

MATLAB INTERFACE FOR RPL DODAG VISUALIZATION

Software Recommended: NetSim Standard v12.1/v12.2 (32bit/ 64bit), Visual Studio 2015/2017/2019, MATLAB (32bit/ 64bit)

Note: This project works only with MATLAB v2015b or higher versions.

Follow the instructions specified in the following link to clone/download the project folder from GitHub using Visual Studio:

<https://tetcos.freshdesk.com/support/solutions/articles/14000099351-how-to-clone-netsim-file-exchange-project-repositories-from-github->

Other tools such as GitHub Desktop, SVN Client, Sourcetree, Git from the command line, or any client you like to clone the Git repository.

Note: It is recommended not to download the project as an archive (compressed zip) to avoid incompatibility while importing workspaces into NetSim.

Secure URL for the GitHub repository:

v12.1: https://github.com/NetSim-TETCOS/RPL_DODAG_Visualization_in_IOT_v12.1.git

v12.2: https://github.com/NetSim-TETCOS/RPL_DODAG_Visualization_in_IOT_v12.2.git

Note: The cloned project directory will contain the documentation specific to the NetSim version (v12.1/v12.2).

RPL (Routing Protocol for Low-Power and Lossy Networks) is a routing protocol for wireless networks with low power consumption and generally susceptible to packet loss. It is a proactive protocol based on distance vectors and operates on IEEE 802.15.4, optimized for multi-hop and many-to-one communication, but also supports one-to-one messages.

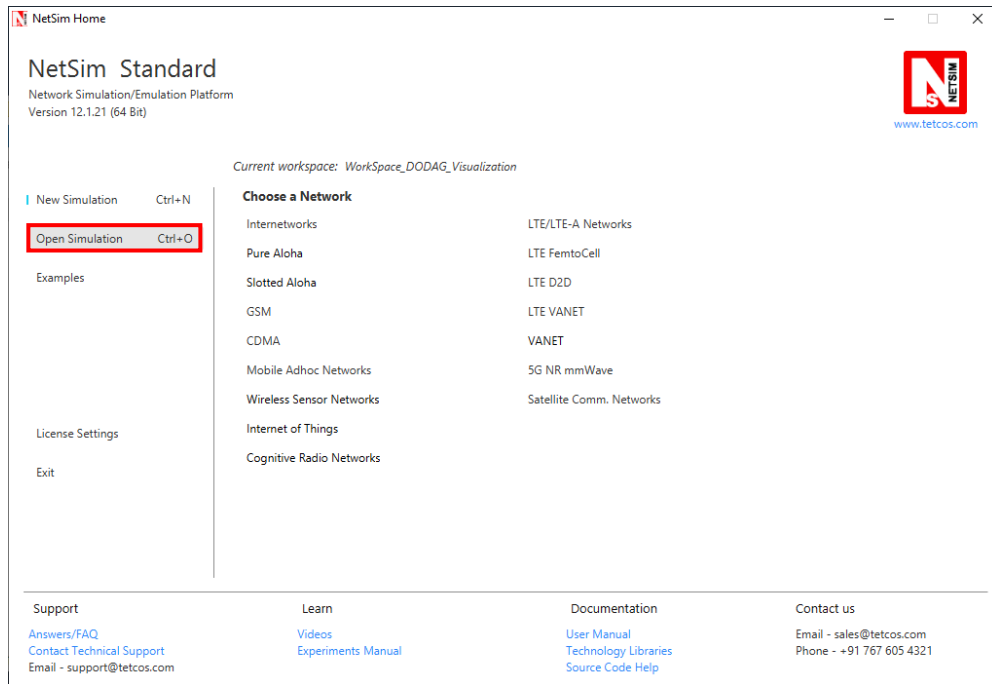
This protocol is specified in RFC 6550 with special applications in RFCs 5867, 5826, 5673 and 5548. RPL can support a wide variety of link layers, including those with limitations, with potential losses or that are used in devices with limited resources. This protocol can quickly create network routes, share routing knowledge and adapt the topology in an efficient way.

RPL creates a topology similar to a tree (DAG or directed acyclic graph). Each node within the network has an assigned rank (Rank), which increases as the teams move away from the root node (DODAG). The nodes resend packets using the lowest range as the route selection criteria.

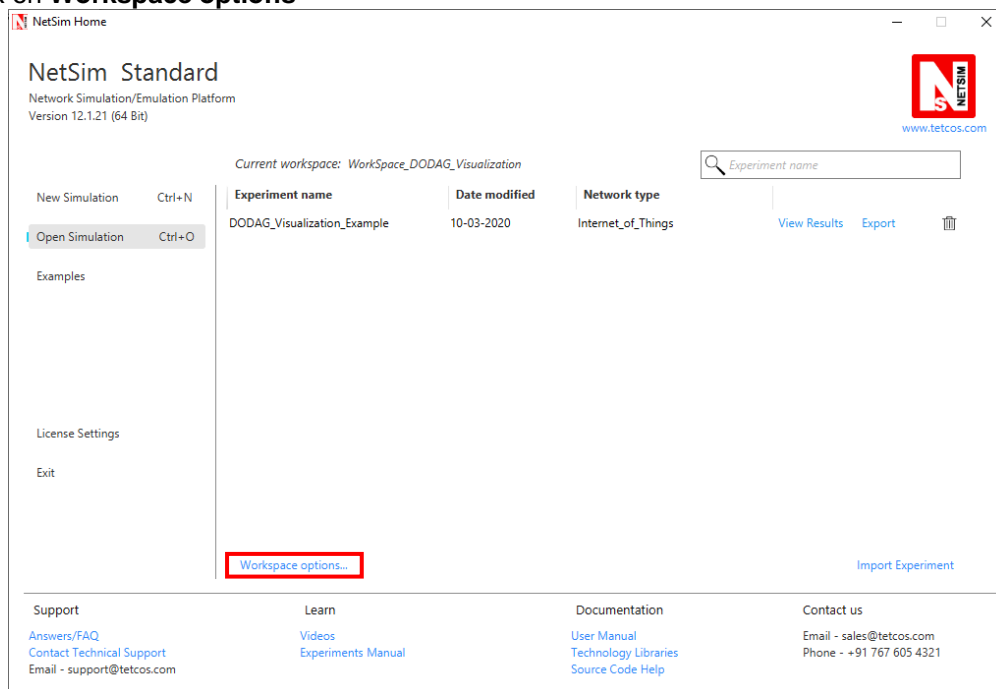
The DODAG formed as part of RPL protocol in NetSim can be visualized by interfacing NetSim with MATLAB.

Steps to run MATLAB interface

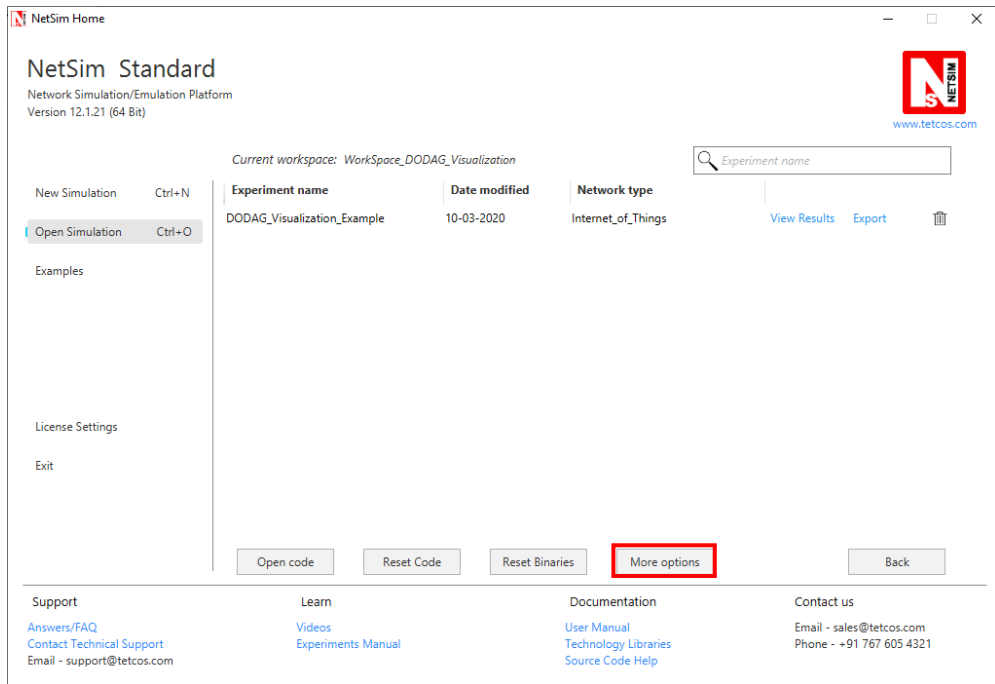
1. After you unzip the downloaded project folder, Open NetSim Home Page click on **Open Simulation** option,



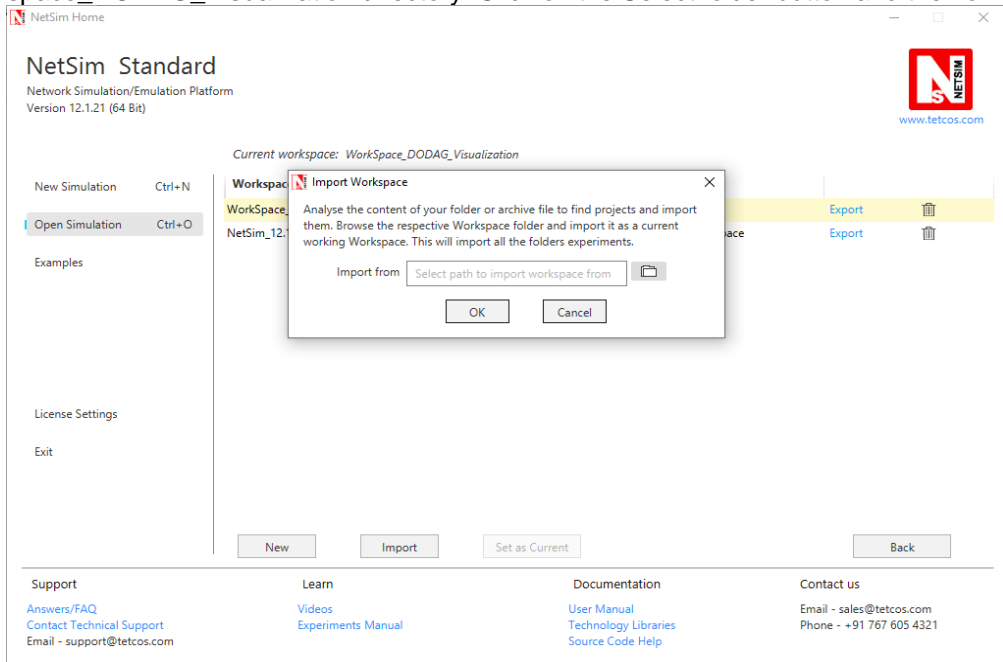
2. Click on **Workspace options**



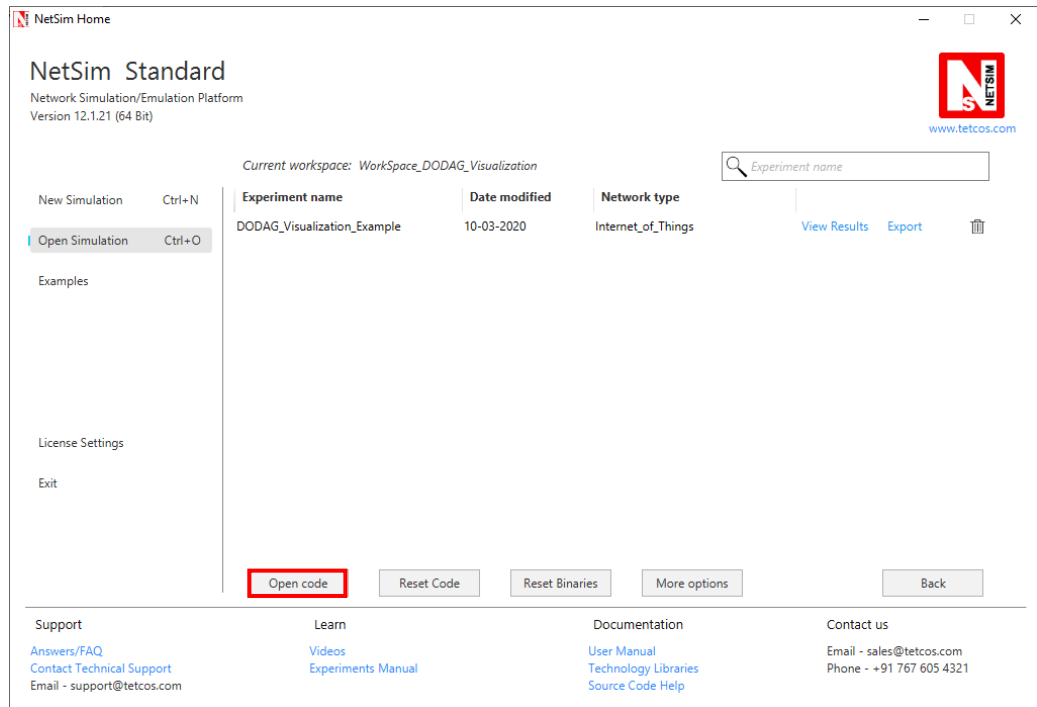
3. Click on **More Options**,



- Click on **Import**, browse the extracted folder path and go into the **Workspace_DODAG_Visualization** directory. Click on the **Select folder** button and then on **OK**.



- Go to home page, Click on **Open Simulation**  **Workspace options**  **Open code**



6. Place **PlotDAG.m** file inside the MATLAB root directory. For Eg: “<MATLAB installed path>\MATLAB\R2015b”, (Note: **PlotDAG.m** is provided inside the MATLAB_Code directory)
7. Following modifications were done to the RPL project for this implementation:
 - a. Open RPL.c file and add **fn_netsim_matlab_init()**, **fn_netsim_matlab_DODDAG_run()** and **fn_netsim_matlab_DODDAG_Init()** inside **fn_NetSim_RPL_Init()** and **fn_netsim_matlab_finish()** inside **fn_NetSim_RPL_Finish()**.

```

Neighbor.c  MATLAB_Interface.c  RPL.c  x
RPL (Global Scope)  fn_NetSim_RPL_Init(stru_NetSim_Networ
25  /**
26  RPL Init function initializes the RPL parameters.
27  */
28  _declspec(dllexport) int fn_NetSim_RPL_Init(struct stru_NetSim_Network *NETWORK_Forma
29  NetSim_EVENTDETAILS *pstruEventDetails_Forma
30  char *pszAppPath_Forma
31  char *pszWritePath_Forma
32  int nVersion_Type,
33  void **fnPointer)
34  {
35  fn_netsim_matlab_init();
36  fn_netsim_matlab_DODDAG_Init();
37  fn_netsim_matlab_DODDAG_run();
38  _getch();
39  return fn_NetSim_RPL_Init_F();
40  }
41
42  /** ... */
46  _declspec(dllexport) int fn_NetSim_RPL_Run({ ... })
104
105  /** ... */
109  _declspec(dllexport) int fn_NetSim_RPL_Finish()
110  {
111  _getch();
112  fn_netsim_matlab_finish();
113  return fn_NetSim_RPL_Finish_F();
114  }
115
99 %

```

- b. Add definitions of the following functions inside **RPL.h** file
 - a. **double** fn_netsim_matlab_init();
 - b. **double** fn_netsim_matlab_DODDAG_Init();
 - c. **double** fn_netsim_matlab_DODDAG_run();
- c. **double** fn_netsim_matlab_finish();

```

21 #define _NETSIM_RPL_H_
22 #ifdef __cplusplus
23 extern "C" {
24 #endif
25
26 //Log settings
27 #define DEBUG_RPL
28 #ifdef DEBUG_RPL
29 //define DEBUG_RPL_PRINT_IP_TABLE
30 #define DEBUG_RPL_PRINT_DAO_ROUTE_INFOMATION
31 //define DEBUG_RPL_TRICKLE
32 #endif
33
34
35 #include "RPL_Message.h"
36
37 //Include necessary lib's
38 #pragma comment(lib,"NetworkStack.lib")
39 #pragma comment(lib,"RPL.lib.lib")
40 double fn_netsim_matlab_init();
41 double fn_netsim_matlab_DODDAG_Init();
42 double fn_netsim_matlab_DODDAG_run();
43 double fn_netsim_matlab_finish();
44
45 /*
46 * Maximum amount of timer doubling.
47 */

```

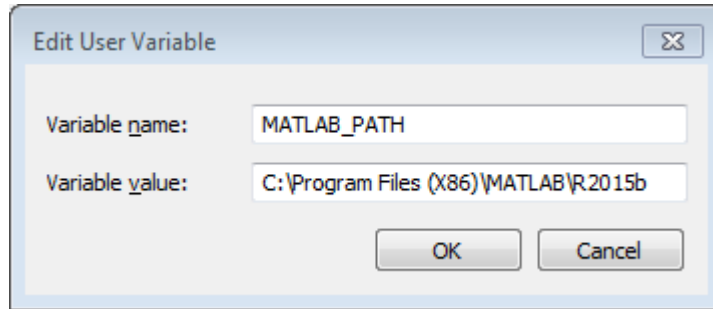
- d. Go to the **Neighbor.c** file. Inside Function **void** choose_parents_and_siblings(NETSIM_ID d) add fn_netsim_matlab_DODDAG_run() below rpl_add_route_to_parent()

```

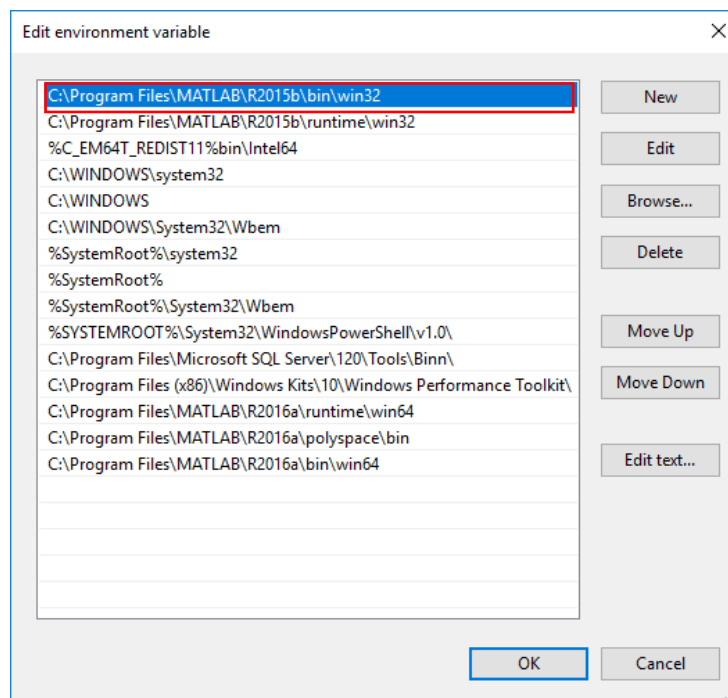
306 dodag->pref_parent = rpl->neighbor_list[best_rank_index];
307 if (dodag->pref_parent->nodeId != old_pref_parent)
308 {
309     NetSim_EVENTDETAILS pevent;
310     memset(&pevent, 0, sizeof pevent);
311     pevent.dEventTime = pstruEventDetails->dEventTime;
312     pevent.nDeviceId = d;
313     pevent.nDeviceType = DEVICE_TYPE(d);
314     pevent.nEventType = TIMER_EVENT;
315     pevent.nProtocolId = NW_PROTOCOL_RPL;
316     pevent.nSubEventType = RPL_NEW_PREF_PARENT;
317     fnpAddEvent(&pevent);
318 }
319 rpl_add_route_to_parent(d, dodag->pref_parent->nodeId);
320 fn_netsim_matlab_DODDAG_run();
321
322
323 for (i = 0; i < rpl->neighbor_count; i++)
324 {
325     PRPL_NEIGHBOR neighbor = rpl->neighbor_list[i];
326
327     if (matching_ranks[i] >= INFINITE_RANK)
328     {
329         if (neighbor->lastDIOMSG != NULL)
330         { /* forget messages from other DODAG iterations */
331             rpl_dio_pdu_free(neighbor->lastDIOMSG);
332             neighbor->lastDIOMSG = NULL;

```

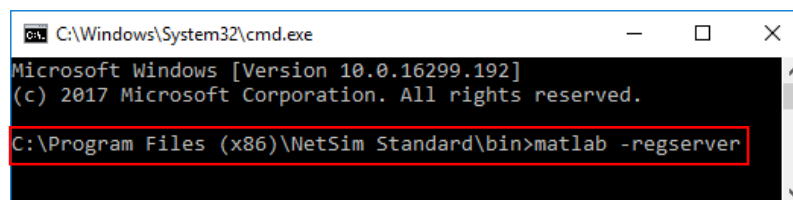
8. Create a user variable with the name of MATLAB_PATH and provide the path of the installation directory of user's respective MATLAB version.



9. Make sure that the following directory is in the PATH(Environment variable)
<Path where MATLAB is installed>\bin\win32



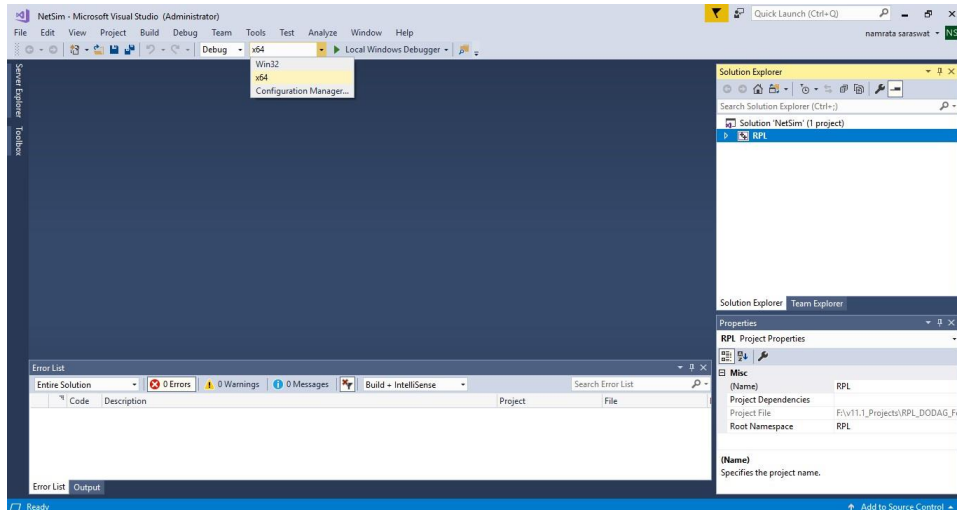
(Note: To run this code 32-bit version of MATLAB must be installed in your system. If you are interfacing for the first time then open command window and go to the <NetSim installed directory>\bin and type `matlab -regserver`)



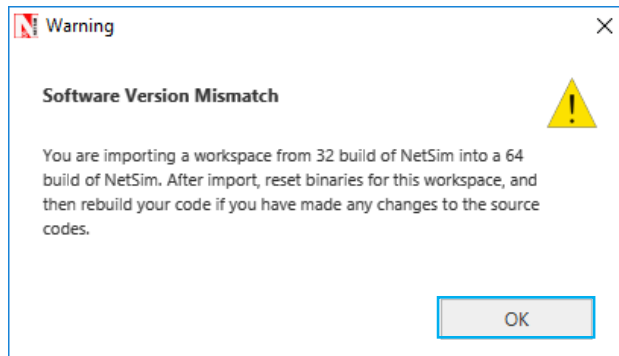
10. Now Right Click on RPL project and select Rebuild.
11. Upon rebuilding, **RPL.dll** will automatically be updated in the respective bin folder of the current workspace.

Note:

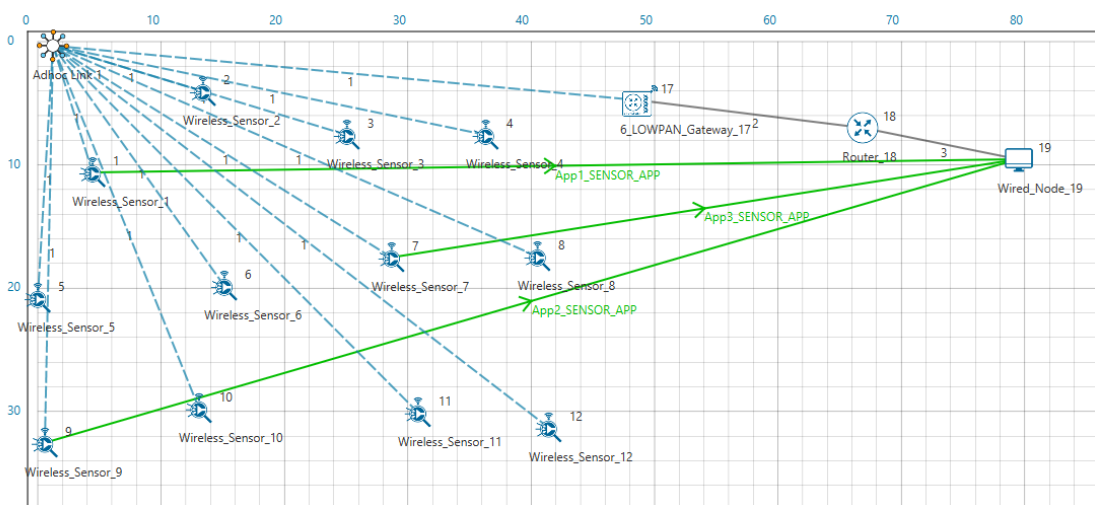
- Based on whether you are using NetSim 32 bit or 64 bit setup you can configure Visual studio to build 32 bit or 64 bit Dll files respectively as shown below:



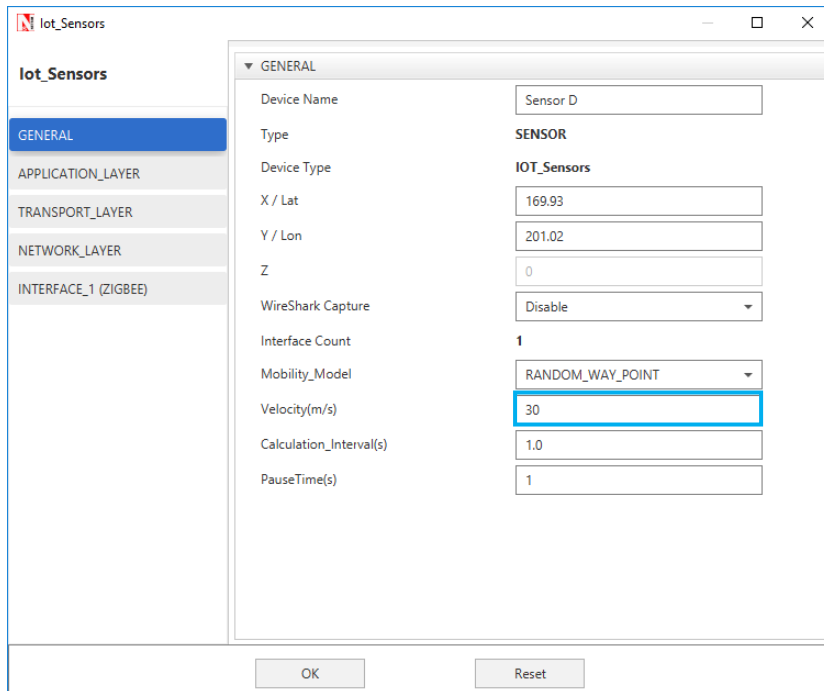
- While importing the workspace, if the following warning message indicating Software Version Mismatch is displayed, you can ignore it and proceed.



- Go to NetSim home page, click on **Open Simulation**, Click on **DODAG_Visualization_Example**.



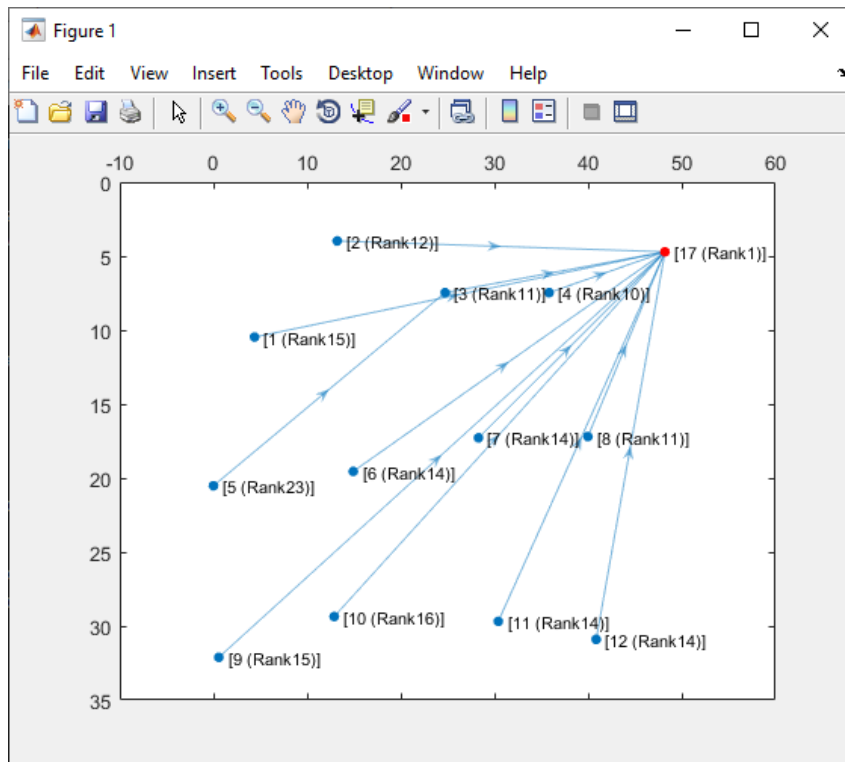
Set Velocity to the sensors



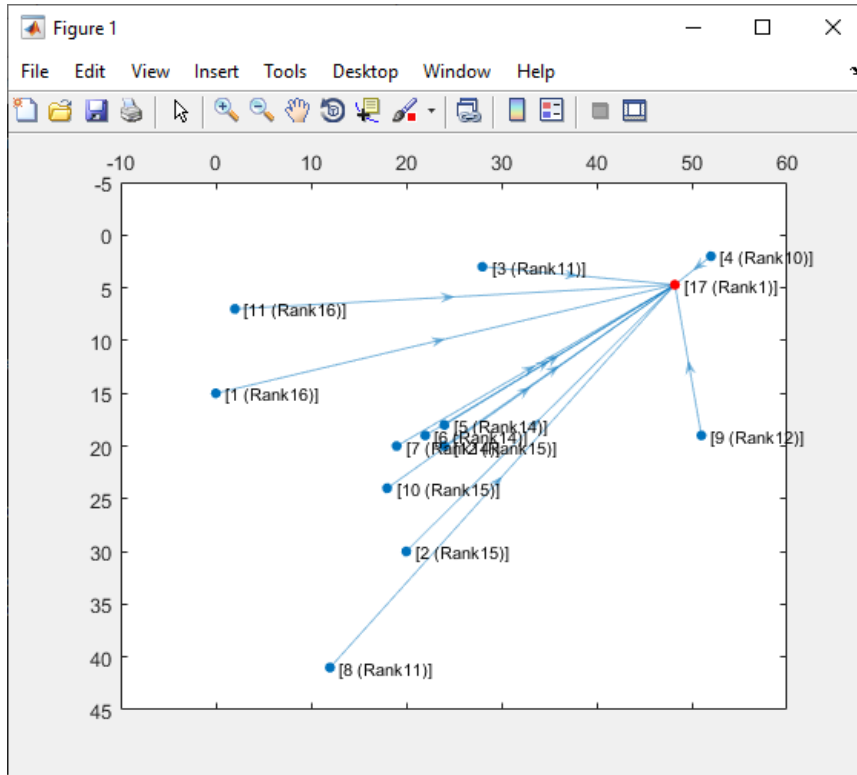
Output:

A plot will open, showing the DODAG when the simulation is started and the first route is formed between sink node and the sensor. And the DODAG will be dynamically updated.

Initially formed DODAG



DODAG formed after some time due to movement in sensors



After simulation press any key in the NetSim command window to close the MATLAB.