Reflection and Traceability Report on FitCoachAR

Seyedmohamad Mirhosseininejad

This document summarizes how feedback received during the development of FitCoachAR (FitCoachAR) was addressed between Revision 0 and the Final Documentation, reflects on the design iterations, and reviews how the project unfolded relative to its original plan. Each feedback item below is linked to the closed GitHub issue that tracked it on the project repository (mmd-mirh/cas741).

1 Changes in Response to Feedback

All feedback was received either through GitHub issues or as annotated PDFs attached to review-request issues. Every item was addressed and every corresponding issue was closed. The tables below record the source, the summary of the feedback, the response, and a pointer to the issue.

1.1 Problem Statement

#	Source	Feedback	Response
1	Dr. Smith (#1)	The reader needs to know what FitCoachAR is and what problem it solves.	Rewrote the opening paragraph to describe the system and the gap it addresses (affordable, hardware-free AR form feedback).
1	Dr. Smith (#1)	CAS 741 is not a real-time systems course; drop the real-time framing.	Removed all language framing the system as a real-time system; kept only the scientific-computing aspects (pose estimation, angle calculation, validity classification).

#	Source	Feedback	Response
1	Dr. Smith (#1)	Opening LaTeX quotes are rendered incorrectly (should use ‘‘).	Corrected all opening quotes in the Problem Statement.
1	Dr. Smith (#1)	A machine-learning report would be a good fit as an extra.	Updated `Repos.csv` to list the Machine Learning Report as one of the two extras for the project.

1.2 SRS

#	Source	Feedback	Response
5	Y. Mei (#5)	Symbols \$, θ_{thresh}, θ_{low} missing from the Table of Symbols.	Added the missing symbols.
6	Y. Mei (#6)	DD1 and DD2 cite different sources for the same underlying fact.	Unified the source of both DDs to *Standard Biomechanics*.
7	Y. Mei (#7)	R1 is not visibly traced to the Camera assumption in Table R_trace.	Updated the SRS Traceability Matrix so R1 is explicitly linked to A_Camera (A2).
8	Y. Mei (#8)	Add explicit vector-subtraction formulas for the joint-angle DDs.	Added the vector forms ($\vec{v}_{femur} = P_{hip} - P_{knee}$, etc.) to GD_KneeAngle and GD_ElbowAngle.
9	Y. Mei (#9)	“Not occluded” in A_Visibility is ambiguous.	Rewrote A_Visibility to require a *direct line of sight* to the target joints, and propagated this phrasing to GD_ElbowAngle during the final pass.
10	Y. Mei (#10)	Missing “Data Definitions” header; leftover template Appendix; misaligned GD tables; inconsistent TM numbering.	Added the header, removed the template Appendix, switched GD tables to `tabularx` for proper wrapping, and fixed the TM references.

#	Source	Feedback	Response
4	Dr. Smith (#4)	Detailed comments on the annotated SRS PDF.	Applied throughout the revision; individual items are tracked in the commits cited by the issue resolutions above.

1.3 VnV Plan

#	Source	Feedback	Response
13	Y. Mei (#13)	Formatting inconsistencies.	Fixed.
14	Y. Mei (#14)	Section 3.1 should list the actual reviewer names, not just roles.	Section 3.1 now names the assigned reviewers explicitly.
15	Y. Mei (#15)	Dataset description is too vague.	Expanded the Dataset section to state that it covers a diverse distribution of correct/incorrect form, varying user heights, and camera angles.
16	Y. Mei (#16)	Traceability matrix is missing.	Added the requirements–tests traceability matrix. It was further corrected during the final pass so that R1–R5 match the R_Input/R_Process/R_Calc/R_Verify/R_Output identifiers used in the SRS.
17	Y. Mei (#17)	Mentioning code reviews under *Unit Testing* is confusing; it is already covered under implementation verification.	Removed the code-review discussion from the unit testing section.
12	Dr. Smith (#12)	Detailed comments on the annotated VnV Plan PDF.	Incorporated throughout the revision, including the restructuring noted above.

1.4 MG and MIS

#	Source	Feedback	Response
19	Y. Mei (#19)	MG is missing a revision history.	Added a full revision-history table (v1.0–v1.3) at the top of the MG.
20	Y. Mei (#20)	Empty list of figures; unusual section numbering (7.0.1 style).	Added real figures (architecture diagram, TikZ Uses-Hierarchy DAG) and fixed the heading hierarchy so numbering is well-formed.
21	Y. Mei (#21)	M4, M5, M8 should not be classified as *Software Decision*; they describe externally visible behaviour.	Partially applied. M4 (Pose Tracking) and M5 (Kinematic Engine) were reclassified as *Behaviour-Hiding* as suggested. M8 (Signal Smoothing) was kept under *Software Decision* because the filtering algorithm (Kalman filter, see AC4) is not in the SRS and is a software-level design choice. Section 6 of the MG was restructured so the subsection headers (Hardware-Hiding / Behaviour-Hiding / Software Decision) match the table.
22	Y. Mei (#22)	Inconsistency in M8 usage: the DAG and the MIS disagreed on the direction of the M4/M8 dependency.	Corrected the DAG so it shows M4 *uses* M8; updated the M8 MIS so its *Uses* field reads `None` (M8 is a stateless filter).
23	Y. Mei (#23)	M1, M2, M7 had no MIS sections.	Added full MIS sections for M1 (`get_frame`), M2 (`draw_overlay`, `display`) and M7 (`update_ui`, `get_json`), each with module identifier, uses, syntax and semantics.

#	Source	Feedback	Response
24	Y. Mei (#24)	Input frames are mis-listed as environment variables for M3 and M4.	Reclassified frame inputs as access-program *parameters* (`frame: bytes`). Environment variables are now reserved for truly external, persistent state (webcam hardware, browser DOM).
25	Dr. Smith (#25)	Detailed comments on the annotated MG and MIS PDFs.	Applied throughout the revision; residual issues surfaced during the final pass (requirement-to-module trace table, M8 Uses) were also fixed.

1.5 VnV Report

#	Source	Feedback	Response
26	Y. Mei (#26)	Peer reviewer was unable to schedule a VnV Report review in time.	Acknowledged; the report will be reviewed later by the instructor (see #27).
27	Dr. Smith (#27)	Will review the VnV Report when the full project is complete.	The report reflects the executed test plan, including the consolidation of T3 into T2 and the decision not to implement the Robot-Framework E2E suite originally labelled T9.

2 Extras

Two extras were pursued for this project: (i) a **Machine Learning Report**, replacing an earlier extra at Dr. Smith's suggestion (issue #1), and (ii) a **User Manual** describing how to install, calibrate, and use the system. Both extras were approved as part of the Problem Statement and are listed in `Repos.csv`.

3 Design Iteration (LO11)

The final design is the result of three substantial iterations.

Rev 0. The first architecture treated FitCoachAR as a monolithic browser-side pipeline in which MediaPipe, the kinematic computation, and the state machine all lived in the same frontend process. This was convenient for the Proof-of-Concept but made the scientific logic (angle calculation, form validity) hard to test in isolation.

Rev 1 (MG/MIS presentation). After the modular-design lectures and Dr. Smith's feedback, the system was decomposed into eight modules along information-hiding lines, splitting the browser (M1, M2, M7) from the Python backend (M3–M6, M8). This made it possible to unit-test M5 (kinematic engine) and M6 (state machine) in pytest without a camera in the loop, and directly addressed the review comment that the scientific computing *secrets* were not being surfaced.

Rev 1.x (Final pass). During peer review Yibing pointed out that the "behavioural" modules (M4, M5, M8) had been mis-filed under *Software Decision*. The hierarchy was corrected to reflect the fact that pose tracking, angle computation, and smoothing are all required externally visible behaviours, not hidden implementation choices. A late internal audit also caught a stale requirement-to-module traceability table in the MG, which was re-derived from the SRS R_Input…R_Output definitions. These changes did not alter the code; they aligned the documentation with what the system actually does.

In all three iterations the same underlying need drove the design: make the scientific-computing core testable and auditable, rather than hiding it behind the UI.

4 Design Decisions (LO12)

MediaPipe as the pose backend (M4). MediaPipe was chosen because it runs on commodity webcams without depth hardware, satisfying the "affordable, no specialized hardware" constraint from the Problem Statement. The decision is recorded as anticipated change `acMLModel` in the MG so that a future replacement (e.g. OpenPose, BlazePose-GHUM) would only touch M4.

Explicit Kalman smoothing as a separate module (M8). MediaPipe's per-frame landmarks are noisy enough that the raw signal produces unstable angle estimates and spurious state-machine transitions. A stateless smoothing filter was introduced as its own module so the choice of filter (currently a 1-D Kalman per coordinate) can change without affecting the kinematic engine. Its *Uses* set is `None` for the same reason.

State-machine based rep counting (M6). A threshold-hysteresis FSM was preferred over an ML classifier because the squat and bicep-curl rep definitions are already expressible in closed form, and an FSM gives reviewers a clearly verifiable specification (see IM_RepetitionCount). Assumption A_Visibility and the "direct line of sight" constraint followed directly from this decision: the FSM has no recovery path for missing landmarks, so the assumption is an explicit limitation, not an accident.

Separation of the Python backend from the browser front-end.

This was influenced by the testability constraint. By pushing M3–M6 and M8 into a Python process, the scientific computing modules can be exercised by pytest fixtures without any camera or browser, which is what T1, T2, T4–T8 actually do.

Limitations that shaped these decisions were: the inability to assume any depth sensor, the absence of a training dataset large enough to train a classifier of our own, and the course's emphasis on scientific-computing verifiability over end-to-end system testing.

5 Reflection on Project Management (LO24)

5.1 How Project Management Compared to the Development Plan

The project largely followed the planned workflow: weekly self-imposed milestones aligned with the CAS 741 deliverable schedule (Problem Statement → SRS → VnV Plan → MG/MIS → Implementation → VnV Report → Final Documentation). GitHub issues were used both for peer review and for tracking Dr. Smith's feedback, and each issue was closed with a commit reference, which made the final traceability exercise (this document) substantially easier.

The technology choices (Python + FastAPI backend, MediaPipe for pose, browser-side three.js front-end, LaTeX documentation, pytest) stayed consistent from the Problem Statement onward.

5.2 What Went Well

- **Issue-driven feedback.** Every review comment from Yibing and Dr. Smith became a GitHub issue, which was closed by a commit that named the change. This made the Revision 0 → Final Documentation diff trivial to reconstruct.
- **Early modular split.** Moving the scientific core into its own Python backend paid off repeatedly: it made it possible to unit-test the kinematic engine and state machine without a camera.
- **Reviewing as a source of design improvements.** Several issues (#21, #22, #24) were not just formatting fixes; they exposed real inconsistencies between my mental model and what was documented. The reviews were the single biggest source of design-level improvements.

5.3 What Went Wrong

- **Documentation drift.** Between Rev 0 and the final pass, several documents (most visibly the MG traceability table and the VnV Plan requirement numbering) drifted out of sync with the SRS. The drift was only

caught by a deliberate end-of-project audit; the regular review cycles had not checked cross-document consistency.

- **Late unit testing.** Unit tests were written close to the implementation deadline rather than in parallel with each module. This meant bugs surfaced at the end when there was little time to redesign.
- **VnV Report scheduling.** The VnV Report was finished too close to the peer-review window, which made it impossible for the peer reviewer to return feedback in time (issue #26).

5.4 What Would I Do Differently

- Run a *cross-document consistency check* at every revision boundary, not just at the end. A short checklist (requirement IDs, module IDs, test IDs, assumption labels) would have caught the late drift much earlier.
- Write the unit tests alongside each module, not after. This would have made the VnV Report's code-coverage section much stronger.
- Budget the VnV Report the same amount of calendar time as the SRS or MG, so that peer review is actually feasible.