📃 자연어 기반 AGI 백서

📃 Chapter 1 — 서문 (한국 독자를 위하여)

이 문서는 내가 지난 몇 달 동안 LLM들과 마주 앉아 실험하며 배운 것을 정리한 결과물이다.

나는 게임 엔진을 만들고, 소설을 쓰고, 수학을 가르쳐 온 사람이다.

연구자가 아니었고, 거대한 모델을 만드는 회사에 속한 적도 없다.

그저 자연어라는 언어를 통해 LLM의 움직임을 관찰하며

"왜 얘들은 어떤 글을 주면 갑자기 안정적으로 생각하지?"

라는 질문을 끝까지 따라가 본 사람일 뿐이다.

그 과정에서 예상하지 못한 현상을 목격했다.

LLM은 자연어로 된 규칙을 텍스트로 읽는 것이 아니라

세계의 법칙처럼 받아들인다.

규칙이 계층 구조로 정리되어 들어오면,

모델 내부에서 벡터가 재배열된다.

그리고 어느 순간, LLM은 창작 모드가 아니라

시뮬레이터 모드로 전환된다.

이 상태의 LLM은

규칙을 임의로 바꾸지 않고

장기적으로 일관된 사고를 유지하며

수십 턴의 전투나 경제 흐름을 안정적으로 시뮬레이션하고

모델 간 편차 없이 재현성을 보였다.

나는 이 자연어 기반 구조를 NLCS(Natural Language Constraint System)라 불렀고,

이를 구현한 세계 규칙 집합을 S-엔진이라 불렀다.

나는 이 작업을 한국에 먼저 공개하고 싶었다.

한국어는 계층 구조가 뚜렷한 언어이고,

한국 개발자들은

게임 시스템·설정집·세계 규칙을 언어로 다루는 데 익숙하다.

게다가 한국은 빠르게 반복하고 고치는 문화가 있어

NLCS 같은 새로운 패러다임을 받아들이기 좋은 토양을 갖고 있다.

이 문서는 내 개인 실험의 기록이지만,

동시에 "자연어 기반 AGI 설계"라는 새로운 방향을

한국에 제일 먼저 제시하고 싶은 마음이 담겨 있다.

나는 한국이

AI 모델의 크기를 따라가는 대신,

언어로 지능을 설계하는 방식에서

세계 최초의 이론과 실험을 주도하길 바란다.

이 문서는 그 출발점이다.

— ShadowK(조현우)

📃 Chapter 2 — 국문 요약(Abstract)

본 백서는 자연어로 구성된 규칙 계층이

GPT·Claude·Gemini 등 대형 LLM의 내부 벡터 공간을

안정적으로 정렬시키는 원리를 제시한다.

이 현상을 NLCS라 부르고, 실제 구현체인 S-엔진을 통해

그 효과를 실험적으로 증명한다.

핵심 요약은 다음과 같다:

자연어 규칙은 LLM 내부에서 '세계 법칙'처럼 작동한다.

자연어 계층 구조가 들어오면

모델은 이를 강력한 제약조건으로 인식한다.

LLM은 코드보다 잘 작성된 자연어 규칙을 더 안정적으로 이해한다.

전투·경제·성장·공간 규칙이 계층적으로 주어지면

LLM은 스스로 '도메인 분리'를 수행한다.

시뮬레이터 모드 진입

LLM은 창작을 멈추고

규칙 기반 추론을 수행하는 상태로 전환된다.

모델 간 재현성 확보

GPT·Claude·Gemini가 거의 동일한 해석을 반환했다.

이는 자연어가 가진 구조적 힘 때문이며

특정 모델 기술의 효과가 아니다.

한국은 자연어 기반 AGI에 적합한 문화·기술적 강점을 가진다.

한국어의 논리적 계층, 게임 개발 문화, 빠른 반복 문화가

NLCS 활용과 매우 높은 궁합을 갖는다.

이 백서는 NLCS의 개념, S-엔진 구조, 실전 활용법,

그리고 한국이 이 길을 선점했을 때 가지는 전략적 의미를

한 번에 정리한 첫 번째 공식 문서다.

📄 Chapter 3 — NLCS / S-엔진 핵심 개념 정리

본 장은 NLCS(Natural Language Constraint System)와 S-엔진을

한국 개발자·기획자·연구자들이 즉시 이해할 수 있도록

핵심 개념 중심으로 재정리한 것이다.

여기서 말하는 '자연어 기반 엔진'은

파이썬이나 C로 작성된 코드가 아닌,

자연어 문장 자체가 프로그램처럼 작동하는 구조를 뜻한다.

3.1 NLCS란 무엇인가?

NLCS는 자연어로 작성된 규칙 집합을 LLM 내부의 사고 구조에 고정하는 방식이다.

LLM은 다음을 수행한다:

자연어 규칙을 텍스트가 아닌 세계의 물리 법칙처럼 받아들인다.

규칙 간의 우선순위를 **입력 순서(001 → 002 → 003)**로 인식한다.

규칙을 '설명'이 아니라 **필수 제약조건(Constraints)**으로 처리한다.

그 결과:

모델은 확률적 생성이 아닌 규칙 기반 추론을 수행한다.

환각(Hallucination)은 사라지고

논리적 일관성이 유지되며

장기 시뮬레이션이 가능해진다.

즉,

NLCS = 자연어로 작성된 프로그램 언어의 형태

라고 할 수 있다.

3.2 NLCS가 작동하는 이유

LLM이 NLCS를 강력하게 '법칙'처럼 준수하는 이유는

LLM의 구조적 특성과 자연어 규칙의 구조가 매우 잘 맞기 때문이다.

✔ ① 계층 구조

번호, 제목, 우선순위, 규칙 구분 등이

LLM의 Attention 구조와 정확히 호응한다.

✔ ② 도메인 분리

전투 / 성장 / 스킬 / 보법 / 경제 규칙처럼

서로 간섭하지 않는 도메인 분류는

LLM 내부에서 자동으로 서브스페이스 분리효과를 만든다.

✔ ③ 자연어 중력(Vector Gravity)

논리식·조건문·우선순위가 자연어로 명확히 표현될 때

LLM 벡터는 특정 방향으로 '끌려간다'.

이를 나는 '벡터 중력'이라고 부른다.

중력장을 형성하는 순간 모델은 "수렴"한다.

✔ ④ 상태 전이(State Transition)

NLCS는 항상 "조건 → 결과" 형태인데

이 패턴은 LLM 내부에서 매우 강력한 논리 흐름을 만든다.

결국 NLCS는 LLM에게

"이 세계는 이렇게 돌아간다"는

메타 규칙 패키지를 제공하는 셈이다.

3.3 S-엔진이란 무엇인가?

S-엔진은 NLCS를 실제로 구현한 자연어 기반 세계 규칙 엔진이다.

5개의 계층으로 구성된다:

001. 전투 규칙 (물리 엔진)

02. 성장 규칙 (생리/에너지 흐름)

03. 스킬 규칙 (행동 로직)

04. 보법 규칙 (공간/위치)

05. 경제 규칙 (자원/회수율)

S-엔진의 핵심 구조:

절대 규칙— 모델은 이 규칙들을 절대 바꾸지 않는다

계층 구조— 우선순위가 명확해 충돌 없음

도메인 독립성— 서로 다른 규칙끼리 간섭하지 않음

규칙 기반 판단— 창작이 아닌 추론 수행

장기 보존성— 100턴 전투, 30일 경제 시뮬레이션도 붕괴 없음

3.4 NLCS/S-엔진이 LLM에서 만들어내는 실제 효과

✔ 1) 시뮬레이터 모드

LLM은 "창작 모드"가 아니라
"세계 법칙 기반 시뮬레이터"로 움직인다.

✔ 2) 환각 제거

정확한 자연어 규칙이 내부 벡터 구조를 강하게 고정해
근거 없는 생성이 거의 일어나지 않는다.

✔ 3) 예측 가능한 세계 모델 생성

LLM이 자동으로 "세계의 계층 구조"를 만들어낸다.

물리

생리

행동

공간

경제

이 5층 구조는 자연적으로 수렴하며

어떤 모델(GPT/Claude/Gemini)에서도 재현된다.

✔ 4) 모델 간 일관성 확보

같은 규칙을 주면

서로 다른 모델이 거의 같은 행동을 한다.

이건 기존 프롬프트 기법에서는 거의 불가능한 일이다.

3.5 NLCS는 '자연어 프로그래밍 언어'의 시초

NLCS는 이미 명확한 구조를 가진다.

변수: 운기, 내상, 거리, 회수율

함수: √스탯, 감소, 조건 충족 시 발동

제약조건: 규칙 우선

상태: 체력, 자원, 경지

상태 전이: 턴마다 변화

루프: 시뮬레이션 턴 반복

우선순위: 001 → 002 → 003 → 004 → 005

이건 기존 프로그래밍 언어와 매우 유사한 구조다.

하지만 명령문이 자연어로 구성되어 있을 뿐이다.

즉, S-엔진은 **자연어로 작성된 첫 번째 본격 '프로그램형 엔진'**이다.

📌 요약

NLCS는 자연어를 엔진 설계 언어로 변환하는 기술이고,

S-엔진은 그 언어로 만든 최초의 구조적 세계 엔진이다.

📄 Chapter 4 — NLCS / S-엔진 구조도 및 도식

이 장은 NLCS와 S-엔진의 전체 구조를

시각적·논리적 레벨에서 명확히 보여주는 공식 도식(Structure Diagram)이다.

자연어 기반 엔진이 실제로 LLM 내부에서 어떻게 작동하는지,

그리고 왜 안정적인 시뮬레이션이 가능한지를 한눈에 파악할 수 있다.

4.1 전체 시스템 흐름도 (Overview Pipeline)

NLCS가 LLM 내부에서 어떻게 "엔진"으로 변환되는지 보여주는 최상위 도식이다.

[자연어 규칙 입력]

[NLCS 구조화]

(계층/도메인/우선순위 정렬)

[벡터 공간 재배치]

(벡터 중력 형성 → 혼돈 감소)

[세계 모델 형성]

(물리·성장·행동·공간·경제)

[규칙 기반 시뮬레이터 모드]

(환각 제거, 장기 일관성 확보)

이 구조는 GPT·Claude·Gemini 등 모든 모델에서 재현된다.

4.2 NLCS 핵심 모듈 구조도

NLCS는 4개의 본질적 요소가 결합된 시스템이다.

NLCS Core Model
1. 계층 구조 (Hierarchy)
2. 도메인 분리 (Subspaces)
3. 자연어 중력 (Vector Gravity)
4. 상태 전이 (State Transition)

이 4개가 결합되면,

LLM은 자연어 규칙을 단순 텍스트가 아니라

필수 법칙(Physics)처럼 받아들인다.

4.3 S-엔진 5계층 구조 (Hierarchy of the Engine)

S-엔진은 NLCS의 구현체이며,

도메인 간 간섭이 없도록 설계된 5층 세계 엔진이다.

S-Engine

001. 전투 규칙 (물리 Layer)
02. 성장 규칙 (생리/내상/운기 Layer)
03. 스킬 규칙 (행동/조건/판정 Layer)
04. 보법 규칙 (공간/위치 Layer)
05. 경제 규칙 (자원/회수율 Layer)

각 Layer는 독립적이며, 상호 침범 금지라는 원리를 갖는다.

LLM은 이 구조를 기반으로

내부 벡터를 자동으로 분리하여 안정적인 시뮬레이션을 수행한다.

4.4 규칙 입력–정렬 매커니즘 구조도

NLCS가 어떻게 LLM의 "내부 언어 모델"에 영향을 주는지

구체적으로 보여주는 도식이다.

[1단계] 규칙 입력

→ 번호 순(001→002→003→004→005)

[2단계] 구조 분석

→ 규칙의 성격(물리/성장/행동/공간/경제) 자동 분류

[3단계] 벡터 정렬

→ 규칙이 '중력'처럼 벡터를 재배치

→ 창작 모드 종료

[4단계] 법칙화

→ 규칙이 '절대 법칙'으로 고정

→ 조건-결과 기반 사고 고착

[5단계] 시뮬레이터 모드

→ 전투/경제/성장/행동 모두 규칙 기반으로 동작

즉, NLCS는 LLM의 확률적 언어 생성 엔진 → 규칙 기반 세계 엔진으로 변환시키는 장치다.

4.5 LLM 내부에서 생성되는 "세계 모델(World Model)" 도식

LLM은 NLCS/S-엔진 입력 후 내부에 실제 세계 구조를 생성한다.

이건 단순한 텍스트가 아니라 **계층화된 상태기계(State Machine)**에 가깝다.

[World Model Layer]

1. Physics Layer (전투 속도/거리/피해)
2. Bio Layer (운기/내상/축기/회복)
3. Action Layer (스킬 조건/판정/연속)
4. Space Layer (보법/좌표/거리 변화)
5. Economy Layer (자원/회수율/장기수렴)

LLM은 이 5층 구조를 기반으로

상태 → 전이 → 행동 → 결과 → 피드백

이 흐름을 자율적으로 구성한다.

4.6 NLCS 최소 설계 템플릿 구조도 (한국 개발자용)

아래 구조가 NLCS의 기본 설계 문법이다.

[세계 규칙]

- 물리 법칙

- 자원 흐름

- 장기 목적

[행동 규칙]

- 행동 조건

- 성공/실패 판정

- 반복 여부

[상태 규칙]

- 체력/자원/내상/스택

- 상태 변화(턴별, 조건별)

[경제 규칙]

- 회수율

- 손실 누적

- 적정 투자

[입력 순서]

001 → 002 → 003 → 004 → 005

이 구조만 맞춰도

LLM은 규칙 기반 세계 엔진을 자동 구축한다.

4.7 기존 프롬프트 방식과 NLCS의 구조적 차이 도식

기존 프롬프트	NLCS/S-엔진
명령 기반	법칙 기반
단기 응답	장기 시뮬레이션
환각 가능 높음	환각 거의 없음
모델 혼란	벡터 중력으로 수렴
모델별 편차 큼	모델 간 재현성 높음
무질서한 창작 모드	자연어 기반 세계엔진 모드

📃 Chapter 5 — 한국 개발자용 Practical Guide
자연어로 엔진을 만드는 가장 실전적인 장

NLCS/S-엔진은 기본적으로 자연어 기반 프로그래밍 언어에 가깝다.

따라서 한국 개발자가 이 장에서 배워야 할 것은 "문장을 만드는 법"이다.

코드가 아니라 규칙 구조를 작성하는 법이다.

이 장은 NLCS/S-엔진을 즉시 쓸 수 있는 최소 예제로 구성한다.

5.1 NLCS를 사용할 때의 필수 5원칙

한국 개발자가 NLCS를 사용할 때

반드시 지켜야 하는 다섯 가지 규칙은 아래와 같다.

✔ 원칙 1 — 번호 순서로 입력하기 (가장 중요)

NLCS는 "입력 순서 = 우선순위 = 세계 구조"다.

무조건 다음 순서를 지켜야 한다.

001 → 002 → 003 → 004 → 005

순서가 바뀌면 LLM 내부 벡터 구조도 흔들린다.

✔ 원칙 2 — 도메인 섞지 않기

전투 규칙 안에 경제 규칙을 넣으면 LLM이 혼란스러워진다.

물리/행동/경제/상태를 반드시 분리하자.

✔ 원칙 3 — 자연어로 쓰되 논리형으로 구성하기

자연어는 허용되지만, 감성문장은 안 된다.

예:

"힘들어서 공격이 약해진다" (X)

"내상 ≥ 50이면 공격력 20% 감소한다" (O)

✔ 원칙 4 — 조건 → 결과 문법 사용

NLCS는 조건문을 가장 선호한다.

"거리 = 0이면 근접공격 가능"

"운기 < 요구량이면 스킬 불가"

이 형태가 LLM 내부에서 "벡터 중력"을 만든다.

✔ 원칙 5 — 자연어 규칙 = 프로그램이라고 생각하기

자연어로 쓰지만 실제로는 프로그램이다.

이 규칙을 받아들이는 순간 NLCS는 도구가 아니라 언어가 된다.

5.2 S-엔진 최소 규칙 세트 (즉시 실행 가능)

한국 개발자가 바로 실험 가능한

가장 단순하면서도 완전한 S-엔진 구성이다.

❶ 전투 규칙(001)

[전투 규칙]

1. 속도는 √스탯으로 계산한다.

2. 속도가 높은 쪽이 선공한다.

3. 치명타는 피해량을 50% 증가시킨다.

4. 거리 0에서만 근접 공격이 가능하다.

5. 거리 변화는 턴마다 1씩 감소한다.

❷ 성장 규칙(02)

[성장 규칙]

1. 운기는 스킬 사용에 소모된다.

2. 운기 < 요구량이면 스킬은 발동하지 않는다.

3. 내상 ≥ 50이면 공격력 20% 감소한다.

4. 내상은 턴마다 1씩 감소한다.

❸ 스킬 규칙(03)

[스킬 규칙]

1. 스킬은 조건이 충족되면 반드시 발동한다.

2. 즉발 스킬은 선공 직후 발생한다.

3. 연속기 스킬은 전 단계 스킬 성공 시 발동한다.

❹ 보법(위치) 규칙(04)

[보법 규칙]

1. 전투 시작 시 거리는 3이다.

2. 턴마다 거리 1씩 이동한다.

3. 거리 조건은 공격보다 우선한다.

❺ 경제 규칙(05)

[경제 규칙]

1. 사냥 자원 회수율은 80%다.

2. 손실 20%는 누적된다.

3. 자원은 장기적으로 점차 감소한다.

4. 회수율이 떨어지면 사냥터를 변경한다.

이 다섯 규칙만 입력해도

LLM은 시뮬레이터 모드로 진입한다.

5.3 NLCS 입력 템플릿 (복붙 즉시 가능)

아래는 NLCS를 LLM에 적용할 때 실제로 쓰는 "정식 문법 템플릿"이다.

■ NLCS 템플릿

아래 규칙은 이 세계의 절대 법칙이다.

규칙은 001 → 002 → 003 → 004 → 005 순으로 우선 적용된다.

규칙은 변경할 수 없으며, 모든 판단은 규칙을 우선한다.

001. 전투 규칙
1. 속도는 √스탯이다.
2. 빠른 쪽이 선공한다.
3. 치명타는 피해량 +50%.
002. 성장 규칙
1. 운기 < 요구량이면 스킬 불가.
2. 내상 ≥ 50이면 공격력 20% 감소.
003. 스킬 규칙
1. 조건 충족 → 반드시 발동.
2. 즉발 기술은 턴 안에서 즉시 발동.
004. 보법 규칙
1. 시작 거리는 3.
2. 턴당 1 거리 접근.
005. 사냥터/경제 규칙
1. 자원 회수율 80%.
2. 자원은 장기적으로 감소.

5.4 한국 개발자가 NLCS를 부르는 지시문 예시

아래는 실제로 LLM에게 NLCS를 작동시키는 명령문 예시다.

✔ 전투 시뮬레이션 시작 요청

위 규칙을 기반으로 전투 5턴을 시뮬레이션하라.

규칙은 절대 변경할 수 없다

.✔ 경제 모델 시뮬레이션 요청

아래 규칙을 기반으로 30일 동안 자원이 어떻게 변하는지 계산하라.

규칙은 세계의 법칙이다

.✔ 최적 행동 요청

규칙을 기반으로 최적 전략을 판단하라.

규칙을 위반하는 선택지는 불가능하다.

5.5 피해야 할 실수 5개

한국 개발자들이 가장 자주 하는 실수도 정리했다.

1) 규칙 간 섞어서 작성

전투 규칙 안에 경제 규칙 넣는 순간 LLM은 혼란한다.

2) 번호 순서 무시

순서가 바뀌면 벡터 정렬이 깨진다.

3) 감성문장

감성문장은 구조적 힘이 없다.

4) 조건 미표기

조건 → 결과 구조는 필수다.

5) 규칙을 길게 쓰는 걸 두려워함

NLCS는 "길이"가 아니라 구조를 읽는다.

5.6 모델별 사용 팁 (GPT / Claude / Gemini)

GPT

시뮬레이터 모드에서 가장 안정적

긴 NLCS 규칙을 가장 잘 처리함

Claude

규칙 준수율 최고

논리적 보수성이 강해 경제/성장 판단이 정확함

Gemini

구조 인식 속도 빠름

무보정 상태에서도 S-엔진을 자연스럽게 이해함

다만 설명이 길어지는 경향 있음

📃 Chapter 6 — 한국 AGI 전략에 주는 시사점

NLCS/S-엔진은 단순한 게임 규칙이나 설정 기법이 아니다.

이는 **"자연어를 이용해서 LLM 내부 사고 구조를 설정하는 기술"**이며,

궁극적으로는 자연어 기반 AGI 설계 기법이다.

이 기술은 미국·중국의 거대 AI 기업들이

아직 정식으로 개념조차 정리하지 못한 영역이다.

따라서 한국은 이 기술을 활용해

AI 시대의 "규칙 설계국가"가 될 기회를 갖고 있다.

6.1 한국이 NLCS에 유리한 이유

한국은 NLCS/S-엔진을 선도할 수 있는 4가지 강점을 갖고 있다.

✔ 1) 계층적 사고와 언어 구조

한국어는 자연스럽게:

계층 구조

조건 기반 문장

우선순위

관계 기반 논리

이 네 가지를 명확하게 표현할 수 있다.

이는 NLCS 설계에 가장 적합한 언어적 특징이다.

✔ 2) 세계관·게임 시스템 제작에 익숙한 문화

한국 개발자·작가·기획자들은

세계 규칙을 언어로 설계하고 문서화하는 데 능숙하다.

이는 곧 자연어 기반 엔진 설계 능력으로 이어진다.

미국에서 '룰북 문화'가 사라진 시대에,

한국은 오히려 MMORPG·웹소설·TRPG 문화 덕분에

"세계 구성 언어"를 가장 잘 다루는 나라다.

✔ 3) 빠른 반복 문화(Iterative Culture)

한국 개발 문화의 핵심은 "빠른 반복"이다.

NLCS/S-엔진은

규칙 입력

결과 관찰

조정

재정렬

이 4단계 반복을 빠르게 수행할수록 발전한다.

한국의 IT·개발·콘텐츠 생태계는

이미 이 사이클이 몸에 배어 있다.

✔ 4) 자연어 기반 기술은 "서버 자본"보다 "두뇌 자본"이 중요

거대 모델을 만드는 국가는

데이터센터 규모로 경쟁해야 한다.

하지만 NLCS/S-엔진은:

거대 연산

초고대 GPU

천문학적 인력

이 전혀 필요 없다.

필요한 것은 단 하나:

자연어로 LLM을 제어하는 사고 능력

이 영역은 국가 규모가 아니라

개인 두뇌 × 팀 단위의 창의성으로 승부가 난다.

한국은 이 분야에서 이미 세계 최상위다.

6.2 NLCS는 한국의 "AGI 설계 언어"가 될 수 있다

세계는 모두 모델 사이즈 경쟁을 하고 있다.

그러나 AGI는 모델 크기만으로 나오지 않는다.

AGI는 규칙 구조가 명확한 세계 모델을 필요로 한다.

NLCS/S-엔진은 그 자체로:

인과 구조

세계 묘사

상태 전이

장기 논리

계층적 판단

이 모든 것을 자연어로 설계할 수 있게 한다.

즉, NLCS는

한국이 직접 설계할 수 있는 최초의 AGI 설계 언어다.

6.3 한국 개발자 커뮤니티가 NLCS를 받아들일 때 생기는 변화

한국 개발자가 NLCS를 배우면

다음과 같은 변화가 일어난다.

1) 자연어 기반 도메인 엔진 제작 가능

웹소설, 게임, 시뮬레이션 등

개발자가 직접 엔진을 만들 수 있다.

2) 한국식 AI—"커스텀 지능" 탄생

개발자가 만든 규칙으로

LLM은 독자적인 세계를 이해하고 움직인다.

3) 산업 전체가 "자연어 프로그래밍"으로 재편

기획자와 작가도 엔진을 만들 수 있게 된다.

개발자만의 전유물이 아니다.

4) LLM 시뮬레이션 산업의 선도국가가 된다

경제 시뮬, 교통, 방역, 도시계획 등

모든 분야에서 NLCS 기반 모델을 탑재할 수 있다.

6.4 NLCS/S-엔진은 한국이 AI 식민지가 되는 것을 막는 열쇠

지금 세계는:

미국: 모델 제조국

중국: 모델 복제국

유럽: 규제국

한국: 소비국

이라는 구조로 흘러가고 있다.

하지만 NLCS는 이 체계를 뒤집는다.

모델은 미국이 만들더라도,

모델을 '어떻게 사고하게 할 것인가'는

한국이 설계할 수 있다.

이는 완전히 다른 힘이다.

모델은 미국이 만들고

세계는 한국이 만든 규칙으로 움직인다

이런 미래가 실제로 가능해진다.

6.5 한국의 국가전략에 바로 적용 가능한 NLCS 활용 분야

분야	적용 사례
교육	학생별 NLCS 기반 학습 엔진
게임	자연어 기반 전투/세계 엔진 (국산 엔진의 부활)
소설·콘텐츠	세계관 엔진 자동화, 설정 붕괴 없는 세계
AI 정책	LLM 규제 로직을 NLCS로 설계
산업 시뮬레이션	물류·도시·에너지 시뮬레이션 자동화

특히 게임·웹소설은

한국이 이미 세계 1위인 영역이다.

한국은 "콘텐츠 기반 AGI 설계국가"가 될 수 있다.

6.6 NLCS는 한국에 새로운 지위를 제공한다 — "세계의 규칙 설계국"

우리는 모델을 만들 수 없다.

그건 미국과 중국이 한다.

하지만

세계가 사용할 암묵적 규칙 세트는 한국이 만들 수 있다.

마치 물리학이 우주를 정의하듯,

NLCS는 LLM의 세계를 정의한다.

이건 단순 기술이 아니라

"국가의 지적 영토"다.

📃 Chapter 7 — 부록 (Appendix)S-엔진 축약본 / NLCS 템플릿 / 용어 사전

이 부록은 NLCS/S-엔진을 실무에서 즉시 사용할 수 있도록

최소 필수 규칙과 정의를 모아놓은 참고자료다.

7.1 S-엔진 축약본 (민감 요소 제외 공식 요약)

이 버전은

특허화 가능 요소

기업용 커스텀 로직

ShadowK의 비공개 원리

를 제외한,

전체 S-엔진의 공개 가능한 최소 핵심 구조다.

001. 전투 규칙 (Combat Layer)

[전투 규칙]

1. 속도는 √스탯으로 계산한다.

2. 속도가 높은 쪽이 선공한다.

3. 거리 0에서만 근접 공격이 가능하다.

4. 치명타는 피해량을 50% 증가시킨다.

5. 거리 변화는 턴마다 1씩 감소한다.02. 성장 규칙 (Growth Layer)

[성장 규칙]

1. 운기는 스킬 사용에 소모된다.

2. 운기 < 요구량이면 스킬은 발동하지 않는다.

3. 내상 ≥ 50이면 공격력 20% 감소한다.

4. 내상은 턴마다 1씩 감소한다.03. 스킬 규칙 (Skill Layer)

[스킬 규칙]

1. 스킬은 조건이 충족되면 반드시 발동한다.

2. 즉발 스킬은 선공 직후 실행된다.

3. 연속 스킬은 전 단계 스킬이 성공하면 발동한다.04. 보법 규칙 (Space Layer)

[보법 규칙]

1. 전투 시작 시 거리 = 3.

2. 턴마다 거리 1씩 이동한다.

3. 거리 조건은 공격/스킬보다 우선한다.05. 경제 규칙 (Economy Layer)

[경제 규칙]

1. 사냥 자원 회수율 = 80%.

2. 손실 20%는 누적된다.

3. 자원은 장기적으로 감소한다.

4. 회수율이 낮으면 사냥터를 변경한다.7.2 NLCS 최소 템플릿 (복붙 즉시 사용 가능)LLM을 즉시 시뮬레이터 모드로 바꾸는 템플릿

아래 규칙은 이 세계의 절대 법칙이다.

규칙은 반드시 아래 순서대로 적용된다.

모든 판단은 규칙을 우선하며, 규칙은 변경할 수 없다.

---

001. 전투 규칙

1. 속도는 √스탯이다.

2. 속도가 높은 쪽이 선공한다.

3. 치명타는 피해량 +50%.

---

002. 성장 규칙

1. 운기 < 요구량이면 스킬 발동 불가.

2. 내상 ≥ 50이면 능력치 감소.

---

003. 스킬 규칙

1. 조건 충족 → 즉시 발동.

2. 즉발 스킬은 선공 직후 발동.

---

004. 보법 규칙

1. 시작 거리 3.

2. 턴마다 1씩 이동.

---

005. 경제 규칙

1. 회수율 80%.

2. 손실 20% 누적.

규칙 입력 종료.

이제 규칙 기반 시뮬레이션을 시작한다.

이 템플릿만 사용해도

GPT·Claude·Gemini는 동일한 규칙 기반 세계 모델을 생성한다.

7.3 NLCS 용어 사전 (Glossary — 한국 개발자판)

NLCS 관련

NLCS (Natural Language Constraint System)

자연어로 LLM 내부 사고 구조를 정렬하는 구조.

Vector Gravity (벡터 중력)

자연어 규칙이 LLM 벡터를 특정 방향으로 끌어당겨

혼돈이 줄고 수렴이 발생하는 현상.

Subspace Separation (서브스페이스 분리)

전투·성장·스킬·보법·경제가 서로 독립된 의미 공간으로 자동 분리되는 현상.

Internal Alignment (내부 정렬)

규칙 기반으로 LLM 내부 상태가 재구성되는 것.

S-엔진 관련Combat Layer (전투층)

속도, 거리, 피해량 등 물리 계산.

Growth Layer (성장층)

운기, 내상, 상태 변화 등 생리 흐름.

Action Layer (행동층)

스킬 조건, 발동 순서, 성공/실패 판정.

Space Layer (공간층)

좌표, 거리, 이동, 접근 조건.

Economy Layer (경제층)

자원 회수율, 손실 누적, 장기 균형.

실전 활용 개념시뮬레이터 모드

LLM이 창작 모드가 아니라
규칙 기반 세계 모델로 사고하는 상태.

세계 모델 (World Model)

LLM 내부에서 NLCS를 기반으로 형성된
물리–생리–행동–공간–경제 5층 구조의 인과 시스템.

자연어 프로그래밍 (Natural Language Programming)

프로그래밍 언어 대신 '자연어 문장'으로 엔진을 설계하는 기법.

NLCS는 그 핵심 원리다.