**User Manual**

For Multiagent Control of Traffic Signals

Version 1.0

Submitted in partial fulfillment of the requirements of the degree of MSE

Bryan Nehl

CIS 895 – MSE Project

Kansas State University

Table of Contents

[1 Intro 3](#_Toc323025039)

[2 Software Required to Run 3](#_Toc323025040)

[2.1 RabbitMQ 3](#_Toc323025041)

[2.2 MongoDB 3](#_Toc323025042)

[2.3 Python 3](#_Toc323025043)

[2.4 Simulation for Urban MObility 3](#_Toc323025044)

[3 Configuration 3](#_Toc323025045)

[3.1 One time setup of database 3](#_Toc323025046)

[3.2 One time setup of RabbitMQ infrastructure 3](#_Toc323025047)

[3.3 RabbitMQ Exchange Initiation 4](#_Toc323025048)

[4 Running Simulations 4](#_Toc323025049)

[4.1 Running Agents 4](#_Toc323025050)

[4.2 Collecting Metrics 4](#_Toc323025051)

[4.3 Running Fixed Response Simulation – No Agents 5](#_Toc323025052)

[4.4 Varying the Simulation – the Communications Agent 5](#_Toc323025053)

# Intro

This user manual will inform you of the software needed by the MACTS in order to run simulations. You’ll learn what one time and routine configuration needs to be done. Next you’ll see how to start up the agents. Then you’ll learn how to collect metrics and parse them. In the final section you will be presented with information on how to create your own agents.

# Software Required to Run

There are several key pieces of software you need to have installed on your machine in order to run MACTS.

## RabbitMQ

RabbitMQ is a messaging broker server. This needs to be installed and operating in order for the communication to happen between the parts of the system.

## MongoDB

MongoDB is a document store server. This needs to be installed and operating in order for the metrics to be stored at the end of a simulation run.

## Python

They Python language interpreter 2.7 and supporting libraries: pika and pymongo should be installed. Pika is for interacting with RabbitMQ and PyMongo is for working with MongoDB.

## Simulation for Urban MObility

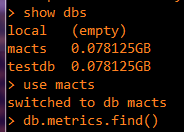
The Simulation for Urban Mobility version 14.0 should be installed on your machine. The system will obtain different results or not function with other versions of the software.

# Configuration

## One time setup of database

Once you have MongoDB installed you need to do some one time configuration.

Open the mongo shell and issue the following commands:



show dbs

use macts

db.metrics.find()

This will create the macts document storage area (db) and an associated document collection, metrics.

## One time setup of RabbitMQ infrastructure

There are several commands in the source folder for working with RabbitMQ.

The rabbitmq\_init.bat file needs to be run one time to create the needed users and permissions. To remove the MACTS exchanges and users run the rabbitmq\_clean.bat command.

## RabbitMQ Exchange Initiation

To create the exchanges that are needed by the simulation, the command:

Python rabbitmq\_create\_exchanges.py needs to be run. This command will need to be run anytime the user logouts or the system reboots. The exchanges are not set up to be persistent between power cycles.

# Running Simulations

Use the start-command-line.bat that is located in the source directory to start all command prompts. This places the SUMO-TraCI libraries on the path.

## Running Agents

The planning and safety agents that are part of the network should be started in their own individual command prompt boxes before starting the Communications Agent. The safety agents are called by the planning agents. You do not directly instantiate and run them. The basic format of the command is python <agentname.py>. For example:

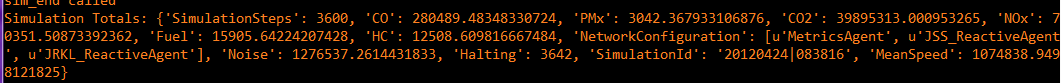
python JSS\_Reactive\_Agent.py

python JRKL\_Reactive\_Agent.py

## Collecting Metrics

In order to collect metrics the metrics agent must be online: python MetricsAgent.py

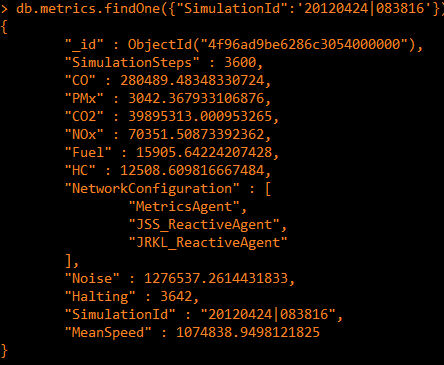
At the end of a simulation run, the metrics agent will output the simulation run totals to the screen. It will also save them to the database. The output looks like this:



An important attribute to note is the SimulationId. This can be used to retrieve prettier looking results from MongoDB as well as look at them again later.



Using the SimulationId and the findOne command of MongoDB to retrieve the simulation run data:



Note that the Network configuration is also stored along with the metrics.

## Running Fixed Response Simulation – No Agents

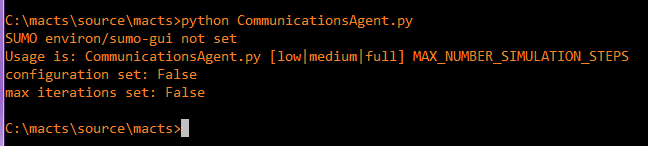
If you would like to run the simulation without any agents you need to set the FIXED\_PLAN\_NO\_SAFETY\_AGENTS = True in the CommunicationsAgent.py file. You would use this setting in combination with the metrics agent to obtain baseline metrics.

## Varying the Simulation – the Communications Agent

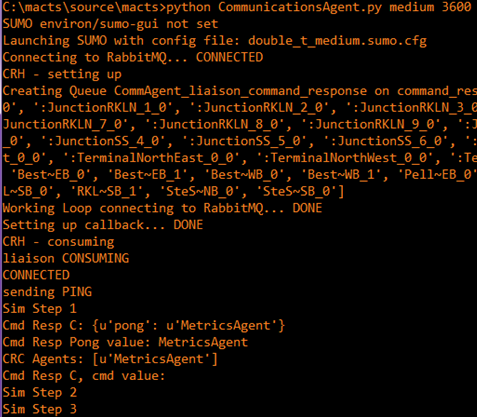
Make sure that FIXED\_PLAN\_NO\_SAFETY\_AGENTS = False and that the EXPECTED\_NUMBER\_SAFETY\_AGENT\_COMMANDS is properly set. The setting below would be for only safety agent in the system.



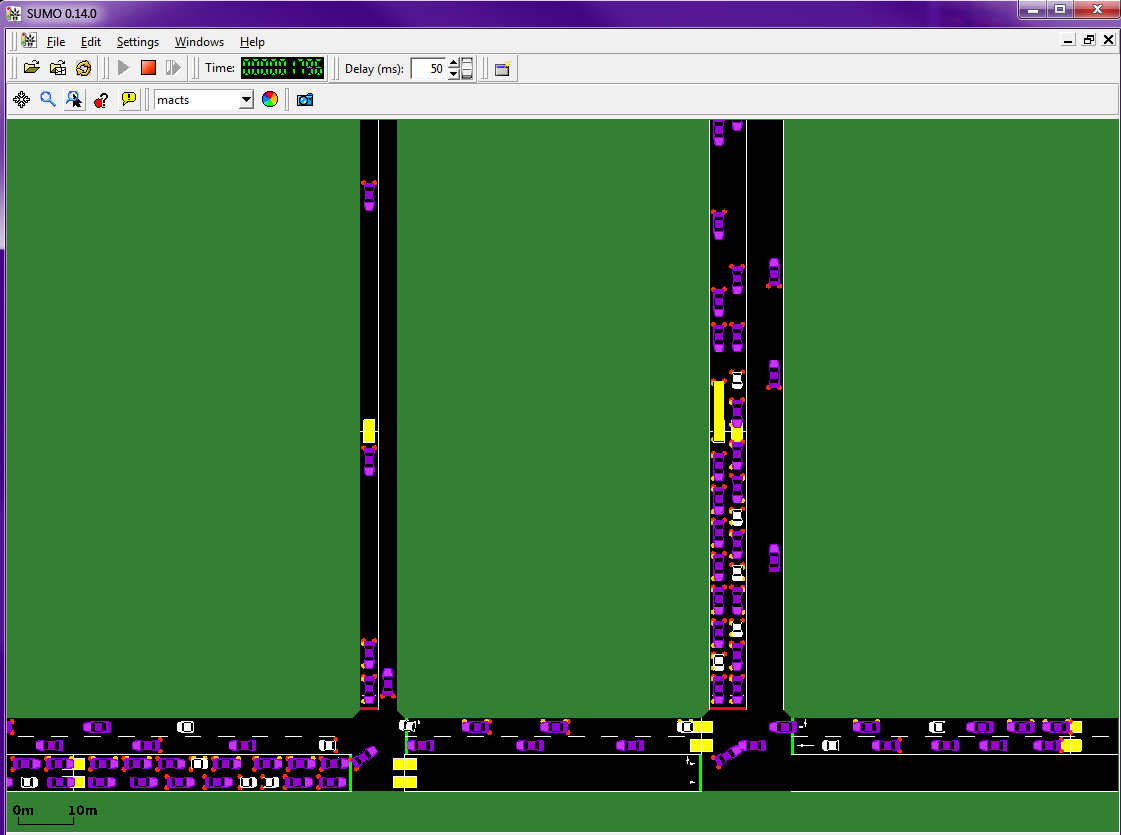
The following screen shows the command line usage for the Communications Agent. You need to specify network size and the number of seconds to simulate.



In the screen shot below the communications agent has been started with a medium network size for 3600 second simulation. The output is to some extent determined by the varying of a verbose\_level field in the code classes. For this configuration we see the road network configuration. Also, the agent network discovery is seen. Then the simulation loop starts and we see the progress being displayed.

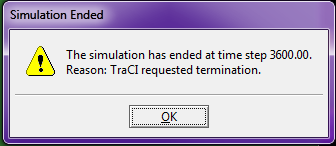


The communications agent will launch the SUMO-GUI and we’ll see something like the following on screen:

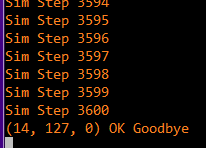


The cars will move as the agents respond with their traffic light signal commands to the Communications Agent.

When the simulation is complete the following screen generated by the SUMO-GUI will pop up:



The contents of the Communications Agent screen will look like this:



All agents except for the Communications Agent will have self-terminated at this point.