**Thuật toán phát hiện va chạm trong hệ thống Bike Safety**

**1. Nguyên lý hoạt động**

Thuật toán phát hiện va chạm trong hệ thống Bike Safety dựa trên việc phân tích dữ liệu từ cảm biến gia tốc MPU6050 được gắn trên thiết bị ESP32C3. Thuật toán này không chỉ đơn thuần phát hiện khi giá trị gia tốc vượt ngưỡng, mà còn áp dụng nhiều kỹ thuật xử lý tín hiệu tiên tiến để tăng độ chính xác và giảm cảnh báo sai.

**2. Cấu hình cảm biến MPU6050**

Trước tiên, cảm biến MPU6050 được cấu hình với các thông số tối ưu:

// Set accelerometer range to ±16g

Wire.beginTransmission(MPU6050\_ADDR);

Wire.write(ACCEL\_CONFIG);

Wire.write(0x18); // 0x18 = 0b00011000 (bits [4:3] = 0b11 cho ±16g)

Wire.endTransmission(true);

* **Thang đo ±16g**: Cho phép đo lường được các va chạm mạnh mà không bị bão hòa cảm biến
* **Tốc độ lấy mẫu 200Hz**: Đủ nhanh để bắt được các va chạm xảy ra trong thời gian ngắn
* **Bộ lọc thông thấp (DLPF) 44Hz**: Loại bỏ nhiễu tần số cao nhưng vẫn giữ được thông tin va chạm

**3. Quá trình xử lý dữ liệu**

**3.1. Tính toán G-force**

Đầu tiên, hệ thống tính toán lực G từ dữ liệu gia tốc thô:

float calculateRawGForce() {

// Đọc dữ liệu gia tốc từ MPU6050

Wire.beginTransmission(MPU6050\_ADDR);

Wire.write(ACCEL\_XOUT\_H);

Wire.endTransmission(false);

Wire.requestFrom((uint8\_t)MPU6050\_ADDR, (size\_t)6, true);

int16\_t ax = Wire.read() << 8 | Wire.read();

int16\_t ay = Wire.read() << 8 | Wire.read();

int16\_t az = Wire.read() << 8 | Wire.read();

// Hệ số chia 2048.0 cho thang đo ±16g

float gx = ax / 2048.0f;

float gy = ay / 2048.0f;

float gz = az / 2048.0f;

// Magnitude của vector gia tốc

float gForce = sqrt(gx\*gx + gy\*gy + gz\*gz);

return gForce;

}

G-force được tính bằng cách lấy độ lớn của vector gia tốc trong không gian 3 chiều, cho phép phát hiện va chạm từ bất kỳ hướng nào.

**3.2. Bộ lọc trung bình di động**

Để giảm nhiễu và tín hiệu giả, dữ liệu G-force thô được đưa qua bộ lọc trung bình di động:

float updateFilter(float newReading) {

// Cập nhật tổng bằng cách trừ giá trị cũ và thêm giá trị mới

accelSum = accelSum - accelReadings[readIndex] + newReading;

// Lưu giá trị mới vào bộ đệm

accelReadings[readIndex] = newReading;

// Cập nhật chỉ số cho lần đọc tiếp theo

readIndex = (readIndex + 1) % NUM\_SAMPLES;

// Tính giá trị trung bình

filteredGForce = accelSum / NUM\_SAMPLES;

return filteredGForce;

}

Bộ lọc này:

* Sử dụng 10 mẫu gần nhất (NUM\_SAMPLES)
* Tính giá trị trung bình để loại bỏ nhiễu ngẫu nhiên
* Tạo đường cơ sở (baseline) ổn định để so sánh khi có sự thay đổi đột ngột

**3.3. Thuật toán phát hiện đỉnh (Peak Detection)**

Phát hiện va chạm dựa trên thuật toán phát hiện đỉnh tinh vi:

bool detectImpact() {

// Kiểm tra ngưỡng sơ bộ để tối ưu hiệu suất

if (filteredGForce < crashThreshold \* 0.7f) {

// Reset buffer nếu giá trị quá thấp để tránh nhiễu

if (filteredGForce < 1.0f) {

for (int i = 0; i < IMPACT\_WINDOW; i++) {

peakValues[i] = filteredGForce;

}

}

return false;

}

// Lưu giá trị hiện tại vào bộ đệm cửa sổ va chạm

peakValues[peakIndex] = filteredGForce;

peakIndex = (peakIndex + 1) % IMPACT\_WINDOW;

// Tìm giá trị cao nhất trong cửa sổ

float maxVal = 0;

for (int i = 0; i < IMPACT\_WINDOW; i++) {

if (peakValues[i] > maxVal) {

maxVal = peakValues[i];

}

}

// Kiểm tra ngưỡng trước khi tiếp tục xử lý

if (maxVal < crashThreshold) {

return false;

}

// Tìm đường cơ sở (baseline) bằng cách lấy giá trị nhỏ nhất

float minVal = 100.0f;

for (int i = 0; i < IMPACT\_WINDOW; i++) {

if (peakValues[i] < minVal && peakValues[i] > 0.5f) {

minVal = peakValues[i];

}

}

// Đảm bảo baseline hợp lý

float baseline = (minVal > 1.0f) ? minVal : 1.0f;

// Tính tỷ lệ tăng đột biến

float spikeRatio = maxVal / baseline;

// Phân tích nhiều tiêu chí để xác định va chạm

bool isPeakRecent = (filteredGForce > maxVal \* 0.7f); // Đảm bảo đỉnh là gần đây

bool isAboveThreshold = (maxVal > crashThreshold);

bool isHighRatio = (spikeRatio > IMPACT\_THRESHOLD);

// Chỉ khi tất cả điều kiện thỏa mãn mới xác nhận va chạm

if (isAboveThreshold && isHighRatio && isPeakRecent) {

return true;

}

return false;

}

**4. Tiêu chí xác định va chạm**

Thuật toán sử dụng 3 tiêu chí chính để xác định va chạm thực sự:

1. **Vượt ngưỡng (isAboveThreshold)**: Giá trị G-force phải vượt ngưỡng do người dùng cài đặt (mặc định 4.0G).
2. **Tỷ lệ tăng cao (isHighRatio)**: Tỷ lệ giữa giá trị đỉnh và đường cơ sở phải vượt ngưỡng IMPACT\_THRESHOLD (1.5). Điều này đảm bảo phát hiện sự thay đổi đột ngột đặc trưng của va chạm, thay vì chỉ dựa vào giá trị tuyệt đối.
3. **Đỉnh xảy ra gần đây (isPeakRecent)**: Giá trị hiện tại phải gần với giá trị đỉnh (> 70% giá trị đỉnh), đảm bảo va chạm vừa mới xảy ra, không phải giá trị đỉnh cũ lưu trong bộ đệm.

**5. Cơ chế chống nhiễu và ngăn cảnh báo lặp lại**

Để tránh nhiều cảnh báo cho một sự kiện va chạm, hệ thống có cơ chế "debounce":

// Trong hàm checkForCrash

if (crashAlarmActive) {

// Nếu đang trong trạng thái báo động, kiểm tra thời gian cooldown

if (currentTime - crashDetectedTime > crashCooldownTime) {

// Hết thời gian cooldown, reset trạng thái

crashAlarmActive = false;

crashNotified = false;

}

return; // Bỏ qua phát hiện va chạm khi đang trong trạng thái báo động

}

* Sau khi phát hiện va chạm, hệ thống chuyển sang trạng thái "báo động" (crashAlarmActive = true)
* Trong khoảng thời gian "cooldown" (10 giây), không xử lý thêm va chạm nào
* Giúp tránh gửi nhiều thông báo liên tiếp cho cùng một sự kiện

**6. Điều chỉnh độ nhạy**

Độ nhạy của thuật toán có thể được điều chỉnh qua giao diện người dùng:

class SensitivityCallbacks: public BLECharacteristicCallbacks {

void onWrite(BLECharacteristic \*pCharacteristic) {

std::string value = pCharacteristic->getValue();

if (value.length() > 0) {

// Parse float value từ string

crashThreshold = atof(value.c\_str());

// Lưu vào preferences

preferences.begin("bike-safety", false);

preferences.putFloat("crash\_threshold", crashThreshold);

preferences.end();

}

}

};

* Người dùng có thể điều chỉnh ngưỡng va chạm (crashThreshold) từ 2.0G (nhạy cao) đến 10.0G (nhạy thấp)
* Giá trị này được lưu trong bộ nhớ không bay hơi để duy trì sau khi khởi động lại
* Ứng dụng cung cấp giao diện trực quan thông qua thanh trượt để điều chỉnh

**7. Quy trình xử lý khi phát hiện va chạm**

Khi xác định có va chạm xảy ra, hệ thống thực hiện các bước sau:

if (detectImpact()) {

// Chỉ xử lý nếu không trong thời gian debounce

if (currentTime - lastCrashTime > DEBOUNCE\_TIME) {

lastCrashTime = currentTime;

crashDetectedTime = currentTime;

crashAlarmActive = true; // Chuyển sang trạng thái báo động

// Chỉ gửi thông báo BLE một lần

if (deviceConnected && !crashNotified) {

uint8\_t crashValue = 1;

pCrashChar->setValue(&crashValue, 1);

pCrashChar->notify();

crashNotified = true;

// Reset crash value after notification

delay(100);

crashValue = 0;

pCrashChar->setValue(&crashValue, 1);

}

// Sound emergency alarm

for (int i = 0; i < 10; i++) {

digitalWrite(BUZZER\_PIN, HIGH);

delay(150);

digitalWrite(BUZZER\_PIN, LOW);

delay(100);

}

}

}

1. **Gửi thông báo BLE**: Thiết bị gửi thông báo đến ứng dụng di động qua kết nối Bluetooth
2. **Kích hoạt còi báo động**: Phát âm thanh cảnh báo từ thiết bị
3. **Ứng dụng di động nhận thông báo**: Xử lý và gửi cảnh báo đến người thân
4. **Xác định vị trí GPS**: Lấy tọa độ hiện tại để đưa vào thông báo
5. **Gửi thông báo khẩn cấp**: Gửi email hoặc tin nhắn Telegram đến danh sách người thân

**8. Tối ưu hóa hiệu suất và tiết kiệm năng lượng**

Thuật toán được tối ưu về hiệu suất và tiết kiệm năng lượng:

* **Kiểm tra sơ bộ**: Bỏ qua phân tích phức tạp nếu giá trị G không vượt 70% ngưỡng
* **Tần số lấy mẫu tối ưu**: 100Hz, đủ để phát hiện va chạm nhưng không tốn quá nhiều năng lượng
* **Bộ lọc thông thấp**: Giảm nhiễu và tác động của rung động thông thường
* **Kỹ thuật debounce**: Tránh xử lý liên tục và tiết kiệm năng lượng sau khi phát hiện va chạm

Qua việc kết hợp các kỹ thuật xử lý tín hiệu tiên tiến và thuật toán phát hiện đỉnh thông minh, hệ thống Bike Safety có thể phân biệt chính xác giữa va chạm thực sự và các dao động thông thường, mang lại độ tin cậy cao trong việc phát hiện tai nạn cho người sử dụng xe đạp.