AI의 실수를 프롬프트로 바꿨다

반복 실패에서 만든 AI 장기 협업 운영체계 - 제출 증빙자료

제출 구성
이 PDF는 증빙 해설서입니다. 실제 원본 화면은 아래의 JPG 4개를 별도 첨부합니다. PDF 1개 + 원본 JPG 4개 = 총 5개 첨부로 구성합니다.

핵심 활용 구조
AI 출력 → 인간 직접 수정 → AI에게 차이 분석 → 반복 오류 패턴 추출 → 제어 명령 생성 → 다음 작업에 재적용 → 회귀 검증

본 사례는 ChatGPT 모델의 내부 파라미터를 재학습한 사례가 아닙니다. 사용자가 반복 오류를 관찰하고 작업 단위·실행 권한·검증 절차를 외부 규칙으로 설계해 장기 협업을 안정화한 실제 활용 사례입니다.

첨부 번호	파일명	역할
1	AI활용사례_증빙자료_제출확정.pdf	증거 해설·전체 구조
2	1000012777.jpg	실제 실패: 최신 명령 편향·검토/수정 혼동
3	1000012779.jpg	반복 오류를 4개 유형으로 분류
4	1000012783.jpg	누적 명령 확인·발화 판정·실행 권한·역검수의 작업 순서
5	1000012791.jpg	회귀 검증·최종 안전 원칙

1. 실제 실패가 먼저 있었다

증거 A | 최신 명령만 따라가고, 검토를 수정으로 오판

별도 첨부 원본 첨부 2 - 1000012777.jpg

원본에서 직접 확인되는 내용
원본 화면에는 모든 조건이 계속 유효한 상황에서도 AI가 최신 지시 중심으로 처리한 문제, 사용자의 “검토해”를 “수정해”로 바꿔 해석한 문제, 원문을 새 대사·설정으로 다시 쓴 문제가 기록되어 있습니다. 사용자 명령 “앞에서 한 명령을 모두 기억하고 적용해.”도 확인됩니다.

오류 → 제어 명령
단순한 사용 불만이 아니라, 장기 작업에서 재발하는 명령 누락과 발화 오분류가 실제 대화 기록으로 존재했음을 증명합니다.

증명하는 것
최신 지시 중심 처리 → 취소·완료되지 않은 이전 명령은 계속 유효
검토를 수정으로 오판 → 분석·검토·질문과 실행·수정·출력을 분리

이 실패 이후 사용자는 복합 명령을 한꺼번에 처리시키는 대신, DOS 프로그램처럼 한 가지 명령 → 결과 확인 → 다음 명령으로 작업 단위를 줄였습니다. 목적은 AI를 내부적으로 재학습시키는 것이 아니라, 한 단계에서 AI가 판단해야 할 선택지를 줄여 오류 위치를 추적 가능하게 만드는 것이었습니다.

2. 반복 실수를 이름 붙이고 명령어로 바꿨다

증거 B | 오류를 4개 유형으로 분류

별도 첨부 원본 첨부 3 - 1000012779.jpg

원본에서 직접 확인되는 내용
원본 화면에는 “문제는 단순한 기억 부족이 아니었습니다.”라는 문장과 함께 ① 직접 명령 편향 ② 발화 유형 오분류 ③ 자동 창작 ④ 출력 권한 오류가 명시되어 있습니다.

오류 → 제어 명령
그때그때 다시 쓰게 하는 수준을 넘어, 실패를 관찰 데이터로 남기고 재사용 가능한 제어 규칙으로 전환했음을 증명합니다.

증명하는 것
오류 발생 → 반복 여부 확인 → 공통 패턴 추출 → 오류 유형 명명 → 방지 명령 작성 → 다음 작업에 재적용

오류 장부의 역할
“AI가 또 틀렸다”로 끝내지 않고, 같은 종류의 실패가 재발하면 원인을 패턴으로 묶었습니다. 이때부터 실패 자체가 다음 프롬프트의 재료가 됐습니다. 사용자가 직접 수정한 결과와 AI 결과를 다시 비교시켜 설명 과잉, 임의 추가, 지시 누락 등 반복 차이를 언어화하고, 그 차이를 새로운 명령으로 고정했습니다.

3. 개별 명령이 고정 작업 순서로 발전했다

증거 C | 실행 전 확인 절차를 고정

별도 첨부 원본 첨부 4 - 1000012783.jpg

원본에서 직접 확인되는 내용
원본 화면에는 “작업 순서도 고정했습니다.”와 함께 정보 확인 → 누적 명령 확인 → 사용자 발화 판정 → 실행 권한 확인 → 장면 구조 확인 → 요청 범위만 실행 → 결과 역검수가 표시되어 있습니다. “너는 자유 창작자가 아니다.”라는 핵심 프롬프트도 확인됩니다.

오류 → 제어 명령
초기의 개별 오류 대응 명령들이 하나의 고정된 실행 순서와 권한 관리 구조로 통합되었음을 보여줍니다.

증명하는 것
복합 지시 누락 → 실행 전 누적 명령 확인
발화 오분류 → 사용자 발화 판정
임의 수정·창작 → 실행 권한과 요청 범위 확인

이 구조에서는 AI가 자연스럽다고 판단해 작업 범위를 넓히지 않습니다. 사용자가 승인한 기준·잠금·정본을 먼저 확인하고, 명시적 실행 지시가 있을 때만 해당 범위를 수정합니다. 정확성이 중요할수록 한 번에 한 항목씩 처리합니다.

4. 수정과 검증을 분리했다

증거 D | 출력 직전 다시 확인하는 역검수

별도 첨부 원본 첨부 5 - 1000012791.jpg

원본에서 직접 확인되는 내용
원본 화면에는 “결과를 역검수한다”, “출력 직전 확인한다”가 보이며, 이전 명령·최신 정보·설정·사용자 요청·작업 범위·인과관계·의도하지 않은 변경을 다시 확인하는 체크가 기록되어 있습니다. 최종 원칙으로 “AI의 기억력과 자율성을 과신하지 않는다.”가 명시되어 있습니다.

오류 → 제어 명령
생성·수정 직후 AI의 자기평가만으로 완료를 판정하지 않고, 별도의 검증 단계를 두어 인간의 승인권을 유지했음을 증명합니다.

증명하는 것
국소 수정 성공 후 다른 조건 재발 → 수정과 검증 분리 + 이전 조건 회귀 검증
성급한 완료 선언 → 확인한 것만 검증 완료, 나머지는 미확인

초기와 현재의 차이
초기: 사용자 명령 → AI 출력 → 오류 → 다시 설명 → 재작성
현재: 누적 조건 확인 → 발화 유형 판정 → 실행 범위 확인 → 개별 실행 → 결과 검증 → 회귀 검증 → 사용자 승인 → 새 오류 발생 시 규칙 갱신

5. 재현 가능한 활용법

이 사례의 목적은 특정 사용자의 긴 프롬프트를 그대로 복사하게 하는 것이 아닙니다. 다른 사용자도 아래의 작은 루프부터 시작할 수 있습니다.

단계	실행
1	AI에게 한 번에 하나만 시킨다.
2	결과가 마음에 들지 않으면 직접 한 번 수정해 본다.
3	AI의 답과 내가 고친 답을 다시 AI에게 비교시킨다.
4	같은 실수가 반복되면 그 실수에 이름을 붙인다.
5	그 실수를 막는 한 줄짜리 명령을 만든다.
6	다음 작업에 적용하고 마지막에는 이전 조건까지 다시 검수한다.

최종 원칙

AI를 잘 사용하는 능력은 좋은 답을 한 번 받아내는 능력만이 아닙니다. AI가 틀렸을 때 실패를 관찰하고, 왜 틀렸는지 패턴을 찾고, 다음에는 같은 실수를 줄일 수 있도록 작업 구조를 바꾸는 능력입니다.

본 증빙 해설서의 주장과 별도 첨부 원본 4개가 서로 대응하도록 구성했습니다. 개인정보·비공개 소설 원문은 심사에 필요한 범위만 사용합니다.