장편 영화 시나리오 번역 파이프라인

Jake Tae-moon Jo · 개발 리드 · https://jake-jo-portfolio.vercel.app/ko/work/screenplay-translation/

AI 에이전트 11개 역할이 번역하고 판정하는 시나리오 번역 파이프라인을 설계해, 장편 네 편을 두 방향으로 다섯 번 끝까지 번역했습니다.

장편 4편 5회 번역
두 방향, 전편을 끝까지

하루
장편 한 편 전 공정

83%
감사관 백테스트 재현율, 세 번째 시도

0건
환각, 다섯 번 모두 (자체 관문)

결과

장편 네 편을 두 방향으로 다섯 번 끝까지 번역했습니다.

제작사의 수정 요청은 모두 24시간 안에 반영했습니다.

장편 네 편을 두 방향으로 다섯 번 끝까지 번역했습니다. 베트남어 → 한국어가 네 번(작품 A의 두 판본, 작품 B, 작품 C), 한국어 → 베트남어가 한 번(작품 D)입니다. 산출물은 업계 조판의 대본 PDF, 편집용 DOCX, 손실 목록이 맨 앞에 오는 보고서입니다.

빈 폴더에서 시작한 날 밤 첫 장편 전편(95쪽)을 끝냈습니다. 작품 C와 D는 전 공정을 각각 하루에 끝냈습니다(파일 시각 기준, 작품 C는 사용량 한도로 약 1시간 멈춘 시간 포함). 여러 모델을 함께 쓰면 보완할 수 있지만, 지금은 비용을 줄이려고 적용하지 않았습니다.

다섯 번 모두 환각 0건이었습니다(파이프라인 자체 관문 기준).

첫 한국어 → 베트남어 작품(작품 D)도 새 역할을 더하지 않고 같은 역할 정의로 처리했습니다. 영어는 한 시퀀스 시험과 외부 평가의 대사 번역까지만 했고, 영어 전편 번역은 없습니다.

기간
2026.09 – 2026.10

역할
사람은 본인 1명 · AI 에이전트 팀 설계·운영

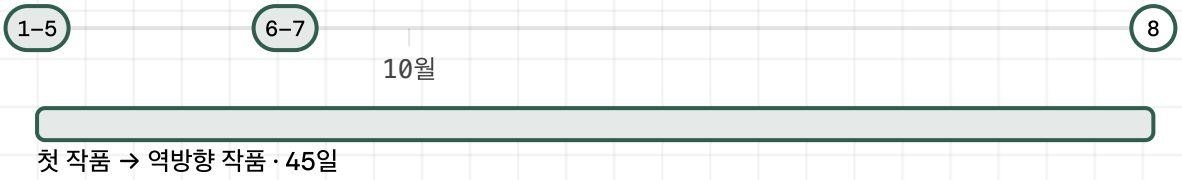
팀
본인 1명 + AI 에이전트 11개 역할

플랫폼
Claude Code 안에서 작업자가 돌리는 하네스(웹 서비스 아님). 산출은 대본 PDF·DOCX·보고서

배경
영화 제작사. 원본 대본, 이전 판본, 사람이 만든 한국어 번역본을 받아 시작

첫 작품에서 역방향 작품까지

장편 네 편을 두 방향으로 다섯 번 끝까지 번역했다.



첫 작품 → 역방향 작품 · 45일

1	2026년 9월 16일	**첫 커밋 묶음** 설계, 에이전트 11개, 오케스트레이터, 인테이크 도구, 백테스트
2	2026년 9월 16일	**감사관 백테스트 합격** 변이 12개 중 10개(83%), 확정 오탐 0, 세 번째 시도
3	2026년 9월 16일	**병렬 구간 번역 폐기** 한 언어 안에서는 순차 이어달리기(본인 판정)
4	2026년 9월 16일	**작품 A(6.5판) 한국어 전편** 95쪽. 빈 폴더를 만든 그날 밤
5	2026년 9월 17일	**작품 A(7.0판) 한국어 전편** 94쪽. 새 발견 0건이던 단계를 뺐다
6	2026년 9월 26일	**작품 B 한국어 전편** 86쪽
7	2026년 10월 1일	**작품 C 한국어 전편** 89쪽, 하루
8	2026년 10월 31일	**작품 D 베트남어 전편** 103쪽, 첫 한국어 → 베트남어, 하루

문제와 배경

「직역하지 말고 다른 언어에서도 같은 문맥과 감정을 지키라」는 요구를, 잴 수 있는 검사로 바꾸는 일이었습니다.

고객은 영화 제작사입니다. 장편 시나리오를 다른 언어로 옮기되, 직역하지 말고 다른 언어에서도 같은 문맥과 감정을 지켜 달라는 요구였습니다. 슬픈 장면이 덜 슬퍼지거나, 농담이 직역으로 죽으면 안 됐습니다.

목표 언어는 베트남어·한국어·영어 셋이었습니다. 전편까지 번역한 것은 베트남어 → 한국어와 한국어 → 베트남어 두 방향이고, 영어는 한 시퀀스(4씬) 시험과 외부 평가의 대사 번역까지만 했습니다.

설계의 출발점은 이 요구가 계측 없이는 판정할 수 없다는 것이었습니다. 그래서 번역하기 전에 지켜야 할 것(비트별 감정 강도, 유머 지점, 복선과 회수)을 수치와 목록으로 고정하고, 번역본만 보고 그것이 살아남았는지 거꾸로 검증하기로 했습니다.

시작 자산은 작품 A의 원본, 사람이 만든 한국어 번역본(이하 기준본), 아직 번역하지 않은 최신 판본이었습니다. 하네스가 해내야 할 일은 셋이었습니다. 판본 사이의 수정 표시를 지켜 다음 판본에서 바뀐 씬만 골라 번역할 수 있어야 했고, 원어의 두 개념을 하나로 합칠 때는 그 손실을 알려야 했으며, 대본 포맷을 지켜야 했습니다.

만든 것은 웹 서비스가 아니라, 작업자가 Claude Code 안에서 돌리는 번역 하네스입니다. 웹 서비스는 「번역 품질 자체가 훌륭하다고 판단된 시점에 다시 검토」하기로 미뤘습니다.

제 역할

에이전트 11개 역할과 관문은 제가 설계해 돌렸고, 방향을 바꾼 세 결정도 제가 내렸으며, 작품마다의 시장·용도·표기는 제작사 결정으로 따로 기록했습니다.

만드는 쪽의 사람은 저 한 명이었습니다. git 작성자는 저 한 사람이고, 커밋 148개 모두에 AI 공동 작성 표시가 있습니다. 손으로 쓴 부분과 에이전트가 쓴 부분을 나눈 기록은 없어서, 역할을 「사람은 본인 1명 · AI 에이전트 팀 설계·운영」으로 적었습니다.

설계한 것은 에이전트 11개 역할, 그 역할을 단계별로 부르는 오케스트레이터 스킬, 철칙 9개, 도구 26개(번역 하네스 17, 외부 평가 9)입니다. 역할은 세 계층입니다. 원작 해부(AI 감독·구조 작가·디테일 작가·호칭 담당), 전이(번역가·문화 자문·연기 감수), 판정(감사관·포맷 관리자·AI 관객·AI 투자사)입니다. 9개는 Opus 계열에, 2개는 Sonnet에 배정했습니다. 오케스트레이터는 스킬입니다. 메인 세션이 단계를 돌리며 정의 파일을 지정해 에이전트를 부릅니다.

호출 기록은 9월 26일 이후 세션에만 남아 있습니다. 작품 C 세션에서는 에이전트를 33번, 작품 D 세션에서는 51번 불렀습니다. 작품 D의 51번 중 47번은 역할 정의 10개를 지정한 호출이고(AI 감독·번역가·감사관 각 8번 등), 4번은 정의 없이 불렀습니다. 포맷 관리자는 남은 기록에 호출이 없습니다. 9월 17일 단계를 줄일 때 포맷 에이전트 단계를 뺐습니다.

방향과 기준을 바꾼 아래 세 결정은 세션 기록에 「사용자」 결정으로 남아 있고, 그 세션을 돌린 사람은 저입니다. 9월 16일에는 병렬로 번역한 구간들이 서로 어긋난 점을 짚어 순차 번역으로 다시 하게 했습니다. 같은 날 효과가 확인된 개선은 그 작품에서 끝내지 말고 파이프라인 전체에 반영한다는 원칙을 정했습니다. 9월 17일에는 「너무 오래 걸리는데」라며 새 발견이 0건이던 단계를 빼게 했습니다. 작품마다 시장·용도·표기는

사람에게 묻게 했습니다. 답이 없는 항목은 직전 작품의 결정을 기본값으로 쓰되, 확인 대기로 표시해 보고서의 「사용자(제작사) 확인」 절에 올립니다. 남은 확인 대기는 작품 B 4건, C 3건, D 8건입니다.

사람의 결정은 AI 감독보다 위에 있습니다. 우선순위는 「사용자 결정 > 감독 잠금 > 렉시콘」이고, 감독은 사용자 결정을 다시 판정하지 않습니다.

원본에서 최종 산출까지

원본은 도구가 정제하고 에이전트가 해부해 씬 계약으로 굳힌 뒤, 새 번역 주자가 구간마다 이어 달리며 관문을 통과해야 다음 구간으로 간다.

① **인테이크** □시스템
워터마크 정제, 성조 부호 검사, 구조 파싱. 하나라도 실패하면 멈춘다

② **원작 해부** ◇에이전트
구조·디테일·호칭 담당이 병렬로 구조 맵, 감정 강도, 렉시콘, 원장을 만든다

③ **씬 계약과 잠금** ◇에이전트
AI 감독이 병합하고 모든 구간이 똑같이 써야 할 문자열을 번역 전에 잠근다

④ **구간 번역** ◇에이전트
구간마다 새 주자, 자기 구간의 원문·패킷과 앞 구간의 확정 대본만 받는다

⑤ **구간 관문** □시스템
씬 누락·인계 누출·따옴표, 환각, 잠금을 검사한다

⑥ **조립 뒤 판정** ◇에이전트
감사관이 번역본만 보고 씬 계약을 역추론해 원본과 대조, AI 관객·투자 시뮬레이션

⑦ **사람의 결정** ○사람
시장·용도·표기, 정하지 않은 항목은 보고서의 확인 대기로

⑧ **최종 산출** □시스템
대본 PDF, 편집용 DOCX, 손실 목록이 맨 앞에 오는 보고서

되돌아가는 길
↺ ⑤ 구간 관문 → ④ 구간 번역: 막히면 그 구간 재번역

누가 무엇을 정했나

방향과 기준은 사람이 정했고, 에이전트 사이 충돌은 AI 감독이 판정했으며, 기계 관문은 도구의 종료 코드로 막았다.

사람

병렬 구간 번역 폐기 → 순차 이어달리기(9월 16일)
효과가 확인된 개선은 파이프라인 전체에(9월 16일)
새 발견 0건이던 단계 제거(9월 17일)
작품마다 시장·용도·표기 결정. 정하지 않은 항목은 보고서의 확인 대기로

AI 에이전트 11개 역할

원작 해부 4 · 전이 3 · 판정 4(Opus 계열 9, Sonnet 2)
AI 감독이 에이전트 사이 충돌을 최종 판정하고, 사람의 결정은 다시 판정하지 않는다
AI 관객 12명·AI 투자 프로파일 5종은 AI 시뮬레이션이고, 보고서에도 그렇게 적는다
남은 세션 기록: 작품 C 호출 33회, 작품 D 호출 51회(11개 중 10개 역할)
커밋 148개 전부에 AI 공동 작성 표시

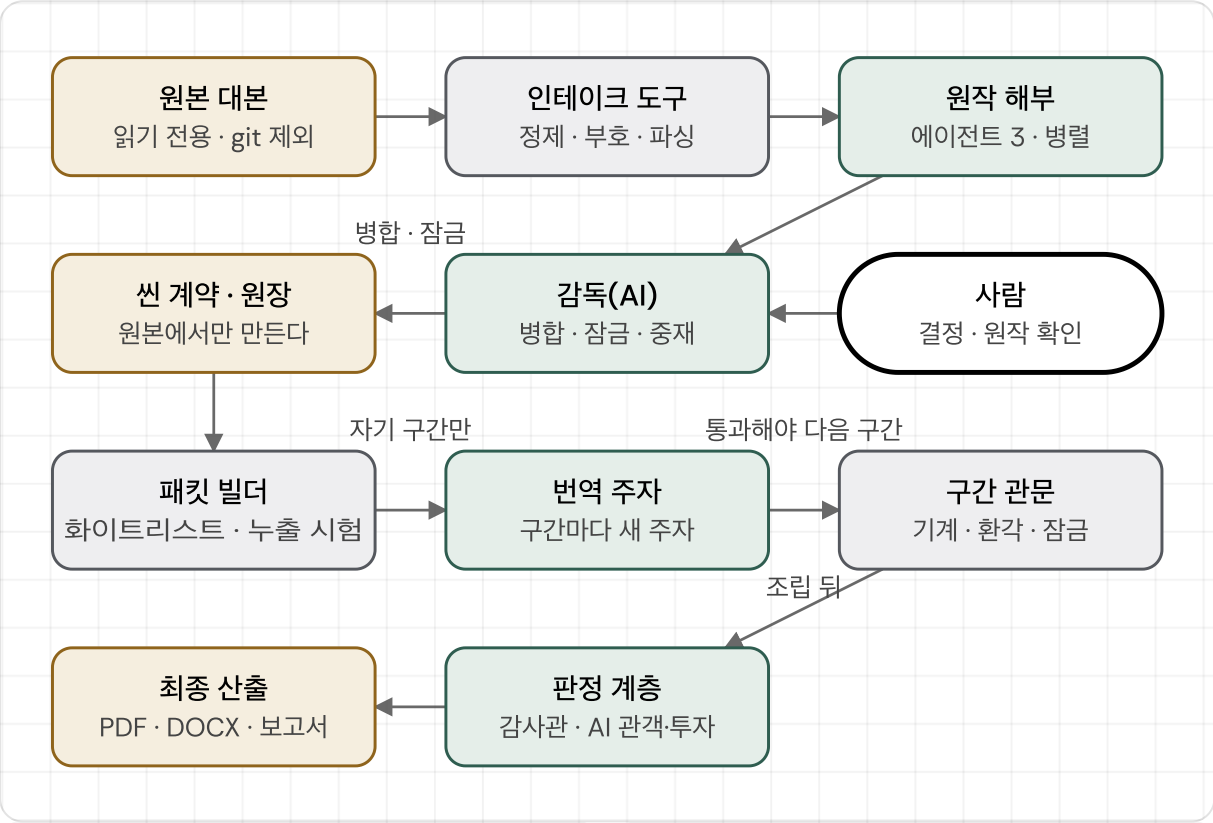
기계 관문

원본의 성조 부호 검사가 실패하면 중단
구간 관문: 씬 누락·인계 누출·대사 따옴표 차단, 관문 자체를 가짜 결함으로 시험
환각 검사가 하나라도 어긋나면 자동 반려(감독 중재 없음)
CI·PR·훅은 없다

시스템 구조

파이프라인 구성

번역 주자는 자기 구간의 원문과 패킷, 앞 구간의 확정 대본과 인계만 보고 순서대로 이어 달리며, 관문을 통과해야 다음 주자가 출발한다.



원본 대본

원본 대본(PDF·DOCX)은 읽기 전용이고 git에 넣지 않는다.

인테이크 도구

도구. 워터마크 정제, 베트남어 성조 부호 무결성 검사(밀도·종류·깨진 글자), 구조 파싱. 하나라도 실패하면 멈춘다. 작품 A 원본에서 워터마크 오염 194곳을 찾아 지웠다.

원작 해부

에이전트. 구조 작가·디테일 작가·호칭 담당이 병렬로 원작을 해부한다. 구조 맵, 비트별 감정 강도(1~10)와 유머 지점, 호칭·용어 렉시콘, 상태 원장을 만든다.

씬 계약 · 원장

씬마다 극적 기능·감정 궤적·서브텍스트·복선과 회수·살릴 대사를 JSON으로 고정한다. 원본에서만 만들고, 번역 산출은 입력으로 쓰지 않는다. 상태 원장(누가 무엇을 아는가 등)도 원본으로 미리 다 만들어 번역가는 읽기만 한다.

감독(AI)

에이전트(AI)다. 해부 결과를 씬 계약으로 합치고, 모든 구간이 똑같이 써야 할 문자열을 번역 전에 잠그며, 감사·시사 결과를 한 번에 중재한다. 사람의 결정은 다시 판정하지 않는다.

사람

사람이 시장·용도·배급·표기를 정한다. 사람의 결정은 AI 감독의 판정 위에 있다. 원작 차원의 문제는 작가에게 확인을 요청한다.

패킷 빌더

도구. 화이트리스트 1~5층만 담는다. 「이 패킷에 없는 정보는 존재하지 않는다. 발명하지 말고 신고하라.」 뒷내용이 새지 않는지는 답을 모르는 에이전트에게 반전을 추리하게 해 시험한다.

번역 주자

에이전트. 한 언어 안의 구간은 순서대로 번역하고, 구간마다 새 주자를 띄운다. 자기 구간의 원문과 패킷, 앞 구간의 확정 대본과 인계만 받고 앞 주자의 노트는 받지 않는다. 언어 사이는 병렬이다.

구간 관문

도구와 감사관. ① 기계 관문(씬 누락·인계 누출·따옴표 차단) ② 환각 관문(화자별 대사 블록·고유명사·수치·원장) ③ 감독 잠금 개정과 소급 치환. 막히면 그 구간을 다시 번역한다. 관문은 스스로 가짜 결함을 심어 시험한다.

최종 산출

Fountain 대본에서 업계 조판 PDF와 편집용 DOCX를 만들고, 손실 목록이 맨 앞에 오는 보고서를 낸다.

판정 계층

에이전트. 감사관이 번역본만 보고 씬 계약을 역추론한 뒤 원본 계약과 대조하고, 감정 강도 차가 3 이상이면 실패다. AI 관객 4명과 AI 투자 프로파일 5종이 번역본 사본만 읽는다. 실제 관객 조사가 아니다. 감독이 결과를 한 번에 판정하고, 보정은 1회다.

backtest/relay_gate.py · 관문이 스스로를 시험한다. 대본 사본에 따옴표 대사·씬 삭제·잠금 문자열 변조를 심고, 하나라도 놓치면 관문 결과를 믿지 않는다. (설명과 이름은 영어로 옮김, 출력·기록 줄은 생략, 씬 삭제 정규식은 drop_scene으로 묶음)

```
def mutate(group):
    """Plant violations in a copy of the script; all must be caught."""
    real = bp.RUN
    base = (real / f"ko_{group}.fountain").read_text(encoding="utf-8")
    ok0 = gate(group)
    start, end = bp.GROUPS[group]
    mid = (start + end) // 2
    present = [a for a in dict.fromkeys(CHECKED)
               if len(a) >= 4 and base.count(a) == 1]
    cases = {
        "dialogue quotes": base.replace("\n", '\n  "fake line"\n', 1),
        f"scene {mid} deleted": drop_scene(base, mid),
    }
    if present:
        s = present[0]
        cases["locked string altered"] = base.replace(s, s[::-1])
    tmp = pathlib.Path(tempfile.mkdtemp())  # … plus the hand-off file
    misses = 0
    for name, t in cases.items():
        if t == base: misses += 1; continue  # mutation did not apply
        (tmp / f"ko_{group}.fountain").write_text(t, encoding="utf-8")
        bp.RUN = tmp
        with contextlib.redirect_stdout(io.StringIO()) as buf:
            ok = gate(group)
        caught = (not ok) or (WARN in buf.getvalue())
        misses += not caught
    bp.RUN = real
    return ok0 and not misses
```

핵심 결정

여섯 결정 중 구간을 차례로 잇는 것과 새 발견이 없던 단계를 빼는 것은 제가 짚어 시작했고, 나머지 넷은 에이전트의 추천 기본값을 구조와 검사로 정한 것입니다.

장편의 문맥은 폐쇄형 패킷과, 원본으로 미리 만든 원장으로 에이전트 기본값 · 확정 기록 없음

문제

장편을 씬 단위로 나눠 번역하면, 번역가는 비어 있는 정보를 지어내 채웁니다. 대본 전체를 주면 한 씬에 집중하지 못합니다. 번역가에게 무엇을 얼마나 보여 줄지 정해야 했습니다.

검토한 선택지

씬만 따로 준다: 정보 공백에서 환각이 생긴다(「환각은 불확실성 아래의 공백 메우기」)

대본 전체를 준다: 주의가 흐려지고, 다른 씬의 디테일이 섞이고, 비용이 폭증한다

원장을 번역하면서 쌓는다: 앞 씬의 오역이 원장을 오염시키고, 뒤 씬이 모두 그 위에 쌓인다

✓ 폐쇄형 패킷 + 원본으로 미리 만든 읽기 전용 원장

결정

번역가에게는 수명이 다른 네 층의 정보만 담은 패킷을 줍니다. 작품 내내 변하지 않는 것, 앞으로만 늘어나는 상태, 그 씬에만 필요한 것, 앞뒤 한 씬의 비트 요약입니다. 원장(누가 무엇을 아는가, 호칭 상태, 등장한 소품, 죽은 인물)은 번역 전에 원본으로 전 씬을 다 만들고, 번역가는 읽기만 합니다. 패킷에는 탈출구를 하나 두었습니다. 「이 패킷에 없는 정보는 존재하지 않는다. 발명하지 말고 신고하라」입니다.

결과

패킷 빌더는 담아도 되는 것만 고르는 화이트리스트 1~5층으로 바뀌었습니다. 신고가 0건이면 오히려 의심하는 규칙도 더했습니다. 패킷이 뒷내용을 흘리는지는 답을 모르는 에이전트에게 반전을 추리하게 해 시험합니다(품질 섹션).

대가·남은 위험

반전이 없는 작품에서는 누출 필터가 꺼져 뒷내용이 샜습니다(10월 1일, 10월 31일에 다시). 누출 검사는 패킷을 만드는 규칙과 같은 규칙이라, 기록에도 「필요조건일 뿐」이라고 적었습니다.

구간 번역은 병렬로 돌려 본 뒤 순차 이어달리기로 본인 입력

문제

장편 한 편(작품 A, 91씬)은 한 번에 번역하기에 깁니다. 처음에는 다섯 구간을 동시에 번역했습니다.

검토한 선택지

한 언어 안의 구간을 동시에 번역한다(실제로 돌려 본 첫 회차)

✓ 한 언어 안의 구간은 순서대로, 언어 사이는 병렬로

결정

동시 번역은 장편을 깼습니다. 반복해서 나오는 가사를 두 구간이 각자 지었고, 세 번 반복한 뒤 결말에서 회수되는 대사를 결말 구간이 대조할 수 없었습니다. 한 호칭이 네 가지로 섞였고, 강조 표기는 세 방식이 됐습니다. 기록에는 「심는 사람과 거두는 사람이 서로 모른 채 동시에 썼다」고 남았습니다. 제가 이 점을 짚어 순차로 바꾸게 했습니다. 구간마다 새 번역 주자를 띄우고, 앞 구간의 확정 대본과 인계만 넘깁니다. 이유는 「순차는 문맥과 누출 둘 다에 더 강하다. 앞 대본에는 뒤 내용이 없다」입니다.

결과

함께 바꾼 것이 셋입니다. 모든 구간이 똑같이 써야 할 문자열(호칭·반복 대사 등)을 번역 전에 잠갔고, 잠금은 25개에서 7차 개정 때 60개가 됐습니다. 패킷은 블랙리스트에서 화이트리스트로 바꿨습니다. 구간 사이에는 관문 세 겹을 세웠습니다. 순차로 다시 번역해 그날 밤 작품 A 6.5판 한국어 전편(95쪽)을 끝냈습니다.

대가·남은 위험

앞 구간을 기다려야 해서 시간이 늘었습니다. 7.0판에서는 구간당 1~2시간이 걸려 「한 회차가 너무 길었다」고 기록했고, 이것이 단계 축소로 이어졌습니다(결정 5).

환각은 검사할 수 있는 사실로 보고, 기계 검사로 자동 반려 에이전트 기본값 · 확정 기록 없음

문제

번역가가 원본에 없는 인물·소품·수치를 만들어 낼 수 있었습니다. 이어달리기에서는 한 구간의 환각이 다음 구간으로 전파됩니다.

검토한 선택지

감정 드리프트처럼 AI 감독의 중재로 보낸다

✓ 기계 검사로 막고 자동으로 반려한다(중재 없음)

결정

「드리프트는 충실도와 현지화 사이의 판단이지만, 발명된 엔티티가 정답인 경우는 없다」고 보고 중재하지 않기로 했습니다. 설계 문서에는 「프롬프트로는 보장되지 않는다. 보장은 기계적 검사와 차단 정책에서만」이라고 적었습니다. 검사는 네 가지입니다. 화자별 대사 블록 수, 고유명사·소품이 원본과 렉시콘 안에 있는지, 수치, 원장과의 정합입니다. 하나라도 어긋나면 그 구간을 다시 번역합니다. 구간마다 관문이 서고, 그중 기계 관문은 스스로 가짜 결함을 심어 시험합니다(아래 코드).

결과

다섯 번의 완주 모두 환각 0건이었습니다(파이프라인 자체 관문 기준). 대사 블록 계수는 설계 때 스크립트의 몫이었고, 설계 문서는 이를 「가장 신뢰도 높은 탐지기」라고 적었습니다. 그런데 스크립트는 +10 오차를 냈고, 화자 정렬 도구로 바꾸는 시도도 네 번 실패했습니다. 그래서 계수는 감사관이 하기로 하고, 설계 문서의 주장을 고쳐 적었습니다.

대가·남은 위험

서브텍스트를 미묘하게 비트는 류의 환각은 이 검사로 잡히지 않습니다. 그 몫은 감정 강도 재측정과 AI 블라인드 시사에 남겼습니다.

채점 대상은 감사관, 방법은 변이 주입 에이전트 기본값 · 확정 기록 없음

문제

번역 품질을 무엇으로 잴지 정해야 했습니다. 사람이 만든 기준본이 있었고, 번역을 판정하는 것도 AI 감사관이었습니다.

검토한 선택지

기준본과의 유사도(BLEU류)로 번역가를 채점한다: 「다르다와 틀리다가 구분되지 않는다」, 기준본에 있을 수 있는 실수까지 정답으로 굳는다

백테스트 실패에서 보정 규칙을 뽑아 에이전트에 심는 레지스트리: 표본이 너무 작고, 감사관과 번역가가 모두 Claude라 측정이 자기참조적이다

✓ 기준본에 일부러 결함을 심고, 감사관이 몇 개를 찾는지 잰다

결정

감사관의 재현율과 오탐률을 재기로 했습니다. 설계 문서는 「오탐률이 더 결정적」이라고 적었습니다. 합격선은 재현율 80% 이상, 오탐 10% 이하이고, 미달이면 실전에 쓰지 않습니다. 실패는 추가만 하는 실패 로그 하나에 쌓고, 같은 실패가 여러 씬에서 반복될 때만 정의 파일에 한 줄로 올립니다.

결과

첫 시도는 58%로 불합격이었습니다. 판정 기준을 더한 두 번째도 58%였습니다. 검사 범위를 모든 비트와 지문으로 넓힌 세 번째에 83%(12개 중 10개)로 합격했고, 확정 오탐은 0이었습니다. 기록에는 「무엇을 보라보다 어디를 보라가 효과적」이라고 남겼습니다. 기계 검사 층은 좋은 의역 9종을 하나도 결함으로 잡지 않았습니다.

대가·남은 위험

감사관도 번역가도 같은 계열 모델이라, 이 채점은 여전히 자기참조적입니다. 외부 기준에서는 같은 모델의 프롬프트 하나에 졌습니다(회고).

새 발견이 0건이던 단계는 넣을 때와 같은 기준으로 뺀다 본인 입력

문제

작품 A 7.0판에서는 한 회차가 너무 길었습니다(구간당 1~2시간). 단계마다 실제로 무엇을 찾았는지는 7.0판 실행에서 측정돼 있었습니다.

결정

9월 17일, 7.0판에서 새 발견이 0건이던 단계를 뺐습니다. 일관성 감사, 전체 연속성 감사, 포맷 에이전트(14분, 위반 0), 확정 문자열의 소급 재감사, 두 번째 보정 루프(루프마다 4줄 변경), 누출 재시험입니다. 효과가 있던 단계는 남겼습니다. 누출 시험은 한 번에 +15.5%p를 적발했고, 기계 관문은 잠금 누락 버그를 찾았습니다. 원칙은 「단계를 넣을 때와 같은 기준으로 뺀다」입니다.

결과

전체 감사 5종은 1종과 기계 검사로, 보정 루프는 2회에서 1회로 줄었고, 시사와 감사는 동시에 돌게 됐습니다. 그 뒤 작품 C와 D는 전 공정을 각각 하루에 끝냈습니다(파일 시각 기준).

대가·남은 위험

실행 중에 찾은 개선은 완주 뒤에 한꺼번에 반영하기로 했습니다. 10월 31일 작품 D에서는 그 대가를 치렀습니다. 미룬 결함 11종을 구간과 단계마다 다시 만났습니다(대문자 구분 문제 5구간 연속, 필드 이름 혼동 6번 등).

장르는 작품 설정의 판정 기준으로 에이전트 기본값 · 확정 기록 없음

문제

작품마다 장르가 달랐습니다. 장르마다 번역가와 감사관을 따로 둘지 정해야 했습니다.

검토한 선택지

장르별 에이전트: 역할 11개 × 장르 N개

✓ 같은 11개 역할 + 작품 설정에 장르별 실패 조건

결정

장르별 에이전트는 세 가지 이유로 기각했습니다. 조합이 폭발하고 정의의 95%가 겹칩니다. 작품은 대개 한 장르가 아닙니다. 장르 지식은 판정 기준에 속합니다. 예외로, 기존 역할에 없는 기술(가사 운율, 시대 고증, 액션 명료성, 전문 용어)이 필요할 때만 역할을 새로 둡니다.

결과

장르 4종을 같은 11개 역할과 작품 설정의 장르별 실패 조건으로 처리했습니다. AI 시사에서 한 명이라도 장르 실패 조건에 걸리면 그 씬을 다시 작업합니다.

품질과 검증

채점 대상은 감사관이었고, 일부러 망가뜨린 대본으로 먼저 시험했습니다.

채점 대상은 감사관이었습니다. 기준본에 9가지 유형의 결함 12개를 일부러 심고, 감사관이 몇 개를 찾는지 쟀습니다. 합격선은 재현율 80% 이상, 오탐 10% 이하이고, 미달이면 실전에 쓰지 않습니다. 세 번째 시도에 83%로 합격했습니다(아래 그림).

번역본 감사는 블라인드로 합니다. 감사관은 번역본만 읽고 씬 계약을 거꾸로 다시 써서 먼저 저장한 뒤, 원본 계약을 열어 대조합니다. 계약을 먼저 열었으면 그 감사는 무효이고 다시 돌립니다. 감정 강도는 비트마다 1~10으로 다시 재서, 원본과 3 이상 차이 나면 실패, 2면 경고입니다. 씬 계약 단위의 역추론입니다.

뜻이 문화 차이로 사라지는지도 따로 봅니다. 문화 자문이 원문 독자가 받는 뜻을 묻는 중립 질문을 만들고, 번역본만 읽은 AI 관객이 답합니다. 답은 도착·일부·손실로 나누고, 손실이면 잠금 항목이어도 AI 감독이 다시 봅니다.

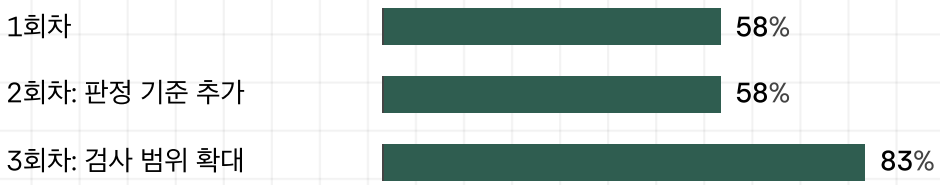
검사 도구도 검사합니다. 구간 관문은 대본 사본에 따옴표 대사·씬 삭제·잠금 문자열 변조를 심고, 하나라도 놓치면 관문 결과를 믿지 않습니다. 패킷의 누출은 답을 모르는 새 에이전트에게 원문만, 또는 원문과 패킷을 주고 반전을 추리하게 해 잽니다. 적중률이 원문만 준 기준선 +10%p 안에 들어와야 합니다.
AI 관객 4명과 AI 투자 프로파일 5종은 프로젝트 밖에 둔 번역본 사본만 읽습니다. 보고서에 적은 대로 실제 관객 조사가 아니어서, 이 사례에는 시사·투자 판정을 성과로 적지 않았습니다. 환각 0건과 대사 블록 수 일치도 파이프라인 자체 관문의 판정입니다. 같은 모델 계열이 같은 계열의 번역을 채점한다는 한계는 기록에도 적혀 있습니다.

CI·PR·훅은 없었습니다. 관문을 강제하는 것은 스킬과 역할 정의의 지시문, 그리고 도구의 종료 코드(구간 관문 차단, 성조 부호 검사 실패)입니다. 이 사례를 정리하며 11월 10일에 도구 자체 시험(selftest)을 다시 돌렸고, 12개 중 11개가 통과했습니다. 1개는 DOCX 라이브러리가 설치되지 않아 돌지 못했습니다. 단언 수를 센 테스트 스위트는 아닙니다. 10월 2일 일괄 개선 때의 회귀 결과(selftest 8/8, 관문 자체 검증 8/8)는 작업 기록이고, 다시 돌리지 않았습니다.

감사관 백테스트 재현율

판정 기준을 더한 2회차는 효과가 없었고, 검사 범위를 모든 비트와 지문으로 넓힌 3회차에 합격선 80%를 넘었다.

출처: 기준본에 변이 12개(9유형)를 심고 감사관이 찾은 비율, 주입 라벨과 엄격 대조. 세 번 모두 확정 오탐 0. 합격선은 재현율 80% 이상·오탐 10% 이하.



기능 하나, 처음부터 끝까지

시험 번역한 한 시퀀스(4씬)에서, 원본과 반대 반응이 나온 비트 하나를 끝까지 따라갑니다. 2026년 9월 16일 하루의 일이고, 판정은 대부분 AI 에이전트(AI 감독·AI 관객)가 했습니다. 여기에는 구조만 적습니다.

1. **시사 — 반응이 뒤집힌 비트 (12:49)**

   원본을 읽은 AI 관객 4명이 모두 의도대로 반응한 비트에서, 영어판 AI 관객 2명과 수정 전 한국어판 AI 관객 4명이 웃었습니다. 원인 후보는 어휘 쪽이었습니다.

2. **선례 반증 (12:51)**

   선례집에서 가져온 의태어 규칙이 이 장면에서는 긴장을 깎는다는 판정이 나왔습니다(AI 관객 기준). 그 뒤로 선례도 검증 대상이 됐습니다.

3. **1라운드 수정 → 불합격 (13:38~13:40)**

   한국어 의태어를 16종에서 4종으로 줄였지만, 웃은 AI 관객은 한국어 4명 → 3명, 영어 2명 → 3명이었습니다. 웃음이 번역가의 새 대체 표현으로 옮겨 갔고, 두 언어가 같은 방식으로 실패했습니다. AI 감독은 한 장면만 먼저 검사하는 게이트를 두어, 나머지 21건을 고치기 전에 멈췄습니다.

4. **빠뜨린 검사 실행 (14:43)**

   실제 번역본에 한 번도 돌리지 않았던 충실도 감사를 돌렸습니다. 감정 강도 실패는 0건이었습니다. 원인은 번역가가 아니라, 시사 뒤에 고친 지침이 그 전에 만든 씬 계약에 반영되지 않은 것이었습니다.

5 **계약 수정 → 재번역 (14:46 → 15:04)**

한국어판에서 웃은 AI 관객은 4명 → 2명, 장르를 원작대로 읽은 AI 관객은 1명 → 4명이 됐습니다. 밋밋하다거나 어둡다는 반응은 0이었습니다. 남은 웃음은 두 언어에서 같은 자리였고, 원본 AI 관객은 그 자리에서 웃지 않았습니다.

6 **원작 차원 판정 (15:32)**

번역으로 풀리지 않는 4건을 나눴습니다. 연출로 넘김 1건, 작가 확인 2건, 원작 유지 1건입니다. 해당 지점은 번역가를 동결하고 번역 품질 점수에서 뺐습니다. 이유는 「원인이 번역에 없는 실패를 번역가 탓으로 남기면 다음 회차에 또 엉뚱한 곳을 고친다」입니다. 가장 위험한 판정이라, 콘티 단계에서 다시 시사하도록 못 박았습니다.

7 **절차로 굳힘**

감사 결과의 목적지에 네 번째로 「원작 차원」을 더했습니다. 원본과 번역본의 AI 관객 반응이 갈리는데 번역이 정확하거나, 원본 관객도 같은 문제를 보이면 이리로 보냅니다. 보정 루프에도 두 가지를 넣었습니다. 바꾼 씬은 반드시 다시 감사하고, 같은 지적이 되풀이되는지 감시합니다.

회고

외부 기준에서는 같은 모델에 프롬프트 하나로 번역한 것에 졌고, 「통과」가 사실은 검사가 돌지 않은 것이었던 일도 두 번 있었습니다.

외부 기준에서는 프롬프트 하나에 졌습니다. 9월 17일, 공개 장편 3편의 공식 자막을 정답으로 놓고 대사만 번역해 비교했습니다(언어당 131쌍). 합격 기준은 데이터를 보기 전에 해시로 잠갔습니다. 공식 자막과 비교하면 파이프라인이 이겼습니다(승률 한국어 76.7%, 영어 84.4%). 같은 모델에 프롬프트 하나로 번역한 것은 공식 자막에 더 크게 이겼습니다(82.4%, 88.5%). 둘을 직접 맞붙이면 파이프라인의 승률은 한국어 32.1%, 영어 24%였고, 신뢰구간 상한도 50%에 못 미쳐 「열세」였습니다(주 판정자 기준. 다른 판정자 기준으로는 한국어 23.3%, 영어 15.6%로 더 낮았습니다). 결과 파일은 한계 넷을 함께 적었고, 그중 셋은 판정자가 번역자와 같은 계열 모델이고, 작품이 기준인 6편보다 적은 3편이며, 대사 전용 모드는 전체 파이프라인의 하한 추정입니다. 판정자는 다른 회사 모델로 하려 했지만 쓸 수 있는 키가 없어 같은 계열로 정했습니다. 이 결과를 해석한 문서는 쓰지 않았습니다.

「통과」가 검사의 통과가 아니었던 일이 두 번 있었습니다. 처음의 누출 검사는 패킷을 걸러 낸 함수와 같은 함수로 「잔존 0건」을 확인했습니다. 「검사 도구와 생성 도구가 같으면 검사가 아니다」라고 기록하고, 답을 모르는 에이전트의 블라인드 추리 시험으로 바꿨습니다. 10월 31일 첫 한국어 → 베트남어 작품에서는, 원문을 베트남어로 전제한 도구가 잠금 대조를 전부 건너뛰면서 경고 없이 「통과」로 보였습니다. 그 뒤 「새 언어쌍의 첫 작품은 검사가 0건인지, 안 돌았는지부터 본다」를 교훈으로 적었습니다.

처음에 일부러 뺀 도구가 관문의 본체가 됐습니다. 설계 문서는 대본 파서와 쪽수 계산을 파이썬으로 짜는 안을 「일부러 뺐다… 병목이 되면 그때 150줄짜리 stdlib 스크립트」라고 적었습니다. 실제로는 첫 커밋 묶음에 이미 파서(85줄) 등 인테이크 도구 3종이 들어갔고, 그 도구는 관문의 본체가 됐으며, 외부 평가 도구까지 합쳐 26개(4,816줄)로 늘었습니다. 다만 판단이 필요한 계수는 기계로 바꾸지 않았습니다(결정 3).

완주한 런의 기록을 두 번 덮어썼습니다. 10월 1일 도구 개선의 회귀 기준선을 뜨다가 작품 B의 패킷 일부 원본을 잃었고, 출력 위치 옵션과 「완주 런 폴더는 기록」 규칙을 더했습니다. 10월 31일에는 오케스트레이터가 다시 작품 D 첫 구간의 패킷과 관문 결과를 덮어썼습니다. 이때는 개정 백업으로 다시 만들어 바이트 단위로 같은지 확인한 뒤 복원했습니다.

일하는 방식

이 프로젝트도 역할별 AI 에이전트 팀으로 만들었습니다. 팀 구성과 게이트는 프로젝트마다 다르고, 이 프로젝트의 것은 위 「제 역할」에 적었습니다. 아래는 EWA 플랫폼을 만들며 정리한 방식입니다.

2026년 EWA 플랫폼을 만들며 세우고 운영한 방식 보기 →

https://jake-jo-portfolio.vercel.app/ko/process/

연락처

whxoans@gmail.com
+82 10-9233-7440