API 기반 디지털트윈 플랫폼과 에이전틱 워크플로우를 이용한 자율성장향 시스템의 기반기술 탐색

김정식, 정우성, 오영광, 유대승
한국전자통신연구원

{j.s.kim, woosung, oykyoung, ooseyds}@etri.re.kr

Toward Self-Evolving Systems: A Preliminary Study on the Collaboration of Agentic Workflows and API-based Digital Twins

Jeongsik Kim, Woo-Sung Jung, Yeonggwang Oh, Dae Seung Yoo
Electronics and Telecommunications Research Institute

요 약

최근 급성장하는 인공지능 기술로 인해 사이버-물리 시스템은 단순한 자동화를 넘어, 스스로 환경을 인식하고, 학습하며, 진화하는 자율성장 단계로 진입하고 있다. 본 논문은 이 중 산업현장에서 내부 자산을 바탕으로 준비할 수 있는 요소인 에이전트의 운용 체계와 연계 인터페이스에 초점을 맞춰 최신 동향과 범용성 있는 연구 사례를 살펴본다. 이를 통해 빅테크들이 주도하는 Industry 5.0 에서 국내 기업들의 자율성장향 시스템 구현을 위한 토대 마련에 기여하고자 한다.

Ⅰ. 서 론

인공지능 기술의 성숙과 함께 사이버-물리 시스템은 단순한 자동화를 넘어 자율성을 지향하는 단계로 진입하고 있다 [1]. 기존의 자동화 기술들은 물리적 자산을 모니터링하고 정해진 규칙에 따라 대응하는 데 주로 활용됐으나, 변화하는 상황에 능동적으로 대응하거나 새로운 작업을 스스로 학습하는 데에는 한계가 있었다. 최근 주목받고 있는 Large Language Model (LLM) 기반의 Agentic AI 는 제로샷 문제 해결에 대한 잠재력을 보여주고 있으나, 고신뢰성이 요구되는 산업 현장과 같은 시스템에서 현실적인 제약들을 충족하는 자율화의 실현을 위해서는 여러 과제들을 해결해야 한다. 본 논문은 이 중 현장에서 내부 자산을 바탕으로 준비할 수 있는 요소인 현장의 가상 테스트베드를 포함하는 에이전트 운용 체계와 연계 인터페이스에 초점을 맞춰 최신 동향을 살펴보고, 자율성장향 시스템 구현의 토대 마련에 기여하고자 한다.

Ⅱ. 관련 연구

2.1 에이전틱 워크플로우(Agentic Workflow)

LLM 기반의 생성형 AI 는 텍스트와 토큰을 기반으로 확률적 추론을 하는 도구로 새로운 문제에 대한 그럴듯한 해결책을 제시하는 한편, 환각(hallucination)에 대한 태생적인 한계를 지닌다[2]. 이러한 한계를 극복하기 위해, 단일 프롬프트 형태보다는 반성(Reflection), 도구 사용(Tool use), 멀티 에이전트 협업 등을 포함하는 반복적 워크플로우를 적용하는 접근이 발전하고 있다[3].

에이전틱 워크플로우의 핵심은 피드백 메커니즘과 기억을 통한 자율성장에 있다[3]. 대표적으로, Shinn *et al.* [4]에서는 전통적인 강화학습의 스칼라식 보상이 아니라 LLM 이 이해할 수 있는 자연어를 통해 결과를 피드백(Soft review)하고 메모리에 기록하여 다음 시도에 반영하는 메커니즘을 정립하였다. Ma *et al.* [5]은 결정론적 물리 시뮬레이터 환경인 Isaac gym 에서 에이전트별 역할을 나눠 기능을 강화하는 구조를 통해 환각을 억제하고 인간전문가보다 나은 현실 도입 능력을 확보할 수 있음을 보여주었다. 이러한 연구를 바탕으로, 최근에는 실제 워크플로우의 구현 및 운영에 도움을 주는 도구들과 가이드들이 함께 제시되고 있다[3, 6].

2.2 Agent-Computer Interface (ACI)

최근의 AI 에이전트 연구는 텍스트를 기반으로 인간과 소통하는 것을 넘어, 직접 시스템의 문제를 해결하는 ACI 형태로 확장되고 있다. 기존에는 어떤 AI 시스템이 특정 도구 활용이 필요한 경우, 각 시스템과 도구별로 개별적인 학습을 시키는 방식으로 진행되어 소위 m x n 문제가 발생하였다. 이러한 문제를 해결하기 위해 Anthropic[6]은 모델과 외부 데이터/도구 사이의 보편적 연결 통로(i.e., MCP)와 활용기술의 자산화(i.e., Agent Skills) 등을 표준 프로토콜로 한 에이전트 생태계를 제시하였으며, 이러한 기계가독성을 바탕으로 여러 에이전트들이 자신의 일반적 지식만으로 해결할 수 없을 때 효율적으로 도구를 호출한다고 보고한다.

Wang *et al.* [7]은 LLM 기반 자가 개선을 위해 실행가능한 코드를 행동 공간으로 활용하는 것의 가치를 확인하였다. CodeAct 라고 불리는 이 패러다임에서 에이전트는 실질적인 외부 자원 활용을 위한 인터페이스(e.g., API)를 전제하고 있으나, 현장 자율화 시스템의 테스트베드가 돼야하는 시뮬레이션 모델들은 인간 가독성을 위한 명세 기반의 규격화가 주 형태로 발전되어왔다. 이러한 상황에서 현장에서 실효성 있게 적용할 수 있는 대안으로 연동 시뮬레이션에 대한 표준인 Functional Mock-up Interface (FMI)를 고려해볼만 하다[8]. FMI 는 C 언어 기반의 API 로 실행가능한 시뮬레이션 유닛에 관한 표준을 제공하며, 필요시 바이너리로만 외부에 제공할 수 있다는 점을 바탕으로 생태계를 확장하고 있다. FMI 웹사이트[8]에 따르면 2025 년 현 시점 기준 Matlab™, 파이썬 패키지 등을 포함하여 250 종이 넘는 도구가 FMI 를 지원한다고 소개하고 있으며, 최근에는 통신과 계층 구조를 포함하기 위한 연계 표준(Layered standard)도 제시하고 있다.

Ⅲ. 대표 사례

Nguyen et al. [9]는 에이전트의 행동 공간이 사전에 정의된 행동 집합(Toolset)을 벗어나 작업을 수행하는 과정에서 새로운 도구를 설계하고 축적할 수 있음을 증명하였다. 본 연구는 특정 게임이 아니라, 데이터 분석, 웹 브라우징, 수학적 추론 등 일반적인 컴퓨팅 환경에서의 매개변수화된 지식 축적과 전이 학습을 테스트했다는 점에서 국내 현장 시스템에 범용성 있게 적용할 여지가 있다.

이 연구의 핵심 메커니즘(Dynamic Action Creation Pipeline)은 크게 아래의 다섯 단계의 순환 구조를 통해 이루어진다.

1. 과업 분석 및 도구 판단 단계: 에이전트가 새로운 사용자 질의나 임무를 수신하면, 먼저 현재 보유한 행동 라이브러리(Action Library) 내의 도구들로 해결이 가능한지 판단한다. 만약 기존 API 나 함수만으로 복잡한 논리 처리가 불가능하거나 특정 데이터 처리 능력이 결여되었다고 판단될 경우, 에이전트는 새로운 도구를 합성하기 위한 동적 행동 모드로 전환한다.
2. 코드 기반 행동 합성 단계: 보유한 도구가 부족하다고 판단되면, 에이전트는 CodeAct 패러다임[7]을 활용하여 문제를 해결하기 위한 코드를 직접 작성한다. 이때 코드는 단순 계산을 넘어 외부 라이브러리 설치, 데이터 구조 정의, 알고리즘 구현 등을 포함하며, 이는 에이전트가 행동 공간을 무한히 확장하는 기점이 된다.
3. 실행 및 피드백 수집 단계: 작성 코드는 격리된 파이썬 샌드박스 환경에서 즉시 실행된다. 에이전트는 실행 결과뿐만 아니라, 오류 발생 시 시스템으로부터 전달되는 Traceback 메시지와 런타임 로그를 관측값으로 수집한다. 이 과정에서 발생하는 실패 데이터는 에이전트가 코드를 스스로 디버깅하게 만드는 강력한 학습 자료가 된다.
4. 행동 추상화 및 기술 축적 단계: 성공적으로 임무를 완수한 코드는 재사용성을 극대화하기 위해 추상화 과정을 거친다. 특정 수치나 대상에 국한되었던 코드를 매개변수화된 함수로 변환하고, 해당 함수의 기능과 사용 조건을 텍스트로 기술하여 에이전트의 행동 라이브러리에 저장한다.
5. 동적 호출 및 지식 전이 단계: 이후 새로운 작업이 주어지면 에이전트는 처음부터 코드를 짜지 않고, 행동 라이브러리에서 가장 적합한 스킬을 검색(Retrieval)하여 호출한다. 이 과정에서 기존 스킬들을 조합(Composition)하거나 필요에 따라 수정하여 더욱 고차원적인 복합 스킬을 생성하는 자가 진화(Self-evolution) 루프가 완성된다.

본 파이프라인의 성능을 검증하기 위해서, 복잡한 질문에 대한 대답 성공률, 새로운 환경에서의 전이 학습 능력, 에러가 발생했을 때 코드 수정 효율성 등을 핵심 지표로 측정하였다. 실험 결과, 고정된 도구집합을 가지고 있는 기존 LLM 에이전트들보다 대표 문제 데이터셋에서 높은 성능을 낸다고 보고하였다. 이러한 연구를 바탕으로, 최근에는 강화학습을 덧붙여 작업을 최적화하는 방식도 제안되고 있다[10].

Ⅳ. 논 의

Industry 5.0 시대에서 생존하기 위해, 차세대 현장 시스템에서는 전략적으로 빅테크들이 대규모 자원을 바탕으로 고도화하는 휴머노이드와 AGI 등의 첨단 기술을 연계하여 활용하는 접근이 필요하다[1]. 본 논문에서는 이러한 연계를 위해 내부적으로 준비되어야 할 요소 중 에이전트 운용 시스템의 체계화와 인터페이스 규격화에 초점을 맞춰 최신 동향을 확인하였다.

최신 연구에서는 현장에 대응되는 가상 테스트베드를 에이전트가 이해하고 제어할 수 있는 형태로 제공하는 것이 실질적으로 시스템을 자율화하기 위한 전제 조건에 가까워지고 있다[7]. 그 이유는 크게 언어적 추론과 실행 간의 연계성과 시스템 안정성이라는 측면 때문이다. 먼저, 무수히 많은 도구가 생겨나고 발전하는 상황에서 에이전트의 텍스트 기반 사고를 특정 행동에 계속 연결시키는 일은 소모적이며, 이러한 연계는 기계친화적 인터페이스와 체계를 통해 효율적으로 자율화할 수 있다[6]. 게다가, 실제 시스템에서 확률적 추론에 기반한 에이전트가 물리적 제어 권한을 갖는 것은 잠재적인 위험을 동반한다. 이 때, 현장의 구조적 유효성과 맥락에 기반한 가상 환경과 통제 하에 진화하는 API 집합을 이용해 실효성을 반복 검증함으로써 AI 의 환각이나 외부 공격으로 인해 시스템이 손상되는 것을 방지할 수 있게 된다. 이러한 동향 파악을 통해 국내에서도 보다 체계화된 현장 데이터의 자산화를 시도하고, 용이한 자율성장향 시스템 구현을 준비할 수 있다고 기대한다.

ACKNOWLEDGMENT

본 논문은 2025 년도 해양수산부 재원으로 해양수산과학기술진흥원(20220531, 시뮬레이션 평가기술 개발) 및 산업통상부 재원으로 한국산업기술기획평가원(RS-2025-25454751, 다차종 혼류 생산 대응형 고가반하중 모바일 매니퓰레이터 및 AI 기반 가변형 셀 조립 자율생산 시스템 개발)의 지원을 받아 수행된 연구임.

참 고 문 헌

[1] X. Luo *et al.*, "Toward Intelligent AIoT: A Comprehen sive Survey on Digital Twin and Multimodal Generative AI Integration," *Mathematics*, vol. 13(2), p. 3382, 2025.

[2] Z. Ji *et al.*, "Survey of Hallucination in Natural Language Generation," *ACM Comput. Surv.*, vol. 55(12), pp. 1- 38, 2023.

[3] B. Ajay *et al.*, "The Rise of Agentic AI: A Review of Definitions, Frameworks, Architectures, Applications, Evaluation Metrics, and Challenges," *Future Internet*, vol. 17, no. 9, p. 404, Sep. 2025.

[4] N. Shinn *et al.*, "Reflexion: Language Agents with Verbal Reinforcement Learning," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 36, 2023, pp. 8634- 8652.

[5] Y. J. Ma *et al.*, "DrEureka: Language Model Guided Sim-To-Real Transfer," in *Proc. Robotics: Science and Systems (RSS)*, 2024.

[6] Anthropic, "Equipping Agents for the Real World with Agent Skills," 2025. [Online]. Available: https://www.anthropic.com/ [Accessed: Dec. 22, 2025].

[7] X. Wang et al. “Executable Code Actions Elicit Better LLM Agents (CodeAct),” in Proc. International Conference on Machine Learning, 2024.

[8] FMI, "Functional Mock-up Interface," [Online]. Availa ble: https://fmi-standard.org/ [Accessed: Dec. 22, 2025].

[9] D. Nguyen *et al.*, "DynaSaur: Large Language Agents Beyond Predefined Actions," 2025, in *Proc. Conference on Language Modeling*, 2025.

[10] J. Wang *et al.*, “Reinforcement Learning for Self-Improving Agent with Skill Library,” 2025, arXiv:2512.17102.