

# Komplexe Anpassung ohne neuronale Netze?

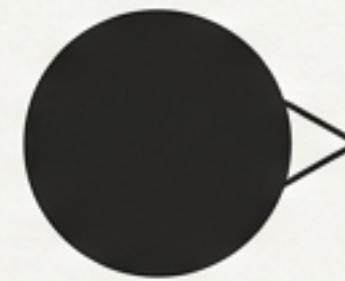
Micro-Swarm: Ein Experiment über Emergenz aus lokalen Regeln und mehrschichtigem Gedächtnis.



- **Emergenz statt Optimierung:** Wir untersuchen, wie globale Strukturen aus lokalen Interaktionen entstehen, ohne ein globales Ziel zu optimieren.
- **Nachvollziehbare Kausalität:** Jeder Effekt ist mechanistisch erklärbar – keine Blackbox.
- **Biologisch inspiriert:** Das System modelliert Konzepte wie Stigmergie (Pheromone), strukturelles Gedächtnis (Mycel) und evolutionäre Strategien (DNA).

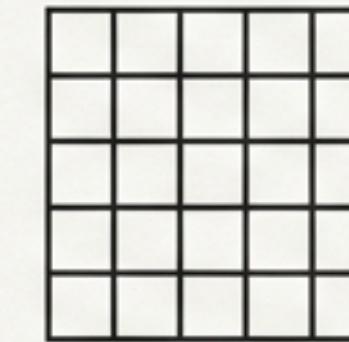
# Die Bausteine: Lokale Agenten in einer dynamischen Umwelt.

## Agenten



- Mobile Einheiten mit internem Zustand (Position, Energie, Genom).
- Handeln ausschließlich auf Basis lokaler Sensorik (links, vorne, rechts).
- Besitzen **keine** globale Sicht oder zentrales Ziel.

## Felder (GridFields)



- Ein 2D-Gitter als Umgebung.
- **Ressourcen:** Regenerieren langsam, werden von Agenten konsumiert.
- **Umwelt-Layer:** Pheromone, Moleküle, etc. dienen als Kommunikationsmedium.

Das Kernprinzip: Globale Ordnung entsteht, obwohl jeder Agent nur seine unmittelbare Nachbarschaft 'sieht'.

# Das Gedächtnis ist nicht monolithisch.



## 1. Moleküle (Kurzzeitgedächtnis)

- Funktion: Unmittelbare Reaktion auf Ereignisse.
- Eigenschaften: Stark verdampfend, sehr lokal.
- Analogie: Flüchtige chemische Signale.

## 2. Pheromone (Mittelzeitgedächtnis)

- Funktion: Stigmergische Kommunikation, Pfadmarkierung.
- Eigenschaften: Diffusiv, verdampfend.
- Analogie: Duftspuren von Ameisen.

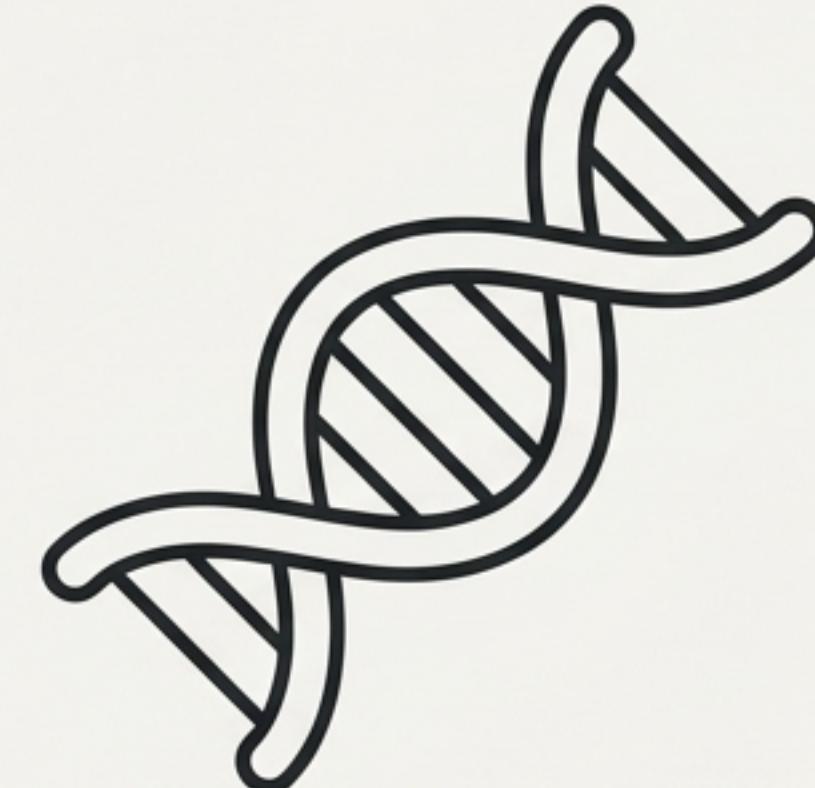
## 3. Mycel-Netzwerk (Strukturelles Gedächtnis)

- Funktion: Stabilisierung von Pfaden durch Speicherung dauerhafter Aktivität.
- Eigenschaften: Langsames, logistisches Wachstum; Transport über Diffusion (Laplacian).
- Analogie: Ein Pilzgeflecht, das Nährstoffautobahnen bildet.

# Das Langzeitgedächtnis speichert Strategien, nicht Zustände.

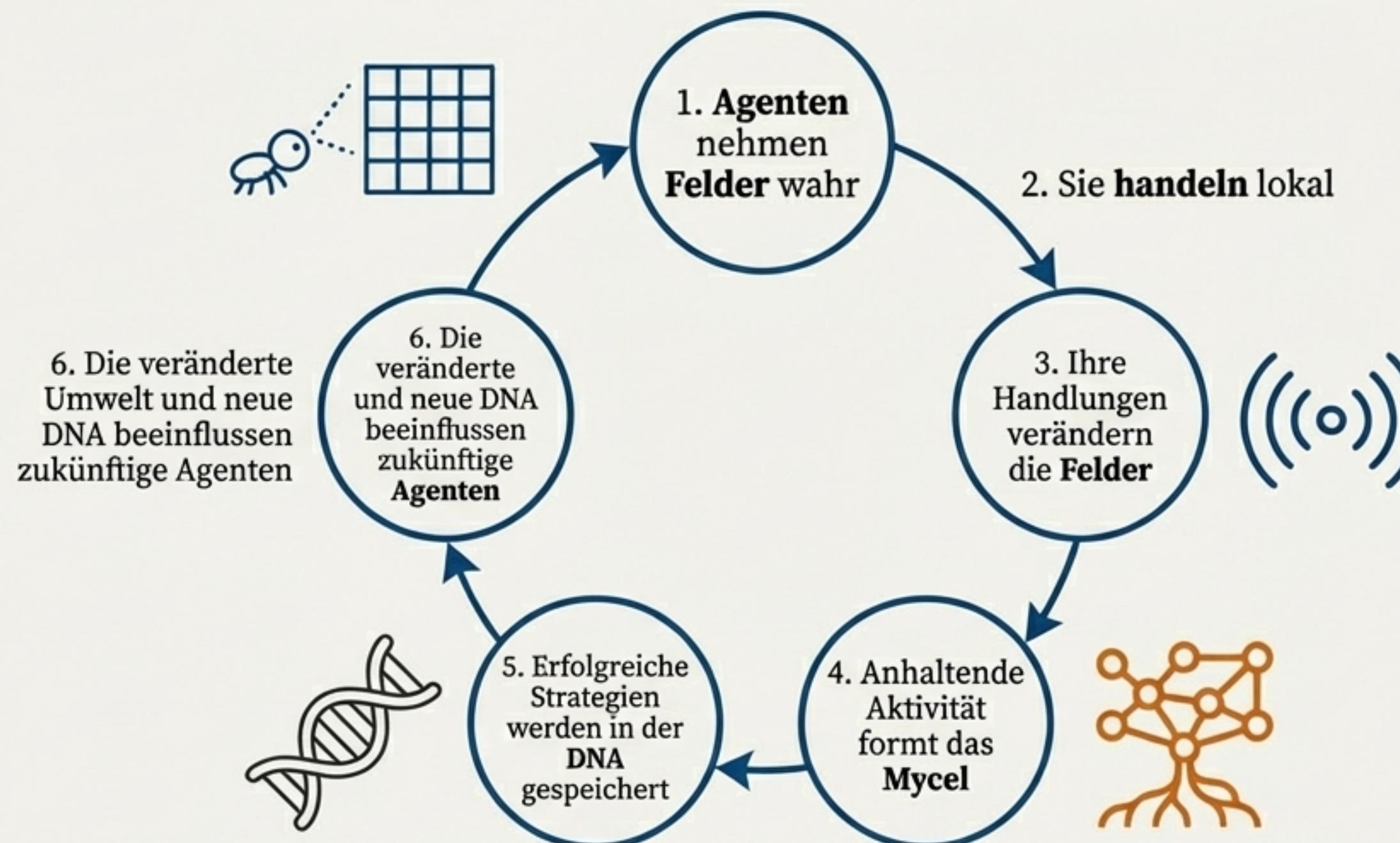
## DNA als evolutionärer Speicher

- Ein globaler Pool erfolgreicher Genome (Parametervektoren).
- Agenten mit niedriger Energie werden mit einem Genom aus dem Pool reinitialisiert (fitness-gewichtetes Sampling).
- Agenten mit hoher Energie tragen ihre Genome in den Pool ein.
- Kontrollierte Mutationen bei der Reproduktion ermöglichen die Entdeckung neuer Strategien.



DNA ermöglicht die Evolution von *Verhalten* (z. B. Effizienz bei der Nahrungssuche) über Generationen von Agenten hinweg.

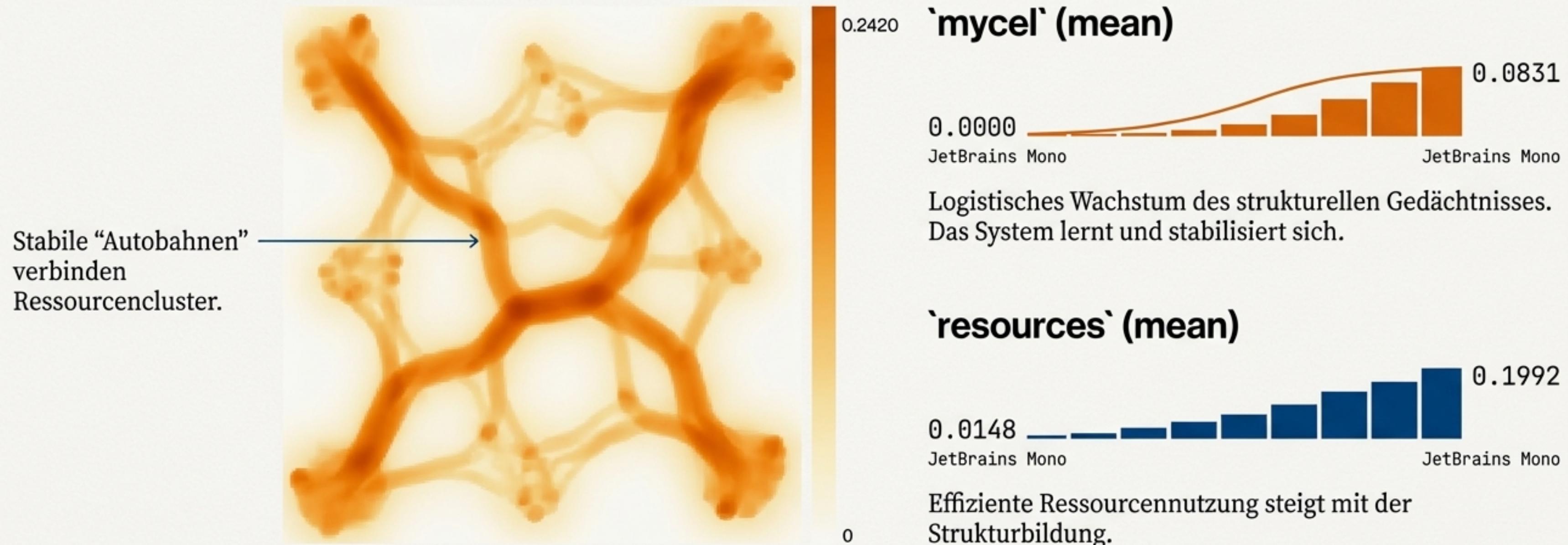
# Vom lokalen Handeln zur globalen Struktur: Der Kreislauf der Emergenz.



Aber funktioniert das in der Praxis? Sehen wir die erwartete Emergenz und Anpassung?

# Der Baseline-Run: Das System formt selbstorganisiert eine stabile Struktur.

Ohne externe Eingriffe oder Störungen entwickelt der Schwarm effiziente Sammelpfade und ein stabiles strukturelles Gedächtnis.



Dieser Zustand ist unsere “Ground Truth” für den Vergleich.

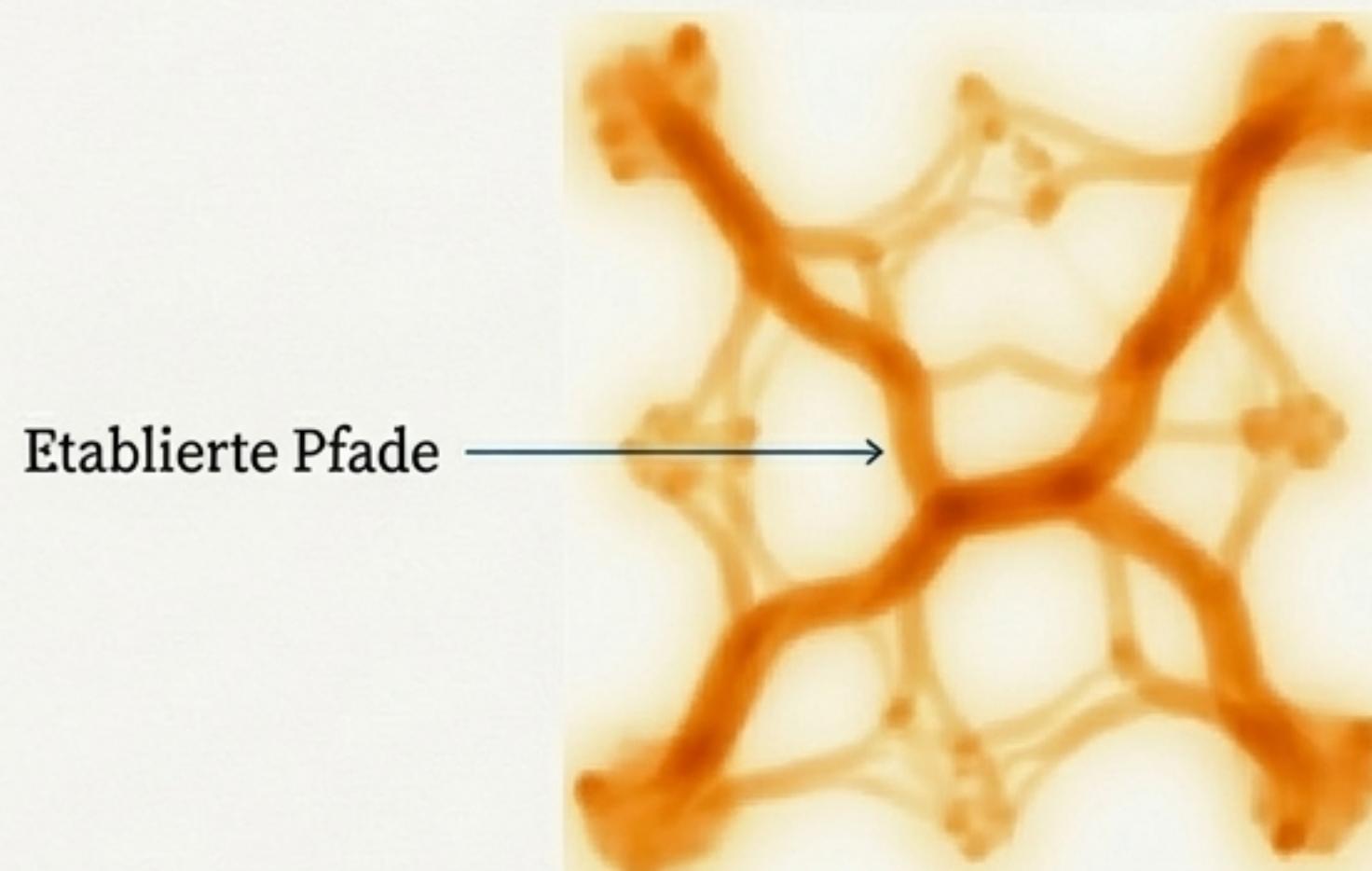
# Was passiert unter Druck? Der Stress-Test.

Der Lauf beginnt identisch zur Baseline.

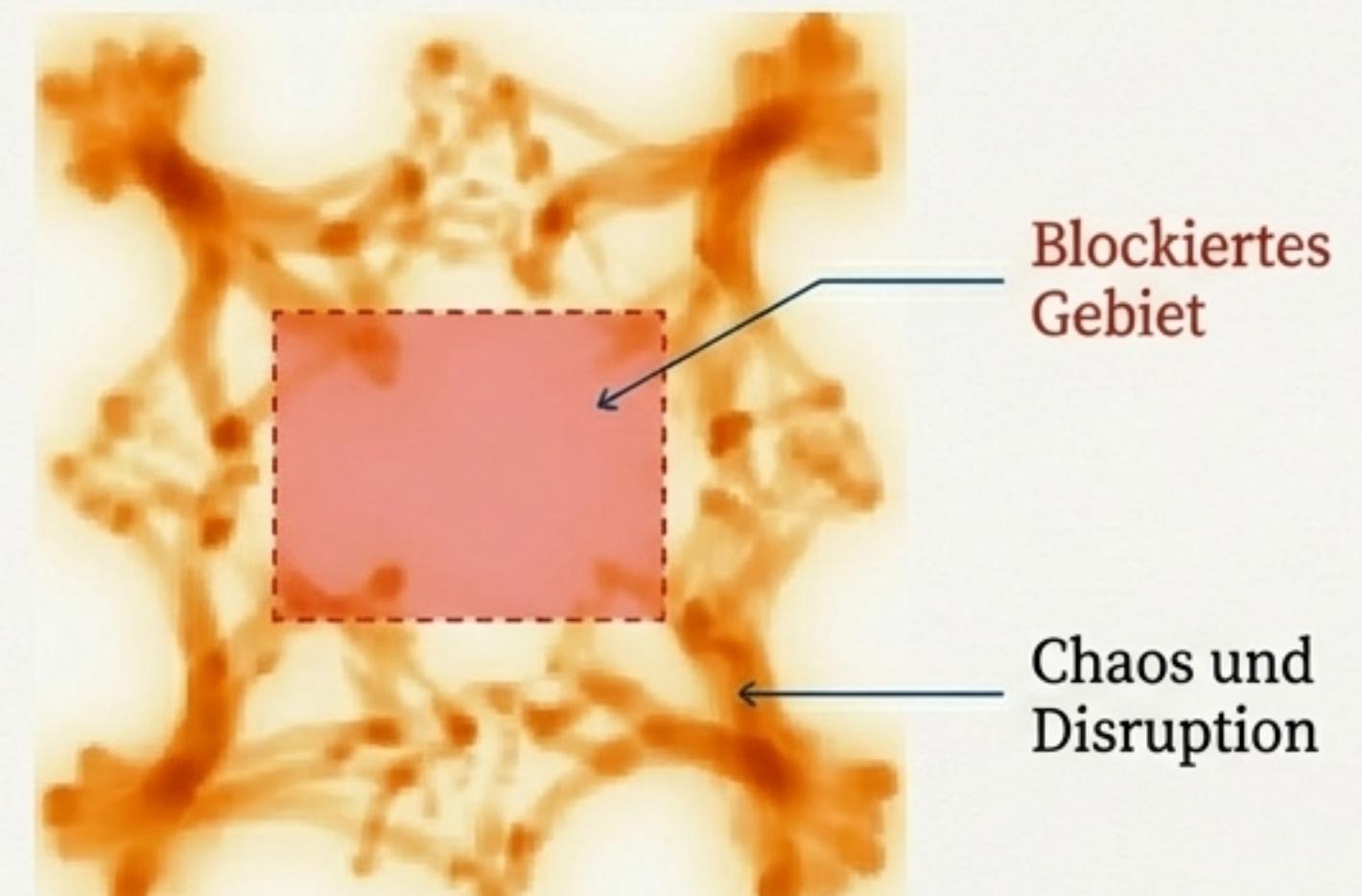
Bei Schritt 120: Eine massive Störung wird eingeführt.

Ein zentrales Ressourcenfeld wird blockiert (`block\_rect=40, 40, 30, 30`).

Die Pheromon-Kommunikation wird durch Rauschen gestört (`pheromone\_noise=0.004`).



Schritt 100 (vor Störung)



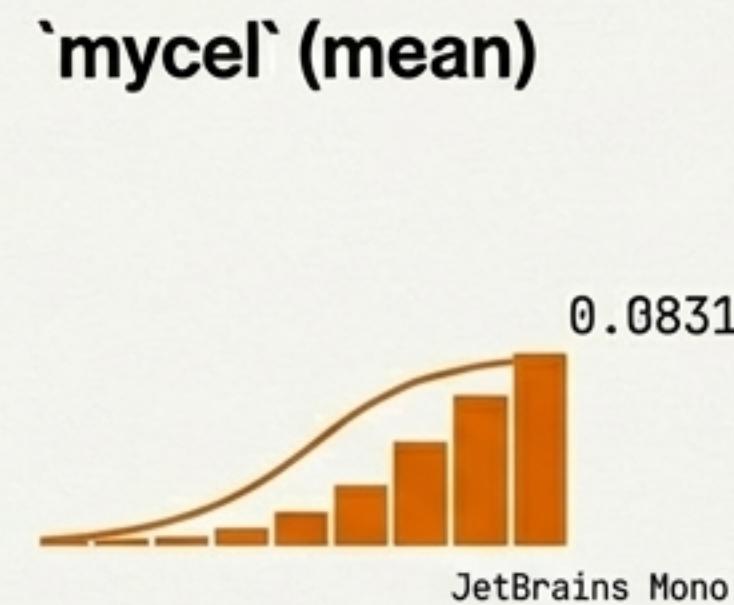
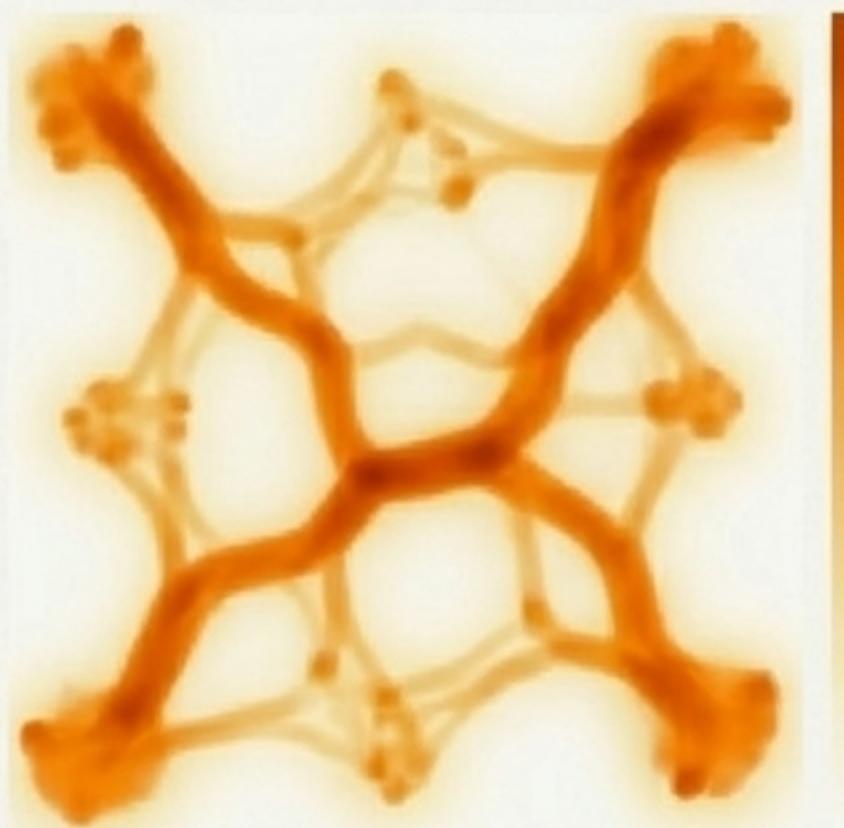
Schritt 150 (nach Störung)

Die etablierten Pfade sind nun nutzlos. Kann der Schwarm umlernen oder kollabiert das System?

# Das System lernt um: Emergenz neuer Pfade als Antwort auf die Störung.

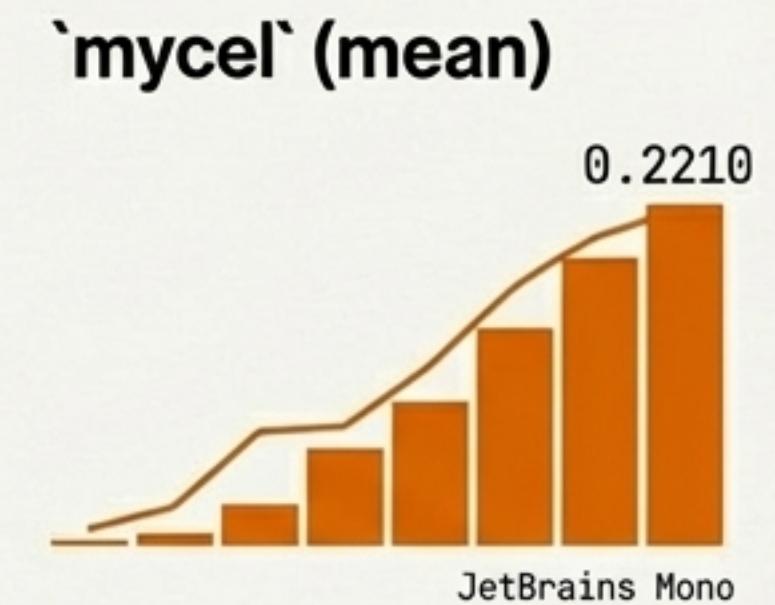
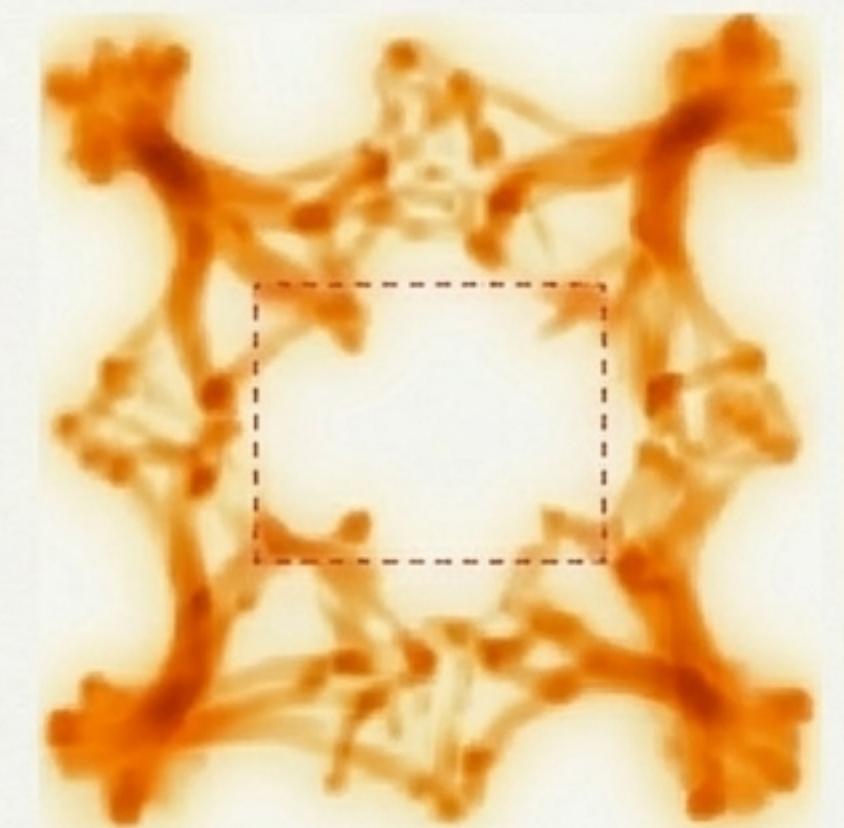
Vergleich der Endzustände (Schritt 450)

## Baseline-Run



Optimale, zentrale Pfade.

## Stress-Test-Run



Neue, dezentrale Pfade bilden sich um das Hindernis herum.

Das System ist nicht nur stabil, sondern **adaptiv**. Das strukturelle Gedächtnis (mycel) ist plastisch und findet neue Lösungen, was sich in einem noch stärkeren Wachstum des mycel-Wertes zeigt.

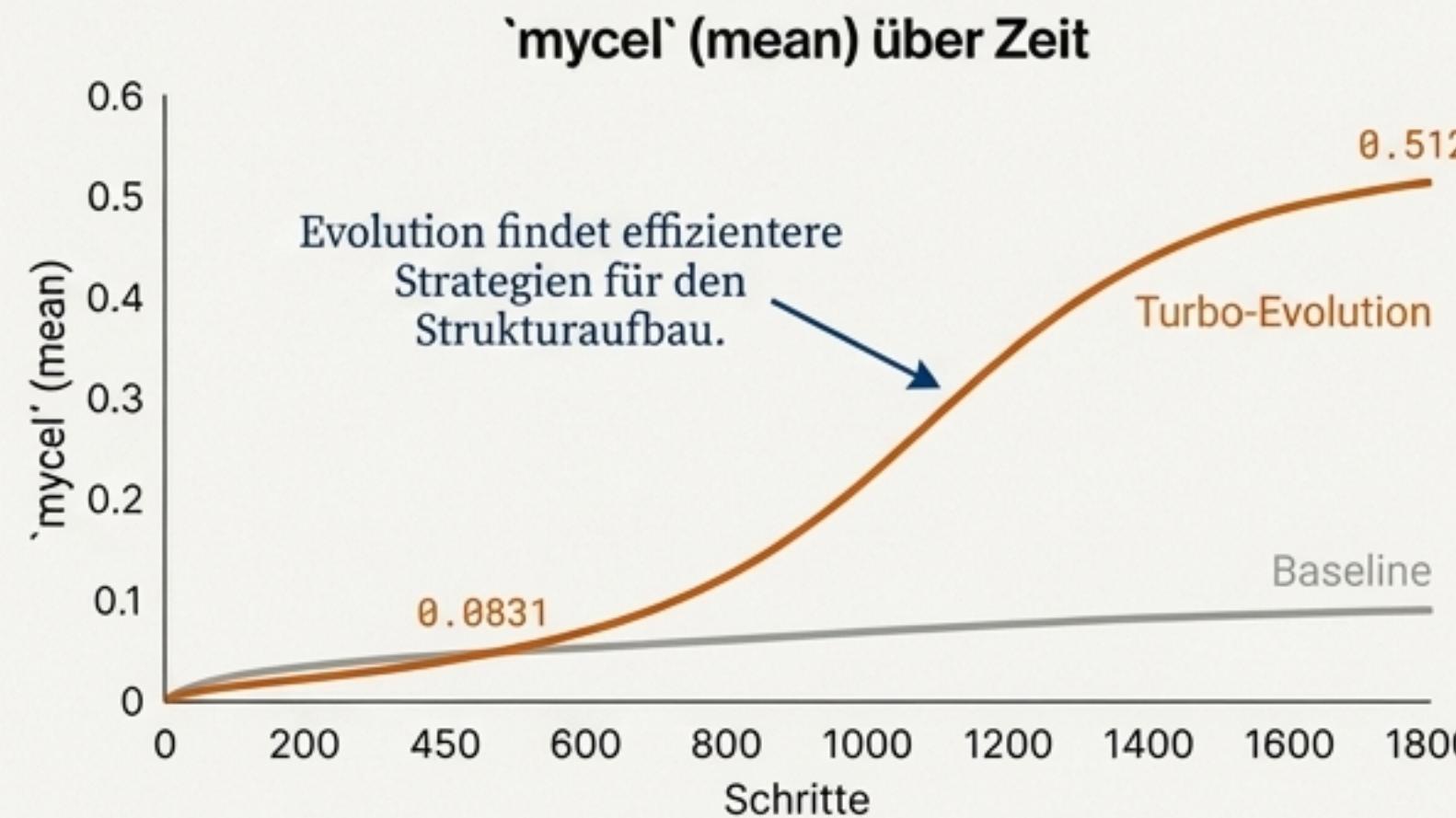
# Mehr als nur Anpassung: Gezielte Evolution beschleunigt die Effizienz

## Szenario

Der "Turbo-Evolution"-Lauf: Elite-Selektion ist aktiviert ( `--evo-enable` ).

Nur Genome, die die Fitness (Energiezuwachs) nachweislich steigern, werden in den DNA-Pool aufgenommen.

## Visueller Beweis

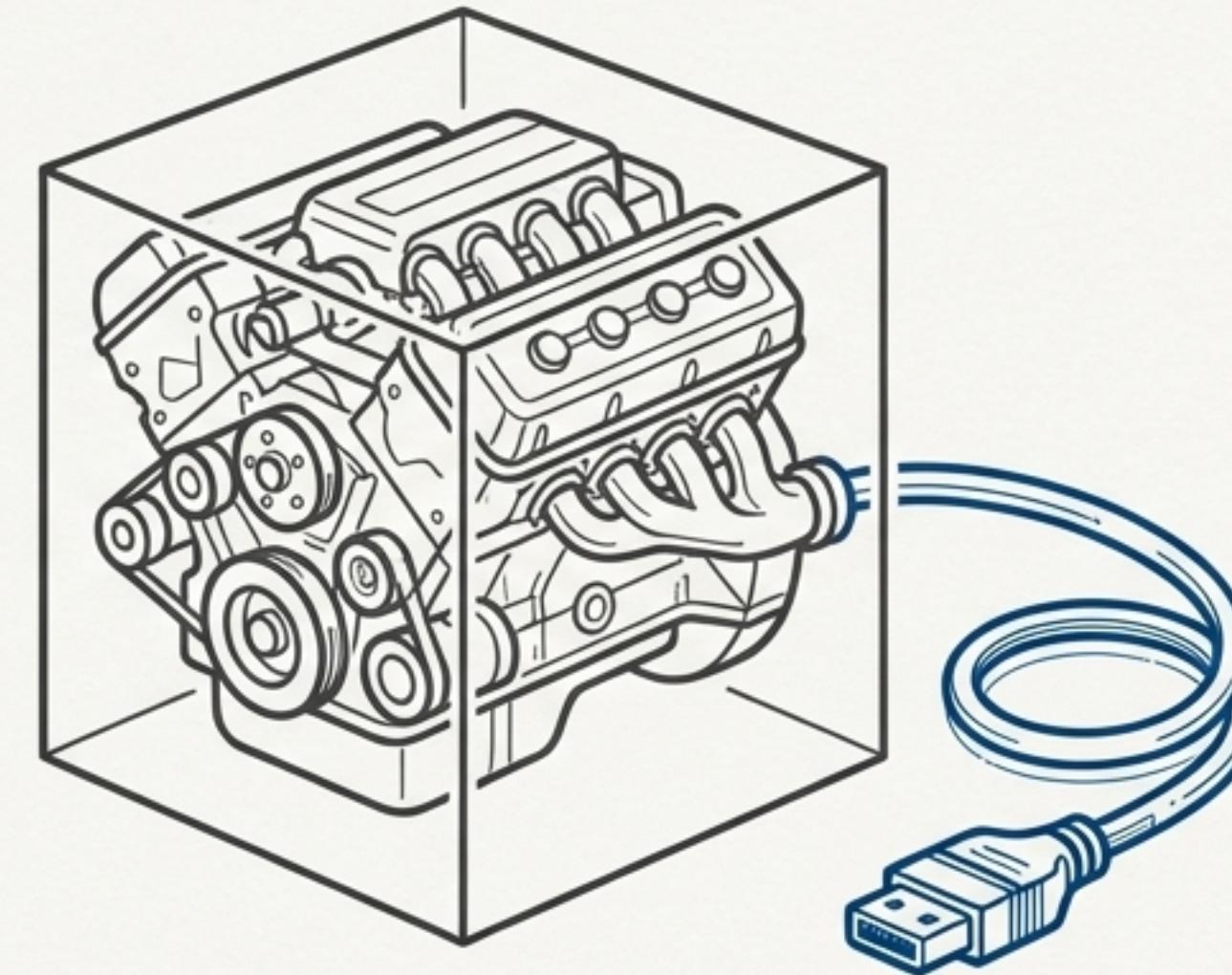


## Kernaussage

Durch Evolution optimiert sich das System selbst und übertrifft die Baseline-Performance deutlich.

# Diese Engine für emergentes Verhalten ist mehr als ein Experiment.

Die gesamte Simulations-Engine ist in einer performanten C++-Bibliothek gekapselt und über eine stabile C-API zugänglich.



- Wir haben die Komplexität der Simulation in eine deterministische, headless-fähige Engine verpackt.
- Sie können diese Engine direkt in Ihren eigenen Anwendungen nutzen.
- Die folgende Sektion zeigt Ihnen, wie.

# Der Kern: Eine stabile und portable C-API.

## Philosophie der API

- **Stabilität:** Semantische Versionierung (MAJOR.MINOR.PATCH) und eine klare Deprecation Policy garantieren Vorhersagbarkeit. ABI-Stabilität ist ein Kernziel.
- **Portabilität:** Eine einzige Header-Datei (`micro_swarm_api.h`) und eine DLL/SO-Datei.
- **Einfachheit:** Reine C-Schnittstelle, `__cdecl` Calling Convention, POD-Structs, keine komplizierten Abhängigkeiten.

## Schlüsselkonzepte der API-Nutzung

- **Handle-basiert:** `'ms_create()'`->`'ms_destroy()'`.
- **Synchron:** Kein internes Threading, volle Kontrolle für den Aufrufer.
- **Datenzugriff:** Direkter Zugriff auf Felder über `'ms_copy_field_in/out'` mit rohen `'float*'` Arrays.

```
// From micro_swarm_api.h

typedef struct ms_config_t {
    int width;
    int height;
    int agent_count;
    int steps;
    unsigned int seed;
    // ... more fields
} ms_config_t;

/**
 * @brief Creates a new micro-swarm context.
 * @return A valid handle, or NULL on failure.
 */
MS_API ms_handle_t ms_create(
    const ms_config_t* config
);
```

# In jede Umgebung einbettbar: Von Python bis Unreal Engine

## Python (ctypes)

```
import ctypes

# use ct.CDLL for __cdecl on Windows
lib = ctypes.CDLL("./micro_swarm.dll")
# ... setup argument types and call functions
```

## Unity (C#)

```
using System.Runtime.InteropServices;

// P/Invoke with Cdecl calling convention
[DllImport("micro_swarm",
    CallingConvention = CallingConvention.Cdecl)]
private static extern IntPtr ms_create(ref Config cfg);
```

Die stabile C-API ermöglicht eine nahtlose Integration in praktisch jede moderne Entwicklungsumgebung.

## Rust (FFI)

```
// extern "C" block with #[repr(C)] structs
#[link(name = "micro_swarm")]
extern "C" {
    fn ms_create(config: *const Config) -> Handle;
    // ... other function signatures
}
```

## Unreal (C++)

```
#include "HAL/PlatformProcess.h"

// Load DLL and bind function pointers
void* DllHandle = FPlatformProcess::GetDllHandle(
    L"micro_swarm.dll"
);
CreateFunc = (ms_create_func)FPlatformProcess::GetDllExport(
    DllHandle, L"ms_create"
);
```

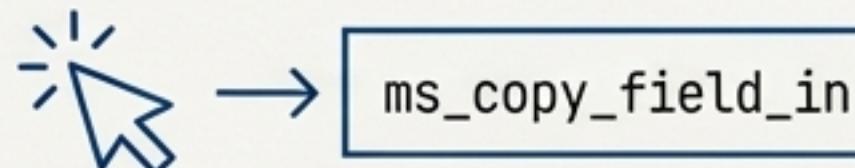
# Anwendungsfälle: Simulation als Treiber für Visuals und Gameplay.

## Unity (C#)

- **Visualisierung:** Felder mit `ms_copy_field_out` auslesen und in eine `Texture2D` (z.B. für Heatmaps) oder einen `ComputeBuffer` schreiben.



- **Interaktion:** Mausposition oder Collider-Events nutzen, um lokale Änderungen in ein `float[]` Array zu schreiben und mit `ms_copy_field_in` in die Simulation zu injizieren.



- **Editor-Tools:** Headless Batch-Läufe direkt aus dem Editor starten.

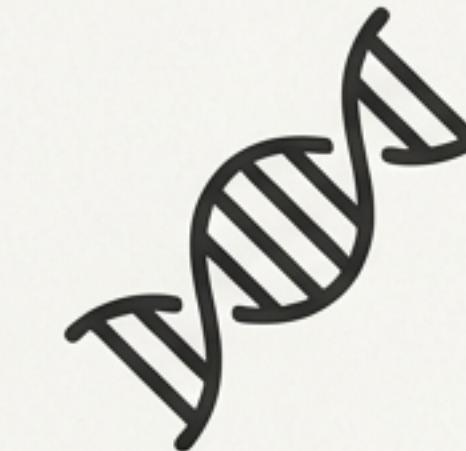
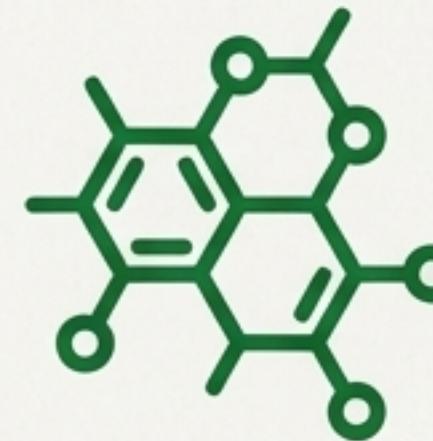
## Unreal (C++)

- **Visualisierung:** Felder nutzen, um Niagara-Systeme zu steuern (z.B. Dichte, Farbe) oder dynamische Material-Overlays zu erzeugen.



- **Gameplay-Logik:** Simulationszustände (z.B. Mycel-Dichte) als Grundlage für Spielmechaniken nutzen (z.B. 'verseuchtes' Gebiet).
- **Replays:** Deterministische Läufe über den Seed für Replays oder Debugging.

# Micro-Swarm: Eine Engine für nachvollziehbare, emergente Komplexität.



- **Die Vision:** Wir haben gezeigt, dass **komplexe Anpassung ohne neuronale Netze** möglich ist.
- **Die Architektur:** Ein **mehrschichtiges Gedächtnissystem** (Moleküle, Pheromone, Mycel, DNA) ermöglicht dies.
- **Der Beweis:** Experimente demonstrieren selbstorganisierte Strukturbildung, adaptive Reaktion auf Störungen und evolutionäre Optimierung.
- **Das Werkzeug:** Eine **performante C++ Engine** mit einer stabilen C-API steht bereit, um diese Dynamiken in Ihre eigenen Projekte zu integrieren.