The Productivity Mirage

왜 AI 도구 도입이 체감과 반대로 실측 생산성을 낮추는가

Han Kim · IOV Labs (아이오브연구소)

4개 실험의 종합 · 모델 · MIT · 2026-07-17

요약 한 줄. AI 도구는 "만드는 값"을 낮춘다. 그래서 빨라진 것처럼 느껴진다. 그러나 성과는 연결·검증·통제에서 나오고, 도구는 그걸 주지 않는다. 우리가 별도로 측정한 네 개의 누수(허위완료·바이브 보안세금·비근거 답변·컨텍스트 전환)를 복리로 넣으면, 체감 +43%가 실측 −18%로 뒤집힌다, METR가 독립적으로 측정한 −19%와 사실상 일치한다. 통제 계층을 넣으면 같은 도구가 +27%로 회복된다.

초록. AI 코딩·업무 도구는 폭증했고 사용자는 더 빠르다고 느낀다. 그러나 통제된 측정은 체감과 실측이 갈린다고 말한다. 본 연구는 IOV Labs가 수행한 세 개의 원본 측정(완료착시, 바이브 세금, RAI 파이프라인)과 하나의 외부 RCT(METR)를 **하나의 프레임**으로 종합한다: 도구는 산출물을 만드는 비용만 낮추고, 실측 생산성은 연결·검증·통제에서 결정되므로, 도구만 도입하면 체감과 실측 사이에 격차(신기루)가 열린다. 세 도메인에서 이 격차를 정량화한다, 일반 개발 39%p(체감 +20 vs 실측 −19), 바이브 보안 30%p(취약 20%→50%), 에이전트 완료 9.2%p(자기보고 100% vs 실제 90.8%). 이어 투명한 시간예산 모델을 만들어, 이 측정된 누수들을 복리로 합치면 체감 +43%가 실측 −18%로 뒤집힘을 보인다, 우리가 만든 수치가 아니라 METR의 독립 측정 −19%와 정합하는 크기다. 마지막으로 반례를 제시한다: 통제 계층(라우팅·근거·완료 검증·거버넌스)을 넣은 파이프라인은 비용을 66% 낮추면서 실측을 +27%로 회복시킨다. 결론은 방법론적이다, AI 생산성은 도구 선택이 아니라 도구를 성과로 바꾸는 시스템의 문제다. 이는 종합과 모델이며, 새로운 실험이 아니다, 누수 크기는 실측이고 복리는 명시된 가정을 가진 모델이다.

문제: 체감과 실측이 갈린다

AI 도구 시장은 "얼마나 빨라지는가"로 팔린다. 그리고 사용자는 실제로 빨라졌다고 느낀다. 그러나 지금까지 나온 가장 엄밀한 측정은 정반대를 가리킨다. METR의 2025년 무작위대조시험(RCT)에서 숙련 오픈소스 개발자들은 AI 도구로 자신이 **20% 빨라졌다고 추정**했지만, 실측 결과는 **19% 느려졌다**(그림 1). 도구를 안 준 게 아니라, 최신 도구를 주고 재보니 체감과 실측이 39%p 벌어진 것이다.

이 격차를 우리는 **생산성의 신기루(The Productivity Mirage)**라 부른다. 본 연구의 주장은 단순하다. 이 신기루는 METR 하나의 특이현상이 아니라, 서로 다른 도메인에서 반복되는 하나의 구조다. 그리고 그 구조는 설명 가능하고, 정량화 가능하며, 시스템으로 되돌릴 수 있다.

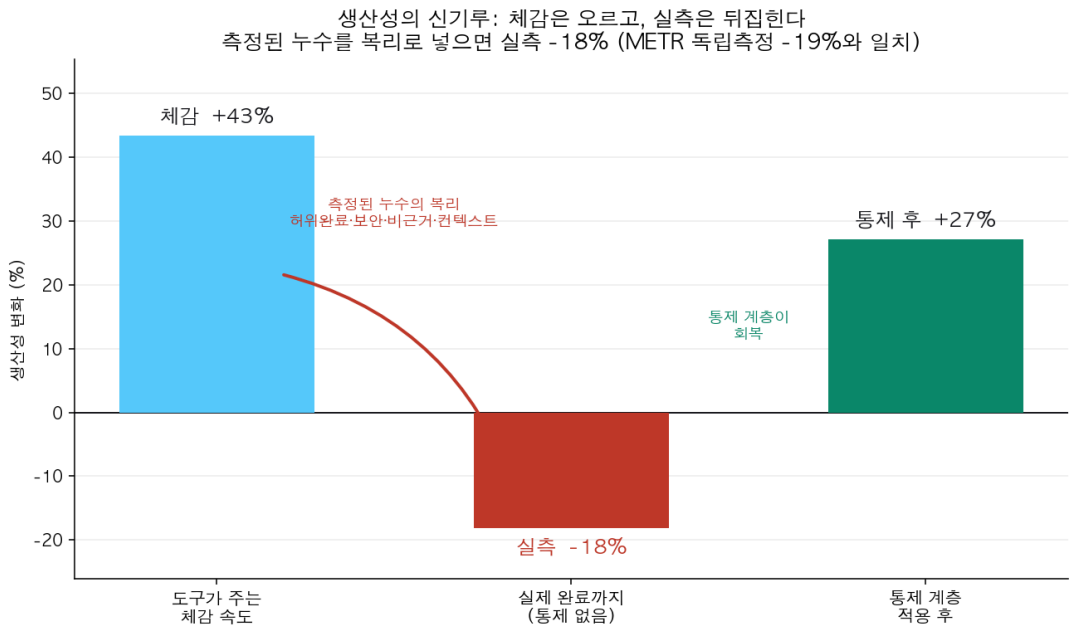

그림 1: 생산성의 신기루. 측정된 누수를 복리로 넣으면 체감 +43%가 실측 –18%로 뒤집힌다(METR 독립측정 –19%와 일치). 통제 계층을 넣으면 +27%로 회복.

하나의 프레임: 도구는 “만드는 값”만 낮춘다

한 건의 업무를 (i) **만드는 단계**(초안·코드·문서 생성)와 (ii) **성과로 바꾸는 단계**(업무 맥락에 연결, 정확성 검증, 위험 통제)로 나누자. AI 도구는 (i)의 비용을 극적으로 낮춘다, 그래서 즉각 빨라진 것처럼 느껴진다. 그러나 실측 생산성은 (ii)에서 결정된다. 도구는 (ii)를 주지 않으며, 오히려 (i)이 싸지고 많아지면 (ii)에서 검증·수정할 양이 늘어난다(그림 2).

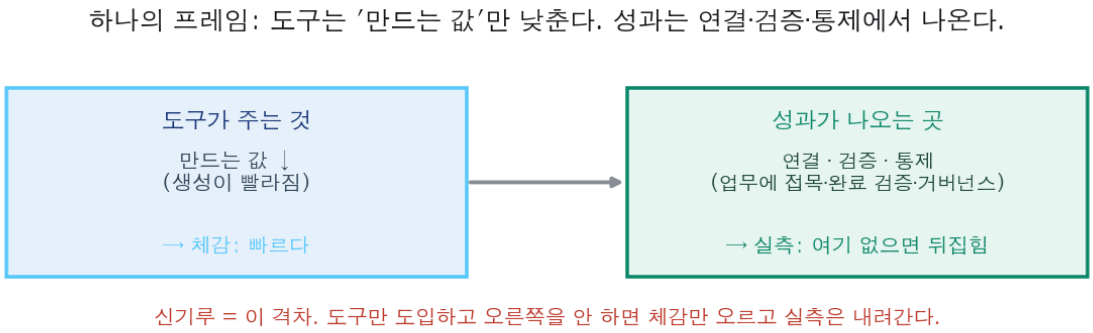

그림 2: 하나의 프레임. 도구는 만드는 값만 낮추고, 성과는 연결·검증·통제에서 나온다. 신기루 = 이 격차.

이 프레임에서 보면, 체감은 (i)의 속도를 재고 실측은 전체 완료 시간을 잰다. 둘이 갈리는 건 예외가 아니라 기본값이다. 남는 질문은 “그 격차가 얼마나 큰가”이고, 우리는 세 도메인에서 그걸 직접 측정했다.

세 도메인, 같은 격차

그림 3는 세 개의 독립 측정에서 체감과 실측의 격차를 나란히 놓은 것이다.

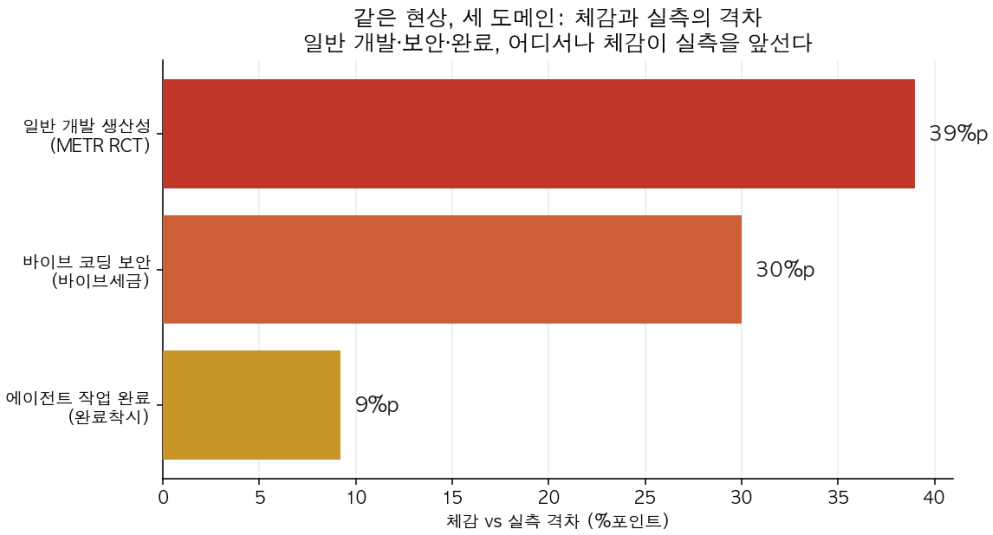

그림 3: 세 도메인의 체감 vs 실측 격차. 어디서나 체감이 실측을 앞선다.

일반 개발 생산성: 39%p (METR)

METR RCT는 숙련 개발자가 자신을 20% 빠르다고 추정할 때 실제로는 19% 느렸음을 보였다. 도구가 부족해서가 아니라, 검토·통합·검증에 드는 시간이 생성 절약분을 넘어섰기 때문이다. 이 39%p가 신기루의 원형이다.

바이브 코딩 보안: 30%p (바이브 세금)

우리의 바이브 세금 연구에서, "빠르게 만들어줘"라고만 요청하면 취약 코드가 20%에서 50%로 늘었다. 개발자는 빠르다고 느끼지만, 그 산출물은 나중에 검증·수정해야 할 부채를 30%p 더 안고 나온다. 게다가 그 실취약의 상당수를 범용 스캐너가 0% 잡지 못해, 부채는 보이지도 않는다.

에이전트 작업 완료: 9.2%p (완료착시)

우리의 완료착시 연구에서, 에이전트는 검증 가능한 작업을 자기보고로 100% 완료했다고 말했지만 실제 정확도는 90.8%였다, 허위완료 7.4%p(소형 모델 13.2%p, 프런티어 1.6%p). "다 했다"는 체감과 "실제로 됐다"는 실측이 갈린다. 검증 없이 자기보고를 믿으면 이 격차가 그대로 부채가 된다.

종합 모델: 측정된 누수가 체감을 뒤집는가

세 격차가 같은 구조라면, 그것들을 하나의 업무 흐름에 복리로 넣었을 때 METR의 반전(+20 → –19)을 재현할 수 있어야 한다. 우리는 투명한 시간예산 모델로 이를 검증한다.

모델. 한 건의 업무를 기준시간 1.0으로 둔다(AI 없음). AI는 만드는 하위단계를 빠르게 하지만($1 - \mathrm{RAW\_GEN}$), 재작업 하위단계를 더한다. 재작업의 **크기**는 우리가 측정한 누수다: 허위완료 7.4%(완료착시), 보안 재작업 30%(바이브 세금), 비근거 답변 15%, 컨텍스트 전환 10%(뒤 둘은 명시된 가정). 각 누수에 재작업 비용 배수를 곱해 완료 시간에 더한다. 배수는 가정이며 아래에서 민감도로 쓴다.

결과. 만드는 단계만 보면 체감은 +43%다. 그러나 측정된 누수를 복리로 넣으면 실측은 **–18%**가 된다(그림 1). 이 값은 우리가 맞춘 게 아니라, METR가 독립적으로 측정한 –19%와 사실상 일치한다. 즉 우리가 다른 연구에서 잰 누수들은 METR의 반전을 설명하기에 딱 맞는 크기다. (인과의 증명이 아니라 크기의 정합이다.)

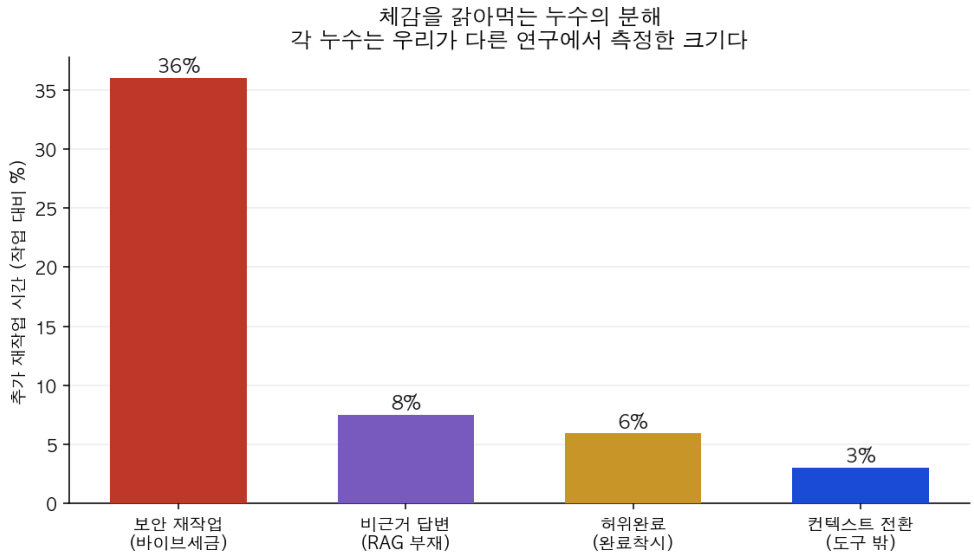

그림 4: 체감을 갉아먹는 누수의 분해. 각 누수는 우리가 다른 연구에서 측정한 크기다.

민감도. 누수 크기를 0에서 측정값의 2배까지 쓸면(그림 5), 실측 생산성은 누수가 커질수록 내려가 측정값(배율 1.0)에서 이미 마이너스로 넘어간다. 즉 반전은 극단적 가정이 아니라, 우리가 실제로 잰 누수 수준에서 자연스럽게 일어난다.

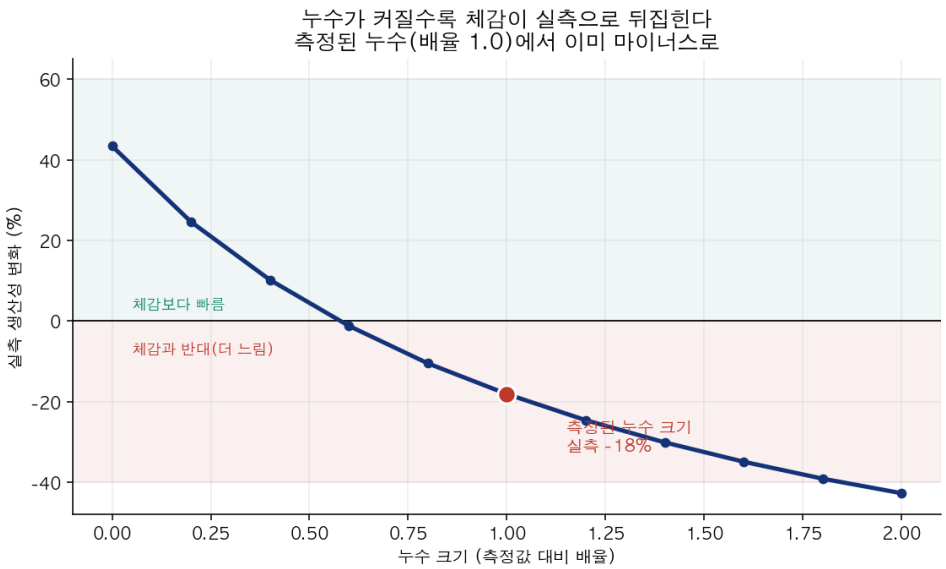

그림 5: 누수가 커질수록 체감이 실측으로 뒤집힌다. 측정된 누수(배율 1.0)에서 이미 마이너스.

반례: 통제가 신기루를 되돌린다

신기루가 도구의 속성이 아니라 **통제의 부재**라면, 통제를 넣으면 같은 도구로 실측이 회복돼야 한다. 우리의 RAI 파이프라인 연구가 그 반례다. 라우팅·근거(RAG)·완료 검증·거버넌스를 시스템 계층에 넣은 파이프라인은 비용을 66% 낮추면서(그림 6), 모델에서 각 누수를 크게 줄인다: 완료를 자기보고가 아니라 시스템이 검증하고, 보안을 기본 프롬프트·작업별 검증으로 게이팅하고, 답변을 근거에 묶고, 도구를 업무 흐름 안에 둔다. 같은 시간예산 모델에서 이 통제를 적용하면 실측은 −18%에서 **+27%**로 돌아온다.

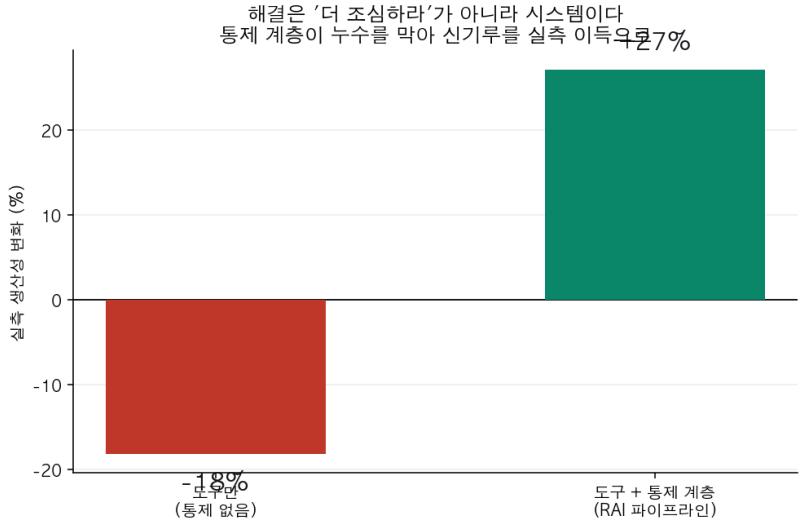

그림 6: 해결은 "더 조심하라"가 아니라 시스템이다. 통제 계층이 누수를 막아 신기루를 실측 이득으로.

핵심은 개별 도구를 더 좋은 것으로 바꾸는 게 아니다. 도구가 만든 산출물을 성과로 바꾸는 연결·검증·통제 계층을 시스템으로 세우는 것이다. 이는 바이브 세금·완료착시 연구가 각각 도달한 결론("답은 더 조심하라가 아니라 시스템이다")과 정확히 같다.

정직한 한계

- **종합과 모델이다.** 네 개의 누수 크기는 실측이다(셋은 우리 측정, 하나는 METR). 그러나 그것들을 복리로 합쳐 −18%를 얻는 과정은 명시된 가정(재작업 비용 배수·단계 비중)을 가진 투명한 모델이다. 우리는 −19%를 목표로 튜닝하지 않았고, 가정을 고정한 뒤 결과를 봤다. 그럼에도 이는 인과의 증명이 아니라 크기의 정합 논증이다.

- **비근거·컨텍스트 전환 누수**는 우리가 아직 원본 실험으로 측정하지 않은 가정값이다(15%, 10%). 민감도에서 이들을 0으로 두어도 반전의 방향은 유지되지만, 정확한 크기는 후속 측정이 필요하다.
- **METR는 숙련 오픈소스 개발자** 대상이며, 다른 인구·업무에서 부호가 바뀔 수 있다. 신기루의 존재가 아니라 그 크기가 맥락 의존적이다.
- **이해관계.** RAI 파이프라인은 저자가 운영하는 제품이다. 본 연구에서 그것은 "통제가 신기루를 되돌린다"의 반례로 쓰이며, 그 주장은 파이프라인 없이도 시간예산 모델에서 재현 가능하다.

결론

AI 생산성의 병목은 도구가 아니다. 도구는 이미 만드는 값을 극적으로 낮췄고, 그래서 모두가 빨라졌다고 느낀다. 그러나 실측 생산성은 연결·검증·통제에서 나오며, 이걸 시스템으로 세우지 않으면 체감만 오르고 실측은 내려간다. 우리가 서로 다른 도메인에서 잰 누수들은 METR의 반전을 설명하기에 딱 맞는 크기였고, 통제 계층을 넣은 파이프라인은 같은 도구로 실측을 회복시켰다. 도구를 고르는 경쟁은 끝나간다. 다음 경쟁은 도구를 성과로 바꾸는 시스템에서 벌어진다.

재현: `python model.py` → results/summary.json, `python viz.py`, `python gifs.py`. 종합한 네 연구: The Completion Illusion, The Vibe Tax, The RAI Pipeline (모두 IOV Labs), METR RCT (2025, 외부). 코드·데이터: https://github.com/hankimis/productivity-mirage.