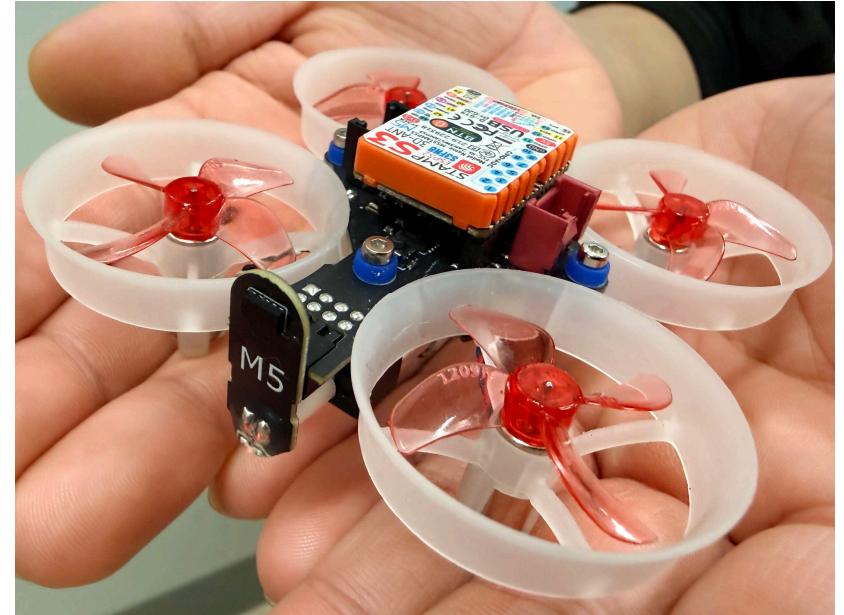


StampFly Ecosystem

ドローン制御を「学び、作り、飛ばす」オープンプラットフォーム



なぜ作ったのか？

ドローン制御を学びたい人の壁

「PID制御を教科書で学んだ。
でも、実際に飛ぶものを作るにはどうすれば...？」

- 市販ドローンはブラックボックス
- 自作はハードウェアから全部やる必要がある
- シミュレータと実機の間に大きなギャップ

私たちの答え

「制御だけに集中できる」プラットフォーム

- ハードウェアは完成品を提供
- ソフトウェアは「スケルトン」を提供
- あなたが実装するのは、制御アルゴリズムだけ

StampFly Ecosystem とは？

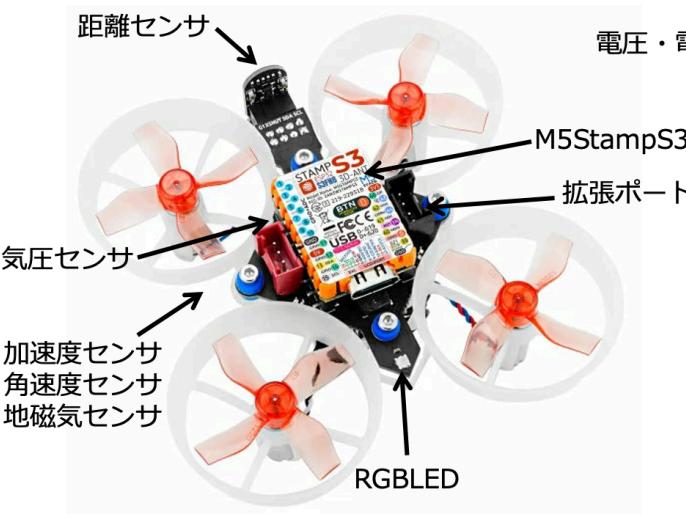
3つの柱

柱	内容
ハードウェア	35gマイクロドローン + 専用コントローラ
ファームウェア	ESP-IDF + FreeRTOS ベースの拡張可能なコード
シミュレータ	実機なしで制御を検証できる3D環境

ハードウェア： StampFly

- 質量: 35g
- MCU: ESP32-S3
- センサ:
 - IMU (BMI270)
 - 気圧計 (BMP280)
 - 地磁気 (BMM150)
 - ToF (VL53L3CX)
 - オプティカルフロー (PMW3901)

StampFly



ファームウェア：スケルトン設計

あなたが実装する部分

- ・姿勢制御（角度→角速度）
- ・位置制御（位置→姿勢）
- ・高度制御（ToF→スロットル）

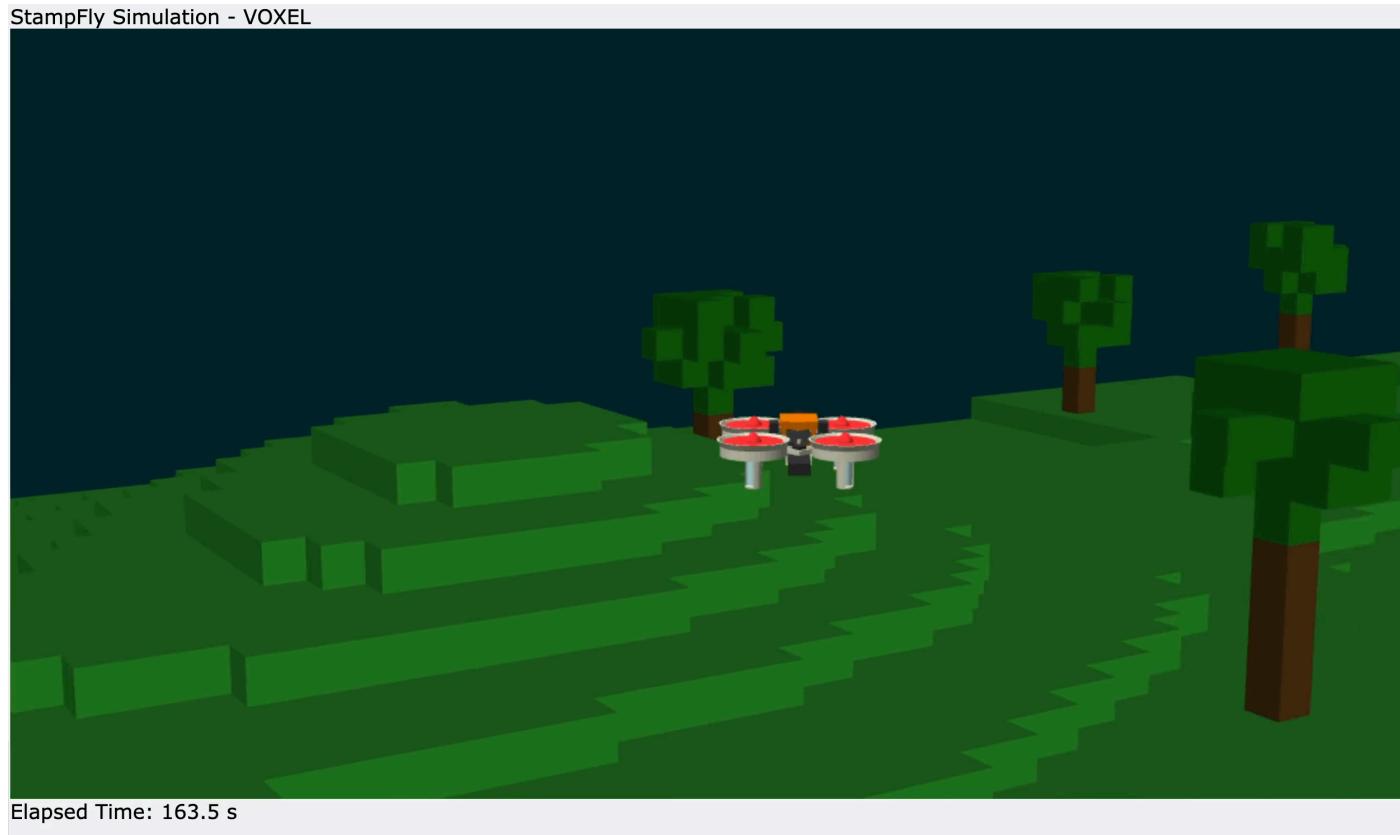
↓ 指令値

提供済みの部分

- ・角速度制御（PID実装済み）
- ・センサ読み取り・姿勢推定
- ・モーターミキサー・通信

シミュレータ：実機なしで検証

- コントローラをUSB接続して操作
- 3Dビューでリアルタイム飛行
- 衝突判定・ランダム地形生成
- 制御パラメータの調整を安全に



誰のためのプロジェクト？

ターゲットユーザー

ユーザー	ニーズ
学生	制御工学の実践的な学習
研究者	飛行実験プラットフォーム
教育者	制御理論の教材
ホビイスト	自分だけの飛行制御を作る楽しさ

教育での活用例

授業シナリオ

1. Week 1-2: シミュレータで基礎操作
2. Week 3-4: PID制御の理論学習
3. Week 5-6: シミュレータで姿勢制御を実装
4. Week 7-8: 実機で検証・調整

座学 → シミュレーション → 実機 の流れ

現在の状態

実装済み機能

カテゴリ	機能	状態
機体FW	角速度制御 (ACRO)	完成
機体FW	センサ読み取り・姿勢推定	完成
機体FW	WiFiテレメトリー	完成
コントローラ	ESP-NOW通信	完成
コントローラ	USB HIDモード	完成
コントローラ	メニューシステム	完成
シミュレータ	3D飛行・衝突判定	完成

これから（ユーザー実装を想定）

カテゴリ	機能	状態
機体FW	姿勢制御 (STABILIZE)	 ユーザー実装を想定
機体FW	高度制御	 ユーザー実装を想定
機体FW	位置制御	 ユーザー実装を想定
解析	フライトログ可視化	 計画中
ドキュメント	チュートリアル	 計画中

デモ

シミュレータデモ

- コントローラをUSB HIDモードでPC接続
- `python run_sim.py` で起動
- 3Dビューでドローンを操作
- 衝突判定・地形生成を確認

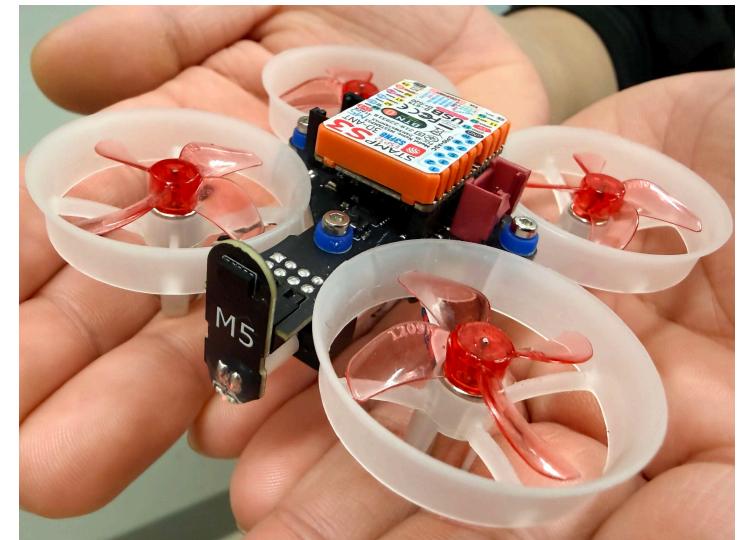
実演します！

まとめ

StampFly Ecosystem

「制御を学びたい」すべての人へ

- 実機がなくてもシミュレータで始められる
- 制御だけに集中できるスケルトン設計
- オープンソースで自由に改変可能
- 教育・研究・趣味、どんな用途にも



Thank You

GitHub: M5Fly-kanazawa/stampfly_ecosystem

Contact: kouhei.ito@itolab-ktc.com

一緒に、ドローン制御の学びを変えましょう

補足資料

技術仕様

項目	仕様
MCU	ESP32-S3 (M5Stamp S3)
フレームワーク	ESP-IDF v5.4.1 + FreeRTOS
姿勢推定	ESKF (Error-State Kalman Filter)
通信	ESP-NOW (50Hz) + WiFi (テレメトリー)
センサ	BMI270, BMM150, BMP280, VL53L3CX, PMW3901

リポジトリ構成

```
stampfly_ecosystem/
└── firmware/
    └── vehicle/      # 機体ファームウェア
        └── controller/ # コントローラファームウェア
    └── simulator/   # 3Dフライトシミュレータ
    └── docs/         # ドキュメント
    └── protocol/    # 通信プロトコル仕様
```

クイックスタート

```
# シミュレータを試す  
cd simulator/scripts  
pip install -r ./requirements.txt  
python run_sim.py
```

```
# ファームウェアをビルド  
cd firmware/vehicle  
idf.py build flash monitor
```

詳細: [docs/getting-started.md](#)