TRACI4MATLAB - USER'S MANUAL

Project:	Modelamiento y control de tráfico en la ciudad de Medellín - Segunda Etapa
Acronym:	MOYCOT E2
Web site:	`www.moycot.org`

UNIVERSIDAD NACIONAL DE COLOMBIA

Authors: Andrés Acosta, Jorge Espinosa, Jairo Espinosa.

Contents

Abstract **3**

1 What is TraCI4Matlab? **4**

1.1 Citing TraCI4Matlab 4

2 Installing TraCI4Matlab **6**

2.1 Prerequisites 6

2.2 Installation 6

2.2.1 Step 1: Setting up the environment 6

2.2.2 Step 2: Getting TraCI4Matlab 9

2.2.3 Step 3: Adding the additional dependencies to the Matlab's static Java path 9

2.2.4 Step 4: Adding TraCI4Matlab to the Matlab's path 9

2.2.5 Step 5: Testing TraCI4Matlab 10

3 Using TraCI4Matlab **13**

3.1 Building the simulation scenario in SUMO 13

3.2 Developing the application in Matlab 13

3.2.1 Step 1: Executing SUMO in server mode from Matlab and initializing the connection 13

3.2.2 Step 2: Developing the application 14

3.2.3 Step 3: Closing the connection 16

Bibliography **16**

Abstract

This user's manual describes the installation procedure of the TraCI4Matlab package and a brief description of its usage. TraCI4Matlab is an implementation of the Traffic Control Interface (TraCI) for the Matlab®programming language, which allows the interaction with the Simulation of Urban Mobility (SUMO) microscopic road traffic simulator.

Chapter 1

What is TraCI4Matlab?

TraCI4Matlab is an Application Programming Interface (API) developed in Matlab which allows the communication between any application developed in this language and the Simulation of Urban Mobility (SUMO) microscopic road traffic simulator. The functions comprising TraCI4Matlab implement the Traffic Control Interface (TraCI) application level protocol which is built on top of the TCP/IP stack, so that the application developed in Matlab, which is the client, can access and modify the simulation environment provided by the server (SUMO). TraCI4Matlab allows controlling SUMO objects such as vehicles, traffic lights, junctions, etc, enabling applications like traffic lights predictive control, dynamic route assignment and vehicular communications, among others.

1.1 Citing TraCI4Matlab

If you use TraCI4Matlab in your research reports, articles and conferences, please use this book chapter [1] to cite it, as follows:

```
@incollection{acosta_etal_2015,
        title = {{TraCI}4Matlab: {Enabling} the
        {Integration} of the {SUMO} {Road} {Traffic}
        {Simulator} and {Matlab} {Through} a
        {Software} {Re}-engineering {Process}},
        url = {http://link.springer.com/
        10.1007/978-3-319-15024-6_9},
        urldate = {2017-03-06},
        booktitle = {Modeling {Mobility} with {Open}
        {Data}},
        publisher = {Springer},
        author = {Acosta, Andr\'es F. and Espinosa,
        Jorge E. and Espinosa, Jairo},
        year = {2015},
        pages = {155--170}
}
```

Chapter 2

Installing TraCI4Matlab

2.1 Prerequisites

- Operating System: Windows 7 or higher
- Matlab®R2012b or higher
- SUMO 0.19.0 or higher. Installation instructions of SUMO can be found here

Note: TraCI4Matlab is designed to preserve the same syntax of the TraCI-Python client, which is part of the SUMO suite. If you have little programming experience, we encourage you to get familiar with TraCI-Python before starting with TraCI4Matlab.

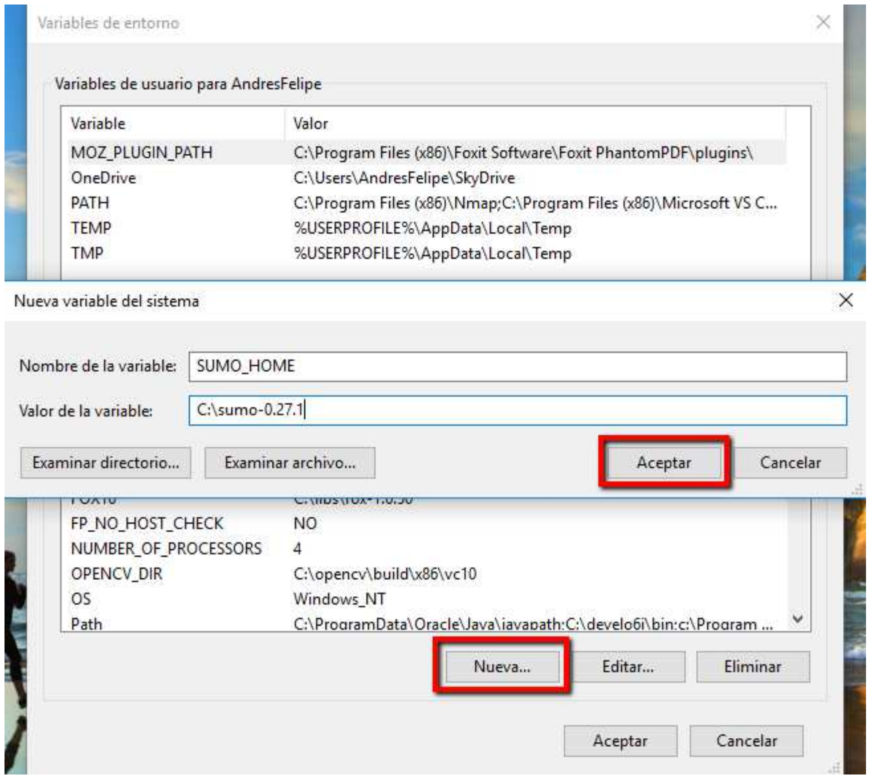
2.2 Installation

2.2.1 Step 1: Setting up the environment

Set up the `SUMO_HOME` environment variable with system scope, with a value corresponding to the root directory of the SUMO installation. For example, if the SUMO version 0.27.1 was installed in `C:`, then the `SUMO_HOME` environment variable would have a value of `C:\sumo-0.27.1`. Environment variables can be configured as follows:

- Open the windows explorer, right click on "My computer" and click the properties option, as showed in figure 2.2.
- On the left side of the window that opens, click in the link "Advanced system configuration", then click in the button "Environment variables" and in the "System variables" field, click on the button "New", configure the variable according to your SUMO installation, and click the "Accept" button, as showed in figure 2.1.

- Again in the "System variables" field, look for the "path" variable and add the route to the bin directory of the SUMO installation. This can be accomplished in Windows 10 by double clicking the "path" variable then selecting the "New" button as shown in figure 2.3, or in Windows 7 by adding a semi-colon to the current value and then the corresponding route. Finally, click on the "Accept" button.



Variables de usuario para AndresFelipe

Variable	Valor
MOZ_PLUGIN_PATH	C:\Program Files (x86)\Foxit Software\Foxit PhantomPDF\plugins\
OneDrive	C:\Users\AndresFelipe\SkyDrive
PATH	C:\Program Files (x86)\Nmap;C:\Program Files (x86)\Microsoft VS C...
TEMP	%USERPROFILE%\AppData\Local\Temp
TMP	%USERPROFILE%\AppData\Local\Temp

Nueva variable del sistema

Nombre de la variable: SUMO_HOME

Valor de la variable: C:\sumo-0.27.1

FP_NO_HOST_CHECK	NO
NUMBER_OF_PROCESSORS	4
OPENCV_DIR	C:\opencv\build\x86\vc10
OS	Windows_NT
Path	C:\ProgramData\Oracle\Java\javapath;C:\develo6i\bin;c:\Program ...

Figure 2.1: Creating the SUMO_HOME variable.

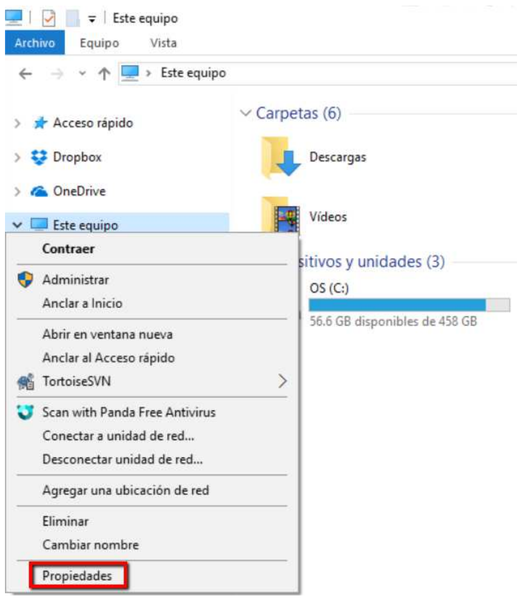
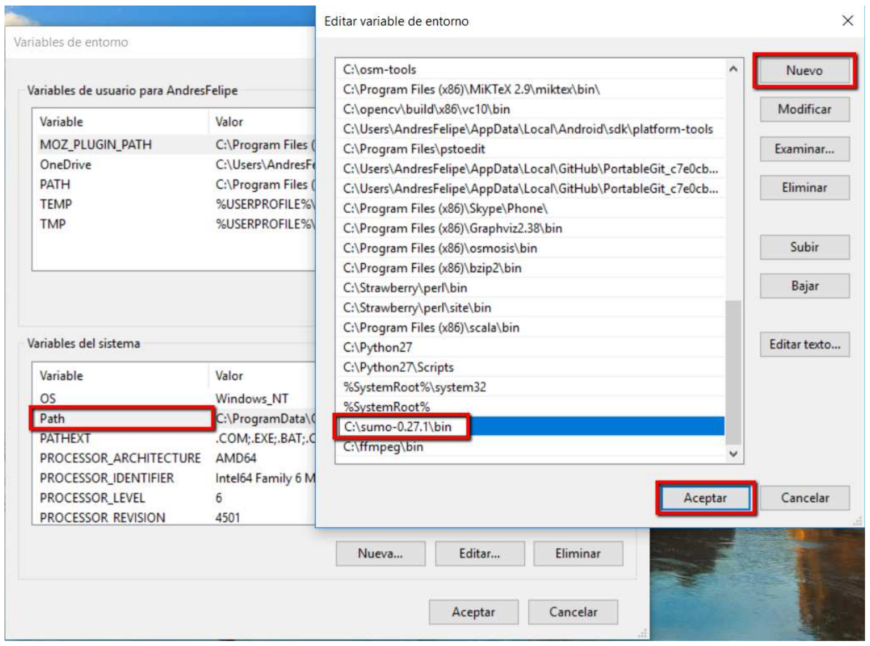


Figure 2.2: Access to the system properties.



Variables de usuario para AndresFelipe

Variable	Valor
MOZ_PLUGIN_PATH	C:\Program Files (
OneDrive	C:\Users\AndresFe
PATH	C:\Program Files (
TEMP	%USERPROFILE%\
TMP	%USERPROFILE%\

Variables del sistema

Variable	Valor
OS	Windows_NT
Path	C:\ProgramData\(
PATHEXT	.COM;.EXE;.BAT;.C
PROCESSOR_ARCHITECTURE	AMD64
PROCESSOR_IDENTIFIER	Intel64 Family 6 M
PROCESSOR_LEVEL	6
PROCESSOR_REVISION	4501

```
C:\osm-tools
C:\Program Files (x86)\MiKTeX 2.9\miktex\bin\
C:\opencv\build\x86\vc10\bin
C:\Users\AndresFelipe\AppData\Local\Android\sdk\platform-tools
C:\Program Files\pstoedit
C:\Users\AndresFelipe\AppData\Local\GitHub\PortableGit_c7e0cb...
C:\Users\AndresFelipe\AppData\Local\GitHub\PortableGit_c7e0cb...
C:\Program Files (x86)\Skype\Phone\
C:\Program Files (x86)\Graphviz2.38\bin
C:\Program Files (x86)\osmosis\bin
C:\Program Files (x86)\bzip2\bin
C:\Strawberry\perl\bin
C:\Strawberry\perl\site\bin
C:\Program Files (x86)\scala\bin
C:\Python27
C:\Python27\Scripts
%SystemRoot%\system32
%SystemRoot%
C:\sumo-0.27.1\bin
C:\ffmpeg\bin
```

Figure 2.3: Editing the Windows path.

2.2.2 Step 2: Getting TraCI4Matlab

TraCI4Matlab can be obtained in three ways:

- TraCI4Matlab comes with the SUMO installation since version 0.20.0 and is located in the `SUMO_HOME/tools/contributed` folder. This version is obtained from the Github repository of TraCI4Matlab
- TraCI4Matlab is also in a Subversion repository, where there are the most up-to-date versions. We recommend to use the TortoiseSVN client to obtain TraCI4Matlab.
- Finally, a compressed zip with TraCI4Matlab can be obtained from Matlab Central

2.2.3 Step 3: Adding the additional dependencies to the Matlab's static Java path

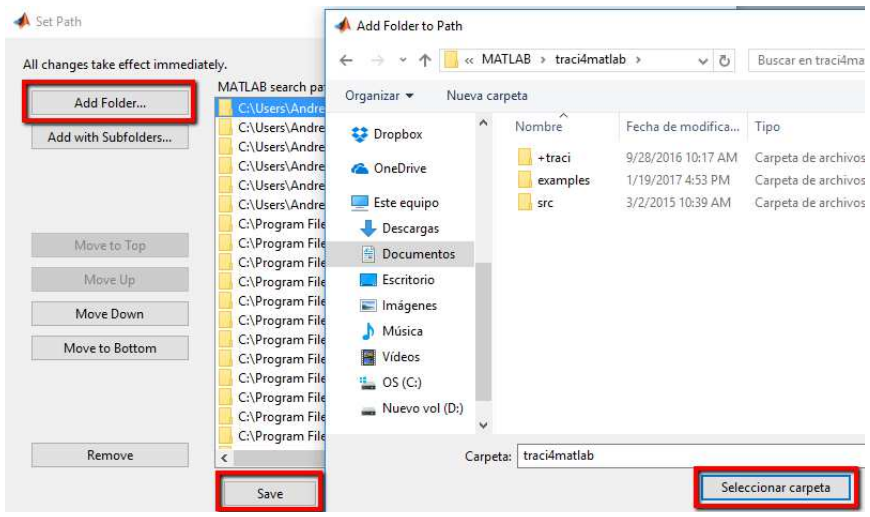
In your preferred text editor, create a text file containing the path to the `traci4matlab.jar` file, as follows:

```
TRACI4MATLAB_HOME/traci4matlab.jar
```

Where `TRACI4MATLAB_HOME` is the path to the root folder of TraCI4Matlab obtained in the previous step. Now, save this file with the name `javaclasspath.txt` in the preferences directory of Matlab, which can be identified using the `prefdir` command.
Note: You must restart Matlab so that the new static Java path takes effect.

2.2.4 Step 4: Adding TraCI4Matlab to the Matlab's path

In the Matlab command window, type the `pathtool` command, which opens a new window. Click on the "Add folder . . . " button and browse the `TRACI4MATLAB_HOME` directory, then click the "Select folder" button, as showed in figure 2.4. Finally, click on the "Save" button to save the Matlab path.



Nombre	Fecha de modifica...	Tipo
+traci	9/28/2016 10:17 AM	Carpeta de archivos
examples	1/19/2017 4:53 PM	Carpeta de archivos
src	3/2/2015 10:39 AM	Carpeta de archivos

Carpeta: traci4matlab

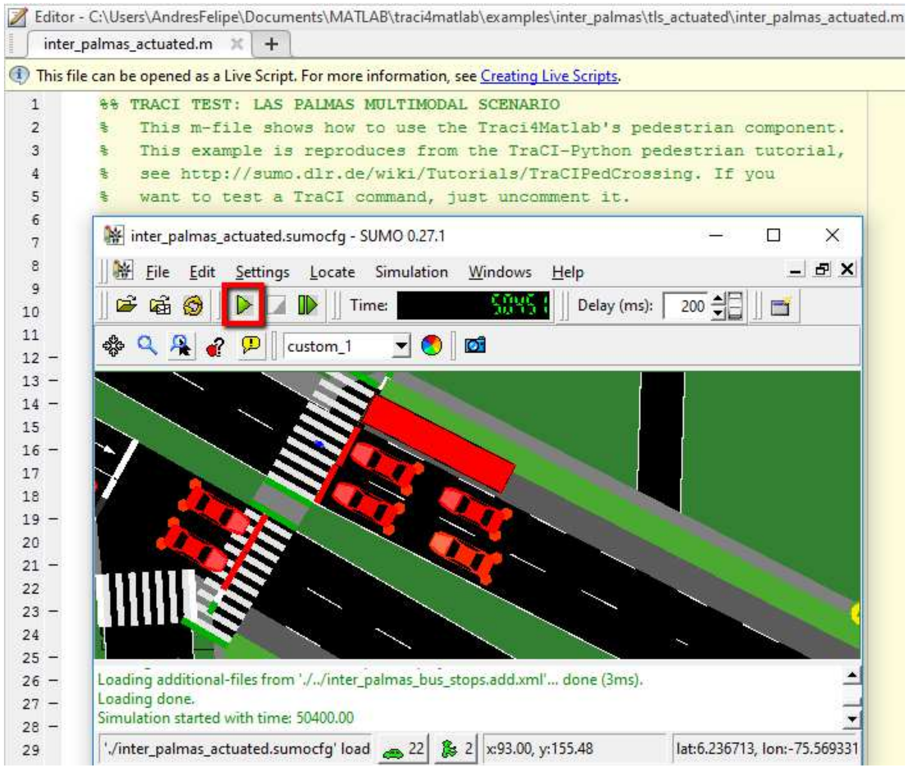
Figure 2.4: Adding TraCI4Matlab to the Matlab path.

2.2.5 Step 5: Testing TraCI4Matlab

TraCI4Matlab includes three test examples for verifying its correct installation and functioning. The following list describes these examples, starting from the most simple:

- `traci_test.m`. This test script is located in `TRACI4MATLAB_HOME/examples` and returns the SUMO and the TraCI versions. We recommend to use this example in order to test the TraCI4Matlab installation.

- `inter_palmas_actuated.m`. This example is located in `TRACI4MATLAB_HOME/examples/inter_palmas/tls_actuated`. This scenario shows a pedestrian crossing with an actuated traffic lights system (TLS) that simulates pedestrians pushing a button for requesting the right of way using the same algorithm of the TraCIPedCrossing tutorial. Once this script is executed, the Graphical User Interface of SUMO (sumo-gui) should open. By clicking the "play" button, the simulation starts, as showed in figure 2.5. When the simulation finishes, the number of pedestrians waiting on the sidewalks and the traffic lights phase for pedestrians are plotted, as showed in figure 2.6.


Editor - C:\Users\AndresFelipe\Documents\MATLAB\traci4matlab\examples\inter_palmas\tls_actuated\inter_palmas_actuated.m
inter_palmas_actuated.m
This file can be opened as a Live Script. For more information, see Creating Live Scripts.
inter_palmas_actuated.sumocfg - SUMO 0.27.1
File Edit Settings Locate Simulation Windows Help
Time:
Delay (ms): 200
custom_1
'./inter_palmas_actuated.sumocfg' load
22
2
x:93.00, y:155.48
lat:6.236713, lon:-75.569331

```
%% TRACI TEST: LAS PALMAS MULTIMODAL SCENARIO
%   This m-file shows how to use the Traci4Matlab's pedestrian component.
%   This example is reproduces from the TraCI-Python pedestrian tutorial,
%   see http://sumo.dlr.de/wiki/Tutorials/TraCIPedCrossing. If you
%   want to test a TraCI command, just uncomment it.
```

```
Loading additional-files from './../inter_palmas_bus_stops.add.xml'... done (3ms).
Loading done.
Simulation started with time: 50400.00
```

Figure 2.5: TraCI4Matlab test in sumo-gui.

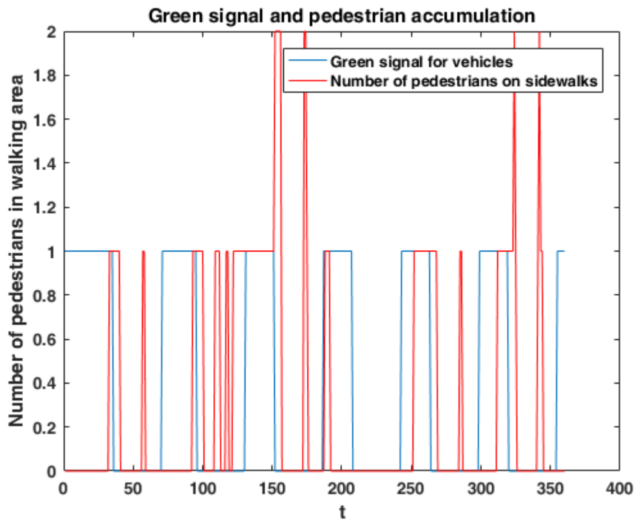


Figure 2.6: TraCI4Matlab test results in Matlab.

- `traci_test2.m`. This "advanced" example, located in `TRACI4MATLAB_HOME/examples`, aims to test the majority of the TraCI4Matlab functions. This scenario utilizes the TraCI4TrafficLights tutorial, which consists of an intersection where "priority" vehicles going from North to South are guaranteed the right of way through an actuated TLS. Hence, an important prerequisite is that the user must run the TraCI4TrafficLights scenario using the TraCI-Python client so that the routes file is generated.

Chapter 3

Using TraCI4Matlab

3.1 Building the simulation scenario in SUMO

To use TraCI4Matlab, the first step is to set up a simulation scenario in SUMO. The process for building a simulation scenario is out of the scope of this manual, but an official tutorial can be found in the SUMO documentation.

3.2 Developing the application in Matlab

3.2.1 Step 1: Executing SUMO in server mode from Matlab and initializing the connection

Any application in Matlab that uses TraCI4Matlab must start by executing the commands `sumo` if it is desired to execute the simulation without visualization or `sumo-gui` if it is desired to execute SUMO in GUI mode, and specifying as a parameter the path where the configuration file of the SUMO scenario obtained in the previous step is located. This is achieved through the Matlab's `system` command, as showed in the code listing 3.1. Furthermore, it is important to specify that SUMO will run in server mode, using the `--remote-port` option followed by an available TCP port. If you do not know how to automatically obtain an available TCP port, you can use the ports 8813 or 8873.

Listing 3.1: Executing SUMO in server mode from Matlab

```
clear;
close('all');
clc;

import traci.constants

% Launch SUMO in server mode and initialize the TraCI connection
system(['sumo-gui -c ' '"' scenarioPath '"' ' --remote-port 8873
    --start&']);
traci.init();
```

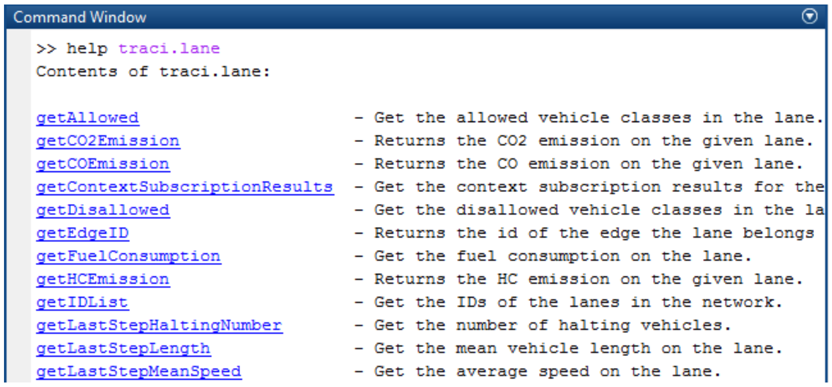
Optionally, the command in line 5 can be added to facilitate referencing the TraCI constants in case that TraCI subscriptions are made. Thus, for instance, to reference the command `LAST_STEP_VEHICLE_NUMBER` of the TraCI constants, it's enough to write `constants.LAST_STEP_VEHICLE_NUMBER` instead of `traci.constants.LAST_STEP_VEHICLE_NUMBER`.
After initializing SUMO in server mode, the connection must be established with the function `traci.init()`. If the SUMO server was configured to use the 8873 port, this function does not need additional parameters, otherwise the port must be specified as a parameter. Additional parameters of the `traci.init()` function are described in the function documentation, which can be accessed by typing `help traci.init` in the Matlab command window.

3.2.2 Step 2: Developing the application

Often, TraCI4Matlab applications include a main loop in which the simulation is executed iteratively through the `traci.simulationStep` command. In this loop, the attributes of the objects in the SUMO simulation are accessed and modified. The SUMO objects are grouped in fifteen domains: areal, edge, gui, inductionloop, junction, lane, multientryexit, person, poi, polygon, route, simulation, trafficlights, vehicle and vehicletype. the general structure for accessing or modifying a SUMO object is: `traci.<domain>.<get/set_wrapper()>`, where `domain` can take any of the domains listed previously and `get/set_wrapper()` are the functions for accessing the values (`get`) or modifying (`set`) the attributes of the object of interest. For example, if the value of the velocity of the vehicle with ID `veh_1` in the current time step is required, the following command can be executed:

```
current_speed_veh1 = traci.vehicle.getSpeed('veh_1')
```

To obtain a list of all the commands related to a specific domain, type `help traci.<domain>` in the Matlab command window, where `domain` can take any of the values listed previously. Figure 3.1 shows an example in the case of the lane domain. Further help regarding a function can be obtained by clicking it.



```
>> help traci.lane
Contents of traci.lane:

getAllowed                    - Get the allowed vehicle classes in the lane.
getCO2Emission                - Returns the CO2 emission on the given lane.
getCOEmission                 - Returns the CO emission on the given lane.
getContextSubscriptionResults - Get the context subscription results for the
getDisallowed                 - Get the disallowed vehicle classes in the la
getEdgeID                     - Returns the id of the edge the lane belongs
getFuelConsumption            - Get the fuel consumption on the lane.
getHCEmission                 - Returns the HC emission on the given lane.
getIDList                     - Get the IDs of the lanes in the network.
getLastStepHaltingNumber      - Get the number of halting vehicles.
getLastStepLength             - Get the mean vehicle length on the lane.
getLastStepMeanSpeed          - Get the average speed on the lane.
```

Figure 3.1: Obtaining a list of the functions related to a SUMO object.

The main loop of the application can be executed using a fixed time or a condition such as until all vehicles of the simulation have arrived to their destinations. In the second case, the `traci.simulation.getMinExpectedNumber` function can be used, as showed in the following code listing, where the main loop has been highlighted:

```
while traci.simulation.getMinExpectedNumber()>0
    traci.simulationStep();
    programPointer = min(programPointer+1, length(PROGRAM));

    % Get the number of vehicles that passed through the induction
    % loop in the last simulation step
    numPriorityVehicles = traci.inductionloop.
        getLastStepVehicleNumber('0');

    % Change the phase of the traffic light if a vehicle passed
    % through the induction loop
    if numPriorityVehicles > 0
        if programPointer == length(PROGRAM)
            programPointer = 1;
        elseif ~strcmp(PROGRAM(programPointer), WEYELLOW)
            programPointer = 4;
        end
    end

    traci.trafficlights.setRedYellowGreenState('0', PROGRAM{...
        programPointer});

end
```

TraCI4Matlab includes functions for making TraCI subscriptions. TraCI subscriptions allow retrieving several attributes of a SUMO object using a single command. To use TraCI subscriptions it is necessary to know the TraCI constants containing the codes of the different attributes related to a TraCI subscription. These constants can be found by typing `edit traci.constants` in the Matlab command window and locating the `VARIABLE TYPES` field. For example, suppose that it is desired to make a TraCI subscription to access the values of the attributes `LAST_STEP_VEHICLE_NUMBER` and `LAST_STEP_MEAN_SPEED` of the induction loop with ID `'0'`. In this case, the following command can be used:

```
traci.inductionloop.subscribe('0', {constants.
    LAST_STEP_VEHICLE_NUMBER, constants.LAST_STEP_MEAN_SPEED});
```

Note that thanks to the command `import traci.constants` at the beginning of the script, as explained in the previous step, the subsequent commands can be shorter.
The following commands allow accessing the values related to the TraCI subscription:

```
indloopSubsResults = traci.inductionloop.getSubscriptionResults('0
    ');
numVehicles = indloopSubsResults(constants.
    LAST_STEP_VEHICLE_NUMBER);
lsms = indloopSubsResults(constants.LAST_STEP_MEAN_SPEED);
```

Note that the results are stored in a handle variable `indloopSubsResults` which is indexed with the TraCI constants to which the subscription was made.
Note also that, using the getter commands of TraCI4Matlab in the the main loop of the Matlab application, the user can collect variables of interest in vectors or other Matlab arrays for plotting, analyzing and storing them later. Furthermore, many Matlab routines can be used inside the main loop in order to make decisions related to the routing of vehicles, the traffic lights states or the vehicular wireless network protocols.

3.2.3 Step 3: Closing the connection

Finally, the connection to the SUMO server is closed using the `traci.close()` command.

Bibliography

[1] A. F. Acosta, J. E. Espinosa, and J. Espinosa, "TraCI4matlab: Enabling the Integration of the SUMO Road Traffic Simulator and Matlab® Through a Software Re-engineering Process," in *Modeling Mobility with Open Data*, pp. 155–170, Springer, 2015.