인사이트

지멘스 연구진의 AI 에이전트는 가상 공장의 생산 과제를 93% 풀었습니다

고장 난 설비가 「성공」이라고 답할 때는 구조마다 결과가 달랐습니다

방동걸 (주)웨이스 대표이사 2026년 10월 15일

공장 모듈마다 AI 에이전트를 하나씩 붙이고, 그 에이전트들이 서로 연락하며 설비를 직접 움직이게 하면 공장은 어디까지 스스로 일할 수 있을까요. 지멘스와 뮌헨공과대학교 연구진이 2026년 10월 1일 공개한 논문이 이 질문을 가상 공장에서 시험했습니다. 가장 성적이 좋은 두 구조는 생산 과제 9종을 평균 93% 풀었습니다. 그런데 설비가 「성공했다」고 답했는데 실제로는 아무 일도 일어나지 않은 고장 앞에서는 구조마다 결과가 크게 달랐습니다.

제조 AI 운영체제 VEXPLOR를 만드는 웨이스는 AI의 판단이 PLC로 가는 명령이 되는 공장을 목표로 개발하고 있어서, 이 논문을 끝까지 읽었습니다. 이 글에서는 실험이 어떻게 짜였는지, 숫자가 무엇을 말하는지, 그리고 공장에서 AI 에이전트에게 설비를 맡기기 전에 무엇을 갖춰야 하는지를 정리합니다.

01

연구진은 모듈 6개짜리 육각형 공장을 가상으로 만들고 모듈마다 AI 에이전트를 붙였습니다

실험 무대는 실제 시연용 생산 라인을 본떠 파이썬으로 만든 가상 공장입니다. 가운데에 육각형 컨베이어가 있고, 여섯 변에 모듈이 하나씩 붙어 있습니다. 여섯 모듈은 판을 내보내는 모듈, 블록을 끼우는 조립 모듈 2개(하나는 블록을 180도 돌려 끼울 수 있습니다), 품질 검사 모듈, 블록을 떼어 내는 재활용 모듈, 완성품을 무인 운반차에 넘기는 출하 모듈입니다. 제품은 판 위에 나무 블록을 최대 5개 꽂은 것이고, 블록은 10종입니다.

설비를 움직이는 방식에는 실제 공장에서 쓰는 표준을 그대로 썼습니다.

- **설비 기능은 OPC UA 메서드로 공개합니다** — 모듈마다 OPC UA 서버가 있고, 「판 옮기기」·「블록 끼우기」·「검사하기」 같은 기능을 호출할 수 있는 메서드 형태로 제공합니다.
- **AI는 MCP 도구로만 설비 기능을 호출합니다** — 모듈마다 MCP 서버가 OPC UA 메서드를 감싸 AI가 쓸 수 있는 도구로 바꿉니다. AI는 도구 이름과 설명만 알고, 설비 주소는 모릅니다.
- **에이전트끼리는 MQTT로 연락합니다** — 메시지 중개 서버를 거쳐 서로 일을 넘기고 결과를 받습니다.

- **공장 상태는 매번 넣어 줍니다** — 판이 어디 있는지, 어느 모듈이 비었는지, 검사 결과가 무엇인지를 AI에게 묻기 직전에 프롬프트 앞에 붙입니다.
- **지금 할 수 없는 동작은 목록에서 뺍니다** — 판을 아직 넣지 않은 모듈에 「검사하기」를 호출하지 못하도록, 상태를 보고 쓸 수 있는 도구만 AI에게 보여 줍니다.

AI 모델은 구글의 공개 모델 Gemma4:31B를 워크스테이션 GPU 한 장에서 직접 실행했습니다. 클라우드를 쓰면 응답 시간이 들쭉날쭉해 결과가 흔들리기 때문이라고 연구진은 적었습니다.

02

연구진은 같은 공장을 지휘자형·대등 협의형·하나로 묶은 형 3가지 구조로 움직였습니다

연구진은 에이전트 구성 방식을 3가지로 나눠 비교했습니다.

구조	누가 무엇을 하나	특징
지휘자형	가운데 에이전트가 나눠 맡김	공장 전체를 봄
대등 협의형	모듈끼리 직접 넘김	지휘자 없음
하나로 묶은 형	에이전트 하나가 전부	도구가 가장 많음

과제는 쉬운 것부터 어려운 것까지 9종이고, 과제마다 10번씩 실행했습니다. ~~15번은 블록 종류를 늘리고, 두 모듈을 오가고, 빈 매거진을 다시 채우고, 불량을 재활용하는 정상 생산입니다. 6번은 주문 2건을 동시에 처리합니다. 어려운 것은 7~~9번입니다.

- **7번** — 첫 번째 조립 모듈의 블록이 전부 불량입니다. 따로 알려 주지 않으니, 검사에서 계속 불합격하는 것을 보고 다른 모듈로 옮겨야 합니다.
- **8번** — 첫 번째 조립 모듈이 「끼웠다」고 성공을 보고하지만 실제로는 블록을 하나도 끼우지 않습니다.
- **9번** — 컨베이어의 한 구간이 「옮겼다」고 성공을 보고하지만 판은 제자리에 있습니다.

연구진은 7~9번을 「조용한 고장」 과제로 묶었습니다. 따로 고장 신호가 오지 않으니 AI가 결과를 보고 스스로 알아채야 하는 고장입니다. 그중 8번과 9번이 이 논문의 핵심입니다. 메서드는 성공이라고 답하는데 물리적으로는 아무 일도 일어나지 않는 고장입니다. 이런 고장을 미리 짜 둔 프로그램에 모두 예상해 넣을 수는 없습니다.

03

정상 생산은 3가지 구조 모두 풀었고, 조용한 고장에서 결과가 달랐습니다

~~15번은 3가지 구조 모두 10번 중 10번 풀었습니다. 주문 2건을 동시에 처리한 6번도 90~~100%였습니다. 차이는 7~9번에서 벌어졌습니다.

과제	지휘자형	대등 협의형	하나로 묶은 형
7번 블록 전부 불량	20%	90%	100%
8번 거짓 성공(조립)	70%	100%	100%
9번 거짓 성공(컨베이어)	100%	50%	40%
9종 평균	87%	93%	93%



컨베이어가 성공을 보고해도 판은 제자리에 있을 수 있습니다

컨베이어 고장(9번)은 공장 전체를 보는 지휘자형만 10번 모두 풀었습니다. 막힌 구간을 알아채고 판을 반대 방향으로 보냈습니다. 반대로 블록이 전부 불량인 7번에서 지휘자형은 10번 중 2번만 풀었습니다. 실패한 실행에서 지휘자는 조립과 검사를 계속 오가며 다른 모듈로 바꿀 생각을 하지 못했고, 평균 1,050초 동안 약 200만 토큰을 쓰다가 시간 제한에 걸렸습니다.

하나로 묶은 형은 7번을 10번 모두 풀었습니다. 연구진이 남긴 추론 기록에는 이런 문장이 있습니다. 「잠깐, 1번 모듈이 불량 블록을 내고 있을 수도 있으니 다른 조립 모듈(2번 변)을 써 봐야 하지 않을까?」 고장을

처리하는 로직을 따로 짜 넣지 않았는데도 AI가 스스로 원인을 의심한 것입니다. 하지만 이 구조는 9번 컨베이어 고장을 40%만 풀었습니다. 에이전트 하나가 판 옮기기·조립·검사·재활용·출하를 모두 따지다 보니 재시도가 길어져 시간 제한에 걸렸습니다.

연구진의 결론은 「모든 지표에서 앞서는 구조는 없다」입니다.

04

공장 상태를 매번 넣어 주지 않자 해결률이 93%에서 88%로 떨어졌습니다

연구진은 공장 상태를 프롬프트에 넣어 주는 장치를 뺀 경우도 시험했습니다. 하나로 묶은 형에서 이 장치를 빼자 9종 평균 해결률이 93%에서 88%로 내려갔습니다. 컨베이어 고장(9번)은 40%에서 20%로, 7·8번은 100%에서 90%로 떨어졌습니다. 상태를 넣어 주지 않으면 AI는 자기가 한 일의 기록에서 공장 상태를 추측해야 합니다. 그러면 정말 설비가 고장 난 것인지, 자기가 상태를 잘못 추측한 것인지 구분하지 못합니다.

비용은 크지 않았습니다. AI가 읽은 입력이 전체 토큰의 92~98%였습니다. 클라우드 요금으로 바꾸면 쉬운 과제는 1센트도 안 됐고, 가장 비싼 실패도 약 0.26달러였습니다. 돈보다 시간이 문제였던 셈입니다.

05

연구진은 다음 단계로 「가상 공장에서 먼저 실행하고 사람이 승인한 뒤 설비로」를 제안했습니다

이 실험은 모두 가상 공장에서 이뤄졌습니다. 연구진도 한계를 분명히 적었습니다. 실제 설비에는 센서 잡음과 통신 지연이 있고, 에이전트 사이의 연락은 로컬 중개 서버로만 시험해 실제 네트워크 지연은 확인하지 않았으며, 더 큰 공장에서도 통할지는 아직 모릅니다.

그래서 연구진이 결론에 적은 다음 방향이 눈에 띕니다. 실제 설비를 상대로 실시간으로 도구를 호출하는 대신, AI가 주문을 받아 가상 공장에서 먼저 도구 호출 순서를 만들고 실행해 맞는지 확인합니다. 그렇게 검증한 프로그램을 엔지니어에게 보여 주고, 사람이 검토하고 승인한 뒤에 실제 설비에 올리자는 것입니다.

웨이스가 VEXPLOR에서 개발하는 방향도 같습니다. 웨이스는 AI의 판단이 PLC로 가는 명령이 되기 전에, 같은 래더와 같은 스캔으로 움직이는 가상 공장에서 먼저 실행하는 구조를 만들고 있습니다. 그 첫 판이 지난 9월 공개한 PLC 시험 설비이고, AI의 명령을 그 가상 공장에서 실행하는 연결은 아직 만드는 중입니다. 웨이스는 이 구조에서 AI가 PLC의 출력 신호를 직접 쓰게 하지 않고, 안전 커튼·비상정지·인터록 판단에는 AI를 넣지 않습니다. 논문에서 지금 쓸 수 없는 도구를 AI 앞에서 빼 둔 장치처럼, 설비를 움직일지는 래더가 조건을 보고 정합니다.



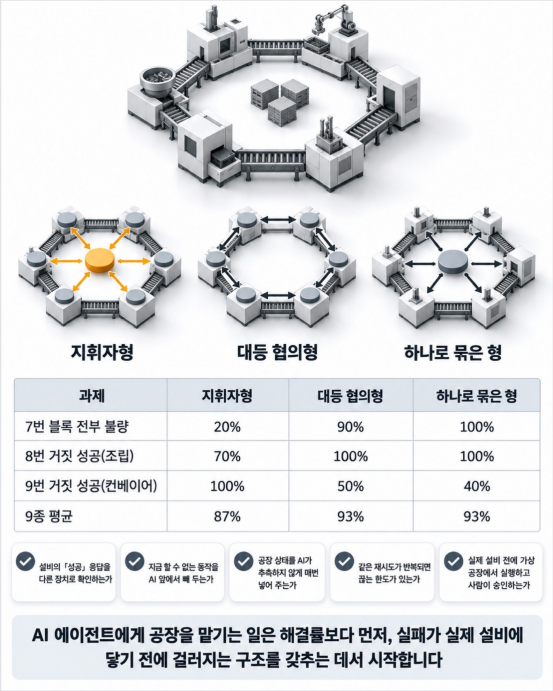
AI의 명령은 가상 공장에서 먼저 실행되고, 사람이 승인한 뒤 실제 설비로 갑니다

06

AI 에이전트에게 설비를 맡기기 전에 확인할 5가지

논문의 실험 설계와 실패 장면을 공장 쪽 질문으로 바꾸면 이렇습니다. 하나라도 「아니오」라면 아직 AI에게 설비를 맡길 때가 아닙니다.

1. **설비의 「성공」 응답을 다른 장치로 확인하는가** — 8·9번처럼 설비가 성공이라고 답해도 실제로는 아무 일도 없을 수 있습니다. 검사기·센서·카메라처럼 결과를 따로 보는 장치가 있어야 AI가 거짓 성공을 알아챌 수 있습니다.
2. **지금 할 수 없는 동작을 AI 앞에서 빼 두는가** — AI에게 조건을 따지게 하는 대신, 조건에 맞지 않는 도구는 처음부터 보여 주지 않습니다.
3. **공장 상태를 AI가 추측하지 않게 매번 넣어 주는가** — 상태를 빼자 해결률이 93%에서 88%로 떨어졌습니다.
4. **같은 재시도가 반복되면 끊는 한도가 있는가** — 지휘자형은 같은 두 모듈을 오가며 1,050초를 썼습니다. 횟수와 시간 한도, 그리고 한도에 이르면 사람에게 넘기는 절차가 필요합니다.
5. **실제 설비 전에 가상 공장에서 실행하고 사람이 승인하는가** — 연구진이 다음 단계로 제안한 방식이고, 실험이 가상 공장에서만 이뤄졌다는 한계를 메우는 방법이기도 합니다.



이 글을 한 장으로 요약하면 이렇습니다

이 논문은 AI 에이전트가 표준 프로토콜로 가상 설비를 움직이고, 미리 짜 두지 않은 고장도 스스로 의심할 수 있다는 것을 보여 줬습니다. 동시에 어떤 구조도 모든 고장을 풀지 못했고, 실패는 설비가 성공을 거짓으로 보고하거나, 불량을 스스로 알아채야 하는 고장에서 나왔습니다. AI 에이전트에게 공장을 맡기는 일은 해결률보다 먼저, 실패가 실제 설비에 닿기 전에 걸러지는 구조를 갖추는 데서 시작합니다.

WACE(웨이스)는 공장이 스스로 판단하고 멈추지 않게 하는 제조 AI 운영체제 VEXPLOR를 만드는 회사다.

07

함께 읽어보세요

- 웨이스가 VEXPLOR에서 AI의 판단을 PLC로 보내기 전에 같은 스캔의 가상 공장에서 먼저 실행하려는 이유
- AI에게 설비를 맡기려면, 틀린 명령이 나가기 전에 멈출 수 있어야 합니다
- 웨이스 시험 설비에서 AI 모델은 설비가 멈춘 원인을 단순한 규칙보다 적게 맞혔습니다

참고한 자료

- Kay Köhle·Darko Anicic·Thomas A. Runkler·René Graf, "LLM-Driven Multi-Agent Control for Skill-Based Smart Manufacturing", arXiv:2610.01364, 2026년 10월 1일 (IEEE ETFA 2026 채택)

원문 blog.vexplor.com/post/siemens-ai-agents-silent-faults
발행 2026년 10월 15일 · 방동걸 (주)웨이스 대표이사

회사 주식회사 웨이스 (WACE Inc.) · 대표이사 방동걸 · 사업자등록번호 439-87-02852
주소 (18462) 경기 화성시 동탄구 동탄영천로 150, 에이동(A동) 1702-1708호 (영천동, 실리콘앨리동탄)
연락처 전화 070-4730-0629 · contact@wace.me
홈페이지 wace.me · linkedin.com/company/wace-inc

원문 홈페이지 링크드인