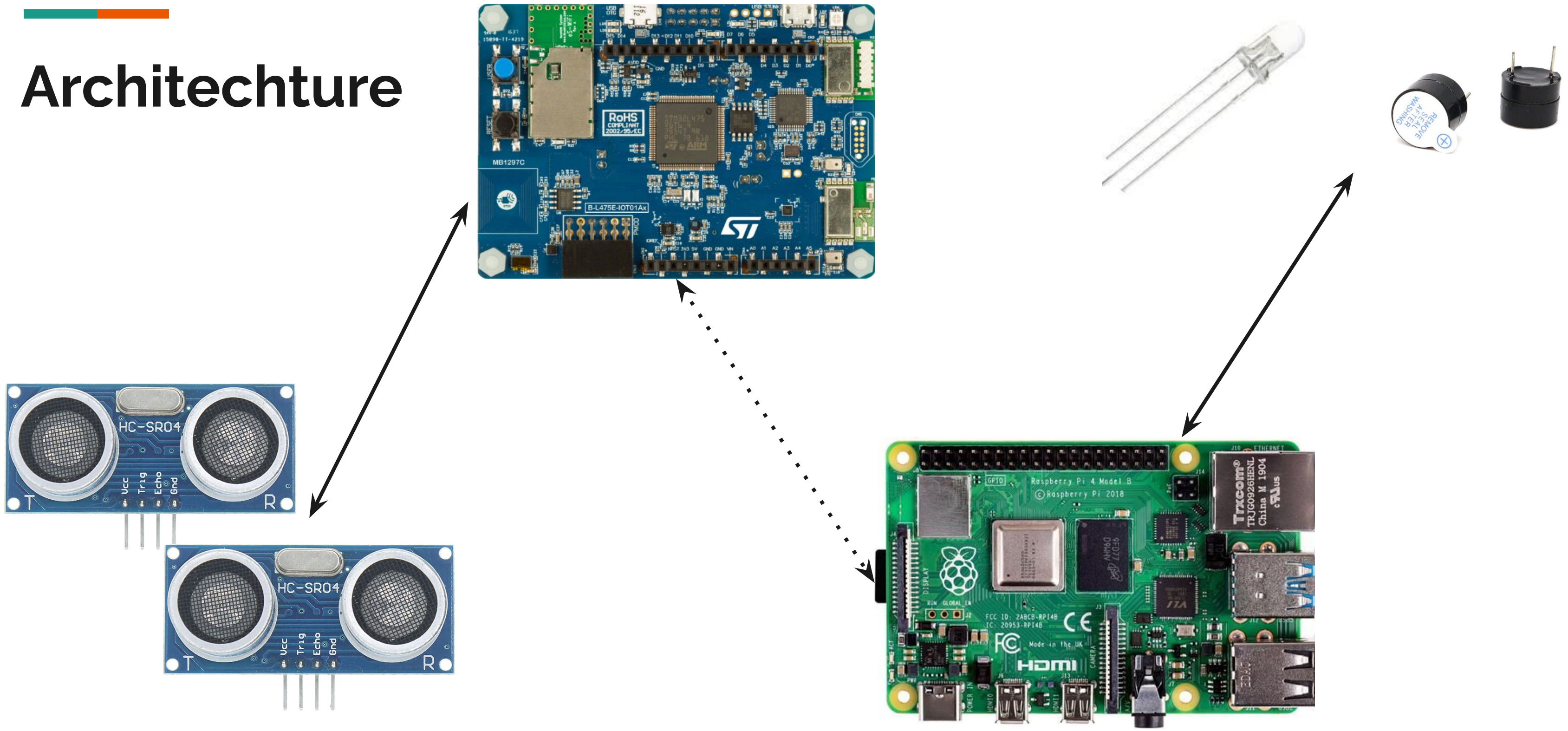
ESLab

陳亮君 曾憲揚 徐楷程

Motivation

- Smart Home
- IoT

Architechture



Architechture

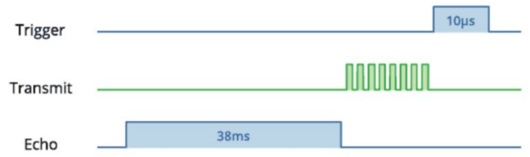
- Technologies that have been taught classes
    - Mbed OS API
        - Interrupt-driven IO (ISR)
    - Sensor Data Collection
    - Wireless communication (WiFi)

Architechture

- Sensor part
- Algorithm part
- Wifi part

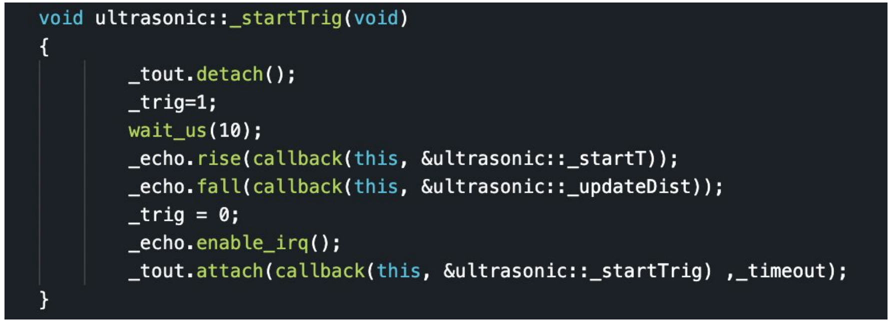
How HC-SR04 works

- Send 10us (or more) high voltage to trigger pin
- HC-SR04 sends eight pulses at 40kHz
- Echo rises after the last pulse
- Echo falls when receive wave



Sensor implementation

- _trig is DigitalOut
  - Assign 0/1 and wait to create the starting signal
- _echo is InterruptIn
  - Register functions for rising/falling edges
- _tout is Timeout
  - To re-call this function regularly



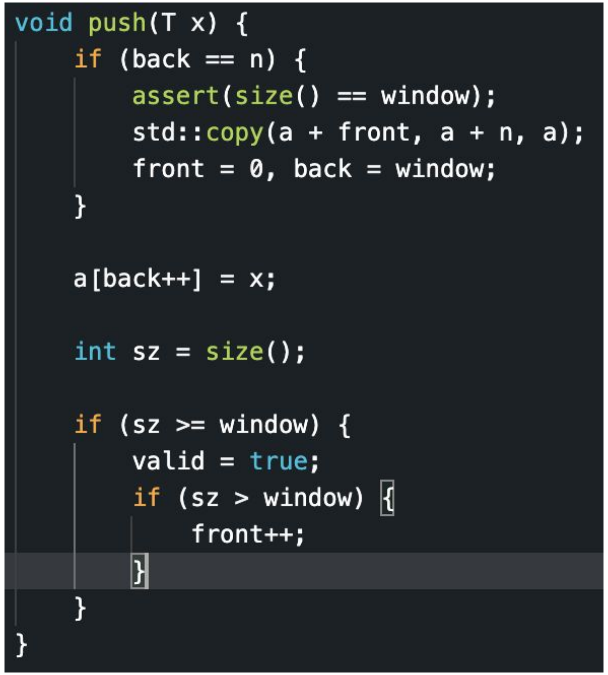
```
void ultrasonic::_startTrig(void)
{
        _tout.detach();
        _trig=1;
        wait_us(10);
        _echo.rise(callback(this, &ultrasonic::_startT));
        _echo.fall(callback(this, &ultrasonic::_updateDist));
        _trig = 0;
        _echo.enable_irq();
        _tout.attach(callback(this, &ultrasonic::_startTrig) ,_timeout);
}
```

Data Storage

- We will need data within a certain interval to solve for gestures
- struct MyQueue
    - Basically a queue with a limited size
    - To avoid dynamic memory usage(std::vector, std::queue), it is implemented as a sliding window on a large array, can only access data within the current size
- Each Sensor object is associated with a queue

MyQueue Implementation

- Not circular, mod (%) operation is expensive!



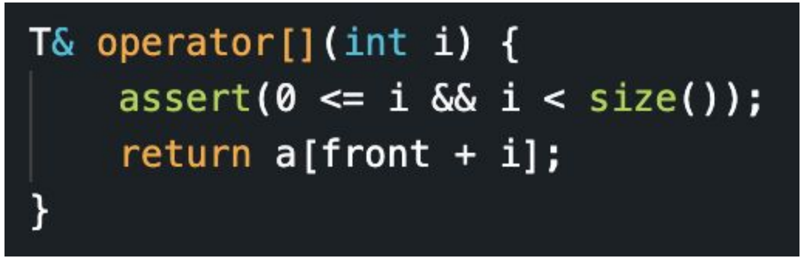
```
void push(T x) {
    if (back == n) {
        assert(size() == window);
        std::copy(a + front, a + n, a);
        front = 0, back = window;
    }

    a[back++] = x;

    int sz = size();

    if (sz >= window) {
        valid = true;
        if (sz > window) {
            front++;
        }
    }
}
```

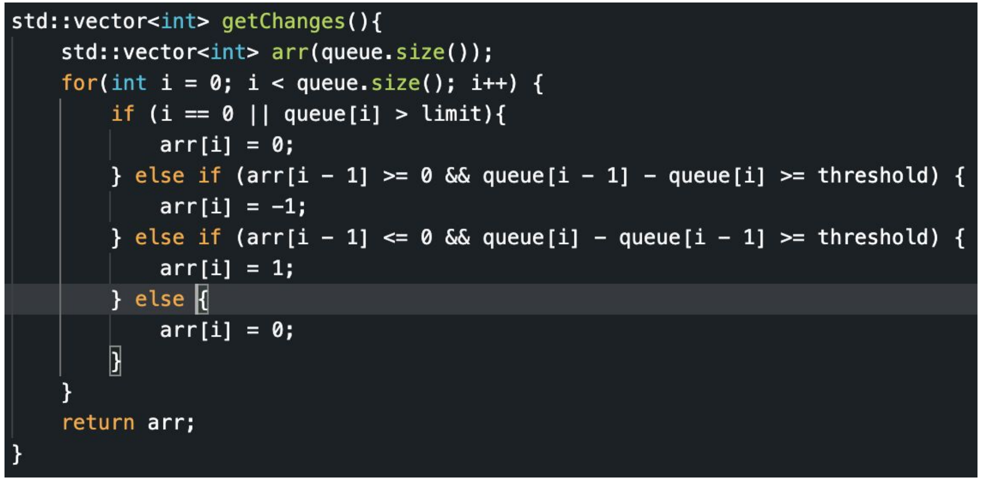
MyQueue Implementation



```
T& operator[](int i) {
    assert(0 <= i && i < size());
    return a[front + i];
}
```

Gesture Algorithm - Information

- For each sensor, we only care about whether an object appeared or left in front of it.
- Only **delta** of data in queue is important for current usage.

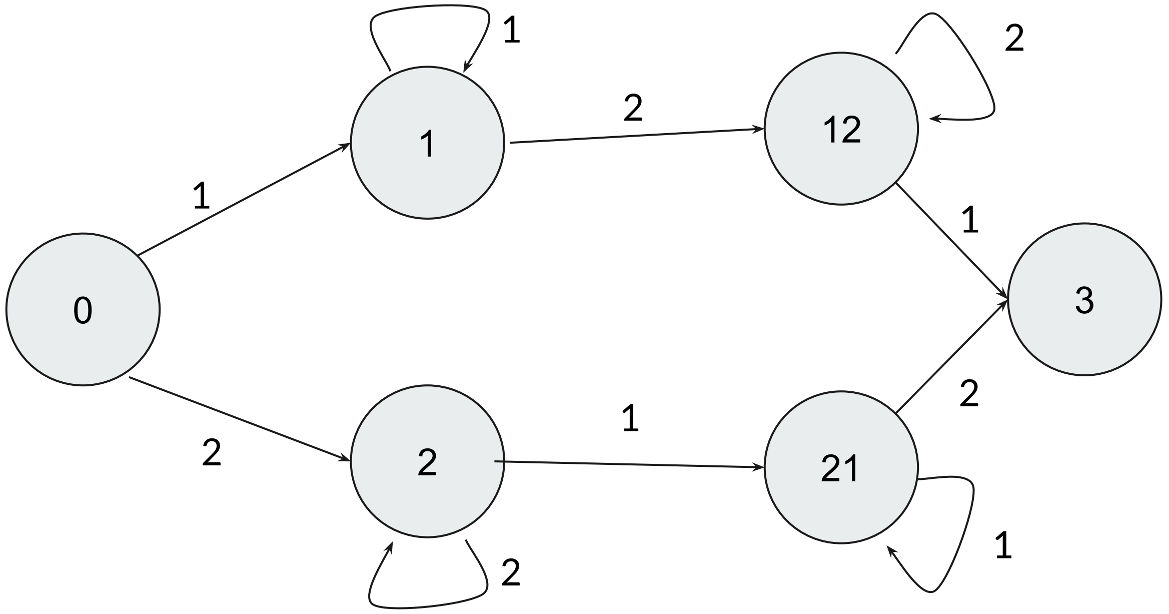


```cpp
std::vector<int> getChanges(){
    std::vector<int> arr(queue.size());
    for(int i = 0; i < queue.size(); i++) {
        if (i == 0 || queue[i] > limit){
            arr[i] = 0;
        } else if (arr[i - 1] >= 0 && queue[i - 1] - queue[i] >= threshold) {
            arr[i] = -1;
        } else if (arr[i - 1] <= 0 && queue[i] - queue[i - 1] >= threshold) {
            arr[i] = 1;
        } else {
            arr[i] = 0;
        }
    }
    return arr;
}
```

Gesture Algorithm- information

- Gesture can be defined as “an order of sensors that detected something”
- Assuming that the sensors are separated enough, that only sensor will response at any instant
- Just need to find the order of peaks in the two sensors’ queue

Gesture Algorithm- state diagram



if idle: go to state 0

Gesture Algorithm - Special Cases

- Multi-step gestures takes time
    - Idle time limit
- If state 3 is detected
    - No need to wait for anything more, can reset the state to 0
    - However, if state 3 did not take a full time to be performed, left over peaks will still exist in the queue
    - Need to reset the queue! -> adds to cycle time
    - This means that the queue's size is changing, not always full
    - Many functions need to return vectors, resulting in dynamic memory

Sensor Fine-tuning

- Detect frequency threshold
- Detect distance threshold

Frequency threshold - Detect frequency

- Detect function timeout time
    - The period that the detect function is executed
    - If this period is too long?
        - Might fail to detect the hand during this interval
    - What if this period is too short?

Frequency Threshold - Intrinsic cycle time

- Each detection cycle, the program needs to:
  - For each of the two sensors:
    - 10us trigger high
    - 8 * 40kHz wave
    - wait until the sound wave bounces back
    - push into queue, calculate something for the previous “window” time interval
  - Solve for gestures
    - possibly clearing the queue
- By experiment, we think the intrinsic cycle time is at least 20~50ms (unstable)
- If too short, the program is basically undefined behaviour

Frequency threshold - Reset and Idle time

- After a maximal gesture (not extendable) is detected, all sensors reset after **Reset time**
- If a non-maximal gesture is detected and nothing new happens in **Idle time**, break the gesture

Frequency threshold - Numerical results

- HC-SR04 operation time ~150 µs to 25 ms
  - timeout = 38ms
- detect function timeout time: 0.1s
- reset time: 0.2s
- idle time: 1s

Distance threshold

- There exists a upper limit on the value of collected data
  - Seems to be meaningless above the upper limit
  - Seems to be way less than the manual suggested
- A threshold on the value of **delta** is imposed, to eliminate errorneous detections caused by random fluctuations

Wifi part

- Using Wifi to connect RPi and STM32
- Open a socket and when receiving gesture STM32 would update it through Wifi and RPi could notice it

Controlling LED/Speaker With RPi

- After receiving the gesture, RPi would control LED/Speaker to act using a 2-bit data
- The first bit controls LED and the second controls speaker, could add more bits to control more different devices

Demo

- https://youtu.be/BIcjqGm-IUg

Reference

- How HC-SR04 Ultrasonic Sensor Works & Interface It With Arduino
- HC_SR04_Ultrasonic_Library
- HC-SR04 manual
- bluetooth v.s. wifi