (*str*) The name of the data set to save

- **data** – (`iter` of `dict`) 2-dimensional data structure, each list entry is a feature vector.
Each feature vector is an associative (named) array of features

4.1 Submodules

4.2 gui.conf module

Provides ConfigFrame, the Tkinter frame that allows modifying the visualiser configuration

class `gui.conf.ConfigFrame` (*root, main*)

Bases: `tkinter.Frame`

Frame allowing the configuration of the visualiser

Contains a number of Tkinter controls allowing the modification of attributes in *SimulatorConfig*

_labelControl (*text, control, ctrlOpts={}*)

Create a label and a widget, which are placed adjacent on the same row.

Parameters

- **text** – `str` to show in the label
- **control** – `object` class of the control to create
- **ctrlOpts** – `dict` of kwargs to use when constructing the control.

Returns The created `Widget` control

_load ()

Load values into the controls from the *SimulatorConfig*

_matches = {'CarScale': 'carScale', 'CarSpeed': 'carSpeed', 'ObstacleInterval': 'ob

SimulatorConfig attributes as keys, associated private attributes (with a `set()` method) as attributes.

These config values are all `float`

_save ()

Save values into the *SimulatorConfig* from the controls

4.3 gui.controls module

Provides custom Tkinter controls

class `gui.controls.LabeledScale` (*root, font, resolution=2, **kwargs*)

Bases: `tkinter.Frame`

`tkinter.ttk.Scale` and a `tkinter.Spinbox` joined in a frame

The Scale is shown to the left of the Spinbox

Parameters

- **root** – Parent `tkinter.Widget`
- **font** – Font to use in the Spinbox
- **resolution** – (*int*) Number of decimal places to round stored and displayed value to

_update (*val=None*)

Callback method for both the Spinbox and Scale so that each can update the other, and the value can be properly formatted

get ()

Returns `float` - Value stored, rounded to specified resolution number of decimal places

set (*val*)

Set value of both the Spinner and Scale

Parameters **val** – `float` to set values to

class `gui.controls.ToolTip` (*widget, text="", delay=500, width=250*)

Bases: `object`

Creates a tooltip that appears above the given widget when hovered

Authors

- **Fuzzyman:** http://voidspace.org.uk/python/weblog/arch_d7_2006_07_01.shtml#e387
- **vegaseat:** <https://daniweb.com/programming/software-development/code/484591>
- **crxguy52:** <https://stackoverflow.com/a/36221216>

Parameters

- **widget** – `tkinter.Widget` to bind to
- **text** – (*str*) Text content of the tooltip
- **delay** – (*int*) Milliseconds before displaying tooltip on hover
- **width** – (*int*) Maximum width of tooltip in characters

_cursorPos (*widget*)

_enter (*event=None*)

_hide ()

_leave (*event=None*)

_schedule ()

_show (*event=None*)

_unsubscribe ()

4.4 gui.main module

The main window and main page which links to the others

```
class gui.main.MainFrame (root, main)
    Bases: tkinter.Frame
```

Main ‘page’ of the application, linking to the *Simulator* and *ConfigFrame* using *Button* widgets. These cause the *MainWindow* to focus on the desired frame

Parameters

- **root** – Widget parent
- **main** – *MainWindow*

```
startSim (mode)
```

```
class gui.main.MainWindow (*args, **kwargs)
    Bases: tkinter.Tk
```

The main Tkinter window of the application

Encapsulates *Simulator* and *ConfigFrame*

```
_keyPress (event)
```

Tkinter key press callback method, sends events to the *Simulator* if it is currently focused

```
_simulator
```

The encapsulated *Simulator*

Getter Get the simulator

Type Widget

```
_tick ()
```

Run *tick()* on the *Simulator*

Repeatedly calls itself in intervals of *TickRate* milliseconds while the *Simulator* window remains focused

```
back ()
```

Return to the main menu by raising the *MainFrame* to the top

```
focus (name)
```

Focus on a specified contained frame. Starts the tick cycle by calling *_tick()* if focusing on the *Simulator*

Parameters **name** – (*str*) Class name of the encapsulated frame that should be focused

One of ‘MainFrame’, ‘ConfigFrame’

4.5 gui.panel module

Provides the side panel shown alongside the visualiser

```
class gui.panel.SimulatorPanel (parent, sim, visualiser, **kwargs)
    Bases: tkinter.Frame
```

Side panel of the window, extends *tk.Frame*. Shows information about the visualisation.

Parameters

- **parent** – Parent `tkinter.Widget`
- **sim** – `Simulator` instance to control
- **visualiser** – `SimulatorVisualiser` instance to take information from
- **kwargs** – keyword arguments to pass to superclass constructor

_call (*func*)

Return a function that focuses the panel then calls *func*

Used for button commands

_makeLabel (*name, text, row*)

Create and attach multiple labels to the given row

Parameters

- **name** – `str` used in property names
- **text** – `str` to use as label
- **row** – `tkinter.Widget` to attach to

_newRow (***kwargs*)

Create a new row to be used

Returns The new `Frame`

tick ()

Update displayed information using the `SimulatorVisualiser`

4.6 gui.sim module

Provides the main simulator, which includes the Pygame visualiser and a Tkinter panel

class `gui.sim.NameDatasetPopup` (*main*)

Bases: `tkinter.Toplevel`

A Tkinter popup that requests a name for a training set.

It will check if the name is in use; if it is, another prompt asks whether it should be overridden or another name should be input.

accept (**args*)

Close the popup and set the input value on the `Simulator`

class `gui.sim.Simulator` (*root, main*)

Bases: `tkinter.Frame`

The GUI component of the visualiser Contains the `SimulatorVisualiser` (Pygame canvas) in a `tkinter.ttk.Frame`. The `SimulatorPanel` is displayed alongside it.

finish ()

Stop the visualiser and return to the main menu

focus ()

If loaded in manual mode, create a `NameDatasetPopup` to choose the training data name.

If in AUDRI mode, train the model

keyPress (*event*)

Pass key press events to visualiser

mode

The current mode in which the simulator should run in.

0 = Manual: Collect training data

1 = AUDRI: Train AUDRI and let it control the car

2 = Compare: Allow both expert and AUDRI to control two different cars, side by side

Getter Get the current mode

Setter Set the current mode, also setting the mode of the *SimulatorVisualiser*. Prevents setting an invalid mode

Type `int`

restart ()

Reset random seed and restart visualiser using *reset ()*

setDataset (name)

Set the `_datasetName` property from the *NameDatasetPopup*, unpause the *SimulatorVisualiser*, and return focus to the main window

Parameters `name` – `str` name of dataset to store training data in

tick ()

Call visualiser and panel tick frequently

Provides the visualiser modules

class `visualiser.SimulatorVisualiser` (*windowID*)

Bases: `object`

The visualiser of the simulator, showing a graphical representation of the highway.

Parameters `windowID` – `str` set as the `SDL_WINDOWID` environment variable. Used to embed the pygame window into the GUI.

`_backgrounds` = `None`

`list` storing the active `BackgroundPiece`

`_bgHeight` = `None`

(`int`) Height of the background sprite

`_cachedSprites` = `None`

(`boolean`) Whether all sprites have been cached

`_fps` = `None`

(`int`) Frames per second target

`_lastAction` = `None`

(`Actions`) last action performed

`_lastDraw` = `None`

(`float`) Timestamp of the last time the screen was drawn

`_lastSpawn` = `None`

(`int`) Last timestamp an obstacle vehicle was spawned

`_lastTick` = `None`

(`float`) Timestamp of the last `tick()` call

`_obstacles` = `None`

`list` storing the currently spawned `Obstacle` instances

`_targetDrawDelay` = `None`

(`float`) Targeted milliseconds in between screen draws

agentCar = None

The *Car* controlled by the agent

agentCollisions = None

int indicating the number of agent collisions in the current run

canvas = None

A reference to the *Surface* for drawing

car = None

The *Car* controlled by the expert

collisions = None

int indicating the number of expert collisions in the current run

distanceTravelled = None

int indicating the number metres travelled in the current run

doAct (*act, agent=False*)

Perform Action *act*

draw ()

Updates the canvas Attempts to achieve target FPS by blocking As such, it should run in its own thread so other things can be done in the background Should run in an endless loop to continuously redraw

fps

The current target frames per second

Getter Get the current FPS

Setter Set the current FPS. Calculates the necessary target draw interval

Type *int*

keyPress (*key*)

Handle key presses and perform the actions they map onto

lastActionTime = None

float Timestamp when the last action was performed

mode

The mode of the visualiser. See *mode* for details.

Getter Return the current set mode

Setter Set the current mode. Also sets the correct dimensions of the display, doubling the width if in compare mode

Type *int*

pause = None

bool indicating whether the visualiser is paused

reset ()

Restart the visualiser by resetting properties

sessionTime = None

float indicating the milliseconds since starting the current run

stateVector (*agent=False*)

Return a dictionary of features: {last action, current lane, distance of obstacles in three lanes, offroad} Some features have been 'pruned' from our decision tree because they did not affect the accuracy of the tree. These have been commented out for clarity.

tick()

This method tries to run at regular intervals of *TickRate* milliseconds. Performs update logic of the cars, obstacles, and background using time since the previous tick. Stores the current tick timestamp in *_lastTick*

togglePause()

Toggle pause state of the game

5.1 Submodules

5.2 visualiser.util module

Utility functions and classes

class `visualiser.util.Actions`

Bases: `enum.Enum`

Enumerations of the actions that the simulator can receive

LEFT = 2

Move into the lane to the left

NONE = 1

No action

PAUSE = 4

Toggle the pause state of the simulator

RIGHT = 3

Move into the lane to the right

class `visualiser.util.Pos` (*vehicle*, *x=0*, *y=0*)

Bases: `visualiser.util.Vector`

2D vector that automatically updates its vehicle's rect position

x

The x component of the vector :getter: Return x value :setter: Set x value, rounded to nearest `int`

y

The y component of the vector :getter: Return y value :setter: Set y value, rounded to nearest `int`

class `visualiser.util.Vector` (*x=0*, *y=0*)

Bases: `object`

A 2D vector, with arithmetic magic methods implemented

x = None

The x component of the vector

y = None

The y component of the vector

`visualiser.util.loadSprite` (*path*, *scale=1*)

Load an image as a `pygame.Surface`. Automatically caches loaded images in the background

Parameters

- **path** – (`str`) Path to desired image, relative to the `sprites/` directory
- **scale** – (`float`) Multiplied against the size of the image in order to scale as desired

5.3 visualiser.vehicles module

Vehicle classes used by the game

class `visualiser.vehicles.Car` (*game*)
Bases: `visualiser.vehicles.Vehicle`

The controlled car

Parameters `game` – The parent `SimulatorVisualiser`

tick (*dt*)

Car tick method, currently does nothing

class `visualiser.vehicles.Obstacle` (*game*)
Bases: `visualiser.vehicles.Vehicle`

An obstacle vehicle that the main car has to avoid

Parameters `game` – The parent `SimulatorVisualiser`

_agentCollided = `None`

`bool` indicating whether this vehicle has collided with the agent car

_collided = `None`

`bool` indicating whether this vehicle has collided with the user car

hasCollided

Getter Return whether the vehicle has just collided this tick. Updates `_collided`

hasCollidedAgent

Getter Returns whether the vehicle has started colliding with agent car this tick. Updates `_agentCollided`

speed

The virtual forward speed of the vehicle, in metres per second

Getter Get the vehicle speed

Setter Set the vehicle speed in m/s. Also sets the real velocity, which is relative to the user car

tick (*dt*)

Called regularly to perform update logic. Return `False` to indicate vehicle is entirely off screen and should be removed

`visualiser.vehicles.SPRITE_SCALES` = `{'car.png': 0.4, 'lorry.png': 0.75, 'police.png': 0.4}`
`dict` mapping obstacle sprites onto their respective `float` scales

class `visualiser.vehicles.Vehicle` (*game*)
Bases: `object`

A vehicle in the game

Parameters `game` – The parent `SimulatorVisualiser`

_sprite = `None`

`str` name of the sprite file, relative to the `sprites/obstacles` directory

_spriteScale = `None`

(`float`) Size multiplier of the sprite

colliding (*veh*)

Return whether this vehicle is colliding with `Vehicle veh`

draw (*canvas*)

Draw the vehicle

lane

The current lane of the vehicle

Getter Get the current lane**Setter** Set the lane of the vehicle. The vehicle is centered in its lane. Will only accept lanes available**Type** `int`**pos**

The position vector of the vehicle

Getter Get position**Setter** Set the position**Type** `Pos`**rect = None**A reference to the vehicle's `Rect`**speed**

The virtual forward speed of the vehicle, in metres per second

Getter Get the vehicle speed**Setter** Set the vehicle speed in m/s**sprite**The sprite object. Uses `_sprite` as the file name of the image to load**Getter** Return the sprite**Type** `pygame.Surface`**tick** (*dt*)

Called regularly to perform update logic. Updates vehicle position using its velocity

5.4 visualiser.visualiser module

The visualiser of the simulator - the game the user plays Uses the PyGame library

class `visualiser.visualiser.SimulatorVisualiser` (*windowID*)Bases: `object`

The visualiser of the simulator, showing a graphical representation of the highway.

Parameters `windowID` – `str` set as the `SDL_WINDOWID` environment variable. Used to embed the pygame window into the GUI.**`_backgrounds = None`**`list` storing the active `BackgroundPiece`**`_bgHeight = None`**`(int)` Height of the background sprite**`_cachedSprites = None`**`(boolean)` Whether all sprites have been cached

_fps = None
(*int*) Frames per second target

_lastAction = None
(*Actions*) last action performed

_lastDraw = None
(*float*) Timestamp of the last time the screen was drawn

_lastSpawn = None
(*int*) Last timestamp an obstacle vehicle was spawned

_lastTick = None
(*float*) Timestamp of the last *tick()* call

_obstacles = None
list storing the currently spawned *Obstacle* instances

_targetDrawDelay = None
(*float*) Targeted milliseconds in between screen draws

agentCar = None
The *Car* controlled by the agent

agentCollisions = None
int indicating the number of agent collisions in the current run

canvas = None
A reference to the *Surface* for drawing

car = None
The *Car* controlled by the expert

collisions = None
int indicating the number of expert collisions in the current run

distanceTravelled = None
int indicating the number metres travelled in the current run

doAct (*act, agent=False*)
Perform Action *act*

draw ()
Updates the canvas Attempts to achieve target FPS by blocking As such, it should run in its own thread so other things can be done in the background Should run in and endless loop to continuously redraw

fps
The current target frames per second

Getter Get the current FPS

Setter Set the current FPS. Calculates the necessary target draw interval

Type *int*

keyPress (*key*)
Handle key presses and perform the actions they map onto

lastActionTime = None
float Timestamp when the last action was performed

mode
The mode of the visualiser. See *mode* for details.

Getter Return the current set mode

Setter Set the current mode. Also sets the correct dimensions of the display, doubling the width if in compare mode

Type `int`

pause = None

`bool` indicating whether the visualiser is paused

reset ()

Restart the visualiser by resetting properties

sessionTime = None

`float` indicating the milliseconds since starting the current run

stateVector (*agent=False*)

Return a dictionary of features: {last action, current lane, distance of obstacles in three lanes, offroad}
Some features have been 'pruned' from our decision tree because they did not affect the accuracy of the tree. These have been commented out for clarity.

tick ()

This method tries to run at regular intervals of `TickRate` milliseconds. Performs update logic of the cars, obstacles, and background using time since the previous tick. Stores the current tick timestamp in `_lastTick`

togglePause ()

Toggle pause state of the game

CHAPTER 6

Indices and tables

- `genindex`
- `modindex`
- `search`

a

audri, 1

c

config, 3

d

data, 7

g

gui, 9

gui.conf, 9

gui.controls, 10

gui.main, 11

gui.panel, 11

gui.sim, 12

v

visualiser, 15

visualiser.util, 17

visualiser.vehicles, 18

visualiser.visualiser, 19

Symbols

- `_agentCollided` (visualiser.vehicles.Obstacle attribute), 18
 - `_backgrounds` (visualiser.SimulatorVisualiser attribute), 15
 - `_backgrounds` (visualiser.visualiser.SimulatorVisualiser attribute), 19
 - `_bgHeight` (visualiser.SimulatorVisualiser attribute), 15
 - `_bgHeight` (visualiser.visualiser.SimulatorVisualiser attribute), 19
 - `_cachedSprites` (visualiser.SimulatorVisualiser attribute), 15
 - `_cachedSprites` (visualiser.visualiser.SimulatorVisualiser attribute), 19
 - `_call()` (gui.panel.SimulatorPanel method), 12
 - `_collided` (visualiser.vehicles.Obstacle attribute), 18
 - `_cursorPos()` (gui.controls.ToolTip method), 10
 - `_enter()` (gui.controls.ToolTip method), 10
 - `_fps` (visualiser.SimulatorVisualiser attribute), 15
 - `_fps` (visualiser.visualiser.SimulatorVisualiser attribute), 19
 - `_hide()` (gui.controls.ToolTip method), 10
 - `_instances` (config.Singleton attribute), 5
 - `_keyPress()` (gui.main.MainWindow method), 11
 - `_labelControl()` (gui.conf.ConfigFrame method), 9
 - `_lastAction` (visualiser.SimulatorVisualiser attribute), 15
 - `_lastAction` (visualiser.visualiser.SimulatorVisualiser attribute), 20
 - `_lastDraw` (visualiser.SimulatorVisualiser attribute), 15
 - `_lastDraw` (visualiser.visualiser.SimulatorVisualiser attribute), 20
 - `_lastSpawn` (visualiser.SimulatorVisualiser attribute), 15
 - `_lastSpawn` (visualiser.visualiser.SimulatorVisualiser attribute), 20
 - `_lastTick` (visualiser.SimulatorVisualiser attribute), 15
 - `_lastTick` (visualiser.visualiser.SimulatorVisualiser attribute), 20
 - `_leave()` (gui.controls.ToolTip method), 10
 - `_load()` (gui.conf.ConfigFrame method), 9
 - `_makeLabel()` (gui.panel.SimulatorPanel method), 12
 - `_matches` (gui.conf.ConfigFrame attribute), 9
 - `_newRow()` (gui.panel.SimulatorPanel method), 12
 - `_obstacles` (visualiser.SimulatorVisualiser attribute), 15
 - `_obstacles` (visualiser.visualiser.SimulatorVisualiser attribute), 20
 - `_save()` (gui.conf.ConfigFrame method), 9
 - `_schedule()` (gui.controls.ToolTip method), 10
 - `_show()` (gui.controls.ToolTip method), 10
 - `_simulator` (gui.main.MainWindow attribute), 11
 - `_sprite` (visualiser.vehicles.Vehicle attribute), 18
 - `_spriteScale` (visualiser.vehicles.Vehicle attribute), 18
 - `_targetDrawDelay` (visualiser.SimulatorVisualiser attribute), 15
 - `_targetDrawDelay` (visualiser.visualiser.SimulatorVisualiser attribute), 20
 - `_tick()` (gui.main.MainWindow method), 11
 - `_unschedule()` (gui.controls.ToolTip method), 10
 - `_update()` (gui.controls.LabeledScale method), 10
- ## A
- `accept()` (gui.sim.NameDatasetPopup method), 12
 - `action()` (audri.Agent method), 1
 - Actions (class in visualiser.util), 17
 - Agent (class in audri), 1
 - `agentCar` (visualiser.SimulatorVisualiser attribute), 16
 - `agentCar` (visualiser.visualiser.SimulatorVisualiser attribute), 20
 - `agentCollisions` (visualiser.SimulatorVisualiser attribute), 16
 - `agentCollisions` (visualiser.visualiser.SimulatorVisualiser attribute), 20
 - audri (module), 1
- ## B
- `back()` (gui.main.MainWindow method), 11
- ## C
- `canvas` (visualiser.SimulatorVisualiser attribute), 16

canvas (visualiser.visualiser.SimulatorVisualiser attribute), 20
 Car (class in visualiser.vehicles), 18
 car (visualiser.SimulatorVisualiser attribute), 16
 car (visualiser.visualiser.SimulatorVisualiser attribute), 20
 CarScale (config.SimulatorConfig attribute), 5
 CarSpeed (config.SimulatorConfig attribute), 5
 colliding() (visualiser.vehicles.Vehicle method), 18
 collisions (visualiser.SimulatorVisualiser attribute), 16
 collisions (visualiser.visualiser.SimulatorVisualiser attribute), 20
 CompareText (config.GUIConfig attribute), 3
 ConfAppearance (config.GUIConfig attribute), 3
 ConfBackground (config.GUIConfig attribute), 3
 ConfCarScale (config.GUIConfig attribute), 3
 ConfCarSpeed (config.GUIConfig attribute), 3
 ConfExperiment (config.GUIConfig attribute), 3
 ConfFPS (config.GUIConfig attribute), 3
 config (module), 3
 ConfigFrame (class in gui.conf), 9
 ConfInnerPadding (config.GUIConfig attribute), 3
 ConfObstFreq (config.GUIConfig attribute), 3
 ConfObstScale (config.GUIConfig attribute), 3
 ConfObstSpeed (config.GUIConfig attribute), 3
 ConfOffroad (config.GUIConfig attribute), 3
 ConfRandomSeed (config.GUIConfig attribute), 3
 ConfRecordFreq (config.GUIConfig attribute), 3
 ConfRowPadding (config.GUIConfig attribute), 3
 ConfSave (config.GUIConfig attribute), 4
 ConfText (config.GUIConfig attribute), 4
 ConfTickrate (config.GUIConfig attribute), 4
 ConfTitle (config.GUIConfig attribute), 4
D
 data (module), 7
 DATA_DIR (in module data), 7
 DATA_ORDER (in module data), 7
 dataExists() (in module data), 7
 DataText (config.GUIConfig attribute), 4
 distanceTravelled (visualiser.SimulatorVisualiser attribute), 16
 distanceTravelled (visualiser.visualiser.SimulatorVisualiser attribute), 20
 doAct() (visualiser.SimulatorVisualiser method), 16
 doAct() (visualiser.visualiser.SimulatorVisualiser method), 20
 draw() (visualiser.SimulatorVisualiser method), 16
 draw() (visualiser.vehicles.Vehicle method), 18
 draw() (visualiser.visualiser.SimulatorVisualiser method), 20

F

finish() (gui.sim.Simulator method), 12
 focus() (gui.main.MainWindow method), 11
 focus() (gui.sim.Simulator method), 12
 Font (config.GUIConfig attribute), 4
 FontBold (config.GUIConfig attribute), 4
 FontHeading (config.GUIConfig attribute), 4
 FontMonospace (config.GUIConfig attribute), 4
 FPS (config.SimulatorConfig attribute), 5
 fps (visualiser.SimulatorVisualiser attribute), 16
 fps (visualiser.visualiser.SimulatorVisualiser attribute), 20

G

get() (gui.controls.LabeledScale method), 10
 gui (module), 9
 gui.conf (module), 9
 gui.controls (module), 10
 gui.main (module), 11
 gui.panel (module), 11
 gui.sim (module), 12
 GUIConfig (class in config), 3

H

hasCollided (visualiser.vehicles.Obstacle attribute), 18
 hasCollidedAgent (visualiser.vehicles.Obstacle attribute), 18
 Height (config.GUIConfig attribute), 4

K

keyPress() (gui.sim.Simulator method), 12
 keyPress() (visualiser.SimulatorVisualiser method), 16
 keyPress() (visualiser.visualiser.SimulatorVisualiser method), 20

L

LabeledScale (class in gui.controls), 10
 lane (visualiser.vehicles.Vehicle attribute), 19
 LaneWidth (config.SimulatorConfig attribute), 5
 lastActionTime (visualiser.SimulatorVisualiser attribute), 16
 lastActionTime (visualiser.visualiser.SimulatorVisualiser attribute), 20
 LEFT (visualiser.util.Actions attribute), 17
 load() (audri.Agent method), 1
 loadData() (in module data), 7
 loadSprite() (in module visualiser.util), 17

M

MainButtonXPad (config.GUIConfig attribute), 4
 MainButtonYPad (config.GUIConfig attribute), 4
 MainFrame (class in gui.main), 11
 MainWindow (class in gui.main), 11

mode (gui.sim.Simulator attribute), 12
mode (visualiser.SimulatorVisualiser attribute), 16
mode (visualiser.visualiser.SimulatorVisualiser attribute), 20
MODEL_DIR (in module data), 7
modelExists() (in module data), 7
ModeText (config.GUIConfig attribute), 4

N

NameDatasetPopup (class in gui.sim), 12
NONE (visualiser.util.Actions attribute), 17

O

Obstacle (class in visualiser.vehicles), 18
ObstacleInterval (config.SimulatorConfig attribute), 5
ObstacleSpeed (config.SimulatorConfig attribute), 5
OffroadAllowed (config.SimulatorConfig attribute), 5
OffroadWidth (config.SimulatorConfig attribute), 5

P

PanelWidth (config.GUIConfig attribute), 4
pause (visualiser.SimulatorVisualiser attribute), 16
PAUSE (visualiser.util.Actions attribute), 17
pause (visualiser.visualiser.SimulatorVisualiser attribute), 21
PixelMetreRatio (config.SimulatorConfig attribute), 5
Pos (class in visualiser.util), 17
pos (visualiser.vehicles.Vehicle attribute), 19

R

RandomSeed (config.SimulatorConfig attribute), 5
RecordInterval (config.SimulatorConfig attribute), 5
rect (visualiser.vehicles.Vehicle attribute), 19
reset() (visualiser.SimulatorVisualiser method), 16
reset() (visualiser.visualiser.SimulatorVisualiser method), 21
restart() (gui.sim.Simulator method), 13
RIGHT (visualiser.util.Actions attribute), 17

S

save() (audri.Agent method), 1
saveData() (in module data), 7
ScrollBackground (config.SimulatorConfig attribute), 5
sessionTime (visualiser.SimulatorVisualiser attribute), 16
sessionTime (visualiser.visualiser.SimulatorVisualiser attribute), 21
set() (gui.controls.LabeledScale method), 10
setDataset() (gui.sim.Simulator method), 13
SimPopupDupWarn (config.GUIConfig attribute), 4
SimPopupPad (config.GUIConfig attribute), 4
SimPopupText (config.GUIConfig attribute), 4
SimPopupTitle (config.GUIConfig attribute), 4
Simulator (class in gui.sim), 12

SimulatorConfig (class in config), 5
SimulatorPanel (class in gui.panel), 11
SimulatorVisualiser (class in visualiser), 15
SimulatorVisualiser (class in visualiser.visualiser), 19
Singleton (class in config), 5
speed (visualiser.vehicles.Obstacle attribute), 18
speed (visualiser.vehicles.Vehicle attribute), 19
sprite (visualiser.vehicles.Vehicle attribute), 19
SPRITE_SCALES (in module visualiser.vehicles), 18
startSim() (gui.main.MainFrame method), 11
stateVector() (visualiser.SimulatorVisualiser method), 16
stateVector() (visualiser.visualiser.SimulatorVisualiser method), 21
Subtitle (config.GUIConfig attribute), 4

T

TestText (config.GUIConfig attribute), 4
Theme (config.GUIConfig attribute), 4
tick() (gui.panel.SimulatorPanel method), 12
tick() (gui.sim.Simulator method), 13
tick() (visualiser.SimulatorVisualiser method), 16
tick() (visualiser.vehicles.Car method), 18
tick() (visualiser.vehicles.Obstacle method), 18
tick() (visualiser.vehicles.Vehicle method), 19
tick() (visualiser.visualiser.SimulatorVisualiser method), 21
TickRate (config.SimulatorConfig attribute), 5
Title (config.GUIConfig attribute), 4
togglePause() (visualiser.SimulatorVisualiser method), 17
togglePause() (visualiser.visualiser.SimulatorVisualiser method), 21
ToolTip (class in gui.controls), 10
TooltipWidth (config.GUIConfig attribute), 4
train() (audri.Agent method), 1

V

Vector (class in visualiser.util), 17
Vehicle (class in visualiser.vehicles), 18
visualiser (module), 15
visualiser.util (module), 17
visualiser.vehicles (module), 18
visualiser.visualiser (module), 19
VisualiserWidth (config.GUIConfig attribute), 4

X

x (visualiser.util.Pos attribute), 17
x (visualiser.util.Vector attribute), 17

Y

y (visualiser.util.Pos attribute), 17
y (visualiser.util.Vector attribute), 17