# BalanceBot 신규 기능 개발 요청서

**문서 ID:** FRD-20251008-01

**작성일:** 2025년 10월 8일

**상태:** 신규 요청

## 1. 개요

본 문서는 BalanceBot 프로젝트의 기능 확장을 위한 신규 개발 항목을 정의합니다. README.md에 명시된 미구현 기능들을 바탕으로, 각 기능의 상세 요구사항, 구현 위치, 우선순위를 명시하여 체계적인 개발을 목표로 합니다.

## 2. 개발 요청 목록 (우선순위 순)

### 2.1. [요청 ID: FEAT-UI-001] 사용자 인터페이스 시스템

* **우선순위: HIGH**
* **요약:** 물리적 버튼과 RGB LED를 통한 직관적인 사용자 인터페이스를 구현합니다. 이는 로봇의 기본 조작 및 상태 확인에 필수적입니다.
* **상세 요구사항:**
  1. **버튼 입력 시스템 (main/input/button\_control.h/c)**
     + GPIO 12번 핀을 사용합니다.
     + **단순 클릭**: 로봇 시작/정지(Balancing) 토글 기능을 구현합니다.
     + **더블 클릭 (500ms 내)**: GPS 추적 모드 ON/OFF 토글 기능을 구현합니다.
     + **길게 누르기 (2초 이상)**: 서보 기립 시퀀스를 수동으로 실행합니다.
     + 채터링(Chattering) 현상 방지를 위한 디바운싱(Debouncing) 로직을 적용해야 합니다.
     + GPIO 인터럽트 기반으로 구현하여 CPU 부하를 최소화합니다.
  2. **RGB LED 상태 표시 시스템 (main/output/rgb\_led.h/c)**
     + GPIO 14(R), 15(G), 16(B) 핀을 사용하며, PWM으로 제어합니다.
     + 메인 상태 머신(main.c의 robot\_state\_t)과 연동하여 아래 표에 따라 색상을 자동으로 변경해야 합니다.

|  |  |  |
| --- | --- | --- |
| 색상 | 상태 | 설명 |
| 🔴 빨강 | ERROR / FALLEN | 오류 또는 넘어짐 |
| 🔵 파랑 | IDLE | 대기 상태, BLE 연결 대기 |
| 🟢 초록 | BALANCING | 정상 밸런싱 중 |
| 🟣 보라 | GPS\_TRACKING | GPS 추적 모드 |
| 🟡 노랑 | STANDING\_UP | 서보 기립 중 |
| 🌈 무지개 | INIT | 시스템 초기화(애니메이션) |

**3. BLE 프로토콜 연동**

* + - 모바일 앱에서 LED 색상과 밝기를 직접 제어할 수 있는 LED\_SET\_COLOR, LED\_SET\_BRIGHTNESS 명령을 처리해야 합니다.
    - 버튼 입력 발생 시 앱으로 BUTTON\_PRESSED 상태를 전송해야 합니다.
* **구현 위치:**
  + **ESP32:** main/input/button\_control.h/c, main/output/rgb\_led.h/c (신규 생성)
  + **POSIX:** posix\_simulator/input/button\_control.h/c, posix\_simulator/output/rgb\_led.h/c (시뮬레이션용 신규 생성)

### 2.2. [요청 ID: FEAT-SERVO-002] 서보 기립 제어 시스템

* **우선순위: HIGH**
* **요약:** 로봇이 넘어졌을 때 서보 모터를 이용해 자동으로 일어나는 시스템을 구현합니다. 이는 로봇의 실용성과 자율성을 크게 향상시킵니다.
* **상세 요구사항:**
  1. **기립 상태 머신 (main/output/servo\_standup.h/c)**
     + EXTENDING → PUSHING → RETRACTING → COMPLETE 순서의 3단계 기립 시퀀스를 구현합니다.
     + 각 단계의 지속 시간은 타이머로 관리하며, config.h에서 조정 가능해야 합니다.
  2. **시스템 통합**
     + 메인 상태 머신(main.c)과 연동하여, FALLEN 상태에서 기립 명령 수신 시 STANDING\_UP 상태로 전환되도록 구현합니다.
     + 기립 동작 중에는 PID 밸런싱 제어와 모터 구동을 일시적으로 중지해야 합니다.
  3. **BLE 프로토콜 연동**
     + 모바일 앱에서 SERVO\_STANDUP 명령을 수신하면 기립 시퀀스를 시작해야 합니다.
     + 기립 진행 상황(EXTENDING, PUSHING 등)을 SERVO\_STATUS 응답으로 앱에 주기적으로 전송해야 합니다.
* **구현 위치:**
  + **ESP32:** main/output/servo\_standup.h/c (신규 생성)
  + **POSIX:** posix\_simulator/output/servo\_standup.h/c (시뮬레이션용 신규 생성)

### 2.3. [요청 ID: FEAT-GPS-003] GPS 사람 추적 시스템

* **우선순위: MEDIUM**
* **요약:** GPS를 이용하여 지정된 타겟(사람)을 자동으로 따라가는 자율 주행 기능을 구현합니다.
* **상세 요구사항:**
  1. **추적 알고리즘 (main/logic/gps\_navigation.h/c)**
     + 현재 로봇의 GPS 좌표와 수신된 타겟의 GPS 좌표 간의 거리와 방향(방위각)을 계산하는 알고리즘을 구현합니다.
     + 계산된 방향 오차를 입력으로 받는 PID 제어기를 구현하여 로봇의 회전(Turn) 값을 출력해야 합니다.
  2. **시스템 통합**
     + 메인 제어 로직(main.c)에 통합되어, GPS 추적 모드가 활성화되면 방향 제어 PID의 출력이 모터 제어에 반영되어야 합니다.
     + 타겟과의 거리가 일정 수준 이하로 가까워지면 정지하는 안전 로직을 포함해야 합니다.
  3. **BLE 프로토콜 연동**
     + GPS\_SET\_TARGET, GPS\_START\_FOLLOW, GPS\_STOP\_FOLLOW 명령을 처리해야 합니다.
     + GPS\_FOLLOW\_STATUS (추적 상태), GPS\_DISTANCE (타겟까지의 거리) 정보를 주기적으로 앱에 전송해야 합니다.
* **구현 위치:**
  + **ESP32:** main/logic/gps\_navigation.h/c (신규 생성)
  + **POSIX:** posix\_simulator/logic/gps\_navigation.h/c (시뮬레이션용 신규 생성)

## 3. 기여 가이드라인

모든 신규 기능 개발은 아래의 가이드라인을 준수해야 합니다.

1. **BSW 아키텍처 준수:** 기능은 input, logic, output 계층에 맞게 분리하여 구현합니다.
2. **멀티태스킹 안전성:** FreeRTOS 환경에서 동작하므로, 공유 자원 접근 시 반드시 뮤텍스를 사용합니다.
3. **POSIX 호환성:** ESP32 펌웨어와 POSIX 시뮬레이터 코드를 동시에 구현하여 PC 환경에서 우선 테스트합니다.
4. **문서화:** 모든 새로운 함수, 구조체, 파일에는 Doxygen 형식의 주석을 상세히 작성합니다.