

批判的意見まとめ：high-precision-sensor / table-level

作成日: 2026-02-25

対象: High Precision Tilt Sensor PWA (v1.2) および Table Level Guide (v1.0)

はじめに

本書は「プロトタイプの問題を直視し、次の改善と意思決定の材料にする」ことを目的とする。批判は否定ではなく、健全な品質向上のための行為として位置づける。

第1部：high-precision-sensor への批判的意見

1. 精度の主張に根拠が薄い

1-1. 「分解能 0.01°」は数値表示桁数であって、計測精度ではない

- スマートフォンの MEMS センサー（DeviceOrientation API）は、ISO 規格の計測器ではない。
- ソフトウェアフィルタで数値を平滑化しても、物理的な精度が向上するわけではない。
- 「0.01° の分解能」は小数点以下の桁数の話に過ぎず、測定不確かさ（uncertainty）の評価が欠如している。
- キャリブレーション後の相対精度「±0.1°」の根拠も、独立した第三者検証が存在しない。

1-2. 端末個体差の矮小化

- 同じアプリを異なる機種で動かした場合の系統誤差は不明であり、機種間比較結果が公表されていない。
- 「スマホで代替できる」という主張は、個体差・温度ドリフト・経年劣化を考慮した上での評価が必要。

1-3. 既存計測器との比較データの欠如

- デジタルプロトラクター（数万円）との比較試験が行われていない。
- 「専用計測器の代替になる」という訴求には、精度・応答速度・耐環境性の比較実験が必要。

2. 実運用への配慮不足

2-1. 現場で誤操作が起きやすい設計

- センサー許可を拒否した場合の復旧導線が「知っている人向け」で、現場作業者には不明瞭。
- 2点キャリブレーション手順が-toast通知に依存しており、現場で見落とされやすい。
- 計測モード（active / locking / measuring）が開発者語で、作業者が状態を理解しにくい。

2-2. バックグラウンド動作の制限

- タブが非アクティブになると requestAnimationFrame が停止し、データロギングも止まる。
- 長時間計測や他アプリ操作しながらの記録ができず、実現場での継続運用が困難。

2-3. 横画面・環境対応の未整備

- 現場では端末を横向きに固定するケースも多いが、横画面対応がスコープ外のまま。
- 建設現場の振動環境での静止判定閾値は、デフォルト値では機能しない可能性が高い。

2-4. 電源断・強制終了時のデータ保護

- 計測中に電源が切れた場合のデータ保全設計が明確でない。
- IndexedDB への自動保存間隔・落ちた場合のリカバリ手順が未整備。

3. コード品質・保守性の問題

3-1. `app.js` の責務過多（改善中だが根本は残る）

- センサー制御・UI・音声・永続化・ライフサイクルを1クラスで管理する構造は、機能追加の度に回帰リスクを生む。
- コードレビューで指摘後に分割対応はされたが、アーキテクチャの思想としてはまだモノリシック傾向が残る。

3-2. TypeScript 非採用による型安全性の欠如

- モジュール間インターフェースを型で保証できておらず、リファクタリング時の不整合は実行時まで発見できない。
- JSDoc 導入も「見送り」扱いで、将来の拡張が困難になる懸念がある。

3-3. Service Worker のキャッシュリストが手動管理

- アセット追加時の手動追記漏れは、「オフラインだけ壊れる」問題の再発源となる。
- Workbox 等のビルド連携が「見送り」のままで、PWA の信頼性が脆すぎる。

3-4. UI・音声制御のテストカバレッジ不足

- ロジック層のテストは充実しているが、`UIManager` と `AudioEngine` の単体テストがない。
- DOM 更新・音声出力系のバグは E2E に頼らざるを得ず、根因特定コストが高い。

4. ビジネス・展開面の問題

4-1. ユーザー層・ペルソナの不明確さ

- 「建設現場の水平出し」「DIY」「設備傾斜監視」と用途が広すぎて、UXが誰に向けたものか曖昧。
- 精度要件・操作習熟度のターゲットが絞り込まれておらず、結果として「惜しいが刺さらない」アプリになっている。

4-2. 「MIT License オープンソース」の位置づけが曖昧

- 事業展開（受注、外注委託、商用利用）とオープンソース公開を同時に行う場合の整合性が未検討。
- ライセンスの選択が深く議論されておらず、後で問題になる可能性がある。

4-3. 継続的なメンテナンス体制が未確立

- 開発者1名で、スマートフォン OS のアップデートに追従し続けられるかの計画がない。
 - DeviceOrientation API は iOS の仕様変更の影響を受けやすく、過去にも破壊的変更が発生している。
-

第2部：table-level への批判的意見

5. アプリとしての独自性・差別化の弱さ

5-1. high-precision-sensor からの焼き直し感が強い

- センサー処理・音声ガイド・PWA 構成は high-precision-sensor とほぼ同一であり、「新しいアプリ」としての独自価値が薄い。
- calculator.js（ボルト回転量計算）が中核だが、これ自体はシンプルな三角関数計算に過ぎない。
- 「スピノフアプリ」の位置づけが未整理で、ユーザーがどちらを使えばいいか判断しにくい。

6. ボルト回転量計算の信頼性

6-1. 計算モデルが単純すぎる

- 机の4隅の高さを2軸傾斜角（Pitch/Roll）から近似しているが、机の天板が剛体でない場合や脚の配置が中心対称でない場合への考慮が不明。
- 「最小変位・CW のみ」戦略でボルトを一方方向にのみ回すことを前提にしているが、摩擦やメカニカルな劣化による予期しない挙動への言及がない。

6-2. 計算精度の限界が未提示

- センサー精度 $\pm 0.1^\circ$ の場合、奥行き 600mm の机では高さ誤差が最大 $\pm 1\text{mm}$ 以上になる。
- ボルトピッチ 1mm の M6 ネジで 1mm の誤差は「1回転の違い」になりうるが、この限界が UI 上で示されていない。

6-3. 机の寸法設定の検証困難

- ユーザーが入力する「机の寸法」（幅・奥行き）が正確でない場合、計算結果が大きくブレる。
- 入力値の妥当性チェック（現実的な机サイズの範囲）が不明確。

7. 実運用・UX の問題

7-1. スマホを机に「置く」という前提の脆弱性

- 計測中にスマホが動く（振動、ケースの滑り）と正確な測定ができないが、設置状態の確認手段がない。
- テーブル上に置いた際の姿勢（縦置き・横置き・角度）による計測精度への影響が未評価。

7-2. 「自動確定」の判断根拠が不透明

- 静止判定後の自動確定タイミングが分散閾値ベースであり、「今の値を信頼してよいか」が数値でわからない。
- タイムアウト（手動確定）に頼るケースが多く、「自動判定の失敗」に対するユーザー体験が悪い。

7-3. ドキュメントの薄さ

- spec フォルダに 1 ファイル（SP-01_Overview.md）しかなく、アルゴリズム根拠・精度評価・ユーザーマニュアルが存在しない。
 - high-precision-sensor に比べてドキュメント整備が大幅に遅れており、保守・改善の基盤が弱い。
-

第3部：両アプリ共通の問題

8. 精度の第三者検証がない

- 両アプリとも「キャリブレーション後の精度」は開発者の自己評価に基づく。
- 独立した試験環境（回転台、精密水平器との比較）での評価が行われていない。

9. セキュリティ・プライバシーの検討不足

- CSP（Content Security Policy）ヘッダーが未設定で、防御層が薄い。
- センサーデータの取扱い方針（ローカルのみ、送信しない等）がプライバシーポリシーとして明文化されていない。

10. アクセシビリティへの配慮不足

- 視覚的に傾きを確認する手段は充実しているが、視覚障害者を対象とした設計は音声のみ。
- ダークモードのコントラスト比がすべての状態で十分か検証されていない。

11. 長期サポート・OSアップデート追従計画の欠如

- iOS / Android の OS アップデートに伴う API 互換性問題へのモニタリング体制がない。
 - 個人開発のため、メンテナンスが途絶えるリスクが非常に高い。
-

批判への反論：high-precision-sensor / table-level

作成日: 2026-02-25

対象: 01_CriticalReview.md への反論・補足

はじめに

批判的意見は改善を促す有益な視点だが、文脈を誤ると「プロトタイプに完成品の基準を適用する」誤謬に陥る。本書は各批判を公平に評価し、妥当な反論と受け入れるべき点を整理する。基本的な立場は「このアプリはモックアップ段階のプロトタイプであり、その段階での完成度としては相当高い」である。

第1部：high-precision-sensor への反論

批判1への反論：精度の主張について

「分解能 0.01° は桁数の話に過ぎない」への反論

- これは正しい批判であり、スペックシート（SP-02）にも明記されている。「分解能 0.01°」と「測定不確かさ」は別物であり、アプリは前者を主張している。
- ただし、静止平均モード（Hybrid Static）では理論上 $1/\sqrt{n}$ でノイズが低減され、長時間平均では精度が向上する。これは確率論的に根拠のある主張である。
- 受け入れるべき点: 不確かさの明示化とキャリブレーション状態の明記は今後の課題として妥当。

「端末個体差・第三者検証がない」への反論

- このアプリは「精密計測器」ではなく「ゼロコストで現場に配備できる補助計測ツール」として設計されている。目標とする比較相手は「デジタルプロトラクター」ではなく「気泡管水準器（目視・ $\pm 0.5^\circ$ ）」である。
- その対比では、カルマンフィルタ + 静止平均の組み合わせが有意に改善していることは自己評価でも確認されており、第三者検証の欠如は「主張の過大さ」の問題ではなく「説明の文脈設定」の問題。
- 妥当な反論: 比較対象の基準を明示すれば、精度の主張は現場の実用ツールとして十分に正当化できる。

批判2への反論：実運用への配慮不足について

「現場で誤操作が起きやすい」への反論

- 現場 UX の改善（F1~F5）はすでにレビュー（RV-05）で特定済みであり、未知の問題ではなくロードマップ上の既知の課題である。
- プロトタイプ段階では「動く・計れる・音が出る」ことが優先であり、誤操作耐性の完成は次フェーズとして合理的に分離されている。
- キャリブレーションウィザードはやや高難度のUX設計（P2, 12~18時間工数）であり、効果が大きいので意図的に後回しにされている。

「バックグラウンド動作の制限」への反論

- `requestAnimationFrame` の停止はブラウザの仕様であり、PWA の制約として不可避。
- 「長時間記録」ユースケースは現在のスコープ外であり、対応が必要と判断された場合は `setInterval` や Web Worker への移行が設計段階から想定されている（RV-01 § 3.2 参照）。

「横画面対応の未整備」への反論

- 横画面対応はスコープ外と明示されており（RV-01 §5）、これは意図的な判断。
- センサーの座標系と画面方向の組み合わせは端末依存・ブラウザ依存が大きく、正確な対応は工数対効果の検討が必要。「未実装 = 欠陥」ではなく「検討済みの保留」。

批判3への反論：コード品質・保守性について

「appjs の責務過多」への反論

- 辛口レビュー（RV-03）で指摘済みであり、ToastManager・LifecycleManager・AppEventBinder への分割・スキーマ駆動化を 2026-02-24 時点で完了済み。
- スコアも 65→82→88 と改善が記録されており、**批判が改善を生む健全なフィードバックループが機能している**。

「TypeScript 非採用」への反論

- 1人開発のプロトタイプ段階での TypeScript 導入は、設定コスト・学習曲線・既存 JS との橋渡し工数が発生する。
- JSDoc 導入は次段として記録されており、段階的移行の意図は明確。「採用していない＝品質が低い」は短絡的すぎる。

「Service Worker のキャッシュリストが手動管理」への反論

- 手動管理のリスクは認識済み（M1 指摘）で、Workbox 導入は機能安定後の中優先課題として位置づけられている。
- 現時点では E2E オフラインスモークテストが回帰検知ラインとして機能しており、**運用リスクをテストでカバーする代替戦略が取られている**。

批判4への反論：ビジネス・展開面について

「ユーザー層・ペルソナが不明確」への反論

- プロトタイプ段階では「できること」を示すことが先であり、ターゲットの絞り込みは本格開発フェーズで行う判断は合理的。
- 「建設現場・DIY・設備監視」が並列に挙げられるのは、アプリが汎用性を持つことの強みでもある。

「MIT License の位置づけが曖昧」への反論

- MIT License はシンプルで利用者フレンドリーな選択であり、商用利用も明示的に許可している。オープンソース＋商用展開の整合性はライセンスとして成立している。
 - ただし、「自社展開と外注委託の整合性」については別途検討の余地があり、批判の一部は妥当。
-

第2部：table-level への反論

批判5への反論：独自性・差別化について

「焼き直し感が強い」への反論

- `calculator.js` で実現する「ボルト回転量への変換」は、単なる三角関数ではなく「傾き角度 → 高さ差 → ボルト回転数」という実用変換であり、現場作業の認知負荷を現役の設備管理者基準まで引き下げる独自価値を持つ。
- 共通モジュール（`shared/js/`）の積極活用は「焼き直し」ではなく、ソフトウェアエンジニアリングの観点で正しい設計判断（DRY 原則）。

high-precision-sensor との棲み分け

- high-precision-sensor: 「今のくらい傾いているか」を知るツール（汎用計測）
- table-level: 「どのボルトを何回転すれば水平になるか」を教えるツール（特定タスク支援）
- 両者は競合ではなく、抽象レベルの異なる相補的なツールである。

批判6への反論：ボルト回転量計算の信頼性について

「計算モデルが単純すぎる」への反論

- 正確に言えば「計算モデルが明快にシンプル」であり、これは現場でのデバッグ・説明のしやすさという設計上の美德。
- 複雑な物理モデルを導入しても、センサー精度の限界（ $\pm 0.1^\circ$ ）がボトルネックになるため過剰設計になる。
- 「剛体でない天板」や「非対称な脚配置」は確かに現実には存在するが、大多数の机は正方・長方形で4点支持であり、初期ターゲットとして合理的。

「計算精度の限界が未提示」への反論

- センサー精度 $\pm 0.1^\circ$ → 奥行き 600mm で $\pm 1\text{mm}$ 以上の誤差が生じる点は誠実な批判。
- ただし、「1mm = ボルト1回転の違い」レベルで精度が必要なユースケース（精密装置の水平出し）は当初スコープ外。一般的な事務机・作業台程度であれば「大体合わせる（ $\pm 2 \sim 3$ 回転の誤差許容）」で十分な精度は確保できる。
- 受け入れるべき点: UI 上での精度限界の明示（「現在の精度ではボルト $\pm X$ 回転程度の誤差を含む」など）は改善として妥当。

批判7への反論：実運用・UX の問題について

「スマホを置くという前提の脆弱性」への反論

- 静止判定（分散監視）は、スマホが滑ったり動いたりした場合に `active` モードに戻るよう設計されており、誤った確定を防ぐメカニズムが機能している。
- 完全な固定を保証するには別途スタンド等のアクセサリが必要だが、これは「アプリの問題」ではなく「使用方法の問題」。

「ドキュメントの薄さ」への反論

- table-level は high-precision-sensor のスピンオフとして開発されており、共通設計は high-precision-sensor 側のドキュメントで賄う構造。SP-01 の薄さは意図的な範囲分離の結果。
 - 受け入れるべき点: アルゴリズム固有部（`calculator.js` の精度評価）と操作マニュアルのドキュメント追加は次フェーズの優先事項として妥当。
-

第3部：両アプリ共通の批判への反論

批判8への反論：第三者検証がない

- 第三者検証は「商用リリース前」の要件であり、プロトタイプ段階では「開発者自身が比較試験を実施・記録した」ことで要件として十分。
- 正確な測定不確かさの評価には校正された標準器が必要であり、個人開発段階でのコスト・設備に鑑みれば現時点での省略は合理的判断。

批判9への反論：セキュリティ・プライバシー

- CSP 未設定は改善余地として認めるが、**両アプリは外部通信を行わず・サーバーを持たず・個人情報を収集しない**。攻撃面が極めて限定的であり、CSP 設定の優先度は実害リスクから見て相対的に低い。
- ただし、プライバシーポリシーの簡易揭示（「センサーデータは端末外に送信されません」）は低コストで追加可能であり、追加すべき。

批判10への反論：アクセシビリティ

- **音声フィードバック（4方向・読み上げ）はまさにアクセシビリティ向上のための設計**であり、視覚に頼らない操作を主要ユースケースとしている。
- aria-label の不足はコードレビューで指摘済みであり、2026-02-15 時点で主要ボタンへの追加が完了している（RV-03 §8 参照）。

批判11への反論：長期メンテナンス

- 個人開発プロジェクトが直面するリスクとして妥当な批判。ただし、MIT License でオープンソース公開している以上、フォーク・引き継ぎ・外注化が可能な状態になっている。
 - DeviceOrientation API の互換性問題については、iOS 13 未満の非互換が既知の課題として文書化されており（SP-04 §Known Issues）、無視されているわけではない。
-

今後の展望：high-precision-sensor / table-level

作成日: 2026-02-25
対象: 両アプリの中長期的な発展シナリオ

はじめに

本書は、現在プロトタイプ段階にある2つのアプリについて、技術・ユーザー体験・ビジネスの3軸で今後の発展方向を整理する。近未来（6ヶ月以内）から長期ビジョン（2～3年）まで時間軸を持って考える。

第1部：high-precision-sensor の今後の展望

短期（～v1.3：3～6ヶ月以内）

A. 運用品質の底上げ（現行ロードマップの完遂）

目的: 実証試験・現場導入で「動かない」「分からない」が起きないレベルへ引き上げる

課題	具体策	期待効果
現場誤操作	キャリブレーションウィザード UI（ステップ固定表示）	手順ミスによる精度低下を排除
状態表示の不明瞭さ	モード表示を運用語に変更（計測中/安定化中/確定値）	非技術者が状態を即座に把握
センサー許可拒否時	復旧ガイド画面（iOS 設定画面へのリンク付き）	現場で担当者に頼らず復旧可能
SW 自動更新	skipWaiting + 更新通知バナー実装	古いキャッシュで動く問題の根絶
アイコン互換性	PNG/maskable アイコン追加（manifest 強化）	ホーム画面への追加品質向上

B. データ品質の強化

- セッションパッケージ出力（CSV + 設定スナップショット + 端末情報を ZIP 化）
 - 再現性の確保と問い合わせ対応の標準化
 - 外注・依頼研究への引き渡しデータとして活用
- 不確かさ表示（信頼度スコアの UI 提示）
 - 「今の値を信じてよいか」を可視化し、計測品質の均質化を支援
- サンプリングレート固定ロギング（10Hz/50Hz の setInterval ベース）
 - バックグラウンド計測問題の部分解消

中期（v2.0：6～18ヶ月）

C. 精度の第三者検証・公表

- デジタルプロトラクターとの比較試験（機種別・温度別）を実施し、測定不確かさを正式評価
- 比較ドキュメントを公開することで、「気泡管水準器の代替として有効」という主張に客観的根拠を付与
- 測定条件（機種・設置姿勢・環境温度）によるブレのマトリクスを作成

D. リアルタイム遠隔化

- WebSocket を用いた親機（PC）との連携
 - スマホを「センサーノード」、PC を「表示端末・管理端末」に分担

- 遠隔地からスマホを設置して水平出しを観察する用途（足場・高所作業など）
- 複数台連携による 3D 計測
 - 2台のスマホを使い、X-Y 平面の傾きに加えてねじれ（Z軸回転）を推定
 - 大型設備の据付精度評価に応用可能

E. プロファイル機能

- 機材別・現場別の設定プリセット保存
- チーム内で設定共有（QR コード読み込み・JSON エクスポート）
- 複数現場にまたがる設備管理への対応

長期ビジョン（v3.0+：1～3年）

F. 据付検収アシスタントへの進化

- 計測→記録→承認（電子署名や写真付き）の検収ワークフローを内包
- 既存の帳票フォーマット（紙）と連携した PDF 生成
- 設備台帳との紐付けと傾斜の経年変化モニタリング

G. クラウドデータ活用（オプション展開）

- オプトイン型でデータをクラウドに集約し、設備の状態変化を遠隔モニタリング
- 同種設備の集計データから「異常な傾き」を早期検知するアノマリー検出
- プライバシーと業務機密の保護を前提とした設計

H. 外注・依頼研究への貢献

- 計測アルゴリズムの教材・研究用データとして活用（「センサー学習アプリ」への派生）
- 建設・製造業向けの受託開発（カスタムプロファイル・特定設備向けチューニング）

技術ロードマップ（コード品質）

フェーズ	主な改善
短期	TypeScript 段階導入（JSDoc → d.ts 生成）、Workbox による SW 自動化
中期	テスト自動化の CI/CD 組み込み、UIManager/AudioEngine の単体テスト整備
長期	マルチアプリ共通基盤（shared/ の npm パッケージ化）、TypeScript 全面移行

第2部：table-level の今後の展望

短期（～v1.1：3～6ヶ月以内）

A. ドキュメント整備と精度の透明化

- アルゴリズム精度評価書の作成
 - センサー誤差とボルト回転量誤差の関係を明示（「機の奥行き 600mm の場合、 $\pm 0.1^\circ$ の誤差は約 $\pm 1\text{mm} \approx \pm 1$ 回転に相当」等）
 - ユーザーが期待値を正しく理解できるように
- ユーザーマニュアル（SP-01 の拡充）
 - 機のサイズ入力方法、スマホの正しい設置位置、ボルト調整手順を写真付きで説明

B. 精度・設置安定性の改善

- スマホ設置状態の確認機能（静止判定中の脱落検出）
 - 計測開始後にスマホが動いた場合、即座に **active** に戻す処理の可視化強化
- 不確かさ表示（high-precision-sensor と共通実装）
 - 「現在の精度での調整限界： $\pm X\text{ mm}$ （ $= \pm Y$ 回転）」をリアルタイム表示
- サポートするボルトピッチの拡張（現行: M6～M12 → 建築用アンカー対応等）

C. UX の改善

- 4隅調整マップのビジュアル強化
 - 「今どの脚を優先すべきか」を色分けやランクで提示
 - 1本調整後に次の目標を自動案内
 - 音声ガイドの強化
 - 「手前左の脚を右に 0.5 回転してください」→「準備ができたらスマホを置いてください」の対話フロー化
 - 機の形状プリセット（正方形・長方形・L字型）
-

中期（v2.0：6～18ヶ月）

D. 計算モデルの精緻化

- 天板の剛体仮定を緩和したモデルの検討
 - 3点支持の場合・脚位置が非対称の場合への対応
 - ユーザーが脚位置を手動入力できる「カスタム配置」モード
- 調整シミュレーター機能
 - 「この脚を N 回転すると、他の脚への影響がどうなるか」を事前プレビュー

E. 多台数管理への拡張

- 案件・台番管理機能
 - 調整した机に番号を付けて記録（台帳エクスポート）
 - チームでの作業進捗管理（「机 3 番：完了」「机 7 番：要再調整」）
- 工場ライン・建設現場での大量調整への対応
 - 10～100 台規模の据付作業を効率化する「バッチ調整モード」

F. high-precision-sensor との連携・統合

- 共通の「センサーコア」モジュールを `shared/` から npm パッケージとして提供
 - table-level の「ボルト回転量計算」を計測アプリの拡張機能として統合
 - ユーザーが用途に応じてモードを切り替えるシームレスな UX
-

長期ビジョン（1～3年）

G. 業務 SaaS への発展

- PWA としての軽量を維持しながら、チームアカウント・案件管理・記録履歴をクラウドバックエンドで提供
- API 連携によって設備管理システム（CMMS）との統合が可能に

H. 精密機械・半導体製造への展開

- より高精度なセンサー（外付け傾斜計）との接続（Bluetooth/USB）
- ppb レベルの精度が求められる用途には「計測スマホ + 外部センサー」ハイブリッド構成で対応

I. 教育・研修用途への展開

- 「水平出し研修」のデジタルツールとしての活用
 - 研修記録（誰が・どの設備を・どの精度で調整したか）を自動記録
-

第3部：両アプリ共通の展望

開発体制の進化

段階	体制
現在	個人開発 + 生成AI コーディングサポート
短期	外部レビュアー（精度検証・UX 評価）との連携
中期	依頼研究・共同開発（大学・研究機関・専門企業）
長期	外注開発チームによる機能拡張・商用化

共通技術基盤の整備

- `shared/js/` の `npm` パッケージ化（カルマンフィルタ・静止判定ロジックをライブラリとして公開）
- TypeScript による型安全な設計への移行
- CI/CD パイプラインの構築（自動テスト・デプロイの標準化）

生成AI活用の今後

- 開発プロセスへの生成AI統合は既の実証済みであり、今後はより**高度なロール**（設計レビュー自動化・テスト自動生成・ドキュメント自動更新）に拡張可能
- 「生成AI × センサー」の組み合わせで、計測データの異常検知・レポート自動生成が現実的な近未来として視野に入る

知財・ライセンス戦略の整備

- MIT License のままコミュニティへ公開し続けるか、商用機のデュアルライセンス化を検討するかを明確化
 - 外注・共同開発時のコード帰属・知財処理方針を事前文書化
-

まとめ：優先度付き アクションリスト

3ヶ月以内にやること（最優先）

1. 両アプリ —— 精度限界を UI 上に明示（「この精度での誤差：約 $\pm XX$ mm」等）
2. high-precision-sensor —— キャリブレーションウィザードと運用語への表示変更
3. table-level —— アルゴリズム精度評価書とユーザーマニュアルの作成

6ヶ月以内にやること（重要）

1. 両アプリ —— 比較試験データの取得・公表（精度の客観的根拠づくり）
2. high-precision-sensor —— セッションパッケージ出力とサンプリングレート固定ロギング
3. table-level —— 不確かさ表示と4隅調整マップの改善

1年以内に判断すること（戦略）

1. 外注・依頼研究の具体的な相手・領域の選定
2. 商用化・SaaS 展開の意思決定
3. TypeScript 移行と npm パッケージ化の開始