Vectors are All You Need: A Retrospective

Jeffrey F. Eastman, Ph.D.
August 2025

Introduction

In 1975, I completed a PhD dissertation titled “An N-Space Model for Visual and Verbal Concepts”, in which the meaning of language was represented as vectors, and understanding was enacted through transformations. The resulting program, called Engraf, parsed English language input using an Augmented Transition Network (ATN), converted tokens into n-dimensional embedding vectors, and executed verb-driven transformations needed to create 3D graphical scenes. This structure anticipated — some fifty years ago — core elements of today’s transformer-based large language models (LLMs), including distributed vector semantics, transformation-based composition and execution-based grounding.

Engraf was intended to drive the 3D Color Computer Graphics System developed by my colleagues David Wooten, Nick England, Deborah Ogden, Glen Williamson and me at North Carolina State University under our mentor, Professor John Staudhammer. The lab ran on an Adage AGT-30 Graphics Terminal and a Varian 620 A minicomputer and output run-length-encoded scan lines to a video disk that drove a Sony Trinitron TV. For many years it was among the fastest 3D modeling system in the world. Unfortunately, Engraf ran on the IBM 370-168 at the Research Triangle Computer Center and was never actually connected to the graphics modeling system.

Although these ideas prefigured later advances in vector-based semantics and neural language models, they were not well received by the natural language processing community at the time. At the 1974 ACL conference, my paper was excluded from the proceedings and was met with skepticism during its presentation and dismissal in private correspondence afterward. In fairness, I was probably seen as a naive electrical engineering student from a small southern university with no NLP presence, trying to apply EE-familiar linear algebra to what was then largely viewed as a symbolic problem — “smashing square pegs into round holes.”

Though vectors were used in information retrieval in 1975 and some systems had used sets of semantic features *representable* as vectors, we are not aware of any NLP systems that explicitly *represented* grammatical, semantic and visual features as vectors within an executable n-space; such ideas were well outside the NLP mainstream. Nevertheless, the conceptual foundation of Engraf — that meaning could be operationalized through vector

transformations — remained compelling, and has only recently become technically realizable and indeed is now, mainstream.

2. Engraf: Verb-Centric Vector Semantics

The architecture of Engraf treated natural language as a set of executable transformations on a multidimensional concept space. Each input word was represented as a vector in this space, with dimensions encoding its basic grammatical categories, grammatical features, spatial coordinates, visual properties, preposition and verb semantics. Today these word/token vectors are called embeddings.

Verbs were the primary drivers of transformation: they acted on noun phrase vectors by applying either additive or multiplicative updates depending on part-of-speech roles identified by the ATN parser (§3.4). For example, 'MAKE IT MUCH MORE TRANSPARENT AND ROUGHER' resulted in a MODIFY command that adjusted multiple dimensions of the referenced object (§1.2.10).

3. Engraf's Embeddings as Attention's Value Vectors

Vectors are necessary but not sufficient for deep learning. In modern transformer architectures, the Attention matrices—Query (Q), Key (K), and Value (V)—are learned from a training corpus and used to compute Value vectors from the input token's embedding. These Value vectors carry the semantic payload of each token.

In Engraf, the vocabulary's embedding vectors served a role much like Value vectors, but they were specified by design, much smaller, and not learned:

- Each token's meaning vector was used directly in scene construction.
- The ATN parser drew on the vector's grammatical categories and features to determine how sentence meanings would be mathematically derived.
- Engraf ran in a 150 KB region on the IBM mainframe—about enough storage for just a few GPT-4 embedding vectors today.
- Transformer-style "attention" would not be possible until decades later, after many iterations of Moore's Law.

Engraf thus operationalized meaning as vectors and vector transformations, rather than symbolic patterns.

4. Grounding, Feedback, and Time Travel

Engraf's grounding was visual: meaning was to be verified by generating or modifying visible objects in 3D space. Incorrect interpretations (e.g., misuse of number agreement or modifiers) were rejected or flagged (§1.2.3, §1.2.23). The system maintained a timeline of operations, enabling rollback and forward replay (§1.2.27–1.2.31). This interactivity

resembled early dialog modeling and shares traits with modern multimodal systems like DALL·E or GPT-4V, which combine text, visual feedback, and iterative refinement.

5. Comparison to Modern LLMs

Today's transformer models such as GPT-4 (Vaswani et al., 2017; OpenAI, 2023) rely on embedding vectors, attention-driven transformation layers, and context-dependent composition of meaning. Engraf anticipated many of these:

- Token Vectors as Operators: In both systems, the representation of meaning is not static but arises through transformation.
- Grammar-Directed Composition: Engraf's POS-driven ATN guided vector operations whereas transformers use learned attention and feed-forward matrices.
- Execution as Understanding: Engraf judged meaning by scene correctness; LLMs use prediction error.
- Distributed Representation: In both systems, concepts are represented with high-dimensional vectors. Engraf's roughly 50 dimensions were manually designed; in LLMs, thousands of dimensions are learned from an immense training corpus.

6. Conclusion

Engraf was an early attempt to unify language, geometry, and computation. It anticipated some key ideas now found in transformer-based language models: embedding spaces, vectorized meaning, execution-based semantics, and interactive grounding. What was once a novel PhD experiment has become a guiding principle in modern AI. In my words:

Language is transformation, and Transformation is mathematics

7. Looking Ahead

Today, a Python implementation of Engraf is under development on GitHub (https://github.com/jeffeastman-2/Engraf). In Engraf's original SNOBOL implementation, the ATN parser used some textual part-of-speech categories and features, but these were conceptually defined in the common n-space alongside the numeric spatial and visual dimensions. In the modern Python reimplementation, these elements are represented numerically in the embedding vectors, as originally intended. This reimplementation revisits the original design goals and leverages modern Python libraries to unify the natural language parsing, vector algebra, and real-time 3D rendering. It reimplements the original ATN-based language analysis and connects it with a real-time graphics engine, finally closing the loop between understanding and visualization that was only theoretical in 1975.

References

Eastman, J. F. (1975). "An N-Space Model for Visual and Verbal Concepts". PhD Dissertation, North Carolina State University.

NCSU EE Dept. Computer Graphics Lab (1970-78). https://www.graphics-history.org/ncsu/

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). "Efficient Estimation of Word Representations in Vector Space". In *Proceedings of the 1st International Conference on Learning Representations (ICLR 2013), Workshop Track*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). "Attention is All You Need". Advances in Neural Information Processing Systems, 30.

OpenAI. (2021). DALL·E: Creating Images from Text. https://openai.com/research/dall-e

OpenAI. (2023). "GPT-4 Technical Report". https://openai.com/research/gpt-4