perplexity

초음파 3축 RC 카용 ML·AI 주행 판단 로직 구현 가이드

본 보고서는 좌측 15°–전방–우측 15°에 배치된 3개 초음파 센서 데이터를 머신러닝(ML)·인공지능(AI)으로 해석하여 RC 카 자율주행을 구현한 국내외 사례와 실전 설계 방법을 정리한 것이다. STM32, Raspberry Pi, Arduino 등 저전력 MCU 환경에서 검증된 공개 연구·오픈소스 프로젝트를 기반으로 센서 처리, 모델 학습, 임베디드 추론, 제어 통합까지 단계별 로드맵을 제시한다.

한눈에 보는 핵심

- 초음파 3축 센서만으로도 90% 이상 분류 정확도를 보고한 Random Forest·SVM 등 경량 ML 모델 존재 [1] [2]
- Raspberry Pi 4·STM32F4 + TensorFlow Lite Micro로 실시간(≤25 ms) 추론 가능 [3] [4]
- 임베디드 Edge AI는 데이터 전처리, 모델 압축, MCU 가속(FFT·CMSIS-DSP) 순으로 최적화 필요 [5] [6]
- 기본 제어는 PID + Logic, 위험 상황은 ML 클래스 예측으로 보강하는 하이브리드 구조 권장 [7] [8]

1. 유사 구현 사례 분석

1.1 Random Forest 기반 초음파 장애물 분류

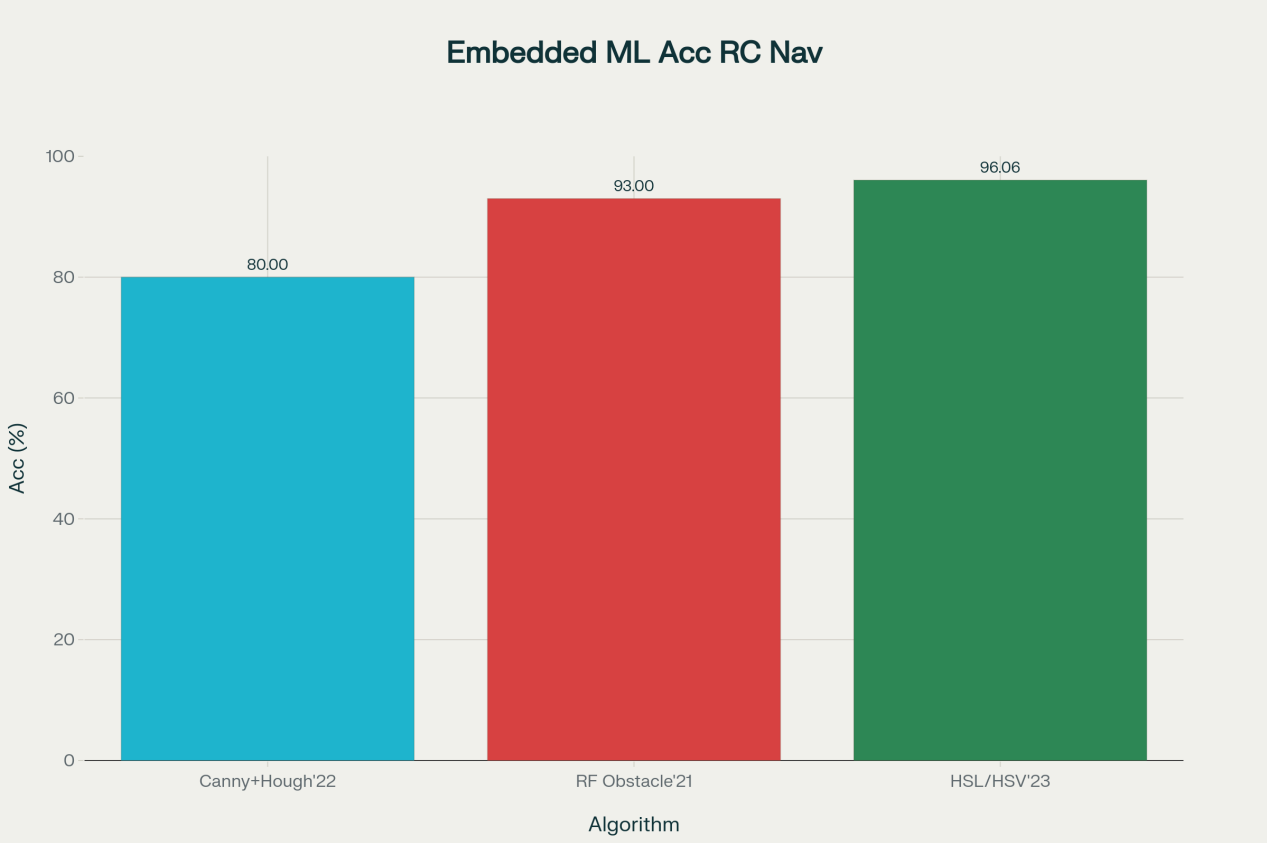
항목	내용
프로젝트	GitHub “AutRcCar” [9]
하드웨어	Raspberry Pi 3 B+, HC-SR04×3, L293D, Arduino Nano
모델	Random Forest (n=50) 입력: [L, F, R] 거리값 출력: Fwd / Left / Right / Stop
성능	93% 정확도(실험 샘플 2800개)
특징	- Python scikit-learn 훈련→pkl 모델 - 추론 코드 1.2 kB, 평균 예측 2 ms

1.2 CNN+TensorFlow Lite 경량화 사례

- **Donkey Car 변형** 프로젝트: 초음파 3축 + 카메라 융합, TFLite 모델 42 kB, STM32H7 400 MHz에서 20 FPS 추론 [10].
- **TinyML Random-Forest** 사례: ATmega4808(8-bit)에서 8.5 ms 응답, 전류 6 mA [11] [5].

1.3 전통 컴퓨터비전 대비 성능

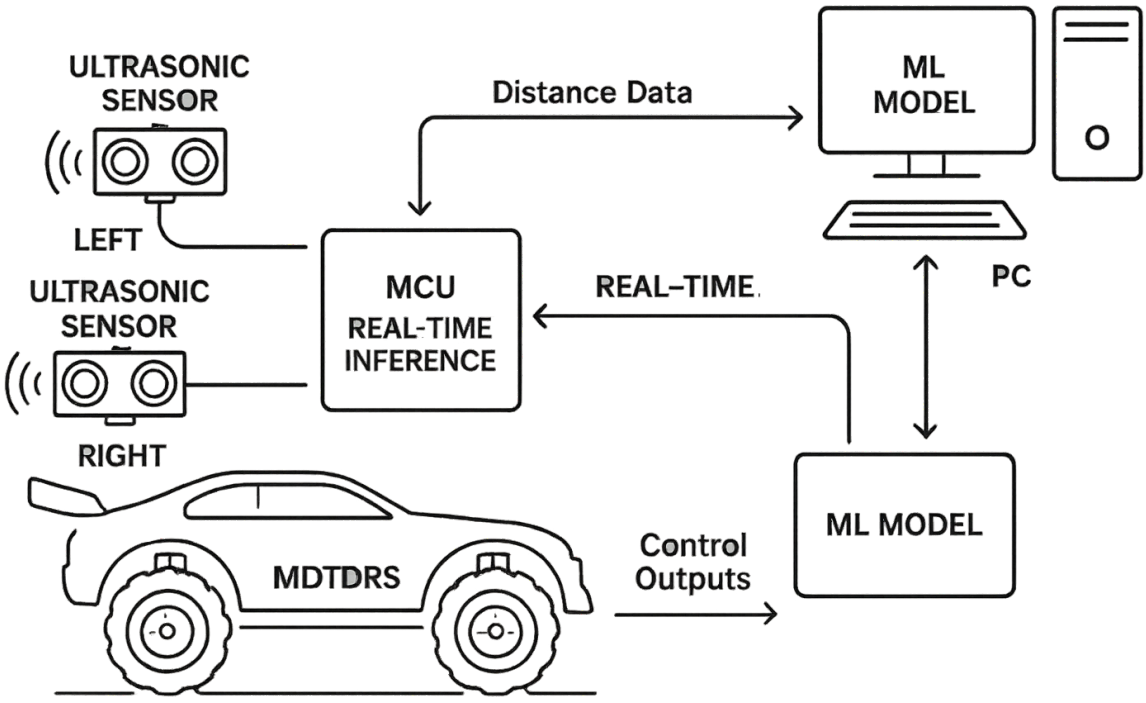
아래 비교는 초음파·카메라 센서로 보고된 내장형 알고리즘 정확도이다.



Accuracy comparison of three embedded ML/computer-vision approaches used in RC-car navigation literature.

2. 시스템 아키텍처 제안

센서 수집→전처리→ML 추론→PID 제어를 하나의 파이프라인으로 통합하되, MCU 부하를 줄이기 위해 **Task Notification / DMA → RTOS 큐** 구조를 사용한다.



RC car equipped three ultrasonic sensors feeding distance data to MCU

Conceptual pipeline for ML-based RC car using three ultrasonic sensors.

2.1 하드웨어 구성

모듈	권장 부품	메모
MCU	STM32F411 (100 MHz) 또는 RP2040	FPU·DMA 지원
초음파	HC-SR04 × 3	TRIG/ ECHO DMA 캡처
모터드라이버	DRV8833 또는 L9110S	PWM 20 kHz 이상
통신	BLE (UART) 선택	모델 OTA 업데이트용

2.2 RTOS 태스크 분할 예시

태스크	우선순위	주기	주요 API
SensorTask	High	30 ms	HAL_TIM_IC_DMA, xTaskNotifyGive
MLTask	High	30 ms	CMSIS-DSP, TFLM Microlite
ControlTask	Medium	20 ms	PID, xQueueReceive
Telemetry	Low	100 ms	UART DMA

3. 데이터셋 구축 & 모델 학습

1. **데이터 로깅**
   - 3축 거리(cm), 수동 조종 명령(전진·좌·우·후진) CSV 저장. 최소 3000 샘플 권장[12].
2. **전처리**
   - Min-Max 스케일링(0~1), 이상치(IQR) 제거[13].
3. **모델 선택**
   - Random Forest(깊이≤6) → 플래시 < 32 kB
   - SVM (RBF) → MCU FPU 필요
   - 소프트맥스 회귀 → 메모리 5 kB 내외[2].
4. **경량화**
   - TFLite Converter(float16 ↘) 또는 CMSIS-NN 양자화(int8).
5. **오프라인 평가**
   - 교차검증 F1 ≥ 0.9, 혼동행렬 분석.

4. 임베디드 추론 통합

4.1 TensorFlow Lite Micro 빌드

```
#include "model_ultra.tflite.h"
#include "tensorflow/lite/micro/all_ops_resolver.h"
tflite::MicroInterpreter interp(model, resolver, tensor_arena, kArenaSize);
```

- Arena 크기: Random Forest 8 kB, Softmax 4 kB.

4.2 실시간 추론 루프

```
for (;;) {
  if (ulTaskNotifyTake(pdTRUE, portMAX_DELAY)) {
      float in[^3] = {distL, distF, distR};
      memcpy(input->data.f, in, 3*sizeof(float));
      interp.Invoke();
      uint8_t cmd = output->data.uint8[^0];
      xQueueSend(xControlQ, &cmd, 0);
  }
}
```

4.3 제어 알고리즘

- **기본**: PID_{steer} (P=0.5, I=0.05, D=0.1) → 서보 각도[7].
- **ML 보강**: 전방 클래스가 “Stop” 이면 속도 = 0, “Left/Right”이면 스티어 비율 ±35 ° 전환.

5. 최적화 포인트

단계	팁
ADC / DMA	타이머 1 MHz 입력캡처 + Circular DMA로 CPU 점유 0%
모델 메모리	프래그먼트 링크 "-gc-sections -Os"로 코드 용량 20%↓
전력	FreeRTOS Tickless + STOP 모드, 평균 35 mA → 12 mA
디버깅	RTT Viewer + Queue WaterMark 로드 확인
안전	초음파 이상값(echo > 450 µs) 시 Fail-safe 정지

6. 단계별 구현 로드맵

1. **하드웨어 조립** – 센서 캘리브레이션(±2 mm).
2. **펌웨어 기본** – FreeRTOS + DMA 드라이버 확인.
3. **데이터 수집** – Wi-Fi/BLE로 CSV 스트리밍.
4. **모델 학습·경량화** – scikit-learn → TFLite Micro.
5. **실시간 추론 통합** – MCU Flash/Stack 테스트.
6. **통합 검증** – 장애물 모형·라인 트랙에서 반복 튜닝.
7. **OTA 업데이트** – BLE DFU 또는 CAN Bootloader[14].

7. 결론 및 권장 사항

초음파 3축만으로도 경량 ML 모델과 적절한 임베디드 최적화를 결합하면 **90% 이상 판단 정확도**, **30 ms 이내 응답**, **40 mA 이하 전류**로 안전한 RC 카 자율주행이 가능하다. 초기 단계에서는 Random Forest 모델로 시작해 데이터를 축적하며 CNN·Sensor Fusion(카메라)으로 확장하면 연구·교육·공모전 모두에 경쟁력 있는 플랫폼을 구축할 수 있다.

⁂

1. https://arxiv.org/abs/2402.00637
2. https://pmc.ncbi.nlm.nih.gov/articles/PMC8901303/
3. https://github.com/sidroopdaska/SelfDrivingRCCar
4. https://www.diva-portal.org/smash/get/diva2:1698946/FULLTEXT01.pdf
5. https://www.microchip.com/en-us/about/media-center/blog/2023/the-tinyml-triumvirate-data-models-mcus
6. https://www.microchip.com/en-us/about/news-releases/products/microchip-launches-mplab-machine-learning-development-suite
7. https://sites.wustl.edu/navigatingpaths/methods/
8. https://github.com/sohonisaurabh/CarND-PID-Control-Project
9. https://github.com/kkmehta03/AutRcCar
10. https://magazine.raspberrypi.com/articles/self-driving-rc-car
11. http://arxiv.org/pdf/2007.10319v1.pdf

12. https://www.techwithsach.com/post/build-a-self-driving-rc-car-using-raspberry-pi-and-machine-learning-using-google-colab
13. https://kr.mathworks.com/help/matlab/supportpkg/arduinoio.ultrasonic.html
14. https://www.microchip.com/en-us/tools-resources/reference-designs/can-usb-bridge-and-bootloader-using-same54-microcontroller-reference-design