**Modelling and Sensing of the Sensing and Communication of a Temperature Sensor using NetSim WSN**

**Software Recommended:** NetSim Standard v10.2 (32bit), Visual Studio 2015, MATLAB 2011a (32bit)

**Objective:** To model physical phenomenon (Temperature) in MATLAB and to simulate the sensing and communication using NetSim WSN model

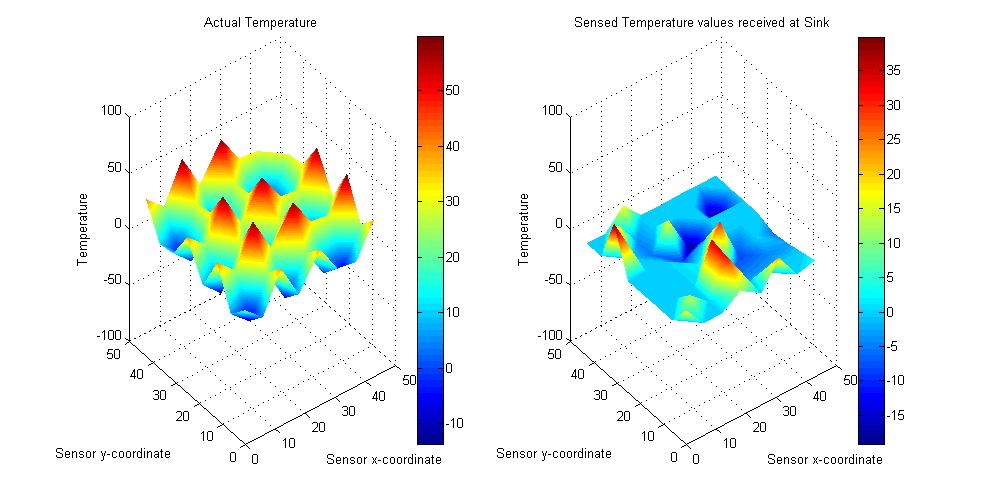
**Solution Overview:**

Let the variation of temperature over time defined by f (t).

Due to a variety of effects in the real world, this signal would appear to the sensor as from g (t) which we define as f (t) plus error (which we model as Additive White Gaussian Noise)

Given below is what g (t) and f (t) would look like. The graphs depict the model of actual temperature and the sensed temperature at a point of time. Note that g(t) and f(t) varies over time.

Simulation of network communication using NetSim: While temperature is modelled in MATLAB, NetSim simulates the 802.15.4 protocol in MAC and PHY layers and provides an easy way of interfacing with MATLAB. NetSim simulation determines the time taken for the packet to reach the sink, and also whether the packet is errored or not. Note that users can also modify the underlying source code of NetSim and write their own algorithm.



In the above plot, the first graph depicts the variation of the actual temperature in the simulation environment. The second graph represents the sensed temperature readings received at the sink node.

Not all the packets generated by the sensor nodes reach the sink node. Packets may get errored or collided during transmission. Delay is also involved between the time of generation of the packet and the time at which the packet is received at the sink node.

**Agent based sensing model in NetSim**

Sensors in NetSim are generic in terms of what they sense. NetSim concentrates on the communication aspects after sensing is performed. NetSim has a dedicated agent model.

The agent is an abstraction for whatever is being sensed by the sensor. So it is an abstraction for any physical phenomenon / property such as temperature, fire, pressure, mositure ...etc.

The sensors "sense" the agent, and upon sensing application packets are generated automatically by NetSim. This happens in fn\_NetSim\_Zigbee\_SensorEvent( ) in sensor.c. Users can modify the underlying source code to set their own packet size / arrival times.

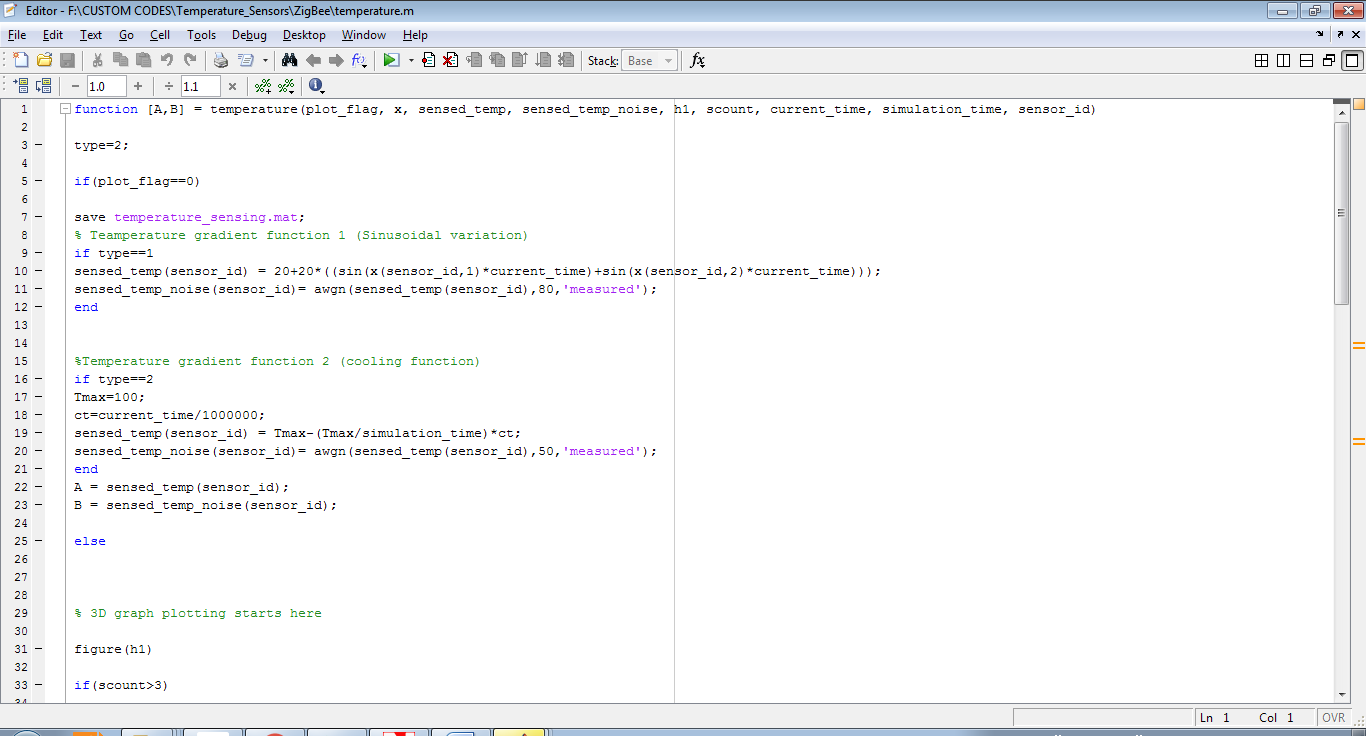
Each sensor has the parameters Sensor Interval and Sensor Range. The parameter sensor interval is used to determine how often a sensor has to perform sensing.

SENSOR\_EVENT is used to initiate sensing process for a sensor after every sensor interval.

During each sensor event of a sensor, it is determined whether the agent is within the sensor range. If an agent is present within the sensor range then the sensor generates a packet, assuming that it has sensed an agent (temperature in this case).

**Temperature modelling in MATLAB:**

1. We have written a temperature.m file which can be used for modelling temperature either with
   1. sinusoidal variation over time f(t) = T + T (sin(xt) + sin (yt)) and T = 20
   2. cooling function where the temperature gradually drops over time f(t) = T – (T / tmax)\* t
2. Following is a screenshot of the temperature.m file:



1. The temperature function in the temperature.m file requires the following inputs from NetSim during runtime:
   1. X and Y coordinates of all the sensors
   2. Actual temperature readings which were modelled and passed to NetSim for all the sensors.
   3. Temperature readings received at the Sinknode
   4. Total number of sensors in the current scenario.
   5. Current simulation time.
   6. Total Simulation time.
   7. The ID of the sensor that is currently performing sensing.

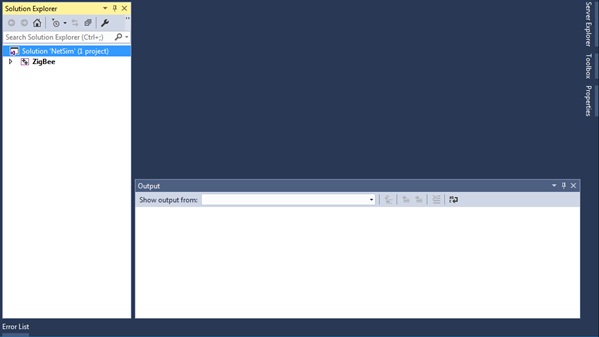
**How temperature sensors are modelled in NetSim?**

The agent model in NetSim is replaced with the calls to the temperature model in MATLAB. Whenever a SENSOR\_EVENT for a sensor is triggered, a call to MATLAB function temperature() in the temperature.m file is made. All the inputs required by the temperature function are passed from NetSim during the call. MATLAB returns to NetSim the actual temperature in the environment and the temperature that is sensed by the sensor (the variation between the two is achieved using Additive White Gaussian Noise).

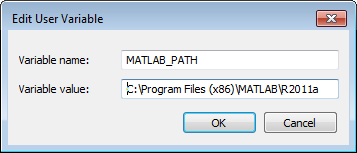
During the process graphs are also plotted, representing the change in temperature in the environment with respect to time and the difference we find between the actual and the sensed temperature.

# Steps:

1. After you unzip the file the folder would look like Open the simulation folder
2. Open the NetSim.sln file inside the Simulation folder.
3. Click on OK and when this opens in MS Visual Studio 2015, it would look like



1. Under the ZigBee project in the solution explorer double click on the MATLAB\_Interface.c file.
2. Place temperature.m file inside the installed directory of MATLAB i.e. **C:\Program Files (x86)\MATLAB\R2011a**.
3. This file is present in the ZigBee folder inside the Simulation\_Temperature\_Sensors directory.
4. Create a user variable with the name of MATLAB\_PATH and provide the path of the installation directory of user’s respective MATLAB version.

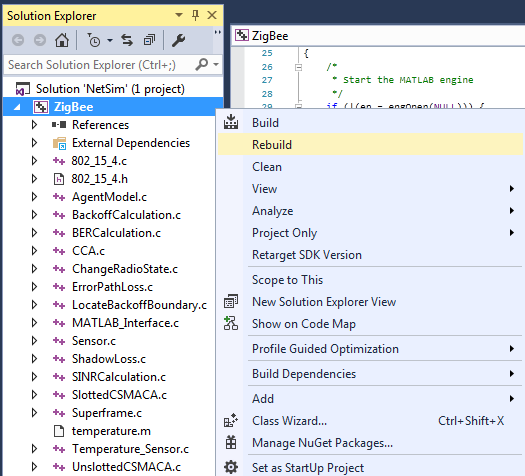


1. Make sure that the following directory is in the PATH(Environment variable)

**<Path where MATLAB is installed>\bin\win32**

**Note:** If the machine has more than one MATLAB installed, the directory for the target platform must be ahead of any other MATLAB directory (for instance, when compiling a 32-bit application, the directory in the MATLAB 32-bit installation must be the first one on the PATH).

1. Right click on ZigBee project in Solution Explorer and select rebuild.



1. Upon rebuilding, libZigBee.dll will get created in the path ..\simulation\DLL
2. Now copy the libZigBee.dll from this DLL folder and paste it in NetSim bin folder present in the NetSim installation directory. The NetSim install director would look something like

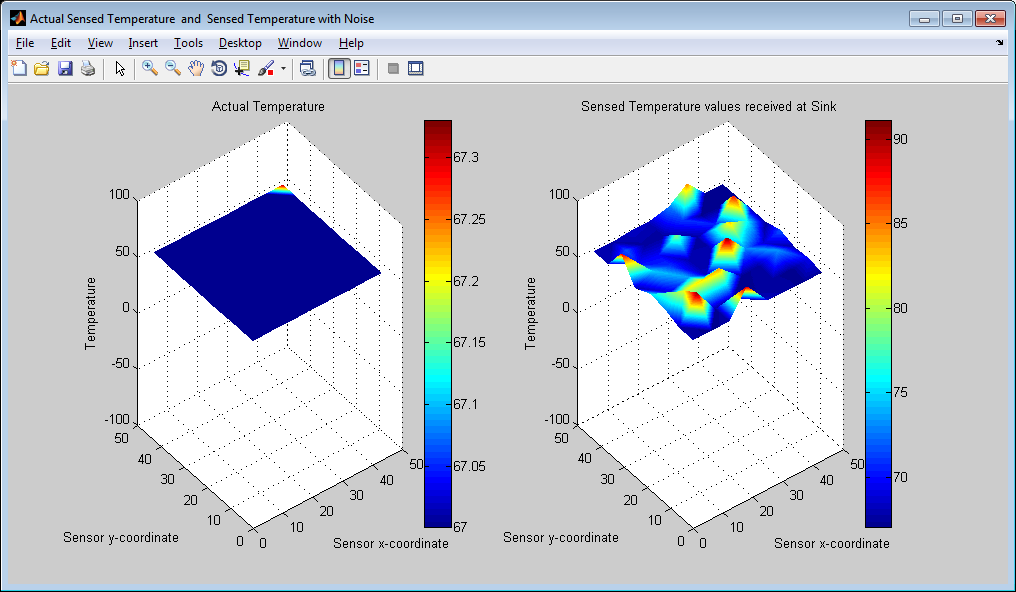
< **C:\Program Files (x86)\NetSim Standard\bin**>

1. Note that there exists a libZigbee.dll in this bin folder. This is the default file being shipped with NetSim. The user is replacing this file with the newly built file.
2. Therefore, take care to rename the original libZigBee.dll file, so that it isn’t lost. For example, you may rename it as libZigbee\_default.dll
3. Run NetSim in administrator mode and open the configuration.xml file provided with the project or you can prefer to create a Network Scenario on your own.
4. Run the simulation.

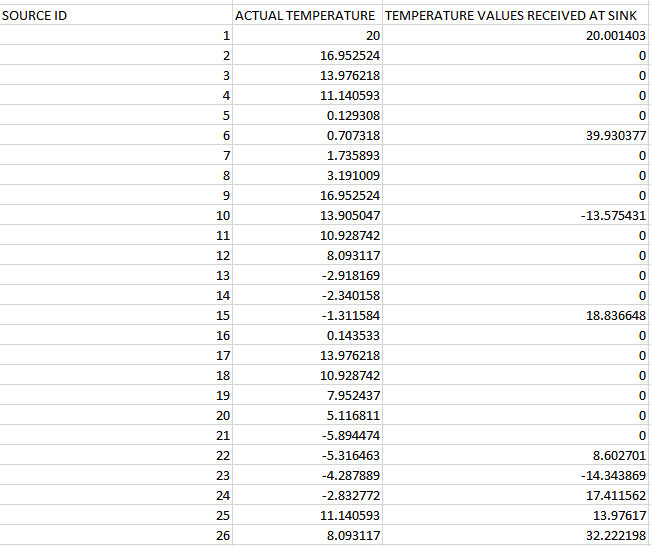
**Simulation output metrics:**

On performing the simulation of this Temperature Sensor Project in NetSim you will get the following metrics in addition to the set of performance metrics that NetSim provides, for any WSN Network Simulation performed.

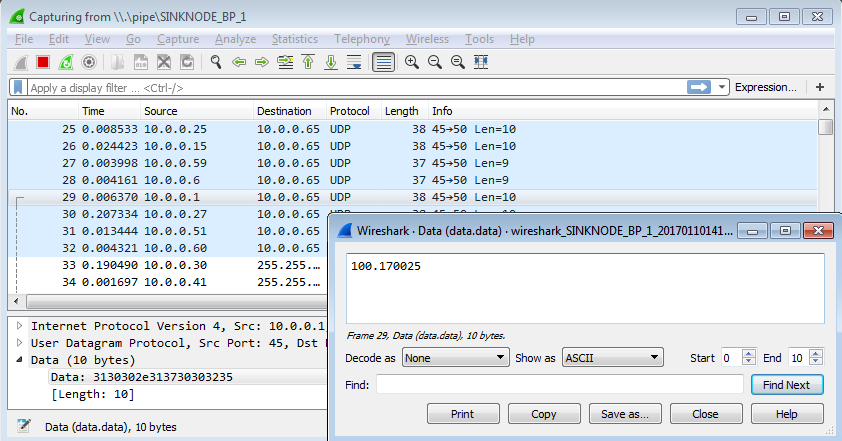
1. A 3D plot showing two graphs, one depicting the variation of the actual temperature in the environment with respect to time and other depicting the temperature readings received at the sinknode at different points of time.



1. A Temperature\_Sensor\_Log.csv file which logs in the timestamps, the actual and sensed temperature at each sensor



1. Wireshark packet capture:



Note: To obtain Wireshark packet capture. Wireshark Online parameter has to be set in Configuration file manually and run in CLI mode. You can use the Configuration file provided along with this project for this purpose.

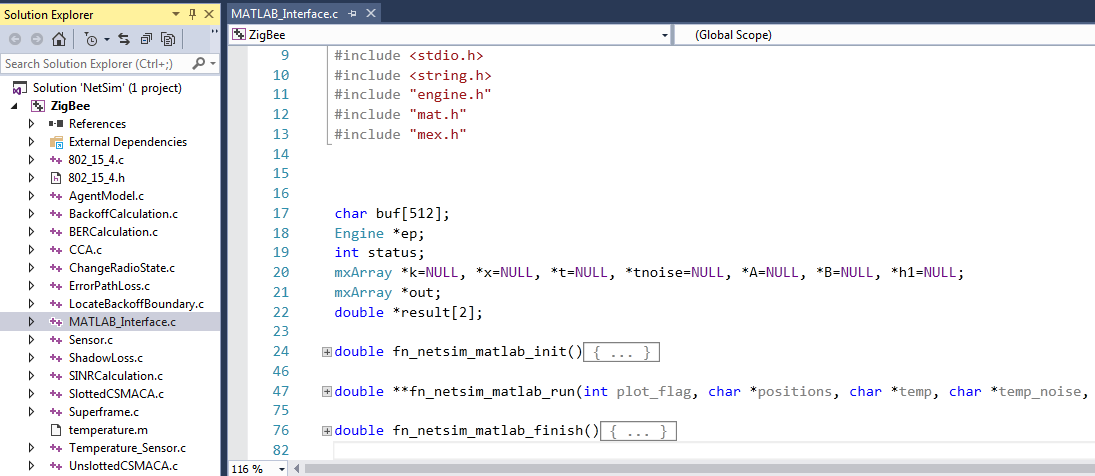
**Code modifications done:**

Please note that as part of the ZigBee project a MATLAB\_Interface.c file is added, which contains the following functions

**a)**  fn\_netsim\_matlab\_init() - Opens the MATLAB Engine

**b)** fn\_netsim\_matlab\_run() - Communicates with MATLAB Command Window

**c)** fn\_netsim\_matlab\_finish() - Closes the MATLAB Engine



We have also added a Temperature\_Sensor.c file to the ZigBee project to provide inputs from NetSim to MATLAB and also to get and update the variables with the values obtained from MATLAB. This file also takes care of writing the Temperature\_Sensor\_Log.csv file.

Then code modifications are done in function, fn\_NetSim\_Zigbee\_SensorEvent() present in Sensor.c file, where the codes related to sensing an agent is commented off and codes for making calls to MATLAB is added.

Here we also write the sensed temperature to the packet that is generated, so that it can be visualized using Wireshark packet capture.