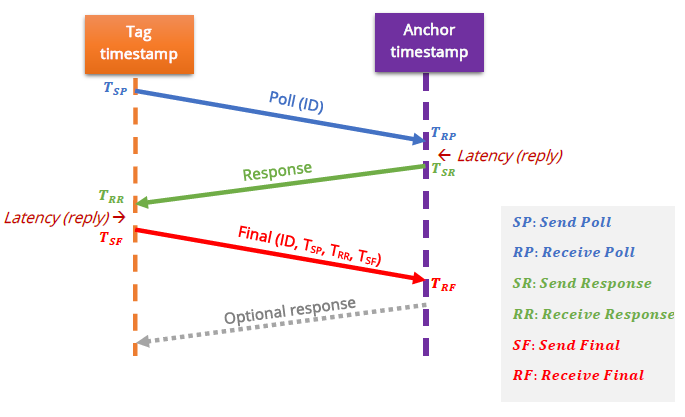
**UWB device:**

The DecaWave UWB device provides a solution to real time localization system (RTLS) network. Many solutions had been done but not many would provide the localization across different AGVs platform. UWB technology has been in used since the 1960s and it provides a lower power spectral density of -40 dBm/MHz so as to avoid interference with other frequency like Wi-Fi or Bluetooth. The accuracy of the UWB technology is also high, error margin is within centimeters. The DecaWave UWB system consist of tags which are located on the AGVs, and anchor nodes which are placed around the needed workspace. In order to setup a network, there are two possibilities: creating the network through DRTLS app from DecaWave or through the serial communication. Both are described in the DW1001-DEV board documentations [LINK].

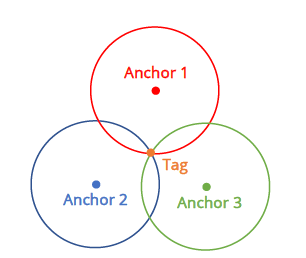
The system uses the Two-Way Ranging method to communicate. This method works similarly to a ping system as seen in figure below. Each message the tag send is included with a time stamp. The tag transmits a poll with its identifier to the anchor. Once a response from the anchor is received, the tag will send a final response with all related time stamps. By determining the time it takes for a signal to finish around trip, the distance between them can be calculated as follows:

𝑇𝑜𝐹 = ((𝑇𝑅𝑅− 𝑇𝑆𝑃) − ( 𝑇𝑆𝑅−𝑇𝑅𝑃) + ( 𝑇𝑅𝐹−𝑇𝑆𝑅) − ( 𝑇𝑆𝐹−𝑇𝑅𝑅))/ 4



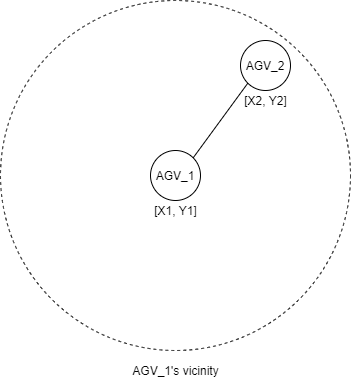
To determine the location of a UWB tag, the trilateration method was used. This method can be understood with the figure below. This method then find the intersecting point of 3 circles with the distance of each anchor to the tag as the radius of the respected circle. Because of this, the tag needs to send out polls to a minimum of 3 anchor nodes. By getting the distance between the tag and the anchor nodes, the UWB device would process the information into localization data using this formula:

Where and are the coordinates of anchor number i, x and y are the coordinates of the tag.

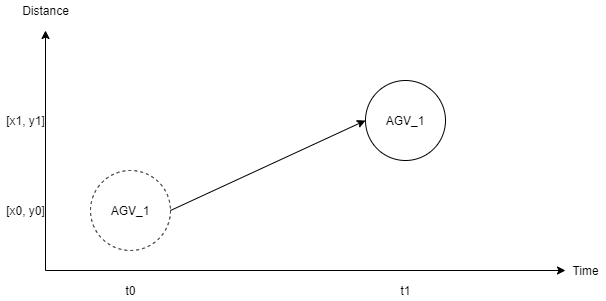


**UWB possibilities:** what functionality of the UWB can be useful

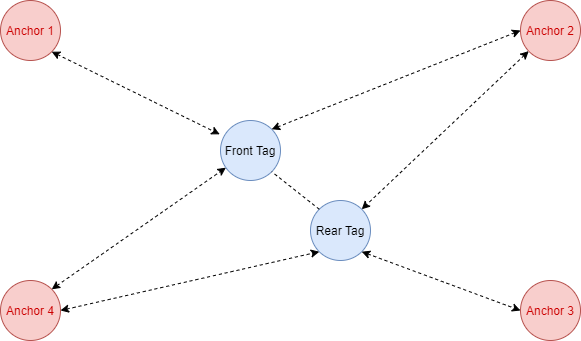
With the location data from the DecaWave UWB, there are many functionalities that can be infer. First and foremost, with these data, the system can have an overview of where different AGVs are and what is in their respective vicinity. This is achieved by finding the difference in coordinate of the tags. A visualization of this functionality is seen below.



Although some AGVs do have the capability to see their respective speed, not all AGV have this feature. By observing how long it takes to reach from one data point to another, there is the possibility of measuring the speed of the AGV.



To find the orientation of the AGV, two tags will be used with each place at either the front or the rear of the AGV. Then by using the arc-tangent rule, the system can formulate the angle of the AGV comparing to the grid of anchors.



**UART serial codes:**

There are many options to access the UWB device for location data. The two communication protocols that was explored are UART serial and SPI. The advantage and disadvantages of both protocols will be discussed below.

SPI is the first type of communication protocol that was researched. The data flow of SPI is continuous as there is no start or stop bit. This could be seen as both good and bad impact. SPI boast a faster speed for communication messages thanks to the continuous flow. But since there is no stop bit, no acknowledgement could be confirmed which mean data could be lost without the knowledge of the system. As there are separate MISO and MOSI line, data can be transfer and receive at the same time. However, SPI requires a dedicated UWB device, besides the two tags, acting as a listener to be able to receive location information. This listener could lead to interference. Although this have not been tested but if two tags are placed within 15 centimetres of each other, the error margin is more significant. Also, since there is no parity bit, SPI cannot error check messages its receiving.

UART serial is the second choice and the communication protocol that was used in this project. UART is simpler to operate than SPI as it uses USB connection to communicate. One drawback of UART is the message size, a maximum of only 9 bits of data. Because the data require a stop bit from 9 bits of data, transfer rate is limited. But with the stop bit, the system can acknowledge if the data arrives correctly. Also, by using parity bit, the system can error check messages before receiving.

For the connection between the CoLAB’s raspberry pi and the UWB chip, UART serial was the chosen protocol. No extra listener is required. Since the location data transferred is small, UART 9-bits data size is more than sufficient to send. As the two tags requires power through USB, combining power and communication is also a reason to choose UART.

**Explanation of UART serial code**

The code below is within the config of the main sendLocation function and is used to set up serial config for both tags such as the correct baud rate that the UWB chip uses. The code also set up the size of the message with eight bits of data and one stop bit. Next, in order to access the UWB’s UART API through serial, a double-tapping of the enter key is needed to. With this, all necessary configuration for serial read to receive information is done.

    #Front tag serial config

    serFront = serial.Serial()

    serFront.port = '/dev/UWBfront'

    serFront.baudrate = 115200

    serFront.bytesize = serial.EIGHTBITS

    serFront.parity =serial.PARITY\_NONE

    serFront.stopbits = serial.STOPBITS\_ONE

    serFront.timeout = 1

    serFront.open()

    serFront.write(b'\r\r')

    time.sleep(1)

    serFront.close()

Within the main sendLocation function, for serial reading, the code will utilize the command “apg” from the DecaWave UWB. This command, when used, returns the location value based on the trilateration method used by DecaWave UWB. The message format is: “apg: x:### y:### z:### qf:###”. Once the message is returned, the code will read the data from the serial ports and pre-process it by removing unnecessary characters. Then the data will be saved globally within the function as a float array. The information within the array will be append to variables for further processing. The code below is an example for the front tag only.

# use “*apg”* command and read location data from UWB Front tag

      serFront.write(b'apg\n')

      serFront.readline()

      dataFront = str(serFront.readline())

# pre-process location data to create array

      dataFront = dataFront.split(' ')

      dataFront[1] = float(dataFront[1].replace("x:",""))

      dataFront[2] = float(dataFront[2].replace("y:",""))

      dataFront[3] = float(dataFront[3].replace("z:",""))

      dataFront[4] = dataFront[4].replace("qf:","")

      dataFront[4] = float(dataFront[4].replace("\\r\\n'",""))

      dataFront.pop(0)

# save data from location array for further processing

      FTagX.append(dataFront[0])

      FTagY.append(dataFront[1])

**How to setup UART serial for the Raspberry Pi:**

Setup the network with the app (**Reminder**: change stationary refresh rate to 10 Hz) [LINK]

Use RaspberryPi terminal to run these commands:

Command: dmesg | grep ttyUSB

to get serial number for individual UWB tags

first connect first tag to be the front tag.

>> Check first serial number

Then connect second tag to be the rear tag

>> Check second serial number

Command: sudo nano /etc/udev/rules.d/10-usb-serial.rules

to setup rules for later uses of the UWB

>> By using the command, a rules.d file will be created

>> Type these lines below and change the serial attribute with the serial number from above

*SUBSYSTEM==”tty”, ATTRS{serial}==”############”, SYMLINK+=”UWBfront”*

*SUBSYSTEM==”tty”, ATTRS{serial}==”############”, SYMLINK+=”UWBrear”*